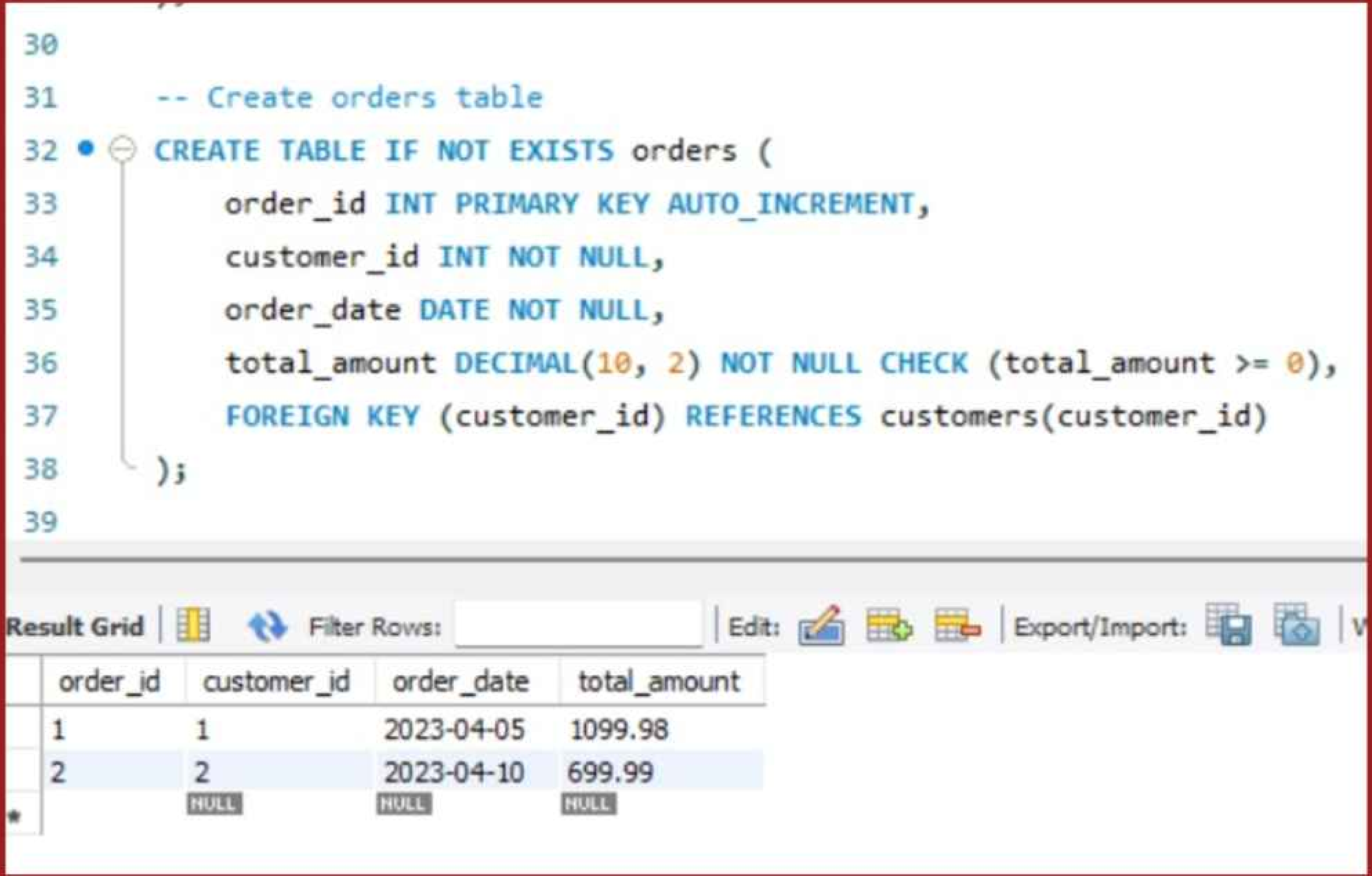
E-COMMERCE DATBASE QUERIES AND OUT IN MYSQL

SMARTPHONE OF THE YEAR

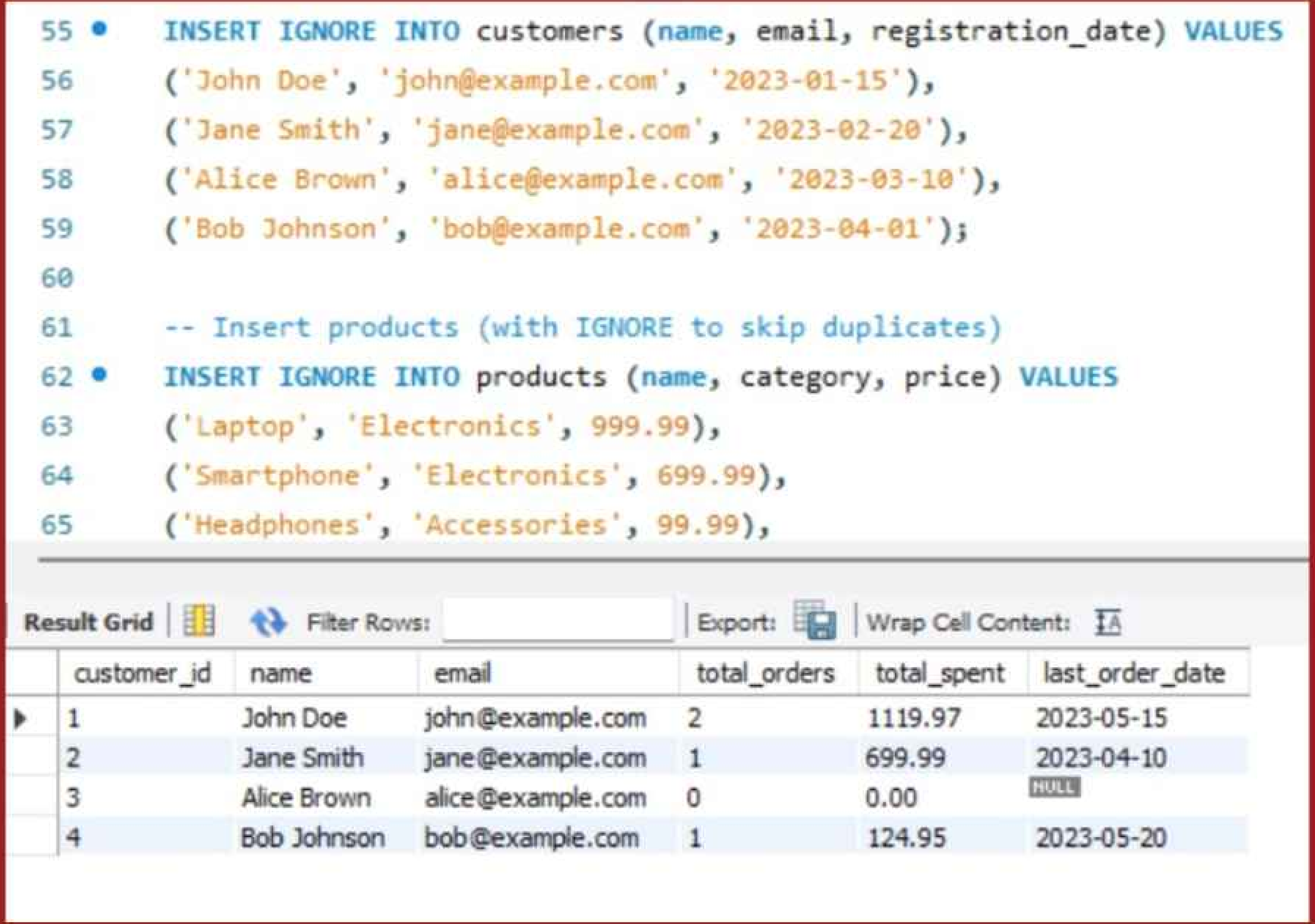
$ 199

- CREDIT CARD
- FREE SHIPPING ALL OVER AMERICA
- 100% RELIABLE STORE
- FREE RETURN



```sql
-- Create orders table
CREATE TABLE IF NOT EXISTS orders (
    order_id INT PRIMARY KEY AUTO_INCREMENT,
    customer_id INT NOT NULL,
    order_date DATE NOT NULL,
    total_amount DECIMAL(10, 2) NOT NULL CHECK (total_amount >= 0),
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id)
