

# Ecommerce

## Q1. Update refrigerator product price to 800.

The screenshot shows the MySQL Workbench interface with a single query window titled "Query 1". The SQL code is:

```
964
965 • UPDATE products
966     SET price = 800.00
967     WHERE product_id = 7;
968
969 • select * from products;
```

The "Result Grid" pane displays the "products" table with 9 rows. The 7th row, which corresponds to the refrigerator, has its "price" column updated to 800.00.

product_id	name	price	description	stockQuantity
1	Laptop	800	High-performance laptop	10
2	Smartphone	600	Latest smartphone	15
3	Tablet	300	Portable tablet	20
4	Headphones	150	Noise-cancelling	30
5	TV	900	4K Smart TV	5
6	Coffee Maker	50	Automatic coffee maker	25
7	Refrigerator	800	Energy-efficient	10
8	Microwave Oven	80	Countertop microwave	15
9	Blender	70	High-speed blender	20

The "Output" pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	16:10:40	UPDATE products SET price = 800.00 WHERE product_id = 7	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.015 sec
2	16:10:41	select * from products LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec

## Q2. Remove all cart items for a specific customer.

The screenshot shows the MySQL Workbench interface with a single query window titled "Query 1". The SQL code is:

```
970
971
972 • DELETE FROM cart
973     WHERE customer_id = 10;
974
975 • select * from cart;
```

The "Result Grid" pane displays the "cart" table with 9 rows. All rows where customer\_id is 10 are deleted.

cart_id	customer_id	product_id	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

The "Output" pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	16:12:30	DELETE FROM cart WHERE customer_id = 10	0 row(s) affected	0.015 sec
2	16:12:30	select * from cart LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec

### Q3. Retrieve Products Priced Below \$100

The screenshot shows the MySQL Workbench interface with a query editor window titled "Query 1". The query is:

```
972 • DELETE FROM cart
973 WHERE customer_id = 10;
974 • select * from cart;
975
976 • SELECT *
977 FROM products
978 WHERE price < 100.00;
979
```

The results grid displays the following data:

product_id	name	price	description	stockQuantity
6	Coffee Maker	50	Automatic coffee maker	25
8	Microwave Oven	80	Countertop microwave	15
9	Blender	70	High-speed blender	20

The status bar at the bottom right indicates the time is 4:14 PM and the date is 12/13/2023.

### Q4. Find Products with Stock Quantity Greater Than 5

The screenshot shows the MySQL Workbench interface with a query editor window titled "Query 1". The query is:

```
978 WHERE price < 100.00;
979
980 • SELECT *
981 FROM products
982 WHERE stockQuantity > 5;
983
984
985
```

The results grid displays the following data:

product_id	name	price	description	stockQuantity
1	Laptop	800	High-performance laptop	10
2	Smartphone	600	Latest smartphone	15
3	Tablet	300	Portable tablet	20
4	Headphones	150	Noise-cancelling	30
6	Coffee Maker	50	Automatic coffee maker	25
7	Refrigerator	800	Energy-efficient	10
8	Microwave Oven	80	Countertop microwave	15
9	Blender	70	High-speed blender	20
10	Vacuum Cleaner	120	Bagless vacuum cleaner	10

The status bar at the bottom right indicates the time is 4:14 PM and the date is 12/13/2023.

## Q5. Retrieve Orders with Total Amount Between \$500 and \$1000.

The screenshot shows the MySQL Workbench interface with the following details:

- Query Editor:** Displays the SQL query:

```
981 FROM products
982 WHERE stockQuantity > 5;
984 • SELECT *
985 FROM orders
986 WHERE total_price BETWEEN 500.00 AND 1000.00;
987
988
```
- Result Grid:** Shows the results of the query, which are two rows of data from the 'orders' table:

order_id	customer_id	order_date	total_price
2	2	2023-02-10	900
7	7	2023-07-05	700
- Output Panel:** Displays the execution log:

```
1 16:15:43 SELECT * FROM orders WHERE total_price BETWEEN 500.00 AND 1000.00 LIMIT 0, 1000
2 row(s) returned
Duration / Fetch
0.000 sec / 0.000 sec
```

## Q6. Find Products which name end with letter 'r'.

The screenshot shows the MySQL Workbench interface with the following details:

