PROGRAM 6. ORDER PROCESSING DATABASE

Consider the following relations for an Order Processing database application in a company.

CUSTOMER (CUST #: int, cname: String, city: String)

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

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

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

WAREHOUSE (warehouse #: int, city: String)

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

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

```
CREATE TABLE `order_processing`.`customer` ( `cust` INT NOT NULL , `cnam
e` VARCHAR(50) NOT NULL , `city` VARCHAR(50) NOT NULL , PRIMARY KEY (`cu
st`));
```

```
CREATE TABLE `order_processing`.`item` ( `item#` INT NOT NULL , `unit-
price` INT NOT NULL , PRIMARY KEY (`item#`));
```

```
CREATE TABLE `order_processing`.`warehouse` ( `warehouse#` INT NOT NULL
, `city` VARCHAR(50) NOT NULL , PRIMARY KEY (`warehouse#`)) ;
```

```
CREATE TABLE `order_processing`.`order_item` ( `orderno` INT NOT NULL ,
`itemno` INT NOT NULL , `qty` INT NOT NULL , PRIMARY KEY (`orderno`));
```

ALTER TABLE order_item ADD FOREIGN KEY (itemno) REFERENCES item(itemno);

CREATE TABLE `order_processing`.`orders` (`orderno` INT NOT NULL , `odate` DATE NOT NULL , `custno` INT NOT NULL , `ord_amt` INT NOT NULL , PRIMARY KEY (`orderno`)) ENGINE = InnoDB;

ALTER TABLE orders ADD FOREIGN KEY(custno) REFERENCES customer(cust)

```
CREATE TABLE `order_processing`.`shipment` ( `orderno` INT NOT NULL , `w
arehouseno` INT NOT NULL , `ship_date` DATE NOT NULL , PRIMARY KEY (`ord
erno`));
```

```
ALTER TABLE shipment ADD FOREIGN KEY (warehouseno) REFERENCES
warehouse(warehouseno);
```

ii. Enter at least five tuples for each relation.

```
INSERT INTO `customer` (`cust`, `cname`, `city`) VALUES ('501', 'rudra',
 'bengaluru'), ('502', 'darshana', 'kolar'), ('503', 'rajiv', 'mandya'),
 ('504', 'kavita', 'mysuru'), ('505', 'shekhar', 'bengaluru')
```

```
SELECT * FROM `customer`
```

Show all | Number of rows: 25 Filter rows

+ Options

	cust	cname	city
Edit Copy Delete	501	rudra	bengaluru
Edit Copy Delete	502	darshana	kolar
Edit Copy Delete	503	rajiv	mandya
Edit Copy Delete	504	kavita	mysuru
Edit Copy Delete	505	shekhar	bengaluru

```
INSERT INTO `item` (`itemno`, `unit-
price`) VALUES ('1001', '150'), ('1002', '199'), ('1003', '450'), ('1004
', '700'), ('1005', '99')
```

```
SELECT * FROM `item`
```

Show all | Number of rows: 25 Filte

+ Options

	itemno	unit-price
Edit Copy Delete	1001	150
Edit Copy Delete	1002	199
Edit Copy Delete	1003	450
Edit Copy Delete	1004	700
Edit Copy Delete	1005	99

```
INSERT INTO `warehouse` (`warehouseno`, `city`) VALUES ('101', 'bengalur
u'), ('102', 'kolar'), ('103', 'mysuru'), ('104', 'bengaluru'), ('105',
'mysuru')
```

```
SELECT * FROM `warehouse`
```

Show all | Number of rows: 25 Filter rows

+ Options

	warehouseno	city
Edit Copy Delete	101	bengaluru
Edit Copy Delete	102	kolar
Edit Copy Delete	103	mysuru
Edit Copy Delete	104	bengaluru
Edit Copy Delete	105	mysuru

```
INSERT INTO `orders` (`orderno`, `odate`, `custno`, `ord_amt`) VALUES ('1', '2021-05-05', '501', '30'), ('2', '2021-05-09', '504', '100'), ('3', '2021-06-03', '503', '500')
```

```
SELECT * FROM `orders`
```

Show all | Number of rows: 25 Filter rows: Search this ta

+ Options

	orderno	odate	custno	ord_amt
Edit Copy Delete	1	2021-05-05	501	30
Edit Copy Delete	2	2021-05-09	504	100
Edit Copy Delete	3	2021-06-03	503	500

```
INSERT INTO `order_item` (`orderno`, `itemno`, `qty`) VALUES ('1', '1005', '100'), ('2', '1002', '520'), ('3', '1004', '2000')
```

```
SELECT * FROM `order_item`
```

Show all | Number of rows: 25 Filter rows

+ Options

	orderno	itemno	qty
Edit Copy Delete	1	1005	100
Edit Copy Delete	2	1002	520
Edit Copy Delete	3	1004	2000

```
INSERT INTO `shipment` (`orderno`, `warehouseno`, `ship_date`) VALUES ('1', '101', '2021-05-10'), ('2', '103', '2021-05-19'), ('3', '104', '2021-06-05')
```

```
SELECT * FROM `shipment`
```

Show all | Number of rows: 25 Filter rows: Search th

+ Options

	orderno	warehouseno	ship_date
Edit Copy Delete	1	101	2021-05-10
Edit Copy Delete	2	103	2021-05-19
Edit Copy Delete	3	104	2021-06-05

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

SELECT c.cname, COUNT(o.orderno) AS nooforders ,AVG(o.ord_amt) AS AVG_order_amt FROM customer c ,orders o WHERE c.cust=o.custno GROUP BY c.cname

