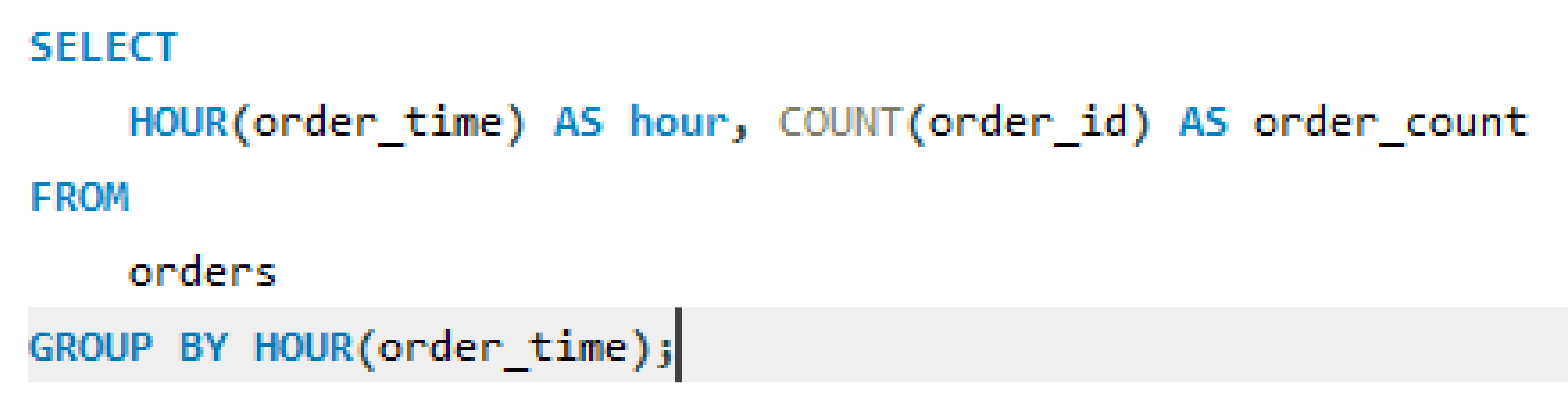
# Sales report

2024

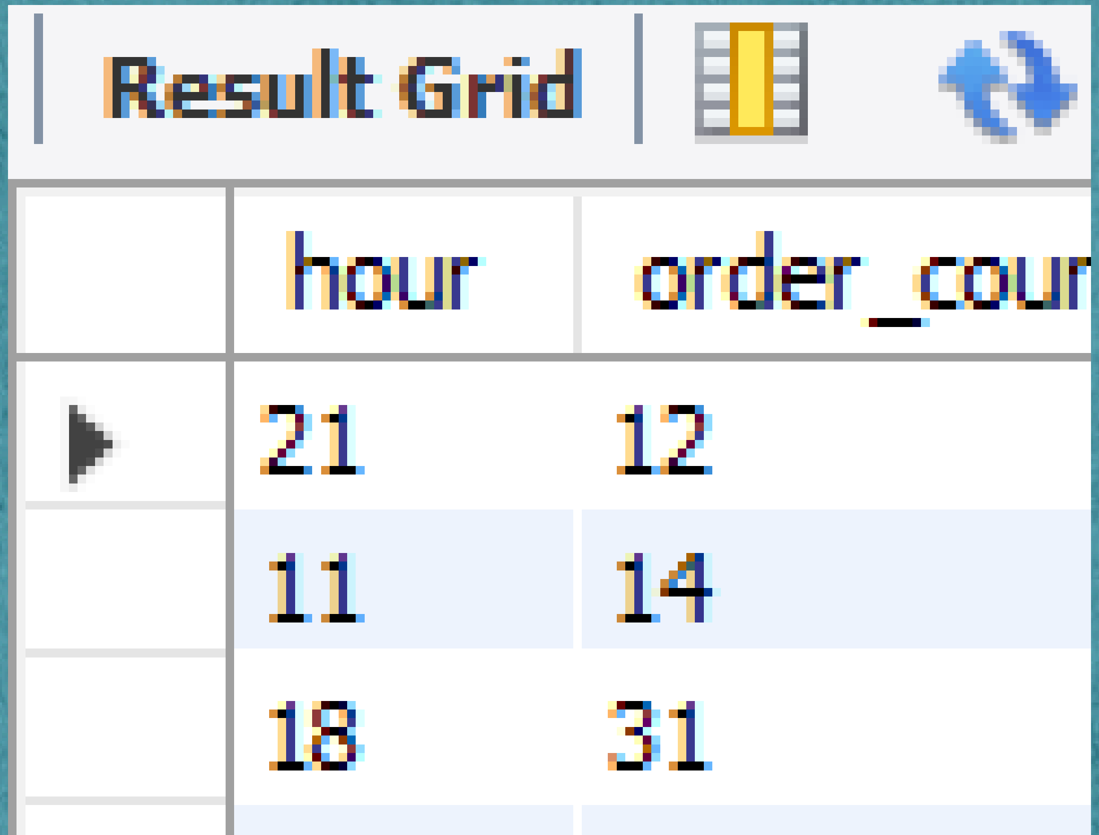
# Introduction

Hello, My name is Md Arshad and in this project, I have utilized SQL queries to solve questions that were related to pizza sales .

# determine the distribution of orders by hour of the day.

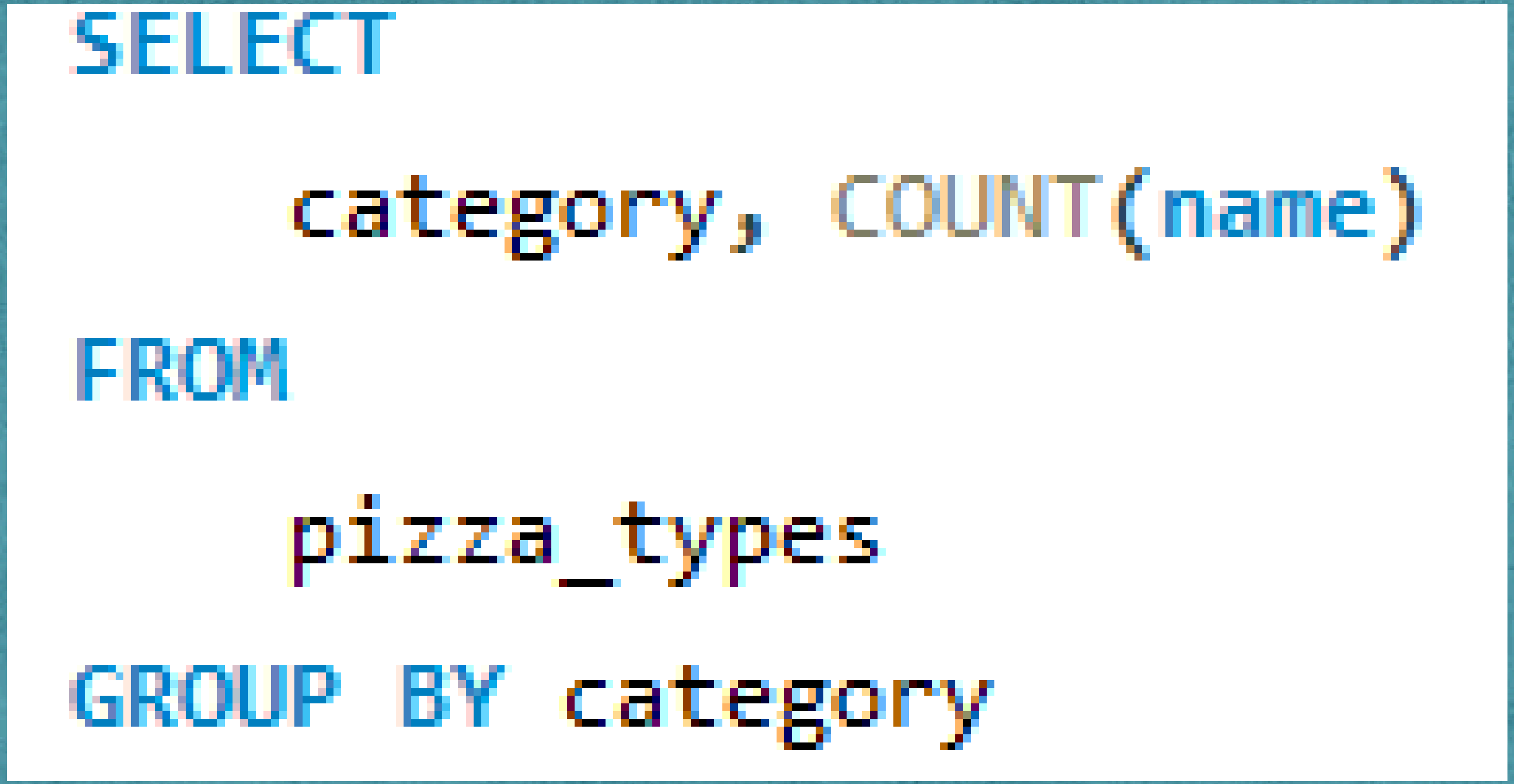


```
SELECT
    HOUR(order_time) AS hour, COUNT(order_id) AS order_count
FROM
    orders
GROUP BY HOUR(order_time);
```



