Assignment-8

Name:Maddula Rupa Sri Manohar

E-mail:maddularupasrimanohar2001@gmail.com

Assignment 1: Write a SELECT query to retrieve all columns from a 'customers' table, and modify it to return only the customer name and email address for customers in a specific city.

Query 1: Retrieve all columns from the 'customers' table

SELECT * FROM customers;

Query 2: Retrieve customer name and email address for customers in a specific city.

SELECT Fname, Lname, email FROM customers WHERE city = 'chennai';

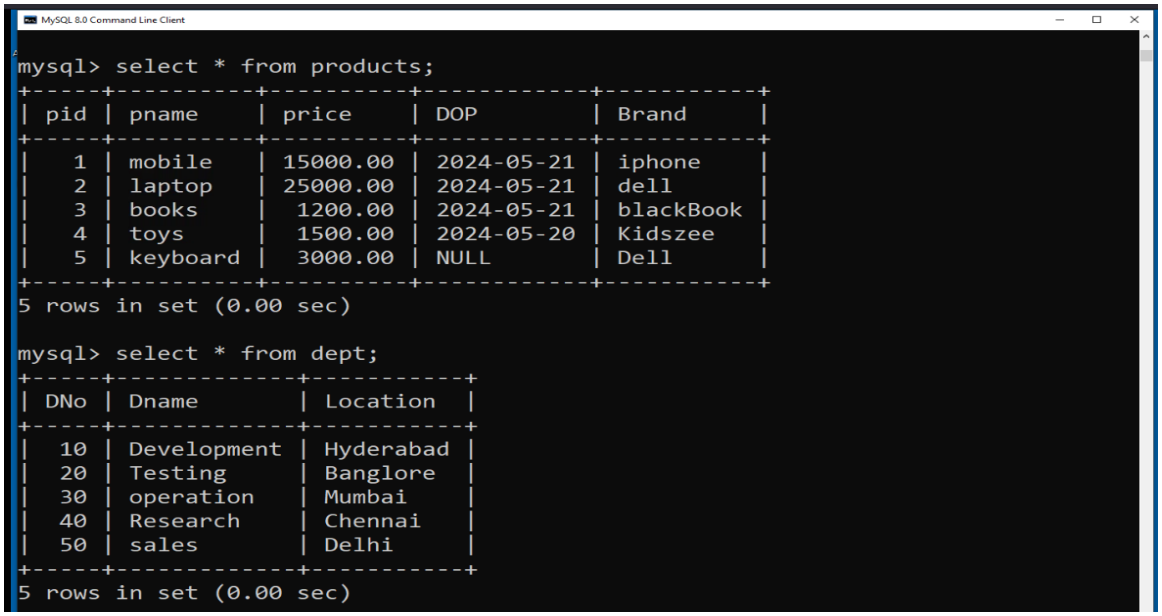
```
mysql> show tables;
+-----------------------------+
| Tables_in_librarymanagement |
+-----------------------------+
| accounts                    |
| authors                     |
| books                       |
| borrowedbooks               |
| customers                   |
| members                     |
| products                    |
+-----------------------------+
7 rows in set (0.01 sec)

mysql> desc customers;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| cust_id | int         | NO   | PRI | NULL    |       |
| Fname   | varchar(15) | NO   |     | NULL    |       |
| Lname   | varchar(15) | NO   |     | NULL    |       |
| email   | varchar(50) | NO   |     | NULL    |       |
| city    | varchar(30) | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
5 rows in set (0.01 sec)

mysql> select * from customers;
```

```
mysql> select * from customers;
+---------+-------+-------+-----------------+---------+
| cust_id | Fname | Lname | email           | city    |
+---------+-------+-------+-----------------+---------+
|       1 | Jonh  | gon   | jonh@example.com | hyd     |
|       2 | rose  | simth | rose@example.com | chennai |
|       3 | Tom   | jerry | tom@example.com  | Delhi   |
+---------+-------+-------+-----------------+---------+
3 rows in set (0.00 sec)

mysql> select fname , email from customers where city = 'chennai';
+-------+-----------------+
| fname | email           |
+-------+-----------------+
| rose  | rose@example.com |
+-------+-----------------+
1 row in set (0.00 sec)

mysql> select fname ,lname, email from customers where city = 'chennai';
+-------+-------+-----------------+
| fname | lname | email           |
+-------+-------+-----------------+
| rose  | simth | rose@example.com |
+-------+-------+-----------------+
1 row in set (0.00 sec)

mysql>
```


Assignment 2: Craft a query using an INNER JOIN to combine 'orders' and 'customers' tables for customers in a specified region, and a LEFT JOIN to display all customers including those without orders.

Here I have create employee table and dept table .

Inserted some records into employee table and dept table.



