SQL project on Pizza sales

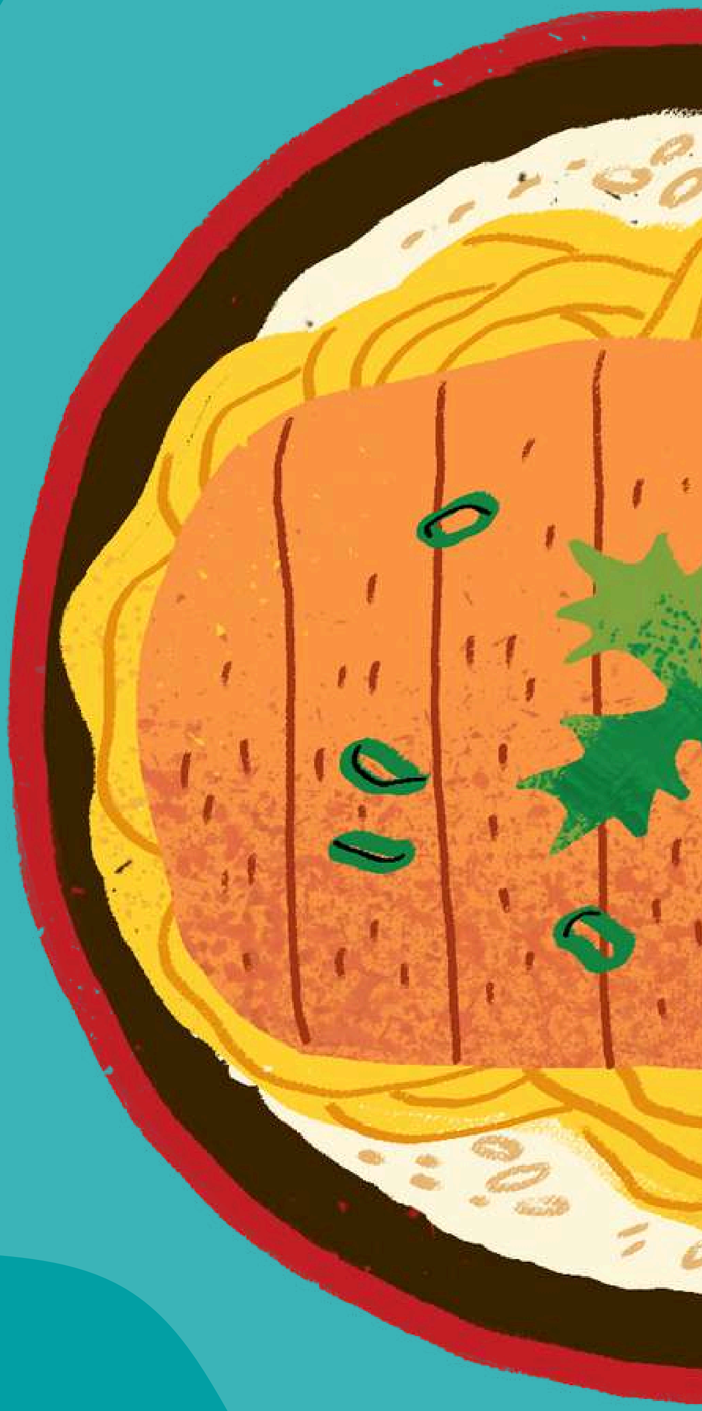
PLAY

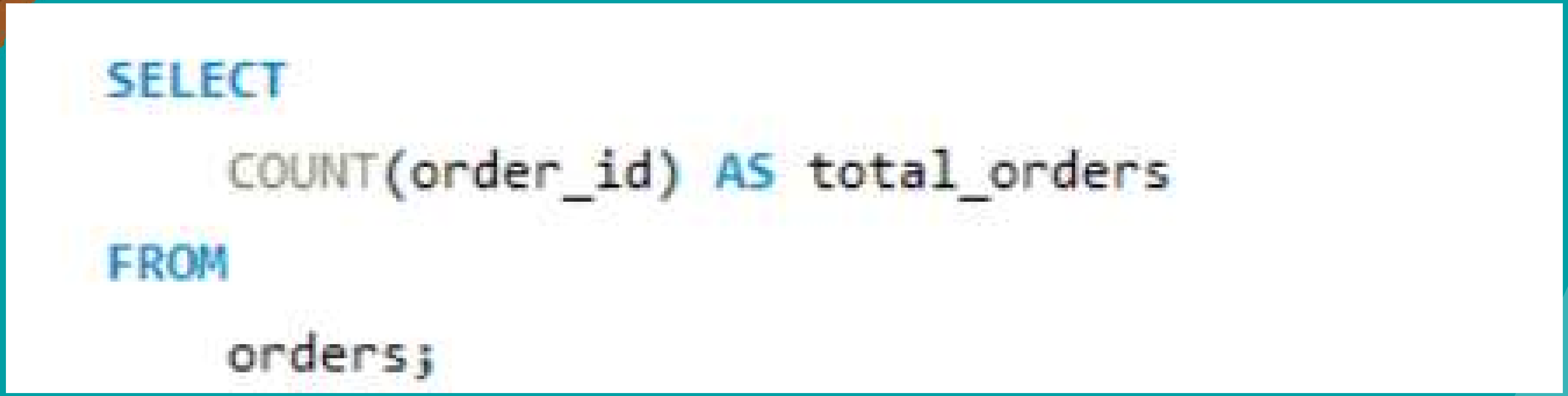
Hello!

Hello everyone, my name is Nikhil Chauhan, and I am excited to present my project on pizza sales analysis using SQL queries. This project aims to explore and analyze the sales patterns, customer preferences, and overall performance of a pizza business through comprehensive data manipulation and querying techniques. By leveraging SQL, I have extracted valuable insights that can help optimize the business operations and enhance customer satisfaction

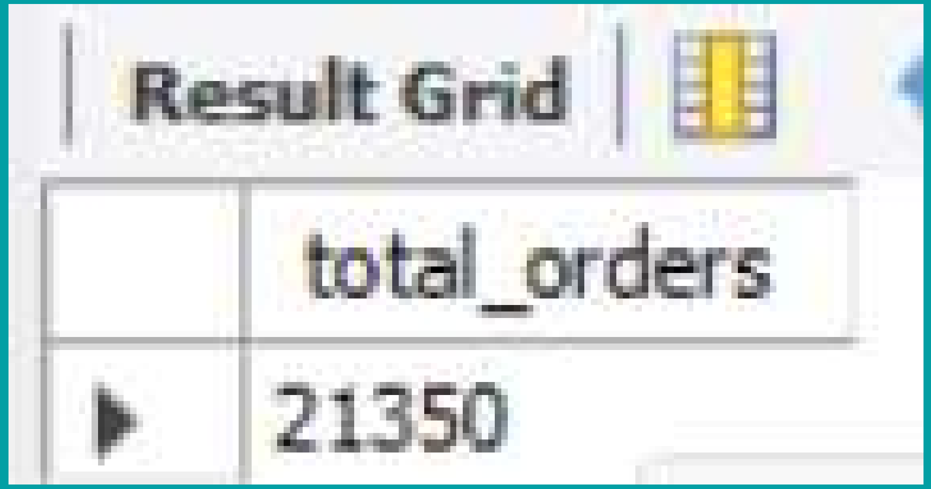
WHAT IS IT?

RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED



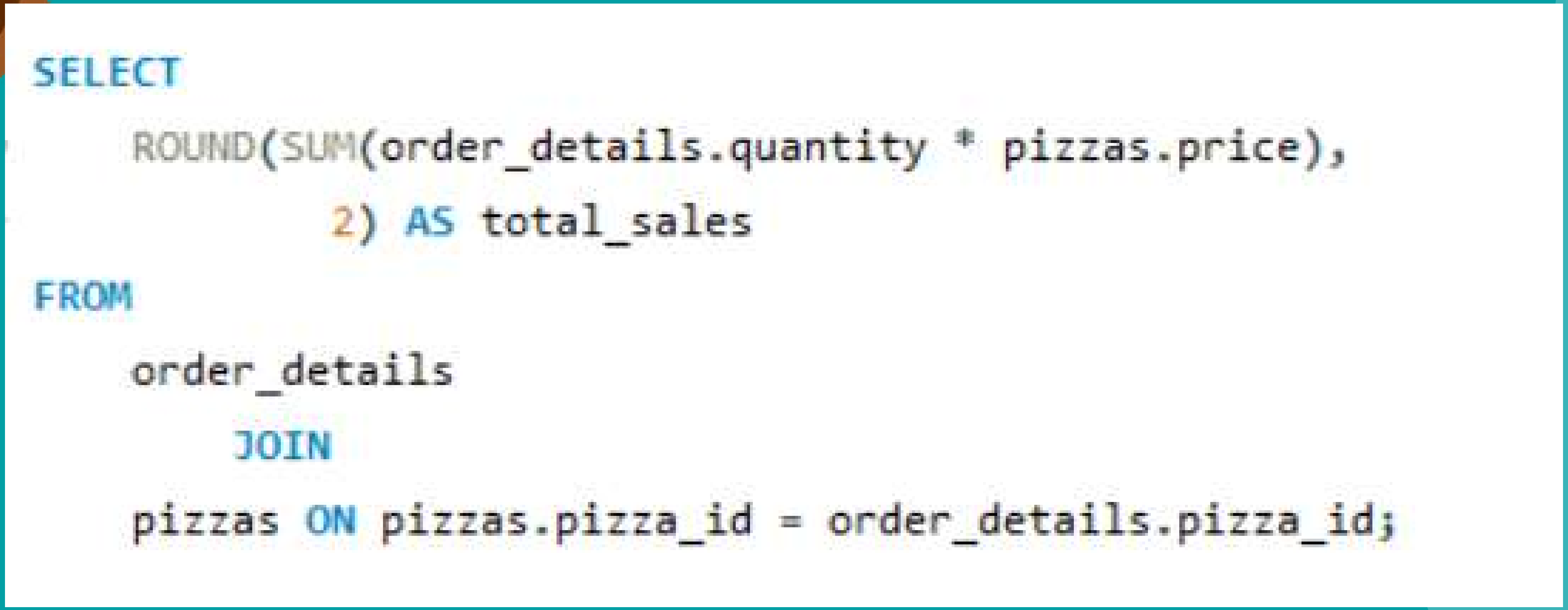


```
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

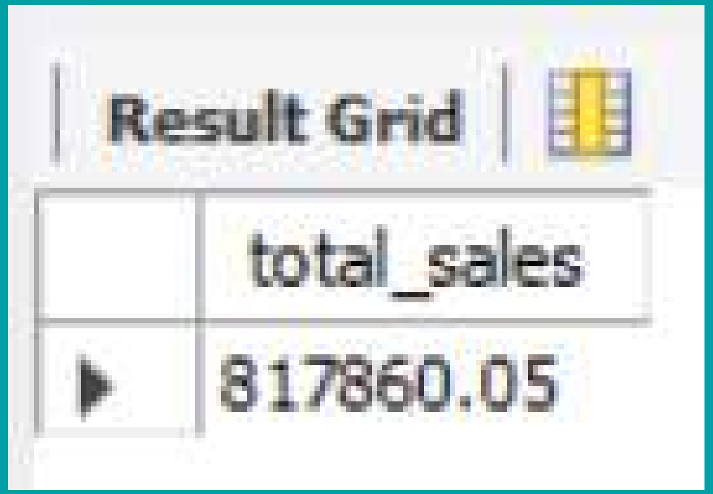


