PIZZA SALES ANALYSIS USING SQL

SANTAJI VHANKADE

PROJECT OBJECTIVE

- To perform a comprehensive analysis of pizza sales data using SQL.
- To uncover key business insights such as sales trends, top-performing products, and customer preferences.
- To practice SQL querying skills across basic, intermediate, and advanced levels.

DATASET OVERVIEW

Data Overview:

- 4 interconnected tables used:

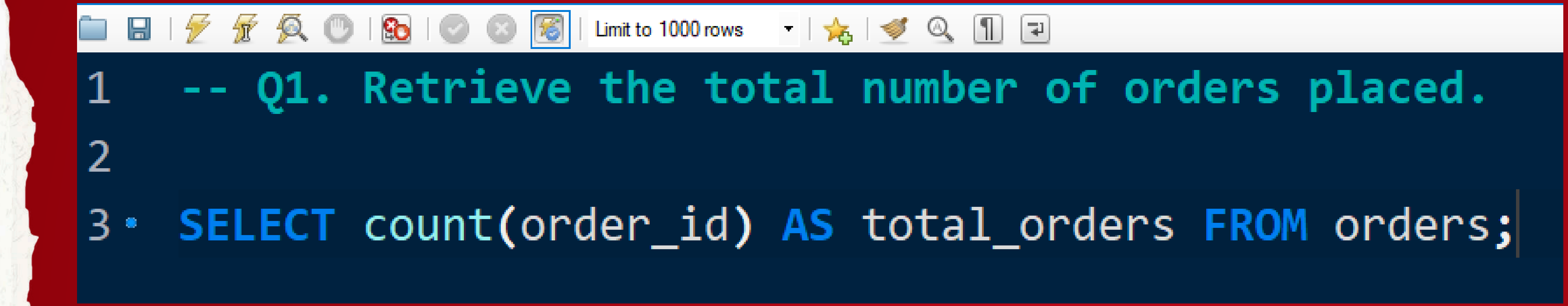
1) pizzas (pizza_id, pizza_type_id, size, price)

2) pizza_types (pizza_type_id, name, category, ingredients)

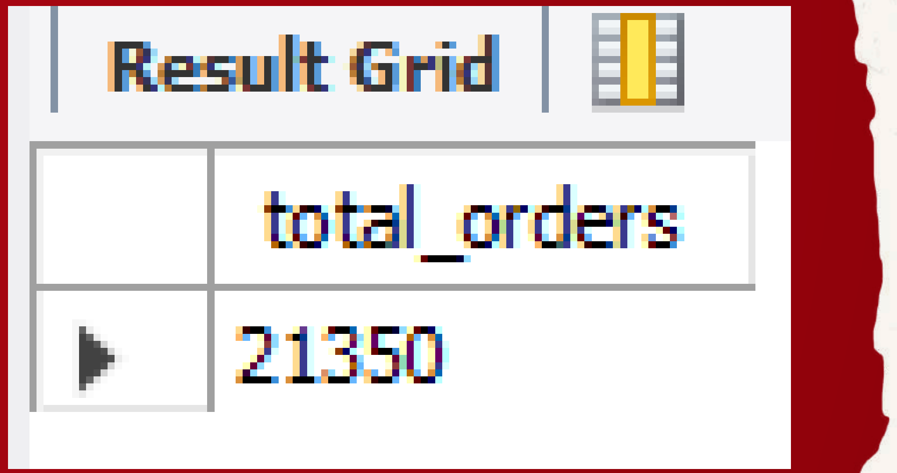
3) orders (order_id, order_date, order_time)

4) order_details (order_details_id, order_id, pizza_id, quantity)

Q1. RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

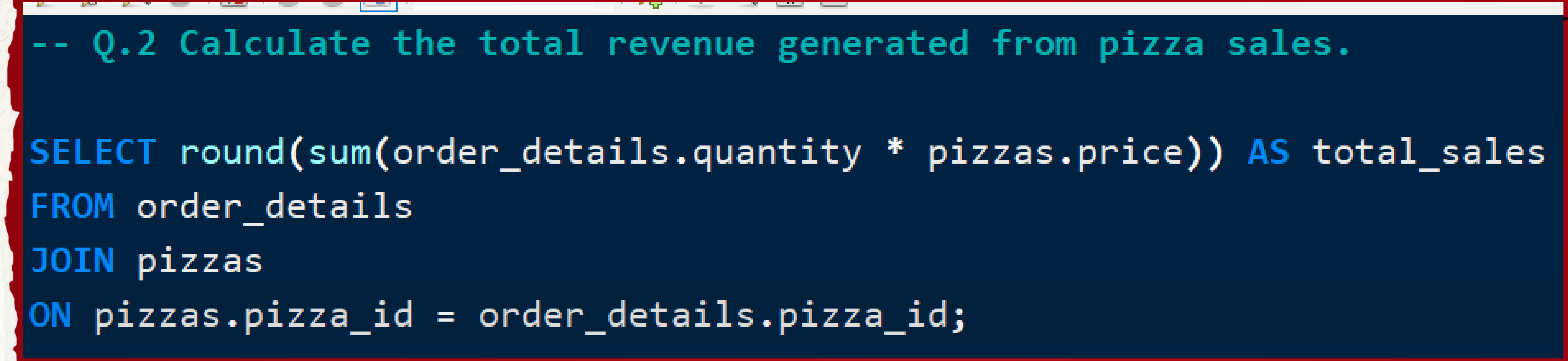


```sql
-- Q1. Retrieve the total number of orders placed.

SELECT count(order_id) AS total_orders FROM orders;
```

