SQL-BASED SALES ANALYSIS FOR PIZZA HUT

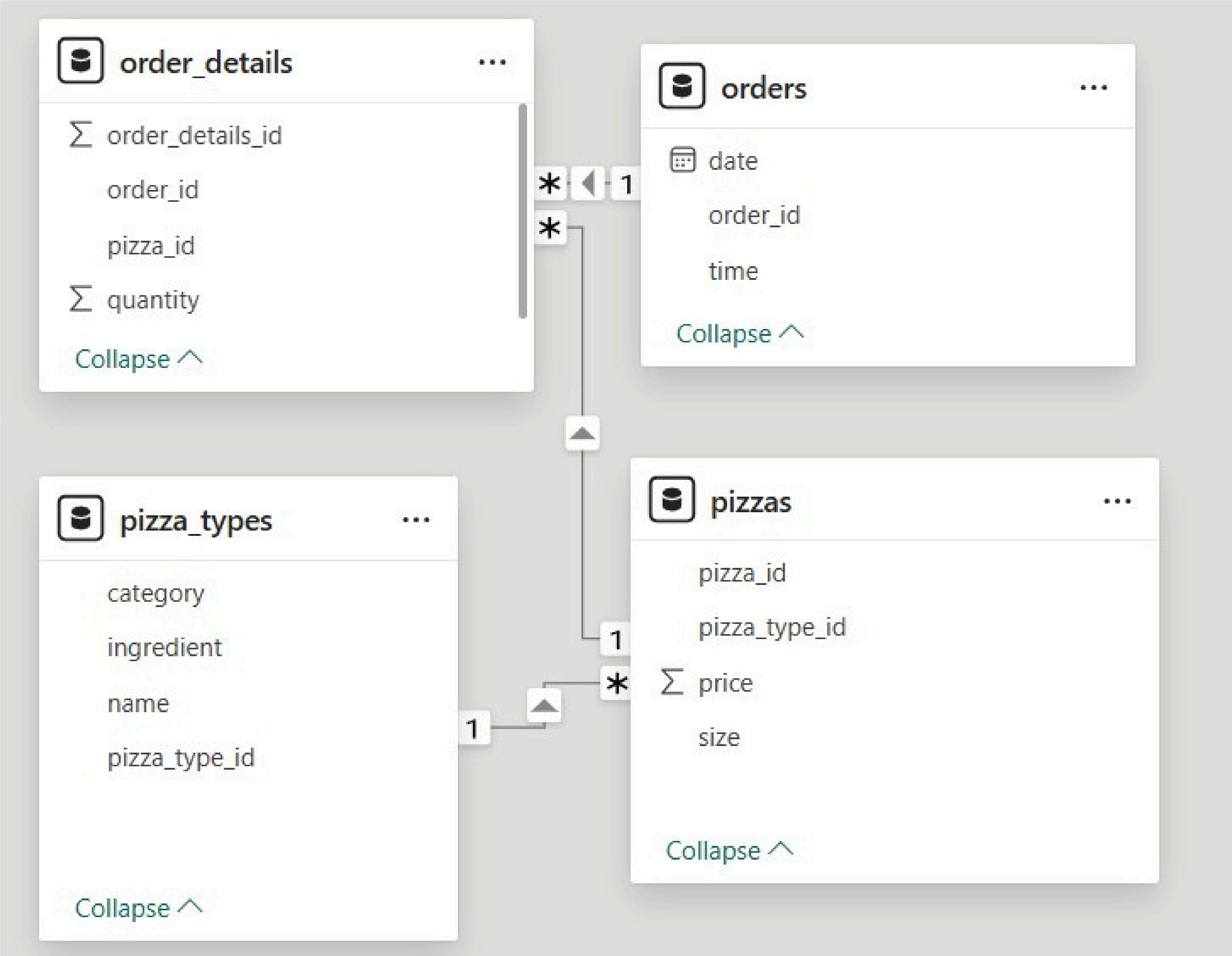
INTRODUCTION

This project focuses on analyzing Pizza Hut’s sales data using SQL to uncover meaningful business insights.
The goal is to understand customer behavior, sales performance, and product trends to support data-driven decision making.

PROBLEM STATEMENT

Businesses generate large amounts of sales data, but without proper analysis, valuable insights remain hidden.
This project aims to transform raw sales data into actionable insights using SQL.