);
```

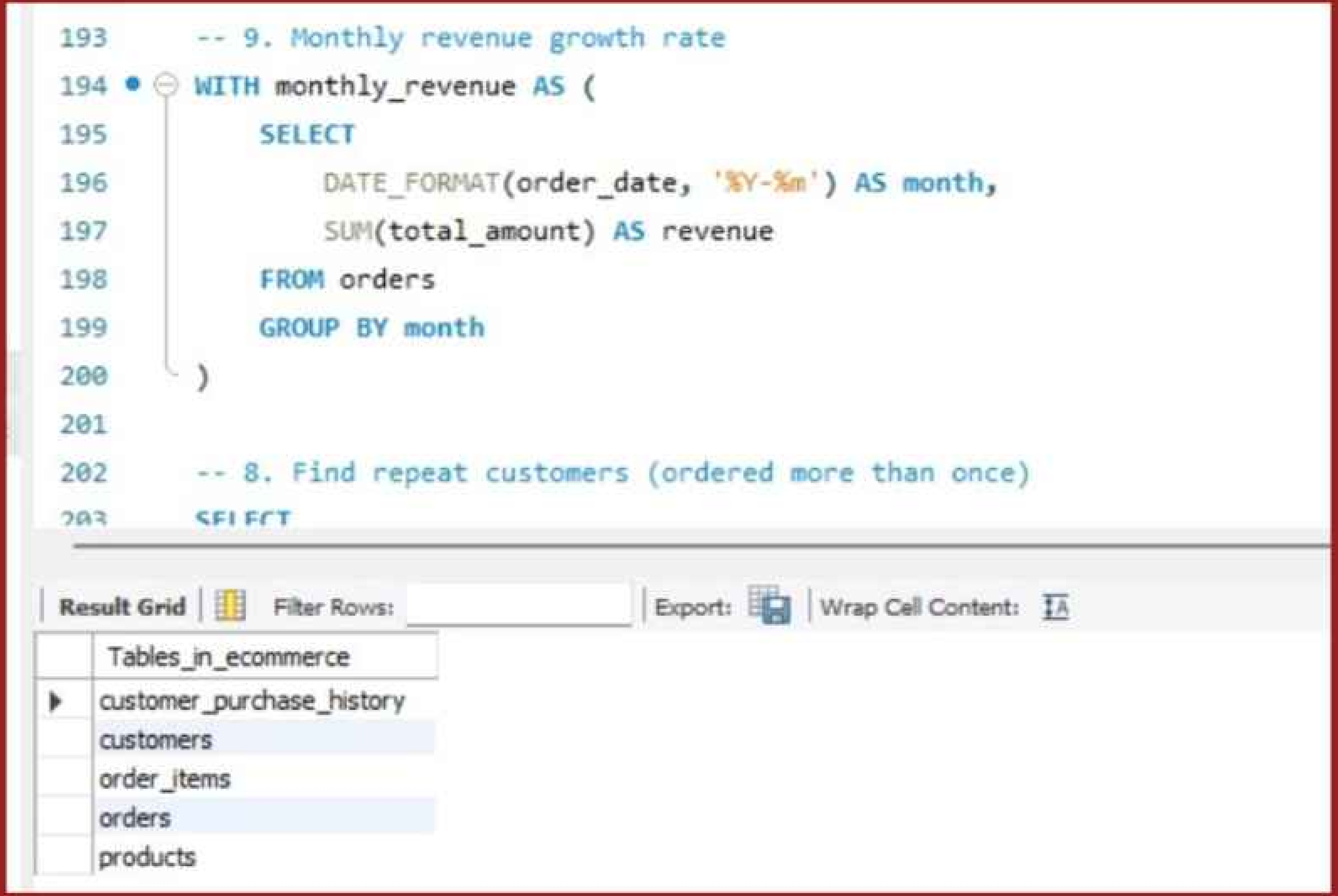
order_id	customer_id	order_date	total_amount
1	1	2023-04-05	1099.98
2	2	2023-04-10	699.99
*	NULL	NULL	NULL


```sql
INSERT IGNORE INTO customers (name, email, registration_date) VALUES
('John Doe', 'john@example.com', '2023-01-15'),
('Jane Smith', 'jane@example.com', '2023-02-20'),
('Alice Brown', 'alice@example.com', '2023-03-10'),
('Bob Johnson', 'bob@example.com', '2023-04-01');

-- Insert products (with IGNORE to skip duplicates)
INSERT IGNORE INTO products (name, category, price) VALUES