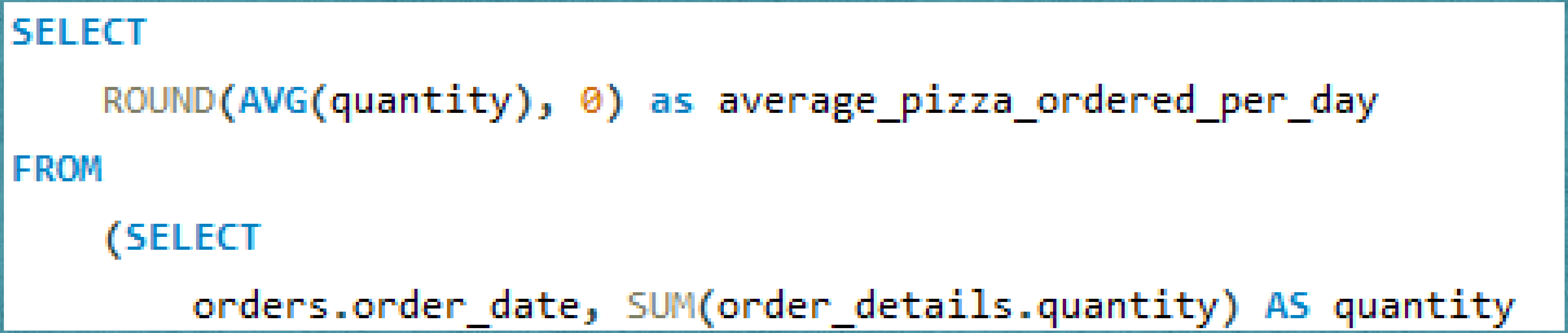
| hour | order_cour |
|---|---|
| 21 | 12 |
| 11 | 14 |
| 18 | 31 |

# join relevant tables to find the category wise distribution of pizza.

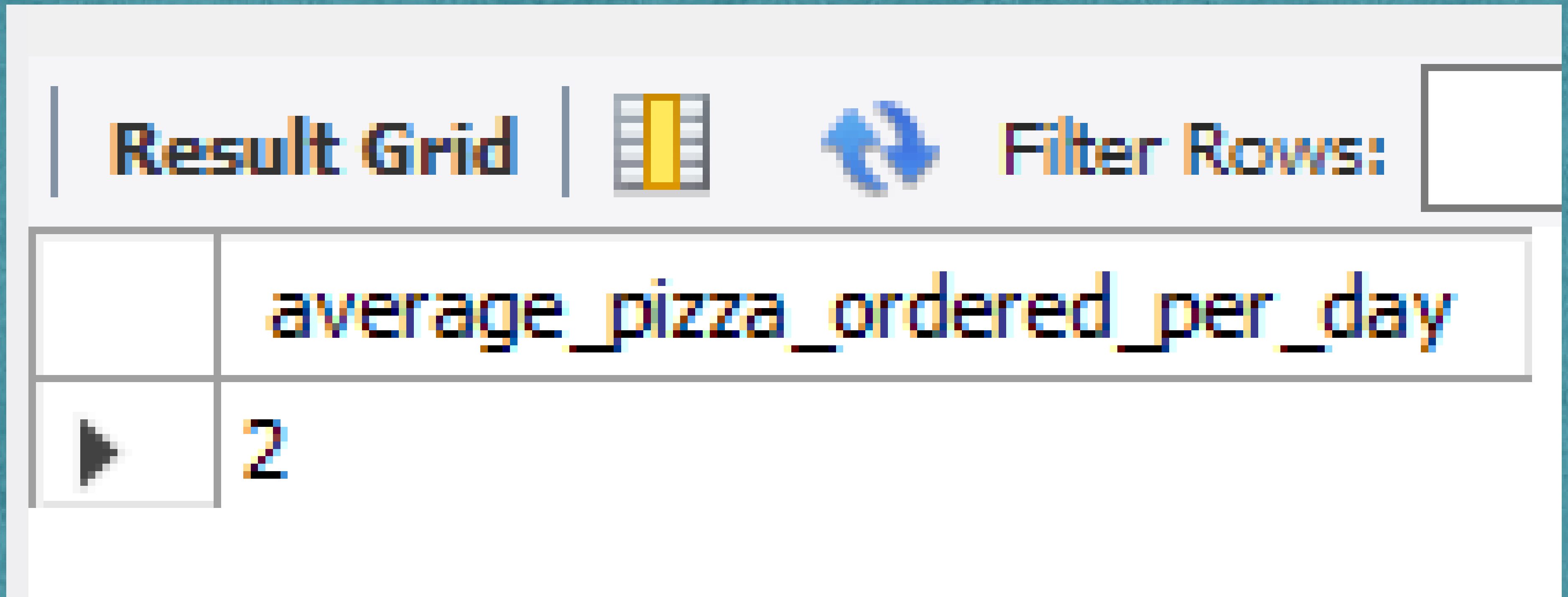


```
SELECT

    category, COUNT(name)

FROM

    pizza_types

GROUP BY category
```

# group the order by date and calculate the average number of order per day.

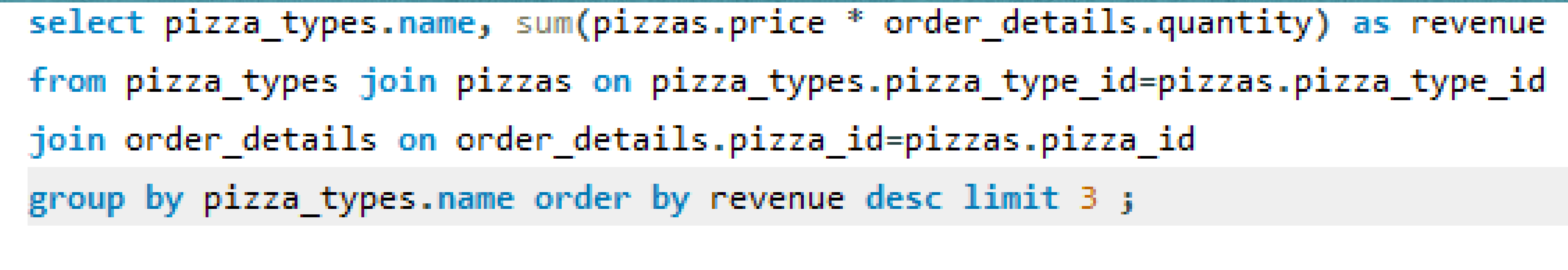


```
SELECT
    ROUND(AVG(quantity), 0) as average_pizza_ordered_per_day
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS quantity
```