```
total_orders
21350
```

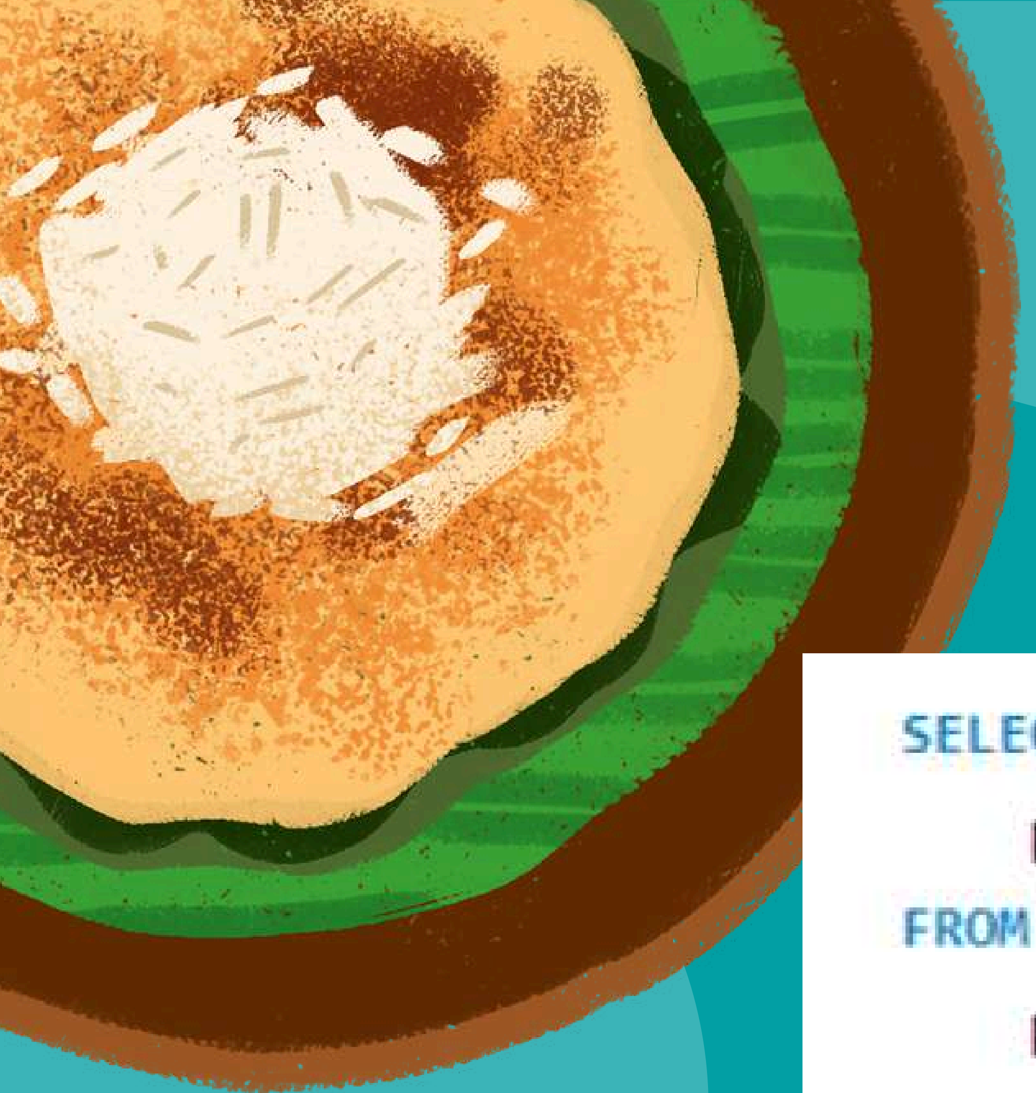
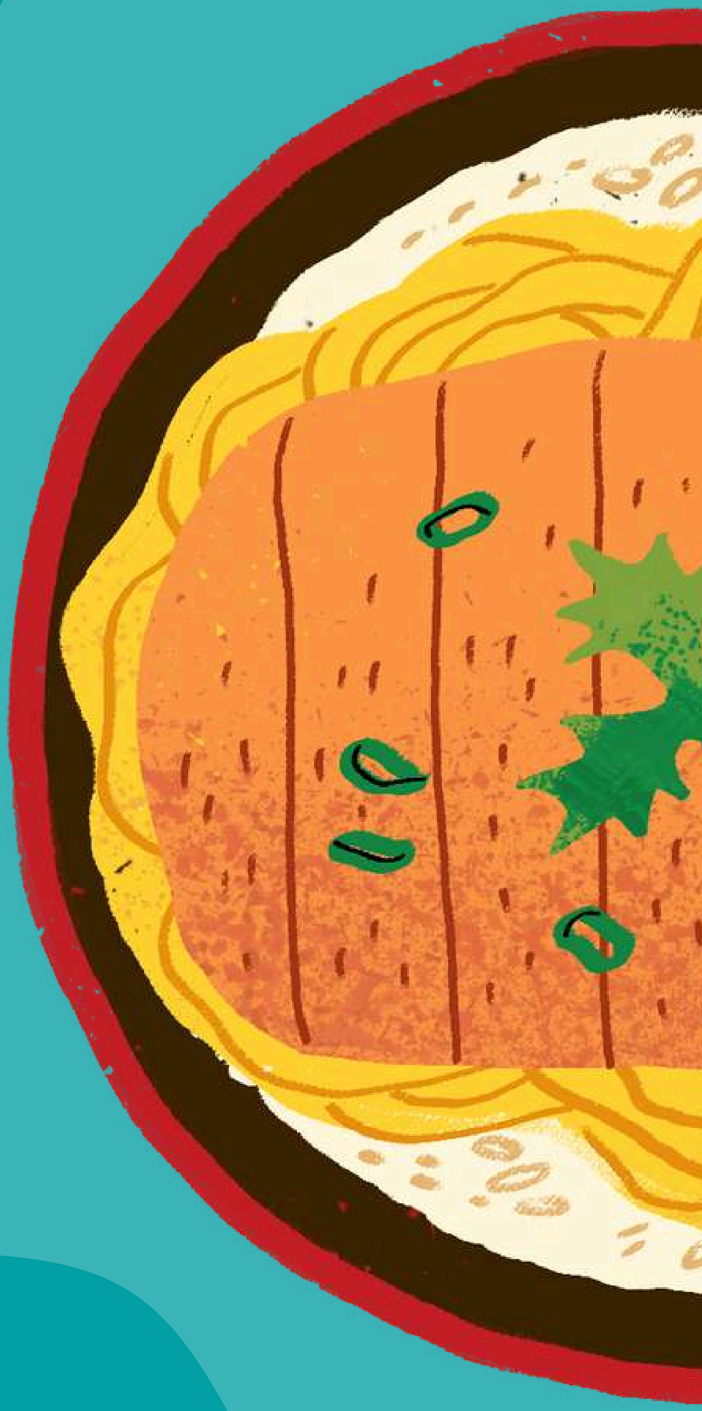
CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES



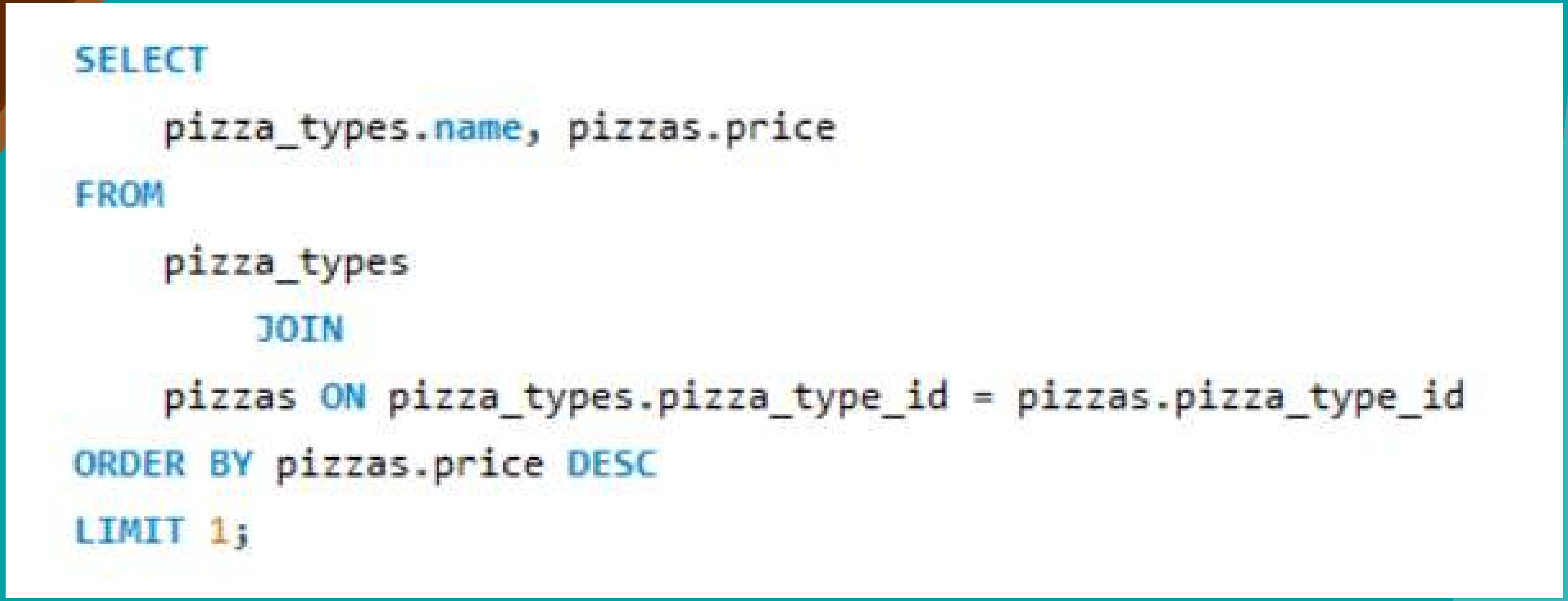
```
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS total_sales
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id;
```



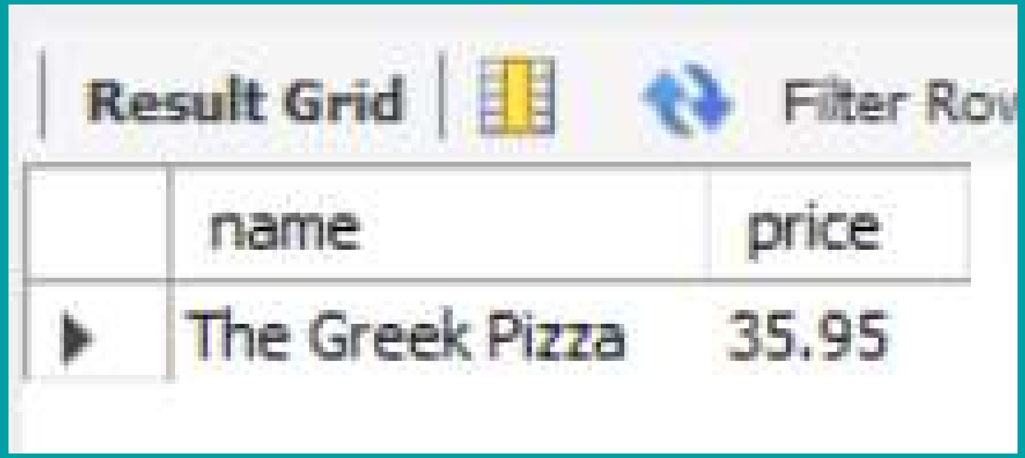
```
total_sales
817860.05
```

IDENTIFY THE HIGHEST-PRICED PIZZA

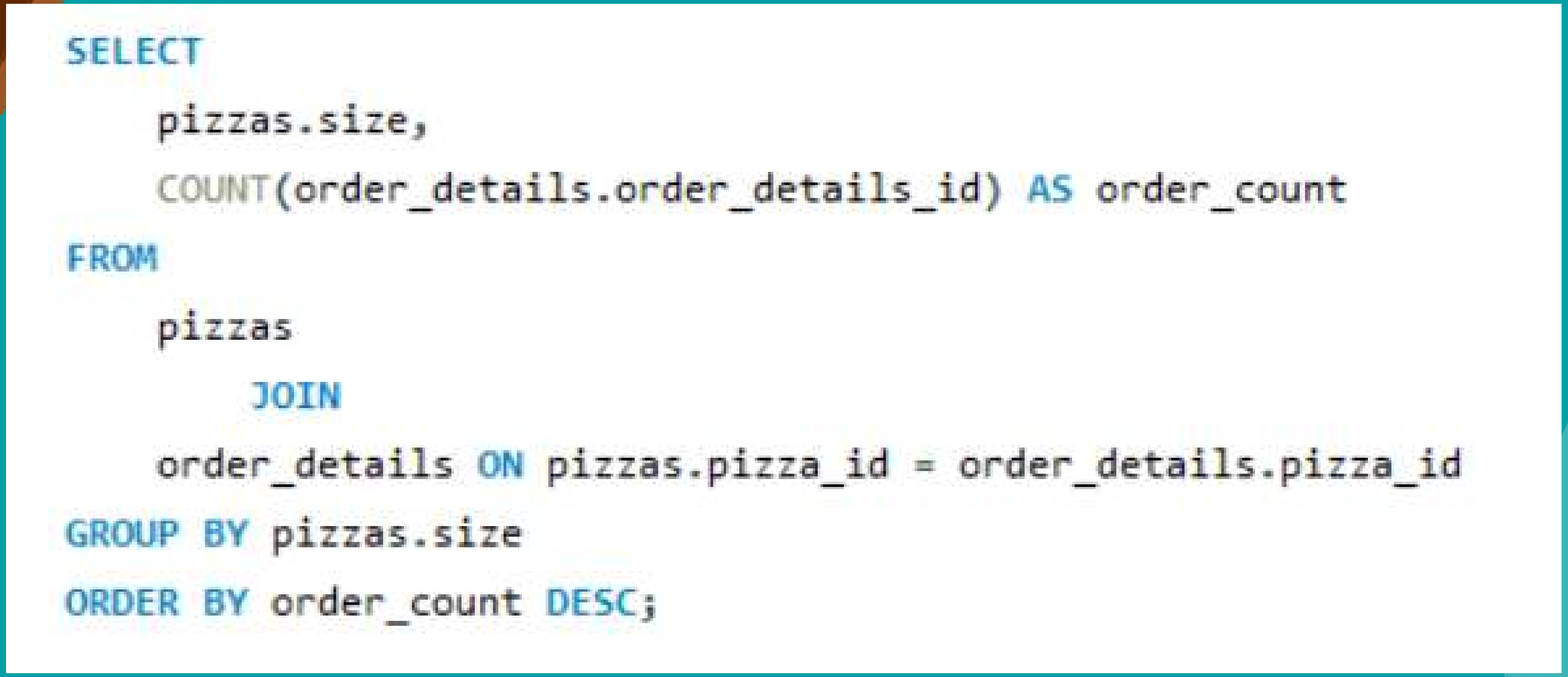


