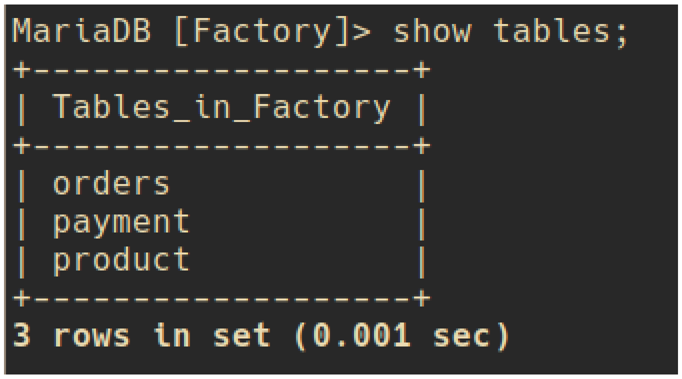
CS Practical File

SECTION – 2 (DATABASE MANAGEMENT AND SQL)

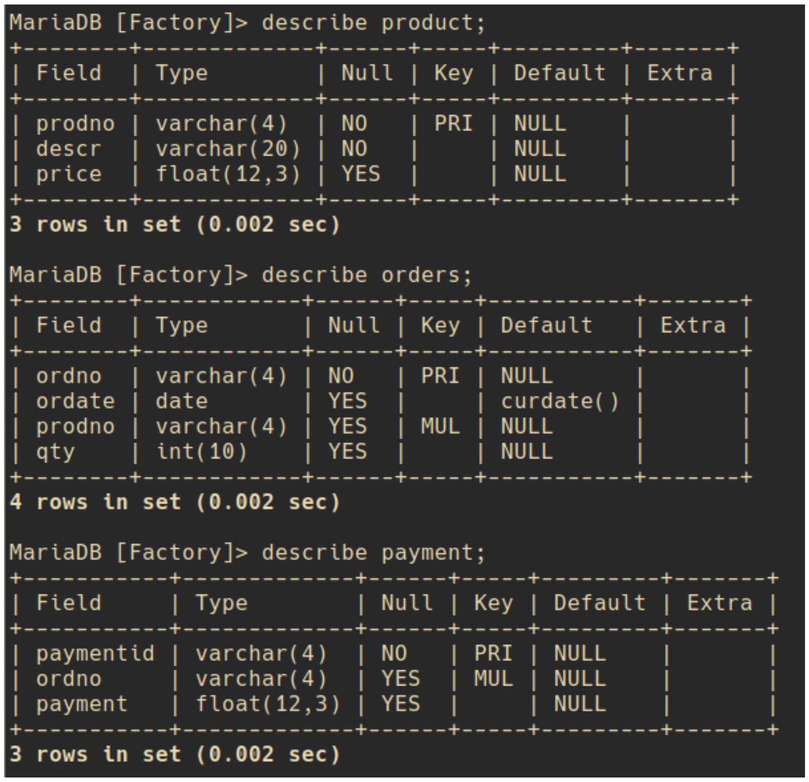
Q2-A:

1) Display the tables existing in the database.



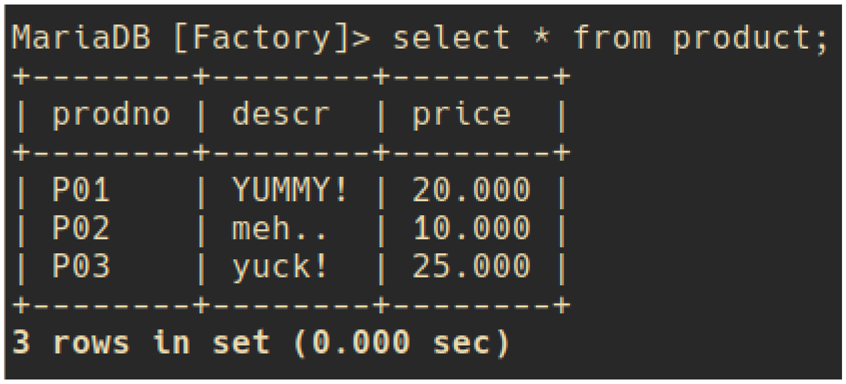
```
MariaDB [Factory]> show tables;
+--------------------+
| Tables_in_Factory |
+--------------------+
| orders            |
| payment           |
| product           |
+--------------------+
3 rows in set (0.001 sec)
```

2) Display the structure of all the tables.



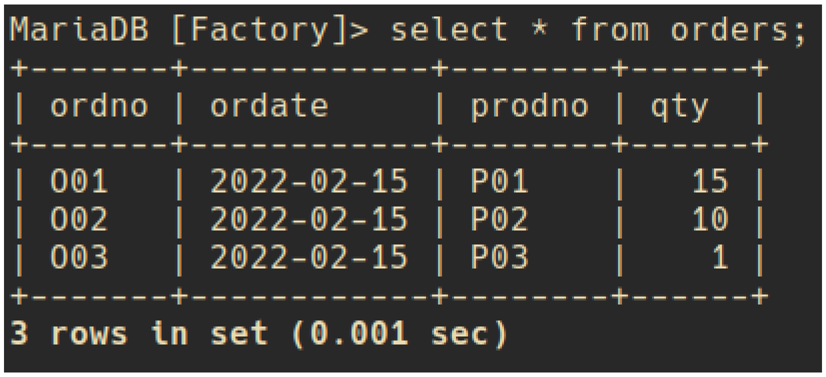
```
MariaDB [Factory]> describe product;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| prodno | varchar(4)  | NO   | PRI | NULL    |       |
| descr  | varchar(20) | NO   |     | NULL    |       |
| price  | float(12,3) | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
3 rows in set (0.002 sec)

MariaDB [Factory]> describe orders;
+--------+------------+------+-----+-----------+-------+
| Field  | Type       | Null | Key | Default   | Extra |
+--------+------------+------+-----+-----------+-------+
| ordno  | varchar(4) | NO   | PRI | NULL      |       |
| ordate | date       | YES  |     | curdate() |       |
| prodno | varchar(4) | YES  | MUL | NULL      |       |
| qty    | int(10)    | YES  |     | NULL      |       |
+--------+------------+------+-----+-----------+-------+
4 rows in set (0.002 sec)

MariaDB [Factory]> describe payment;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| paymentid | varchar(4)  | NO   | PRI | NULL    |       |
| ordno     | varchar(4)  | YES  | MUL | NULL    |       |
| payment   | float(12,3) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
3 rows in set (0.002 sec)
```

3) Display the data from product.



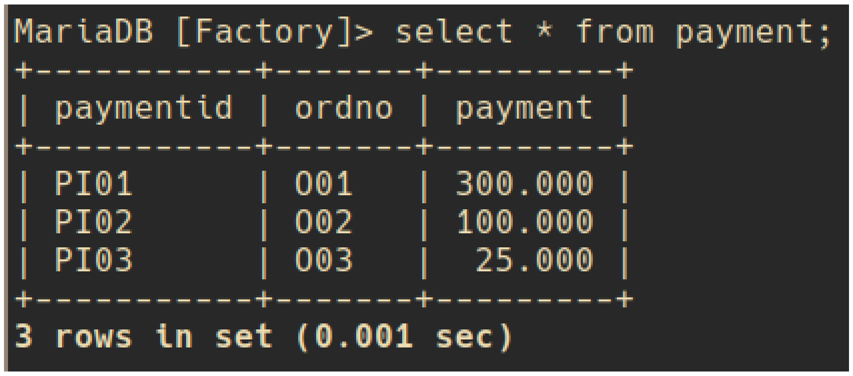
```
MariaDB [Factory]> select * from product;
+--------+--------+--------+
| prodno | descr  | price  |
+--------+--------+--------+
| P01    | YUMMY! | 20.000 |
| P02    | meh..  | 10.000 |
| P03    | yuck!  | 25.000 |
+--------+--------+--------+
3 rows in set (0.000 sec)
```

4) Display the data from orders.



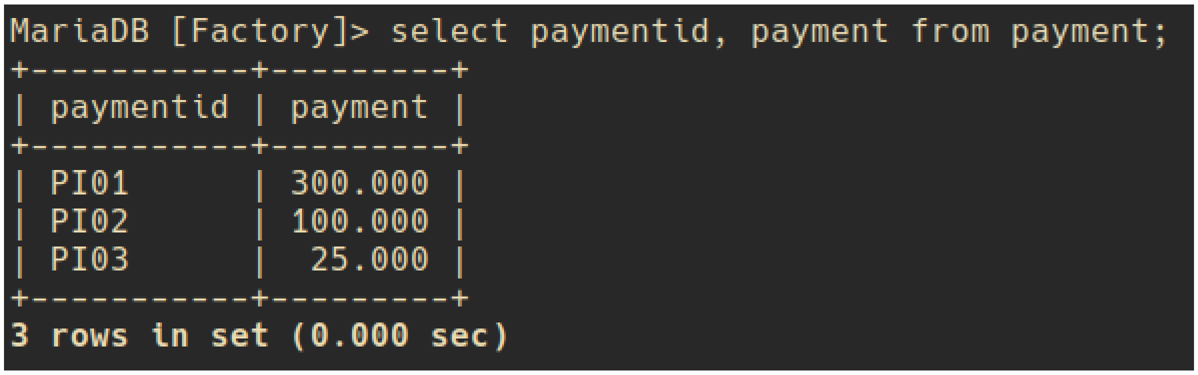
```
MariaDB [Factory]> select * from orders;
+-------+------------+--------+------+
| ordno | ordate     | prodno | qty  |
+-------+------------+--------+------+
| O01   | 2022-02-15 | P01    |   15 |
| O02   | 2022-02-15 | P02    |   10 |
| O03   | 2022-02-15 | P03    |    1 |
+-------+------------+--------+------+
3 rows in set (0.001 sec)
```

5) Display the data from payment.



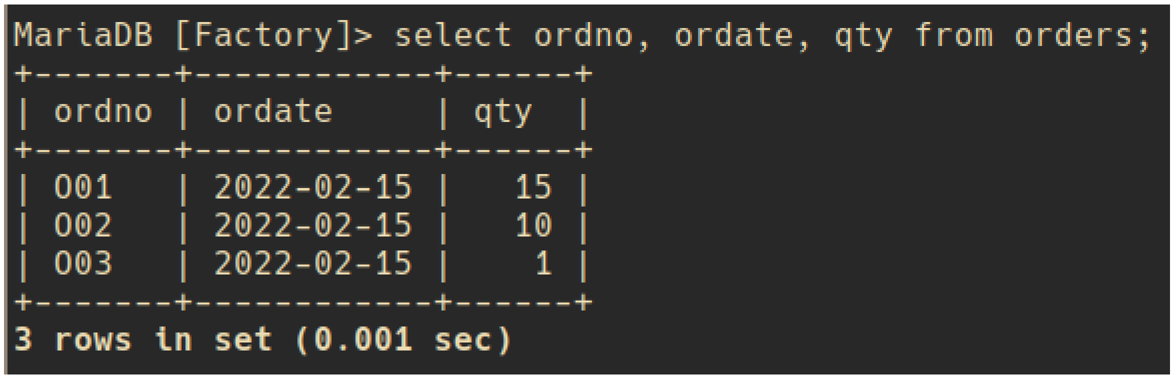
```
MariaDB [Factory]> select * from payment;
+-----------+-------+---------+
| paymentid | ordno | payment |
+-----------+-------+---------+
| PI01      | O01   | 300.000 |
| PI02      | O02   | 100.000 |
| PI03      | O03   |  25.000 |
+-----------+-------+---------+
3 rows in set (0.001 sec)
```

6) Display payment and payment id from payment table.



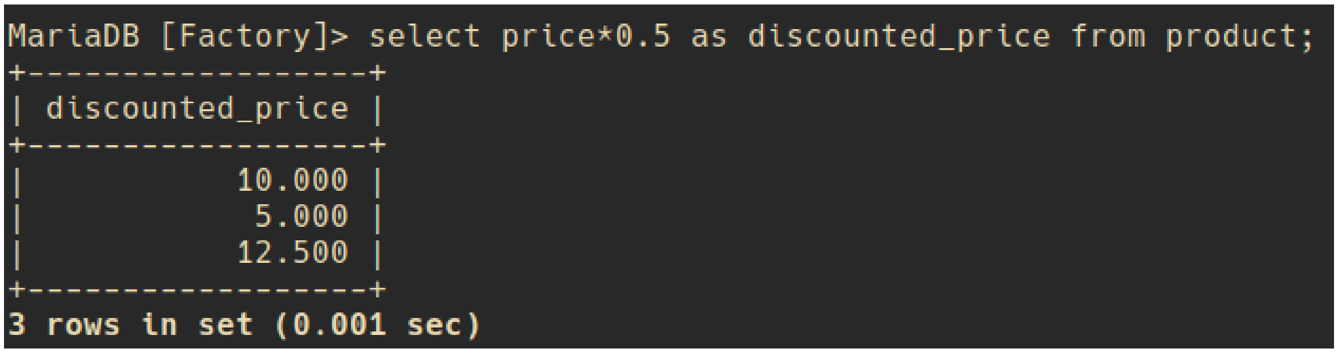
```
MariaDB [Factory]> select paymentid, payment from payment;
+-----------+---------+
| paymentid | payment |
+-----------+---------+
| PI01      | 300.000 |
| PI02      | 100.000 |
| PI03      |  25.000 |
+-----------+---------+
3 rows in set (0.000 sec)
```

7) Display order no, order date, qty from order table.



