Lab questions for RDBMS LAB (17CS502)

Mark Distribution

Table creation and value insertion	Viva	Query 1	Query 2	Query 3
10	**10**	**10**	**10**	**10**

Note:

1. Create the tables by properly specifying the primary keys and the foreign keys.
2. Enter at least four tuples for each relation

I.Insurance Database

Consider the Insurance database given below.

PERSON (<u>driver – id #:</u> String, name: string, address: string)

CAR (<u>regno</u>: string, model: string, year: int)

ACCIDENT (<u>report-number</u>: int, <u>accd-date</u>: date, location: string)

OWNS (<u>driver-id #:</u> string, <u>regno</u>: string)

PARTICIPATED (<u>driver-id</u>: string, <u>Regno</u>: string, <u>report-number</u>: int, damage amount: int)

1. Find the total number of people who owned cars that were involved in accidents in 1989.
2. Find the number of accidents in which the cars belonging to "John Smith" were involved.
3. Update the damage amount for the car with reg number "KA-12" in the accident with report number "1" to $3000.

II. Order Database

Consider the following relations for an order processing database application in a company:

CUSTOMER (<u>cust #:</u> int, cname: string, city: string)

ORDER (order #: int, odate: date, cust #: int, ord-Amt: int)

ORDER – ITEM (order #: int, item #: int, qty: int)

ITEM (item #: int, unit price: int)

SHIPMENT (order #: int, warehouse#: int, ship-date: date)

WAREHOUSE (warehouse #: int, city: string)

1. Produce a listing: CUSTNAME, #oforders, AVG_ORDER_AMT, where the middle column is the total numbers of orders by the customer and the last column is the average order amount for that customer.
2. For each item that has more than two orders , list the item, number of orders that are shipped from atleast two warehouses and total quantity of items shipped
3. List the customers who have ordered for every item that the company produces

III. Consider the following database of student enrollment in courses & books adopted for each course:

STUDENT (regno: string, name: string, major: string, bdate: date)

COURSE (course #: int, cname: string, dept: string)

ENROLL (regno: string, course#: int, sem: int marks: int)

BOOK _ ADOPTION (course#: int, sem: int, book-ISBN: int)

TEXT (book-ISBN: int, book-title: string, publisher: string, author: string)

1. Produce a list of text books (include Course #, Book-ISBN,Book-title) in the alphabetical order for courses offered by th 'CS' department that use more than two books.
2. List any department that has all its adopted books published by a specific publisher
3. List the bookISBNs and book titles of the department that has maximum number of students

VI. The following tables are maintained by a book dealer:

AUTHOR (author-id: int, name: string, city: string, country: string)

PUBLISHER (publisher-id: int, name: string, city: string, country: string)

CATALOG (book-id: int, title: string, author-id: int, publisher-id: int, category-id: int, year: int, price: int)

CATEGORY (category-id: int, description: string)

ORDER-DETAILS (order-no: int, book-id: int, quantity: int)

1. Find the author of the book which has maximum sales.
2. Increase the price of the books published by a specific publisher by 10%
3. Find the number of orders for the book that has minimum sales.

V. Consider the following database for a banking enterprise:

BRANCH (<u>branch-name:</u> string, branch-city: string, assets: real)

ACCOUNT (<u>accno:</u> int, branch-name: string, balance: real)

DEPOSITOR (<u>customer-name:</u> string, <u>accno:</u> int)

CUSTOMER (<u>customer-name:</u> string, customer-street: string, customer-city: string)

LOAN (<u>loan-number:</u> int, branch-name: string, amount: real)

BORROWER (<u>customer-name:</u> string, <u>loan-number:</u> int)

1. Find all the customers who have atleast 2 accounts at all the branches located in a specific city.
2. Find all the customers who have accounts in atleast 1 branch located in all the cities
3. Find all the customers who have accounts in atleast 2 branches located in a specific city.

```
create Database Insurance

use Insurance


CREATE TABLE PERSON (
                driverid varchar(10),
                fname char(15) not null,
                address varchar(30),
                primary key (driverid)
                )


insert into  PERSON values ('111','John Smith' , 'SP Road, Bangalore-
12')
insert into  PERSON values ('112','Ramesh Babu' , 'KP Nagar, Udupi -
13')
insert into  PERSON values ('113','Raju SK' , 'KS Circle, Mangalore-
12')
insert into  PERSON values ('114','Ramesh Babu' , 'AS Road, Bangalore-
14')
insert into  PERSON values ('115','Alica wallace' , 'SS Road, Karkala-
16')



select * from PERSON



CREATE TABLE CAR (
                regno varchar(10),
                model varchar(10)not null,
                cyear int,
                primary key(regno)
           )

insert into  CAR values ('KA-12','FORD' ,1980)
insert into  CAR values ('KA-13','SWIFT' ,1990)
insert into  CAR values ('MH-11','INDIGO' ,1998)
insert into  CAR values ('AP-10','SWIFT' ,1988)
insert into  CAR values ('TN-11','FORD' ,2001)
insert into  CAR values ('TN-12','TOYATA' ,2001)
insert into  CAR values ('MH-14','SWIFT' ,2001)
insert into  CAR values ('KL-15','TOYATA' ,2001)
insert into  CAR values ('KL-4','INDIGO' ,2001)
insert into  CAR values ('AP-05','SANTRO' ,2001)

select * from CAR

CREATE TABLE ACCIDENT  (
                reportno int ,
                accdate datetime,
                location varchar(20),
                primary key(reportno)
                )




insert into  ACCIDENT values (1,'1998-07-22' ,'Nitte')
insert into  ACCIDENT values (2,'1998-07-22','Karkala')
insert into  ACCIDENT values (12,'1998-07-22' ,'Mangalore')
insert into  ACCIDENT values (3,'1998-07-23','Mangalore')
insert into  ACCIDENT values (4,'1990-09-09','Bhatkal')
insert into  ACCIDENT values (5,'2001-02-22' ,'Udupi')
insert into  ACCIDENT values (6,'1990-09-09','Udupi')
insert into  ACCIDENT values (15,'1981-07-22' ,'Udupi')

select * from ACCIDENT
delete from ACCIDENT

insert into  ACCIDENT values (7,'1981-09-09','Karkala')
insert into  ACCIDENT values (8,'1990-09-09','Bhatkal')
insert into  ACCIDENT values (9,'2001-02-22' ,'Udupi')
insert into  ACCIDENT values (10,'1998-02-02','Udupi')
insert into  ACCIDENT values (11,'1998-01-02','Bhatkal')
insert into  ACCIDENT values (13,'1998-07-22','Udupi')
insert into  ACCIDENT values (14,'1998-07-22','Karkala')


delete from ACCIDENT

CREATE TABLE OWNS    (
                driverid varchar(10) ,
                regno varchar(10)
                primary key(driverid,regno)
                foreign key(driverid) references PERSON(driverid)on
delete cascade on update cascade,
                foreign key(regno) references CAR(regno)on delete
cascade on update cascade,
                unique(regno)
               )


drop table OWNS

select * from PERSON
select * from car

insert into  OWNS values ('111','KA-13')
insert into  OWNS values ('111','KA-12')
insert into  OWNS values ('111','MH-11')

insert into  OWNS values ('112','AP-10')
insert into  OWNS values ('112','TN-11')

insert into  OWNS values ('113','TN-12')
insert into  OWNS values ('113','KL-15')

insert into  OWNS values ('114','AP-05')
insert into  OWNS values ('114','KL-4')

insert into  OWNS values ('115','MH-14')


select * from OWNS

delete from OWNS

drop table PARTCIPATED
CREATE TABLE PARTCIPATED (
                   driverid varchar(10) ,
                   regno varchar(10),
                   reportno  int,
                   dmgamt int,
                   primary key(driverid,regno,reportno) ,
                   foreign key(driverid) references PERSON(driverid)on
delete cascade on update cascade,
                   foreign key(regno) references CAR(regno)on delete
cascade on update cascade,
                   foreign key(reportno) references ACCIDENT(reportno)
on delete cascade on update cascade,
                   foreign key(driverid,regno) references
OWNS(driverid,regno),
                   unique(reportno)
                 )


drop table  PARTCIPATED

select * from accident

insert into  PARTCIPATED values ('111','KA-12',1,20000)
insert into  PARTCIPATED values ('111','KA-13',2,10000)
insert into  PARTCIPATED values ('111','KA-12',3,60000)
insert into  PARTCIPATED values ('111','KA-12',4,60000)
insert into  PARTCIPATED values ('111','KA-12',5,60000)
insert into  PARTCIPATED values ('111','KA-12',15,40000)
insert into  PARTCIPATED values ('111','KA-13',6,10000)
insert into  PARTCIPATED values ('111','MH-11',12,20000)

insert into  PARTCIPATED values ('112','AP-10',7,30000)
insert into  PARTCIPATED values ('112','TN-11',8,40000)
insert into  PARTCIPATED values ('112','AP-10',13,20000)
insert into  PARTCIPATED values ('112','TN-11',14,10000)

insert into  PARTCIPATED values ('113','TN-12',9,40000)
insert into  PARTCIPATED values ('113','KL-15',10,50000)
insert into  PARTCIPATED values ('113','TN-12',11,20000)



delete from PARTCIPATED where reportno=5
drop table PERSON
drop table CAR
drop table ACCIDENT
drop table OWNS
drop table PARTCIPATED


select * from PERSON
select * from CAR
select * from ACCIDENT
select * from OWNS
select * from PARTCIPATED

select distinct(location) from ACCIDENT
```

1. Find the total number of people who owned cars that were involved in accidents in 1989.

```
select count (distinct P.driverid)
from accident A, partcipated P
where A.reportno = P.reportno
and A.accdate between  '1998-01-01' and  '1998-12-31'

select count (distinct P.driverid)
from accident A, partcipated P
where A.reportno = P.reportno
and year(A.accdate) = '1998'


select count (distinct P.driverid)
from  partcipated P where P.reportno  in
                                   (
                                        select reportno  from accident
                                        where reportno = P.reportno
and year(accdate)  ='1998')



select count (distinct P.driverid)
from  partcipated P where P.reportno  in
```

```
                                    (
                                         select reportno  from accident
                                         where  year(accdate)  ='1998')
```

2a. Find the number of accidents in which the cars belonging to “John Smith” were involved.

```
select  count (P.reportno) as NO_OF_ACC
from   partcipated P,  person PN
where P.driverid =  PN.driverid
and   PN.fname = 'John Smith'
```

2b. Find the number of accidents in which the cars belonging to specific model were involved.

```
select  count (P.reportno) as NO_OF_ACC
from   partcipated P,  car C
where P.regno =  C.regno
and   C.model  = 'SWIFT'
```

3. Add a new accident to the database; assume any values for required attributes.

We assume the driver was “Ramesh Babu,” although it could be someone else.
Also, we assume “Ramesh Babu” owns one Toyota. First we must find the license of
the given car. Then the participated and accident relations must be updated
in order to both record the accident and tie it to the given car. We assume
values “Berkeley” for location, ’2001-09-01’ for date and date, 4007 for reportnumber
and 3000 for damage amount.

```
insert into accident values (7, '2001-09-01', 'Karkala')

insert into partcipated
```

```
select O.driverid, C.regno, 7, 100000
from person P, owns O, car C
where P.fname = 'Ramesh Babu'
and P.driverid = O.driverid
and O.regno = C.regno
and C.model = 'SWIFT'

select * from partcipated
select * from accident
```