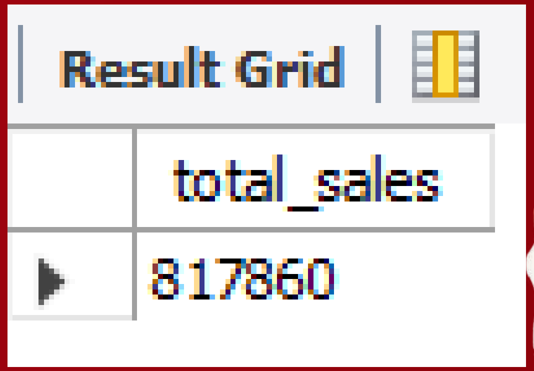


total_orders
21350

Q2. CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

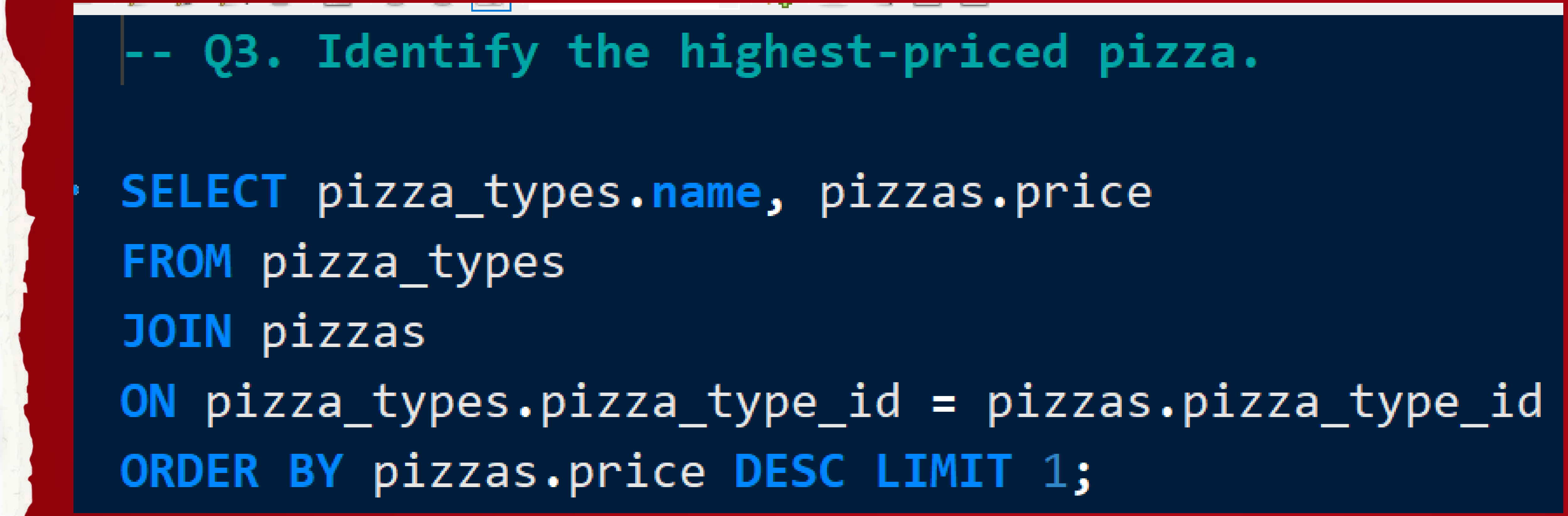


```
-- Q.2 Calculate the total revenue generated from pizza sales.

SELECT round(sum(order_details.quantity * pizzas.price)) AS total_sales
FROM order_details
JOIN pizzas
ON pizzas.pizza_id = order_details.pizza_id;
```

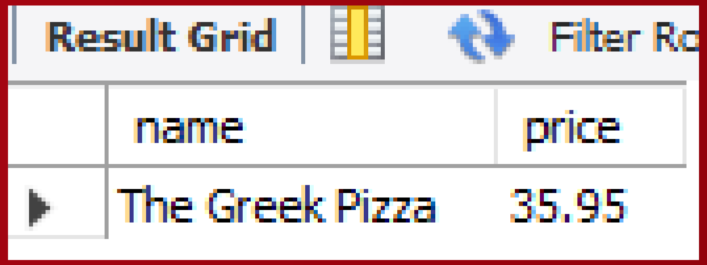


total_sales
817860

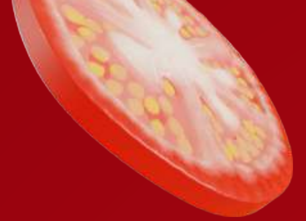
Q3. IDENTIFY THE HIGHEST-PRICED PIZZA.



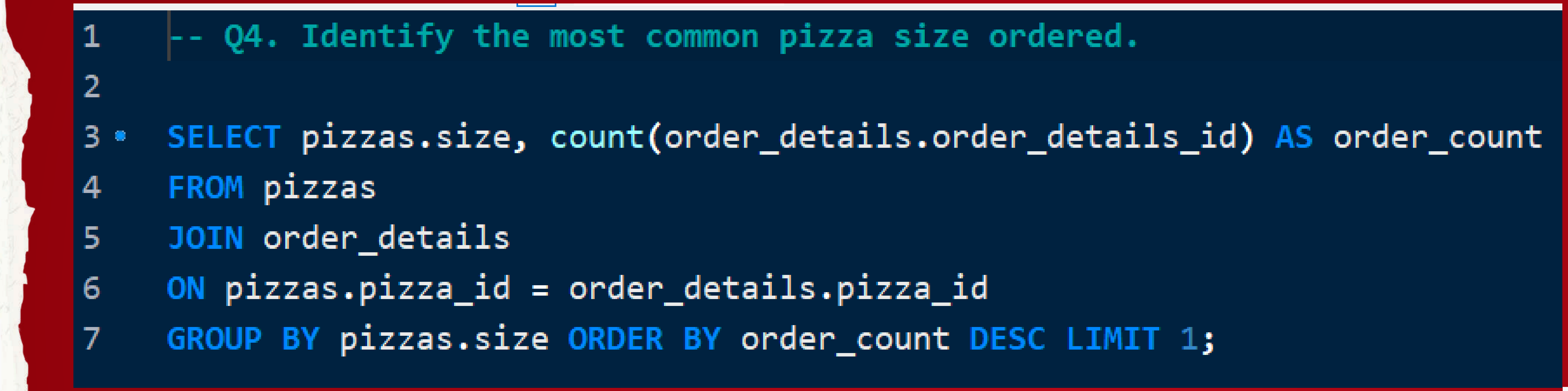
```
-- Q3. Identify the highest-priced pizza.

SELECT pizza_types.name, pizzas.price
FROM pizza_types
JOIN pizzas
ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC LIMIT 1;
```



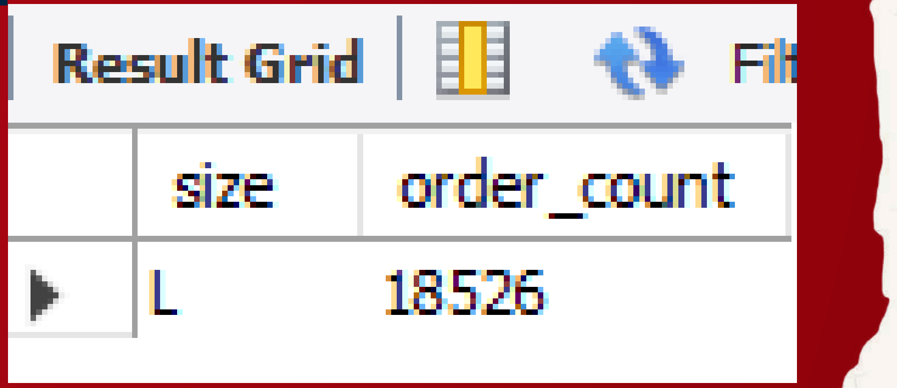
name	price
The Greek Pizza	35.95



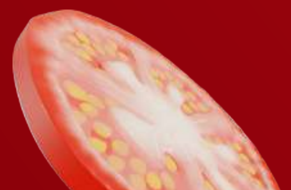
Q4. IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.