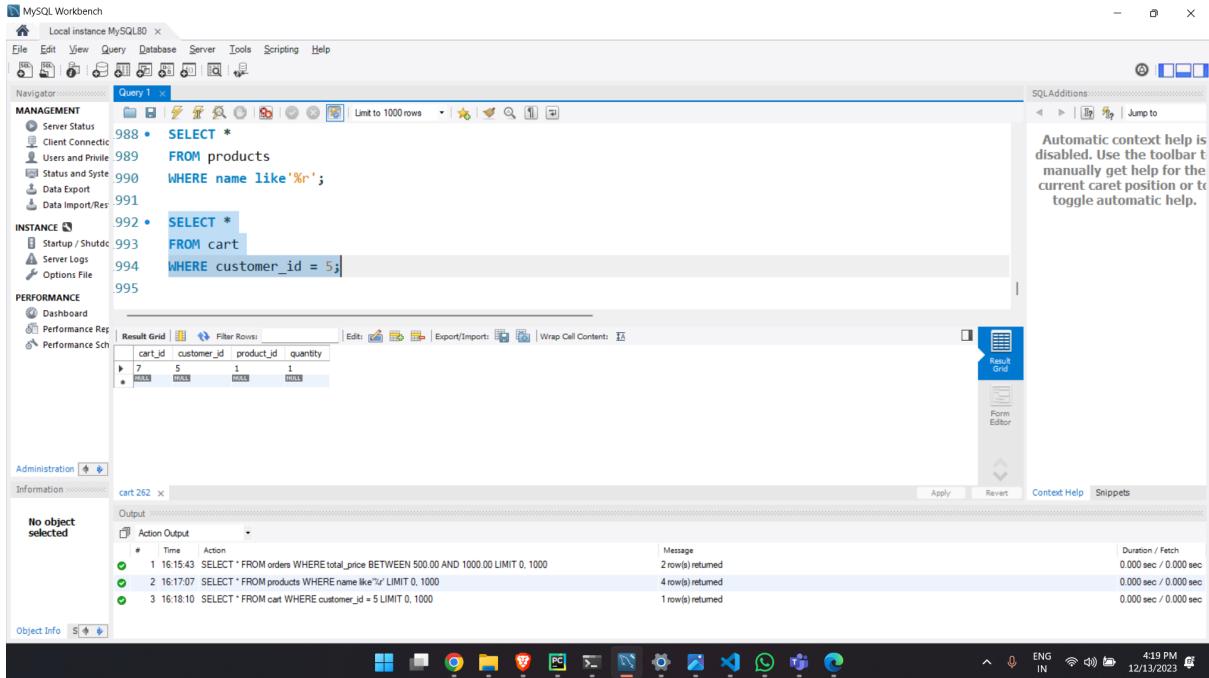
- Query Editor:** Displays the SQL queries:

```
984 • SELECT *
985 FROM orders
986 WHERE total_price BETWEEN 500.00 AND 1000.00;
988 • SELECT *
989 FROM products
990 WHERE name like '%r';
991
```
- Result Grid:** Shows the results of the second query, which are four rows of data from the 'products' table:

product_id	name	price	description	stockQuantity
6	Coffee Maker	50	Automatic coffee maker	25
7	Refrigerator	800	Energy-efficient	10
9	Blender	70	High-speed blender	20
10	Vacuum Cleaner	120	Bagless vacuum cleaner	10
- Output Panel:** Displays the execution logs for both queries:

```
1 16:15:43 SELECT * FROM orders WHERE total_price BETWEEN 500.00 AND 1000.00 LIMIT 0, 1000
2 row(s) returned
Duration / Fetch
0.000 sec / 0.000 sec
2 16:17:07 SELECT * FROM products WHERE name like '%r' LIMIT 0, 1000
4 row(s) returned
Duration / Fetch
0.000 sec / 0.000 sec
```

## Q7. Retrieve Cart Items for Customer 5.



The screenshot shows the MySQL Workbench interface with the following details:

