Borcelle
PIZZA RESTAURANT

Analyzing pizza sales

AN END-TO-END SQL PROJECT

Transforming Raw Data
Into Business Intelligence

BY SARVESH TIWARI

Project Overview

Introduction

This project explores the analysis of pizza sales data using SQL to derive actionable insights. Our goal is to uncover key trends, understand customer preferences, and optimize inventory management to enhance overall business performance.

Objectives

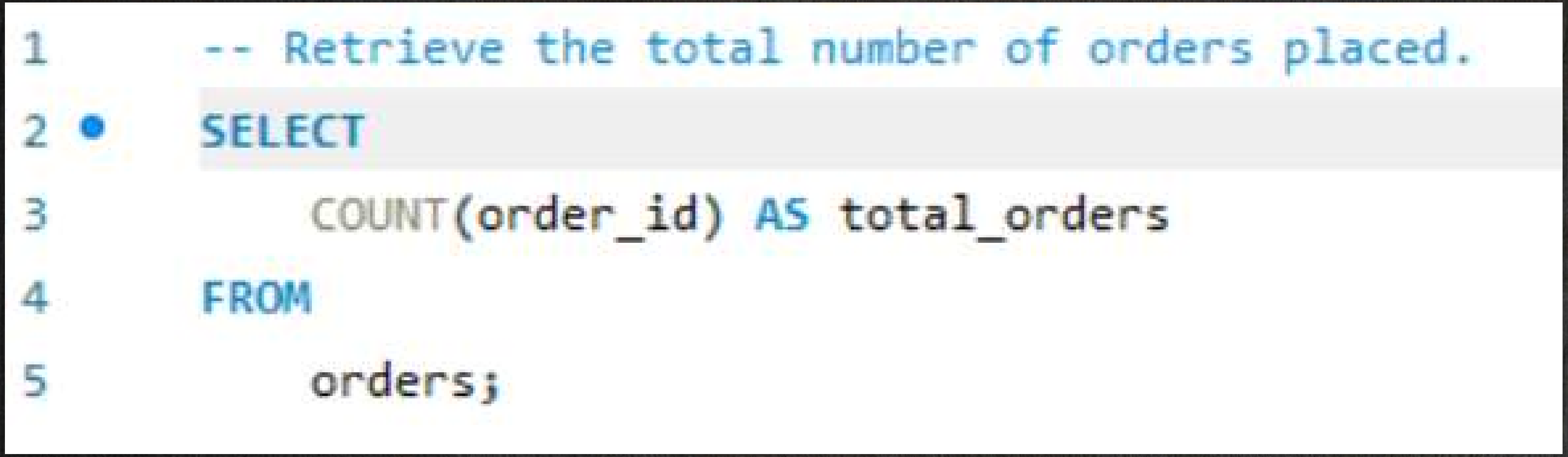
- Identify Sales Trends: Examine sales data to recognize patterns and peak periods in pizza sales.
- Understand Customer Preferences: Analyze customer data to determine popular pizza varieties and purchasing behaviors.
- Optimize Inventory Management: Use sales data to improve inventory practices and reduce waste.
- Enhance Business Strategies: Leverage insights to refine marketing strategies and drive sales growth.

Queries

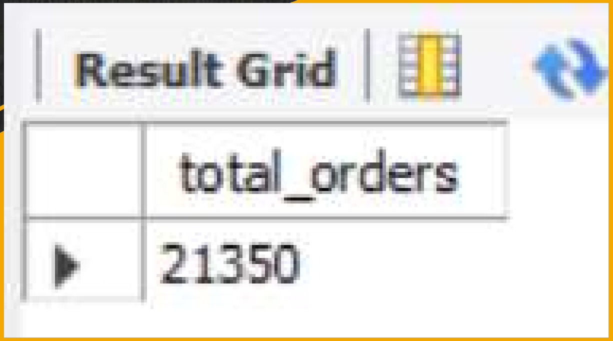
Basic Queries:

- Retrieve the total number of orders placed.
- Calculate the total revenue generated from pizza sales.
- Identify the highest-priced pizza.
- Identify the most common pizza size ordered.
- List the top 5 most ordered pizza types along with their quantities.

1. Retrieve the total number of orders placed.

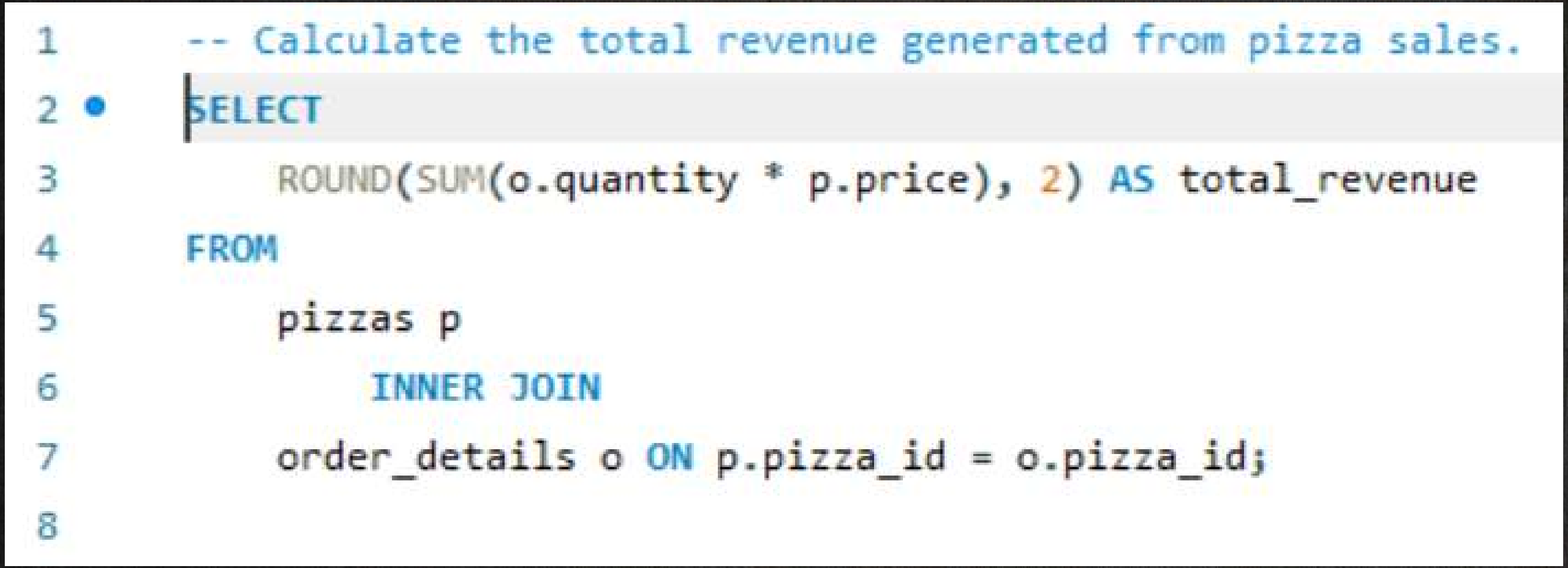


```sql
-- Retrieve the total number of orders placed.
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

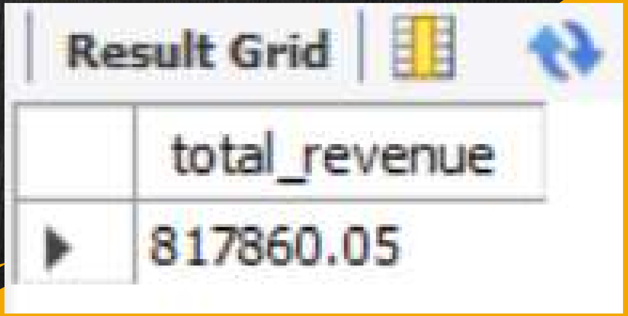


