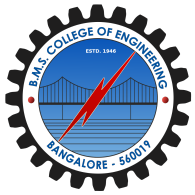


B.M.S. COLLEGE OF ENGINEERING

(AUTONOMOUS COLLEGE UNDER VTU)
BENGALURU-19

DEPARTMENT OF COMPUTER SCIENCE
AND ENGINEERING

DBMS LAB REPORT

(2020-2021)

NAME : VAISHNAVI JAGDALE

USN : 1BM19CS215

COURSE NAME : DATABASE MANAGEMENT SYSTEMS

COURSE TITLE : 19CS4PCDBM

SEM : 4D

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.

create database Insurance2;

use Insurance2;

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

desc person;

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

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

desc car;

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

create table accident(reportno int,adate date,location varchar(20),primary key(reportno));

desc accident;

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

create table owns(driverid varchar(10),regno varchar(10), primary key(driverid,regno),foreign key(driverid) references person(driverid),foreign key(regno) references car(regno));

desc owns;

Field	Type	Null	Key	Default	Extra
driverid	varchar(10)	NO	PRI	NULL	
regno	varchar(10)	NO	PRI	NULL	

create table participated(driverid varchar(10), regno varchar(10),reportno int, damageamt int, primary key(driverid,regno,reportno), foreign key(driverid) references person(driverid), foreign key(regno) references car(regno), foreign key(reportno) references accident(reportno));

desc participated;

Field	Type	Null	Key	Default	Extra
driverid	varchar(10)	NO	PRI	NULL	
regno	varchar(10)	NO	PRI	NULL	
reportno	int	NO	PRI	NULL	
damageamt	int	YES		NULL	

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');

commit;

select * from person;

	driverid	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('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);

commit;

select * from car;

	regno	model	year
▶	KA031181	Lancer	1957
	KA041702	Audi	2005
	KA052250	Indica	1990
	KA053408	Honda	2008
	KA095477	Toyota	1998
*	NULL	NULL	NULL

insert into accident values(11,'03-01-01','Mysore Road');

insert into accident values(12,'04-02-02','Southend Circle');

insert into accident values(13,'03-01-21','Bulltemple Road');

insert into accident values(14,'08-02-17','Mysore Road');

insert into accident values(15,'05-03-04','Kanakpura Road');

commit;

select * from accident;

	reportno	adate	location
▶	11	2003-01-01	Mysore Road
	12	2004-02-02	Southend Circle
	13	2003-01-21	Bulltemple Road
	14	2008-02-17	Mysore Road
	15	2005-03-04	Kanakpura Road

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');

commit;

select * from owns;

	driverid	regno
▶	A04	KA031181
	A05	KA041702
	A01	KA052250
	A02	KA053408
	A03	KA095477
*	NULL	NULL

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);

commit;

select * from participated;

	driverid	regno	reportno	damageamt
▶	A01	KA052250	11	10000
	A02	KA053408	12	50000
	A03	KA095477	13	25000
	A04	KA031181	14	3000
	A05	KA041702	15	5000
*	NULL	NULL	NULL	NULL

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.

update participated set damageamt=25000 where regno='KA053408' and reportno=12;

commit;

select * from participated;

	driverid	regno	reportno	damageamt
▶	A01	KA052250	11	10000
	A02	KA053408	12	25000
	A03	KA095477	13	25000
	A04	KA031181	14	3000
	A05	KA041702	15	5000
*	NULL	NULL	NULL	NULL

insert into accident values(16,'08-03-15','Domlur');

select * from accident;

	reportno	adate	location
▶	11	2003-01-01	Mysore Road
	12	2004-02-02	Southend Circle
	13	2003-01-21	Bulltemple Road
	14	2008-02-17	Mysore Road
	15	2005-03-04	Kanakpura Road
	16	2008-03-15	Domlur
*	NULL	NULL	NULL

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

select count(distinct driverid) CNT from participated a, accident b where a.reportno=b.reportno and year(b.adate)=2008;

	CNT
▶	1

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

select count(reportno) CNT from car c,participated p where c.regno=p.regno and model='Lancer';

