B.M.S. COLLEGE OF ENGINEERING

(AUTONOMOUS COLLEGE UNDER VTU)
BENGALURU-19

LAB TEST 1 REPORT

NAME: Ishan Bhandari

USN: 1BM19CS198

COURSE NAME: DATABASE MANAGEMENT SYSTEMS

COURSE TITLE: 19CS4PCDBM

SEMESTER: 4

SECTION: D

LAB PROGRAMS 1-5:

PROGRAM 1: INSURANCE DATABASE

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

PERSON (driver_id: String, name: String, address: String)

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

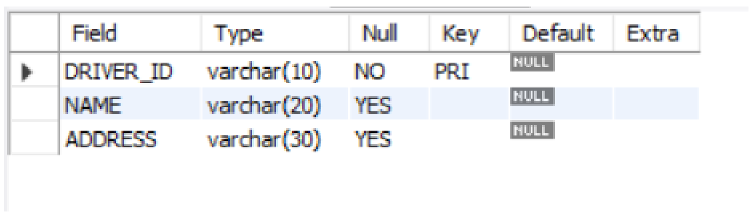
ACCIDENT (report_num: int, accident_date: date, location: String)

OWNS (driver_id: String, reg_num: String)

PARTICIPATED (driver_id: String, reg_num: String, report_num: int, damage_amount: int)

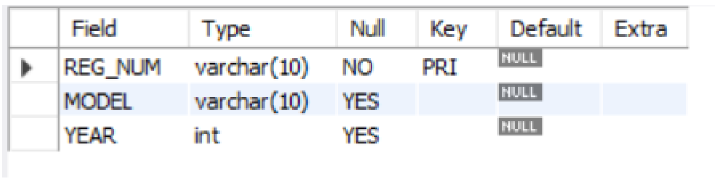
i) Create the above tables by properly specifying the primary keys and the foreign keys.

CREATE TABLE PERSON(DRIVER_ID VARCHAR(10), NAME VARCHAR(20), ADDRESS VARCHAR(30), PRIMARY KEY (DRIVER_ID));



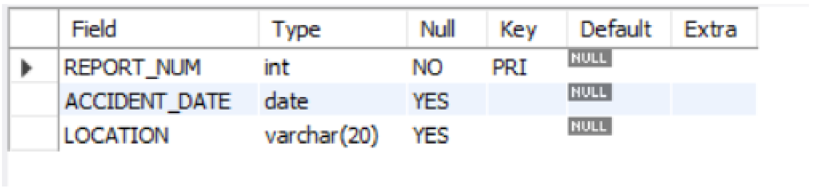
Field	Type	Null	Key	Default	Extra
DRIVER_ID	varchar(10)	NO	PRI	NULL	
NAME	varchar(20)	YES		NULL	
ADDRESS	varchar(30)	YES		NULL	

CREATE TABLE CAR(REG_NUM VARCHAR(10), MODEL VARCHAR(10), YEAR INT, PRIMARY KEY(REG_NUM));



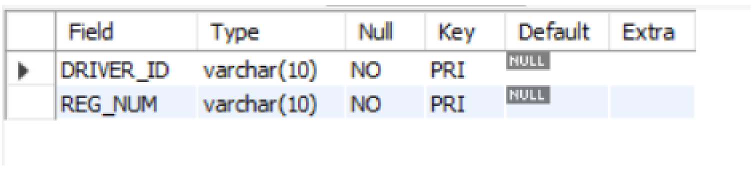
Field	Type	Null	Key	Default	Extra
REG_NUM	varchar(10)	NO	PRI	NULL	
MODEL	varchar(10)	YES		NULL	
YEAR	int	YES		NULL	

CREATE TABLE ACCIDENT(REPORT_NUM INT, ACCIDENT_DATE DATE, LOCATION VARCHAR(20), PRIMARY KEY(REPORT_NUM));



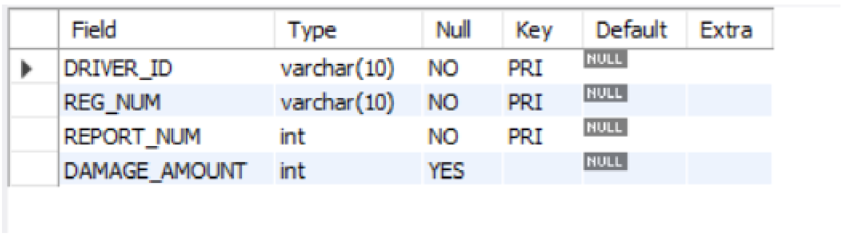
Field	Type	Null	Key	Default	Extra
REPORT_NUM	int	NO	PRI	NULL	
ACCIDENT_DATE	date	YES		NULL	
LOCATION	varchar(20)	YES		NULL	

CREATE TABLE OWNS(DRIVER_ID VARCHAR(10), REG_NUM VARCHAR(10), PRIMARY KEY(DRIVER_ID, REG_NUM), FOREIGN KEY(DRIVER_ID) REFERENCES PERSON(DRIVER_ID), FOREIGN KEY(REG_NUM) REFERENCES CAR (REG_NUM));



Field	Type	Null	Key	Default	Extra
DRIVER_ID	varchar(10)	NO	PRI	NULL	
REG_NUM	varchar(10)	NO	PRI	NULL	

CREATE TABLE PARTICIPATED(DRIVER_ID VARCHAR(10), REG_NUM VARCHAR(10), REPORT_NUM INT, DAMAGE_AMOUNT INT, PRIMARY KEY(DRIVER_ID, REG_NUM, REPORT_NUM), FOREIGN KEY(DRIVER_ID) REFERENCES PERSON(DRIVER_ID), FOREIGN KEY(REG_NUM) REFERENCES CAR(REG_NUM), FOREIGN KEY(REPORT_NUM) REFERENCES ACCIDENT (REPORT_NUM));

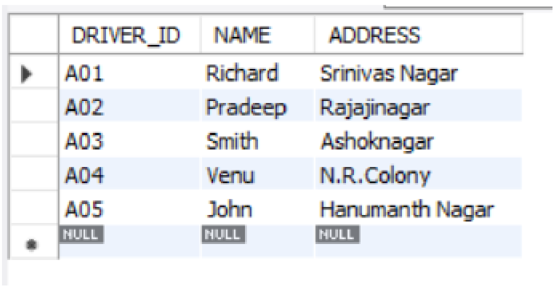


Field	Type	Null	Key	Default	Extra
DRIVER_ID	varchar(10)	NO	PRI	NULL	
REG_NUM	varchar(10)	NO	PRI	NULL	
REPORT_NUM	int	NO	PRI	NULL	
DAMAGE_AMOUNT	int	YES		NULL	

ii)Enter at least five tuples for each relation.

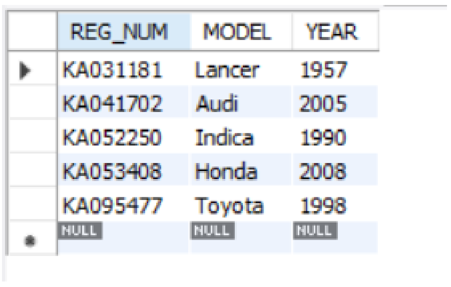
INSERT INTO PERSON VALUES('A01', 'Richard', 'Srinivas Nagar');
INSERT INTO PERSON VALUES('A02', 'Pradeep', 'Rajajinagar');

```
INSERT INTO PERSON VALUES('A03', 'Smith', 'Ashoknagar');
INSERT INTO PERSON VALUES('A04', 'Venu', 'N.R.Colony');
INSERT INTO PERSON VALUES('A05', 'John', 'Hanumanth Nagar');
```



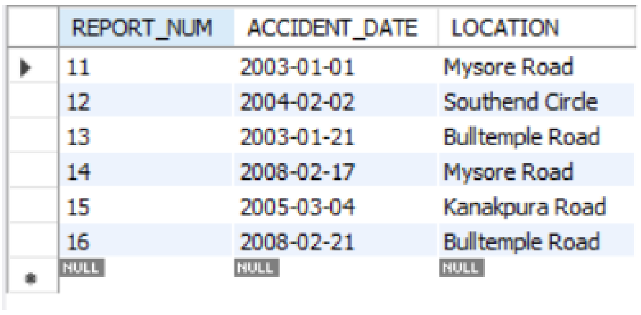
DRIVER_ID	NAME	ADDRESS
A01	Richard	Srinivas Nagar
A02	Pradeep	Rajajinagar
A03	Smith	Ashoknagar
A04	Venu	N.R.Colony
A05	John	Hanumanth Nagar
NULL	NULL	NULL

```
INSERT INTO CAR VALUES('KA052250', 'Indica', 1990);
INSERT INTO CAR VALUES('KA031181', 'Lancer', 1957);
INSERT INTO CAR VALUES('KA095477', 'Toyota', 1998);
INSERT INTO CAR VALUES('KA053408', 'Honda', 2008);
INSERT INTO CAR VALUES('KA041702', 'Audi', 2005);
```



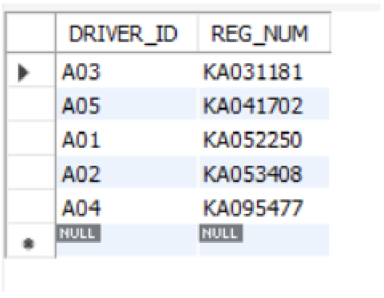
REG_NUM	MODEL	YEAR
KA031181	Lancer	1957
KA041702	Audi	2005
KA052250	Indica	1990
KA053408	Honda	2008
KA095477	Toyota	1998
NULL	NULL	NULL

```
INSERT INTO ACCIDENT VALUES(11, '2003-01-01', 'Mysore Road');
INSERT INTO ACCIDENT VALUES(12, '2004-02-02', 'Southend Circle');
INSERT INTO ACCIDENT VALUES(13, '2003-01-21', 'Bulltemple Road');
INSERT INTO ACCIDENT VALUES(14, '2008-02-17', 'Mysore Road');
INSERT INTO ACCIDENT VALUES(15, '2005-03-04', 'Kanakpura Road');
```



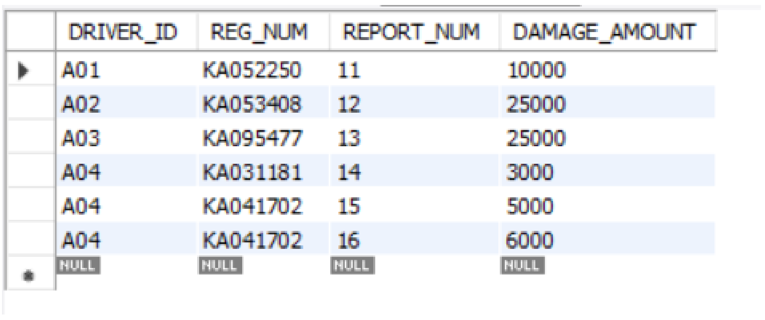
REPORT_NUM	ACCIDENT_DATE	LOCATION
11	2003-01-01	Mysore Road
12	2004-02-02	Southend Circle
13	2003-01-21	Bulltemple Road
14	2008-02-17	Mysore Road
15	2005-03-04	Kanakpura Road
16	2008-02-21	Bulltemple Road
NULL	NULL	NULL

```
INSERT INTO OWNS VALUES('A01', 'KA052250');
INSERT INTO OWNS VALUES('A02', 'KA053408');
INSERT INTO OWNS VALUES('A03', 'KA031181');
INSERT INTO OWNS VALUES('A04', 'KA095477');
INSERT INTO OWNS VALUES('A05', 'KA041702');
```



DRIVER_ID	REG_NUM
A03	KA031181
A05	KA041702
A01	KA052250
A02	KA053408
A04	KA095477
NULL	NULL

```
INSERT INTO PARTICIPATED VALUES('A01', 'KA052250', 11, 10000);
INSERT INTO PARTICIPATED VALUES('A02', 'KA053408', 12, 50000);
INSERT INTO PARTICIPATED VALUES('A03', 'KA095477', 13, 25000);
INSERT INTO PARTICIPATED VALUES('A04', 'KA031181', 14, 3000);
INSERT INTO PARTICIPATED VALUES('A05', 'KA041702', 15, 5000);
```

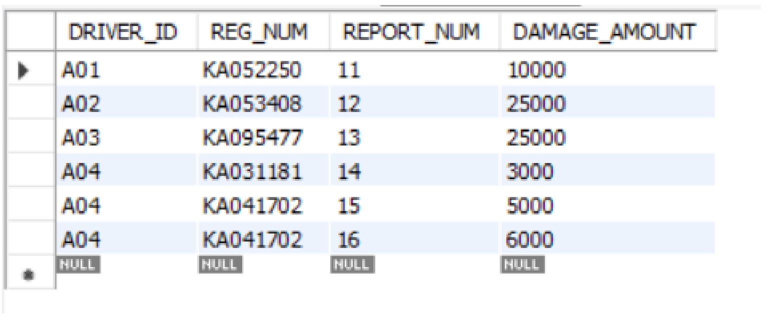


DRIVER_ID	REG_NUM	REPORT_NUM	DAMAGE_AMOUNT
A01	KA052250	11	10000
A02	KA053408	12	25000
A03	KA095477	13	25000
A04	KA031181	14	3000
A04	KA041702	15	5000
A04	KA041702	16	6000
NULL	NULL	NULL	NULL

iii) Demonstrate how you:

a. Update the damage
amount to 25000 for the car with a specific reg-num(example 'K A053408') for which the accident report number was 12.

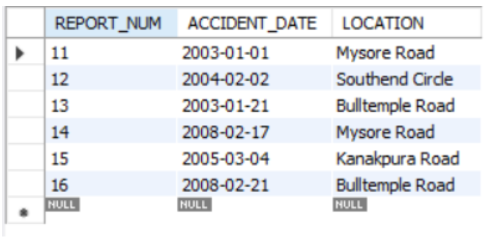
```
UPDATE PARTICIPATED SET DAMAGE_AMOUNT = 25000 WHERE
REPORT_NUM = 12;
```



DRIVER_ID	REG_NUM	REPORT_NUM	DAMAGE_AMOUNT
A01	KA052250	11	10000
A02	KA053408	12	25000
A03	KA095477	13	25000
A04	KA031181	14	3000
A04	KA041702	15	5000
A04	KA041702	16	6000
NULL	NULL	NULL	NULL

b. Add a new accident to the database.

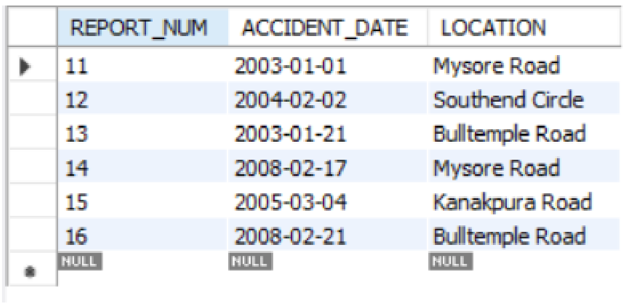
INSERT INTO ACCIDENT VALUES(16, '2008-02-21', 'Bulltemple Road');



REPORT_NUM	ACCIDENT_DATE	LOCATION
11	2003-01-01	Mysore Road
12	2004-02-02	Southend Circle
13	2003-01-21	Bulltemple Road
14	2008-02-17	Mysore Road
15	2005-03-04	Kanakpura Road
16	2008-02-21	Bulltemple Road
NULL	NULL	NULL

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

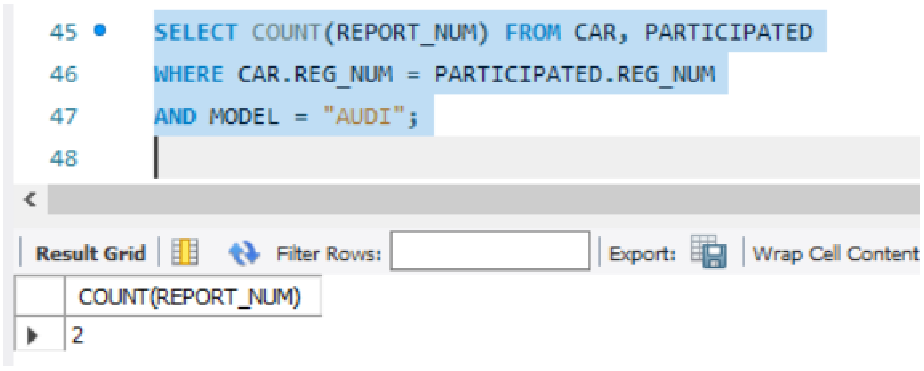
SELECT COUNT(DISTINCT DRIVER_ID) FROM ACCIDENT, PARTICIPATED
WHERE ACCIDENT.REPORT_NUM = PARTICIPATED.REPORT_NUM
AND ACCIDENT_DATE LIKE '2008%';



REPORT_NUM	ACCIDENT_DATE	LOCATION
11	2003-01-01	Mysore Road
12	2004-02-02	Southend Circle
13	2003-01-21	Bulltemple Road
14	2008-02-17	Mysore Road
15	2005-03-04	Kanakpura Road
16	2008-02-21	Bulltemple Road
NULL	NULL	NULL

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

SELECT COUNT(REPORT_NUM) FROM CAR, PARTICIPATED
WHERE CAR.REG_NUM = PARTICIPATED.REG_NUM
AND MODEL = "AUDI";



```
SELECT COUNT(REPORT_NUM) FROM CAR, PARTICIPATED
WHERE CAR.REG_NUM = PARTICIPATED.REG_NUM
AND MODEL = "AUDI";
```

```
COUNT(REPORT_NUM)
2
```

PROGRAM 2: BANKING ENTERPRISE DATABASE

Consider the following database for a banking enterprise.

Branch (branch-name: String, branch-city: String, assets: real)

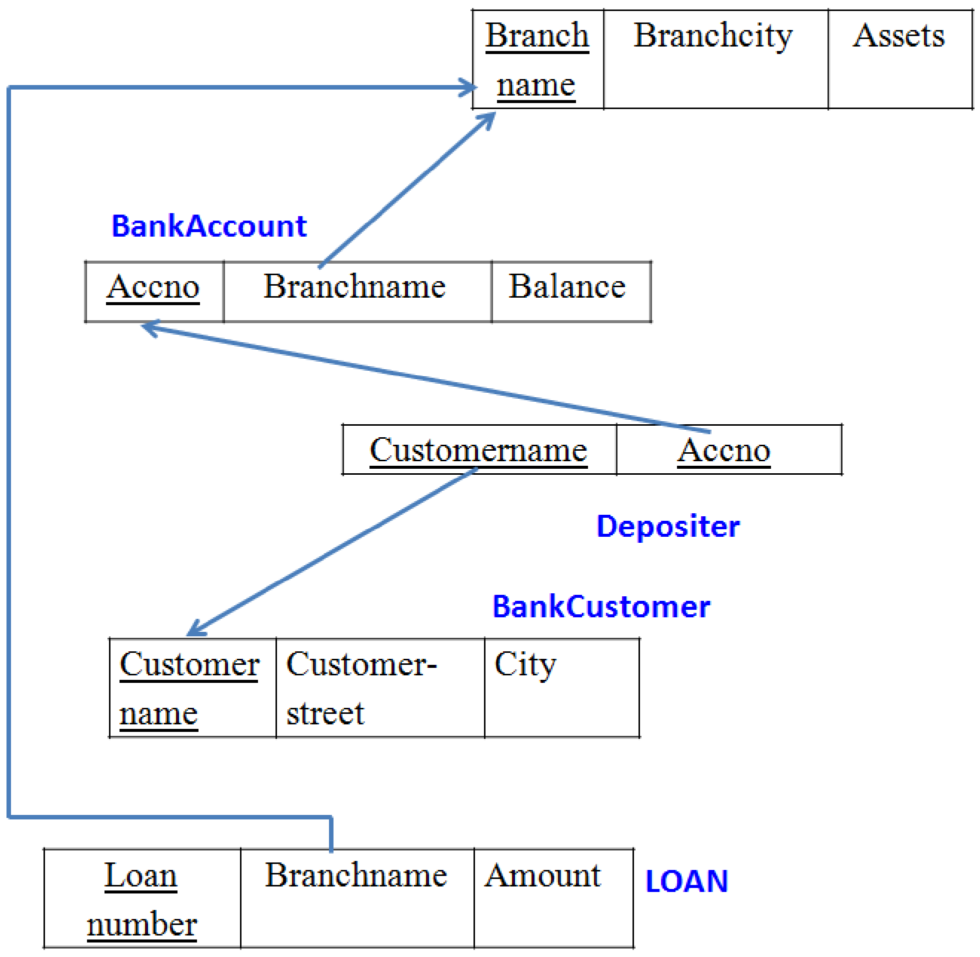
BankAccount(accno: int, branch-name: String, balance: real)

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

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

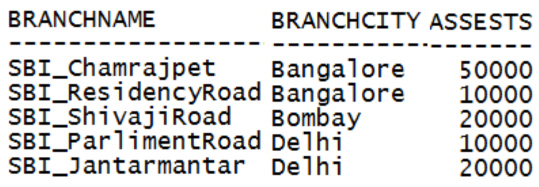
Loan (loan-number: int, branch-name: String, amount: real)

Schema Diagram



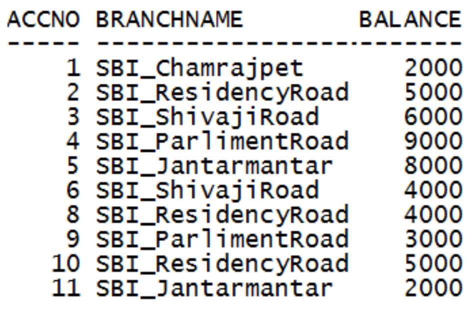
Sample Table data

Branch



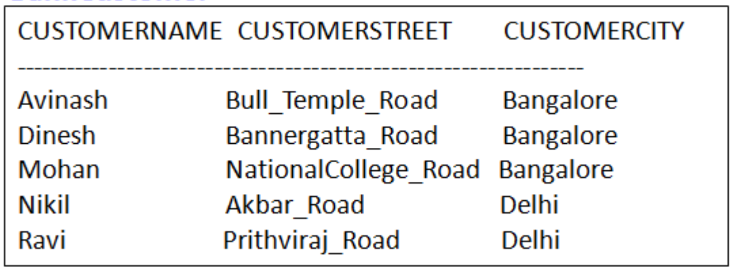
```
BRANCHNAME         BRANCHCITY ASSESTS
------------------ ----------------- 
SBI_Chamrajpet     Bangalore    50000
SBI_ResidencyRoad  Bangalore    10000
SBI_ShivajiRoad    Bombay       20000
SBI_ParlimentRoad  Delhi        10000
SBI_Jantarmantar   Delhi        20000
```

BankAccount



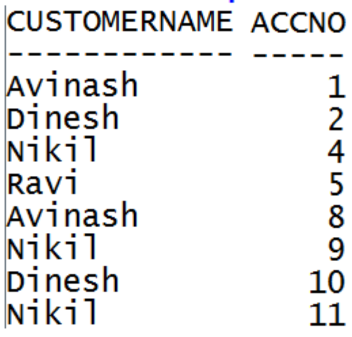
```
ACCNO BRANCHNAME           BALANCE
----- ------------------------------
    1 SBI_Chamrajpet          2000
    2 SBI_ResidencyRoad       5000
    3 SBI_ShivajiRoad         6000
    4 SBI_ParlimentRoad       9000
    5 SBI_Jantarmantar        8000
    6 SBI_ShivajiRoad         4000
    8 SBI_ResidencyRoad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

BankCustomer



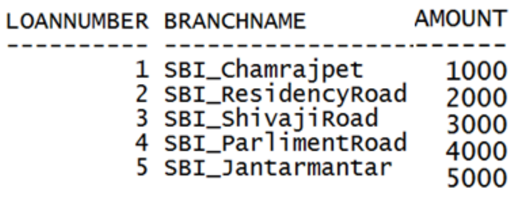
```
CUSTOMERNAME  CUSTOMERSTREET       CUSTOMERCITY
-------------------------------------------------------------
Avinash       Bull_Temple_Road       Bangalore
Dinesh        Bannergatta_Road       Bangalore
Mohan         NationalCollege_Road   Bangalore
Nikil         Akbar_Road             Delhi
Ravi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

Depositer



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

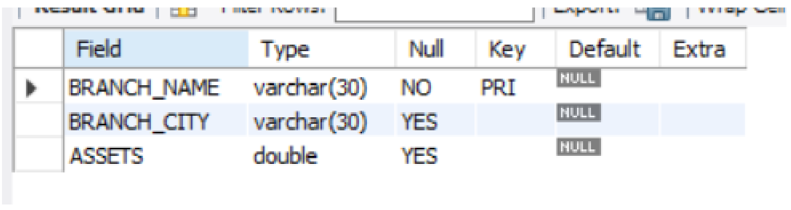
Loan



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

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

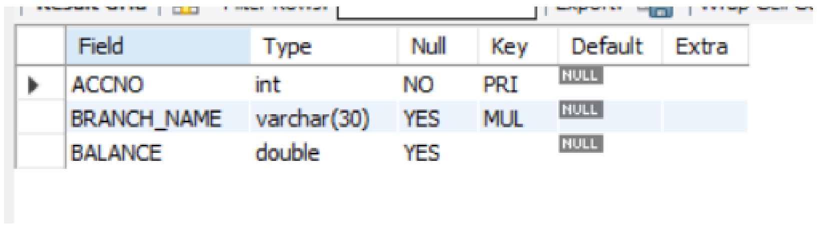
CREATE TABLE BRANCH (BRANCH_NAME VARCHAR(30), BRANCH_CITY VARCHAR(30), ASSETS REAL, PRIMARY KEY (BRANCH_NAME));



Field	Type	Null	Key	Default	Extra
BRANCH_NAME	varchar(30)	NO	PRI	NULL	
BRANCH_CITY	varchar(30)	YES		NULL	
ASSETS	double	YES		NULL	

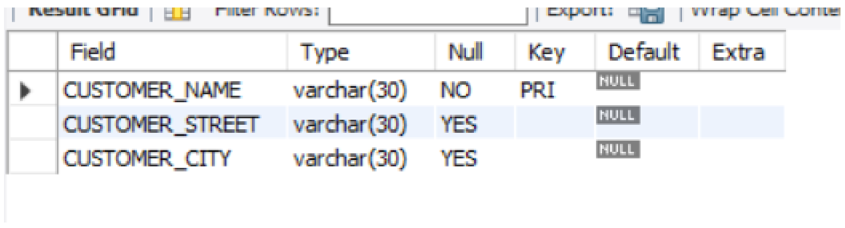
CREATE TABLE BANK_ACCOUNT (ACCNO INT, BRANCH_NAME

VARCHAR(30), BALANCE REAL, PRIMARY KEY (ACCNO), FOREIGN KEY (BRANCH_NAME) REFERENCES BRANCH(BRANCH_NAME));



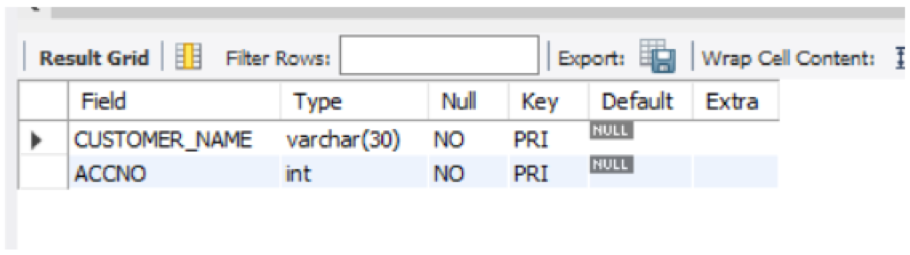
Field	Type	Null	Key	Default	Extra
ACCNO	int	NO	PRI	NULL	
BRANCH_NAME	varchar(30)	YES	MUL	NULL	
BALANCE	double	YES		NULL	

CREATE TABLE BANK_CUSTOMER (CUSTOMER_NAME VARCHAR(30), CUSTOMER_STREET VARCHAR(30), CUSTOMER_CITY VARCHAR(30), PRIMARY KEY(CUSTOMER_NAME));



Field	Type	Null	Key	Default	Extra
CUSTOMER_NAME	varchar(30)	NO	PRI	NULL	
CUSTOMER_STREET	varchar(30)	YES		NULL	
CUSTOMER_CITY	varchar(30)	YES		NULL	

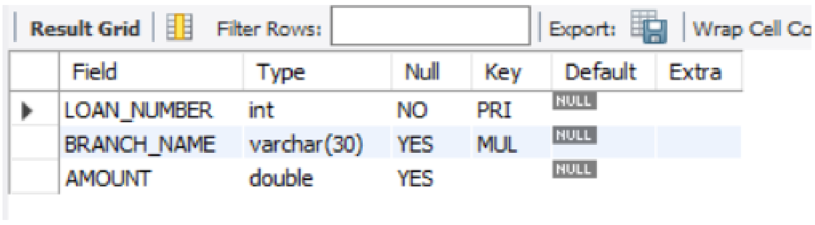
CREATE TABLE DEPOSITER (CUSTOMER_NAME VARCHAR(30), ACCNO INT, PRIMARY KEY(CUSTOMER_NAME, ACCNO), FOREIGN KEY (CUSTOMER_NAME) REFERENCES BANK_CUSTOMER (CUSTOMER_NAME), FOREIGN KEY (ACCNO) REFERENCES BANK_ACCOUNT(ACCNO));



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

Field	Type	Null	Key	Default	Extra
CUSTOMER_NAME	varchar(30)	NO	PRI	NULL	
ACCNO	int	NO	PRI	NULL	

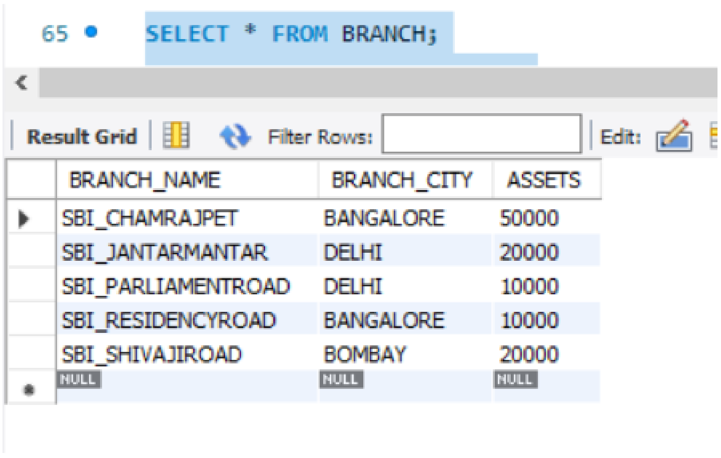
CREATE TABLE LOAN (LOAN_NUMBER INT, BRANCH_NAME VARCHAR(30), AMOUNT REAL, PRIMARY KEY (LOAN_NUMBER), FOREIGN KEY (BRANCH_NAME) REFERENCES BRANCH(BRANCH_NAME));



Field	Type	Null	Key	Default	Extra
LOAN_NUMBER	int	NO	PRI	NULL	
BRANCH_NAME	varchar(30)	YES	MUL	NULL	
AMOUNT	double	YES		NULL	

ii. Enter at least five tuples for each relation.

INSERT INTO BRANCH VALUES ('SBI_CHAMRAJPET', 'BANGALORE', 50000);
INSERT INTO BRANCH VALUES ('SBI_RESIDENCYROAD', 'BANGALORE', 10000);
INSERT INTO BRANCH VALUES ('SBI_SHIVAJIROAD', 'BOMBAY', 20000);
INSERT INTO BRANCH VALUES ('SBI_PARLIAMENTROAD', 'DELHI', 10000);
INSERT INTO BRANCH VALUES ('SBI_JANTARMANTAR', 'DELHI', 20000);

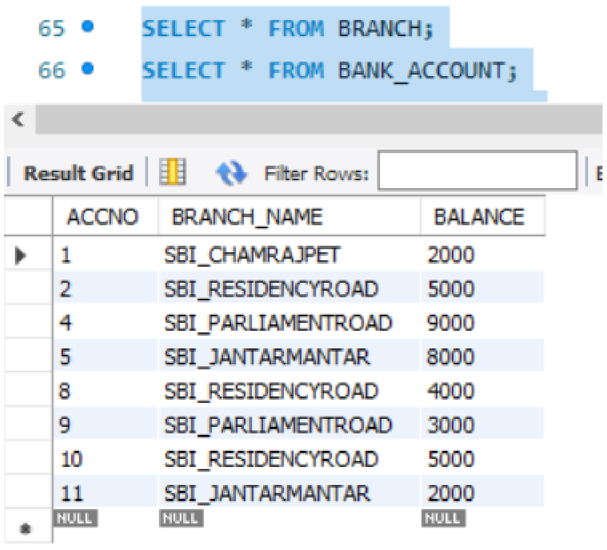


```
SELECT * FROM BRANCH;
```

BRANCH_NAME	BRANCH_CITY	ASSETS
SBI_CHAMRAJPET	BANGALORE	50000
SBI_JANTARMANTAR	DELHI	20000
SBI_PARLIAMENTROAD	DELHI	10000
SBI_RESIDENCYROAD	BANGALORE	10000
SBI_SHIVAJIROAD	BOMBAY	20000
NULL	NULL	NULL

INSERT INTO BANK_ACCOUNT VALUES (1,'SBI_CHAMRAJPET', 2000);
INSERT INTO BANK_ACCOUNT VALUES (2,'SBI_RESIDENCYROAD', 5000);
INSERT INTO BANK_ACCOUNT VALUES (3,'SBI_SHIVAJIROAD', 6000);
INSERT INTO BANK_ACCOUNT VALUES (4,'SBI_PARLIAMENTROAD', 9000);
INSERT INTO BANK_ACCOUNT VALUES (5,'SBI_JANTARMANTAR', 8000);
INSERT INTO BANK_ACCOUNT VALUES (6,'SBI_SHIVAJIROAD', 4000);

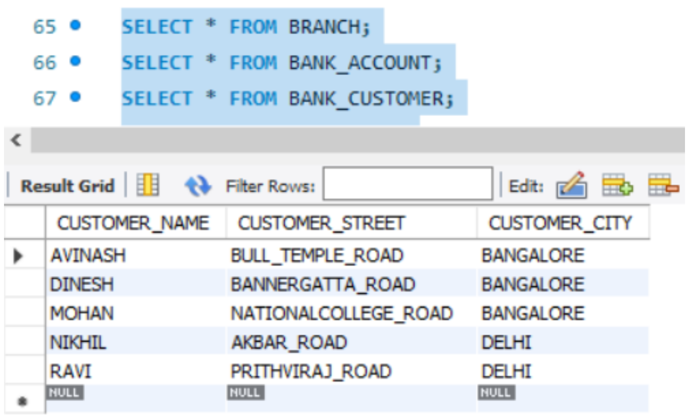
INSERT INTO BANK_ACCOUNT VALUES (8,'SBI_RESIDENCYROAD', 4000);
INSERT INTO BANK_ACCOUNT VALUES (9,'SBI_PARLIAMENTROAD', 3000);
INSERT INTO BANK_ACCOUNT VALUES (10,'SBI_RESIDENCYROAD', 5000);
INSERT INTO BANK_ACCOUNT VALUES (11,'SBI_JANTARMANTAR', 2000);



```
SELECT * FROM BRANCH;
SELECT * FROM BANK_ACCOUNT;
```

ACCNO	BRANCH_NAME	BALANCE
1	SBI_CHAMRAJPET	2000
2	SBI_RESIDENCYROAD	5000
4	SBI_PARLIAMENTROAD	9000
5	SBI_JANTARMANTAR	8000
8	SBI_RESIDENCYROAD	4000
9	SBI_PARLIAMENTROAD	3000
10	SBI_RESIDENCYROAD	5000
11	SBI_JANTARMANTAR	2000
NULL	NULL	NULL

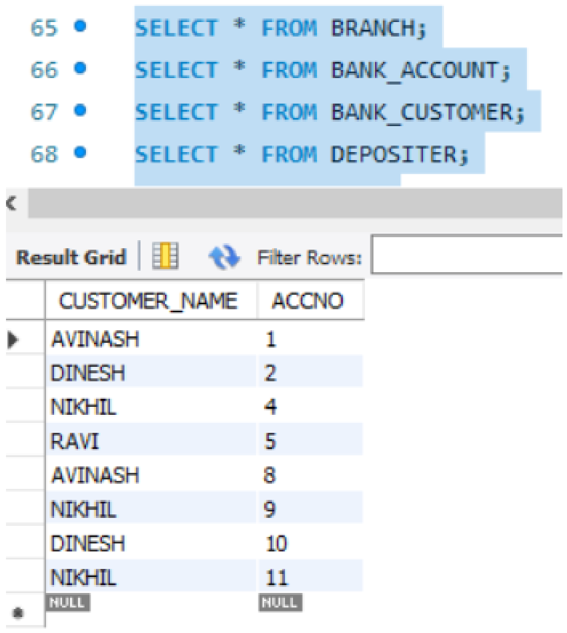
INSERT INTO BANK_CUSTOMER VALUES ('AVINASH', 'BULL_TEMPLE_ROAD', 'BANGALORE');
INSERT INTO BANK_CUSTOMER VALUES ('DINESH', 'BANNERGATTA_ROAD', 'BANGALORE');
INSERT INTO BANK_CUSTOMER VALUES ('MOHAN', 'NATIONALCOLLEGE_ROAD', 'BANGALORE');
INSERT INTO BANK_CUSTOMER VALUES ('NIKHIL', 'AKBAR_ROAD', 'DELHI');
INSERT INTO BANK_CUSTOMER VALUES ('RAVI', 'PRITHVIRAJ_ROAD', 'DELHI');



```
SELECT * FROM BRANCH;
SELECT * FROM BANK_ACCOUNT;
SELECT * FROM BANK_CUSTOMER;
```

CUSTOMER_NAME	CUSTOMER_STREET	CUSTOMER_CITY
AVINASH	BULL_TEMPLE_ROAD	BANGALORE
DINESH	BANNERGATTA_ROAD	BANGALORE
MOHAN	NATIONALCOLLEGE_ROAD	BANGALORE
NIKHIL	AKBAR_ROAD	DELHI
RAVI	PRITHVIRAJ_ROAD	DELHI
NULL	NULL	NULL

INSERT INTO DEPOSITER VALUES('AVINASH', 1);
INSERT INTO DEPOSITER VALUES('DINESH', 2);
INSERT INTO DEPOSITER VALUES('NIKHIL', 4);
INSERT INTO DEPOSITER VALUES('RAVI', 5);
INSERT INTO DEPOSITER VALUES('AVINASH', 8);
INSERT INTO DEPOSITER VALUES('NIKHIL', 9);
INSERT INTO DEPOSITER VALUES('DINESH', 10);
INSERT INTO DEPOSITER VALUES('NIKHIL', 11);

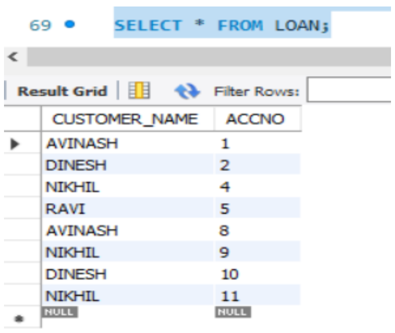


```
SELECT * FROM BRANCH;
SELECT * FROM BANK_ACCOUNT;
SELECT * FROM BANK_CUSTOMER;
SELECT * FROM DEPOSITER;
```

CUSTOMER_NAME	ACCNO
AVINASH	1
DINESH	2
NIKHIL	4
RAVI	5
AVINASH	8
NIKHIL	9
DINESH	10
NIKHIL	11
NULL	NULL

INSERT INTO LOAN VALUES (1, 'SBI_CHAMRAJPET', 1000);
INSERT INTO LOAN VALUES (2, 'SBI_RESIDENCYROAD', 2000);
INSERT INTO LOAN VALUES (3, 'SBI_SHIVAJIROAD', 3000);

INSERT INTO LOAN VALUES (4, 'SBI_PARLIAMENTROAD', 4000);
INSERT INTO LOAN VALUES (5, 'SBI_JANTARMANTAR', 5000);

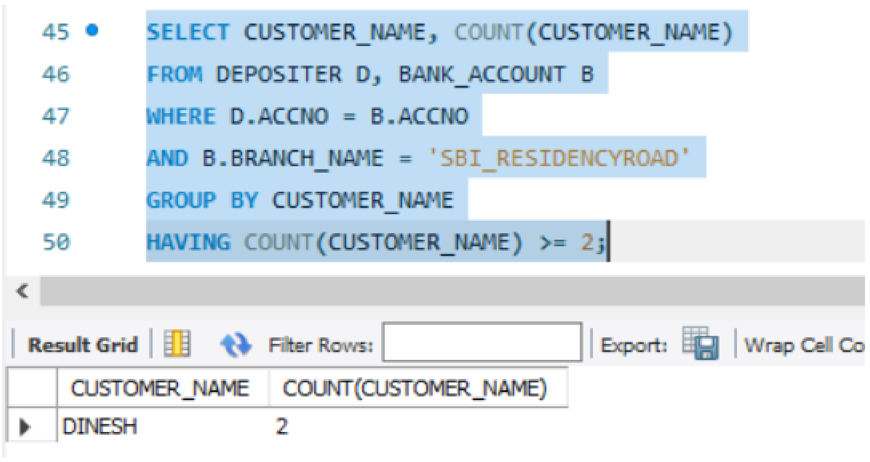


```
SELECT * FROM LOAN;
```

CUSTOMER_NAME	ACCNO
AVINASH	1
DINESH	2
NIKHIL	4
RAVI	5
AVINASH	8
NIKHIL	9
DINESH	10
NIKHIL	11
NULL	NULL

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

SELECT CUSTOMER_NAME, COUNT(CUSTOMER_NAME)
FROM DEPOSITER D, BANK_ACCOUNT B
WHERE D.ACCNO = B.ACCNO
AND B.BRANCH_NAME = 'SBI_RESIDENCYROAD'
GROUP BY CUSTOMER_NAME
HAVING COUNT(CUSTOMER_NAME) >= 2;



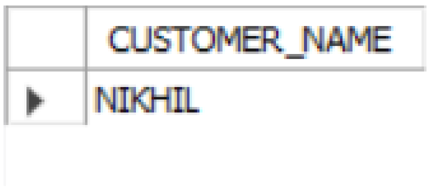
```
SELECT CUSTOMER_NAME, COUNT(CUSTOMER_NAME)
FROM DEPOSITER D, BANK_ACCOUNT B
WHERE D.ACCNO = B.ACCNO
AND B.BRANCH_NAME = 'SBI_RESIDENCYROAD'
GROUP BY CUSTOMER_NAME
HAVING COUNT(CUSTOMER_NAME) >= 2;
```

CUSTOMER_NAME	COUNT(CUSTOMER_NAME)
DINESH	2

iv. Find all the customers who have an account at all the branches located in a specific city (Ex. Delhi).