4. Delete the Mazda belonging to “John Smith”.

```
delete from  car
where model = 'INDIGO' and regno in
(select regno
from person P, owns O
where P.fname = 'John Smith' and P.driverid = O.driverid)
```

5. Update the damage amount for the car with reg number “KA-12” in the accident with report number “1” to $3000.

```
update PARTCIPATED  set dmgamt = 29000
where reportno = 1 and  driverid in
(select driverid
from owns
where regno = 'KA-13')



select * from person
select * from car
select * from accident
select * from PARTCIPATED
select * from OWNS

delete from  PARTCIPATED

create view  veiw1  as
select distinct(driverid)
from (
select driverid,regno from OWNS  as T
where not exists
              (
                      select distinct (location) from accident  A
                      where  location not in
                           (
                                select location from accident
A1,PARTCIPATED P1
                                           where A1.reportno =
P1.reportno and P1.driverid = T.driverid
```

```
                                and P1.regno=T.regno

                                 )
                  )

         )   as T




select driverid ,fname  from  person P
where  exists
 (
     select regno from car  where not exists
            (
                    select distinct (location) from accident
                    where  location not in
                        (
                            select location from accident
A,PARTCIPATED PT
                                      where A.reportno =
PT.reportno and PT.driverid = P.driverid
                                  )
            )
 )



select P.driverid , P.regno , count(*) as no_of_accidents from
PARTCIPATED P
where P.driverid  in ( select driverid from OWNS group by driverid
having count(*) >= 2)

                 and
       P.regno  in  ( select regno from  PARTCIPATED where driverid =
P.driverid group by regno
                     having sum(dmgamt) >= all ( select  sum(dmgamt)
from  PARTCIPATED

where driverid = P.driverid group by regno))

 group by P.driverid , P.regno


select driverid,count(*) from OWNS  group by driverid
select driverid,regno from OWNS   where driverid =112
select * from person
```

1. List the names of people who owned cars that were involved in accidents in 2008.

```
select distinct fname
```

```
from PERSON P,PARTCIPATED R,ACCIDENT A
where P.driverid=R.driverid and R.reportno=A.reportno and
year(A.accdate)='1998'
```

2.   Find the name of owner and his car that has maximum number of accidents in 2008

```
select distinct P.fname,O.regno
from PERSON P,OWNS O,ACCIDENT A,PARTCIPATED R
where P.driverid=R.driverid and O.regno=R.regno and
R.reportno=A.reportno and year(A.accdate)='1998'
group by P.fname,O.regno
having count(*) >= all ( select count(distinct A1.reportno)
                        from OWNS O1,PERSON P1,ACCIDENT A1,PARTCIPATED
R1
                        where P1.driverid=R1.driverid and
O1.regno=R1.regno and R1.reportno=A1.reportno and
year(A1.accdate)='1998'
                        group by P1.fname,O1.regno)
```

3.   List the name of owners who own atleast two TOYOTA cars.

```
select P.fname
from PERSON P,OWNS O,CAR C
where P.driverid=O.driverid and C.regno=O.regno and C.model='TOYATA'
group by P.fname
having count(C.regno)>=2
```

4.   List the name of owner who owns maximum TOYOTA cars.

```
select P.fname
from PERSON P,CAR C,OWNS O
where P.driverid=O.driverid and C.regno=O.regno and C.model='TOYATA'
group by P.fname
having count(C.regno)>=all(select count(distinct C.regno)
                           from PERSON P1,CAR C1,OWNS O1
                           where P1.driverid=O1.driverid and
C1.regno=O1.regno and C1.model='TOYATA'
                           group by P1.fname)
```

5.   Find the name of owner who owns cars having minimum damage amount for accidents in 2008

```
select P.fname,R.dmgamt
from PERSON P,OWNS C,PARTCIPATED R,ACCIDENT A
where P.driverid=R.driverid and C.regno=R.regno and
R.reportno=A.reportno and year(A.accdate)='1998'
group by P.fname,R.dmgamt
having R.dmgamt in (select min(dmgamt)
                    from PARTCIPATED B,ACCIDENT C
                    where B.reportno=C.reportno and
year(C.accdate)='1998')
```

6.   List the names of   owners whose every car is involved in accidents in 2008

```
select P.fname
from PERSON P
where not exists(
                  select Z.regno from OWNS Z
                  where P.driverid=Z.driverid and
                   Z.regno not in
                  (select C.regno
                   from CAR C,ACCIDENT A,PARTCIPATED R
                   where P.driverid=R.driverid and C.regno=R.regno
                   and A.reportno=R.reportno and
year(A.accdate)='1998'))
```

7.   List the names of   owners whose every car is involved in accidents on a specific day.

```
select P.fname
from PERSON P
where not exists( select O.regno
                  from OWNS O
                  where O.driverid=P.driverid
                  and O.regno not in( select R.regno
                                      from PARTCIPATED R,CAR
C,ACCIDENT A
                                      where R.driverid=P.driverid and
C.regno=R.regno
                                      and A.reportno=R.reportno and
A.accdate='22 july 1998'))
```

8.   List the names of people who owned cars that were involved in accidents on a specific day and
 atleast two cars of each owner are involved.

```
 select P.fname
 from PERSON P,OWNS C,PARTCIPATED R,ACCIDENT A
 where P.driverid=R.driverid and C.regno=R.regno and
R.reportno=A.reportno
 and A.accdate='22 july 1998'
 group by P.fname
 having count(C.regno)>=2
```

 9.   List Owner-Name, Car Regno, Number of accidents, and average damage amount for the year 2008.

```
 select P.fname,C.regno,count(A.reportno),avg(R.dmgamt)
 from PERSON P,OWNS C,PARTCIPATED R,ACCIDENT A
 where P.driverid=R.driverid and C.regno=R.regno and
A.reportno=R.reportno and year(A.accdate)='1998'
 group by P.fname,C.regno
```

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

```
select count(distinct O.driverid)
from OWNS O,PARTCIPATED P,ACCIDENT A
where O.driverid=P.driverid and P.reportno=A.reportno and
year(A.accdate)='2001'
```

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

```
select count(P.reportno)
from CAR C,PARTCIPATED P
where C.regno=P.regno and C.model='TOYATA'
```

13.   Find the location at which maximum accidents occur in 2008

```
select A.location
from ACCIDENT A,PARTCIPATED P
where A.reportno=P.reportno and year(A.accdate)='1998'
group by A.location
having count(A.reportno)>=ALL( select count(A1.reportno)
                               from ACCIDENT A1,PARTCIPATED P1
                               where A1.reportno=P1.reportno and
year(A1.accdate)='1998'
                               group by A1.location)
```

14.   Find the people who owned cars that were involved in accidents at every location.

```
select P.fname
from PERSON P
where not exists( select distinct A.location
                  from ACCIDENT A
                  where A.location not in( select A1.location
                                           from ACCIDENT A1,PARTCIPATED
P1,OWNS O
                                           where A1.reportno=P1.reportno
and P1.driverid=P.driverid
                                           and P1.regno=O.regno))
```

15. Find the number of owners whose every car is involved in accidents in 2008.

```
select count(distinct O.driverid) as No_of_Owners
from OWNS O
where not exists( select C.regno
                  from CAR C
                  where C.regno=O.regno
                  and C.regno not in( select distinct P.regno
                                      from ACCIDENT A,PARTCIPATED P
```

```
                                    where P.regno=O.regno and
A.reportno=P.reportno
                                    and year(A.accdate)='1998'))
```

16. Find the location at which maximum number of Mazda cars are involved in accidents

```
 select A.location
 from ACCIDENT A,PARTCIPATED P,CAR C
 where A.reportno=P.reportno and C.regno=P.regno and C.model='SWIFT'
 group by A.location
 having count(distinct C.regno)>=ALL( select count(distinct C1.regno)
                                    from CAR C1,ACCIDENT
A1,PARTCIPATED P1
                                    where C1.regno=P1.regno and
A1.reportno=P1.reportno and C1.model='SWIFT'
                                    group by A1.location)
```

1)list the damage amount for each car

```
select C.regno,C.model,sum(P.dmgamt)
from CAR C,PARTCIPATED P
where C.regno=P.regno
group by C.regno,C.model
```

2)list the owner/owners of the car with the maximum damage amount

```
select P.fname,R.dmgamt
from OWNS O,PERSON P,PARTCIPATED R
where P.driverid=O.driverid and R.driverid=O.driverid and
R.regno=O.regno
group by P.fname,R.dmgamt
having R.dmgamt=(select max(dmgamt) from PARTCIPATED)
```

3)give the details of all the owners(ownerid,name,regno,model)

```
select O.driverid,P.fname,C.regno,C.model
from OWNS O,PERSON P,CAR C
where P.driverid=O.driverid and O.regno=C.regno
```

4)find the no.of cars owned by each owner

```
select O.driverid,P.fname,count(C.regno)
from PERSON P,OWNS O,CAR C
where O.driverid=P.driverid and C.regno=O.regno
group by O.driverid,P.fname
```

5)Give the details of each car showing the regno,model, no. of accident

```
select C.regno,C.model,count(P.reportno)
```

```
from CAR C,PARTCIPATED P
where C.regno=P.regno
group by C.regno,C.model
```

6)Give the details of the owners who owned 2 or more cars and registered after 2000
    and total damage amount for a car is between 10,000 and 25,000

```
select P.driverid,P.fname
from PERSON P,PARTCIPATED R,CAR C
where C.cyear > 1979 and R.driverid=P.driverid  and C.regno=R.regno
group by P.driverid,P.fname
having (sum(R.dmgamt)>=10000 and sum(dmgamt)<=210000)and
count(C.regno)>=2
```

7)Give the owners details which include ownerid,name,car model,no.of cars owned.

```
select P.driverid,P.fname,C.model,count(C.regno)
from OWNS O,CAR C,PERSON P
where P.driverid=O.driverid and C.regno=O.regno
group by P.driverid,P.fname,C.model
```

