SQL CODING CHALLENGE – ECOMMERCE

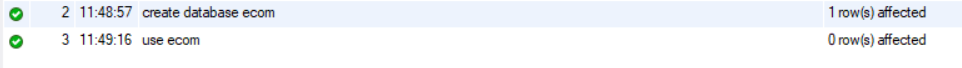
1. **Creating a Database to store the Ecom tables:**

**Query:**

*create database ecom;*

*use ecom;*

**Output:**



1. **Creation of SQL Tables:**

1. **customers table:** • customer\_id (Primary Key) • name • email • Address.

**Query:**

*CREATE TABLE customers (*

*customer\_id INT PRIMARY KEY,*

*first\_name VARCHAR(50),*

*last\_name VARCHAR(50),*

*email VARCHAR(100) UNIQUE,*

*address VARCHAR(255)*

*);*

2. **products table:** • product\_id (Primary Key) • name • price • description • stockQuantity.

**Query:**

*CREATE TABLE products (*

*product\_id INT PRIMARY KEY,*

*name VARCHAR(100),*

*description TEXT,*

*price DECIMAL(10,2),*

*stockQuantity INT*

*);*

3. **cart table:** • cart\_id (Primary Key) • customer\_id (Foreign Key) • product\_id (Foreign Key) • quantity.

**Query:**

*CREATE TABLE cart (*

*cart\_id INT PRIMARY KEY AUTO\_INCREMENT,*

*customer\_id INT,*

*product\_id INT,*

*quantity INT,*

*FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id) ON DELETE CASCADE,*

*FOREIGN KEY (product\_id) REFERENCES products(product\_id) ON DELETE CASCADE*

*);*

4. **orders table:** • order\_id (Primary Key) • customer\_id (Foreign Key) • order\_date • total\_price • shipping\_address.

**Query:**

*CREATE TABLE orders (*

*order\_id INT PRIMARY KEY AUTO\_INCREMENT,*

*customer\_id INT,*

*order\_date DATE,*

*total\_price DECIMAL(10,2),*

*shipping\_address VARCHAR(255),*

*FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id) ON DELETE CASCADE);*

**5. order\_items table** (to store order details): • order\_item\_id (Primary Key) • order\_id (Foreign Key) • product\_id (Foreign Key) • quantity • item\_amount.

**Query:**

*CREATE TABLE order\_items (*

*order\_item\_id INT PRIMARY KEY AUTO\_INCREMENT,*

*order\_id INT,*

*product\_id INT,*

*quantity INT,*

*item\_amount DECIMAL(10,2),*

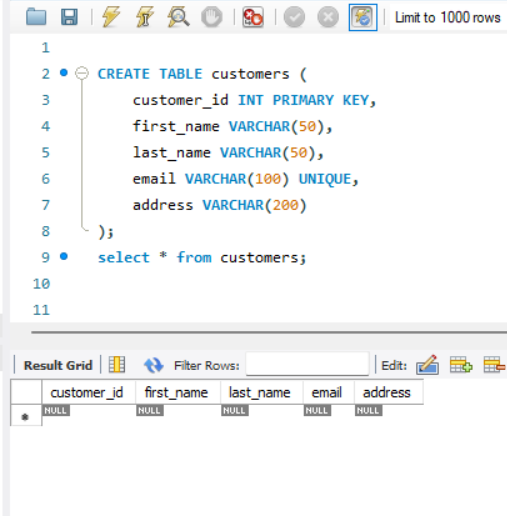
*FOREIGN KEY (order\_id) REFERENCES orders(order\_id) ON DELETE CASCADE,*

