

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

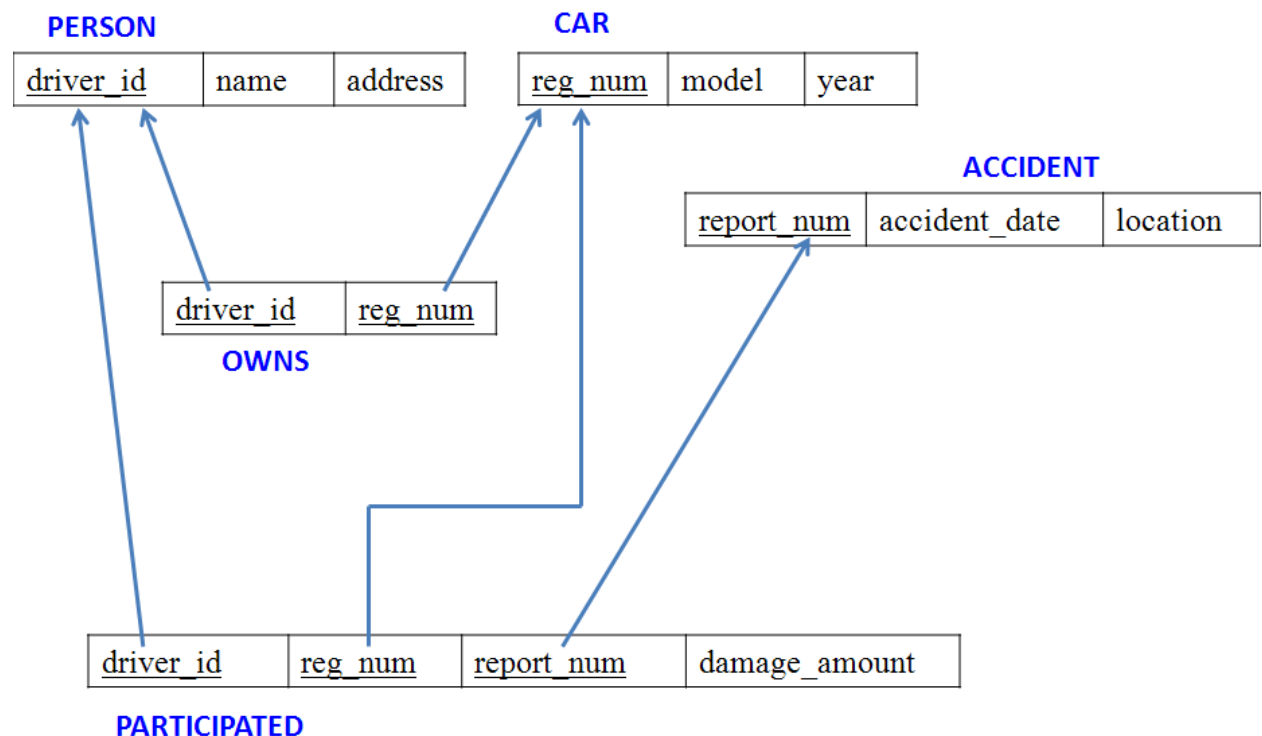
a. Update the damage amount to 25000 for the car with a specific reg-num (example 'KA053408') for which the accident report number was 12.

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



Tables

PERSON

<u>driver_id</u>	name	address
A01	Richard	Srinivas nagar
A02	Pradeep	Rajaji nagar
A03	Smith	Ashok nagar
A04	Venu	N R Colony
A05	Jhon	Hanumanth nagar

CAR

<u>reg_num</u>	model	year
KA052250	Indica	1990
KA031181	Lancer	1957
KA095477	Toyota	1998
KA053408	Honda	2008
KA041702	Audi	2005

OWNS

<u>driver_id</u>	<u>reg_num</u>
A01	KA052250
A02	KA053408
A03	KA031181
A04	KA095477
A05	KA041702

ACCIDENT

<u>report_num</u>	<u>accident_date</u>	<u>location</u>
11	01-JAN-03	Mysore Road
12	02-FEB-04	South end Circle
13	21-JAN-03	Bull temple Road
14	17-FEB-08	Mysore Road
15	04-MAR-05	Kanakpura Road