4.   Find the names of owners whose atleast one car is involved in accidents every year.

```
select P.fname
from PERSON P
where not exists( select distinct year(A.accdate)
                  from ACCIDENT A
                  where year(A.accdate) not in( select distinct
year(A1.accdate)
                                                from ACCIDENT
A1,PARTCIPATED P1,OWNS O
                                                where
A1.reportno=P1.reportno and O.driverid=P.driverid
                                                and
P1.driverid=O.driverid))
```

```
create database ord_proc

use ord_proc

CREATE TABLE CUSTOMER (
                custid int,
                cname char(15) not null,
                city varchar(30),
                primary key (custid)
                )

select count(*) as No_Of_Emp from  CUSTOMER

insert into CUSTOMER values (111,'John Smith', 'Karkala')
insert into CUSTOMER values (112,'Ramesh N', 'Nitte')
insert into CUSTOMER values (113,'Franklin', 'Karkala')
insert into CUSTOMER values (114,'Alica', 'mangalore')
insert into CUSTOMER values (115,'Raju', 'Udupi')

drop table customer
drop table c_order
drop table item
drop TABLE ORDER_ITEM
drop table shipment
drop table warehouse

CREATE TABLE C_ORDER (
                orderid int,
                odate datetime,
                custid int,
                ordamt int,
                primary key (orderid)  ,
                foreign key(custid) references CUSTOMER(custid)on
delete cascade on update cascade
                )

insert into C_ORDER values (201,'2001-08-03', 111,null)
insert into C_ORDER values (202,'2002-08-03', 111,null)
insert into C_ORDER values (203,'2001-08-04', 112,null)
insert into C_ORDER values (204,'2004-02-01', 113,null)
insert into C_ORDER values (205,'2001-04-02', 114,null)
insert into C_ORDER values (206,'2005-02-01', 115,null)
insert into C_ORDER values (207,'2008-04-01', 115,null)
insert into C_ORDER values (209,'2008-02-01', 114,null)
insert into C_ORDER values (208,'2008-12-01', 111,null)
insert into C_ORDER values (200,'2008-11-01', 111,null)
insert into C_ORDER values (210,'2008-10-01', 111,null)

update C_ORDER set ordamt = (select sum(O.qty * T.price) from
ORDER_ITEM O, ITEM T
```

```
                              where O.itemid = T.itemid and O.orderid
= 201)
where orderid = 201

select * from C_ORDER

select * from  ITEM

CREATE TABLE ITEM (
                itemid  int,
                price int,
                primary key (itemid)
             )

insert into ITEM values (301,2000)
insert into ITEM values (302,2000)
insert into ITEM values (303,1000)
insert into ITEM values (304,5000)
insert into ITEM values (305,4000)


CREATE TABLE ORDER_ITEM (
                orderid int,
                itemid int,
                qty int,
                primary key (orderid,itemid),
                foreign key(orderid) references C_ORDER(orderid) on
delete cascade on update cascade,
                foreign key(itemid) references ITEM(itemid) on delete
cascade on update cascade
                )

insert into ORDER_ITEM values (201,301,2)
insert into ORDER_ITEM values (201,302,4)
insert into ORDER_ITEM values (201,303,4)
insert into ORDER_ITEM values (201,304,4)
insert into ORDER_ITEM values (201,305,3)

insert into ORDER_ITEM values (202,303,2)
insert into ORDER_ITEM values (202,305,4)
insert into ORDER_ITEM values (203,302,1)
insert into ORDER_ITEM values (204,305,2)
insert into ORDER_ITEM values (205,301,3)
insert into ORDER_ITEM values (206,301,5)

select * from ORDER_ITEM

CREATE TABLE WAREHOUSE (
                warehouseid int,
                city varchar(20)not null,
                primary key (warehouseid)
              )
```

```
insert into WAREHOUSE values (1,'MAGALORE')
insert into WAREHOUSE values (2,'MAGALORE')
insert into WAREHOUSE values (3,'MAGALORE')
insert into WAREHOUSE values (4,'UDUPI')
insert into WAREHOUSE values (5,'UDUPI')
insert into WAREHOUSE values (6,'KARKALA')


select count(*)
from SHIPMENT s,WAREHOUSE w
where w.warehouseid=s.warehouseid and city='MAGALORE'

select count(distinct city)  from WAREHOUSE

select distinct city  from WAREHOUSE

CREATE TABLE SHIPMENT (
                orderid int,
                warehouseid int,
                ship_dt datetime,
                primary key (orderid,warehouseid)  ,
                foreign key(orderid) references C_ORDER(orderid) on
delete cascade on update cascade,
                foreign key(warehouseid) references
WAREHOUSE(warehouseid) on delete cascade on update cascade
              )

SELECT * FROM CUSTOMER

SELECT * FROM SHIPMENT

insert into SHIPMENT values (201,1,'2001-04-02')
insert into SHIPMENT values (201,2,'2001-04-04')
insert into SHIPMENT values (202,1,'2001-05-02')

insert into SHIPMENT values (202,2,'2002-05-12')
insert into SHIPMENT values (202,3,'2003-06-01')
insert into SHIPMENT values (202,4,'2003-06-01')
insert into SHIPMENT values (203,1,'2004-02-01')
insert into SHIPMENT values (203,2,'2004-02-01')
insert into SHIPMENT values (203,3,'2004-02-01')
insert into SHIPMENT values (204,4,'2004-06-02')
insert into SHIPMENT values (204,2,'2004-06-02')


SELECT * FROM WAREHOUSE
SELECT * FROM SHIPMENT
```

```
--
**********************************************************************
***************
1. Produce a listing: CUSTNAME, #oforders, AVG_ORDER_AMT, where the
middle column is the total
numbers of orders by the customer and the last column is the average
order amount for that customer.

select C.cname , count(O.orderid) as NO_OF_ORDR, avg(O.ordamt) as
AVG_ORD_AMT
from CUSTOMER C, C_ORDER O
where C.custid = O.custid group  by C.cname


2.List the order# for orders that were shipped from all the warehouses
that the company has in a specific city.

using not in
------------
select O.orderid from C_ORDER O
where not exists (select warehouseid from WAREHOUSE where city =
'MAGALORE' and warehouseid not in
                                      (select warehouseid from
SHIPMENT  where orderid = O.orderid)
                 )



select O.orderid from C_ORDER O
where not exists (
                        (select warehouseid from WAREHOUSE where city
= 'MAGALORE' and warehouseid not in
                                      (select warehouseid from
SHIPMENT  where orderid = O.orderid))
                                         union

                (select warehouseid from SHIPMENT  where orderid =
O.orderid and   warehouseid not in
                                      (select warehouseid from
WAREHOUSE where city = 'MAGALORE'))

                 )




select O.orderid from C_ORDER O
where not exists (select warehouseid from WAREHOUSE where city =
'MAGALORE' and warehouseid not in
                                      (select warehouseid from
SHIPMENT  where orderid = O.orderid)
```

```
                 )


using count
-----------

select A.orderid from shipment A,warehouse B
where A.warehouseid = B.warehouseid and B.city='MAGALORE' group by
A.orderid
having count(*) = (select count(*) from warehouse where
city='MAGALORE')


using left outer join
----------------------

select O.orderid from C_ORDER O
where not exists (select orderid  from (
                    (select warehouseid from WAREHOUSE where city =
'MAGALORE') as R1
                        left outer join
                     (select warehouseid, orderid
                             from SHIPMENT
                             where orderid = O.orderid) as R2 on
R1.warehouseid = R2.warehouseid)
                  where orderid is null
                 )
```

3. Demonstrate the deletion of an item from the ITEM table and
demonstrate a method of handling the rows in the
ORDER_ITEM table that contain this particular item.

1.   Produce a listing: CUSTNAME, #oforders,   AVG_ORDER_AMT, where
the middle column is the total
 numbers of orders by the customer and the last column is the average
order amount for that customer.

```
select C.cname,count(O.orderid),avg(O.ordamt)
from CUSTOMER C,C_ORDER O
where C.custid=O.custid
group by C.cname
```

2.   List the order# for orders that were shipped from all the
warehouses that the company has
in a specific city.

```
select distinct O.orderid
from C_ORDER O
where not exists( select warehouseid  from WAREHOUSE
                  where city='MAGALORE'
                  and warehouseid not in( select W.warehouseid
                                          from WAREHOUSE W,SHIPMENT S
                                          where
W.warehouseid=S.warehouseid
                                          and S.orderid=O.orderid))
```

3. Retrieve the details of customer whose average order amount for the year 2008 exceeds the
average order amount of the same customer for the year 2007.

```
select C.cname,C.custid,C.city,AVG(O.ordamt) as avg_amt into tb1
from CUSTOMER C,C_ORDER O
where C.custid=O.custid and YEAR(O.odate)='2001'
group by C.cname,C.custid,C.city

 select * from tb1

 select C.cname,C.custid,C.city,AVG(O.ordamt) as avg_amt into tb2
from CUSTOMER C,C_ORDER O
where C.custid=O.custid and YEAR(O.odate)='2002'
group by C.cname,C.custid,C.city

select * from tb2

select T.cname,T.custid,T.city,T.avg_amt
from tb1 T
group by T.cname,T.custid,T.city,T.avg_amt
having T.avg_amt > ( select avg_amt from tb2
                     where T.custid=custid)
```

4. Find the customer with maximum order amount for the year 2008

```
select C.cname,O.ordamt
from CUSTOMER C,C_ORDER O
where C.custid=O.custid and YEAR(O.odate)='2008'

group by C.cname,O.ordamt
having O.ordamt in ( select max(A.ordamt)
                     from C_ORDER A
                     where YEAR(A.odate)='2008')
```

5. Find the customer who has ordered least number of items.

```
select C.cname,I.qty
from CUSTOMER C,C_ORDER O,ORDER_ITEM I
where C.custid=O.custid and O.orderid=I.orderid
group by C.cname,I.qty
having I.qty in( select MIN(qty)
                 from ORDER_ITEM)
```

6. Find the item on which the company makes highest profit for the year 2008

```
select I.itemid,T.qty,I.price
from ITEM I,C_ORDER O,ORDER_ITEM T
where I.itemid=T.itemid and O.orderid=T.orderid and
YEAR(O.odate)='2008'
group by I.itemid,T.qty,I.price
having T.qty*I.price in ( select max(B.qty*A.price)
                          from  ITEM A,ORDER_ITEM B,C_ORDER C
                          where A.itemid=B.itemid and
C.orderid=B.orderid and YEAR(odate)='2008')
```

OR

```
 select I.itemid
from ITEM I,C_ORDER O,ORDER_ITEM T
where I.itemid=T.itemid and O.orderid=T.orderid and
YEAR(O.odate)='2001'
group by I.itemid
having sum(T.qty*I.price)>=ALL ( select sum(B.qty*A.price)
                          from  ITEM A,ORDER_ITEM B,C_ORDER C
                          where A.itemid=B.itemid and
C.orderid=B.orderid and YEAR(odate)='2001'
                          group by A.itemid)
```

7.   List the order# for orders that have been ordered for every item that the company produces.

```
select C.orderid
from C_ORDER C
where not exists( select itemid
                  from ITEM
                  where itemid not in( select itemid
                                       from ORDER_ITEM I
                                       where C.orderid=I.orderid))
```

8.   Find the year of maximum items sales.

```
select YEAR(O.odate)as max_sales_year
from C_ORDER O,ORDER_ITEM I
where O.orderid=I.orderid
group by YEAR(O.odate)
having sum(I.qty) >=ALL ( select sum(qty)
                           from C_ORDER O1,ORDER_ITEM I1
                           where O1.orderid=I1.orderid
                           group by YEAR(O1.odate))
```

9.   Find the city which ships maximum number of items

```
select W.city
from WAREHOUSE W,SHIPMENT S,ORDER_ITEM I
```

```
where W.warehouseid=S.warehouseid and S.orderid=I.orderid
group by W.city
having SUM(I.qty) >=ALL ( select SUM(C.qty)
                          from WAREHOUSE A,SHIPMENT B,ORDER_ITEM C
                          where A.warehouseid=B.warehouseid and
B.orderid=C.orderid
                          group by A.city )
```

