CREATE DATABASE ECOMMERCE;

USE ECOMMERCE;

TABLE CREATION:

mysql> create table customers (customer\_id int primary key Auto\_increment,name varchar(20) not null,email varchar(30) unique not null,password varchar(30));

Query OK, 0 rows affected (0.09 sec)

mysql> create table products(product\_id int primary key Auto\_increment,name varchar(20) not null,price decimal(10,2) not null default 0.00,description varchar(50) ,stock\_quantity int default 0);

Query OK, 0 rows affected (0.03 sec)

mysql> create table cart(cart\_id int primary key Auto\_increment,customer\_id int,product\_id int,quantity int default 0,foreign key(customer\_id) references customers (customer\_id),foreign key(product\_id) references products(product\_id));

Query OK, 0 rows affected (0.04 sec)

mysql> CREATE TABLE ORDERS(order\_id int primary key,customer\_id int,order\_date date,total\_price decimal(10,2) default 0.00,shipping\_address varchar(50),foreign key (customer\_id) references customers(customer\_id) on delete cascade on update cascade);

Query OK, 0 rows affected (0.05 sec)

mysql> create table order\_items(order\_item\_id int primary key,order\_id int,product\_id int,quantity int default 0,foreign key(order\_id) references orders (order\_id) on delete cascade on update cascade,foreign key(product\_id) references products(product\_id) on delete cascade on update cascade);

Query OK, 0 rows affected (0.05 sec)

INSERTING VALUES::

mysql> INSERT INTO PRODUCTS (name, description, price, stock\_quantity) VALUES

-> ('laptop', 'high-performance laptop', 800.00, 10),

-> ('smartphone', 'latest smartphone', 600.00, 15),

-> ('tablet', 'portable tablet', 300.00, 20),

-> ('headphones', 'noise-canceling headphones', 150.00, 30),

-> ('TV', '4K Smart TV', 900.00, 5),

-> ('coffee Maker', 'automatic coffee maker', 50.00, 25),

-> ('refrigerator', 'energy-efficient refrigerator', 700.00, 10),

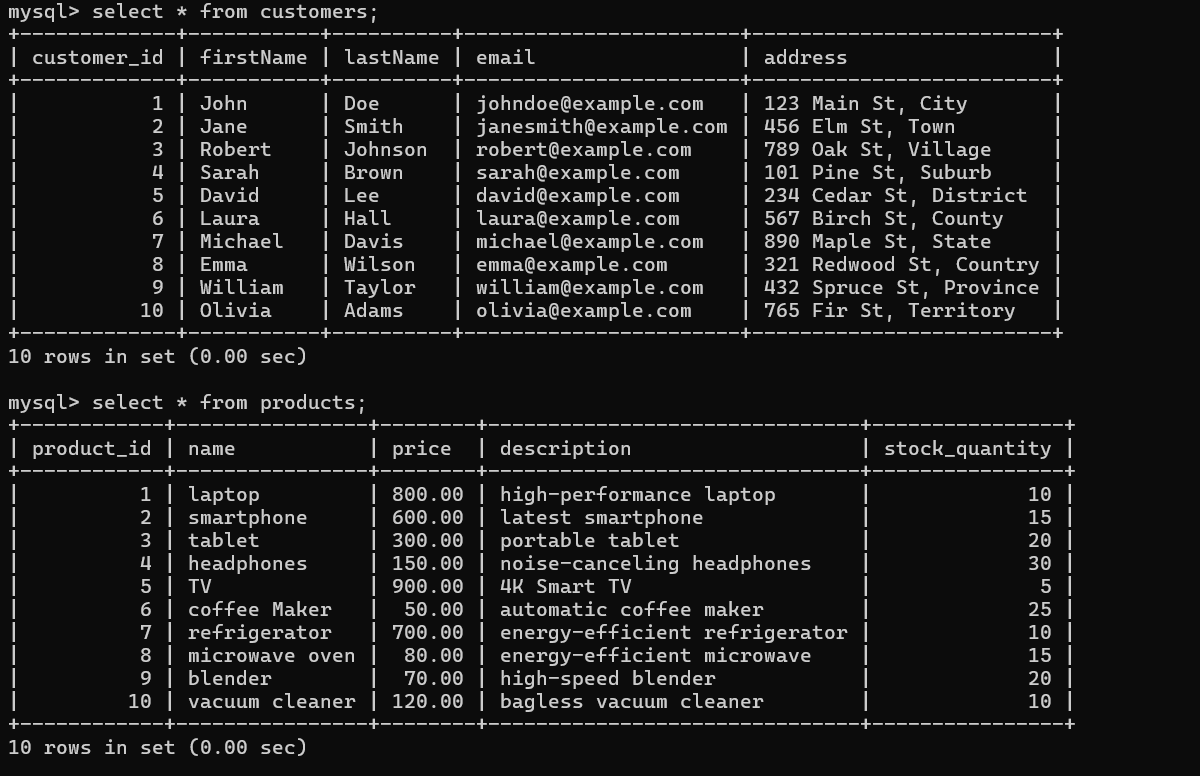
-> ('microwave oven', 'energy-efficient microwave', 80.00, 15),