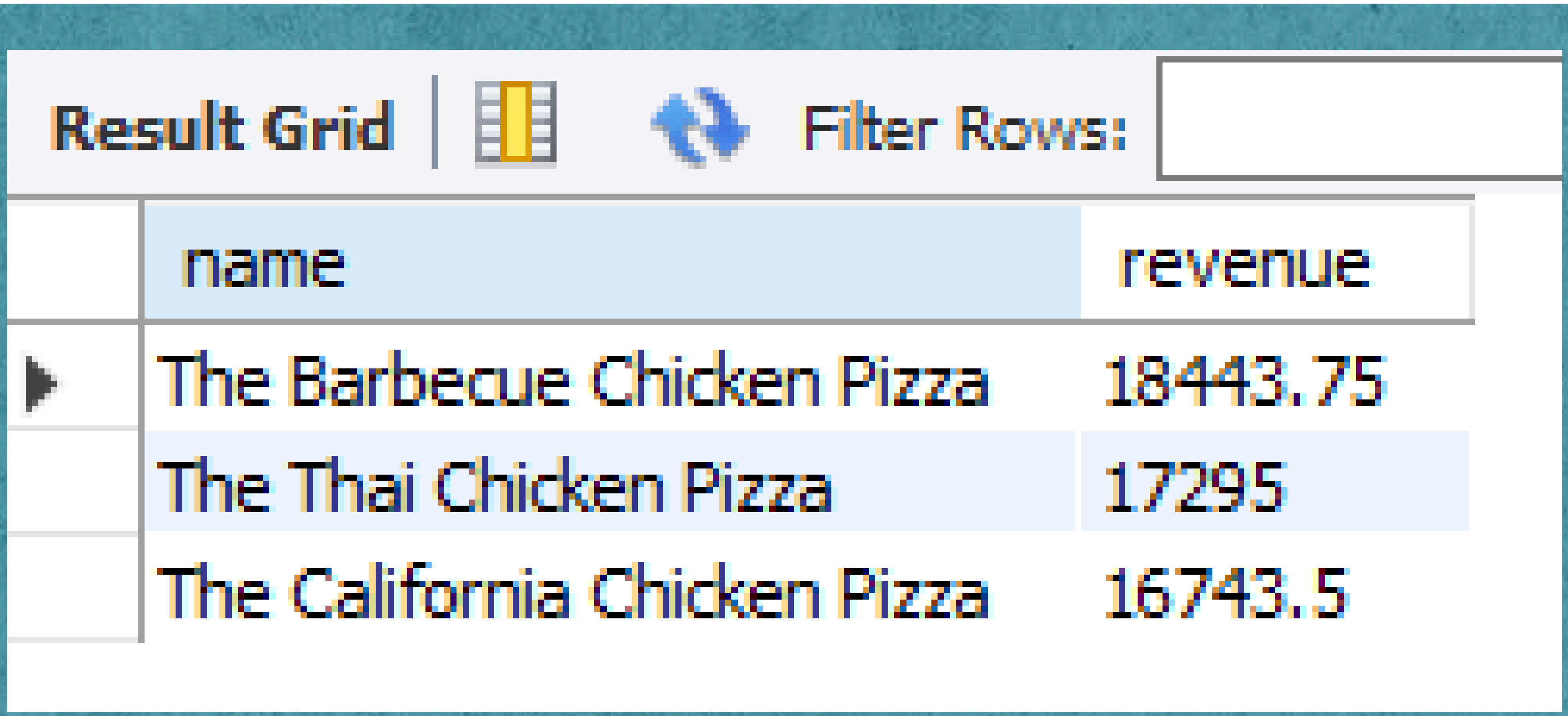


```
average_pizza_ordered_per_day
2
```

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

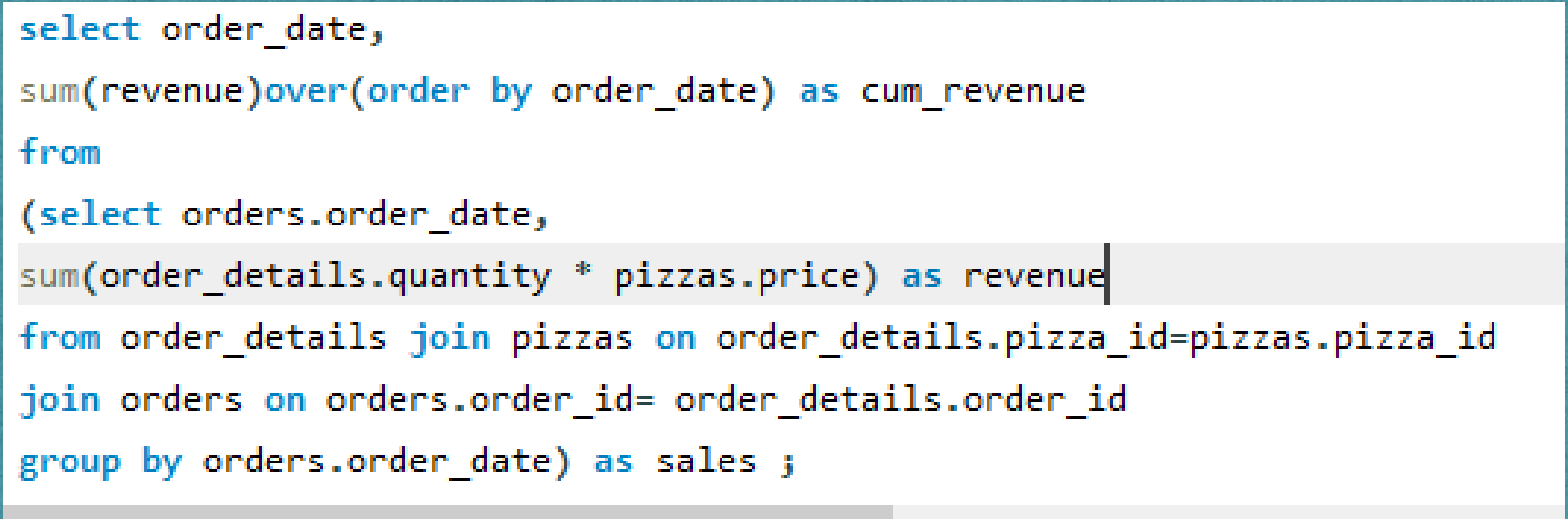


```
select pizza_types.name, sum(pizzas.price * order_details.quantity) as revenue
from pizza_types join pizzas on pizza_types.pizza_type_id=pizzas.pizza_type_id
join order_details on order_details.pizza_id=pizzas.pizza_id
group by pizza_types.name order by revenue desc limit 3 ;
```

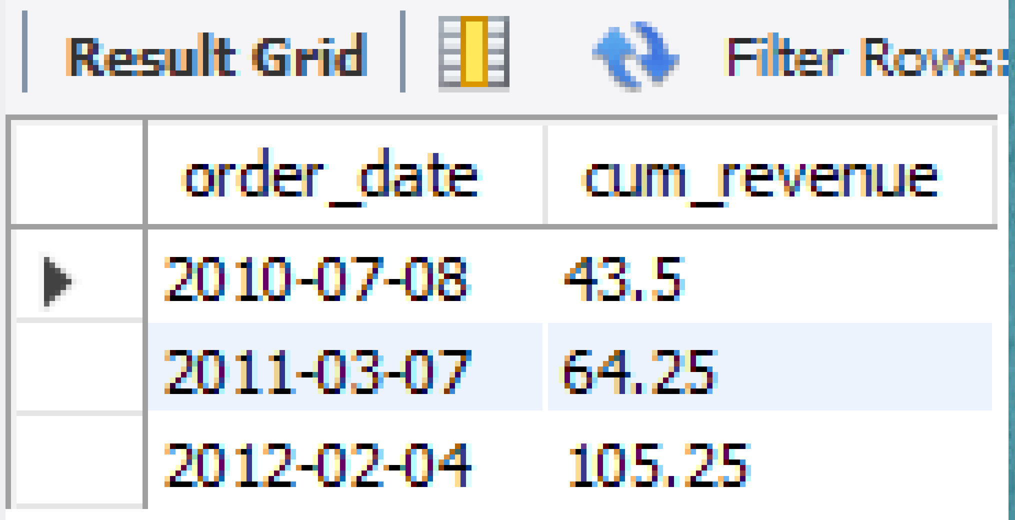


| name | revenue |
|---|---|
| The Barbecue Chicken Pizza | 18443.75 |
| The Thai Chicken Pizza | 17295 |
| The California Chicken Pizza | 16743.5 |