DATABASE SCHEMA



order_details

- ∑ order_details_id
- order_id
- pizza_id
- ∑ quantity

orders

- date
- order_id
- time

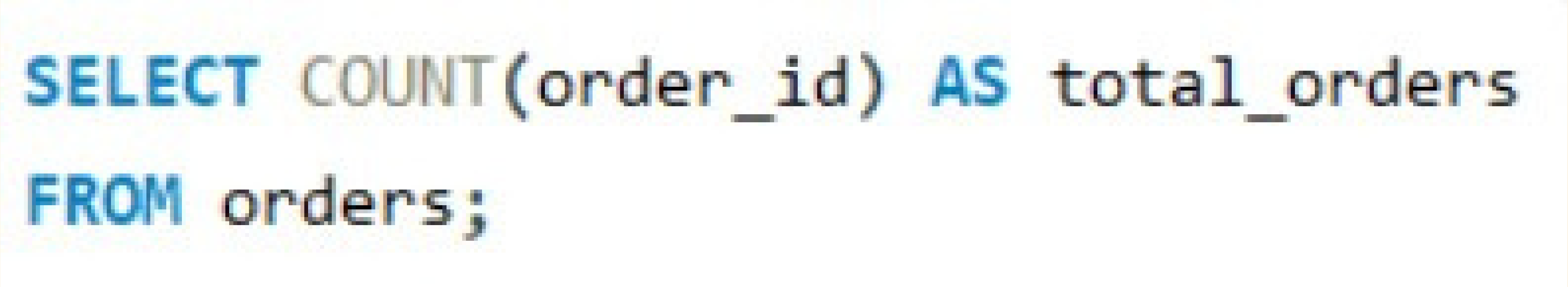
pizza_types

- category
- ingredient
- name
- pizza_type_id

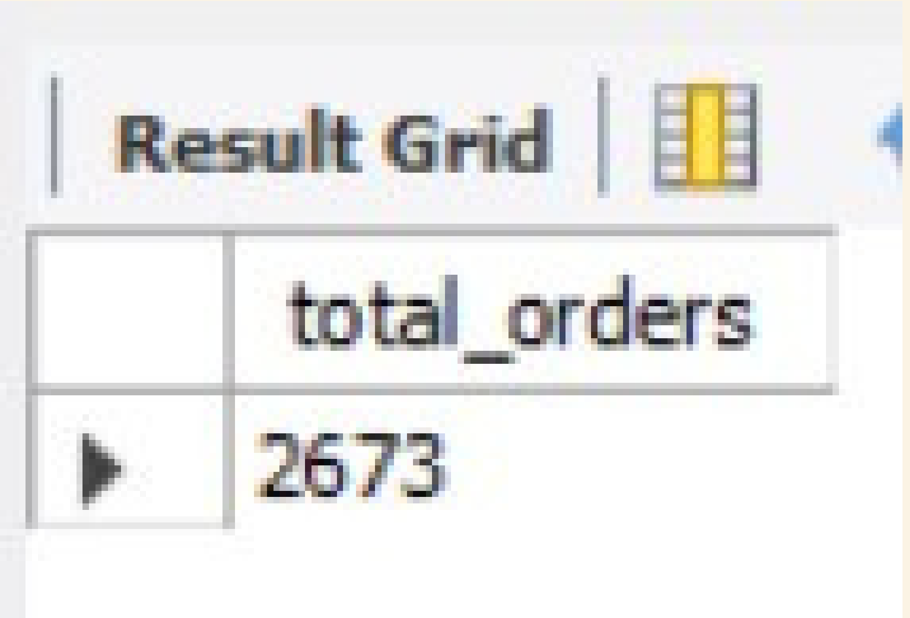
pizzas

- pizza_id
- pizza_type_id
- ∑ price
- size

RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED



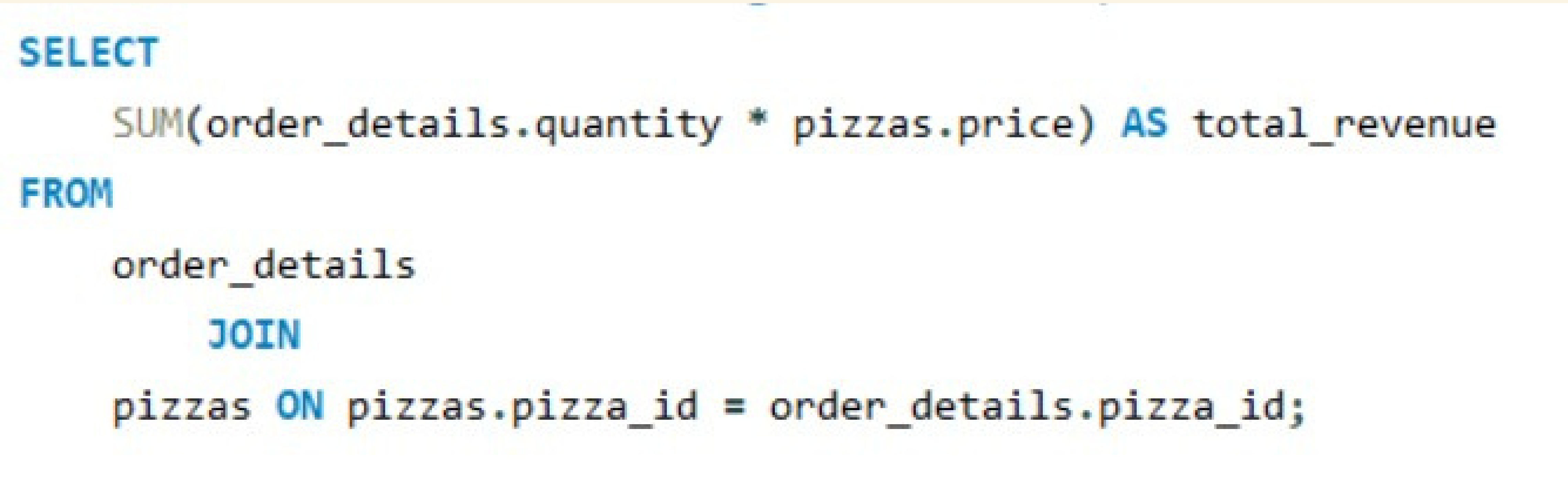
```sql
SELECT COUNT(order_id) AS total_orders
FROM orders;
```



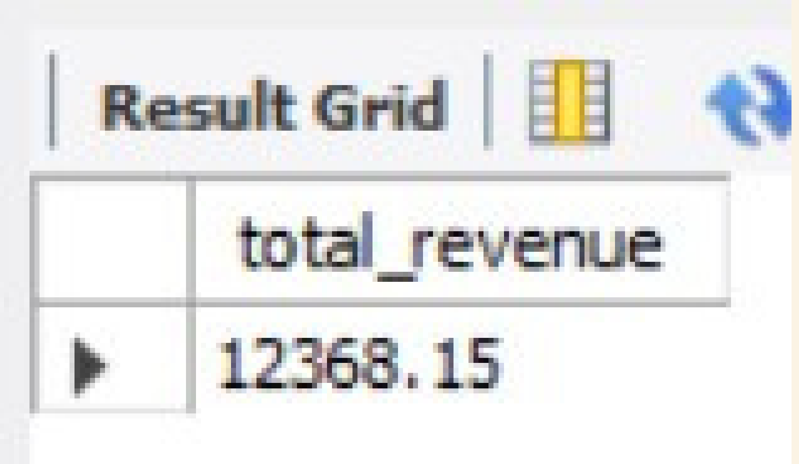
```
total_orders
2673
```

2,673 orders were placed , reflecting consistent customer activity

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES



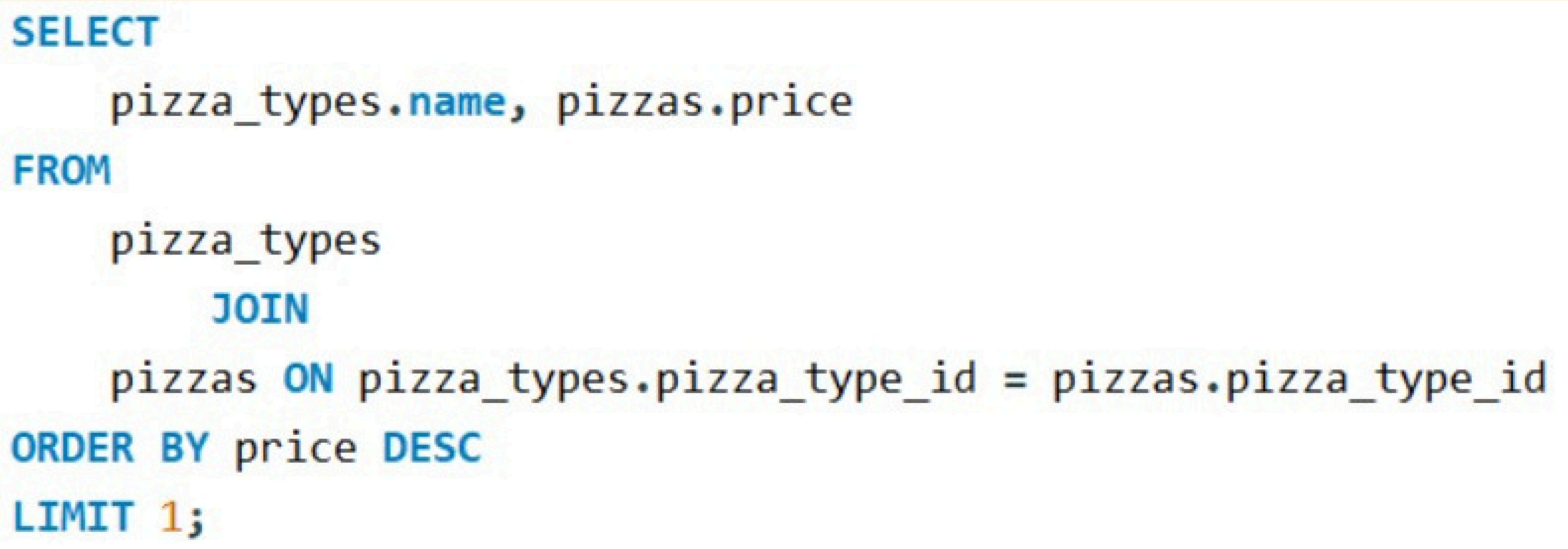
```
SELECT
    SUM(order_details.quantity * pizzas.price) AS total_revenue
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id;
```