Showing rows 0 - 2 (3 total, Query took 0.0036 seconds.)

```
SELECT c.cname, COUNT(o.orderno) AS nooforders ,AVG(o.ord_amt) AS AVG_order_amt FROM customer c ,orders o WHERE c.cust=o.custno GROUP BY c.cname
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

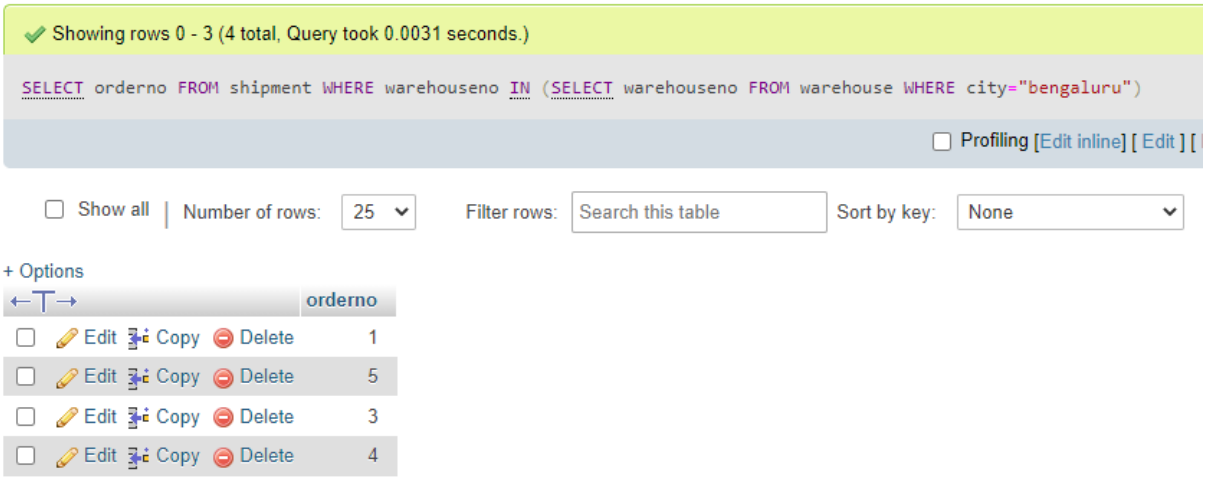
Show all | Number of rows: 25 Filter rows: Search this table

+ Options

cname	nooforders	AVG_order_amt
kavita	1	100.0000
rajiv	1	500.0000
rudra	1	30.0000

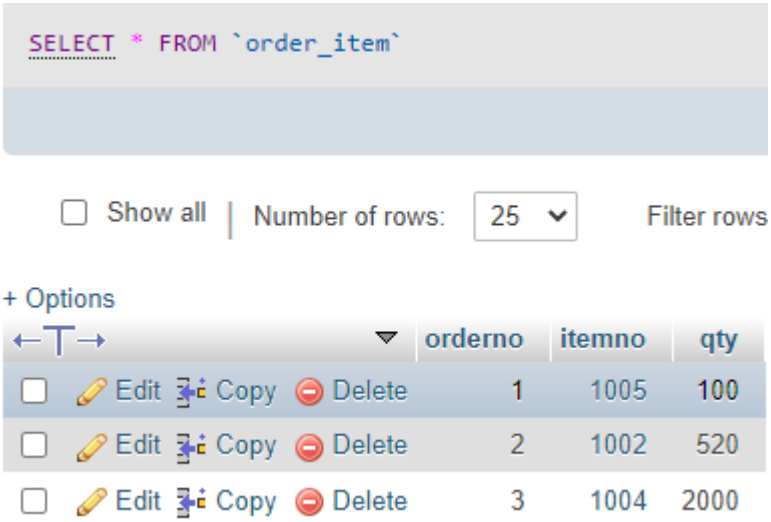
iii. List the order# for orders that were shipped from all warehouses that the company has in a specific city.

SELECT orderno FROM shipment WHERE warehouseno IN (SELECT warehouseno FROM warehouse WHERE city="bengaluru")

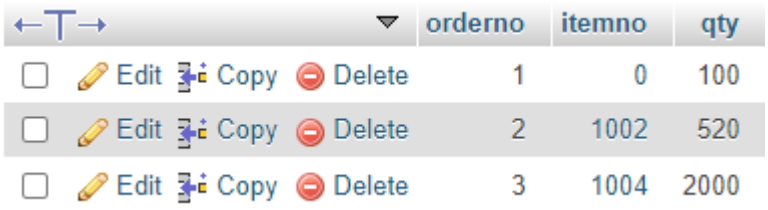


iv. Demonstrate how you delete item# 10 from the ITEM table and make that field null in the ORDER_ITEM table.

ALTER TABLE order_item ADD FOREIGN KEY (itemno) REFERENCES item(itemno) ON DELETE SET NULL ;



DELETE FROM `item` WHERE `item`.`itemno` = 1005"



PROGRAM 7. BOOK DEALER DATABASE

The following tables are maintained by a book dealer:

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

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

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

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

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

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

- create table author (

author_id INT,

author_name VARCHAR(20),

author_city VARCHAR(20),

author_country VARCHAR(20),

PRIMARY KEY(author_id));

- create table publisher (

publisher_id INT,

publisher_name VARCHAR(20),

publisher_city VARCHAR(20),

publisher_country VARCHAR(20),

PRIMARY KEY(publisher_id));

- create table category (

category_id INT,

description VARCHAR(30),

PRIMARY KEY(category_id));

- CREATE TABLE catalog(

book_id INT,

book_title VARCHAR(30),

author_id INT,

publisher_id INT,

category_id INT,

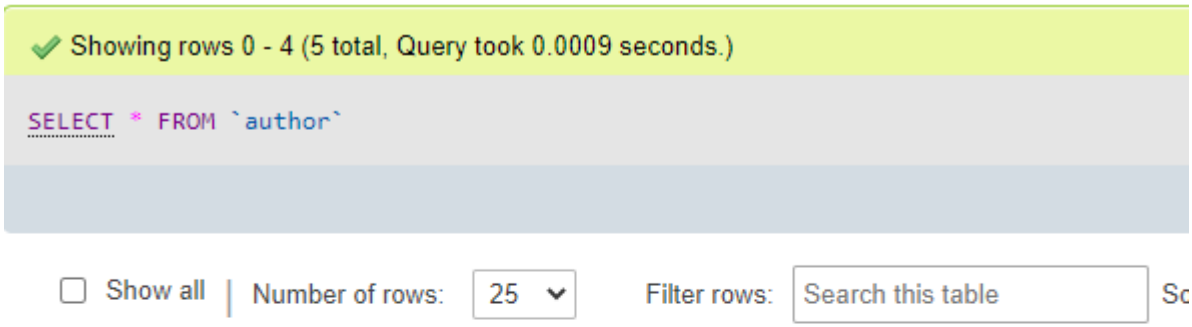
```
year INT,
price INT,
PRIMARY KEY(book_id),
FOREIGN KEY(author_id) REFERENCES author(author_id)ON UPDATE CASCADE,
FOREIGN KEY(publisher_id) REFERENCES publisher(publisher_id)ON UPDATE CASCADE,
FOREIGN KEY(category_id) REFERENCES category(category_id)ON UPDATE CASCADE );
```

- CREATE TABLE orderdetails(