('Laptop', 'Electronics', 999.99),
('Smartphone', 'Electronics', 699.99),
('Headphones', 'Accessories', 99.99),
```

customer_id	name	email	total_orders	total_spent	last_order_date
1	John Doe	john@example.com	2	1119.97	2023-05-15
2	Jane Smith	jane@example.com	1	699.99	2023-04-10
3	Alice Brown	alice@example.com	0	0.00	NULL
4	Bob Johnson	bob@example.com	1	124.95	2023-05-20

MONTHLY REVEMNUE GROWTH RATE

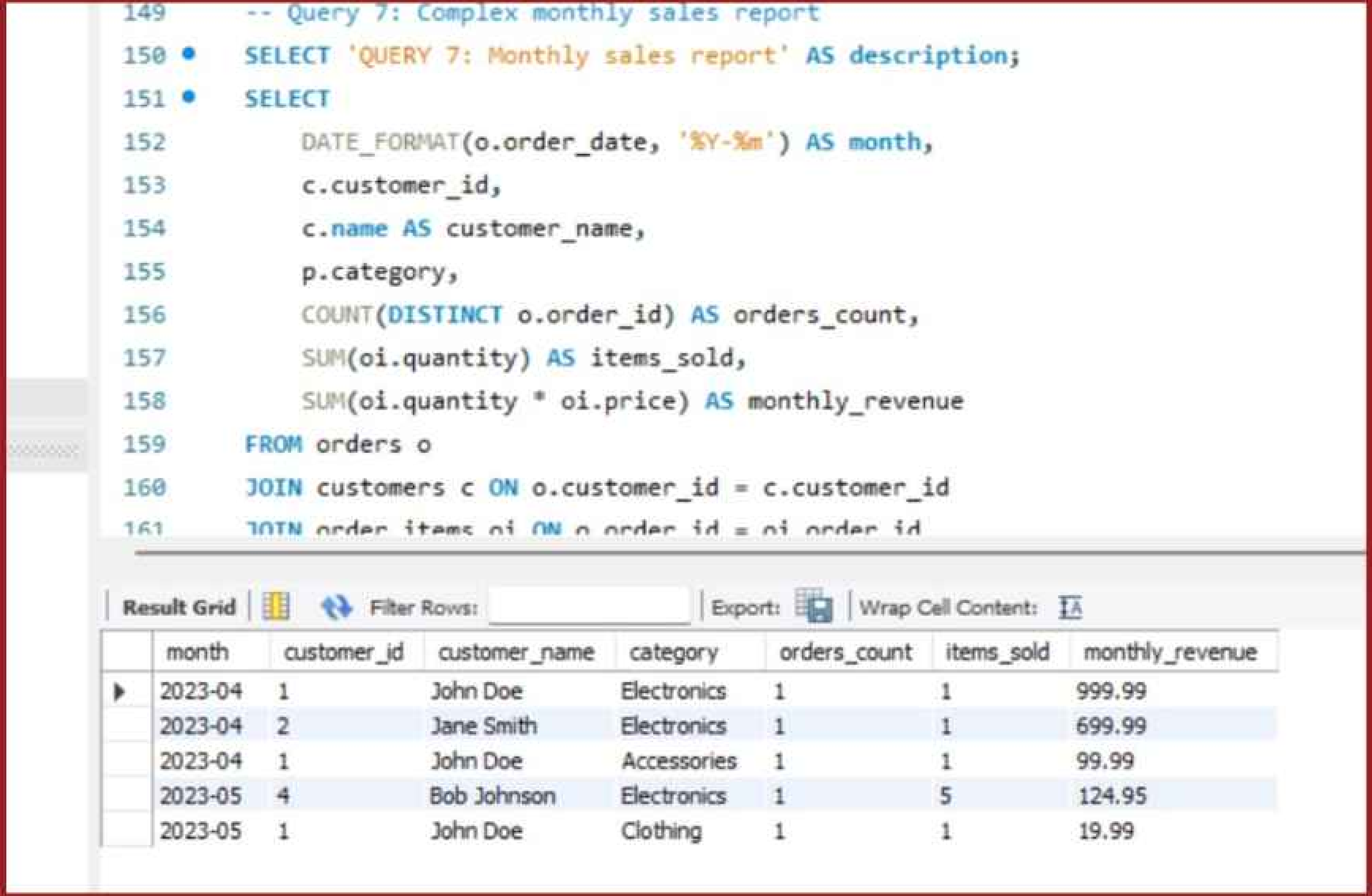


```
-- 9. Monthly revenue growth rate
WITH monthly_revenue AS (
    SELECT