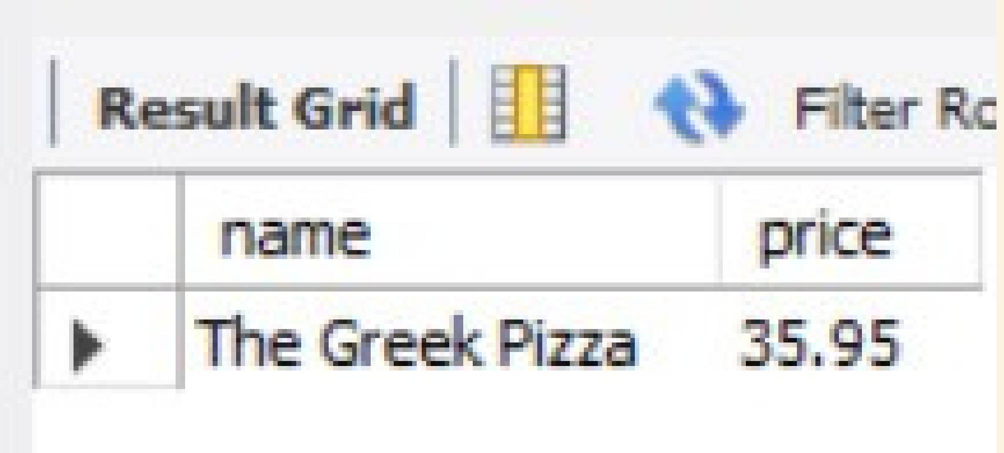
```
total_revenue
12368.15
```

The total revenue is $123,68.50

IDENTIFY THE HIGHEST-PRICED PIZZA



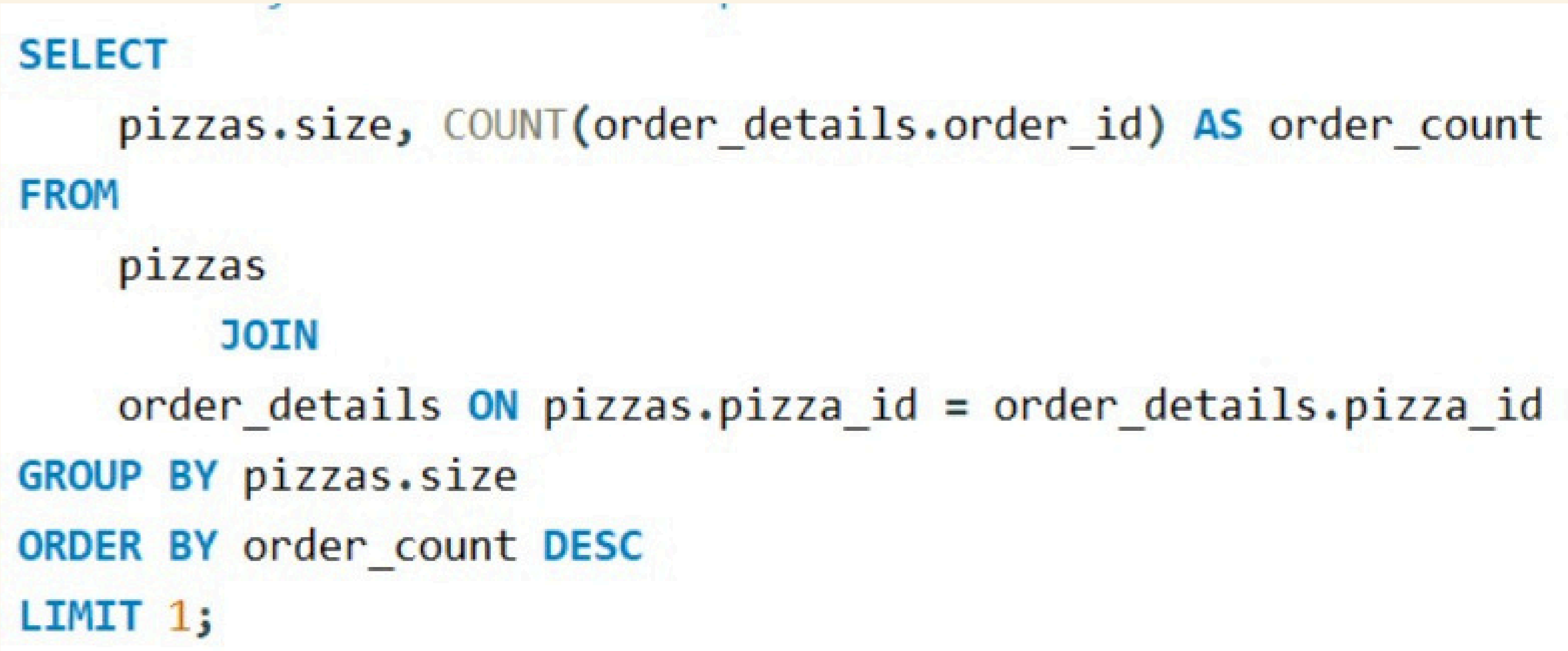
```sql
SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY price DESC
LIMIT 1;
```



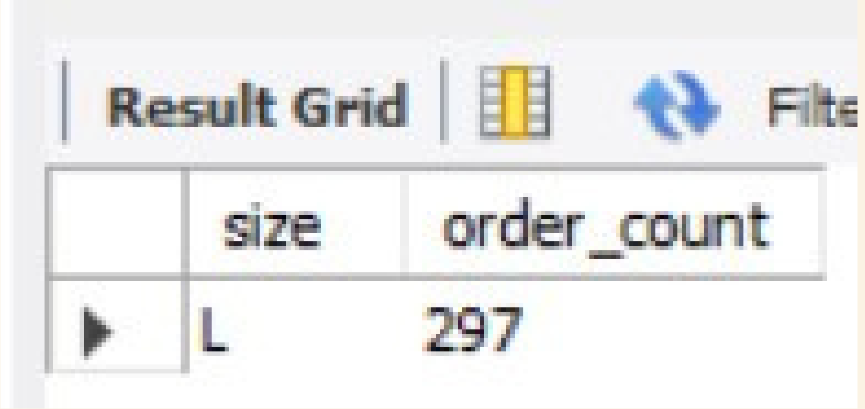
name	price
The Greek Pizza	35.95

The Greek Pizza is the highest priced item at $35.95

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED



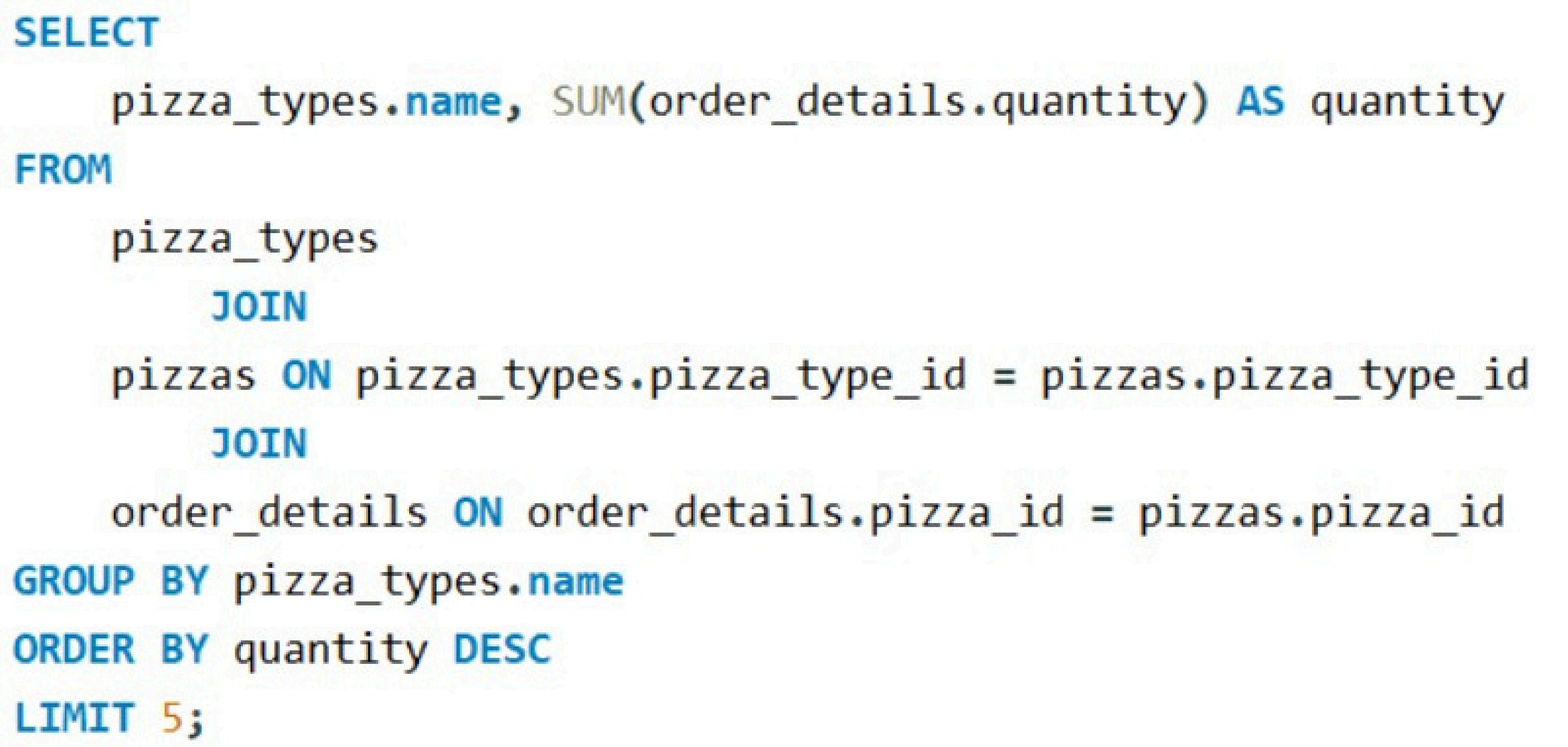
```
SELECT
    pizzas.size, COUNT(order_details.order_id) AS order_count
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC
LIMIT 1;
```