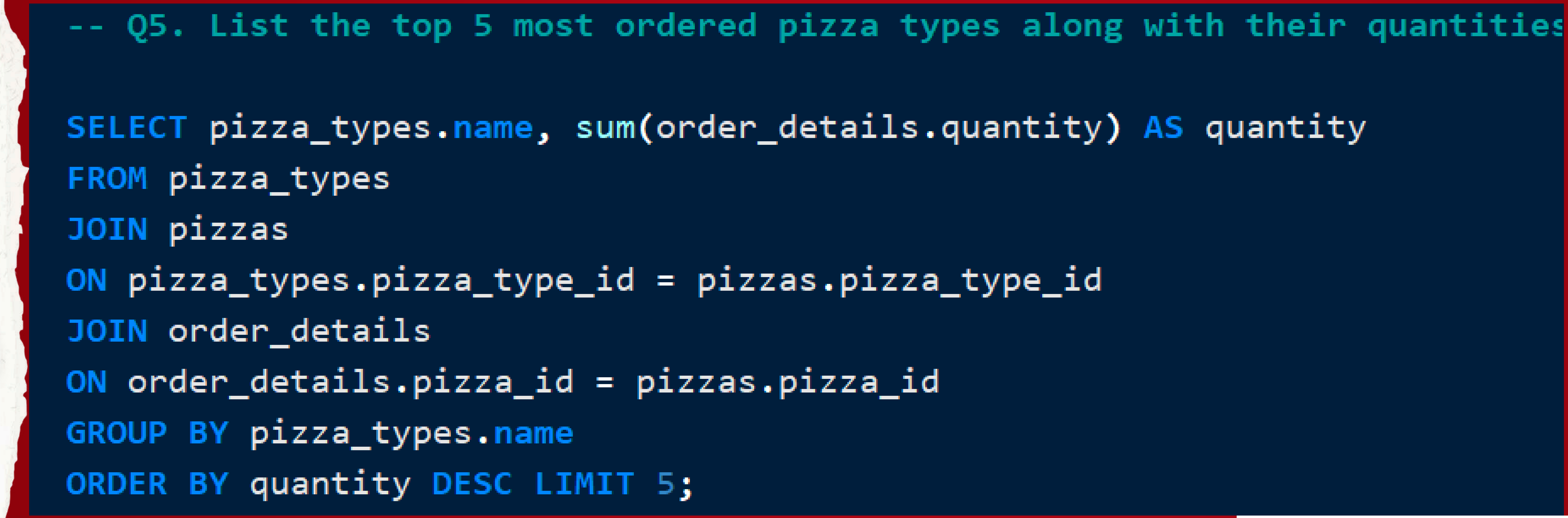
```
-- Q4. Identify the most common pizza size ordered.

SELECT pizzas.size, count(order_details.order_details_id) AS order_count
FROM pizzas
JOIN order_details
ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size ORDER BY order_count DESC LIMIT 1;
```



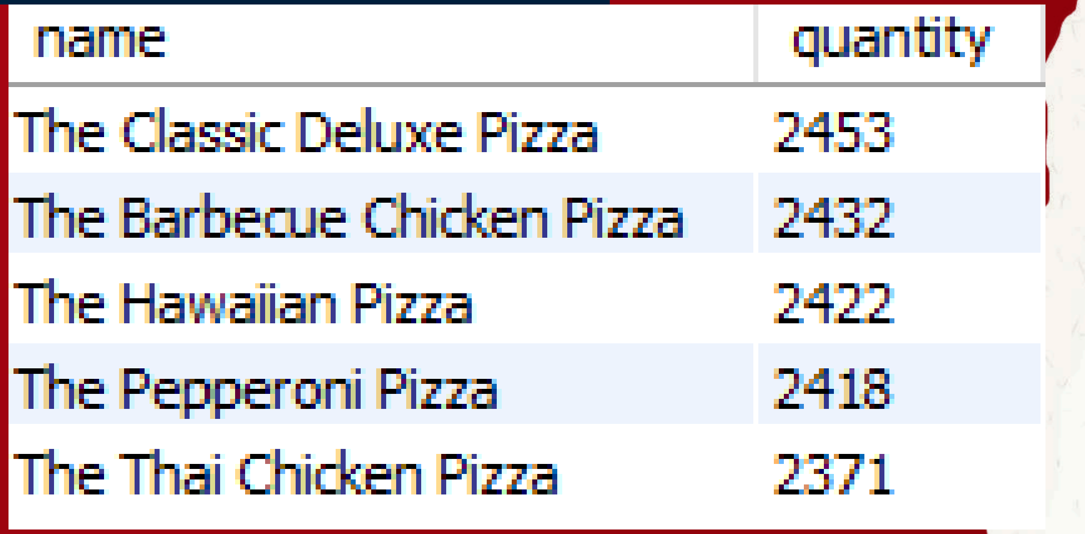
size	order_count
L	18526



Q5. LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.

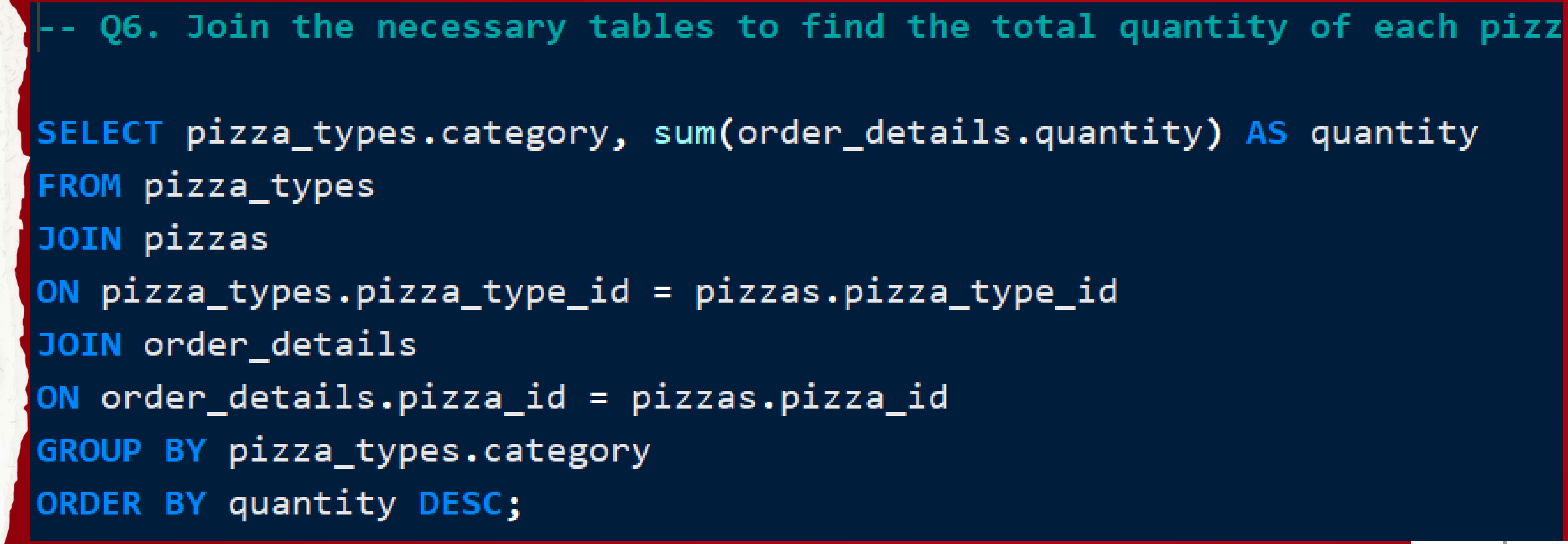


```
-- Q5. List the top 5 most ordered pizza types along with their quantities

SELECT pizza_types.name, sum(order_details.quantity) AS quantity
FROM pizza_types
JOIN pizzas
ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN order_details
ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC LIMIT 5;
```

