MYSQL PROJECT ON PIZZA SALES

INTRODUCTION TO PIZZA SALES DATA ANALYSIS PROJECT

This project analyzes pizza sales data to uncover insights into revenue trends, top-selling pizzas, and customer order patterns. By using SQL queries, extract meaningful insights that can help a restaurant improve its sales strategy and inventory management.

DATASET INFORMATION

The dataset consists of the following tables:

- **Orders Table:** Contains order_id, including order_date, order_time.
- **order_details Table:** order_details_id, order_id, pizza_id, quantity.
- **Pizzas Table:** pizza_id, Lists different pizza types, their prices, and size.
- **pizza_types:** pizza_type_id, pizza_names, categories, ingredients.

QUESTIONS

Basic :

- Retrieve the total number of orders placed.
- Calculate the total revenue generated from pizza sales.
- Identify the most common pizza size ordered.
- List the top 5 most ordered pizza types along with their quantities.
- which day of the week are the most orders placed
- determine the best and worst month in terms of order volume

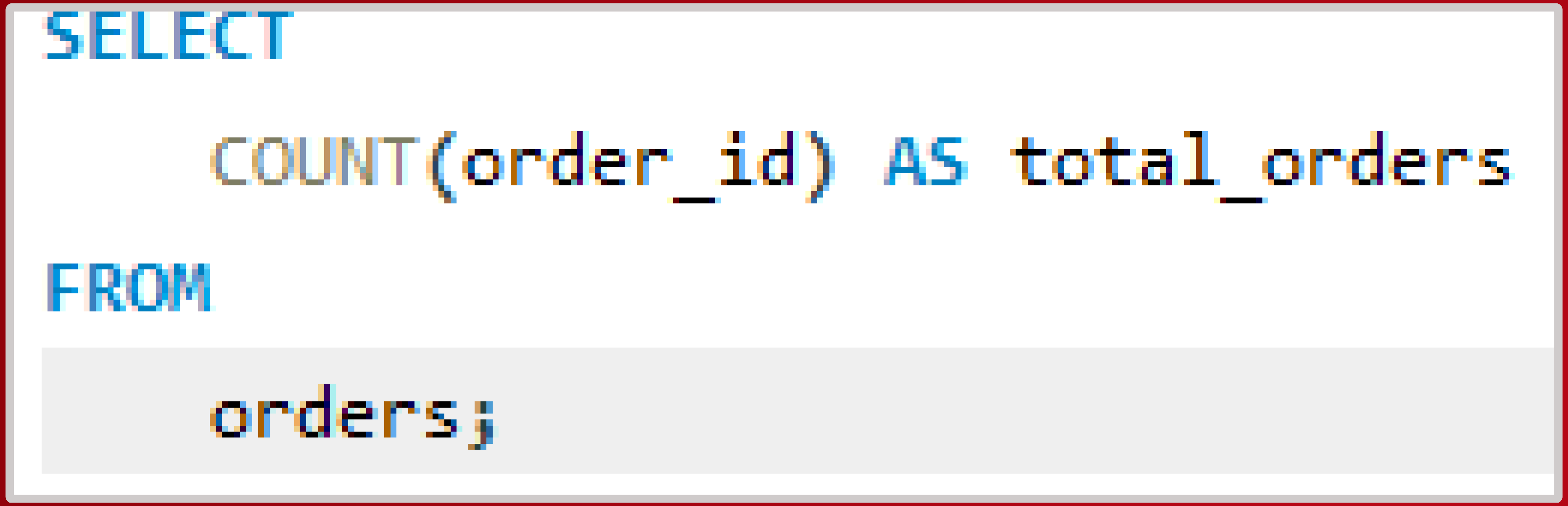
Intermidiate :

- Join the necessary tables to find the total quantity of each pizza category ordered.
- Determine the distribution of orders by hour of the day.
- Join relevant tables to find the category-wise distribution of pizzas.
- Group the orders by date and calculate the average number of pizzas ordered per day.
- Determine the top 3 most ordered pizza types based on revenue.
- Determine the top 3 Least ordered pizza types based on revenue.

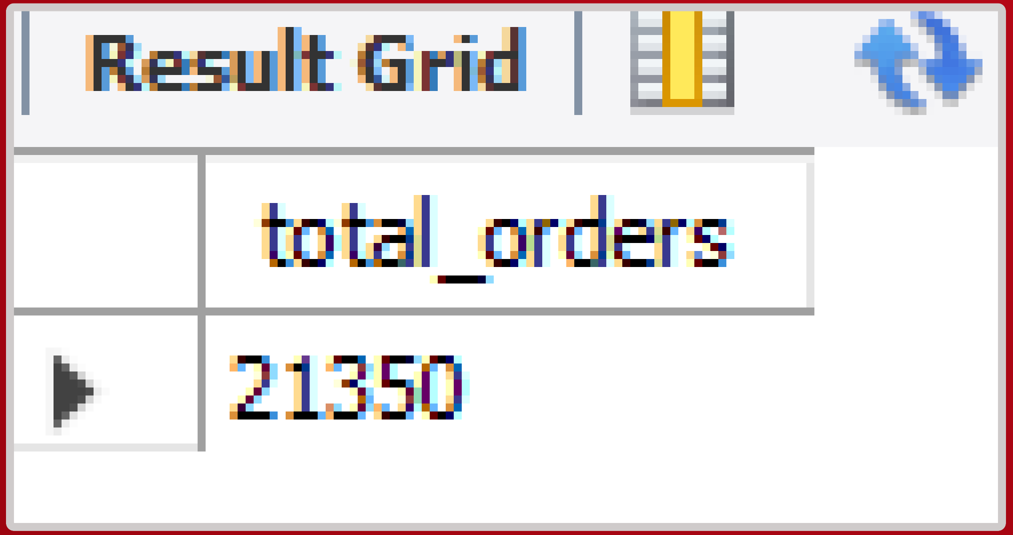
Advanced :

- Calculate the percentage contribution of each pizza type to total revenue.
- Analyze the cumulative revenue generated over time.
- Determine the top 3 most ordered pizza types based on revenue for each pizza category.

Retrieve the total number of orders placed

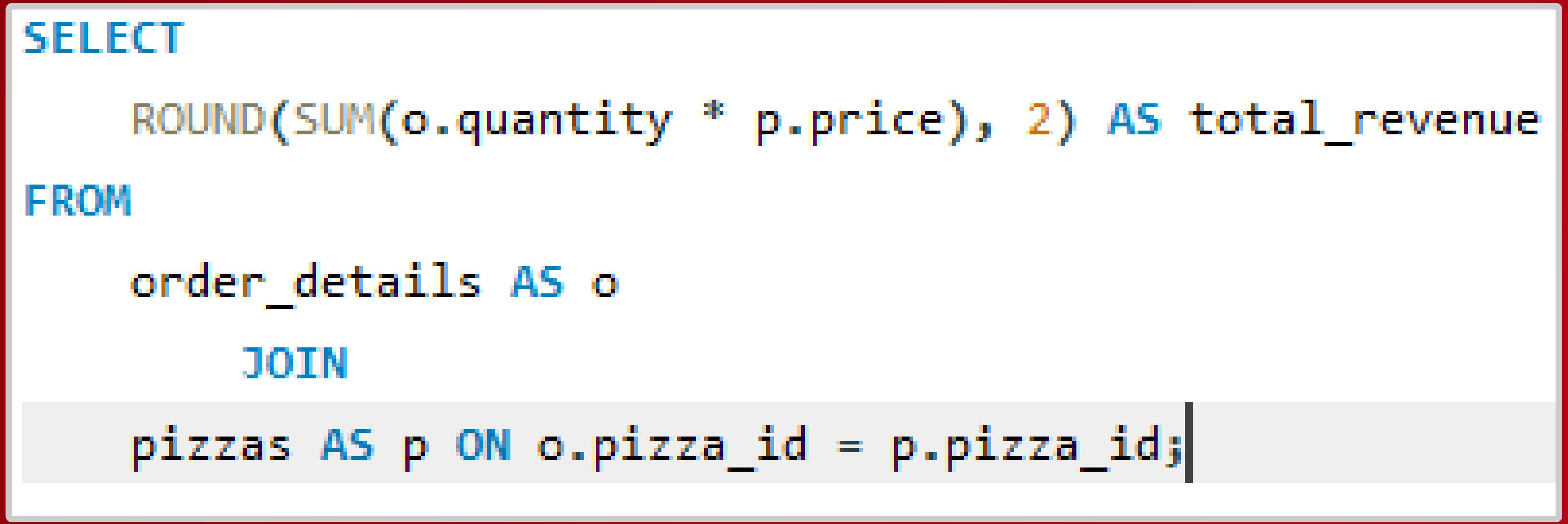


```
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

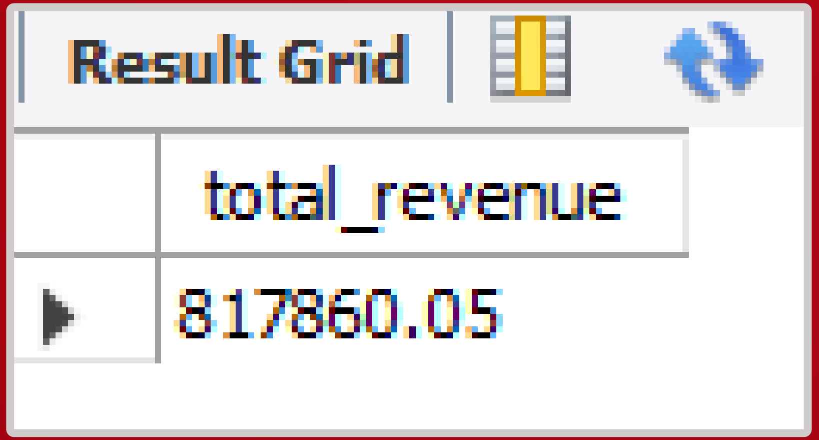