Inner Join:

Select eid,ename,Salary,job,d.dno,dname,location from employee E INNER JOIN Dept D where E.dno = D.dno;

Select eid,ename,Salary,job,d.dno,dname,location from employee E INNER JOIN Dept D where E.dno != D.dno;

```
mysql> select eid,ename, salary,job,d.dno,dname, location from employee E INNER
JOIN Dept D where E.dno = D.dno;
+-----+-------+----------+-----------+-----+-------------+-----------+
| eid | ename | salary   | job       | dno | dname       | location  |
+-----+-------+----------+-----------+-----+-------------+-----------+
| 101 | King  | 50000.00 | President |  40 | Research    | Chennai   |
| 102 | smith | 45000.00 | Manager   |  10 | Development | Hyderabad |
| 103 | Ford  | 40000.00 | Manager   |  20 | Testing     | Banglore  |
| 104 | Tom   | 30000.00 | Developer |  10 | Development | Hyderabad |
| 105 | Scott | 35000.00 | Developer |  10 | Development | Hyderabad |
| 106 | Jerry | 25000.00 | Tester    |  20 | Testing     | Banglore  |
| 107 | Ravi  | 22000.00 | Tester    |  20 | Testing     | Banglore  |
| 108 | Adam  | 47000.00 | Analayst  |  30 | operation   | Mumbai    |
+-----+-------+----------+-----------+-----+-------------+-----------+
8 rows in set (0.00 sec)

mysql> select eid,ename, salary,job,d.dno,dname, location from employee E INNER
JOIN Dept D where E.dno != D.dno;
+-----+-------+----------+-----------+-----+-------------+-----------+
| eid | ename | salary   | job       | dno | dname       | location  |
+-----+-------+----------+-----------+-----+-------------+-----------+
| 101 | King  | 50000.00 | President |  50 | sales       | Delhi     |
| 101 | King  | 50000.00 | President |  30 | operation   | Mumbai    |
| 101 | King  | 50000.00 | President |  20 | Testing     | Banglore  |
| 101 | King  | 50000.00 | President |  10 | Development | Hyderabad |
| 102 | smith | 45000.00 | Manager   |  50 | sales       | Delhi     |
+-----+-------+----------+-----------+-----+-------------+-----------+
```


Left Outer Join:

Select eid,ename,salary,job,D.dno,dname,location from employee E left outer join Dept D ON (E.DNo = D.DNo);

```
32 rows in set (0.00 sec)

mysql> select eid,ename,salary,job,D.dno,dname,location from employee E LEFT OUT
ER JOIN Dept D ON (E.Dno = D.Dno);
+-----+-------+----------+-----------+------+-------------+-----------+
| eid | ename | salary   | job       | dno  | dname       | location  |
+-----+-------+----------+-----------+------+-------------+-----------+
| 101 | King  | 50000.00 | President |   40 | Research    | Chennai   |
| 102 | smith | 45000.00 | Manager   |   10 | Development | Hyderabad |
| 103 | Ford  | 40000.00 | Manager   |   20 | Testing     | Banglore  |
| 104 | Tom   | 30000.00 | Developer |   10 | Development | Hyderabad |
| 105 | Scott | 35000.00 | Developer |   10 | Development | Hyderabad |
| 106 | Jerry | 25000.00 | Tester    |   20 | Testing     | Banglore  |
| 107 | Ravi  | 22000.00 | Tester    |   20 | Testing     | Banglore  |
| 108 | Adam  | 47000.00 | Analayst  |   30 | operation   | Mumbai    |
| 110 | Anil  |     NULL | NULL      | NULL | NULL        | NULL      |
+-----+-------+----------+-----------+------+-------------+-----------+
9 rows in set (0.00 sec)
```

Assignment 3: Utilize a subquery to find customers who have placed orders above the average order value, and write a UNION query to combine two SELECT statements with the same number of columns.

Here I have create employee table and dept table .