	CNT
▶	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)

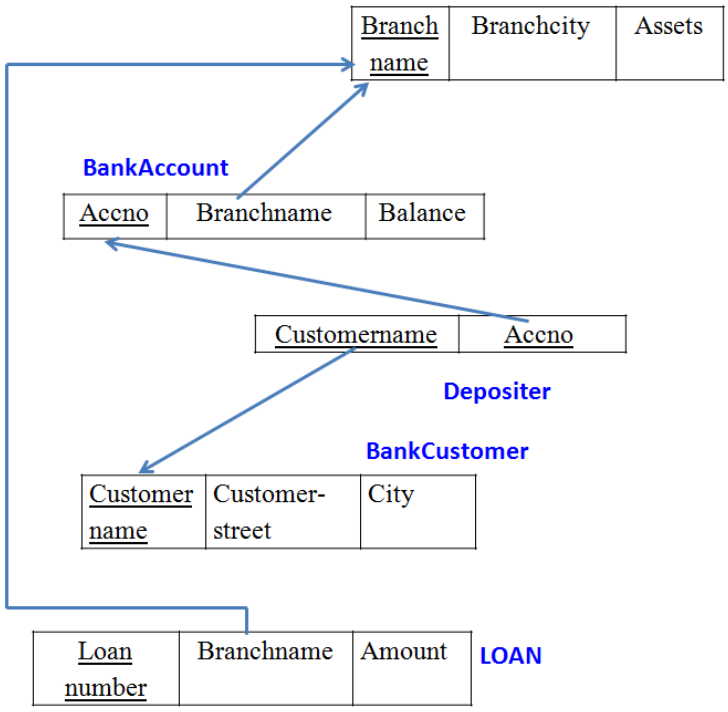
BankCustomer (customer-name: String, customer-street: String, customer-city: String)

Depositer(customer-name: String, accno: int)

Loan (loan-number: int, branch-name: String, amount: real)

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 bank;

use bank;

create table Branch(branchname varchar(30),branchcity varchar(30),assets real,primary key(branchname));

desc Branch;

Field	Type	Null	Key	Default	Extra
branchname	varchar(30)	NO	PRI	NULL	
branchcity	varchar(30)	YES		NULL	
assets	double	YES		NULL	

create table BankAccount(accno int,branchname varchar(30),balance real,primary key(accno),foreign key(branchname) references Branch(branchname));

desc BankAccount;

Field	Type	Null	Key	Default	Extra
accno	int	NO	PRI	NULL	
branchname	varchar(30)	YES	MUL	NULL	
balance	double	YES		NULL	

create table BankCustomer(customername varchar(30), customerstreet varchar(30), customercity varchar(30), primary key(customername));

desc BankCustomer;

Field	Type	Null	Key	Default	Extra
customername	varchar(30)	NO	PRI	NULL	
customerstreet	varchar(30)	YES		NULL	
customercity	varchar(30)	YES		NULL	

create table Depositer(customername varchar(30),accno integer, primary key(customername,accno),foreign key(customername) references BankCustomer(customername), foreign key(accno) references BankAccount (accno));

desc Depositer;

	Field	Type	Null	Key	Default	Extra
▶	customername	varchar(30)	NO	PRI	NULL	
	accno	int	NO	PRI	NULL	

create table Loan (loannumber int, branchname varchar(30),amount real, primary key(loannumber), foreign key (branchname) references Branch(branchname));

desc Loan;

	Field	Type	Null	Key	Default	Extra
▶	loannumber	int	NO	PRI	NULL	
	branchname	varchar(30)	YES	MUL	NULL	
	amount	double	YES		NULL	

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);

commit;

select * from Branch;

	branchname	branchcity	assets
▶	SBI_Chamrajpet	Bangalore	50000
	SBI_Jantarmantar	Delhi	20000
	SBI_ParlimentRoad	Delhi	10000
	SBI_ResidencyRoad	Bangalore	10000
	SBI_ShivajiRoad	Bombay	20000
*	NULL	NULL	NULL