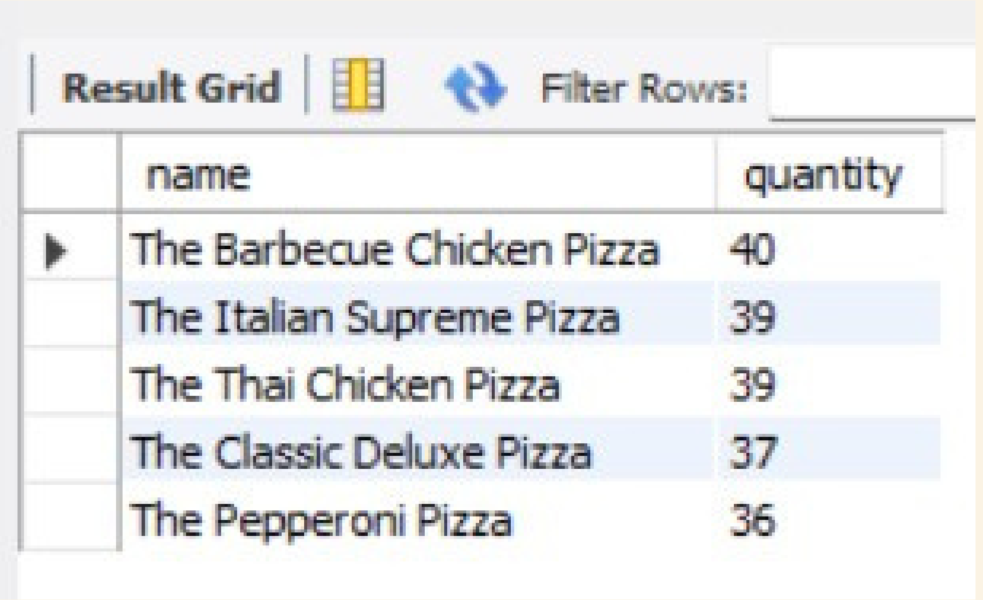
```
   size   order_count
▶  L      297
```

Large-sized pizzas are the most popular choice, showing customers prefer value-sized orders.