# analyze the cumulative revenue generated over time.

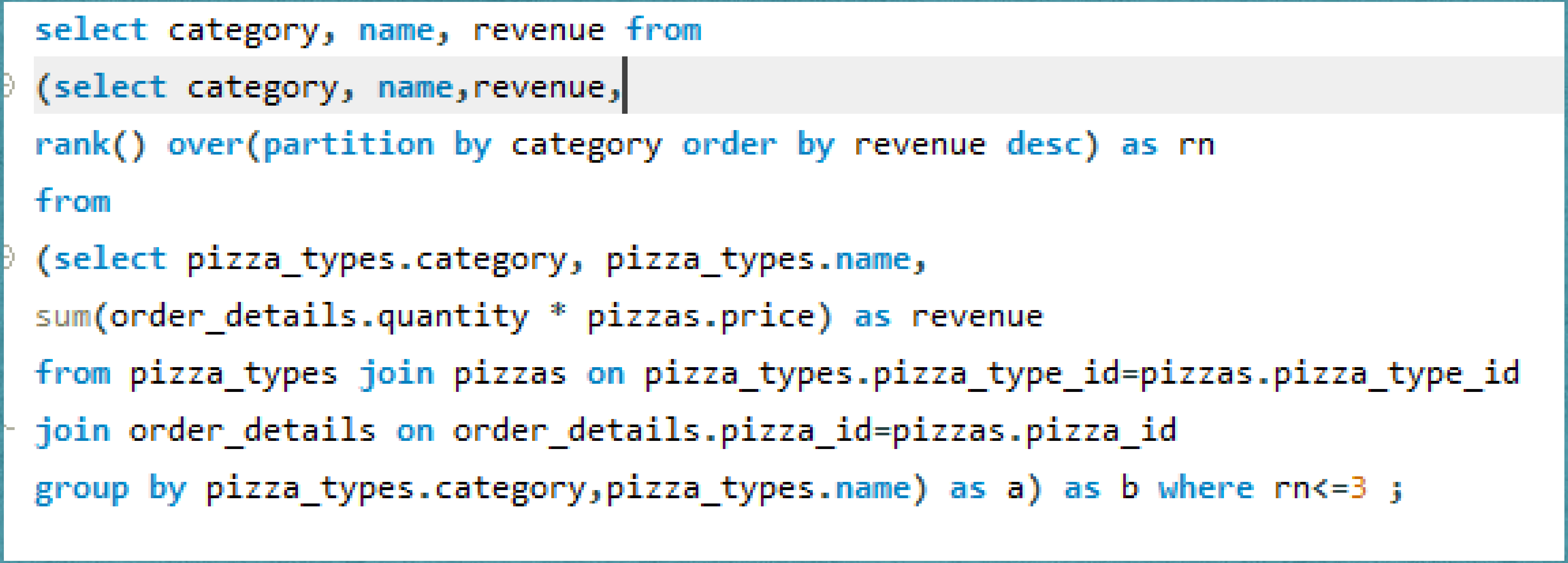


```
select order_date,
sum(revenue)over(order by order_date) as cum_revenue
from
(select orders.order_date,
sum(order_details.quantity * pizzas.price) as revenue
from order_details join pizzas on order_details.pizza_id=pizzas.pizza_id
join orders on orders.order_id= order_details.order_id
group by orders.order_date) as sales ;
```

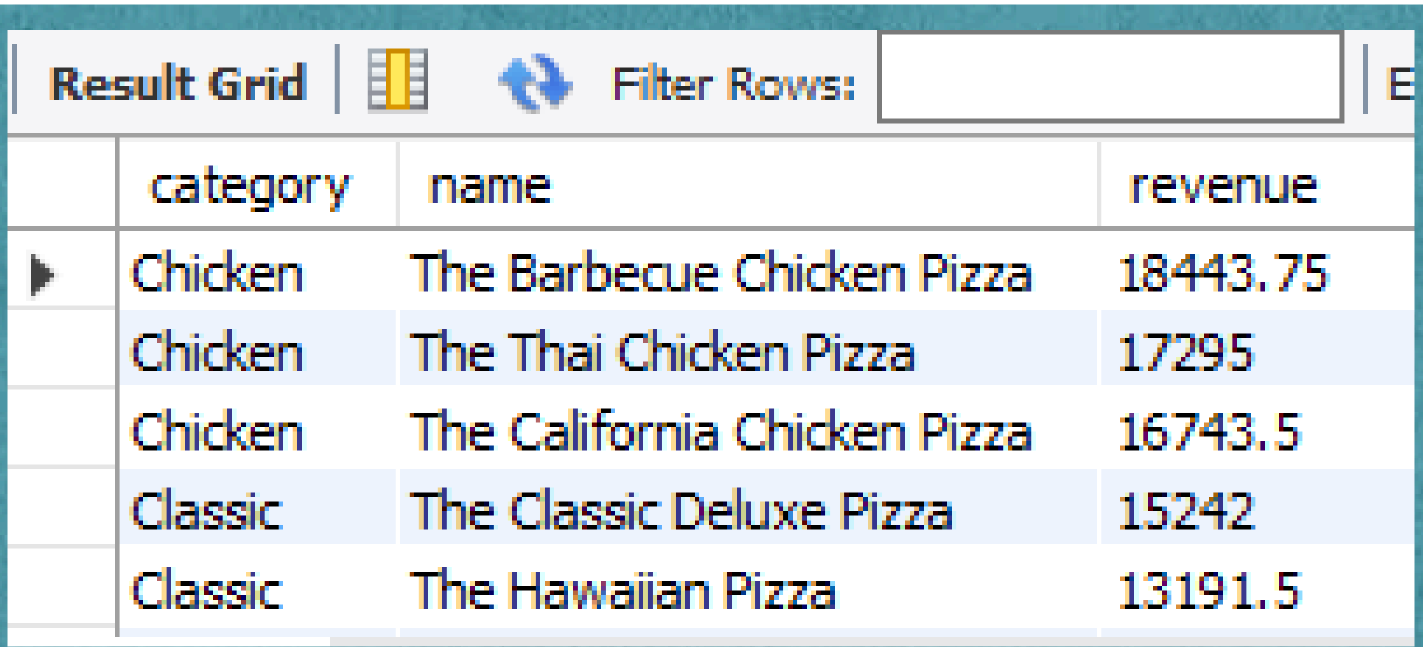


| order_date | cum_revenue |
|---|---|
| 2010-07-08 | 43.5 |
| 2011-03-07 | 64.25 |
| 2012-02-04 | 105.25 |