```
total_orders
21350
```

2. Calculate the total revenue generated from pizza sales.

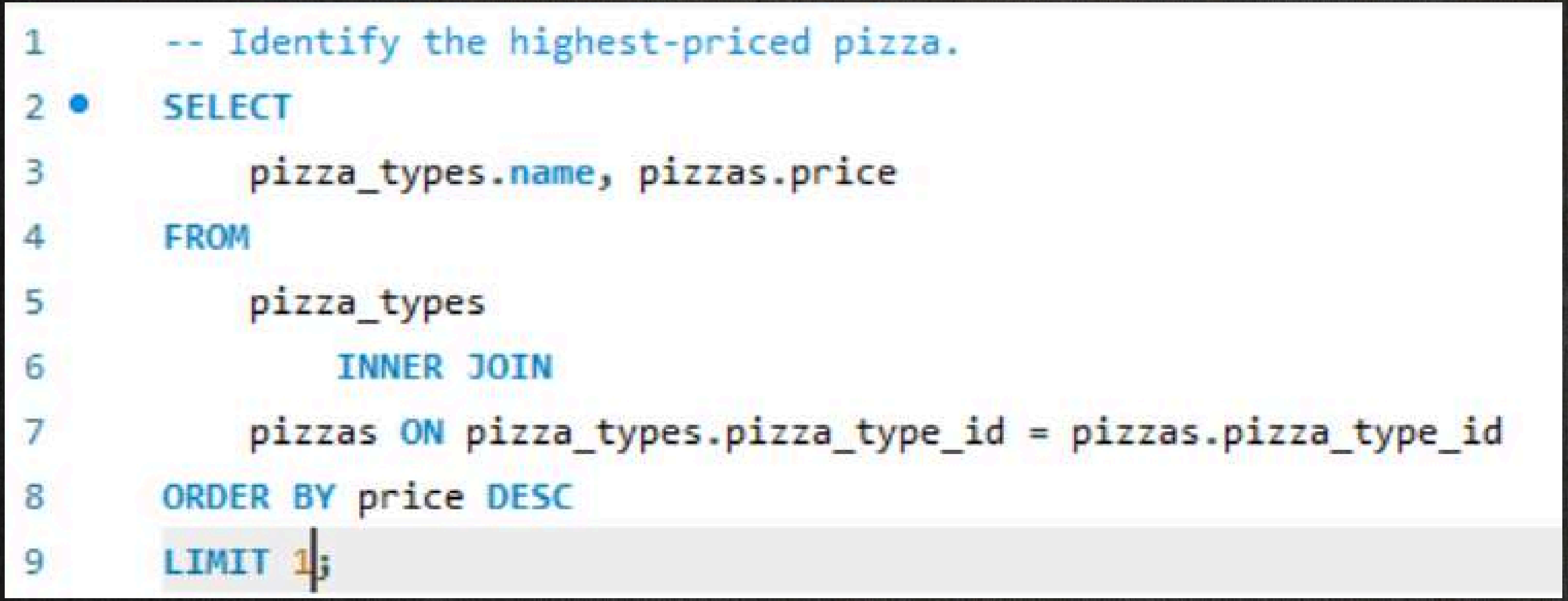


```sql
-- Calculate the total revenue generated from pizza sales.
SELECT
    ROUND(SUM(o.quantity * p.price), 2) AS total_revenue
FROM
    pizzas p
        INNER JOIN
    order_details o ON p.pizza_id = o.pizza_id;
```

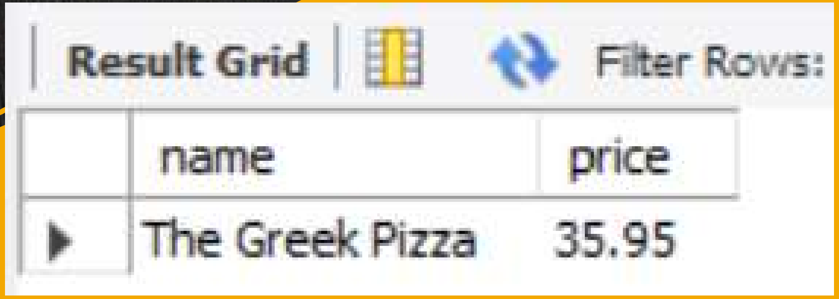


```
total_revenue
817860.05
```

3. Identify the highest-priced pizza.



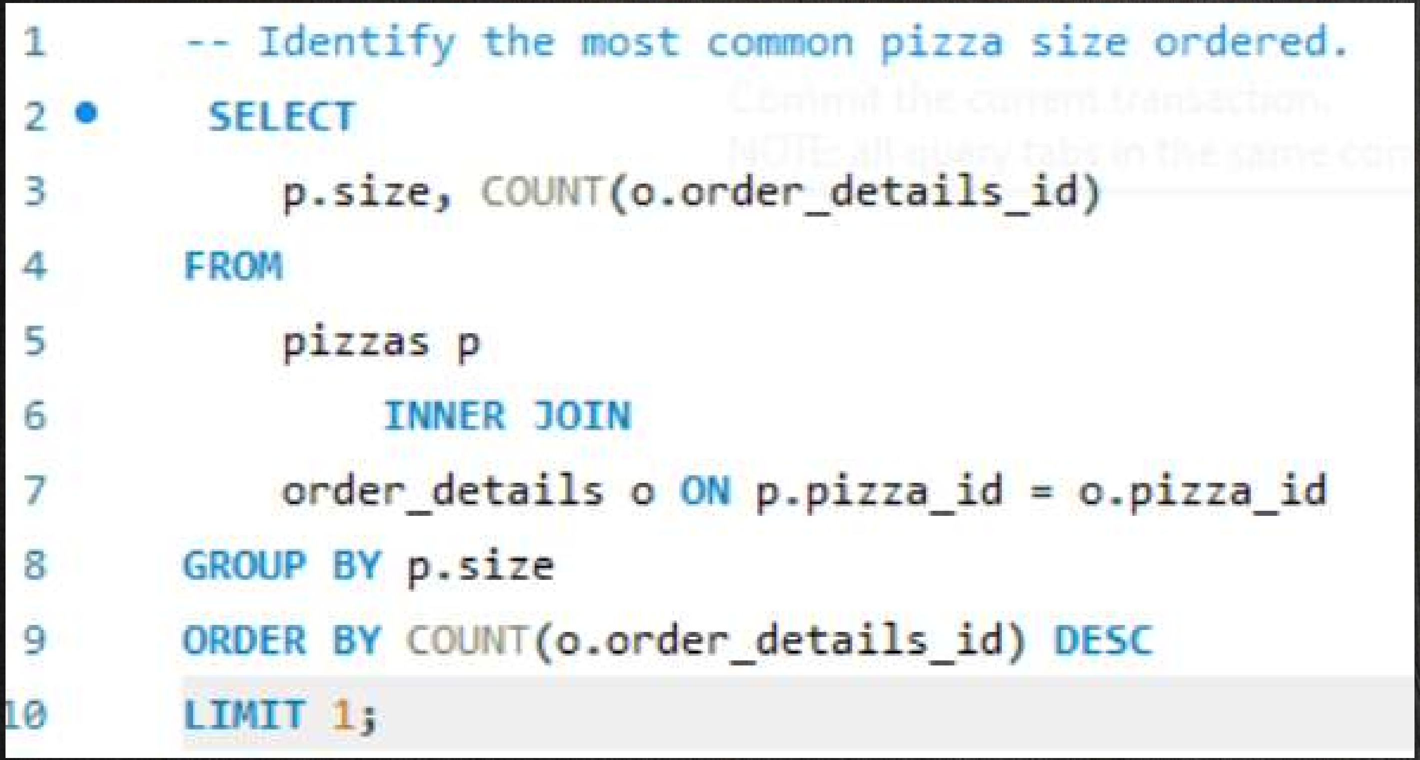
```
-- Identify the highest-priced pizza.
SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
        INNER JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY price DESC
LIMIT 1;
```



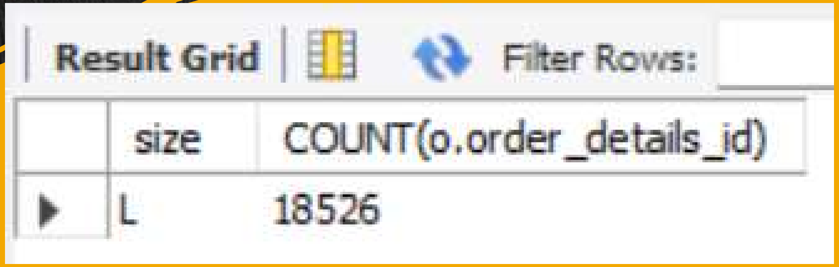
Result Grid | Filter Rows:

name	price
The Greek Pizza	35.95

4. Identify the most common pizza size ordered.

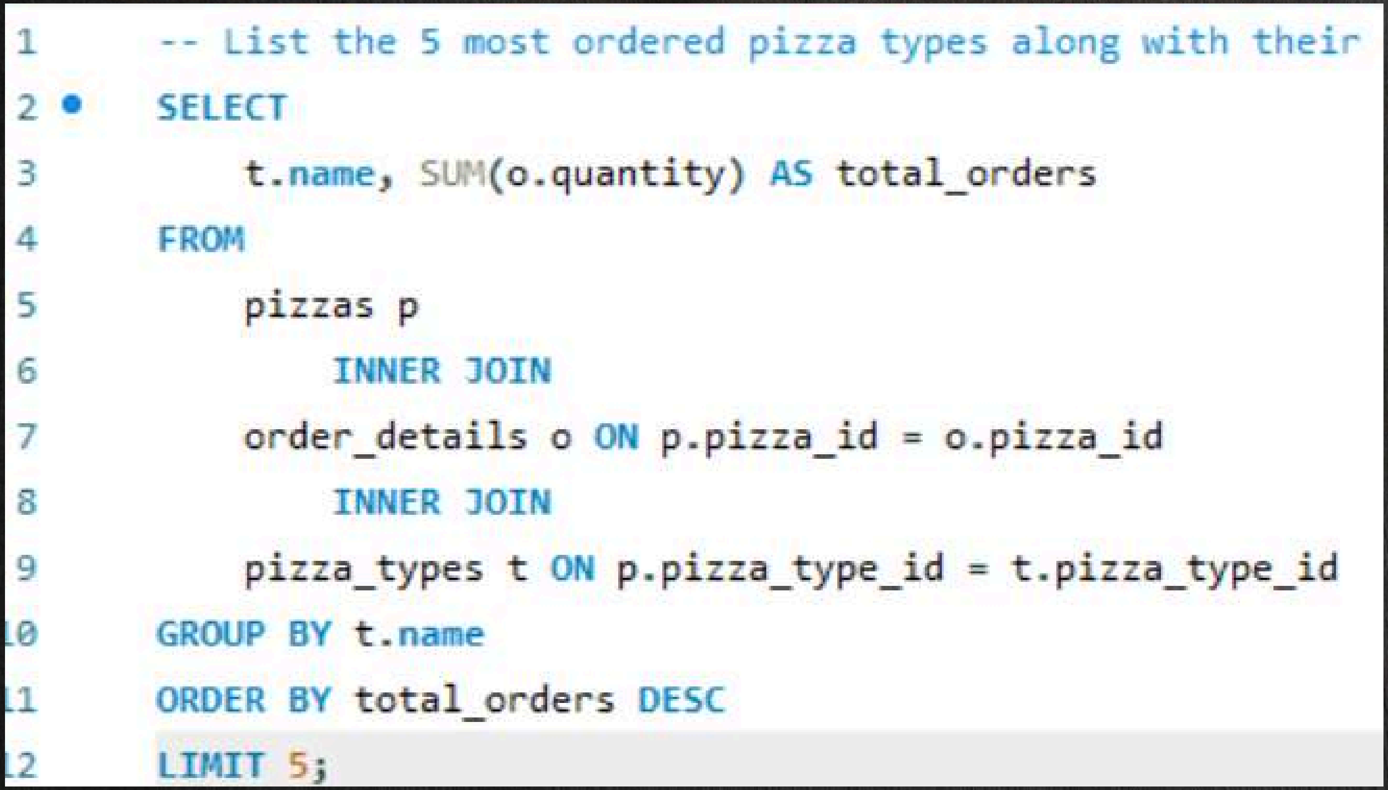