PARTICIPATED

<u>driver_id</u>	<u>reg_num</u>	<u>report_num</u>	damage_amount
A01	KA052250	11	10000
A02	KA053408	12	50000
A03	KA095477	13	25000
A04	KA031181	14	3000
A05	KA041702	15	5000

```
create database s1;

use s1;

create table PERSON(
driver_id char(20) NOT NULL,
Name_ char(30),
address char(50),
PRIMARY KEY (driver_id)
);
```

```
create table car(
reg_num char(20),
model char(30),
year int,
PRIMARY KEY(reg_num)
);
```

```
create table ACCIDENT(
report_num int,
accident_date date,
location char(50),
PRIMARY KEY(report_num)
);
```

```
create table OWNS(
driver_id char(20),
reg_num char(20),
FOREIGN KEY(driver_id) references PERSON(driver_id),
FOREIGN KEY(reg_num) references car(reg_num)
);
```

```
create table PARTICIPATED(  
    driver_id char(20),  
    reg_num char(20),  
    report_num int,  
    damage_amount int,  
    FOREIGN KEY(driver_id) references PERSON(driver_id),  
    FOREIGN KEY(reg_num) references car(reg_num),  
    FOREIGN KEY(report_num) references ACCIDENT(report_num)  
);
```

```
insert into PERSON
```

```
values ("A01","RICHARD","SRINIVAS NAGAR"),("A02","PRADEEP","RAJAJI  
NAGAR"),("A03","SMITH","ASHOK NAGAR")  
,"A04","VENU","N R COLONY"),("A05","JHON","HANUMANYH NAGAR");
```

```
insert into car
```

```
values ("KA052250","INDICA",1990),("KA031181","LANCER",1957),("KA095477","TOYOTA",1998)  
,"KA053408","HONDA",2008),("KA041702","AUDI",2005);
```

```
insert into ACCIDENT
```

```
values (11,"2003-01-01","MYSORE ROAD"),(12,"2004-02-02","SOUTH END CIRCLE"),(13,"2003-01-  
21","BULL TEMPLE ROAD");
```

```
insert into ACCIDENT
```

```
values (14,"2008-02-17","MYSORE ROAD"),(15,"2005-03-04","KANAKPURA ROAD");
```

insert into OWNS

values

("A01","KA052250"),("A02","KA053408"),("A03","KA031181"),("A04","KA095477"),("A05","KA041702");

insert into PARTICIPATED

values ("A01","KA052250",11,10000),("A02","KA053408",12,50000),("A03","KA095477",13,25000)

,("A04","KA031181",14,3000),("A05","KA041702",15,5000);

select* from PARTICIPATED;

update PARTICIPATED

SET damage_amount=25000

WHERE reg_num="KA053408";

insert into ACCIDENT

values (16,"2018-03-29","KORMANGLA");

SELECT COUNT(accident_date) AS accidentsin2008 FROM ACCIDENT

WHERE YEAR(accident_date)=2008;

SELECT COUNT(model) AS carwithhondaomodel FROM car

WHERE model="HONDA";

select* from ACCIDENT

where accident_date="2008-02-17";

insert into PERSON

values ("A06","JOHN","BANSHANKARI SATGE 2");

insert into car

values ("KA05MC001","AUDI",2018);

select * from car;

insert into ACCIDENT

values (17,"2019-03-01","BULL TEMPLE RD");

select * from ACCIDENT;

insert into OWNS

values ("A06","KA05MC001");

select * from OWNS;

insert into PARTICIPATED

values ("A06","KA05MC001",17,75000);

select* from PERSON;

	driver_id	Name_	address
▶	A01	RICHARD	SRINIVAS NAGAR
	A02	PRADEEP	RAJAJI NAGAR
	A03	SMITH	ASHOK NAGAR
	A04	VENU	N R COLONY
	A05	JHON	HANUMANYH NAGAR

select* from car;

	reg_num	model	year
▶	KA031181	LANCER	1957
	KA041702	AUDI	2005
	KA052250	INDICA	1990
	KA053408	HONDA	2008
	KA095477	TOYOTA	1998

select* from ACCIDENT;

	report_num	accident_date	location
▶	11	2003-01-01	MYSORE ROAD
	12	2004-02-02	SOUTH END CIRCLE
	13	2003-01-21	BULL TEMPLE ROAD
	14	2008-02-17	MYSORE ROAD
	15	2005-03-04	KANAKPURA ROAD
	16	2018-03-29	KORMANGLA
•	NULL	NULL	NULL

select* from OWNS;

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

select * from PARTICIPATED;

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

SELECT driver_id, reg_num FROM OWNS WHERE reg_num IN(SELECT reg_num FROM car WHERE model="indica");

	driver_id	reg_num
▶	A01	KA052250

SELECT reg_num FROM PARTICIPATED WHERE report_num IN(SELECT report_num FROM ACCIDENT WHERE YEAR(accident_date)>=2003 AND YEAR(accident_date)<=2005);

	reg_num
▶	KA052250
	KA053408
	KA095477
	KA041702

SELECT COUNT(*) FROM ACCIDENT WHERE location LIKE "%MYSORE%";