```
order_id INT,
book_id INT,
quantity INT,
PRIMARY KEY(order_id),
FOREIGN KEY(book_id) REFERENCES catalog(book_id) ON UPDATE CASCADE);
```

ii. Enter at least five tuples for each relation.

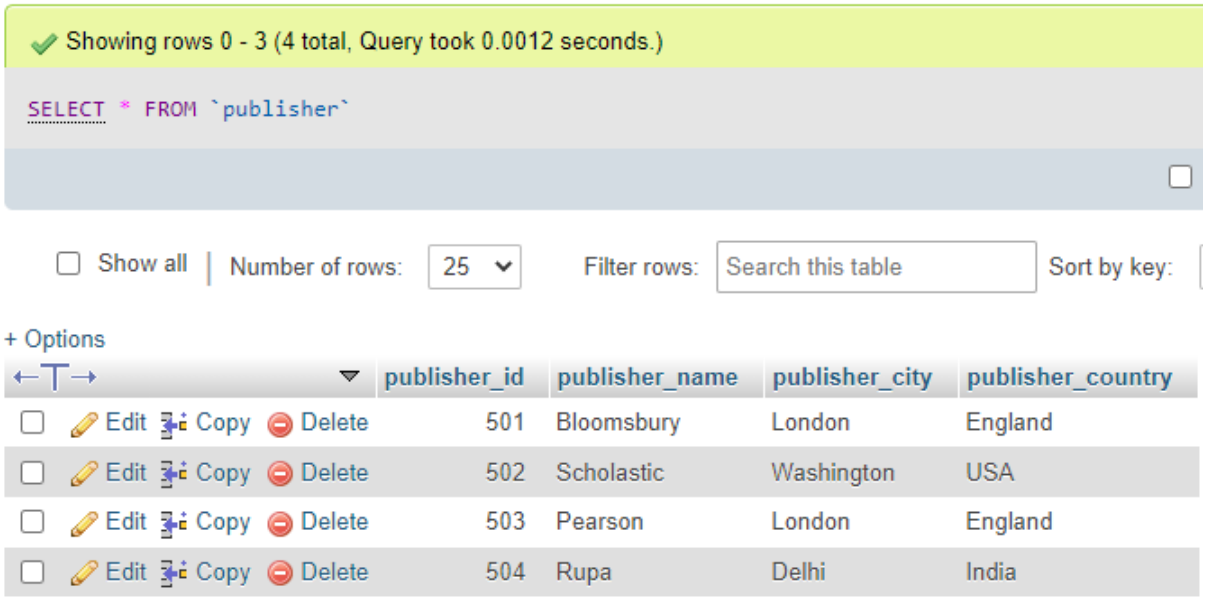
```
INSERT INTO `author` (`author_id`, `author_name`, `author_city`, `author_country`)
 VALUES ('101', 'R          K          Narayan', 'Bengaluru', 'India'), ('102', 'JK
Rowling', 'london', 'England'), ('103', 'Chetan
Bhagat', 'Mumbai', 'India'), ('104', 'Jules
Verne', 'Paris', 'France'), ('105', 'Dan Brown', 'California', 'USA')
```



	author_id	author_name	author_city	author_country
Edit Copy Delete	101	R K Narayan	Bengaluru	India
Edit Copy Delete	102	JK Rowling	london	England
Edit Copy Delete	103	Chetan Bhagat	Mumbai	India
Edit Copy Delete	104	Jules Verne	Paris	France
Edit Copy Delete	105	Dan Brown	California	USA

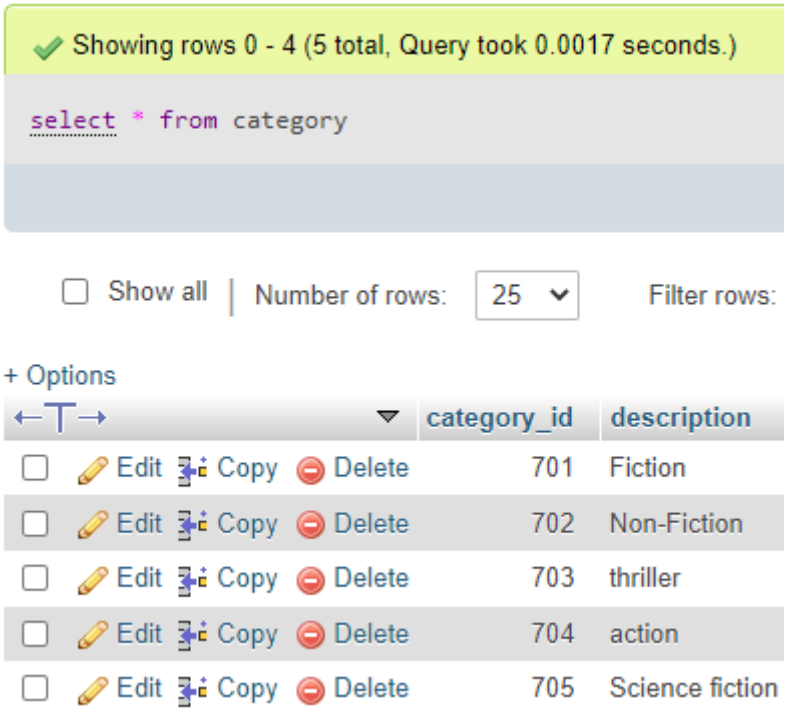
```
INSERT INTO publisher (publisher_id,publisher_name,publisher_city,publisher_country) VALUES
```

```
(101,'Bloomsbury','London','England'),
(102,'Scholastic','Washington','USA'),
(103,'Pearson','London','England'),
(104,'Rupa','Delhi','India');
```



	publisher_id	publisher_name	publisher_city	publisher_country
Edit Copy Delete	501	Bloomsbury	London	England
Edit Copy Delete	502	Scholastic	Washington	USA
Edit Copy Delete	503	Pearson	London	England
Edit Copy Delete	504	Rupa	Delhi	India

```
INSERT INTO category (category_id,description) VALUES (701,'Fiction'), (702,'Non-Fiction'), (703,'thriller'), (704,'action'), (705,'Science fiction')
```



	category_id	description
Edit Copy Delete	701	Fiction
Edit Copy Delete	702	Non-Fiction
Edit Copy Delete	703	thriller
Edit Copy Delete	704	action
Edit Copy Delete	705	Science fiction

```
INSERT INTO `catalog` (`book_id`, `book_title`, `author_id`, `publisher_id`, `category_id`, `year`, `price`) VALUES ('2001', 'Around the World in Eighty Days', '104', '503', '703', '1873', '200'), ('2002', 'Malgudi days', '101', '504', '701', '1942', '500'), ('2003', 'HP and Goblet Of
```

```
Fire', '102', '501', '705', '2002', '600'), ('2004', 'Two
States', '103', '504', '702', '2009', '75'), ('2005', 'HP and Order Of
Phoenix', '102', '503', '704', '2005', '650'), ('2006', 'Da Vinci
Code', '105', '502', '702', '2004', '400'), ('2007', 'Swami and
Friends', '101', '503', '701', '1967', '399')
```

Showing rows 0 - 6 (7 total, Query took 0.0013 seconds.)

```
SELECT * FROM `catalog`
```

Profiling [Edit inline] [Edit]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	book_id	book_title	author_id	publisher_id	category_id	year	price
Edit Copy Delete	2001	Around the World in Eighty Day	104	503	703	1873	200
Edit Copy Delete	2002	Malgudi days	101	504	701	1942	500
Edit Copy Delete	2003	HP and Goblet Of Fire	102	501	705	2002	600
Edit Copy Delete	2004	Two States	103	504	702	2009	75
Edit Copy Delete	2005	HP and Order Of Phoenix	102	503	704	2005	650
Edit Copy Delete	2006	Da Vinci Code	105	502	702	2004	400
Edit Copy Delete	2007	Swami and Friends	101	503	701	1967	399

```
INSERT INTO `orderdetails` (`order_id`, `book_id`, `quantity`) VALUES ('5001', '20
01', '150'), ('5010', '2005', '800'), ('5100', '2002', '700'), ('5110', '2007', '2
50')
```

Showing rows 0 - 3 (4 total, Query took 0.0010 seconds.)