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_ParlimentRoad',4000);
insert into Loan values(5, 'SBI_Jantarmantar',5000);
commit;
select * from Loan;
```

loannumber	branchname	amount
1	SBI_Chamrajpet	1000
2	SBI_ResidencyRoad	2000
3	SBI_ShivajiRoad	3000
4	SBI_ParlimentRoad	4000
5	SBI_Jantarmantar	5000
NULL	NULL	NULL

```
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);
commit;
select * from BankAccount;
```

accno	branchname	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
NULL	NULL	NULL

insert into BankCustomer values("Avinash","Bull_Temple_Road","Bangalore");

insert into BankCustomer values("Dinesh","Bannergatta_Road","Bangalore");

insert into BankCustomer values("Mohan","NationalCollege_Road","Bangalore");

insert into BankCustomer values("Nikil","Akbar_Road","Delhi");

insert into BankCustomer values("Ravi","Prithiviraj_Road","Delhi");

commit;

select * from BankCustomer;

customername	customerstreet	customercity
Avinash	Bull_Temple_Road	Bangalore
Dinesh	Bannergatta_Road	Bangalore
Mohan	NationalCollege_Road	Bangalore
Nikil	Akbar_Road	Delhi
Ravi	Prithiviraj_Road	Delhi
NULL	NULL	NULL

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

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

insert into Depositer values("Nikil",4);

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

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

insert into Depositer values("Nikil",9);

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

insert into Depositer values("Nikil",11);

commit;

select * from Depositer;

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

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

SELECT c.customername FROM BankCustomer c WHERE EXISTS(SELECT

d.customername,COUNT(d.customername) FROM Depositer d, BankAccount ba WHERE

d.accno=ba.accno AND c.customername=d.customername AND ba.branchname='SBI_ResidencyRoad' GROUP BY d.customername HAVING COUNT(d.customername)>=2);

customername
Dinesh
NULL

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

specific city (Ex. Delhi).

select distinct d.customername from Depositer d where exists(select * from BankAccount ba

where ba.accno=d.accno and exists (select * from Branch b where b.branchname =

ba.branchname and b.branchcity='Delhi'));

	customername
▶	Ravi
	Nikil

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

a specific city (Ex. Bombay).

DELETE FROM BankAccount WHERE branchname IN (SELECT branchname FROM BRANCH

WHERE branchcity='Bombay');

select * from BankAccount;

	accno	branchname	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
	NULL	NULL	NULL

PROGRAM 3: Supplier Database

Consider the following schema:

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

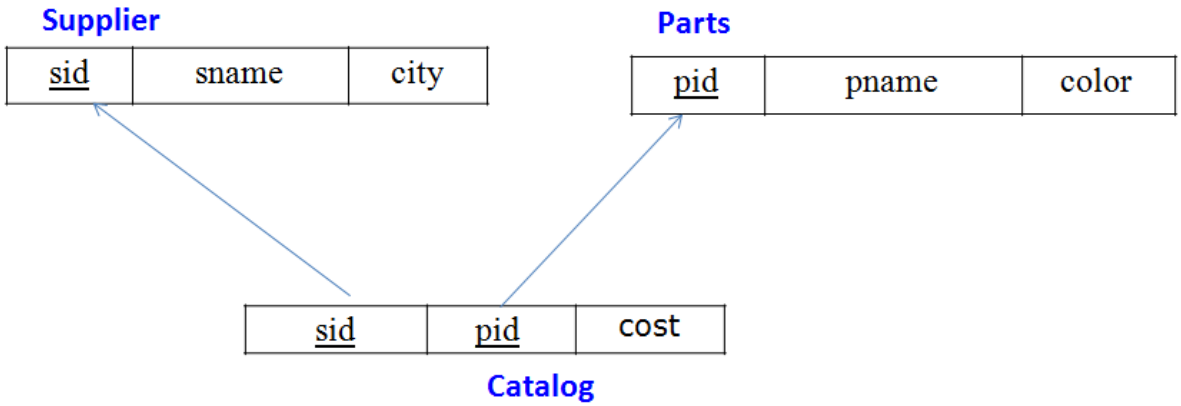
PARTS(pid: integer, pname: string, color: string)

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

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

Write the following queries in SQL:

Schema Diagram