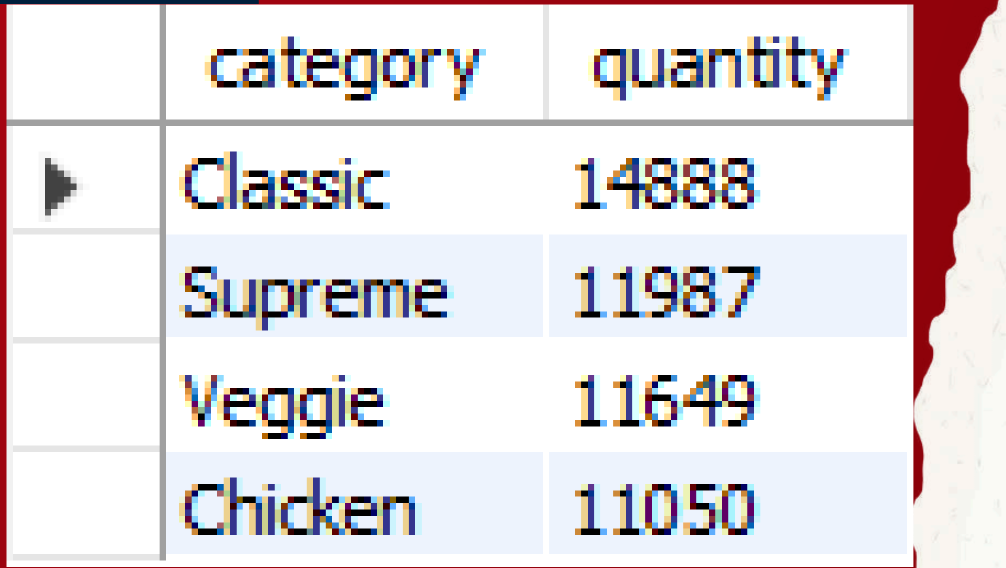


name	quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

Q6. JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.

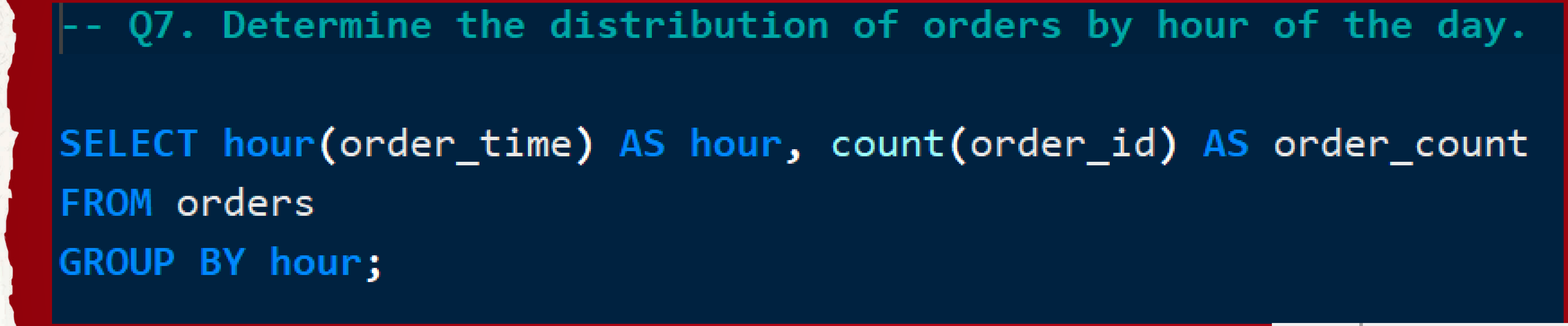


```
-- Q6. Join the necessary tables to find the total quantity of each pizz

SELECT pizza_types.category, sum(order_details.quantity) AS quantity
FROM pizza_types
JOIN pizzas
ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN order_details
ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY quantity DESC;
```

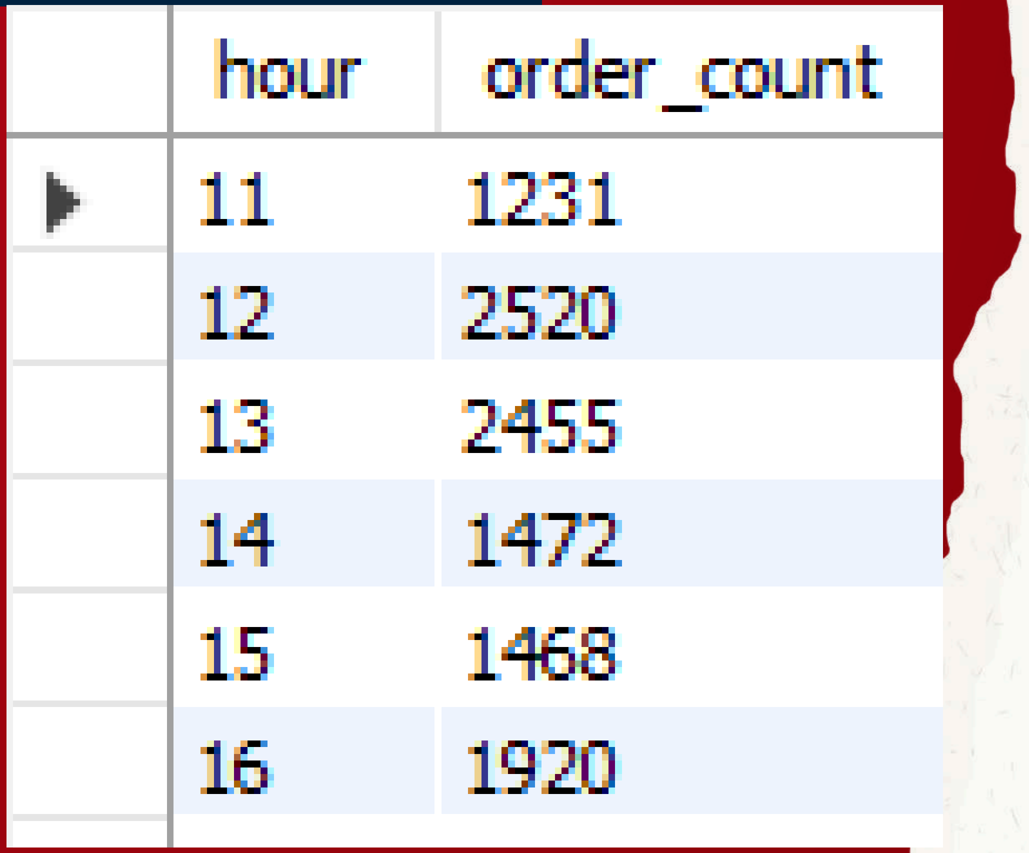


category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Q7. DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

