Pizza Sales Data Analysis using MySQL

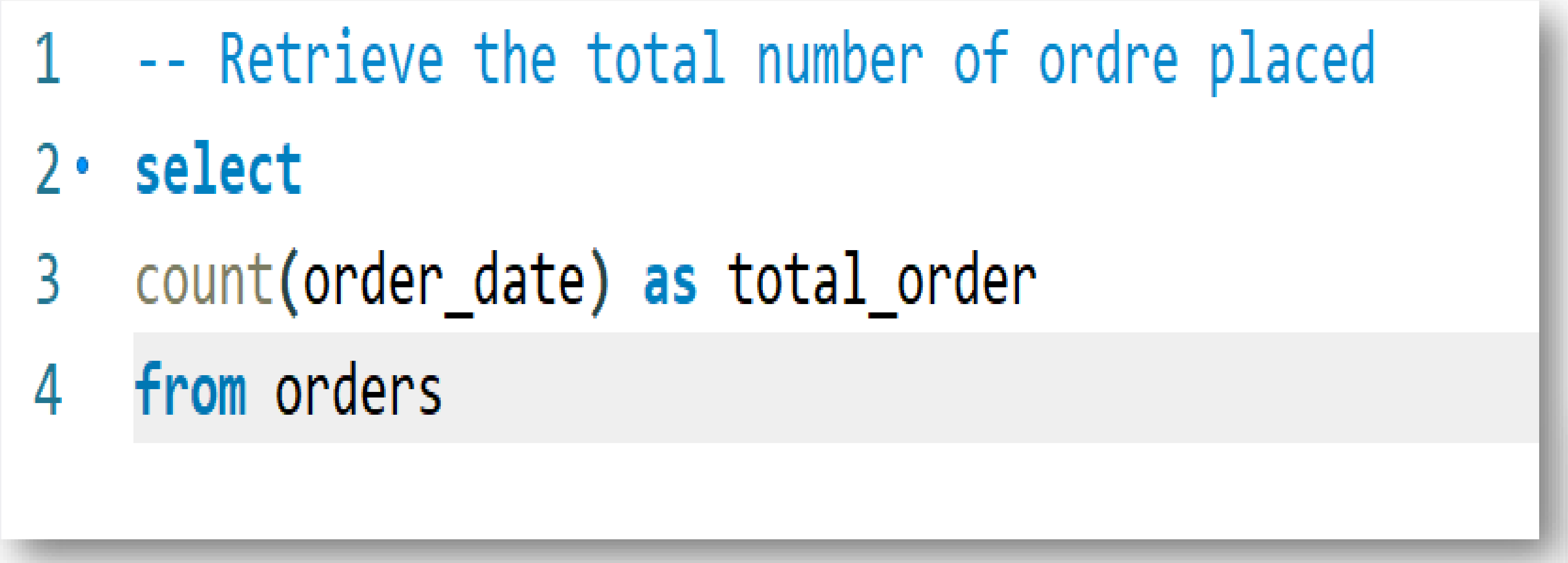
Insights into Orders, Revenue, and Customer Preferences

Presented by: Sameer Haider

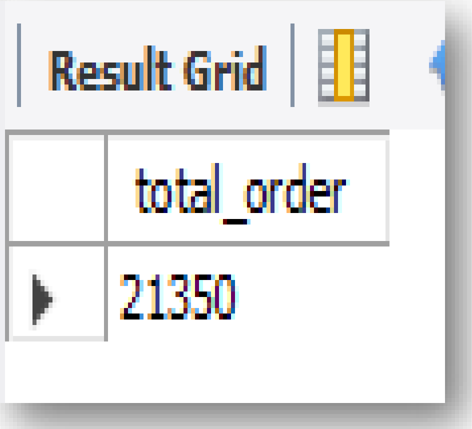
About This Project

This project represents the first comprehensive application of my SQL training. The objective was to move beyond syntax learning and solve actual business problems using data. By analyzing the Pizza Sales dataset, I have applied fundamental to intermediate SQL concepts—including Joins, Aggregates, and Grouping—to derive meaningful insights related to sales trends and inventory management

Retrieve the total number of orders placed

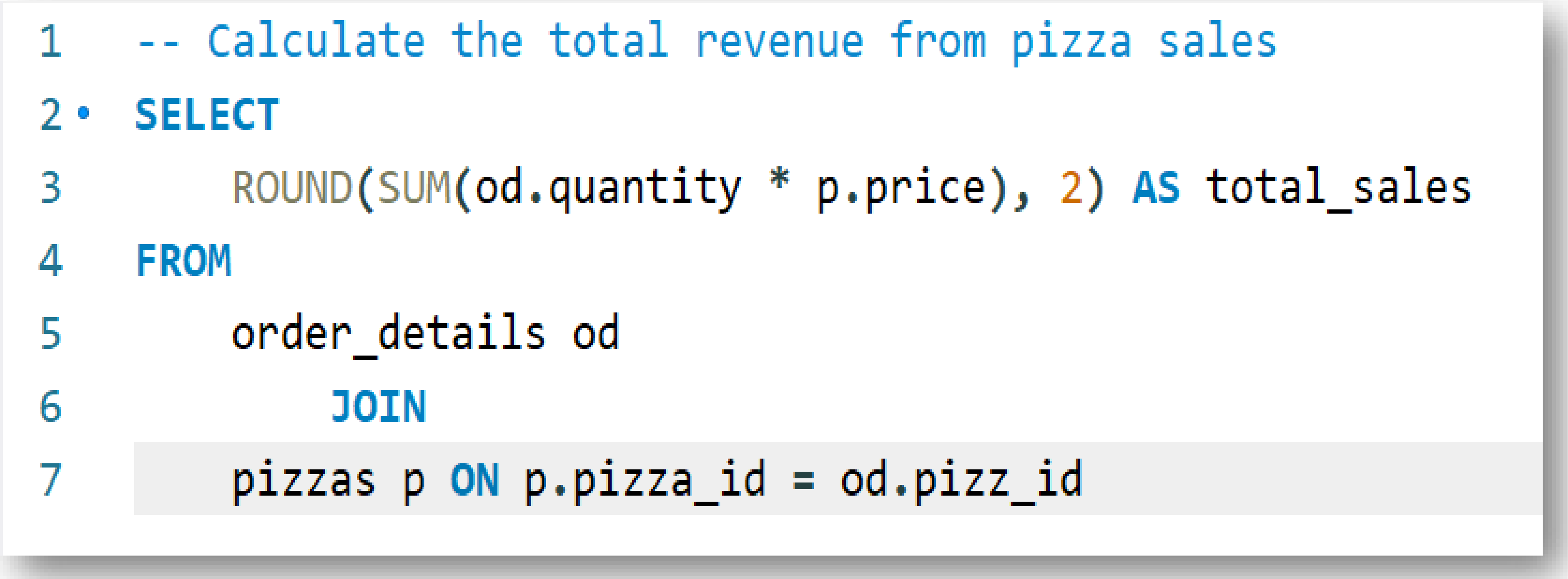


```sql
-- Retrieve the total number of ordre placed
select
count(order_date) as total_order
from orders
```

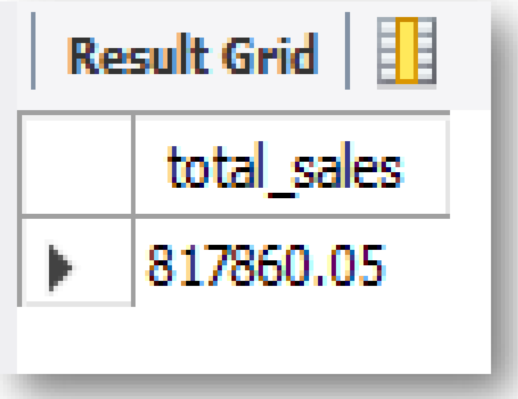


```
total_order
21350
```

Calculate the total revenue from pizza sales.