```
SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```

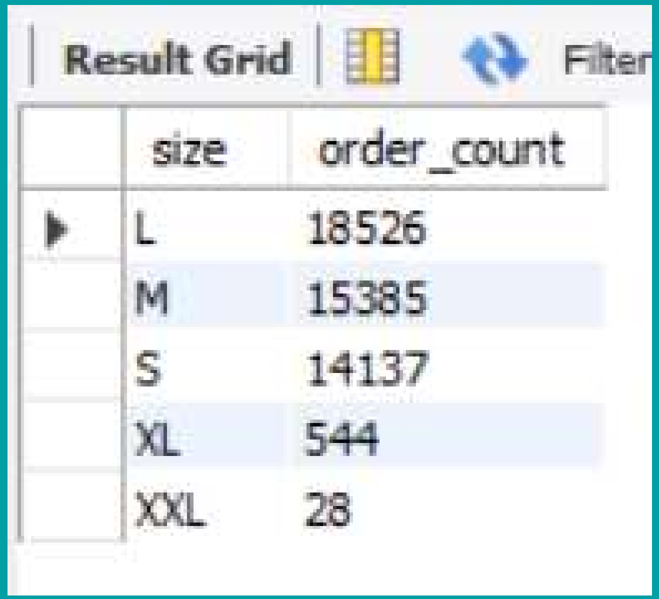


name	price
The Greek Pizza	35.95

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED

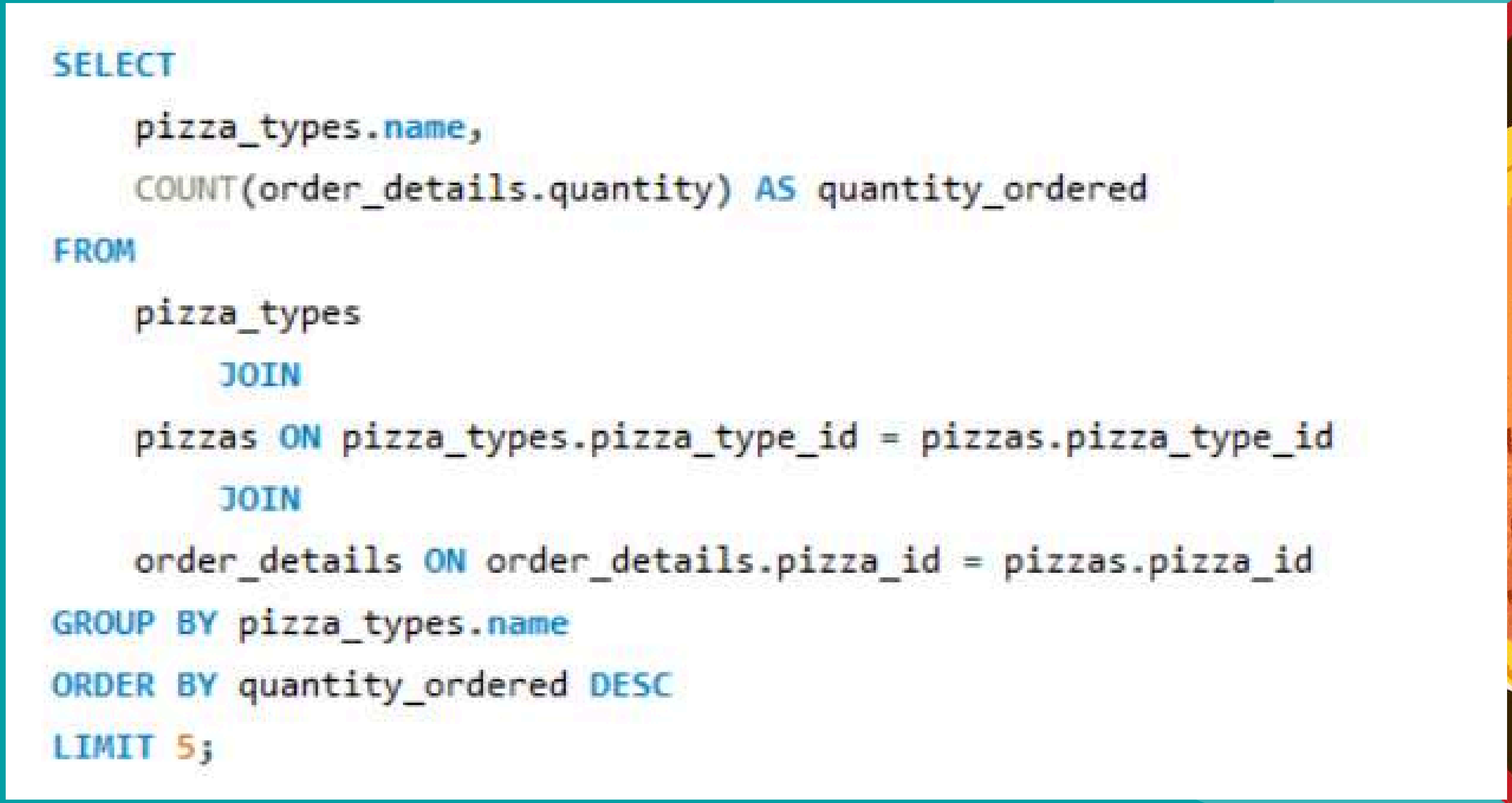


```
SELECT
    pizzas.size,
    COUNT(order_details.order_details_id) AS order_count
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC;
```

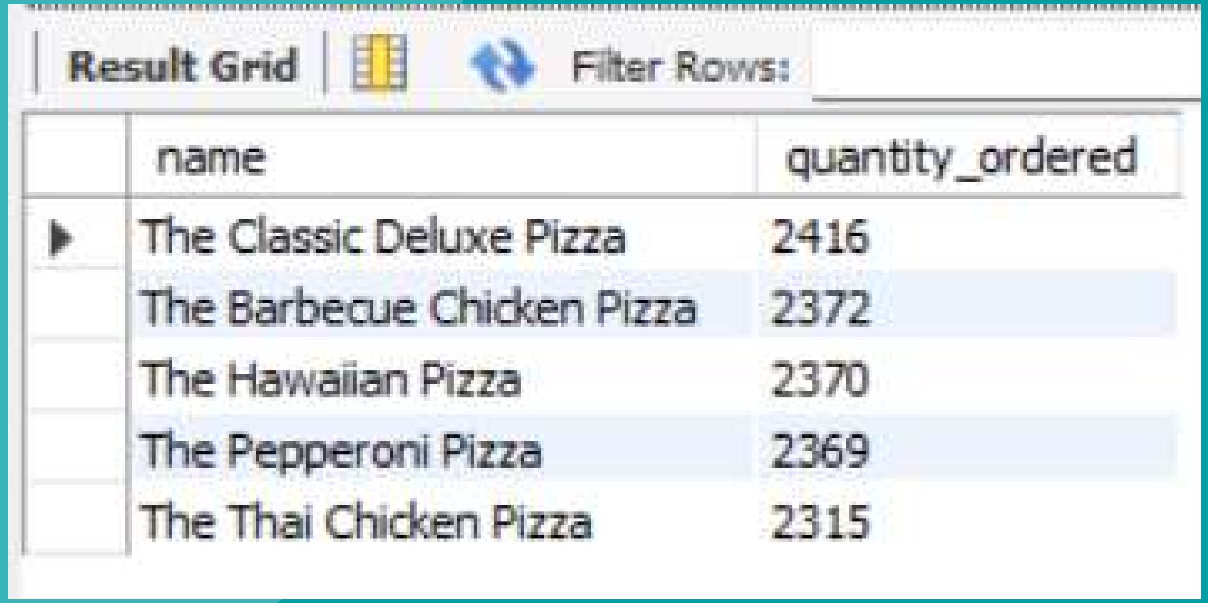


```
size   order_count
L      18526
M      15385
S      14137
XL     544
XXL    28
```

