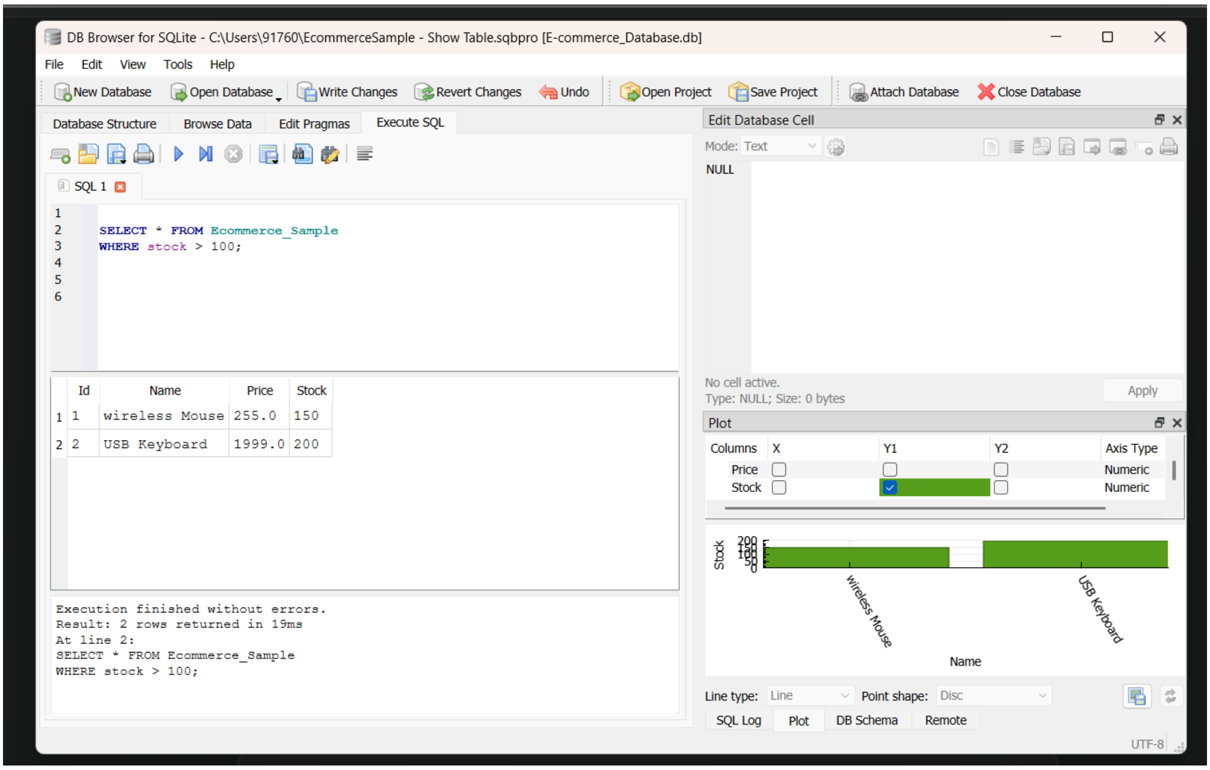
Queries I Practiced

1. Show only products with stock greater than 100



```sql
SELECT * FROM Ecommerce_Sample
WHERE stock > 100;
```

	Id	Name	Price	Stock
1	1	wireless Mouse	255.0	150
2	2	USB Keyboard	1999.0	200

```
Execution finished without errors.
Result: 2 rows returned in 19ms
At line 2:
SELECT * FROM Ecommerce_Sample
WHERE stock > 100;
```

Mode: Text

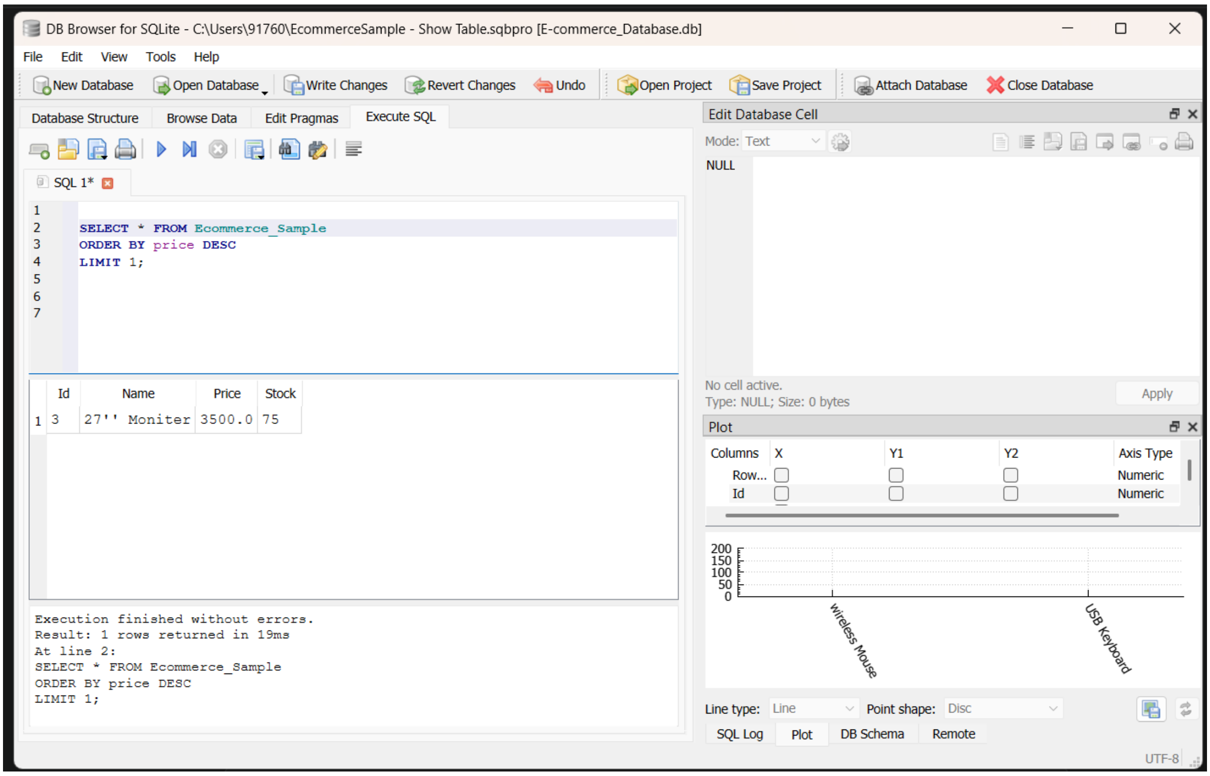
NULL

No cell active.
Type: NULL; Size: 0 bytes

Columns	X	Y1	Y2	Axis Type
Price	☐	☐	☐	Numeric
Stock	☐	☑	☐	Numeric

Line type: Line
Point shape: Disc

2. Show the most expensive product



```sql
SELECT * FROM Ecommerce_Sample
ORDER BY price DESC
LIMIT 1;
```

	Id	Name	Price	Stock
1	3	27'' Moniter	3500.0	75

```
Execution finished without errors.
Result: 1 rows returned in 19ms
At line 2:
SELECT * FROM Ecommerce_Sample
ORDER BY price DESC
LIMIT 1;
```