	COUNT(*)
▶	2

SELECT driver_id FROM PARTICIPATED WHERE damage_amount>=(SELECT AVG(damage_amount) FROM PARTICIPATED);

	driver_id
▶	A02
	A03

SELECT COUNT(*) FROM ACCIDENT WHERE YEAR(accident_date)=2008;

	COUNT(*)
▶	1

DELETE FROM PERSON WHERE Name_="JOHN";

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)

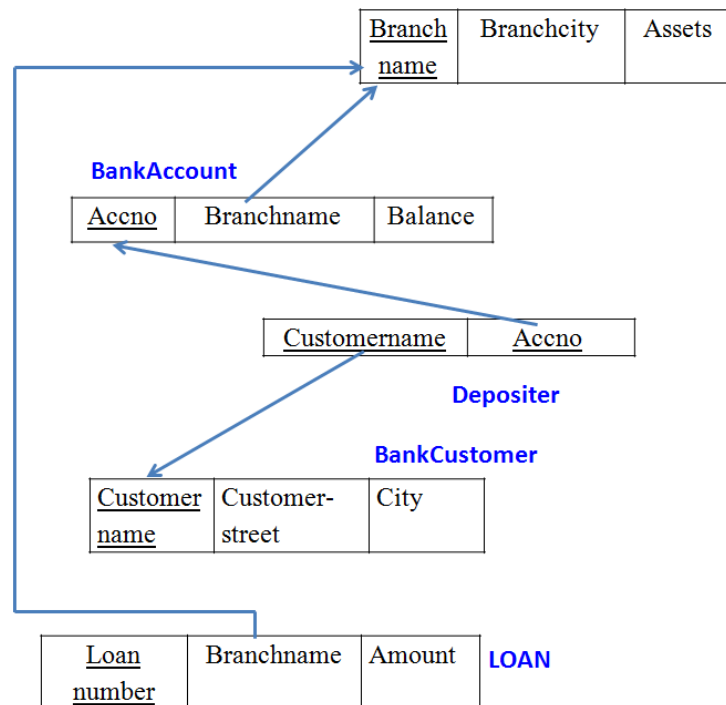
Depositer(customer-name: String, accno: int)

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

- Create the above tables by properly specifying the primary keys and the foreign keys.
- Enter at least five tuples for each relation.
- Find all the customers who have at least two accounts at the *Main* branch (ex. SBI_ResidencyRoad).
- Find all the customers who have an account at *all* the branches located in a specific city (Ex. Delhi).
- 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



Sample Table data

Branch

BRANCHNAME	BRANCHCITY	ASSETS
SBI_Chamrajpet	Bangalore	50000
SBI_ResidencyRoad	Bangalore	10000
SBI_ShivajiRoad	Bombay	20000
SBI_ParlimentRoad	Delhi	10000
SBI_Jantarmentar	Delhi	20000

BankAccount

ACCNO	BRANCHNAME	BALANCE
1	SBI_Chamrajpet	2000
2	SBI_ResidencyRoad	5000
3	SBI_ShivajiRoad	6000
4	SBI_ParlimentRoad	9000
5	SBI_Jantarmentar	8000
6	SBI_ShivajiRoad	4000
8	SBI_ResidencyRoad	4000
9	SBI_ParlimentRoad	3000
10	SBI_ResidencyRoad	5000
11	SBI_Jantarmentar	2000

BankCustomer

CUSTOMERNAME	CUSTOMERSTREET	CUSTOMERCITY
Avinash	Bull_Temple_Road	Bangalore
Dinesh	Bannergatta_Road	Bangalore
Mohan	NationalCollege_Road	Bangalore
Nikil	Akbar_Road	Delhi
Ravi	Prithviraj_Road	Delhi

Depositer

CUSTOMERNAME	ACCNO
Avinash	1
Dinesh	2
Nikil	4
Ravi	5
Avinash	8
Nikil	9
Dinesh	10
Nikil	11

Loan

LOANNUMBER	BRANCHNAME	AMOUNT
1	SBI_Chamrajpet	1000
2	SBI_ResidencyRoad	2000
3	SBI_ShivajiRoad	3000
4	SBI_ParlimentRoad	4000
5	SBI_Jantarmentar	5000

create database sample11;

use sample11;

CREATE TABLE branch

(branch_name VARCHAR(20),

branch_city VARCHAR(20),

assets REAL,

PRIMARY KEY(branch_name)

);