```
total_orders
21350
```

Calculate the total revenue generated from pizza sales.

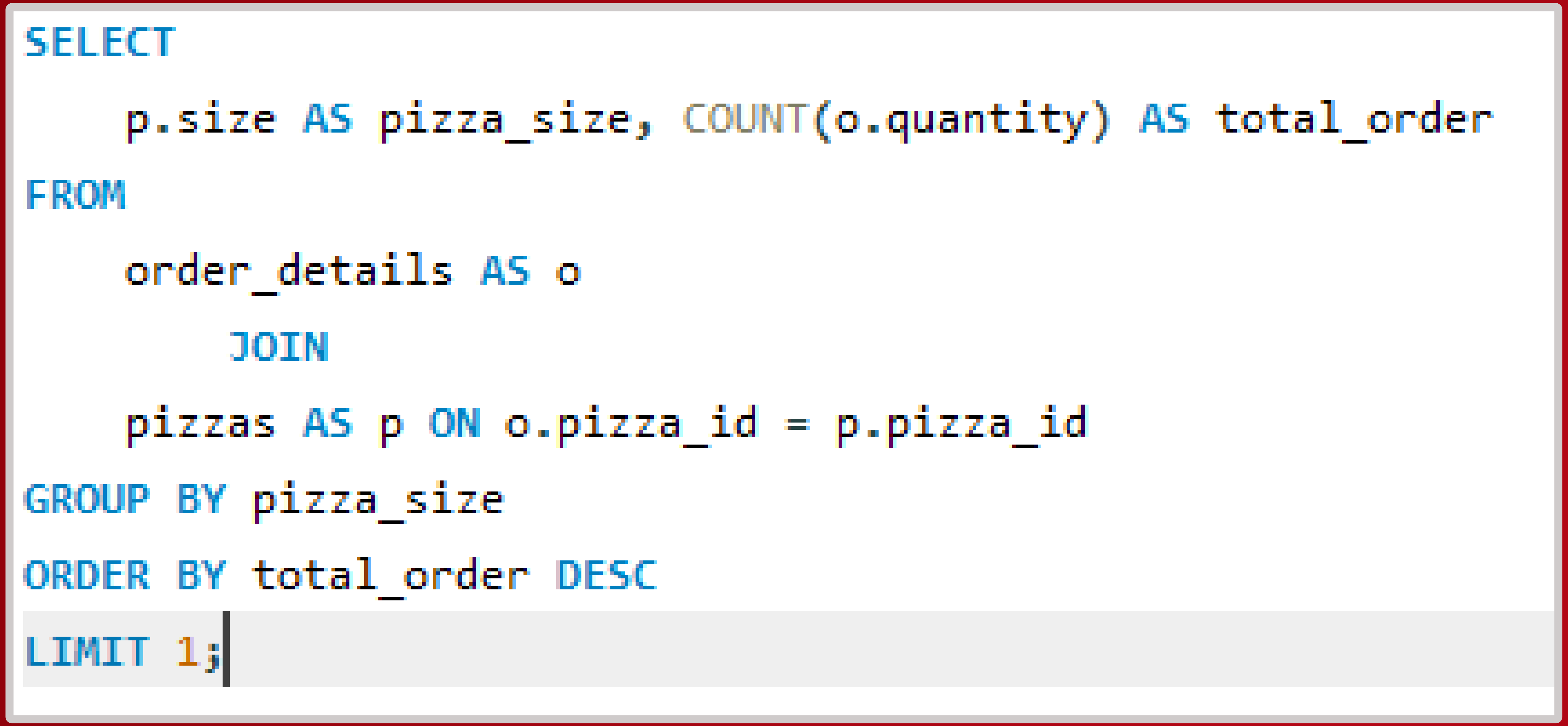


```
SELECT
    ROUND(SUM(o.quantity * p.price), 2) AS total_revenue
FROM
    order_details AS o
        JOIN
    pizzas AS p ON o.pizza_id = p.pizza_id;
```

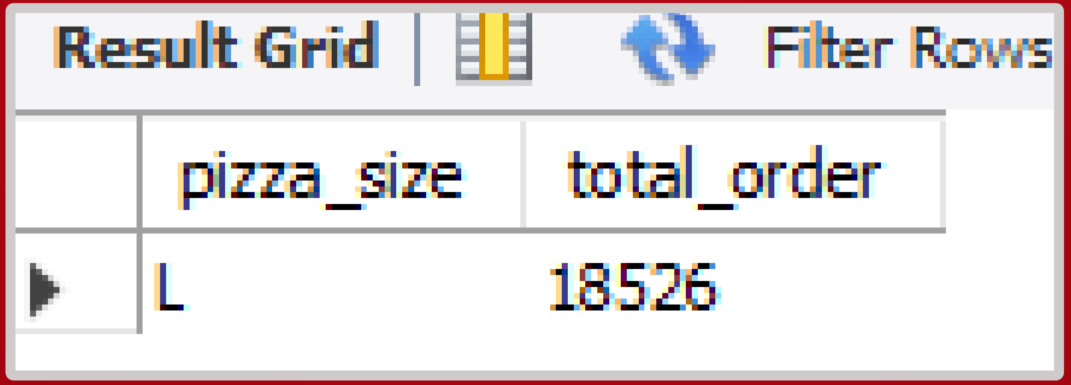


```
total_revenue
817860.05
```

Identify the most common pizza size ordered

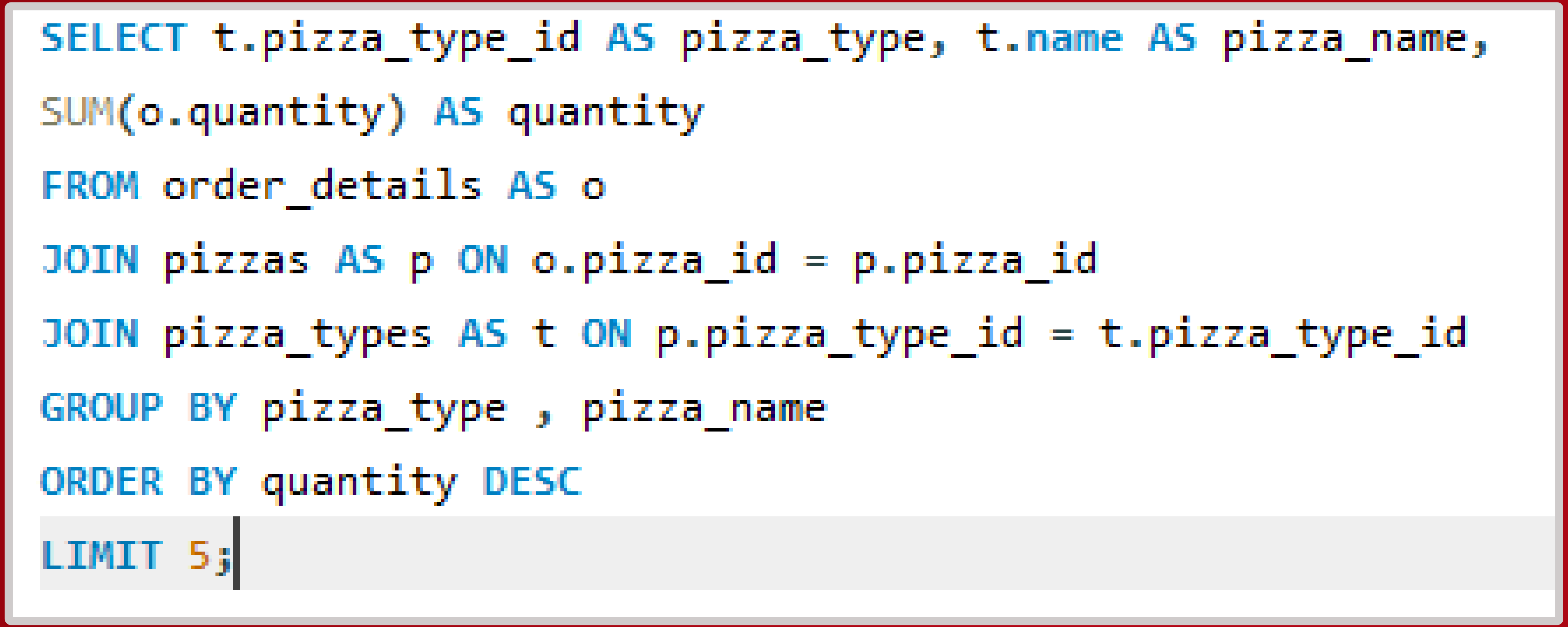


```
SELECT
    p.size AS pizza_size, COUNT(o.quantity) AS total_order
FROM
    order_details AS o
        JOIN
    pizzas AS p ON o.pizza_id = p.pizza_id
GROUP BY pizza_size
ORDER BY total_order DESC
LIMIT 1;
```

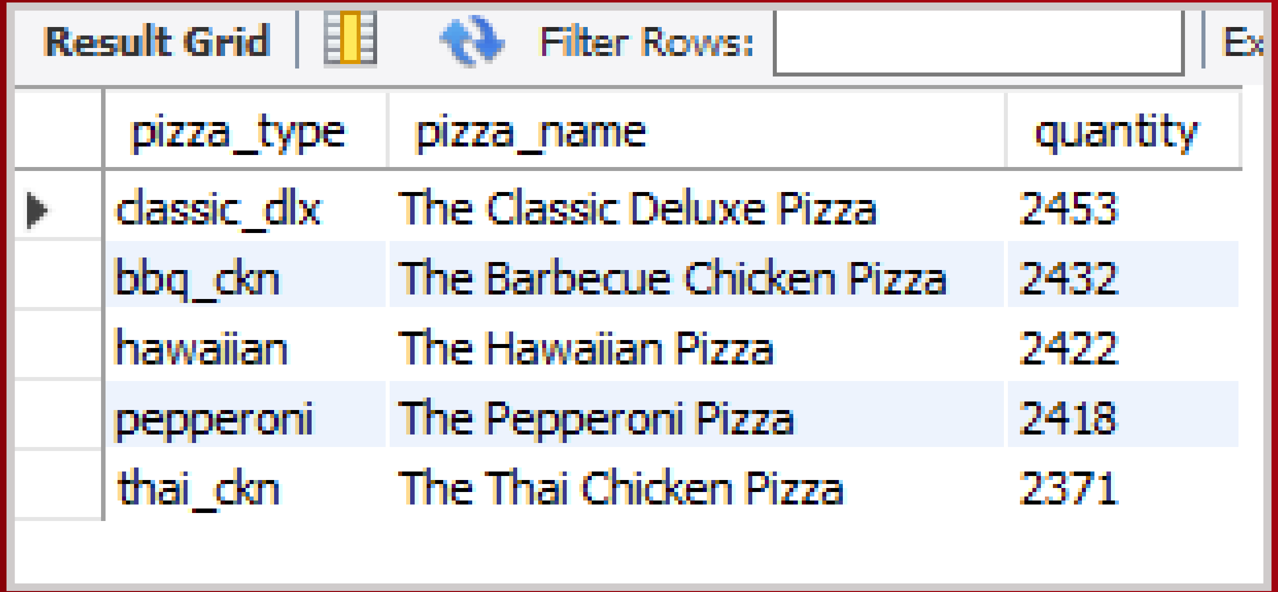


pizza_size	total_order
L	18526

List the top 5 most ordered pizza types along with their quantities

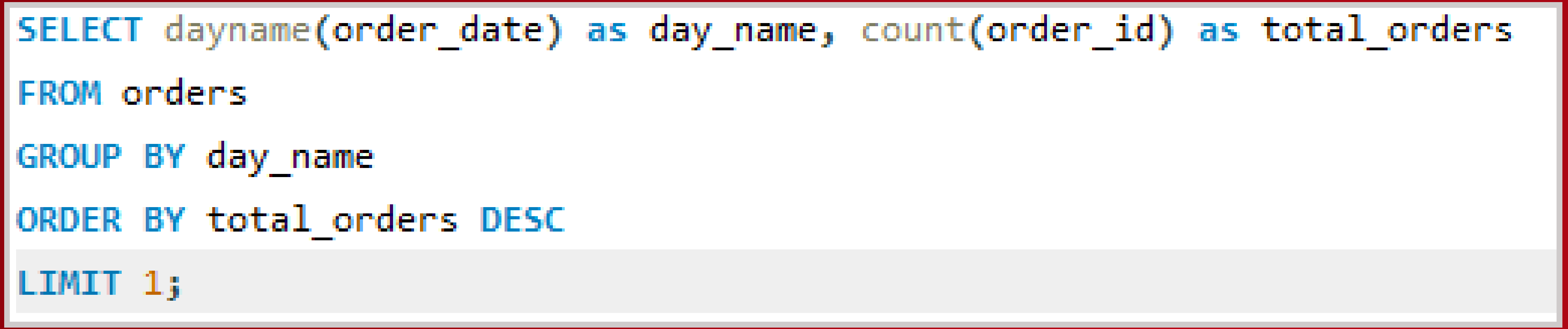