```
SELECT * FROM `orderdetails`
```

Show all | Number of rows: 25 Filter rows: Se

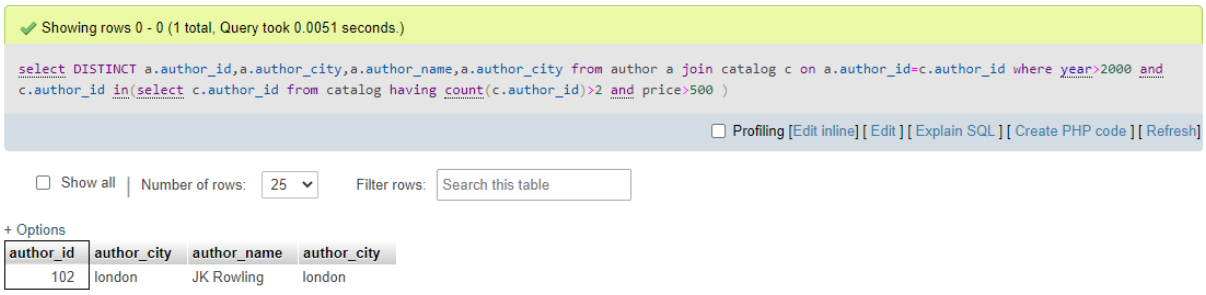
+ Options

	order_id	book_id	quantity
Edit Copy Delete	5001	2001	150
Edit Copy Delete	5010	2005	800
Edit Copy Delete	5100	2002	700
Edit Copy Delete	5110	2007	250

iii. Give the details of the authors who have 2 or more books in the catalog and the price of the books in the catalog is above 500 and the year of publication is after 2000.

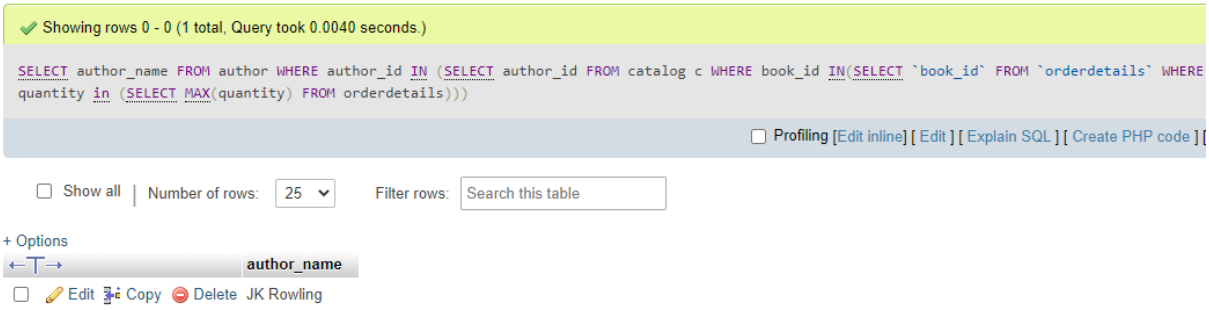
```
select DISTINCT a.author_id,a.author_city,a.author_name,a.author_city

from author a join catalog c on a.author_id=c.author_id where
```

year>2000 and c.author_id in(select c.author_id from catalog having count(c.author_id)>2 and price>500)



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

SELECT author_name FROM author WHERE author_id IN (SELECT author_id FROM catalog c WHERE book_id IN(SELECT `book_id` FROM `orderdetails` WHERE quantity in (SELECT MAX(quantity) FROM orderdetails)))



v. Demonstrate how you increase the price of books published by a specific publisher by 10%.

Program 8:

Consider the following database of student enrollment in courses and books adopted for each course.

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

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

ENROLL (regno: String, cname: String, 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),
    NAME VARCHAR(20),
    MAJOR VARCHAR(5),
    BDATE DATE,
    PRIMARY KEY(REGNO)
    );


CREATE TABLE COURSE(
    COURSE_ID INT,
    CNAME VARCHAR(20),
    DEPT VARCHAR(4),
    PRIMARY KEY(COURSE_ID)
    );


CREATE TABLE TEXT(
    ISBN INT,
    BOOK_TITLE VARCHAR(50),
    PUBLISHER VARCHAR(20),
    AUTHOR VARCHAR(20),
```

```
    PRIMARY KEY(ISBN)
    );

CREATE TABLE ADOPTION(
    COURSE_ID INT,
    SEM INT,
    ISBN INT,
    PRIMARY KEY(COURSE_ID, ISBN),
    FOREIGN KEY(COURSE_ID) REFERENCES COURSE(COURSE_ID),
    FOREIGN KEY(ISBN) REFERENCES TEXT(ISBN)
    );

CREATE TABLE ENROLL(
    REGNO VARCHAR(20),
    COURSE_ID INT,
    SEM INT,
    MARKS INT,
    PRIMARY KEY(REGNO, COURSE_ID),
    FOREIGN KEY(REGNO) REFERENCES STUDENT(REGNO),
    FOREIGN KEY(COURSE_ID) REFERENCES COURSE(COURSE_ID)
    );
```

ii. Enter at least five tuples for each relation.

```
INSERT INTO `student` (`REGNO`, `NAME`, `MAJOR`, `BDATE`) VALUES ('101', '
ravi', 'CS', '2001-06-01'), ('102', 'murali', 'IS', '2001-06-
04'), ('103', 'Subash', 'CV', '2001-05-
11'), ('104', 'Akshay', 'CS', '2001-05-
27'), ('105', 'Rajeev', 'ME', '2001-07-21')
```

	REGNO	NAME	MAJOR	BDATE
Edit Copy Delete	101	ravi	CS	2001-06-01
Edit Copy Delete	102	murali	IS	2001-06-04
Edit Copy Delete	103	Subash	CV	2001-05-11
Edit Copy Delete	104	Akshay	CS	2001-05-27
Edit Copy Delete	105	Rajeev	ME	2001-07-21

```
INSERT INTO `course` (`COURSE_ID`, `CNAME`, `DEPT`) VALUES ('201', 'Data
Structures', 'IS'), ('202', 'Fluid mechanics', 'ME'), ('203', 'Building
materials', 'CV'), ('204', 'Java', 'CS'), ('205', 'DBMS', 'CS')
```

	COURSE_ID	CNAME	DEPT
Edit Copy Delete	201	Data Structures	IS
Edit Copy Delete	202	Fluid mechanics	ME
Edit Copy Delete	203	Building materials	CV
Edit Copy Delete	204	Java	CS
Edit Copy Delete	205	DBMS	CS

```
INSERT INTO `text` (`ISBN`, `BOOK_TITLE`, `PUBLISHER`, `AUTHOR`) VALUES ('
301', 'Fluid Mechanics', 'Rachana Sagar', 'Ramesh N R'), ('302', 'Building
Materials', 'Woodhead', 'Haimei Zang'), ('303', 'The complete java
ref', 'Tata McGraw Hill', 'Herbert'), ('304', 'Data
structures', 'Technical Publications', 'A Puntambekar'), ('305', 'DBMS And
MySQL', 'Tata McGraw Hill', 'Paul DuBois')
```

	ISBN	BOOK_TITLE	PUBLISHER	AUTHOR
Edit Copy Delete	301	Fluid Mechanics	Rachana Sagar	Ramesh N R
Edit Copy Delete	302	Building Materials	Woodhead	Haimei Zang
Edit Copy Delete	303	The complete java ref	Tata McGraw Hill	Herbert
Edit Copy Delete	304	Data structures	Technical Publicatio	A Puntambekar
Edit Copy Delete	305	DBMS And MySQL	Tata McGraw Hill	Paul DuBois

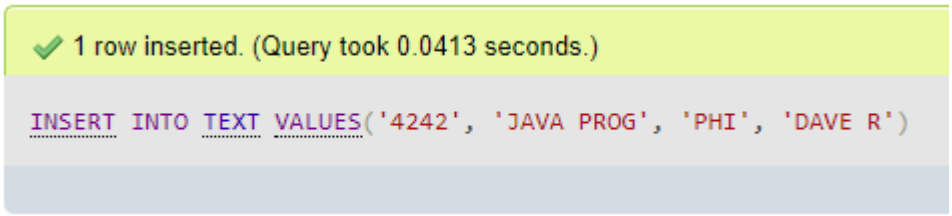
```
INSERT INTO `enroll` (`REGNO`, `COURSE_ID`, `SEM`, `MARKS`) VALUES ('101',
 '205', '4', '89'), ('103', '203', '3', '78'), ('105', '202', '3', '76'),
('104', '204', '3', '88')
```