Mode: Text

NULL

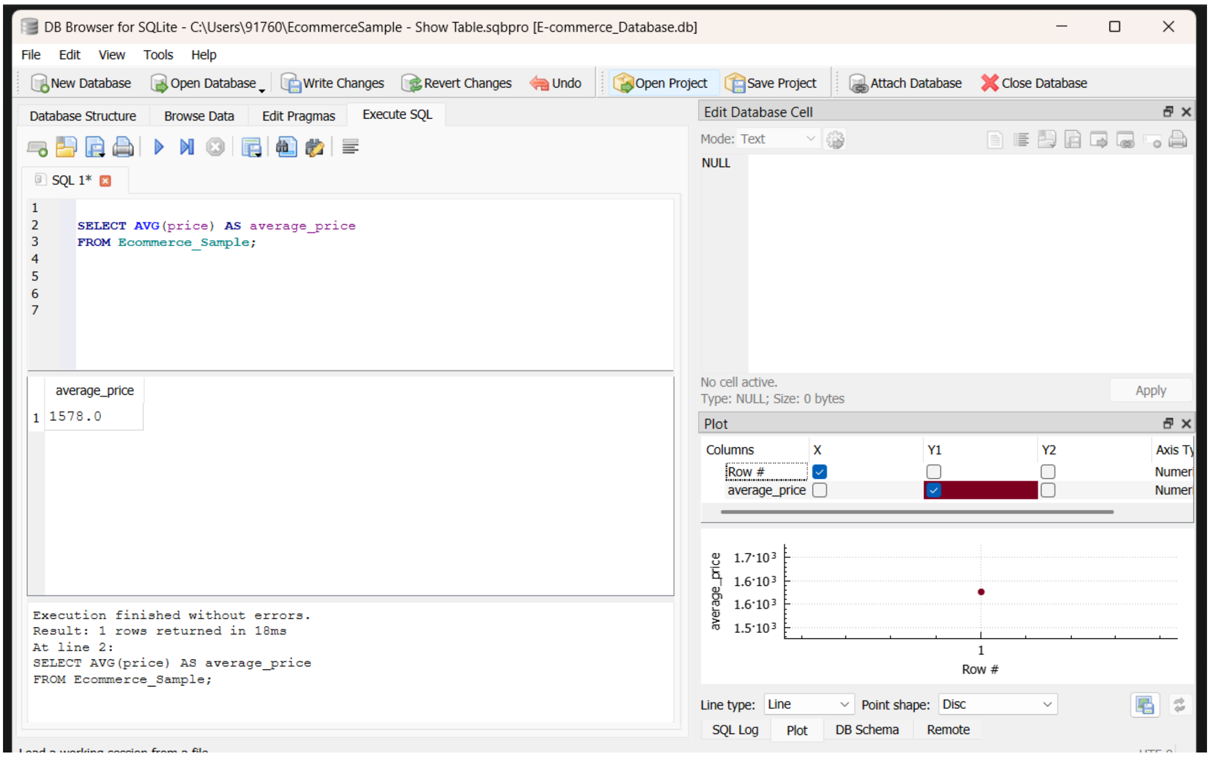
No cell active.
Type: NULL; Size: 0 bytes

Columns	X	Y1	Y2	Axis Type
Row...	☐	☐	☐	Numeric
Id	☐	☐	☐	Numeric

Line type: Line
Point shape: Disc

Queries I Practiced

3. Calculate the average price of products

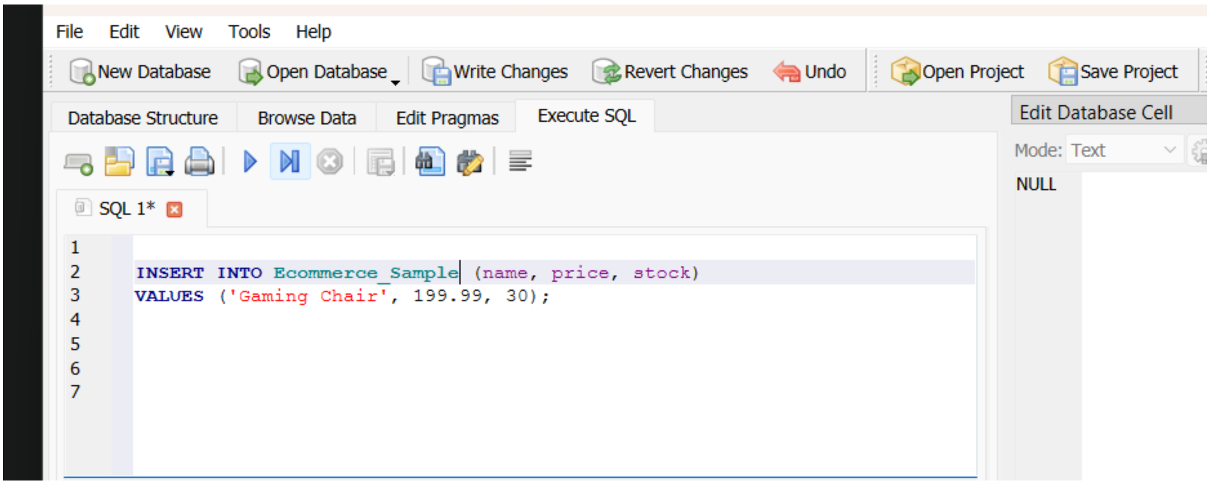


```
SELECT AVG(price) AS average_price
FROM Ecommerce_Sample;
```

	average_price
1	1578.0

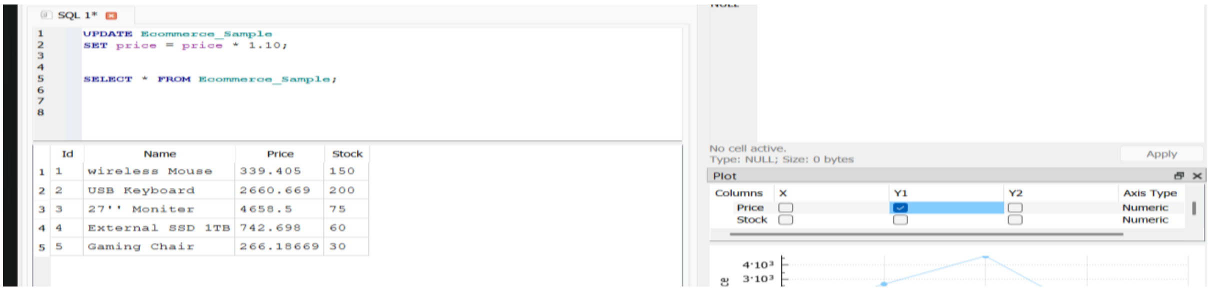
```
Execution finished without errors.
Result: 1 rows returned in 18ms
At line 2:
SELECT AVG(price) AS average_price
FROM Ecommerce_Sample;
```