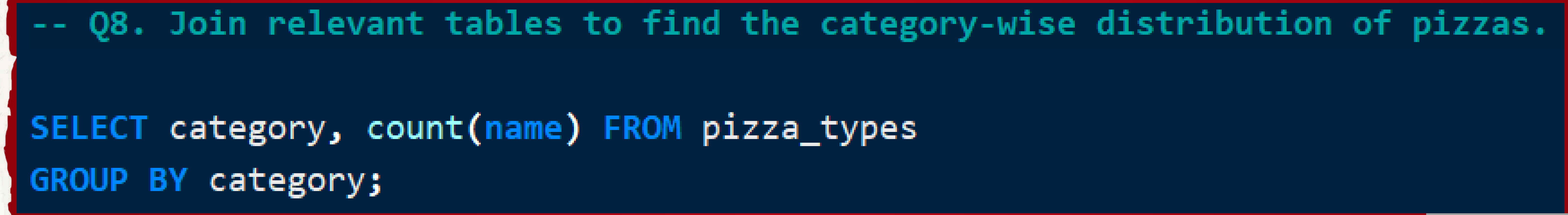


```
-- Q7. Determine the distribution of orders by hour of the day.

SELECT hour(order_time) AS hour, count(order_id) AS order_count
FROM orders
GROUP BY hour;
```

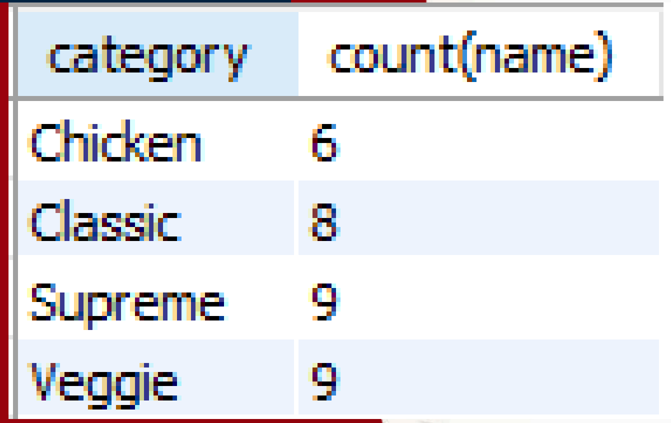


hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920

Q8. JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.

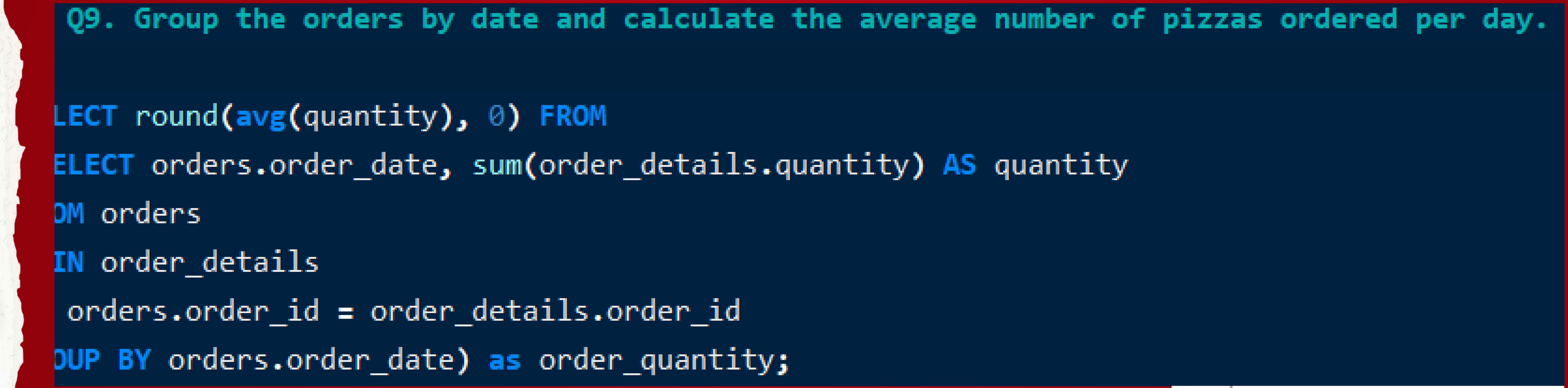


```
-- Q8. Join relevant tables to find the category-wise distribution of pizzas.


SELECT category, count(name) FROM pizza_types
GROUP BY category;
```