```sql
-- Identify the most common pizza size ordered.
 SELECT
    p.size, COUNT(o.order_details_id)
FROM
    pizzas p
        INNER JOIN
    order_details o ON p.pizza_id = o.pizza_id
GROUP BY p.size
ORDER BY COUNT(o.order_details_id) DESC
LIMIT 1;
```

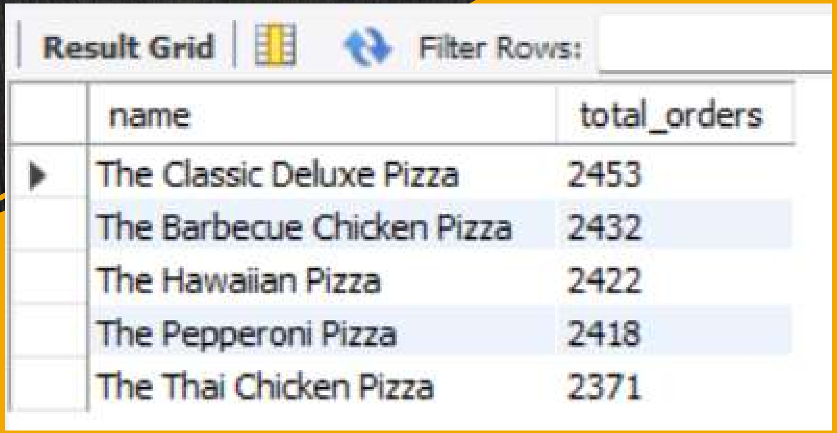


```
size   COUNT(o.order_details_id)
L      18526
```

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