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

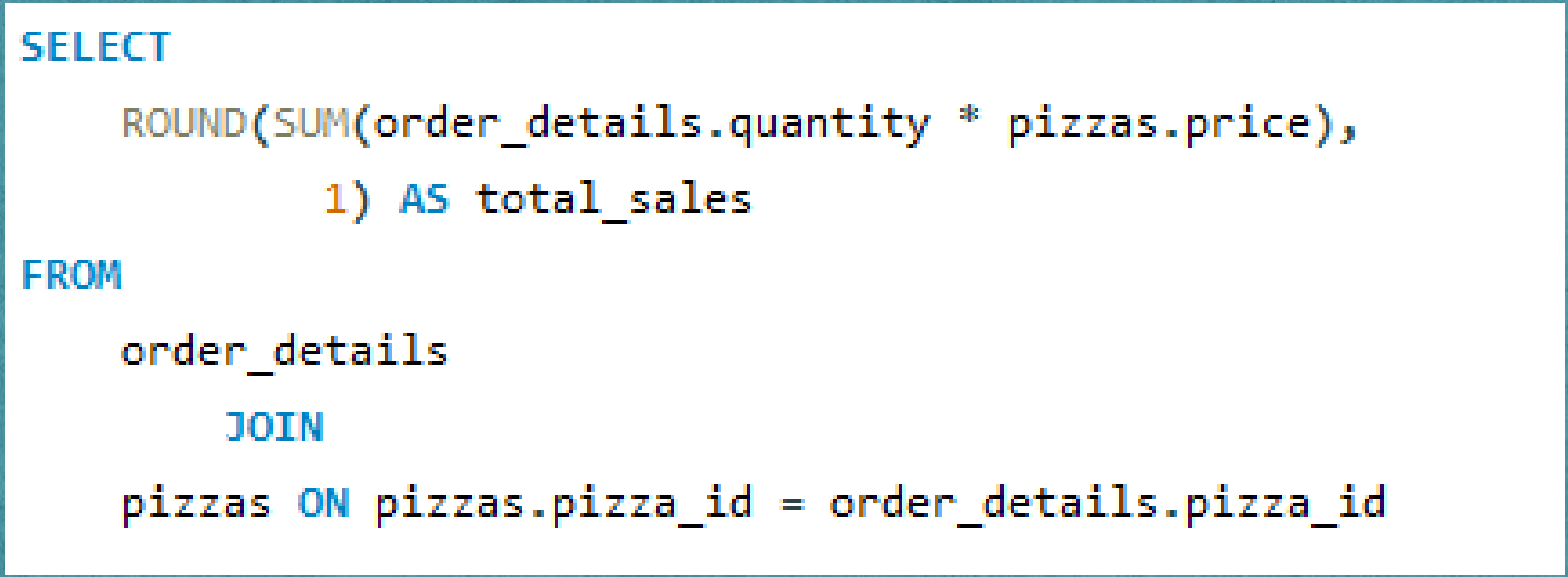


```sql
select category, name, revenue from
(select category, name,revenue,
rank() over(partition by category order by revenue desc) as rn
from
(select pizza_types.category, pizza_types.name,
sum(order_details.quantity * pizzas.price) as revenue
from pizza_types join pizzas on pizza_types.pizza_type_id=pizzas.pizza_type_id
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 ;
```

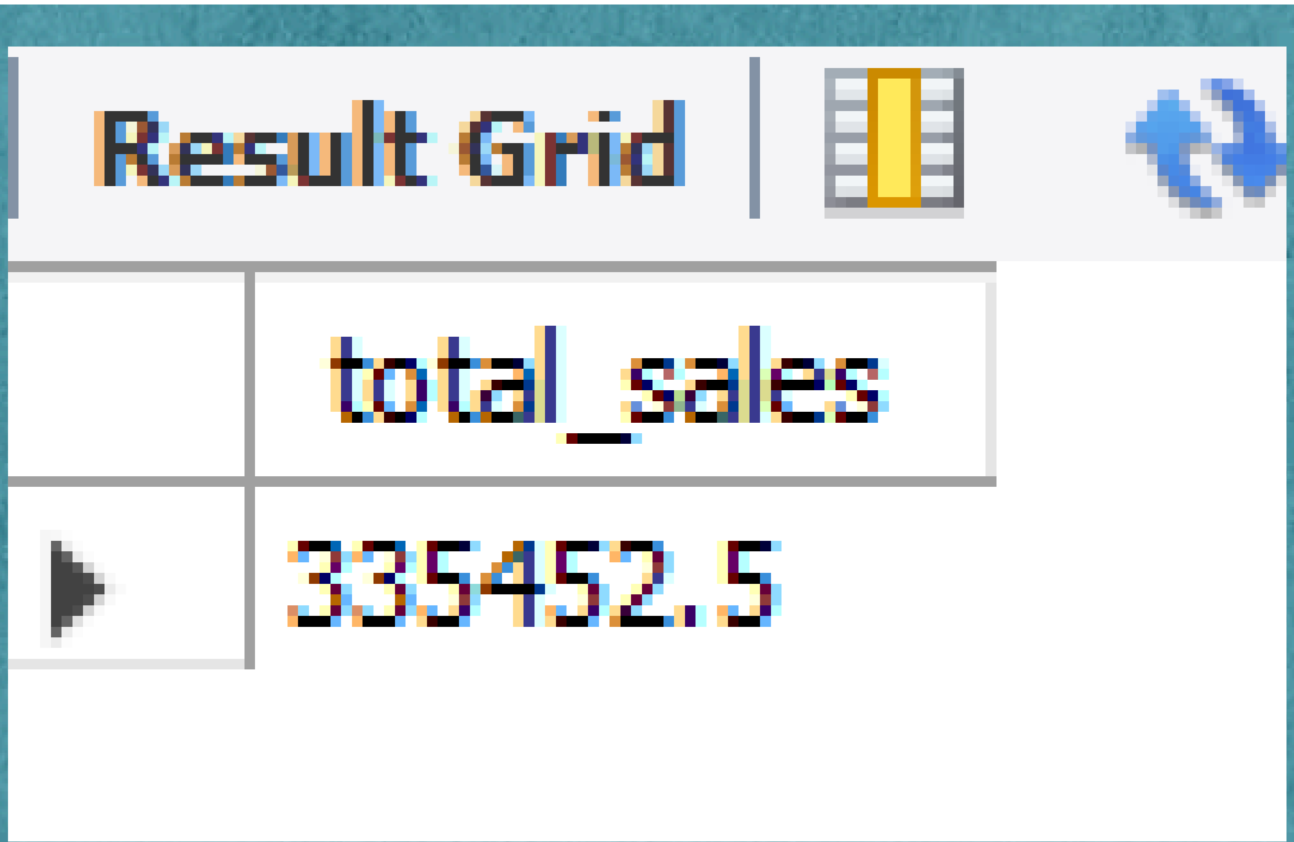


| category | name | revenue |
|---|---|---|
| Chicken | The Barbecue Chicken Pizza | 18443.75 |
| Chicken | The Thai Chicken Pizza | 17295 |
| Chicken | The California Chicken Pizza | 16743.5 |
| Classic | The Classic Deluxe Pizza | 15242 |
| Classic | The Hawaiian Pizza | 13191.5 |