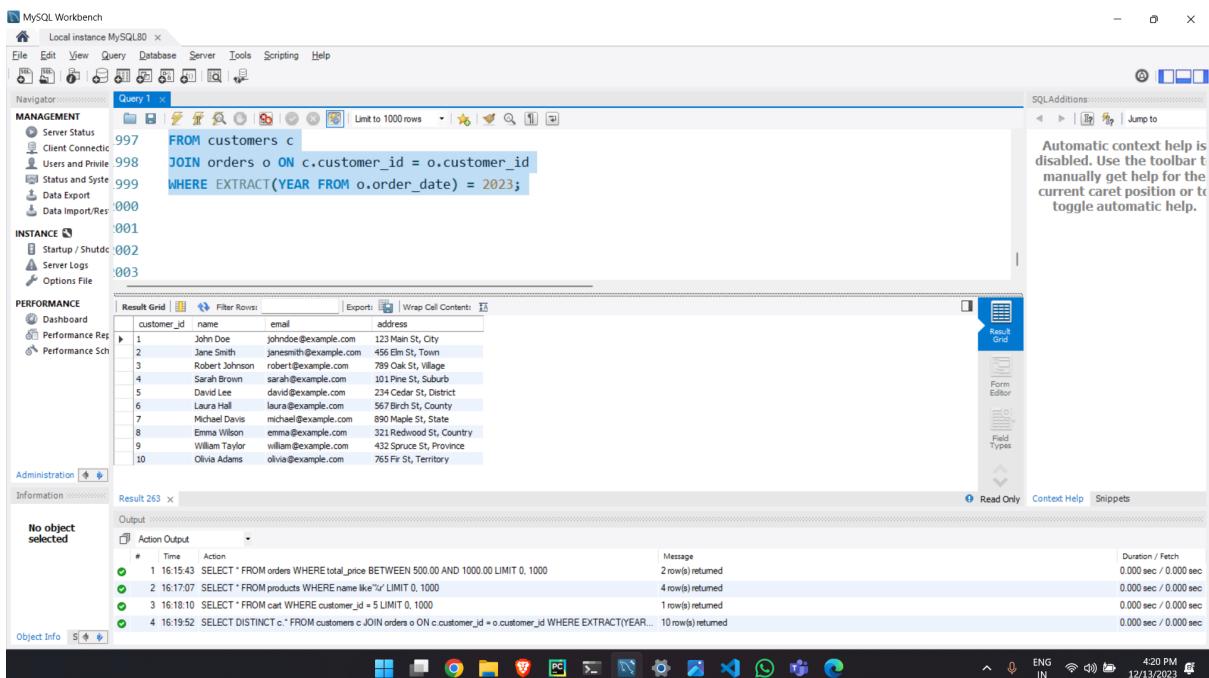
- Query Editor:** Displays two queries:

```
988 • SELECT *  
989   FROM products  
990   WHERE name like '%r';  
  
992 • SELECT *  
993   FROM cart  
994   WHERE customer_id = 5;
```
- Result Grid:** Shows the output of the second query:

cart_id	customer_id	product_id	quantity
7	5	1	1
- Output Window:** Shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	16:15:43	SELECT * FROM orders WHERE total_price BETWEEN 500.00 AND 1000.00 LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec
2	16:17:07	SELECT * FROM products WHERE name like "%r" LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec
3	16:18:10	SELECT * FROM cart WHERE customer_id = 5 LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

## Q8. Find Customers Who Placed Orders in 2023.



The screenshot shows the MySQL Workbench interface with the following details:

- Query Editor:** Displays the query:

```
997   FROM customers c  
998   JOIN orders o ON c.customer_id = o.customer_id  
999   WHERE EXTRACT(YEAR FROM o.order_date) = 2023;
```
- Result Grid:** Shows the output of the query:

customer_id	name	email	address
1	John Doe	john.doe@example.com	123 Main St, City
2	Jane Smith	jane.smith@example.com	456 Elm St, Town
3	Robert Johnson	robert.johnson@example.com	789 Oak St, Village
4	Sarah Brown	sarah.brown@example.com	101 Pine St, Suburb
5	David Lee	david.lee@example.com	234 Cedar St, District
6	Laura Hall	laura.hall@example.com	567 Birch St, County
7	Michael Davis	michael.davis@example.com	890 Maple St, State
8	Emma Wilson	emma.wilson@example.com	321 Redwood St, Country
9	William Taylor	william.taylor@example.com	432 Spruce St, Province
10	Olivia Adams	olivia.adams@example.com	765 Fir St, Territory
- Output Window:** Shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	16:15:43	SELECT * FROM orders WHERE total_price BETWEEN 500.00 AND 1000.00 LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec
2	16:17:07	SELECT * FROM products WHERE name like "%r" LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec
3	16:18:10	SELECT * FROM cart WHERE customer_id = 5 LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec
4	16:19:52	SELECT DISTINCT c.* FROM customers c JOIN orders o ON c.customer_id = o.customer_id WHERE EXTRACT(YEAR...	10 row(s) returned	0.000 sec / 0.000 sec