10. List the order# for orders that were shipped from atmost two warehouses that the company has
in a specific city

```
select S.orderid
from SHIPMENT S,WAREHOUSE W
where S.warehouseid=W.warehouseid and W.city='UDUPI'
group by S.orderid
having count(W.warehouseid)<=2
```

a)List all the items that were ordered by each customer.(Details include custid,name,itemno)

```
select C.custid,C.cname,I.itemid
from C_ORDER O,CUSTOMER C,ORDER_ITEM I
where C.custid=O.custid and O.orderid=I.orderid
group by C.custid,C.cname,I.itemid
```

b) Give the details of the customer who has maximum orders

```
select C.cname,C.custid,C.city
from CUSTOMER C,C_ORDER O
where C.custid=O.custid
group by C.cname,C.custid,C.city
having COUNT(O.orderid) >=ALL ( select COUNT(orderid)
                                from CUSTOMER A,C_ORDER B
                                where A.custid=B.custid
                                group by A.custid)
```

c) Find the item which has maximum orders.

```
select I.itemid
from ORDER_ITEM O,ITEM I
where I.itemid=O.itemid
group by I.itemid
having COUNT(O.orderid) >=ALL ( select COUNT(O.orderid)
                                from ORDER_ITEM O,ITEM I
                                where I.itemid=O.itemid
                                group by I.itemid)
```

d)Find the item which has maximum sales.

```
select I.itemid
from ORDER_ITEM O,ITEM I
```

```
where I.itemid=O.itemid
group by I.itemid
having sum(O.qty) >=ALL ( select SUM(A.qty)
                          from ORDER_ITEM A,ITEM B
                          where A.itemid=B.itemid
                          group by B.itemid)
```

e) Give the details of warehouses from which items were shipped(include ware house city).

```
select distinct W.warehouseid,W.city
from WAREHOUSE W,SHIPMENT S
where W.warehouseid=S.warehouseid
```

f)Give the details of total amount earned for each item .(itemno, total amount earned)

```
select I.itemid,SUM(I.price*O.qty)
from ITEM I,ORDER_ITEM O
where I.itemid=O.itemid
group by I.itemid
```

g) List any customer whose   all   ordered items are shipped from a specific warehouse.

```
select C.cname
from CUSTOMER C
where not exists( select O.orderid
                  from C_ORDER O
                  where O.custid=C.custid and O.orderid not in(select
S.orderid
                                                               from
WAREHOUSE W,SHIPMENT S
                                                               where
W.warehouseid=S.warehouseid
                                                               and
W.warehouseid=2))
```

4.   Find the total price of the items that were shipped between 2005 and 2008

```
select SUM(I.price*O.qty) as total_amount
from ITEM I,ORDER_ITEM O,SHIPMENT S
where I.itemid=O.itemid and S.orderid=O.orderid and S.ship_dt between
'2001-01-01' and '2003-12-31'
```

2.Find the customer with minimum number of orders but with maximum order amount

```
select C.cname into tb1
from CUSTOMER C,C_ORDER O
where C.custid=O.custid
```

```
group by C.cname
having COUNT(O.orderid)<= ALL( select COUNT(O1.orderid)
                               from CUSTOMER C1,C_ORDER O1
                               where C1.custid=O1.custid
                               group by C1.cname)

select * from tb1

select T.cname
from tb1 T,CUSTOMER C,C_ORDER O
where T.cname=C.cname and C.custid=O.orderid
group by T.cname
having SUM(O.ordamt)>=ALL(select SUM(O1.ordamt)
                          from CUSTOMER C1,C_ORDER O1,tb1 T1
                          where T1.cname=C1.cname and
C1.custid=O1.orderid
                          group by C1.cname)
```

```
create database st_enroll

use st_enroll

create table STUDENT (
                regno varchar(10),
                fname char(15),
                major char (20),
                bdate datetime
                primary key(regno)
               )

insert into STUDENT values ('111','ravi','academic','1989-11-09')
insert into STUDENT values ('112','sudha','academic','1979-07-04')
insert into STUDENT values ('113','kumar','academic','1979-01-06')
insert into STUDENT values ('114','raju','academic','1999-10-02')
insert into STUDENT values ('115','hemanth','academic','1988-11-04')

create table COURSE (
                course int,
                cname varchar(15),
                dept  char (20),
                primary key(course)
               )


insert into COURSE values (1,'DBMS','CS')
insert into COURSE values (2,'COMPILER','CS')
insert into COURSE values (3,'JAVA','CS')
insert into COURSE values (4,'SIG PROCESSING','ENC')
insert into COURSE values (5,'DIGTAL CIRCUITS','ENC')
insert into COURSE values (6,'MACHINE DESIGN','MECH')
insert into COURSE values (7,'THEMODYNAICS','MECH')
insert into COURSE values (8,'AUTOCAD','MECH')

select * from COURSE

create table TEXTBOOK (
                bookISBN int,
                title varchar(50),
                publisher  varchar(20),
                author  char(20),
                primary key (bookISBN)
                )

drop table TEXTBOOK

insert into TEXTBOOK  values (201,'Fundamentals of
DBMS','McGraw','NAVATHE')
insert into TEXTBOOK  values (202,'Database Design','McGraw','Raghu
Rama')
```

```
insert into TEXTBOOK  values (203,'Compiler design','Pearson','Ulman')
insert into TEXTBOOK  values (204,'JAVA complete
Reference','McGraw','BALAGURU')
insert into TEXTBOOK  values (205,'Singals and
Fundumentals','McGraw','NITHIN')
insert into TEXTBOOK  values (206,'Machine Theory','McGraw','Ragavan')
insert into TEXTBOOK  values (208,'Circuit
design','McGraw','Rajkamal')
insert into TEXTBOOK  values (207,'Thermodynamics','McGraw','Alfred')
insert into TEXTBOOK  values (209,'Electronic
Circuits','McGraw','Alfred')
insert into TEXTBOOK  values (210,'Circuits Theory','McGraw','Alfred')

select * from TEXTBOOK

create table BOOK_ADAPTION (
                course int,
                sem int,
                bookISBN int,
                primary key(course, sem,bookISBN),
                foreign key(course) references COURSE(course) on
delete cascade on update cascade,
                foreign key(bookISBN) references TEXTBOOK (bookISBN)
on delete cascade on update cascade,
               )

insert into BOOK_ADAPTION  values (1,5,201)
insert into BOOK_ADAPTION  values (1,7,202)
insert into BOOK_ADAPTION  values (2,5,203)
insert into BOOK_ADAPTION  values (2,6,203)
insert into BOOK_ADAPTION  values (3,7,204)
insert into BOOK_ADAPTION  values (4,3,205)
insert into BOOK_ADAPTION  values (4,5,209)
insert into BOOK_ADAPTION  values (5,5,205)
insert into BOOK_ADAPTION  values (5,6,208)
insert into BOOK_ADAPTION  values (5,2,210)
insert into BOOK_ADAPTION  values (6,7,206)
insert into BOOK_ADAPTION  values (7,3,207)
insert into BOOK_ADAPTION  values (7,3,206)
insert into BOOK_ADAPTION  values (8,3,207)

delete from BOOK_ADAPTION

select * from BOOK_ADAPTION


create table ENROLL (
                regno varchar(10),
                course  int,
                sem int ,
                marks int,
                primary key(regno,course,sem),
                foreign key(regno) references STUDENT(regno)on delete
cascade on update cascade,
                foreign key(course) references COURSE(course)on delete
cascade on update cascade,
               )


drop table ENROLL

drop table BOOK_ADAPTION



insert into ENROLL  values (111,1,5,59)
insert into ENROLL  values (111,2,5,70)
insert into ENROLL  values (111,3,5,75)
insert into ENROLL  values (112,1,5,49)
insert into ENROLL  values (113,2,5,80)
insert into ENROLL  values (114,3,7,79)
insert into ENROLL  values (115,4,3,79)

select * from ENROLL
```

1. Produce a list of text books (include Course #, Book-ISBN, Book-title) in the alphabetical order for courses offered by the ‘CS’ department that use more than two books.

```
select A.bookISBN,A.title,B.course,B.cname  from TEXTBOOK A,COURSE
B,BOOK_ADAPTION C
where  A.bookISBN = C.bookISBN and B.course=C.course
and B.dept='CS' and B.course in (select course from BOOK_ADAPTION
group by course having count(*)>=2)
order by A.title
```

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

```
select distinct(C.dept) from course C
   where not exists (


                       select bookISBN from  BOOK_ADAPTION
                       where  course in
                          (select course from  course where dept =
C.dept) and bookISBN not in


(select bookISBN from TEXTBOOK where publisher='McGraw')
```

```
             )


OR

select distinct(C1.dept) from course C1
   where not exists (


                       select B.bookISBN from  BOOK_ADAPTION B ,
COURSE C
                       where  B.course = C.course
                       and C.dept = C1.dept   and bookISBN not in

                                             (select bookISBN from
TEXTBOOK where publisher='McGraw')
             )




using count (*)(will not work for this)
-----------


select distinct(C1.dept) from COURSE C1
where  exists (

                   select  count (distinct BA.bookISBN) from COURSE C,
BOOK_ADAPTION BA
              where C.course = BA.course and C.dept = C1.dept
              having  count (distinct BA.bookISBN) =   (select count
(distinct BA.bookISBN)  from  COURSE C,TEXTBOOK T, BOOK_ADAPTION BA
                                                where C.course =
BA.course and T.bookISBN = BA.bookISBN  and  T.publisher='McGraw'
                                           and C.dept = C1.dept)
                )







select * from TEXTBOOK
select * from COURSE
select * from BOOK_ADAPTION
select * from COURSE
select * from enroll


select C.dept , C.course from course C, enroll E
```

```
where C.course = E.course group by  C.dept  , C.course

select C.dept,  count(distinct E.regno) from course C, enroll E
where C.course = E.course group by  C.dept
having count (distinct E.regno) >= all(select   count(distinct
E.regno) from course C, enroll E where C.course = E.course group by
C.dept )


select C.dept, C.course ,count(distinct B.bookISBN) from course C,
BOOK_ADAPTION B
where C.course = B.course group by  C.dept ,C.course

select C.dept,count(distinct B.bookISBN) from course C, BOOK_ADAPTION
B
where C.course = B.course group by  C.dept


select C.dept from course C, BOOK_ADAPTION B
where C.course = B.course group by  C.dept

select C.dept, count(distinct E.regno) , count(distinct B.bookISBN)
from  COURSE C, BOOK_ADAPTION B, ENROLL E
where C.course = E.course and B.course = C.course
and C.dept in
           (select C.dept from course C
           group by  C.dept
           having count(*) > 2)
group by C.dept


create view temp as

1. for each dept list course that adopts maximum number of books

select C.dept, C.course,count(distinct B.bookISBN)  from course C,
BOOK_ADAPTION B
where C.course = B.course group by  C.dept ,C.course
having count(distinct B.bookISBN) > = all
                 (select count(distinct B1.bookISBN)  from course C1,
BOOK_ADAPTION B1
                           where C1.course = B1.course and C1.dept =
C.dept group by  C1.dept ,C1.course)


select * from temp
drop view temp


select T.dept,T.course from temp T where
T.no_of_books in ( select max(no_of_books) from temp where dept =
T.dept)
```

1. Produce a list of text books (include Course #, Book-ISBN, Book-title) in the alphabetical order for courses offered by the ‘CS’ department that use more than two books.(*)

```
select C.course,T.bookISBN,T.title
from COURSE C,BOOK_ADAPTION B,TEXTBOOK T
where C.course=B.course and B.bookISBN=T.bookISBN and C.dept='CS'
and C.course in( select course
                 from BOOK_ADAPTION
                 where course=C.course
                 group by course
                 having COUNT(distinct bookISBN)>=2)
order by T.title
```