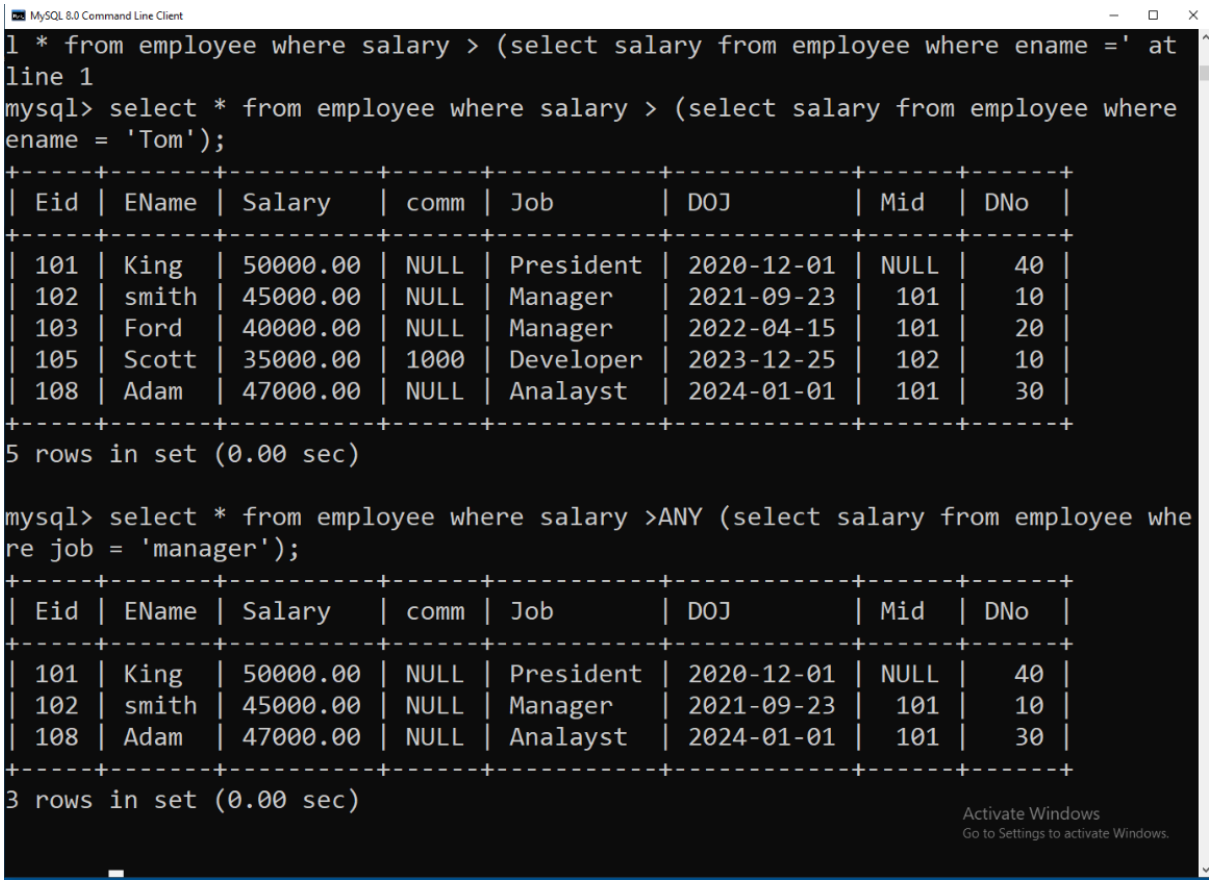
Inserted some records into employee table and dept table.

SubQuery:

sub query a query inside another query

select * from employee where salary > (select salary from employee where ename = ‘Tom’);

select * from employee where salary >ANY (Select salary from employee where job = ‘manager’);



select * from employee where salary <ANY (Select salary from employee where job = ‘manager’);

select * from employee where salary >ALL (Select salary from employee where job = ‘manager’);

select * from employee where salary <ALL (Select salary from employee where job = ‘manager’);

```
mysql> select * from employee where salary <ANY (select salary from employee whe
re job = 'manager');
+-----+-------+----------+------+-----------+------------+------+------+
| Eid | EName | Salary   | comm | Job       | DOJ        | Mid  | DNo  |
+-----+-------+----------+------+-----------+------------+------+------+
| 103 | Ford  | 40000.00 | NULL | Manager   | 2022-04-15 |  101 |   20 |
| 104 | Tom   | 30000.00 | 1500 | Developer | 2023-10-18 |  102 |   10 |
| 105 | Scott | 35000.00 | 1000 | Developer | 2023-12-25 |  102 |   10 |
| 106 | Jerry | 25000.00 | 3000 | Tester    | 2024-05-20 |  103 |   20 |
| 107 | Ravi  | 22000.00 | 4000 | Tester    | 2024-05-19 |  103 |   20 |
+-----+-------+----------+------+-----------+------------+------+------+
5 rows in set (0.00 sec)

mysql> select * from employee where salary <ALL (select salary from employee whe
re job = 'manager');
+-----+-------+----------+------+-----------+------------+------+------+
| Eid | EName | Salary   | comm | Job       | DOJ        | Mid  | DNo  |
+-----+-------+----------+------+-----------+------------+------+------+
| 104 | Tom   | 30000.00 | 1500 | Developer | 2023-10-18 |  102 |   10 |
| 105 | Scott | 35000.00 | 1000 | Developer | 2023-12-25 |  102 |   10 |
| 106 | Jerry | 25000.00 | 3000 | Tester    | 2024-05-20 |  103 |   20 |
| 107 | Ravi  | 22000.00 | 4000 | Tester    | 2024-05-19 |  103 |   20 |
+-----+-------+----------+------+-----------+------------+------+------+
4 rows in set (0.00 sec)
```