-> ('blender', 'high-speed blender', 70.00, 20),

-> ('vacuum cleaner', 'bagless vacuum cleaner', 120.00, 10);

Query OK, 10 rows affected (0.01 sec)

Records: 10 Duplicates: 0 Warnings: 0



mysql> INSERT INTO customers (firstName, lastName, email, address) VALUES

-> ('John', 'Doe', 'johndoe@example.com', '123 Main St, City'),

-> ('Jane', 'Smith', 'janesmith@example.com', '456 Elm St, Town'),

-> ('Robert', 'Johnson', 'robert@example.com', '789 Oak St, Village'),

-> ('Sarah', 'Brown', 'sarah@example.com', '101 Pine St, Suburb'),

-> ('David', 'Lee', 'david@example.com', '234 Cedar St, District'),

-> ('Laura', 'Hall', 'laura@example.com', '567 Birch St, County'),

-> ('Michael', 'Davis', 'michael@example.com', '890 Maple St, State'),

-> ('Emma', 'Wilson', 'emma@example.com', '321 Redwood St, Country'),

-> ('William', 'Taylor', 'william@example.com', '432 Spruce St, Province'),

-> ('Olivia', 'Adams', 'olivia@example.com', '765 Fir St, Territory');

Query OK, 10 rows affected (0.01 sec)

Records: 10 Duplicates: 0 Warnings: 0

mysql> insert into orders(order\_id,customer\_id,order\_date,total\_price) values