```sql
SELECT t.pizza_type_id AS pizza_type, t.name AS pizza_name,
SUM(o.quantity) AS quantity
FROM order_details AS o
JOIN pizzas AS p ON o.pizza_id = p.pizza_id
JOIN pizza_types AS t ON p.pizza_type_id = t.pizza_type_id
GROUP BY pizza_type , pizza_name
ORDER BY quantity DESC
LIMIT 5;
```

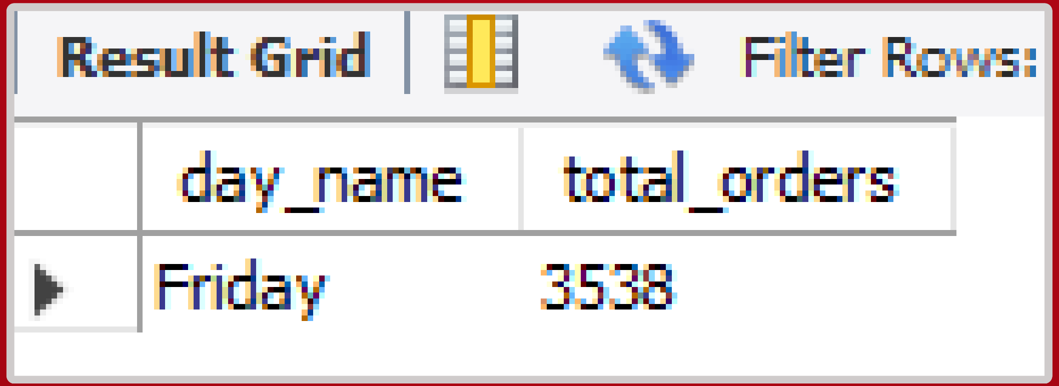


pizza_type	pizza_name	quantity
classic_dlx	The Classic Deluxe Pizza	2453
bbq_ckn	The Barbecue Chicken Pizza	2432
hawaiian	The Hawaiian Pizza	2422
pepperoni	The Pepperoni Pizza	2418
thai_ckn	The Thai Chicken Pizza	2371

which day of the week are the most orders placed

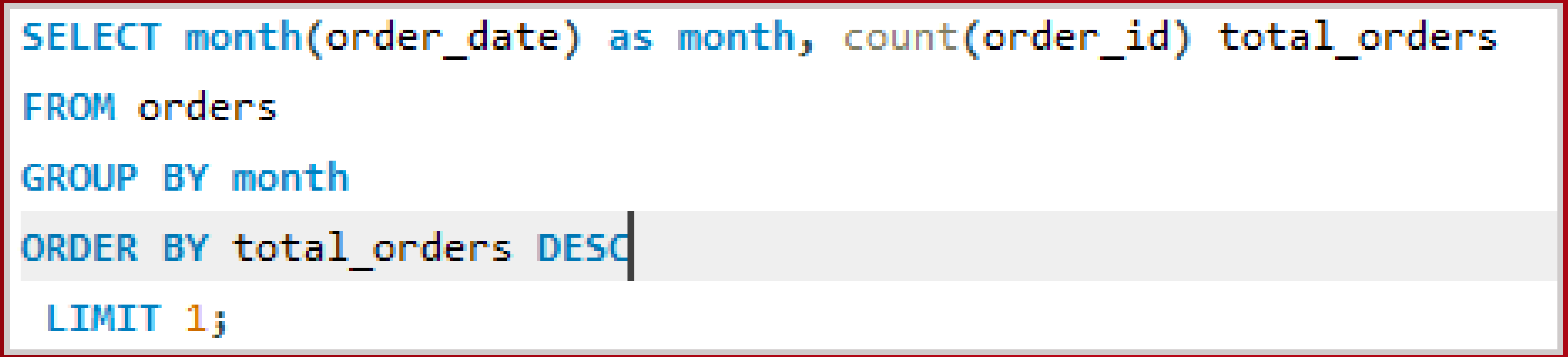


```
SELECT dayname(order_date) as day_name, count(order_id) as total_orders
FROM orders
GROUP BY day_name
ORDER BY total_orders DESC
LIMIT 1;
```

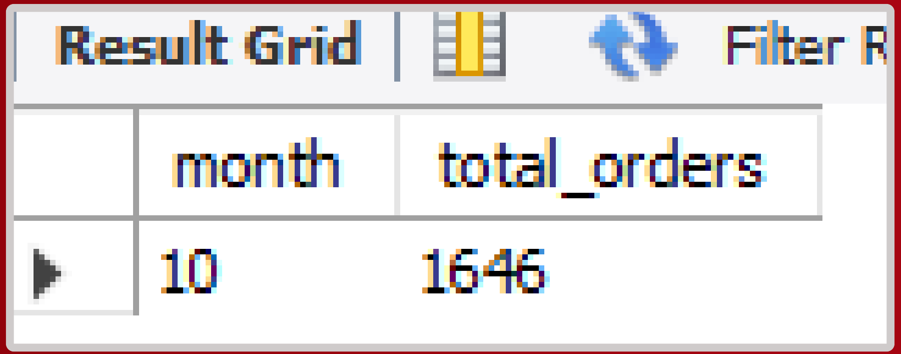


```
day_name   total_orders
Friday     3538
```

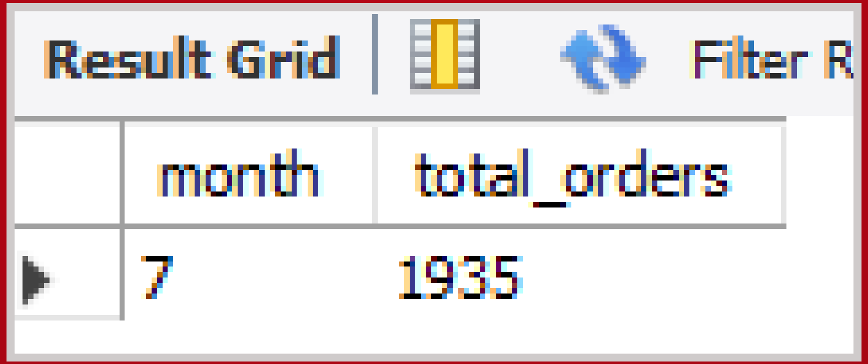
determine the best and worst month in terms of order volume



```
SELECT month(order_date) as month, count(order_id) total_orders
FROM orders
GROUP BY month
ORDER BY total_orders DESC
 LIMIT 1;
```

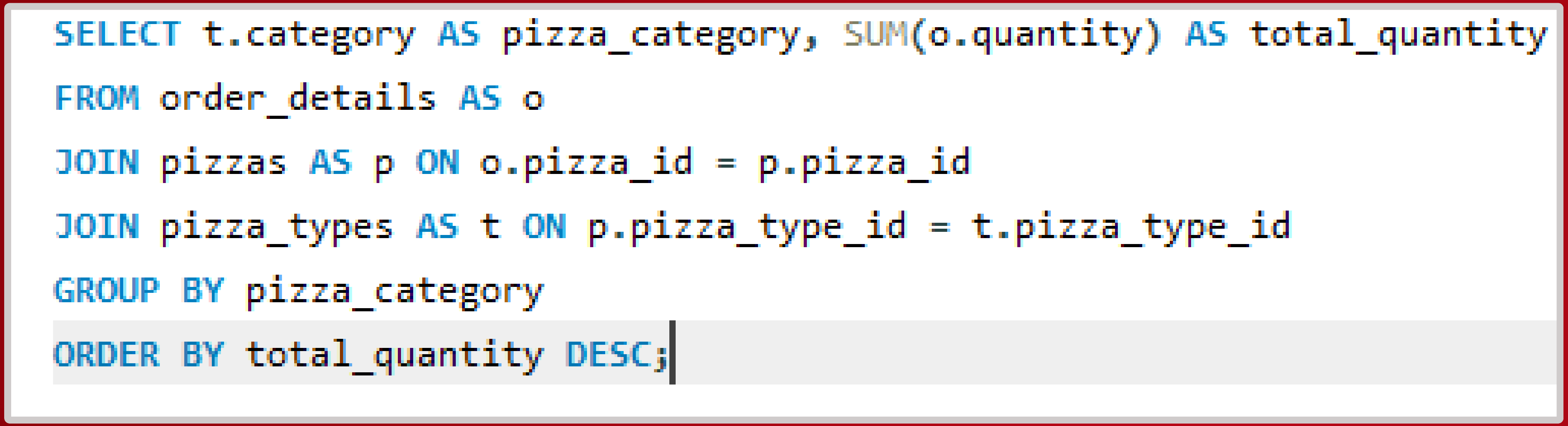