        DATE_FORMAT(order_date, '%Y-%m') AS month,
        SUM(total_amount) AS revenue
    FROM orders
    GROUP BY month
)

-- 8. Find repeat customers (ordered more than once)
SELECT
```

Tables_in_ecommerce
customer_purchase_history
customers
order_items
orders
products

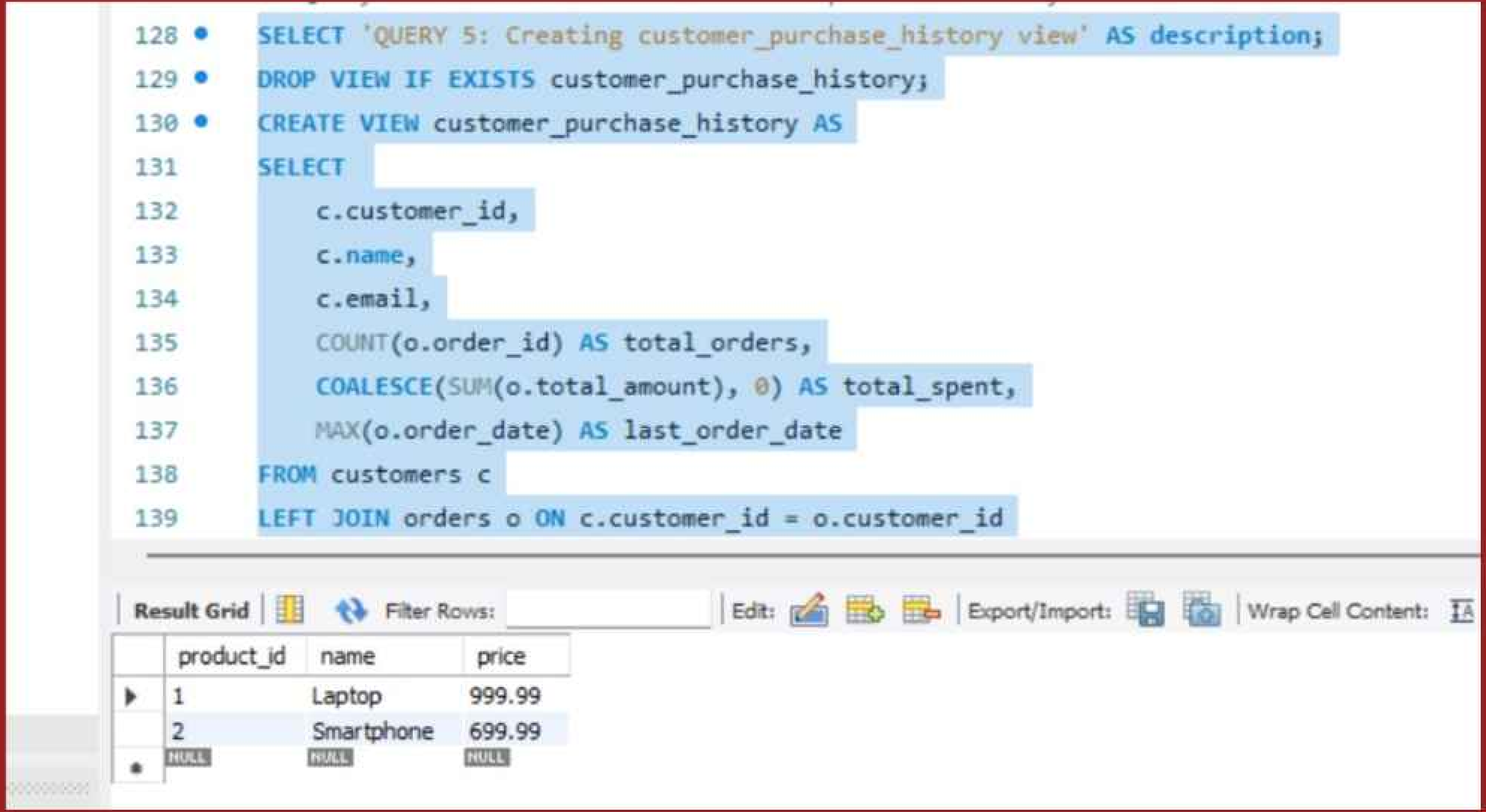
COMPLEX MONHLY SALES REPORT



```sql
-- Query 7: Complex monthly sales report
SELECT 'QUERY 7: Monthly sales report' AS description;
SELECT
    DATE_FORMAT(o.order_date, '%Y-%m') AS month,
    c.customer_id,