## Q9. Determine the Minimum Stock Quantity for Each Product Category.

The screenshot shows the MySQL Workbench interface with two queries in the Query Editor:

```
996 • SELECT DISTINCT c.*  
      FROM customers c  
997      JOIN orders o ON c.customer_id = o.customer_id  
998      WHERE EXTRACT(YEAR FROM o.order_date) = 2023;  
999  
000  
001 • SELECT name,stockQuantity FROM products;  
002
```

The Result Grid displays the following data:

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

The Output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	16:25:15	SELECT name,quantity FROM products LIMIT 0, 1000	Error Code: 1054. Unknown column 'quantity' in field list'	0.000 sec
2	16:25:42	SELECT name,stockQuantity FROM products LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec

## Q10. Calculate the Total Amount Spent by Each Customer.

The screenshot shows the MySQL Workbench interface with three queries in the Query Editor:

```
004 • SELECT c.customer_id,c.name AS customer_name,COALESCE(SUM(o.total_price), 0) AS total_amount  
      FROM customers c  
005      LEFT JOIN orders o ON c.customer_id = o.customer_id  
006      GROUP BY c.customer_id, c.name;  
007  
008  
009
```

The Result Grid displays the following data:

customer_id	customer_name	total_amount
1	John Doe	1200
2	Jane Smith	900
3	Robert Johnson	300
4	Sarah Brown	150
5	David Lee	1800
6	Laura Hall	400
7	Michael Davis	700
8	Emma Wilson	160
9	William Taylor	140
10	Olivia Adams	1400

The Output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	16:25:15	SELECT name,quantity FROM products LIMIT 0, 1000	Error Code: 1054. Unknown column 'quantity' in field list'	0.000 sec
2	16:25:42	SELECT name,stockQuantity FROM products LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec
3	16:26:55	SELECT c.customer_id,c.name AS customer_name,COALESCE(SUM(o.total_price), 0) AS total_amount_spent FROM customers c LEFT JOIN orders o ON c.customer_id = o.customer_id GROUP BY c.customer_id, c.name;	10 row(s) returned	0.047 sec / 0.000 sec
4	16:27:17	SELECT c.customer_id,c.name AS customer_name,COALESCE(SUM(o.total_price), 0) AS total_amount FROM customers c LEFT JOIN orders o ON c.customer_id = o.customer_id GROUP BY c.customer_id, c.name;	10 row(s) returned	0.000 sec / 0.000 sec

## Q11. Find the Average Order Amount for Each Customer.

The screenshot shows the MySQL Workbench interface with a query editor and results grid. The query is:

```
008 •    SELECT c.customer_id, c.name AS customer_name, COALESCE(AVG(o.total_price), 0) AS average_order_amount
009 •      FROM customers c
010     LEFT JOIN orders o ON c.customer_id = o.customer_id
011
012   GROUP BY c.customer_id, c.name;
```

The results grid displays the following data:

customer_id	customer_name	average_order_amount
1	John Doe	1200
2	Jane Smith	900
3	Robert Johnson	300
4	Sarah Brown	150
5	David Lee	1800
6	Laura Hall	400
7	Michael Davis	700
8	Emma Wilson	160
9	William Taylor	140
10	Olivia Adams	1400

The output pane shows the following log entries:

#	Time	Action	Message	Duration / Fetch
2	16:25:42	SELECT name.stockQuantity FROM products LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec
3	16:26:55	SELECT c.customer_id,c.name AS customer_name,COALESCE(SUM(o.total_price), 0) AS total_amount_spent FROM ...	10 row(s) returned	0.047 sec / 0.000 sec
4	16:27:17	SELECT c.customer_id,c.name AS customer_name,COALESCE(SUM(o.total_price), 0) AS total_amount FROM custom... 10 row(s) returned	10 row(s) returned	0.000 sec / 0.000 sec
5	16:28:13	SELECT c.customer_id,c.name AS customer_name, COALESCE(AVG(o.total_price), 0) AS average_order_amount FR...	10 row(s) returned	0.000 sec / 0.000 sec