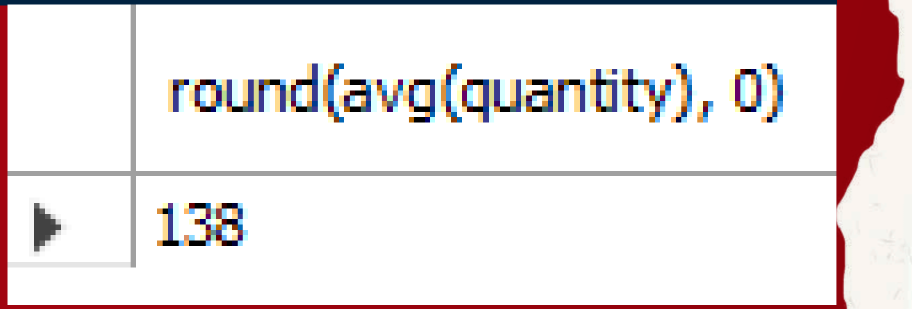


category	count(name)
Chicken	6
Classic	8
Supreme	9
Veggie	9

Q9. GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS

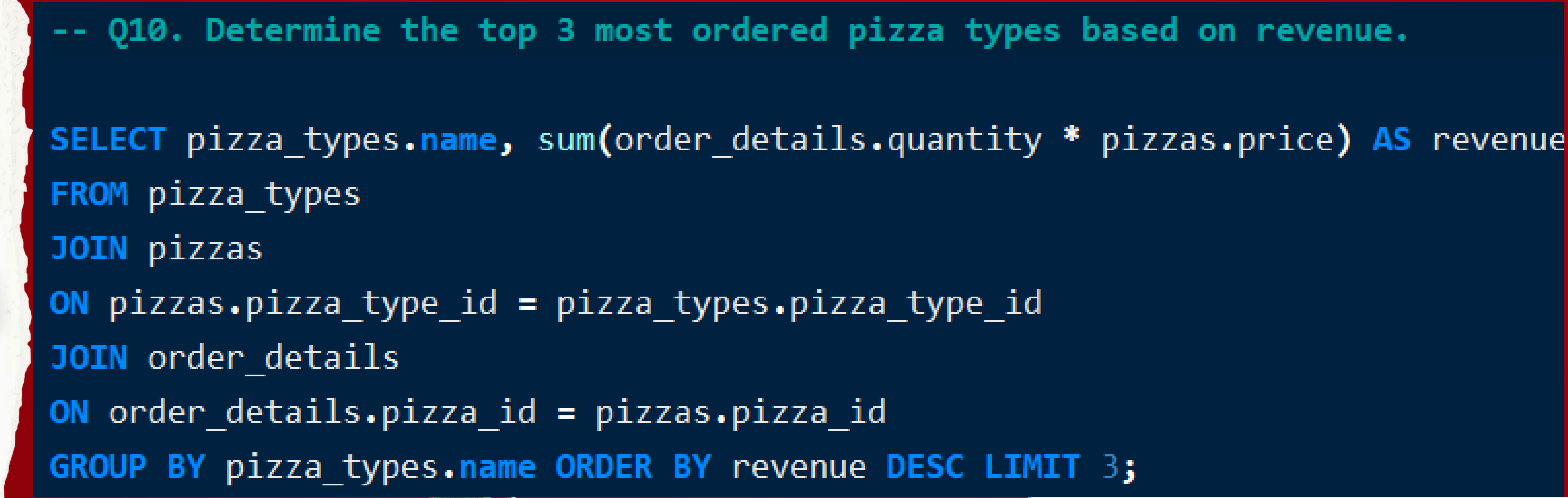


```
 Q9. Group the orders by date and calculate the average number of pizzas ordered per day.

LECT round(avg(quantity), 0) FROM
ELECT orders.order_date, sum(order_details.quantity) AS quantity
OM orders
IN order_details
 orders.order_id = order_details.order_id
OUP BY orders.order_date) as order_quantity;
```

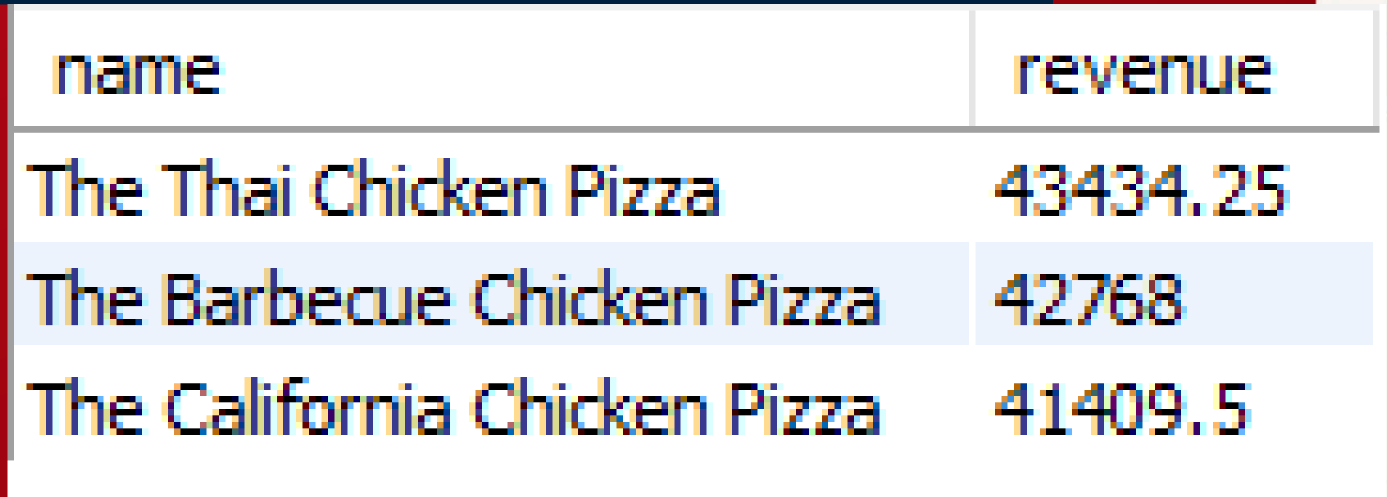


```
   round(avg(quantity), 0)
▶  138
```

Q10. GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

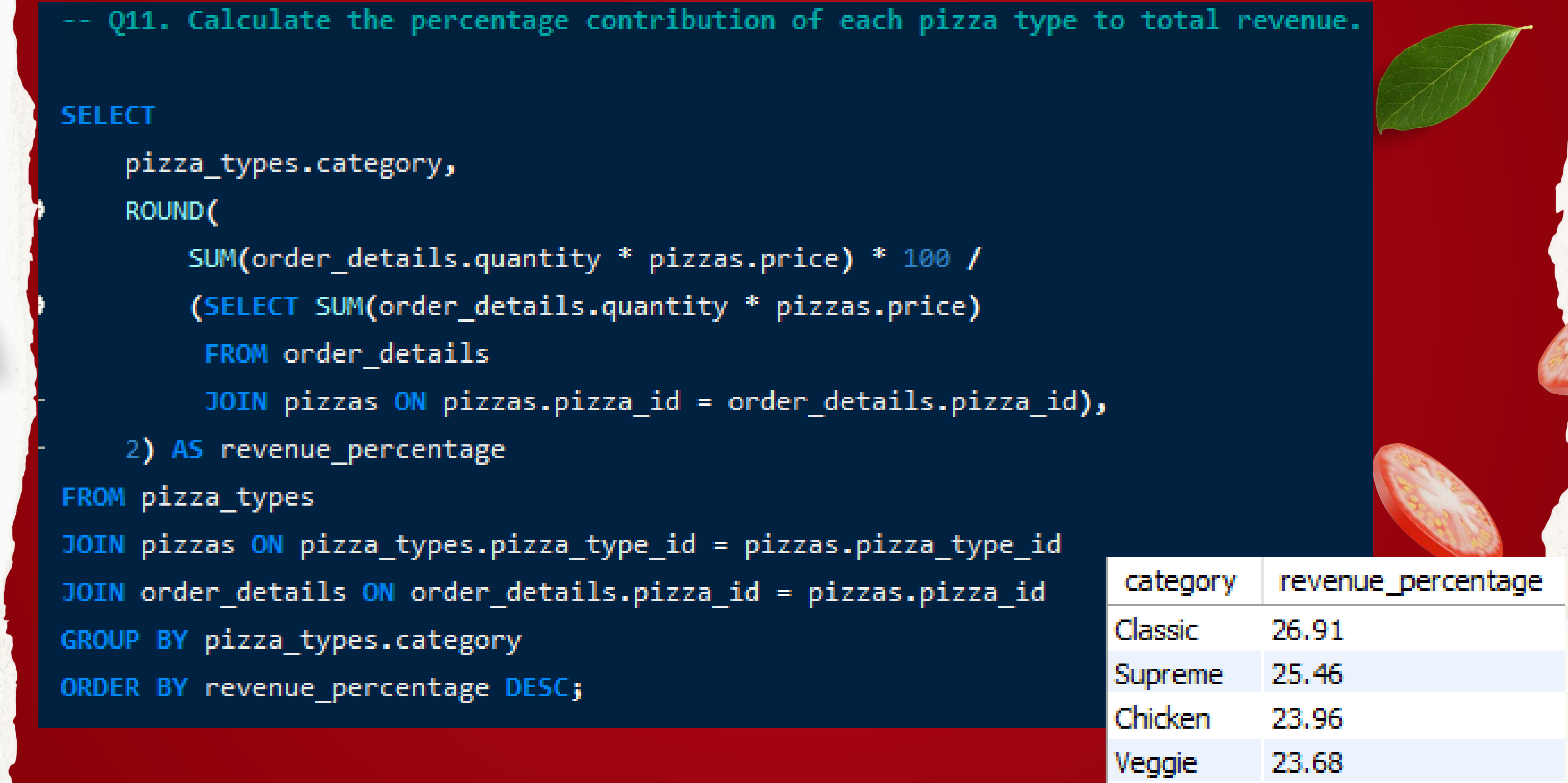


```
-- Q10. Determine the top 3 most ordered pizza types based on revenue.

SELECT pizza_types.name, sum(order_details.quantity * pizzas.price) AS revenue
FROM pizza_types
JOIN pizzas
ON pizzas.pizza_type_id = pizza_types.pizza_type_id
JOIN order_details
ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name ORDER BY revenue DESC LIMIT 3;
```