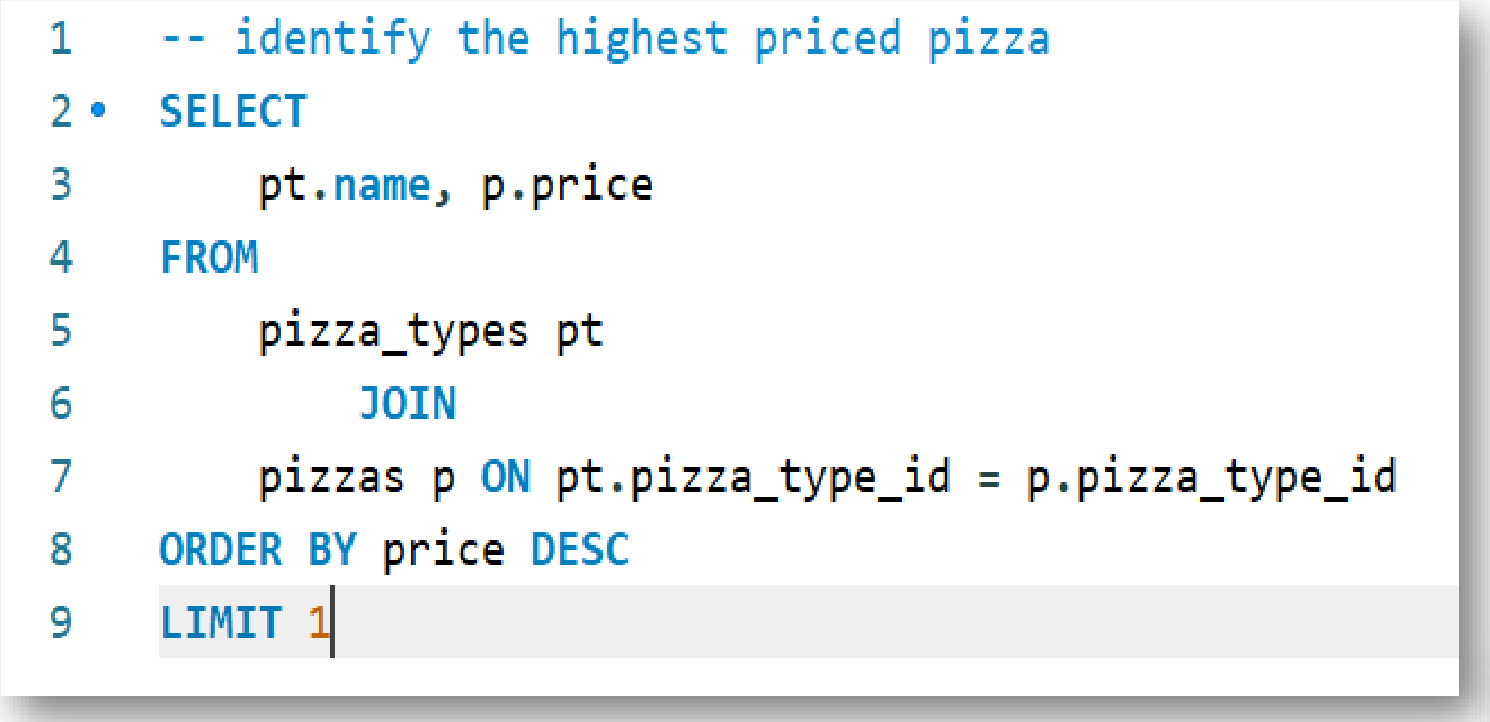


```sql
-- Calculate the total revenue from pizza sales
SELECT
    ROUND(SUM(od.quantity * p.price), 2) AS total_sales
FROM
    order_details od
        JOIN
    pizzas p ON p.pizza_id = od.pizz_id
```

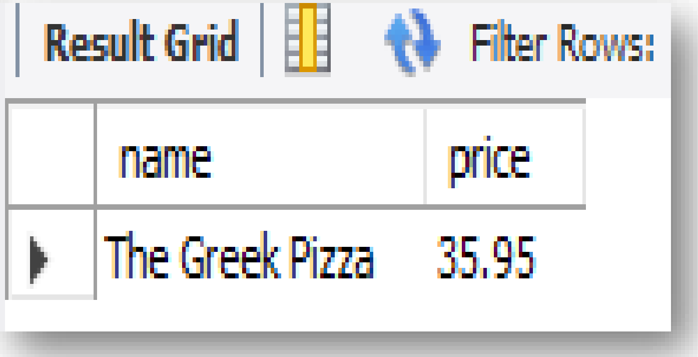


	total_sales
▶	817860.05

Identify the highest-priced pizza

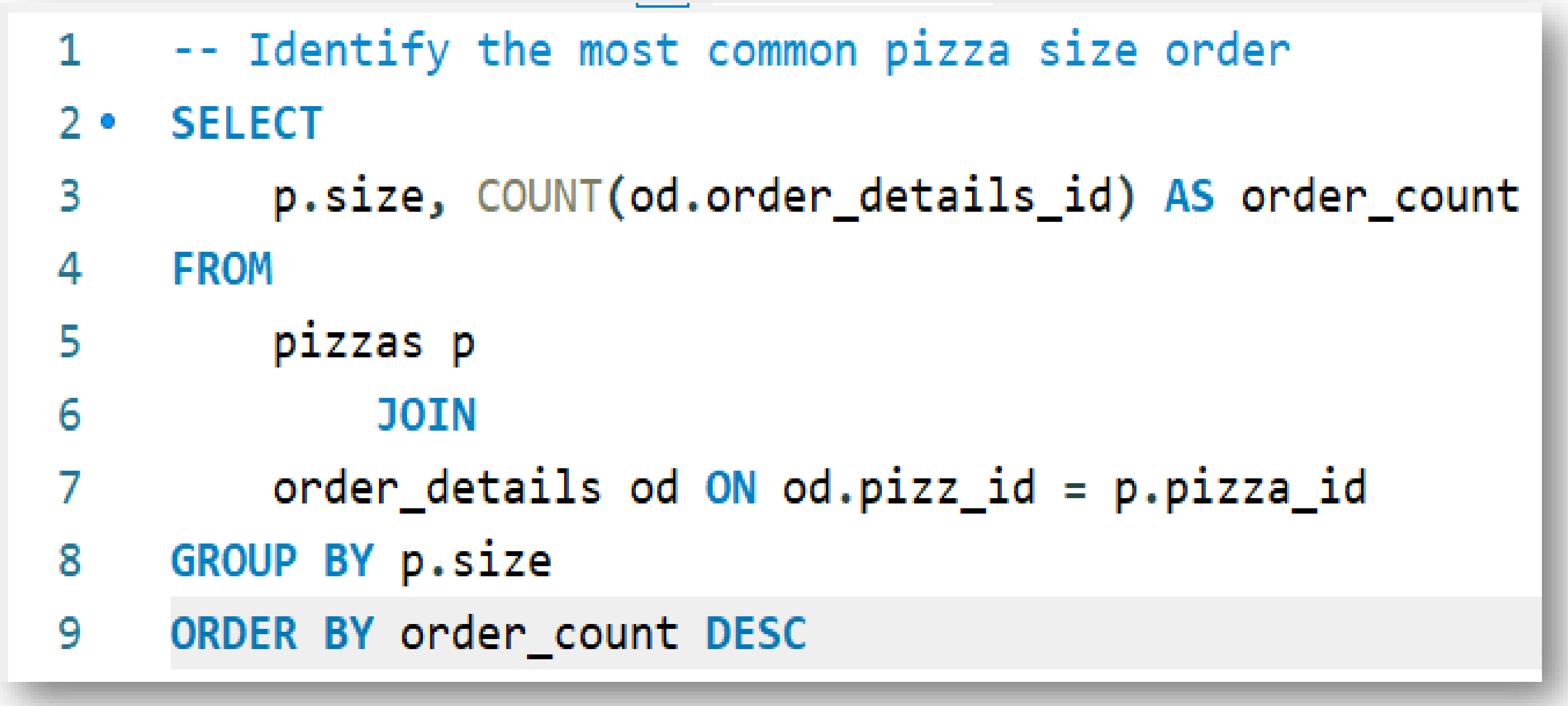


```
-- identify the highest priced pizza
SELECT
    pt.name, p.price
FROM
    pizza_types pt
        JOIN
    pizzas p ON pt.pizza_type_id = p.pizza_type_id
ORDER BY price DESC
LIMIT 1
```