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

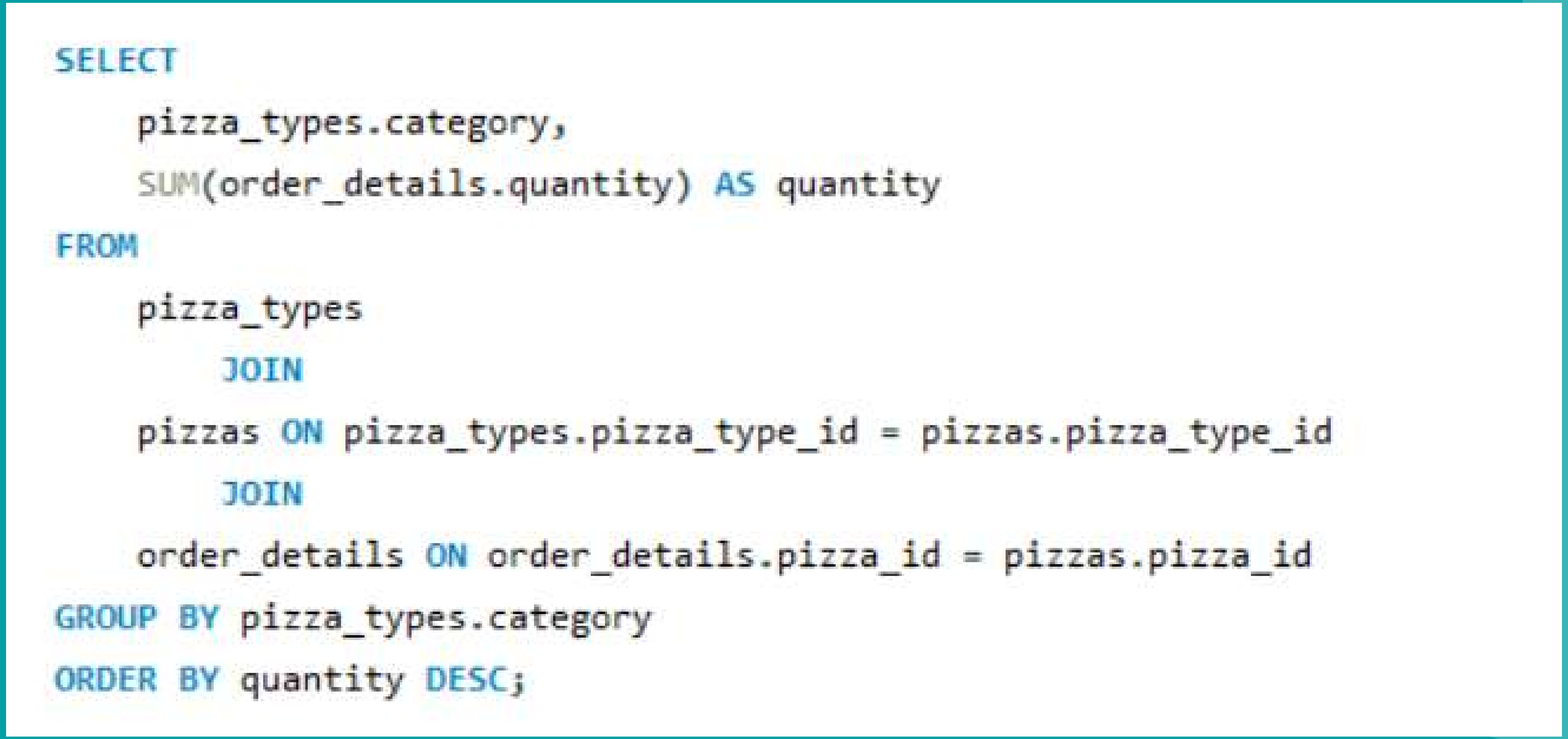


```sql
SELECT
    pizza_types.name,
    COUNT(order_details.quantity) AS quantity_ordered
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity_ordered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

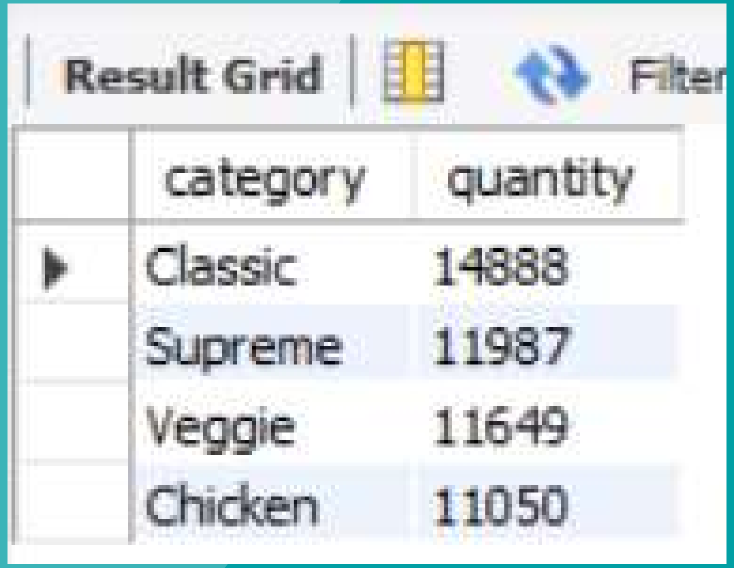


name	quantity_ordered
The Classic Deluxe Pizza	2416
The Barbecue Chicken Pizza	2372
The Hawaiian Pizza	2370
The Pepperoni Pizza	2369
The Thai Chicken Pizza	2315

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

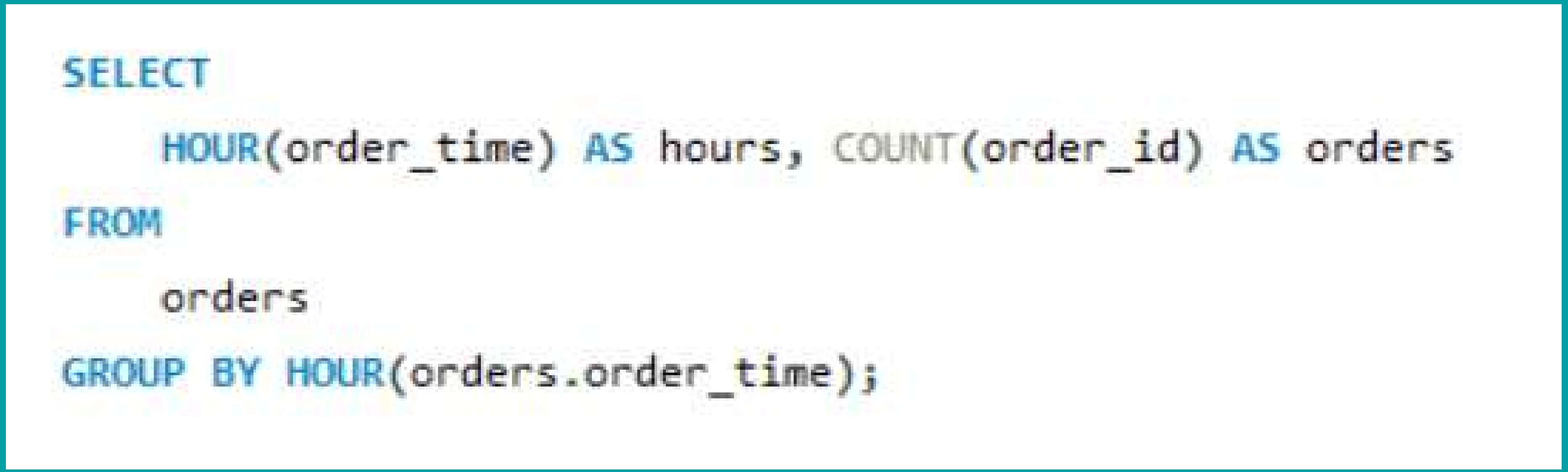


```
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY quantity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