```
-- List the 5 most ordered pizza types along with their
SELECT
    t.name, SUM(o.quantity) AS total_orders
FROM
    pizzas p
        INNER JOIN
    order_details o ON p.pizza_id = o.pizza_id
        INNER JOIN
    pizza_types t ON p.pizza_type_id = t.pizza_type_id
GROUP BY t.name
ORDER BY total_orders DESC
LIMIT 5;
```

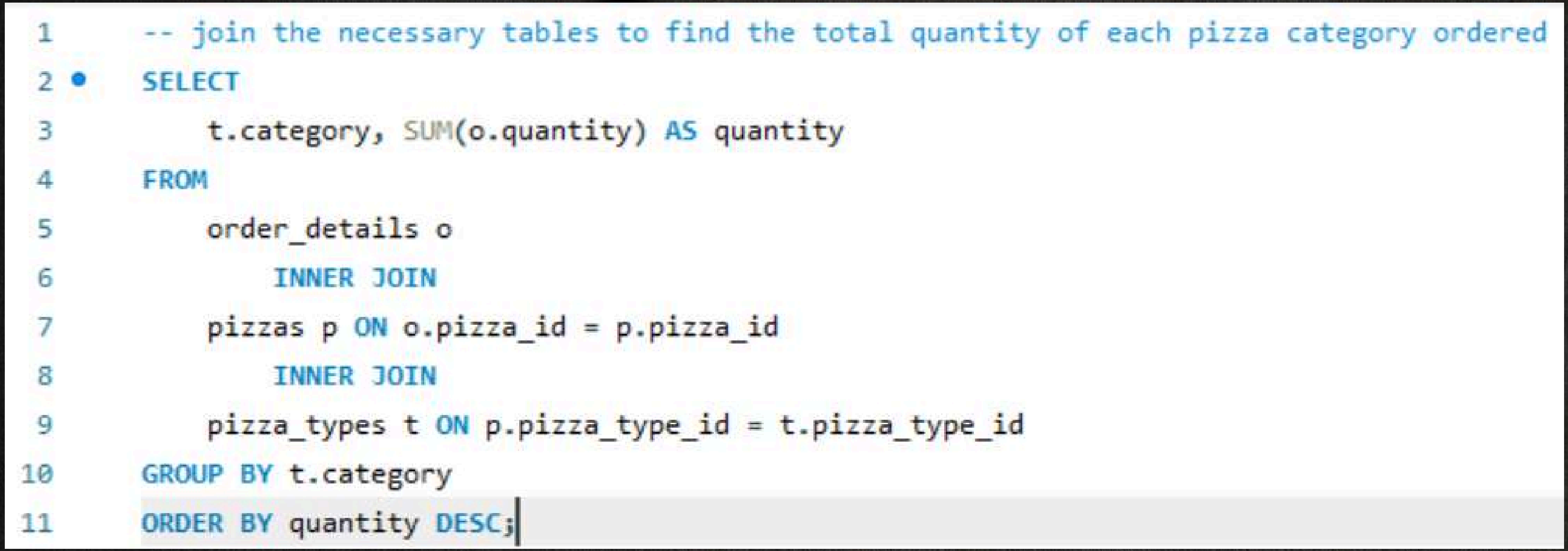


name	total_orders
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

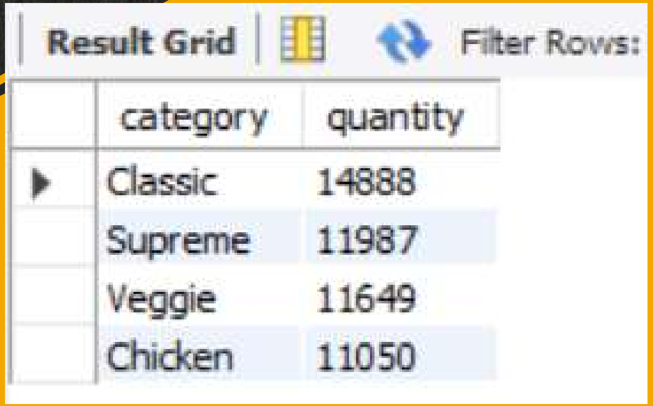
Intermediate Queries:

- 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.

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

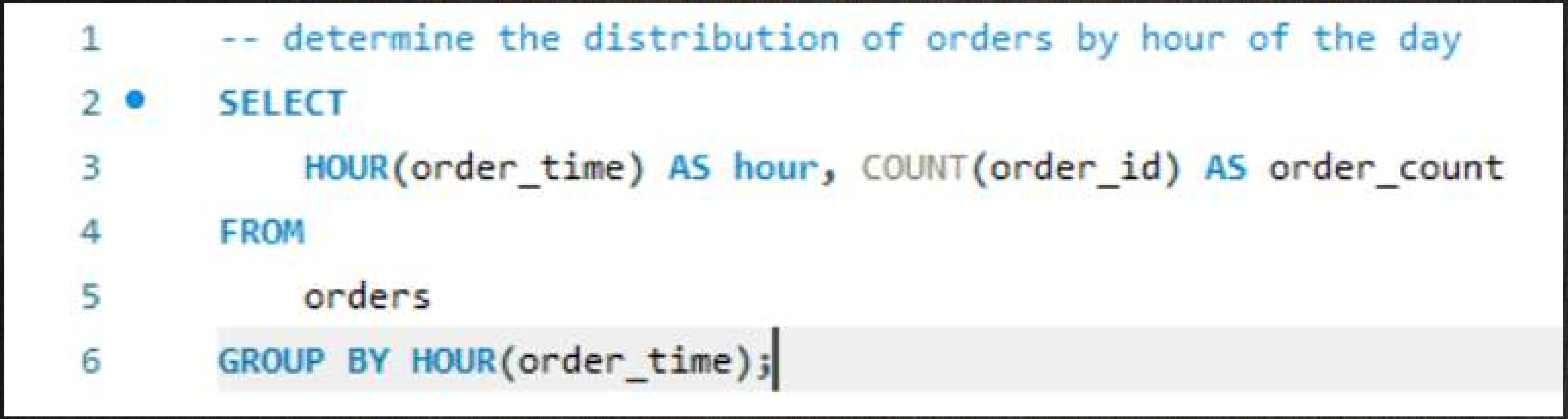


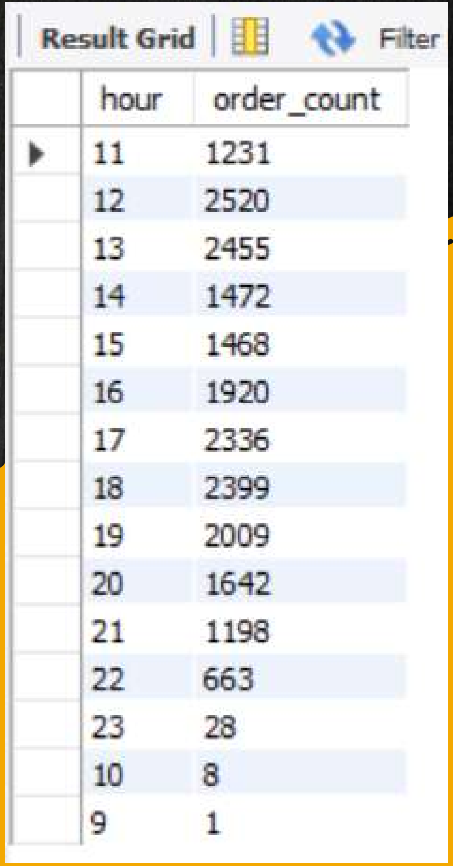
```
-- join the necessary tables to find the total quantity of each pizza category ordered
SELECT
    t.category, SUM(o.quantity) AS quantity
FROM
    order_details o
        INNER JOIN
    pizzas p ON o.pizza_id = p.pizza_id
        INNER JOIN
    pizza_types t ON p.pizza_type_id = t.pizza_type_id
GROUP BY t.category
ORDER BY quantity DESC;
```



category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

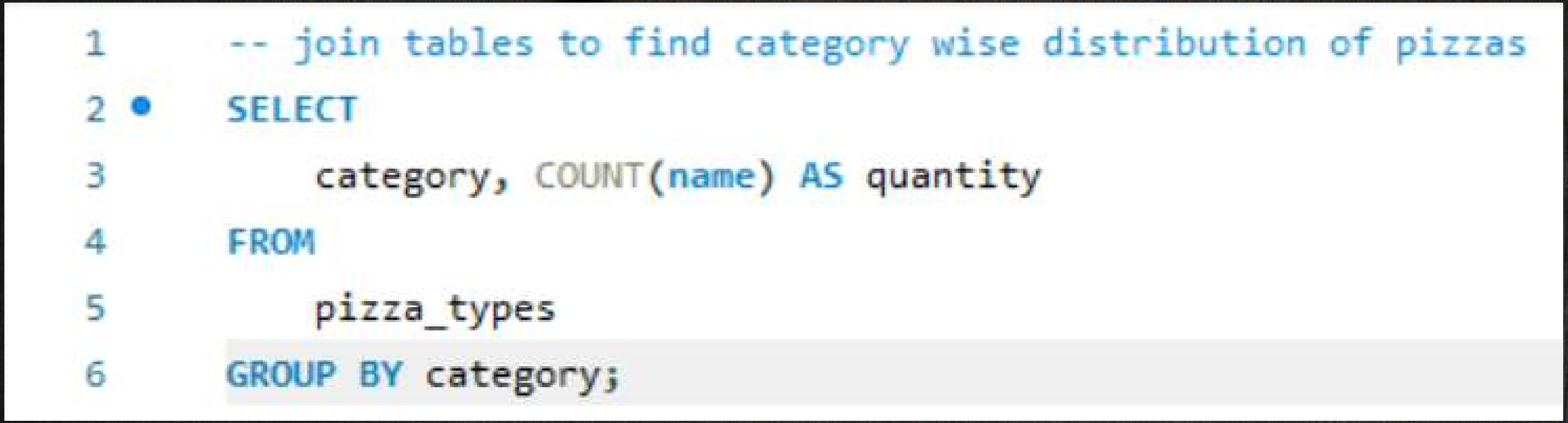
2. Determine the distribution of orders by hour of the day.