-> (1,1,'2023-01-05',1200.00),

-> (2,2,'2023-02-10',900.00),

-> (3,3,'2023-03-15',300.00),

-> (4,4,'2023-04-20',150.00),

-> (5,5,'2023-05-25',1800.00),

-> (6,6,'2023-06-30',400.00),

-> (7,7,'2023-07-05',700.00),

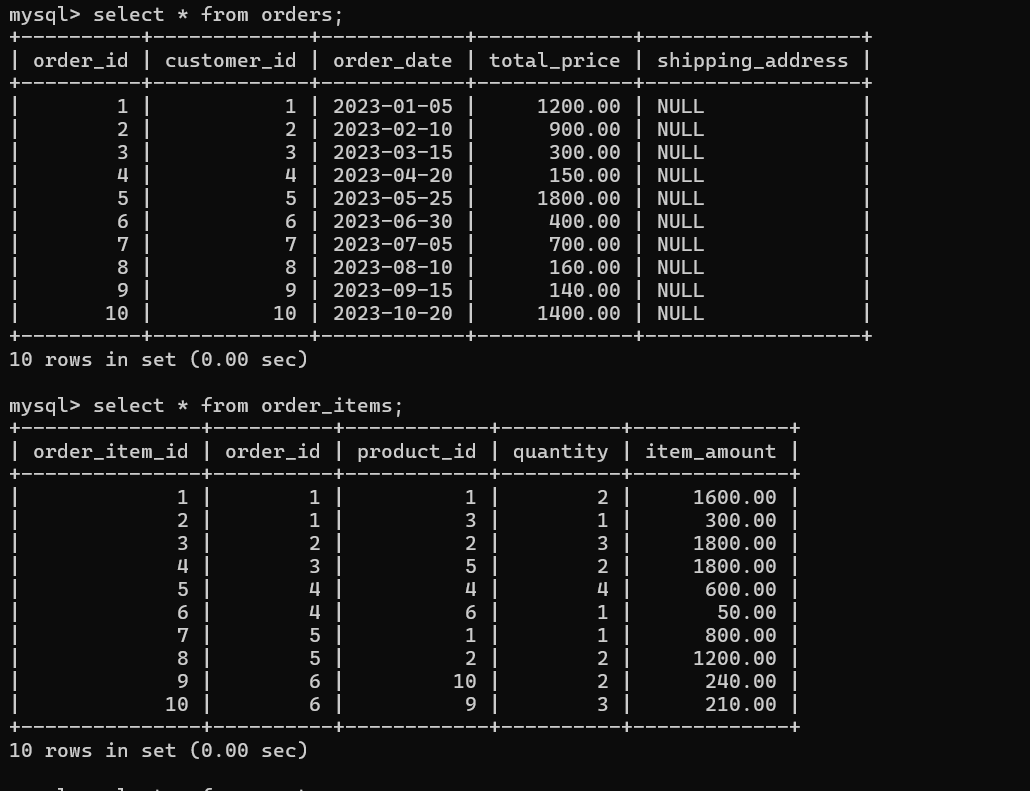
-> (8,8,'2023-08-10',160.00),

-> (9,9,'2023-09-15',140.00),

-> (10,10,'2023-10-20',1400.00);

Query OK, 10 rows affected (0.01 sec)

Records: 10 Duplicates: 0 Warnings: 0

mysql> insert into order\_items (order\_item\_id,order\_id,product\_id,quantity,item\_amount) values

-> (1, 1, 1, 2, 1600.00),

-> (2, 1, 3, 1, 300.00),

-> (3, 2, 2, 3, 1800.00),

-> (4, 3, 5, 2, 1800.00),

-> (5, 4, 4, 4, 600.00),

-> (6, 4, 6, 1, 50.00),

-> (7, 5, 1, 1, 800.00),

-> (8, 5, 2, 2, 1200.00),

-> (9, 6, 10, 2, 240.00),

-> (10, 6, 9, 3, 210.00);

Query OK, 10 rows affected (0.01 sec)

Records: 10 Duplicates: 0 Warnings: 0

mysql> insert into cart(customer\_id,product\_id,quantity) values

-> (1,1,2),

-> (1,3,1),

-> (2,2,3),

-> (3,4,4),

-> (3,5,2),

-> (4,6,1),

-> (5,1,1),

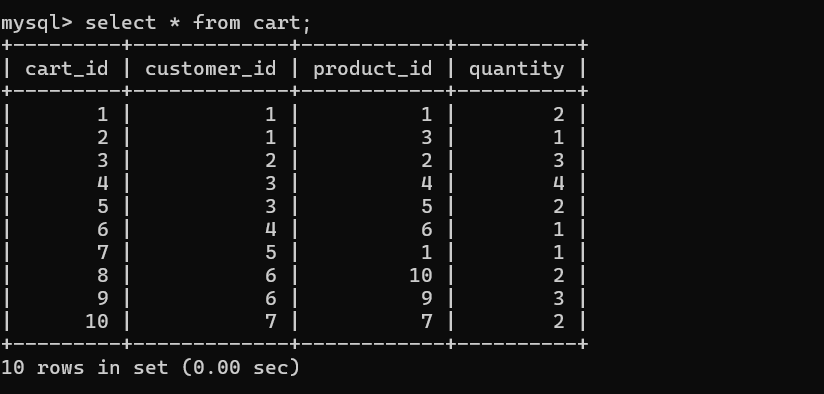
-> (6,10,2),

-> (6,9,3),

-> (7,7,2);

Query OK, 10 rows affected (0.01 sec)

Records: 10 Duplicates: 0 Warnings: 0

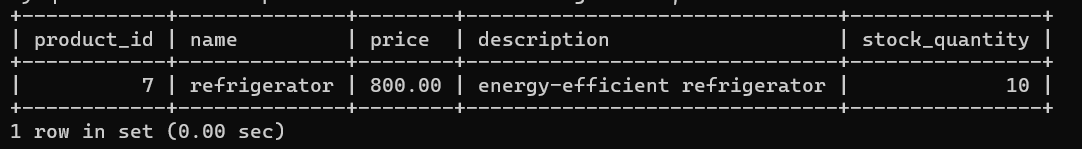


questions and answers:

1. Update refrigerator product price to 800.

mysql> update products set price=800.00 where name='refrigerator';

Query OK, 1 row affected (0.01 sec)



Rows matched: 1 Changed: 1 Warnings: 0

2. Remove all cart items for a specific customer.

mysql> delete from cart where customer\_id=5;