*FOREIGN KEY (product\_id) REFERENCES products(product\_id) ON DELETE CASCADE*

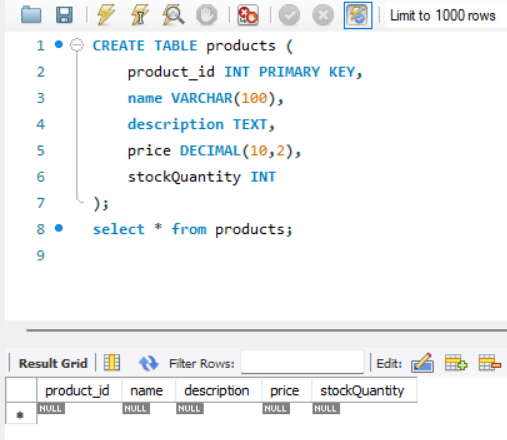
*);*

**Outputs:**

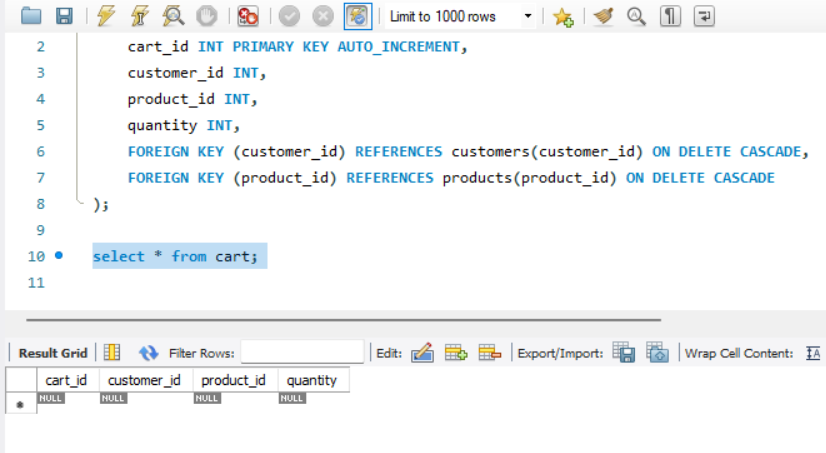
1. *Select \* from customers;*

****

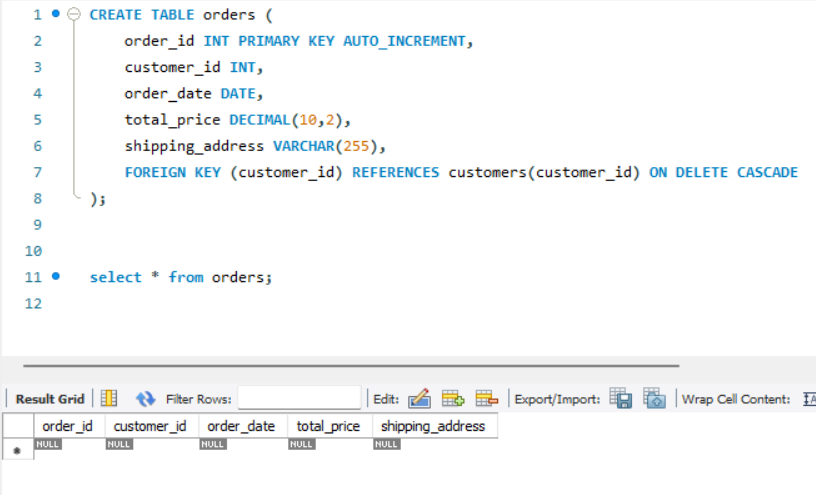
1. *Select \* from products;*

**

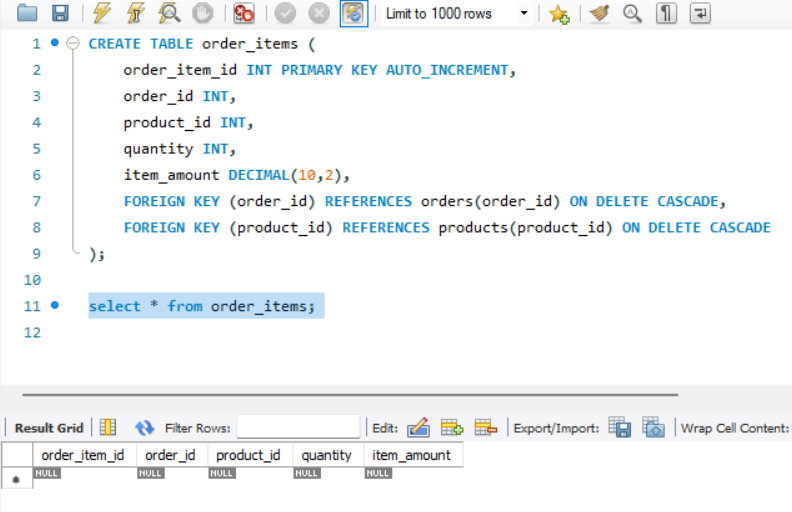
1. *Select \* from cart;*

**

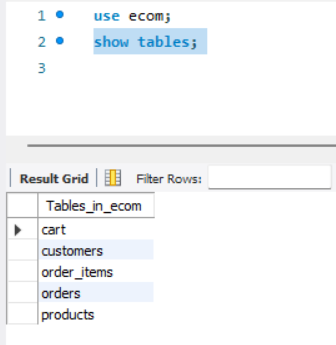
1. *Select \* from orders;*

**

1. *Select \* from order\_items;*

**

**Tables in Ecom Database:**



1. **Inserting Records into the Tables:**
2. **Insert data into Customers Table -**

**Query:**

*INSERT INTO customers (customer\_id, first\_name, last\_name, email, address) VALUES*