```
-- 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(order_time);
```

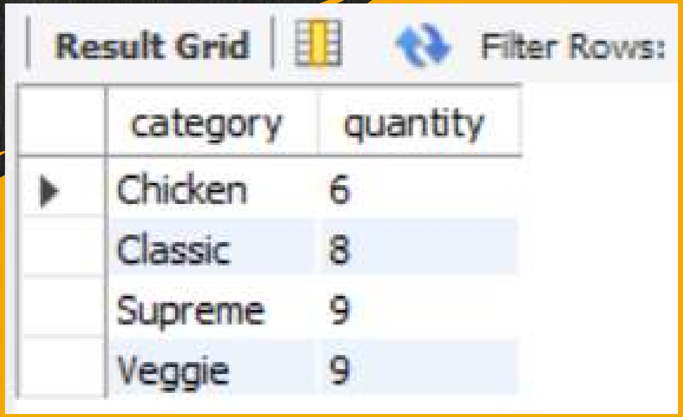


hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

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

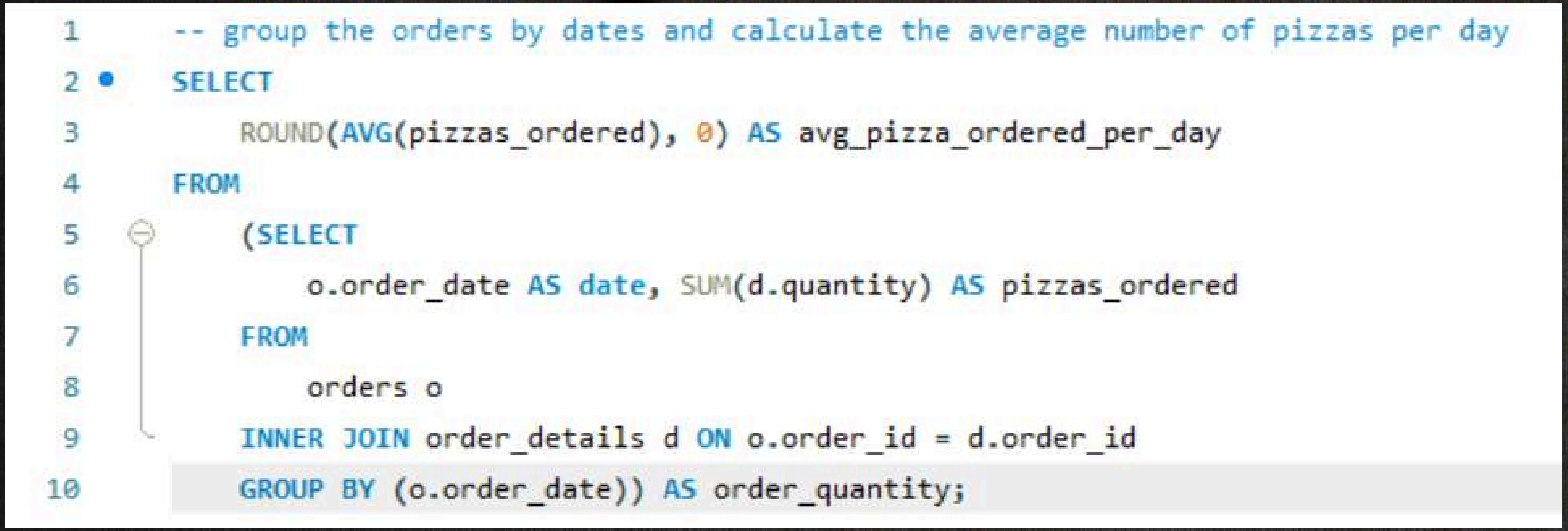


```sql
-- join tables to find category wise distribution of pizzas
SELECT
    category, COUNT(name) AS quantity
FROM
    pizza_types
GROUP BY category;
```

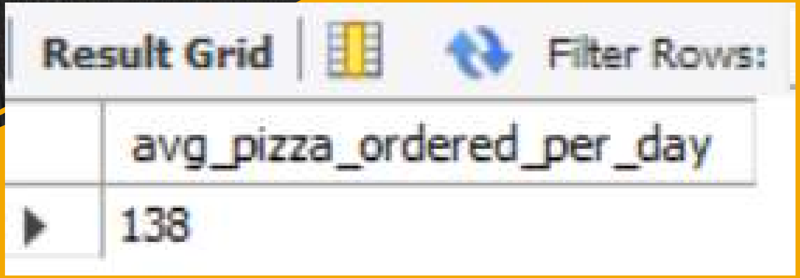


category	quantity
Chicken	6
Classic	8
Supreme	9
Veggie	9

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

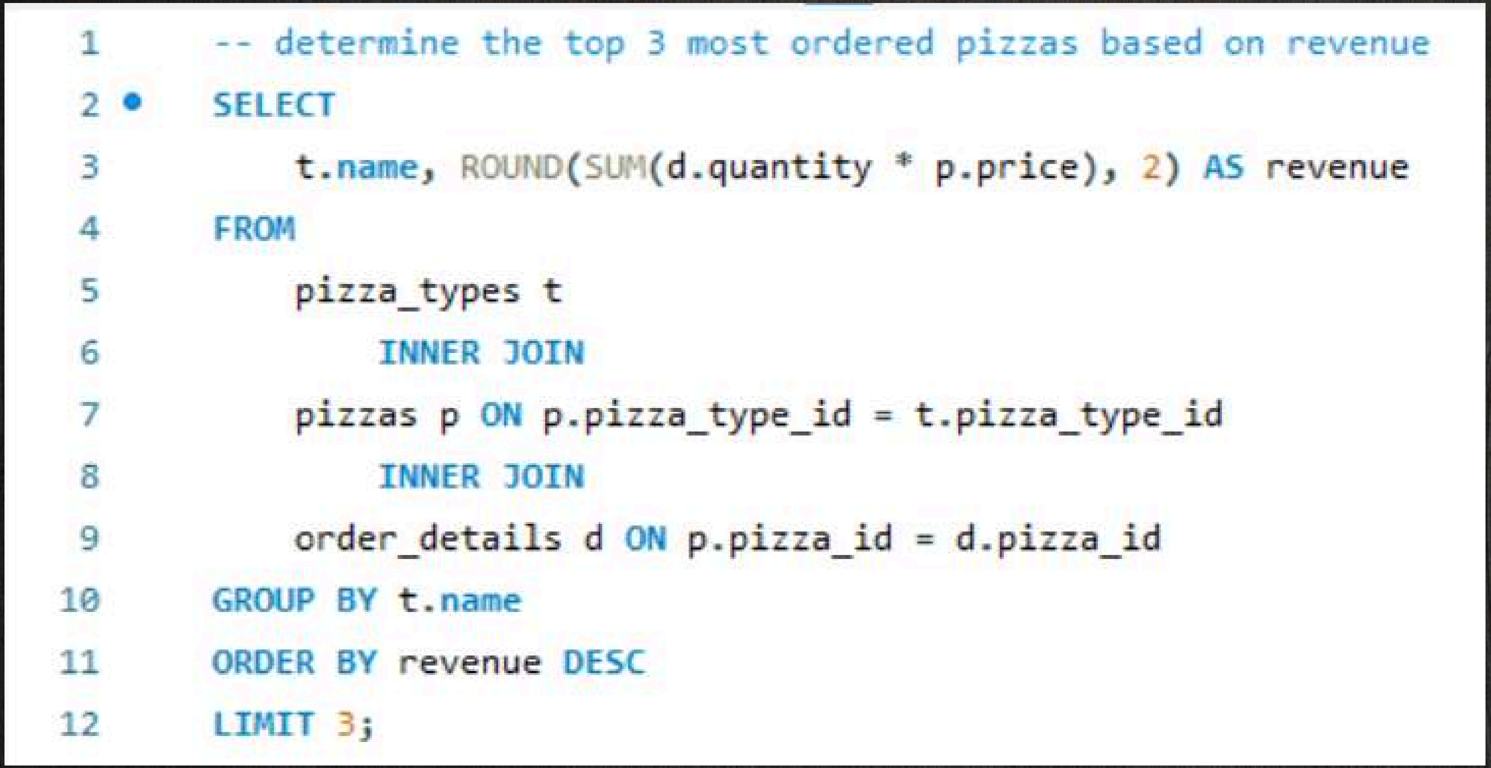


```sql
-- group the orders by dates and calculate the average number of pizzas per day
SELECT
    ROUND(AVG(pizzas_ordered), 0) AS avg_pizza_ordered_per_day
FROM
    (SELECT
        o.order_date AS date, SUM(d.quantity) AS pizzas_ordered
    FROM
        orders o
    INNER JOIN order_details d ON o.order_id = d.order_id
    GROUP BY (o.order_date)) AS order_quantity;
```

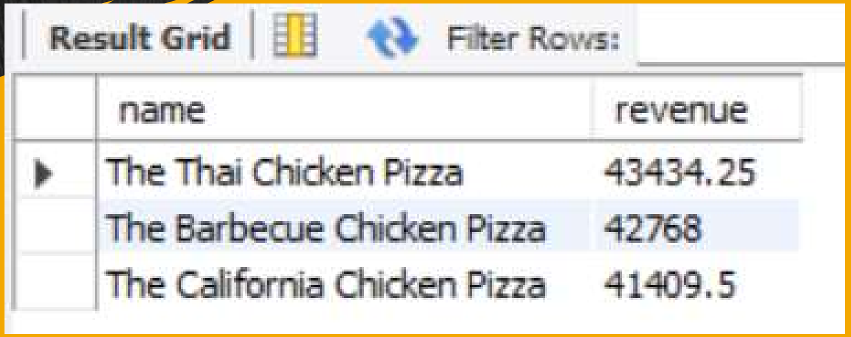


```
avg_pizza_ordered_per_day
138
```

5. Determine the top 3 most ordered pizza types based on revenue.



