create database Ecommerce;

use Ecommerce;

CREATE TABLE customers (

customer\_id INT PRIMARY KEY,

name VARCHAR(100),

email VARCHAR(100),

address VARCHAR(255)

);

CREATE TABLE products (

product\_id INT PRIMARY KEY,

name VARCHAR(100),

description TEXT,

price DECIMAL(10,2),

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,

order\_date DATE,

total\_price DECIMAL(10,2),

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,

item\_amount DECIMAL(10,2),

FOREIGN KEY (order\_id) REFERENCES orders(order\_id),

FOREIGN KEY (product\_id) REFERENCES products(product\_id)

);

INSERT INTO customers 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');

INSERT INTO products 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 orders 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 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);

INSERT INTO cart 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.

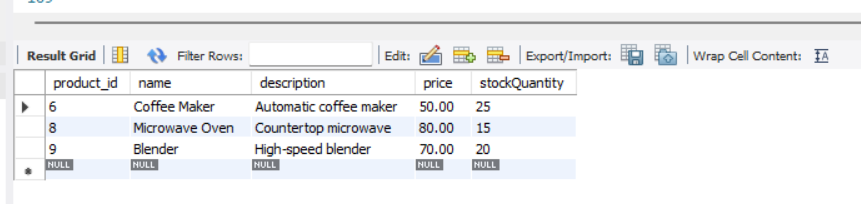
UPDATE products SET price = 800.00 WHERE name = 'Refrigerator';

2. Remove all cart items for a specific customer.

DELETE FROM cart WHERE customer\_id = 3;

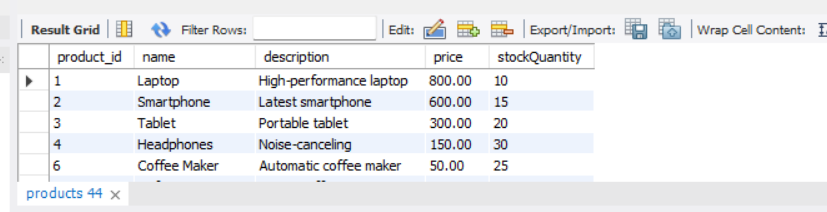
3. Retrieve Products Priced Below $100.

SELECT \* FROM products WHERE price < 100;



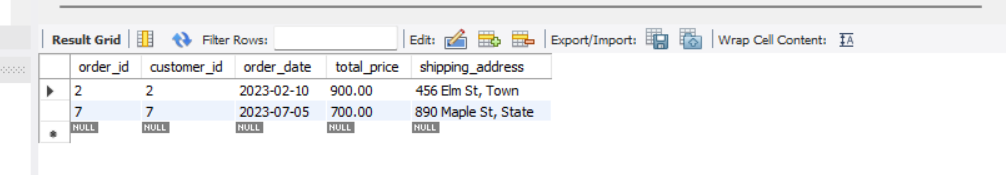
4. Find Products with Stock Quantity Greater Than 5.

SELECT \* FROM products WHERE stockQuantity > 5;



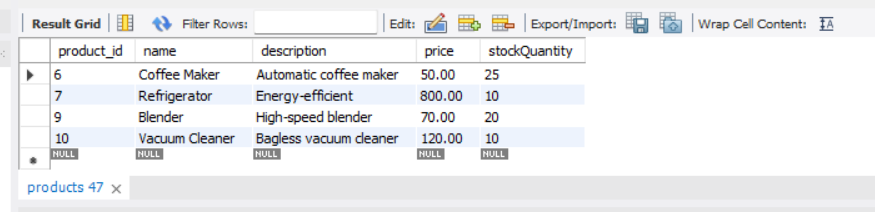
5. Retrieve Orders with Total Amount Between $500 and $1000.

SELECT \* FROM orders WHERE total\_price BETWEEN 500 AND 1000;



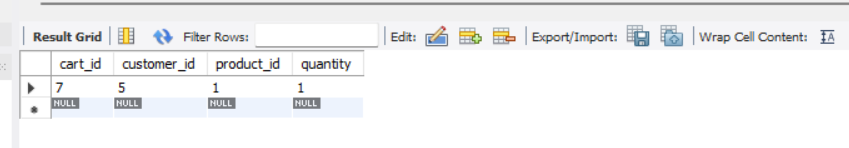
6. Find Products which name end with letter ‘r’.