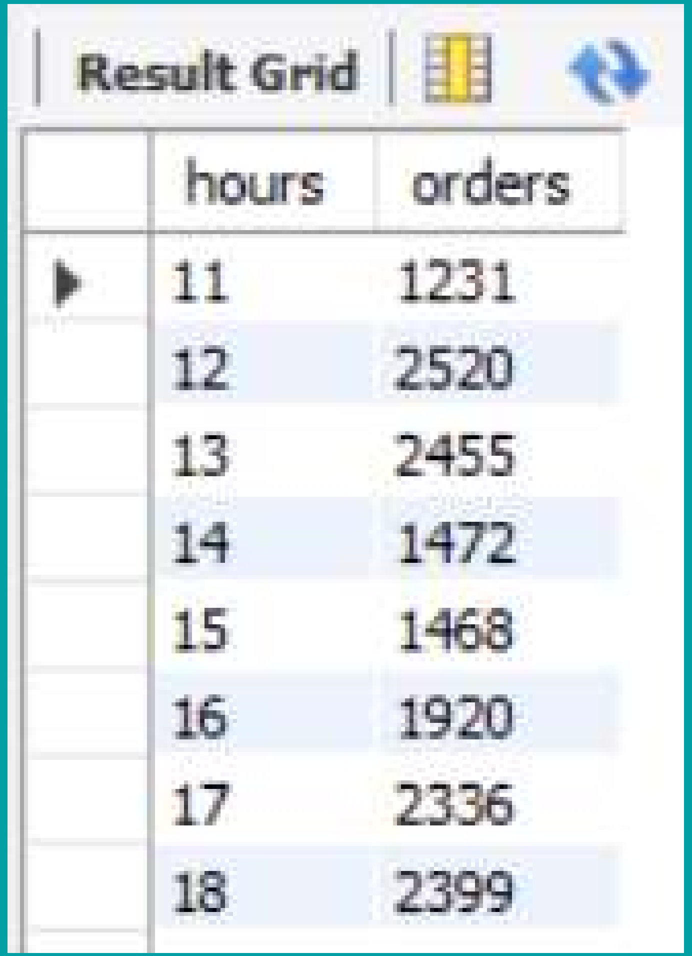


category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY

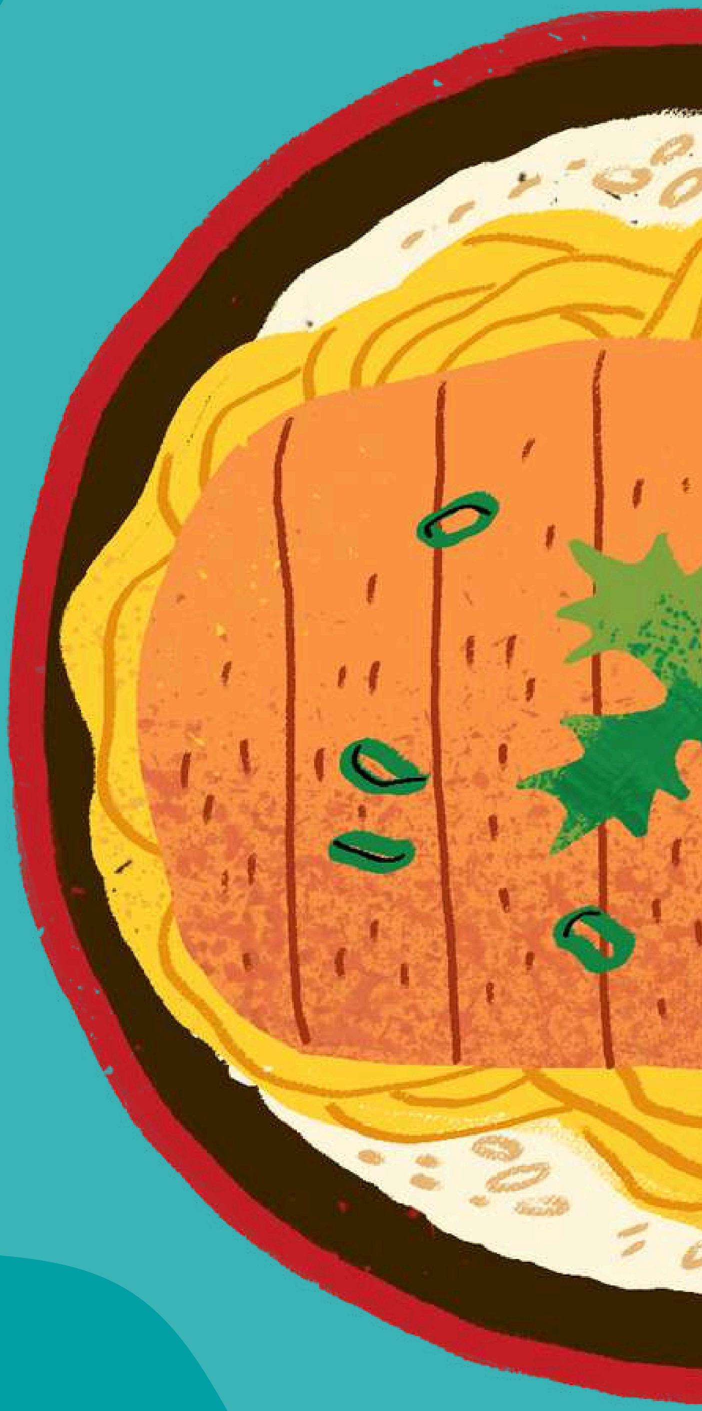


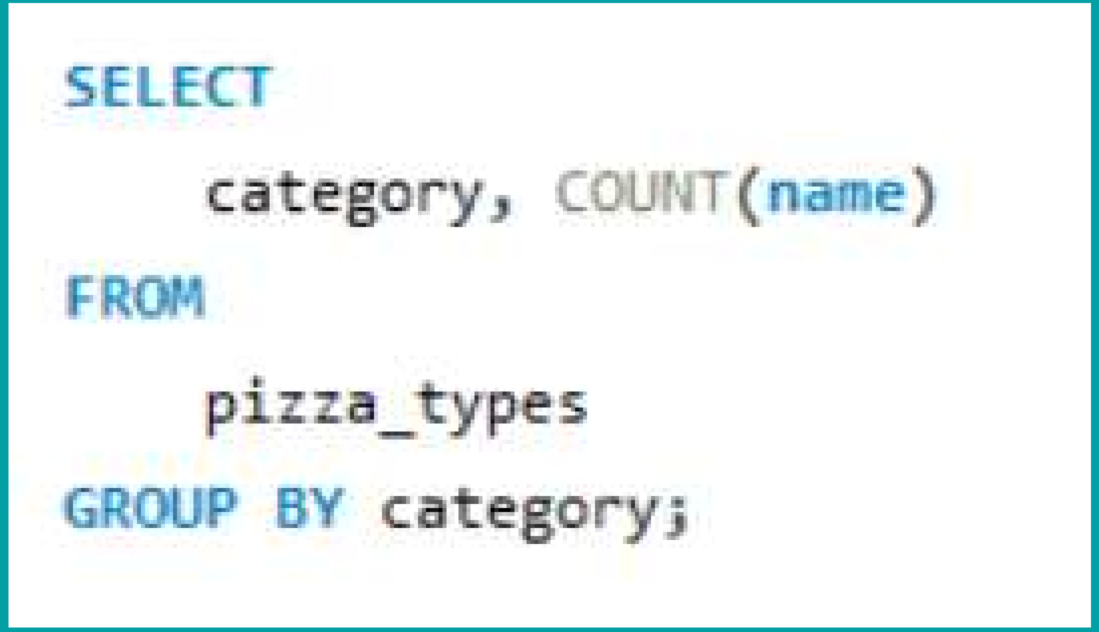
```
SELECT
    HOUR(order_time) AS hours, COUNT(order_id) AS orders
FROM
    orders
GROUP BY HOUR(orders.order_time);
```



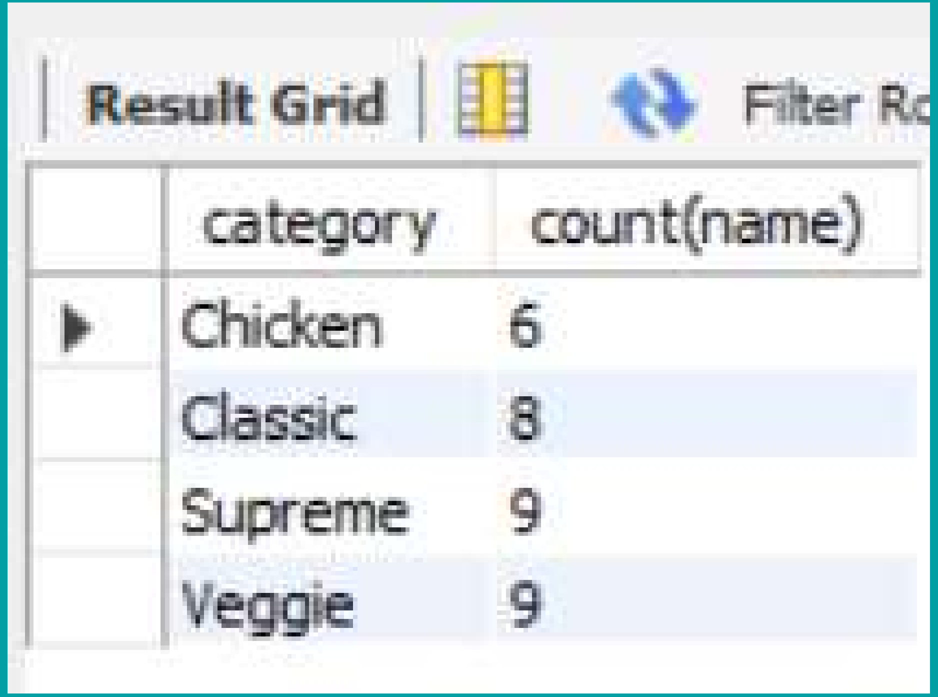
hours	orders
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399

JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS



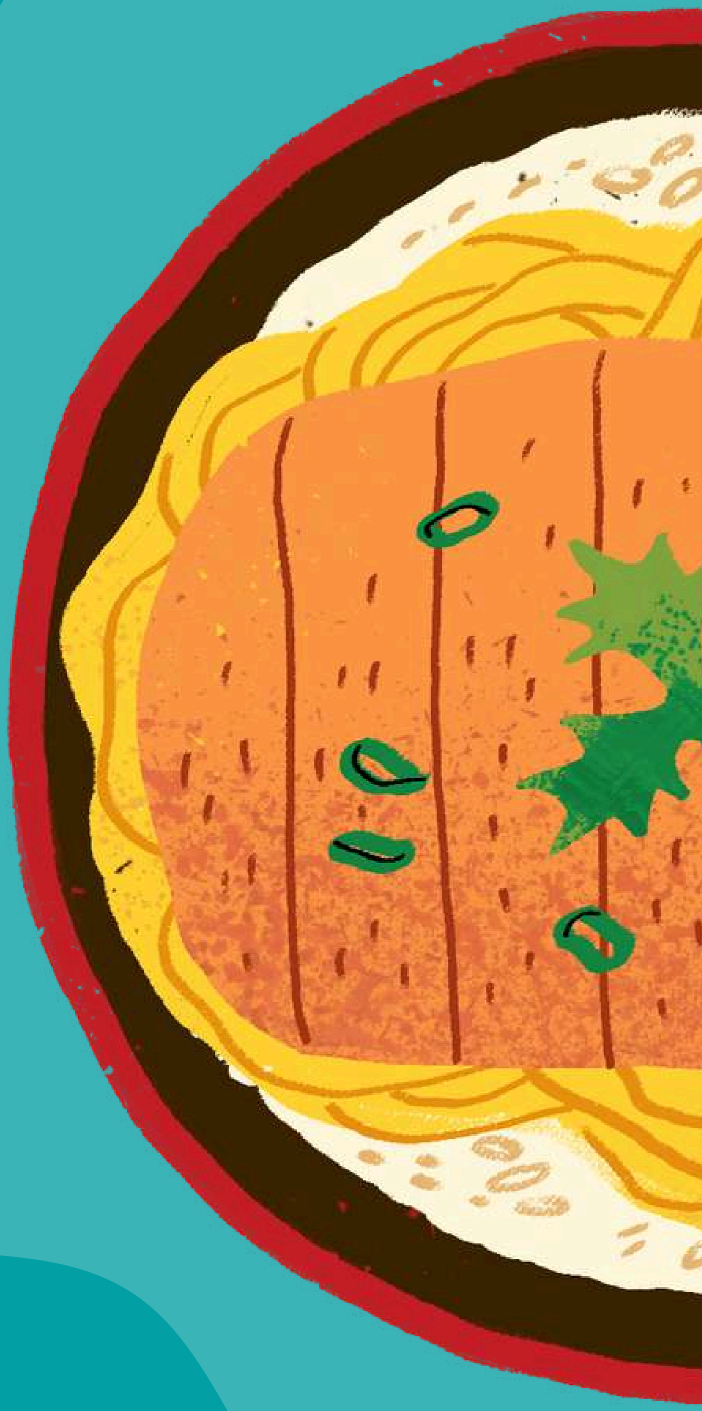


```
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```

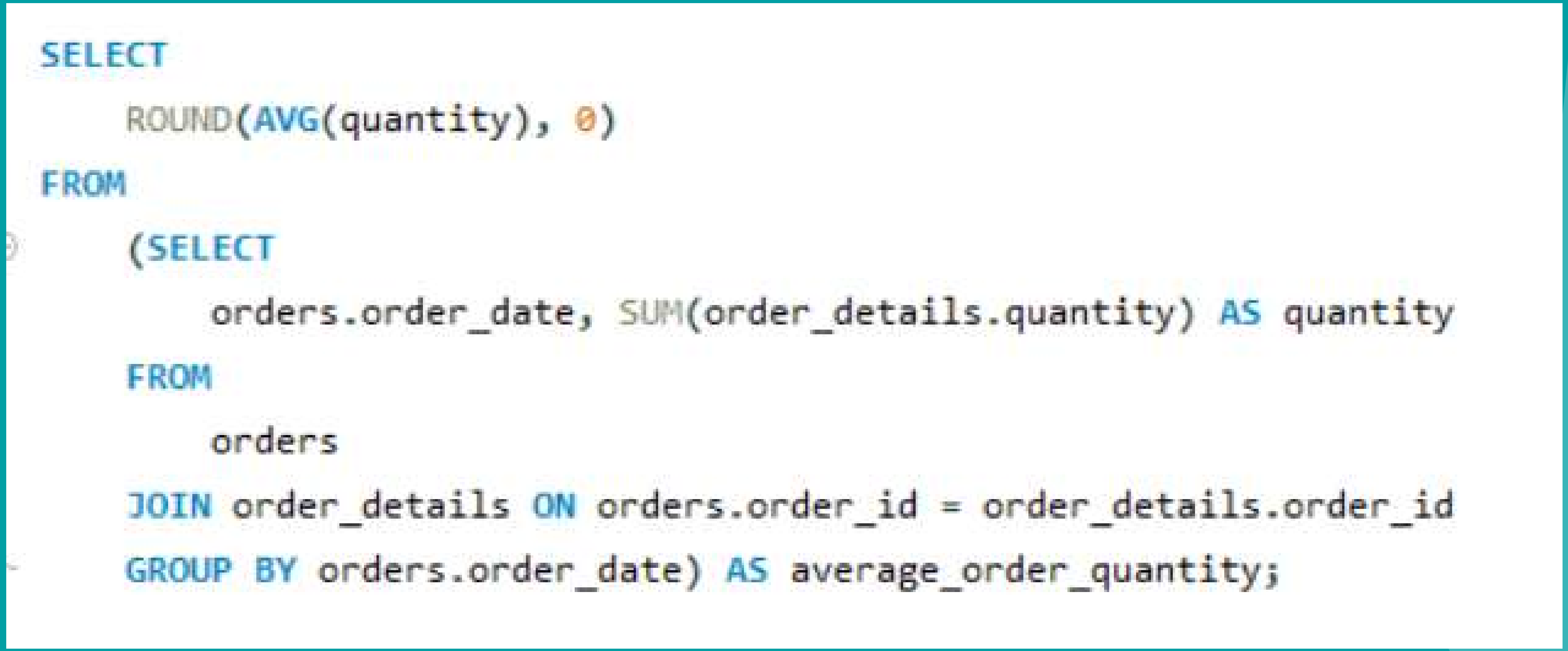


Result Grid | Filter Ro

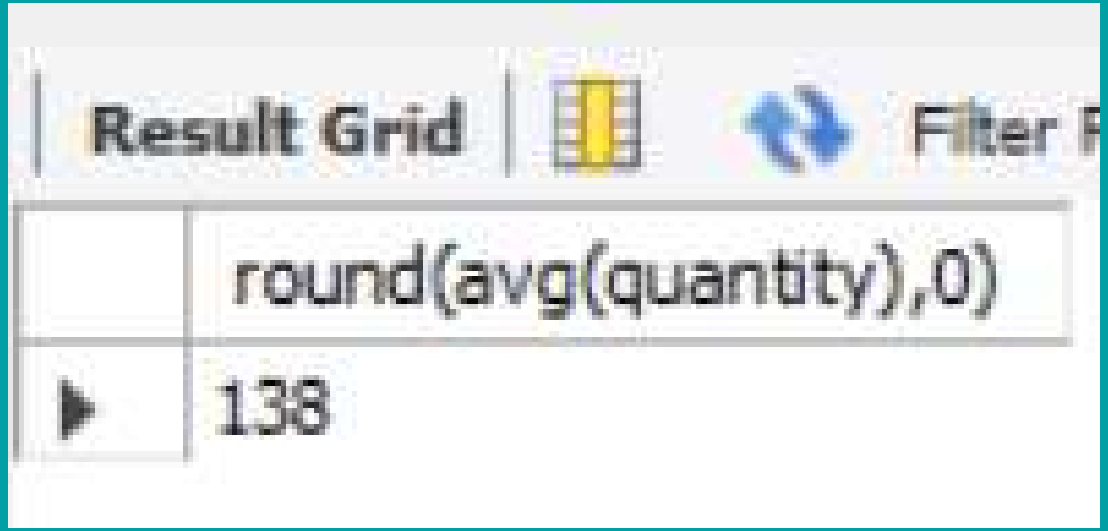
	category	count(name)
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9



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

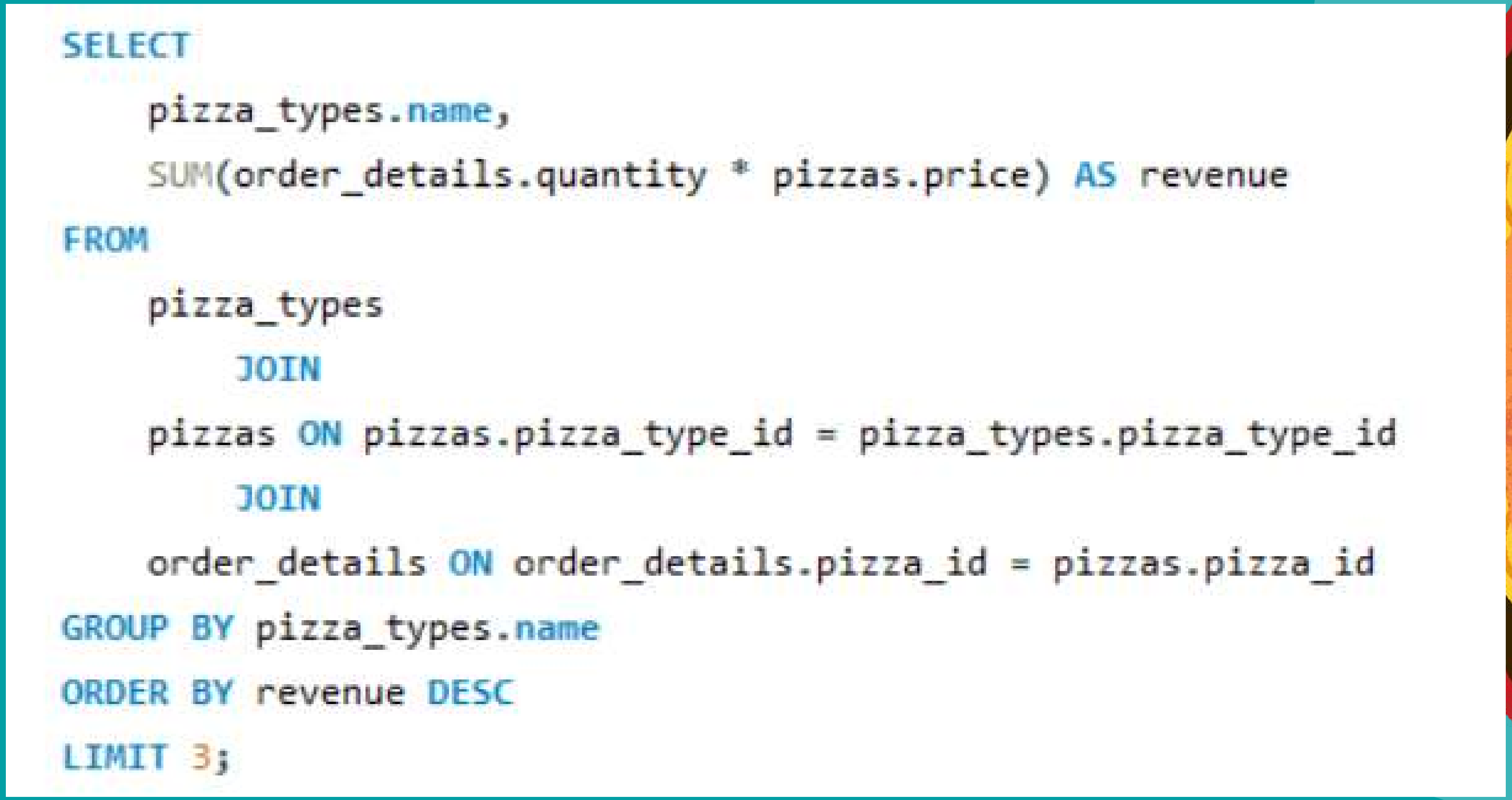


```
SELECT
    ROUND(AVG(quantity), 0)
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS average_order_quantity;
```

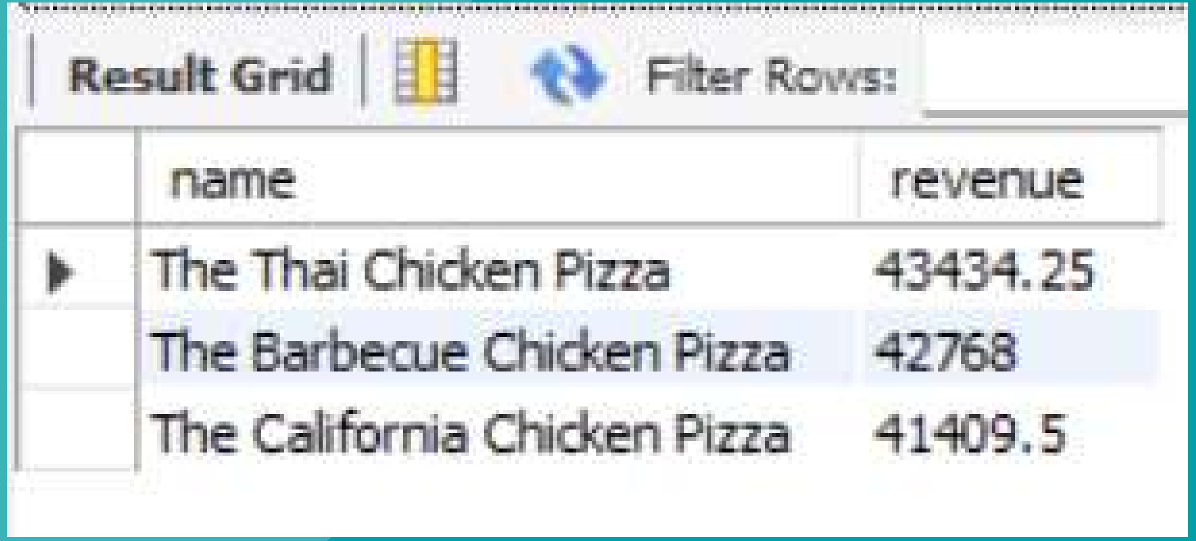