```sql
-- determine the top 3 most ordered pizzas based on revenue
SELECT
    t.name, ROUND(SUM(d.quantity * p.price), 2) AS revenue
FROM
    pizza_types t
        INNER JOIN
    pizzas p ON p.pizza_type_id = t.pizza_type_id
        INNER JOIN
    order_details d ON p.pizza_id = d.pizza_id
GROUP BY t.name
ORDER BY revenue DESC
LIMIT 3;
```

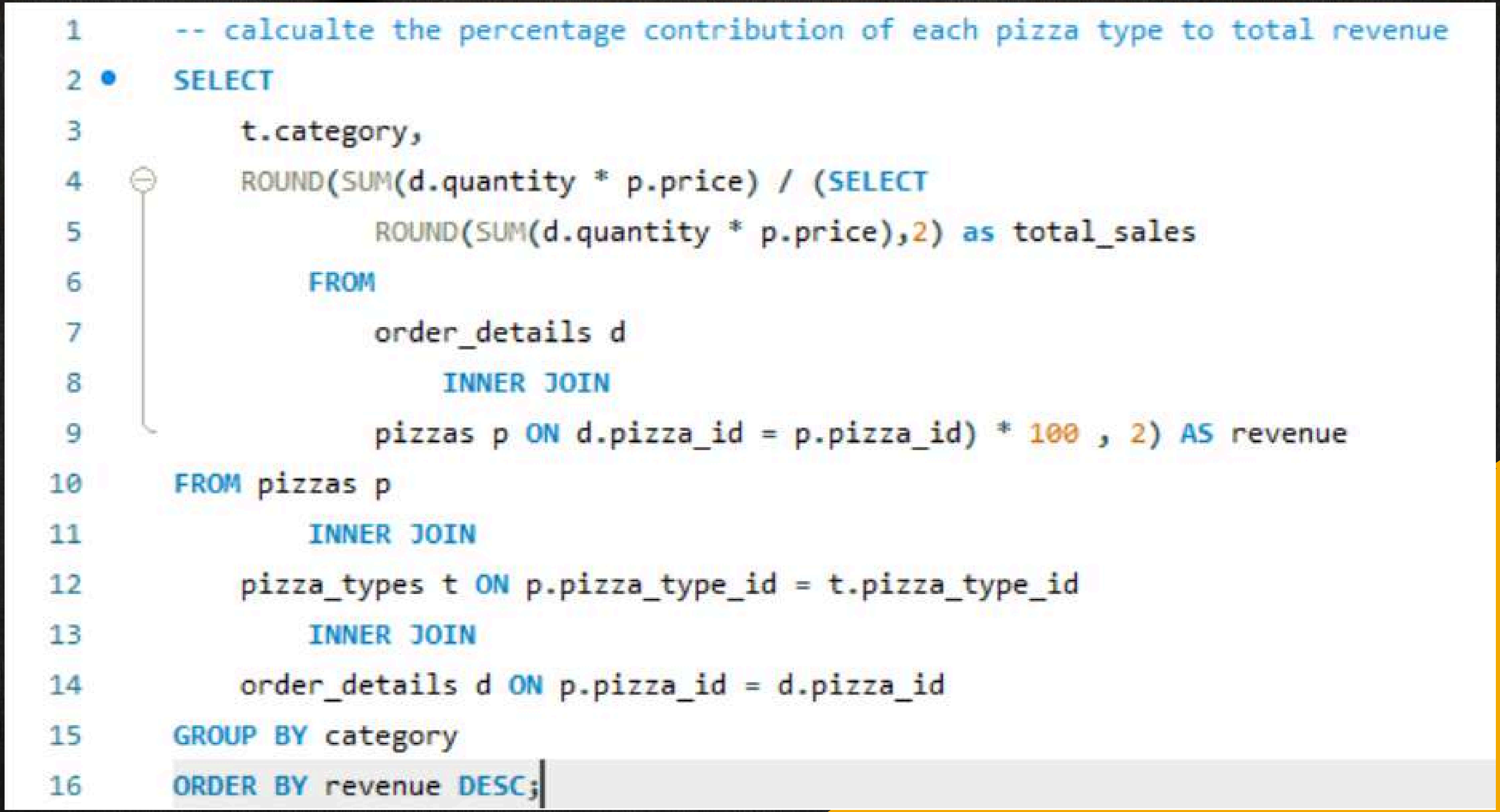


```
name                           revenue
The Thai Chicken Pizza         43434.25
The Barbecue Chicken Pizza     42768
The California Chicken Pizza   41409.5
```

Advanced Queries:

- 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.

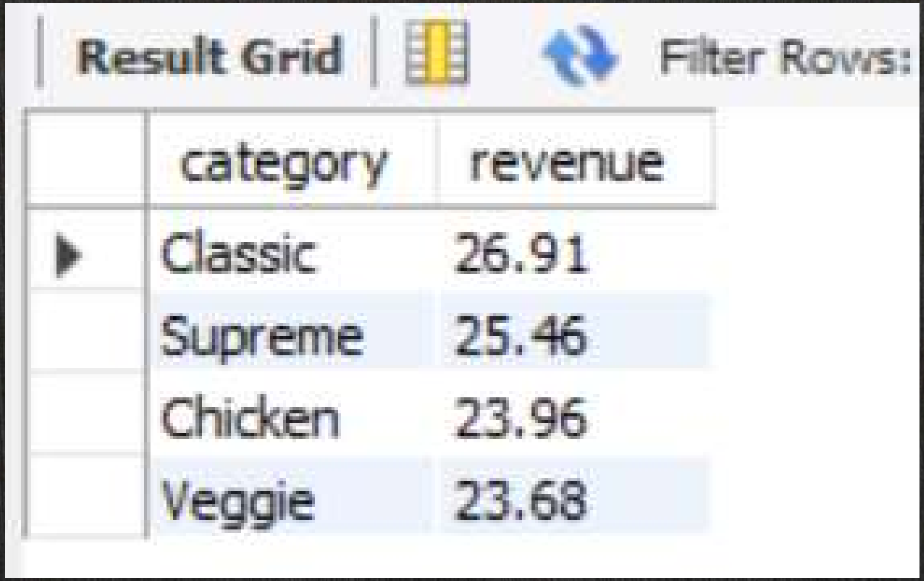
1.Calculate the percentage contribution of each pizza type to total revenue