```
create database supplier_database;
use supplier_database;
create table SUPPLIERS(sid int(5) primary key, sname varchar(20), city varchar(20));
desc SUPPLIERS;
```

	Field	Type	Null	Key	Default	Extra
▶	sid	int	NO	PRI	NULL	
	sname	varchar(20)	YES		NULL	
	city	varchar(20)	YES		NULL	

create table PARTS(pid int(5) primary key, pname varchar(20), color varchar(10));

desc PARTS;

	Field	Type	Null	Key	Default	Extra
▶	pid	int	NO	PRI	NULL	
	pname	varchar(20)	YES		NULL	
	color	varchar(10)	YES		NULL	

create table CATALOG(sid int(10), pid int(15), foreign key(sid) references SUPPLIERS(sid), foreign key(pid) references PARTS(pid), cost float(6), primary key(sid, pid));

desc CATALOG;

	Field	Type	Null	Key	Default	Extra
▶	sid	int	NO	PRI	NULL	
	pid	int	NO	PRI	NULL	
	cost	float	YES		NULL	

insert into suppliers values(10001, 'Acme Widget','Bangalore');

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

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

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

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

commit;

select * from SUPPLIERS;

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

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');
commit;
select * from PARTS;

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

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');
commit;

select * from CATALOG;

	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

i)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

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

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

	sname
▶	Acme Widget

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

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

	sname
▶	Acme Widget
	Johns

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

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

	pname
▶	Mobile
	Charger

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

SELECT DISTINCT C.sid FROM Catalog C
WHERE C.cost > (SELECT AVG (C1.cost)
FROM Catalog C1
WHERE C1.pid = C.pid);

	sid
▶	10002
	10004

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

```
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);
```

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.

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));

desc student;

	Field	Type	Null	Key	Default	Extra
▶	snum	int	NO	PRI	NULL	
	sname	varchar(10)	YES		NULL	
	major	varchar(2)	YES		NULL	
	lvl	varchar(2)	YES		NULL	
	age	int	YES		NULL	

create table faculty(fid INT,fname VARCHAR(20),deptid INT, PRIMARY KEY(fid));

desc faculty;

	Field	Type	Null	Key	Default	Extra
▶	fid	int	NO	PRI	NULL	
	fname	varchar(20)	YES		NULL	
	deptid	int	YES		NULL	

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

desc class;

Field	Type	Null	Key	Default	Extra
cname	varchar(20)	NO	PRI	NULL	
metts_at	timestamp	YES		NULL	
room	varchar(10)	YES		NULL	
fid	int	YES	MUL	NULL	

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;

Field	Type	Null	Key	Default	Extra
snum	int	NO	PRI	NULL	
cname	varchar(20)	NO	PRI	NULL	

insert into student values(1, 'john', '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);

commit;

select * from student;

	snum	sname	major	lvl	age
▶	1	John	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

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);
commit;
select * from faculty;

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

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);

commit;

select * from class;

	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

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');

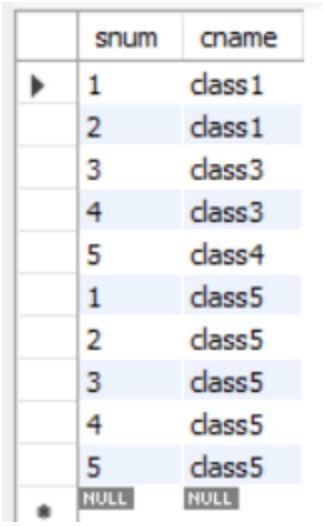
insert into enrolled values(3, 'class5');

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

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

commit;

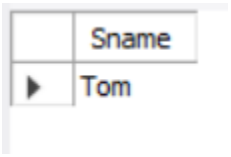
select * from enrolled;



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

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

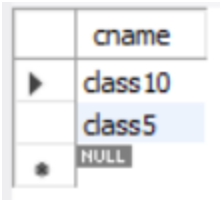
```
SELECT DISTINCT S.Sname FROM Student S, Class C, Enrolled E, Faculty F
WHERE S.snum = E.snum AND E.cname = C.cname AND C.fid = F.fid AND
F.fname = 'Harish' AND S.lvl = 'Jr';
```



	Sname
▶	Tom

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

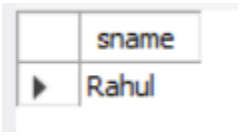
```
SELECT C.cname
FROM class C
WHERE C.room = 'R128'
OR C.cname IN (SELECT E.cname
                FROM enrolled E
                GROUP BY E.cname
                HAVING COUNT(*) >= 5);
```



	cname
▶	class10
	class5
*	NULL

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.metts_at = C2.metts_at);
```