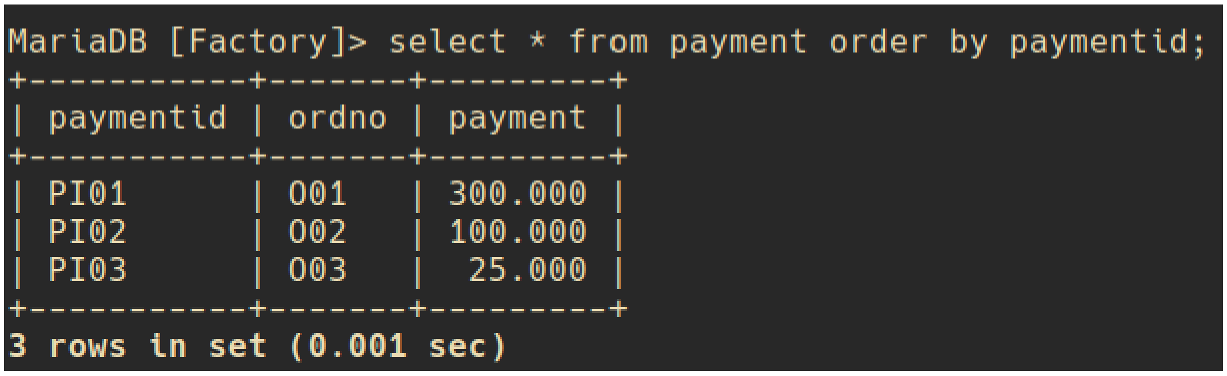
```
MariaDB [Factory]> select ordno, ordate, qty from orders;
+-------+------------+------+
| ordno | ordate     | qty  |
+-------+------------+------+
| 001   | 2022-02-15 |   15 |
| 002   | 2022-02-15 |   10 |
| 003   | 2022-02-15 |    1 |
+-------+------------+------+
3 rows in set (0.001 sec)
```

8) Display the discounted price of all products from product table where discount is 5% of the price.



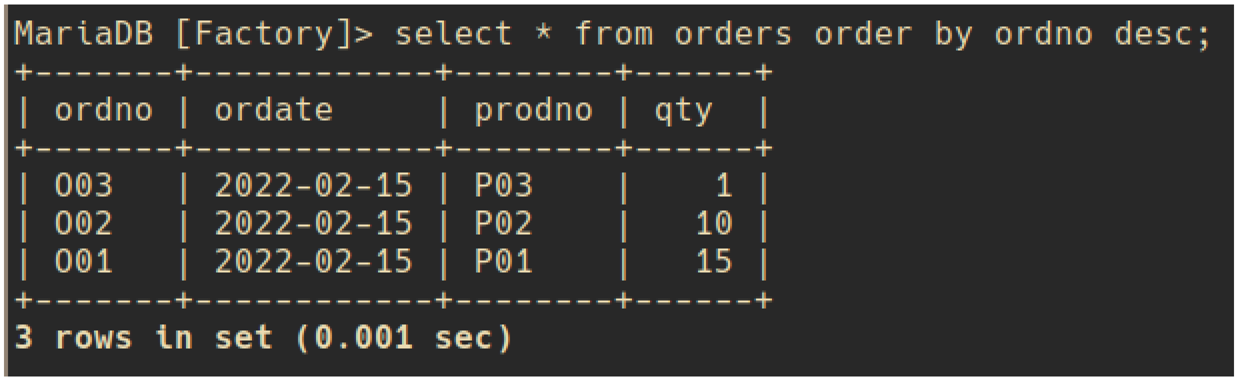
```
MariaDB [Factory]> select price*0.5 as discounted_price from product;
+------------------+
| discounted_price |
+------------------+
|           10.000 |
|            5.000 |
|           12.500 |
+------------------+
3 rows in set (0.001 sec)
```

9) Display the data from payment table, arranged on payment id.



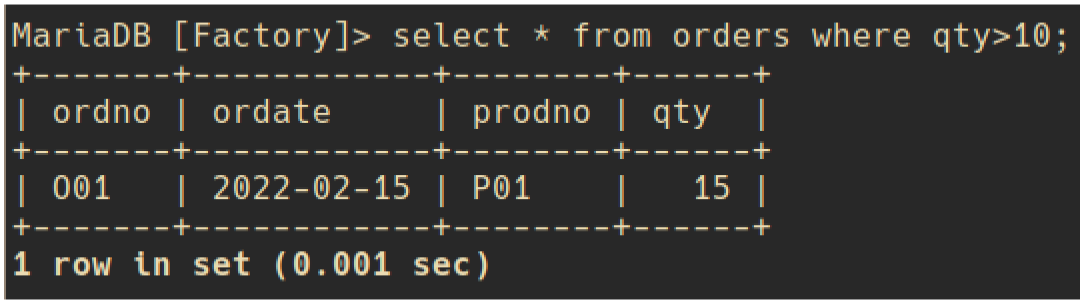
```
MariaDB [Factory]> select * from payment order by paymentid;
+-----------+-------+---------+
| paymentid | ordno | payment |
+-----------+-------+---------+
| PI01      | 001   | 300.000 |
| PI02      | 002   | 100.000 |
| PI03      | 003   |  25.000 |
+-----------+-------+---------+
3 rows in set (0.001 sec)
```

10) Display the data from order table , arranged in descending order on order id.



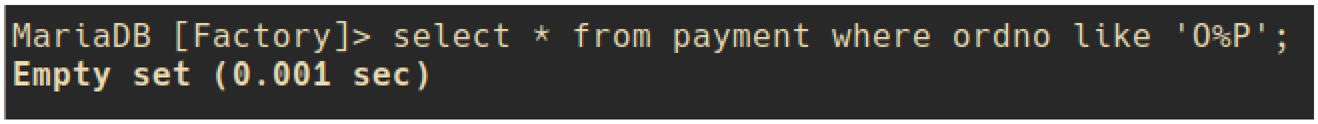
```
MariaDB [Factory]> select * from orders order by ordno desc;
+-------+------------+--------+------+
| ordno | ordate     | prodno | qty  |
+-------+------------+--------+------+
| 003   | 2022-02-15 | P03    |    1 |
| 002   | 2022-02-15 | P02    |   10 |
| 001   | 2022-02-15 | P01    |   15 |
+-------+------------+--------+------+
3 rows in set (0.001 sec)
```

11) Display those order details from order table where qty is more than 10.



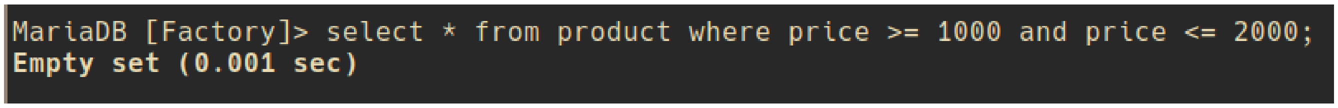
```
MariaDB [Factory]> select * from orders where qty>10;
+-------+------------+--------+------+
| ordno | ordate     | prodno | qty  |
+-------+------------+--------+------+
| 001   | 2022-02-15 | P01    |   15 |
+-------+------------+--------+------+
1 row in set (0.001 sec)
```

12) Display those payment details order id starts with O and ends with P.



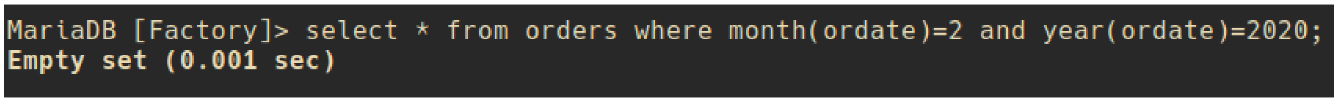
```
MariaDB [Factory]> select * from payment where ordno like 'O%P';
Empty set (0.001 sec)
```

13) Display those product details where price is between 1000 to 2000(both included)



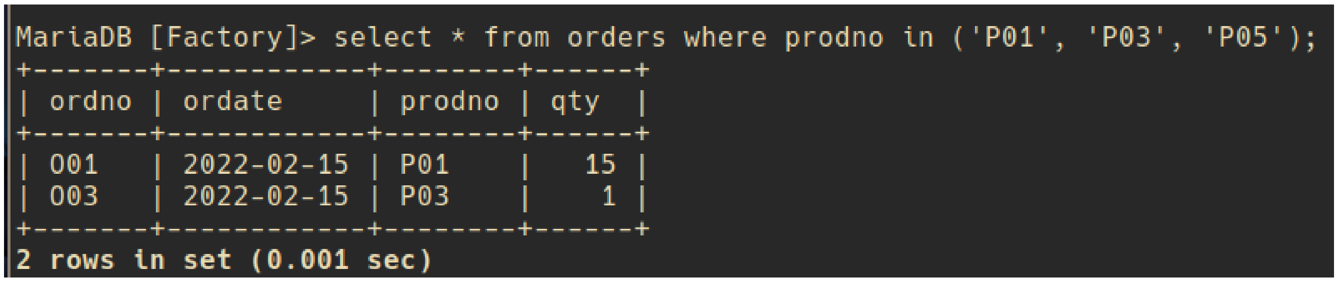
```
MariaDB [Factory]> select * from product where price >= 1000 and price <= 2000;
Empty set (0.001 sec)
```

14) Display those orders which were placed in the month of February 2020.



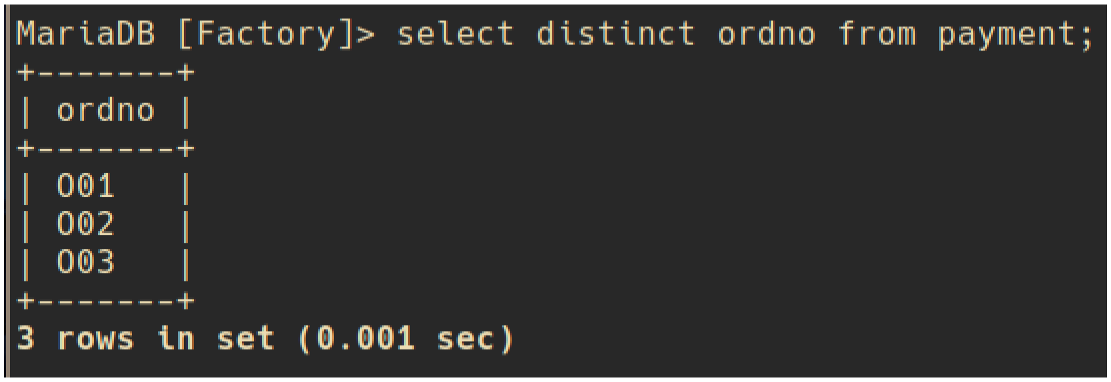
```
MariaDB [Factory]> select * from orders where month(ordate)=2 and year(ordate)=2020;
Empty set (0.001 sec)
```

15) Display those orders where products are ‘P01’, ‘P03’,’P05’ (using in operator)