*(1, 'John', 'Doe', 'johndoe@example.com', '123 Main St, City'),*

*(2, 'Jane', 'Smith', 'janesmith@example.com', '456 Elm St, Town'),*

*(3, 'Robert', 'Johnson', 'robert@example.com', '789 Oak St, Village'),*

*(4, 'Sarah', 'Brown', 'sarah@example.com', '101 Pine St, Suburb'),*

*(5, 'David', 'Lee', 'david@example.com', '234 Cedar St, District'),*

*(6, 'Laura', 'Hall', 'laura@example.com', '567 Birch St, County'),*

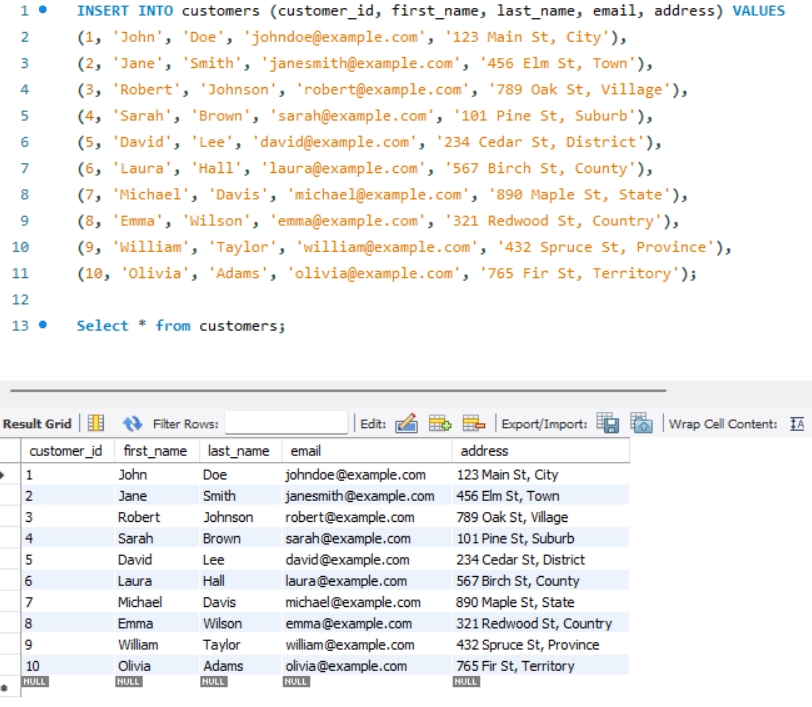
*(7, 'Michael', 'Davis', 'michael@example.com', '890 Maple St, State'),*

*(8, 'Emma', 'Wilson', 'emma@example.com', '321 Redwood St, Country'),*

*(9, 'William', 'Taylor', 'william@example.com', '432 Spruce St, Province'),*

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

**Output:**

****

1. **Insert data into products table –**

**Query:**

*INSERT INTO products (product\_id, name, description, price, stockQuantity) VALUES*

*(1, 'Laptop', 'High-performance laptop', 1200.00, 5),*

*(2, 'Smartphone', 'Latest smartphone', 800.00, 10),*

*(3, 'Tablet', 'Portable tablet', 600.00, 15),*

*(4, 'Headphones', 'Noise-canceling', 300.00, 20),*

*(5, 'TV', '4K Smart TV', 150.00, 30),*

*(6, 'Coffee Maker', 'Automatic coffee maker', 50.00, 25),*

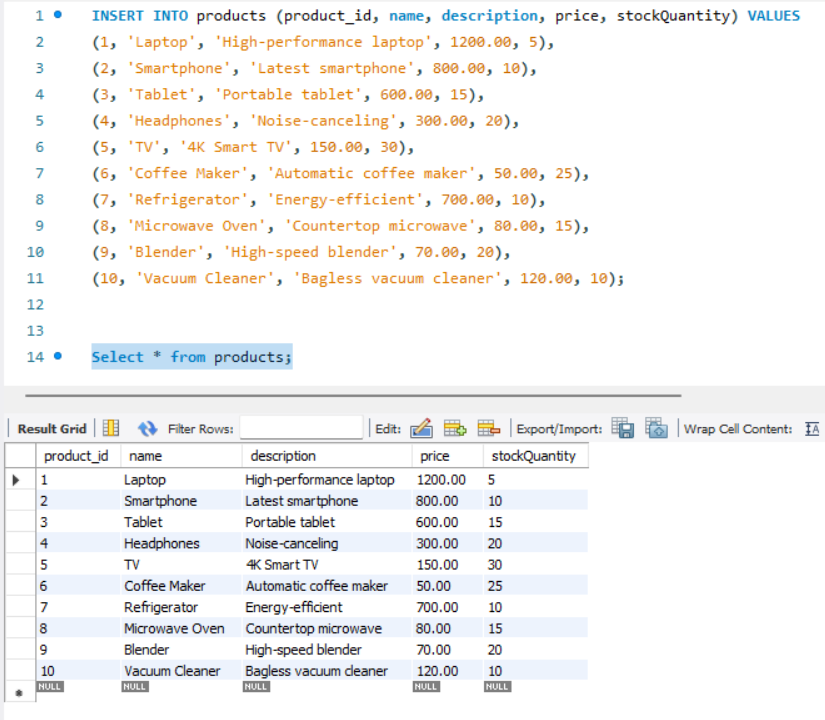
*(7, 'Refrigerator', 'Energy-efficient', 700.00, 10),*