2. List any department that has all its adopted books published by a specific publisher. (*)

```
select distinct C.dept
from COURSE C
where not exists( select bookISBN
                  from BOOK_ADAPTION
                  where course in( select COURSE
                                   from COURSE
                                   where dept=C.dept) and bookISBN not
in( select T.bookISBN

from TEXTBOOK T

where T.publisher='Mcgraw'))
```

3. Find the department that has maximum number of adopted books.

```
select C.dept
from COURSE C,BOOK_ADAPTION B,TEXTBOOK A
where C.course=B.course and B.bookISBN=A.bookISBN
group by C.dept
having COUNT(B.bookISBN) >=ALL (select COUNT(distinct D.bookISBN)
                             from COURSE F,BOOK_ADAPTION D,TEXTBOOK E
                             where F.course=D.course and
D.bookISBN=E.bookISBN
                             group by F.dept)
```

4. List any course# offered by “CS “ department that adapts book titled “RDBMS”

```
select C.course
from COURSE C,BOOK_ADAPTION B,TEXTBOOK A
where C.course=B.course and B.bookISBN=A.bookISBN and C.dept='CS'
and A.title='Fundamentals of DBMS'
```

5.   List the publishers and authors of the books adopted for a specific course offered by
“CS” department.

```
select distinct A.publisher,A.author
from COURSE C,BOOK_ADAPTION B,TEXTBOOK A
where C.course=B.course and B.bookISBN=A.bookISBN and C.dept='CS' and
C.cname='JAVA'
```

6.   List the details of students (regno, name, major) who have enrolled in all courses offered by
“CS” department

```
select S.regno,S.fname,S.major
from STUDENT S
where not exists(select C.course
                 from COURSE C
                 where C.dept='CS' and C.course not in(select course
                                                       from ENROLL
                                                       where
regno=S.regno))
```

7.   List the details (regno, name, major) in the alphabetical order for students who have enrolled
in courses offered by “CS” department that use more than 3 books.

```
select s.regno,S.fname,S.major
from STUDENT S
where not exists ( select distinct C.course
                   from COURSE C,BOOK_ADAPTION B,TEXTBOOK A
                   where C.course=B.course and B.bookISBN=A.bookISBN
and C.dept='CS'
                   and C.course in (select course
                                    from BOOK_ADAPTION
                                    group by course
                                    having COUNT(distinct
bookISBN)>=2)and C.course not in( select D.course

from ENROLL D

where S.regno=D.regno))
order by S.fname
```

8.   Find the department that has maximum number of students.

```
select C.dept
from COURSE C,ENROLL E
where C.course=E.course
group by C.dept
having COUNT(distinct E.regno)>=ALL ( select COUNT(distinct F.regno)
                                      from ENROLL F,COURSE D
```

```
                                       where F.course=D.course
                                       group by D.dept)
```

9.   Produce a list of text books (include book_ISBN, book-title, author) in the alphabetical order for the course offered by the ‘CS’ department that has 10 enrolled students.

```
select T.bookISBN,T.title,T.author
from TEXTBOOK T,BOOK_ADAPTION B,COURSE C,ENROLL E
where T.bookISBN=B.bookISBN and C.course=E.course and
B.course=E.course and C.dept='CS'
group by C.course,T.bookISBN,T.title,T.author
having COUNT(distinct E.regno)=2
order by T.title
```

10. Produce the details: COURSENO, COURSE_NAME, DEPT, #NO_OF_BOOKS in the alphabetical order for courses that use atleast 3 books published by a specific publisher.

```
select C.course,C.cname,C.dept,COUNT(B.bookISBN)as No_of_Books
from COURSE C,BOOK_ADAPTION B,TEXTBOOK T
where T.bookISBN=B.bookISBN and C.course=B.course and
T.publisher='Mcgraw'
group by C.course,C.cname,C.dept
having COUNT(distinct B.bookISBN)>=3
order by C.cname
```

11. For each department that offers more than 2 courses, list the dept name, total number of students enrolled in those courses and total number of books adapted by those courses.

```
select C.dept,C.course,COUNT(distinct E.regno)as
No_of_students,COUNT(distinct B.bookISBN) as No_of_books
from COURSE C,BOOK_ADAPTION B,ENROLL E
where C.course=B.course and C.course=E.course
group by C.dept,C.course
having COUNT(C.course)>=2
```

a) Give the   details of students who have enrolled in the courses conducted by CS department(include regno,name,coursename).

```
select S.regno,S.fname,C.course
from STUDENT S,COURSE C,ENROLL E
where S.regno=E.regno and C.course=E.course and C.dept='CS'
group by C.course,S.regno,S.fname
```

b) Give the details of books adopted for each courses.
(Details include bookisbn, title, publisher, author)

```
select distinct C.course,T.bookISBN,T.title,T.publisher,T.author
from TEXTBOOK T,BOOK_ADAPTION B,COURSE C
where T.bookISBN=B.bookISBN and C.course=B.course
group by C.course,T.bookISBN,T.title,T.publisher,T.author
```

c)Find the no. of books adopted for each course conducted by each dept.
(Details include courseno, name, dept ,no of books)

```
select C.dept,C.course,count(distinct B.bookISBN)as No_of_books
from BOOK_ADAPTION B,COURSE C
where C.course=B.course
group by C.dept,C.course
```

d)List the books(if any) which are adopted by more than one dept.

```
select B.bookISBN
from BOOK_ADAPTION B,COURSE C
where C.course=B.course
group by B.bookISBN
having COUNT(distinct C.dept)>1
```

e)List the dept which has maximum no. of adopted books .

```
select C.dept
from COURSE C,BOOK_ADAPTION B
where C.course=B.course
group by C.dept
having COUNT(distinct B.bookISBN)>=ALL( select COUNT(distinct
D.bookISBN)
                                        from COURSE A,BOOK_ADAPTION D
                                        where A.course=D.course
                                        group by A.dept)
```

e1)List the dept which has minimum no. of adopted books .

```
select C.dept
from COURSE C,BOOK_ADAPTION B
where C.course=B.course
group by C.dept
having COUNT(distinct B.bookISBN)<=ALL( select COUNT(distinct
D.bookISBN)
                                        from COURSE A,BOOK_ADAPTION D
                                        where A.course=D.course
                                        group by A.dept)
```

f)List the no. of students in each dept.

```
select COUNT(distinct S.regno)as No_of_students
from STUDENT S,COURSE C,ENROLL E
where S.regno=E.regno and C.course=E.course
group by C.dept
```

g)Find the dept having maximum no. of students.

```
select C.dept
from STUDENT S,COURSE C,ENROLL E
```

```
where S.regno=E.regno and C.course=E.course
group by C.dept
having COUNT(distinct S.regno) >=ALL( select COUNT(distinct S1.regno)
                                      from STUDENT S1,COURSE B,ENROLL
E1
                                      where S1.regno=E1.regno and
B.course=E1.course
                                      group by B.dept)
```

h) Find the dept having maximum no. of courses.

```
select C.dept
from COURSE C
group by C.dept
having COUNT(distinct C.course) >=ALL(select COUNT(distinct course)
                                   from COURSE
                                   group by dept)
```

i)Give the details of the student who has taken maximum no. of courses

```
select S.regno,S.fname,S.major
from STUDENT S,ENROLL E
where S.regno=E.regno
group by S.regno,S.fname,S.major
having count(distinct E.course) >= ALL( select count(distinct
E1.course)
                                        from STUDENT S1,ENROLL E1
                                        where S1.regno=E1.regno
                                        group by
S1.regno,S1.fname,S1.major)
```

j) Give the details of the student who has obtained maximum marks

```
select S.regno,S.fname,S.major
from STUDENT S,ENROLL E
where S.regno=E.regno
group by S.regno,S.fname,S.major
having AVG(E.marks)>=ALL( select AVG(E1.marks)
                          from ENROLL E1,STUDENT S1
                          where S1.regno=E1.regno
                          group by S1.regno,S1.fname,S1.major)
```

List the departments that adopt atleast one book published by a specific publisher for
every course it offers.

```
select distinct C.dept
from COURSE C
where not exists( select C1.course
                  from COURSE C1
                  where C1.dept=C.dept and C1.course not in( select
distinct C2.course
                                                              from
BOOK_ADAPTION B,COURSE C2
                                                              where
C2.course=B.course and C2.dept=C.dept
                                                              and
C2.course in ( select B1.course from BOOK_ADAPTION B1,TEXTBOOK T

where B1.bookISBN=T.bookISBN and T.publisher='McGraw'

group by B1.course

having count(distinct B1.bookISBN)>=1)))
```

list the course details of the department having maximum number of students.

```
select C.dept into tb1
from COURSE C,ENROLL E
where C.course=E.course
group by C.dept
having COUNT(E.regno)>=all(select COUNT(E1.regno)
                           from ENROLL E1,COURSE C1
                           where C1.course=E1.course
                           group by C1.dept)

 select * from tb1

 select C.cname,C.course
 from COURSE C,tb1 T
 where C.dept=T.dept
```

```
create database bk_shop

use bk_shop

create table AUTHOR
           (
                authorid int primary key,
                aname  varchar(20),
                city varchar(20),
                country    varchar(20)
           )


insert into AUTHOR values(110,'Elmasri','Houston','Canada')
insert into AUTHOR values(111,'sebesta','mangalore','India')
insert into AUTHOR values(112,'Elmasri','Houston','Canada')
insert into AUTHOR values(113,'Bharath K','Bangalore','India')
insert into AUTHOR values(114,'Willy Z','California','USA')
insert into AUTHOR values(115,'Salma','Dakha','Bangladesh')



create table PUBLISHER
           (
                pubid int primary key,
                pname  varchar(20),
                city varchar(20),
                country    varchar(20)
           )


insert into PUBLISHER values(201,'McGRAW','mangalore','India')
insert into PUBLISHER values(202,'Pearson','Bangalore','India')
insert into PUBLISHER values(203,'GKP','Bangalore','India')
insert into PUBLISHER values(204,'MediTech','Delhi','India')
insert into PUBLISHER values(205,'Sun','Ahmadbad','India')



create table CATEGORY
           (
                catid int primary key ,
                descript varchar(30),
           )


insert into CATEGORY values(1,'All children Books')
insert into CATEGORY values(2,'Cooking Books')
insert into CATEGORY values(3,'Popular Novels')
insert into CATEGORY values(4,'Small Story Books')
insert into CATEGORY values(5,'Medical Books')
```

```
create table CATALOGUE
           (
                bookid int primary key,
                title  varchar(20),
                pubid int,
                authorid int,
                catid int,
                yr int,
                price int,
                foreign key(pubid) references PUBLISHER(pubid) on
delete cascade on update cascade,
                foreign key(authorid) references AUTHOR(authorid) on
delete cascade on update cascade,
                foreign key(catid) references CATEGORY(catid) on
delete cascade on update cascade
           )


select * from PUBLISHER

insert into CATALOGUE values(301,'Panchatantra',201,111,1,2000,300)
insert into CATALOGUE values(302,'Vegetables',202,111,2,2000,400)
insert into CATALOGUE values(303,'Yogasana',203,112,5,2002,600)
insert into CATALOGUE values(304,'Stories of
Village',204,113,4,2005,100)
insert into CATALOGUE values(305,'Triangle',205,114,3,2008,1000)
insert into CATALOGUE values(306,'Naughtiest
Girl',201,110,3,2007,1500)
insert into CATALOGUE values(307,'Cookery',205,115,2,2006,100)


select * from CATALOGUE

create table ORDER_DET
           (
                ordno int ,
                bookid int,
                qty int,
                primary key (ordno,bookid),
                foreign key(bookid) references CATALOGUE(bookid) on
delete cascade on update cascade,
           )


insert into ORDER_DET values(1,301,10)
insert into ORDER_DET values(1,302,6)
insert into ORDER_DET values(1,307,23)

insert into ORDER_DET values(2,301,15)
insert into ORDER_DET values(2,304,11)

insert into ORDER_DET values(3,304,15)

insert into ORDER_DET values(4,301,3)
insert into ORDER_DET values(4,305,8)

insert into ORDER_DET values(5,303,20)
insert into ORDER_DET values(5,306,6)
insert into ORDER_DET values(5,305,7)

select * from ORDER_DET
```