```
round(avg(quantity),0)
138
```

DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE

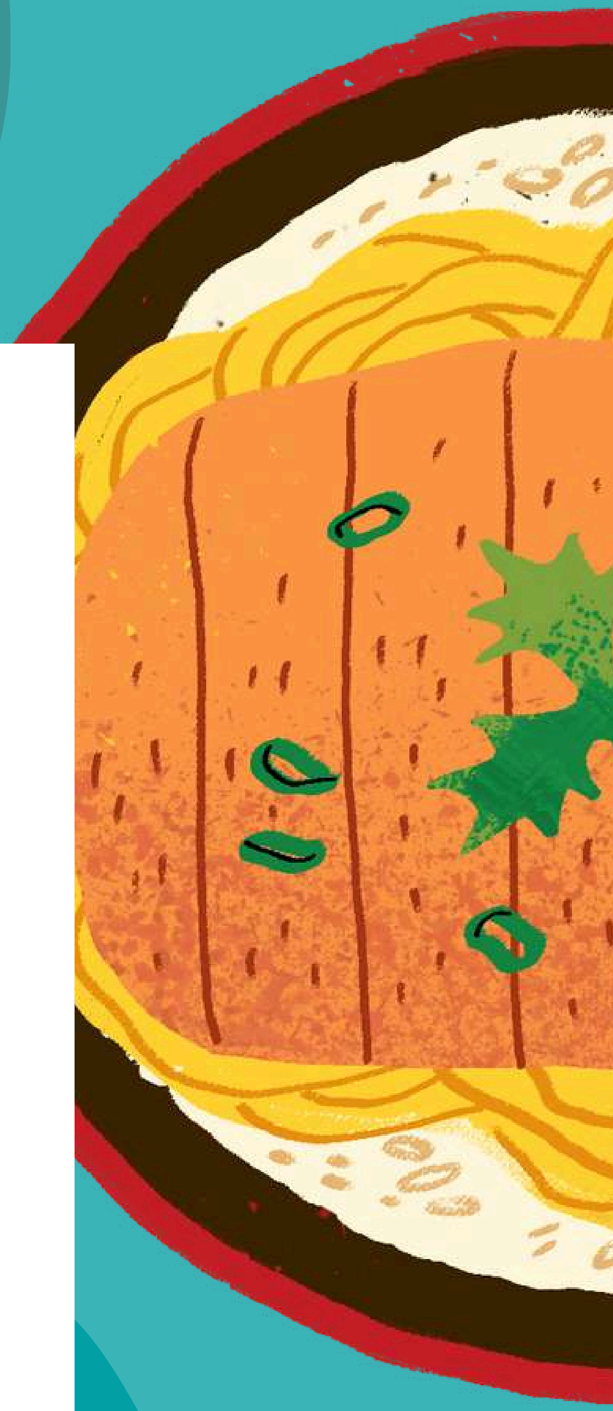


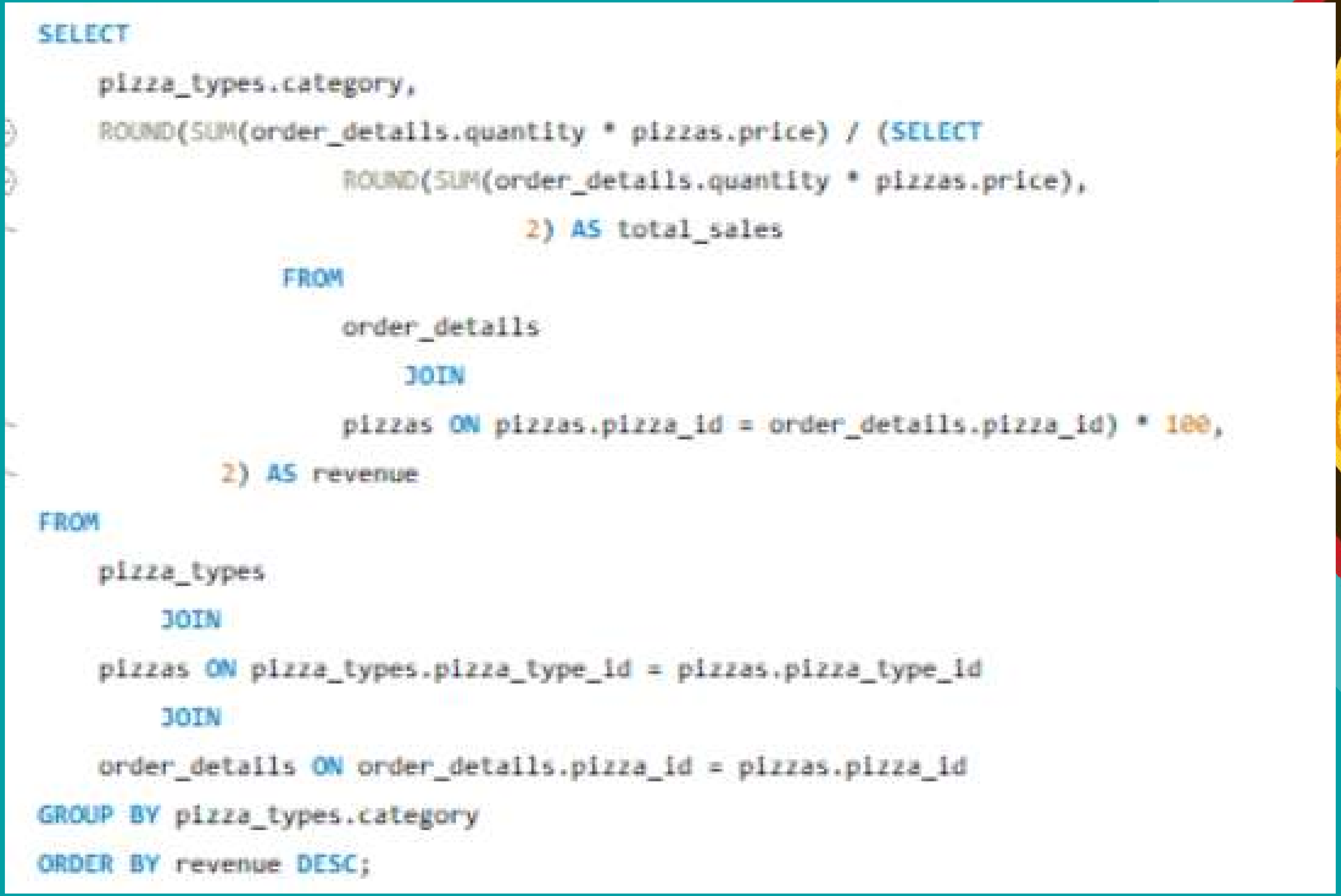
```
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue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



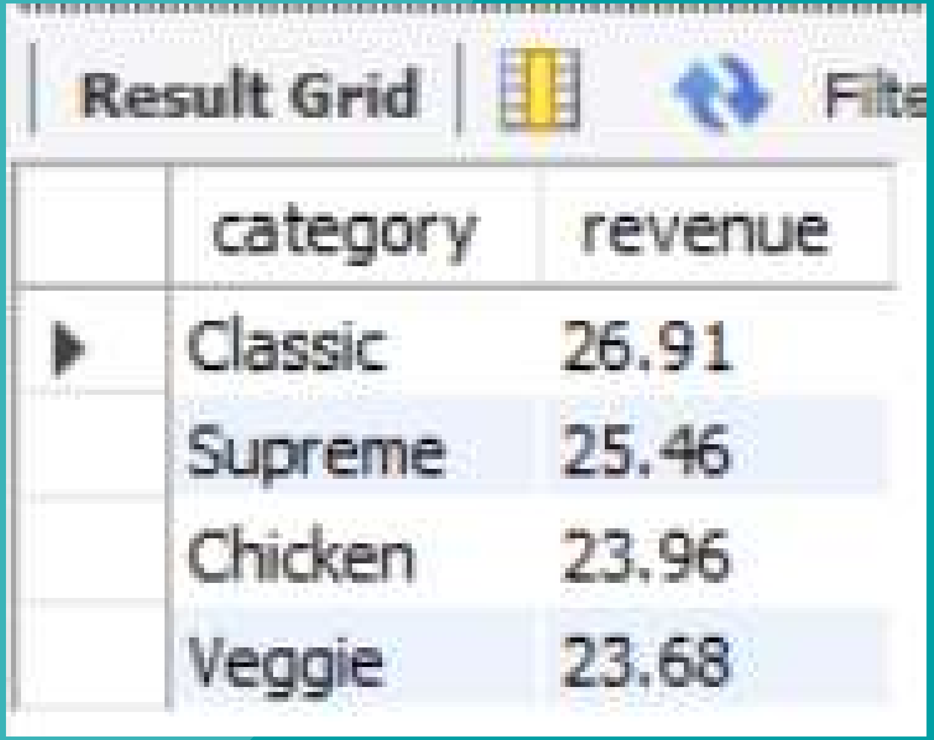
name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE





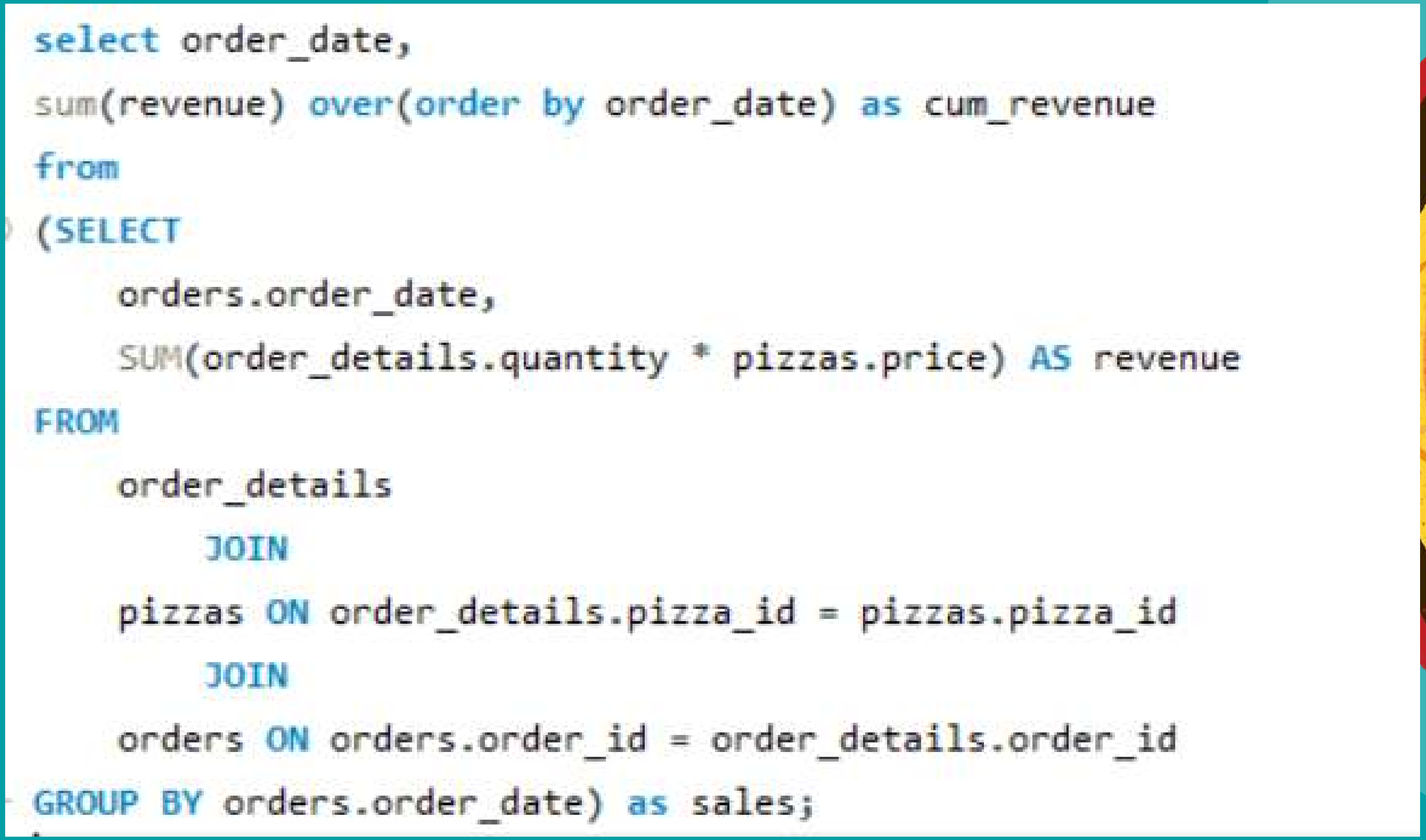
```
SELECT
    pizza_types.category,
    ROUND(SUM(order_details.quantity * pizzas.price) / (SELECT
                    ROUND(SUM(order_details.quantity * pizzas.price),
                                2) AS total_sales
                FROM
                    order_details
                        JOIN
                    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;
```