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



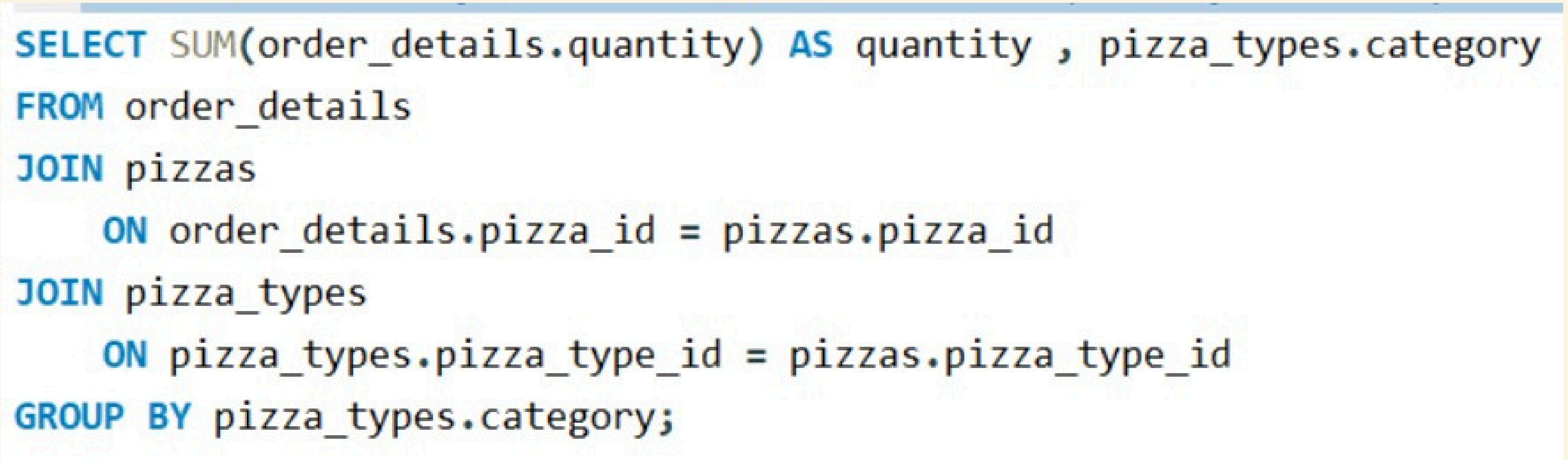
```
SELECT
    pizza_types.name, SUM(order_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC
LIMIT 5;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



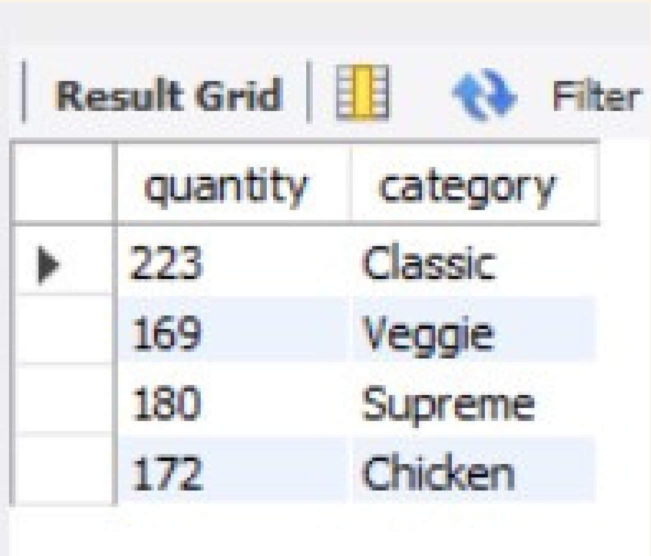
name	quantity
The Barbecue Chicken Pizza	40
The Italian Supreme Pizza	39
The Thai Chicken Pizza	39
The Classic Deluxe Pizza	37
The Pepperoni Pizza	36

The Barbeque Chicken Pizza leads in demand, followed by Italian Supreme and Thai Chicken, indicating strong preference for classic and chicken-based flavors.

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



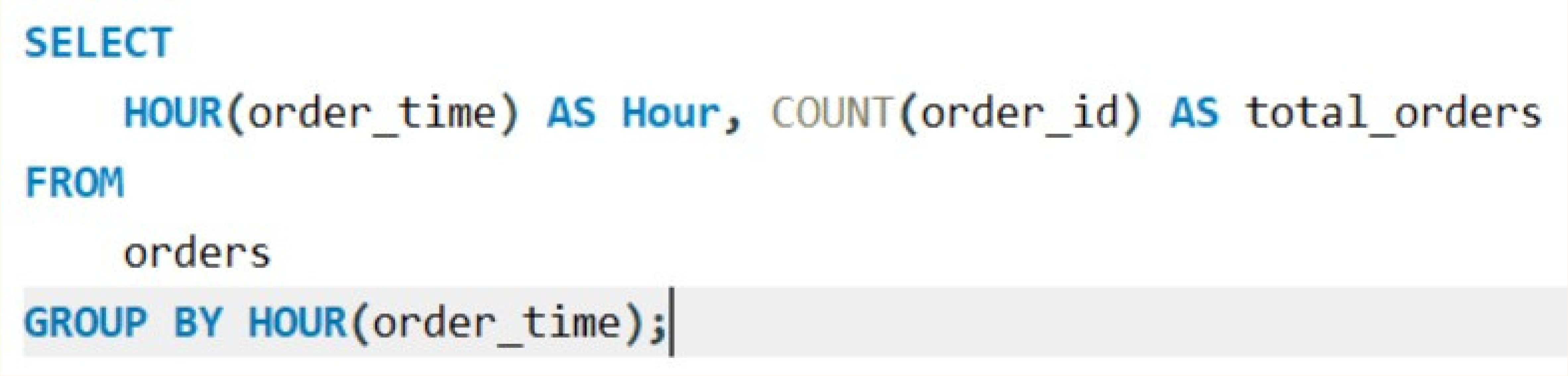
```sql
SELECT SUM(order_details.quantity) AS quantity , pizza_types.category
FROM order_details
JOIN pizzas
    ON order_details.pizza_id = pizzas.pizza_id
JOIN pizza_types
    ON pizza_types.pizza_type_id = pizzas.pizza_type_id
GROUP BY pizza_types.category;
```



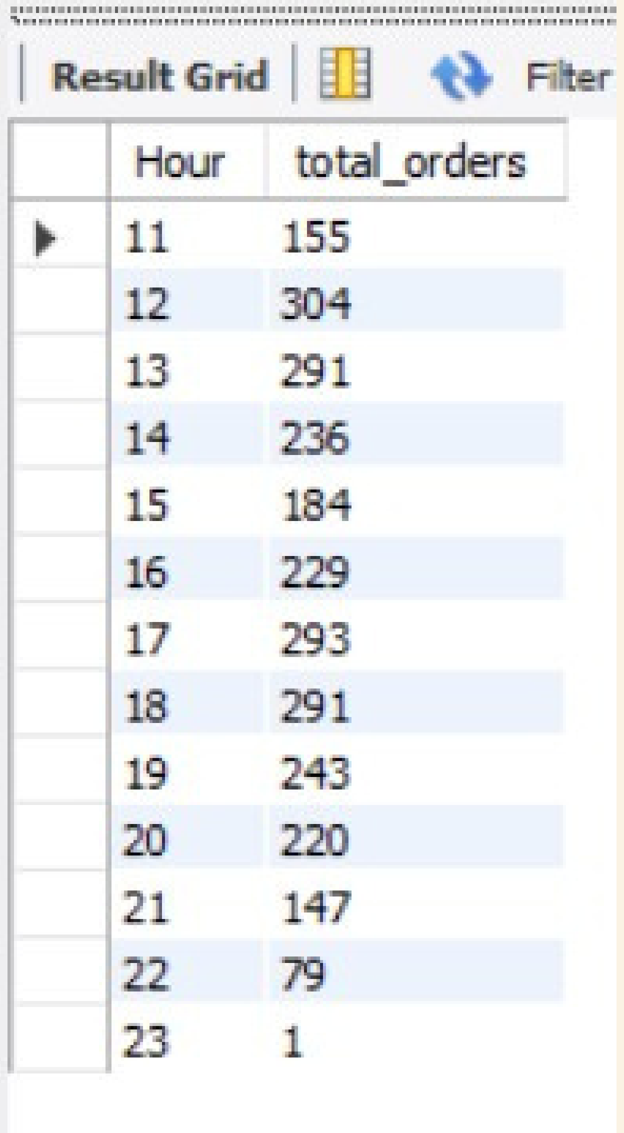
quantity	category
223	Classic
169	Veggie
180	Supreme
172	Chicken

The Classic category has the highest quantity sold, making it the most popular pizza category

DETERMAINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY



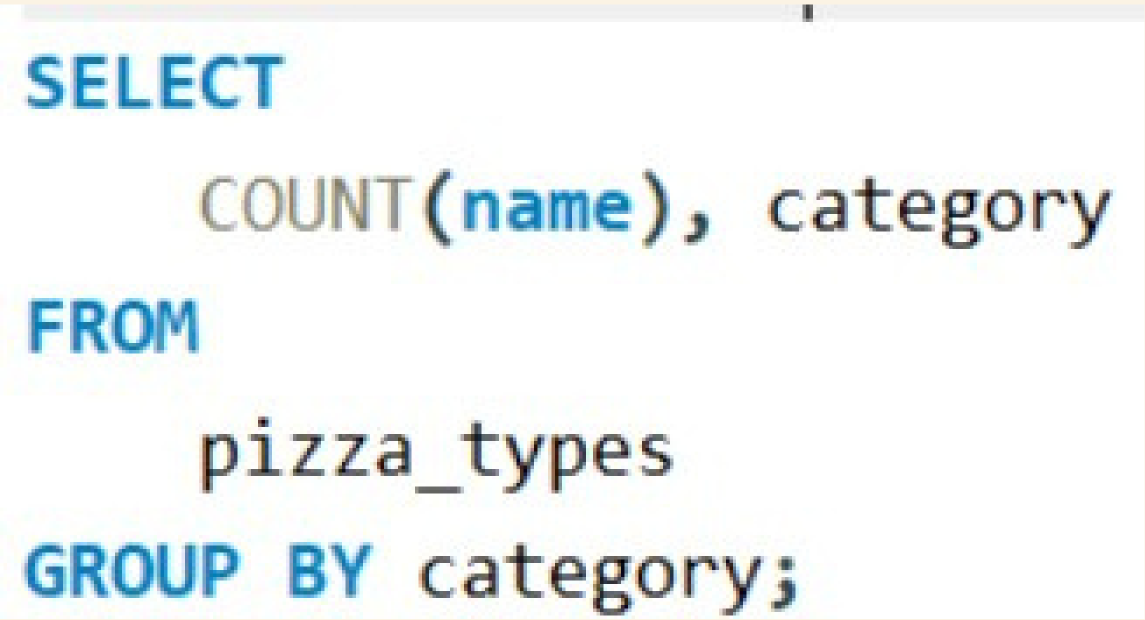
```
SELECT
    HOUR(order_time) AS Hour, COUNT(order_id) AS total_orders
FROM
    orders
GROUP BY HOUR(order_time);
```