*(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);*

**Output:**

****

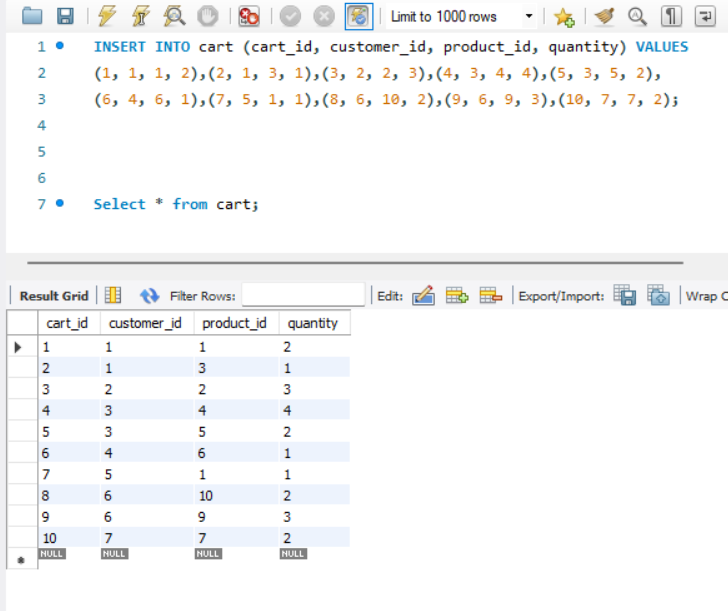
1. **Insert data into cart table –**

**Query:**

*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);*

**Output:**

****

1. **Insert data into orders table –**

**Query:**

*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'),*

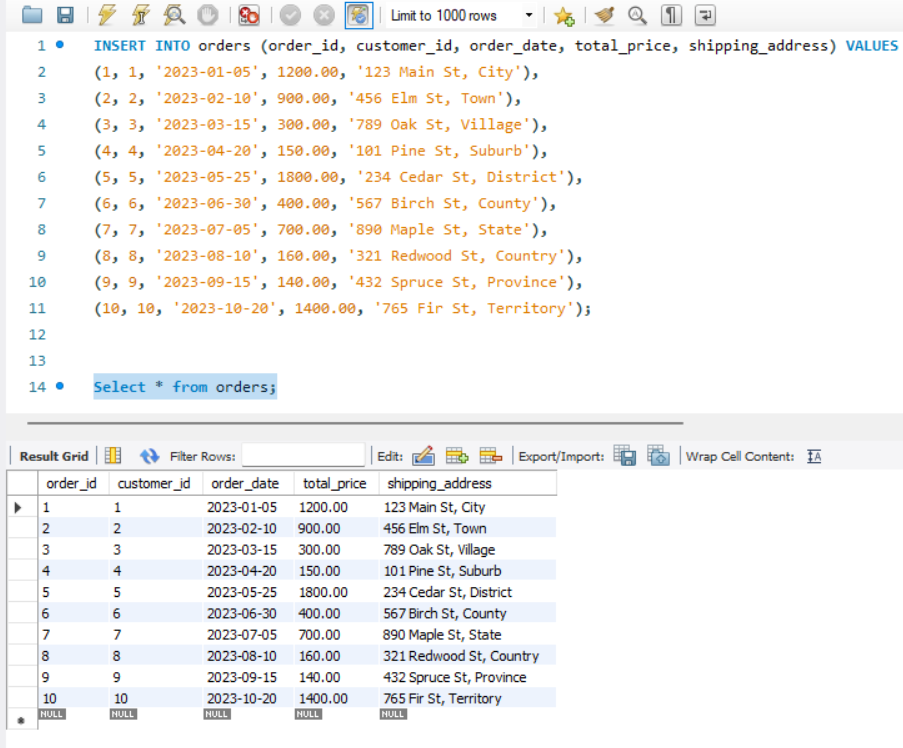
*(7, 7, '2023-07-05', 700.00, '890 Maple St, State'),*

*(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');*

**Output:**

****

1. **Insert data into order\_items table –**

**Query:**

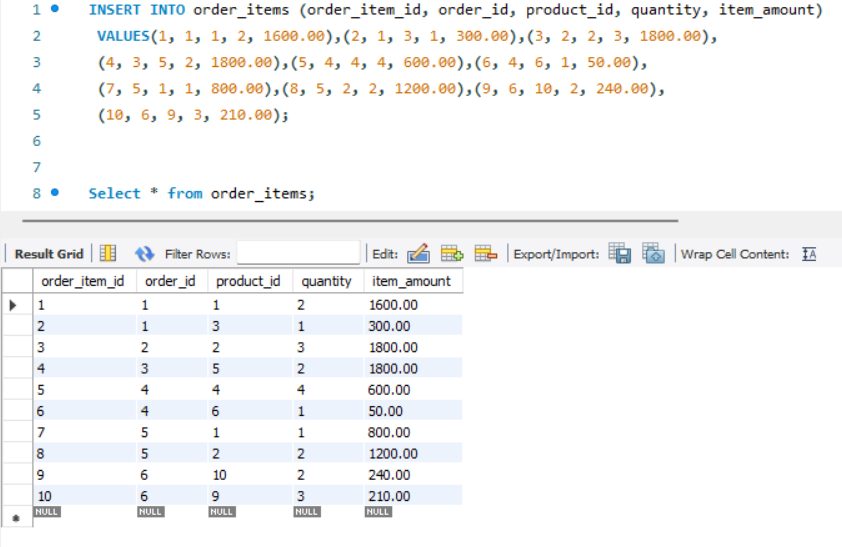
*INSERT INTO order\_items (order\_item\_id, order\_id, product\_id, quantity, item\_amount) VALUES*

*(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);*

**Output:**

****

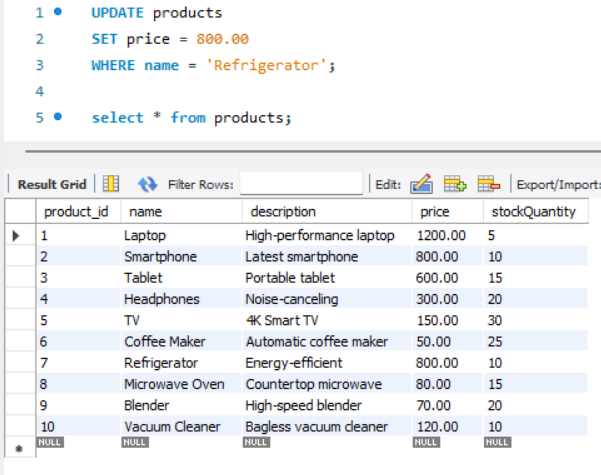
1. **SQL CHALLENGES:**
2. **Update refrigerator product price to 800.**