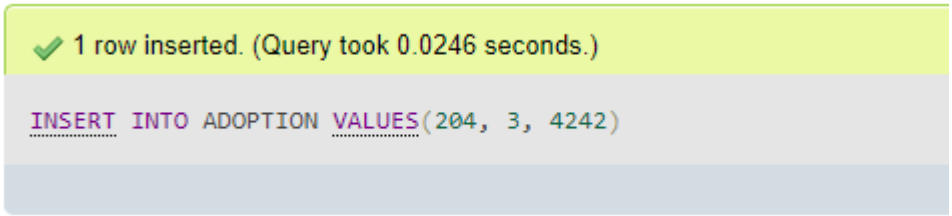
	REGNO	COURSE_ID	SEM	MARKS
Edit Copy Delete	101	205	4	89
Edit Copy Delete	103	203	3	78
Edit Copy Delete	104	204	3	88
Edit Copy Delete	105	202	3	76

```
INSERT INTO `adoption` (`COURSE_ID`, `SEM`, `ISBN`) VALUES ('205', '4', '3
05'), ('204', '3', '303'), ('203', '3', '302'), ('202', '3', '301')
```

	COURSE_ID	SEM	ISBN
Edit Copy Delete	202	3	301
Edit Copy Delete	203	3	302
Edit Copy Delete	204	3	303
Edit Copy Delete	205	4	305

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

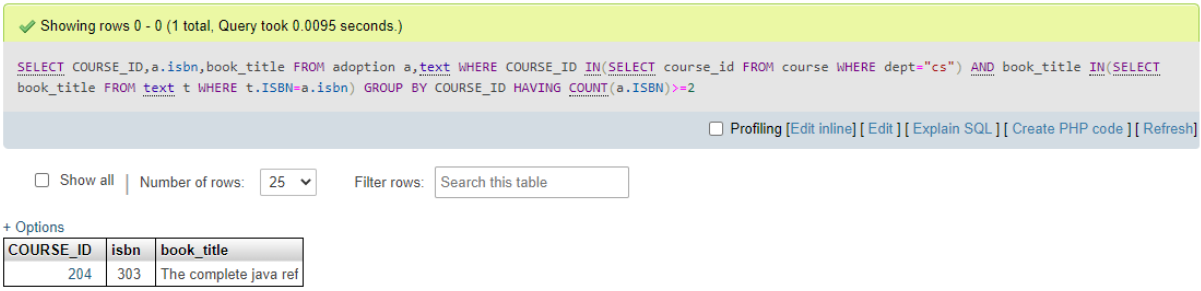




	ISBN	BOOK_TITLE	PUBLISHER	AUTHOR
Edit Copy Delete	301	Fluid Mechanics	Rachana Sagar	Ramesh N R
Edit Copy Delete	302	Building Materials	Woodhead	Haimei Zang
Edit Copy Delete	303	The complete java ref	Tata McGraw Hill	Herbert
Edit Copy Delete	304	Data structures	Technical Publicatio	A Puntambekar
Edit Copy Delete	305	DBMS And MySQL	Tata McGraw Hill	Paul DuBois
Edit Copy Delete	4242	JAVA PROG	PHI	DAVE R

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 COURSE_ID,a.isbn,book_title FROM adoption a,text  WHERE
COURSE_ID IN(SELECT course_id FROM course WHERE dept="cs") AND
book_title IN(SELECT book_title FROM text t WHERE t.ISBN=a.isbn)
GROUP BY COURSE_ID HAVING COUNT(a.ISBN)>=2
```



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

SELECT c1.dept FROM course c1 NATURAL JOIN adoption a1 NATURAL JOIN text t1 ,course c2 NATURAL JOIN adoption a2 NATURAL JOIN text t2 WHERE (c1.COURSE_ID!=c2.COURSE_ID AND c1.DEPT=c2.DEPT and t1.PUBLISHER=t2.PUBLISHER)

****

****

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)

```
CREATE TABLE ACTOR(
    ACT_ID INT,
    ACT_NAME VARCHAR(10),
    ACT_GENDER VARCHAR(10),
    PRIMARY KEY(ACT_ID)
    );
```

```
INSERT INTO `actor` (`ACT_ID`, `ACT_NAME`, `ACT_GENDER`) VALUES ('1', 'Tracy', 'MA
LE'), ('2', 'Keanu', 'MALE'), ('3', 'Khan', 'MALE'), ('4', 'Brad', 'MALE'), ('5',
'Rai', 'FEMALE')
```

	ACT_ID	ACT_NAME	ACT_GENDER
Edit Copy Delete	1	Tracy	MALE
Edit Copy Delete	2	Keanu	MALE
Edit Copy Delete	3	Khan	MALE
Edit Copy Delete	4	Brad	MALE
Edit Copy Delete	5	Rai	FEMALE

```
CREATE TABLE DIRECTOR(
    DIR_ID INT,
    DIR_NAME VARCHAR(10),
    DIR_PHONE INT,
    PRIMARY KEY(DIR_ID)
    );
```

```
INSERT INTO `director` (`DIR_ID`, `DIR_NAME`, `DIR_PHONE`) VALUES ('10', 'Bont', '
301742'), ('20', 'Steve', '541829'), ('30', 'Barton', '947382'), ('40', 'Downey',
'840656'), ('50', 'Kevin', '729146')
```

	DIR_ID	DIR_NAME	DIR_PHONE
Edit Copy Delete	10	Bont	301742
Edit Copy Delete	20	Steve	541829
Edit Copy Delete	30	Barton	947382
Edit Copy Delete	40	Downey	840656
Edit Copy Delete	50	Kevin	729146

```
CREATE TABLE MOVIES(
    MOV_ID INT,
    MOV_TITLE VARCHAR(10),
    MOV_YEAR VARCHAR(10),
    MOV_LANG VARCHAR(10),
    DIR_ID INT,PRIMARY KEY(MOV_ID),
    FOREIGN KEY(DIR_ID) REFERENCES DIRECTOR(DIR_ID)
```

```
    );
INSERT INTO `movies` (`MOV_ID`, `MOV_TITLE`, `MOV_YEAR`, `MOV_LANG`, `DIR_ID`) VALUES ('11', 'NOTEBOOK', '2018', 'HINDI', '20'), ('22', 'AVENGERS', '2007', 'ENGLISH', '20'), ('33', '1917', '1999', 'ENGLISH', '10'), ('44', 'GUILTY', '2020', 'HINDI', '10'), ('55', 'WAR', '2018', 'HINDI', '50')
```

	MOV_ID	MOV_TITLE	MOV_YEAR	MOV_LANG	DIR_ID
Edit Copy Delete	11	NOTEBOOK	2018	HINDI	20
Edit Copy Delete	22	AVENGERS	2007	ENGLISH	20
Edit Copy Delete	33	1917	1999	ENGLISH	10
Edit Copy Delete	44	GUILTY	2020	HINDI	10
Edit Copy Delete	55	WAR	2018	HINDI	50