SELECT D.CUSTOMER_NAME
FROM DEPOSITER D,BRANCH B,BANK_ACCOUNT A
WHERE B.BRANCH_NAME=A.BRANCH_NAME
AND A.ACCNO=D.ACCNO
AND BRANCH_CITY='DELHI'

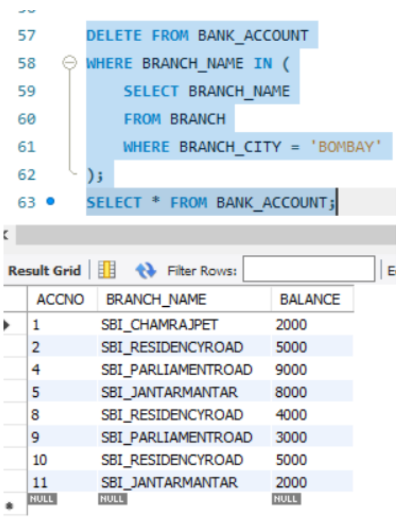
```
GROUP BY D.CUSTOMER_NAME
 HAVING COUNT(DISTINCT B.BRANCH_NAME)=(
               SELECT COUNT(BRANCH_NAME)
               FROM BRANCH
               WHERE BRANCH_CITY='DELHI');
```



CUSTOMER_NAME
NIKHIL

v. Demonstrate how you delete all account tuples at every branch located in a specific city (Ex. Bombay).

```
DELETE FROM BANK_ACCOUNT
WHERE BRANCH_NAME IN (
         SELECT BRANCH_NAME
    FROM BRANCH
    WHERE BRANCH_CITY = 'BOMBAY'
);
SELECT * FROM BANK_ACCOUNT;
```



```
DELETE FROM BANK_ACCOUNT
WHERE BRANCH_NAME IN (
    SELECT BRANCH_NAME
    FROM BRANCH
    WHERE BRANCH_CITY = 'BOMBAY'
);
SELECT * FROM BANK_ACCOUNT;
```

ACCNO	BRANCH_NAME	BALANCE
1	SBI_CHAMRAJPET	2000
2	SBI_RESIDENCYROAD	5000
4	SBI_PARLIAMENTROAD	9000
5	SBI_JANTARMANTAR	8000
8	SBI_RESIDENCYROAD	4000
9	SBI_PARLIAMENTROAD	3000
10	SBI_RESIDENCYROAD	5000
11	SBI_JANTARMANTAR	2000
NULL	NULL	NULL

PROGRAM 3: SUPPLIER DATABASE

Consider the following schema:

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

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

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

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

Schema Diagram

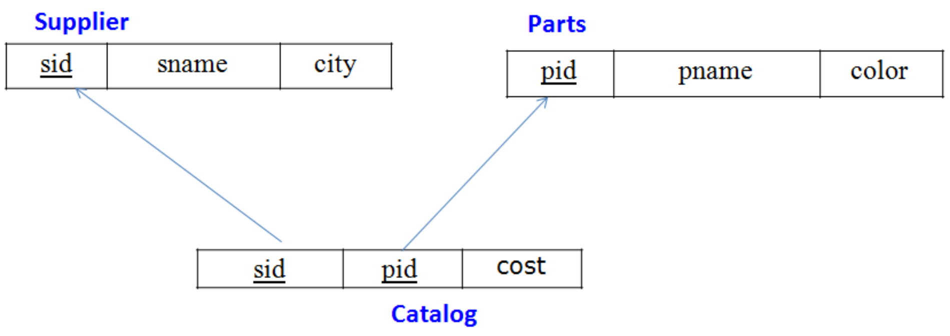
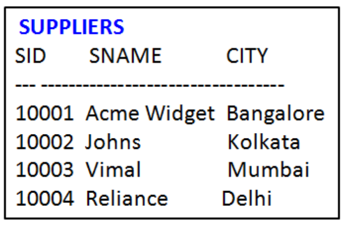
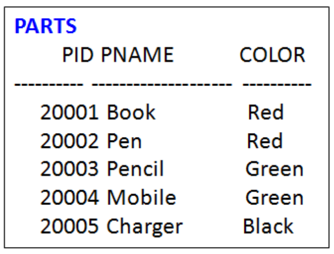


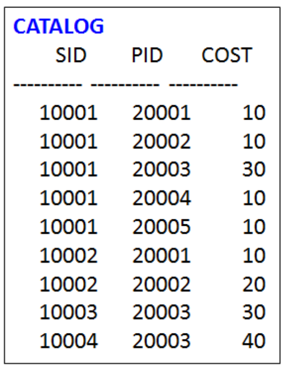
Table Data


SUPPLIERS

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


PARTS

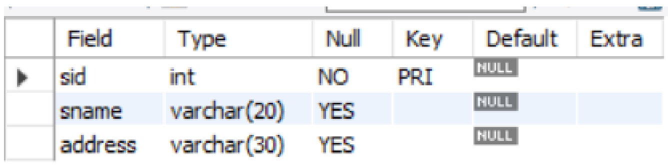
PID	PNAME	COLOR
20001	Book	Red
20002	Pen	Red
20003	Pencil	Green
20004	Mobile	Green
20005	Charger	Black


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

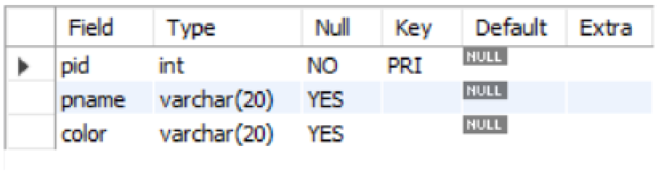
Creation of Tables:

```
CREATE TABLE suppliers (
        sid INT,
   sname VARCHAR(20),
   address VARCHAR(30),
   PRIMARY KEY (sid)
);
```



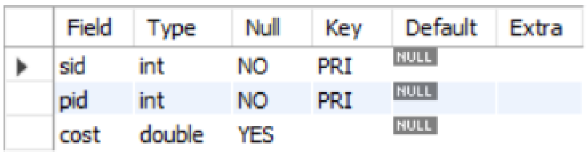
Field	Type	Null	Key	Default	Extra
sid	int	NO	PRI	NULL	
sname	varchar(20)	YES		NULL	
address	varchar(30)	YES		NULL	

```
CREATE TABLE parts (
        pid INT,
   pname VARCHAR(20),
   color VARCHAR(20),
   PRIMARY KEY (pid)
);
```



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

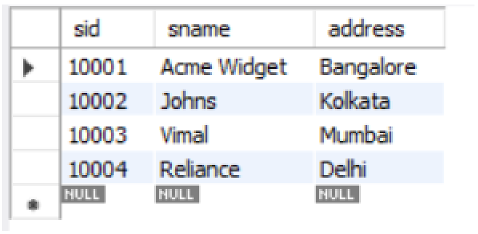
```
CREATE TABLE catalog (
   sid INT,
    pid INT,
    cost REAL,
    PRIMARY KEY(sid, pid),
    FOREIGN KEY (sid) REFERENCES suppliers(sid),
    FOREIGN KEY (pid) REFERENCES parts(pid)
);
```



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

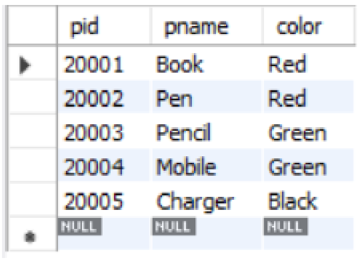
Inserting Values into the tables:

```
INSERT INTO suppliers VALUES (10001, 'Acme Widget', 'Bangalore');
INSERT INTO suppliers VALUES (10002, 'Johns', 'Kolkata');
INSERT INTO suppliers VALUES (10003, 'Vimal', 'Mumbai');
INSERT INTO suppliers VALUES (10004, 'Reliance', 'Delhi');
```



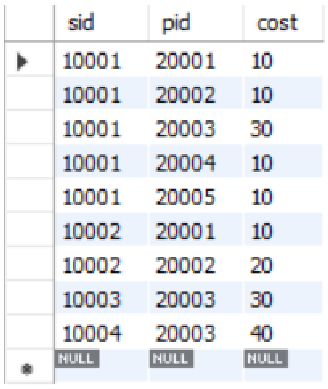
	sid	sname	address
▶	10001	Acme Widget	Bangalore
	10002	Johns	Kolkata
	10003	Vimal	Mumbai
	10004	Reliance	Delhi
*	NULL	NULL	NULL

```
INSERT INTO parts VALUES (20001, 'Book', 'Red');
INSERT INTO parts VALUES (20002, 'Pen', 'Red');
INSERT INTO parts VALUES (20003, 'Pencil', 'Green');
INSERT INTO parts VALUES (20004, 'Mobile', 'Green');
INSERT INTO parts VALUES (20005, 'Charger', 'Black');
```



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

```
INSERT INTO catalog VALUES (10001, 20001, 10);

INSERT INTO catalog VALUES (10001, 20002, 10);

INSERT INTO catalog VALUES (10001, 20003, 30);

INSERT INTO catalog VALUES (10001, 20004, 10);

INSERT INTO catalog VALUES (10001, 20005, 10);

INSERT INTO catalog VALUES (10002, 20001, 10);

INSERT INTO catalog VALUES (10002, 20002, 20);

INSERT INTO catalog VALUES (10003, 20003, 30);

INSERT INTO catalog VALUES (10004, 20003, 40);
```

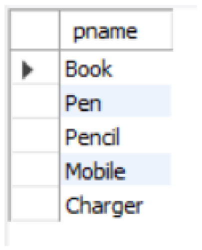


sid	pid	cost
10001	20001	10
10001	20002	10
10001	20003	30
10001	20004	10
10001	20005	10
10002	20001	10
10002	20002	20
10003	20003	30
10004	20003	40
NULL	NULL	NULL

Write the following queries in SQL:

1. **Find the pnames of parts for which there is some supplier.**

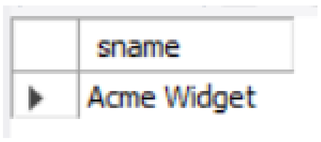
```
SELECT DISTINCT(pname)
FROM parts p, catalog c
WHERE p.pid = c.pid
AND c.sid IS NOT NULL;
```



pname
Book
Pen
Pencil
Mobile
Charger

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

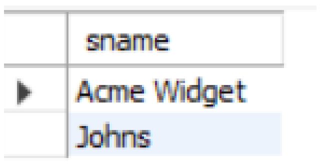
```
SELECT s.sname
FROM suppliers s
WHERE NOT EXISTS (
 SELECT p.pid
 FROM parts p
 WHERE NOT EXISTS (
 SELECT c.sid
 FROM catalog c
 WHERE c.sid = s.sid
 AND c.pid = p.pid
 )
);
```



sname
Acme Widget

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

```
SELECT s.sname
FROM suppliers s
WHERE NOT EXISTS (
 SELECT p.pid
 FROM parts p
 WHERE p.color = 'Red'
 AND NOT EXISTS (
 SELECT c.sid
 FROM catalog c
 WHERE c.sid = s.sid
 AND c.pid = p.pid
 )
);
```

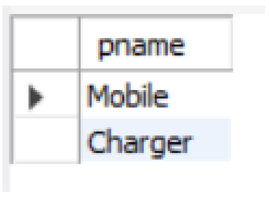


sname
Acme Widget
Johns

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

```
SELECT p.pname
FROM parts p, suppliers s, catalog c
WHERE c.sid = s.sid
```

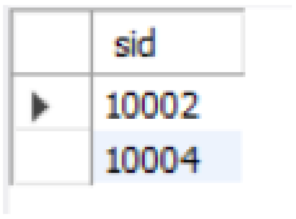
```
AND p.pid = c.pid
AND s.sname = 'Acme Widget'
AND NOT EXISTS (
SELECT c1.pid
FROM catalog c1, suppliers s1
WHERE c1.pid = p.pid
AND c1.sid = s1.sid
AND s1.sname <> 'Acme Widget'
);
```



	pname
▶	Mobile
	Charger

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

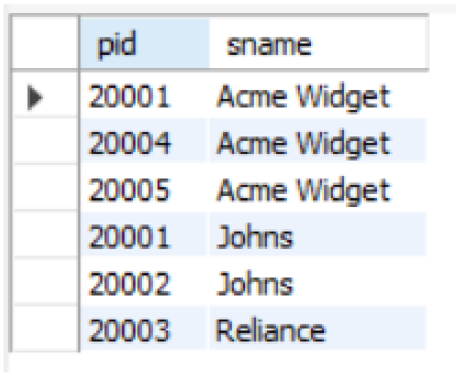
```
SELECT DISTINCT sid
FROM catalog c
WHERE c.cost > (
 SELECT AVG(c1.cost)
 FROM catalog c1
    WHERE c1.pid = c.pid
);
```



	sid
▶	10002
	10004

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

```
SELECT p.pid, s.sname
FROM parts p, suppliers s, catalog c
WHERE c.pid = p.pid
AND c.sid = s.sid
AND c.cost = (
 SELECT MAX(c1.cost)
 FROM catalog c1
 WHERE c1.pid = p.pid
);
```



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

PROGRAM 4: STUDENT FACULTY DATABASE

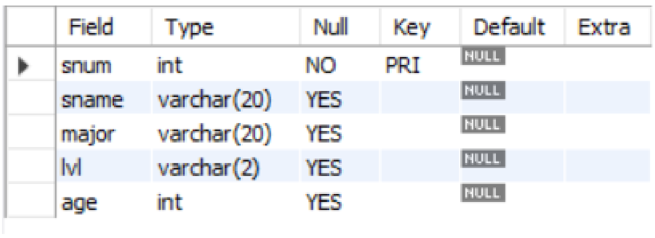
Consider the following database for student enrolment for course:
STUDENT(snum: integer, sname: string, major: string, lvl: string, age: integer)
CLASS(cname: string, meetsat: time, room: string, fid: integer)
ENROLLED(snum: integer, cname: string)
FACULTY(fid: integer, fname: string, deptid: integer)

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.

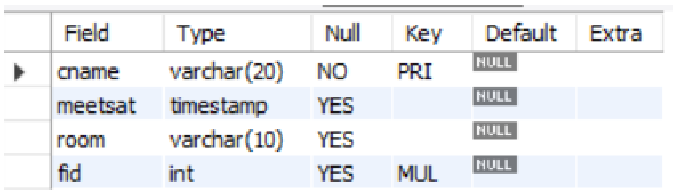
Creation of Tables:

```
CREATE TABLE student (
        snum int,
   sname varchar(20),
   major varchar(20),
   lvl varchar(2),
   age int,
   primary key(snum)
);
```



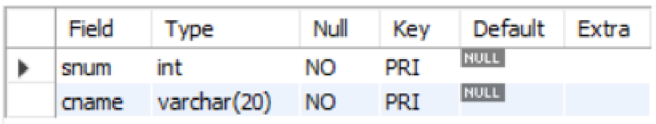
Field	Type	Null	Key	Default	Extra
snum	int	NO	PRI	NULL	
sname	varchar(20)	YES		NULL	
major	varchar(20)	YES		NULL	
lvl	varchar(2)	YES		NULL	
age	int	YES		NULL	

```
CREATE TABLE class (
        cname varchar(20),
   meetsat timestamp,
   room varchar(10),
   fid int,
   primary key(cname),
   foreign key(fid) references faculty(fid)
);
```



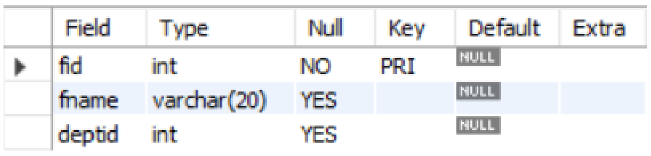
Field	Type	Null	Key	Default	Extra
cname	varchar(20)	NO	PRI	NULL	
meetsat	timestamp	YES		NULL	
room	varchar(10)	YES		NULL	
fid	int	YES	MUL	NULL	

```
CREATE TABLE enrolled (
        snum int,
   cname varchar(20),
   primary key(snum, cname),
   foreign key (snum) references student(snum),
   foreign key (cname) references class(cname)
);
```



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

```
CREATE TABLE faculty (
        fid int,
   fname varchar(20),
   deptid int,
   primary key(fid)
);
```



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

Inserting Values into the tables:

INSERT INTO student VALUES (1, 'John', 'CS', 'Sr', 19);

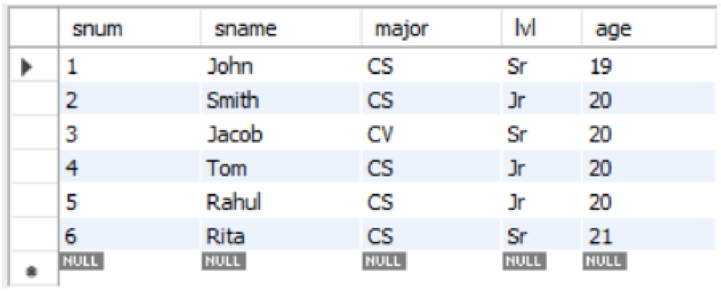
INSERT INTO student VALUES (2, 'Smith', 'CS', 'Jr', 20);

INSERT INTO student VALUES (3, 'Jacob', 'CV', 'Sr', 20);

INSERT INTO student VALUES (4, 'Tom', 'CS', 'Jr', 20);

INSERT INTO student VALUES (5, 'Rahul', 'CS', 'Jr', 20);

INSERT INTO student VALUES (6, 'Rita', 'CS', 'Sr', 21);



	snum	sname	major	lvl	age
▶	1	John	CS	Sr	19
	2	Smith	CS	Jr	20
	3	Jacob	CV	Sr	20
	4	Tom	CS	Jr	20
	5	Rahul	CS	Jr	20
	6	Rita	CS	Sr	21
*	NULL	NULL	NULL	NULL	NULL

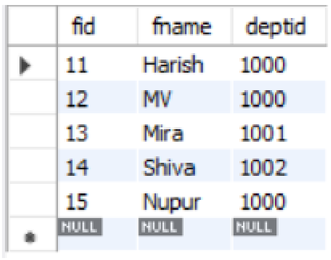
INSERT INTO faculty VALUES(11, 'Harish', 1000);

INSERT INTO faculty VALUES(12, 'MV', 1000);

INSERT INTO faculty VALUES(13, 'Mira', 1001);

INSERT INTO faculty VALUES(14, 'Shiva', 1002);

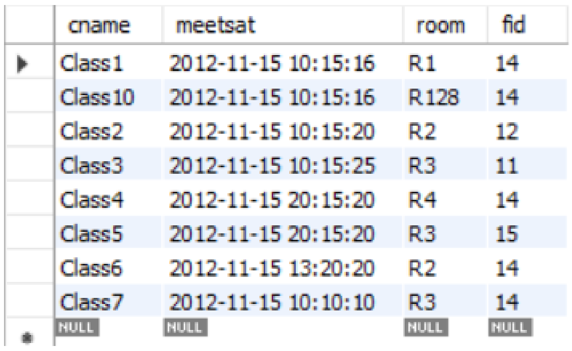
INSERT INTO faculty VALUES(15, 'Nupur', 1000);



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

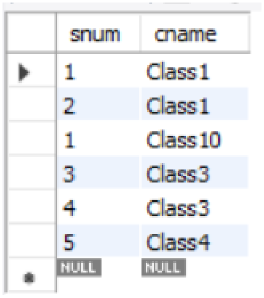
INSERT INTO class VALUES ('Class1', '12/11/15 10:15:16.00000', 'R1', 14);

```
INSERT INTO class VALUES ('Class10', '12/11/15 10:15:16.00000', 'R128', 14);
INSERT INTO class VALUES ('Class2', '12/11/15 10:15:20.000000', 'R2', 12);
INSERT INTO class VALUES ('Class3', '12/11/15 10:15:25.000000', 'R3', 11);
INSERT INTO class VALUES ('Class4', '12/11/15 20:15:20.000000', 'R4', 14);
INSERT INTO class VALUES ('Class5', '12/11/15 20:15:20.000000', 'R3', 15);
INSERT INTO class VALUES ('Class6', '12/11/15 13:20:20.000000', 'R2', 14);
INSERT INTO class VALUES ('Class7', '12/11/15 10:10:10.000000', 'R3', 14);
```



cname	meetsat	room	fid
Class1	2012-11-15 10:15:16	R1	14
Class10	2012-11-15 10:15:16	R128	14
Class2	2012-11-15 10:15:20	R2	12
Class3	2012-11-15 10:15:25	R3	11
Class4	2012-11-15 20:15:20	R4	14
Class5	2012-11-15 20:15:20	R3	15
Class6	2012-11-15 13:20:20	R2	14
Class7	2012-11-15 10:10:10	R3	14
NULL	NULL	NULL	NULL

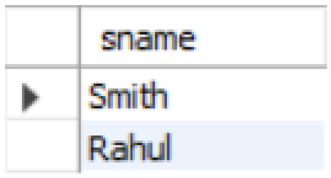
```
INSERT INTO enrolled VALUES (1, 'Class1');
INSERT INTO enrolled VALUES (2, 'Class1');
INSERT INTO enrolled VALUES (3, 'Class3');
INSERT INTO enrolled VALUES (4, 'Class3');
INSERT INTO enrolled VALUES (5, 'Class4');
```



snum	cname
1	Class1
2	Class1
1	Class10
3	Class3
4	Class3
5	Class4
NULL	NULL

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

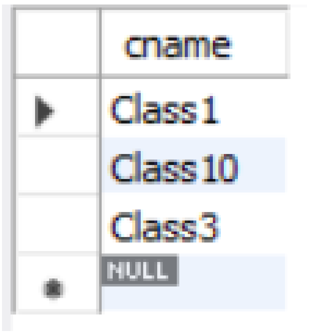
```
SELECT s.sname
FROM student s, enrolled e, class c, faculty f
WHERE s.lvl = 'Jr'
AND s.snum = e.snum
AND c.cname = e.cname
AND c.fid = f.fid
AND f.fname = 'Shiva';
```



```
sname
Smith
Rahul
```

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

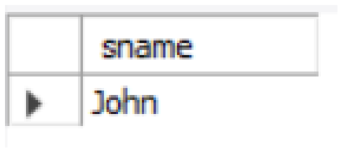
```
SELECT c.cname
FROM class c
WHERE c.room = 'R128'
OR c.cname
IN (
SELECT e.cname
FROM enrolled e
GROUP BY e.cname
HAVING COUNT(e.cname) >= 2
);
```



```
cname
Class1
Class10
Class3
NULL
```

3. **Find the names of all students who are enrolled in two classes that meet at the same time.**

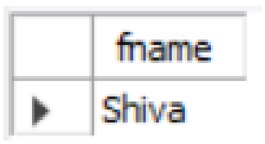
```
SELECT s.sname
FROM student s
WHERE s.snum IN (
SELECT e1.snum
FROM enrolled e1, enrolled e2, class c1, class c2
WHERE e1.snum = e2.snum
AND e1.cname <> e2.cname
AND e1.cname = c1.cname
AND c1.meetsat = c2.meetsat
);
```



```
  sname
▶ John
```

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

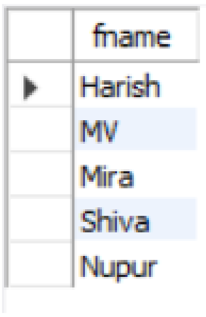
```
SELECT DISTINCT f.fname
FROM faculty f, class c
WHERE f.fid
IN (
SELECT fid
FROM class c
GROUP BY fid
HAVING COUNT(*) = (
SELECT COUNT(DISTINCT room)
FROM class
)
);
```



```
  fname
▶ Shiva
```

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

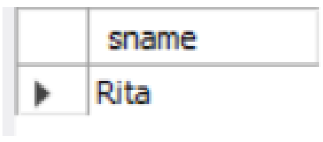
```
SELECT f.fname
FROM faculty f
WHERE 5 > (
SELECT COUNT(e.snum)
FROM class c, enrolled e
WHERE c.cname = e.cname
AND c.fid = f.fid
);
```



	fname
▶	Harish
	MV
	Mira
	Shiva
	Nupur

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

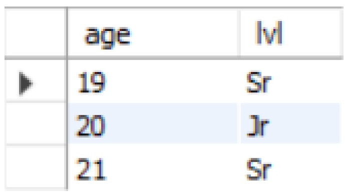
```
SELECT sname
FROM student
WHERE snum NOT IN (
SELECT e.snum
FROM enrolled e
);
```



	sname
▶	Rita

7. **For each age value that appears in Students, find the level value that appears most often. For example, if there are more FR level students aged 18 than SR, JR, or SO students aged 18, you should print the pair (18, FR).**

```
SELECT s.age, s.lvl
FROM student s
GROUP BY s.age, s.lvl
HAVING s.lvl IN (
        SELECT s1.lvl FROM student s1
        WHERE s1.age = s.age
        GROUP BY s1.lvl, s1.age
        HAVING COUNT(*) >= ALL (
                SELECT COUNT(*)
                FROM Student s2
                WHERE s1.age = s2.age
                GROUP BY s2.lvl, s2.age
   )
);
```



age	lvl
19	Sr
20	Jr
21	Sr

PROGRAM 5: AIRLINE FLIGHT DATABASE

Consider the following database that keeps track of airline flight information:

FLIGHTS(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)

AIRCRAFT(aid: integer, aname: string, cruisingrange: integer)

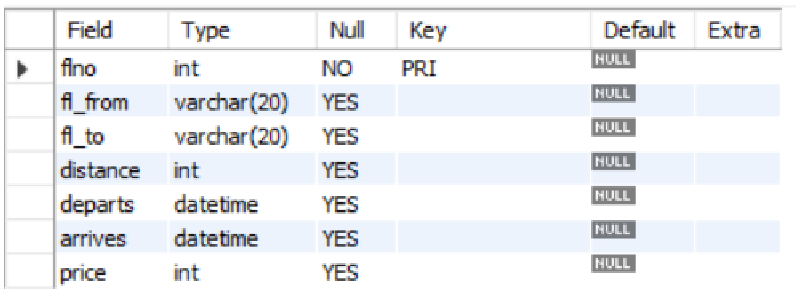
CERTIFIED(eid: integer, aid: integer)

EMPLOYEES(eid: integer, ename: string, salary: integer)

Note that the Employees relation describes pilots and other kinds of employees as well; Every pilot is certified for some aircraft, and only pilots are certified to fly. Write each of the following queries in SQL.

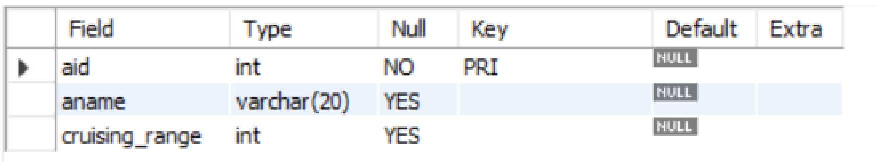
Creation of Tables:

```
CREATE TABLE flights(
        flno INT,
   fl_from VARCHAR(20),
   fl_to VARCHAR(20),
   distance INT,
   departs DATETIME,
   arrives DATETIME,
   price INT,
   PRIMARY KEY(flno)
);
```



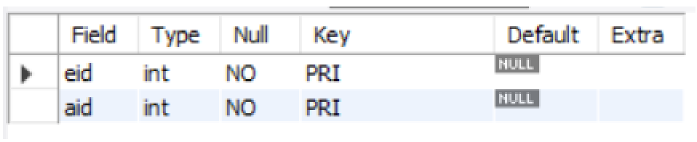
	Field	Type	Null	Key	Default	Extra
▶	flno	int	NO	PRI	NULL	
	fl_from	varchar(20)	YES		NULL	
	fl_to	varchar(20)	YES		NULL	
	distance	int	YES		NULL	
	departs	datetime	YES		NULL	
	arrives	datetime	YES		NULL	
	price	int	YES		NULL	

```
CREATE TABLE aircraft (
        aid INT,
   aname VARCHAR(20),
   cruising_range INT,
   PRIMARY KEY(aid)
);
```



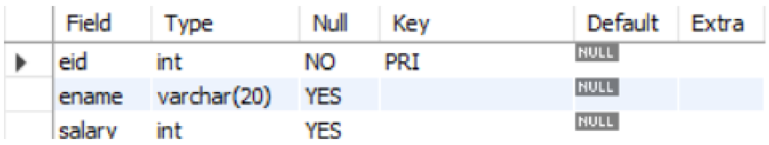
	Field	Type	Null	Key	Default	Extra
▶	aid	int	NO	PRI	NULL	
	aname	varchar(20)	YES		NULL	
	cruising_range	int	YES		NULL	

```
CREATE TABLE certified (
        eid INT,
   aid INT,
   PRIMARY KEY(eid, aid),
   FOREIGN KEY (eid) REFERENCES employees(eid),
   FOREIGN KEY(aid) REFERENCES aircraft(aid)
);
```



	Field	Type	Null	Key	Default	Extra
▶	eid	int	NO	PRI	NULL	
	aid	int	NO	PRI	NULL	

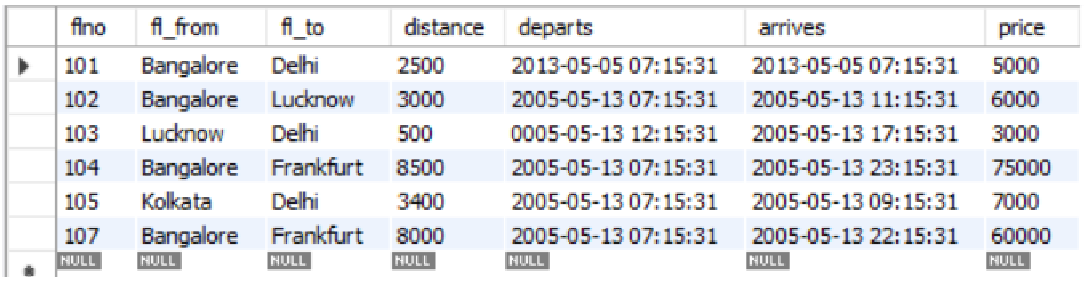
```
CREATE TABLE employees (
        eid INT,
   ename VARCHAR(20),
   salary INT,
   PRIMARY KEY(eid)
);
```



Field	Type	Null	Key	Default	Extra
eid	int	NO	PRI	NULL	
ename	varchar(20)	YES		NULL	
salary	int	YES		NULL	

Inserting Values into the tables:

```
INSERT INTO flights VALUES (101, 'Bangalore', 'Delhi', 2500, '13-05-05
07.15.31.000000', '13-05-05 07.15.31.000000', 5000);
INSERT INTO flights VALUES (102, 'Bangalore', 'Lucknow', 3000, '05/05/13
07:15:31', '05/05/13 11:15:31', 6000);
INSERT INTO flights VALUES (103, 'Lucknow', 'Delhi', 500, '5/05/13 12:15:31',
'05/05/13 17:15:31', 3000);
INSERT INTO flights VALUES (107, 'Bangalore', 'Frankfurt', 8000, '05/05/13
07:15:31', '05/05/13 22:15:31', 60000);
INSERT INTO flights VALUES (104, 'Bangalore', 'Frankfurt', 8500, '05/05/13
07:15:31', '05/05/13 23:15:31', 75000);
INSERT INTO flights VALUES (105, 'Kolkata', 'Delhi', 3400, '05/05/13 07:15:31', '
05/05/13 09:15:31', 7000);
```



fno	fl_from	fl_to	distance	departs	arrives	price
101	Bangalore	Delhi	2500	2013-05-05 07:15:31	2013-05-05 07:15:31	5000
102	Bangalore	Lucknow	3000	2005-05-13 07:15:31	2005-05-13 11:15:31	6000
103	Lucknow	Delhi	500	0005-05-13 12:15:31	2005-05-13 17:15:31	3000
104	Bangalore	Frankfurt	8500	2005-05-13 07:15:31	2005-05-13 23:15:31	75000
105	Kolkata	Delhi	3400	2005-05-13 07:15:31	2005-05-13 09:15:31	7000
107	Bangalore	Frankfurt	8000	2005-05-13 07:15:31	2005-05-13 22:15:31	60000
NULL	NULL	NULL	NULL	NULL	NULL	NULL

