

Coding Challenge – 25/03/2025

Name: Monish Coumar S

Roll Number: J208

Git link: [https://github.com/mc-monish28/HM-bank-SQL-Assignment/tree/main/SQL%20Coding%20Challenge%20\(25.03.2025\)](https://github.com/mc-monish28/HM-bank-SQL-Assignment/tree/main/SQL%20Coding%20Challenge%20(25.03.2025))

Queries:

1. Update refrigerator product price to 800.

```
UPDATE products SET price = 800 WHERE name = 'Refrigerator';  
select * from products;
```

Result Grid

Filter Rows:

Edit:

Export/Import

	product_id	name	price	description	stockQuantity
▶	1	Laptop	800.00	High-performance laptop	10
	2	Smartphone	600.00	Latest smartphone	15
	3	Tablet	300.00	Portable tablet	20
	4	Headphones	150.00	Noise-canceling	30
	5	TV	900.00	4K Smart TV	5
	6	Coffee Maker	50.00	Automatic coffee maker	25
	7	Refrigerator	800.00	Energy-efficient	10
	8	Microwave Oven	80.00	Countertop microwave	15
	9	Blender	70.00	High-speed blender	20
	10	Vacuum Cleaner	120.00	Bagless vacuum cleaner	10
•	NULL	NULL	NULL	NULL	NULL

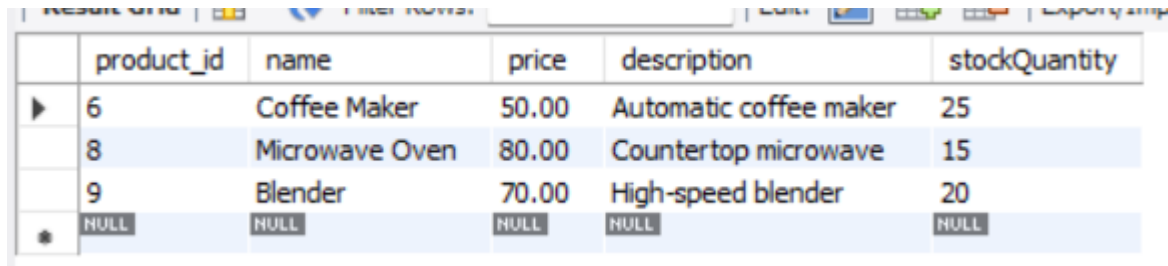
2. Remove all cart items for a specific customer.

```
DELETE FROM cart WHERE customer_id = 5;
```

✓ 183 12:53:21 DELETE FROM cart WHERE customer_id = 5

3. Retrieve Products Priced Below \$100.

```
SELECT * FROM products WHERE price < 100;
```



A screenshot of a database application's 'Result Grid' window. The window has a toolbar with icons for 'Filter Rows', 'Edit', and 'Export/Import'. The grid displays the results of the SQL query 'SELECT * FROM products WHERE price < 100;'. The table has six columns: 'product_id', 'name', 'price', 'description', and 'stockQuantity'. The data rows are: product_id 6, name 'Coffee Maker', price 50.00, description 'Automatic coffee maker', stockQuantity 25; product_id 8, name 'Microwave Oven', price 80.00, description 'Countertop microwave', stockQuantity 15; product_id 9, name 'Blender', price 70.00, description 'High-speed blender', stockQuantity 20; and a row with all NULL values. The last row is preceded by a star icon.

	product_id	name	price	description	stockQuantity
▶	6	Coffee Maker	50.00	Automatic coffee maker	25
	8	Microwave Oven	80.00	Countertop microwave	15
	9	Blender	70.00	High-speed blender	20
★	NULL	NULL	NULL	NULL	NULL