```
CREATE TABLE MOVIE_CAST(
    ACT_ID INT,MOV_ID INT,
    ROLE_PLAYED VARCHAR(10),
    FOREIGN KEY(ACT_ID) REFERENCES ACTOR(ACT_ID),
    FOREIGN KEY(MOV_ID) REFERENCES MOVIES(MOV_ID)
    );
```

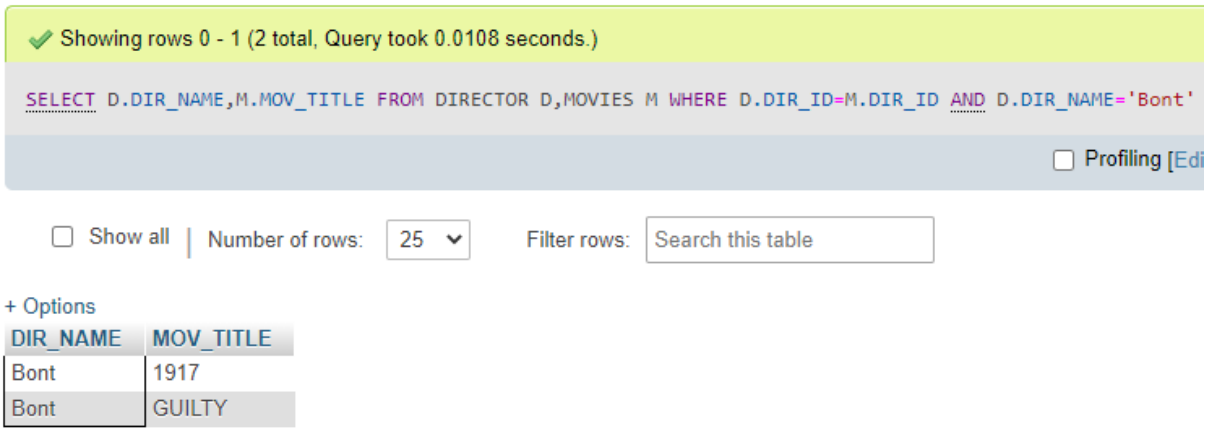
ACT_ID	MOV_ID	ROLE_PLAYED
5	55	VILLAIN
1	55	MAIN LEAD
5	22	SISTER
4	22	MAIN LEAD
4	55	BROTHER
2	44	VILLAIN

```
CREATE TABLE RATING(
    MOV_ID INT,REV_STARS FLOAT,
    FOREIGN KEY(MOV_ID) REFERENCES MOVIES(MOV_ID)
    );
```

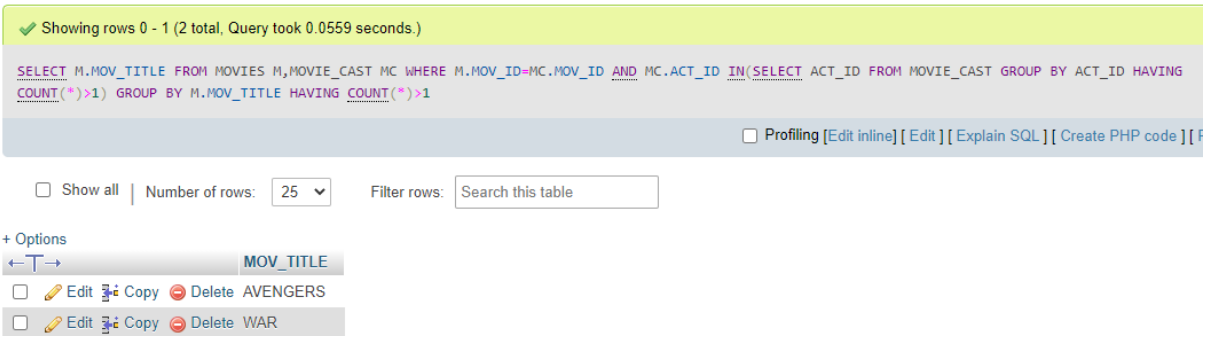
MOV_ID	REV_STARS
11	4
55	3.5
22	2.5
11	3.5
11	4

Write SQL queries to

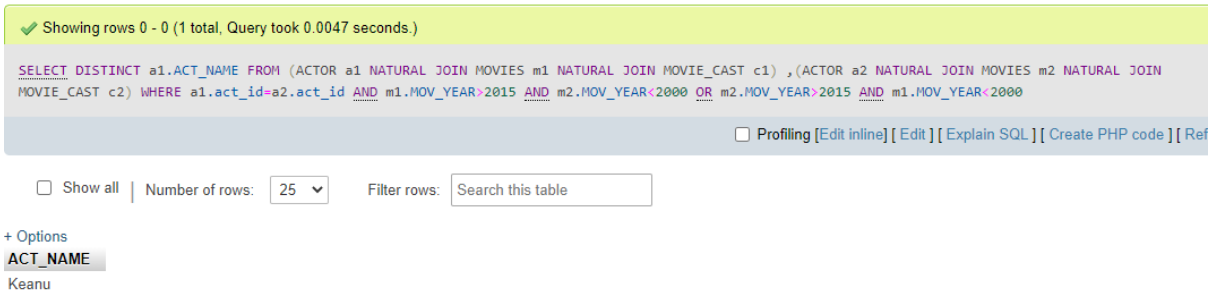
i. List the titles of all movies directed by a specific director.



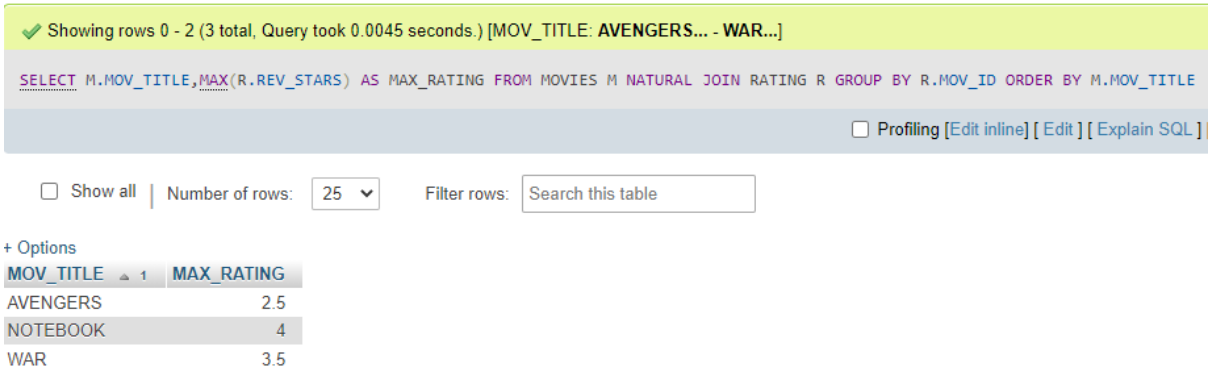
ii. Find the movie names where one or more actors acted in two or more movies.



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



iv. 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.



v. Update rating of all movies directed by ‘Steve’ to 5.

```
UPDATE RATING SET REV_STARS=5 WHERE MOV_ID IN (SELECT M.MOV_ID FROM MOVIES M NATURAL JOIN DIRECTOR D WHERE D.DIR_NAME='Steve')
```

[Edit inline] [E

⚠ Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

✔ Showing rows 0 - 4 (5 total, Query took 0.0008 seconds.)