4. Add a new product



```
INSERT INTO Ecommerce_Sample (name, price, stock)
VALUES ('Gaming Chair', 199.99, 30);
```

5. Increase all prices by 10%

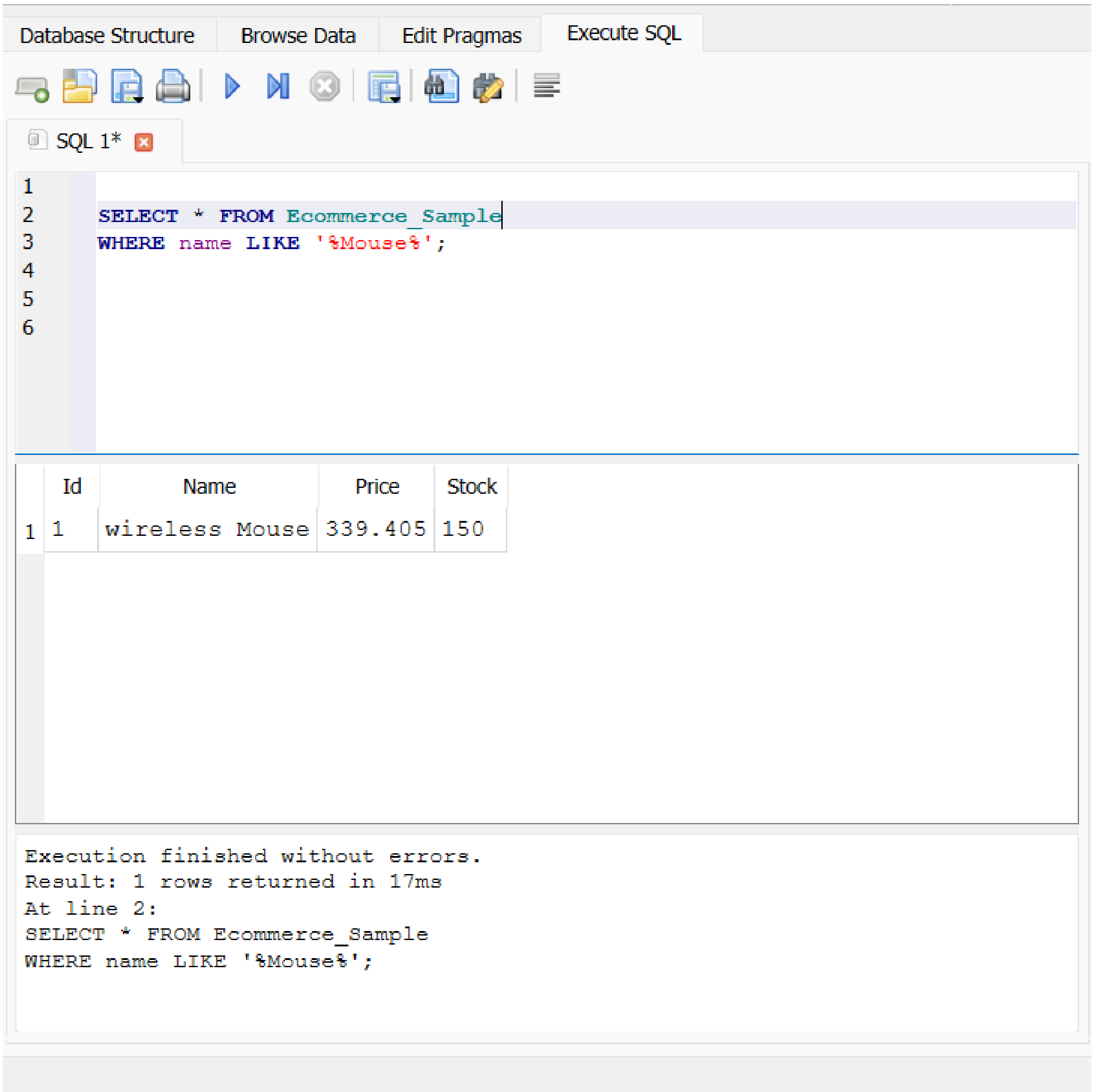


```
UPDATE Ecommerce_Sample
SET price = price * 1.10;


SELECT * FROM Ecommerce_Sample;
```

	Id	Name	Price	Stock
1	1	wireless Mouse	339.405	150
2	2	USB Keyboard	2660.669	200
3	3	27'' Moniter	4658.5	75
4	4	External SSD 1TB	742.698	60
5	5	Gaming Chair	266.18669	30

Queries I Practiced

6. **Search for products containing a specific word**

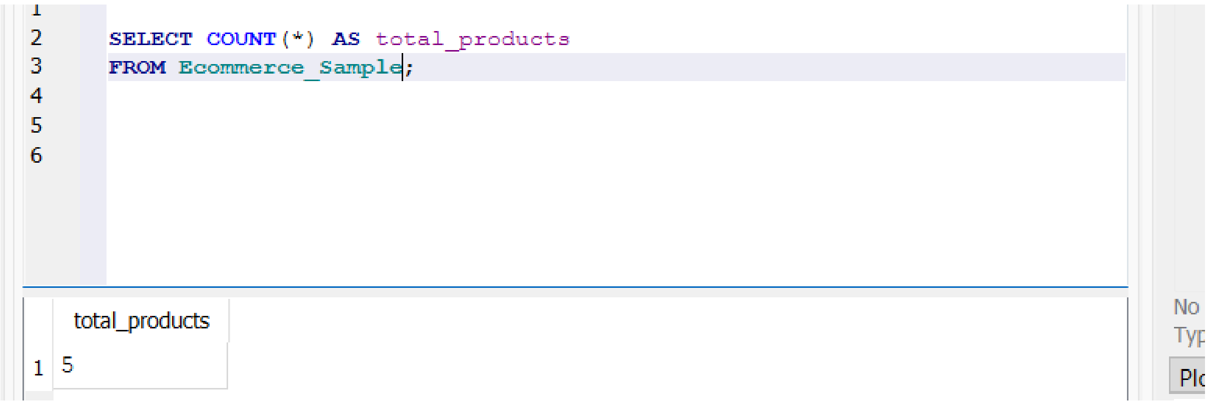


```
SELECT * FROM Ecommerce_Sample
WHERE name LIKE '%Mouse%';
```

	Id	Name	Price	Stock
1	1	wireless Mouse	339.405	150

```
Execution finished without errors.
Result: 1 rows returned in 17ms
At line 2:
SELECT * FROM Ecommerce_Sample
WHERE name LIKE '%Mouse%';
```

7. **Count total number of products**

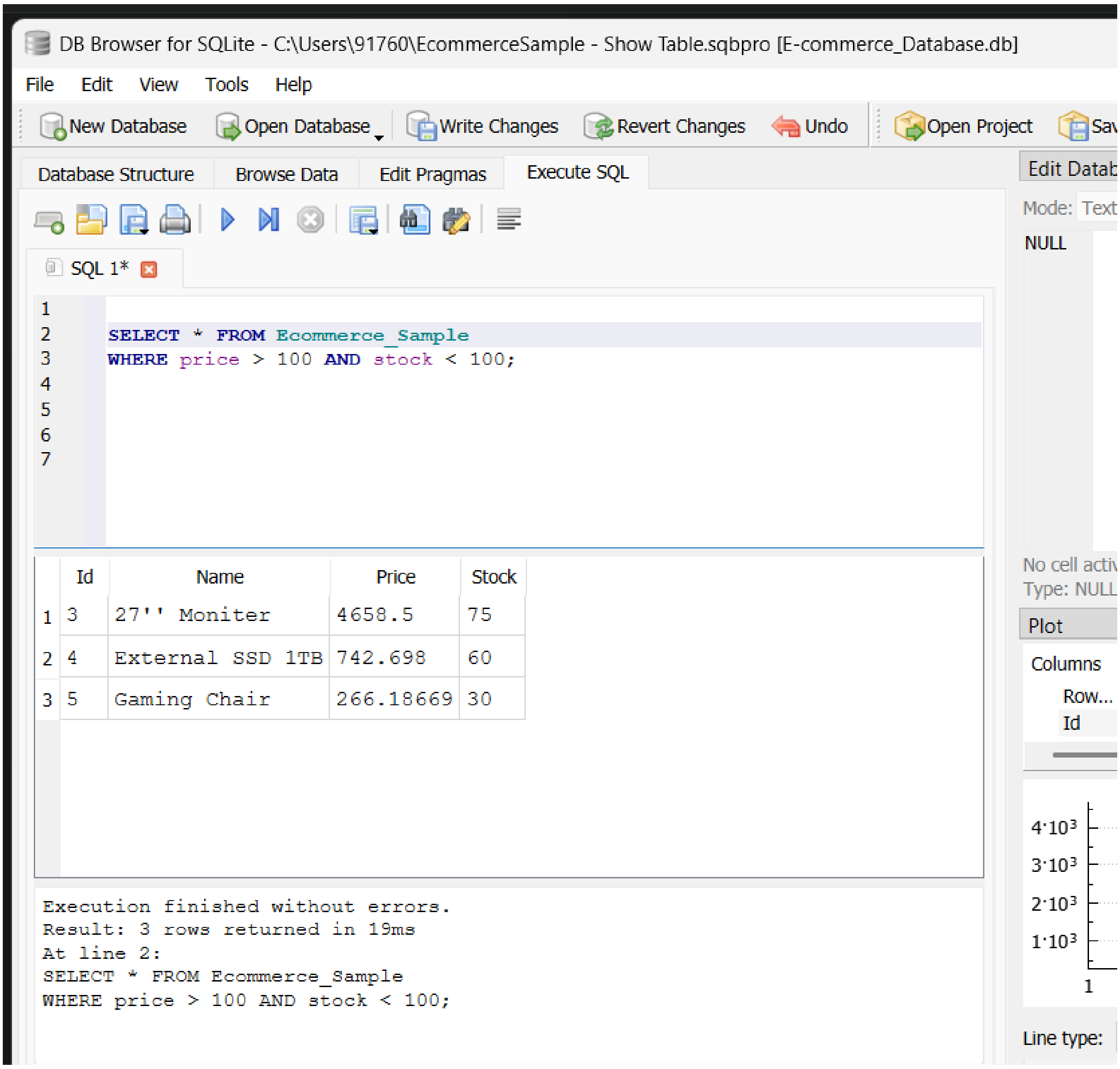


```
SELECT COUNT(*) AS total_products
FROM Ecommerce_Sample;
```