CREATE TABLE accounts

```
( acc_no INT,  
branch_name VARCHAR(50),  
balance REAL,  
PRIMARY KEY(acc_no),  
FOREIGN KEY(branch_name) REFERENCES branch(branch_name)  
ON UPDATE CASCADE ON DELETE CASCADE  
);
```

```
CREATE TABLE customer  
( customer_name VARCHAR(20),  
customer_street VARCHAR(50),  
customer_city VARCHAR(20),  
PRIMARY KEY(customer_name)  
);
```

```
CREATE TABLE depositor  
( customer_name VARCHAR(20),  
acc_no INT,  
PRIMARY KEY(customer_name, acc_no),  
FOREIGN KEY(customer_name) REFERENCES customer(customer_name)  
ON UPDATE CASCADE ON DELETE CASCADE,  
FOREIGN KEY(acc_no) REFERENCES accounts(acc_no)  
ON UPDATE CASCADE ON DELETE CASCADE  
);
```

```
CREATE TABLE loan  
( loan_number INT,  
branch_name VARCHAR(50),  
amount REAL,
```

```
PRIMARY KEY(loan_number),  
FOREIGN KEY(branch_name) REFERENCES branch(branch_name)  
ON UPDATE CASCADE ON DELETE CASCADE  
);
```

```
INSERT INTO branch(branch_name,branch_city,assets) VALUES  
('SBI_Chamrajpet','Bangalore',50000),('SBI_ResidencyRoad','Bangalore',10000),('SBI_ShivajiRoad','Bomb  
ay',20000),('SBI_ParliamentRoad','Delhi',10000),('SBI_Jantarmanatar','Delhi',20000);
```




```
INSERT INTO accounts(acc_no,branch_name,balance) VALUES  
(1,'SBI_Chamrajpet',2000),(2,'SBI_ResidencyRoad',5000),(3,'SBI_ShivajiRoad',6000),(4,'SBI_ParliamentRoa  
d',9000),(5,'SBI_Jantarmanatar',8000),(6,'SBI_ShivajiRoad',4000),(8,'SBI_ResidencyRoad',4000),(9,'SBI_Par  
liamentRoad',3000),(10,'SBI_ResidencyRoad',5000),(11,'SBI_Jantarmanatar',2000);
```

```
INSERT INTO customer(customer_name,customer_street,customer_city) VALUES  
('Avinash','Bull_Temple_Road','Bangalore'),('Dinesh','Bannerghatta_Road','Bangalore'),('Mohan','National  
College_Road','Bangalore'),('Nikil','Akbar_Road','Delhi'),('Ravi','Prithviraj_Road','Delhi');
```




```
INSERT INTO depositor(customer_name,acc_no) VALUES  
('Avinash',1),('Dinesh',2),('Nikil',4),('Ravi',5),('Avinash',8),('Nikil',9),('Dinesh',10),('Nikil',11);
```

```
INSERT INTO loan(loan_number,branch_name,amount) VALUES  
(1,'SBI_Chamrajpet',1000),(2,'SBI_ResidencyRoad',2000),(3,'SBI_ShivajiRoad',3000),(4,'SBI_ParliamentRoa  
d',4000),(5,'SBI_Jantarmanatar',5000);
```




```
SELECT * FROM branch;
```

Result Grid			
Filter Rows: <input type="text"/>			
Edit:    Exp			
	branch_name	branch_city	assets
▶	SBI_Chamrajpet	Bangalore	50000
	SBI_Jantarmanatar	Delhi	20000
	SBI_ParliamentRoad	Delhi	10000
	SBI_ResidencyRoad	Bangalore	10000
	SBI_ShivajiRoad	Bombay	20000
•	NULL	NULL	NULL

SELECT * FROM accounts;

Result Grid			
Filter Rows: <input type="text"/>			
Edit:    Export/Imp			
	acc_no	branch_name	balance
▶	1	SBI_Chamrajpet	2000
	2	SBI_ResidencyRoad	5000
	4	SBI_ParliamentRoad	9000
	5	SBI_Jantarmanatar	8000
	8	SBI_ResidencyRoad	4000
	9	SBI_ParliamentRoad	3000
	10	SBI_ResidencyRoad	5000
	11	SBI_Jantarmanatar	2000
•	NULL	NULL	NULL

SELECT * FROM customer;

Result Grid			
Filter Rows: <input type="text"/>			
Edit:    Export/Imp			
	customer_name	customer_street	customer_city
▶	Avinash	Bull_Temple_Road	Bangalore
	Dinesh	Bannergatta_Road	Bangalore
	Mohan	NationalCollege_Road	Bangalore
	Nikil	Akbar_Road	Delhi
	Ravi	Prithviraj_Road	Delhi
•	NULL	NULL	NULL