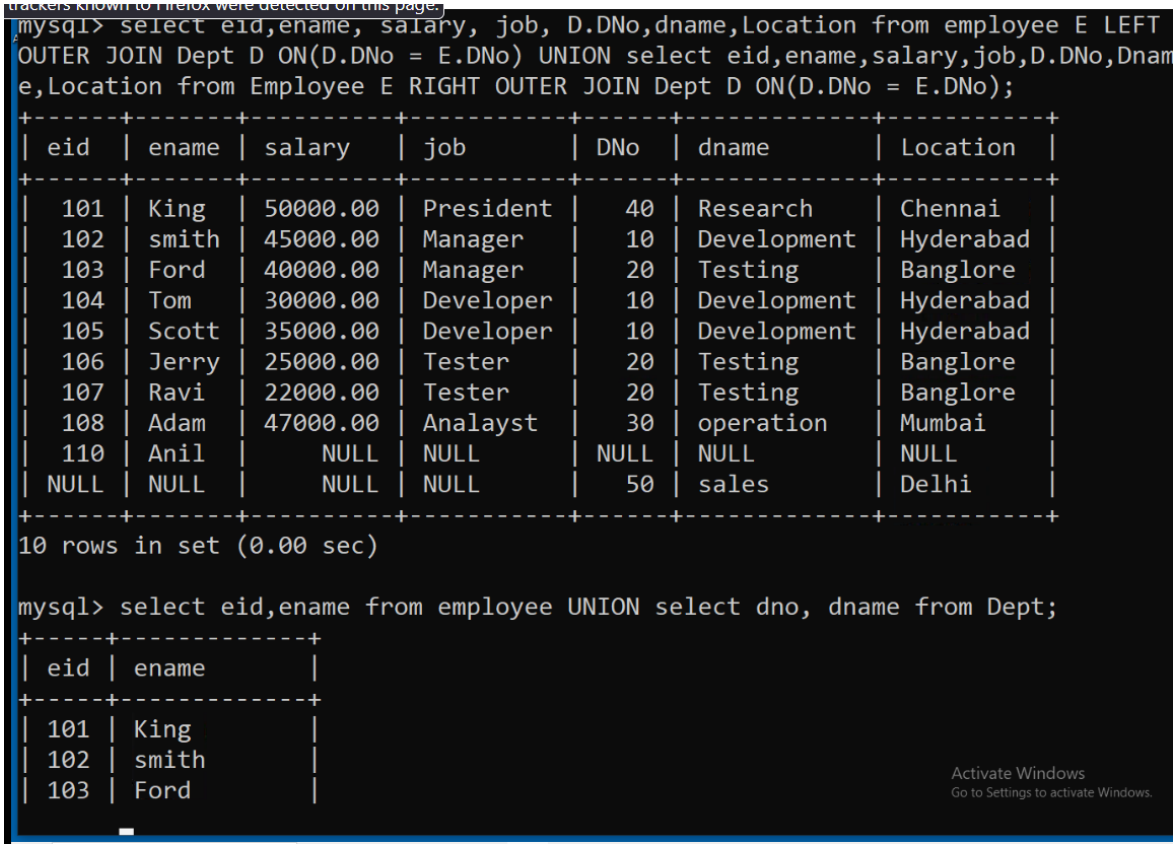


```
Database changed
mysql> select * from employee where DNo = (select DNo from dept where location =
(select location from Dept where Dname = 'Research'));
+-----+-------+----------+------+-----------+------------+------+------+
| Eid | EName | Salary   | comm | Job       | DOJ        | Mid  | DNo  |
+-----+-------+----------+------+-----------+------------+------+------+
| 101 | King  | 50000.00 | NULL | President | 2020-12-01 | NULL |   40 |
+-----+-------+----------+------+-----------+------------+------+------+
1 row in set (0.00 sec)
```

Union:

Select eid,ename ,salary ,job,D.dno,dname,location from employee E Left outer Join dept D ON(D.No = E.No) UNION select eid, ename, salary, job,D.DNo , dname, location from employee E Right outer Join Dept ON(D.No =E.No);

Select eid, ename from employee UNION select dno, dname from Dept;