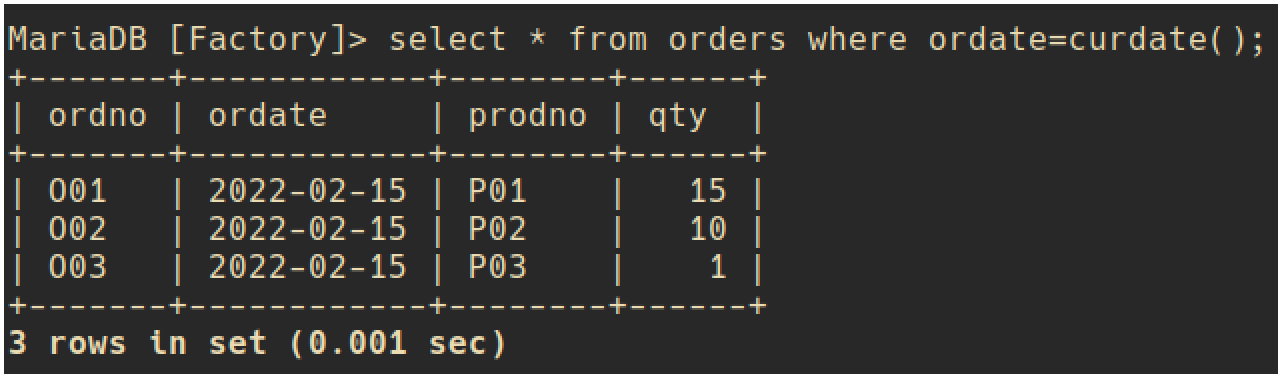
```
MariaDB [Factory]> select * from orders where prodno in ('P01', 'P03', 'P05');
+-------+------------+--------+------+
| ordno | ordate     | prodno | qty  |
+-------+------------+--------+------+
| O01   | 2022-02-15 | P01    |   15 |
| O03   | 2022-02-15 | P03    |    1 |
+-------+------------+--------+------+
2 rows in set (0.001 sec)
```

17) Display all the orders from payment table (orders shouldn’t be duplicated)



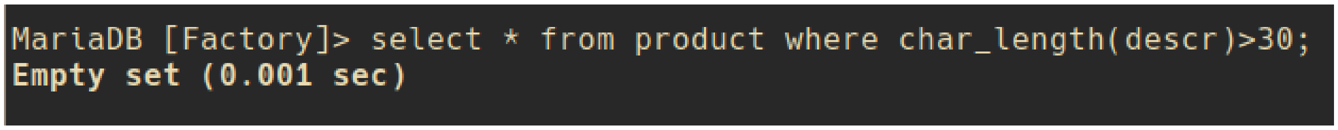
```
MariaDB [Factory]> select distinct ordno from payment;
+-------+
| ordno |
+-------+
| O01   |
| O02   |
| O03   |
+-------+
3 rows in set (0.001 sec)
```

18) Display those orders which are placed today.



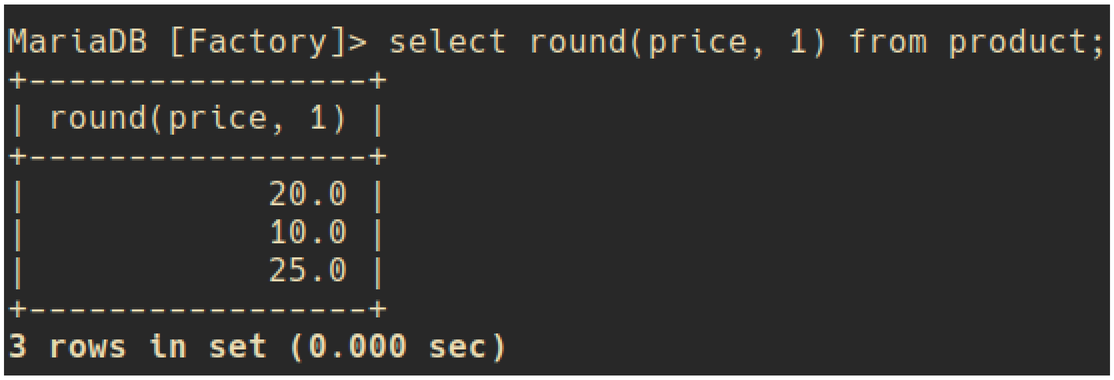
```
MariaDB [Factory]> select * from orders where ordate=curdate();
+-------+------------+--------+------+
| ordno | ordate     | prodno | qty  |
+-------+------------+--------+------+
| 001   | 2022-02-15 | P01    |   15 |
| 002   | 2022-02-15 | P02    |   10 |
| 003   | 2022-02-15 | P03    |    1 |
+-------+------------+--------+------+
3 rows in set (0.001 sec)
```

19) Display those products where description is more than 30 characters long.



```
MariaDB [Factory]> select * from product where char_length(descr)>30;
Empty set (0.001 sec)
```

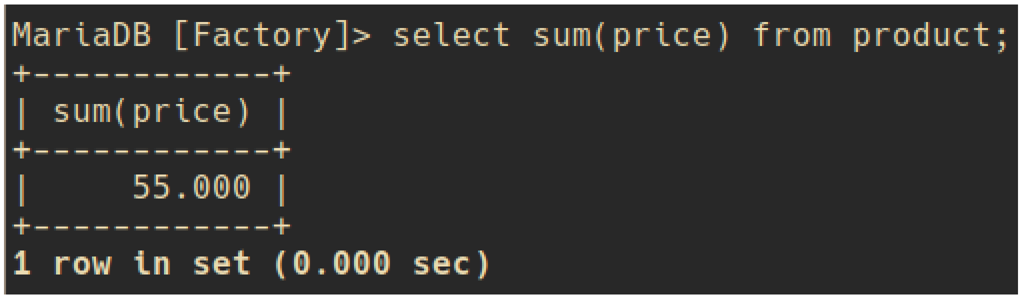
20) Display prices of product, rounded to one place.



```
MariaDB [Factory]> select round(price, 1) from product;
+-----------------+
| round(price, 1) |
+-----------------+
|            20.0 |
|            10.0 |
|            25.0 |
+-----------------+
3 rows in set (0.000 sec)
```

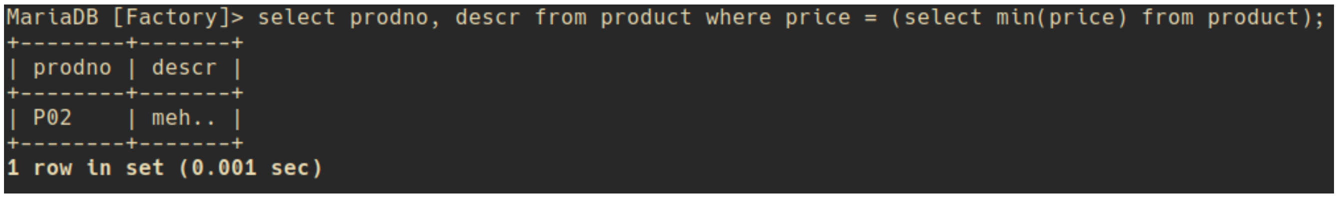
Q2-B:

1) Display the total of price from the product table.



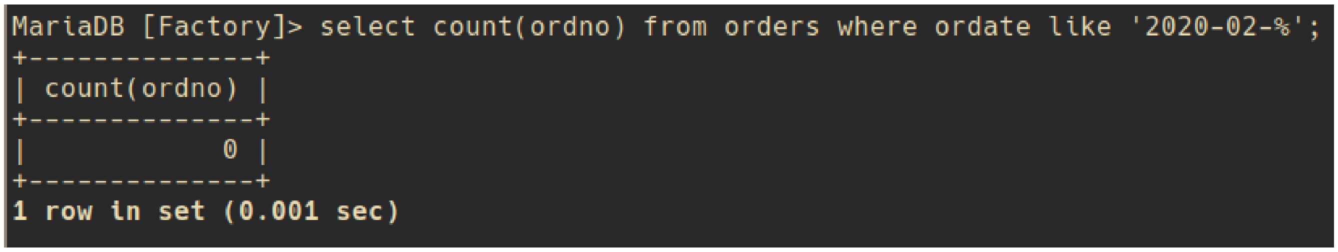
```
MariaDB [Factory]> select sum(price) from product;
+------------+
| sum(price) |
+------------+
|     55.000 |
+------------+
1 row in set (0.000 sec)
```

2) Display the product having minimum price.



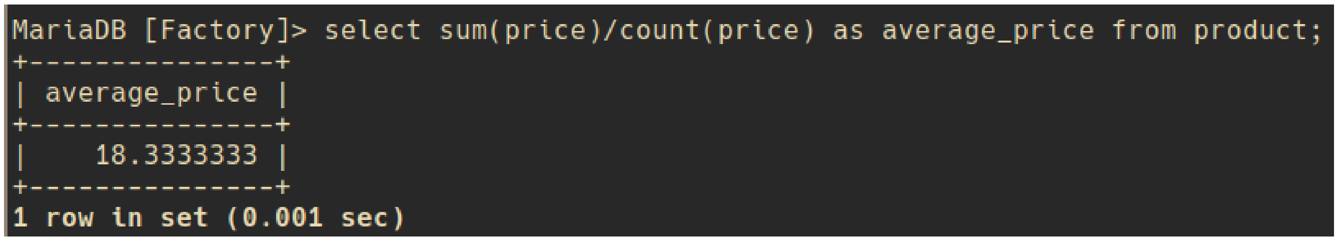
```
MariaDB [Factory]> select prodno, descr from product where price = (select min(price) from product);
+--------+-------+
| prodno | descr |
+--------+-------+
| P02    | meh.. |
+--------+-------+
1 row in set (0.001 sec)
```

3) Display the number of orders placed in the month of February 2020.



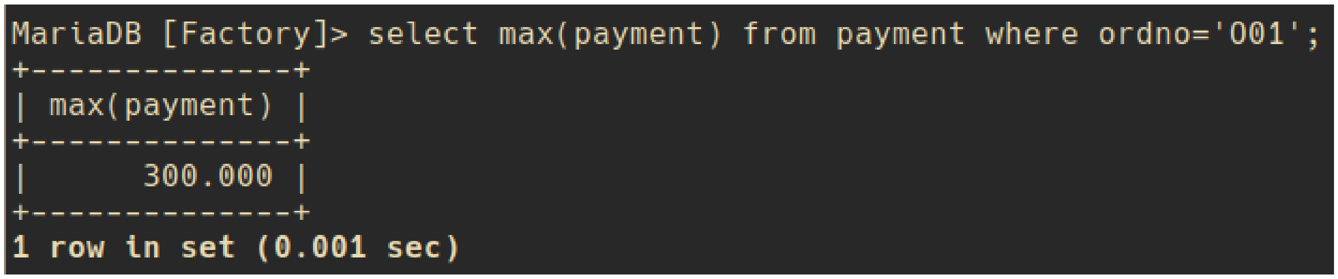
```
MariaDB [Factory]> select count(ordno) from orders where ordate like '2020-02-%';
+--------------+
| count(ordno) |
+--------------+
|            0 |
+--------------+
1 row in set (0.001 sec)
```

4) Display the average of price of all the products.



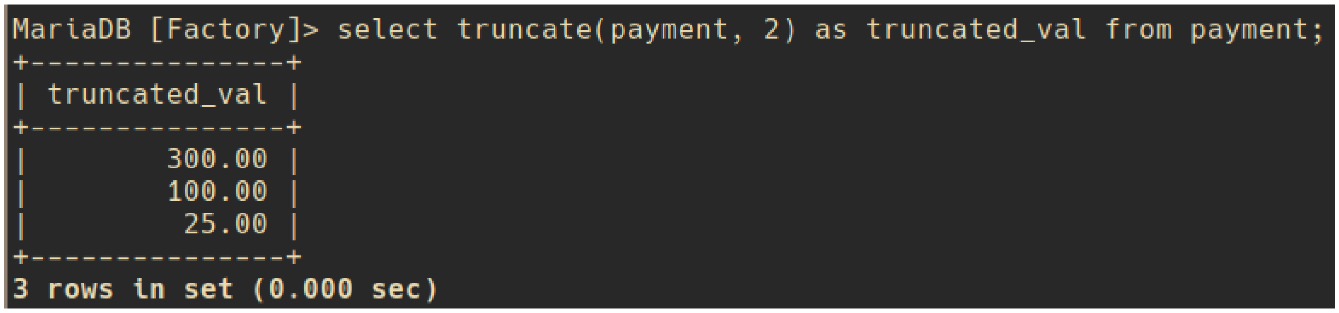
```
MariaDB [Factory]> select sum(price)/count(price) as average_price from product;
+---------------+
| average_price |
+---------------+
|    18.3333333 |
+---------------+
1 row in set (0.001 sec)
```

5) Display the maximum payment paid on order ‘O01’



```
MariaDB [Factory]> select max(payment) from payment where ordno='O01';
+--------------+
| max(payment) |
+--------------+
|      300.000 |
+--------------+
1 row in set (0.001 sec)
```