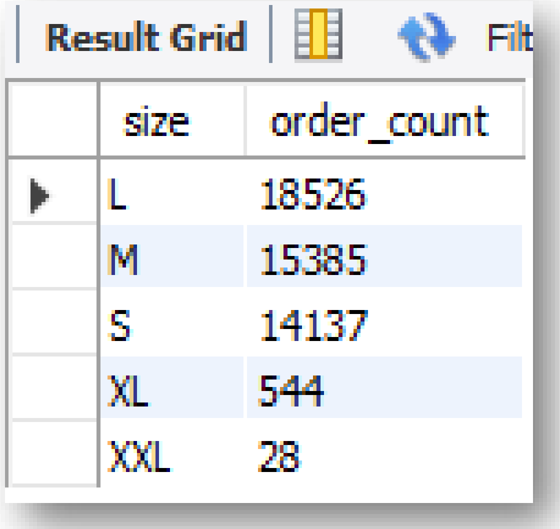


name	price
The Greek Pizza	35.95

Identify the most common pizza size ordered

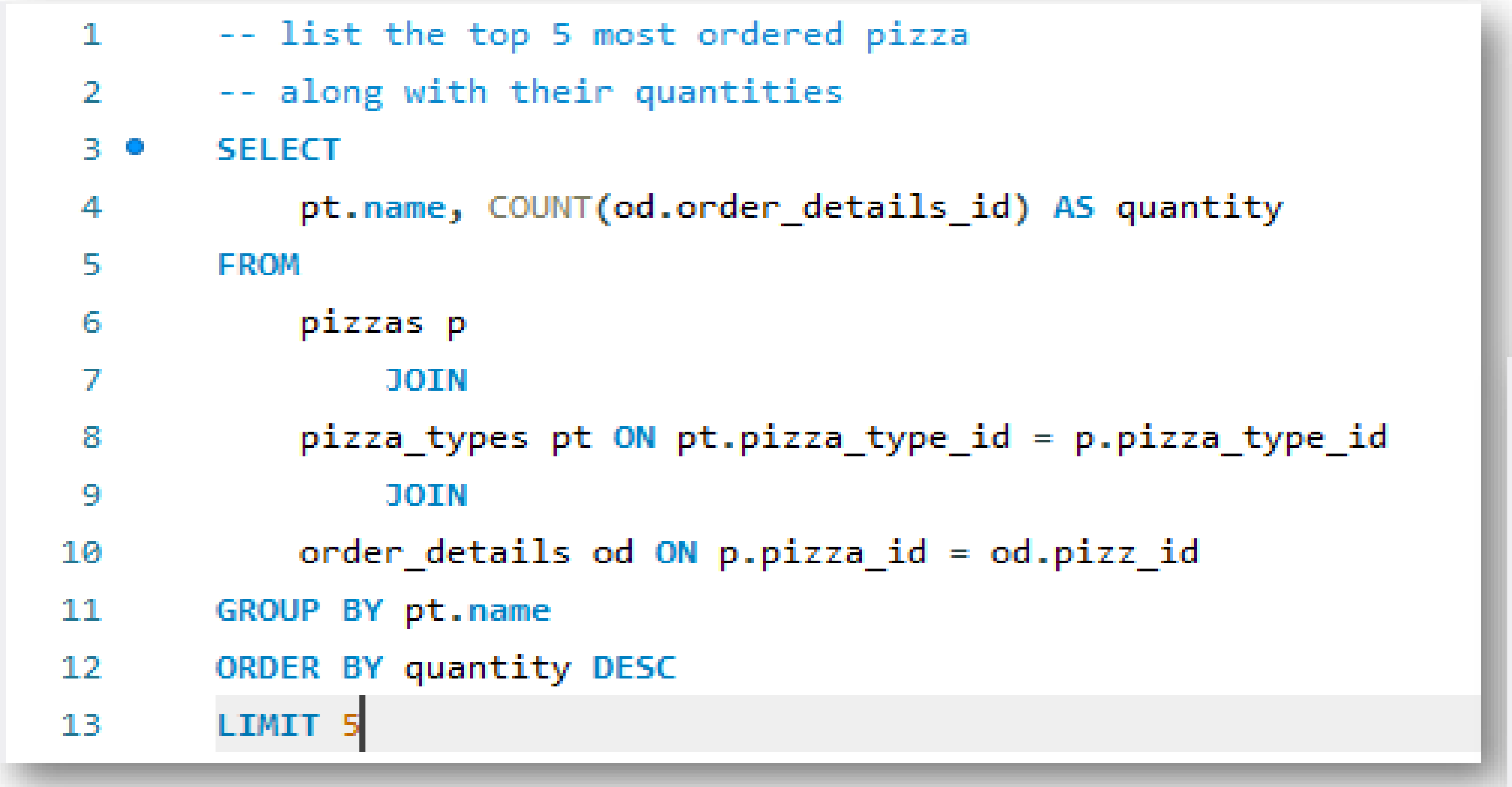


```sql
-- Identify the most common pizza size order
SELECT
    p.size, COUNT(od.order_details_id) AS order_count
FROM
    pizzas p
        JOIN
    order_details od ON od.pizz_id = p.pizza_id
GROUP BY p.size
ORDER BY order_count DESC
```