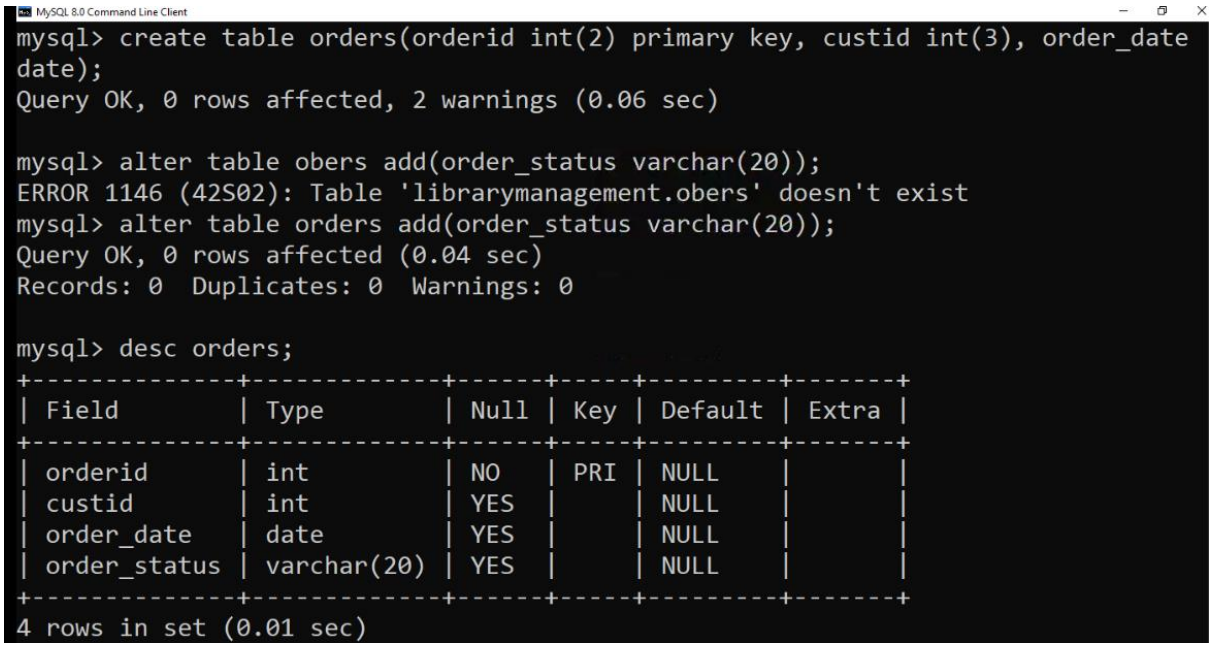
```
mysql> select eid,ename, salary, job, D.DNo,dname,Location from employee E LEFT
OUTER JOIN Dept D ON(D.DNo = E.DNo) UNION select eid,ename,salary,job,D.DNo,Dnam
e,Location from Employee E RIGHT OUTER JOIN Dept D ON(D.DNo = E.DNo);
+------+-------+----------+-----------+------+-------------+-----------+
| eid  | ename | salary   | job       | DNo  | dname       | Location  |
+------+-------+----------+-----------+------+-------------+-----------+
|  101 | King  | 50000.00 | President |   40 | Research    | Chennai   |
|  102 | smith | 45000.00 | Manager   |   10 | Development | Hyderabad |
|  103 | Ford  | 40000.00 | Manager   |   20 | Testing     | Banglore  |
|  104 | Tom   | 30000.00 | Developer |   10 | Development | Hyderabad |
|  105 | Scott | 35000.00 | Developer |   10 | Development | Hyderabad |
|  106 | Jerry | 25000.00 | Tester    |   20 | Testing     | Banglore  |
|  107 | Ravi  | 22000.00 | Tester    |   20 | Testing     | Banglore  |
|  108 | Adam  | 47000.00 | Analayst  |   30 | operation   | Mumbai    |
|  110 | Anil  |     NULL | NULL      | NULL | NULL        | NULL      |
| NULL | NULL  |     NULL | NULL      |   50 | sales       | Delhi     |
+------+-------+----------+-----------+------+-------------+-----------+
10 rows in set (0.00 sec)

mysql> select eid,ename from employee UNION select dno, dname from Dept;
+-----+-------------+
| eid | ename       |
+-----+-------------+
| 101 | King        |
| 102 | smith       |
| 103 | Ford        |
```


Assignment 4: Compose SQL statements to BEGIN a transaction, INSERT a new record into the 'orders' table, COMMIT the transaction, then UPDATE the 'products' table, and ROLLBACK the transaction.

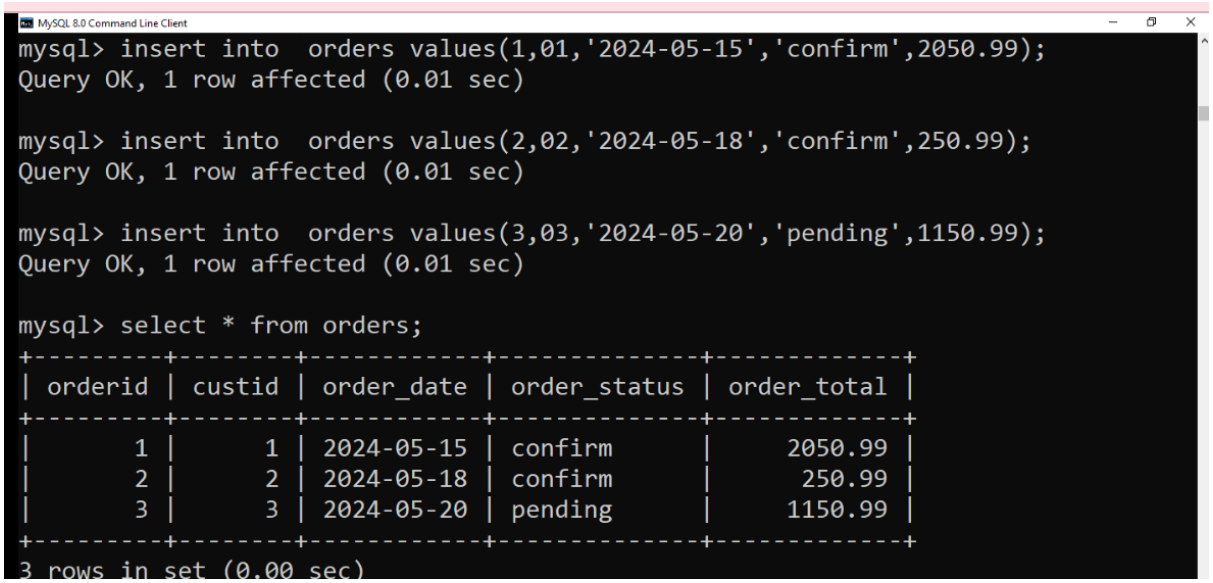
Create orders table and insert values into orders table:

```
CREATE TABLE orders (
  order_id INT(2) PRIMARY Key,
  customer_id INT(3),
  order_date DATE ,
  order_total DECIMAL(10,2) ,
  order_status VARCHAR(50)
);
```