```
month   total_orders
10      1646
```

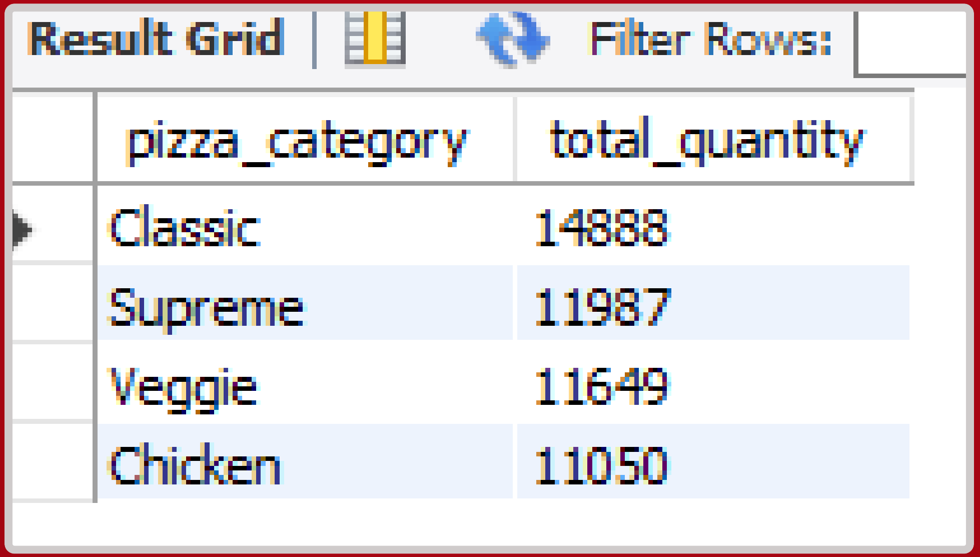


```
month   total_orders
7       1935
```

Join the necessary tables to find the total quantity of each pizza category ordered

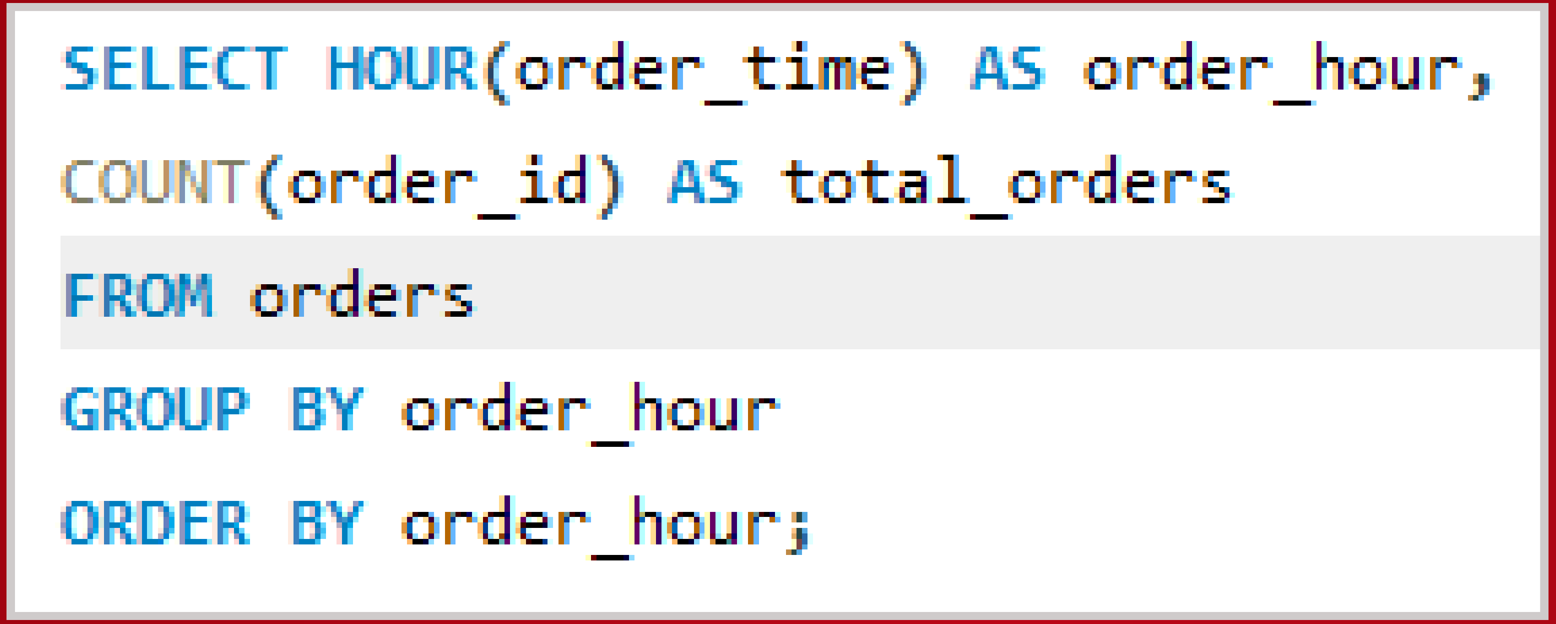


```
SELECT t.category AS pizza_category, SUM(o.quantity) AS total_quantity
FROM order_details AS o
JOIN pizzas AS p ON o.pizza_id = p.pizza_id
JOIN pizza_types AS t ON p.pizza_type_id = t.pizza_type_id
GROUP BY pizza_category
ORDER BY total_quantity DESC;
```

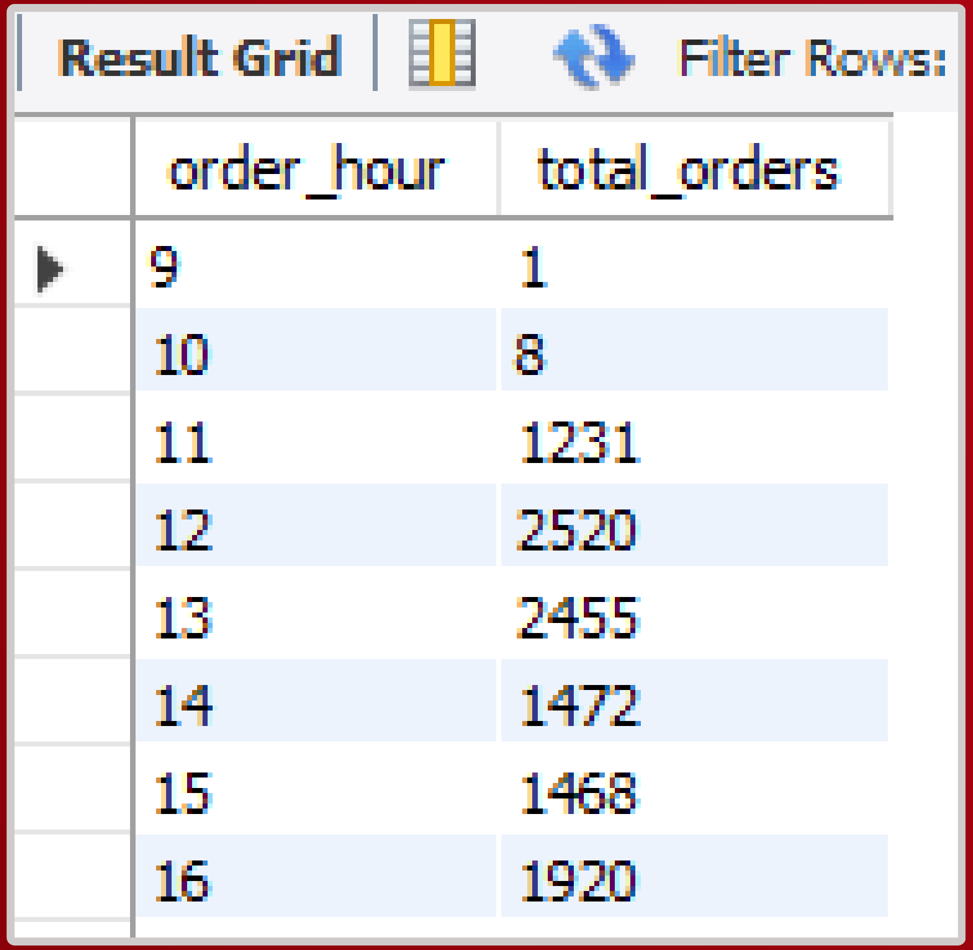


pizza_category	total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Determine the distribution of orders by hour of the day

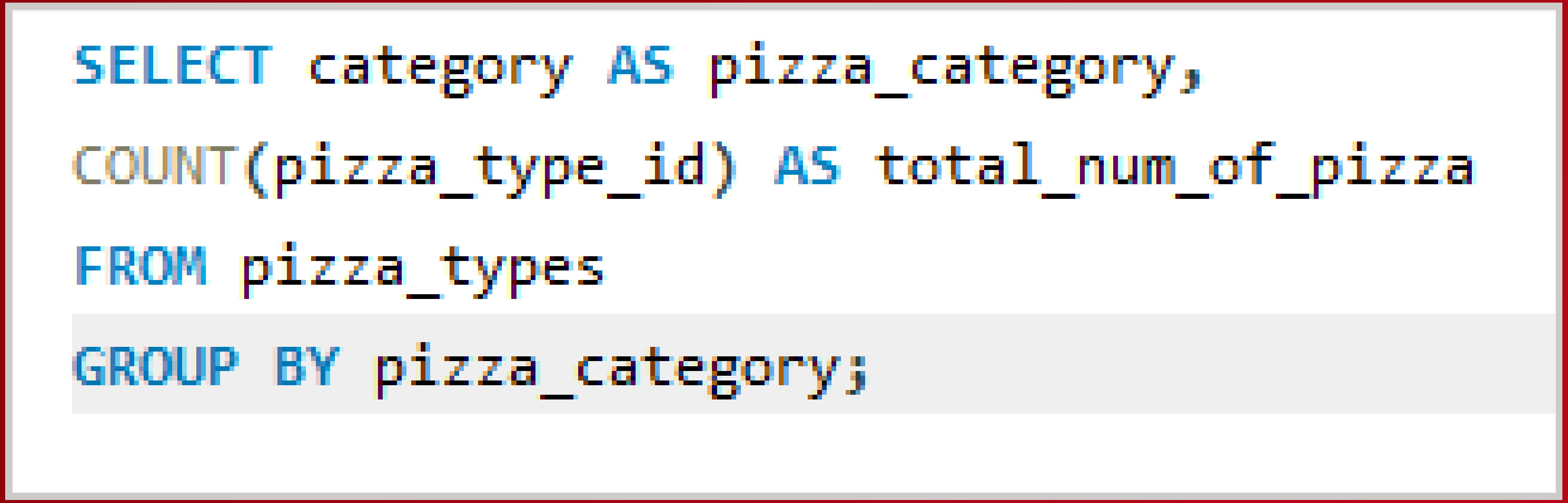