name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

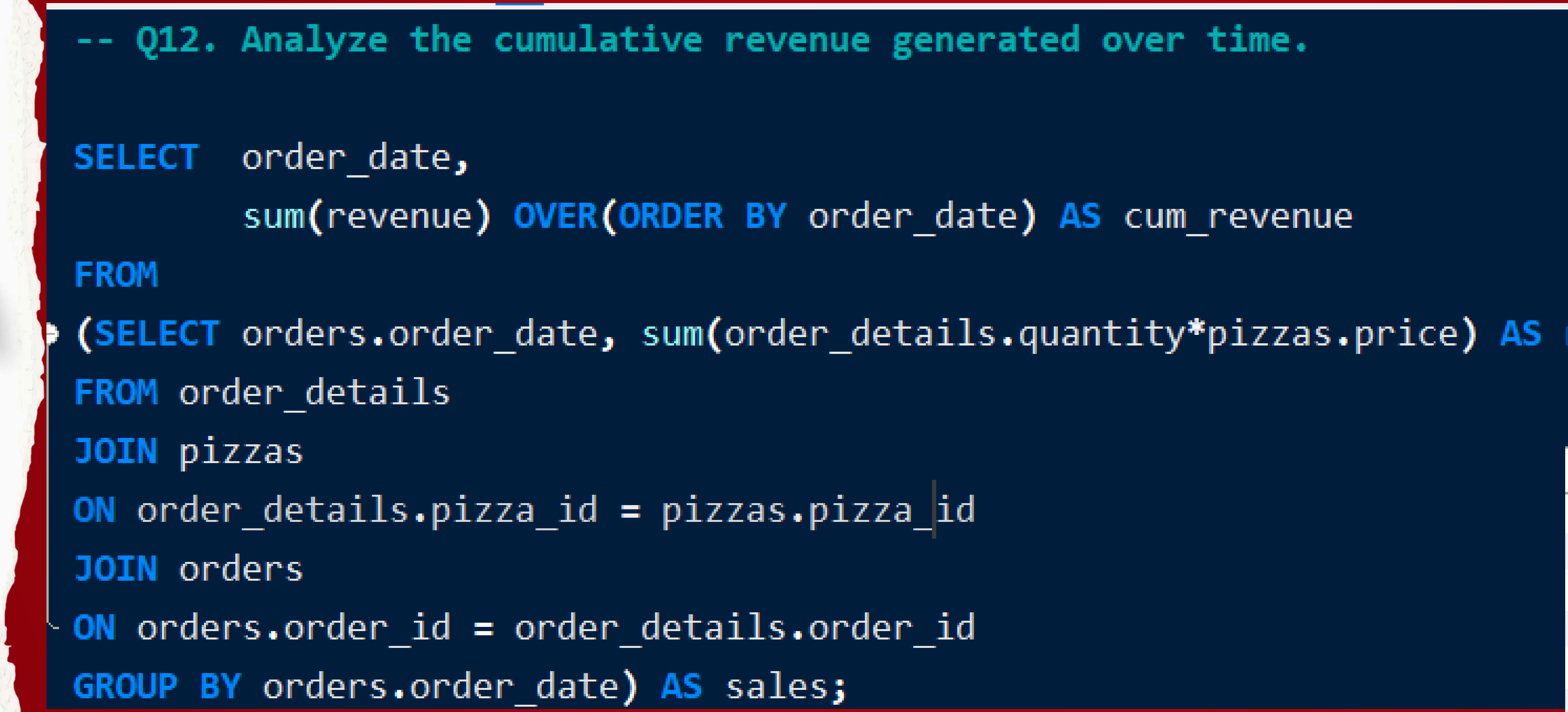
Q11. CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE.



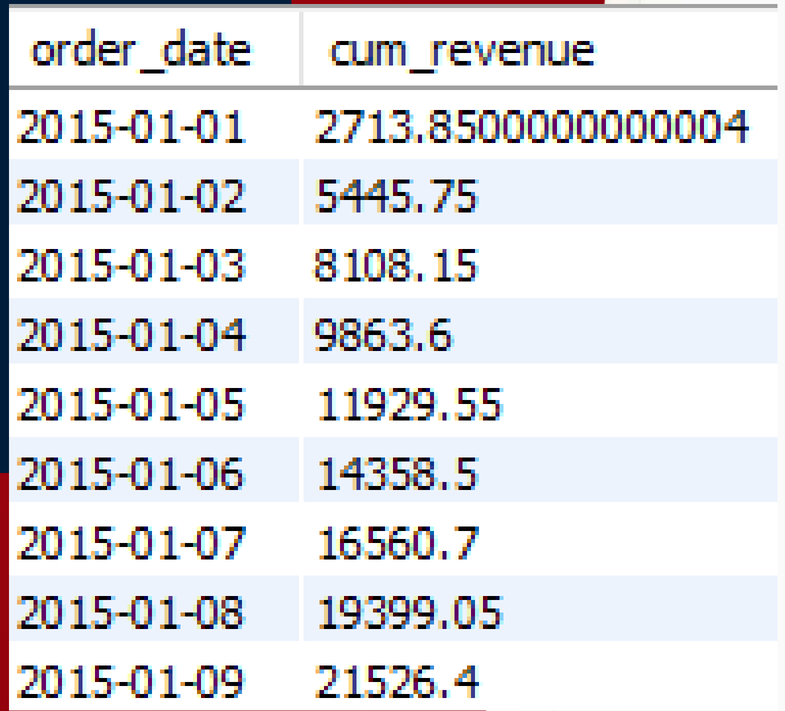
```sql
-- Q11. Calculate the percentage contribution of each pizza type to total revenue.


SELECT
    pizza_types.category,
    ROUND(
        SUM(order_details.quantity * pizzas.price) * 100 /
        (SELECT SUM(order_details.quantity * pizzas.price)
         FROM order_details
         JOIN pizzas ON pizzas.pizza_id = order_details.pizza_id),
    2) AS revenue_percentage
FROM pizza_types
JOIN pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue_percentage DESC;
```

category	revenue_percentage
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Q12. ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