# calculate the total revenue generated from pizza sales.

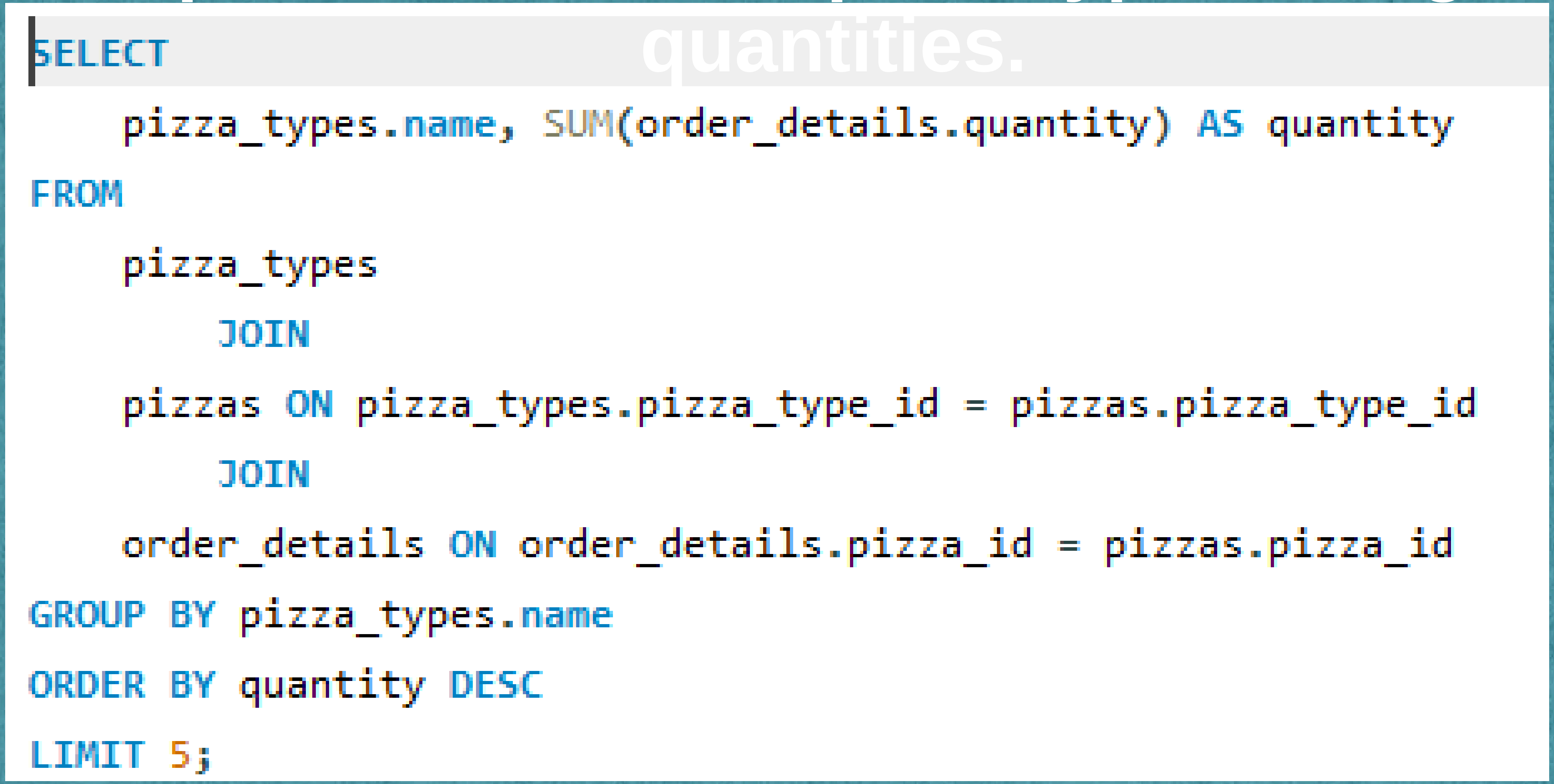


```
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            1) AS total_sales
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id
```