INSERT INTO aircraft VALUES (101, '747', 3000);

INSERT INTO aircraft VALUES (102, 'Boeing', 900);

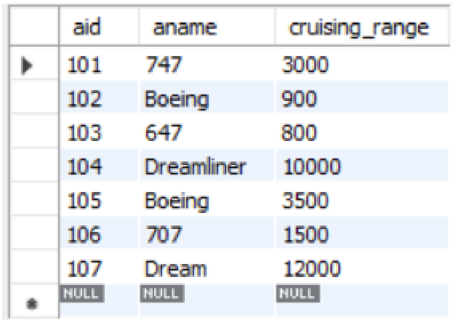
INSERT INTO aircraft VALUES (103, '647', 800);

INSERT INTO aircraft VALUES (104, 'Dreamliner', 10000);

INSERT INTO aircraft VALUES (105, 'Boeing', 3500);

INSERT INTO aircraft VALUES (106, '707', 1500);

INSERT INTO aircraft VALUES (107, 'Dream', 12000);



aid	aname	cruising_range
101	747	3000
102	Boeing	900
103	647	800
104	Dreamliner	10000
105	Boeing	3500
106	707	1500
107	Dream	12000
NULL	NULL	NULL

INSERT INTO certified VALUES (701, 101);

INSERT INTO certified VALUES (701, 102);

INSERT INTO certified VALUES (701, 106);

INSERT INTO certified VALUES (701, 105);

INSERT INTO certified VALUES (702, 104);

INSERT INTO certified VALUES (703, 104);

INSERT INTO certified VALUES (704, 104);

INSERT INTO certified VALUES (702, 107);

INSERT INTO certified VALUES (703, 107);

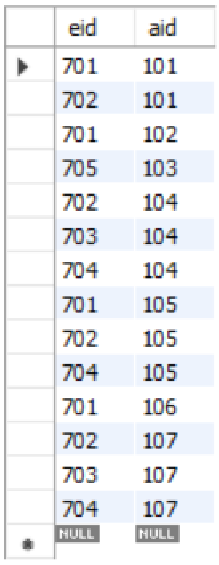
INSERT INTO certified VALUES (704, 107);

INSERT INTO certified VALUES (702, 101);

INSERT INTO certified VALUES (702, 105);

INSERT INTO certified VALUES (704, 105);

INSERT INTO certified VALUES (705, 103);



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

INSERT INTO employees VALUES (701, 'A', 50000);

INSERT INTO employees VALUES (702, 'B', 100000);

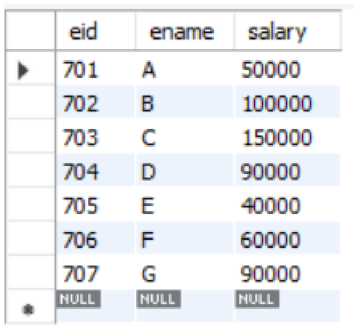
INSERT INTO employees VALUES (703, 'C', 150000);

INSERT INTO employees VALUES (704, 'D', 90000);

INSERT INTO employees VALUES (705, 'E', 40000);

INSERT INTO employees VALUES (706, 'F', 60000);

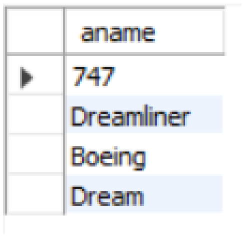
INSERT INTO employees VALUES (707, 'G', 90000);



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

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

```
SELECT DISTINCT a.aname
FROM aircraft a, certified c, employees e
WHERE a.aid = c.aid
AND c.eid = e.eid
AND e.salary>80000;
```

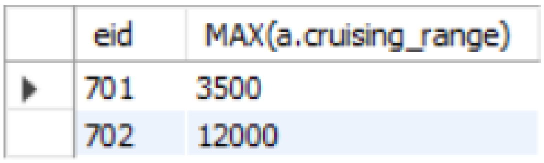


aname
747
Dreamliner
Boeing
Dream

2. For each pilot who is certified for more than three aircrafts, find the eid and the maximum cruisingrange of the aircraft for which she or he is certified.

```
SELECT c.eid, MAX(a.cruising_range)
FROM aircraft a, certified c, employees e
WHERE e.eid = c.eid
AND a.aid = c.aid
```

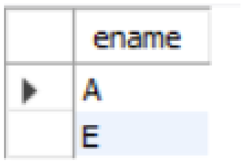
```
GROUP BY c.eid
HAVING COUNT(*) > 3;
```



```
eid   MAX(a.cruising_range)
701   3500
702   12000
```

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

```
SELECT e.ename
FROM employees e
WHERE e.salary < (
 SELECT MIN(f.price)
 FROM flights f
 WHERE f.fl_from = 'Bangalore'
 AND f.fl_to = 'Frankfurt'
);
```



```
ename
A
E
```

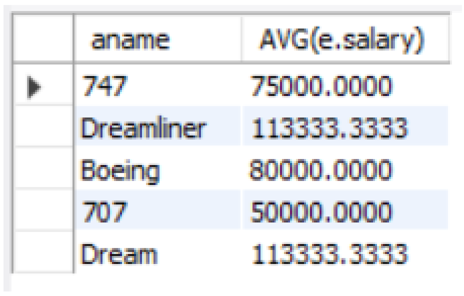
4. For all aircraft with cruisingrange over 1000 Kms, find the name of the aircraft and the average salary of all pilots certified for this aircraft.

```
SELECT a.aname, AVG(e.salary)
FROM aircraft a, certified c, employees e
WHERE a.cruising_range > 1000
```

AND a.aid = c.aid

AND e.eid = c.eid

GROUP BY a.aname;



aname	AVG(e.salary)
747	75000.0000
Dreamliner	113333.3333
Boeing	80000.0000
707	50000.0000
Dream	113333.3333

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

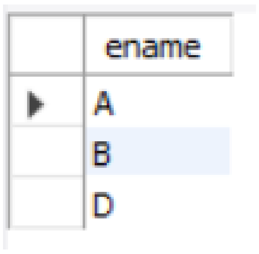
SELECT DISTINCT e.ename

FROM employees e, aircraft a, certified c

WHERE e.eid = c.eid

AND a.aid = c.aid

AND a.aname = 'Boeing';



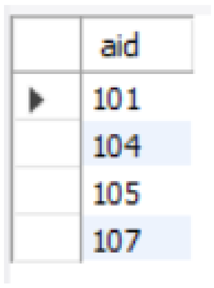
ename
A
B
D

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

SELECT a.aid

FROM aircraft a, flights f

```
WHERE a.cruising_range >= f.distance
AND f.fl_from = 'Bangalore'
AND f.fl_to = 'Delhi';
```



	aid
▶	101
	104
	105
	107

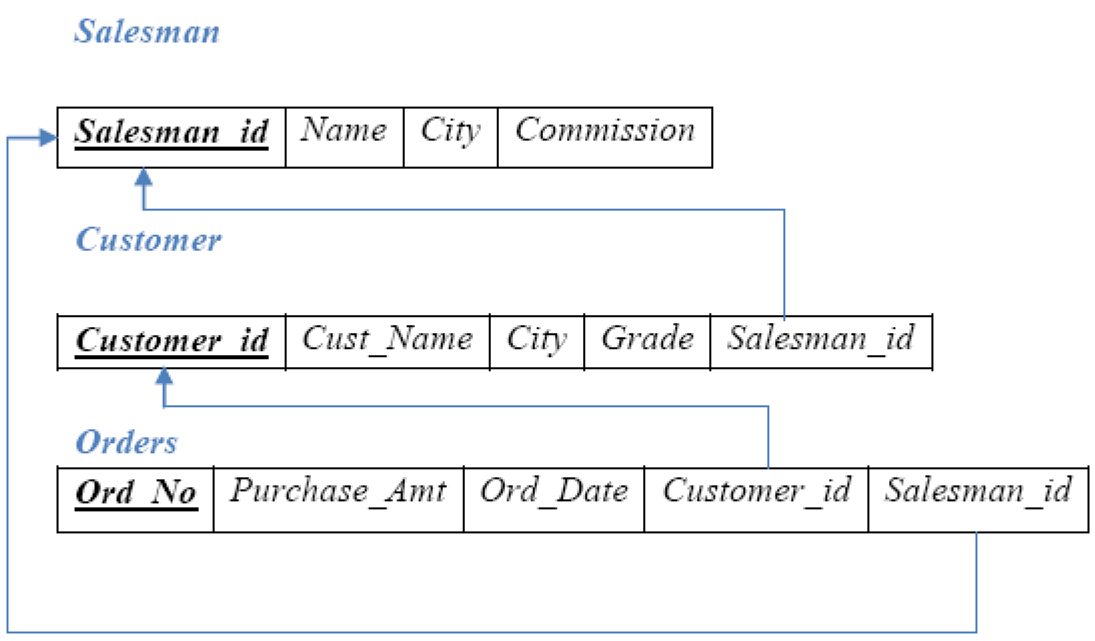
7. **A customer wants to travel from Bangalore to Kolkata New with no more than two changes of flight. List the choice of departure times from Madison if the customer wants to arrive in Kolkata by 6 p.m.**

PROGRAM 6 : ORDER DATABASE

Consider the following schema for Order Database:

SALESMAN (*Salesman_id, Name, City, Commission*)
CUSTOMER (*Customer_id, Cust_Name, City, Grade, Salesman_id*)
ORDERS (*Ord_No, Purchase_Amt, Ord_Date, Customer_id, Salesman_id*)

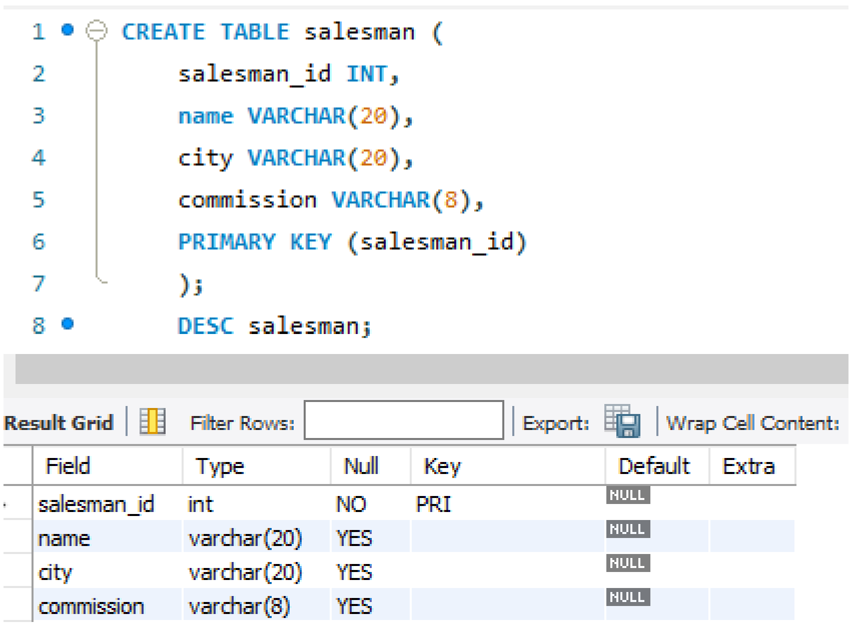
Schema Diagram



CREATION OF TABLES:

```
CREATE TABLE salesman (

        salesman_id INT,

   name VARCHAR(20),

   city VARCHAR(20),

   commission VARCHAR(8),

   PRIMARY KEY (salesman_id)
```

);

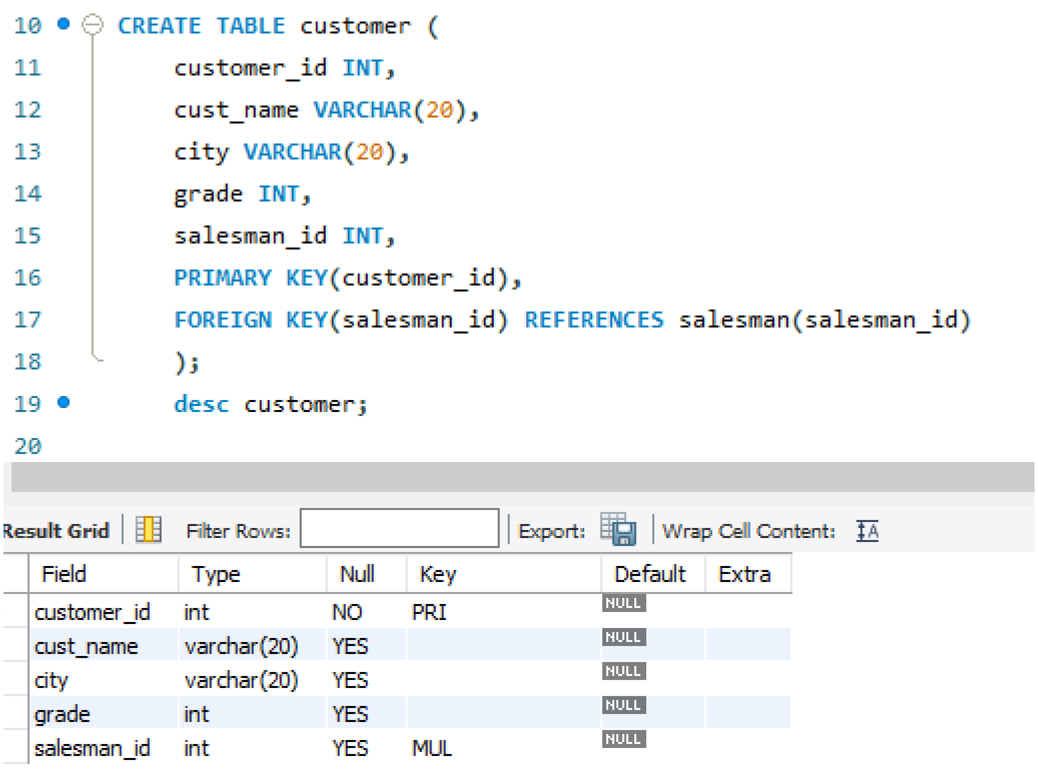


```
CREATE TABLE salesman (
        salesman_id INT,
        name VARCHAR(20),
        city VARCHAR(20),
        commission VARCHAR(8),
        PRIMARY KEY (salesman_id)
        );
        DESC salesman;
```

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

Field	Type	Null	Key	Default	Extra
salesman_id	int	NO	PRI	NULL	
name	varchar(20)	YES		NULL	
city	varchar(20)	YES		NULL	
commission	varchar(8)	YES		NULL	

```
CREATE TABLE customer (
        customer_id INT,
    cust_name VARCHAR(20),
    city VARCHAR(20),
    grade INT,
    salesman_id INT,
    PRIMARY KEY(customer_id),
    FOREIGN KEY(salesman_id) REFERENCES salesman(salesman_id)
    );
```



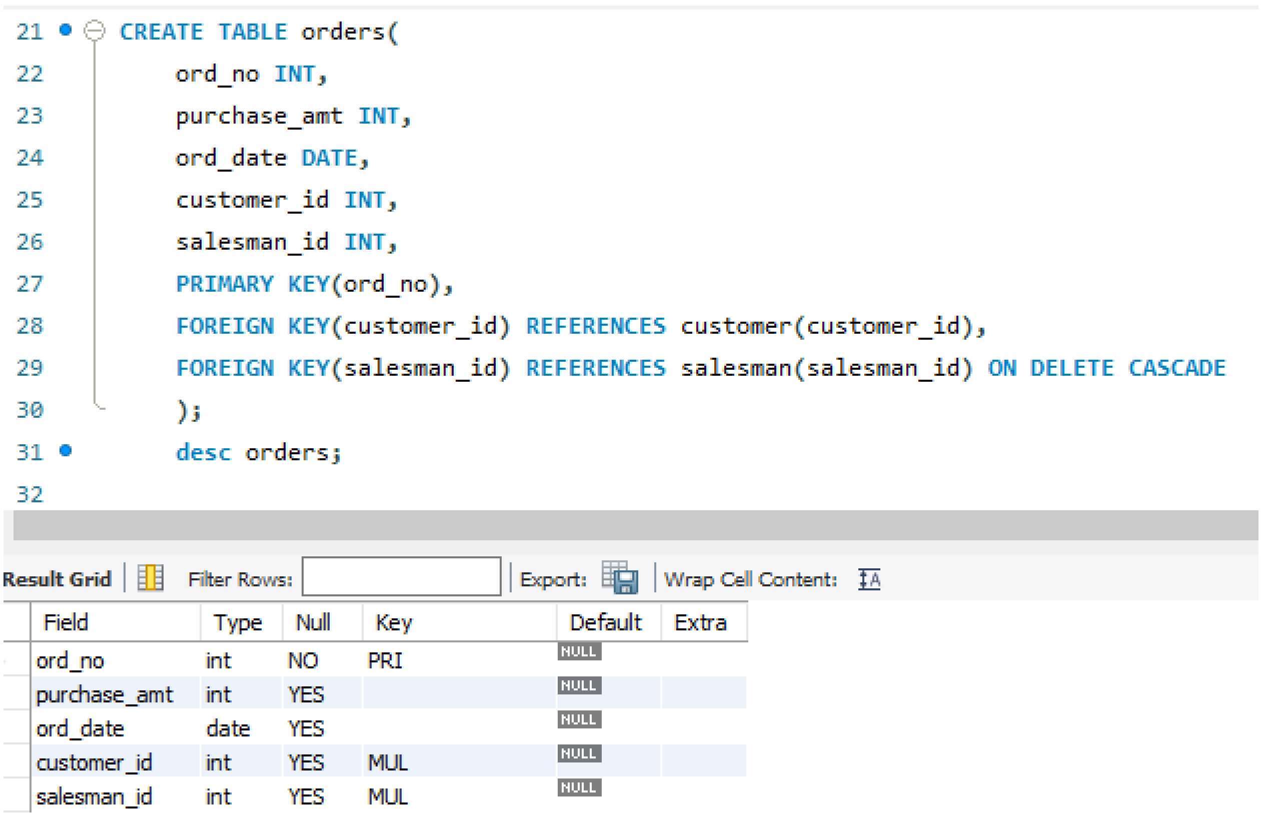
```
CREATE TABLE customer (
        customer_id INT,
        cust_name VARCHAR(20),
        city VARCHAR(20),
        grade INT,
        salesman_id INT,
        PRIMARY KEY(customer_id),
        FOREIGN KEY(salesman_id) REFERENCES salesman(salesman_id)
        );
        desc customer;
```

Field	Type	Null	Key	Default	Extra
customer_id	int	NO	PRI	NULL	
cust_name	varchar(20)	YES		NULL	
city	varchar(20)	YES		NULL	
grade	int	YES		NULL	
salesman_id	int	YES	MUL	NULL	

```
CREATE TABLE orders(
        ord_no INT,
   purchase_amt INT,
   ord_date DATE,
   customer_id INT,
   salesman_id INT,
   PRIMARY KEY(ord_no),
   FOREIGN KEY(customer_id) REFERENCES customer(customer_id),
```

FOREIGN KEY(salesman_id) REFERENCES salesman(salesman_id) ON DELETE CASCADE

);



```
CREATE TABLE orders(
      ord_no INT,
      purchase_amt INT,
      ord_date DATE,
      customer_id INT,
      salesman_id INT,
      PRIMARY KEY(ord_no),
      FOREIGN KEY(customer_id) REFERENCES customer(customer_id),
      FOREIGN KEY(salesman_id) REFERENCES salesman(salesman_id) ON DELETE CASCADE
      );
      desc orders;
```

Field	Type	Null	Key	Default	Extra
ord_no	int	NO	PRI	NULL	
purchase_amt	int	YES		NULL	
ord_date	date	YES		NULL	
customer_id	int	YES	MUL	NULL	
salesman_id	int	YES	MUL	NULL	

Inserting Values into the tables:

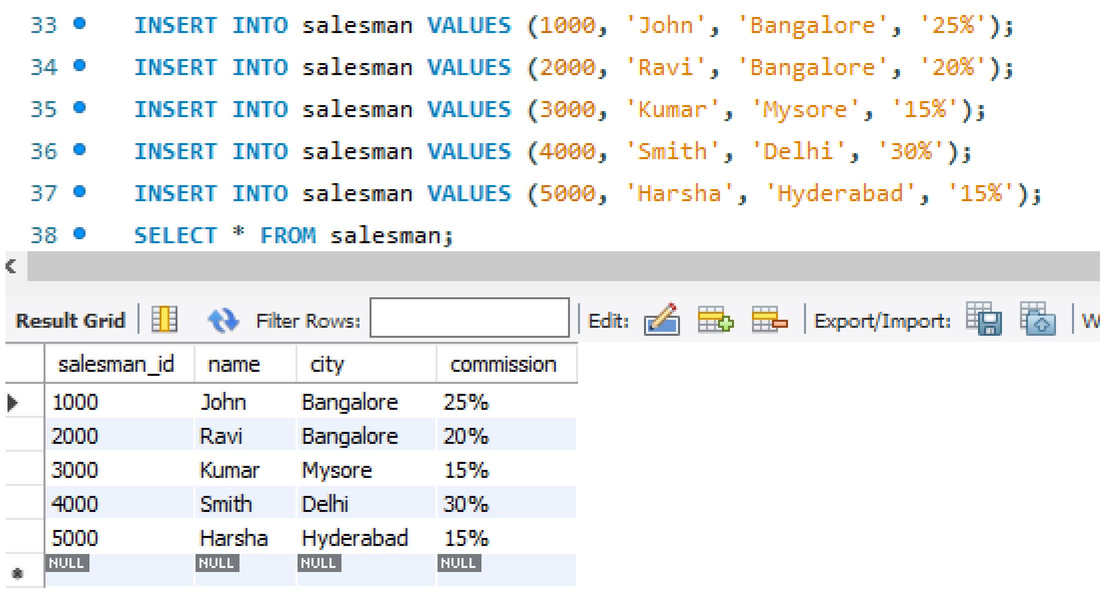
INSERT INTO salesman VALUES (1000, 'John', 'Bangalore', '25%');

INSERT INTO salesman VALUES (2000, 'Ravi', 'Bangalore', '20%');

INSERT INTO salesman VALUES (3000, 'Kumar', 'Mysore', '15%');

INSERT INTO salesman VALUES (4000, 'Smith', 'Delhi', '30%');

INSERT INTO salesman VALUES (5000, 'Harsha', 'Hyderabad', '15%');



```sql
INSERT INTO salesman VALUES (1000, 'John', 'Bangalore', '25%');
INSERT INTO salesman VALUES (2000, 'Ravi', 'Bangalore', '20%');
INSERT INTO salesman VALUES (3000, 'Kumar', 'Mysore', '15%');
INSERT INTO salesman VALUES (4000, 'Smith', 'Delhi', '30%');
INSERT INTO salesman VALUES (5000, 'Harsha', 'Hyderabad', '15%');
SELECT * FROM salesman;
```

salesman_id	name	city	commission
1000	John	Bangalore	25%
2000	Ravi	Bangalore	20%
3000	Kumar	Mysore	15%
4000	Smith	Delhi	30%
5000	Harsha	Hyderabad	15%
NULL	NULL	NULL	NULL

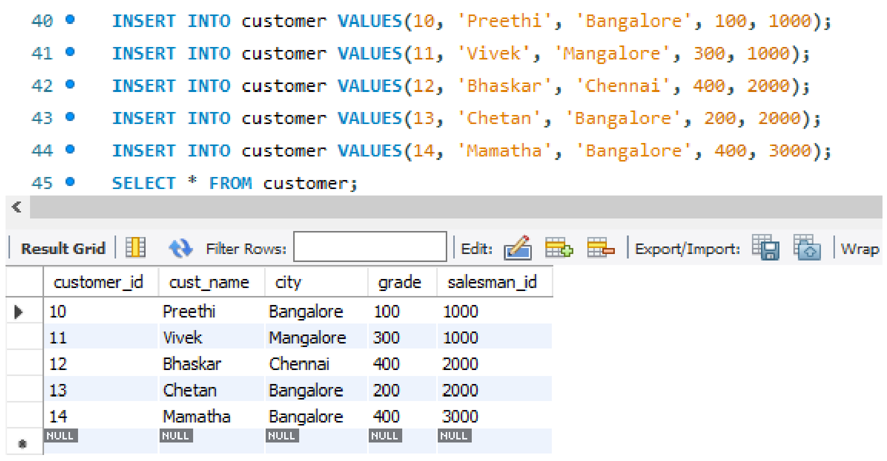
INSERT INTO customer VALUES(10, 'Preethi', 'Bangalore', 100, 1000);

INSERT INTO customer VALUES(11, 'Vivek', 'Mangalore', 300, 1000);

INSERT INTO customer VALUES(12, 'Bhaskar', 'Chennai', 400, 2000);

INSERT INTO customer VALUES(13, 'Chetan', 'Bangalore', 200, 2000);

INSERT INTO customer VALUES(14, 'Mamatha', 'Bangalore', 400, 3000);



```sql
INSERT INTO customer VALUES(10, 'Preethi', 'Bangalore', 100, 1000);
INSERT INTO customer VALUES(11, 'Vivek', 'Mangalore', 300, 1000);
INSERT INTO customer VALUES(12, 'Bhaskar', 'Chennai', 400, 2000);
INSERT INTO customer VALUES(13, 'Chetan', 'Bangalore', 200, 2000);
INSERT INTO customer VALUES(14, 'Mamatha', 'Bangalore', 400, 3000);
SELECT * FROM customer;
```

customer_id	cust_name	city	grade	salesman_id
10	Preethi	Bangalore	100	1000
11	Vivek	Mangalore	300	1000
12	Bhaskar	Chennai	400	2000
13	Chetan	Bangalore	200	2000
14	Mamatha	Bangalore	400	3000
NULL	NULL	NULL	NULL	NULL

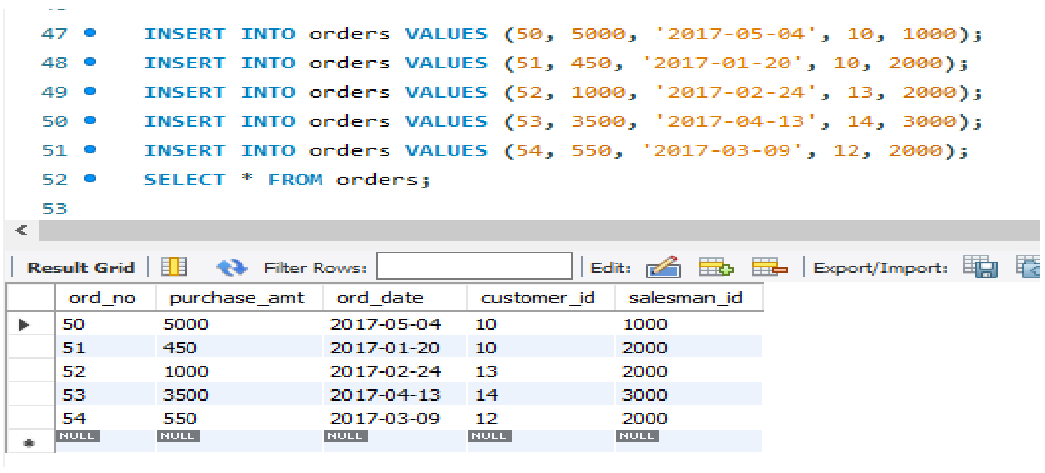
INSERT INTO orders VALUES (50, 5000, '2017-05-04', 10, 1000);

INSERT INTO orders VALUES (51, 450, '2017-01-20', 10, 2000);

INSERT INTO orders VALUES (52, 1000, '2017-02-24', 13, 2000);

INSERT INTO orders VALUES (53, 3500, '2017-04-13', 14, 3000);

INSERT INTO orders VALUES (54, 550, '2017-03-09', 12, 2000);

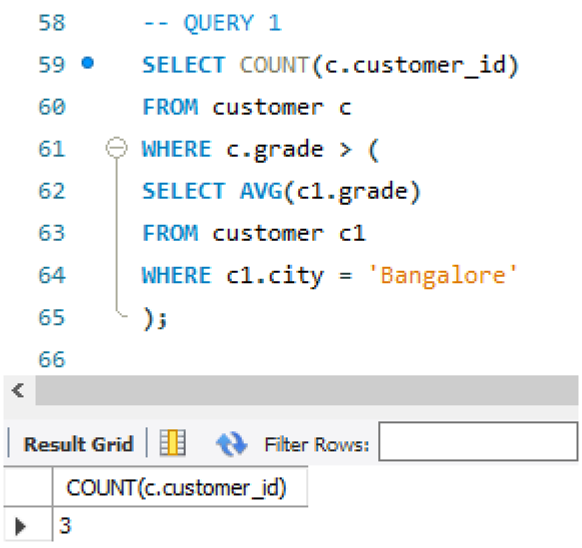


```
INSERT INTO orders VALUES (50, 5000, '2017-05-04', 10, 1000);
INSERT INTO orders VALUES (51, 450, '2017-01-20', 10, 2000);
INSERT INTO orders VALUES (52, 1000, '2017-02-24', 13, 2000);
INSERT INTO orders VALUES (53, 3500, '2017-04-13', 14, 3000);
INSERT INTO orders VALUES (54, 550, '2017-03-09', 12, 2000);
SELECT * FROM orders;
```

ord_no	purchase_amt	ord_date	customer_id	salesman_id
50	5000	2017-05-04	10	1000
51	450	2017-01-20	10	2000
52	1000	2017-02-24	13	2000
53	3500	2017-04-13	14	3000
54	550	2017-03-09	12	2000
NULL	NULL	NULL	NULL	NULL

1. Count the customers with grades above Bangalore's average.

SELECT COUNT(c.customer_id)
FROM customer c
WHERE c.grade > (
SELECT AVG(c1.grade)
FROM customer c1
WHERE c1.city = 'Bangalore'
);

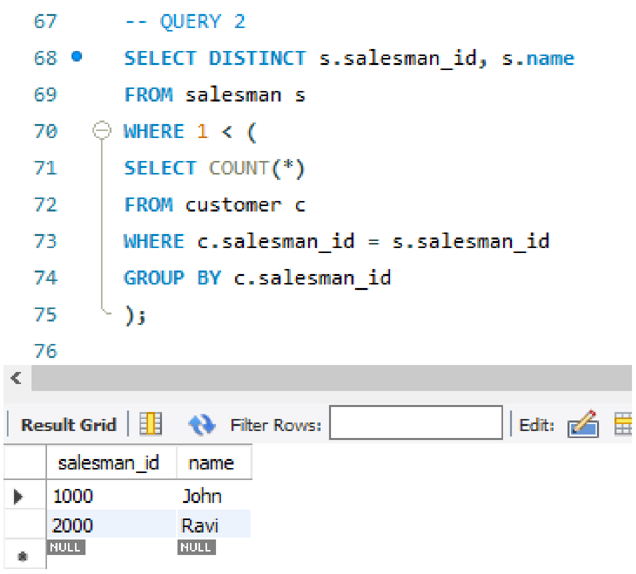


```
-- QUERY 1
SELECT COUNT(c.customer_id)
FROM customer c
WHERE c.grade > (
SELECT AVG(c1.grade)
FROM customer c1
WHERE c1.city = 'Bangalore'
);
```

```
COUNT(c.customer_id)
3
```

2. Find the name and numbers of all salesmen who had more than one customer.

SELECT DISTINCT s.salesman_id, s.name
FROM salesman s
WHERE 1 < (
SELECT COUNT(*)
FROM customer c
WHERE c.salesman_id = s.salesman_id
GROUP BY c.salesman_id
);

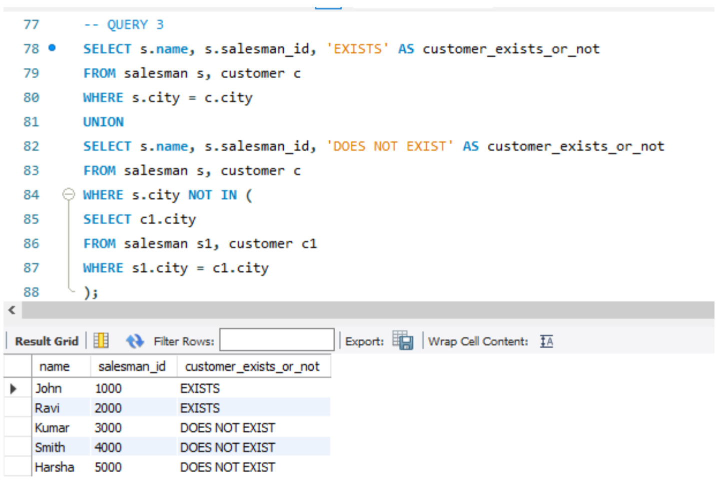


```
-- QUERY 2
SELECT DISTINCT s.salesman_id, s.name
FROM salesman s
WHERE 1 < (
SELECT COUNT(*)
FROM customer c
WHERE c.salesman_id = s.salesman_id
GROUP BY c.salesman_id
);
```

```
salesman_id  name
1000         John
2000         Ravi
NULL         NULL
```

3. List all salesmen and indicate those who have and don’t have customers in their cities (Use UNION operation.)

```
SELECT s.name, s.salesman_id, 'EXISTS' AS customer_exists_or_not
FROM salesman s, customer c
WHERE s.city = c.city
UNION
SELECT s.name, s.salesman_id, 'DOES NOT EXIST' AS customer_exists_or_not
FROM salesman s, customer c
WHERE s.city NOT IN (
SELECT c1.city
FROM salesman s1, customer c1
WHERE s1.city = c1.city
);
```

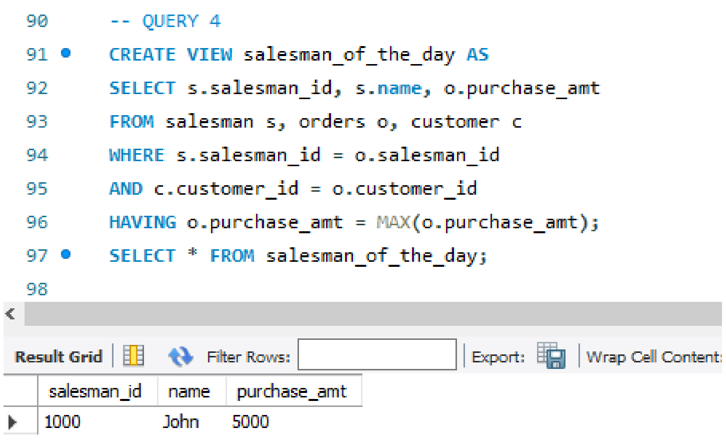


```
-- QUERY 3
SELECT s.name, s.salesman_id, 'EXISTS' AS customer_exists_or_not
FROM salesman s, customer c
WHERE s.city = c.city
UNION
SELECT s.name, s.salesman_id, 'DOES NOT EXIST' AS customer_exists_or_not
FROM salesman s, customer c
WHERE s.city NOT IN (
SELECT c1.city
FROM salesman s1, customer c1
WHERE s1.city = c1.city
);
```

```
name     salesman_id  customer_exists_or_not
John     1000         EXISTS
Ravi     2000         EXISTS
Kumar    3000         DOES NOT EXIST
Smith    4000         DOES NOT EXIST
Harsha   5000         DOES NOT EXIST
```

4. Create a view that finds the salesman who has the customer with the highest order of a day.

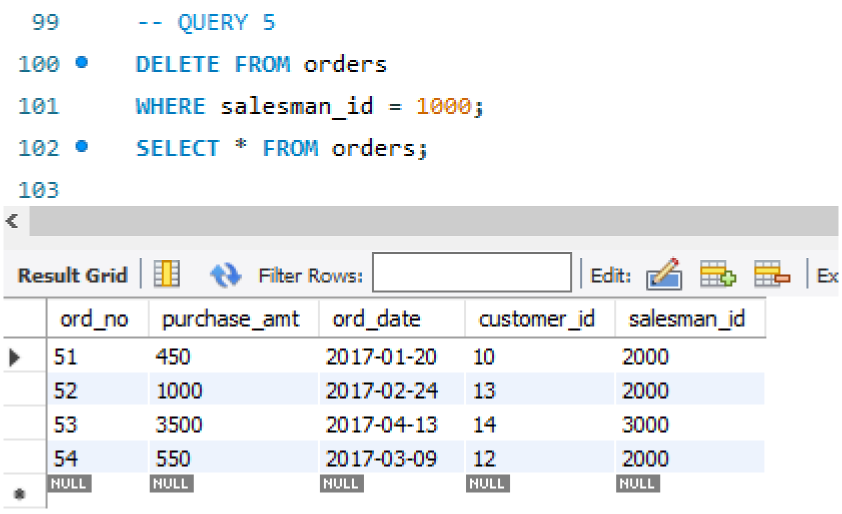
```
CREATE VIEW salesman_of_the_day AS
SELECT s.salesman_id, s.name, o.purchase_amt
FROM salesman s, orders o, customer c
WHERE s.salesman_id = o.salesman_id
AND c.customer_id = o.customer_id
HAVING o.purchase_amt = MAX(o.purchase_amt);

SELECT * FROM salesman_of_the_day;
```


```
-- QUERY 4
CREATE VIEW salesman_of_the_day AS
SELECT s.salesman_id, s.name, o.purchase_amt
FROM salesman s, orders o, customer c
WHERE s.salesman_id = o.salesman_id
AND c.customer_id = o.customer_id
HAVING o.purchase_amt = MAX(o.purchase_amt);
SELECT * FROM salesman_of_the_day;
```

salesman_id	name	purchase_amt
1000	John	5000

5. Demonstrate the DELETE operation by removing salesman with id 1000. All his orders must also be deleted.

DELETE FROM orders
WHERE salesman_id = 1000;