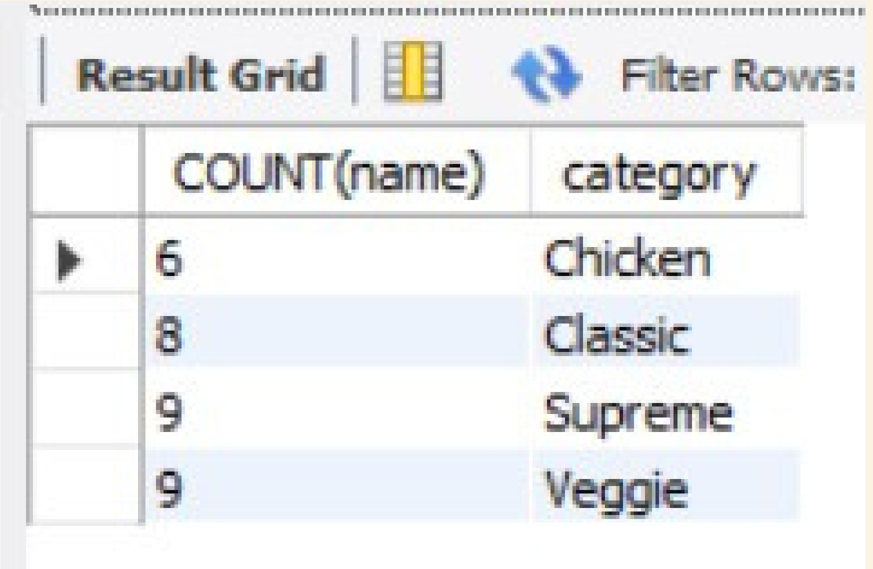
Hour	total_orders
11	155
12	304
13	291
14	236
15	184
16	229
17	293
18	291
19	243
20	220
21	147
22	79
23	1

Order volume peaks around 12 PM, highlighting lunch hours as the busiest period

JOIN RELEVANT TABLES TO FIND THE CATEGORY WISE DISTRIBUTION OF PIZZAS



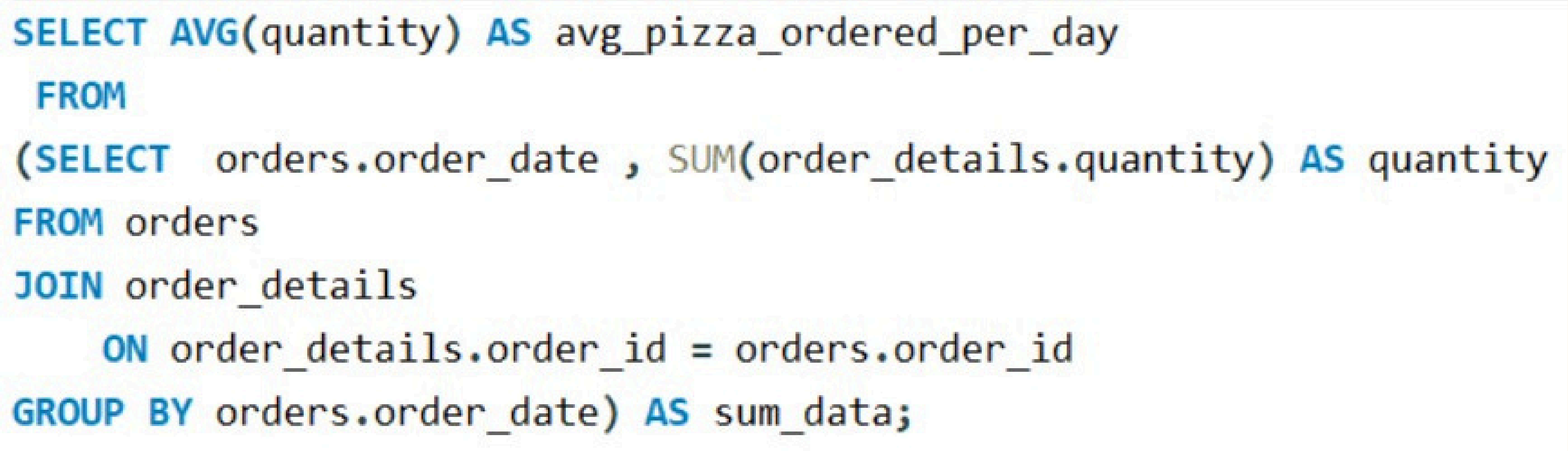
```
SELECT
    COUNT(name), category
FROM
    pizza_types
GROUP BY category;
```



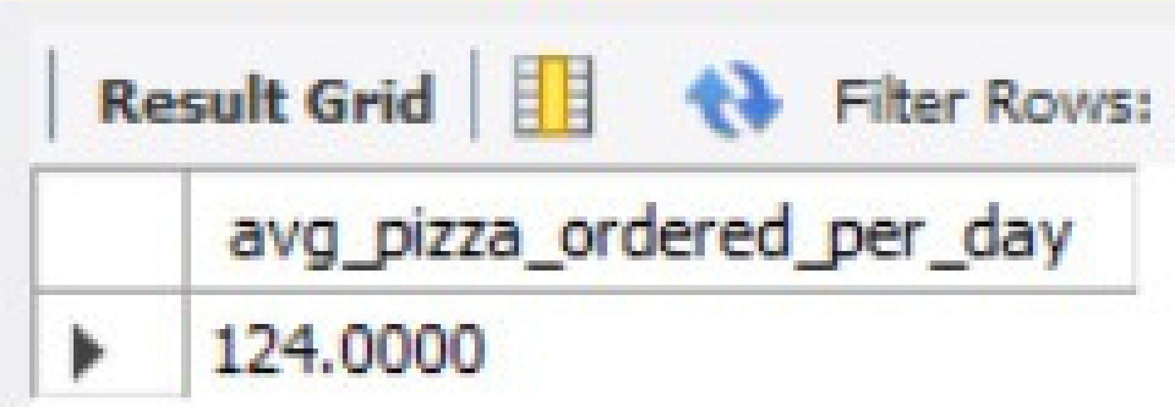
COUNT(name)	category
6	Chicken
8	Classic
9	Supreme
9	Veggie

The menu offers a balanced distribution across categories, ensuring variety for different customer preferences.

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



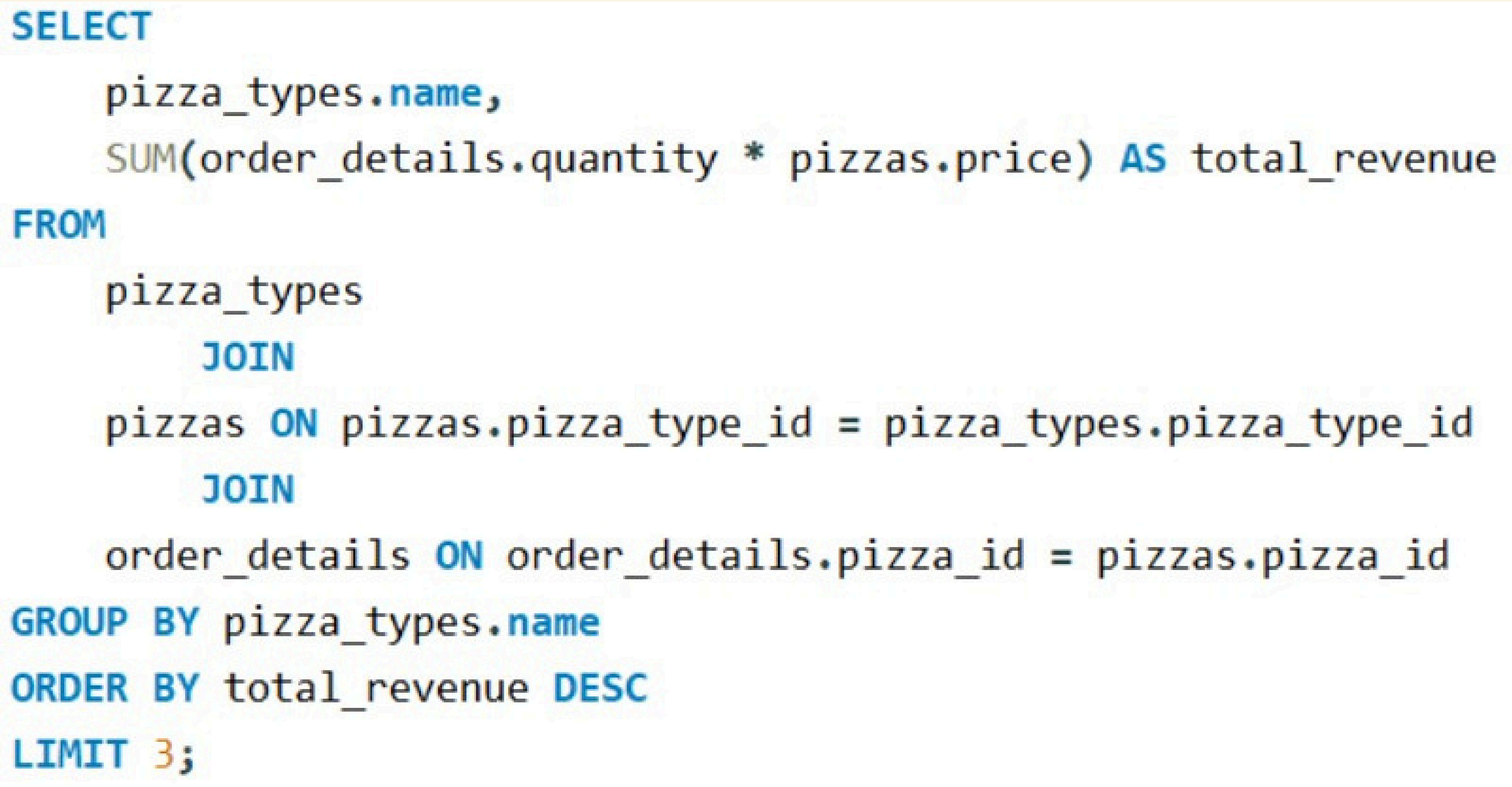
```
SELECT AVG(quantity) AS avg_pizza_ordered_per_day
 FROM
(SELECT  orders.order_date , SUM(order_details.quantity) AS quantity
FROM orders
JOIN order_details
    ON order_details.order_id = orders.order_id
GROUP BY orders.order_date) AS sum_data;
```



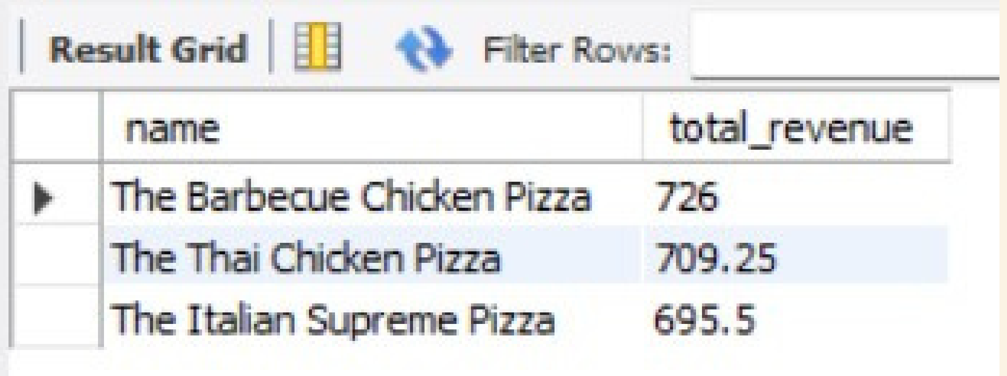
```
avg_pizza_ordered_per_day
124.0000
```

On average, 124 pizzas are ordered daily, indicating stable daily sales activity

DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE



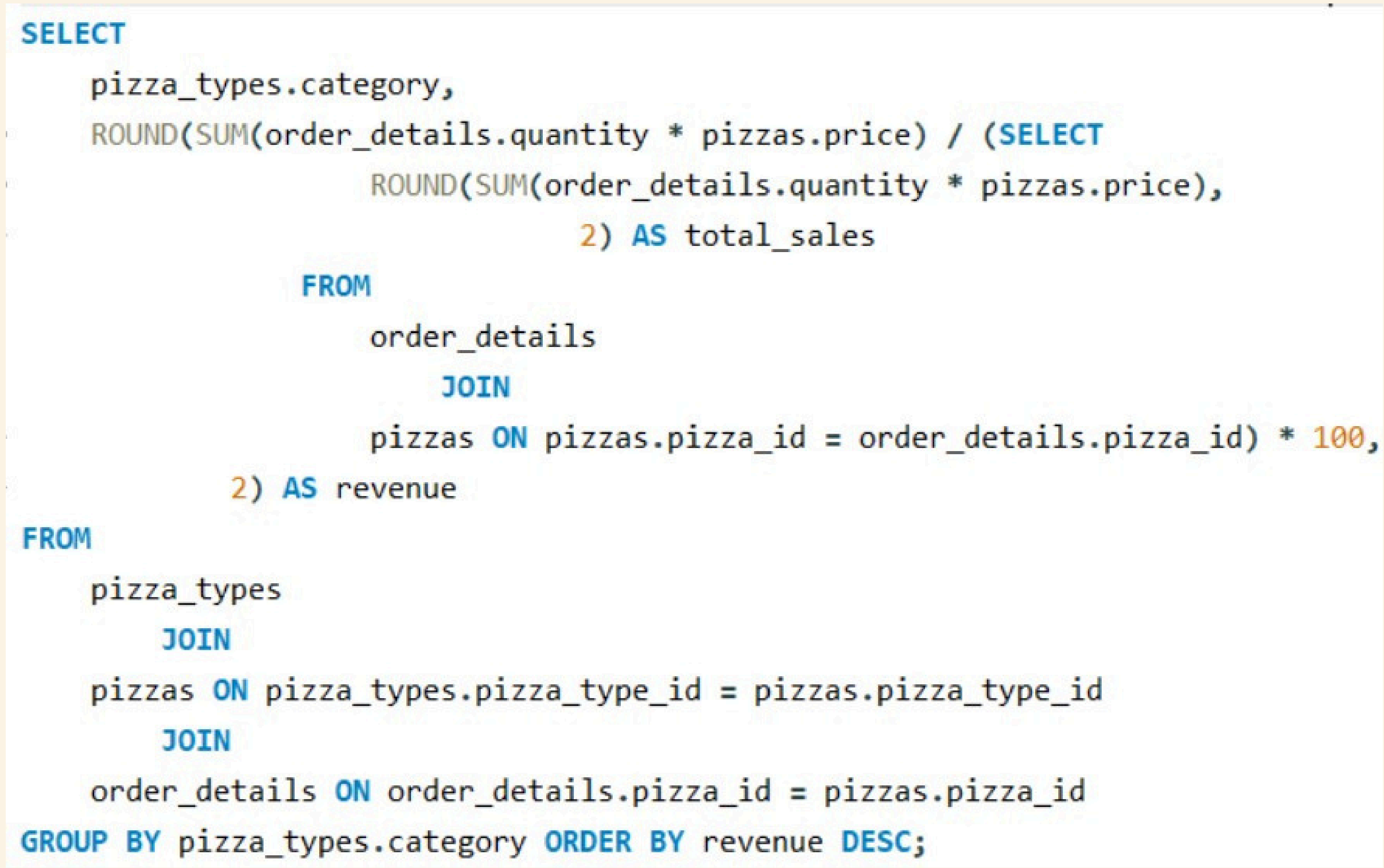
```
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) AS total_revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER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