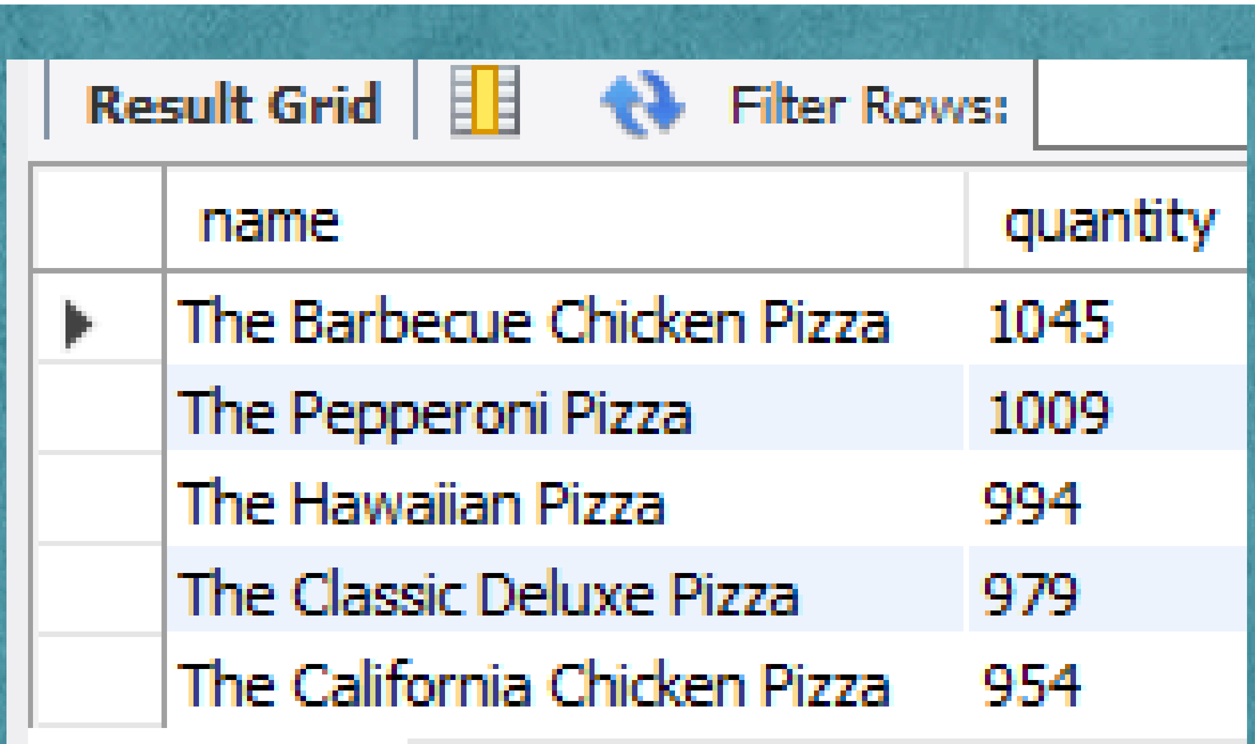


| total_sales |
|---|
| 335452.5 |

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

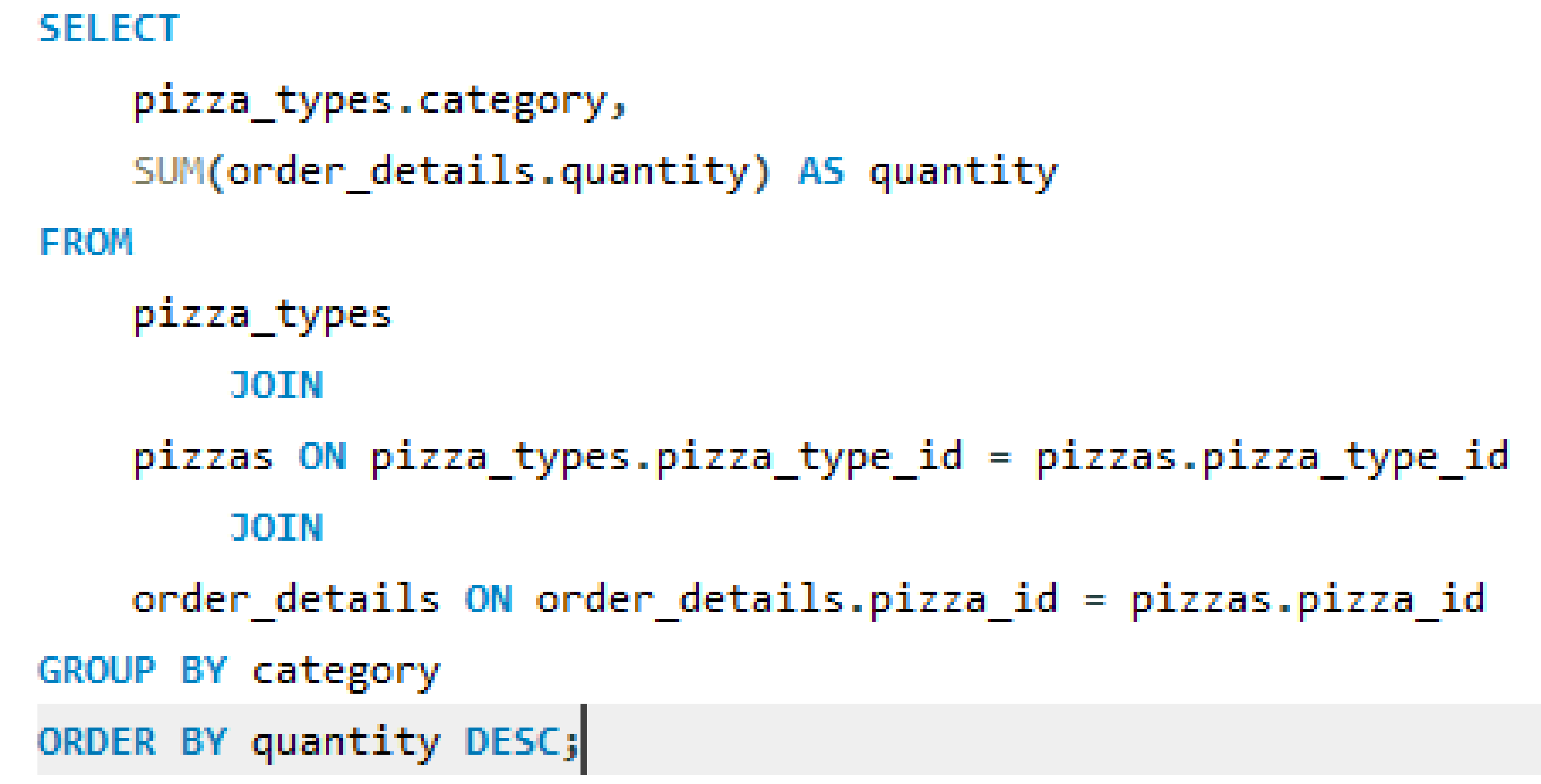


```
SELECT
    pizza_types.name, SUM(order_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC
LIMIT 5;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

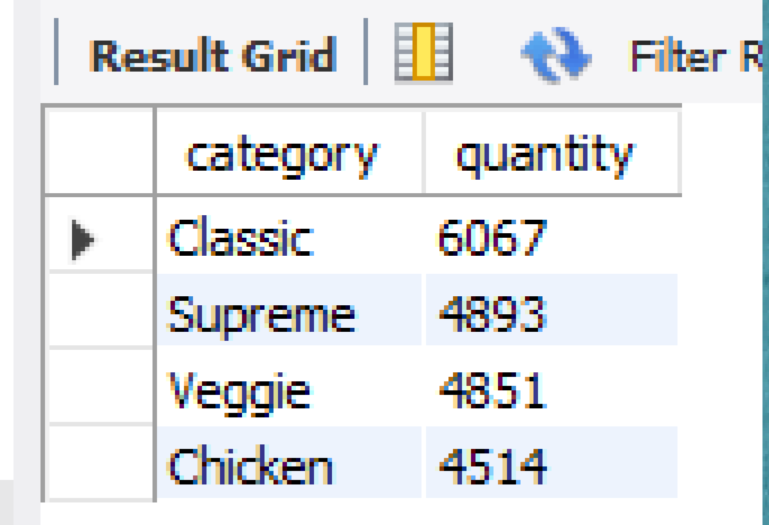


| name | quantity |
|---|---|
| The Barbecue Chicken Pizza | 1045 |
| The Pepperoni Pizza | 1009 |
| The Hawaiian Pizza | 994 |
| The Classic Deluxe Pizza | 979 |
| The California Chicken Pizza | 954 |

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

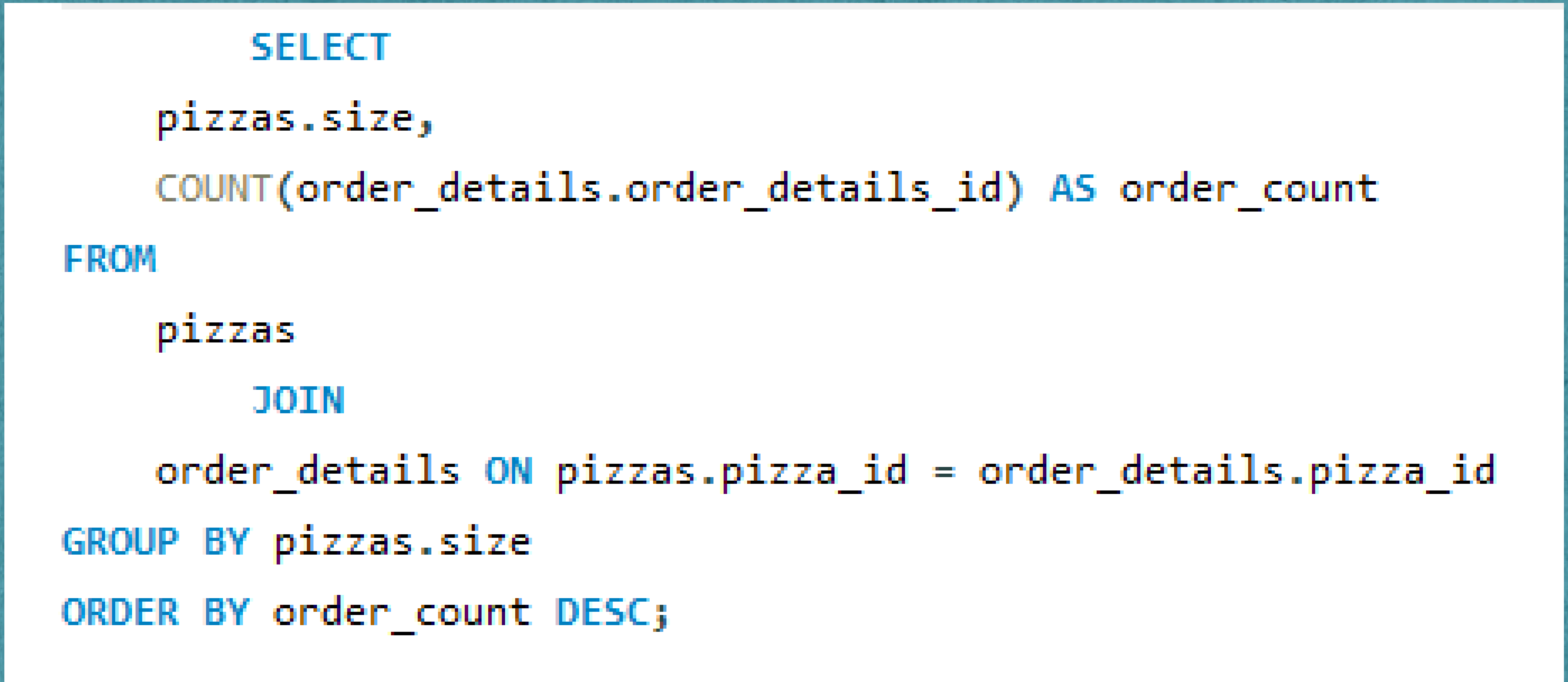