SELECT * FROM depositor;

Result Grid			Filter Rows:	Edit:	Export
	customer_name	acc_no			
▶	Avinash	1			
	Dinesh	2			
	Nikil	4			
	Ravi	5			
	Avinash	8			
	Nikil	9			
	Dinesh	10			
	Nikil	11			
•	NULL	NULL			

SELECT * FROM loan;

Result Grid				Filter Rows:	Edit:	Export
	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			

SELECT * FROM customer WHERE customer_name IN(SELECT customer_name FROM depositor group by customer_name having COUNT(customer_name)>=2);

Result Grid			
Filter Rows: <input type="text"/>			
Edit:    Export			
	customer_name	customer_street	customer_city
▶	Avinash	Bull_Temple_Road	Bangalore
	Dinesh	Bannergatta_Road	Bangalore
	Nikil	Akbar_Road	Delhi
•	NULL	NULL	NULL

```
SELECT d.customer_name FROM accounts a, depositor d, branch b WHERE d.acc_no=a.acc_no AND
b.branch_name=a.branch_name AND b.branch_city="Delhi" GROUP BY d.customer_name having
count(distinct b.branch_name)=(SELECT COUNT(branch_name) FROM branch WHERE
branch_city="Delhi");
```

Result Grid	
Filter Rows: <input type="text"/>	
Export	
	customer_name
▶	Nikil

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

PROGRAM 3: SUPPLIER DATABASE

Consider the following schema:

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

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

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

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

Write the following queries in SQL:

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

Schema Diagram

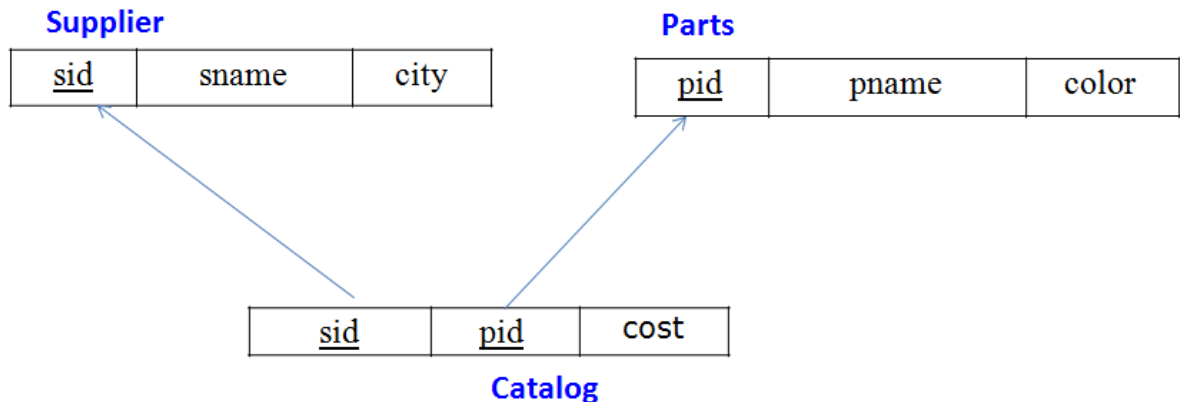


Table Data

SUPPLIERS		
SID	SNAME	CITY

10001	Acme Widget	Bangalore
10002	Johns	Kolkata
10003	Vimal	Mumbai
10004	Reliance	Delhi

PARTS		
PID	PNAME	COLOR

20001	Book	Red
20002	Pen	Red
20003	Pencil	Green
20004	Mobile	Green
20005	Charger	Black

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

```

create database supplier;
use supplier;
CREATE TABLE suppliers(
    sid INT,
    sname VARCHAR(20),
    address VARCHAR(50),
    PRIMARY KEY (sid)
);
CREATE TABLE parts(
    pid INT,
    pname VARCHAR(20),
    color VARCHAR(10),
    PRIMARY KEY (pid)
);
CREATE TABLE catalog(
    sid INT,
    pid INT,
    cost REAL,
    PRIMARY KEY(sid,pid),

```

```

FOREIGN KEY(sid) REFERENCES suppliers(sid)
ON delete CASCADE ON update CASCADE,
FOREIGN KEY(pid) REFERENCES parts(pid)
ON delete CASCADE ON update CASCADE
);

```

```

insert into suppliers values (10001,'Acme Widget','Bangalore'), (10002,'Johns','Kolkata'),
(10003,'Vimal','Mumbai'),(10004,'Reliance','Delhi');
insert into parts values
(20001,'Book','Red'),(20002,'Pen','Red'),(20003,'Pencil','Green'),(20004,'Mobile','Green'),(20005,
'Charger','Black');
insert into catalog
values(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);