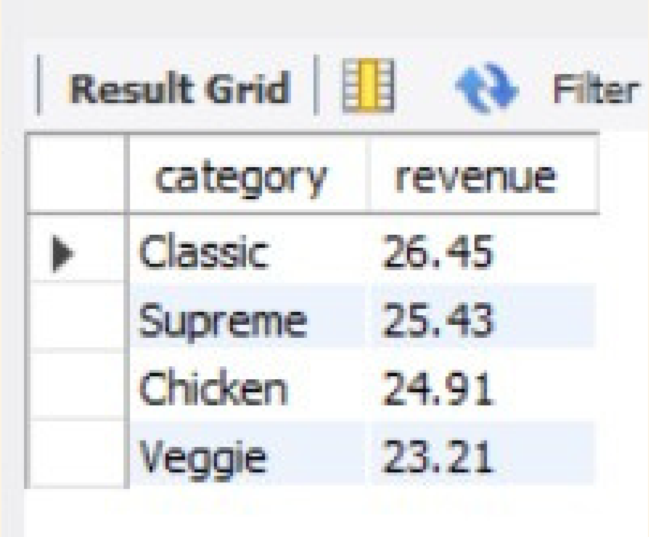
name	total_revenue
The Barbecue Chicken Pizza	726
The Thai Chicken Pizza	709.25
The Italian Supreme Pizza	695.5