```
-- calcualte the percentage contribution of each pizza type to total revenue
SELECT
    t.category,
    ROUND(SUM(d.quantity * p.price) / (SELECT
            ROUND(SUM(d.quantity * p.price),2) as total_sales
        FROM
            order_details d
                INNER JOIN
            pizzas p ON d.pizza_id = p.pizza_id) * 100 , 2) AS revenue
FROM pizzas p
        INNER JOIN
    pizza_types t ON p.pizza_type_id = t.pizza_type_id
        INNER JOIN
    order_details d ON p.pizza_id = d.pizza_id
GROUP BY category
ORDER BY revenue DESC;
```

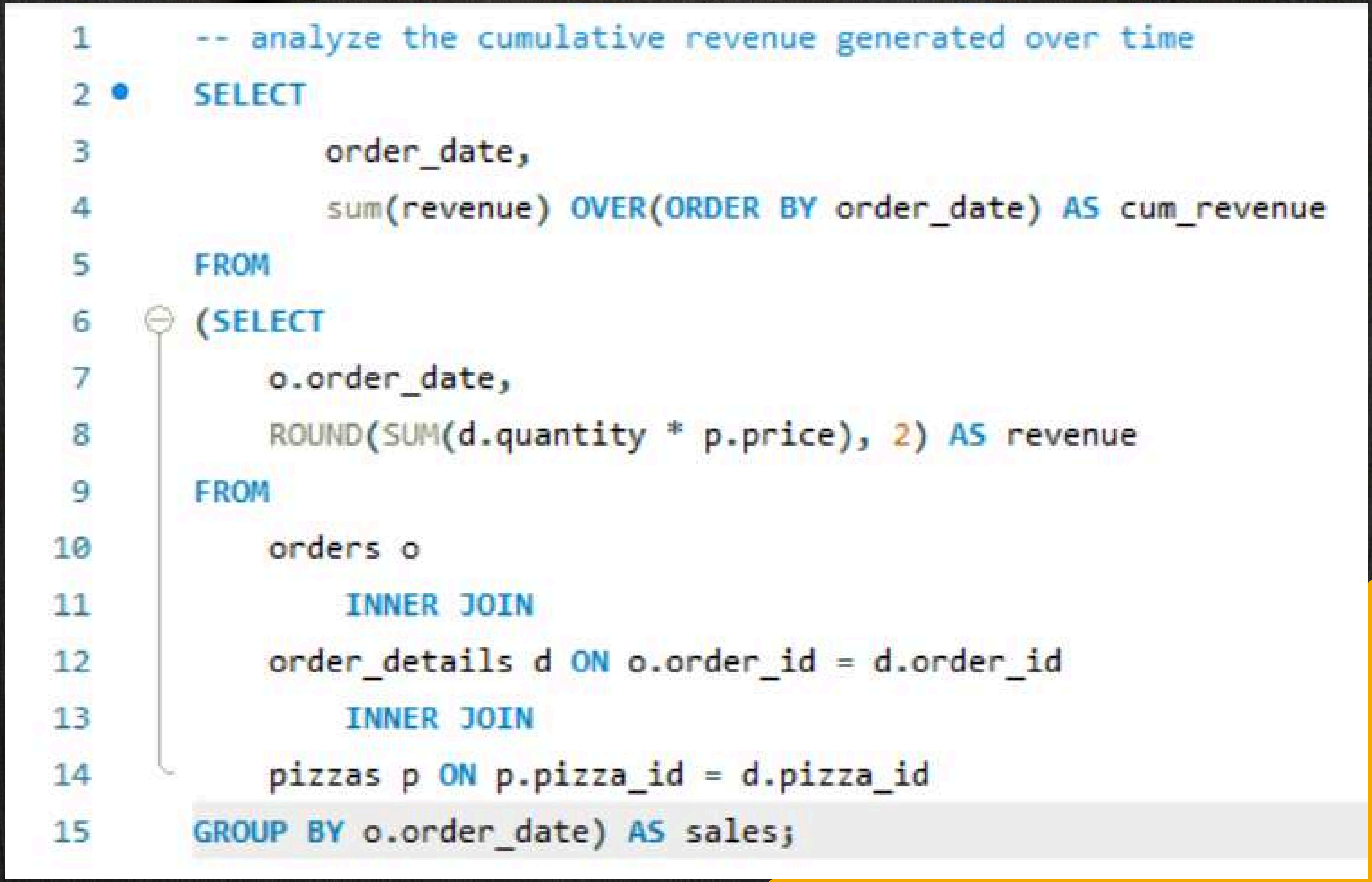
1.Calculate the percentage contribution of each pizza type to total revenue

Output:-



category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

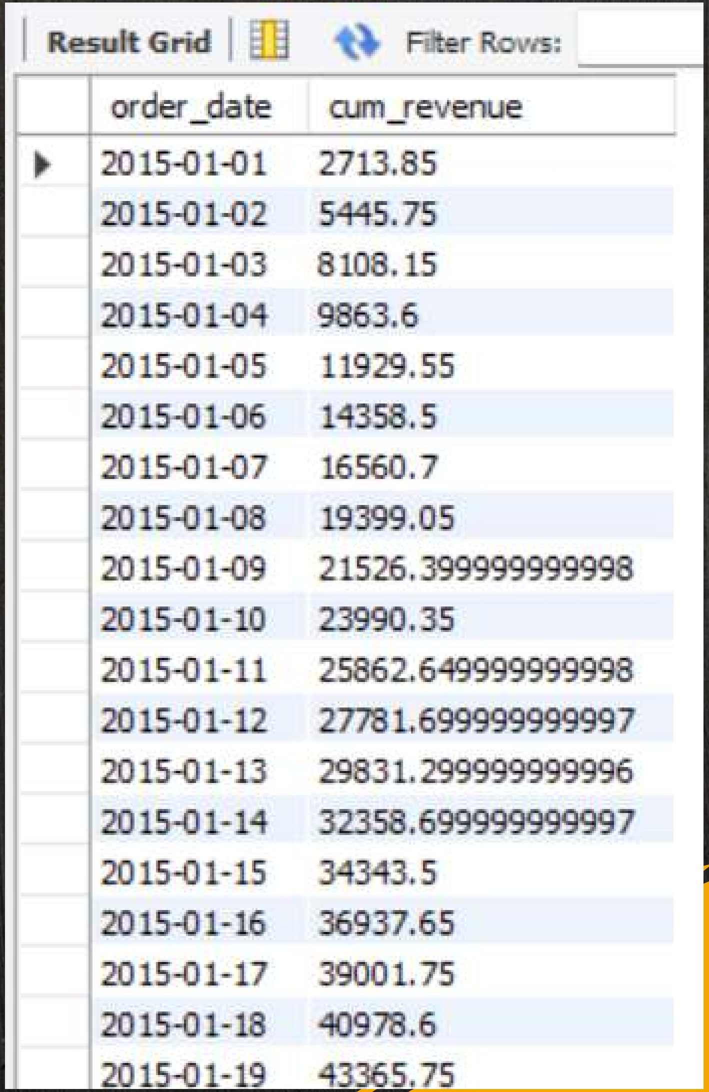
2.Analyze the cumulative revenue generated over time.