```
SELECT HOUR(order_time) AS order_hour,
COUNT(order_id) AS total_orders
FROM orders
GROUP BY order_hour
ORDER BY order_hour;
```

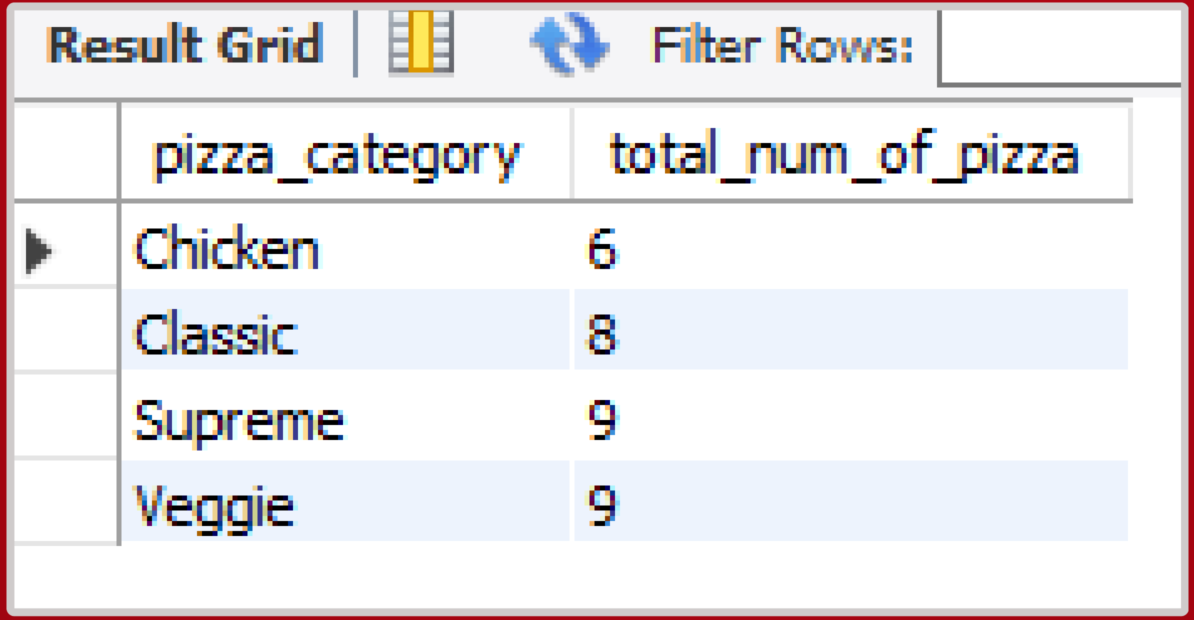


order_hour	total_orders
9	1
10	8
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920

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

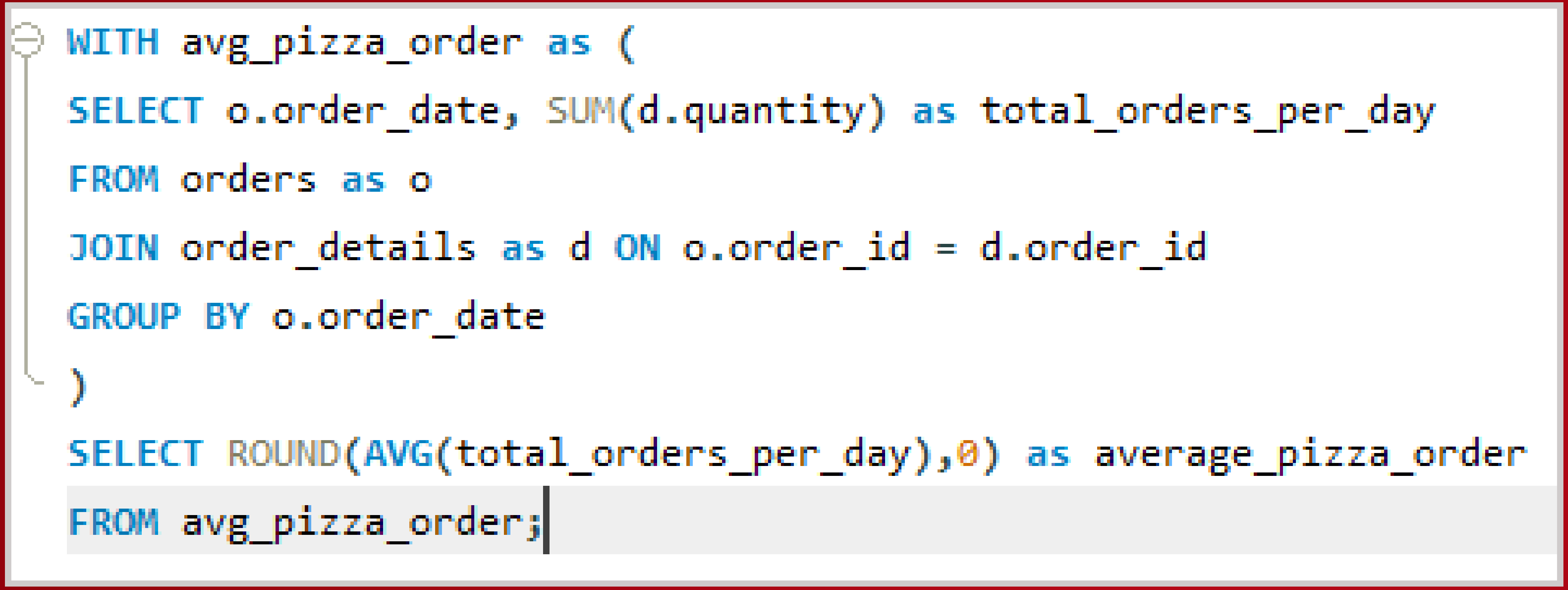


```
SELECT category AS pizza_category,
COUNT(pizza_type_id) AS total_num_of_pizza
FROM pizza_types
GROUP BY pizza_category;
```

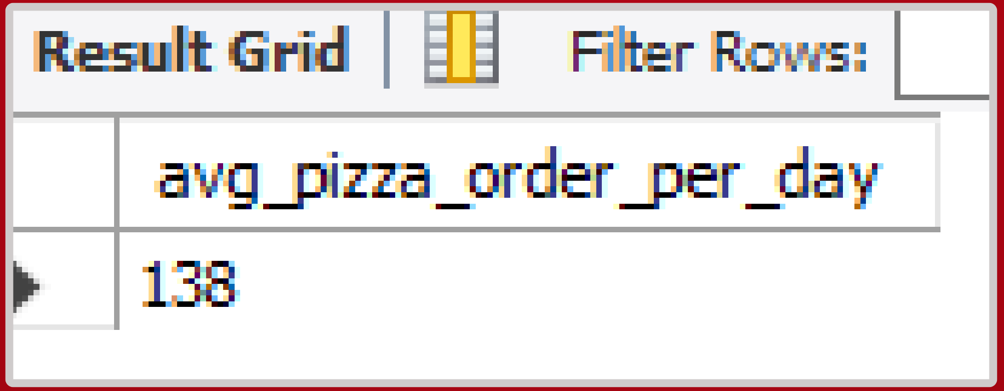


pizza_category	total_num_of_pizza
Chicken	6
Classic	8
Supreme	9
Veggie	9

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

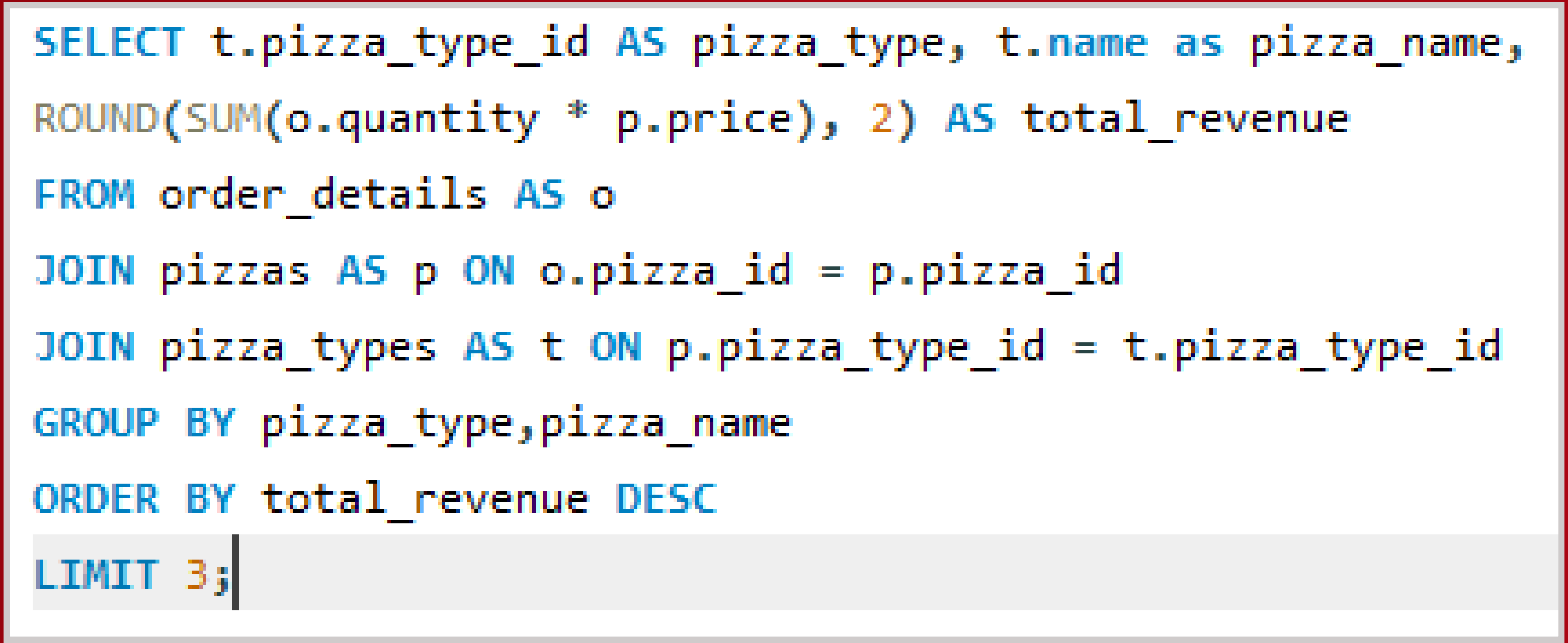