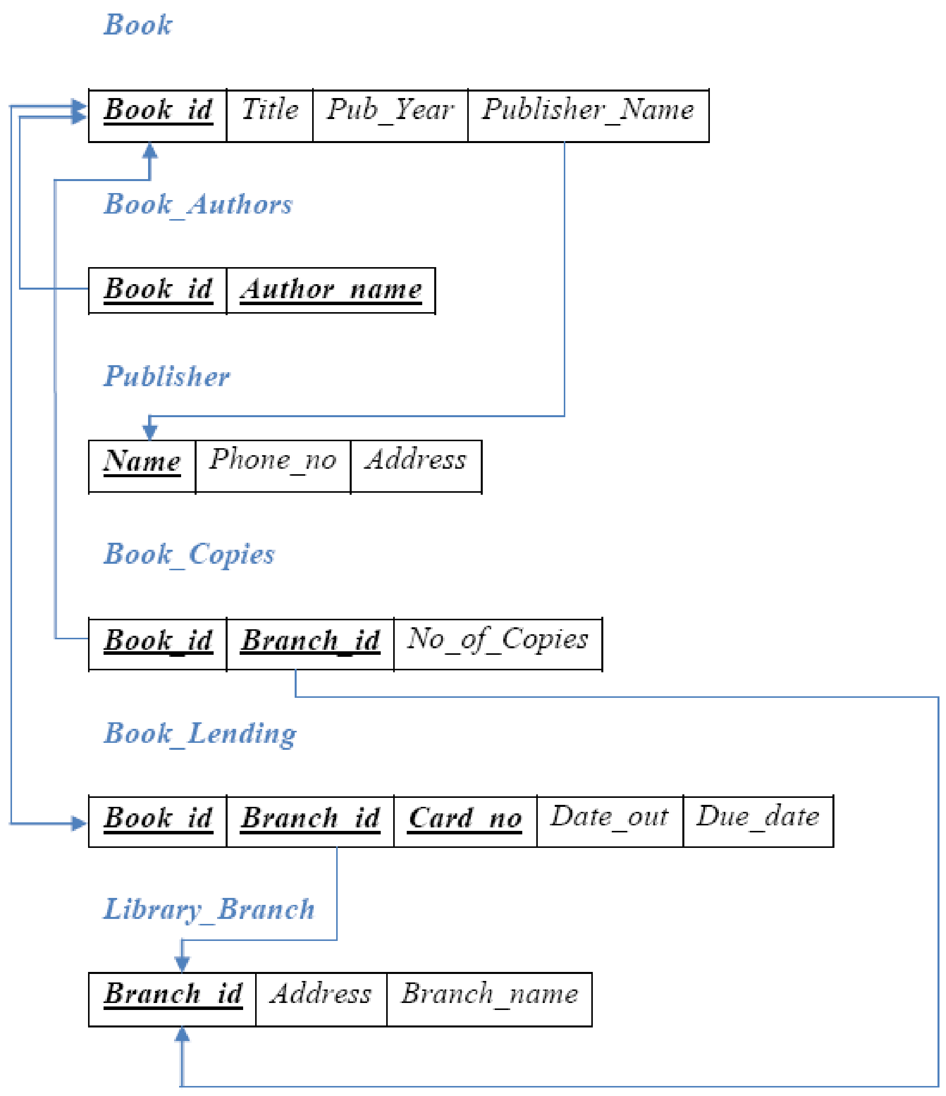

```
-- QUERY 5
DELETE FROM orders
WHERE salesman_id = 1000;
SELECT * FROM orders;
```

ord_no	purchase_amt	ord_date	customer_id	salesman_id
51	450	2017-01-20	10	2000
52	1000	2017-02-24	13	2000
53	3500	2017-04-13	14	3000
54	550	2017-03-09	12	2000
NULL	NULL	NULL	NULL	NULL

PROGRAM 7 : BOOK DATABASE

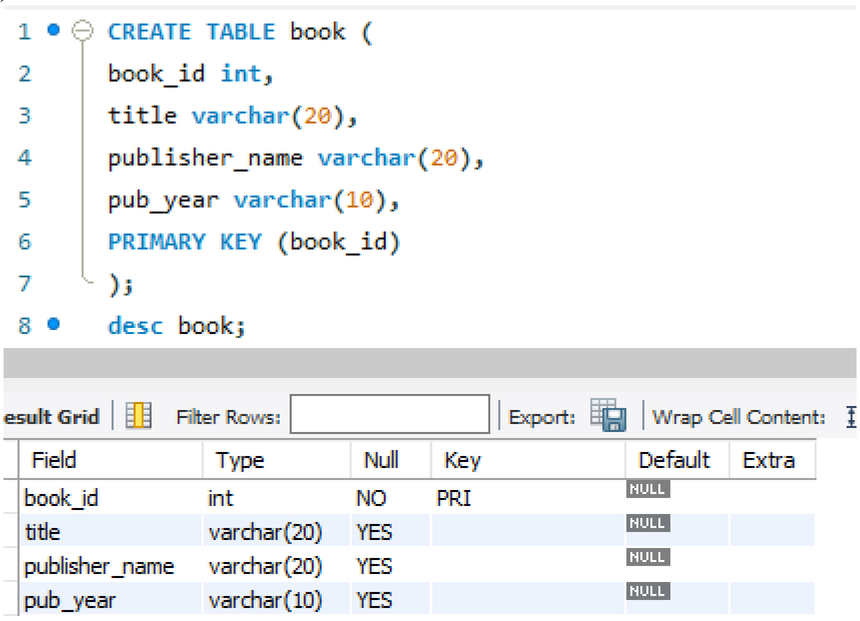
BOOK (Book_id, Title, Publisher_Name, Pub_Year)
BOOK_AUTHORS (Book_id, Author_Name)
PUBLISHER (Name, Address, Phone)
BOOK_COPIES (Book_id, Branch_id, No-of_Copies)
BOOK_LENDING (Book_id, Branch_id, Card_No, Date_Out, Due_Date)
LIBRARY_BRANCH (Branch_id, Branch_Name, Address)

Schema Diagram



CREATION OF TABLES:

```
CREATE TABLE book (
book_id int,
title varchar(20),
publisher_name varchar(20),
pub_year varchar(10),
PRIMARY KEY (book_id)
);
```

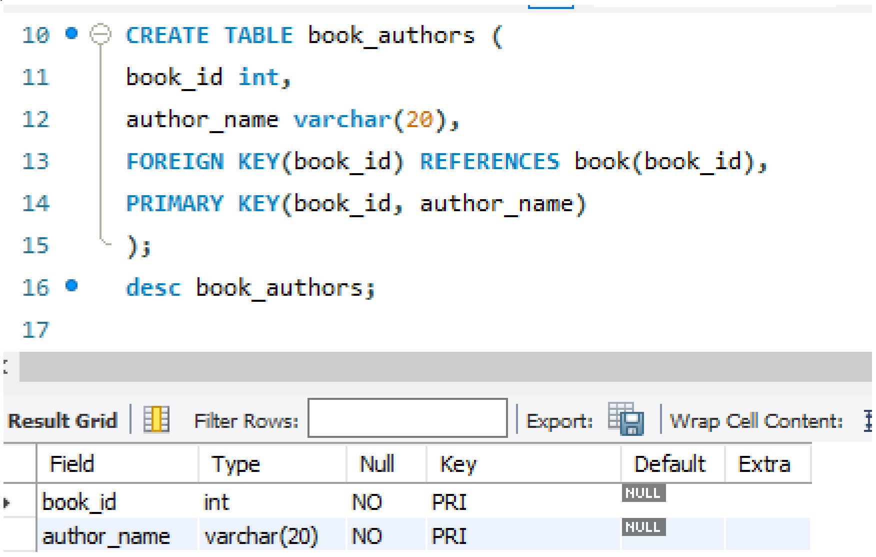


```
CREATE TABLE book (
book_id int,
title varchar(20),
publisher_name varchar(20),
pub_year varchar(10),
PRIMARY KEY (book_id)
);
desc book;
```

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

Field	Type	Null	Key	Default	Extra
book_id	int	NO	PRI	NULL	
title	varchar(20)	YES		NULL	
publisher_name	varchar(20)	YES		NULL	
pub_year	varchar(10)	YES		NULL	

```
CREATE TABLE book_authors (
book_id int,
author_name varchar(20),
FOREIGN KEY(book_id) REFERENCES book(book_id),
PRIMARY KEY(book_id, author_name)
);
```

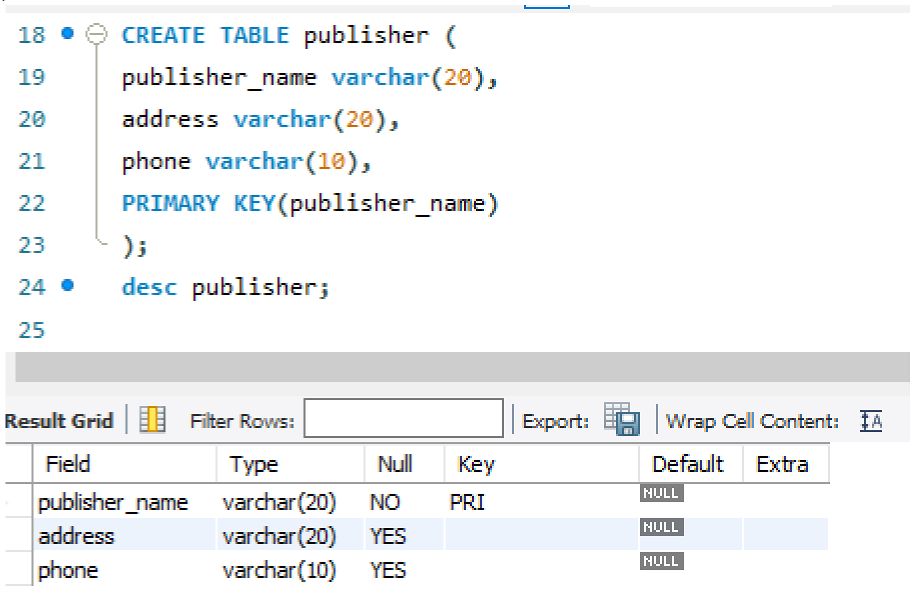


```
CREATE TABLE book_authors (
book_id int,
author_name varchar(20),
FOREIGN KEY(book_id) REFERENCES book(book_id),
PRIMARY KEY(book_id, author_name)
);
desc book_authors;

```

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

Field	Type	Null	Key	Default	Extra
book_id	int	NO	PRI	NULL	
author_name	varchar(20)	NO	PRI	NULL	

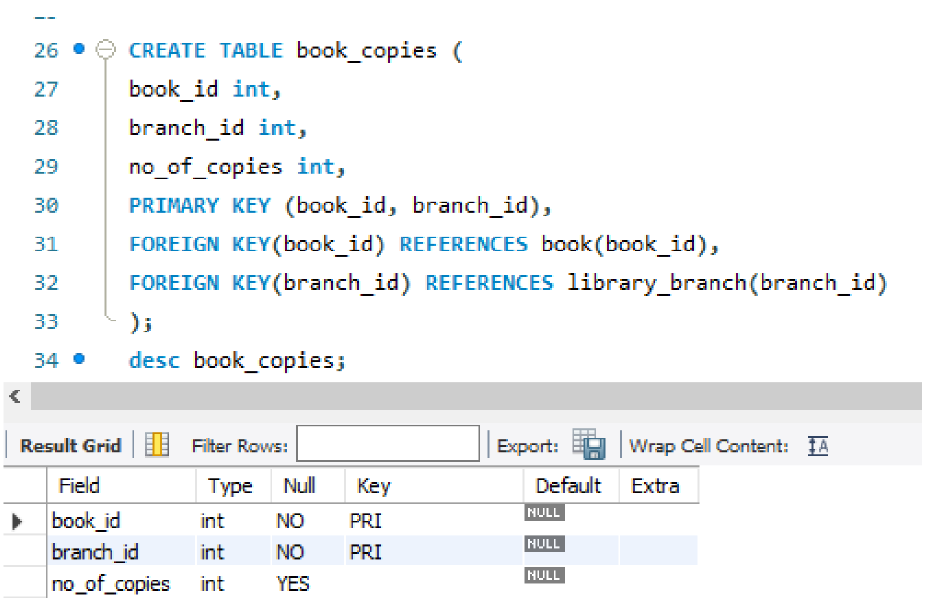
```
CREATE TABLE publisher (
publisher_name varchar(20),
address varchar(20),
phone varchar(10),
PRIMARY KEY(publisher_name)
);
```



```
CREATE TABLE publisher (
publisher_name varchar(20),
address varchar(20),
phone varchar(10),
PRIMARY KEY(publisher_name)
);
desc publisher;
```

Field	Type	Null	Key	Default	Extra
publisher_name	varchar(20)	NO	PRI	NULL	
address	varchar(20)	YES		NULL	
phone	varchar(10)	YES		NULL	

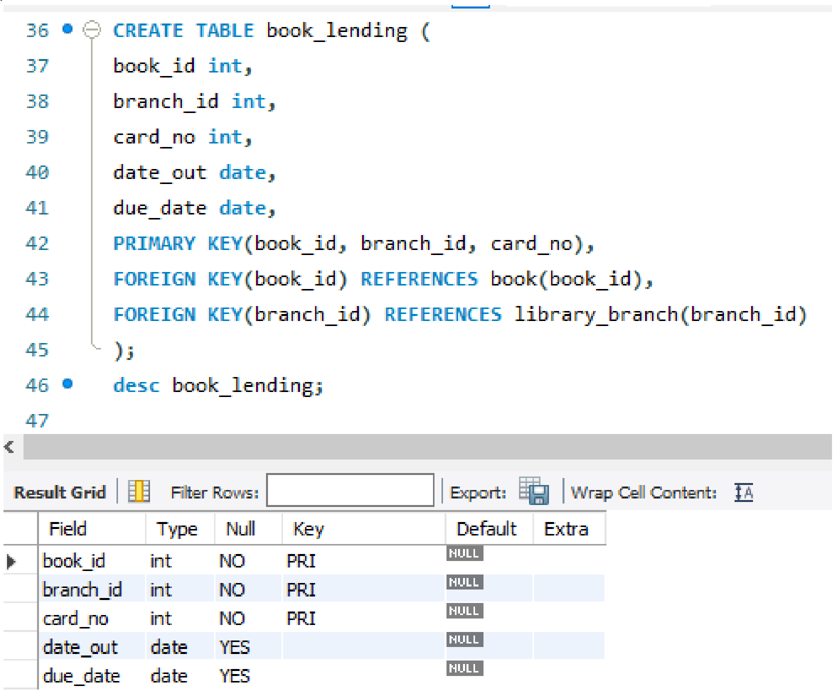
```
CREATE TABLE book_copies (
book_id int,
branch_id int,
no_of_copies int,
PRIMARY KEY (book_id, branch_id),
FOREIGN KEY(book_id) REFERENCES book(book_id),
FOREIGN KEY(branch_id) REFERENCES library_branch(branch_id)
);
```



```sql
--
CREATE TABLE book_copies (
book_id int,
branch_id int,
no_of_copies int,
PRIMARY KEY (book_id, branch_id),
FOREIGN KEY(book_id) REFERENCES book(book_id),
FOREIGN KEY(branch_id) REFERENCES library_branch(branch_id)
);
desc book_copies;
```

Field	Type	Null	Key	Default	Extra
book_id	int	NO	PRI	NULL	
branch_id	int	NO	PRI	NULL	
no_of_copies	int	YES		NULL	

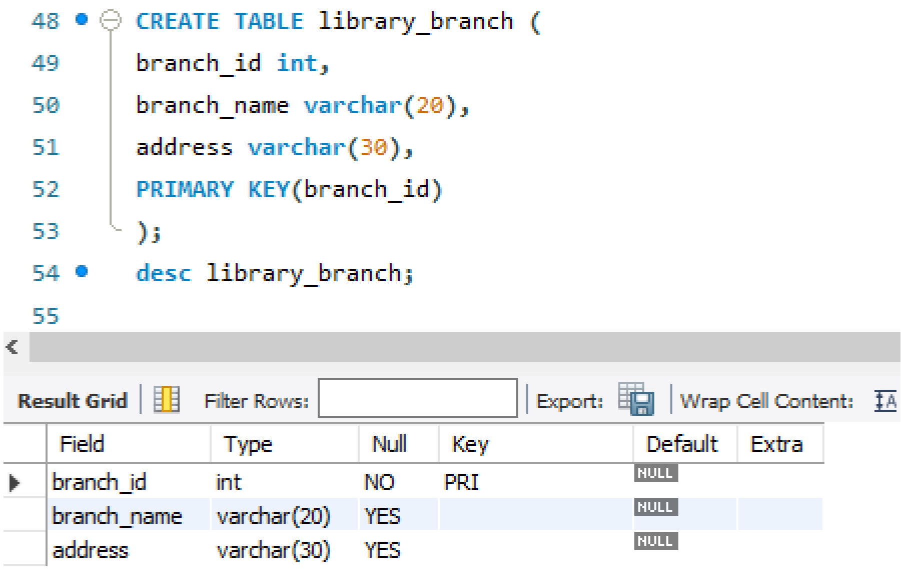
CREATE TABLE book_lending (
book_id int,
branch_id int,
card_no int,
date_out date,
due_date date,
PRIMARY KEY(book_id, branch_id, card_no),
FOREIGN KEY(book_id) REFERENCES book(book_id),
FOREIGN KEY(branch_id) REFERENCES library_branch(branch_id)
);



```sql
CREATE TABLE book_lending (
book_id int,
branch_id int,
card_no int,
date_out date,
due_date date,
PRIMARY KEY(book_id, branch_id, card_no),
FOREIGN KEY(book_id) REFERENCES book(book_id),
FOREIGN KEY(branch_id) REFERENCES library_branch(branch_id)
);
desc book_lending;
```

Field	Type	Null	Key	Default	Extra
book_id	int	NO	PRI	NULL	
branch_id	int	NO	PRI	NULL	
card_no	int	NO	PRI	NULL	
date_out	date	YES		NULL	
due_date	date	YES		NULL	

CREATE TABLE library_branch (
branch_id int,
branch_name varchar(20),
address varchar(30),
PRIMARY KEY(branch_id)
);

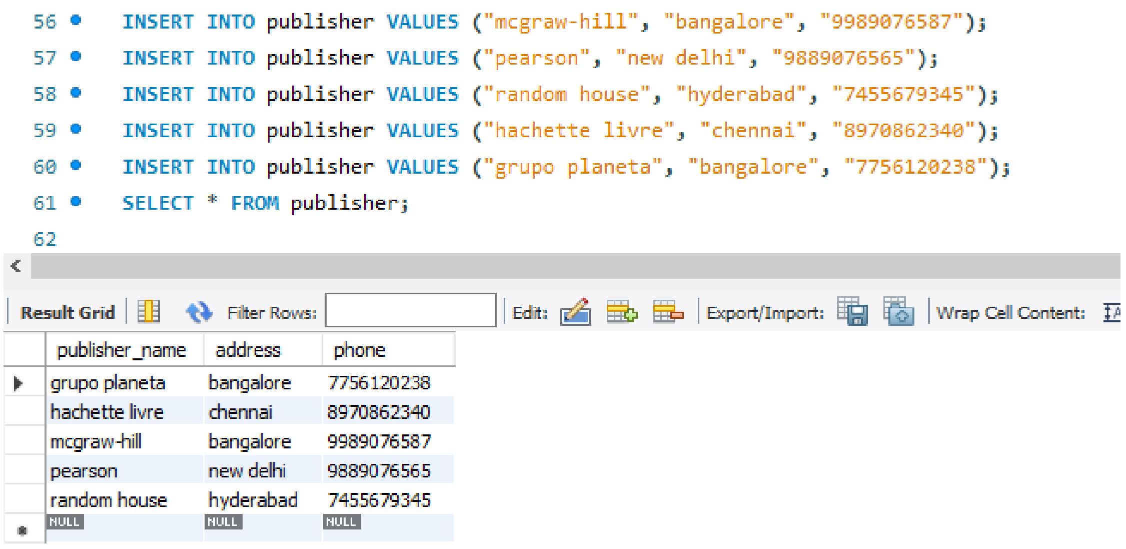


```
CREATE TABLE library_branch (
branch_id int,
branch_name varchar(20),
address varchar(30),
PRIMARY KEY(branch_id)
);
desc library_branch;
```

Field	Type	Null	Key	Default	Extra
branch_id	int	NO	PRI	NULL	
branch_name	varchar(20)	YES		NULL	
address	varchar(30)	YES		NULL	

INSERTING VALUES INTO THE TABLES:

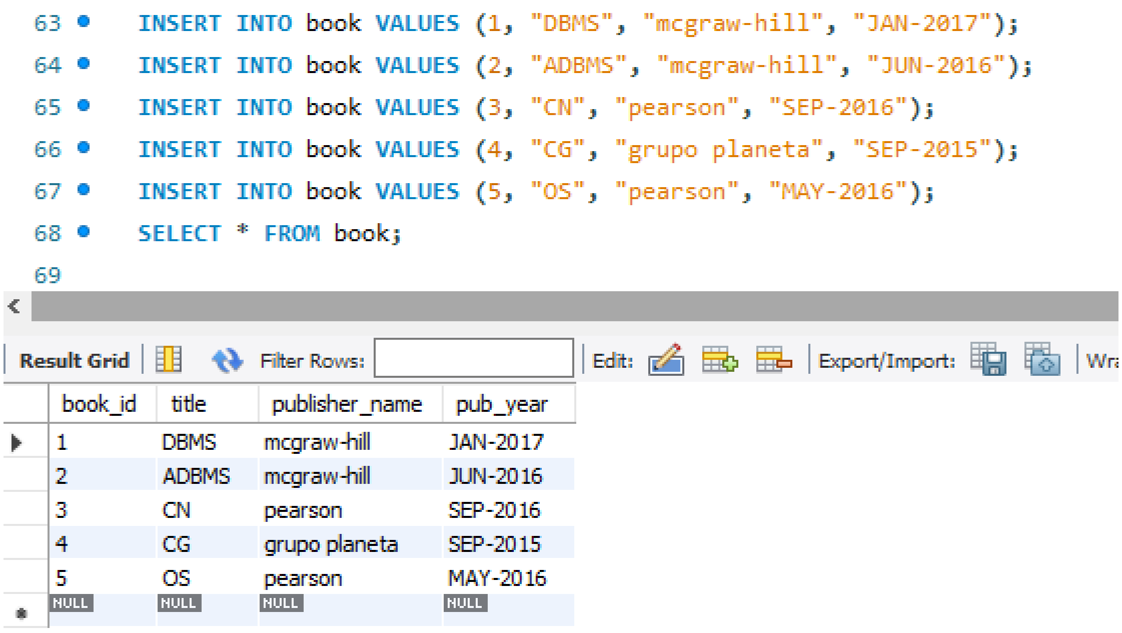
INSERT INTO publisher VALUES ("mcgraw-hill", "bangalore", "9989076587");
INSERT INTO publisher VALUES ("pearson", "new delhi", "9889076565");
INSERT INTO publisher VALUES ("random house", "hyderabad", "7455679345");
INSERT INTO publisher VALUES ("hachette livre", "chennai", "8970862340");
INSERT INTO publisher VALUES ("grupo planeta", "bangalore", "7756120238");



```
INSERT INTO publisher VALUES ("mcgraw-hill", "bangalore", "9989076587");
INSERT INTO publisher VALUES ("pearson", "new delhi", "9889076565");
INSERT INTO publisher VALUES ("random house", "hyderabad", "7455679345");
INSERT INTO publisher VALUES ("hachette livre", "chennai", "8970862340");
INSERT INTO publisher VALUES ("grupo planeta", "bangalore", "7756120238");
SELECT * FROM publisher;
```

publisher_name	address	phone
grupo planeta	bangalore	7756120238
hachette livre	chennai	8970862340
mcgraw-hill	bangalore	9989076587
pearson	new delhi	9889076565
random house	hyderabad	7455679345
NULL	NULL	NULL

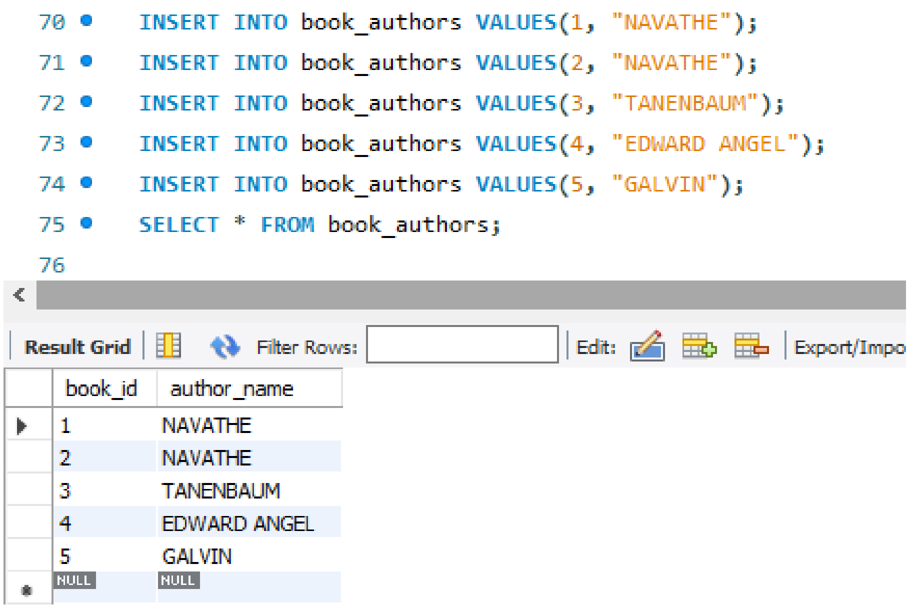
INSERT INTO book VALUES (1, "DBMS", "mcgraw-hill", "JAN-2017");
INSERT INTO book VALUES (2, "ADBMS", "mcgraw-hill", "JUN-2016");
INSERT INTO book VALUES (3, "CN", "pearson", "SEP-2016");
INSERT INTO book VALUES (4, "CG", "grupo planeta", "SEP-2015");
INSERT INTO book VALUES (5, "OS", "pearson", "MAY-2016");



```
INSERT INTO book VALUES (1, "DBMS", "mcgraw-hill", "JAN-2017");
INSERT INTO book VALUES (2, "ADBMS", "mcgraw-hill", "JUN-2016");
INSERT INTO book VALUES (3, "CN", "pearson", "SEP-2016");
INSERT INTO book VALUES (4, "CG", "grupo planeta", "SEP-2015");
INSERT INTO book VALUES (5, "OS", "pearson", "MAY-2016");
SELECT * FROM book;
```

book_id	title	publisher_name	pub_year
1	DBMS	mcgraw-hill	JAN-2017
2	ADBMS	mcgraw-hill	JUN-2016
3	CN	pearson	SEP-2016
4	CG	grupo planeta	SEP-2015
5	OS	pearson	MAY-2016
NULL	NULL	NULL	NULL

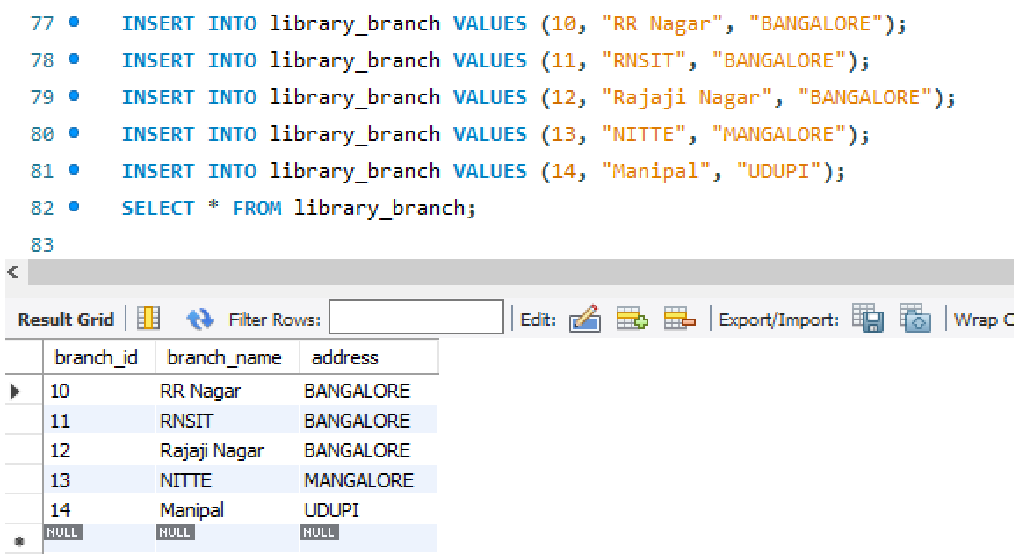
INSERT INTO book_authors VALUES(1, "NAVATHE");
INSERT INTO book_authors VALUES(2, "NAVATHE");
INSERT INTO book_authors VALUES(3, "TANENBAUM");
INSERT INTO book_authors VALUES(4, "EDWARD ANGEL");
INSERT INTO book_authors VALUES(5, "GALVIN");



```
INSERT INTO book_authors VALUES(1, "NAVATHE");
INSERT INTO book_authors VALUES(2, "NAVATHE");
INSERT INTO book_authors VALUES(3, "TANENBAUM");
INSERT INTO book_authors VALUES(4, "EDWARD ANGEL");
INSERT INTO book_authors VALUES(5, "GALVIN");
SELECT * FROM book_authors;
```

book_id	author_name
1	NAVATHE
2	NAVATHE
3	TANENBAUM
4	EDWARD ANGEL
5	GALVIN
NULL	NULL

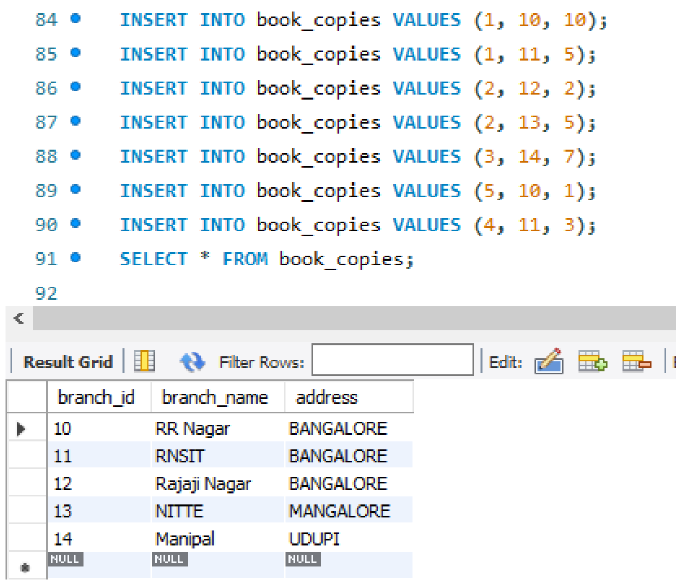
INSERT INTO library_branch VALUES (10, "RR Nagar", "BANGALORE");
INSERT INTO library_branch VALUES (11, "RNSIT", "BANGALORE");
INSERT INTO library_branch VALUES (12, "Rajaji Nagar", "BANGALORE");
INSERT INTO library_branch VALUES (13, "NITTE", "MANGALORE");
INSERT INTO library_branch VALUES (14, "Manipal", "UDUPI");



```
INSERT INTO library_branch VALUES (10, "RR Nagar", "BANGALORE");
INSERT INTO library_branch VALUES (11, "RNSIT", "BANGALORE");
INSERT INTO library_branch VALUES (12, "Rajaji Nagar", "BANGALORE");
INSERT INTO library_branch VALUES (13, "NITTE", "MANGALORE");
INSERT INTO library_branch VALUES (14, "Manipal", "UDUPI");
SELECT * FROM library_branch;
```

branch_id	branch_name	address
10	RR Nagar	BANGALORE
11	RNSIT	BANGALORE
12	Rajaji Nagar	BANGALORE
13	NITTE	MANGALORE
14	Manipal	UDUPI
NULL	NULL	NULL

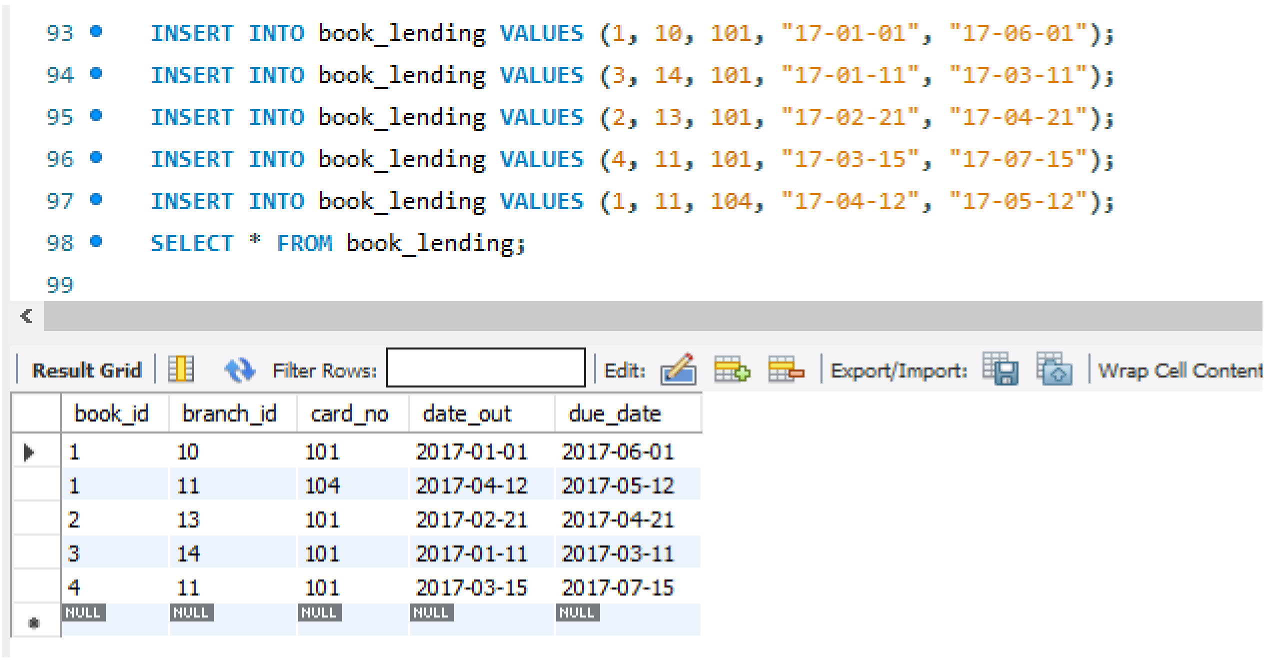
INSERT INTO book_copies VALUES (1, 10, 10);
INSERT INTO book_copies VALUES (1, 11, 5);
INSERT INTO book_copies VALUES (2, 12, 2);
INSERT INTO book_copies VALUES (2, 13, 5);
INSERT INTO book_copies VALUES (3, 14, 7);
INSERT INTO book_copies VALUES (5, 10, 1);
INSERT INTO book_copies VALUES (4, 11, 3);



```
INSERT INTO book_copies VALUES (1, 10, 10);
INSERT INTO book_copies VALUES (1, 11, 5);
INSERT INTO book_copies VALUES (2, 12, 2);
INSERT INTO book_copies VALUES (2, 13, 5);
INSERT INTO book_copies VALUES (3, 14, 7);
INSERT INTO book_copies VALUES (5, 10, 1);
INSERT INTO book_copies VALUES (4, 11, 3);
SELECT * FROM book_copies;
```

branch_id	branch_name	address
10	RR Nagar	BANGALORE
11	RNSIT	BANGALORE
12	Rajaji Nagar	BANGALORE
13	NITTE	MANGALORE
14	Manipal	UDUPI
NULL	NULL	NULL

INSERT INTO book_lending VALUES (1, 10, 101, "17-01-01", "17-06-01");
INSERT INTO book_lending VALUES (3, 14, 101, "17-01-11", "17-03-11");
INSERT INTO book_lending VALUES (2, 13, 101, "17-02-21", "17-04-21");
INSERT INTO book_lending VALUES (4, 11, 101, "17-03-15", "17-07-15");
INSERT INTO book_lending VALUES (1, 11, 104, "17-04-12", "17-05-12");



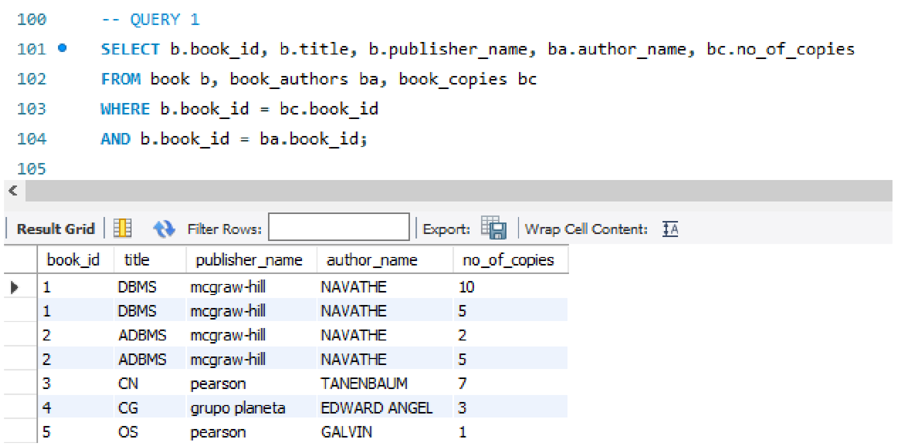
```
INSERT INTO book_lending VALUES (1, 10, 101, "17-01-01", "17-06-01");
INSERT INTO book_lending VALUES (3, 14, 101, "17-01-11", "17-03-11");
INSERT INTO book_lending VALUES (2, 13, 101, "17-02-21", "17-04-21");
INSERT INTO book_lending VALUES (4, 11, 101, "17-03-15", "17-07-15");
INSERT INTO book_lending VALUES (1, 11, 104, "17-04-12", "17-05-12");
SELECT * FROM book_lending;
```

book_id	branch_id	card_no	date_out	due_date
1	10	101	2017-01-01	2017-06-01
1	11	104	2017-04-12	2017-05-12
2	13	101	2017-02-21	2017-04-21
3	14	101	2017-01-11	2017-03-11
4	11	101	2017-03-15	2017-07-15
NULL	NULL	NULL	NULL	NULL