Query OK, 1 row affected (0.01 sec)

3. Retrieve Products Priced Below $100.

mysql> select \* from products where price<100;

+------------+----------------+-------+----------------------------+----------------+

| product\_id | name | price | description | stock\_quantity |

+------------+----------------+-------+----------------------------+----------------+

| 6 | coffee Maker | 50.00 | automatic coffee maker | 25 |

| 8 | microwave oven | 80.00 | energy-efficient microwave | 15 |

| 9 | blender | 70.00 | high-speed blender | 20 |

+------------+----------------+-------+----------------------------+----------------+

3 rows in set (0.00 sec)

4. Find Products with Stock Quantity Greater Than 5.

mysql> select name from products where stock\_quantity >5;

+----------------+

| name |

+----------------+

| laptop |

| smartphone |

| tablet |

| headphones |

| coffee Maker |

| refrigerator |

| microwave oven |

| blender |

| vacuum cleaner |

+----------------+

9 rows in set (0.00 sec)

5. Retrieve Orders with Total Amount Between $500 and $1000.

mysql> select \* from orders where total\_price between 500 and 1000;

+----------+-------------+------------+-------------+------------------+

| order\_id | customer\_id | order\_date | total\_price | shipping\_address |

+----------+-------------+------------+-------------+------------------+

| 2 | 2 | 2023-02-10 | 900.00 | NULL |

| 7 | 7 | 2023-07-05 | 700.00 | NULL |

+----------+-------------+------------+-------------+------------------+

2 rows in set (0.00 sec)

6. Find Products which name end with letter ‘r’.

mysql> select name from products where name like "%r";

+----------------+

| name |

+----------------+

| coffee Maker |

| refrigerator |

| blender |

| vacuum cleaner |

+----------------+

4 rows in set (0.00 sec)

7. Retrieve Cart Items for Customer 5.

mysql> select \* from cart where customer\_id=5;

Empty set (0.00 sec)

8. Find Customers Who Placed Orders in 2023.

mysql> select customer\_id from orders where year(order\_date)=2023;

+-------------+

| customer\_id |

+-------------+

| 1 |

| 2 |

| 3 |

| 4 |

| 5 |

| 6 |

| 7 |

| 8 |

| 9 |

| 10 |

+-------------+

10 rows in set (0.00 sec)

9. Determine the Minimum Stock Quantity for Each Product Category.

mysql> select name,MIN(stock\_quantity) as min\_stock from products group by name;

+----------------+-----------+

| name | min\_stock |

+----------------+-----------+

| laptop | 10 |

| smartphone | 15 |

| tablet | 20 |

| headphones | 30 |

| TV | 5 |

| coffee Maker | 25 |

| refrigerator | 10 |

| microwave oven | 15 |

| blender | 20 |

| vacuum cleaner | 10 |

+----------------+-----------+

10 rows in set (0.00 sec)

10. Calculate the Total Amount Spent by Each Customer.

mysql> select c.customer\_id,c.firstName,sum(o.total\_price) as total\_amount from customers c join orders o on c.customer\_id=o.customer\_id group by c.customer\_id;

+-------------+-----------+--------------+

| customer\_id | firstName | total\_amount |

+-------------+-----------+--------------+

| 1 | John | 1200.00 |

| 2 | Jane | 900.00 |

| 3 | Robert | 300.00 |

| 4 | Sarah | 150.00 |

| 5 | David | 1800.00 |

| 6 | Laura | 400.00 |

| 7 | Michael | 700.00 |

| 8 | Emma | 160.00 |

| 9 | William | 140.00 |

| 10 | Olivia | 1400.00 |

+-------------+-----------+--------------+

10 rows in set (0.00 sec)10 rows in set (0.00 sec)

11. Find the Average Order Amount for Each Customer.

mysql> select c.customer\_id,c.firstName,avg(o.total\_price) as avg\_amount from customers c join orders o on c.customer\_id=o.customer\_id group by c.customer\_id;

+-------------+-----------+-------------+

| customer\_id | firstName | avg\_amount |

+-------------+-----------+-------------+

| 1 | John | 1200.000000 |

| 2 | Jane | 900.000000 |

| 3 | Robert | 300.000000 |

| 4 | Sarah | 150.000000 |

| 5 | David | 1800.000000 |

| 6 | Laura | 400.000000 |

| 7 | Michael | 700.000000 |

| 8 | Emma | 160.000000 |

| 9 | William | 140.000000 |

| 10 | Olivia | 1400.000000 |

+-------------+-----------+-------------+

10 rows in set (0.00 sec)