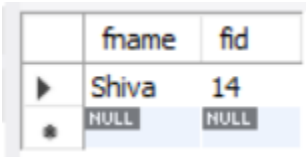


	sname
▶	Rahul

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) );
```



	fname	fid
▶	Shiva	14
*	NULL	NULL

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);
```

	fname
▶	Harish
	MV
	Mira
	Shiva

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

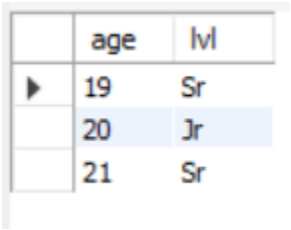
```
SELECT DISTINCT S.sname  FROM Student S
WHERE S.snum NOT IN (SELECT E.snum  FROM Enrolled E );
```

	sname
▶	Rita

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).

```
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));



	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.

```
create database airlines;
 use airlines;
```

```
 create table flights(flno integer primary key, ffrom varchar(15) , tto varchar(15) , distance integer, departs timestamp, arrives timestamp, price integer );
desc flights;
```

	Field	Type	Null	Key	Default	Extra
▶	FLNO	int	NO	PRI	NULL	
	FFROM	varchar(15)	YES		NULL	
	TTO	varchar(15)	YES		NULL	
	DISTANCE	int	YES		NULL	
	DEPARTS	timestamp	YES		NULL	
	ARRIVES	timestamp	YES		NULL	
	PRICE	int	YES		NULL	

```
create table aircraft
  (aid integer primary key,
  aname varchar(10),
  cruisingrange integer);
 desc aircraft;
```

	Field	Type	Null	Key	Default	Extra
▶	AID	int	NO	PRI	NULL	
	ANAME	varchar(10)	YES		NULL	
	CRUISINGRANGE	int	YES		NULL	

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

	Field	Type	Null	Key	Default	Extra
▶	EID	int	NO	PRI	NULL	
	ENAME	varchar(15)	YES		NULL	
	SALARY	int	YES		NULL	

```
create table certified
 (eid integer not null,
 aid integer not null,
 primary key (eid, aid),
 foreign key (eid) references employees (eid),
 foreign key (aid) references aircraft (aid));
desc certified;
```

	Field	Type	Null	Key	Default	Extra
▶	EID	int	NO	PRI	NULL	
	AID	int	NO	PRI	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);

commit;

select * from aircraft;

AID	ANAME	CRUISINGRANGE
101	747	3000
102	Boeing	900
103	647	800
104	Dreamliner	10000
105	Boeing	3500
106	707	1500
107	Dream	120000
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);

select * from employees;

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(703,105);
insert into certified values(704,105);
insert into certified values(705,103);
select * from certified;
```

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

```
insert into flights values(101,'Bangalore','Delhi',2500,TIMESTAMP '2005-05-13 07:15:31',TIMESTAMP '2005-05-13 17:15:31',5000);
```

```
insert into flights values(102,'Bangalore','Lucknow',3000,TIMESTAMP '2005-05-13 07:15:31',TIMESTAMP '2005-05-13 11:15:31',6000);
insert into flights values(103,'Lucknow','Delhi',500,TIMESTAMP '2005-05-13 12:15:31',TIMESTAMP ' 2005-05-13 17:15:31',3000);
insert into flights values(107,'Bangalore','Frankfurt',8000,TIMESTAMP '2005-05-13 07:15:31',TIMESTAMP '2005-05-13 22:15:31',60000);
insert into flights values(104,'Bangalore','Frankfurt',8500,TIMESTAMP '2005-05-13 07:15:31',TIMESTAMP '2005-05-13 23:15:31',75000);
insert into flights values(105,'Kolkata','Delhi',3400,TIMESTAMP '2005-05-13 07:15:31',TIMESTAMP  '2005-05-13 09:15:31',7000);
select * from Flights;
```