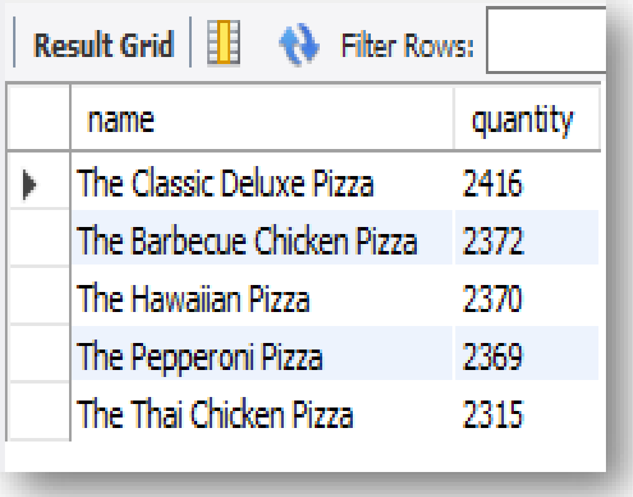


size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

List the Top 5 Most Ordered Pizzas Along with Their Quantities

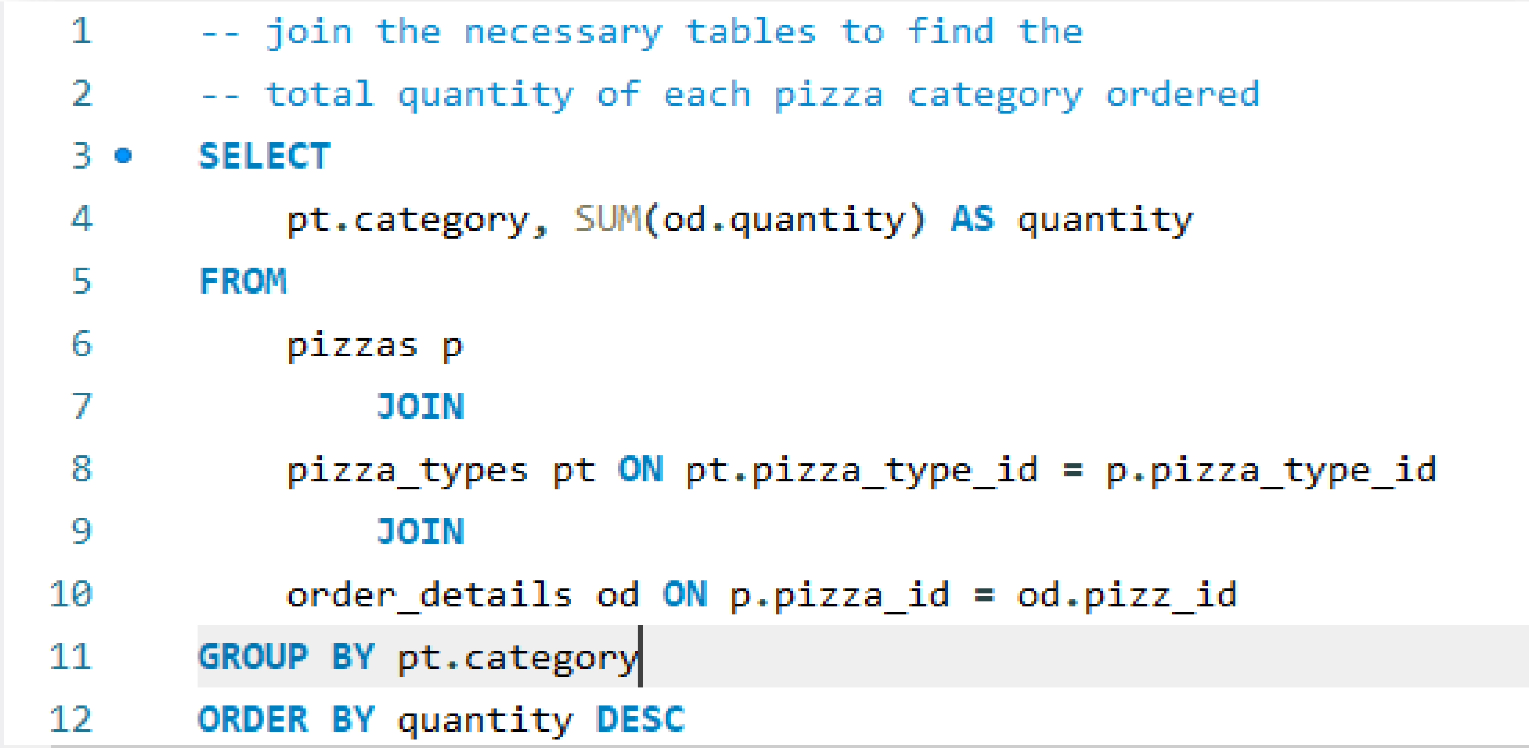


```sql
-- list the top 5 most ordered pizza
-- along with their quantities
SELECT
    pt.name, COUNT(od.order_details_id) AS quantity
FROM
    pizzas p
        JOIN
    pizza_types pt ON pt.pizza_type_id = p.pizza_type_id
        JOIN
    order_details od ON p.pizza_id = od.pizz_id
GROUP BY pt.name
ORDER BY quantity DESC
LIMIT 5
```

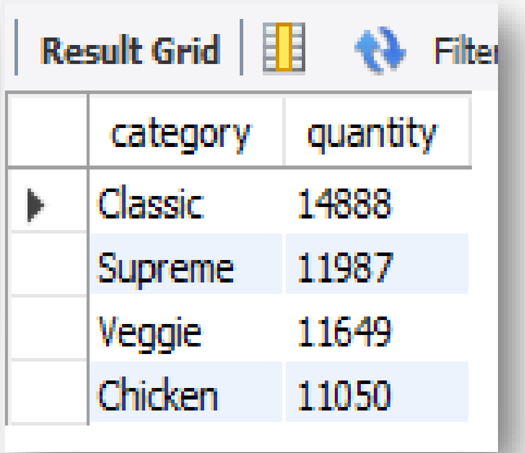


name	quantity
The Classic Deluxe Pizza	2416
The Barbecue Chicken Pizza	2372
The Hawaiian Pizza	2370
The Pepperoni Pizza	2369
The Thai Chicken Pizza	2315

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

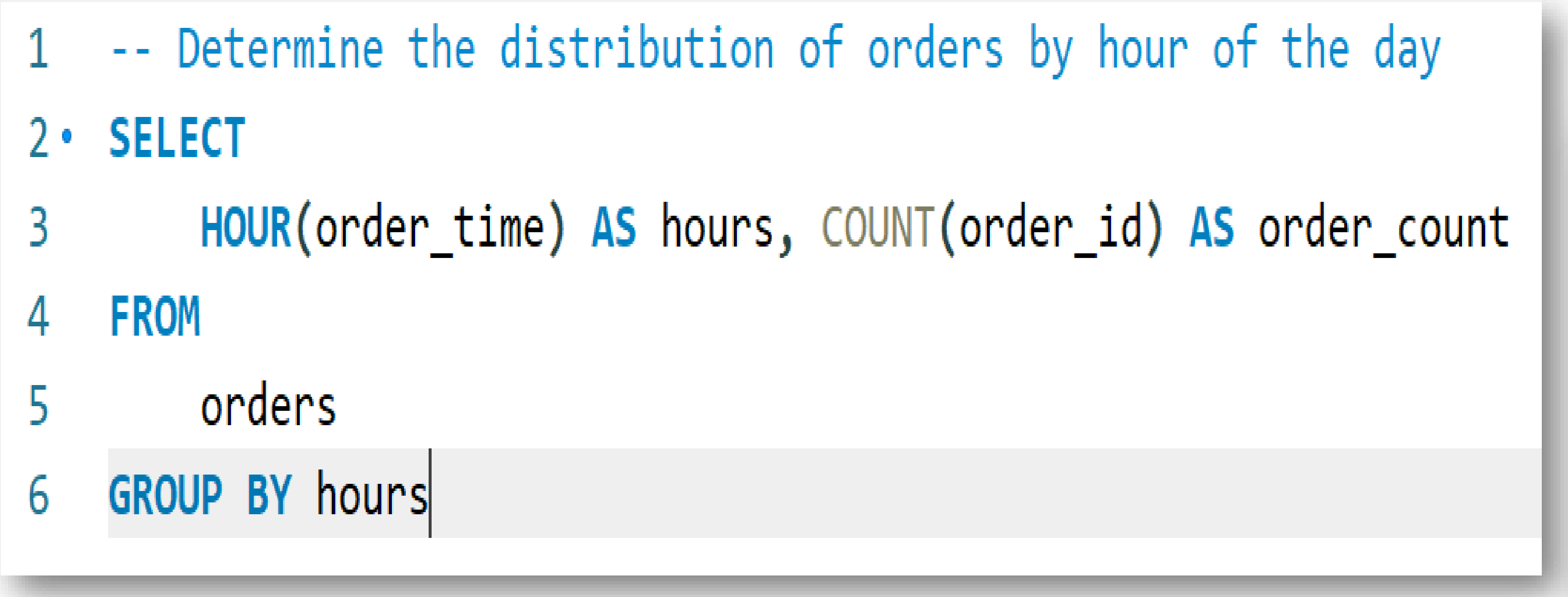


```
-- join the necessary tables to find the
-- total quantity of each pizza category ordered
SELECT
    pt.category, SUM(od.quantity) AS quantity
FROM
    pizzas p
        JOIN
    pizza_types pt ON pt.pizza_type_id = p.pizza_type_id
        JOIN
    order_details od ON p.pizza_id = od.pizz_id
GROUP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

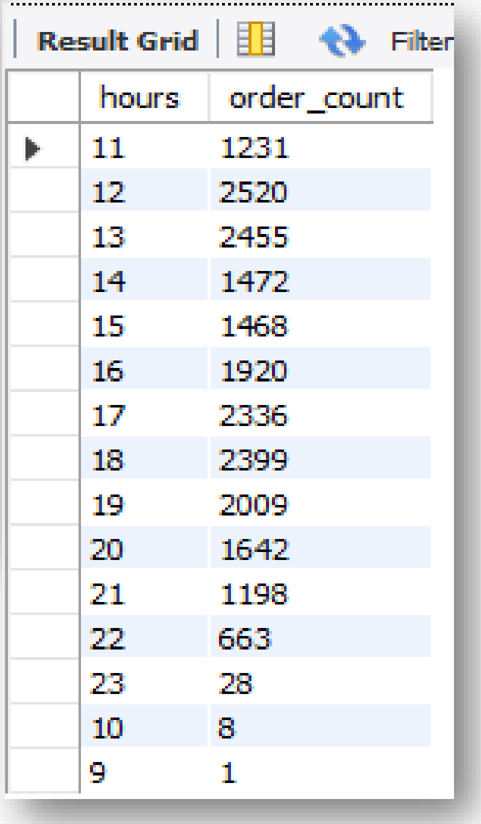


category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Determine the distribution of orders by hour of the day

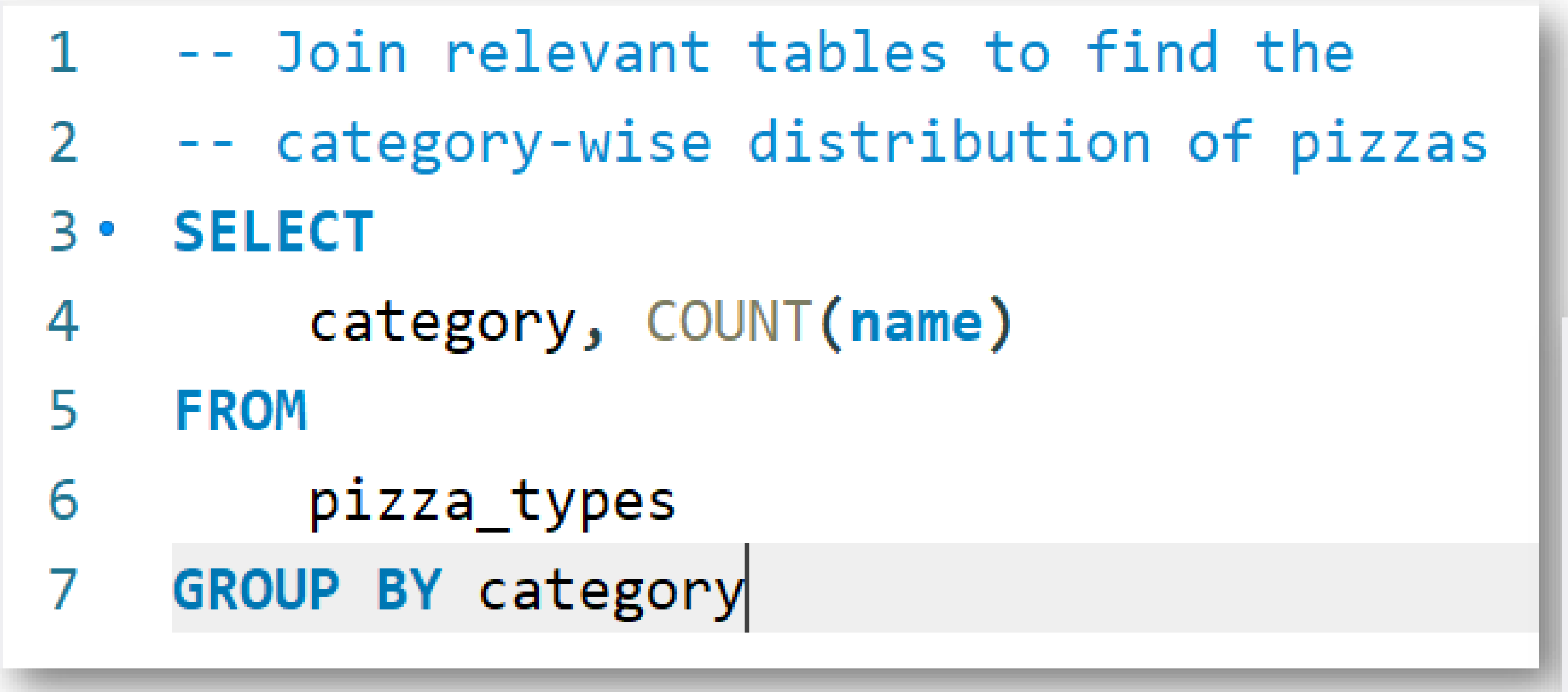