	total_products
1	5

Queries I Practiced

8. **Show products that cost more than 100 and have stock less than 100**

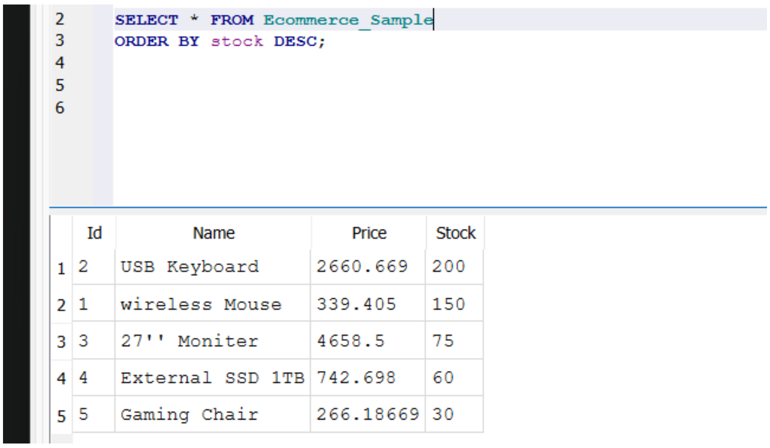


```
SELECT * FROM Ecommerce_Sample
WHERE price > 100 AND stock < 100;
```

	Id	Name	Price	Stock
1	3	27'' Moniter	4658.5	75
2	4	External SSD 1TB	742.698	60
3	5	Gaming Chair	266.18669	30

```
Execution finished without errors.
Result: 3 rows returned in 19ms
At line 2:
SELECT * FROM Ecommerce_Sample
WHERE price > 100 AND stock < 100;
```

9. **Show products sorted by stock from highest to lowest**

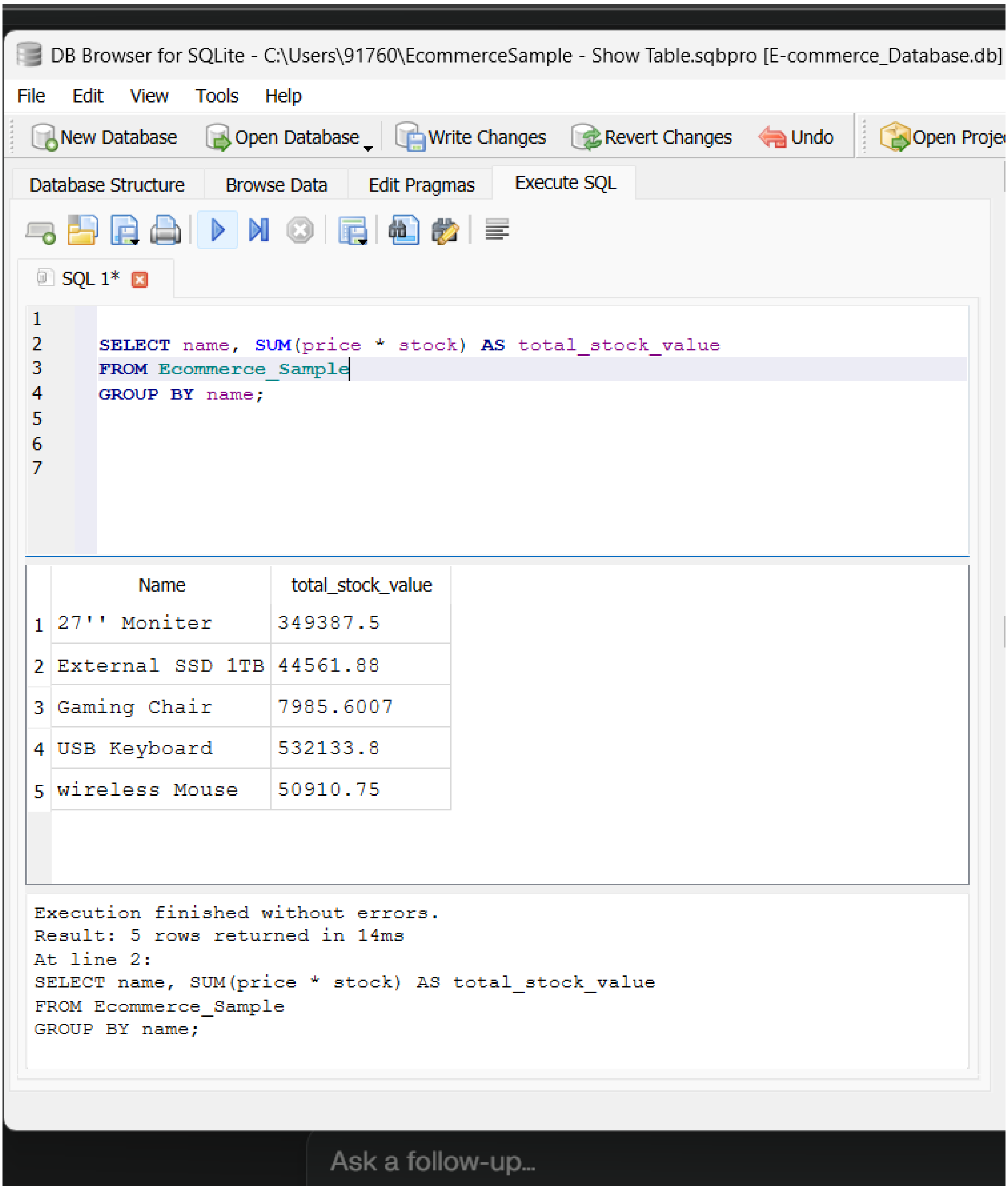


```
SELECT * FROM Ecommerce_Sample
ORDER BY stock DESC;
```

	Id	Name	Price	Stock
1	2	USB Keyboard	2660.669	200
2	1	wireless Mouse	339.405	150
3	3	27'' Moniter	4658.5	75
4	4	External SSD 1TB	742.698	60
5	5	Gaming Chair	266.18669	30