**Query:**

*UPDATE products SET price = 800*

*WHERE name = 'Refrigerator';*

**Result:**

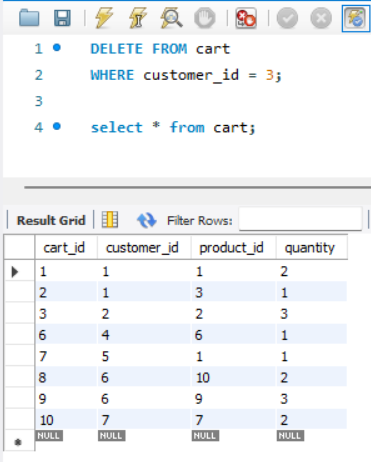
****

1. **Remove all cart items for a specific customer. (eg : Customer\_id = 3)**

**Query:**

*DELETE FROM cart WHERE customer\_id = 3;*

**Result:**

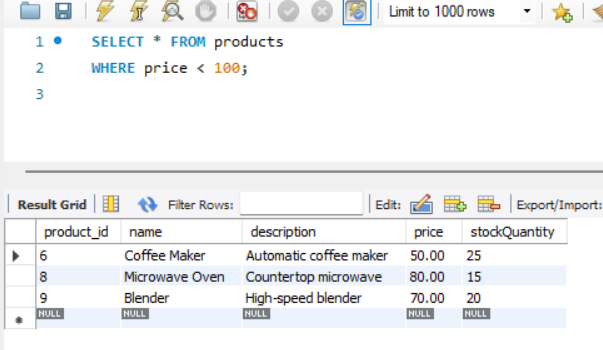
**

1. **Retrieve Products Priced Below $100.**

**Query:**

*Select \* from products WHERE price < 100;*

**Result:**

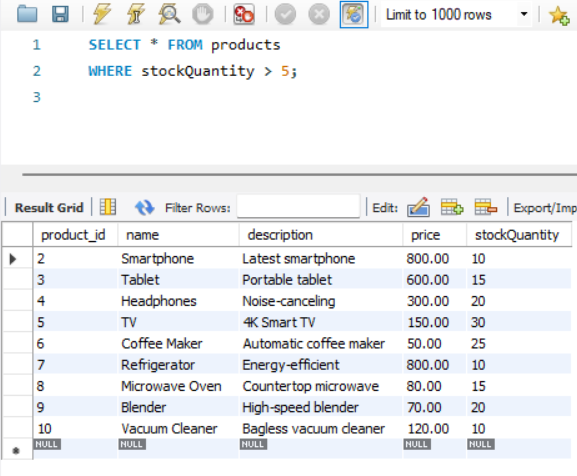
****

1. **Find Products with Stock Quantity Greater Than 5.**

**Query:**

*Select \* from products WHERE stockQuantity > 5;*

**Result:**

****

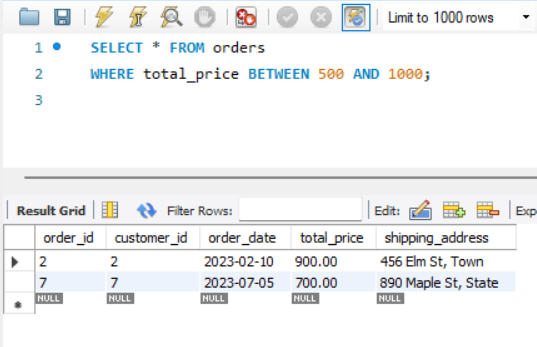
1. **Retrieve Orders with Total Amount Between $500 and $1000.**

**Query:**

*SELECT \* FROM orders*

*WHERE total\_price BETWEEN 500 AND 1000;*

**Result:**

****

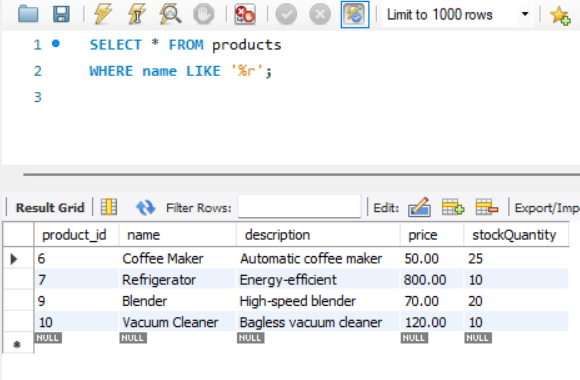
1. **Find Products which name end with letter ‘r’.**