```sql
-- Determine the distribution of orders by hour of the day
SELECT
    HOUR(order_time) AS hours, COUNT(order_id) AS order_count
FROM
    orders
GROUP BY hours
```

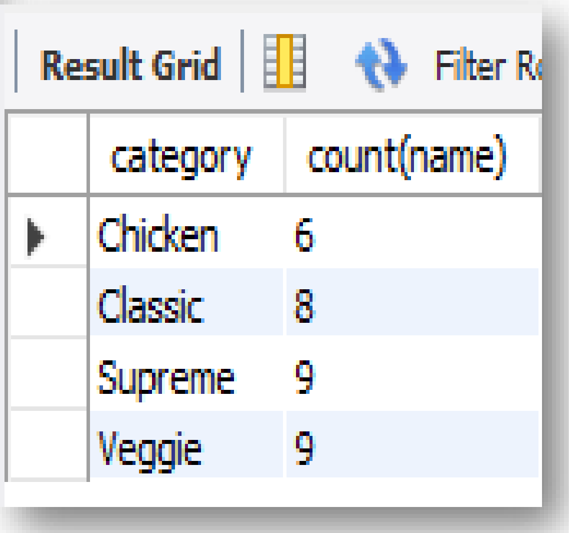


hours	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

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

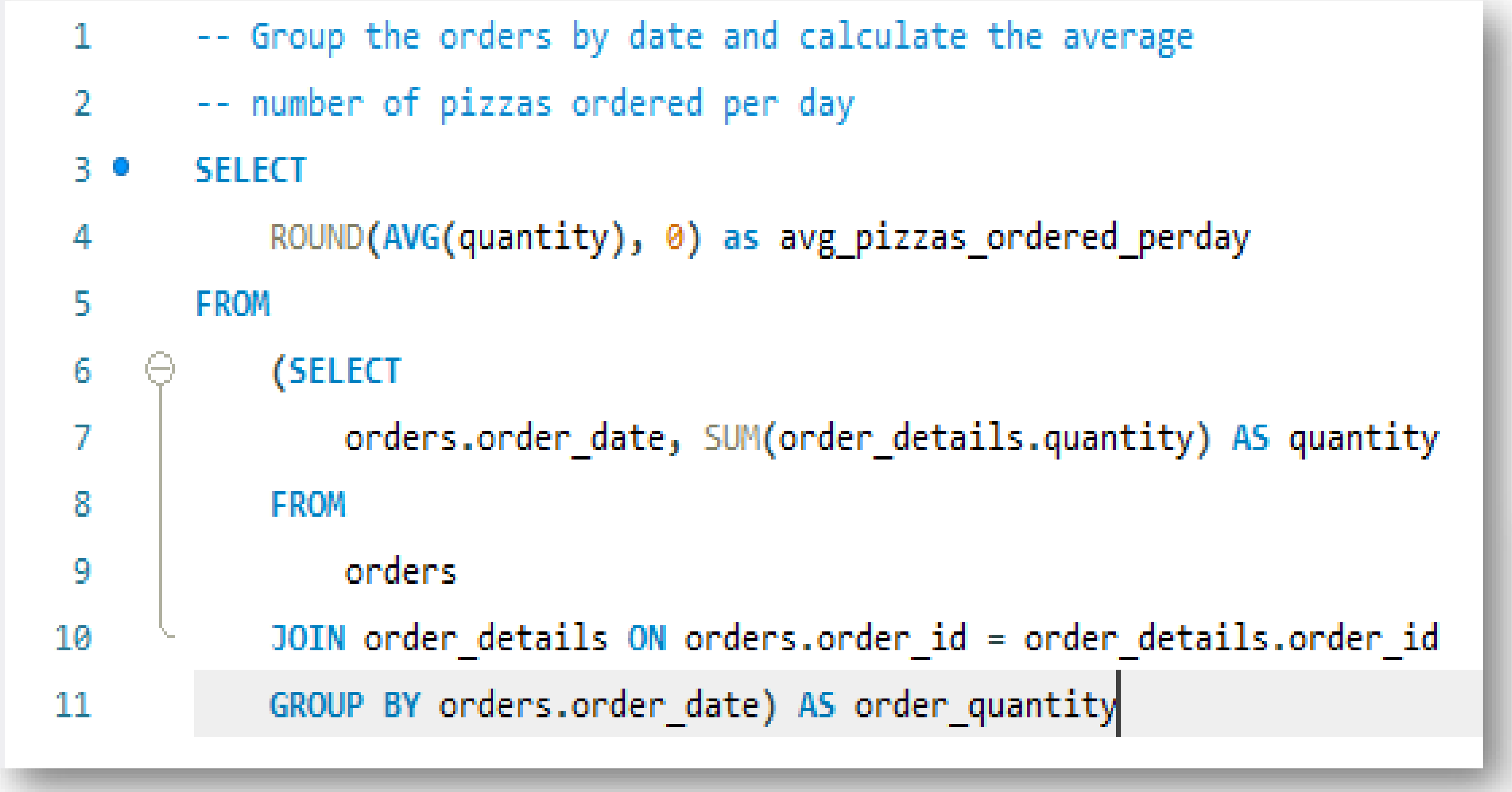