Queries I Practiced

10. Total stock value for each type of product.



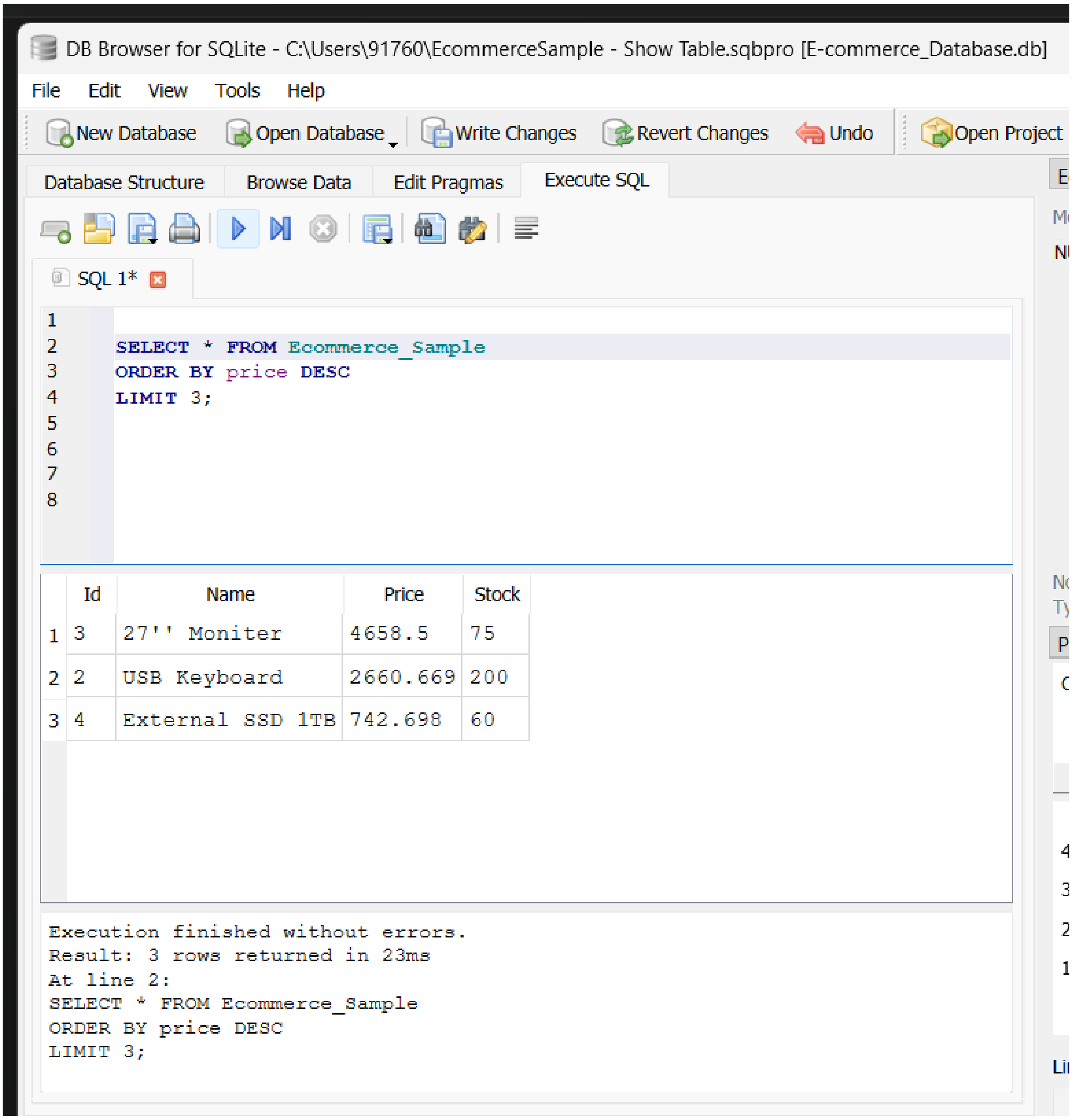
```sql

SELECT name, SUM(price * stock) AS total_stock_value
FROM Ecommerce_Sample
GROUP BY name;
```

	Name	total_stock_value
1	27'' Moniter	349387.5
2	External SSD 1TB	44561.88
3	Gaming Chair	7985.6007
4	USB Keyboard	532133.8
5	wireless Mouse	50910.75

```
Execution finished without errors.
Result: 5 rows returned in 14ms
At line 2:
SELECT name, SUM(price * stock) AS total_stock_value
FROM Ecommerce_Sample
GROUP BY name;
```

Queries I Practiced

11. Top 3 most expensive products.



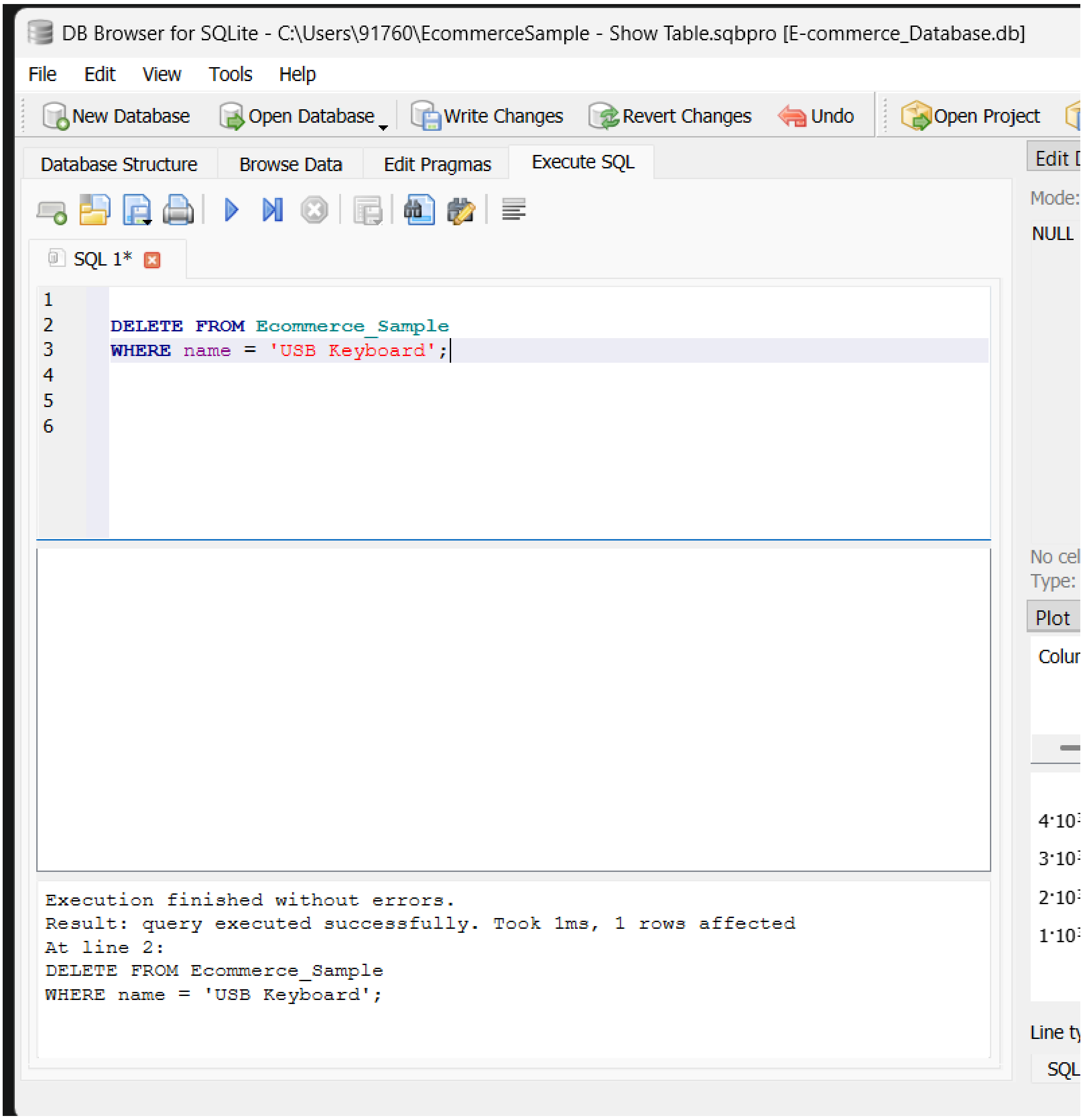
```sql

SELECT * FROM Ecommerce_Sample
ORDER BY price DESC
LIMIT 3;
```

	Id	Name	Price	Stock
1	3	27'' Moniter	4658.5	75
2	2	USB Keyboard	2660.669	200
3	4	External SSD 1TB	742.698	60

```
Execution finished without errors.
Result: 3 rows returned in 23ms
At line 2:
SELECT * FROM Ecommerce_Sample
ORDER BY price DESC
LIMIT 3;
```