    c.name AS customer_name,
    p.category,
    COUNT(DISTINCT o.order_id) AS orders_count,
    SUM(oi.quantity) AS items_sold,
    SUM(oi.quantity * oi.price) AS monthly_revenue
FROM orders o
JOIN customers c ON o.customer_id = c.customer_id
JOIN order_items oi ON o.order_id = oi.order_id
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

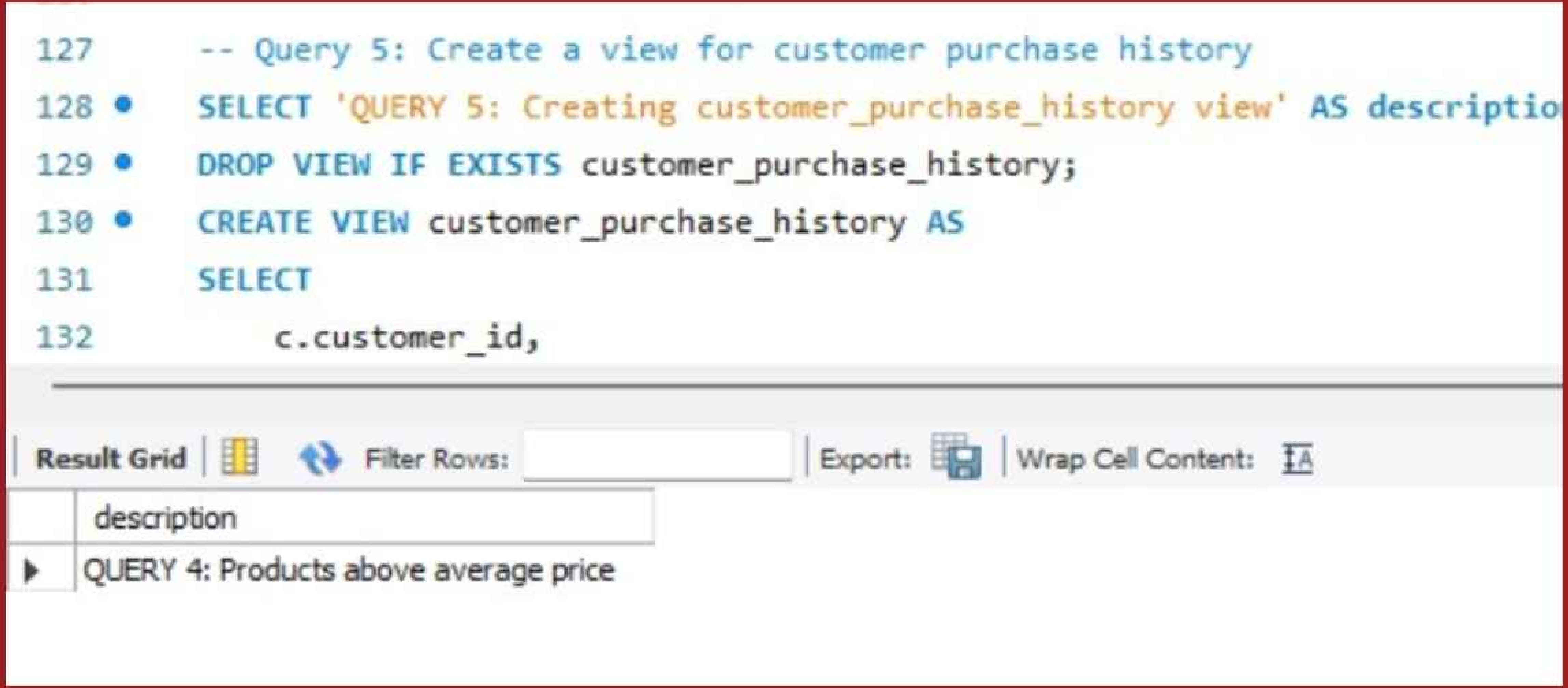
month	customer_id	customer_name	category	orders_count	items_sold	monthly_revenue
2023-04	1	John Doe	Electronics	1	1	999.99
2023-04	2	Jane Smith	Electronics	1	1	699.99