Write SQL queries to

1. **Retrieve details of all books in the library – id, title, name of publisher, authors, number of copies in each branch, etc.**

   SELECT b.book_id, b.title, b.publisher_name, ba.author_name, bc.no_of_copies
   FROM book b, book_authors ba, book_copies bc
   WHERE b.book_id = bc.book_id
   AND b.book_id = ba.book_id;

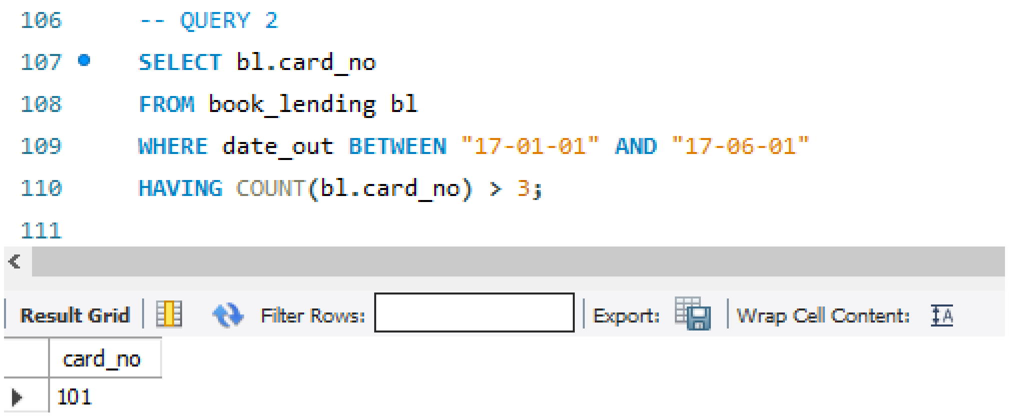


```
-- QUERY 1
SELECT b.book_id, b.title, b.publisher_name, ba.author_name, bc.no_of_copies
FROM book b, book_authors ba, book_copies bc
WHERE b.book_id = bc.book_id
AND b.book_id = ba.book_id;
```

book_id	title	publisher_name	author_name	no_of_copies
1	DBMS	mcgraw-hill	NAVATHE	10
1	DBMS	mcgraw-hill	NAVATHE	5
2	ADBMS	mcgraw-hill	NAVATHE	2
2	ADBMS	mcgraw-hill	NAVATHE	5
3	CN	pearson	TANENBAUM	7
4	CG	grupo planeta	EDWARD ANGEL	3
5	OS	pearson	GALVIN	1

2. **Get the particulars of borrowers who have borrowed more than 3 books, but from Jan 2017 to Jun 2017**

```
SELECT bl.card_no
FROM book_lending bl
WHERE date_out BETWEEN "17-01-01" AND "17-06-01"
HAVING COUNT(bl.card_no) > 3;
```

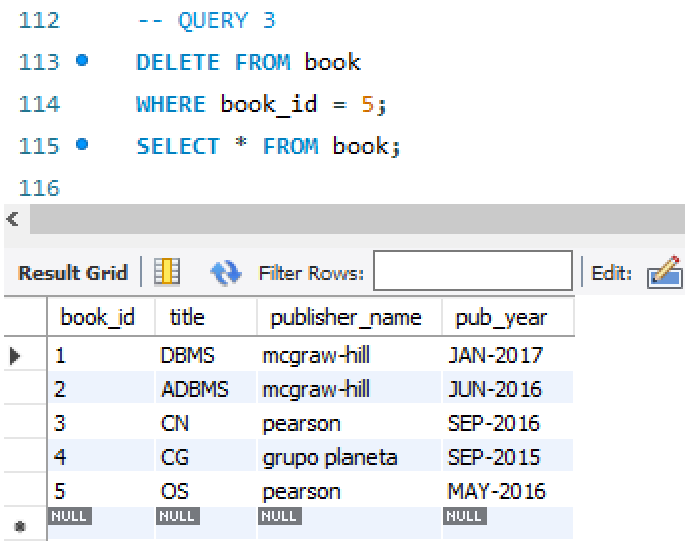


```
-- QUERY 2
SELECT bl.card_no
FROM book_lending bl
WHERE date_out BETWEEN "17-01-01" AND "17-06-01"
HAVING COUNT(bl.card_no) > 3;
```

card_no
101

3. **Delete a book in BOOK table. Update the contents of other tables to reflect this data manipulation operation.**

```
DELETE FROM book
WHERE book_id = 5;
SELECT * FROM book;
```

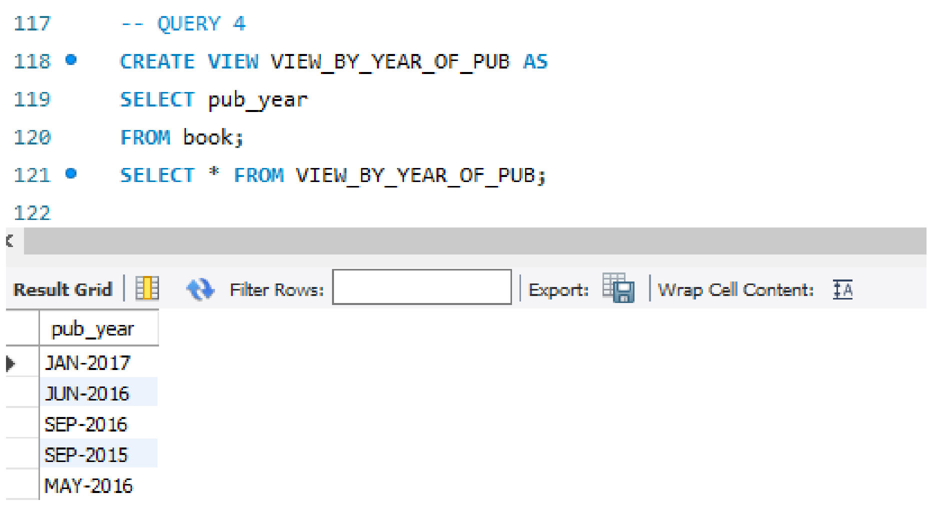


```
-- QUERY 3
DELETE FROM book
WHERE book_id = 5;
SELECT * FROM book;
```

book_id	title	publisher_name	pub_year
1	DBMS	mcgraw-hill	JAN-2017
2	ADBMS	mcgraw-hill	JUN-2016
3	CN	pearson	SEP-2016
4	CG	grupo planeta	SEP-2015
5	OS	pearson	MAY-2016
NULL	NULL	NULL	NULL

4. **Partition the BOOK table based on year of publication. Demonstrate its working with a simple query.**

   CREATE VIEW VIEW_BY_YEAR_OF_PUB AS
   SELECT pub_year
   FROM book;
   SELECT * FROM VIEW_BY_YEAR_OF_PUB;

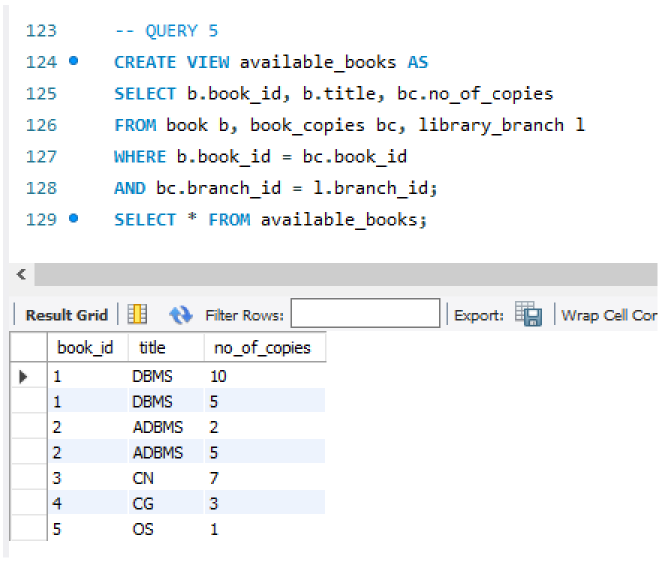


```
-- QUERY 4
CREATE VIEW VIEW_BY_YEAR_OF_PUB AS
SELECT pub_year
FROM book;
SELECT * FROM VIEW_BY_YEAR_OF_PUB;
```

pub_year
JAN-2017
JUN-2016
SEP-2016
SEP-2015
MAY-2016

5. **Create a view of all books and its number of copies that are currently available in the Library.**

   CREATE VIEW available_books AS
   SELECT b.book_id, b.title, bc.no_of_copies
   FROM book b, book_copies bc, library_branch l
   WHERE b.book_id = bc.book_id
   AND bc.branch_id = l.branch_id;
   SELECT * FROM available_books;



```
-- QUERY 5
CREATE VIEW available_books AS
SELECT b.book_id, b.title, bc.no_of_copies
FROM book b, book_copies bc, library_branch l
WHERE b.book_id = bc.book_id
AND bc.branch_id = l.branch_id;
SELECT * FROM available_books;
```

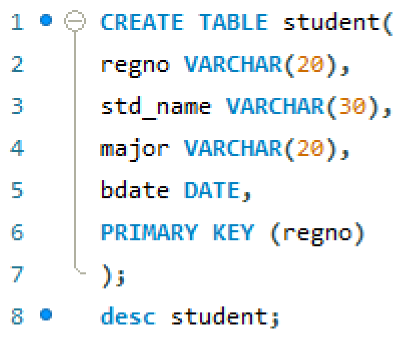
book_id	title	no_of_copies
1	DBMS	10
1	DBMS	5
2	ADBMS	2
2	ADBMS	5
3	CN	7
4	CG	3
5	OS	1

PROGRAM 8: STUDENT ENROLLMENT

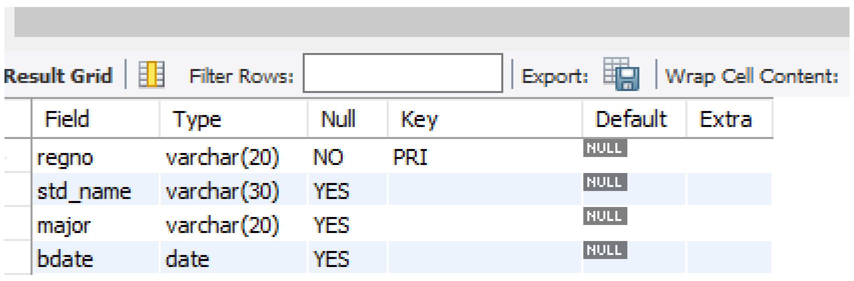
Consider the following database of student enrollment in courses & books adopted for each course.
STUDENT (regno: string, name: string, major: string, bdate:date)
COURSE (course #:int, cname:string, dept:string)
ENROLL (regno:string, course#:int, sem:int, marks:int)
BOOK _ ADOPTION (course# :int, sem:int, book-ISBN:int)
TEXT (book-ISBN:int, book-title:string, publisher:string, author:string)

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

```
CREATE TABLE student(
regno VARCHAR(20),
std_name VARCHAR(30),
major VARCHAR(20),
bdate DATE,
PRIMARY KEY (regno)
);
```



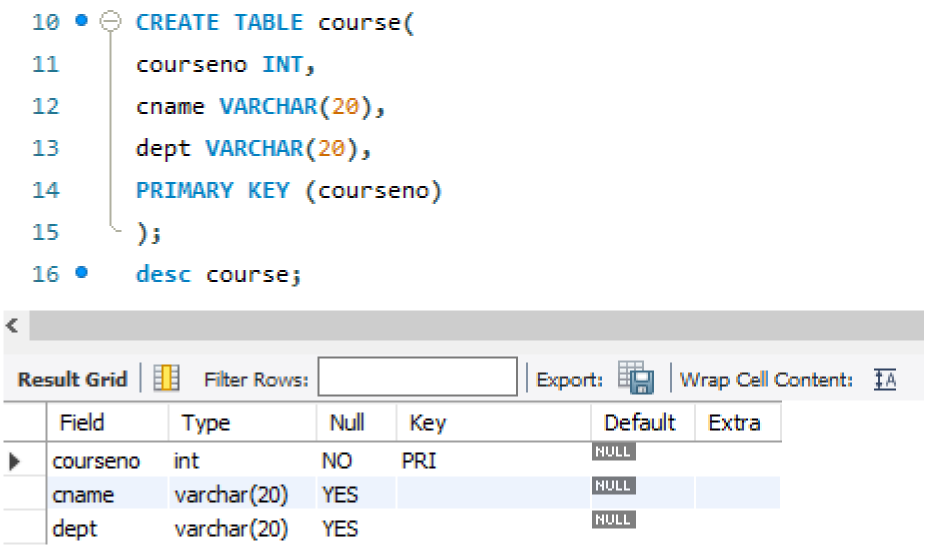
```
CREATE TABLE student(
regno VARCHAR(20),
std_name VARCHAR(30),
major VARCHAR(20),
bdate DATE,
PRIMARY KEY (regno)
);
desc student;
```



Field	Type	Null	Key	Default	Extra
regno	varchar(20)	NO	PRI	NULL	
std_name	varchar(30)	YES		NULL	
major	varchar(20)	YES		NULL	
bdate	date	YES		NULL	

```
CREATE TABLE course(
courseno INT,
cname VARCHAR(20),
dept VARCHAR(20),
PRIMARY KEY (courseno)
```

);

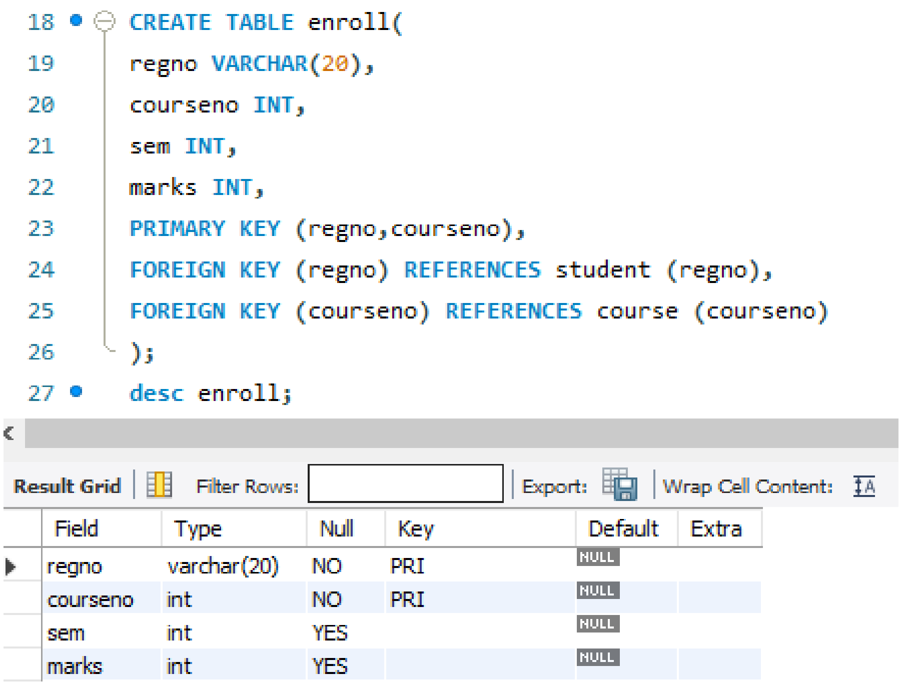


```
CREATE TABLE course(
courseno INT,
cname VARCHAR(20),
dept VARCHAR(20),
PRIMARY KEY (courseno)
);
desc course;
```

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

Field	Type	Null	Key	Default	Extra
courseno	int	NO	PRI	NULL	
cname	varchar(20)	YES		NULL	
dept	varchar(20)	YES		NULL	

CREATE TABLE enroll(
regno VARCHAR(20),
courseno INT,
sem INT,
marks INT,
PRIMARY KEY (regno,courseno),
FOREIGN KEY (regno) REFERENCES student (regno),
FOREIGN KEY (courseno) REFERENCES course (courseno)
);

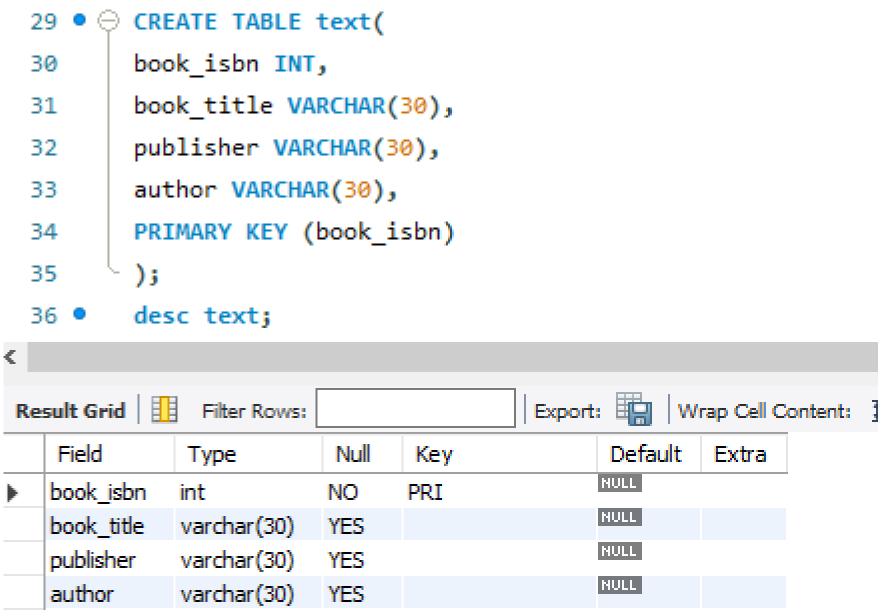


```
CREATE TABLE enroll(
regno VARCHAR(20),
courseno INT,
sem INT,
marks INT,
PRIMARY KEY (regno,courseno),
FOREIGN KEY (regno) REFERENCES student (regno),
FOREIGN KEY (courseno) REFERENCES course (courseno)
);
desc enroll;
```

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

Field	Type	Null	Key	Default	Extra
regno	varchar(20)	NO	PRI	NULL	
courseno	int	NO	PRI	NULL	
sem	int	YES		NULL	
marks	int	YES		NULL	

CREATE TABLE text(
book_isbn INT,
book_title VARCHAR(30),
publisher VARCHAR(30),
author VARCHAR(30),
PRIMARY KEY (book_isbn)
);

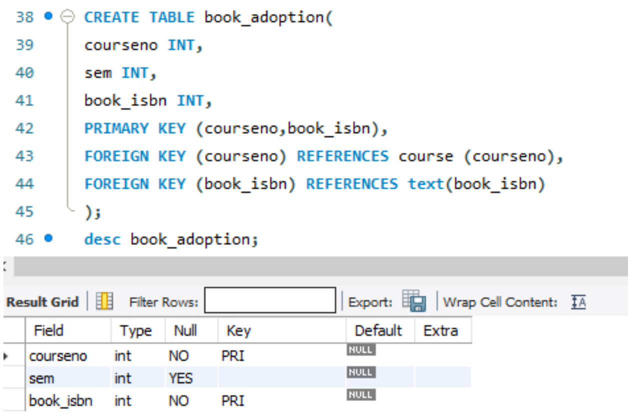


```
CREATE TABLE text(
book_isbn INT,
book_title VARCHAR(30),
publisher VARCHAR(30),
author VARCHAR(30),
PRIMARY KEY (book_isbn)
);
desc text;
```

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

```
Field       Type         Null  Key  Default  Extra
book_isbn   int          NO    PRI  NULL
book_title  varchar(30)  YES        NULL
publisher   varchar(30)  YES        NULL
author      varchar(30)  YES        NULL
```

CREATE TABLE book_adoption(
courseno INT,
sem INT,
book_isbn INT,
PRIMARY KEY (courseno,book_isbn),
FOREIGN KEY (courseno) REFERENCES course (courseno),
FOREIGN KEY (book_isbn) REFERENCES text(book_isbn)
);



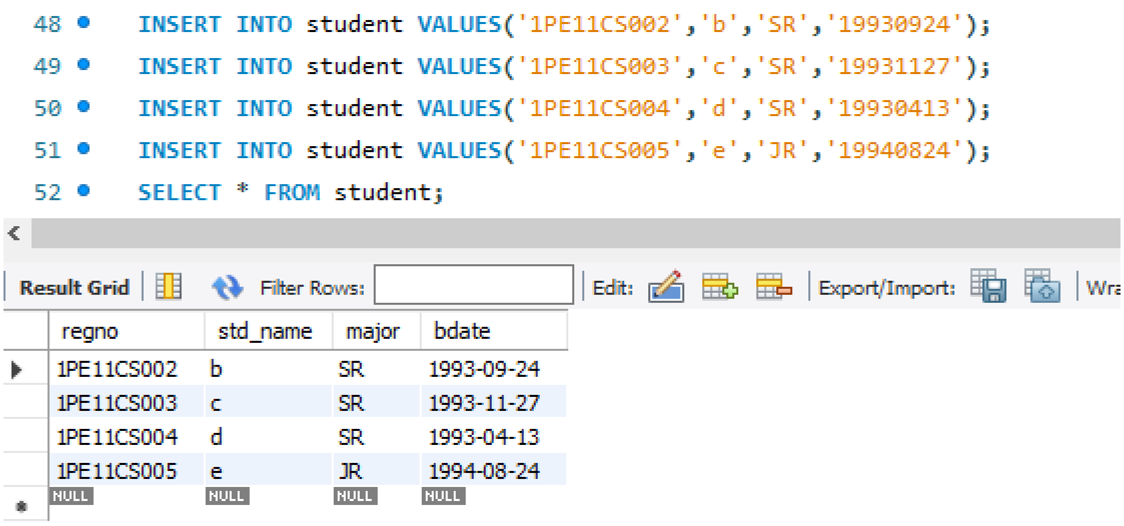
```
CREATE TABLE book_adoption(
courseno INT,
sem INT,
book_isbn INT,
PRIMARY KEY (courseno,book_isbn),
FOREIGN KEY (courseno) REFERENCES course (courseno),
FOREIGN KEY (book_isbn) REFERENCES text(book_isbn)
);
desc book_adoption;
```

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

```
Field      Type  Null  Key  Default  Extra
courseno   int   NO    PRI  NULL
sem        int   YES        NULL
book_isbn  int   NO    PRI  NULL
```

ii. Enter at least five tuples for each relation.

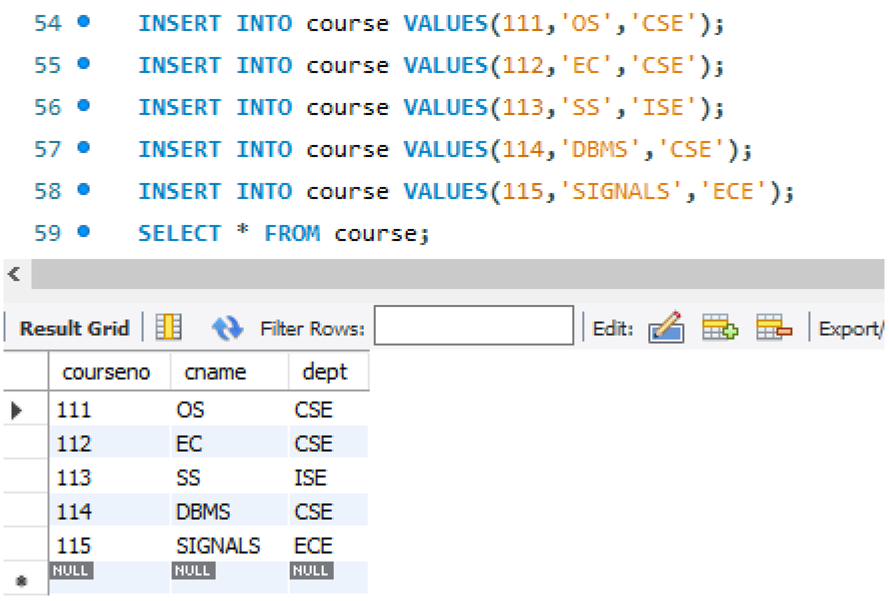
INSERT INTO student VALUES('1PE11CS002','b','SR','19930924');
INSERT INTO student VALUES('1PE11CS003','c','SR','19931127');
INSERT INTO student VALUES('1PE11CS004','d','SR','19930413');
INSERT INTO student VALUES('1PE11CS005','e','JR','19940824');



```
INSERT INTO student VALUES('1PE11CS002','b','SR','19930924');
INSERT INTO student VALUES('1PE11CS003','c','SR','19931127');
INSERT INTO student VALUES('1PE11CS004','d','SR','19930413');
INSERT INTO student VALUES('1PE11CS005','e','JR','19940824');
SELECT * FROM student;
```

regno	std_name	major	bdate
1PE11CS002	b	SR	1993-09-24
1PE11CS003	c	SR	1993-11-27
1PE11CS004	d	SR	1993-04-13
1PE11CS005	e	JR	1994-08-24
NULL	NULL	NULL	NULL

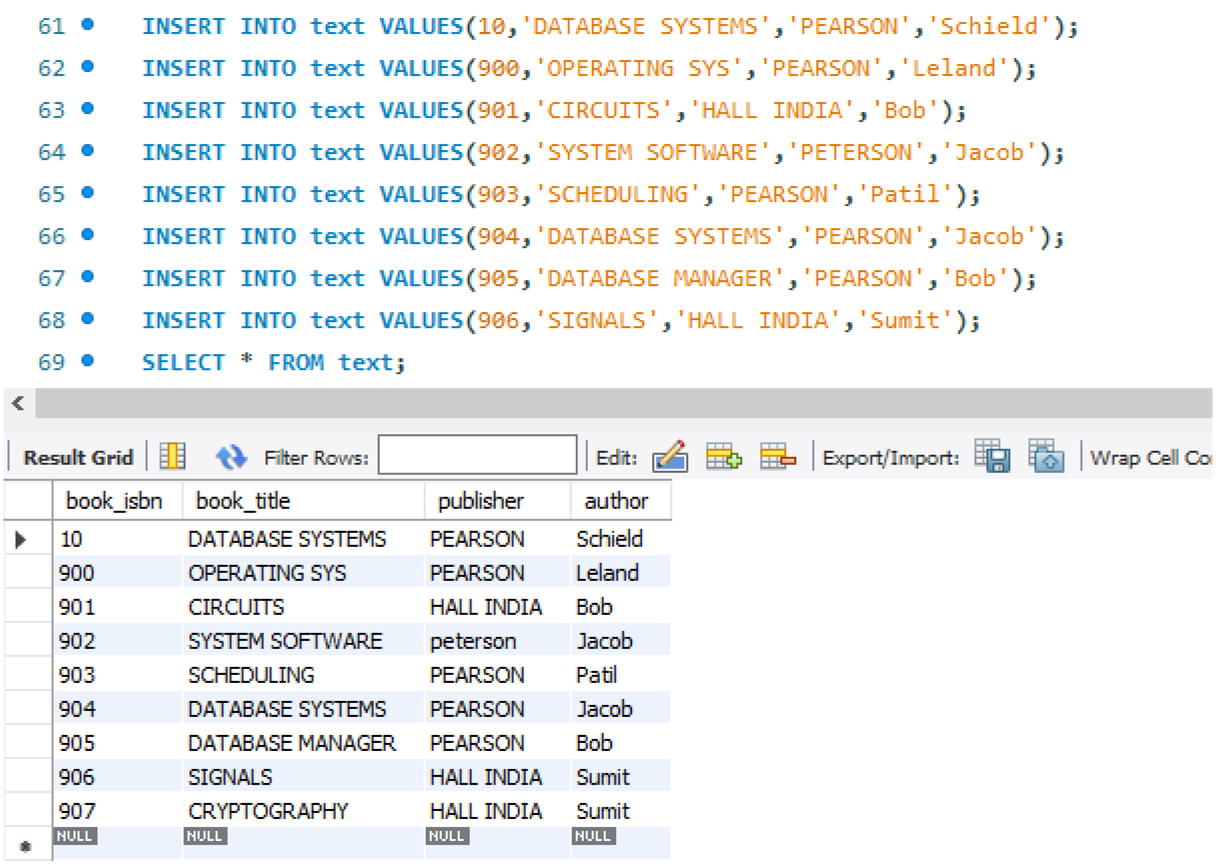
INSERT INTO course VALUES(111,'OS','CSE');
INSERT INTO course VALUES(112,'EC','CSE');
INSERT INTO course VALUES(113,'SS','ISE');
INSERT INTO course VALUES(114,'DBMS','CSE');
INSERT INTO course VALUES(115,'SIGNALS','ECE');



```
INSERT INTO course VALUES(111,'OS','CSE');
INSERT INTO course VALUES(112,'EC','CSE');
INSERT INTO course VALUES(113,'SS','ISE');
INSERT INTO course VALUES(114,'DBMS','CSE');
INSERT INTO course VALUES(115,'SIGNALS','ECE');
SELECT * FROM course;
```

courseno	cname	dept
111	OS	CSE
112	EC	CSE
113	SS	ISE
114	DBMS	CSE
115	SIGNALS	ECE
NULL	NULL	NULL

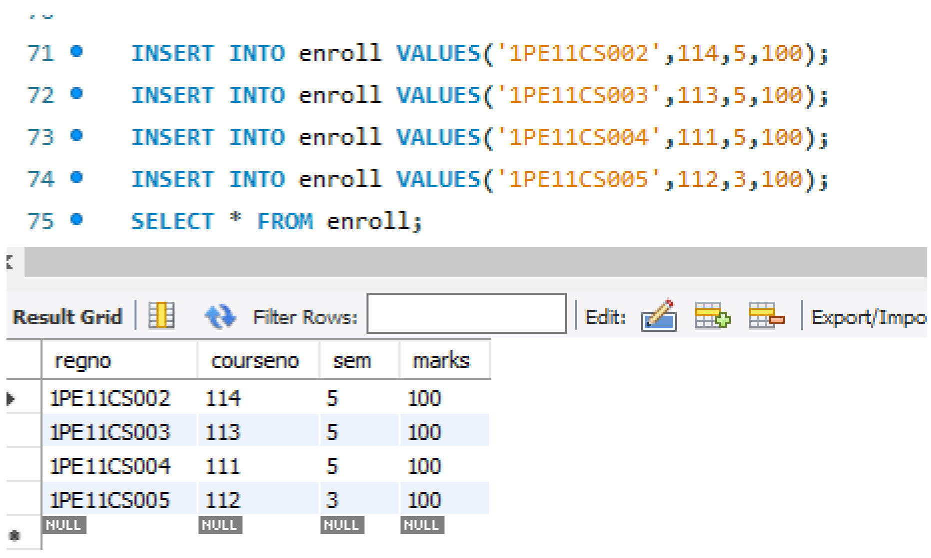
INSERT INTO text VALUES(10,'DATABASE SYSTEMS','PEARSON','Schield');
INSERT INTO text VALUES(900,'OPERATING SYS','PEARSON','Leland');
INSERT INTO text VALUES(901,'CIRCUITS','HALL INDIA','Bob');
INSERT INTO text VALUES(902,'SYSTEM SOFTWARE','PETERSON','Jacob');
INSERT INTO text VALUES(903,'SCHEDULING','PEARSON','Patil');
INSERT INTO text VALUES(904,'DATABASE SYSTEMS','PEARSON','Jacob');
INSERT INTO text VALUES(905,'DATABASE MANAGER','PEARSON','Bob');
INSERT INTO text VALUES(906,'SIGNALS','HALL INDIA','Sumit');



```
INSERT INTO text VALUES(10,'DATABASE SYSTEMS','PEARSON','Schield');
INSERT INTO text VALUES(900,'OPERATING SYS','PEARSON','Leland');
INSERT INTO text VALUES(901,'CIRCUITS','HALL INDIA','Bob');
INSERT INTO text VALUES(902,'SYSTEM SOFTWARE','PETERSON','Jacob');
INSERT INTO text VALUES(903,'SCHEDULING','PEARSON','Patil');
INSERT INTO text VALUES(904,'DATABASE SYSTEMS','PEARSON','Jacob');
INSERT INTO text VALUES(905,'DATABASE MANAGER','PEARSON','Bob');
INSERT INTO text VALUES(906,'SIGNALS','HALL INDIA','Sumit');
SELECT * FROM text;
```

book_isbn	book_title	publisher	author
10	DATABASE SYSTEMS	PEARSON	Schield
900	OPERATING SYS	PEARSON	Leland
901	CIRCUITS	HALL INDIA	Bob
902	SYSTEM SOFTWARE	peterson	Jacob
903	SCHEDULING	PEARSON	Patil
904	DATABASE SYSTEMS	PEARSON	Jacob
905	DATABASE MANAGER	PEARSON	Bob
906	SIGNALS	HALL INDIA	Sumit
907	CRYPTOGRAPHY	HALL INDIA	Sumit
NULL	NULL	NULL	NULL

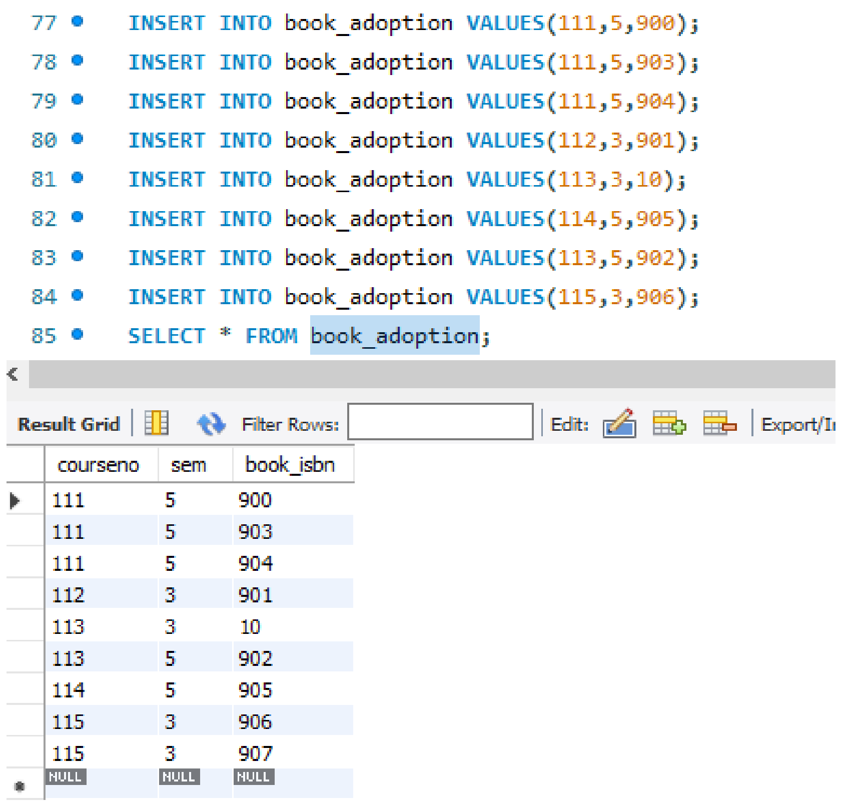
INSERT INTO enroll VALUES('1PE11CS002',114,5,100);
INSERT INTO enroll VALUES('1PE11CS003',113,5,100);
INSERT INTO enroll VALUES('1PE11CS004',111,5,100);
INSERT INTO enroll VALUES('1PE11CS005',112,3,100);



```
INSERT INTO enroll VALUES('1PE11CS002',114,5,100);
INSERT INTO enroll VALUES('1PE11CS003',113,5,100);
INSERT INTO enroll VALUES('1PE11CS004',111,5,100);
INSERT INTO enroll VALUES('1PE11CS005',112,3,100);
SELECT * FROM enroll;
```

regno	courseno	sem	marks
1PE11CS002	114	5	100
1PE11CS003	113	5	100
1PE11CS004	111	5	100
1PE11CS005	112	3	100
NULL	NULL	NULL	NULL

INSERT INTO book_adoption VALUES(111,5,900);
INSERT INTO book_adoption VALUES(111,5,903);
INSERT INTO book_adoption VALUES(111,5,904);
INSERT INTO book_adoption VALUES(112,3,901);
INSERT INTO book_adoption VALUES(113,3,10);
INSERT INTO book_adoption VALUES(114,5,905);
INSERT INTO book_adoption VALUES(113,5,902);
INSERT INTO book_adoption VALUES(115,3,906);

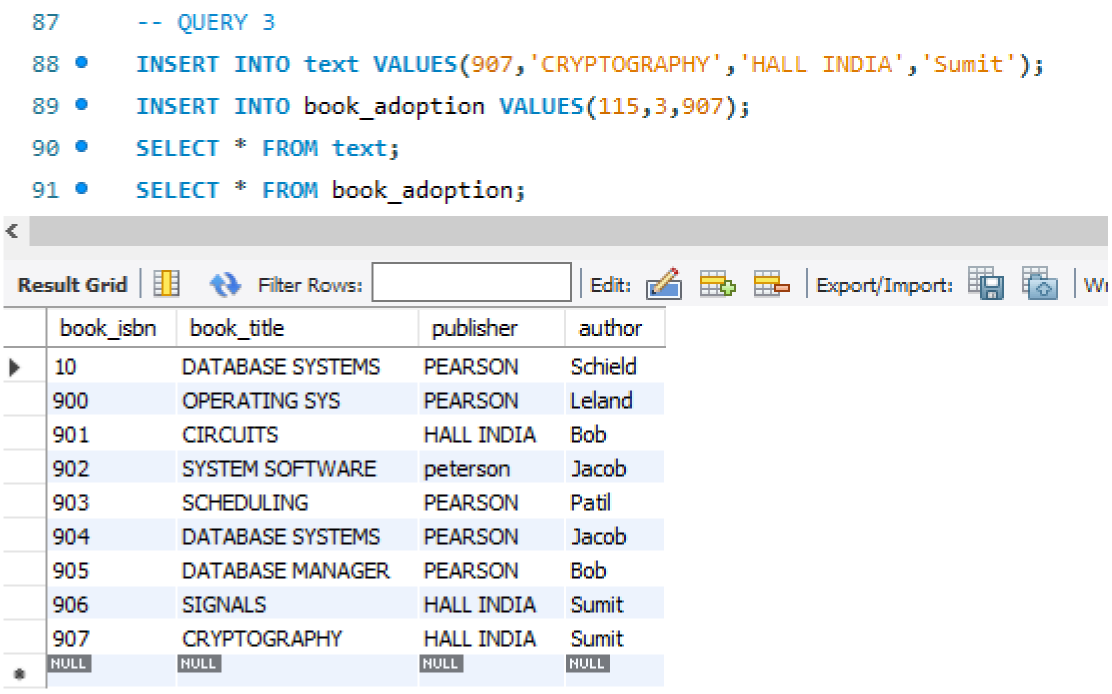


```
INSERT INTO book_adoption VALUES(111,5,900);
INSERT INTO book_adoption VALUES(111,5,903);
INSERT INTO book_adoption VALUES(111,5,904);
INSERT INTO book_adoption VALUES(112,3,901);
INSERT INTO book_adoption VALUES(113,3,10);
INSERT INTO book_adoption VALUES(114,5,905);
INSERT INTO book_adoption VALUES(113,5,902);
INSERT INTO book_adoption VALUES(115,3,906);
SELECT * FROM book_adoption;
```

courseno	sem	book_isbn
111	5	900
111	5	903
111	5	904
112	3	901
113	3	10
113	5	902
114	5	905
115	3	906
115	3	907
NULL	NULL	NULL

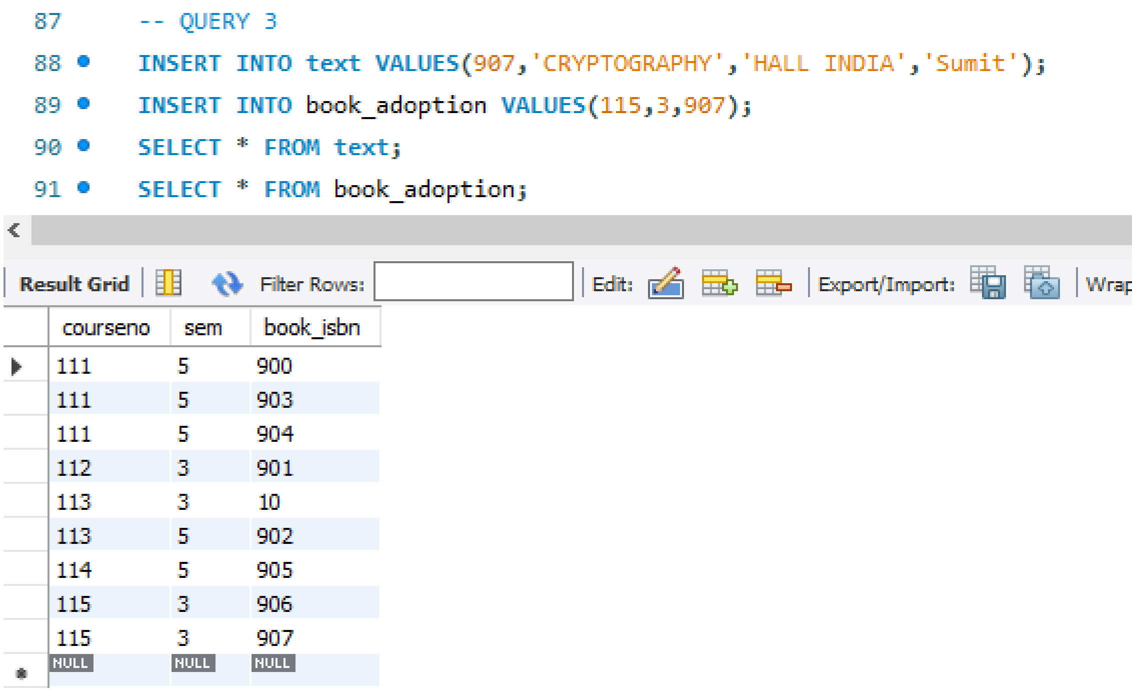
iii. Demonstrate how you add a new text book to the database and make this book be adopted by some department.

