Coding Challenge
Ecommerce

Mubeenakatika07@gmail.com

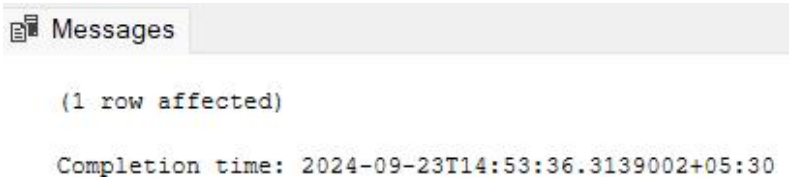
Name-Katika Mubeena

```sql
CREATE DATABASE CC1;
USE CC1
CREATE TABLE customers (
    customer_id INT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(100) NOT NULL UNIQUE,
    password VARCHAR(100) NOT NULL
);
CREATE TABLE products (
    product_id INT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    price DECIMAL(10, 2) NOT NULL,
    description TEXT,
    stockQuantity INT NOT NULL
);
CREATE TABLE cart (
    cart_id INT PRIMARY KEY,
    customer_id INT,
    product_id INT,
    quantity INT NOT NULL,
    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,
    order_date DATE NOT NULL,
    total_price DECIMAL(10, 2) NOT NULL,
    shipping_address VARCHAR(255),
    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 NOT NULL,
    FOREIGN KEY (order_id) REFERENCES orders(order_id),
    FOREIGN KEY (product_id) REFERENCES products(product_id)
);
INSERT INTO products (product_id, name, description, price, stockQuantity)
VALUES
(1, 'Laptop', 'High-performance laptop', 800.00, 10),
(2, 'Smartphone', 'Latest smartphone', 600.00, 15),
(3, 'Tablet', 'Portable tablet', 300.00, 20),
(4, 'Headphones', 'Noise-canceling', 150.00, 30),
(5, 'TV', '4K Smart TV', 900.00, 5),
```

```
(6, 'Coffee Maker', 'Automatic coffee maker', 50.00, 25),
(7, 'Refrigerator', 'Energy-efficient', 700.00, 10),
(8, 'Microwave Oven', 'Countertop microwave', 80.00, 15),
(9, 'Blender', 'High-speed blender', 70.00, 20),
(10, 'Vacuum Cleaner', 'Bagless vacuum cleaner', 120.00, 10);
INSERT INTO customers (customer_id, name, email, password) VALUES
(1, 'John Doe', 'johndoe@example.com', 'password123'),
(2, 'Jane Smith', 'janesmith@example.com', 'password123'),
(3, 'Robert Johnson', 'robert@example.com', 'password123'),
(4, 'Sarah Brown', 'sarah@example.com', 'password123'),
(5, 'David Lee', 'david@example.com', 'password123'),
(6, 'Laura Hall', 'laura@example.com', 'password123'),
(7, 'Michael Davis', 'michael@example.com', 'password123'),
(8, 'Emma Wilson', 'emma@example.com', 'password123'),
(9, 'William Taylor', 'william@example.com', 'password123'),
(10, 'Olivia Adams', 'olivia@example.com', 'password123');

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);
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);

--1.Update refrigerator product price to 800.
```

```sql
UPDATE products
SET price = 800
WHERE name='refrigerator';
```

OUTPUT




```sql
--2. Remove all cart items for a specific customer.
delete from cart where customer_id = 3;
```

OUTPUT


```sql
--3. Retrieve Products Priced Below $100.
select * From products
where price<100.00;
```

OUTPUT

Results Messages

	product_id	name	Description	price	stockQuantity
1	6	Coffee Maker	Automatic coffee maker	50	25
2	8	Microwave Oven	Countertop microwave	80	15
3	9	Blender	High-speed blender	70	20

```
--4. Find Products with Stock Quantity Greater Than 5

select product_id,name,stockQuantity From products
where stockQuantity>5;
```

