Foundation training Coding Challenge

-Sarthak Niranjan Kulkarni (Maverick)

- [sarthakkul2311@gmail.com](mailto:sarthakkul2311@gmail.com) - (+91) 93256 02791

**Day 1 – 4/11/2024 (Monday)**

**The queries below demonstrate all the basic SQL operations I practiced during my foundation training.**

1. **Update refrigerator product price to 800.**

🡪 update products

set Price=800.00

where name='Refrigerator';

select \* from products;

A screenshot of a computer

Description automatically generated

1. **Remove all cart items for a specific customer.**

🡪 delete from cart where customerId = 3;

select \* from cart;

A screenshot of a computer

Description automatically generated

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

🡪 Select \* from products where Price<100;

A screenshot of a computer

Description automatically generated

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

🡪 Select \* from products where stockQuantity>5;

A screenshot of a computer

Description automatically generated

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

🡪 select \* from orders where total\_price between 500 and 1000;

A screenshot of a computer

Description automatically generated

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

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

A screenshot of a computer

Description automatically generated

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

🡪 select \* from cart where customerId = 5;

A screenshot of a computer

Description automatically generated

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

🡪 SELECT DISTINCT c.CustomerId, c.FirstName, c.LastName, c.Email

FROM customers c

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

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

A screenshot of a computer

Description automatically generated

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

🡪 UPDATE products

SET category = CASE

WHEN productID IN (1, 2, 3,4) THEN 'Electronics'

WHEN productID IN (5, 6, 7) THEN 'Accessories'

WHEN productID IN (8, 9, 10) THEN 'Appliances'

END;

select \* from products;

A screenshot of a computer

Description automatically generated

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

🡪 SELECT c.CustomerId, c.FirstName, c.LastName, SUM(o.total\_price) AS TotalAmountSpent

FROM customers c

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

GROUP BY c.CustomerId, c.FirstName, c.LastName;

A screenshot of a computer

Description automatically generated

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

🡪 select customer\_id, avg(total\_price) as 'Average order amount' from orders group by customer\_id;

A screenshot of a computer

Description automatically generated

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

🡪 SELECT customer\_id, COUNT(\*) as 'No. of Orders'

FROM orders

GROUP BY customer\_id;

A screenshot of a computer

Description automatically generated

**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;

A screenshot of a computer

Description automatically generated

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

🡪 SELECT customer\_id FROM orders

GROUP BY customer\_id

HAVING SUM(total\_price) > 1000;

A screenshot of a computer

Description automatically generated

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

🡪 SELECT \* FROM products

WHERE productID NOT IN (SELECT productID FROM cart);

A screenshot of a computer

Description automatically generated

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

🡪 SELECT \* FROM customers

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

---(Everyone has place order so output should not display anything)

A screenshot of a computer

Description automatically generated

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

🡪 SELECT name,

(SELECT SUM(itemamount) FROM order\_items WHERE product\_id = p.productID) /

(SELECT SUM(itemamount) FROM order\_items) \* 100 AS revenue\_percentage

FROM products p;

A screenshot of a computer

Description automatically generated

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

🡪 SELECT \* FROM products

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

A screenshot of a computer

Description automatically generated

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

🡪 select c.\* from customers c

join orders o on c.CustomerId = o.customer\_id

where o.total\_price > (select avg (total\_price) from orders);

A screenshot of a computer

Description automatically generated