```sql
-- Join relevant tables to find the
-- category-wise distribution of pizzas
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category
```

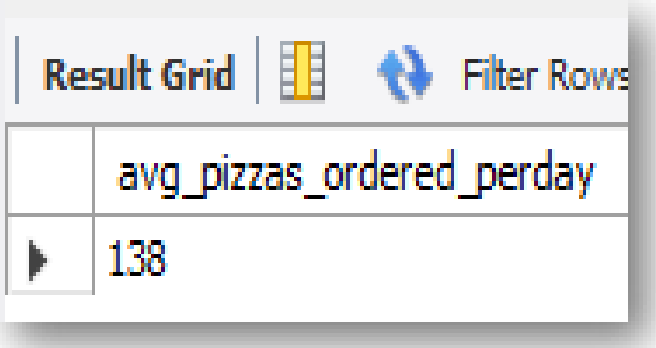


category	count(name)
Chicken	6
Classic	8
Supreme	9
Veggie	9

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

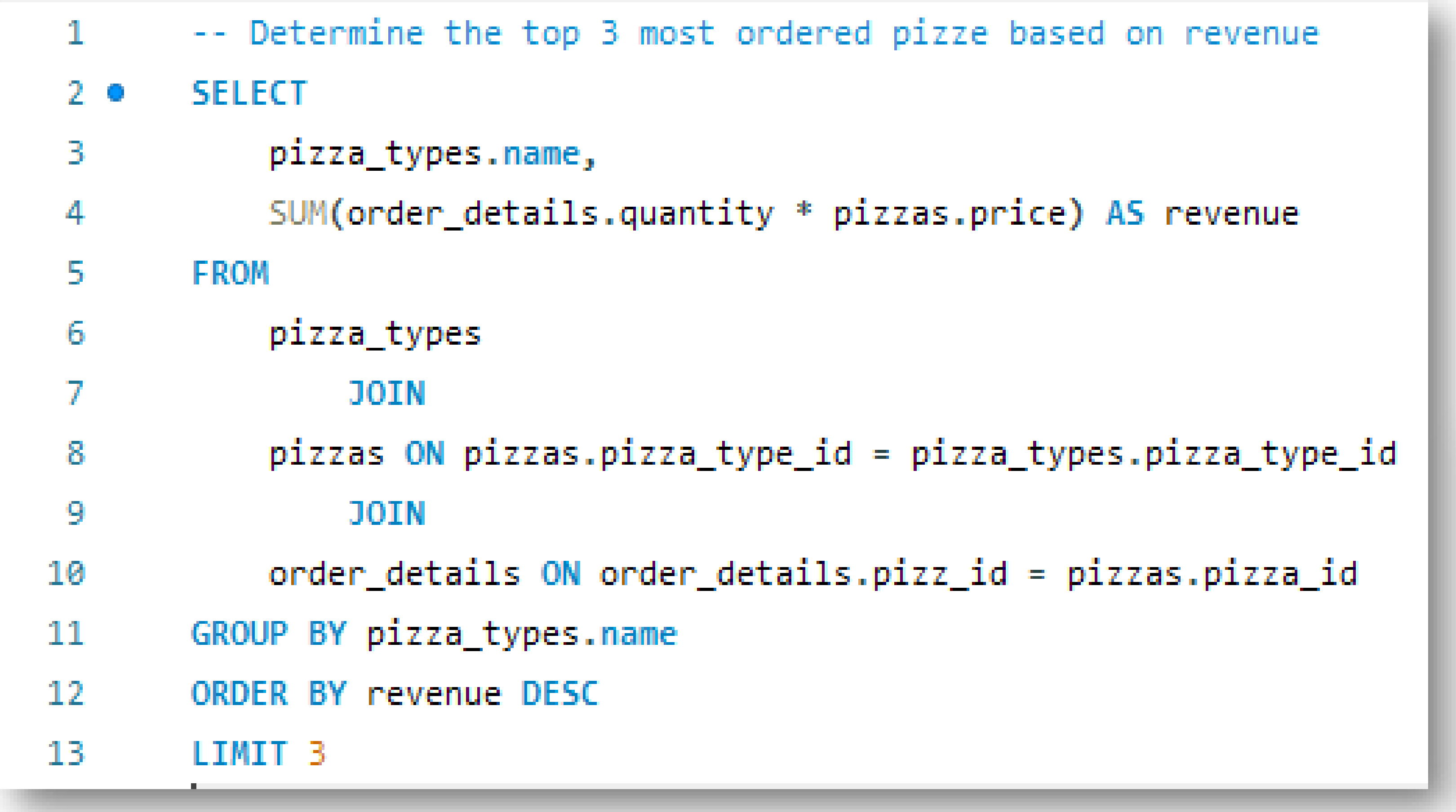


```sql
-- Group the orders by date and calculate the average
-- number of pizzas ordered per day
SELECT
    ROUND(AVG(quantity), 0) as avg_pizzas_ordered_perday
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS order_quantity
```

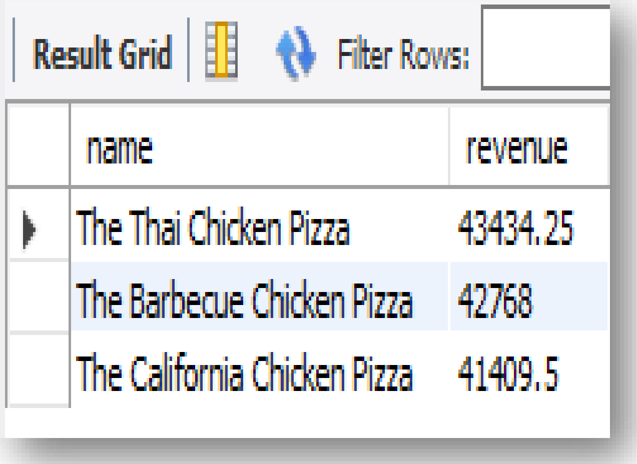