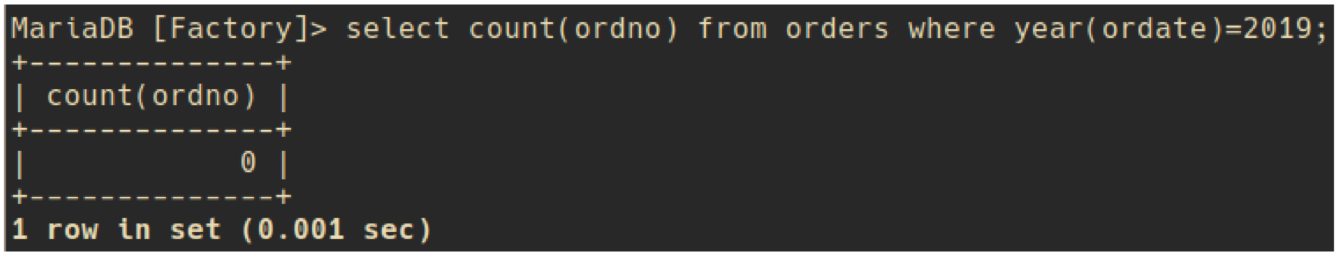
6) Display the payment truncated to two places from the payment table and differentiate between round() and truncate() function in MySQL.



```
MariaDB [Factory]> select truncate(payment, 2) as truncated_val from payment;
+---------------+
| truncated_val |
+---------------+
|        300.00 |
|        100.00 |
|         25.00 |
+---------------+
3 rows in set (0.000 sec)
```

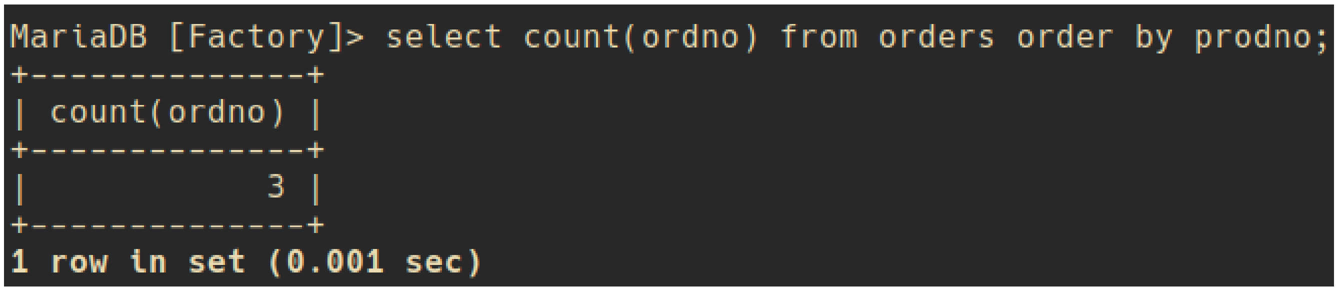
ROUND	**TRUNCATE**
Rounds off the numbers after specified decimal point mathematically.	Simple removes the numbers after the specified decimal point.
Example: 146.248 would round off to 146.25 for 2 decimal points.	Example: 146.248 would truncate to 146.24 for 2 decimal points.

7) Display the count of items ordered in the year 2019.



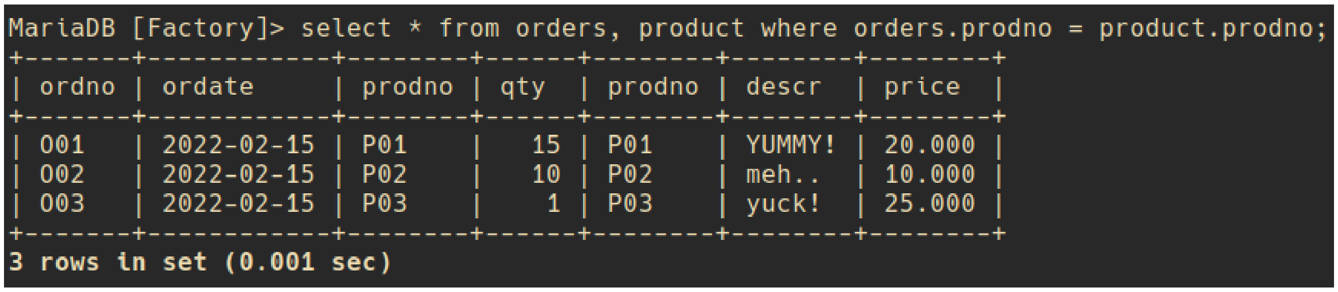
```
MariaDB [Factory]> select count(ordno) from orders where year(ordate)=2019;
+--------------+
| count(ordno) |
+--------------+
|            0 |
+--------------+
1 row in set (0.001 sec)
```

8) Display the no of orders, product wise from the table orders.



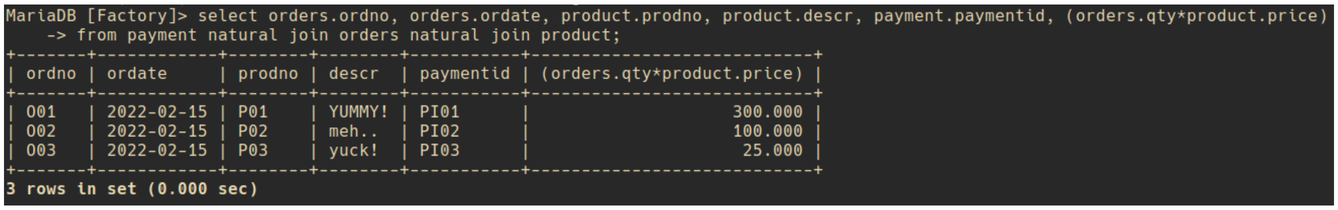
```
MariaDB [Factory]> select count(ordno) from orders order by prodno;
+--------------+
| count(ordno) |
+--------------+
|            3 |
+--------------+
1 row in set (0.001 sec)
```

9) Display the data from order and product table with the matching product number.



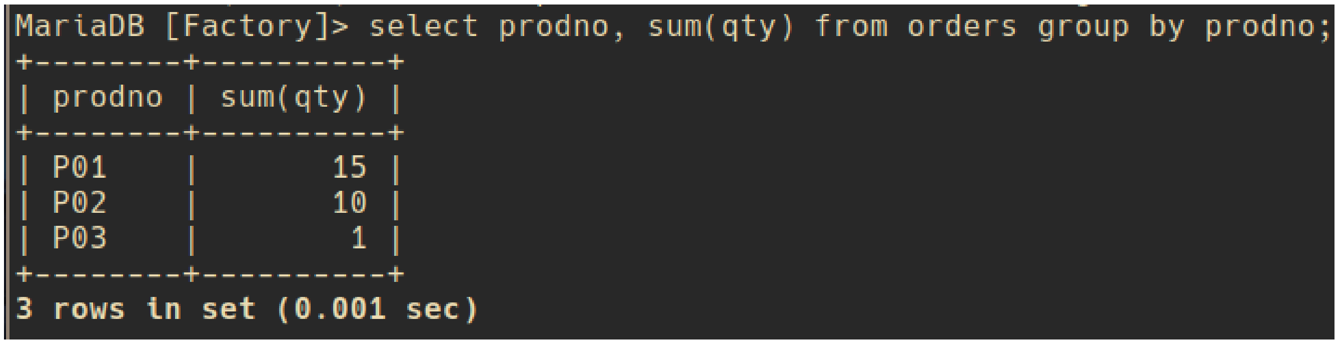
```
MariaDB [Factory]> select * from orders, product where orders.prodno = product.prodno;
+-------+------------+--------+------+--------+--------+--------+
| ordno | ordate     | prodno | qty  | prodno | descr  | price  |
+-------+------------+--------+------+--------+--------+--------+
| 001   | 2022-02-15 | P01    |   15 | P01    | YUMMY! | 20.000 |
| 002   | 2022-02-15 | P02    |   10 | P02    | meh..  | 10.000 |
| 003   | 2022-02-15 | P03    |    1 | P03    | yuck!  | 25.000 |
+-------+------------+--------+------+--------+--------+--------+
3 rows in set (0.001 sec)
```

10) Display orderno, orderdate, product no, descr, payment id, payment(qty*price) from three tables.



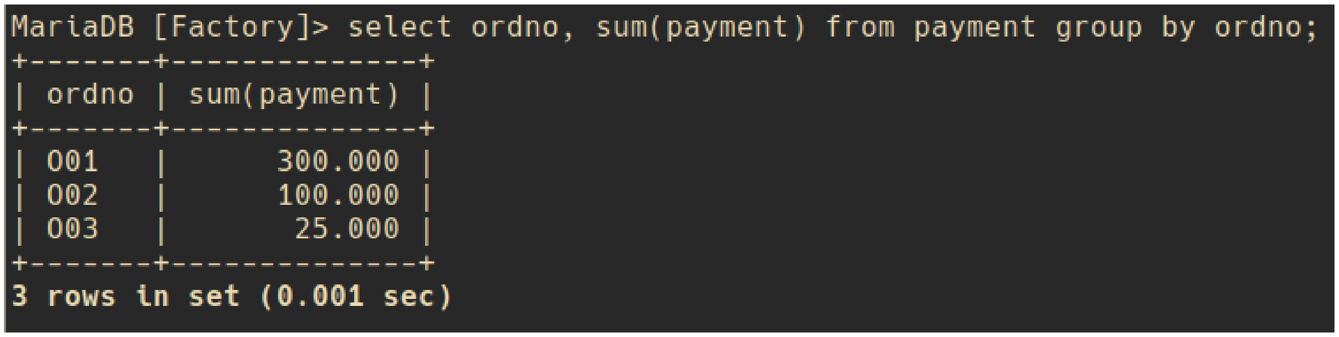
```
MariaDB [Factory]> select orders.ordno, orders.ordate, product.prodno, product.descr, payment.paymentid, (orders.qty*product.price)
    -> from payment natural join orders natural join product;
+-------+------------+--------+--------+-----------+----------------------------+
| ordno | ordate     | prodno | descr  | paymentid | (orders.qty*product.price) |
+-------+------------+--------+--------+-----------+----------------------------+
| 001   | 2022-02-15 | P01    | YUMMY! | PI01      |                    300.000 |
| 002   | 2022-02-15 | P02    | meh..  | PI02      |                    100.000 |
| 003   | 2022-02-15 | P03    | yuck!  | PI03      |                     25.000 |
+-------+------------+--------+--------+-----------+----------------------------+
3 rows in set (0.000 sec)
```

11) Display the product wise total qty ordered.



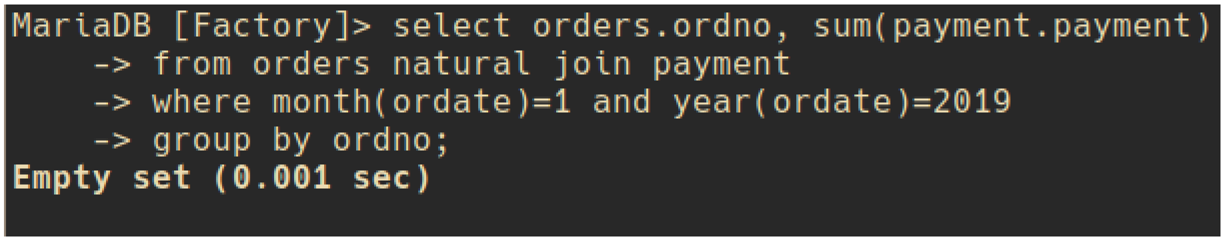
```
MariaDB [Factory]> select prodno, sum(qty) from orders group by prodno;
+--------+----------+
| prodno | sum(qty) |
+--------+----------+
| P01    |       15 |
| P02    |       10 |
| P03    |        1 |
+--------+----------+
3 rows in set (0.001 sec)
```

12) Display the total payment collected for each product ordered.