Queries I Practiced

12. Delete a Product (use with caution!)

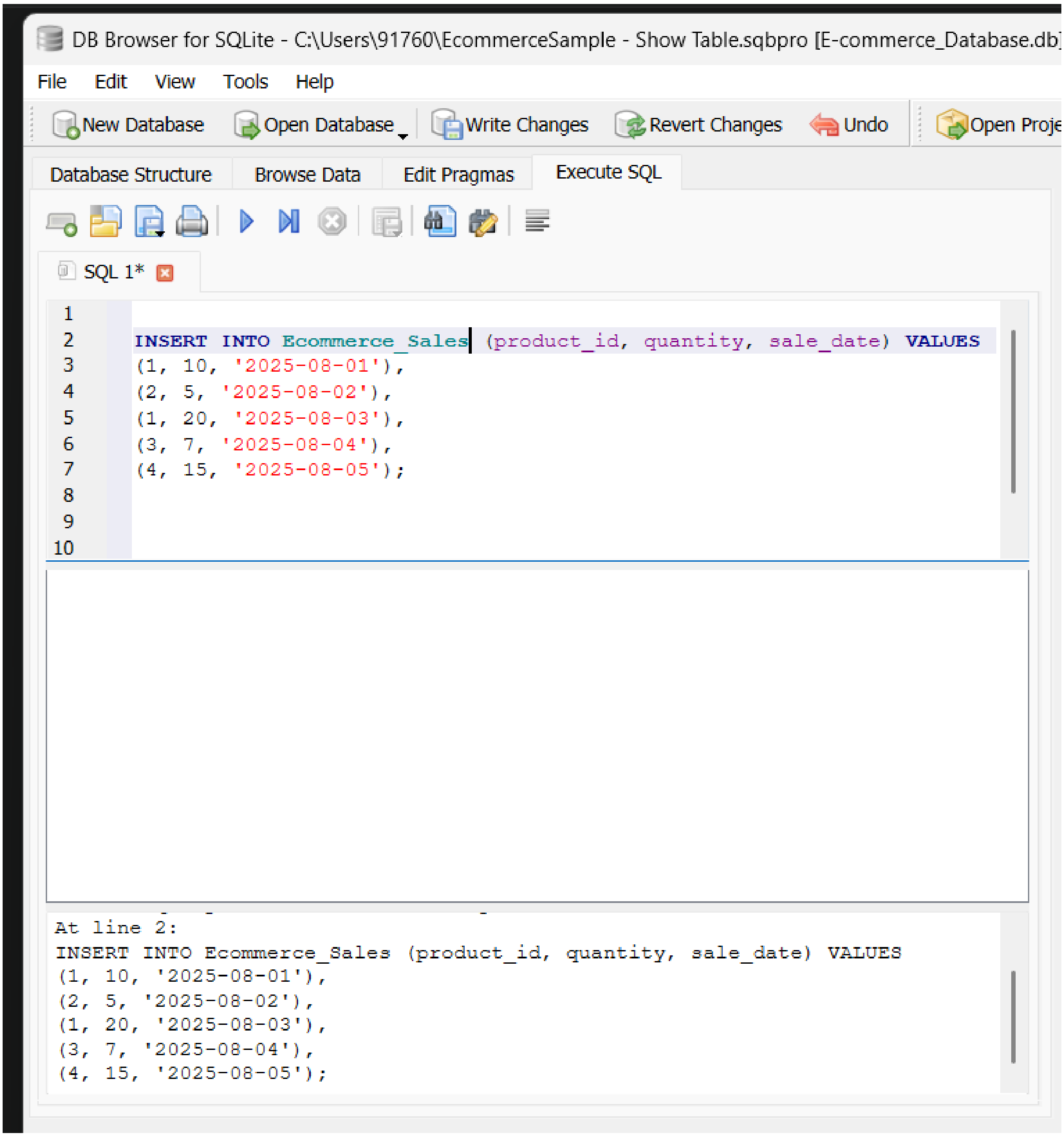


```
DELETE FROM Ecommerce_Sample
WHERE name = 'USB Keyboard';
```

```
Execution finished without errors.
Result: query executed successfully. Took 1ms, 1 rows affected
At line 2:
DELETE FROM Ecommerce_Sample
WHERE name = 'USB Keyboard';
```

Queries I Practiced

13. Insert Sample Data into sales



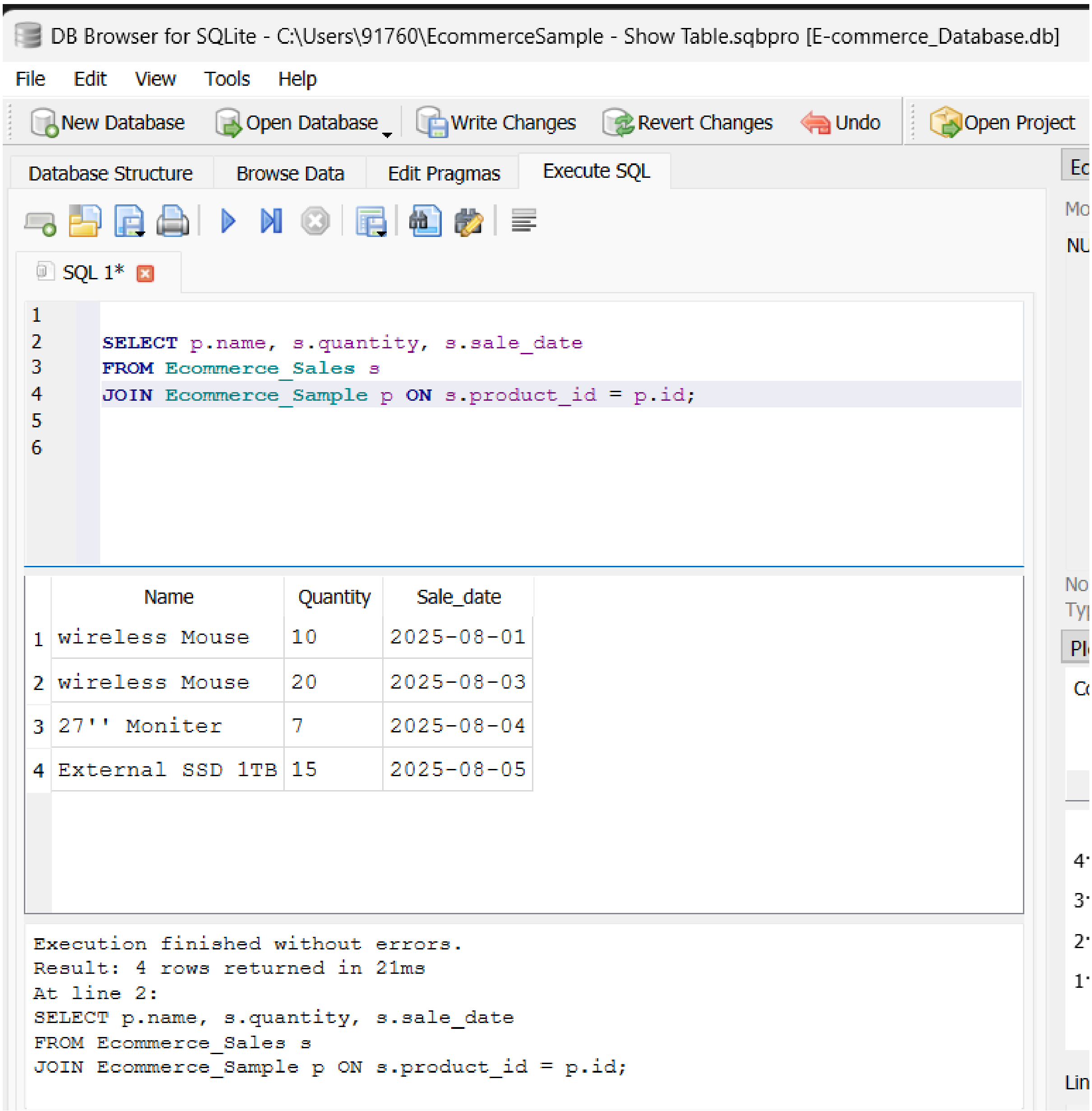
DB Browser for SQLite - C:\Users\91760\EcommerceSample - Show Table.sqbpro [E-commerce_Database.db]
File Edit View Tools Help
New Database
Open Database
Write Changes
Revert Changes
Undo
Open Proje
Database Structure
Browse Data
Edit Pragmas
Execute SQL
SQL 1*

```sql

INSERT INTO Ecommerce_Sales (product_id, quantity, sale_date) VALUES
(1, 10, '2025-08-01'),
(2, 5, '2025-08-02'),
(1, 20, '2025-08-03'),
(3, 7, '2025-08-04'),
(4, 15, '2025-08-05');
```

```
At line 2:
INSERT INTO Ecommerce_Sales (product_id, quantity, sale_date) VALUES
(1, 10, '2025-08-01'),
(2, 5, '2025-08-02'),
(1, 20, '2025-08-03'),
(3, 7, '2025-08-04'),
(4, 15, '2025-08-05');
```