INSERT INTO text VALUES(907,'CRYPTOGRAPHY','HALL INDIA','Sumit');
INSERT INTO book_adoption VALUES(115,3,907);
SELECT * FROM text;
SELECT * FROM book_adoption;



```
-- QUERY 3
INSERT INTO text VALUES(907,'CRYPTOGRAPHY','HALL INDIA','Sumit');
INSERT INTO book_adoption VALUES(115,3,907);
SELECT * FROM text;
SELECT * FROM book_adoption;
```

book_isbn	book_title	publisher	author
10	DATABASE SYSTEMS	PEARSON	Schield
900	OPERATING SYS	PEARSON	Leland
901	CIRCUITS	HALL INDIA	Bob
902	SYSTEM SOFTWARE	peterson	Jacob
903	SCHEDULING	PEARSON	Patil
904	DATABASE SYSTEMS	PEARSON	Jacob
905	DATABASE MANAGER	PEARSON	Bob
906	SIGNALS	HALL INDIA	Sumit
907	CRYPTOGRAPHY	HALL INDIA	Sumit
NULL	NULL	NULL	NULL

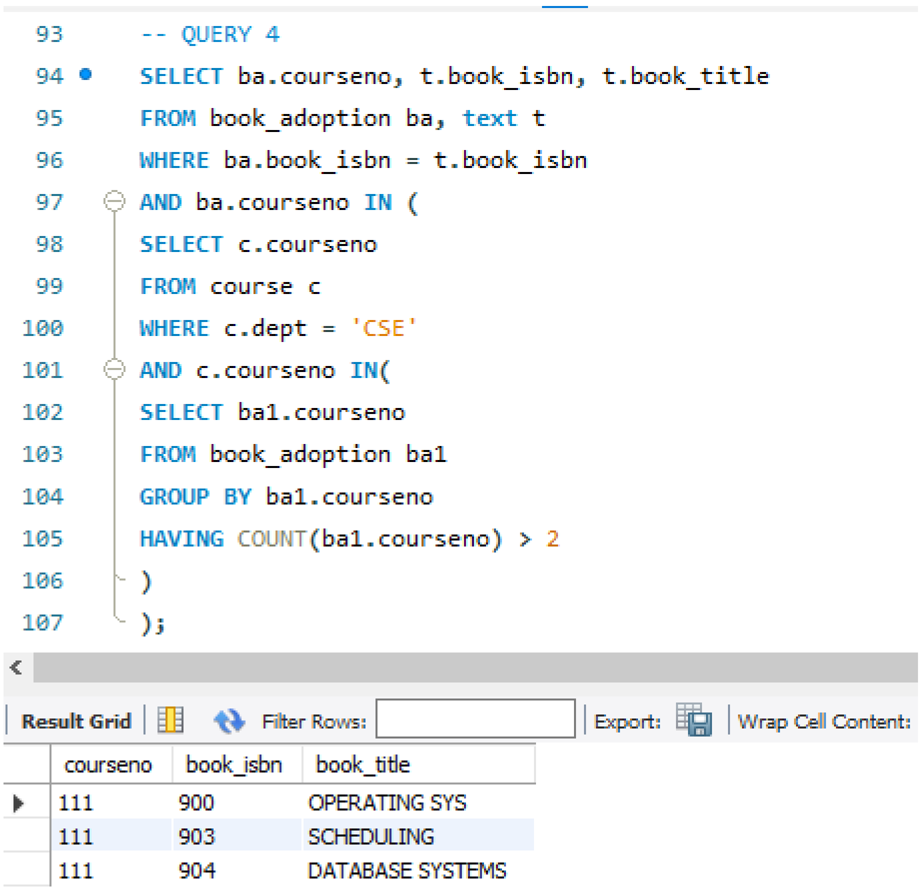


```
-- QUERY 3
INSERT INTO text VALUES(907,'CRYPTOGRAPHY','HALL INDIA','Sumit');
INSERT INTO book_adoption VALUES(115,3,907);
SELECT * FROM text;
SELECT * FROM book_adoption;
```

courseno	sem	book_isbn
111	5	900
111	5	903
111	5	904
112	3	901
113	3	10
113	5	902
114	5	905
115	3	906
115	3	907
NULL	NULL	NULL

iv. 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 ba.courseno, t.book_isbn, t.book_title
FROM book_adoption ba, text t
WHERE ba.book_isbn = t.book_isbn
AND ba.courseno IN (
SELECT c.courseno
FROM course c
WHERE c.dept = 'CSE'
AND c.courseno IN(
SELECT ba1.courseno
FROM book_adoption ba1
GROUP BY ba1.courseno
HAVING COUNT(ba1.courseno) > 2
)
);
```

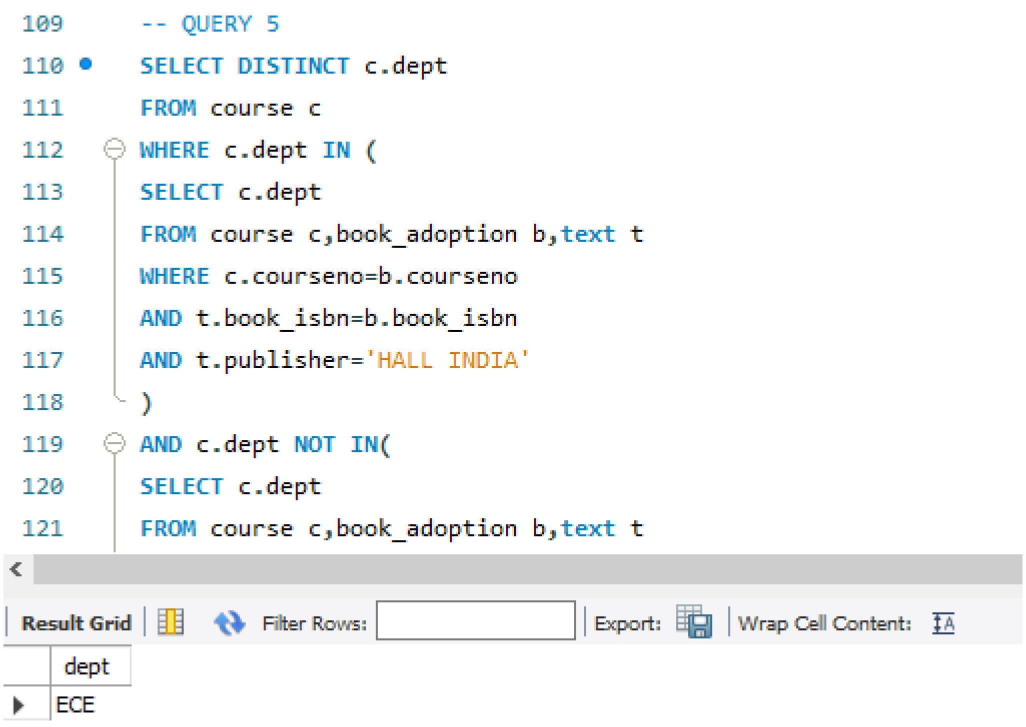


```
-- QUERY 4
SELECT ba.courseno, t.book_isbn, t.book_title
FROM book_adoption ba, text t
WHERE ba.book_isbn = t.book_isbn
AND ba.courseno IN (
SELECT c.courseno
FROM course c
WHERE c.dept = 'CSE'
AND c.courseno IN(
SELECT ba1.courseno
FROM book_adoption ba1
GROUP BY ba1.courseno
HAVING COUNT(ba1.courseno) > 2
)
);
```

courseno	book_isbn	book_title
111	900	OPERATING SYS
111	903	SCHEDULING
111	904	DATABASE SYSTEMS

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

```
SELECT DISTINCT c.dept
FROM course c
WHERE c.dept IN (
SELECT c.dept
FROM course c,book_adoption b,text t
WHERE c.courseno=b.courseno
AND t.book_isbn=b.book_isbn
AND t.publisher='HALL INDIA'
)
AND c.dept NOT IN(
SELECT c.dept
FROM course c,book_adoption b,text t
WHERE c.courseno=b.courseno
AND t.book_isbn=b.book_isbn
AND t.publisher != 'HALL INDIA'
);
```



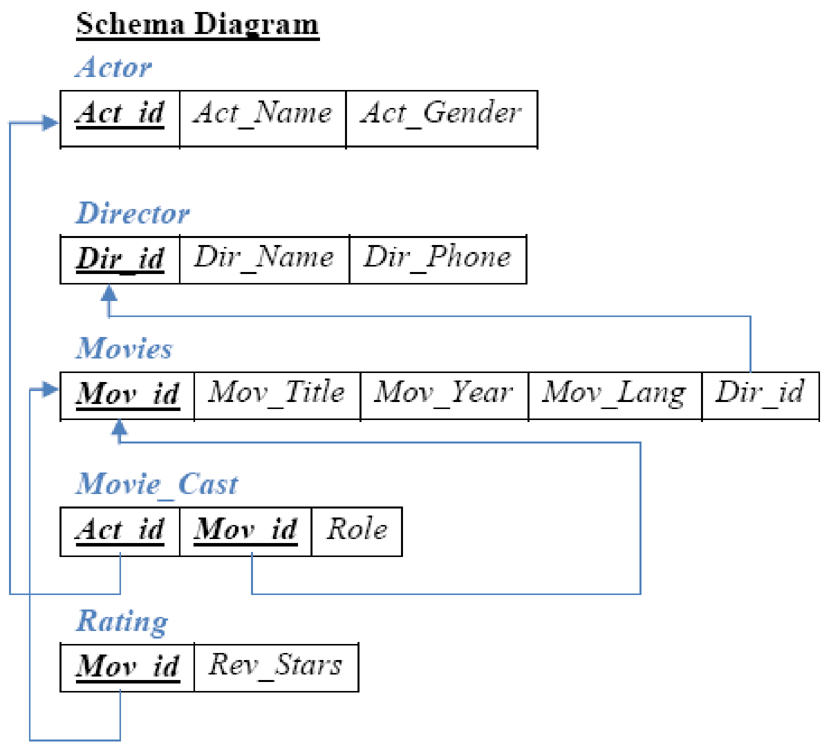
```
-- QUERY 5
SELECT DISTINCT c.dept
FROM course c
WHERE c.dept IN (
SELECT c.dept
FROM course c,book_adoption b,text t
WHERE c.courseno=b.courseno
AND t.book_isbn=b.book_isbn
AND t.publisher='HALL INDIA'
)
AND c.dept NOT IN(
SELECT c.dept
FROM course c,book_adoption b,text t
```

dept
ECE

PROGRAM 9: MOVIE DATABASE

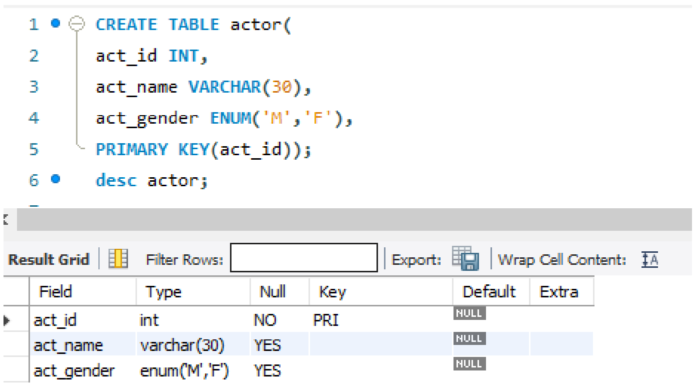
Consider the schema for Movie Database:

ACTOR (*Act_id, Act_Name, Act_Gender*)
DIRECTOR (*Dir_id, Dir_Name, Dir_Phone*)
MOVIES (*Mov_id, Mov_Title, Mov_Year, Mov_Lang, Dir_id*)
MOVIE_CAST (*Act_id, Mov_id, Role*)
RATING (*Mov_id, Rev_Stars*)



CREATION OF TABLES:

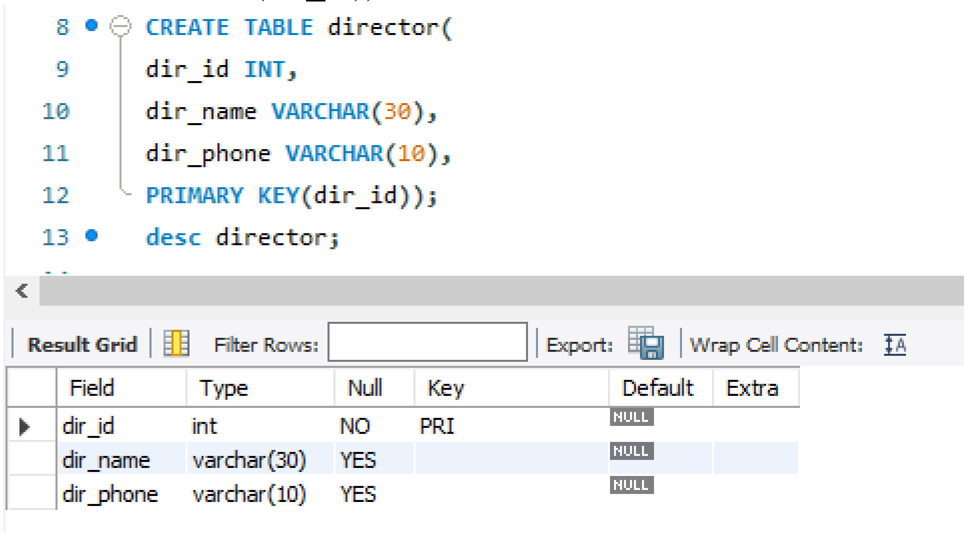
```
CREATE TABLE actor(
act_id INT,
act_name VARCHAR(30),
act_gender ENUM('M','F'),
PRIMARY KEY(act_id));
```



```
CREATE TABLE actor(
act_id INT,
act_name VARCHAR(30),
act_gender ENUM('M','F'),
PRIMARY KEY(act_id));
desc actor;
```

Field	Type	Null	Key	Default	Extra
act_id	int	NO	PRI	NULL	
act_name	varchar(30)	YES		NULL	
act_gender	enum('M','F')	YES		NULL	

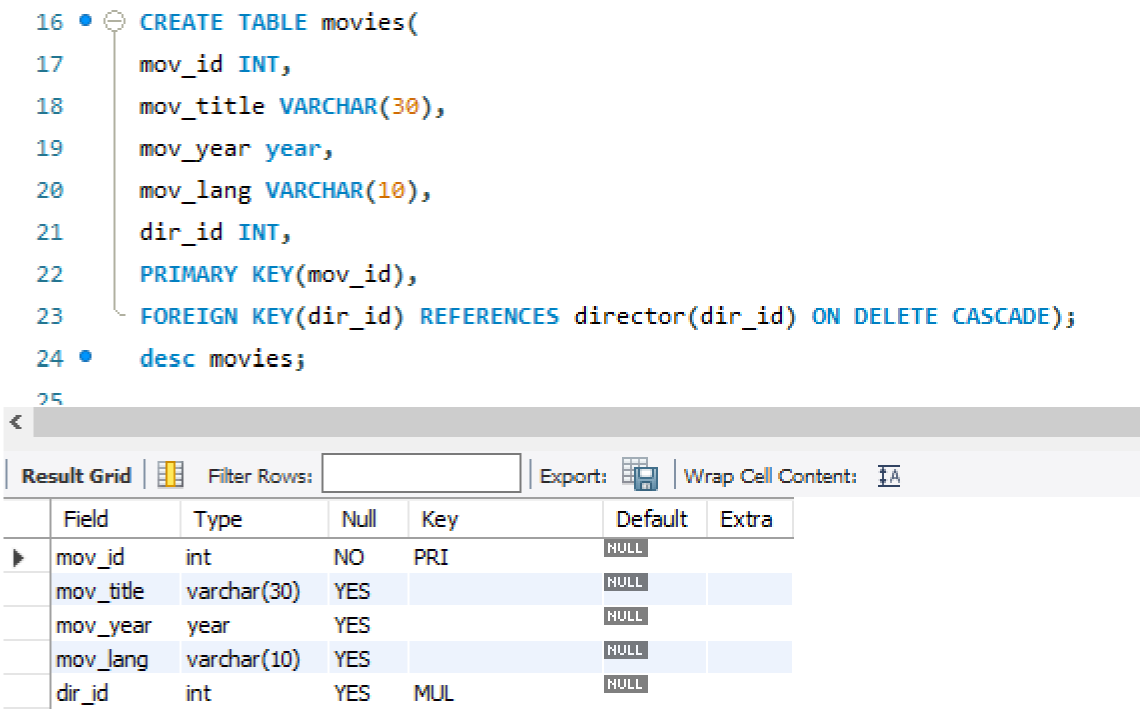
```
CREATE TABLE director(
dir_id INT,
dir_name VARCHAR(30),
dir_phone VARCHAR(10),
PRIMARY KEY(dir_id));
```



```
CREATE TABLE director(
dir_id INT,
dir_name VARCHAR(30),
dir_phone VARCHAR(10),
PRIMARY KEY(dir_id));
desc director;
```

Field	Type	Null	Key	Default	Extra
dir_id	int	NO	PRI	NULL	
dir_name	varchar(30)	YES		NULL	
dir_phone	varchar(10)	YES		NULL	

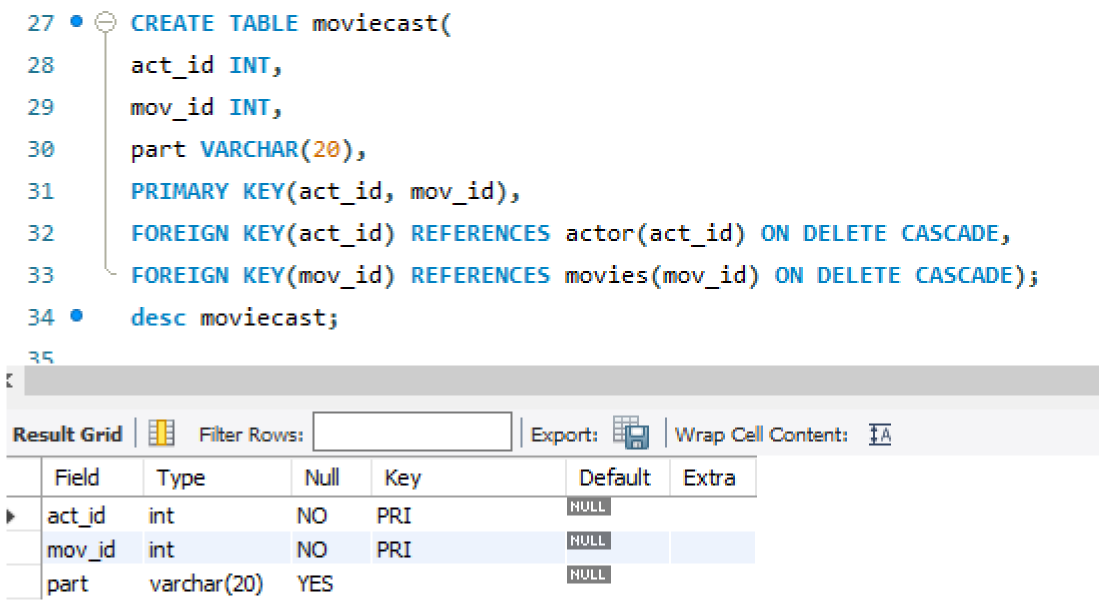
```
CREATE TABLE movies(
mov_id INT,
mov_title VARCHAR(30),
mov_year year,
mov_lang VARCHAR(10),
dir_id INT,
PRIMARY KEY(mov_id),
FOREIGN KEY(dir_id) REFERENCES director(dir_id) ON DELETE CASCADE);
```



```sql
CREATE TABLE movies(
mov_id INT,
mov_title VARCHAR(30),
mov_year year,
mov_lang VARCHAR(10),
dir_id INT,
PRIMARY KEY(mov_id),
FOREIGN KEY(dir_id) REFERENCES director(dir_id) ON DELETE CASCADE);
desc movies;
```

Field	Type	Null	Key	Default	Extra
mov_id	int	NO	PRI	NULL	
mov_title	varchar(30)	YES		NULL	
mov_year	year	YES		NULL	
mov_lang	varchar(10)	YES		NULL	
dir_id	int	YES	MUL	NULL	

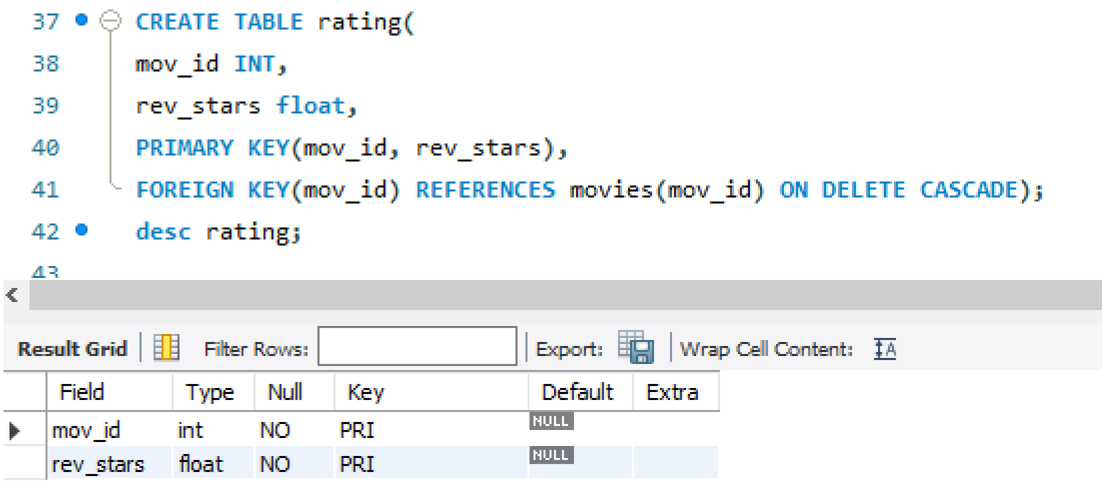
CREATE TABLE moviecast(
act_id INT,
mov_id INT,
part VARCHAR(20),
PRIMARY KEY(act_id, mov_id),
FOREIGN KEY(act_id) REFERENCES actor(act_id) ON DELETE CASCADE,
FOREIGN KEY(mov_id) REFERENCES movies(mov_id) ON DELETE CASCADE);



```sql
CREATE TABLE moviecast(
act_id INT,
mov_id INT,
part VARCHAR(20),
PRIMARY KEY(act_id, mov_id),
FOREIGN KEY(act_id) REFERENCES actor(act_id) ON DELETE CASCADE,
FOREIGN KEY(mov_id) REFERENCES movies(mov_id) ON DELETE CASCADE);
desc moviecast;
```

Field	Type	Null	Key	Default	Extra
act_id	int	NO	PRI	NULL	
mov_id	int	NO	PRI	NULL	
part	varchar(20)	YES		NULL	

```
CREATE TABLE rating(
mov_id INT,
rev_stars float,
PRIMARY KEY(mov_id, rev_stars),
FOREIGN KEY(mov_id) REFERENCES movies(mov_id) ON DELETE CASCADE);
```

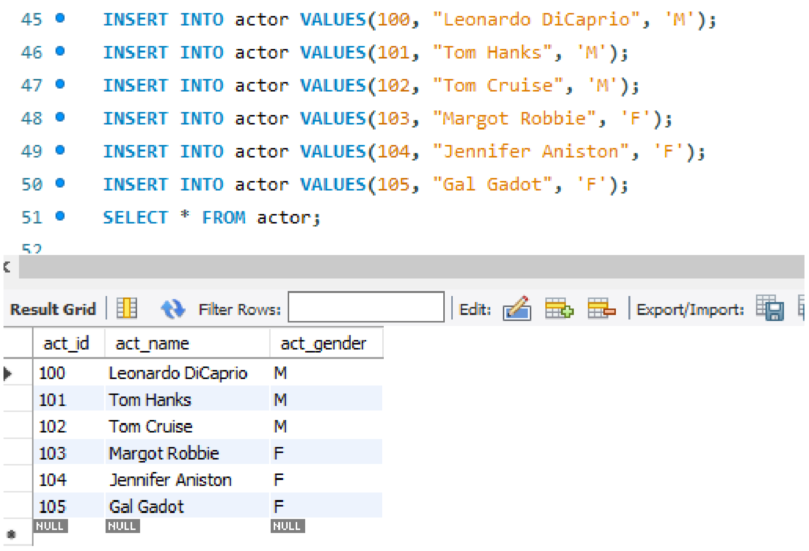


```
CREATE TABLE rating(
mov_id INT,
rev_stars float,
PRIMARY KEY(mov_id, rev_stars),
FOREIGN KEY(mov_id) REFERENCES movies(mov_id) ON DELETE CASCADE);
desc rating;
```

Field	Type	Null	Key	Default	Extra
mov_id	int	NO	PRI	NULL	
rev_stars	float	NO	PRI	NULL	

INSERTING VALUES INTO THE TABLES:

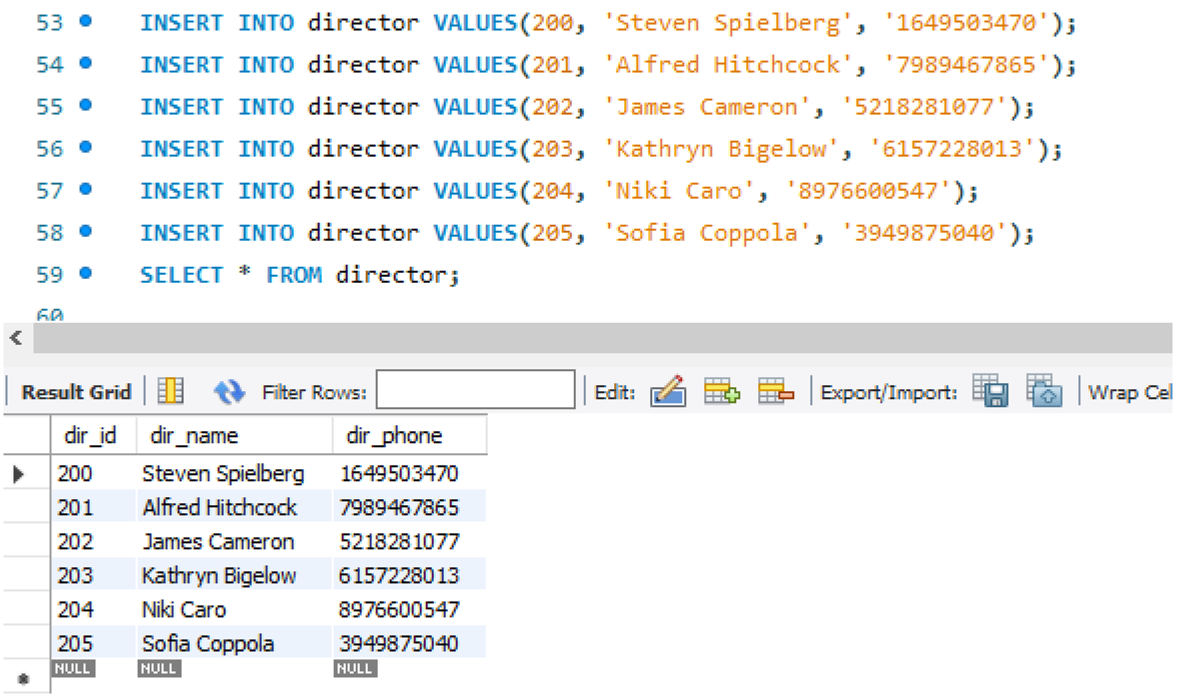
```
INSERT INTO actor VALUES(100, "Leonardo DiCaprio", 'M');
INSERT INTO actor VALUES(101, "Tom Hanks", 'M');
INSERT INTO actor VALUES(102, "Tom Cruise", 'M');
INSERT INTO actor VALUES(103, "Margot Robbie", 'F');
INSERT INTO actor VALUES(104, "Jennifer Aniston", 'F');
INSERT INTO actor VALUES(105, "Gal Gadot", 'F');
SELECT * FROM actor;
```



```
INSERT INTO actor VALUES(100, "Leonardo DiCaprio", 'M');
INSERT INTO actor VALUES(101, "Tom Hanks", 'M');
INSERT INTO actor VALUES(102, "Tom Cruise", 'M');
INSERT INTO actor VALUES(103, "Margot Robbie", 'F');
INSERT INTO actor VALUES(104, "Jennifer Aniston", 'F');
INSERT INTO actor VALUES(105, "Gal Gadot", 'F');
SELECT * FROM actor;
```

act_id	act_name	act_gender
100	Leonardo DiCaprio	M
101	Tom Hanks	M
102	Tom Cruise	M
103	Margot Robbie	F
104	Jennifer Aniston	F
105	Gal Gadot	F
NULL	NULL	NULL

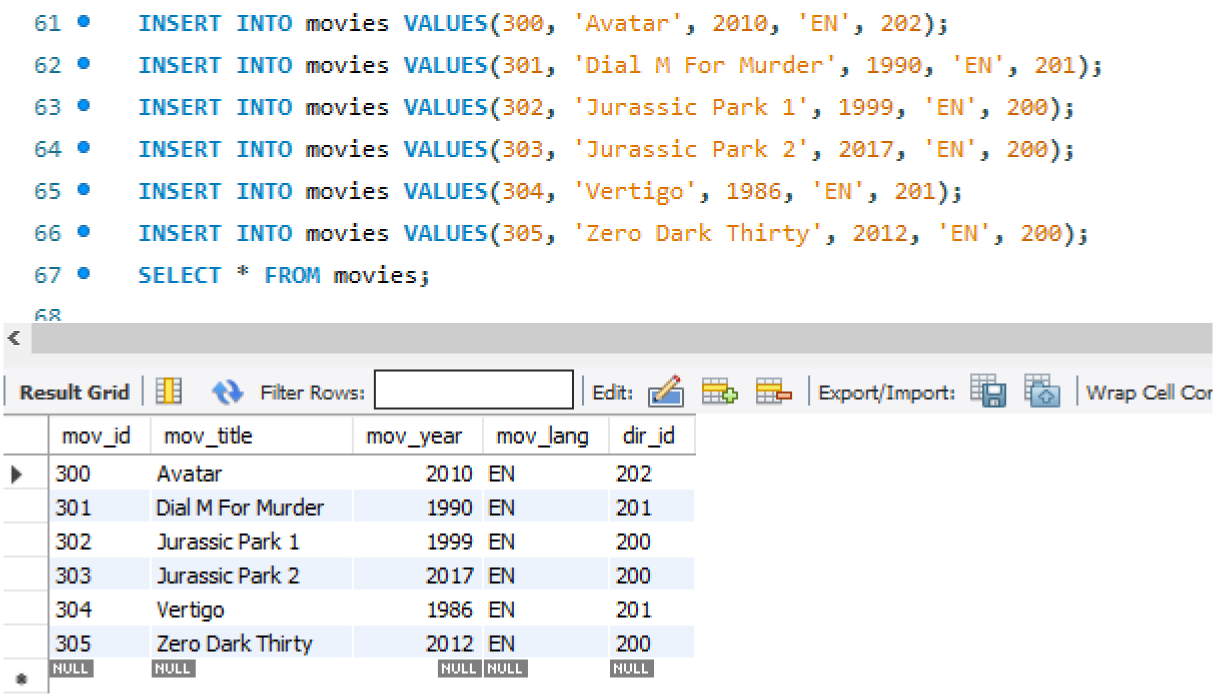
```
INSERT INTO director VALUES(200, 'Steven Spielberg', '1649503470');
INSERT INTO director VALUES(201, 'Alfred Hitchcock', '7989467865');
INSERT INTO director VALUES(202, 'James Cameron', '5218281077');
INSERT INTO director VALUES(203, 'Kathryn Bigelow', '6157228013');
INSERT INTO director VALUES(204, 'Niki Caro', '8976600547');
INSERT INTO director VALUES(205, 'Sofia Coppola', '3949875040');
SELECT * FROM director;
```