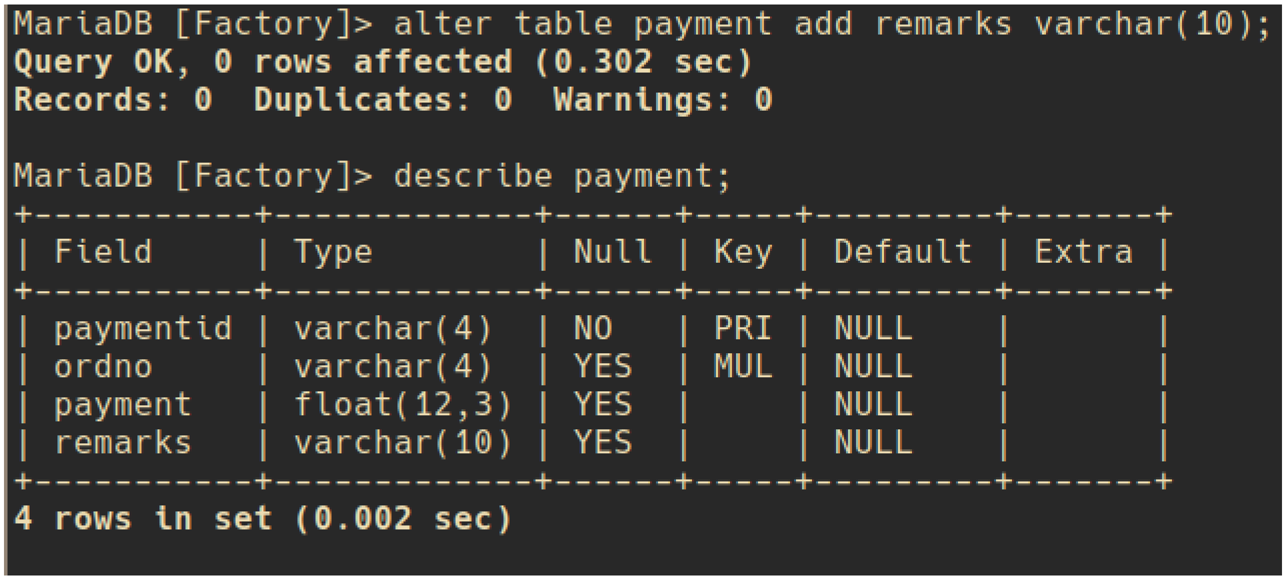
```
MariaDB [Factory]> select ordno, sum(payment) from payment group by ordno;
+-------+--------------+
| ordno | sum(payment) |
+-------+--------------+
| 001   |      300.000 |
| 002   |      100.000 |
| 003   |       25.000 |
+-------+--------------+
3 rows in set (0.001 sec)
```

13) Display the total payment collected for only those products which were ordered in January 2019.



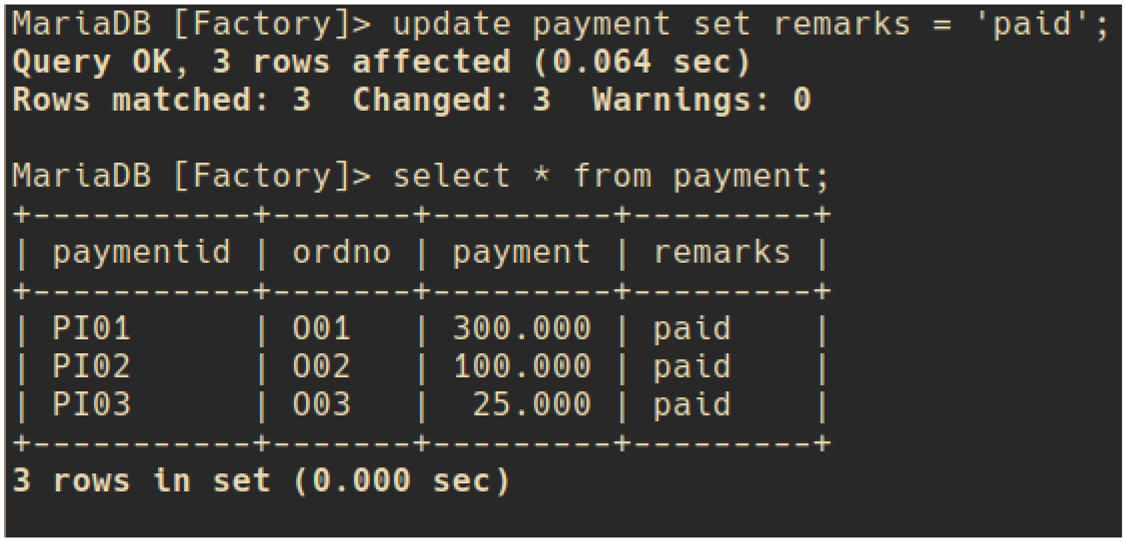
```
MariaDB [Factory]> select orders.ordno, sum(payment.payment)
    -> from orders natural join payment
    -> where month(ordate)=1 and year(ordate)=2019
    -> group by ordno;
Empty set (0.001 sec)
```

15) Add a column REMARKS in payment table.



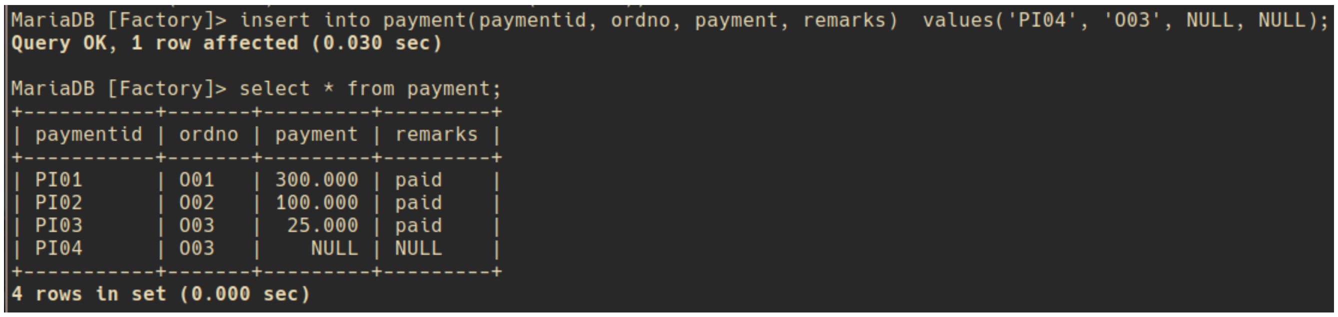
```
MariaDB [Factory]> alter table payment add remarks varchar(10);
Query OK, 0 rows affected (0.302 sec)
Records: 0  Duplicates: 0  Warnings: 0

MariaDB [Factory]> describe payment;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| paymentid | varchar(4)  | NO   | PRI | NULL    |       |
| ordno     | varchar(4)  | YES  | MUL | NULL    |       |
| payment   | float(12,3) | YES  |     | NULL    |       |
| remarks   | varchar(10) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
4 rows in set (0.002 sec)
```

16) Update the value of REMARKS with "paid" for all the payment ids.



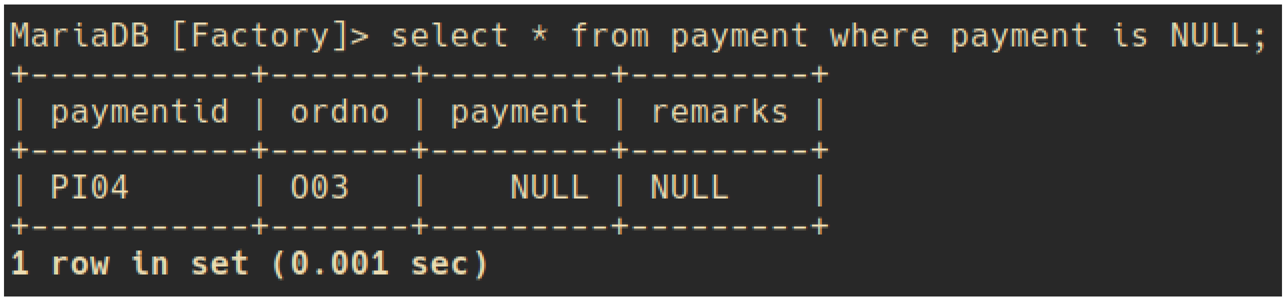
```
MariaDB [Factory]> update payment set remarks = 'paid';
Query OK, 3 rows affected (0.064 sec)
Rows matched: 3  Changed: 3  Warnings: 0

MariaDB [Factory]> select * from payment;
+-----------+-------+---------+---------+
| paymentid | ordno | payment | remarks |
+-----------+-------+---------+---------+
| PI01      | 001   | 300.000 | paid    |
| PI02      | 002   | 100.000 | paid    |
| PI03      | 003   |  25.000 | paid    |
+-----------+-------+---------+---------+
3 rows in set (0.000 sec)
```

17) Insert one row in payment table where remarks should not be entered.



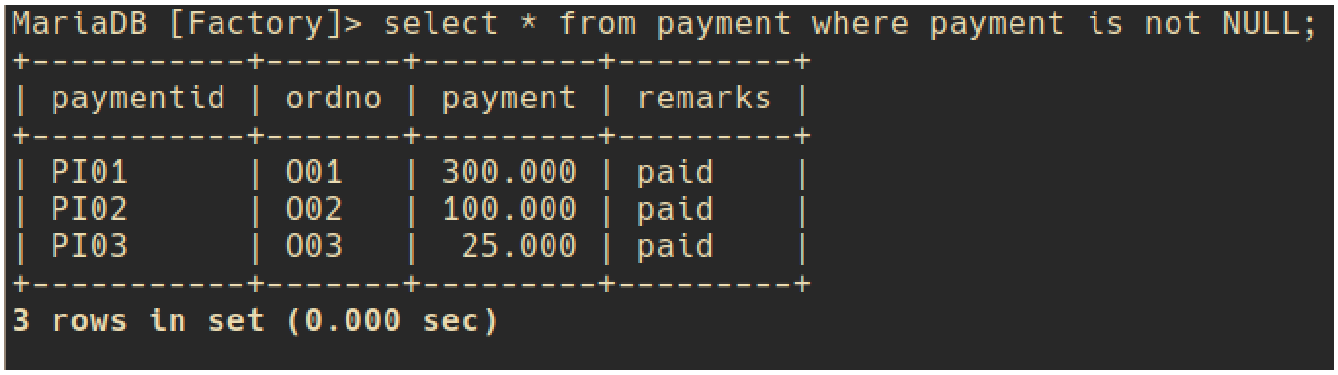
```
MariaDB [Factory]> insert into payment(paymentid, ordno, payment, remarks)  values('PI04', '003', NULL, NULL);
Query OK, 1 row affected (0.030 sec)

MariaDB [Factory]> select * from payment;
+-----------+-------+---------+---------+
| paymentid | ordno | payment | remarks |
+-----------+-------+---------+---------+
| PI01      | 001   | 300.000 | paid    |
| PI02      | 002   | 100.000 | paid    |
| PI03      | 003   |  25.000 | paid    |
| PI04      | 003   |    NULL | NULL    |
+-----------+-------+---------+---------+
4 rows in set (0.000 sec)
```

18) Display those payments which are not to be paid (having NULL).



```
MariaDB [Factory]> select * from payment where payment is NULL;
+-----------+-------+---------+---------+
| paymentid | ordno | payment | remarks |
+-----------+-------+---------+---------+
| PI04      | 003   |    NULL | NULL    |
+-----------+-------+---------+---------+
1 row in set (0.001 sec)
```

19) Display those payments which are not null.



```
MariaDB [Factory]> select * from payment where payment is not NULL;
+-----------+-------+---------+---------+
| paymentid | ordno | payment | remarks |
+-----------+-------+---------+---------+
| PI01      | 001   | 300.000 | paid    |
| PI02      | 002   | 100.000 | paid    |
| PI03      | 003   |  25.000 | paid    |
+-----------+-------+---------+---------+
3 rows in set (0.000 sec)
```

20)

a)

DELETE	DROP
Deletes rows and columns from a table.	Deletes the whole table or database.
DML command.	DDL command.

