**Ecommerce – SQL**

**Instructions**

* Coding Challenge submissions should be done through the partcipants’ Github repository, and the link should be shared with trainers and Hexavarsity.

**SQL Tables:**

1. **customers** table:
   * customer\_id (Primary Key)
   * first\_name
   * last\_name
   * email
   * address

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');

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

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);

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

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

INSERT INTO orders 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);

1. **order\_items** table (to store order details):
   * order\_item\_id (Primary Key)
   * order\_id (Foreign Key)
   * product\_id (Foreign Key)
   * quantity
   * amount

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);

**Product Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **productID** | **name** | **Description** | **price** | **stockQuantity** |
| 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 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **productID** | **name** | **Description** | **price** | **stockQuantity** |
| 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 |

**Customer Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **customerID** | **firstName** | **lastName** | **Email** | **address** |
| 1 | John | Doe | [johndoe@example.com](mailto:johndoe@example.com) | 123 Main St, City |
| 2 | Jane | Smith | [janesmith@example.com](mailto:janesmith@example.com) | 456 Elm St, Town |
| 3 | Robert | Johnson | [robert@example.com](mailto:robert@example.com) | 789 Oak St, Village |
| 4 | Sarah | Brown | [sarah@example.com](mailto:sarah@example.com) | 101 Pine St, Suburb |
| 5 | David | Lee | [david@example.com](mailto:david@example.com) | 234 Cedar St, District |
| 6 | Laura | Hall | [laura@example.com](mailto:laura@example.com) | 567 Birch St, County |
| 7 | Michael | Davis | [michael@example.com](mailto:michael@example.com) | 890 Maple St, State |
| 8 | Emma | Wilson | [emma@example.com](mailto:emma@example.com) | 321 Redwood St, Country |
| 9 | William | Taylor | [william@example.com](mailto:william@example.com) | 432 Spruce St, Province |
| 10 | Olivia | Adams | [olivia@example.com](mailto:olivia@example.com) | 765 Fir St, Territory |

**Order Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **orderID** | **customerID** | **orderDate** | **totalAmount** |
| 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 |

**OrderItem Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **orderItemID** | **orderID** | **productID** | **quantity** | **itemAmount** |
| 1 | 1 | 1 | 2 | 1600.00 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **orderItemID** | **orderID** | **productID** | **quantity** | **itemAmount** |
| 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 |

**Cart Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **cartID** | **customerID** | **productid** | **quantity** |
| 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';



1. Remove all cart items for a specific customer.

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

DELETE FROM cart WHERE customer\_id = 5;

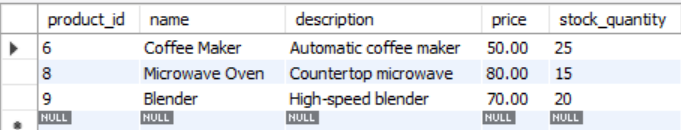
SELECT \* FROM cart WHERE customer\_id = 5;



1. Retrieve Products Priced Below $100.

SELECT \* FROM products

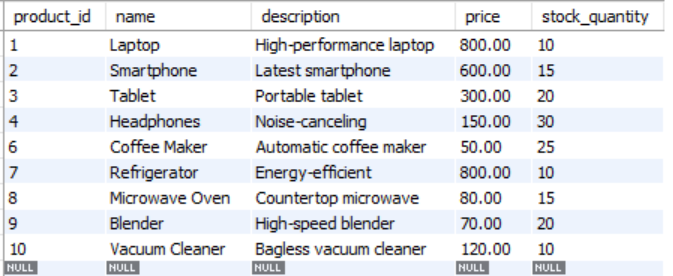
WHERE price < 100.00;



1. Find Products with Stock Quantity Greater Than 5.

SELECT \* FROM products

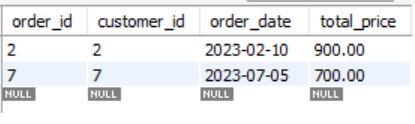
WHERE stock\_quantity > 5;



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

SELECT \* FROM orders

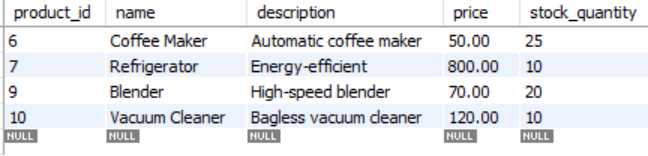
WHERE total\_price BETWEEN 500.00 AND 1000.00;



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

SELECT \* FROM products

WHERE name LIKE '%r';



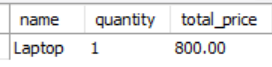
1. Retrieve Cart Items for Customer 5.

SELECT p.name, c.quantity, (p.price \* c.quantity) AS total\_price

FROM cart c

JOIN products p ON c.product\_id = p.product\_id

WHERE c.customer\_id = 5;



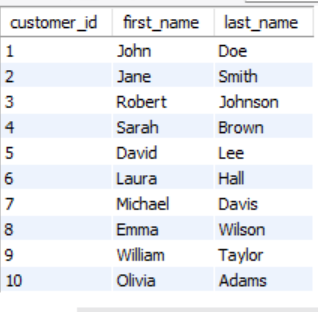
1. Find Customers Who Placed Orders in 2023.