Insert into orders values(1,01,'2024-05-15', 2025.99, 'confirm'), (2,02,'2024-05-18' 250.99,'confirm'),(3,03,'2024-05-20',1050.00,'pending'));



```
MySQL 8.0 Command Line Client
mysql> create table orders(orderid int(2) primary key, custid int(3), order_date
date);
Query OK, 0 rows affected, 2 warnings (0.06 sec)

mysql> alter table obers add(order_status varchar(20));
ERROR 1146 (42S02): Table 'librarymanagement.obers' doesn't exist
mysql> alter table orders add(order_status varchar(20));
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc orders;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| orderid      | int         | NO   | PRI | NULL    |       |
| custid       | int         | YES  |     | NULL    |       |
| order_date   | date        | YES  |     | NULL    |       |
| order_status | varchar(20) | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
4 rows in set (0.01 sec)
```



```
MySQL 8.0 Command Line Client
mysql> insert into  orders values(1,01,'2024-05-15','confirm',2050.99);
Query OK, 1 row affected (0.01 sec)

mysql> insert into  orders values(2,02,'2024-05-18','confirm',250.99);
Query OK, 1 row affected (0.01 sec)

mysql> insert into  orders values(3,03,'2024-05-20','pending',1150.99);
Query OK, 1 row affected (0.01 sec)

mysql> select * from orders;
+---------+--------+------------+--------------+-------------+
| orderid | custid | order_date | order_status | order_total |
+---------+--------+------------+--------------+-------------+
|       1 |      1 | 2024-05-15 | confirm      |     2050.99 |
|       2 |      2 | 2024-05-18 | confirm      |      250.99 |
|       3 |      3 | 2024-05-20 | pending      |     1150.99 |
+---------+--------+------------+--------------+-------------+
3 rows in set (0.00 sec)
```

Create a products table and insert values into products table :

Create table products(prod_id int(5) primary key, pname varchar(20), category varchar(50), pprice decimal(7,2));

Insert into products values(1,'dress','clothing',1050.00), (2,'eyeliner','cosmetics',1050.99);

```
Empty set (0.00 sec)

mysql> desc products;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| prod_id  | int          | NO   | PRI | NULL    |       |
| Pname    | varchar(50)  | YES  |     | NULL    |       |
| category | varchar(50)  | YES  |     | NULL    |       |
| pprice   | decimal(9,2) | YES  |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
4 rows in set (0.01 sec)

mysql> insert into products values(01,'dress','clothing',1050.00);
Query OK, 1 row affected (0.01 sec)

mysql> insert into products values(02,'eyeliner','cosmetics',1050.00);
Query OK, 1 row affected (0.01 sec)

mysql> select * from products;
+---------+----------+-----------+---------+
| prod_id | Pname    | category  | pprice  |
+---------+----------+-----------+---------+
|       1 | dress    | clothing  | 1050.00 |
|       2 | eyeliner | cosmetics | 1050.00 |
+---------+----------+-----------+---------+
2 rows in set (0.00 sec)
```


Start transaction;