```sql
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY category
ORDER BY quantity DESC;
```

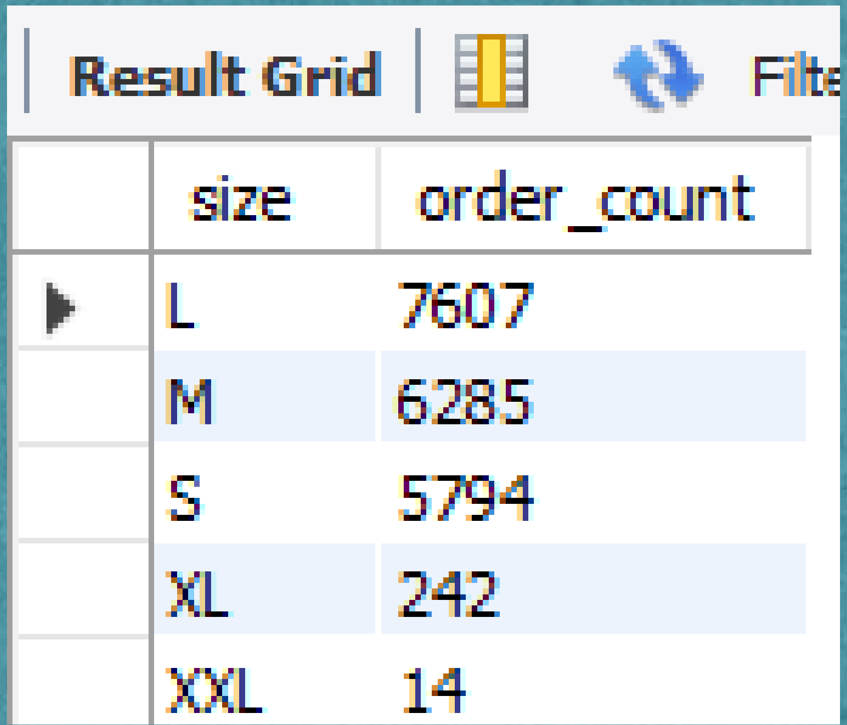


| category | quantity |
|---|---|
| Classic | 6067 |
| Supreme | 4893 |
| Veggie | 4851 |
| Chicken | 4514 |

# identify the most common pizza size order.

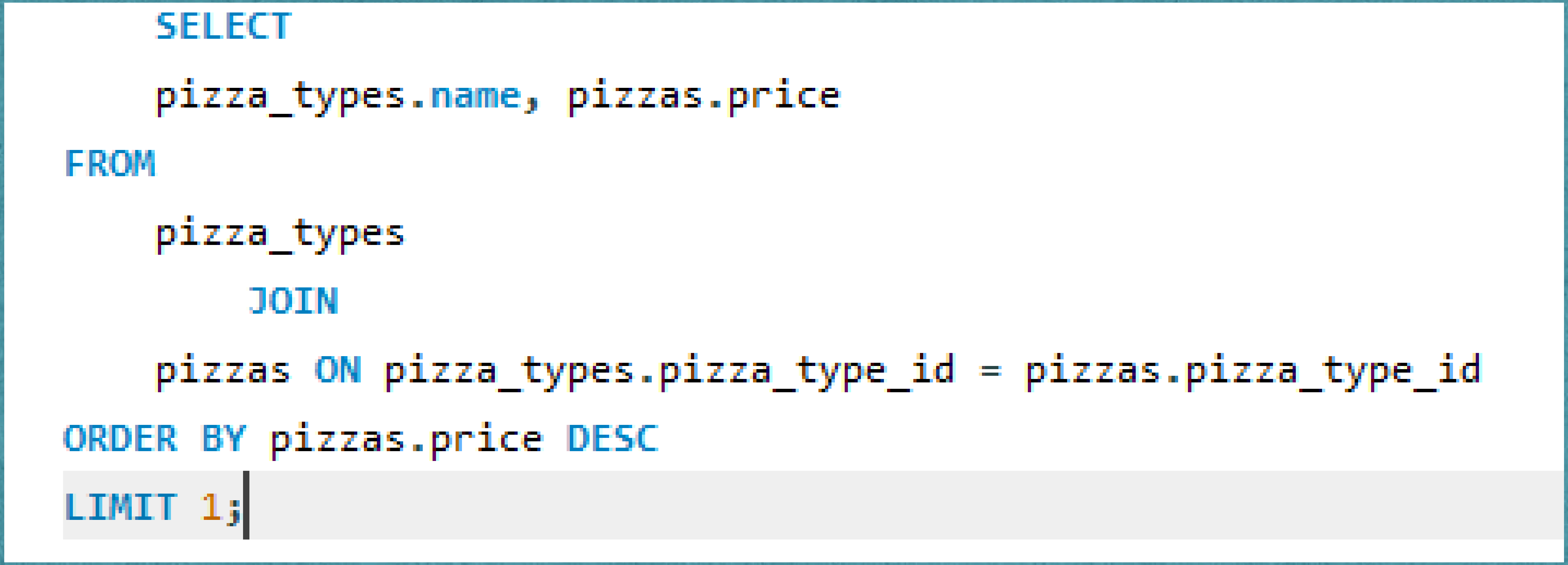


```
        SELECT
    pizzas.size,
    COUNT(order_details.order_details_id) AS order_count
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC;
```

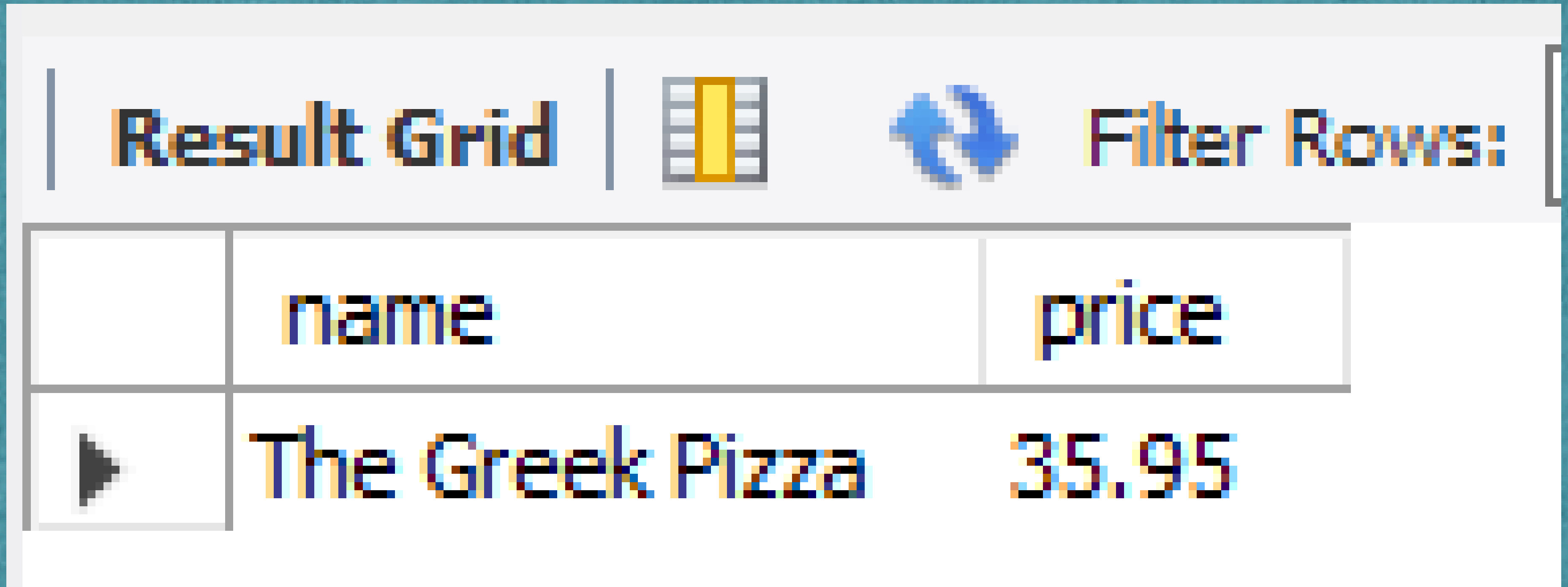


| size | order_count |
|---|---|
| L | 7607 |
| M | 6285 |
| S | 5794 |
| XL | 242 |
| XXL | 14 |

# 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 |

# Contact

☎ 7870739904

✉ mdarshad130123@gmail.com

🌐 https://arshad7u8u.github.io/mdarshad.github.io/

# Thank you!

2024