**Rakshitha Devi J**
**Superset Id: 5369940**
**Saveetha Engineering College**

**CODING CHALLENGE SQL**
**ECOMMERCE – SQL**

## 1.CREATE DATABASE

```
create database ecomdb;
use ecomdb;
```

## 2.TABLES:

```
create table customers (
    customer_id int primary key,
    name varchar(255),
    email varchar(255),
    password varchar(255)
);

create table products (
    product_id int primary key,
    name varchar(255),
    price decimal(10,2),
    description text,
    stockquantity int
);

create table cart (
    cart_id int primary key,
    customer_id int,
    product_id int,
    quantity int,
    foreign key (customer_id) references customers(customer_id),
    foreign key (product_id) references products(product_id)
);

create table orders (
    order_id int primary key,
    customer_id int,
```

```sql
    order_date date,
    total_price decimal(10,2),
    shipping_address text,
    foreign key (customer_id) references customers(customer_id)
);

create table order_items (
    order_item_id int primary key,
    order_id int,
    product_id int,
    quantity int,
    foreign key (order_id) references orders(order_id),
    foreign key (product_id) references products(product_id)
);
```

## 3.DATA'S:

```sql
insert into customers (customer_id, name, email, password) values
(1, 'john doe', 'johndoe@example.com', 'password1'),
(2, 'jane smith', 'janesmith@example.com', 'password2'),
(3, 'robert johnson', 'robert@example.com', 'password3'),
(4, 'sarah brown', 'sarah@example.com', 'password4'),
(5, 'david lee', 'david@example.com', 'password5'),
(6, 'laura hall', 'laura@example.com', 'password6'),
(7, 'michael davis', 'michael@example.com', 'password7'),
(8, 'emma wilson', 'emma@example.com', 'password8'),
(9, 'william taylor', 'william@example.com', 'password9'),
(10, 'olivia adams', 'olivia@example.com', 'password10');

insert into products (product_id, name, price, description, stockquantity) values
(1, 'laptop', 800.00, 'high-performance laptop', 10),
(2, 'smartphone', 600.00, 'latest smartphone', 15),
(3, 'tablet', 300.00, 'portable tablet', 20),
(4, 'headphones', 150.00, 'noise-canceling', 30),
(5, 'tv', 900.00, '4k smart tv', 5),
(6, 'coffee maker', 50.00, 'automatic coffee maker', 25),
(7, 'refrigerator', 700.00, 'energy-efficient', 10),
(8, 'microwave oven', 80.00, 'countertop microwave', 15),
```

```sql
(9, 'blender', 70.00, 'high-speed blender', 20),
(10, 'vacuum cleaner', 120.00, 'bagless vacuum cleaner', 10);

insert into cart (cart_id, customer_id, product_id, quantity) values
(1, 1, 1, 2),
(2, 1, 3, 1),
(3, 2, 2, 3),
(4, 3, 4, 4),
(5, 3, 5, 2),
(6, 4, 6, 1),
(7, 5, 1, 1),
(8, 6, 10, 2),
(9, 6, 9, 3),
(10, 7, 7, 2);

insert into orders (order_id, customer_id, order_date, total_price,
shipping_address) values
(1, 1, '2023-01-05', 1200.00, '123 main st, city'),
(2, 2, '2023-02-10', 900.00, '456 elm st, town'),
(3, 3, '2023-03-15', 300.00, '789 oak st, village'),
(4, 4, '2023-04-20', 150.00, '101 pine st, suburb'),
(5, 5, '2023-05-25', 1800.00, '234 cedar st, district'),
(6, 6, '2023-06-30', 400.00, '567 birch st, county'),
(7, 7, '2023-07-05', 700.00, '890 maple st, state'),
(8, 8, '2023-08-10', 160.00, '321 redwood st, country'),
(9, 9, '2023-09-15', 140.00, '432 spruce st, province'),
(10, 10, '2023-10-20', 1400.00, '765 fir st, territory');

insert into order_items (order_item_id, order_id, product_id, quantity) values
(1, 1, 1, 2),
(2, 1, 3, 1),
(3, 2, 2, 3),
(4, 3, 5, 2),
(5, 4, 4, 4),
(6, 4, 6, 1),
(7, 5, 1, 1),
(8, 5, 2, 2),
```

(9, 6, 10, 2),
(10, 6, 9, 3);

## 1. update refrigerator product price to 800.

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

| product_id | name | price | description | stockquantit |
|---|---|---|---|---|
| 1 | Laptop | 800.00 | High-performance laptop | 10 |
| 2 | Smartphone | 600.00 | Latest smartphone | 15 |
| 3 | Tablet | 300.00 | Portable tablet | 20 |
| 4 | Headphones | 150.00 | Noise-canceling | 30 |
| 5 | TV | 900.00 | 4K Smart TV | 5 |
| 6 | Coffee Maker | 50.00 | Automatic coffee maker | 25 |
| 7 | Refrigerator | 800.00 | Energy-efficient | 10 |
| 8 | Microwave Oven | 80.00 | Countertop microwave | 15 |
| 9 | Blender | 70.00 | High-speed blender | 20 |
| 10 | Vacuum Cleaner | 120.00 | Bagless vacuum cleaner | 10 |
| NULL | NULL | NULL | NULL | NULL |