```
WITH avg_pizza_order as (
SELECT o.order_date, SUM(d.quantity) as total_orders_per_day
FROM orders as o
JOIN order_details as d ON o.order_id = d.order_id
GROUP BY o.order_date
)
SELECT ROUND(AVG(total_orders_per_day),0) as average_pizza_order
FROM avg_pizza_order;
```

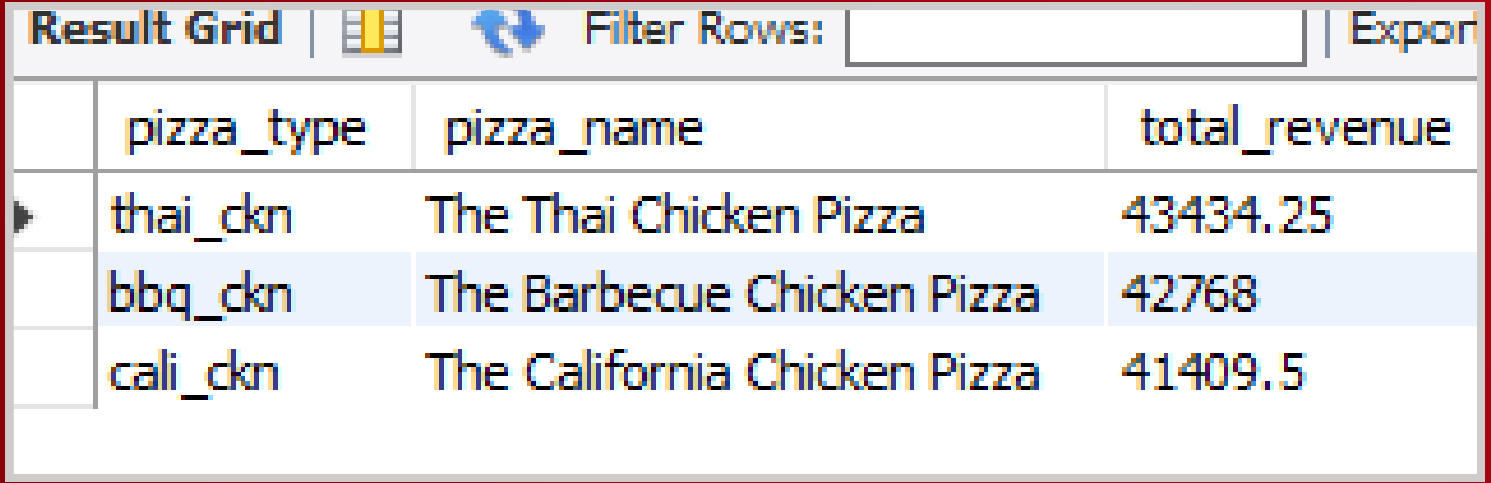


```
avg_pizza_order_per_day
138
```

Determine the top 3 most ordered pizza types based on revenue



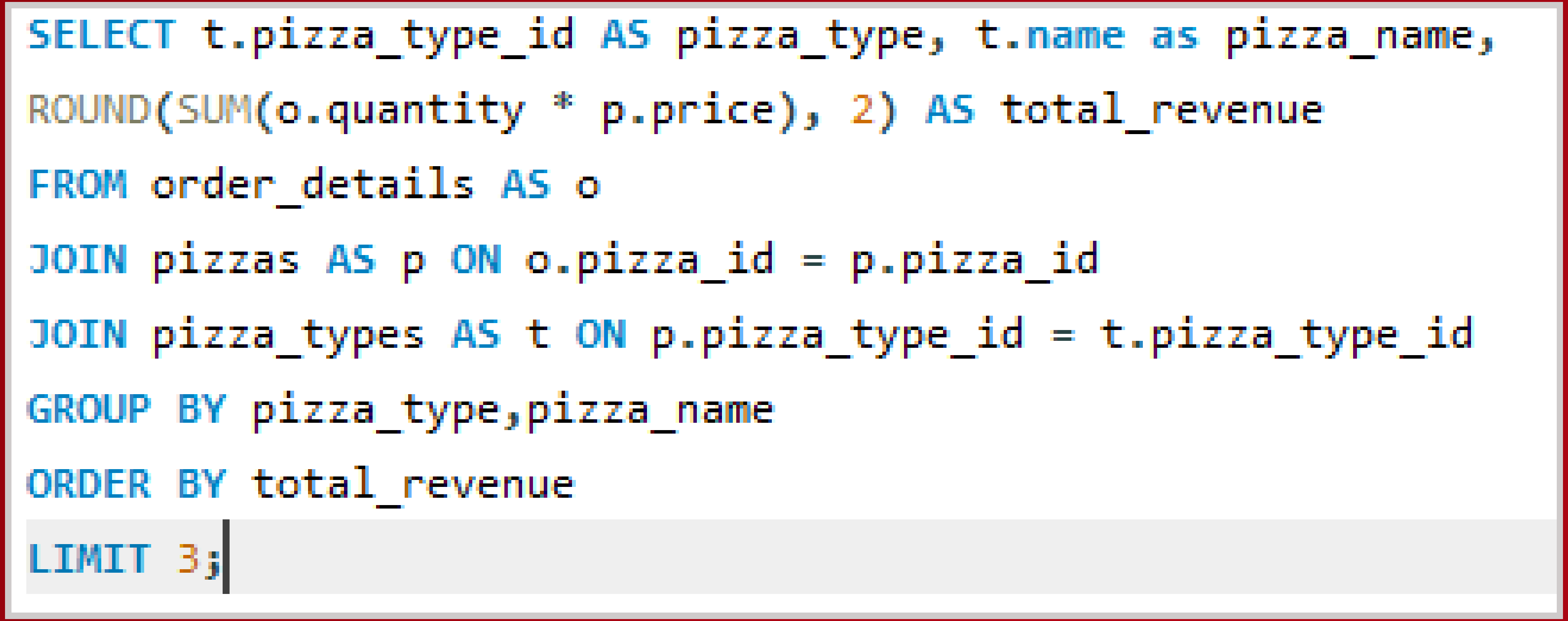
```
SELECT t.pizza_type_id AS pizza_type, t.name as pizza_name,
ROUND(SUM(o.quantity * p.price), 2) AS total_revenue
FROM order_details AS o
JOIN pizzas AS p ON o.pizza_id = p.pizza_id
JOIN pizza_types AS t ON p.pizza_type_id = t.pizza_type_id
GROUP BY pizza_type,pizza_name
ORDER BY total_revenue DESC
LIMIT 3;
```



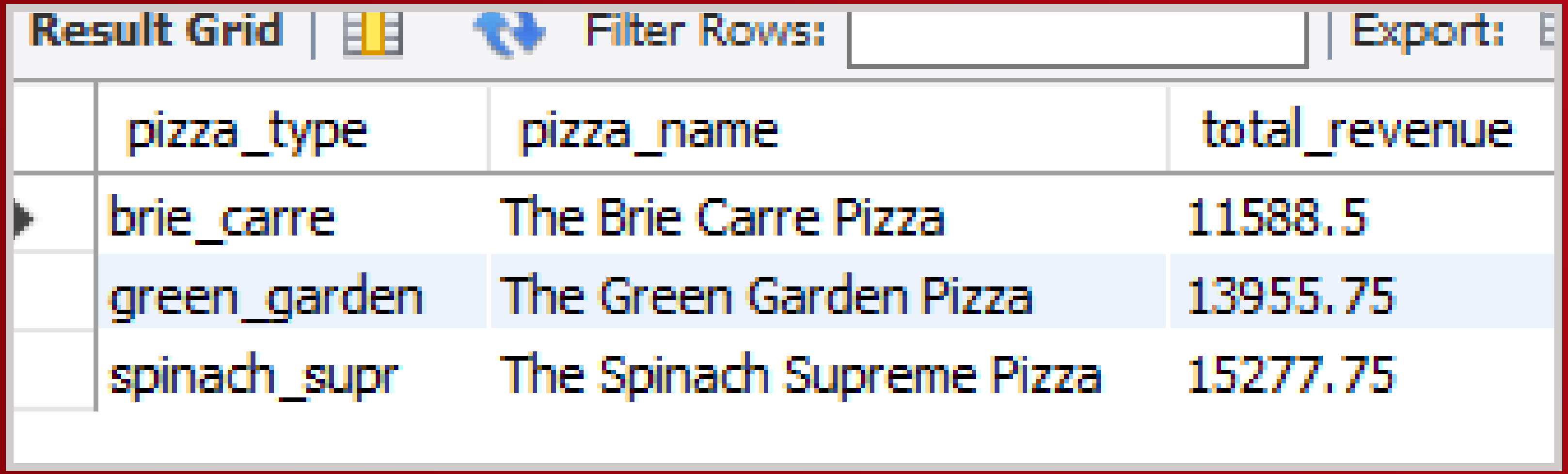
Result Grid | Filter Rows: | Expor

pizza_type	pizza_name	total_revenue
thai_ckn	The Thai Chicken Pizza	43434.25
bbq_ckn	The Barbecue Chicken Pizza	42768
cali_ckn	The California Chicken Pizza	41409.5

Determine the top 3 Least ordered pizza types based on revenue

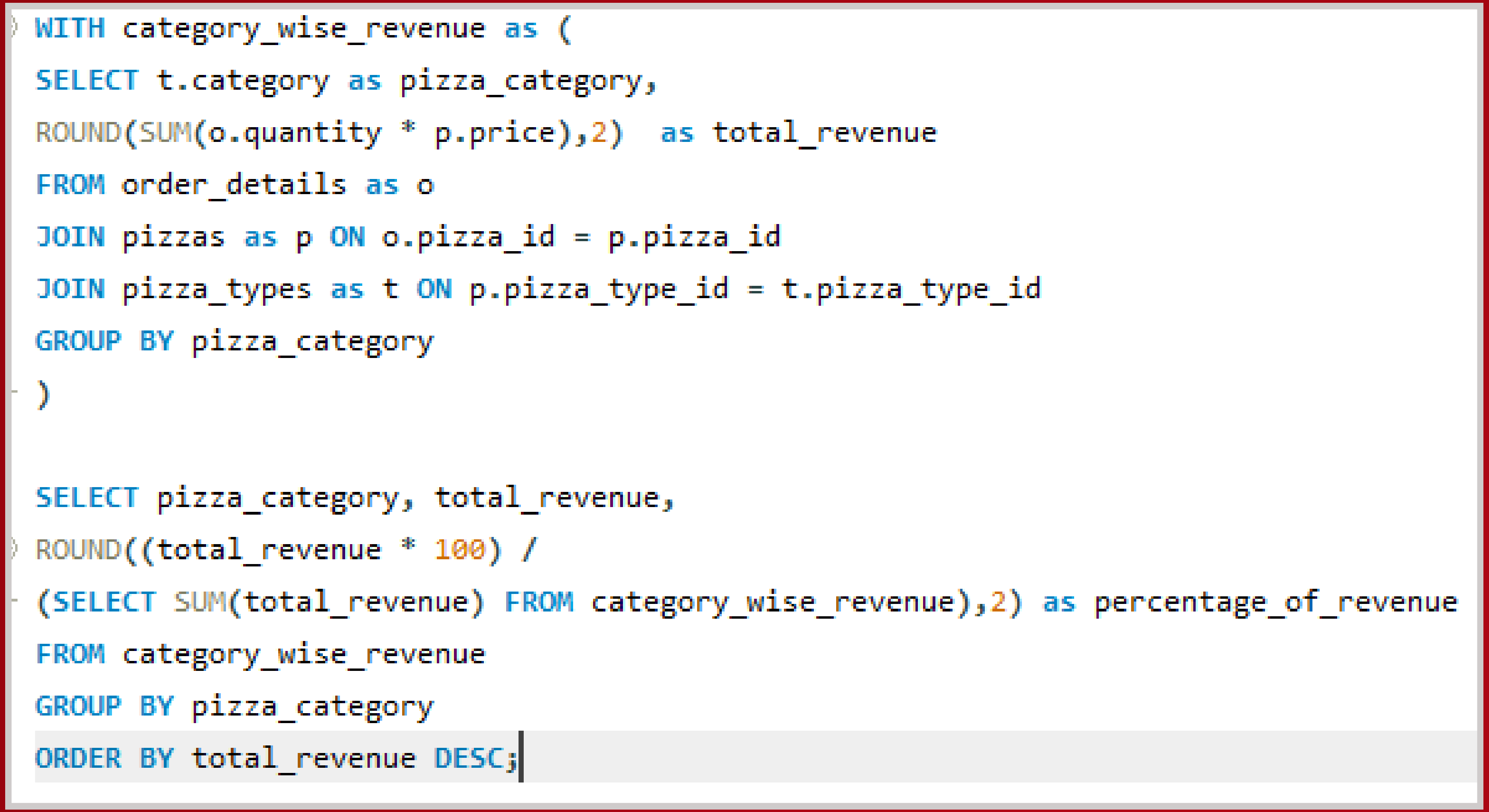