1. Give the details of the authors who have 2 or more books in the catalog and the price of the books is greater than the average price of the books in the catalog and the year of publication is after 2000.

```
select A.authorid,A.aname,A.city from AUTHOR A, CATALOGUE C
where A.authorid  =  C.authorid group by A.authorid, A.aname,A.city
having sum(C.price) > (select avg(price) from CATALOGUE)
and count(*)>=2
```

2. Find the author of the book which has maximum sales.

```
select A.authorid ,A.aname  ,A.city ,C.bookid,sum(O.qty) as QTY_SUM
into tb_auth  from AUTHOR A, CATALOGUE C,ORDER_DET O
where A.authorid  =  C.authorid
and C.bookid = O.bookid  group by A.authorid, A.aname,A.city,C.bookid

select * from  tb_auth   where QTY_SUM in (select max(QTY_SUM) from
tb_auth)


select A.authorid ,A.aname  ,A.city ,sum(O.qty) as QTY_SUM   from
author A, catalog C,order_det O
where A.authorid  =  C.authorid
and C.bookid = O.bookid  group by A.authorid, A.aname,A.city,C.bookid
having sum(qty) >= all (select sum(qty)  from order_det group by
bookid)

(select A.authorid ,A.aname  ,A.city ,C.bookid,sum(O.qty)  from author
A, catalog C,order_det O
where A.authorid  =  C.authorid
and C.bookid = O.bookid)   group by A.authorid,
A.aname,A.city,C.bookid
having sum(qty) = (select max(qty)  from temp1 )

create view temp as
select A.authorid ,A.aname  ,A.city ,C.bookid,sum(O.qty) as QTY_SUM
from author A, catalog C,order_det O
```

```
where A.authorid  =  C.authorid
and C.bookid = O.bookid  group by A.authorid, A.aname,A.city,C.bookid

select * from  temp   where QTY_SUM = (select max(QTY_SUM) from temp)


select * from  tb_auth   where QTY_SUM in (select max(QTY_SUM) from
tb_auth)
```

3. Increase the price of the books published by a specific publisher by 10%

```
update  catalog set price = price * 1.1 where pubid in ( select pubid
from publisher where pname ='Pearson')


select count(*) as no_of_orders from order_det
where bookid in (

     select bookid  from order_det  group by bookid
     having sum(qty) >= all (select sum(qty)  from order_det group by
bookid)
    )
group by bookid
```

1. Give the details of the authors who have 2 or more books in the catalog and the price of the books is greater than the average price of the books in the catalog and the year of publication is after 2000.(*)

```
select A.authorid,A.aname,A.city,A.country
from AUTHOR A,CATALOGUE C
where C.authorid=A.authorid and C.yr=2000
group by A.authorid,A.aname,A.city,A.country
having count(distinct C.bookid)>=2 and sum(C.price) >(select
AVG(price) from CATALOGUE)
```

2. Find the author of the book which has maximum sales.(*)

```
select A.authorid,A.aname
from AUTHOR A,CATALOGUE C,ORDER_DET O
where A.authorid=C.authorid and C.bookid=O.bookid
group by A.authorid,A.aname,C.bookid
having SUM(O.qty)>=ALL( select SUM(O1.qty)
                        from AUTHOR A1,CATALOGUE C1,ORDER_DET O1
                        where A1.authorid=C1.authorid and
C1.bookid=O1.bookid
                        group by A1.authorid,A1.aname,C1.bookid)
```

3.   List the order-no# for orders that were ordered for every book of a specific author.

```
select distinct O.ordno
from ORDER_DET O
where not exists( select C.bookid
                  from CATALOGUE C,AUTHOR A
                  where A.authorid=C.authorid and A.aname='Elmasri'
                  and C.bookid not in( select O1.bookid
                                       from ORDER_DET O1
                                       where O.ordno=O1.ordno))
```

4.   List the order-no# for orders that were ordered for every book published by a specific
publisher.

```
select distinct O.ordno
from ORDER_DET O
where not exists( select C.bookid
                  from CATALOGUE C,PUBLISHER P
                  where C.pubid=P.pubid and P.pname='Pearson'
                  and C.bookid not in( select O1.bookid
                                       from ORDER_DET O1
                                       where O.ordno=O1.ordno))
```

5.   List the order-no# for orders that were ordered for every book of a specific category.

```
select distinct O.ordno
from ORDER_DET O
where not exists( select C.bookid
                  from CATALOGUE C,CATEGORY E
                  where C.catid=E.catid and E.catid=1
                  and C.bookid not in( select O1.bookid
                                       from ORDER_DET O1
                                       where O1.ordno=O.ordno))
```

6.   List names of authors who have written atleast one book in every category.

```
select A.aname
from AUTHOR A
where not exists( select distinct catid
                  from CATEGORY
                  where catid not in( select distinct C.catid
                                        from CATALOGUE C
                                        where C.authorid=A.authorid))
```

7.   List names of authors who have written atleast two books in every category.

```
select A.aname
```

```
from AUTHOR A
where not exists( select distinct catid
                  from CATEGORY
                  where catid not in( select C.catid
                                      from CATALOGUE C
                                      where C.authorid=A.authorid
                                      group by C.catid
                                      having count(C.bookid)>=2))
```

8.	List the order-no# for orders that were ordered for every book published by a specific
publisher and written by a specific author.

```
select distinct O.ordno
from ORDER_DET O
where not exists( select C.bookid
                  from CATALOGUE C,AUTHOR A,PUBLISHER P
                  where C.authorid=A.authorid and C.pubid=P.pubid
                  and A.aname='sebesta' and P.pname='McGRAW'
                  and C.bookid not in( select bookid
                                       from ORDER_DET
                                       where O.ordno= ordno))
```

9.	Find the category of the book which has maximum sales.

```
select C.catid
from CATEGORY C,ORDER_DET O,CATALOGUE E
where C.catid=E.catid and O.bookid=E.bookid
group by C.catid
having SUM(O.qty) >=ALL( select SUM(O1.qty)
                         from CATEGORY C1,ORDER_DET O1,CATALOGUE E1
                         where C1.catid=E1.catid and
O1.bookid=E1.bookid
                         group by C1.catid)
```

10.	Find the publisher of the book which has maximum sales.

```
select P.pname,P.pubid
from PUBLISHER P,CATALOGUE C,ORDER_DET O
where C.pubid=P.pubid and O.bookid=C.bookid
group by P.pubid,P.pname
having count(O.qty)>=all( select count(O1.qty)
                          from PUBLISHER P1,CATALOGUE C1,ORDER_DET O1
                          where C1.pubid=P1.pubid and
O1.bookid=C1.bookid
                          group by P1.pubid,P1.pname)
```

11.	Find the price of the book which has maximum sales.

```
select C.price
from CATALOGUE C,ORDER_DET O
where C.bookid=O.bookid
```

```
group by C.bookid,C.price
having sum(O.qty)>=ALL( select SUM(O1.qty)
                        from CATALOGUE C1,ORDER_DET O1
                        where C1.bookid=O1.bookid
                        group by C1.bookid)
```

13.   Find the average amount earned from the book which has maximum sales.

```
select avg(C.price*O.qty) as average_amount
from CATALOGUE C,ORDER_DET O
where C.bookid=O.bookid
group by C.bookid
having sum(O.qty)>=ALL( select SUM(O1.qty)
                        from CATALOGUE C1,ORDER_DET O1
                        where C1.bookid=O1.bookid
                        group by C1.bookid)
```

14.   Find the number of books that were sold for the book which has maximum sales.

```
select SUM(O.qty) as max_books
from CATALOGUE C,ORDER_DET O
where C.bookid=O.bookid
group by C.bookid
having sum(O.qty)>=ALL( select SUM(O1.qty)
                        from CATALOGUE C1,ORDER_DET O1
                        where C1.bookid=O1.bookid
                        group by C1.bookid)
```

15.   Find the publication year of the book which has maximum sales.

```
select C.yr
from CATALOGUE C,ORDER_DET O
where C.bookid=O.bookid
group by C.bookid,C.yr
having SUM(O.qty) >=ALL( select SUM(O1.qty)
                         from CATALOGUE C1,ORDER_DET O1
                         where C1.bookid=O1.bookid
                         group by C1.bookid)
```

16.   List CATEGORY, #BOOKID, #OFBOOKS, #OFPRICE where   #OFBOOKS is the total number of books ordered and #OFPRICE is the total amount earned by selling that book

```
select C.catid,C1.bookid,SUM(O.qty) as total_no_of_books,
SUM(O.qty*C1.price) as Total_amt
from CATEGORY C,CATALOGUE C1,ORDER_DET O
where C.catid=C1.catid and C1.bookid=O.bookid
group by C.catid,C1.bookid
```

17. List the details of publishers (include name, city, country) for publishers who have published at least 2 books in every category

```
select P.pname,P.city,P.country
from PUBLISHER P
where not exists( select catid
                  from CATEGORY
                  where catid not in (select C.catid
                                      from CATALOGUE C,CATEGORY C1
                                      where C.catid=C1.catid
                                      and C.pubid=P.pubid
                                      group by C.catid,C.pubid
                                      having count(C.bookid)>=1))
```

a)Give   the details of available books in   each   category.

```
select C.catid,C.bookid,A.aname,P.pname,C.title
from AUTHOR A,CATALOGUE C,PUBLISHER P,CATEGORY B
where C.catid=B.catid and C.authorid=A.authorid and C.pubid=P.pubid
group by C.catid,C.bookid,A.aname,P.pname,C.title
```

b)Give the details of total quantity   for each   book.
(Details include orderno , bookid, title, authored ,author name, total qnty).

```
select O.ordno,C.bookid,C.authorid,A.aname,O.qty
from ORDER_DET O,AUTHOR A,CATALOGUE C
where O.bookid=C.bookid and A.authorid=C.authorid
group by O.ordno,C.bookid,C.authorid,A.aname,O.qty
```

c) Give the details the book having maximum orders

```
select C.bookid,A.aname,P.pname,C.title
from AUTHOR A,CATALOGUE C,PUBLISHER P,ORDER_DET O
where P.pubid=C.pubid and A.authorid=C.authorid and O.bookid=C.bookid
group by C.bookid,A.aname,P.pname,C.title
having count(O.ordno)>=ALL ( select count(O1.ordno)
                           from AUTHOR A1,CATALOGUE C1,PUBLISHER
P1,ORDER_DET O1
                           where P1.pubid=C1.pubid and
A1.authorid=C1.authorid
                           and O1.bookid=C1.bookid
                           group by
C1.bookid,A1.aname,P1.pname,C1.title)
```