```
avg_pizzas_ordered_perday
138
```

Determine the top 3 most ordered pizzas based on revenue

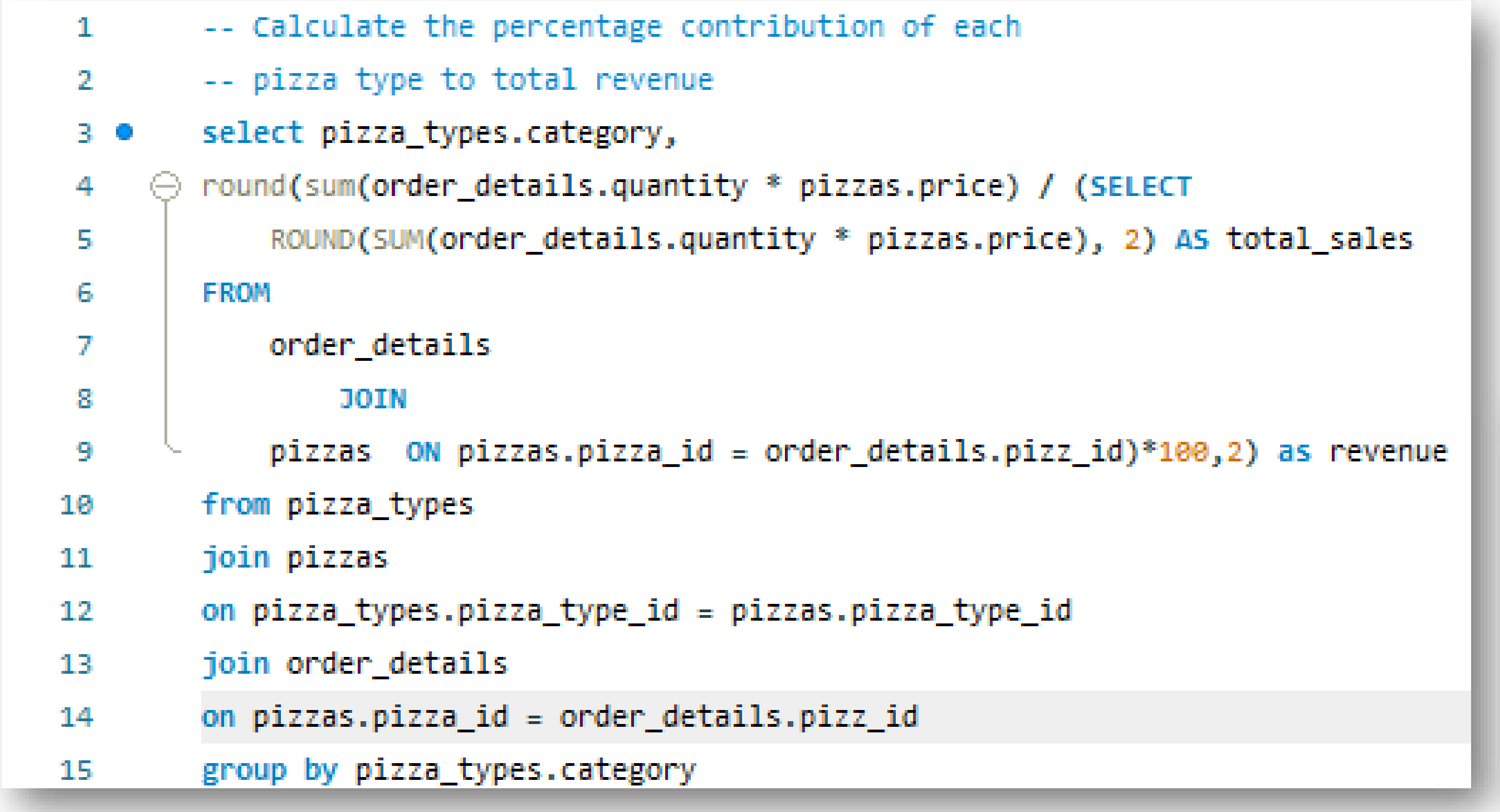


```
-- Determine the top 3 most ordered pizze based on revenue
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON order_details.pizz_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3
```

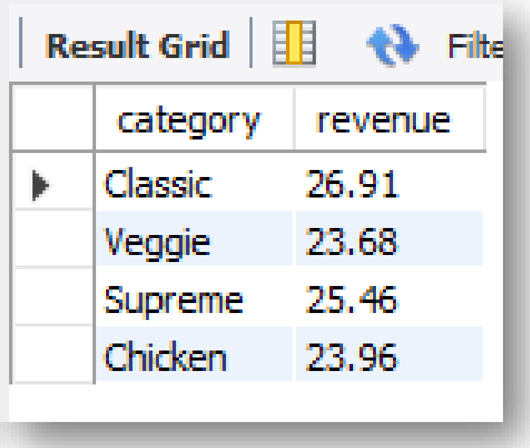


name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

Calculate the percentage contribution of each pizza type to total revenue

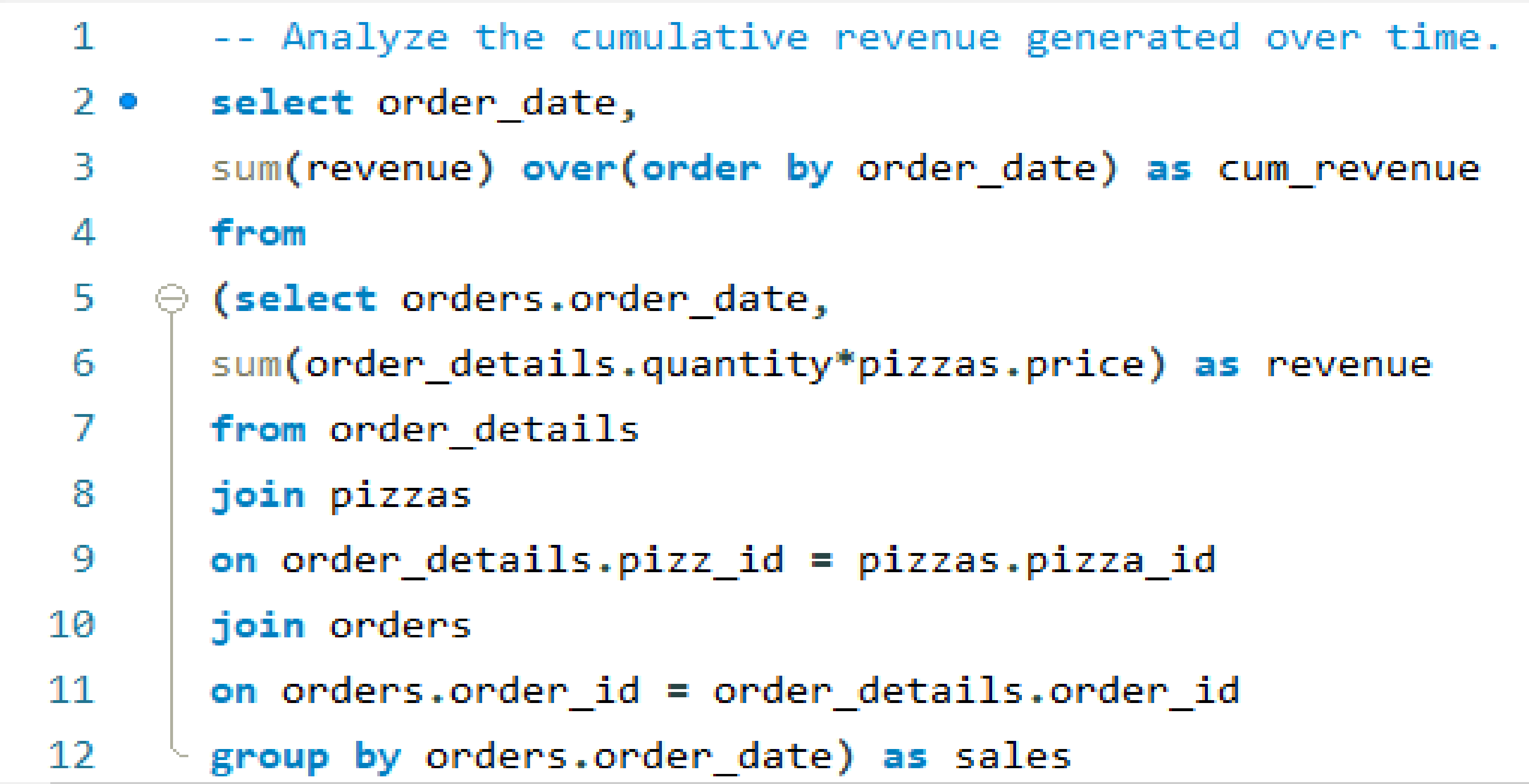