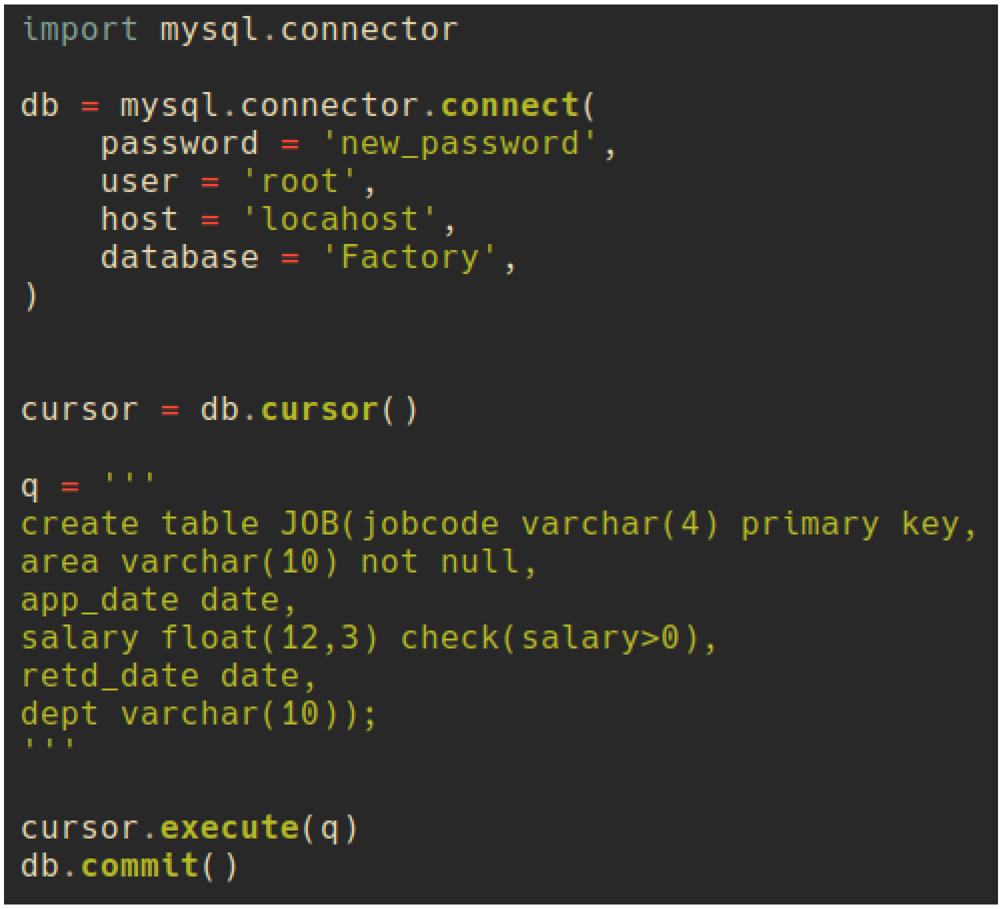
b)

TABLE	DEGREE	CARDINALITY
Product	3	3
Orders	4	3
Payment	4	4

Q3-A:

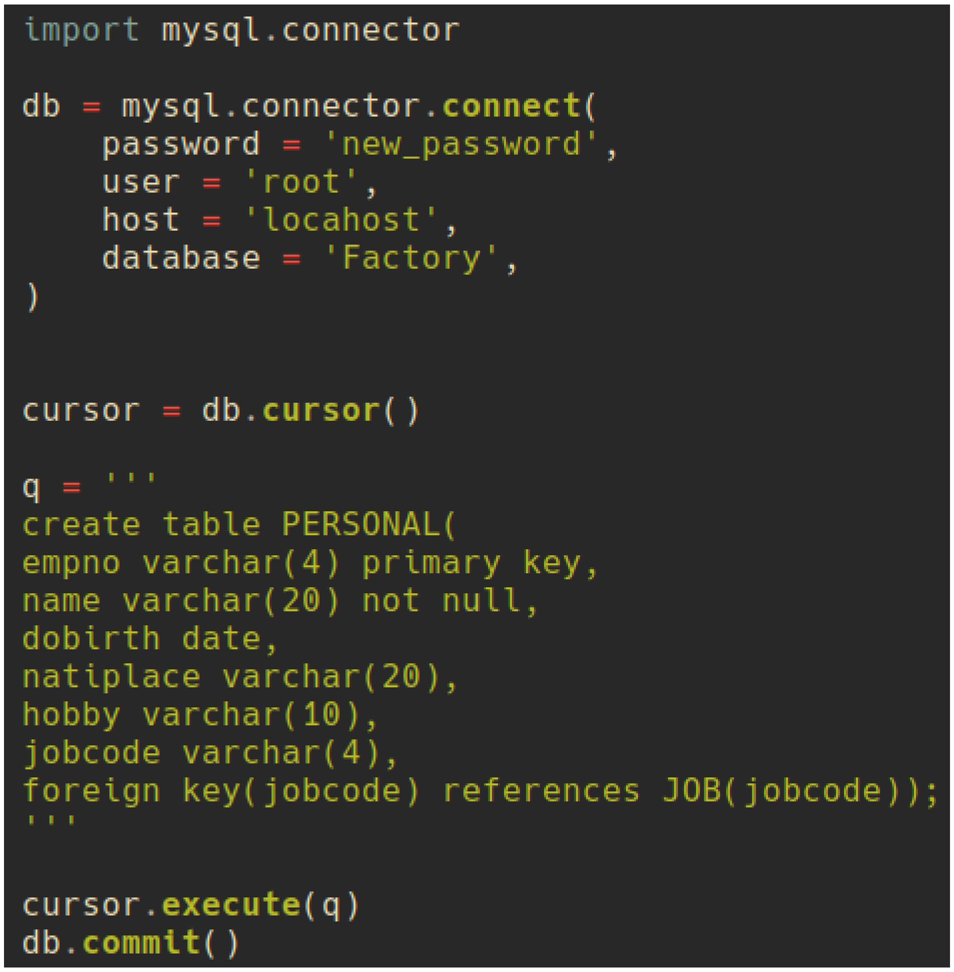
1) Create the following two tables:

a) JOB (jobcode text primary key, area text not null, app_date ,salary decimal should be positive, retd_date, dept)



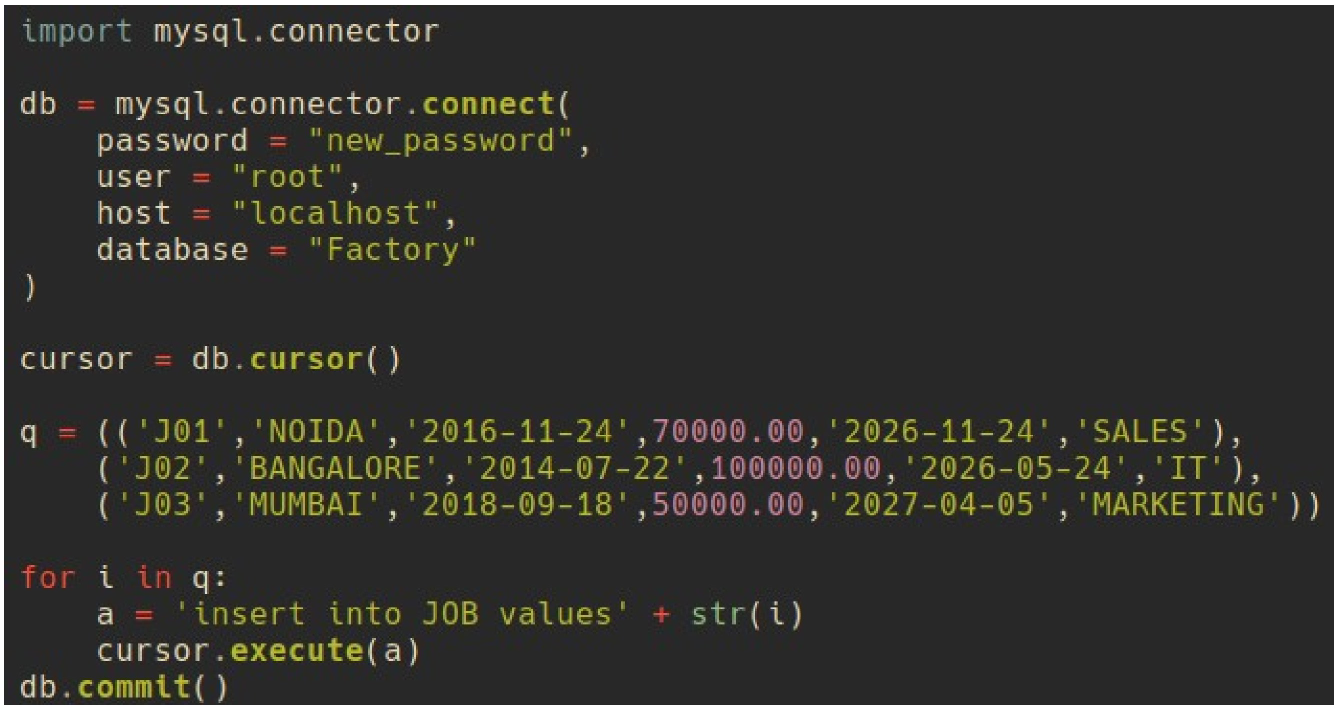
```
import mysql.connector

db = mysql.connector.connect(
    password = 'new_password',
    user = 'root',
    host = 'locahost',
    database = 'Factory',
)


cursor = db.cursor()

q = '''
create table JOB(jobcode varchar(4) primary key,
area varchar(10) not null,
app_date date,
salary float(12,3) check(salary>0),
retd_date date,
dept varchar(10));
'''

cursor.execute(q)
db.commit()
```

b) PERSONAL(empno text primary key, name text not null, dobirth, nativeplace text, hobby text ,jobcode text and foreign key)

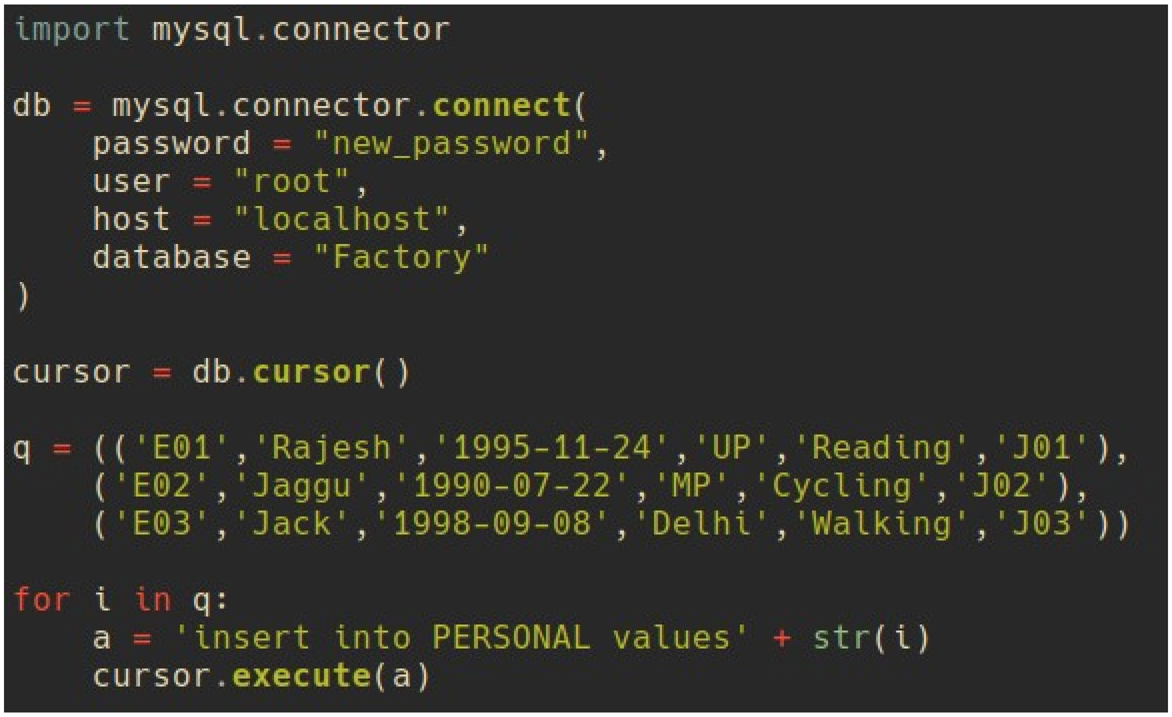


```
import mysql.connector

db = mysql.connector.connect(
    password = 'new_password',
    user = 'root',
    host = 'locahost',
    database = 'Factory',
)


cursor = db.cursor()

q = '''
create table PERSONAL(
empno varchar(4) primary key,
name varchar(20) not null,
dobirth date,
natiplace varchar(20),
hobby varchar(10),
jobcode varchar(4),
foreign key(jobcode) references JOB(jobcode));
'''

cursor.execute(q)
db.commit()
```

c) Insert three rows in both the tables.