d)Find the category of the book which has maximum sales

```
select C.catid,C.descript
from CATEGORY C,CATALOGUE C1,ORDER_DET O
where C.catid=C1.catid and O.bookid=C1.bookid
group by C1.bookid,C.catid,C.descript
having SUM(O.qty)>=ALL( select SUM(O1.qty)
                        from CATEGORY C3,CATALOGUE C2,ORDER_DET O1
                        where C3.catid=C2.catid and
O1.bookid=C2.bookid
                        group by C2.bookid)
```

e)Find the category/author of the book which has minimum orders.

```
select C.descript
from CATEGORY C,CATALOGUE B,ORDER_DET O
where C.catid=B.catid and O.bookid=B.bookid
group by B.bookid,C.descript
having count(O.ordno)<=ALL( select count(O1.ordno)
                        from CATEGORY C1,CATALOGUE B1,ORDER_DET O1
                        where C1.catid=B1.catid and
O1.bookid=B1.bookid
                        group by B1.bookid)
```

f)What is the total amount earned by the dealer from the book having maximum sales.

```
select SUM(O.qty*C.price) as total_profit
from CATALOGUE C,ORDER_DET O
where C.bookid=O.bookid
group by C.bookid
having SUM(O.qty)>= ALL( select SUM(O1.qty)
                         from CATALOGUE C1,ORDER_DET O1
                         where C1.bookid=O1.bookid
                         group by C1.bookid)
```

h) Find the category(if any) having all its books ordered.

```
select C.catid
from CATEGORY C
where not exists( select B.bookid
                  from CATALOGUE B
                  where B.catid=C.catid and B.bookid not in( select
distinct bookid
                                                             from
ORDER_DET))
```

```
create database  bank

use bank

create table BRANCH(
                bname varchar(15)primary key,
                bcity varchar(15),
                assets real
              )

insert into BRANCH values('synd_nitte','karkala',200000)
insert into BRANCH values('Corp_nitte','karkala',300000)
insert into BRANCH values('PNB_nitte','karkala',100000)
insert into BRANCH values('Corp_mang','Mangalore',300000)
insert into BRANCH values('PNB_mang','Mangalore',500000)
insert into BRANCH values('state_udupi','Udupi',500000)
insert into BRANCH values('synd_udupi','Udupi',500000)

select * from BRANCH

create table ACCOUNT(
                accno int,
                bname varchar(15),
                balance real,
                primary key(accno),
                foreign key(bname) references BRANCH(bname) on delete
cascade on update cascade
               )


insert into ACCOUNT values(12345,'synd_nitte',6000)
insert into ACCOUNT values(12340,'synd_nitte',6000)
insert into ACCOUNT values(21345,'synd_nitte',10000)

insert into ACCOUNT values(14341,'Corp_nitte',15000)
insert into ACCOUNT values(14345,'Corp_nitte',15000)
insert into ACCOUNT values(12455,'Corp_nitte',17000)

insert into ACCOUNT values(13345,'PNB_nitte',11000)
insert into ACCOUNT values(13346,'PNB_nitte',11000)
insert into ACCOUNT values(13347,'PNB_nitte',11000)
insert into ACCOUNT values(13340,'PNB_nitte',11000)

insert into ACCOUNT values(15345,'synd_udupi',11000)

insert into ACCOUNT values(12453,'PNB_mang',17000)
insert into ACCOUNT values(21346,'PNB_mang',10000)
insert into ACCOUNT values(12450,'PNB_mang',17000)
insert into ACCOUNT values(12452,'PNB_mang',17000)

insert into ACCOUNT values(13245,'state_udupi',5000)
insert into ACCOUNT values(13241,'state_udupi',5000)
insert into ACCOUNT values(12375,'state_udupi',12000)
insert into ACCOUNT values(12377,'state_udupi',12000)
insert into ACCOUNT values(12378,'state_udupi',12000)
insert into ACCOUNT values(15342,'state_udupi',19000)
insert into ACCOUNT values(12451,'state_udupi',17000)


select * from depositor D , account A where D.accno = A.accno and
D.cname = 'rakesh'

select * from ACCOUNT
select * from DEPOSITOR


create table CUSTOMER(
                cname varchar(20)primary key,
                cstreet varchar(25),
                ccity varchar(20)
                )


insert into CUSTOMER values('Rakesh','3rd main','karkala')
insert into CUSTOMER values('Ramesh','4th main','karkala')
insert into CUSTOMER values('Rajesh','4th block','mangalore')
insert into CUSTOMER values('Kareem','456 nagar','mangalore')
insert into CUSTOMER values('John smith','452 street','Udupi')

create table DEPOSITOR(
                cname varchar(20),
                accno int,
                primary key(cname,accno),
                foreign key(cname) references CUSTOMER(cname) on
delete cascade on update cascade,
                foreign key(accno) references ACCOUNT(accno) on delete
cascade on update cascade,
                unique(accno)
                 )

select * from account
insert into DEPOSITOR values('Rakesh',12340)
insert into DEPOSITOR values('Rakesh',13345)
insert into DEPOSITOR values('Rakesh',14345)
insert into DEPOSITOR values('Rakesh',13346)
insert into DEPOSITOR values('Rakesh',15342)
insert into DEPOSITOR values('Rakesh',14341)

insert into DEPOSITOR values('Ramesh',12345)
insert into DEPOSITOR values('Ramesh',12375)
insert into DEPOSITOR values('Ramesh',12377)
insert into DEPOSITOR values('Ramesh',12378)
insert into DEPOSITOR values('Ramesh',12450)
insert into DEPOSITOR values('Ramesh',13340)
insert into DEPOSITOR values('Ramesh',12451)
insert into DEPOSITOR values('Ramesh',12452)
insert into DEPOSITOR values('Ramesh',12455)

insert into DEPOSITOR values('Kareem',21346)
insert into DEPOSITOR values('Kareem',13245)

insert into DEPOSITOR values('Rajesh',15345)
insert into DEPOSITOR values('Rajesh',13241)

insert into DEPOSITOR values('John smith',21345)
insert into DEPOSITOR values('John smith',12453)
insert into DEPOSITOR values('John smith',13347)

delete from DEPOSITOR where accno = 12450

select * from depositor D , account A where D.accno = A.accno and
D.cname = 'rakesh'

select * from depositor D , account A where D.accno = A.accno and
D.cname = 'John smith'

select * from customer


select distinct bcity from branch
select  bcity from branch

select * from DEPOSITOR
select * from ACCOUNT


select * from CUSTOMER


create table LOAN (
                loanno int,
                bname varchar(15),
                amount real,
                primary key(loanno),
                foreign key(bname) references BRANCH(bname) on delete
cascade on update cascade

            )


insert into LOAN values(1,'Corp_mang',12000)
insert into LOAN values(2,'Corp_mang',11000)
insert into LOAN values(3,'Corp_mang',10000)

insert into LOAN values(4,'Corp_nitte',16000)
insert into LOAN values(5,'Corp_nitte',13000)


insert into LOAN values(6,'PNB_mang',12000)
insert into LOAN values(11,'Corp_mang',10000)

insert into LOAN values(7,'state_udupi',20000)
insert into LOAN values(8,'state_udupi',23000)
insert into LOAN values(12,'synd_nitte',10000)

insert into LOAN values(9,'synd_nitte',32000)

insert into LOAN values(10,'PNB_nitte',12000)
insert into LOAN values(13,'state_udupi',12000)
insert into LOAN values(14,'synd_udupi',12000)
select * from LOAN

create table BORROWER(
                cname varchar(20),
                loanno int
                primary key(cname,loanno),
                foreign key(cname) references CUSTOMER(cname) on
delete cascade on update cascade,
                foreign key(loanno) references LOAN(loanno) on delete
cascade on update cascade,
                unique(loanno)
                )


insert into BORROWER values('John smith',1)
insert into BORROWER values('John smith',2)
insert into BORROWER values('John smith',3)
insert into BORROWER values('John smith',13)
insert into BORROWER values('John smith',14)

insert into BORROWER values('Kareem',4)
insert into BORROWER values('Kareem',5)
insert into BORROWER values('Rajesh',6)
insert into BORROWER values('Rajesh',11)
insert into BORROWER values('Rajesh',12)
insert into BORROWER values('Rajesh',7)
insert into BORROWER values('Rajesh',8)

insert into BORROWER values('Rakesh',9)
insert into BORROWER values('Ramesh',10)



bselect * from BORROWER

select * from BRANCH
```

1. Find all the customers who have at least two accounts at the Main branch.

```
select D.cname  from DEPOSITOR D , ACCOUNT A
where D.accno = A.accno and A.bname = 'state_udupi'  group by D.cname
having  count(*) >= 2
```

2A. Find all the customers who have an account at all the branches located in a specific city.

```
--select C.cname from CUSTOMER  C
--where not exists(
--     select bname from  BRANCH where bcity  = 'karkala' and bname not
in
--
--                (select distinct(A.bname) from ACCOUNT A , BRANCH
B,DEPOSITOR D
--                 where A.bname = B.bname
--                 and D.accno = A.accno
--                and B.bcity  = 'karkala'
--                and D.cname = C.cname )
--          )
--
--
```

OR

```
select C.cname from CUSTOMER  C
where not exists(
     select B.bname from  BRANCH B where bcity  = 'karkala' and
B.bname not in

                (select distinct(A.bname) from ACCOUNT A,DEPOSITOR D
                 where D.accno = A.accno
                 and  A.bname = B.bname
                and D.cname = C.cname )
          )
```

OR

```
--select C.cname from CUSTOMER  C
--where not exists(
--            select B.bname from  BRANCH B where  B.bcity  =
'karkala'
--              and not exists(
--                        (select *  from ACCOUNT A ,  DEPOSITOR D
--                         where  D.accno = A.accno
--                         and A.bname  = B.bname
--                         and D.cname = C.cname ))
--         )





2B. Find all the customers who have atleast 2  accounts at all the
branches
located in a specific city.

select C.cname from CUSTOMER  C
where not exists(
     select B.bname from  BRANCH B where B.bcity  = 'karkala' and
bname not in

                (select A.bname from ACCOUNT A ,DEPOSITOR D
                 where D.accno = A.accno
                 and A.bname = B.bname
                and D.cname = C.cname  group by A.bname having
count(*) >= 2)
           )



or

select * from CUSTOMER

4)Find all the customers who have accounts in atleast 1 branch located
in all the cities

select C.cname from CUSTOMER  C
where not exists(
                select distinct(B.bcity)   from  BRANCH B
             where  not exists
                        (

                 select A.bname from ACCOUNT A ,DEPOSITOR D
                 where  D.accno = A.accno
                 and D.cname =C.cname  and  A.bname  in (select bname
from BRANCH where bcity = B.bcity)

                )
           )

OR


select C.cname from CUSTOMER  C
where  not  exists(

          select distinct(B1.bcity)   from  BRANCH B1
          where not exists(

           select  count( distinct B.bname) from BRANCH B, ACCOUNT A
,DEPOSITOR D
              where A.bname = B.bname
            and D.accno = A.accno
            and B.bcity  = B1.bcity
            and D.cname = C.cname   group by B.bcity having count(*)
>=1))



select * from customer
select * from branch
```