FLNO	FFROM	TTO	DISTANCE	DEPARTS	ARRIVES	PRICE
101	Bangalore	Delhi	2500	2005-05-13 07:15:31	2005-05-13 17:15:31	5000
102	Bangalore	Lucknow	3000	2005-05-13 07:15:31	2005-05-13 11:15:31	6000
103	Lucknow	Delhi	500	2005-05-13 12:15:31	2005-05-13 17:15:31	3000
104	Bangalore	Frankfurt	8500	2005-05-13 07:15:31	2005-05-13 23:15:31	75000
105	Kolkata	Delhi	3400	2005-05-13 07:15:31	2005-05-13 09:15:31	7000
107	Bangalore	Frankfurt	8000	2005-05-13 07:15:31	2005-05-13 22:15:31	60000
NULL	NULL	NULL	NULL	NULL	NULL	NULL

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 ));
```

	aname
▶	747
	Dreamliner
	Boeing
	Dream

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 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

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

```
SELECT DISTINCT E.ename

FROM Employees E
WHERE E.salary <( SELECT MIN(F.price)
                  FROM Flights F
                  WHERE F.ffrom = 'Bangalore' AND F.tto = 'Frankfurt' );
```

	ename
▶	A
	E

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 Temp.name, Temp.AvgSalary

FROM ( SELECT A.aid, A.aname AS name, AVG (E.salary)  AS AvgSalary
 FROM Aircraft A, Certified C, Employees E
```

```
WHERE A.aid = C.aid AND C.eid = E.eid AND A.cruisingrange > 1000
GROUP BY A.aid, A.aname )  Temp;
```

name	AvgSalary
747	75000.0000
Dreamliner	113333.3333
Boeing	96666.6667
707	50000.0000
Dream	113333.3333

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

```
SELECT DISTINCT E.ename
FROM Employees E, Certified C, Aircraft A
WHERE E.eid = C.eid AND C.aid = A.aid AND A.aname  LIKE 'Boeing%';
```

ename
A
C
D

vi. 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(F.distance)
                         FROM Flights F
                         WHERE F.ffrom = 'Bangalore' AND F.tto = 'Frankfurt' );
```

aid
104
107

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.

```
SELECT F.departs
FROM Flights F
WHERE F.flno IN ( ( SELECT F0.flno
 FROM Flights F0
 WHERE F0.ffrom = 'Bangalore' AND F0.tto = 'Delhi'
 AND extract(hour from F0.arrives) < 18 )
 UNION
( SELECT F0.flno
 FROM Flights F0, Flights F1
 WHERE F0.ffrom = 'Bangalore' AND F0.tto <> 'Delhi'
 AND F0.tto = F1.ffrom AND F1.tto = 'Delhi'
 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.ffrom = 'Bangalore'
 AND F0.tto = F1.ffrom
 AND F1.tto = F2.ffrom
 AND F2.tto = 'Delhi'
 AND F0.tto <> 'Delhi'
 AND F1.tto <> 'Delhi'
 AND F1.departs > F0.arrives
 AND F2.departs > F1.arrives
 AND extract(hour from F2.arrives) < 18));
```

	departs
▶	2005-05-13 07:15:31
	2005-05-13 07:15:31

viii. Print the name and salary of every non-pilot whose salary is more than the average salary for pilots.

```
SELECT E.ename, E.salary
FROM Employees E
WHERE E.eid NOT IN ( SELECT DISTINCT C.eid
 FROM Certified C )
AND E.salary >( SELECT AVG (E1.salary)
 FROM Employees E1
 WHERE E1.eid IN
( SELECT DISTINCT C1.eid
 FROM Certified C1 ) );
```

	ename	salary
▶	G	90000