```
SELECT t.pizza_type_id AS pizza_type, t.name as pizza_name,
ROUND(SUM(o.quantity * p.price), 2) AS total_revenue
FROM order_details AS o
JOIN pizzas AS p ON o.pizza_id = p.pizza_id
JOIN pizza_types AS t ON p.pizza_type_id = t.pizza_type_id
GROUP BY pizza_type,pizza_name
ORDER BY total_revenue
LIMIT 3;
```

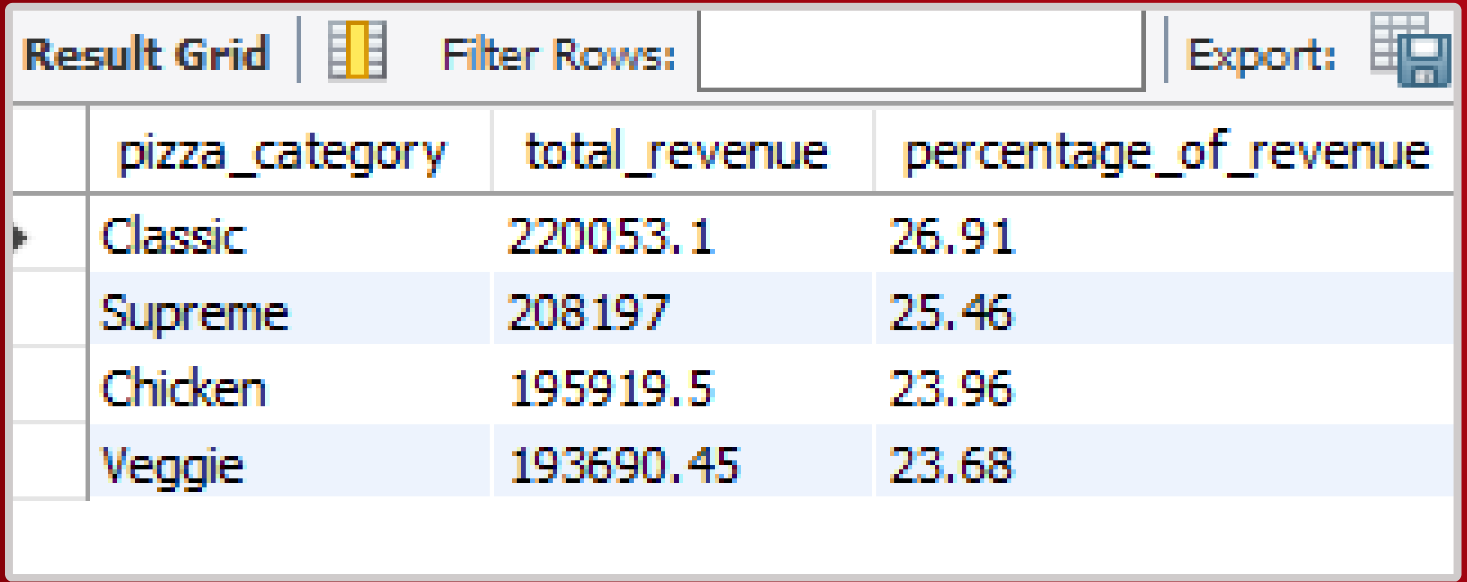


pizza_type	pizza_name	total_revenue
brie_carre	The Brie Carre Pizza	11588.5
green_garden	The Green Garden Pizza	13955.75
spinach_supr	The Spinach Supreme Pizza	15277.75

Calculate the percentage contribution of each pizza type to total revenue



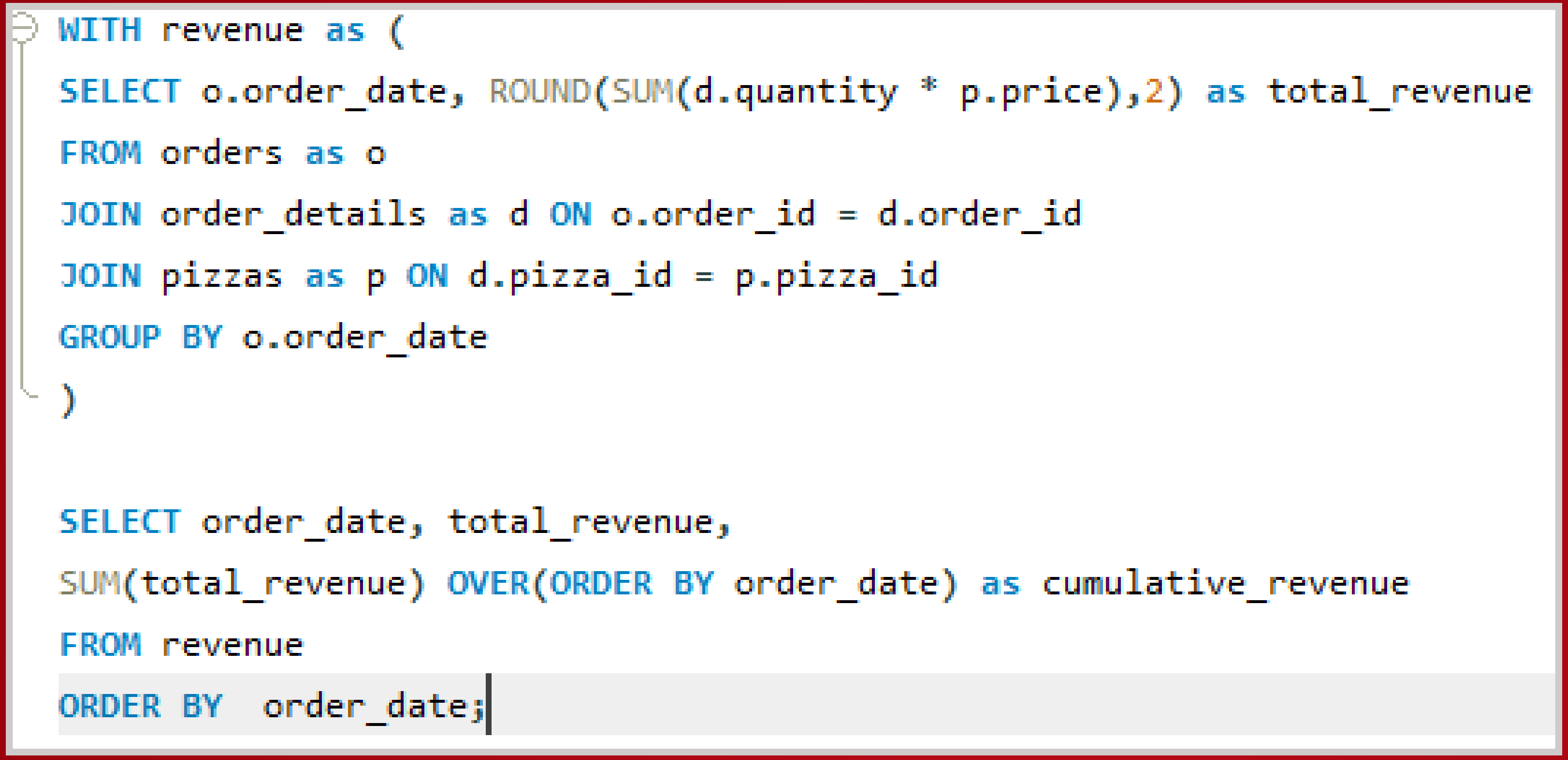
```
WITH category_wise_revenue as (
SELECT t.category as pizza_category,
ROUND(SUM(o.quantity * p.price),2)  as total_revenue
FROM order_details as o
JOIN pizzas as p ON o.pizza_id = p.pizza_id
JOIN pizza_types as t ON p.pizza_type_id = t.pizza_type_id
GROUP BY pizza_category
)