Queries I Practiced

14. Join sales and products to Analyze Sales Data



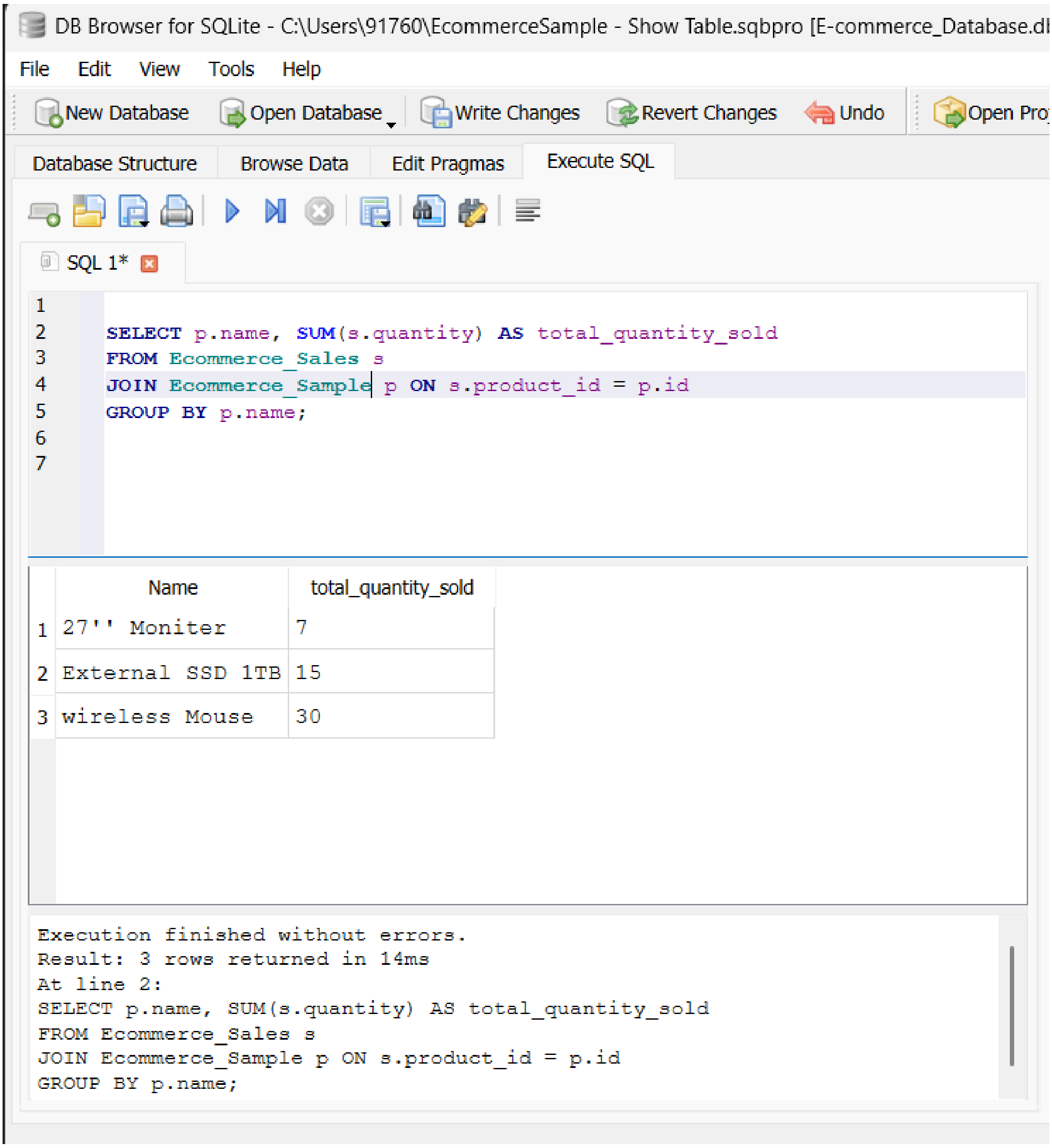
```sql

SELECT p.name, s.quantity, s.sale_date
FROM Ecommerce_Sales s
JOIN Ecommerce_Sample p ON s.product_id = p.id;


```

	Name	Quantity	Sale_date
1	wireless Mouse	10	2025-08-01
2	wireless Mouse	20	2025-08-03
3	27'' Moniter	7	2025-08-04
4	External SSD 1TB	15	2025-08-05

```
Execution finished without errors.
Result: 4 rows returned in 21ms
At line 2:
SELECT p.name, s.quantity, s.sale_date
FROM Ecommerce_Sales s
JOIN Ecommerce_Sample p ON s.product_id = p.id;
```