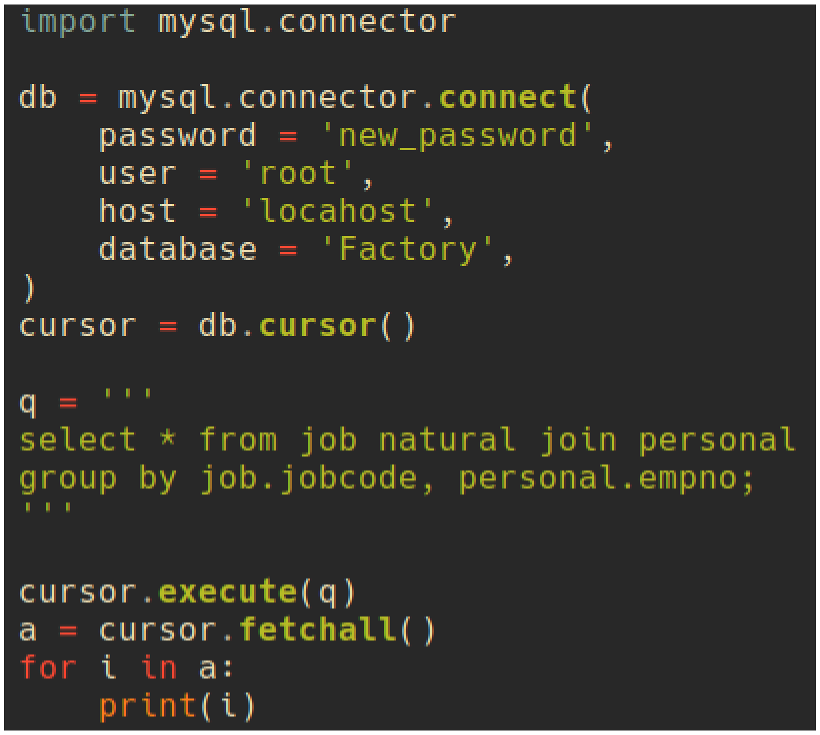


```python
import mysql.connector

db = mysql.connector.connect(
    password = "new_password",
    user = "root",
    host = "localhost",
    database = "Factory"
)

cursor = db.cursor()

q = (('J01','NOIDA','2016-11-24',70000.00,'2026-11-24','SALES'),
    ('J02','BANGALORE','2014-07-22',100000.00,'2026-05-24','IT'),
    ('J03','MUMBAI','2018-09-18',50000.00,'2027-04-05','MARKETING'))

for i in q:
    a = 'insert into JOB values' + str(i)
    cursor.execute(a)
db.commit()
```



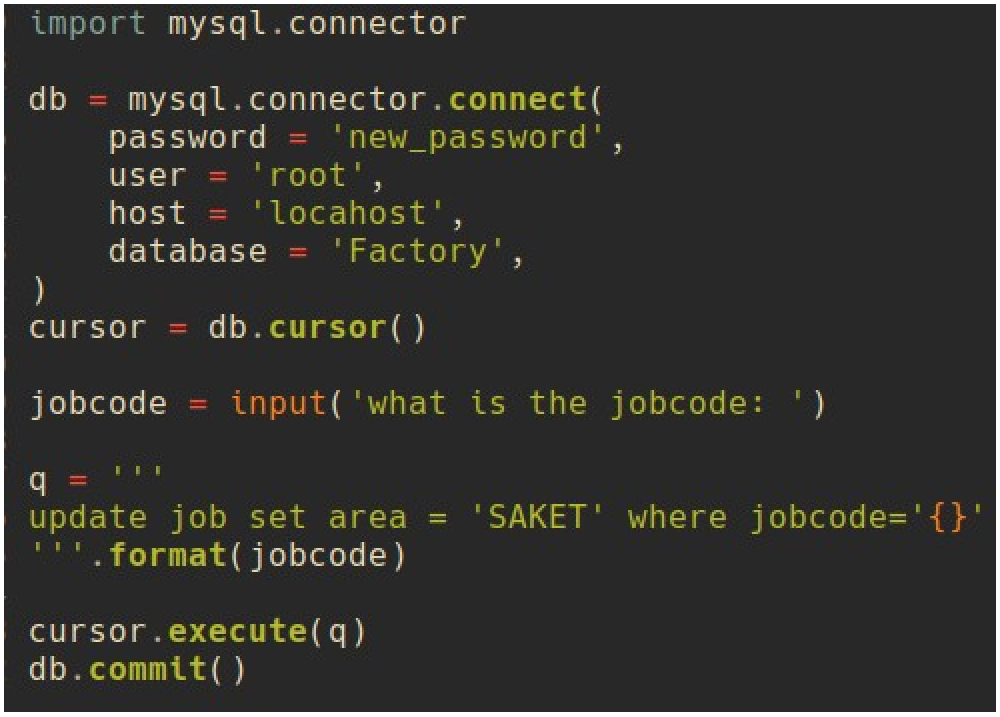
```python
import mysql.connector

db = mysql.connector.connect(
    password = "new_password",
    user = "root",
    host = "localhost",
    database = "Factory"
)

cursor = db.cursor()

q = (('E01','Rajesh','1995-11-24','UP','Reading','J01'),
    ('E02','Jaggu','1990-07-22','MP','Cycling','J02'),
    ('E03','Jack','1998-09-08','Delhi','Walking','J03'))

for i in q:
    a = 'insert into PERSONAL values' + str(i)
    cursor.execute(a)
```

d) Display the entire data in the order of jobcode from JOB and empno from PERSONAL.



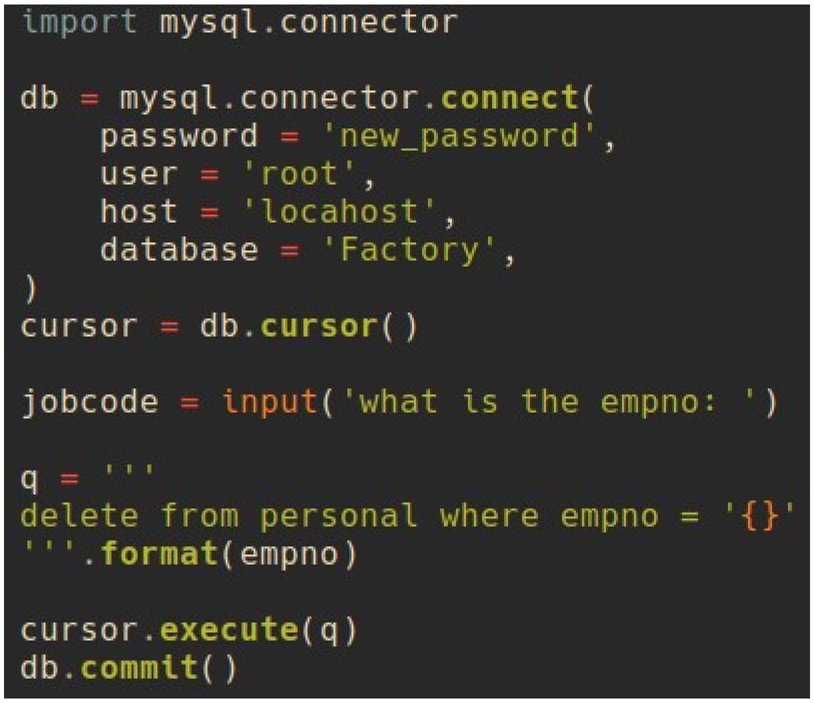
```
import mysql.connector

db = mysql.connector.connect(
    password = 'new_password',
    user = 'root',
    host = 'locahost',
    database = 'Factory',
)
cursor = db.cursor()

q = '''
select * from job natural join personal
group by job.jobcode, personal.empno;
'''

cursor.execute(q)
a = cursor.fetchall()
for i in a:
    print(i)
```

e) Enter the value of jobcode to update the area with "Saket New Delhi ".



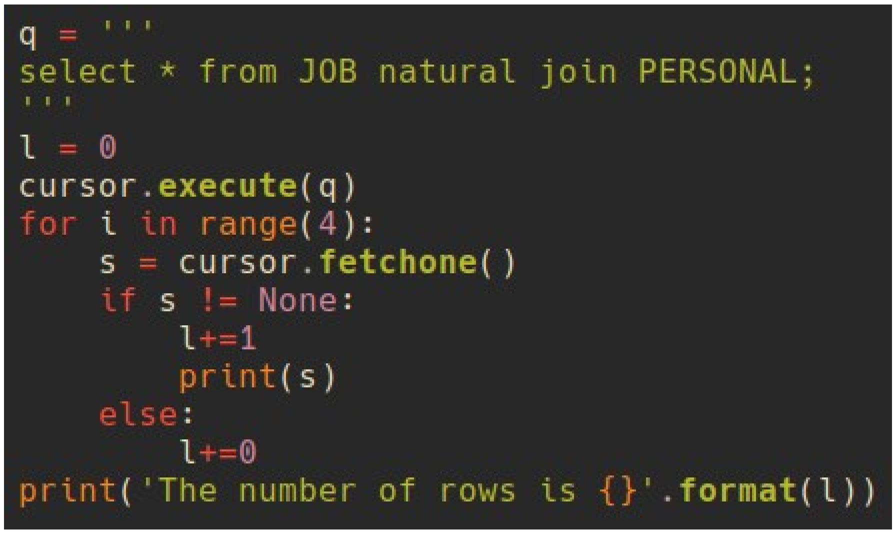
```
import mysql.connector

db = mysql.connector.connect(
    password = 'new_password',
    user = 'root',
    host = 'locahost',
    database = 'Factory',
)
cursor = db.cursor()

jobcode = input('what is the jobcode: ')

q = '''
update job set area = 'SAKET' where jobcode='{}'
'''.format(jobcode)

cursor.execute(q)
db.commit()
```

f) Enter the value of empno to delete the record from the table.



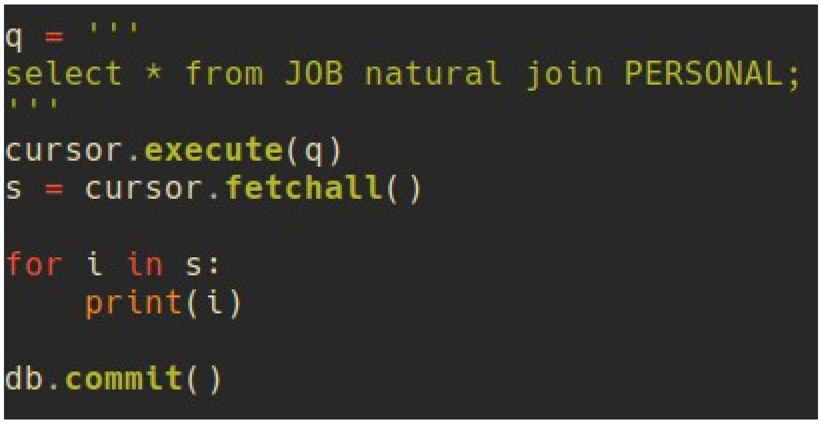
```
import mysql.connector

db = mysql.connector.connect(
    password = 'new_password',
    user = 'root',
    host = 'locahost',
    database = 'Factory',
)
cursor = db.cursor()

jobcode = input('what is the empno: ')

q = '''
delete from personal where empno = '{}'
'''.format(empno)

cursor.execute(q)
db.commit()
```

g) Fetch one by one record from the result set and display on the screen. Also display the number of rows retrieved from the resultset.



```
q = '''
select * from JOB natural join PERSONAL;
'''
l = 0
cursor.execute(q)
for i in range(4):
    s = cursor.fetchone()
    if s != None:
        l+=1
        print(s)
    else:
        l+=0
print('The number of rows is {}'.format(l))
```

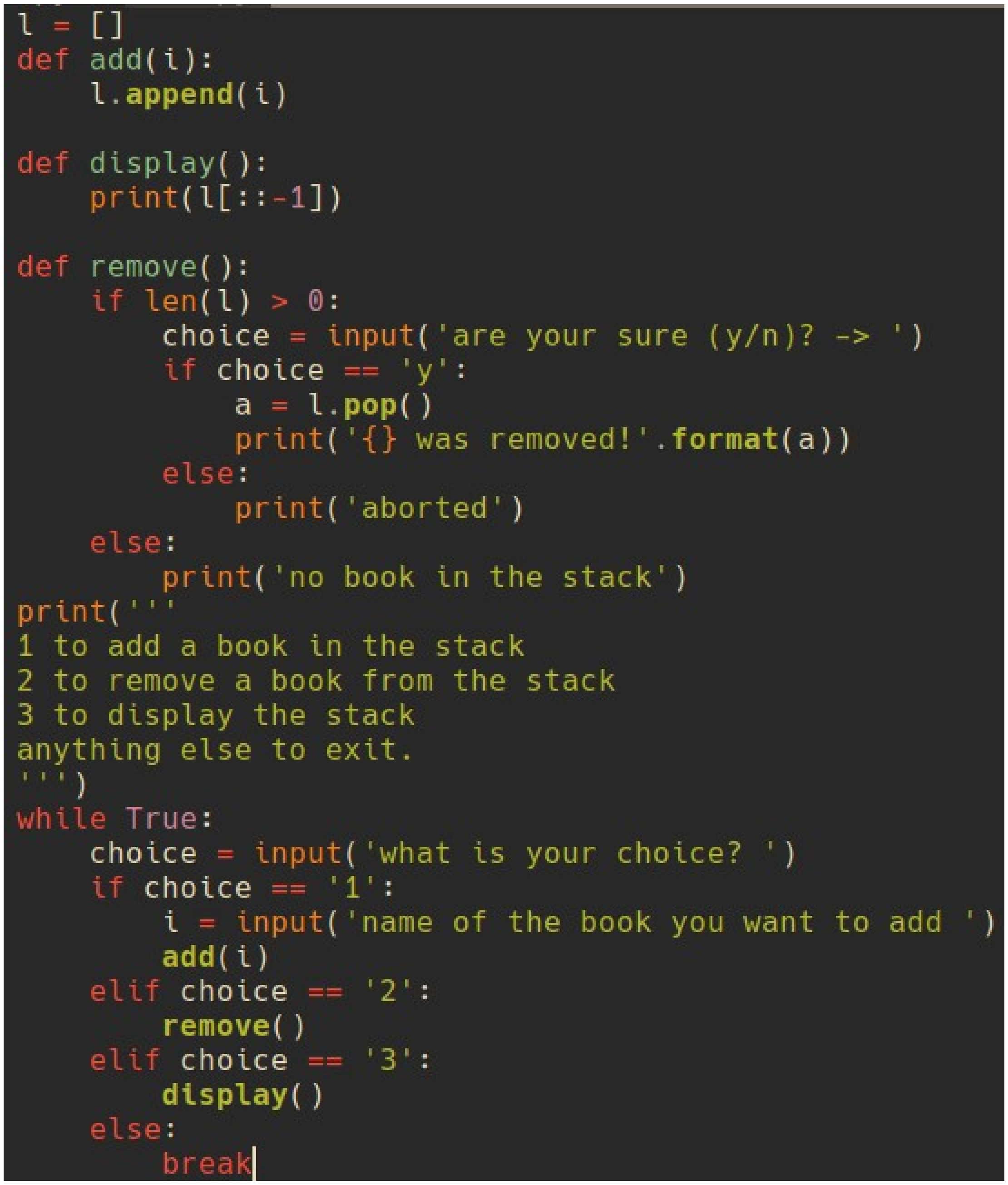
h) Fetch all the rows from the result set and display.