```

```

SELECT * FROM suppliers;

```

Result Grid			
Filter Rows: <input type="text"/>			
	sid	sname	address
▶	10001	Acme Widget	Bangalore
	10002	Johns	Kolkata
	10003	Vimal	Mumbai
	10004	Reliance	Delhi
✱	NULL	NULL	NULL

```

SELECT * FROM parts;

```

Result Grid			
Filter Rows: <input type="text"/>			
	pid	pname	color
▶	20001	Book	Red
	20002	Pen	Red
	20003	Pencil	Green
	20004	Mobile	Green
	20005	Charger	Black
✱	NULL	NULL	NULL

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

SELECT DISTINCT p.pname FROM parts p, catalog c WHERE p.pid = c.pid;

	pname
▶	Book
	Pen
	Pencil
	Mobile
	Charger

select suppliers.sname from suppliers where suppliers.sid in(select catalog.sid from catalog
inner join
parts on catalog.pid=parts.pid group by catalog.sid having count(*)=(select count(parts.pid) from
parts));

	sname
▶	Acme Widget

```
select suppliers.sname from suppliers where suppliers.sid in (select catalog.sid from catalog
inner join
parts on catalog.pid=parts.pid where catalog.pid in (select parts.pid from parts where
parts.color='Red') group by catalog.sid having count(*)=(select count(parts.color) from parts
where
parts.color='Red'));
```

Result Grid	Filter Rows:	Export:
sname		
▶ Acme Widget		
Johns		




```
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:
pname		
▶ Mobile		
Charger		

```
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		

```
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:	<input type="text"/>	Export:	
	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, meetsat: time, room: string, fid: integer)

ENROLLED(snum: integer, cname:string)

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

The meaning of these relations is straightforward; for example, Enrolled has one record per student-class pair such that the student is enrolled in the class. Level(lvl) is a two character code with 4 different values (example: Junior: JR etc)

Write the following queries in SQL.

No duplicates should be printed in any of the answers.

- i. Find the names of all Juniors (level = JR) who are enrolled in a class taught by “name”
- ii. Find the names of all classes that either meet in room R128 or have five or more Students enrolled.
- iii. Find the names of all students who are enrolled in two classes that meet at the same time.
- iv. Find the names of faculty members who teach in every room in which some class is taught.
- v. Find the names of faculty members for whom the combined enrollment of the courses that they teach is less than five.
- vi. Find the names of students who are not enrolled in any class.
- vii. For each age value that appears in Students, find the level value that appears most often. For example, if there are more FR level students aged 18 than SR, JR, or SO students aged 18, you should print the pair (18, FR).

SQL> select * from student;

SNUM	SNAME	MA	LV	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

SQL> select * from faculty;

FID	FNAME	DEPTID

11	Harish	1000
12	MV	1000
13	Mira	1001
14	Shiva	1002
15	Nupur	1000

SQL> select * from class;

CNAME	METTS_A	ROOM	FID
Class1	12/11/15 10:15:16.000000	R1	14
Class10	12/11/15 10:15:16.000000	R128	14
Class2	12/11/15 10:15:20.000000	R2	12
Class3	12/11/15 10:15:25.000000	R3	11
Class4	12/11/15 20:15:20.000000	R4	14
Class5	12/11/15 20:15:20.000000	R3	15
Class6	12/11/15 13:20:20.000000	R2	14
Class7	12/11/15 10:10:10.000000	R3	14

SQL> select * from enrolled;

SNUM	CNAME

1	class1
2	class1
3	class3
4	class3
5	class4

Program:-