## Q12. Count the Number of Orders Placed by Each Customer.

The screenshot shows the MySQL Workbench interface with a query editor and results grid. The query is:

```
013
014 •    SELECT c.customer_id, c.name AS customer_name, COUNT(o.order_id) AS num_orders
015   FROM customers c
016     LEFT JOIN orders o ON c.customer_id = o.customer_id
017
018   GROUP BY c.customer_id, c.name;
```

The results grid displays the following data:

customer_id	customer_name	num_orders
1	John Doe	1
2	Jane Smith	1
3	Robert Johnson	1
4	Sarah Brown	1
5	David Lee	1
6	Laura Hall	1
7	Michael Davis	1
8	Emma Wilson	1
9	William Taylor	1
10	Olivia Adams	1

The output pane shows the following log entries:

#	Time	Action	Message	Duration / Fetch
3	16:26:55	SELECT c.customer_id,c.name AS customer_name,COALESCE(SUM(o.total_price), 0) AS total_amount_spent FROM ...	10 row(s) returned	0.047 sec / 0.000 sec
4	16:27:17	SELECT c.customer_id,c.name AS customer_name,COALESCE(SUM(o.total_price), 0) AS total_amount FROM custom... 10 row(s) returned	10 row(s) returned	0.000 sec / 0.000 sec
5	16:28:13	SELECT c.customer_id,c.name AS customer_name, COALESCE(AVG(o.total_price), 0) AS average_order_amount FR...	10 row(s) returned	0.000 sec / 0.000 sec
6	16:29:39	SELECT c.customer_id,c.name AS customer_name, COUNT(o.order_id) AS num_orders FROM customers c LEFT JOI...	10 row(s) returned	0.000 sec / 0.000 sec

### Q13. Find the Maximum Order Amount for Each Customer.

The screenshot shows the MySQL Workbench interface with a query editor and results grid. The query retrieves the customer ID, name, and maximum total price for each customer from the 'customers' and 'orders' tables.

```
018
019 •  SELECT c.customer_id, c.name AS customer_name, COALESCE(MAX(o.total_price), 0) AS max_order_amount
020   FROM customers c
021     LEFT JOIN orders o ON c.customer_id = o.customer_id GROUP BY c.customer_id, c.name;
022
```

customer_id	customer_name	max_order_amount
1	John Doe	1200
2	Jane Smith	900
3	Robert Johnson	300
4	Sarah Brown	150
5	David Lee	1800
6	Laura Hall	400
7	Michael Davis	700
8	Emma Wilson	160
9	William Taylor	140
10	Olivia Adams	1400

The results grid shows 10 rows of data. The output pane indicates 10 row(s) returned.

### Q14. Get Customers Who Placed Orders Totaling Over \$1000

The screenshot shows the MySQL Workbench interface with a query editor and results grid. The query retrieves the customer ID, name, and total amount spent for each customer from the 'customers' and 'orders' tables, filtering for customers whose total spending is greater than 1000.

```
021   LEFT JOIN orders o ON c.customer_id = o.customer_id GROUP BY c.customer_id, c.name;
022
023 •  SELECT c.customer_id, c.name AS customer_name, COALESCE(SUM(o.total_price), 0) AS total_amount_spent
024   FROM customers c
025   LEFT JOIN orders o ON c.customer_id = o.customer_id
026   GROUP BY c.customer_id, c.name HAVING COALESCE(SUM(o.total_price), 0) > 1000;
027
```

customer_id	customer_name	total_amount_spent
1	John Doe	1200
5	David Lee	1800
10	Olivia Adams	1400

The results grid shows 3 rows of data. The output pane indicates 3 row(s) returned.