```
-- analyze the cumulative revenue generated over time
SELECT
        order_date,
        sum(revenue) OVER(ORDER BY order_date) AS cum_revenue
FROM
(SELECT
    o.order_date,
    ROUND(SUM(d.quantity * p.price), 2) AS revenue
FROM
    orders o
        INNER JOIN
    order_details d ON o.order_id = d.order_id
        INNER JOIN
    pizzas p ON p.pizza_id = d.pizza_id
GROUP BY o.order_date) AS sales;
```

2.Analyze the cumulative revenue generated over time.

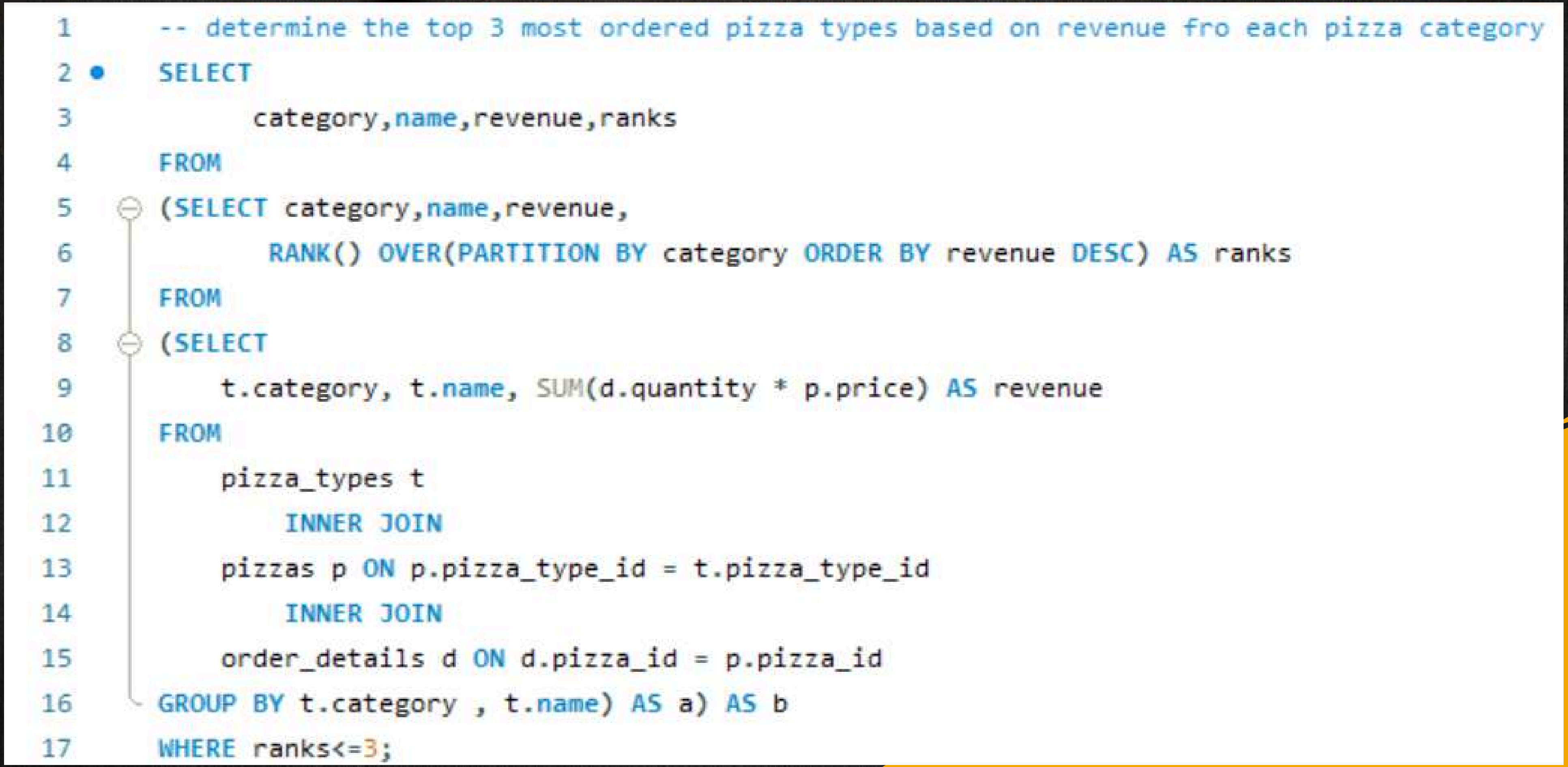
Output:-



Result Grid | Filter Rows:

order_date	cum_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	14358.5
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526.399999999998
2015-01-10	23990.35
2015-01-11	25862.649999999998
2015-01-12	27781.699999999997
2015-01-13	29831.299999999996
2015-01-14	32358.699999999997
2015-01-15	34343.5
2015-01-16	36937.65
2015-01-17	39001.75
2015-01-18	40978.6
2015-01-19	43365.75

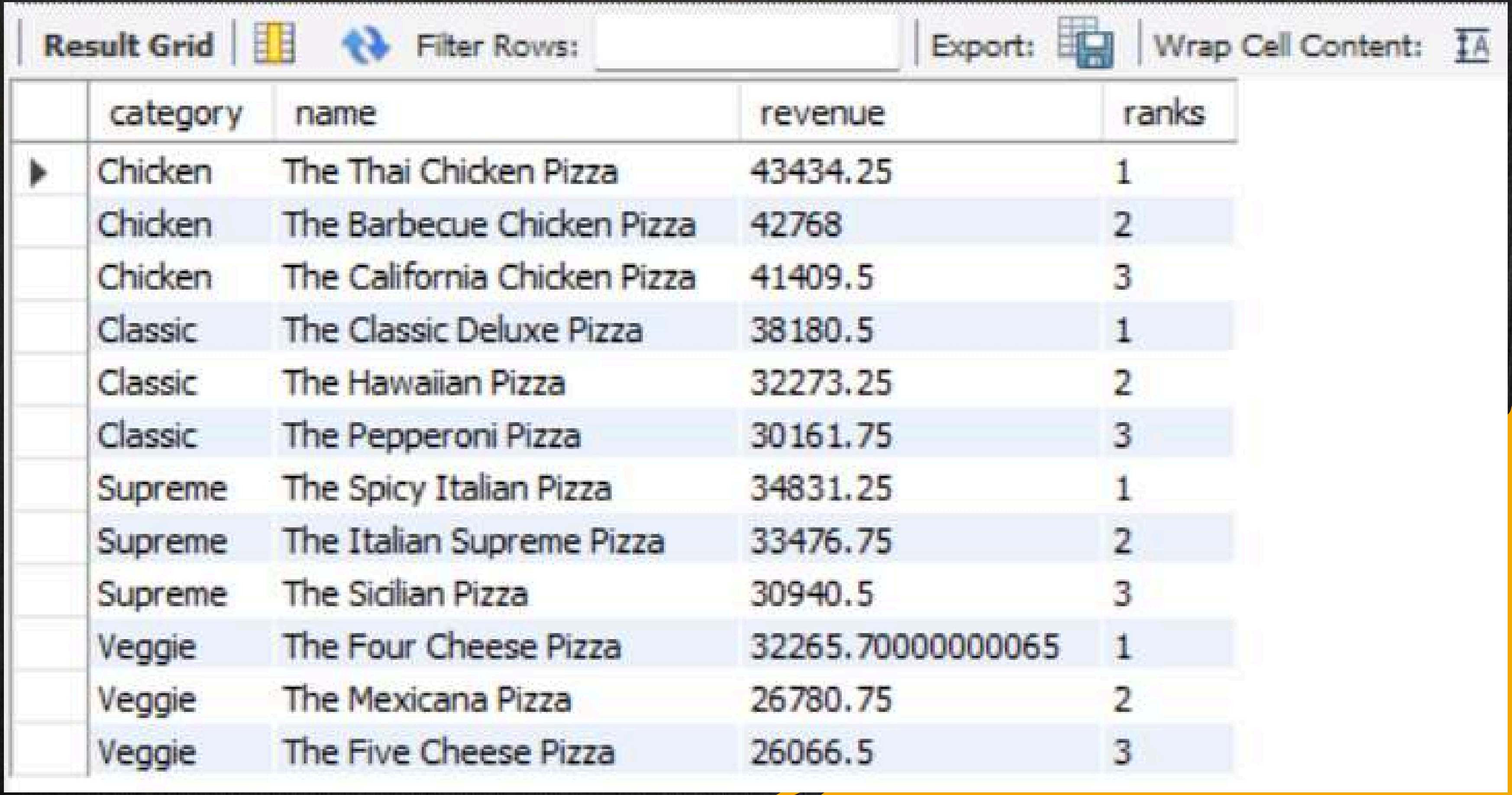
3.Determine the top 3 most ordered pizza types based on revenue for each pizza category.



```sql
-- determine the top 3 most ordered pizza types based on revenue fro each pizza category
SELECT
        category,name,revenue,ranks
FROM
(SELECT category,name,revenue,
         RANK() OVER(PARTITION BY category ORDER BY revenue DESC) AS ranks
FROM
(SELECT
     t.category, t.name, SUM(d.quantity * p.price) AS revenue
FROM
     pizza_types t
         INNER JOIN
     pizzas p ON p.pizza_type_id = t.pizza_type_id
         INNER JOIN
     order_details d ON d.pizza_id = p.pizza_id
GROUP BY t.category , t.name) AS a) AS b
WHERE ranks<=3;
```

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

Output:-



category	name	revenue	ranks
Chicken	The Thai Chicken Pizza	43434.25	1
Chicken	The Barbecue Chicken Pizza	42768	2
Chicken	The California Chicken Pizza	41409.5	3
Classic	The Classic Deluxe Pizza	38180.5	1
Classic	The Hawaiian Pizza	32273.25	2
Classic	The Pepperoni Pizza	30161.75	3
Supreme	The Spicy Italian Pizza	34831.25	1
Supreme	The Italian Supreme Pizza	33476.75	2
Supreme	The Sicilian Pizza	30940.5	3
Veggie	The Four Cheese Pizza	32265.70000000065	1
Veggie	The Mexicana Pizza	26780.75	2
Veggie	The Five Cheese Pizza	26066.5	3

Key Learnings from the Project

1. SQL Proficiency

- Advanced Query Techniques: Gained experience with complex SQL queries, including joins, subqueries, and aggregations.
- Data Manipulation: Enhanced skills in data cleaning and transformation to ensure accuracy and consistency.
- Optimization: Learned how to optimize SQL queries for better performance, especially with large datasets.

Key Learnings from the Project

2. Project Management

- Data Handling: Improved capability in managing and preparing data for analysis, including handling incomplete or inconsistent data.
- Problem-Solving: Enhanced problem-solving skills by addressing challenges encountered during data analysis.
- Documentation: Gained experience in documenting methodologies and findings.