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

FROM customers c

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

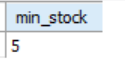
WHERE YEAR(o.order\_date) = 2023;



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

SELECT MIN(stock\_quantity) AS min\_stock

FROM products;



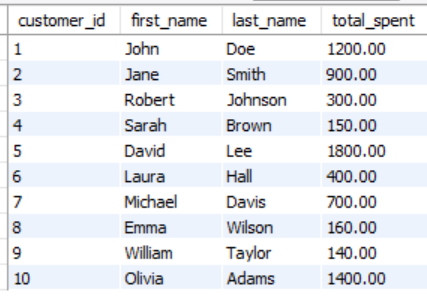
1. Calculate the Total Amount Spent by Each Customer.

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;



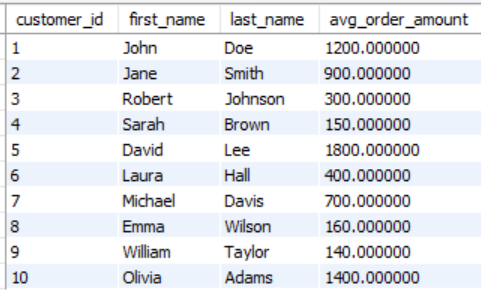
1. Find the Average Order Amount for Each Customer.

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, c.first\_name, c.last\_name;



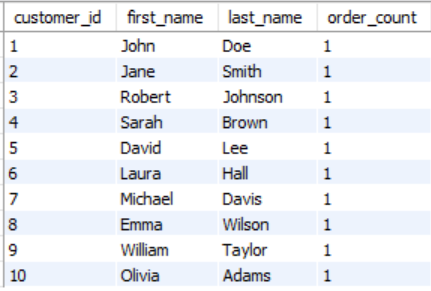
1. Count the Number of Orders Placed by Each Customer.

SELECT c.customer\_id, c.first\_name, c.last\_name, COUNT(o.order\_id) AS order\_count

FROM customers c

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

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



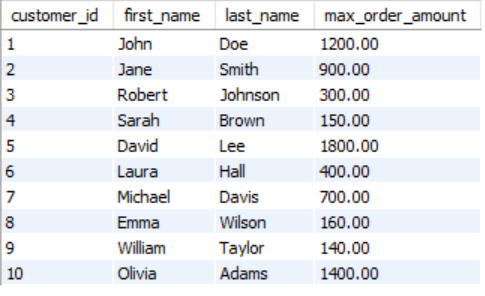
1. Find the Maximum Order Amount for Each Customer.

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

FROM customers c

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

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



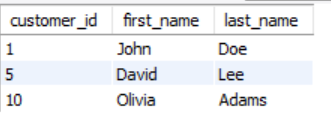
1. Get Customers Who Placed Orders Totaling Over $1000.

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

FROM customers c

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

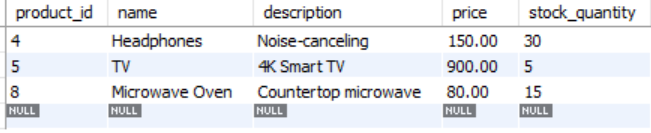
WHERE o.total\_price > 1000.00;



1. Subquery to Find Products Not in the Cart.

SELECT \* FROM products

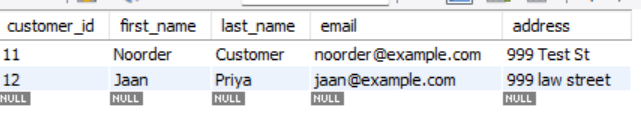
WHERE product\_id NOT IN (SELECT DISTINCT product\_id FROM cart);



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

SELECT \* FROM customers

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



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

SELECT

p.product\_id,

p.name,

SUM(oi.item\_amount) AS product\_revenue,

(SUM(oi.item\_amount) / (SELECT SUM(item\_amount) FROM order\_items)) \* 100 AS revenue\_percentage

FROM

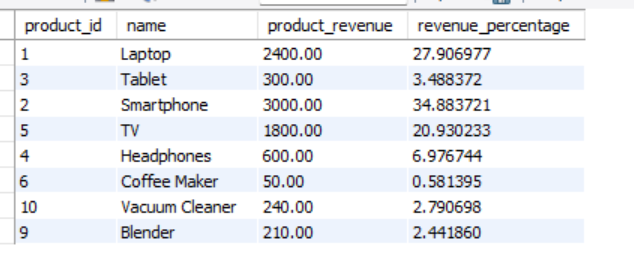
products p

JOIN

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

GROUP BY

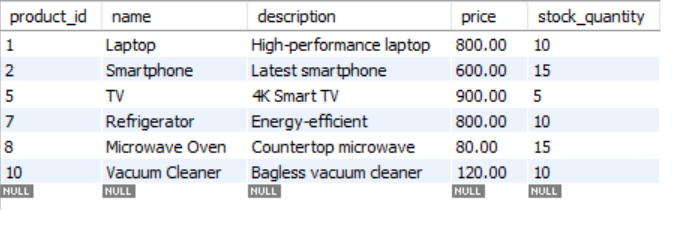
p.product\_id, p.name;



1. Subquery to Find Products with Low Stock.

SELECT \* FROM products

WHERE stock\_quantity < (SELECT AVG(stock\_quantity) FROM products);



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

SELECT

c.customer\_id,

c.first\_name,

c.last\_name,

o.total\_price AS order\_amount

FROM

customers c

JOIN

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

WHERE

o.total\_price > (SELECT AVG(total\_price) FROM orders)

ORDER BY

o.total\_price DESC;