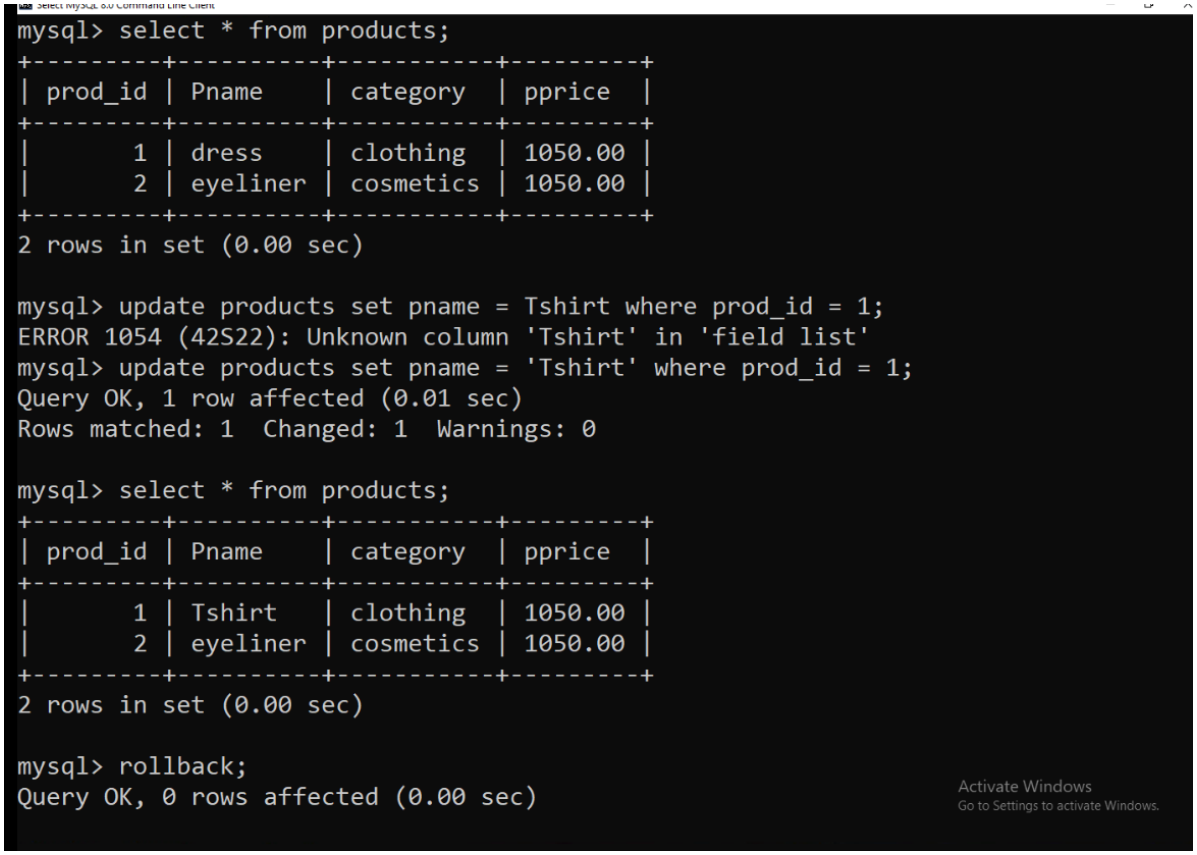
Insert into orders (orderid,custid,order_date order_total) values (4,4,'2024-05-21', 150.09);

Commit;

```
Select MySQL 8.0 Command Line Client
mysql> insert into orders (orderid,custid,order_date,order_total) values (4,4, '2
024-05-21' ,150.09) ;
Query OK, 1 row affected (0.00 sec)

mysql> select * from orders;
+---------+--------+------------+--------------+-------------+
| orderid | custid | order_date | order_status | order_total |
+---------+--------+------------+--------------+-------------+
|       1 |      1 | 2024-05-15 | confirm      |     2050.99 |
|       2 |      2 | 2024-05-18 | confirm      |      250.99 |
|       3 |      3 | 2024-05-20 | pending      |     1150.99 |
|       4 |      4 | 2024-05-21 | NULL         |      150.09 |
+---------+--------+------------+--------------+-------------+
4 rows in set (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from orders;
+---------+--------+------------+--------------+-------------+
| orderid | custid | order_date | order_status | order_total |
+---------+--------+------------+--------------+-------------+
|       1 |      1 | 2024-05-15 | confirm      |     2050.99 |
|       2 |      2 | 2024-05-18 | confirm      |      250.99 |
|       3 |      3 | 2024-05-20 | pending      |     1150.99 |
|       4 |      4 | 2024-05-21 | NULL         |      150.09 |
+---------+--------+------------+--------------+-------------+
```


Update products set pname = 'tshirt' where prod_id = 1;

Rollback;



Assignment 5: Begin a transaction, perform a series of INSERTs into 'orders', setting a SAVEPOINT after each, rollback to the second SAVEPOINT, and COMMIT the overall transaction.

Start transaction;

Insert into orders (orderid, custid,order_date,order_total) values(5,05,’2024-04-09’,’1000.00);

Savepoint savepoint_1;

Insert into orders (orderid, custid,order_date,order_total) values(6,06,’2024-04-18’,’1090.99);

Savepoint savepoint_2;

Rollback savepoint_2 // Any changes made after savepoint_2 will be rolled back.