```
SELECT * FROM RATING
```

Profiling [Edit inline] [Edit] [Explain SQL] [Crea

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

MOV_ID	REV_STARS
11	5
55	3.5
22	5
11	5
11	5

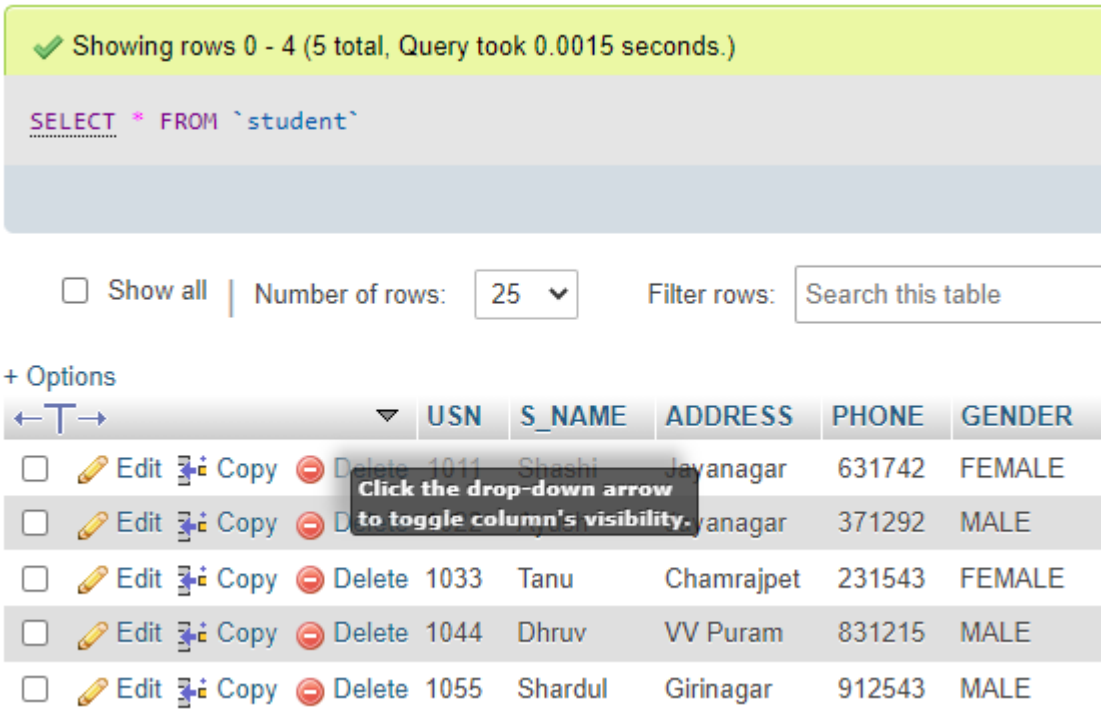
****

********** Program 10 in next page**********

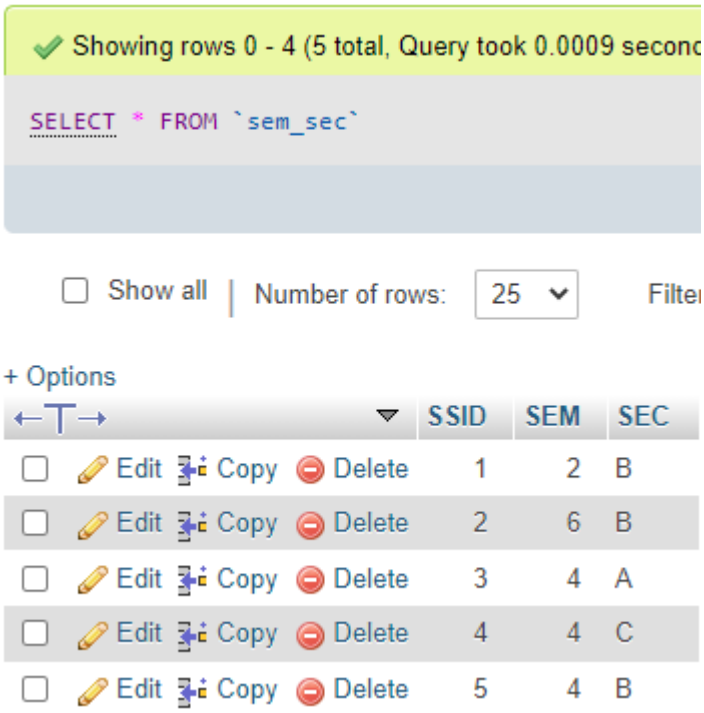
POGRAM 10:COLLEGE DATABASE

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)

```
CREATE TABLE STUDENT(
    USN INT,S_NAME VARCHAR(10),
    ADDRESS VARCHAR(20),
    PHONE INT,
    GENDER VARCHAR(10),
    PRIMARY KEY(USN)
    );
```

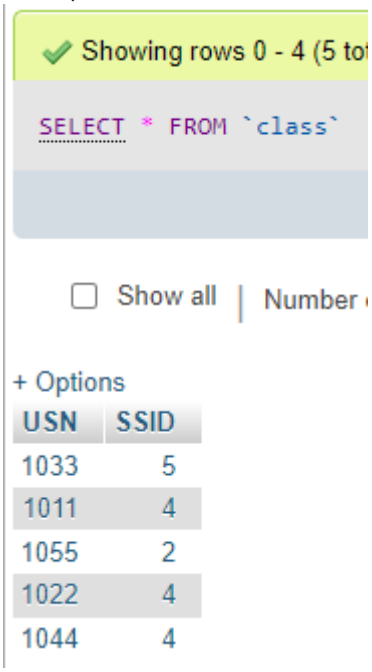



```
CREATE TABLE SEM_SEC(
    SSID INT,
    SEM INT,
    SEC VARCHAR(5),
    PRIMARY KEY(SSID)
    );
```



	SSID	SEM	SEC
Edit Copy Delete	1	2	B
Edit Copy Delete	2	6	B
Edit Copy Delete	3	4	A
Edit Copy Delete	4	4	C
Edit Copy Delete	5	4	B

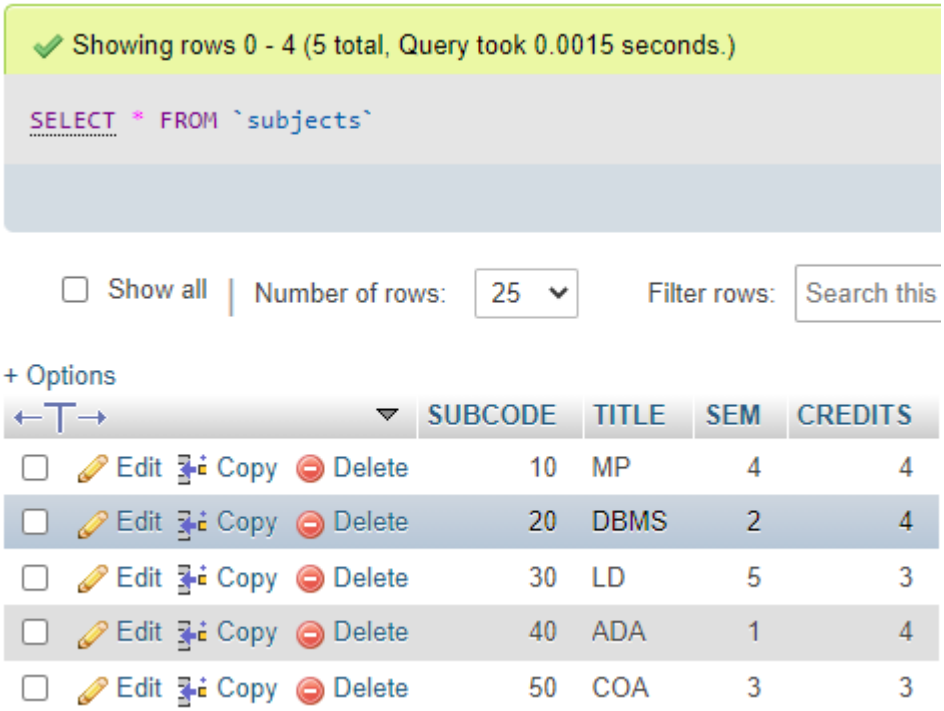
```
CREATE TABLE CLASS(
    USN INT,
    SSID INT,
    FOREIGN KEY(USN) REFERENCES STUDENT(USN),
    FOREIGN KEY(SSID) REFERENCES SEM_SEC(SSID)
    );
```



USN	SSID
1033	5
1011	4
1055	2
1022	4
1044	4

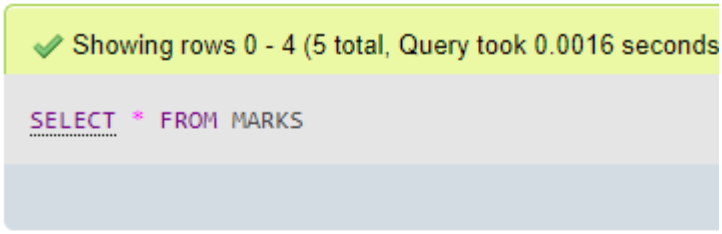
```
CREATE TABLE SUBJECTS(
    SUBCODE INT,
    TITLE VARCHAR(20),
    SEM INT,
```