```
create database studentfaculty4;
```

```
use studentfaculty4;
```

```
create table STUDENT(
```

```
  snum int,
```

```
  sname varchar(60),
```

```
  major varchar(50),
```

```
  lvl varchar(50),
```

```
  age int,
```

```
  primary key(snum)
```

```
);
```

```
create table CLASS(
```

```
  cname varchar(60),
```

```
  meetsat timestamp,
```

```
  room varchar(60),
```

```
  fid int,
```

```
  primary key (cname)
```

```
);
```

```
create table enrolled(
```

```
  snum int,
```

```
  cname varchar(60),
```

```
  primary key(snum,cname),
```

```
  foreign key(snum) references STUDENT(snum)
```

```
  on update cascade on delete cascade,
```

```
  foreign key(cname) references CLASS(cname)
```


on update cascade on delete cascade

);

create table FACULTY(

fid int,

fname varchar(60),

deptid int,

primary key(fid)

);

insert into STUDENT values (1,'Jhon','CS','Jr',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);

update STUDENT set lvl='Sr' where sname='Jhon';

insert into CLASS values ('Class1','12/11/15 10:15:16.00000','R1',14);

select * from CLASS;

delete from CLASS where cname='Class1';

select * from CLASS;

insert into CLASS values ('Class1','15/11/12 10:15:16.00000','R1',14);

select * from CLASS;

insert into CLASS values ('Class10','15/11/12 10:15:16.00000','R128',14), ('Class2','15/11/12
10:15:20.00000','R2',12),

('Class3','15/11/12 10:15:25.00000','R3',11), ('Class4','15/11/12 10:15:20.00000','R4',14),

('Class5','15/11/12 10:15:20.00000','R3',15),

('Class6','15/11/12 13:20:20.00000','R2',14), ('Class7','15/11/12 10:10:10.00000','R3',14);

insert into ENROLLED values (1,'Class1'),(2,'Class1'),(3,'Class3'),(4,'Class3'),(5,'Class4');

insert into FACULTY values

(11,'Harish',1000),(12,'MV',1000),(13,'Mira',1001),(14,'Shiva',1002),(15,'Nupur',1000);

select * from STUDENT;

	snum	sname	major	lvl	age
▶	1	Jhon	CS	Jr	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

select * from CLASS;

	cname	meetsat	room	fid
▶	Class1	2015-11-12 10:15:16	R1	14
	Class10	2015-11-12 10:15:16	R128	14
	Class2	2015-11-12 10:15:20	R2	12
	Class3	2015-11-12 10:15:25	R3	11
	Class4	2015-11-12 10:15:20	R4	14
	Class5	2015-11-12 10:15:20	R3	15
	Class6	2015-11-12 13:20:20	R2	14
	Class7	2015-11-12 10:10:10	R3	14
	NULL	NULL	NULL	NULL

select * from FACULTY;

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

select * from ENROLLED;

	snum	cname
▶	1	Class1
	2	Class1
	3	Class3
	4	Class3
	5	Class4
✱	NULL	NULL

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 = 'Shiva' AND s.lvl = 'Jr';

	sname
▶	Jhon
	Smith
	Rahul

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(e.snum)>4);

	cname
▶	Class10
✱	NULL

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);

	fname
▶	Harish
	MV
	Mira
	Shiva
	Nupur

SELECT DISTINCT s.sname FROM STUDENT s WHERE s.snum NOT IN(SELECT e.snum FROM ENROLLED e);

	sname
▶	Rita

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	Jr
	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.

- Find the names of aircraft such that all pilots certified to operate them have salaries more than Rs.80,000.
- 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.
- Find the names of pilots whose salary is less than the price of the cheapest route from Bengaluru to Frankfurt.
- 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.
- Find the names of pilots certified for some Boeing aircraft.
- Find the aids of all aircraft that can be used on routes from Bengaluru to New Delhi.
- 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.

SQL> select * from Flights;

FLNO	FFROM	TO	DISTANCE	DEPARTS	ARRIVES	PRICE
101	Bangalore	Delhi	2500	13-MAY-05 07:15:31.000000	AM13-MAY-05 07:15:31.000000	A 5000
102	Bangalore	Lucknow	3000	05/05/13 07:15:31.000000	05/05/13 11:15:31.000000	6000
103	Lucknow	Delhi	500	05/05/13 12:15:31.000000	05/05/13 17:15:31.000000	3000
107	Bangalore	Frankfurt	8000	05/05/13 07:15:31.000000	05/05/13 22:15:31.000000	60000
104	Bangalore	Frankfurt	8500	05/05/13 07:15:31.000000	05/05/13 23:15:31.000000	75000
105	Kolkata	Delhi	3400	05/05/13 07:15:31.000000	05/05/13 09:15:31.000000	7000

SQL> 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

7 rows selected.

SQL> select * from Certified;

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

EID	AID
703	105
704	105
705	103

14 rows selected.

SQL> 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

7 rows selected.

Program :-

```
CREATE DATABASE AIRLINE_FLIGHT_DATABASE;
```

```
USE AIRLINE_FLIGHT_DATABASE;
```

```
CREATE TABLE FLIGHTS
```

```
(  
    flno int,  
    ffrom varchar(40),  
    tto varchar(40),  
    distance int,  
    departs datetime,  
    arrives datetime,  
    price int,  
    primary key(flno)  
);
```

```
CREATE TABLE AIRCRAFT
```

```
(  
    aid int,  
    aname varchar(40),  
    cruisingrange int,  
    primary key(aid)  
);
```