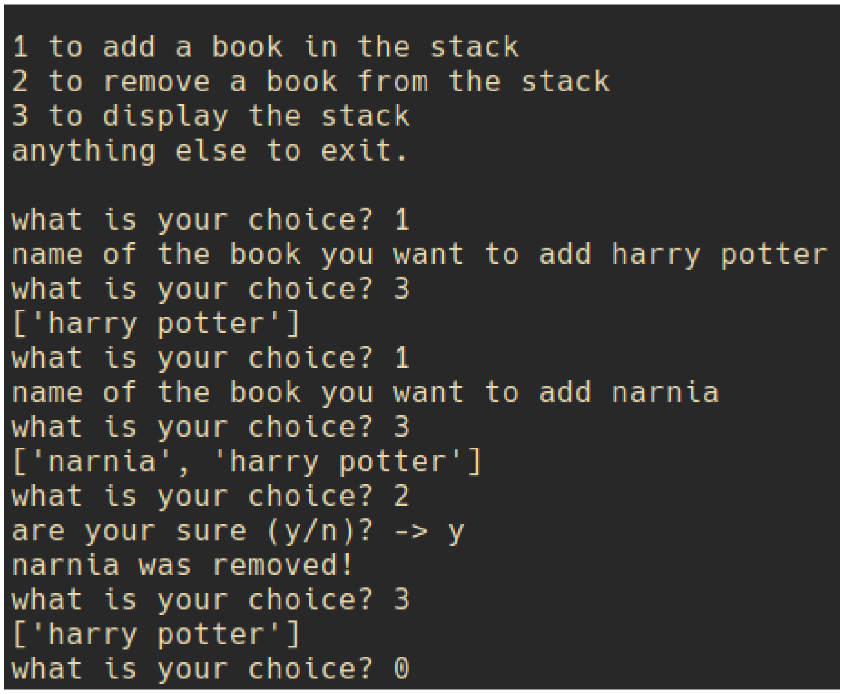
```
q = '''
select * from JOB natural join PERSONAL;
'''
cursor.execute(q)
s = cursor.fetchall()

for i in s:
    print(i)

db.commit()
```

SECTION – 1 (PYTHON PROGRAM)

Q19: Considering a list of book details (bookno and name), write a program to implement STACK defining following three functions: 1. Add a book in stack 2. Delete a book from stack 3. Display the elements of stack

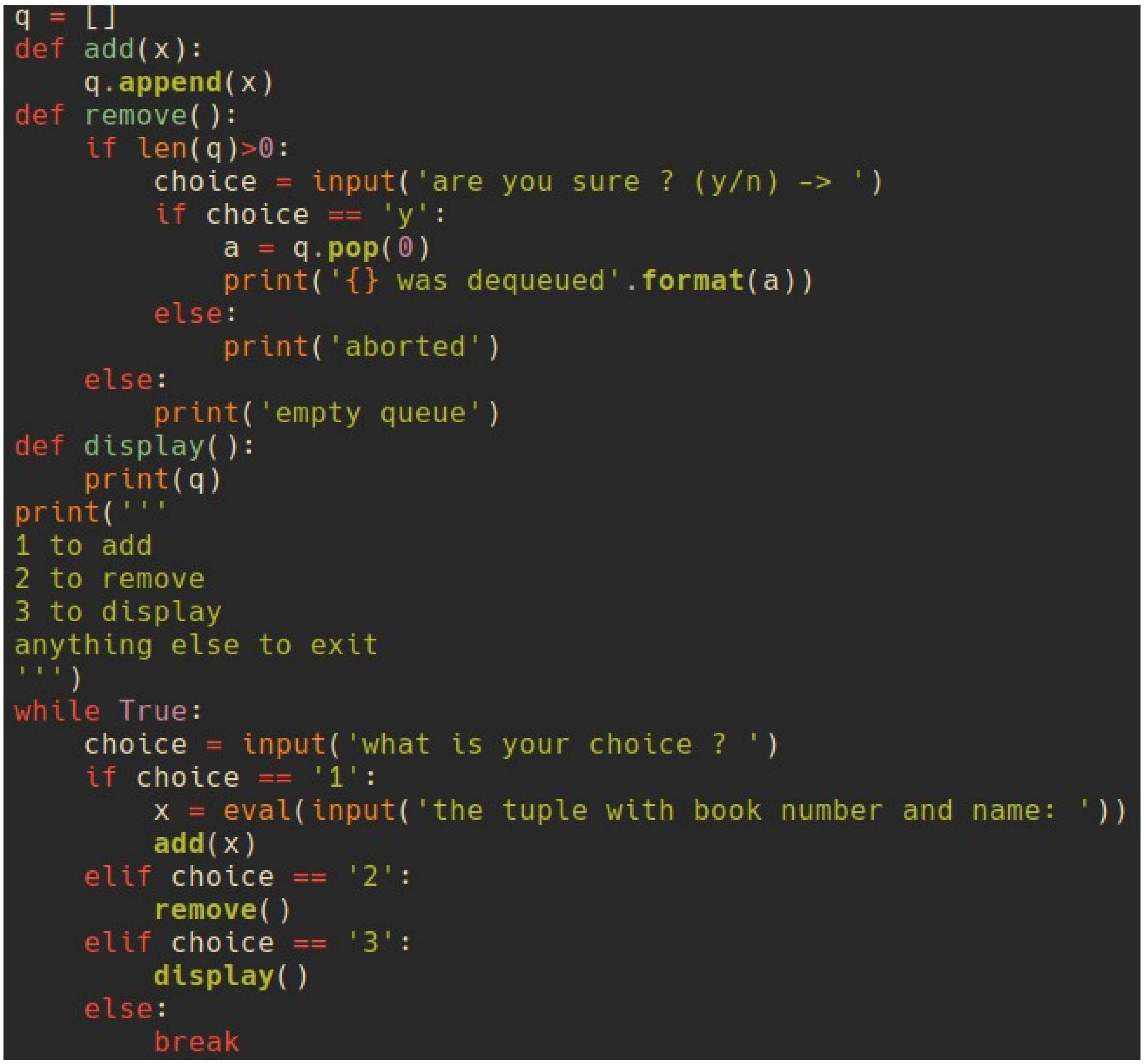


```python
l = []
def add(i):
    l.append(i)

def display():
    print(l[::-1])

def remove():
    if len(l) > 0:
        choice = input('are your sure (y/n)? -> ')
        if choice == 'y':
            a = l.pop()
            print('{} was removed!'.format(a))
        else:
            print('aborted')
    else:
        print('no book in the stack')
print('''
1 to add a book in the stack
2 to remove a book from the stack
3 to display the stack
anything else to exit.
''')
while True:
    choice = input('what is your choice? ')
    if choice == '1':
        i = input('name of the book you want to add ')
        add(i)
    elif choice == '2':
        remove()
    elif choice == '3':
        display()
    else:
        break
```

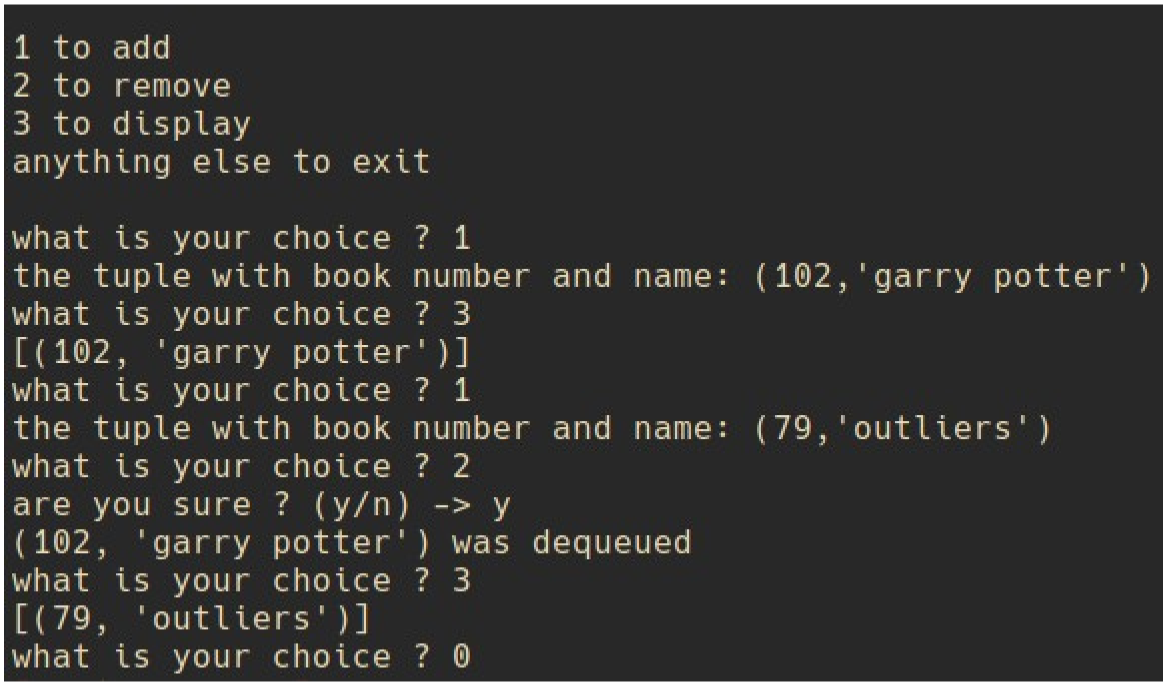


```
1 to add a book in the stack
2 to remove a book from the stack
3 to display the stack
anything else to exit.

what is your choice? 1
name of the book you want to add harry potter
what is your choice? 3
['harry potter']
what is your choice? 1
name of the book you want to add narnia
what is your choice? 3
['narnia', 'harry potter']
what is your choice? 2
are your sure (y/n)? -> y
narnia was removed!
what is your choice? 3
['harry potter']
what is your choice? 0
```

Q20 Considering a list of book details (bookno and name), write a program to implement QUEUE defining following three functions: 1 Add a book, 2 Delete a book and 3 Display the queue.



```
q = []
def add(x):
    q.append(x)
def remove():
    if len(q)>0:
        choice = input('are you sure ? (y/n) -> ')
        if choice == 'y':
            a = q.pop(0)
            print('{} was dequeued'.format(a))
        else:
            print('aborted')
    else:
        print('empty queue')
def display():
    print(q)
print('''
1 to add
2 to remove
3 to display
anything else to exit
''')
while True:
    choice = input('what is your choice ? ')
    if choice == '1':
        x = eval(input('the tuple with book number and name: '))
        add(x)
    elif choice == '2':
        remove()
    elif choice == '3':
        display()
    else:
        break
```



```
1 to add
2 to remove
3 to display
anything else to exit

what is your choice ? 1
the tuple with book number and name: (102,'garry potter')
what is your choice ? 3
[(102, 'garry potter')]
what is your choice ? 1
the tuple with book number and name: (79,'outliers')
what is your choice ? 2
are you sure ? (y/n) -> y
(102, 'garry potter') was dequeued
what is your choice ? 3
[(79, 'outliers')]
what is your choice ? 0
```