3)Find all the customers who have accounts in atleast 2 branches located in a specific city.

```
select C.cname from CUSTOMER  C
where  exists(
            select  count( distinct B.bname) from BRANCH B, ACCOUNT A
,DEPOSITOR D
              where A.bname = B.bname
            and D.accno = A.accno
            and B.bcity  = 'karkala'
            and D.cname = C.cname   group by B.bcity having count(*)
>=2)
```

Find all the customers who have accounts in atleast 2 branches located in all the cities

```
select C.cname from CUSTOMER  C
where  not  exists(

          select distinct(B1.bcity)   from  BRANCH B1
          where not exists(

           select  count( distinct B.bname) from BRANCH B, ACCOUNT A
,DEPOSITOR D
              where A.bname = B.bname
            and D.accno = A.accno
            and B.bcity  = B1.bcity
            and D.cname = C.cname   group by B.bcity having count(*)
>=2))



select * from customer
select * from branch
select * from BORROWER

select bname from  BRANCH B where B.bcity  = 'karkala'

select L.bname from  BORROWER B , LOAN L   where L.loanno = B.loanno
and B.cname = 'Rajesh'
```

Find the branch name that has maximum number of customers in a specific city

```
select D.cname, A.bname, count(*) from ACCOUNT A, DEPOSITOR D
where A.accno = D.accno group by D.cname , A.bname


select   A.bname,count(distinct D.cname)   from ACCOUNT A, DEPOSITOR D
where A.accno = D.accno group by  A.bname
having   count(distinct D.cname) >= all (select   count(distinct
D.cname)    from ACCOUNT A, DEPOSITOR D
where A.accno = D.accno group by  A.bname)

select * from ACCOUNT
```

1)Give the details of all the branches having more than two account .

```
select B.bname,B.bcity
from BRANCH B,ACCOUNT A
where B.bname=A.bname
group by B.bname,B.bcity
having count(A.accno)>=2
```

2)Display the loan details of each customer.
(Details include custname,branchname,no of loans , total amount at the branch)

```
select B.cname,L.bname,COUNT(L.loanno),SUM(L.amount)
from LOAN L,BORROWER B
where L.loanno=B.loanno
group by B.cname,L.bname
```

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

```
select D.cname
from DEPOSITOR D,ACCOUNT A
where A.accno=D.accno and A.bname='state_udupi'
group by D.cname
having count(*)>=2
```

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

```
select distinct D.cname
from DEPOSITOR D
where not exists( select B.bname
                  from BRANCH B
                  where B.bcity='karkala'
                  and B.bname not in( select bname
                                      from ACCOUNT A,DEPOSITOR D1
                                      where A.accno=D1.accno
                                      and A.bname=B.bname
                                      and D1.cname=D.cname))
```

3.   Find all the customers who have accounts in atleast 2 branches located in a specific city.

```
select D.cname
from DEPOSITOR D,ACCOUNT A,BRANCH B
where D.accno=A.accno and A.bname=B.bname and B.bcity='karkala'
group by D.cname
having count(*)>=2
```

4.   Find all the customers who have accounts in atleast 1 branch located in all the cities

```
select C.cname
from CUSTOMER  C
where  not  exists(select distinct(B1.bcity)
                   from  BRANCH B1
                    where not exists(select  count( distinct B.bname)
                                     from BRANCH B, ACCOUNT A
,DEPOSITOR D
                                    where A.bname = B.bname
                                     and D.accno =A.accno  and B.bcity
= B1.bcity
                                     and D.cname =C.cname
                                     group by B.bcity
                                     having count(*) >=1))
```

5.   Find all the customers who have accounts in atleast 2 branches located in all the cities

```
select C.cname
from CUSTOMER C
where not exists( select distinct B1.bname
                  from BRANCH B1
                  where not exists( select COUNT(B.bname)
                                    from BRANCH B,DEPOSITOR D,ACCOUNT
A
                                    where B.bname=A.bname and
D.accno=A.accno
                                    and B.bcity=B1.bcity and
D.cname=C.cname
                                    group by B.bcity
                                    having COUNT(*)>=2))
```

6. Find the branch name that has maximum number of customers in a specific city

```
select B.bname
from BRANCH B,ACCOUNT A,DEPOSITOR D
where A.accno=D.accno and A.bname=B.bname and B.bcity='karkala'
group by B.bname
having COUNT(distinct D.cname) >=ALL (select COUNT(distinct D1.cname)
                                      from BRANCH B1,ACCOUNT
A1,DEPOSITOR D1
                                      where A1.accno=D1.accno and
A1.bname=B1.bname
                                      and B1.bcity='karkala'
                                      group by B1.bname)
```

7. Find the branch name that has maximum number of accounts in a specific city

```
select B.bname
from BRANCH B,ACCOUNT A,DEPOSITOR D
where A.accno=D.accno and A.bname=B.bname and B.bcity='karkala'
group by B.bname
having COUNT(distinct A.accno) >=ALL (select COUNT(A1.accno)
                                      from BRANCH B1,ACCOUNT
A1,DEPOSITOR D1
                                      where A1.accno=D1.accno and
A1.bname=B1.bname
                                      and B1.bcity='karkala'
                                      group by B1.bname)
```

8. Find the customer name who has deposited maximum amount at branches located in a specific city.

```
select D.cname
from DEPOSITOR D,ACCOUNT A,BRANCH B
where A.accno=D.accno and A.bname=B.bname and B.bcity='karkala'
group by D.cname
having SUM(A.balance) >= ALL(select SUM(A1.balance)
                             from DEPOSITOR D1,ACCOUNT A1,BRANCH B1
                             where A1.accno=D1.accno and
A1.bname=B1.bname and B1.bcity='karkala'
                             group by D1.cname)
```

9.List CUSTOMER_NAME,#AMOUNT where #AMOUNT is total amount at a branch located in different cities.

```
 select D.cname,A.balance,B.bcity
 from ACCOUNT A,DEPOSITOR D,BRANCH B
 where D.accno=A.accno and B.bname=A.bname
 group by B.bcity,D.cname,A.balance
```

10. Find the customers who have borrowed loan from all the branches located in a specific city

```
select distinct B.cname
from BORROWER B
where not exists( select B1.bname
                  from BRANCH B1
                  where B1.bcity='Mangalore'
                  and B1.bname not in( select L.bname
                                       from BORROWER B2,LOAN L
                                       where B2.loanno=L.loanno and
B2.cname=B.cname))
```

11. Find the customers who have borrowed loan from atleast one branch located in all the cities

```
select distinct B.cname
from BORROWER B
where not exists( select distinct B1.bcity
                  from BRANCH B1
                  where not exists( select COUNT(distinct L.bname)
                                    from LOAN L,BORROWER B2,BRANCH C
                                    where L.loanno=B2.loanno and
L.bname=C.bname
                                    and B2.cname=B.cname and
C.bcity=B1.bcity
                                    group by C.bcity
                                    having COUNT(*)>=1))
```

12. Find the customers who have borrowed loan from atleast 2 branch located in all the cities

```
select distinct B.cname
from BORROWER B
where not exists( select distinct B1.bcity
                  from BRANCH B1
                  where not exists( select count(distinct L.bname)
                                    from LOAN L,BORROWER B2,BRANCH C
                                    where L.loanno=B.loanno and
B2.cname=B.cname
                                    and C.bcity=B1.bcity and
L.bname=C.bname
                                    group by C.bcity
                                    having COUNT(*)>=2))
```

a). Give the details of all the branches having more than two account

```
select B.bname,B.bcity
from BRANCH B,ACCOUNT A
where B.bname=A.bname
```

```
group by B.bname ,B.bcity
having COUNT(distinct A.accno)>=2
```

b)Display the loan details of each customer.

```
select L.loanno,L.bname,B1.cname
from LOAN L,BRANCH B,BORROWER B1
where L.loanno=B1.loanno and B.bname=L.bname
```

find the no.of loans in the branch having maximum customers

```
select B.bname into tb1
from DEPOSITOR D,BRANCH B,ACCOUNT A
where A.bname=B.bname and D.accno=A.accno
group by B.bname
having COUNT(distinct D.cname)>=ALL( select COUNT(distinct D1.cname)
                                     from DEPOSITOR D1,BRANCH
B1,ACCOUNT A1
                                     where A1.bname=B1.bname and
D1.accno=A1.accno
                                     group by B1.bname)

select * from tb1

select COUNT(distinct L.loanno)
from LOAN L,tb1 T
where L.bname=T.bname
```

list the customers who have borrowed money from every branch located in a specific city

```
select C.cname
from CUSTOMER C
where not exists( select B.bname
                  from BRANCH B
                  where B.bcity='Mangalore'
                  and B.bname not in( select distinct L.bname
                                      from BORROWER B1,LOAN L
                                      where B1.loanno=L.loanno and
L.bname=B.bname
                                      and B1.cname=C.cname))
```

list the customer name,NO_OF_LOANS,TOTAL_LOAN_AMOUNT for the customers who have borrowed money from
atleast two branches in their own city

```
select C.cname into tb
from CUSTOMER C
where exists( select count(distinct B.bname)
              from BRANCH B,LOAN L,BORROWER A
              where L.loanno=A.loanno and B.bname=L.bname
              and A.cname=C.cname and B.bcity=C.ccity
              group by B.bcity
              having count(distinct B.bname)>=2)

select * from tb

select T.cname,COUNT(L.loanno) as No_of_loans,sum(L.amount) as
Total_Amount
from tb T,LOAN L,BORROWER B
where T.cname=B.cname and L.loanno=B.loanno
group by T.cname
```

c)Find the customer who is having maximum loans

```
select C.cname
from BORROWER C,LOAN L
where C.loanno=L.loanno
group by C.cname
having COUNT(distinct L.loanno)>=ALL( select COUNT(distinct L1.loanno)
                                      from LOAN L1,BORROWER B
                                      where L1.loanno=B.loanno
                                      group by B.cname)
```

d)display the customer’s balance amount at each branch.

```
select C.cname,L.bname,sum(L.amount) as total_amount
from BORROWER C,LOAN L
where L.loanno=C.loanno
group by C.cname,L.bname
```

e)Give the details of any branch which has maximum customers.

```
select B.bname,B.bcity
from BRANCH B,ACCOUNT A,DEPOSITOR D
where B.bname=A.bname and D.accno=A.accno
group by B.bname,B.bcity
having COUNT(distinct D.cname)>=ALL( select COUNT(distinct D1.cname)
                                     from BRANCH B1,ACCOUNT
A1,DEPOSITOR D1
                                     where B1.bname=A1.bname and
D1.accno=A1.accno
                                     group by B1.bname,B1.bcity)
```

g)Find the customer(if any) who does not have an account at all the
branch located in
a specific city.

```
select C.cname
from CUSTOMER C
where not exists( select B.bname
                  from BRANCH B
                  where B.bcity='Udupi'
                  and B.bname in( select B1.bname
                                  from ACCOUNT A,BRANCH B1,DEPOSITOR D
                                  where B1.bname=A.bname and
D.accno=A.accno

                                  and D.cname=C.cname
                                  group by B1.bname))
```

h)Find the customer having maximum accounts.

```
select D.cname
from ACCOUNT A,DEPOSITOR D
where D.accno=A.accno
group by D.cname
having COUNT(distinct A.accno) >=ALL (select COUNT(distinct A1.accno)
                                      from ACCOUNT A1,DEPOSITOR D1
                                      where A1.accno=D1.accno
                                      group by D1.cname )
```