Queries I Practiced

15. **See total quantity sold for each product**



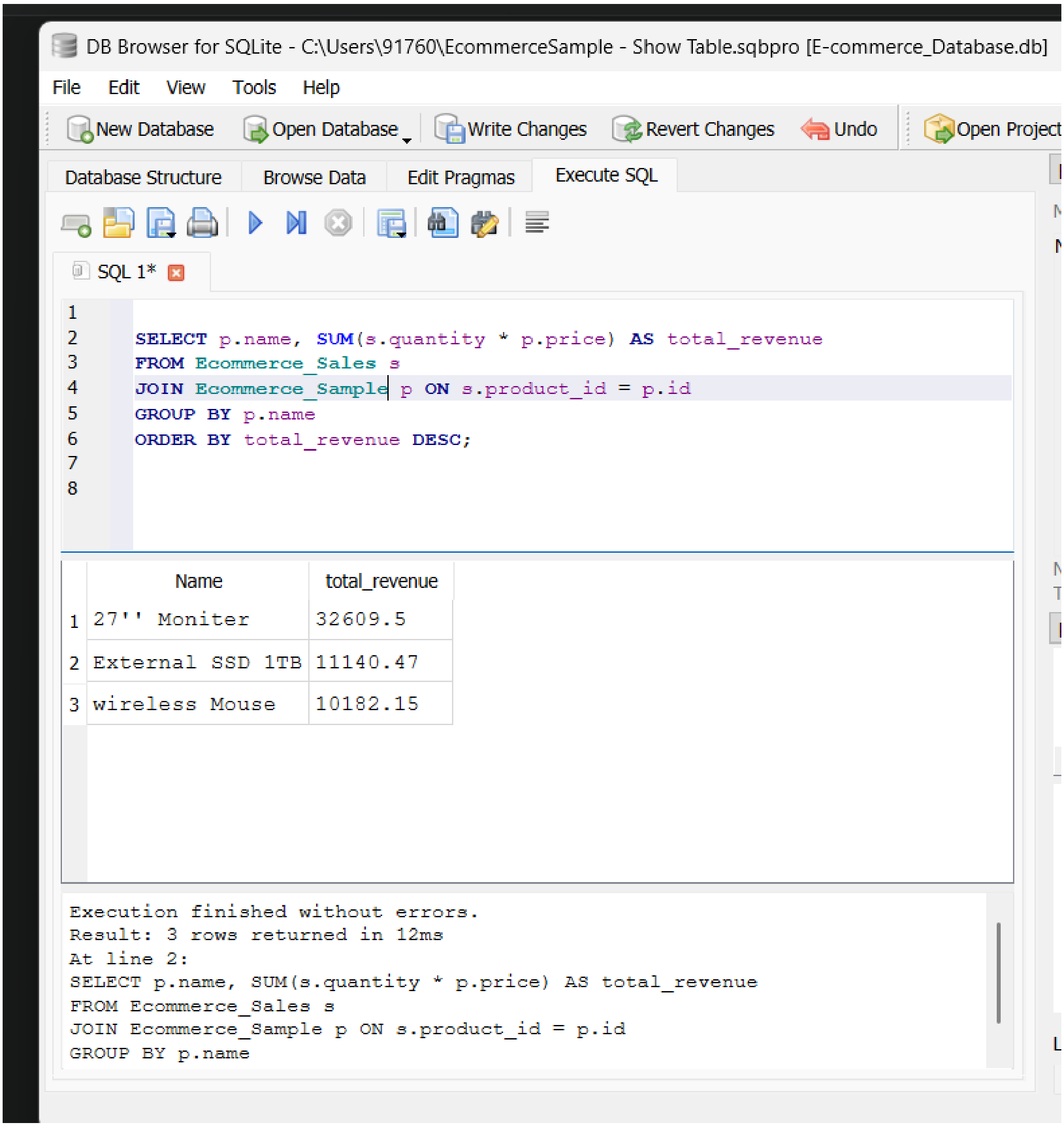
```

SELECT p.name, SUM(s.quantity) AS total_quantity_sold
FROM Ecommerce_Sales s
JOIN Ecommerce_Sample p ON s.product_id = p.id
GROUP BY p.name;


```

	Name	total_quantity_sold
1	27'' Moniter	7
2	External SSD 1TB	15
3	wireless Mouse	30

```
Execution finished without errors.
Result: 3 rows returned in 14ms
At line 2:
SELECT p.name, SUM(s.quantity) AS total_quantity_sold
FROM Ecommerce_Sales s
JOIN Ecommerce_Sample p ON s.product_id = p.id
GROUP BY p.name;
```

Queries I Practiced

16. **Calculate total revenue for each product**



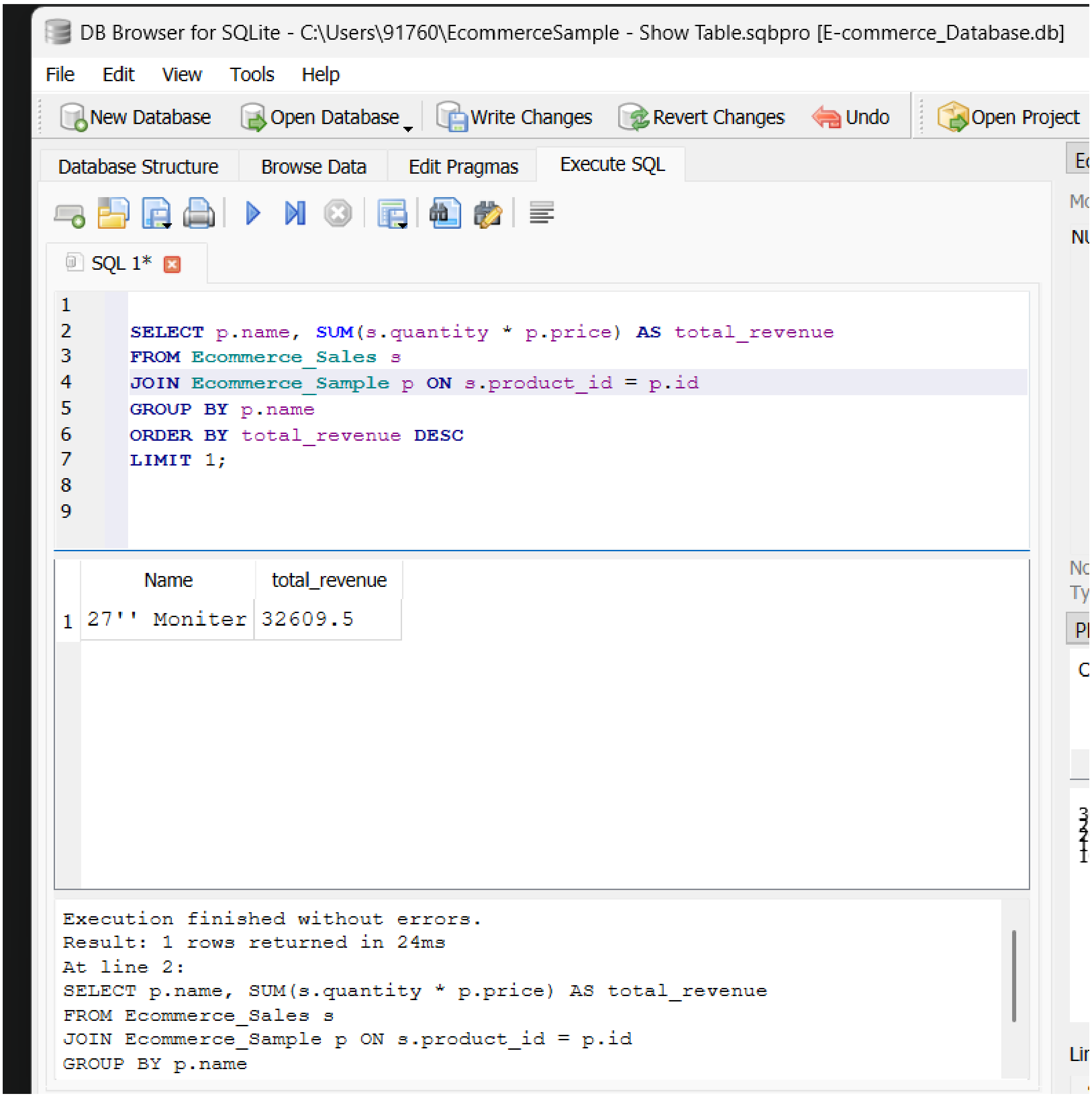
```

SELECT p.name, SUM(s.quantity * p.price) AS total_revenue
FROM Ecommerce_Sales s
JOIN Ecommerce_Sample p ON s.product_id = p.id
GROUP BY p.name
ORDER BY total_revenue DESC;
```

	Name	total_revenue
1	27'' Moniter	32609.5
2	External SSD 1TB	11140.47
3	wireless Mouse	10182.15

```
Execution finished without errors.
Result: 3 rows returned in 12ms
At line 2:
SELECT p.name, SUM(s.quantity * p.price) AS total_revenue
FROM Ecommerce_Sales s
JOIN Ecommerce_Sample p ON s.product_id = p.id
GROUP BY p.name
```

Queries I Practiced

17. Find the best-selling product (by total revenue)



```sql

SELECT p.name, SUM(s.quantity * p.price) AS total_revenue
FROM Ecommerce_Sales s
JOIN Ecommerce_Sample p ON s.product_id = p.id
GROUP BY p.name
ORDER BY total_revenue DESC
LIMIT 1;
```

	Name	total_revenue
1	27'' Moniter	32609.5

```
Execution finished without errors.
Result: 1 rows returned in 24ms
At line 2:
SELECT p.name, SUM(s.quantity * p.price) AS total_revenue
FROM Ecommerce_Sales s
JOIN Ecommerce_Sample p ON s.product_id = p.id
GROUP BY p.name
```