Barbeque Chicken Pizza generates the highest revenue

CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTA REVENUE



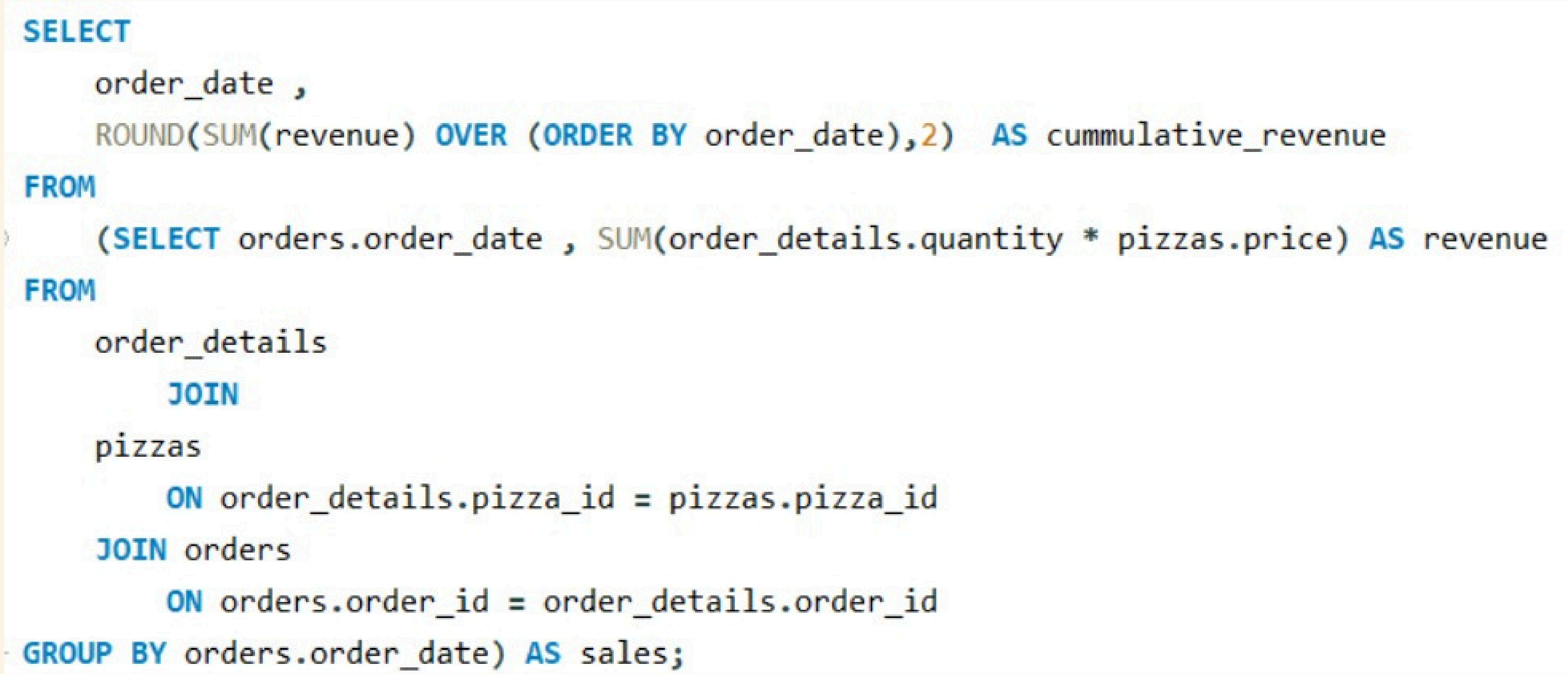
```
SELECT
    pizza_types.category,
    ROUND(SUM(order_details.quantity * pizzas.price) / (SELECT
                    ROUND(SUM(order_details.quantity * pizzas.price),
                                2) AS total_sales
                FROM
                    order_details
                        JOIN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



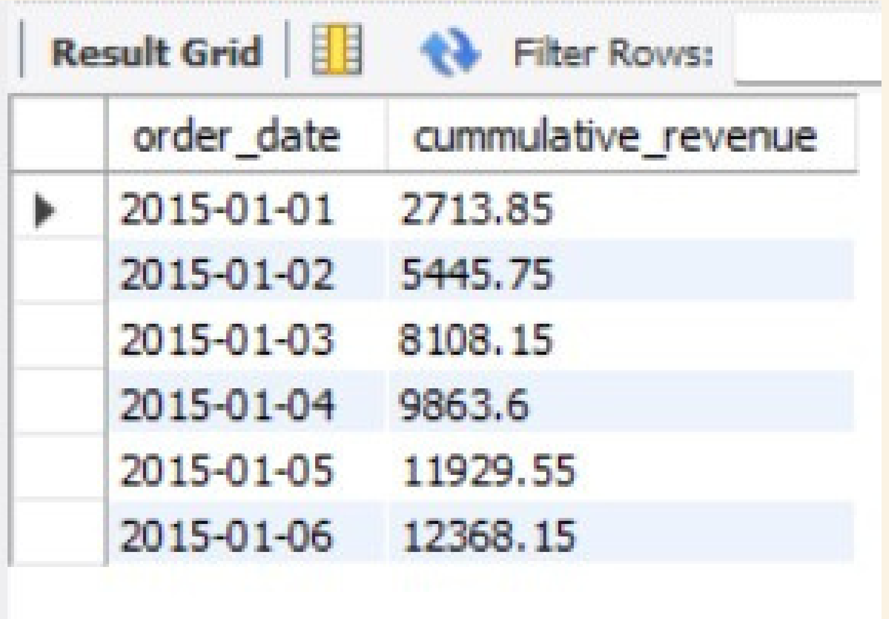
category	revenue
Classic	26.45
Supreme	25.43
Chicken	24.91
Veggie	23.21

Revenue is evenly distributed across categories, with Classic contributing the highest share at 26.45%.

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME



```
SELECT
    order_date ,
    ROUND(SUM(revenue) OVER (ORDER BY order_date),2)  AS cummulative_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   cummulative_revenue
▶ 2015-01-01   2713.85
  2015-01-02   5445.75
  2015-01-03   8108.15
  2015-01-04   9863.6
  2015-01-05   11929.55
  2015-01-06   12368.15
```

The steady upward trend shows consistent revenue growth over time.