2023-04	1	John Doe	Accessories	1	1	99.99
2023-05	4	Bob Johnson	Electronics	1	5	124.95
2023-05	1	John Doe	Clothing	1	1	19.99



```sql
SELECT 'QUERY 5: Creating customer_purchase_history view' AS description;
DROP VIEW IF EXISTS customer_purchase_history;
CREATE VIEW customer_purchase_history AS
SELECT
    c.customer_id,
    c.name,
    c.email,
    COUNT(o.order_id) AS total_orders,
    COALESCE(SUM(o.total_amount), 0) AS total_spent,
    MAX(o.order_date) AS last_order_date
FROM customers c
LEFT JOIN orders o ON c.customer_id = o.customer_id
```

product_id	name	price
1	Laptop	999.99
2	Smartphone	699.99
NULL	NULL	NULL

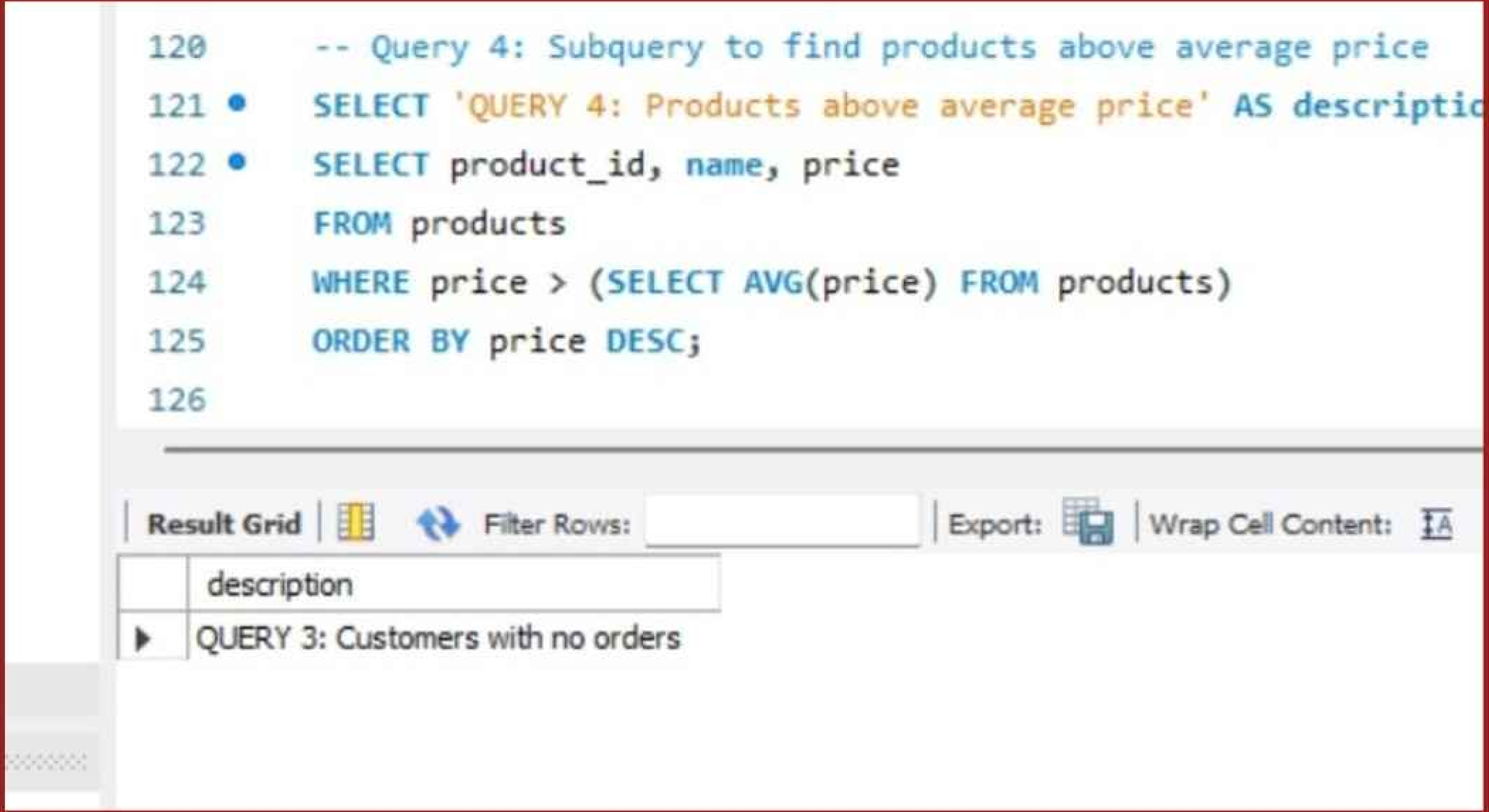
CREATE A VIEW FOR CUSTOMER PURCHASE HISTORY



```
-- Query 5: Create a view for customer purchase history
SELECT 'QUERY 5: Creating customer_purchase_history view' AS descriptio
DROP VIEW IF EXISTS customer_purchase_history;
CREATE VIEW customer_purchase_history AS
SELECT
    c.customer_id,
```

```
description
QUERY 4: Products above average price
```

SUBQUERY TO FIND PRODUCTS ABOVE AVERAGE PRICE



```sql
-- Query 4: Subquery to find products above average price
SELECT 'QUERY 4: Products above average price' AS descriptic
SELECT product_id, name, price
FROM products
WHERE price > (SELECT AVG(price) FROM products)
ORDER BY price DESC;
```

	description
▶	QUERY 3: Customers with no orders

SUMMARY OF THIS QUERY:

- TO SOLVE THIS PROBLEM USE MYSQL WORKBENCH.
-
- USE SELECT TO FIND MONTHLY SALES.
-
- USE COUNT ,MAX TO FIND PRICE.
-
- USE WHERE TO FIND AVERAGE PRICE.
-
- USE JOIN TO FIND SALES REPORT.

THANK YOU