OUTPUT

Results Messages

	product_id	name	stockQuantity
1	1	Laptop	10
2	2	Smartphone	15
3	3	Tablet	20
4	4	Headphones	30
5	6	Coffee Maker	25
6	7	Refrigerator	10
7	8	Microwave Oven	15
8	9	Blender	20
9	10	Vacuum Cleaner	10

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

select * from orders
where total_amount between 500 and 1000;
```

OUTPUT

Results Messages

	order_id	customer_id	order_date	total_amount
1	2	2	2023-02-10	900
2	7	7	2023-07-05	700

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

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

OUTPUT

Results | Messages

	product_id	name	Description	price	stockQuantity
1	6	Coffee Maker	Automatic coffee maker	50	25
2	7	Refrigerator	Energy-efficient	800	10
3	9	Blender	High-speed blender	70	20
4	10	Vacuum Cleaner	Bagless vacuum cleaner	120	10

--7. Retrieve Cart Items for Customer 5.

```
SELECT *
FROM cart
WHERE customer_id = 5;
```

```
select * from cart
where customer_id = 5;
```

OUTPUT

	cart_id	customer_id	product_id	quantity
1	7	5	1	1

Results | Messages

	cart_id	customer_id	product_id	quantity
1	7	5	1	1

-- 8. Find Customers Who Placed Orders in 2023.

```
select c.first_name, c. last_name, o.order_date
from customers c join orders o on c.customer_id = o.customer_id
where o.order_date like '2023%';
```

OUTPUT

Results Messages

	first_name	last_name	order_date
1	John	Doe	2023-01-05
2	Jane	Smith	2023-02-10
3	Robert	Johnson	2023-03-15
4	Sarah	Brown	2023-04-20
5	David	Lee	2023-05-25
6	Laura	Hall	2023-06-30
7	Michael	Davis	2023-07-05
8	Emma	Wilson	2023-08-10
9	William	Taylor	2023-09-15
10	Olivia	Adams	2023-10-20

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

```
select MIN(stockQuantity) AS min_stockQuantity
from products
```

OUTPUT

Results Messages

	min_stockQuantity
1	5

```
--10. Calculate the Total Amount Spent by Each Customer
select customer_id, total_amount from orders
```

Results Messages

	customer_id	total_amount
1	1	1200
2	2	900
3	3	300
4	4	150
5	5	1800
6	6	400
7	7	700
8	8	160
9	9	140
10	10	1400

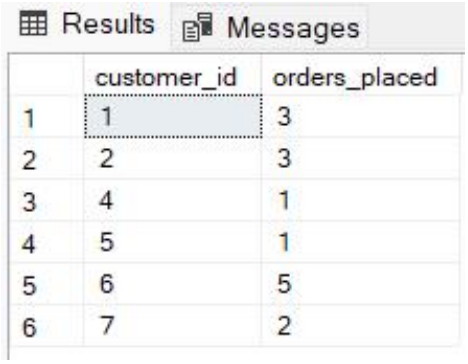
```sql
--11. Find the Average Order Amount for Each Customer.

SELECT c.customer_id, c.name, AVG(o.total_price) AS avgOrderAmount
FROM customers c
JOIN orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_id, c.name;
```

OUTPUT

	customer_id	name	avgOrderAmoun
1	1	Mubeena	1200.000000
2	2	sahil	900.000000
3	3	Robert Johnson	300.000000
4	4	Sarah Brown	150.000000
5	5	David Lee	1800.000000
6	6	Laura Hall	400.000000
7	7	Michael Davis	700.000000
8	8	Emma Wilson	160.000000
9	9	William Taylor	140.000000
10	10	Olivia Adams	1400.000000