```
-- Calculate the percentage contribution of each
-- pizza type to total revenue
select pizza_types.category,
round(sum(order_details.quantity * pizzas.price) / (SELECT
    ROUND(SUM(order_details.quantity * pizzas.price), 2) AS total_sales
FROM
    order_details
        JOIN
    pizzas  ON pizzas.pizza_id = order_details.pizz_id)*100,2) as revenue
from pizza_types
join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
join order_details
on pizzas.pizza_id = order_details.pizz_id
group by pizza_types.category
```

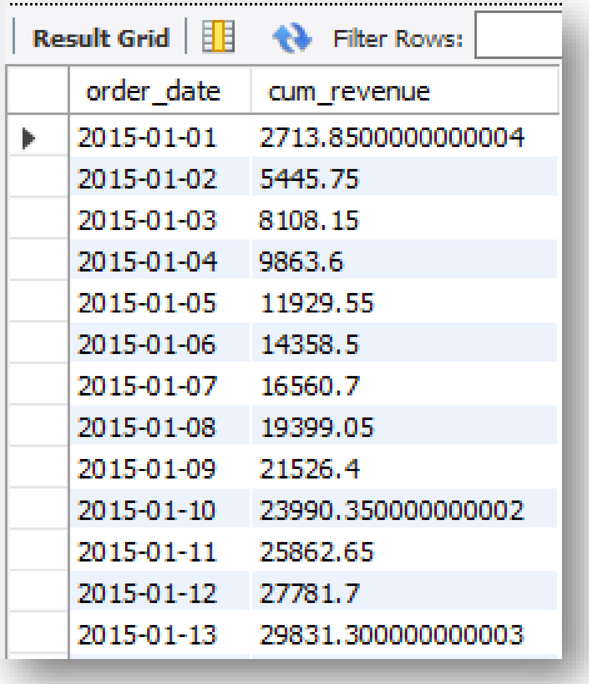


category	revenue
Classic	26.91
Veggie	23.68
Supreme	25.46
Chicken	23.96

Analyze the cumulative revenue generated over time

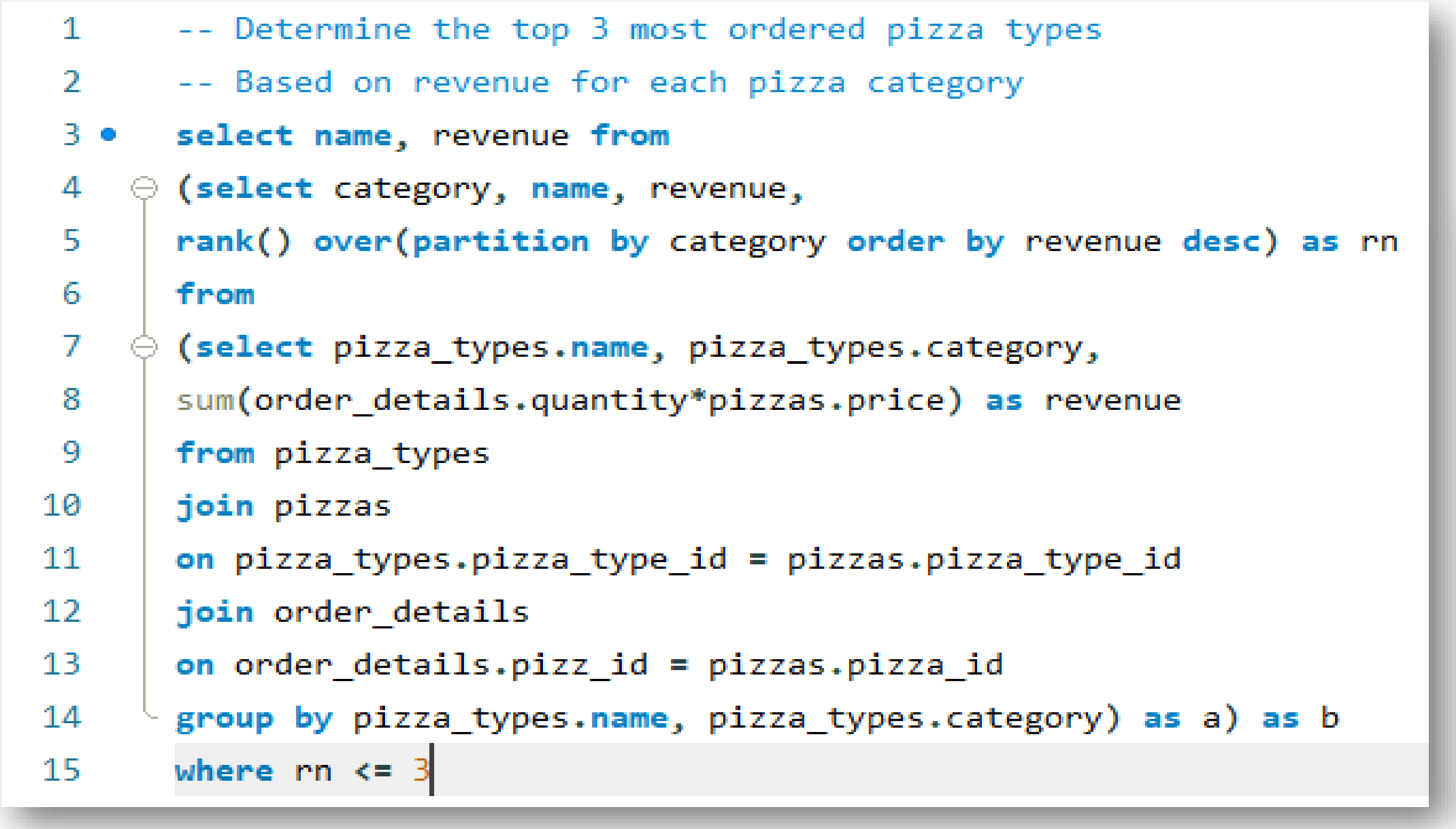


```sql
-- Analyze the cumulative revenue generated over time.
select order_date,
sum(revenue) over(order by order_date) as cum_revenue
from
(select orders.order_date,
sum(order_details.quantity*pizzas.price) as revenue
from order_details
join pizzas
on order_details.pizz_id = pizzas.pizza_id
join orders
on orders.order_id = order_details.order_id
group by orders.order_date) as sales
```

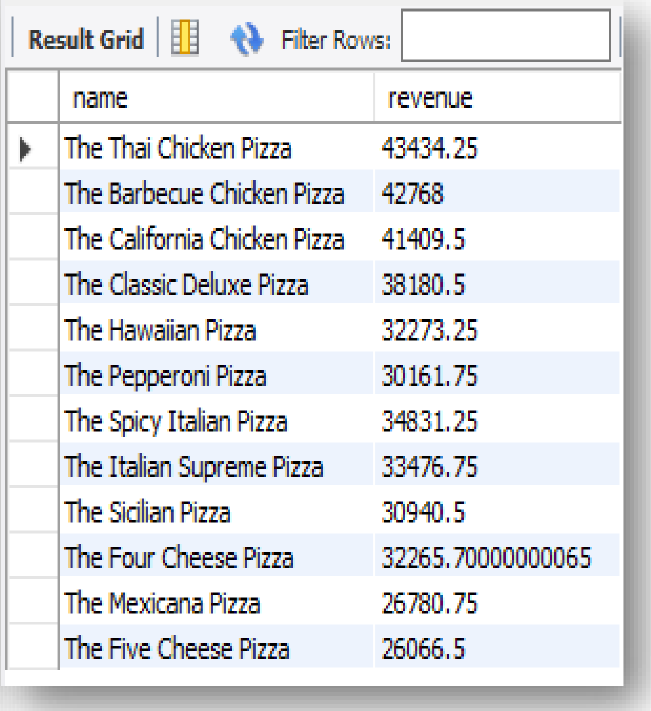


order_date	cum_revenue
2015-01-01	2713.8500000000004
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55
2015-01-06	14358.5
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526.4
2015-01-10	23990.350000000002
2015-01-11	25862.65
2015-01-12	27781.7
2015-01-13	29831.300000000003

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 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.name, pizza_types.category,
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.pizz_id = pizzas.pizza_id
group by pizza_types.name, pizza_types.category) as a) as b
where rn <= 3
```



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