SELECT \* FROM products WHERE name LIKE '%r';



7. Retrieve Cart Items for Customer 5.

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



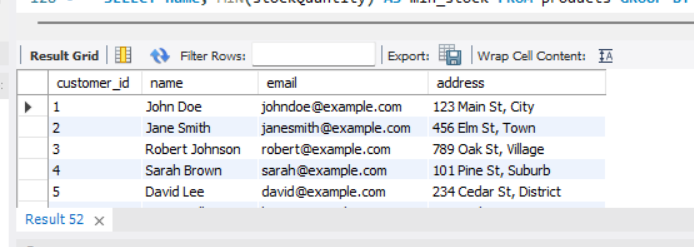
8. Find Customers Who Placed Orders in 2023.

SELECT DISTINCT c.\*

FROM customers c

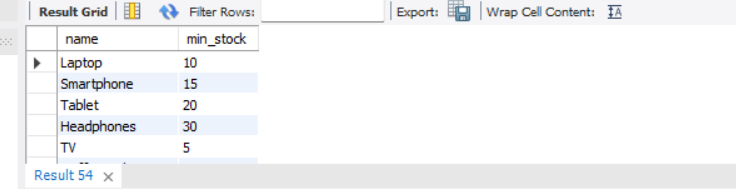
JOIN orders o ON c.customer\_id = o.customer\_id

WHERE YEAR(order\_date) = 2023;



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

SELECT name, MIN(stockQuantity) AS min\_stock FROM products GROUP BY name;



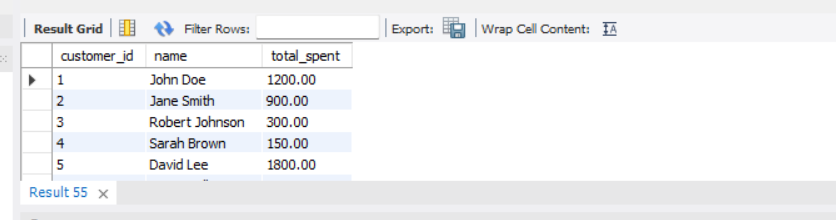
10. Calculate the Total Amount Spent by Each Customer.

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;



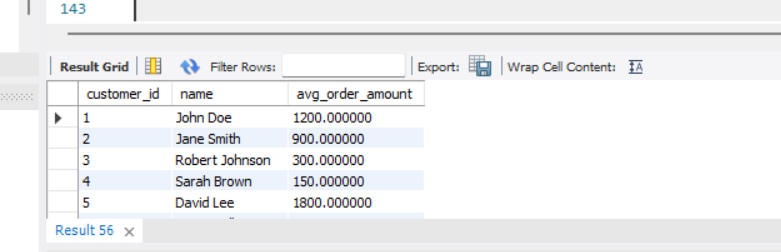
11. Find the Average Order Amount for Each Customer.

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;

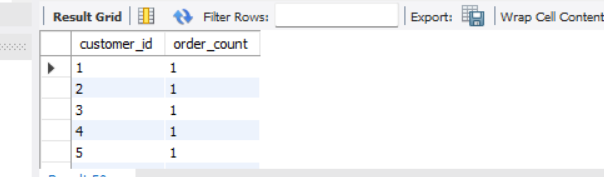


12. Count the Number of Orders Placed by Each Customer.

SELECT customer\_id, COUNT(\*) AS order\_count

FROM orders

GROUP BY customer\_id;

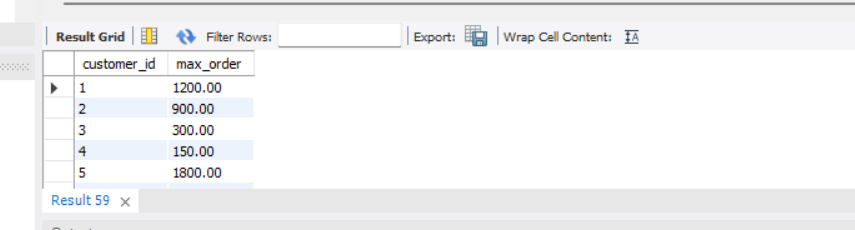


13. Find the Maximum Order Amount for Each Customer

SELECT customer\_id, MAX(total\_price) AS max\_order

FROM orders

GROUP BY customer\_id;