4. Find Products with Stock Quantity Greater Than 5.

```
SELECT * FROM products WHERE stockQuantity > 5;
```



A screenshot of a database application's 'Result Grid' window. The window has a toolbar with icons for 'Filter Rows', 'Edit', and 'Export/Import'. The grid displays the results of the SQL query 'SELECT * FROM products WHERE stockQuantity > 5;'. The table has six columns: 'product_id', 'name', 'price', 'description', and 'stockQuantity'. The data rows are: product_id 1, name 'Laptop', price 800.00, description 'High-performance laptop', stockQuantity 10; product_id 2, name 'Smartphone', price 600.00, description 'Latest smartphone', stockQuantity 15; product_id 3, name 'Tablet', price 300.00, description 'Portable tablet', stockQuantity 20; product_id 4, name 'Headphones', price 150.00, description 'Noise-canceling', stockQuantity 30; product_id 6, name 'Coffee Maker', price 50.00, description 'Automatic coffee maker', stockQuantity 25; product_id 7, name 'Refrigerator', price 800.00, description 'Energy-efficient', stockQuantity 10; product_id 8, name 'Microwave Oven', price 80.00, description 'Countertop microwave', stockQuantity 15; product_id 9, name 'Blender', price 70.00, description 'High-speed blender', stockQuantity 20; product_id 10, name 'Vacuum Cleaner', price 120.00, description 'Bagless vacuum cleaner', stockQuantity 10; and a row with all NULL values. The last row is preceded by a star icon.

	product_id	name	price	description	stockQuantity
▶	1	Laptop	800.00	High-performance laptop	10
	2	Smartphone	600.00	Latest smartphone	15
	3	Tablet	300.00	Portable tablet	20
	4	Headphones	150.00	Noise-canceling	30
	6	Coffee Maker	50.00	Automatic coffee maker	25
	7	Refrigerator	800.00	Energy-efficient	10
	8	Microwave Oven	80.00	Countertop microwave	15
	9	Blender	70.00	High-speed blender	20
	10	Vacuum Cleaner	120.00	Bagless vacuum cleaner	10
★	NULL	NULL	NULL	NULL	NULL

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

```
SELECT * FROM orders WHERE total_amount BETWEEN 500 AND 1000;
```

Result Grid					
		Filter Rows:		Edit:	
	order_id	customer_id	order_date	shipping_address	total_amount
▶	2	2	2023-02-10 00:00:00	456 Elm St, Town	900.00
	7	7	2023-07-05 00:00:00	890 Maple St, State	700.00
*	NULL	NULL	NULL	NULL	NULL

6. Find Products which name end with letter 'r' .

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

Result Grid					
		Filter Rows:		Edit:	
	product_id	name	price	description	stockQuantity
▶	6	Coffee Maker	50.00	Automatic coffee maker	25
	7	Refrigerator	800.00	Energy-efficient	10
	9	Blender	70.00	High-speed blender	20
	10	Vacuum Cleaner	120.00	Bagless vacuum cleaner	10
	NULL	NULL	NULL	NULL	NULL

7. Retrieve Cart Items for Customer 5.

SELECT * FROM cart WHERE customer_id = 4;

Result Grid				
		Filter Rows:		
	cart_id	customer_id	product_id	quantity
▶	6	4	6	1
*	NULL	NULL	NULL	NULL

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(o.order_date) = 2023;

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	customer_id	name	email	address
▶	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

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

SELECT name, MIN(stockQuantity) AS min_stock

FROM products

GROUP BY name;

Result Grid

Filter Rows:



	name	min_stock
▶	Laptop	10
	Smartphone	15
	Tablet	20
	Headphones	30
	TV	5
	Coffee Maker	25
	Refrigerator	10
	Microwave Oven	15
	Blender	20
	Vacuum Cleaner	10

10. Calculate the Total Amount Spent by Each Customer.

SELECT customer_id, SUM(total_amount) AS total_spent

FROM orders

GROUP BY customer_id;



Result Grid   Filter Rows: <input type="text"/>		
	customer_id	total_spent
▶	1	1200.00
	2	900.00
	3	300.00
	4	150.00
	5	1800.00
	6	400.00
	7	700.00
	8	160.00
	9	140.00
	10	1400.00

11. Find the Average Order Amount for Each Customer.

SELECT customer_id, AVG(total_amount) AS avg_order_amount

FROM orders

GROUP BY customer_id;



Result Grid   Filter Rows: <input type="text"/>		
	customer_id	avg_order_amount
▶	1	1200.000000
	2	900.000000
	3	300.000000
	4	150.000000
	5	1800.000000
	6	400.000000
	7	700.000000
	8	160.000000
	9	140.000000
	10	1400.000000

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

```
SELECT customer_id, COUNT(order_id) AS order_count
```

```
FROM orders
```

```
GROUP BY customer_id;
```

Result Grid   Filter Rows:		
	customer_id	order_count
▶	1	1
	2	1
	3	1
	4	1
	5	1
	6	1
	7	1
	8	1
	9	1
	10	1