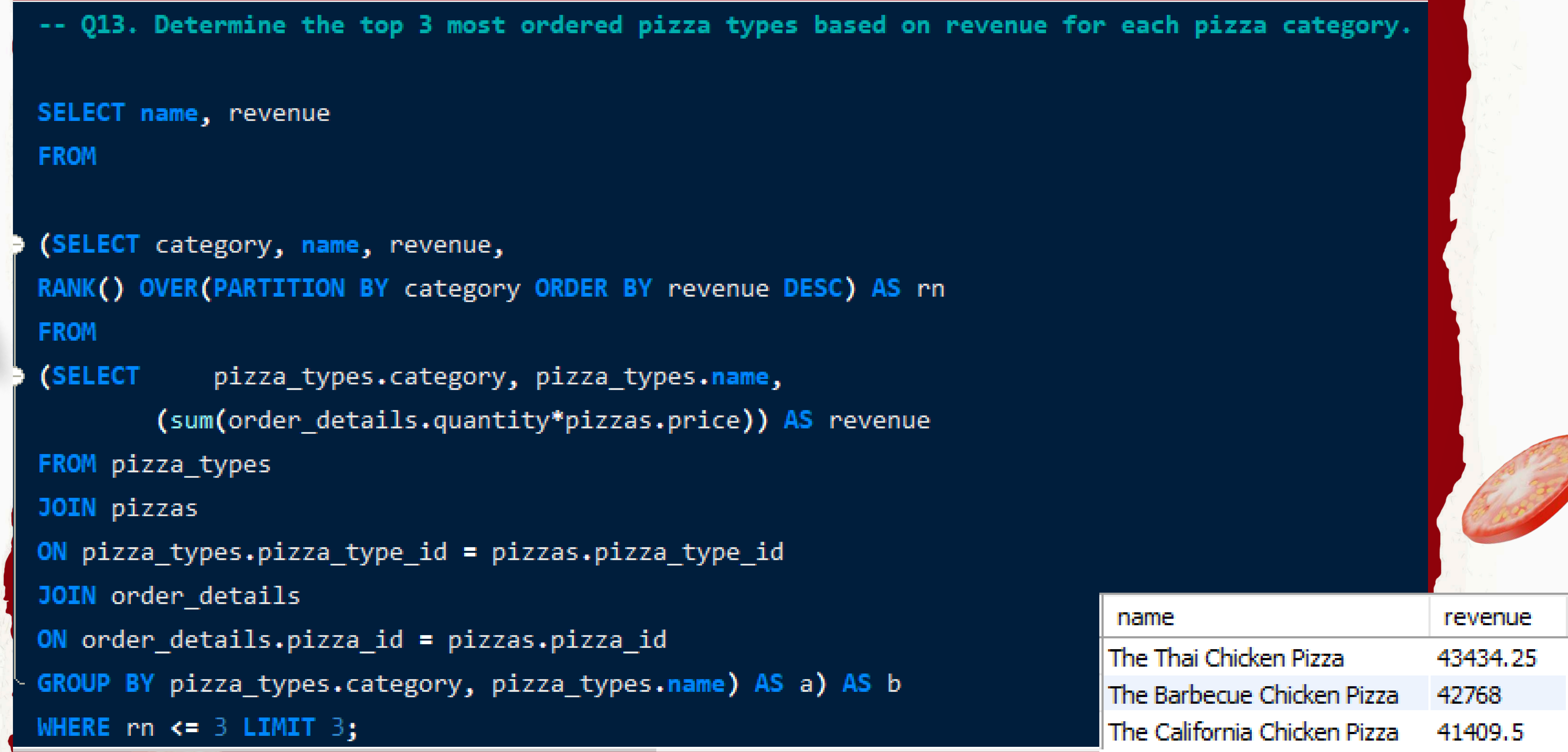


```
-- Q12. Analyze the cumulative revenue generated over time.

SELECT  order_date,
        sum(revenue) OVER(ORDER BY order_date) AS cum_revenue
FROM
(SELECT orders.order_date, sum(order_details.quantity*pizzas.price) AS revenue
FROM order_details
JOIN pizzas
ON order_details.pizza_id = pizzas.pizza_id
JOIN orders
ON orders.order_id = order_details.order_id
GROUP BY orders.order_date) AS sales;
```



order_date	cum_revenue
2015-01-01	2713.8500000000004
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55
2015-01-06	14358.5
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526.4

Q13. DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY.



```
-- Q13. Determine the top 3 most ordered pizza types based on revenue for each pizza category.


SELECT name, revenue
FROM


(SELECT category, name, revenue,
RANK() OVER(PARTITION BY category ORDER BY revenue DESC) AS rn
FROM
(SELECT     pizza_types.category, pizza_types.name,
        (sum(order_details.quantity*pizzas.price)) AS revenue
FROM pizza_types
JOIN pizzas
ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN order_details
ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category, pizza_types.name) AS a) AS b
WHERE rn <= 3 LIMIT 3;
```

name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

INSIGHTS

- 21,350 ORDERS SHOW STRONG CUSTOMER DEMAND.
- PIZZA SALES EARNED $817,860, INDICATING A HEALTHY MARKET.
- THE GREEK PIZZA IS THE HIGHEST-PRICED PREMIUM OPTION.
- LARGE PIZZAS ARE THE MOST POPULAR SIZE.
- FIVE PIZZA TYPES DOMINATE CUSTOMER PREFERENCES.
- CLASSIC CATEGORY LEADS ORDERS; OTHERS ALSO SELL WELL.
- HOURLY ORDER ANALYSIS CAN IMPROVE STAFFING AND PROMOTIONS.
- SUPREME AND VEGGIE CATEGORIES OFFER THE MOST VARIETY.
- AVERAGE DAILY SALES ARE 138 PIZZAS, ENSURING STEADY REVENUE.
- CHICKEN PIZZAS GENERATE THE HIGHEST REVENUE.
- REVENUE IS EVENLY SPREAD, WITH CLASSIC AND SUPREME LEADING.
- REVENUE STEADILY GROWS, SHOWING INCREASING CUSTOMER BASE.

STRATEGIC OUTLOOK

- FOCUS MARKETING AND PROMOTIONS ON LARGE PIZZAS AND POPULAR VARIETIES LIKE BARBECUE CHICKEN AND CLASSIC DELUXE TO DRIVE HIGHER SALES VOLUME.
- DEVELOP NEW PREMIUM SPECIALTY PIZZAS INSPIRED BY THE GREEK PIZZA TO ATTRACT CUSTOMERS WILLING TO PAY MORE AND INCREASE AVERAGE ORDER VALUE.
- OPTIMIZE STAFFING AND DELIVERY CAPACITY DURING PEAK HOURS IDENTIFIED FROM ORDER TIMING DATA TO IMPROVE SERVICE SPEED AND CUSTOMER SATISFACTION.

THANK YOU!