```
CREATE TABLE EMPLOYEES
```

```
(  
    eid int,
```



```
    ename varchar(40),  
    salary int,  
    primary key(eid)  
);
```

CREATE TABLE CERTIFIED

```
(  
    eid int,  
    aid int,  
    FOREIGN KEY(aid) REFERENCES AIRCRAFT(aid),  
    FOREIGN KEY(eid) REFERENCES EMPLOYEES(eid)  
);
```

INSERT INTO FLIGHTS

```
VALUES (101,"Bangalore","Delhi",2500,'2005-05-13:07:15:31.000000','2005-05-  
13:07:15:31.000000',5000),  
    (102,"Bangalore","Lucknow",3000,'2013-05-05:07:15:31.000000','2013-05-  
05:11:15:31.000000',6000),  
    (103,"Lucknow","Delhi",500,'2013-05-05:12:15:31.000000','2013-05-05:17:15:31.000000',3000),  
    (107,"Bangalore","Frankfurt",8000,'2013-05-05:07:15:31.000000','2013-05-  
05:22:15:31.000000',60000),  
    (104,"Bangalore","Frankfurt",8500,'2013-05-05:07:15:31.000000','2013-05-  
05:23:15:31.000000',75000),  
    (105,"Kolkata","Delhi",3400,'2013-05-05:07:15:31.000000','2013-05-05:09:15:31.000000',7000);  
SELECT * FROM FLIGHTS;
```

	fno	ffrom	tto	distance	departs	arrives	price
▶	101	Bangalore	Delhi	2500	2005-05-13 07:15:31	2005-05-13 07:15:31	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
	107	Bangalore	Frankfurt	8000	2013-05-05 07:15:31	2013-05-05 22:15:31	60000
•	NULL	NULL	NULL	NULL	NULL	NULL	NULL

INSERT INTO AIRCRAFT

VALUES (101,747,3000),(102,"Boeing",900),(103,647,800),(104,"Dreamliner",10000),
(105,"Boeing",3500),(106,707,1500),(107,"Dream",120000);

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),(702,"B",100000),(703,"C",150000),(704,"D",90000),
(705,"E",40000),(706,"F",60000),(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),(701,102),(701,106),(701,105),(702,104),(703,104),(704,104),(702,107),
(703,107),(704,107),(702,101),(703,105),(704,105),(705,103);

SELECT * FROM CERTIFIED;

	eid	aid
▶	701	101
	701	102
	701	106
	701	105
	702	104
	703	104
	704	104
	702	107
	703	107
	704	107
	702	101
	703	105
	704	105
	705	103

SELECT distinct a.aname

FROM AIRCRAFT a,EMPLOYEES e,CERTIFIED c

WHERE a.aid=c.aid and e.eid=c.eid and e.salary>80000;

	aname
▶	Dreamliner
	Dream
	747
	Boeing

```

SELECT e.eid,e.ename,max(a. cruisingrange)
FROM EMPLOYEES e,CERTIFIED c,AIRCRAFT a
WHERE e.eid=c.eid and a.aid=c.aid
group by e.ename
having count(c.aid)>3;

```

	eid	ename	max(a. cruisingrange)
▶	701	A	3500

```

SELECT e.ename
FROM EMPLOYEES e
WHERE salary < (select min(price)
                from FLIGHTS
                where ffrom="Bangalore" and tto="Frankfurt");

```

	ename
▶	A
	E

```

SELECT a.aname,a.cruisingrange,avg(e.salary)

```

FROM AIRCRAFT a,EMPLOYEES e,CERTIFIED c

WHERE c.eid=e.eid and c.aid=a.aid

group by a.aname

having a.cruisingrange > 1000;

	aname	cruisingrange	avg(e.salary)
▶	747	3000	75000.0000
	Dreamliner	10000	113333.3333
	707	1500	50000.0000
	Dream	120000	113333.3333

SELECT distinct e.ename

FROM EMPLOYEES e,CERTIFIED c,AIRCRAFT a

WHERE e.eid=c.eid and a.aid=c.aid and aname like "Boeing";

	ename
▶	A
	C
	D

SELECT a.aid

FROM AIRCRAFT a

WHERE a. cruisingrange >= (select distance

from FLIGHTS

where ffrom="Bangalore" and tto="Delhi");

	aid
▶	101
	104
	105
	107
•	NULL

```
SELECT f.ffrom,f.tto,f.arrives
```

```
FROM   FLIGHTS f
```

```
WHERE (f.ffrom="Bangalore" and f.tto=(select ffrom
```

```
from FLIGHTS
```

```
where tto="Kolkata")) or f.tto="Kolkata";
```

	ffrom	tto	arrives