```
--12. Count the Number of Orders Placed by Each Customer.
select customer_id,sum(quantity) AS orders_placed from cart
group by customer_id
```

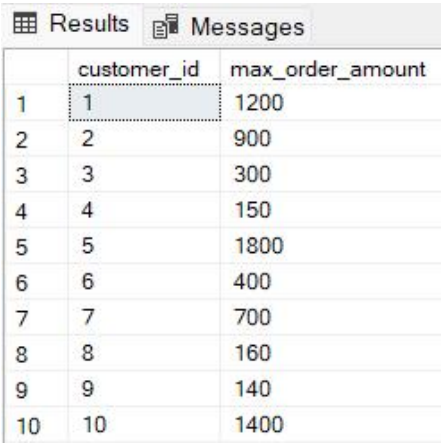
OUTPUT



Results | Messages

	customer_id	orders_placed
1	1	3
2	2	3
3	4	1
4	5	1
5	6	5
6	7	2

```
--13. Find the Maximum Order Amount for Each Customer.
SELECT customer_id, MAX(total_amount) AS max_order_amount
FROM orders
GROUP BY customer_id;
```



Results | Messages

	customer_id	max_order_amount
1	1	1200
2	2	900
3	3	300
4	4	150
5	5	1800
6	6	400
7	7	700
8	8	160
9	9	140
10	10	1400

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

```
select * from orders
where total_amount>1000;
```

OUTPUT

Results | Messages

	order_id	customer_id	order_date	total_amount
1	1	1	2023-01-05	1200
2	5	5	2023-05-25	1800
3	10	10	2023-10-20	1400

--15. Subquery to Find Products Not in the Cart.

```
select p.product_id,c.quantity from products p  left join cart c
on p.product_id = c.product_id
where quantity IS NULL
```

OUTPUT

Results | Messages

	product_id	quantity
1	4	NULL
2	5	NULL
3	8	NULL

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

```
select c.* from customers c  left join orders o
on c.customer_id = o.customer_id
where o.order_id IS NULL
```

OUTPUT

Results Messages

customer_id	first_name	last_name	email	address

(Empty Record as all customers placed order)

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

```
select *, (price*stockQuantity) AS total_revenue,
        ((price*stockQuantity)/100) AS revenue_percentage
from products
```

OUTPUT

Results Messages

	product_id	name	Description	price	stockQuantity	total_revenue	revenue_percentage
1	1	Laptop	High-performance laptop	800	10	8000	80.000000
2	2	Smartphone	Latest smartphone	600	15	9000	90.000000
3	3	Tablet	Portable tablet	300	20	6000	60.000000
4	4	Headphones	Noise-canceling	150	30	4500	45.000000
5	5	TV	4K Smart TV	900	5	4500	45.000000
6	6	Coffee Maker	Automatic coffee maker	50	25	1250	12.500000
7	7	Refrigerator	Energy-efficient	800	10	8000	80.000000
8	8	Microwave Oven	Countertop microwave	80	15	1200	12.000000
9	9	Blender	High-speed blender	70	20	1400	14.000000
10	10	Vacuum Cleaner	Bagless vacuum cleaner	120	10	1200	12.000000

```
--18. Subquery to Find Products with Low Stock.
```

```
SELECT *
FROM products
WHERE stockQuantity < (SELECT AVG(stockQuantity) FROM products);
```

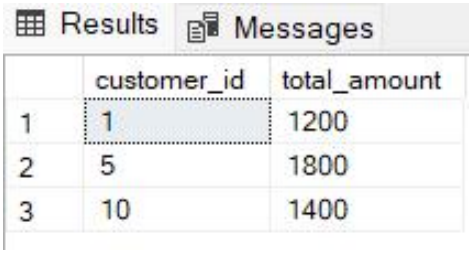
OUTPUT

Results Messages

	product_id	name	Description	price	stockQuantity
1	1	Laptop	High-performance laptop	800	10
2	5	TV	4K Smart TV	900	5
3	7	Refrigerator	Energy-efficient	800	10
4	10	Vacuum Cleaner	Bagless vacuum cleaner	120	10

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

SELECT DISTINCT c.customer_id, o.total_price
FROM customers c
JOIN orders o ON c.customer_id = o.customer_id
WHERE o.total_price > 1000;
```

OUTPUT



Results Messages

	customer_id	total_amount
1	1	1200
2	5	1800
3	10	1400