**Query:**

*SELECT \* FROM products*

*WHERE name LIKE '%r';*

**Result:**

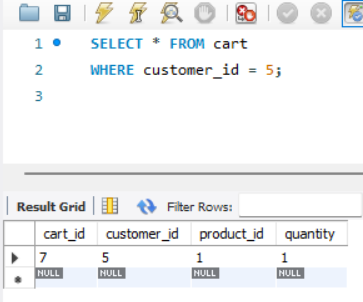
****

1. **Retrieve Cart Items for Customer 5.**

**Query:**

*SELECT \* FROM cart WHERE customer\_id = 5;*

**Result:**

****

1. **Find Customers Who Placed Orders in 2023.**

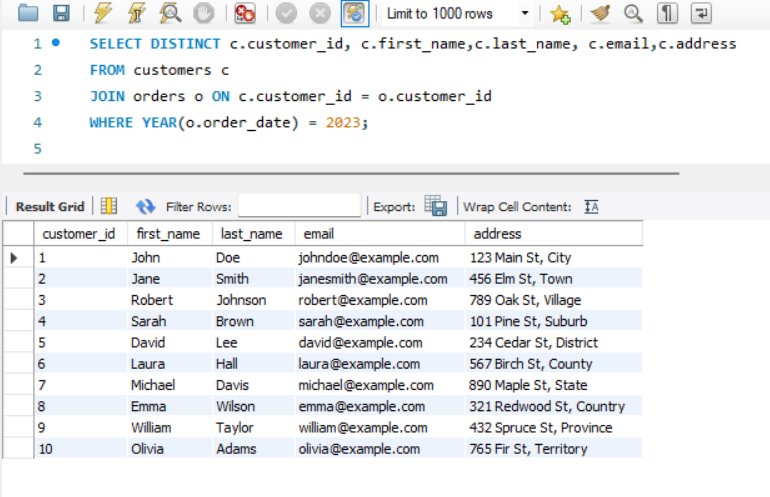
**Query:**

*SELECT DISTINCT c.customer\_id, c.first\_name,c.last\_name, c.email,c.address FROM customers c*

*JOIN orders o ON c.customer\_id = o.customer\_id*

*WHERE YEAR(o.order\_date) = 2023;*

**Result:**

****

1. **Determine the Minimum Stock Quantity for Each Product Category.**

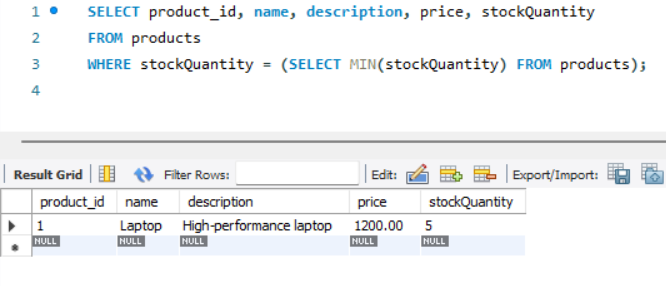
**Query:**

*SELECT product\_id, name, description, price, stockQuantity*

*FROM products*

*WHERE stockQuantity = (SELECT MIN(stockQuantity) FROM products);*

**Result:**

****

1. **Calculate the Total Amount Spent by Each Customer.**

**Query:**

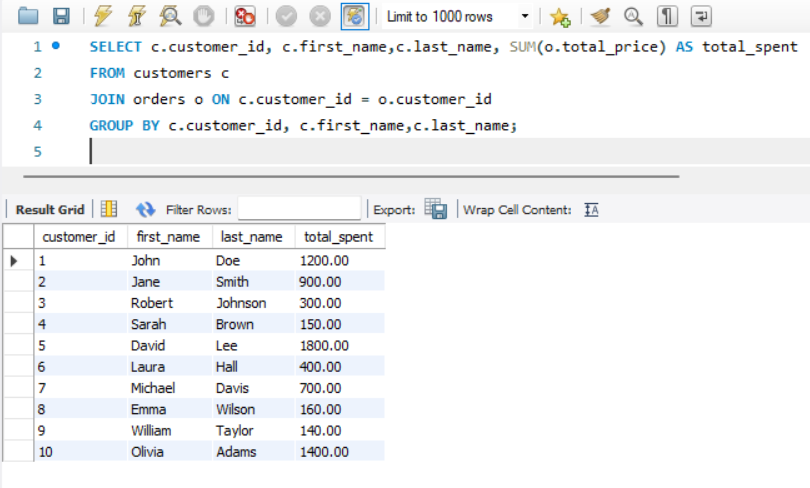
*SELECT c.customer\_id, c.name, SUM(o.total\_price) AS total\_spent*