## Q15. Subquery to Find Products Not in the Cart.

The screenshot shows the MySQL Workbench interface with the following details:

- Query Editor:** Displays the SQL query:

```
028
029
030 • SELECT p.product_id, p.name
  FROM products p
031 WHERE p.product_id NOT IN (SELECT product_id FROM cart);
032
033
034
```
- Result Grid:** Shows the output of the query:

product_id	name
8	Microwave Oven
- Output Panel:** Shows the execution log:

```
Action Output
# Time Action
1 16:34:22 SELECT p.product_id, p.name FROM products p WHERE p.product_id NOT IN (SELECT product_id FROM cart) LIMIT... 1 row(s) returned
Duration / Fetch
0.016 sec / 0.000 sec
```
- System Tray:** Shows the date and time as 12/13/2023 4:34 PM.

## Q16. Subquery to Find Customers Who Haven't Placed Orders.

The screenshot shows the MySQL Workbench interface with the following details:

- Query Editor:** Displays the SQL query:

```
032 WHERE p.product_id NOT IN (SELECT product_id FROM cart);
033
034 • SELECT c.customer_id, c.name
  FROM customers c
035
036 WHERE c.customer_id NOT IN (SELECT DISTINCT customer_id FROM orders);
037
038
```
- Result Grid:** Shows the output of the query:

customer_id	name
1001	Mark
- Output Panel:** Shows the execution log:

```
Action Output
# Time Action
1 16:34:22 SELECT p.product_id, p.name FROM products p WHERE p.product_id NOT IN (SELECT product_id FROM cart) LIMIT... 1 row(s) returned
2 16:36:15 SELECT c.customer_id,c.name FROM customers c WHERE c.customer_id NOT IN (SELECT DISTINCT customer_id F... 0 row(s) returned
Duration / Fetch
0.016 sec / 0.000 sec
0.000 sec / 0.000 sec
```
- System Tray:** Shows the date and time as 12/13/2023 4:36 PM.

## Q17. Subquery to Calculate the Percentage of Total Revenue for a Product.

The screenshot shows the MySQL Workbench interface with the following details:

- Query Editor:** Contains the following SQL code:

```
037 •    SELECT product_name, (COALESCE(SUM(oi.item_amount), 0) / (SELECT COALESCE(SUM(item_amount), 0) FROM order_items)) * 100
038 •        AS percentage_of_total_revenue
039
040     WHERE product_id = oi.product_id
041
042
```
- Result Grid:** Displays the results for 10 products, showing their names and calculated percentages of total revenue.

product_id	product_name	percentage_of_total_revenue
1	Laptop	27.90714401989758
2	Smartphone	34.883639805488826
3	Tablet	3.488363980548882
4	Headphones	6.97623951097764
5	TV	20.930183883293292
6	Coffee Maker	0.5813939967581471
7	Refrigerator	0
8	Microwave Oven	0
9	Blender	2.4418547863842175
10	Vacuum Cleaner	2.7906911844391056
- Output Window:** Shows the execution log with the following entries:
  - 1 16:34:22 SELECT p.product\_id, p.name FROM products p WHERE p.product\_id NOT IN (SELECT product\_id FROM cat) LIMIT... 1 row(s) returned Duration / Fetch sec 0.016 sec / 0.000 sec
  - 2 16:36:15 SELECT c.customer\_id, c.name FROM customers c WHERE c.customer\_id NOT IN (SELECT DISTINCT customer\_id F... 0 row(s) returned 0.000 sec / 0.000 sec
  - 3 16:38:00 p.product\_id,p.name AS product\_name,(COALESCE(SUM(oi.item\_amount), 0) / (SELECT COALESCE(SUM(item\_amou... Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server versi... 0.000 sec
  - 4 16:38:18 SELECT p.product\_id,p.name AS product\_name,(COALESCE(SUM(oi.item\_amount), 0) / (SELECT COALESCE(SUM(item... 10 row(s) returned 0.016 sec / 0.000 sec