Commit;

MySQL 8.0 Command Line Client

```
mysql> select * from orders;
+---------+--------+------------+--------------+-------------+
| orderid | custid | order_date | order_status | order_total |
+---------+--------+------------+--------------+-------------+
|       1 |      1 | 2024-05-15 | confirm      |     2050.99 |
|       2 |      2 | 2024-05-18 | confirm      |      250.99 |
|       3 |      3 | 2024-05-20 | pending      |     1150.99 |
|       4 |      4 | 2024-05-21 | NULL         |      150.09 |
+---------+--------+------------+--------------+-------------+
4 rows in set (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> insert into orders(orderid, custid, order_date, order_total) values(5,05,'
2024-04-9',1000.00);
Query OK, 1 row affected (0.00 sec)

mysql> select * from orders;
+---------+--------+------------+--------------+-------------+
| orderid | custid | order_date | order_status | order_total |
+---------+--------+------------+--------------+-------------+
|       1 |      1 | 2024-05-15 | confirm      |     2050.99 |
|       2 |      2 | 2024-05-18 | confirm      |      250.99 |
|       3 |      3 | 2024-05-20 | pending      |     1150.99 |
|       4 |      4 | 2024-05-21 | NULL         |      150.09 |
|       5 |      5 | 2024-04-09 | NULL         |     1000.00 |
+---------+--------+------------+--------------+-------------+
```



```
mysql> savepoint savepoint_1;
Query OK, 0 rows affected (0.00 sec)

mysql> insert into orders(orderid, custid, order_date, order_total) values(6,06,'
2024-04-16',1090.00);
Query OK, 1 row affected (0.00 sec)

mysql> select * from orders;
+---------+--------+------------+--------------+-------------+
| orderid | custid | order_date | order_status | order_total |
+---------+--------+------------+--------------+-------------+
|       1 |      1 | 2024-05-15 | confirm      |     2050.99 |
|       2 |      2 | 2024-05-18 | confirm      |      250.99 |
|       3 |      3 | 2024-05-20 | pending      |     1150.99 |
|       4 |      4 | 2024-05-21 | NULL         |      150.09 |
|       5 |      5 | 2024-04-09 | NULL         |     1000.00 |
|       6 |      6 | 2024-04-16 | NULL         |     1090.00 |
+---------+--------+------------+--------------+-------------+
6 rows in set (0.00 sec)

mysql> savepoint savepoint_2;
Query OK, 0 rows affected (0.00 sec)

mysql> rollback to savepoint_2;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from orders;
```

```
mysql> commit;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from orders;
+---------+--------+------------+--------------+-------------+
| orderid | custid | order_date | order_status | order_total |
+---------+--------+------------+--------------+-------------+
|       1 |      1 | 2024-05-15 | confirm      |     2050.99 |
|       2 |      2 | 2024-05-18 | confirm      |      250.99 |
|       3 |      3 | 2024-05-20 | pending      |     1150.99 |
|       4 |      4 | 2024-05-21 | NULL         |      150.09 |
|       5 |      5 | 2024-04-09 | NULL         |     1000.00 |
|       6 |      6 | 2024-04-16 | NULL         |     1090.00 |
+---------+--------+------------+--------------+-------------+
6 rows in set (0.00 sec)

mysql>
```

Assignment 6: Draft a brief report on the use of transaction logs for data recovery and create a hypothetical scenario where a transaction log is instrumental in data recovery after an unexpected shutdown.

Transaction logs are crucial components of database management systems that record all changes made to a database. These logs serve as a reliable source of information for recovering data in the event of system failures or unexpected shutdowns.

Importance of Transaction Logs:

1. Data Integrity: Transaction logs ensure data integrity by recording every transaction before it is committed to the database. This allows for rollbacks or recovery to a specific point in time.

2. Recovery Point: They provide a recovery point in case of system failures, allowing databases to be restored to a consistent state prior to the failure.

3. Performance Monitoring: Transaction logs also aid in performance monitoring and troubleshooting, as they track changes and can identify potential issues.

Hypothetical Scenario:

Imagine a scenario where a large e-commerce company experiences an unexpected server shutdown during a peak shopping period, resulting in potential data loss and customer disruption. However, due to the

implementation of transaction logs, the company's database administrator can initiate a successful data recovery process.

Scenario Details:

1. Unexpected Shutdown: The e-commerce platform experiences a sudden server shutdown due to a power outage.

2. Data Loss Concerns: Concerns arise about potential data loss, including ongoing transactions and customer orders that were being processed.

3. Transaction Logs Utilization: The database administrator leverages transaction logs to restore the database to its state just before the shutdown.

4. Recovery Process: By analysing the transaction logs, the administrator identifies the last committed transactions before the shutdown.

5. Database Restoration: Using this information, the administrator restores the database to the point just before the unexpected shutdown, ensuring minimal data loss and maintaining data consistency.

6. Customer Impact Mitigation: The quick recovery minimizes disruption for customers, allowing them to resume their transactions seamlessly.

Conclusion:

Transaction logs play a vital role in data recovery, especially in scenarios of unexpected shutdowns or system failures. By maintaining a record of all database transactions, transaction logs enable organizations to restore data integrity and minimize downtime, ultimately ensuring business continuity and customer satisfaction.