*FROM customers c*

*JOIN orders o ON c.customer\_id = o.customer\_id*

*GROUP BY c.customer\_id, c.name;*

**Result:**

****

1. **Find the Average Order Amount for Each Customer.**

**Query:**

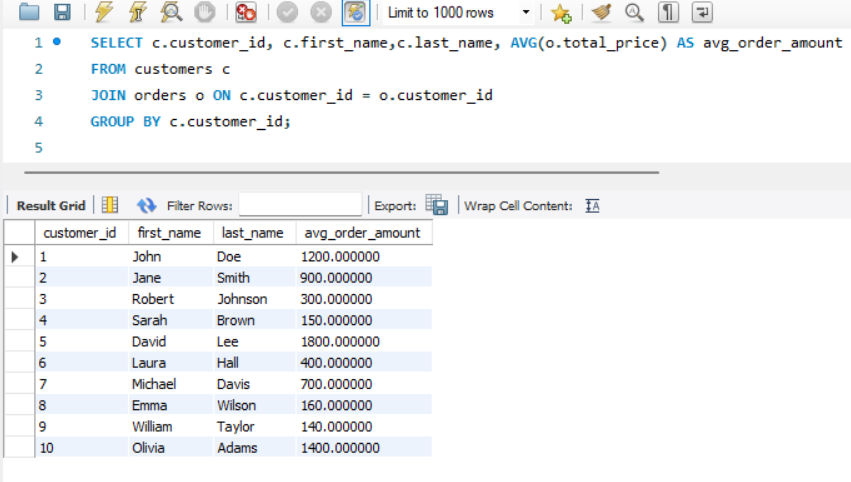
*SELECT c.customer\_id, c.first\_name,c.last\_name, AVG(o.total\_price) AS avg\_order\_amount*

*FROM customers c*

*JOIN orders o ON c.customer\_id = o.customer\_id*

*GROUP BY c.customer\_id;*

**Result:**

****

1. **Count the Number of Orders Placed by Each Customer.**

**Query:**

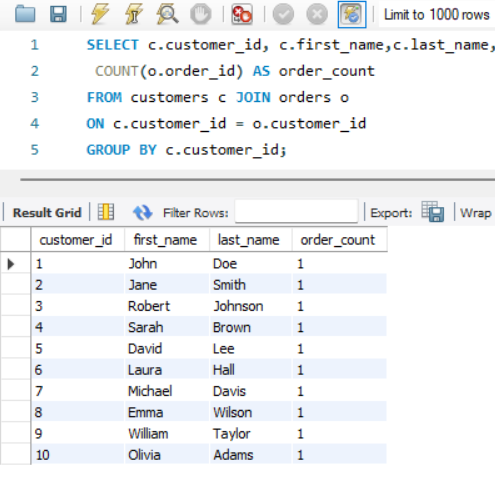
*SELECT c.customer\_id, c.first\_name,c.last\_name,*

*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;*

**Result:**

****

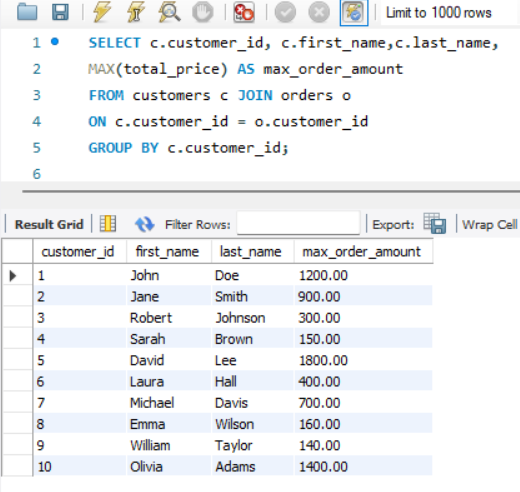
1. **Find the Maximum Order Amount for Each Customer**

**Query:**

*SELECT c.customer\_id, c.first\_name,c.last\_name, MAX(total\_price) AS max\_order\_amount FROM customers c JOIN orders o*

*ON c.customer\_id = o.customer\_id GROUP BY c.customer\_id;*

**Result:**

****

1. **Get Customers Who Placed Orders Totaling Over $1000.**

**Query:**

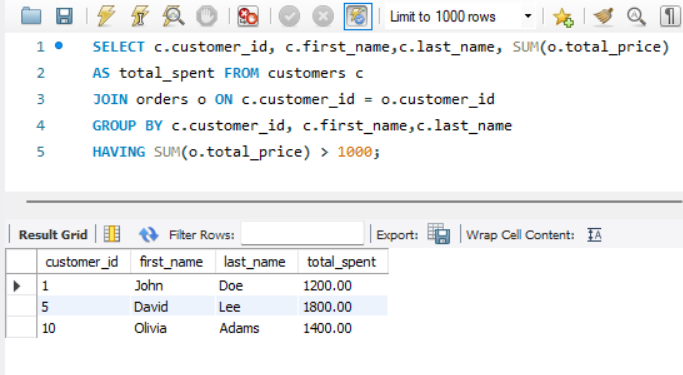
*SELECT c.customer\_id, c.first\_name,c.last\_name, SUM(o.total\_price) AS total\_spent FROM customers c*

*JOIN orders o ON c.customer\_id = o.customer\_id*

*GROUP BY c.customer\_id, c.first\_name,c.last\_name*

*HAVING SUM(o.total\_price) > 1000;*

**Result:**

****

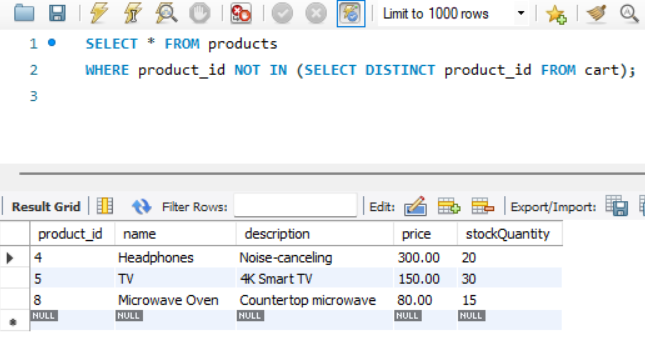
1. **Subquery to Find Products Not in the Cart.**