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

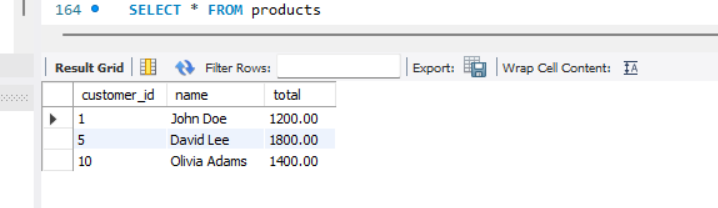
SELECT c.customer\_id, c.name, SUM(o.total\_price) AS total

FROM customers c

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

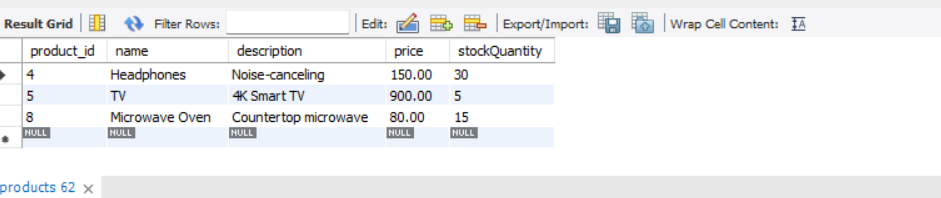
GROUP BY c.customer\_id

HAVING total > 1000;



15. Subquery to Find Products Not in the Cart

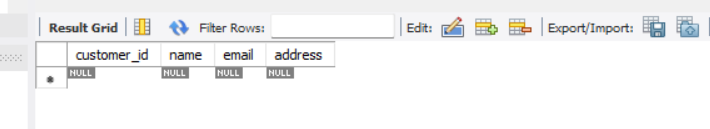
WHERE product\_id NOT IN (SELECT product\_id FROM cart);



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

SELECT \* FROM customers

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



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

SELECT

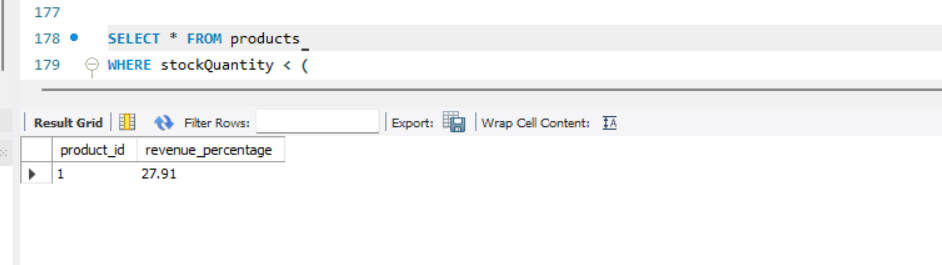
product\_id,

ROUND(SUM(item\_amount) / (SELECT SUM(item\_amount) FROM order\_items) \* 100, 2) AS revenue\_percentage

FROM order\_items

WHERE product\_id = 1

GROUP BY product\_id;

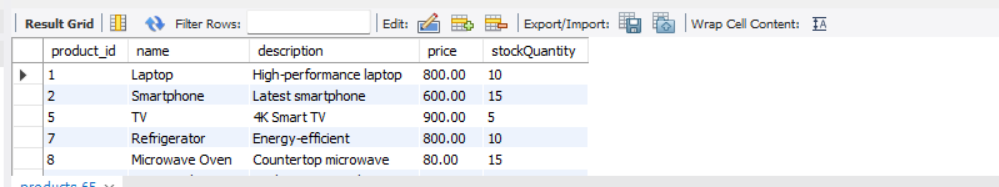


18. Subquery to Find Products with Low Stock.  
SELECT \* FROM products

WHERE stockQuantity < (

SELECT AVG(stockQuantity) FROM products

);



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

);