```
INSERT INTO director VALUES(200, 'Steven Spielberg', '1649503470');
INSERT INTO director VALUES(201, 'Alfred Hitchcock', '7989467865');
INSERT INTO director VALUES(202, 'James Cameron', '5218281077');
INSERT INTO director VALUES(203, 'Kathryn Bigelow', '6157228013');
INSERT INTO director VALUES(204, 'Niki Caro', '8976600547');
INSERT INTO director VALUES(205, 'Sofia Coppola', '3949875040');
SELECT * FROM director;
```

dir_id	dir_name	dir_phone
200	Steven Spielberg	1649503470
201	Alfred Hitchcock	7989467865
202	James Cameron	5218281077
203	Kathryn Bigelow	6157228013
204	Niki Caro	8976600547
205	Sofia Coppola	3949875040
NULL	NULL	NULL

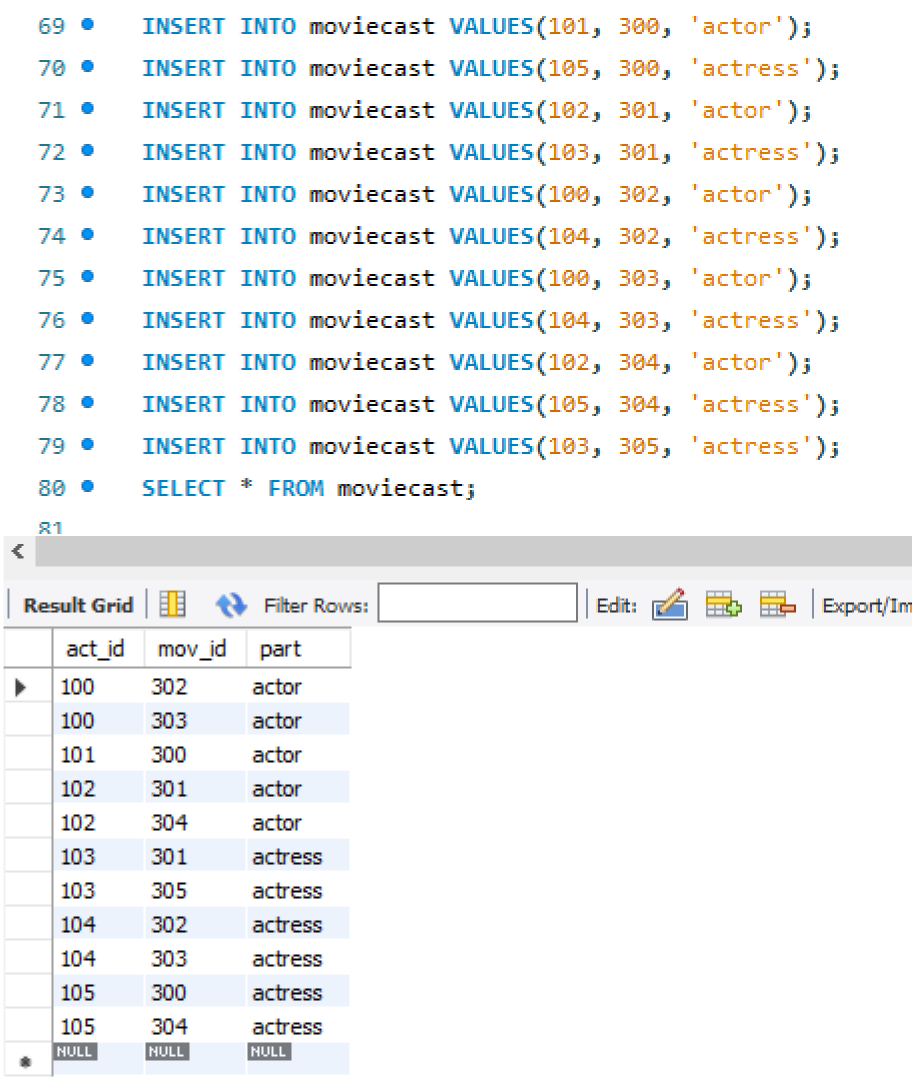
```
INSERT INTO movies VALUES(300, 'Avatar', 2010, 'EN', 202);
INSERT INTO movies VALUES(301, 'Dial M For Murder', 1990, 'EN', 201);
INSERT INTO movies VALUES(302, 'Jurassic Park 1', 1999, 'EN', 200);
INSERT INTO movies VALUES(303, 'Jurassic Park 2', 2017, 'EN', 200);
INSERT INTO movies VALUES(304, 'Vertigo', 1986, 'EN', 201);
INSERT INTO movies VALUES(305, 'Zero Dark Thirty', 2012, 'EN', 200);
SELECT * FROM movies;
```



```
INSERT INTO movies VALUES(300, 'Avatar', 2010, 'EN', 202);
INSERT INTO movies VALUES(301, 'Dial M For Murder', 1990, 'EN', 201);
INSERT INTO movies VALUES(302, 'Jurassic Park 1', 1999, 'EN', 200);
INSERT INTO movies VALUES(303, 'Jurassic Park 2', 2017, 'EN', 200);
INSERT INTO movies VALUES(304, 'Vertigo', 1986, 'EN', 201);
INSERT INTO movies VALUES(305, 'Zero Dark Thirty', 2012, 'EN', 200);
SELECT * FROM movies;
```

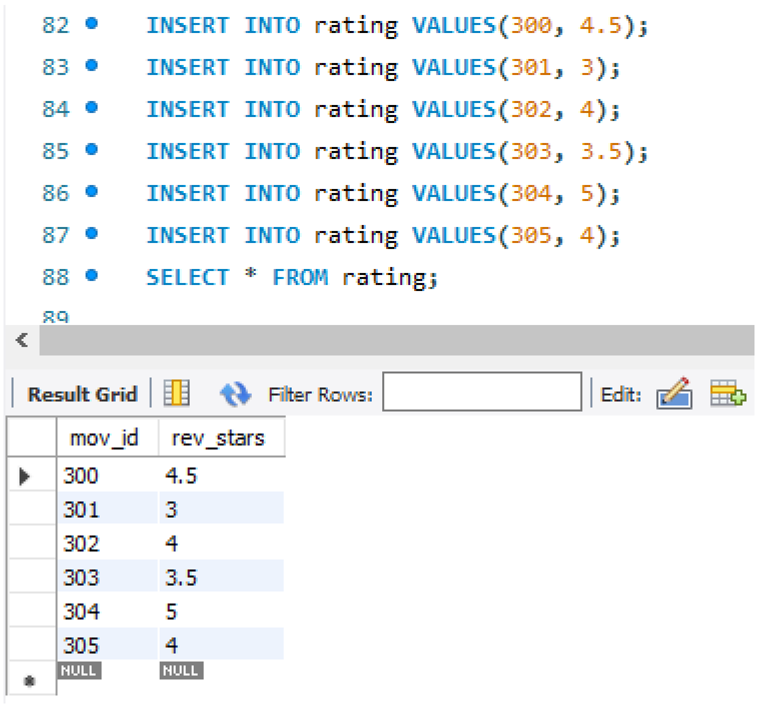
mov_id	mov_title	mov_year	mov_lang	dir_id
300	Avatar	2010	EN	202
301	Dial M For Murder	1990	EN	201
302	Jurassic Park 1	1999	EN	200
303	Jurassic Park 2	2017	EN	200
304	Vertigo	1986	EN	201
305	Zero Dark Thirty	2012	EN	200
NULL	NULL	NULL	NULL	NULL

```
INSERT INTO moviecast VALUES(101, 300, 'actor');
INSERT INTO moviecast VALUES(105, 300, 'actress');
INSERT INTO moviecast VALUES(102, 301, 'actor');
INSERT INTO moviecast VALUES(103, 301, 'actress');
INSERT INTO moviecast VALUES(100, 302, 'actor');
INSERT INTO moviecast VALUES(104, 302, 'actress');
INSERT INTO moviecast VALUES(100, 303, 'actor');
INSERT INTO moviecast VALUES(104, 303, 'actress');
INSERT INTO moviecast VALUES(102, 304, 'actor');
INSERT INTO moviecast VALUES(105, 304, 'actress');
INSERT INTO moviecast VALUES(103, 305, 'actress');
SELECT * FROM moviecast;
```


```
INSERT INTO moviecast VALUES(101, 300, 'actor');
INSERT INTO moviecast VALUES(105, 300, 'actress');
INSERT INTO moviecast VALUES(102, 301, 'actor');
INSERT INTO moviecast VALUES(103, 301, 'actress');
INSERT INTO moviecast VALUES(100, 302, 'actor');
INSERT INTO moviecast VALUES(104, 302, 'actress');
INSERT INTO moviecast VALUES(100, 303, 'actor');
INSERT INTO moviecast VALUES(104, 303, 'actress');
INSERT INTO moviecast VALUES(102, 304, 'actor');
INSERT INTO moviecast VALUES(105, 304, 'actress');
INSERT INTO moviecast VALUES(103, 305, 'actress');
SELECT * FROM moviecast;
```

act_id	mov_id	part
100	302	actor
100	303	actor
101	300	actor
102	301	actor
102	304	actor
103	301	actress
103	305	actress
104	302	actress
104	303	actress
105	300	actress
105	304	actress
NULL	NULL	NULL

INSERT INTO rating VALUES(300, 4.5);
INSERT INTO rating VALUES(301, 3);
INSERT INTO rating VALUES(302, 4);
INSERT INTO rating VALUES(303, 3.5);
INSERT INTO rating VALUES(304, 5);
INSERT INTO rating VALUES(305, 4);
SELECT * FROM rating;



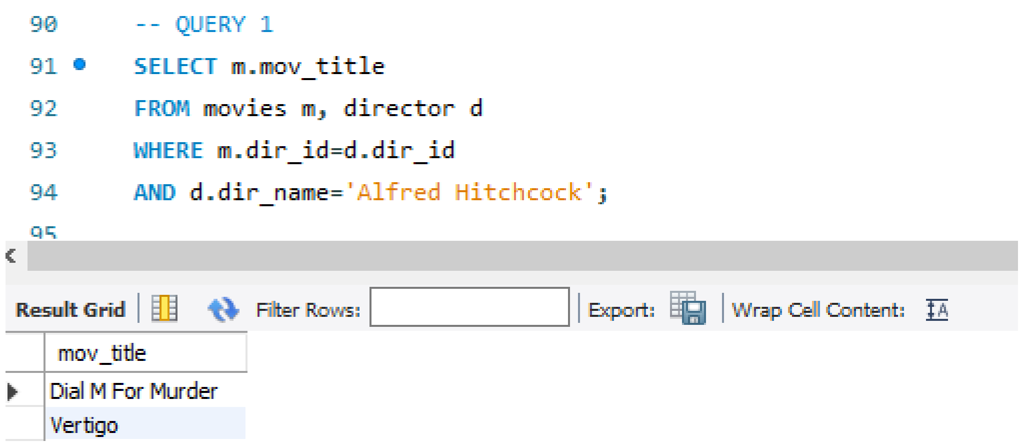
```
INSERT INTO rating VALUES(300, 4.5);
INSERT INTO rating VALUES(301, 3);
INSERT INTO rating VALUES(302, 4);
INSERT INTO rating VALUES(303, 3.5);
INSERT INTO rating VALUES(304, 5);
INSERT INTO rating VALUES(305, 4);
SELECT * FROM rating;
```

mov_id	rev_stars
300	4.5
301	3
302	4
303	3.5
304	5
305	4
NULL	NULL

Write SQL queries to

1. List the titles of all movies directed by ‘Hitchcock’.

SELECT m.mov_title
FROM movies m, director d
WHERE m.dir_id=d.dir_id
AND d.dir_name='Alfred Hitchcock';

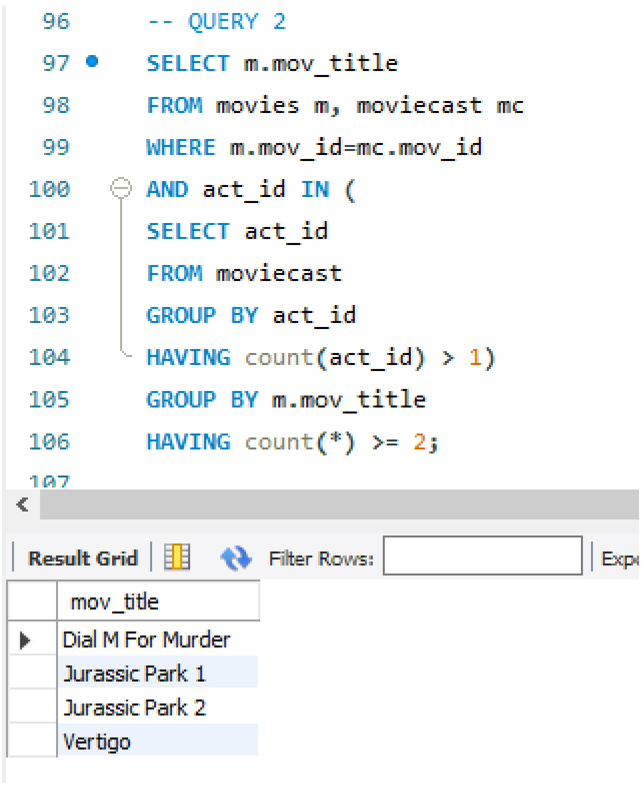


```
-- QUERY 1
SELECT m.mov_title
FROM movies m, director d
WHERE m.dir_id=d.dir_id
AND d.dir_name='Alfred Hitchcock';
```

mov_title
Dial M For Murder
Vertigo

2. Find the movie names where one or more actors acted in two or more movies.

```
SELECT m.mov_title
FROM movies m, moviecast mc
WHERE m.mov_id=mc.mov_id
AND act_id IN (
SELECT act_id
FROM moviecast
GROUP BY act_id
HAVING count(act_id) > 1)
GROUP BY m.mov_title
HAVING count(*) >= 2;
```



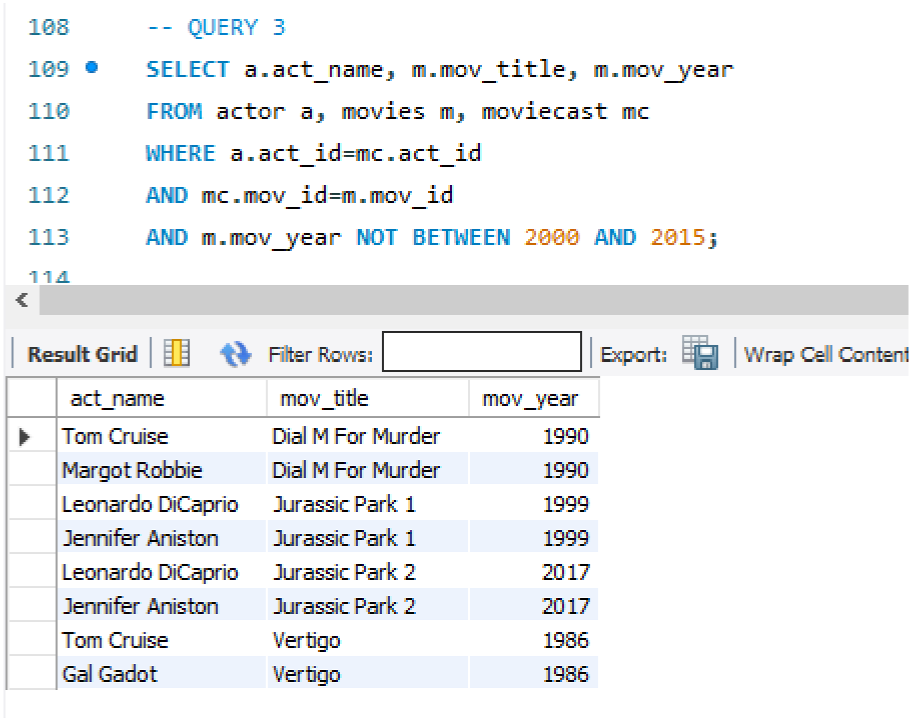
```
-- QUERY 2
SELECT m.mov_title
FROM movies m, moviecast mc
WHERE m.mov_id=mc.mov_id
AND act_id IN (
SELECT act_id
FROM moviecast
GROUP BY act_id
HAVING count(act_id) > 1)
GROUP BY m.mov_title
HAVING count(*) >= 2;
```

mov_title
Dial M For Murder
Jurassic Park 1
Jurassic Park 2
Vertigo

3. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).

```
SELECT a.act_name, m.mov_title, m.mov_year
FROM actor a, movies m, moviecast mc
WHERE a.act_id=mc.act_id
AND mc.mov_id=m.mov_id
```

AND m.mov_year NOT BETWEEN 2000 AND 2015;

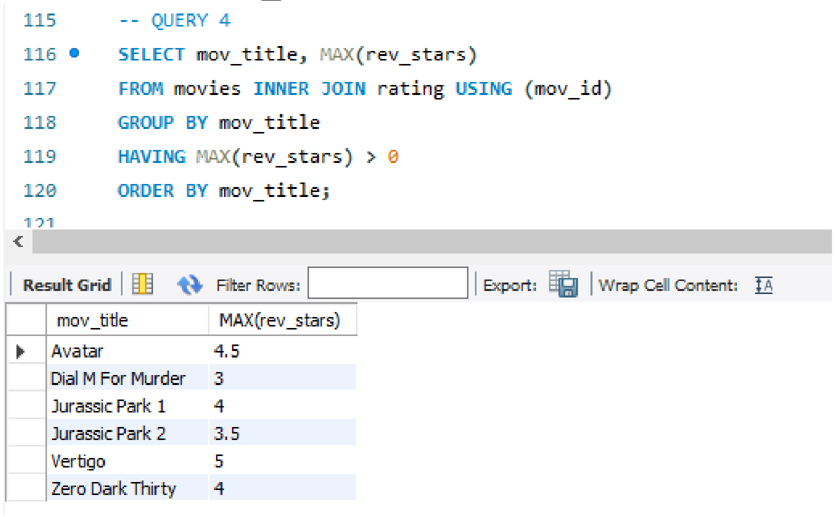


```
-- QUERY 3
SELECT a.act_name, m.mov_title, m.mov_year
FROM actor a, movies m, moviecast mc
WHERE a.act_id=mc.act_id
AND mc.mov_id=m.mov_id
AND m.mov_year NOT BETWEEN 2000 AND 2015;
```

act_name	mov_title	mov_year
Tom Cruise	Dial M For Murder	1990
Margot Robbie	Dial M For Murder	1990
Leonardo DiCaprio	Jurassic Park 1	1999
Jennifer Aniston	Jurassic Park 1	1999
Leonardo DiCaprio	Jurassic Park 2	2017
Jennifer Aniston	Jurassic Park 2	2017
Tom Cruise	Vertigo	1986
Gal Gadot	Vertigo	1986

4. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.

SELECT mov_title, MAX(rev_stars)
FROM movies INNER JOIN rating USING (mov_id)
GROUP BY mov_title
HAVING MAX(rev_stars) > 0
ORDER BY mov_title;

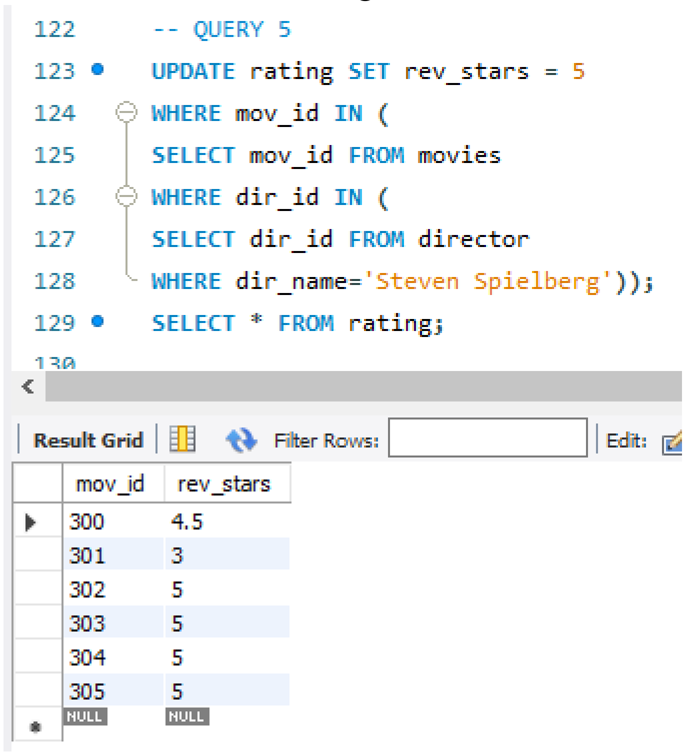


```
-- QUERY 4
SELECT mov_title, MAX(rev_stars)
FROM movies INNER JOIN rating USING (mov_id)
GROUP BY mov_title
HAVING MAX(rev_stars) > 0
ORDER BY mov_title;
```

mov_title	MAX(rev_stars)
Avatar	4.5
Dial M For Murder	3
Jurassic Park 1	4
Jurassic Park 2	3.5
Vertigo	5
Zero Dark Thirty	4

5. Update rating of all movies directed by 'Steven Spielberg' to 5.

```
UPDATE rating SET rev_stars = 5
WHERE mov_id IN (
SELECT mov_id FROM movies
WHERE dir_id IN (
SELECT dir_id FROM director
WHERE dir_name='Steven Spielberg'));
SELECT * FROM rating;
```



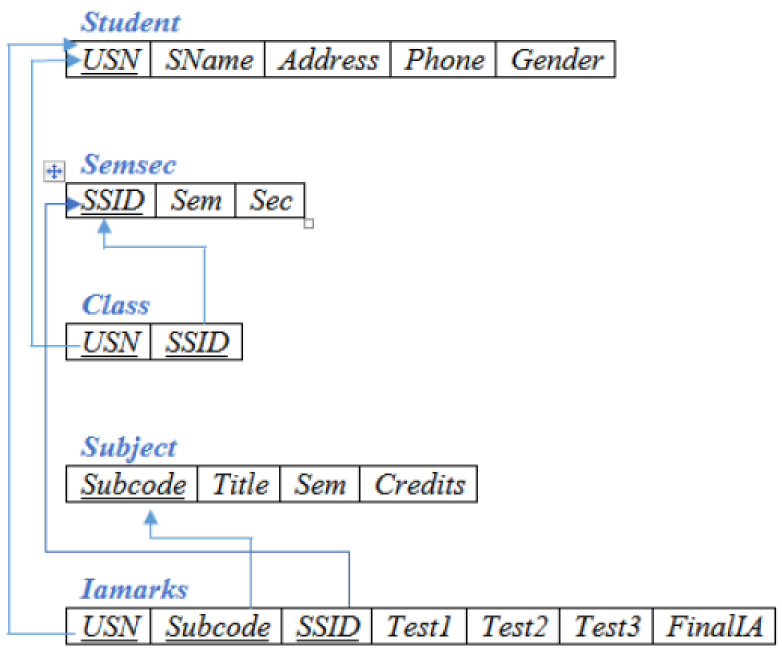
```
-- QUERY 5
UPDATE rating SET rev_stars = 5
WHERE mov_id IN (
SELECT mov_id FROM movies
WHERE dir_id IN (
SELECT dir_id FROM director
WHERE dir_name='Steven Spielberg'));
SELECT * FROM rating;
```

mov_id	rev_stars
300	4.5
301	3
302	5
303	5
304	5
305	5
NULL	NULL

PROGRAM 10: COLLEGE DATABASE

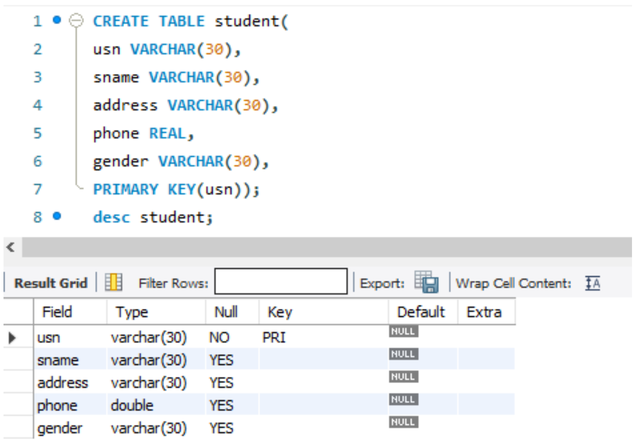
Consider the schema for College Database:
STUDENT (*USN, SName, Address, Phone, Gender*)
SEMSEC (*SSID, Sem, Sec*)
CLASS (*USN, SSID*)
SUBJECT (*Subcode, Title, Sem, Credits*)
IAMARKS (*USN, Subcode, SSID, Test1, Test2, Test3, FinalIA*)

Schema Diagram


Student
USN | SName | Address | Phone | Gender
Semsec
SSID | Sem | Sec
Class
USN | SSID
Subject
Subcode | Title | Sem | Credits
Iamarks
USN | Subcode | SSID | Test1 | Test2 | Test3 | FinalIA

CREATION OF TABLES:

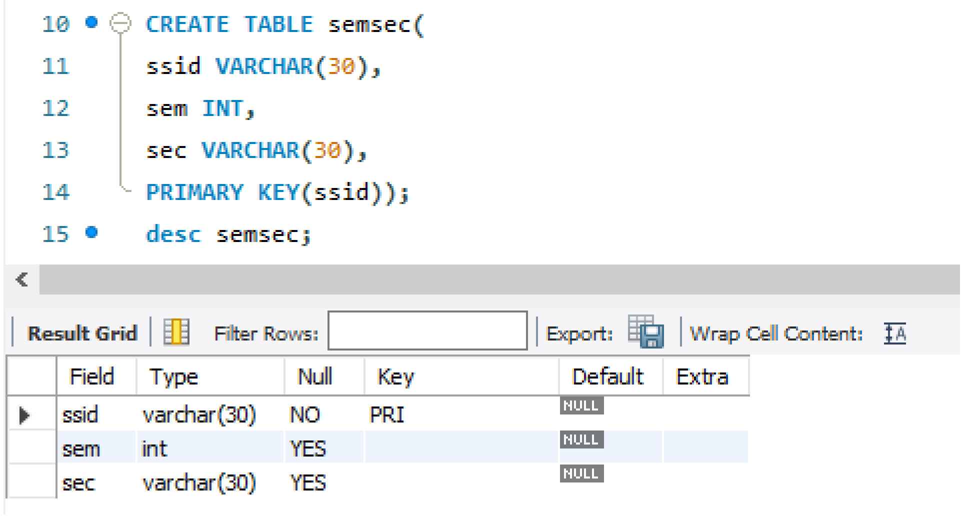
```
CREATE TABLE student(
usn VARCHAR(30),
sname VARCHAR(30),
address VARCHAR(30),
phone REAL,
gender VARCHAR(30),
PRIMARY KEY(usn));
```



```
CREATE TABLE student(
usn VARCHAR(30),
sname VARCHAR(30),
address VARCHAR(30),
phone REAL,
gender VARCHAR(30),
PRIMARY KEY(usn));
desc student;
```

Field	Type	Null	Key	Default	Extra
usn	varchar(30)	NO	PRI	NULL	
sname	varchar(30)	YES		NULL	
address	varchar(30)	YES		NULL	
phone	double	YES		NULL	
gender	varchar(30)	YES		NULL	

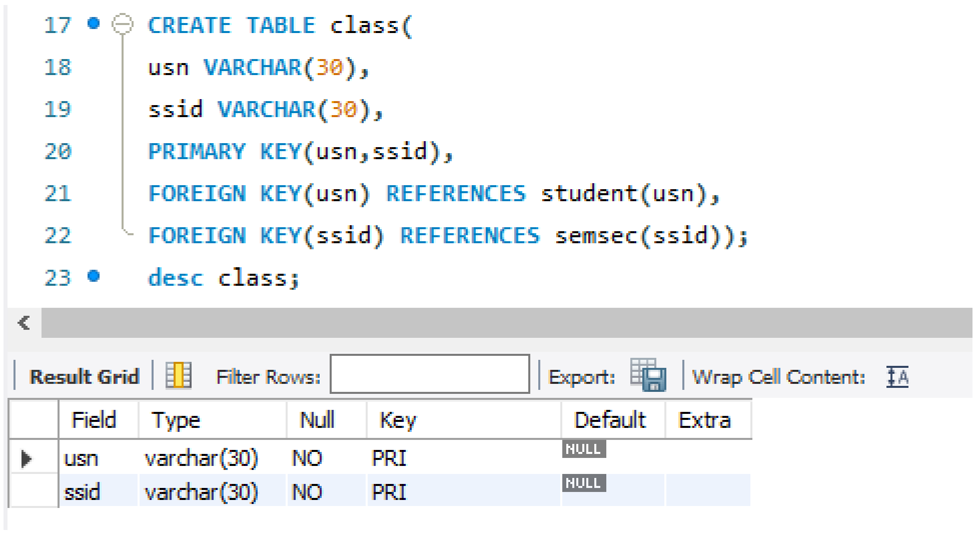
CREATE TABLE semsec(
ssid VARCHAR(30),
sem INT,
sec VARCHAR(30),
PRIMARY KEY(ssid));



```
CREATE TABLE semsec(
ssid VARCHAR(30),
sem INT,
sec VARCHAR(30),
PRIMARY KEY(ssid));
desc semsec;
```

Field	Type	Null	Key	Default	Extra
ssid	varchar(30)	NO	PRI	NULL	
sem	int	YES		NULL	
sec	varchar(30)	YES		NULL	

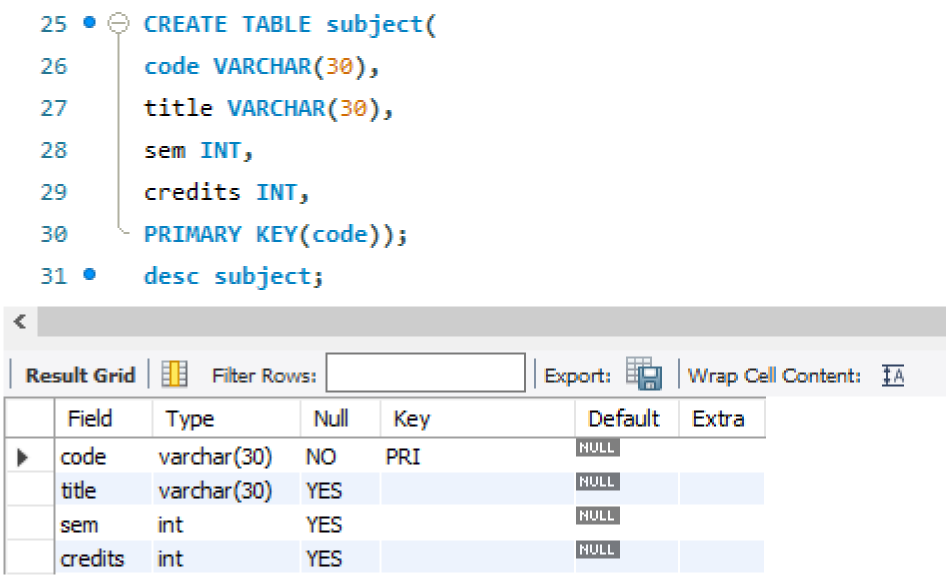
CREATE TABLE class(
usn VARCHAR(30),
ssid VARCHAR(30),
PRIMARY KEY(usn,ssid),
FOREIGN KEY(usn) REFERENCES student(usn),
FOREIGN KEY(ssid) REFERENCES semsec(ssid));



```
CREATE TABLE class(
usn VARCHAR(30),
ssid VARCHAR(30),
PRIMARY KEY(usn,ssid),
FOREIGN KEY(usn) REFERENCES student(usn),
FOREIGN KEY(ssid) REFERENCES semsec(ssid));
desc class;
```

Field	Type	Null	Key	Default	Extra
usn	varchar(30)	NO	PRI	NULL	
ssid	varchar(30)	NO	PRI	NULL	

CREATE TABLE subject(
code VARCHAR(30),
title VARCHAR(30),
sem INT,
credits INT,
PRIMARY KEY(code));

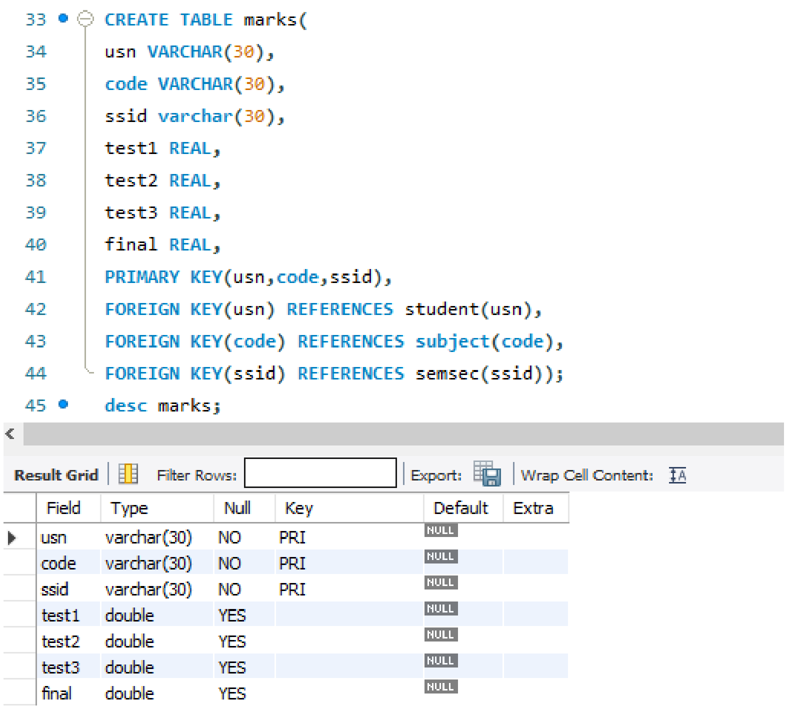


```
CREATE TABLE subject(
code VARCHAR(30),
title VARCHAR(30),
sem INT,
credits INT,
PRIMARY KEY(code));
desc subject;
```

Field	Type	Null	Key	Default	Extra
code	varchar(30)	NO	PRI	NULL	
title	varchar(30)	YES		NULL	
sem	int	YES		NULL	
credits	int	YES		NULL	

CREATE TABLE marks(
usn VARCHAR(30),
code VARCHAR(30),
ssid varchar(30),
test1 REAL,
test2 REAL,
test3 REAL,

```
final REAL,
PRIMARY KEY(usn,code,ssid),
FOREIGN KEY(usn) REFERENCES student(usn),
FOREIGN KEY(code) REFERENCES subject(code),
FOREIGN KEY(ssid) REFERENCES semsec(ssid));
```

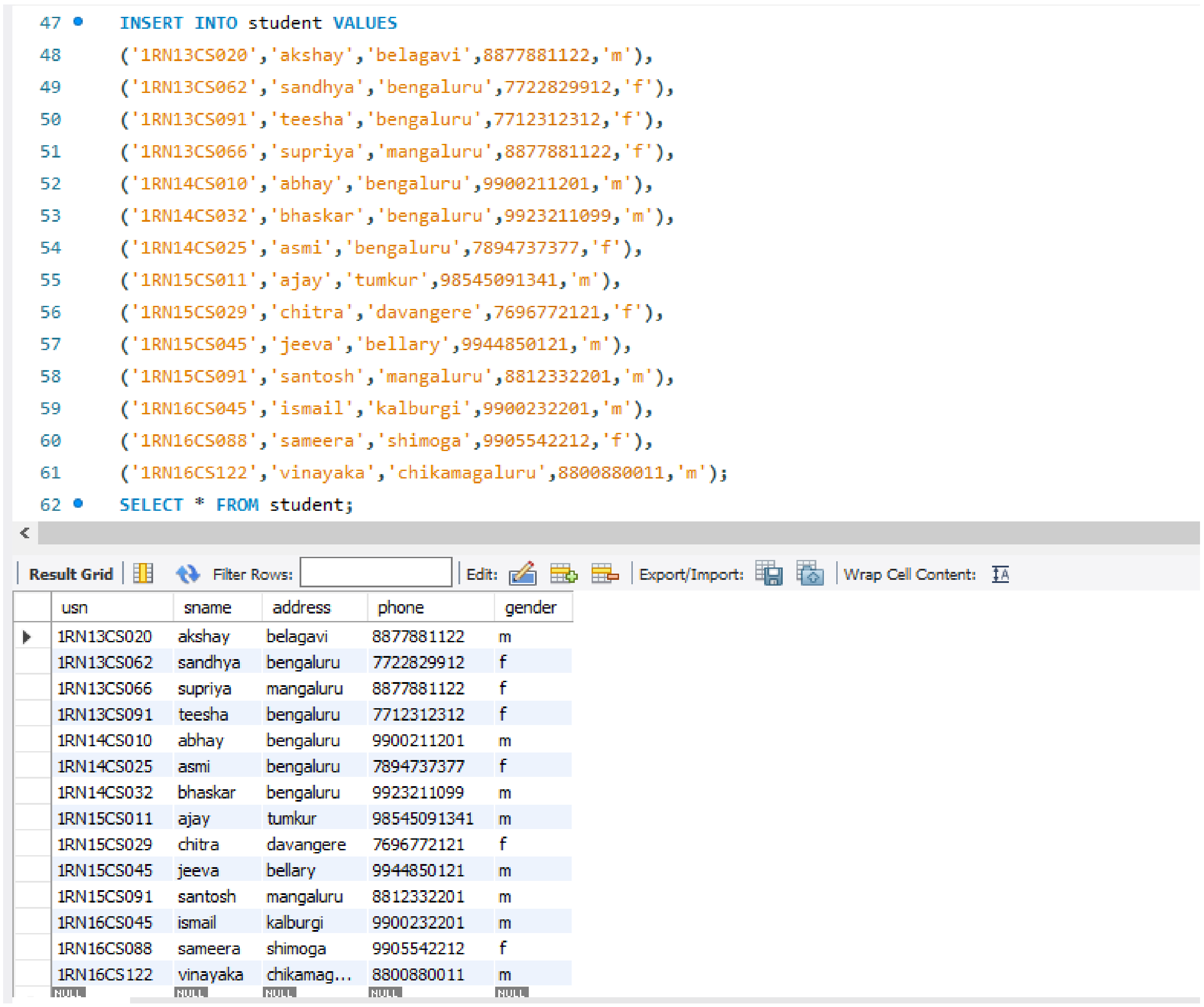