```
    CREDITS INT,
    PRIMARY KEY(SUBCODE)
    );
```



	SUBCODE	TITLE	SEM	CREDITS
Edit Copy Delete	10	MP	4	4
Edit Copy Delete	20	DBMS	2	4
Edit Copy Delete	30	LD	5	3
Edit Copy Delete	40	ADA	1	4
Edit Copy Delete	50	COA	3	3

```
CREATE TABLE MARKS(
    USN INT,
    SUBCODE INT,
    SSID INT,
    TEST1 INT,
    TEST2 INT,
    TEST3 INT,
    FOREIGN KEY(USN) REFERENCES STUDENT(USN),
    FOREIGN KEY(SSID) REFERENCES SEM_SEC(SSID),
    FOREIGN KEY(SUBCODE) REFERENCES SUBJECTS(SUBCODE)
    );
```

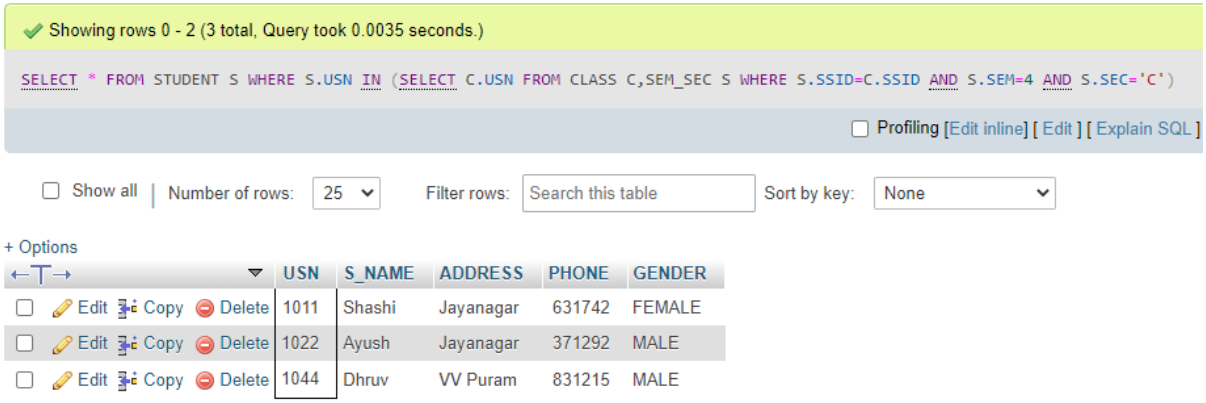


+ Options

USN	SUBCODE	SSID	TEST1	TEST2	TEST3
1033	10	5	19	19	20
1055	30	2	19	19	19
1022	40	4	12	18	16
1044	10	4	10	12	11
1011	20	4	15	14	13

Write SQL queries to

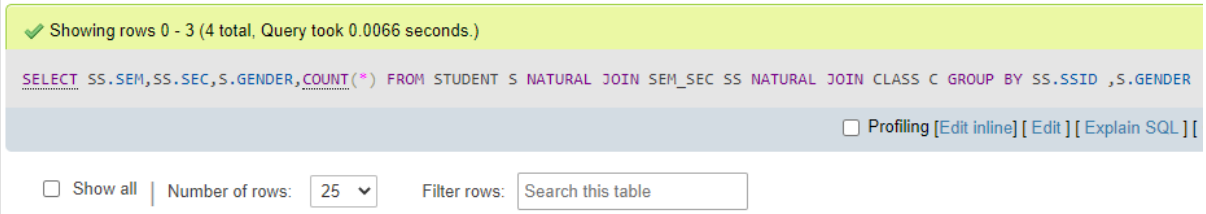
i. List all the student details studying in fourth semester 'C' section



+ Options

	USN	S_NAME	ADDRESS	PHONE	GENDER
Edit Copy Delete	1011	Shashi	Jayanagar	631742	FEMALE
Edit Copy Delete	1022	Ayush	Jayanagar	371292	MALE
Edit Copy Delete	1044	Dhruv	VV Puram	831215	MALE

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



+ Options

SEM	SEC	GENDER	COUNT(*)
6	B	MALE	1
4	C	FEMALE	1
4	C	MALE	2
4	B	FEMALE	1

iii. Create a view of Test1 marks of student USN '1022' in all subjects.

```
CREATE VIEW USN_22(USN,SUB,MARKS) AS SELECT M.USN,S.TITLE,M.TEST1 FROM
MARKS M,SUBJECTS S WHERE M.SUBCODE=S.SUBCODE AND M.USN=1022;
```

```
SELECT * FROM USN_22
```

+ Options

	USN	SUB	MARKS
Edit Copy Delete	1022	ADA	12

iv. Calculate the FinalIA (average of best two test marks) and update the corresponding table for all students.

```
ALTER TABLE MARKS ADD COLUMN FINAL_ALL FLOAT;
UPDATE MARKS SET FINAL_ALL=((TEST1+TEST2+TEST3)-
LEAST(TEST1,TEST2,TEST3))/2;
SELECT * FROM MARKS;
```

USN	SUBCODE	SSID	TEST1	TEST2	TEST3	FINAL_ALL
1033	10	5	19	19	20	19.5
1055	30	2	19	19	19	19
1022	40	4	12	18	16	17
1044	10	4	10	12	11	11.5
1011	20	4	15	14	13	14.5

v. 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.

```
ALTER TABLE MARKS ADD COLUMN CATEGORY VARCHAR(20);
UPDATE MARKS SET CATEGORY=
                        CASE
           WHEN FINAL_ALL>=17 AND FINAL_ALL<=20 THEN 'OUTSTANDING'
                    WHEN FINAL_ALL>=12 AND FINAL_ALL<17 THEN 'AVERAGE'
                    WHEN FINAL_ALL<12 THEN 'WEAK'
                        END;
SELECT * FROM MARKS;
```

+ Options

USN	SUBCODE	SSID	TEST1	TEST2	TEST3	FINAL_ALL	CATEGORY
1033	10	5	19	19	20	19.5	OUTSTANDING
1055	30	2	19	19	19	19	OUTSTANDING
1022	40	4	12	18	16	17	OUTSTANDING
1044	10	4	10	12	11	11.5	WEAK
1011	20	4	15	14	13	14.5	AVERAGE

****