12. Count the Number of Orders Placed by Each Customer.

mysql> select c.customer\_id,c.firstName,count(o.order\_id) as order\_count from customers c join orders o on c.customer\_id=o.customer\_id group by c.customer\_id;

+-------------+-----------+-------------+

| customer\_id | firstName | order\_count |

+-------------+-----------+-------------+

| 1 | John | 1 |

| 2 | Jane | 1 |

| 3 | Robert | 1 |

| 4 | Sarah | 1 |

| 5 | David | 1 |

| 6 | Laura | 1 |

| 7 | Michael | 1 |

| 8 | Emma | 1 |

| 9 | William | 1 |

| 10 | Olivia | 1 |

+-------------+-----------+-------------+

10 rows in set (0.00 sec) 10 rows in set (0.00 sec)

13. Find the Maximum Order Amount for Each Customer.

mysql> select c.customer\_id,c.firstName,max(o.total\_price) as max\_amount from customers c join orders o on c.customer\_id=o.customer\_id group by c.customer\_id;

+-------------+-----------+------------+

| customer\_id | firstName | max\_amount |

+-------------+-----------+------------+

| 1 | John | 1200.00 |

| 2 | Jane | 900.00 |

| 3 | Robert | 300.00 |

| 4 | Sarah | 150.00 |

| 5 | David | 1800.00 |

| 6 | Laura | 400.00 |

| 7 | Michael | 700.00 |

| 8 | Emma | 160.00 |

| 9 | William | 140.00 |

| 10 | Olivia | 1400.00 |

+-------------+-----------+------------+

10 rows in set (0.00 sec)

14. Get Customers Who Placed Orders Totaling Over $1000.

mysql> select c.customer\_id,c.firstName,c.lastName from customers c join orders o on c.customer\_id=o.customer\_id where o.total\_price>1000;

+-------------+-----------+----------+

| customer\_id | firstName | lastName |

+-------------+-----------+----------+

| 1 | John | Doe |

| 5 | David | Lee |

| 10 | Olivia | Adams |

+-------------+-----------+----------+

3 rows in set (0.00 sec)

15. Subquery to Find Products Not in the Cart.

mysql> select product\_id,name from products where product\_id not in (select distinct product\_id from cart);

+------------+----------------+

| product\_id | name |

+------------+----------------+

| 8 | microwave oven |

+------------+----------------+

1 row in set (0.00 sec)

16. Subquery to Find Customers Who Haven't Placed Orders.

mysql> select customer\_id,firstName,lastName from customers where customer\_id not in (select distinct customer\_id from orders);

Empty set (0.00 sec)

17. Subquery to Calculate the Percentage of Total Revenue for a Product.

mysql> select p.product\_id,p.name,(select sum(oi.item\_amount) from order\_items oi where oi.product\_id=p.product\_id)/(select sum(total\_price) from orders) \*

100 as total\_revenue from products p;

+------------+----------------+---------------+

| product\_id | name | total\_revenue |

+------------+----------------+---------------+

| 1 | laptop | 33.566434 |

| 2 | smartphone | 41.958042 |

| 3 | tablet | 4.195804 |

| 4 | headphones | 8.391608 |

| 5 | TV | 25.174825 |

| 6 | coffee Maker | 0.699301 |

| 7 | refrigerator | NULL |

| 8 | microwave oven | NULL |

| 9 | blender | 2.937063 |

| 10 | vacuum cleaner | 3.356643 |

+------------+----------------+---------------+

10 rows in set (0.00 sec)

18. Subquery to Find Products with Low Stock.

mysql> select product\_id,name from products where stock\_quantity<(select avg(stock\_quantity) from products) / 2;

+------------+------+

| product\_id | name |

+------------+------+

| 5 | TV |

+------------+------+

1 row in set (0.00 sec)

19. Subquery to Find Customers Who Placed High-Value Orders.

mysql> select customer\_id,concat(firstName,lastName) as full\_name from customers where customer\_id in(select distinct customer\_id from orders where total\_pr

ice>1000);

+-------------+-------------+

| customer\_id | full\_name |

+-------------+-------------+

| 1 | JohnDoe |

| 5 | DavidLee |

| 10 | OliviaAdams |

+-------------+-------------+

3 rows in set (0.01 sec)