```
CREATE TABLE marks(
usn VARCHAR(30),
code VARCHAR(30),
ssid varchar(30),
test1 REAL,
test2 REAL,
test3 REAL,
final REAL,
PRIMARY KEY(usn,code,ssid),
FOREIGN KEY(usn) REFERENCES student(usn),
FOREIGN KEY(code) REFERENCES subject(code),
FOREIGN KEY(ssid) REFERENCES semsec(ssid));
desc marks;
```

Field	Type	Null	Key	Default	Extra
usn	varchar(30)	NO	PRI	NULL	
code	varchar(30)	NO	PRI	NULL	
ssid	varchar(30)	NO	PRI	NULL	
test1	double	YES		NULL	
test2	double	YES		NULL	
test3	double	YES		NULL	
final	double	YES		NULL	

INSERTING VALUES INTO THE TABLES:

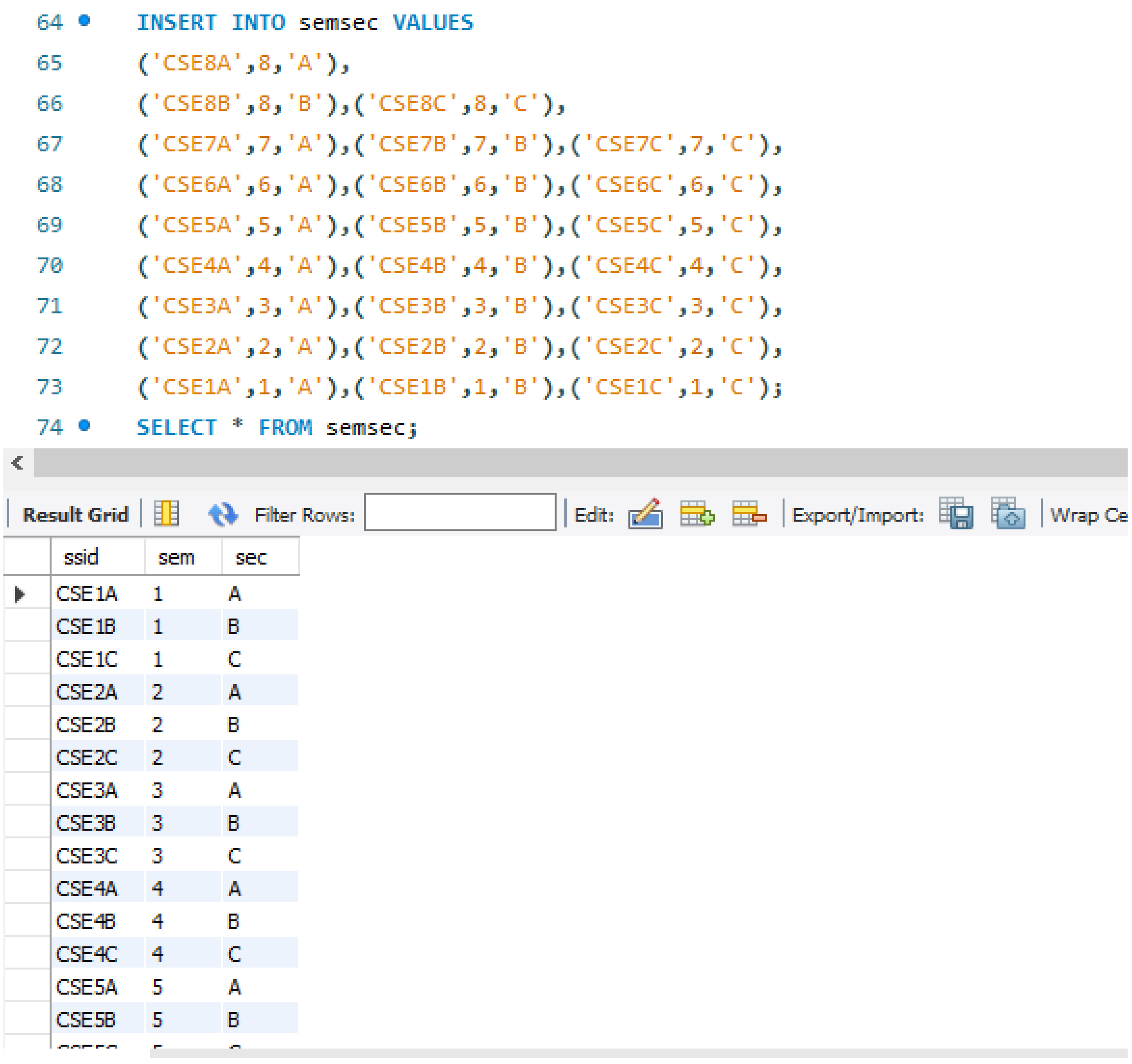
```
INSERT INTO student VALUES
('1RN13CS020','akshay','belagavi',8877881122,'m'),
('1RN13CS062','sandhya','bengaluru',7722829912,'f'),
('1RN13CS091','teesha','bengaluru',7712312312,'f'),
('1RN13CS066','supriya','mangaluru',8877881122,'f'),
('1RN14CS010','abhay','bengaluru',9900211201,'m'),
('1RN14CS032','bhaskar','bengaluru',9923211099,'m'),
('1RN14CS025','asmi','bengaluru',7894737377,'f'),
('1RN15CS011','ajay','tumkur',98545091341,'m'),
('1RN15CS029','chitra','davangere',7696772121,'f'),
('1RN15CS045','jeeva','bellary',9944850121,'m'),
('1RN15CS091','santosh','mangaluru',8812332201,'m'),
('1RN16CS045','ismail','kalburgi',9900232201,'m'),
('1RN16CS088','sameera','shimoga',9905542212,'f'),
('1RN16CS122','vinayaka','chikamagaluru',8800880011,'m');
SELECT * FROM student;
```



```sql
INSERT INTO student VALUES
('1RN13CS020','akshay','belagavi',8877881122,'m'),
('1RN13CS062','sandhya','bengaluru',7722829912,'f'),
('1RN13CS091','teesha','bengaluru',7712312312,'f'),
('1RN13CS066','supriya','mangaluru',8877881122,'f'),
('1RN14CS010','abhay','bengaluru',9900211201,'m'),
('1RN14CS032','bhaskar','bengaluru',9923211099,'m'),
('1RN14CS025','asmi','bengaluru',7894737377,'f'),
('1RN15CS011','ajay','tumkur',98545091341,'m'),
('1RN15CS029','chitra','davangere',7696772121,'f'),
('1RN15CS045','jeeva','bellary',9944850121,'m'),
('1RN15CS091','santosh','mangaluru',8812332201,'m'),
('1RN16CS045','ismail','kalburgi',9900232201,'m'),
('1RN16CS088','sameera','shimoga',9905542212,'f'),
('1RN16CS122','vinayaka','chikamagaluru',8800880011,'m');
SELECT * FROM student;
```

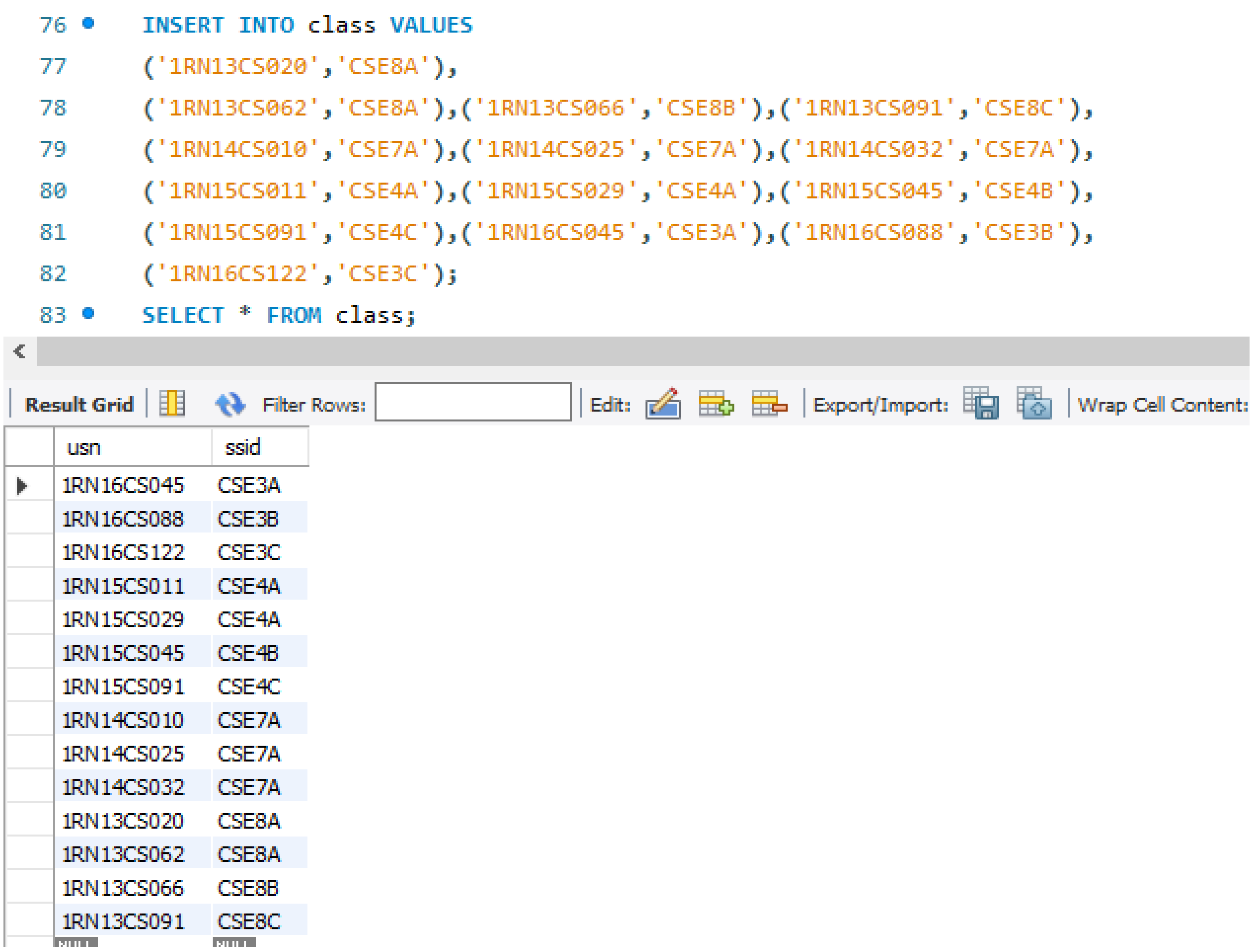
usn	sname	address	phone	gender
1RN13CS020	akshay	belagavi	8877881122	m
1RN13CS062	sandhya	bengaluru	7722829912	f
1RN13CS066	supriya	mangaluru	8877881122	f
1RN13CS091	teesha	bengaluru	7712312312	f
1RN14CS010	abhay	bengaluru	9900211201	m
1RN14CS025	asmi	bengaluru	7894737377	f
1RN14CS032	bhaskar	bengaluru	9923211099	m
1RN15CS011	ajay	tumkur	98545091341	m
1RN15CS029	chitra	davangere	7696772121	f
1RN15CS045	jeeva	bellary	9944850121	m
1RN15CS091	santosh	mangaluru	8812332201	m
1RN16CS045	ismail	kalburgi	9900232201	m
1RN16CS088	sameera	shimoga	9905542212	f
1RN16CS122	vinayaka	chikamag...	8800880011	m
NULL	NULL	NULL	NULL	NULL

INSERT INTO semsec VALUES
('CSE8A',8,'A'),
('CSE8B',8,'B'),('CSE8C',8,'C'),
('CSE7A',7,'A'),('CSE7B',7,'B'),('CSE7C',7,'C'),
('CSE6A',6,'A'),('CSE6B',6,'B'),('CSE6C',6,'C'),
('CSE5A',5,'A'),('CSE5B',5,'B'),('CSE5C',5,'C'),
('CSE4A',4,'A'),('CSE4B',4,'B'),('CSE4C',4,'C'),
('CSE3A',3,'A'),('CSE3B',3,'B'),('CSE3C',3,'C'),
('CSE2A',2,'A'),('CSE2B',2,'B'),('CSE2C',2,'C'),
('CSE1A',1,'A'),('CSE1B',1,'B'),('CSE1C',1,'C');
SELECT * FROM semsec;


```
INSERT INTO semsec VALUES
('CSE8A',8,'A'),
('CSE8B',8,'B'),('CSE8C',8,'C'),
('CSE7A',7,'A'),('CSE7B',7,'B'),('CSE7C',7,'C'),
('CSE6A',6,'A'),('CSE6B',6,'B'),('CSE6C',6,'C'),
('CSE5A',5,'A'),('CSE5B',5,'B'),('CSE5C',5,'C'),
('CSE4A',4,'A'),('CSE4B',4,'B'),('CSE4C',4,'C'),
('CSE3A',3,'A'),('CSE3B',3,'B'),('CSE3C',3,'C'),
('CSE2A',2,'A'),('CSE2B',2,'B'),('CSE2C',2,'C'),
('CSE1A',1,'A'),('CSE1B',1,'B'),('CSE1C',1,'C');
SELECT * FROM semsec;
```

ssid	sem	sec
CSE1A	1	A
CSE1B	1	B
CSE1C	1	C
CSE2A	2	A
CSE2B	2	B
CSE2C	2	C
CSE3A	3	A
CSE3B	3	B
CSE3C	3	C
CSE4A	4	A
CSE4B	4	B
CSE4C	4	C
CSE5A	5	A
CSE5B	5	B

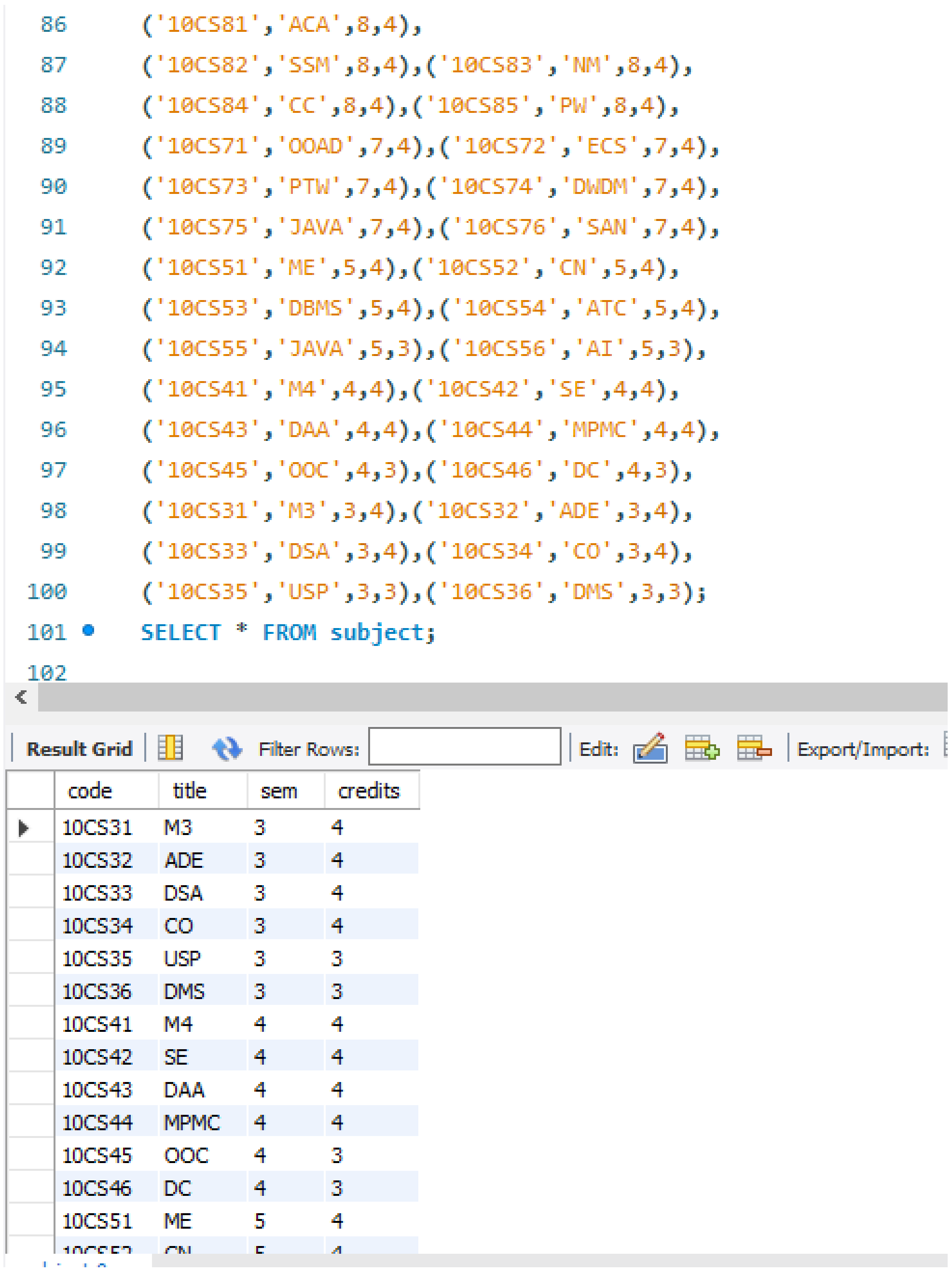
```
INSERT INTO class VALUES
('1RN13CS020','CSE8A'),
('1RN13CS062','CSE8A'),('1RN13CS066','CSE8B'),('1RN13CS091','CSE8C'),
('1RN14CS010','CSE7A'),('1RN14CS025','CSE7A'),('1RN14CS032','CSE7A'),
('1RN15CS011','CSE4A'),('1RN15CS029','CSE4A'),('1RN15CS045','CSE4B'),
('1RN15CS091','CSE4C'),('1RN16CS045','CSE3A'),('1RN16CS088','CSE3B'),
('1RN16CS122','CSE3C');
SELECT * FROM class;
```



```
INSERT INTO class VALUES
('1RN13CS020','CSE8A'),
('1RN13CS062','CSE8A'),('1RN13CS066','CSE8B'),('1RN13CS091','CSE8C'),
('1RN14CS010','CSE7A'),('1RN14CS025','CSE7A'),('1RN14CS032','CSE7A'),
('1RN15CS011','CSE4A'),('1RN15CS029','CSE4A'),('1RN15CS045','CSE4B'),
('1RN15CS091','CSE4C'),('1RN16CS045','CSE3A'),('1RN16CS088','CSE3B'),
('1RN16CS122','CSE3C');
SELECT * FROM class;
```

usn	ssid
1RN16CS045	CSE3A
1RN16CS088	CSE3B
1RN16CS122	CSE3C
1RN15CS011	CSE4A
1RN15CS029	CSE4A
1RN15CS045	CSE4B
1RN15CS091	CSE4C
1RN14CS010	CSE7A
1RN14CS025	CSE7A
1RN14CS032	CSE7A
1RN13CS020	CSE8A
1RN13CS062	CSE8A
1RN13CS066	CSE8B
1RN13CS091	CSE8C

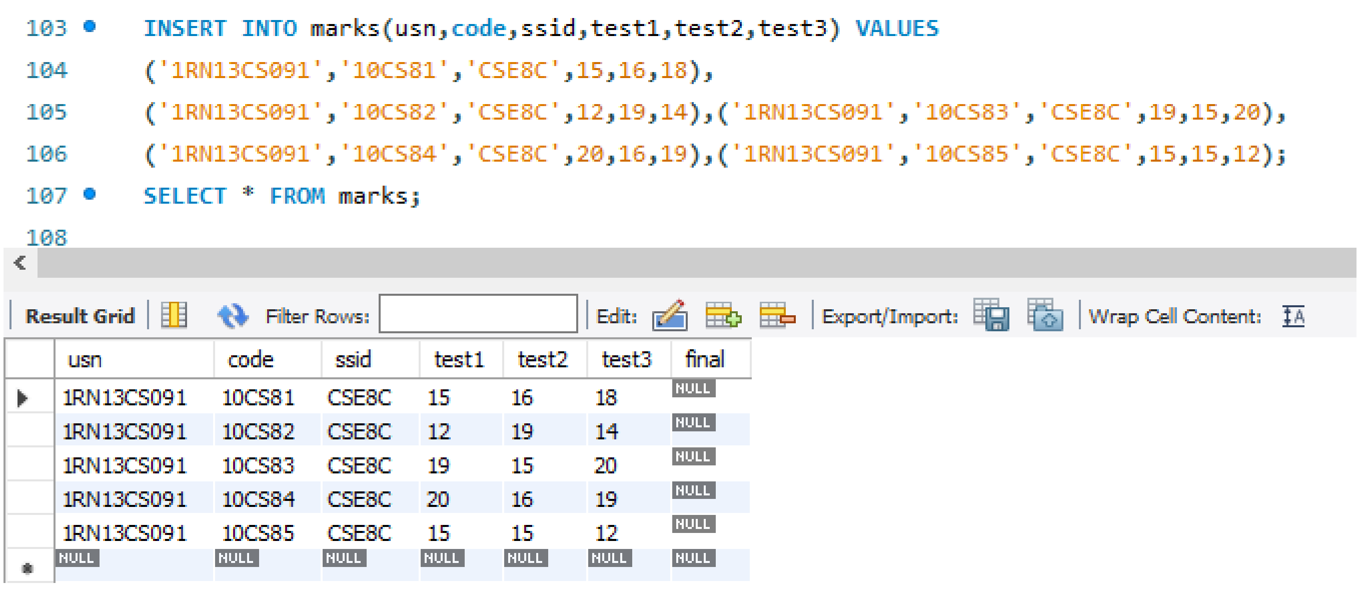
```
INSERT INTO subject VALUES
('10CS81','ACA',8,4),
('10CS82','SSM',8,4),('10CS83','NM',8,4),
('10CS84','CC',8,4),('10CS85','PW',8,4),
('10CS71','OOAD',7,4),('10CS72','ECS',7,4),
('10CS73','PTW',7,4),('10CS74','DWDM',7,4),
('10CS75','JAVA',7,4),('10CS76','SAN',7,4),
('10CS51','ME',5,4),('10CS52','CN',5,4),
('10CS53','DBMS',5,4),('10CS54','ATC',5,4),
('10CS55','JAVA',5,3),('10CS56','AI',5,3),
('10CS41','M4',4,4),('10CS42','SE',4,4),
('10CS43','DAA',4,4),('10CS44','MPMC',4,4),
('10CS45','OOC',4,3),('10CS46','DC',4,3),
('10CS31','M3',3,4),('10CS32','ADE',3,4),
('10CS33','DSA',3,4),('10CS34','CO',3,4),
('10CS35','USP',3,3),('10CS36','DMS',3,3);
SELECT * FROM subject;
```



```
    ('10CS81','ACA',8,4),
    ('10CS82','SSM',8,4),('10CS83','NM',8,4),
    ('10CS84','CC',8,4),('10CS85','PW',8,4),
    ('10CS71','OOAD',7,4),('10CS72','ECS',7,4),
    ('10CS73','PTW',7,4),('10CS74','DWDM',7,4),
    ('10CS75','JAVA',7,4),('10CS76','SAN',7,4),
    ('10CS51','ME',5,4),('10CS52','CN',5,4),
    ('10CS53','DBMS',5,4),('10CS54','ATC',5,4),
    ('10CS55','JAVA',5,3),('10CS56','AI',5,3),
    ('10CS41','M4',4,4),('10CS42','SE',4,4),
    ('10CS43','DAA',4,4),('10CS44','MPMC',4,4),
    ('10CS45','OOC',4,3),('10CS46','DC',4,3),
    ('10CS31','M3',3,4),('10CS32','ADE',3,4),
    ('10CS33','DSA',3,4),('10CS34','CO',3,4),
    ('10CS35','USP',3,3),('10CS36','DMS',3,3);
SELECT * FROM subject;
```

code	title	sem	credits
10CS31	M3	3	4
10CS32	ADE	3	4
10CS33	DSA	3	4
10CS34	CO	3	4
10CS35	USP	3	3
10CS36	DMS	3	3
10CS41	M4	4	4
10CS42	SE	4	4
10CS43	DAA	4	4
10CS44	MPMC	4	4
10CS45	OOC	4	3
10CS46	DC	4	3
10CS51	ME	5	4
10CS52	CN	5	4

INSERT INTO marks(usn,code,ssid,test1,test2,test3) VALUES
('1RN13CS091','10CS81','CSE8C',15,16,18),
('1RN13CS091','10CS82','CSE8C',12,19,14),('1RN13CS091','10CS83','CSE8C',19,15,20),
('1RN13CS091','10CS84','CSE8C',20,16,19),('1RN13CS091','10CS85','CSE8C',15,15,12);
SELECT * FROM marks;



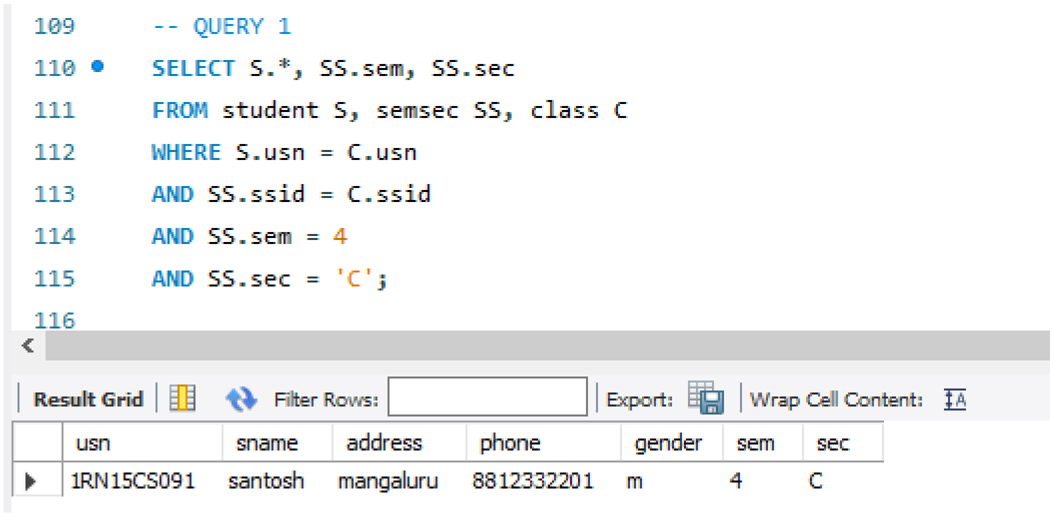
```sql
INSERT INTO marks(usn,code,ssid,test1,test2,test3) VALUES
('1RN13CS091','10CS81','CSE8C',15,16,18),
('1RN13CS091','10CS82','CSE8C',12,19,14),('1RN13CS091','10CS83','CSE8C',19,15,20),
('1RN13CS091','10CS84','CSE8C',20,16,19),('1RN13CS091','10CS85','CSE8C',15,15,12);
SELECT * FROM marks;
```

usn	code	ssid	test1	test2	test3	final
1RN13CS091	10CS81	CSE8C	15	16	18	NULL
1RN13CS091	10CS82	CSE8C	12	19	14	NULL
1RN13CS091	10CS83	CSE8C	19	15	20	NULL
1RN13CS091	10CS84	CSE8C	20	16	19	NULL
1RN13CS091	10CS85	CSE8C	15	15	12	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Write SQL queries to

1. List all the student details studying in fourth semester ‘C’ section.

SELECT S.*, SS.sem, SS.sec
FROM student S, semsec SS, class C
WHERE S.usn = C.usn
AND SS.ssid = C.ssid
AND SS.sem = 4
AND SS.sec = 'C';

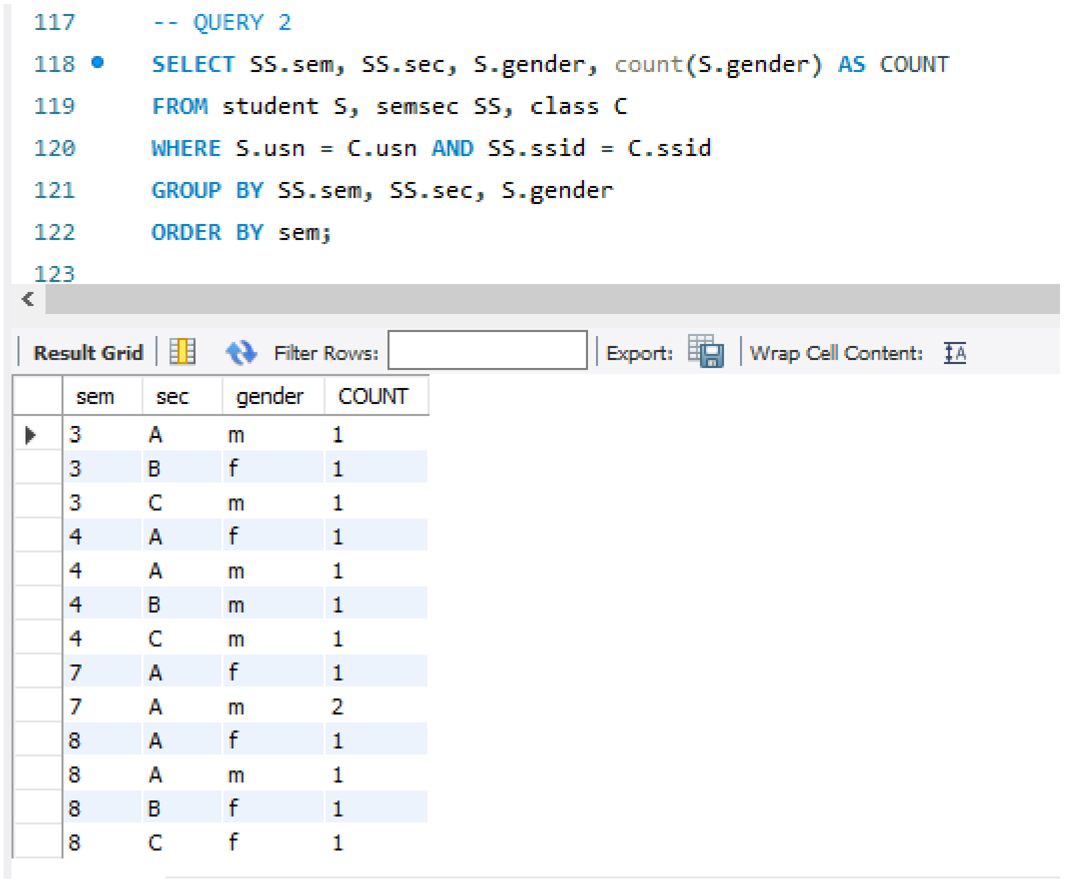


```sql
-- QUERY 1
SELECT S.*, SS.sem, SS.sec
FROM student S, semsec SS, class C
WHERE S.usn = C.usn
AND SS.ssid = C.ssid
AND SS.sem = 4
AND SS.sec = 'C';
```

usn	sname	address	phone	gender	sem	sec
1RN15CS091	santosh	mangaluru	8812332201	m	4	C

2. Compute the total number of male and female students in each semester and in each section.

```
SELECT SS.sem, SS.sec, S.gender, count(S.gender) AS COUNT
FROM student S, semsec SS, class C
WHERE S.usn = C.usn AND SS.ssid = C.ssid
GROUP BY SS.sem, SS.sec, S.gender
ORDER BY sem;
```

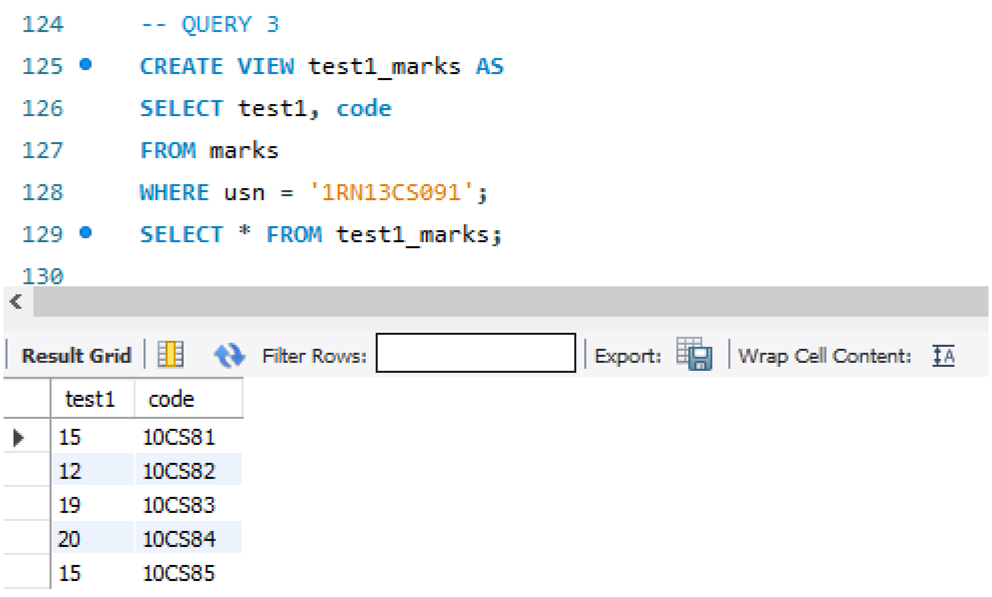


```
-- QUERY 2
SELECT SS.sem, SS.sec, S.gender, count(S.gender) AS COUNT
FROM student S, semsec SS, class C
WHERE S.usn = C.usn AND SS.ssid = C.ssid
GROUP BY SS.sem, SS.sec, S.gender
ORDER BY sem;
```

sem	sec	gender	COUNT
3	A	m	1
3	B	f	1
3	C	m	1
4	A	f	1
4	A	m	1
4	B	m	1
4	C	m	1
7	A	f	1
7	A	m	2
8	A	f	1
8	A	m	1
8	B	f	1
8	C	f	1

3. Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.

```
CREATE VIEW test1_marks AS
SELECT test1, code
FROM marks
WHERE usn = '1RN13CS091';
SELECT * FROM test1_marks;
```

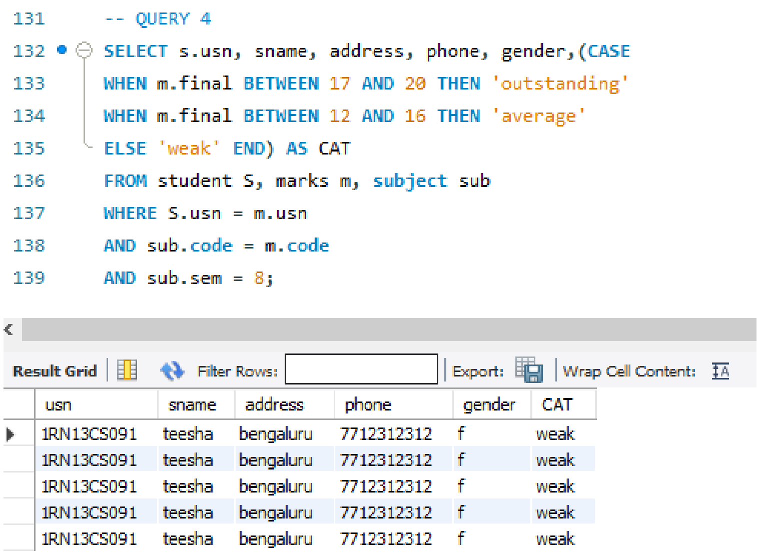


```
-- QUERY 3
CREATE VIEW test1_marks AS
SELECT test1, code
FROM marks
WHERE usn = '1RN13CS091';
SELECT * FROM test1_marks;
```

test1	code
15	10CS81
12	10CS82
19	10CS83
20	10CS84
15	10CS85

4. Categorize students based on the following criterion:
If FinalIA = 17 to 20 then CAT = ‘Outstanding’
If FinalIA = 12 to 16 then CAT = ‘Average’
If FinalIA< 12 then CAT = ‘Weak’
Give these details only for 8th semester A, B, and C section students.

```
SELECT s.usn, sname, address, phone, gender,(CASE
WHEN m.final BETWEEN 17 AND 20 THEN 'outstanding'
WHEN m.final BETWEEN 12 AND 16 THEN 'average'
ELSE 'weak' END) AS CAT
FROM student S, marks m, subject sub
WHERE S.usn = m.usn
AND sub.code = m.code
AND sub.sem = 8;
```



```
-- QUERY 4
SELECT s.usn, sname, address, phone, gender,(CASE
WHEN m.final BETWEEN 17 AND 20 THEN 'outstanding'
WHEN m.final BETWEEN 12 AND 16 THEN 'average'
ELSE 'weak' END) AS CAT
FROM student S, marks m, subject sub
WHERE S.usn = m.usn
AND sub.code = m.code
AND sub.sem = 8;
```

usn	sname	address	phone	gender	CAT
1RN13CS091	teesha	bengaluru	7712312312	f	weak
1RN13CS091	teesha	bengaluru	7712312312	f	weak
1RN13CS091	teesha	bengaluru	7712312312	f	weak
1RN13CS091	teesha	bengaluru	7712312312	f	weak
1RN13CS091	teesha	bengaluru	7712312312	f	weak