## 2. remove all cart items for a specific customer.
delete from cart where customer_id = 2;

| cart_id | customer_id | product_id | quantity |
|---|---|---|---|
| 1 | 1 | 1 | 2 |
| 2 | 1 | 3 | 1 |
| 4 | 3 | 4 | 4 |
| 5 | 3 | 5 | 2 |
| 6 | 4 | 6 | 1 |
| 7 | 5 | 1 | 1 |
| 8 | 6 | 10 | 2 |
| 9 | 6 | 9 | 3 |
| 10 | 7 | 7 | 2 |
| NULL | NULL | NULL | NULL |

### 3. retrieve products priced below $100.

select * from products where price < 100;

| | product_id | name | price | description | stockquantit |
|---|---|---|---|---|---|
| ▶ | 6 | Coffee Maker | 50.00 | Automatic coffee maker | 25 |
| | 8 | Microwave Oven | 80.00 | Countertop microwave | 15 |
| | 9 | Blender | 70.00 | High-speed blender | 20 |
| * | NULL | NULL | NULL | NULL | NULL |

### 4. find products with stock quantity greater than 5.

select * from products where stockquantity > 5;

| | product_id | name | price | description | stockquan |
|---|---|---|---|---|---|
| ▶ | 1 | Laptop | 800.00 | High-performance laptop | 10 |
| | 2 | Smartphone | 600.00 | Latest smartphone | 15 |
| | 3 | Tablet | 300.00 | Portable tablet | 20 |
| | 4 | Headphones | 150.00 | Noise-canceling | 30 |
| | 6 | Coffee Maker | 50.00 | Automatic coffee maker | 25 |
| | 7 | Refrigerator | 800.00 | Energy-efficient | 10 |
| | 8 | Microwave Oven | 80.00 | Countertop microwave | 15 |
| | 9 | Blender | 70.00 | High-speed blender | 20 |
| | 10 | Vacuum Cleaner | 120.00 | Bagless vacuum cleaner | 10 |
| * | NULL | NULL | NULL | NULL | NULL |

### 5. retrieve orders with total amount between $500 and $1000.

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 | 456 Elm St, Town |
| | 7 | 7 | 2023-07-05 | 700.00 | 890 Maple St, State |
| * | NULL | NULL | NULL | NULL | NULL |

## 6. find products which name end with letter 'r'.

select * from products where name like '%r';

| product_id | name | price | description | stockquantity |
|---|---|---|---|---|
| 6 | Coffee Maker | 50.00 | Automatic coffee maker | 25 |
| 7 | Refrigerator | 800.00 | Energy-efficient | 10 |
| 9 | Blender | 70.00 | High-speed blender | 20 |
| 10 | Vacuum Cleaner | 120.00 | Bagless vacuum cleaner | 10 |
| NULL | NULL | NULL | NULL | NULL |

## 7. retrieve cart items for customer 5.

select * from cart where customer_id = 5;

| cart_id | customer_id | product_id | quantity |
|---|---|---|---|
| 7 | 5 | 1 | 1 |
| NULL | NULL | NULL | NULL |

## 8. find customers who placed orders in 2023.

select distinct c.* from customers c
join orders o on c.customer_id = o.customer_id
where year(o.order_date) = 2023;

| customer_id | name | email | password |
|---|---|---|---|
| 1 | John Doe | johndoe@example.com | password1 |
| 2 | Jane Smith | janesmith@example.com | password2 |
| 3 | Robert Johnson | robert@example.com | password3 |
| 4 | Sarah Brown | sarah@example.com | password4 |
| 5 | David Lee | david@example.com | password5 |
| 6 | Laura Hall | laura@example.com | password6 |
| 7 | Michael Davis | michael@example.com | password7 |
| 8 | Emma Wilson | emma@example.com | password8 |
| 9 | William Taylor | william@example.com | password9 |
| 10 | Olivia Adams | olivia@example.com | password10 |

## 9. determine the minimum stock quantity for each product category.

select min(stockquantity) as min_stock from products;

| min_stock |
|-----------|
| ▶ 5 |

## 10. calculate the total amount spent by each customer.

select c.customer_id, c.name, sum(o.total_price) as total_spent
from customers c
join orders o on c.customer_id = o.customer_id
group by c.customer_id, c.name;

| customer_id | name | total_spent |
|-------------|------|-------------|
| ▶ 1 | John Doe | 1200.00 |
| 2 | Jane Smith | 900.00 |
| 3 | Robert Johnson | 300.00 |
| 4 | Sarah Brown | 150.00 |
| 5 | David Lee | 1800.00 |
| 6 | Laura Hall | 400.00 |
| 7 | Michael Davis | 700.00 |
| 8 | Emma Wilson | 160.00 |
| 9 | William Taylor | 140.00 |
| 10 | Olivia Adams | 1400.00 |