## Q18. Subquery to Find Products with Low Stock.

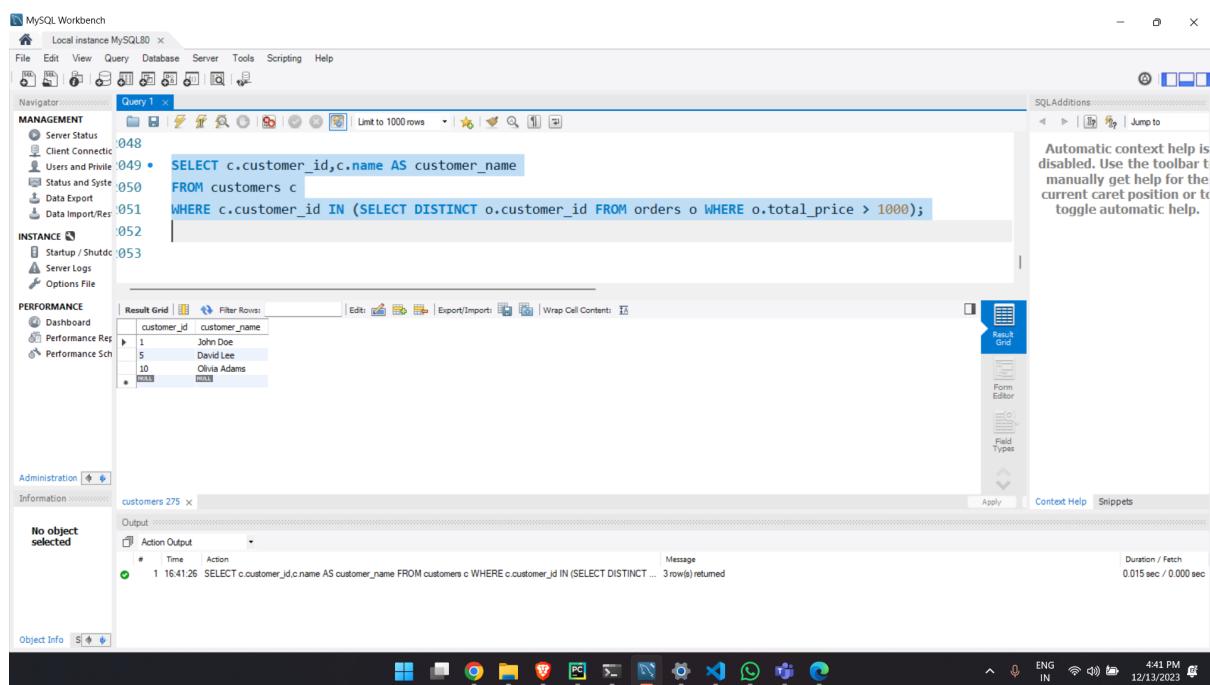
The screenshot shows the MySQL Workbench interface with the following details:

- Query Editor:** Contains the following SQL code:

```
042
043
044 •    SELECT product_id, name AS product_name, stockQuantity
045     FROM products
046
047     WHERE stockQuantity < (SELECT AVG(stockQuantity) FROM products);
048
```
- Result Grid:** Displays the results for 10 products, showing their names and current stock quantities.

product_id	product_name	stockQuantity
1	Laptop	10
2	Smartphone	15
5	TV	5
7	Refrigerator	10
8	Microwave Oven	15
10	Vacuum Cleaner	10
11	None	None
- Output Window:** Shows the execution log with the following entries:
  - 2 16:36:15 SELECT c.customer\_id,c.name FROM customers c WHERE c.customer\_id NOT IN (SELECT DISTINCT customer\_id F... 0 row(s) returned 0.000 sec / 0.000 sec
  - 3 16:38:00 p.product\_id,p.name AS product\_name,(COALESCE(SUM(oi.item\_amount), 0) / (SELECT COALESCE(SUM(item\_amou... Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server versi... 0.000 sec
  - 4 16:38:18 SELECT p.product\_id,p.name AS product\_name,(COALESCE(SUM(oi.item\_amount), 0) / (SELECT COALESCE(SUM(item... 10 row(s) returned 0.016 sec / 0.000 sec
  - 5 16:39:33 SELECT product\_id,name AS product\_name,stockQuantity FROM products WHERE stockQuantity < (SELECT AVG(s... 6 row(s) returned 0.000 sec / 0.000 sec

## Q19. Subquery to Find Customers Who Placed High-Value Orders.



The screenshot shows the MySQL Workbench interface with a query editor and a results grid. The query is:

```
048 • SELECT c.customer_id, c.name AS customer_name
FROM customers c
WHERE c.customer_id IN (SELECT DISTINCT o.customer_id FROM orders o WHERE o.total_price > 1000);
```

The results grid displays the following data:

customer_id	customer_name
1	John Doe
5	David Lee
10	Olivia Adams