SELECT pizza_category, total_revenue,
ROUND((total_revenue * 100) /
(SELECT SUM(total_revenue) FROM category_wise_revenue),2) as percentage_of_revenue
FROM category_wise_revenue
GROUP BY pizza_category
ORDER BY total_revenue DESC;
```



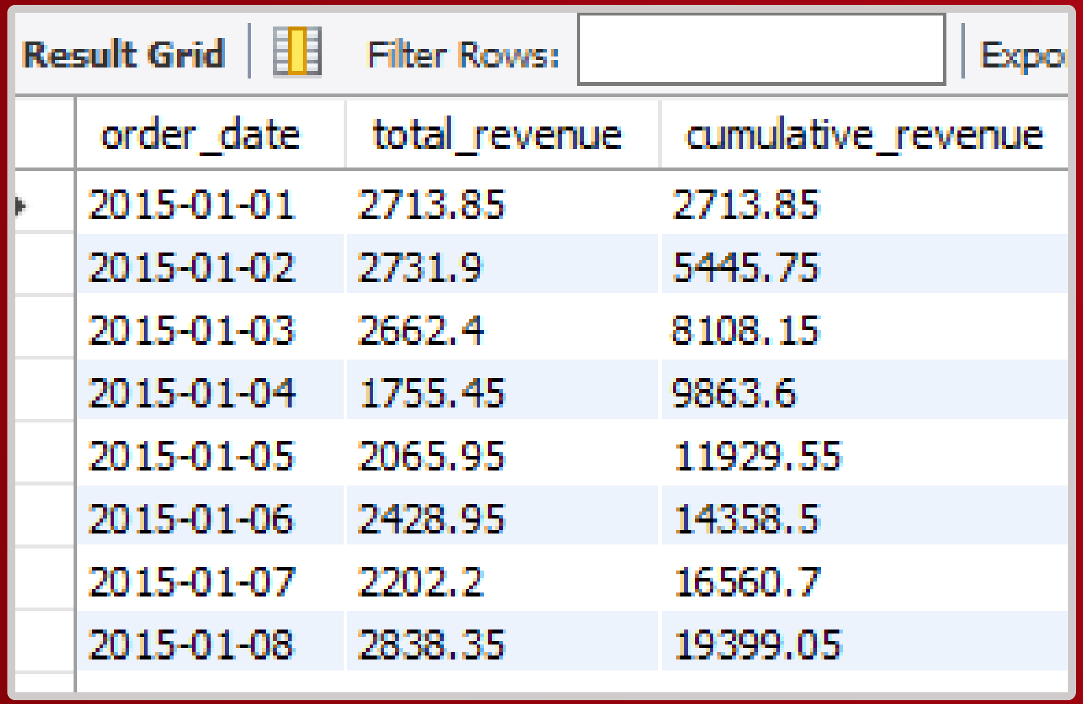
Result Grid | Filter Rows: | Export:

pizza_category	total_revenue	percentage_of_revenue
Classic	220053.1	26.91
Supreme	208197	25.46
Chicken	195919.5	23.96
Veggie	193690.45	23.68

Analyze the cumulative revenue generated over time



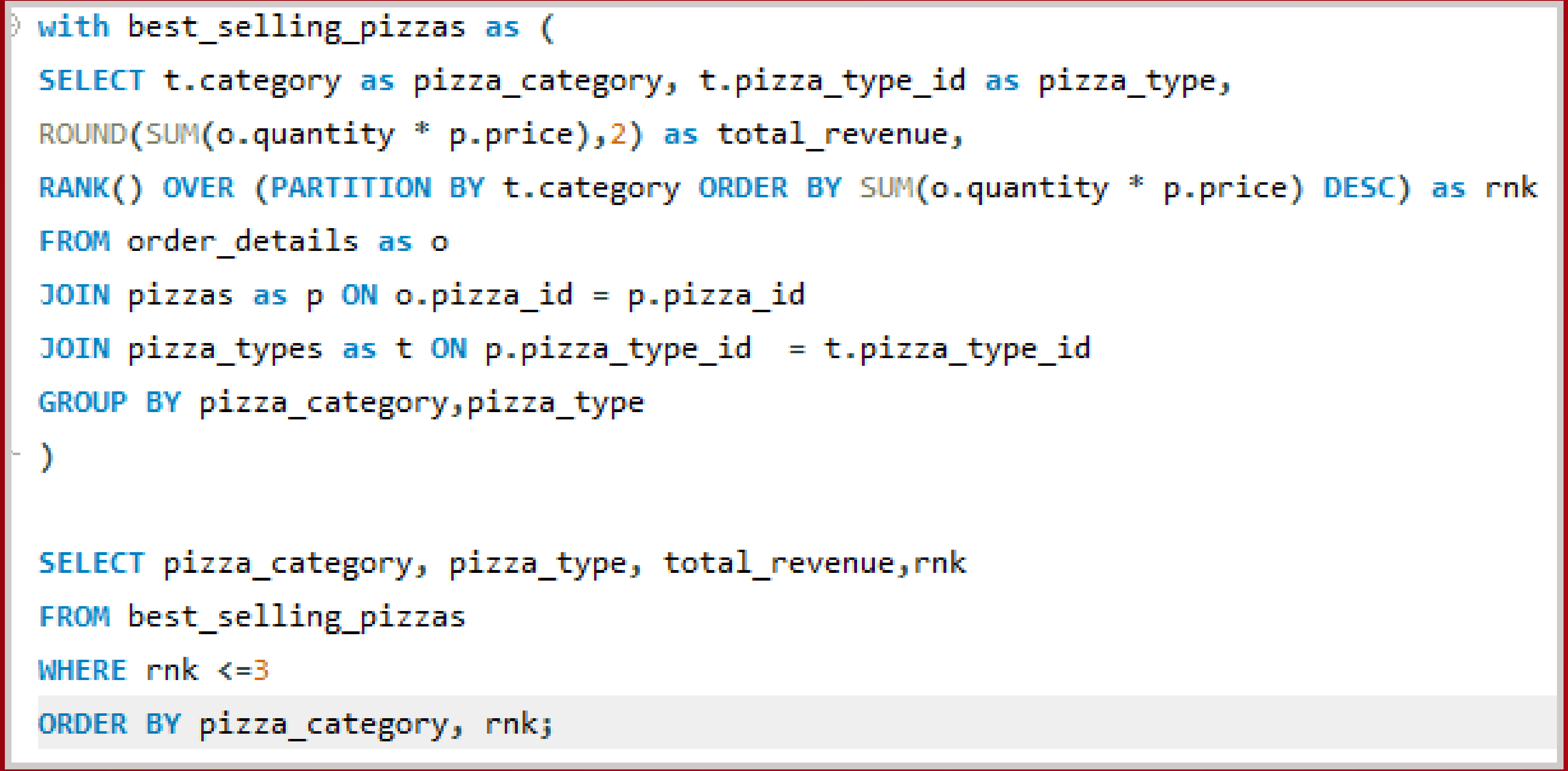
```
WITH revenue as (
SELECT o.order_date, ROUND(SUM(d.quantity * p.price),2) as total_revenue
FROM orders as o
JOIN order_details as d ON o.order_id = d.order_id
JOIN pizzas as p ON d.pizza_id = p.pizza_id
GROUP BY o.order_date
)

SELECT order_date, total_revenue,
SUM(total_revenue) OVER(ORDER BY order_date) as cumulative_revenue
FROM revenue
ORDER BY  order_date;
```



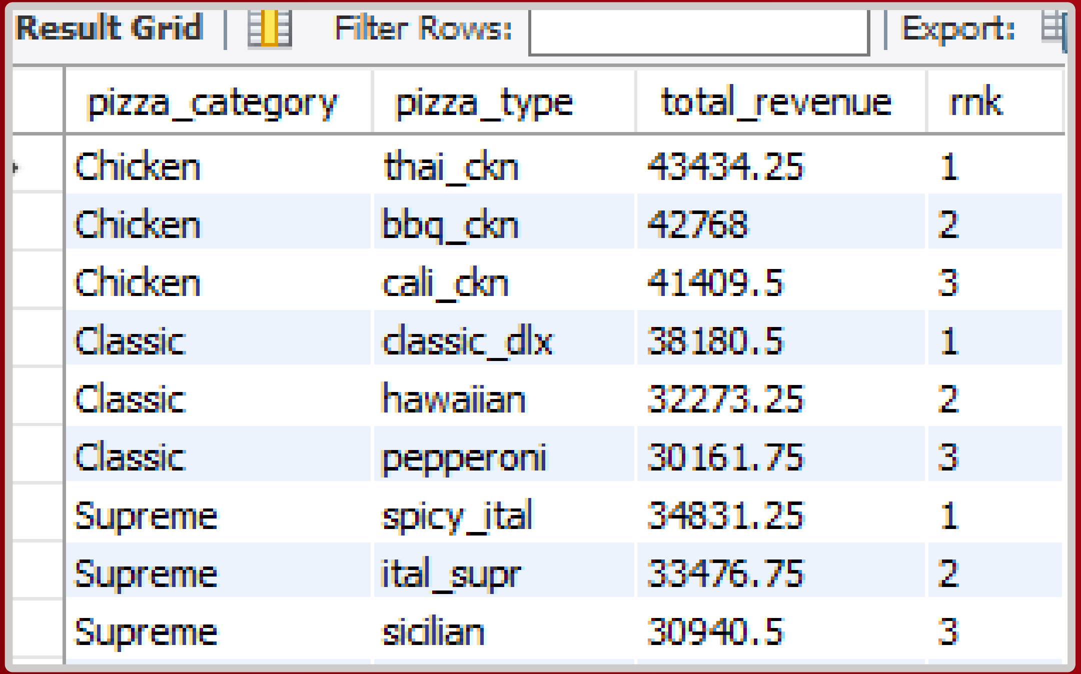
Result Grid | Filter Rows: | Expo

order_date	total_revenue	cumulative_revenue
2015-01-01	2713.85	2713.85
2015-01-02	2731.9	5445.75
2015-01-03	2662.4	8108.15
2015-01-04	1755.45	9863.6
2015-01-05	2065.95	11929.55
2015-01-06	2428.95	14358.5
2015-01-07	2202.2	16560.7
2015-01-08	2838.35	19399.05

Determine the top 3 most ordered pizza types based on revenue for each pizza category



```
with best_selling_pizzas as (
SELECT t.category as pizza_category, t.pizza_type_id as pizza_type,
ROUND(SUM(o.quantity * p.price),2) as total_revenue,
RANK() OVER (PARTITION BY t.category ORDER BY SUM(o.quantity * p.price) DESC) as rnk
FROM order_details as o
JOIN pizzas as p ON o.pizza_id = p.pizza_id
JOIN pizza_types as t ON p.pizza_type_id  = t.pizza_type_id
GROUP BY pizza_category,pizza_type
)

SELECT pizza_category, pizza_type, total_revenue,rnk
FROM best_selling_pizzas
WHERE rnk <=3
ORDER BY pizza_category, rnk;
```



pizza_category	pizza_type	total_revenue	rnk
Chicken	thai_ckn	43434.25	1
Chicken	bbq_ckn	42768	2
Chicken	cali_ckn	41409.5	3
Classic	classic_dlx	38180.5	1
Classic	hawaiian	32273.25	2
Classic	pepperoni	30161.75	3
Supreme	spicy_ital	34831.25	1
Supreme	ital_supr	33476.75	2
Supreme	sicilian	30940.5	3

Key Insights and Recommendations

- Feature the top 3 pizzas that generate the highest revenue Use them in combo deals, digital ads, or social media Encourage staff to upsell these popular items. Review pizzas with the lowest revenue. Consider improving recipes, changing names, or removing them.
- Friday is the busiest day of the week, with 3,538 orders. Offer Friday-specific promotions or discounts to capitalize on the busiest day of the week.
- Large pizzas are the most popular size, with 18526 ordered. Consider offering discounts or promotions for ordering larger sizes, enticing customers to upgrade.
- July had the highest order volume (1,935 orders), while October had the lowest (1,646 orders). Analyze seasonal factors that affect order volume. Plan marketing campaigns or seasonal specials to boost sales during slower months like October.
- Based on order patterns, peak hours should be staffed well. Introduce "Happy Hour" discounts during low-traffic times.

THANK YOU