## 11. find the average order amount for each customer.

select c.customer_id, c.name, avg(o.total_price) as avg_order_amount
from customers c
join orders o on c.customer_id = o.customer_id
group by c.customer_id, c.name;

| customer_id | name | avg_order_amount |
|---|---|---|
| 1 | John Doe | 1200.000000 |
| 2 | Jane Smith | 900.000000 |
| 3 | Robert Johnson | 300.000000 |
| 4 | Sarah Brown | 150.000000 |
| 5 | David Lee | 1800.000000 |
| 6 | Laura Hall | 400.000000 |
| 7 | Michael Davis | 700.000000 |
| 8 | Emma Wilson | 160.000000 |
| 9 | William Taylor | 140.000000 |
| 10 | Olivia Adams | 1400.000000 |

## 12. count the number of orders placed by each customer.

select customer_id, count(*) as order_count from orders group by customer_id;

| customer_id | order_count |
|---|---|
| 1 | 1 |
| 2 | 1 |
| 3 | 1 |
| 4 | 1 |
| 5 | 1 |
| 6 | 1 |
| 7 | 1 |
| 8 | 1 |
| 9 | 1 |
| 10 | 1 |

## 13. find the maximum order amount for each customer.

select customer_id, max(total_price) as max_order_amount from orders group by customer_id;

| customer_id | max_order_amount |
|---|---|
| 1 | 1200.00 |
| 2 | 900.00 |
| 3 | 300.00 |
| 4 | 150.00 |
| 5 | 1800.00 |
| 6 | 400.00 |
| 7 | 700.00 |
| 8 | 160.00 |
| 9 | 140.00 |
| 10 | 1400.00 |

## 14. get customers who placed orders totaling over $1000.

select c.customer_id, c.name, sum(o.total_price) as total_spent
from customers c
join orders o on c.customer_id = o.customer_id
group by c.customer_id, c.name
having total_spent > 1000;

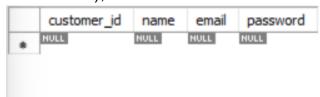| customer_id | name | total_spent |
|---|---|---|
| 1 | John Doe | 1200.00 |
| 5 | David Lee | 1800.00 |
| 10 | Olivia Adams | 1400.00 |

## 15. subquery to find products not in the cart.

select * from products where product_id not in (select distinct product_id from cart);

| product_id | name | price | description | stockquantity |
|---|---|---|---|---|
| 2 | Smartphone | 600.00 | Latest smartphone | 15 |
| 8 | Microwave Oven | 80.00 | Countertop microwave | 15 |
| NULL | NULL | NULL | NULL | NULL |

## 16. subquery to find customers who haven't placed orders.

select * from customers where customer_id not in (select distinct customer_id from orders);

| customer_id | name | email | password |
|---|---|---|---|
| NULL | NULL | NULL | NULL |

## 17. subquery to calculate the percentage of total revenue for a product.

select p.product_id, p.name,
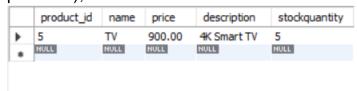(sum(oi.quantity * p.price) / (select sum(total_price) from orders)) * 100 as revenue_percentage
from order_items oi
join products p on oi.product_id = p.product_id
group by p.product_id, p.name;

| product_id | name | revenue_percentage |
|---|---|---|
| 1 | Laptop | 33.566434 |
| 3 | Tablet | 4.195804 |
| 2 | Smartphone | 41.958042 |
| 5 | TV | 25.174825 |
| 4 | Headphones | 8.391608 |
| 6 | Coffee Maker | 0.699301 |
| 10 | Vacuum Cleaner | 3.356643 |
| 9 | Blender | 2.937063 |

## 18. subquery to find products with low stock.
select * from products where stockquantity = (select min(stockquantity) from products);

| product_id | name | price | description | stockquantity |
|---|---|---|---|---|
| 5 | TV | 900.00 | 4K Smart TV | 5 |
| NULL | NULL | NULL | NULL | NULL |

## 19. subquery to find customers who placed high-value orders.
select * from customers where customer_id in (
select customer_id from orders where total_price > 1000

| customer_id | name | email | password |
|---|---|---|---|
| 1 | John Doe | johndoe@example.com | password1 |
| 5 | David Lee | david@example.com | password5 |
| 10 | Olivia Adams | olivia@example.com | password10 |

);