**Query:**

*SELECT \* FROM products*

*WHERE product\_id NOT IN (SELECT DISTINCT product\_id FROM cart);*

***Result:***

******

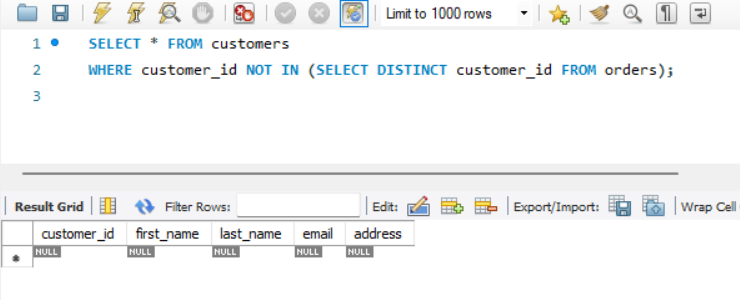
1. **Subquery to Find Customers Who Haven't Placed Orders.**

**Query:**

*SELECT \* FROM customers*

*WHERE customer\_id NOT IN (SELECT DISTINCT customer\_id FROM orders);*

**Result:**

****

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

**Query:**

*SELECT p.product\_id, p.name,*

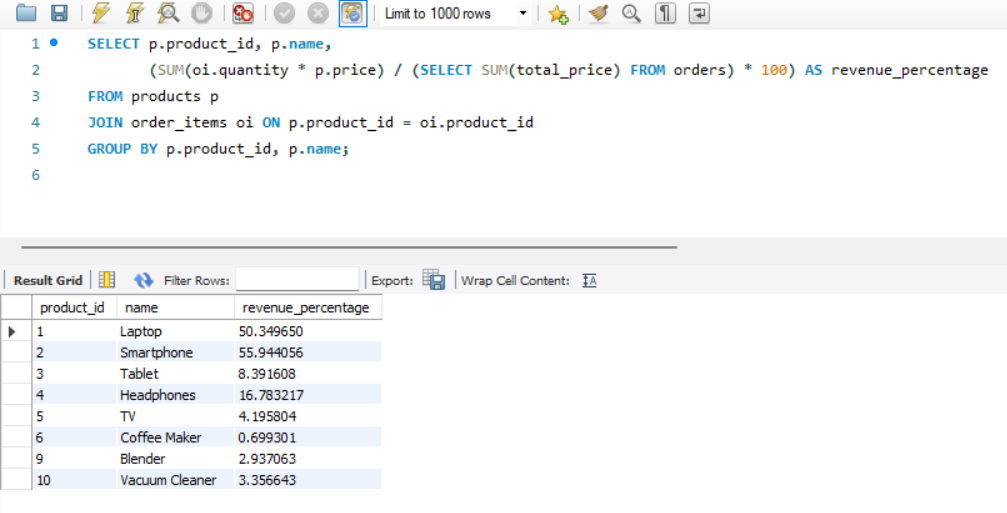
*(SUM(oi.quantity \* p.price) / (SELECT SUM(total\_price) FROM orders) \* 100) AS revenue\_percentage*

*FROM products p*

*JOIN order\_items oi ON p.product\_id = oi.product\_id*

*GROUP BY p.product\_id, p.name;*

**Result:**

****

1. **Subquery to Find Products with Low Stock.**

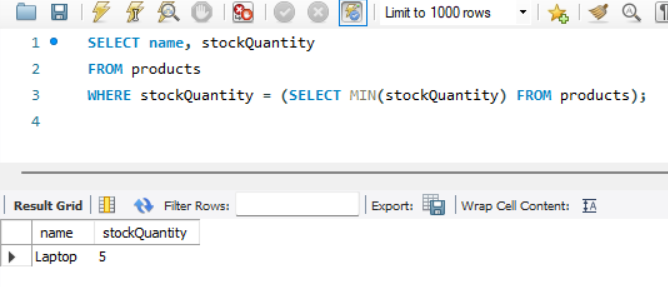
**Query:**

*SELECT name, stockQuantity*

*FROM products*

*WHERE stockQuantity = (SELECT MIN(stockQuantity) FROM products);*

**Result:**

**

1. **Subquery to Find Customers Who Placed High-Value Orders.**

**Query:**

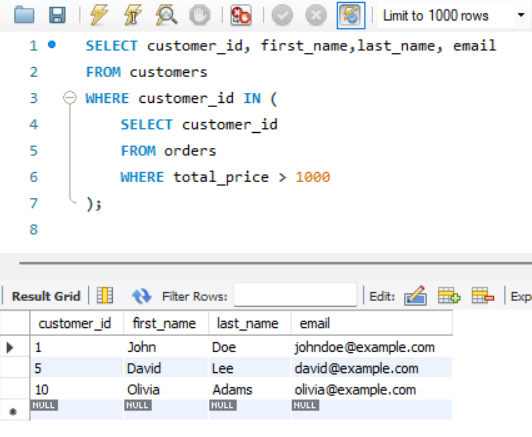
*SELECT customer\_id, first\_name,last\_name, email*

*FROM customers*

*WHERE customer\_id IN (SELECT customer\_id FROM orders*

*WHERE total\_price > 1000);*

**Result:**

****