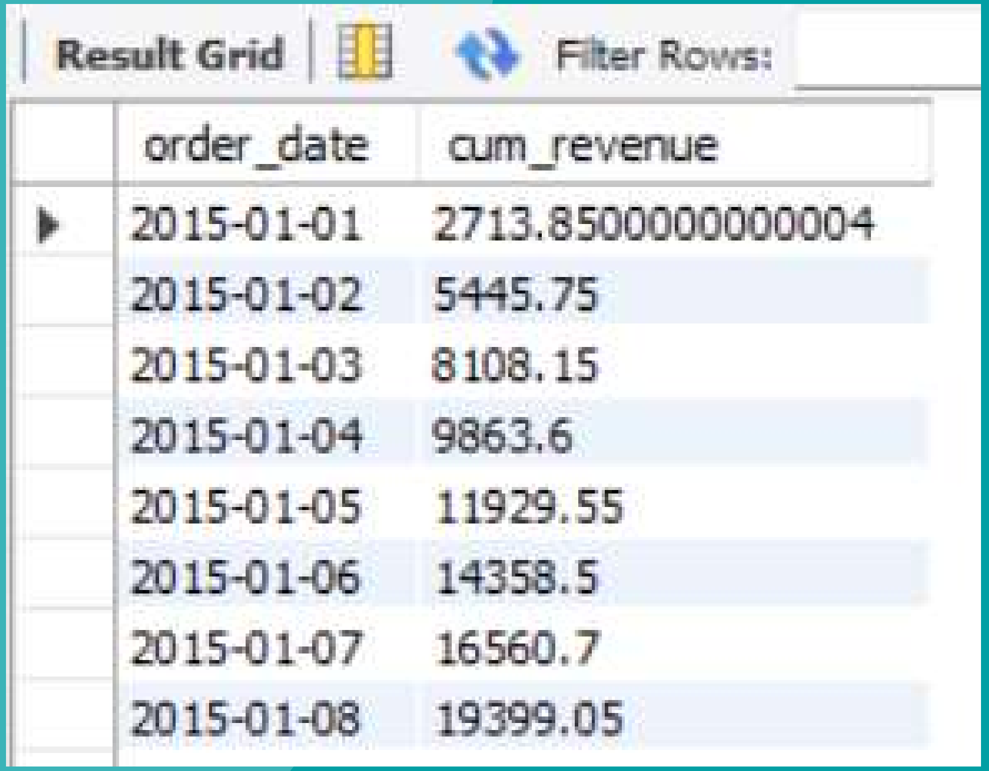
Result Grid | Filte

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

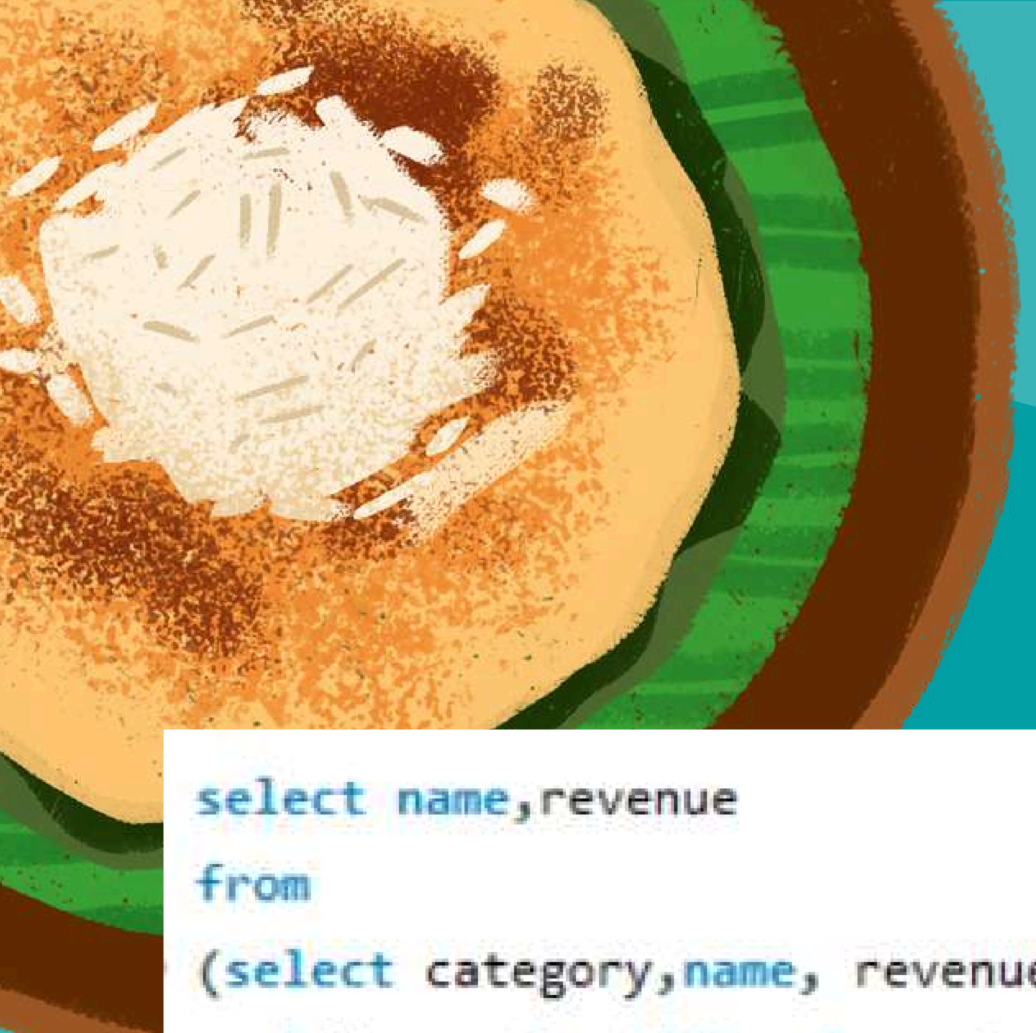
ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME



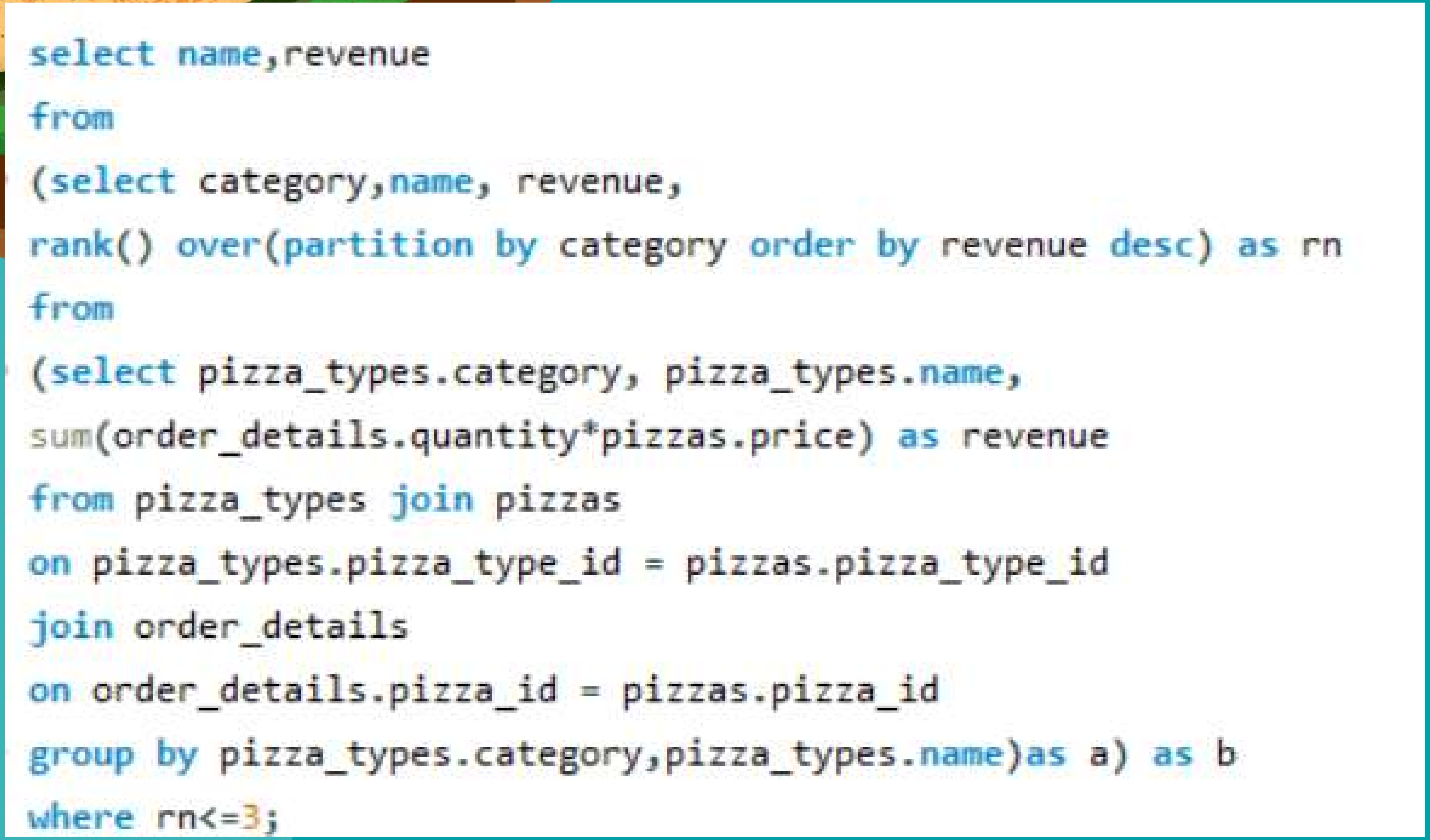
```
select order_date,
sum(revenue) over(order by order_date) as cum_revenue
from
(SELECT
    orders.order_date,
    SUM(order_details.quantity * pizzas.price) AS revenue
FROM
    order_details
        JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id
        JOIN
    orders ON orders.order_id = order_details.order_id
GROUP BY orders.order_date) as sales;
```



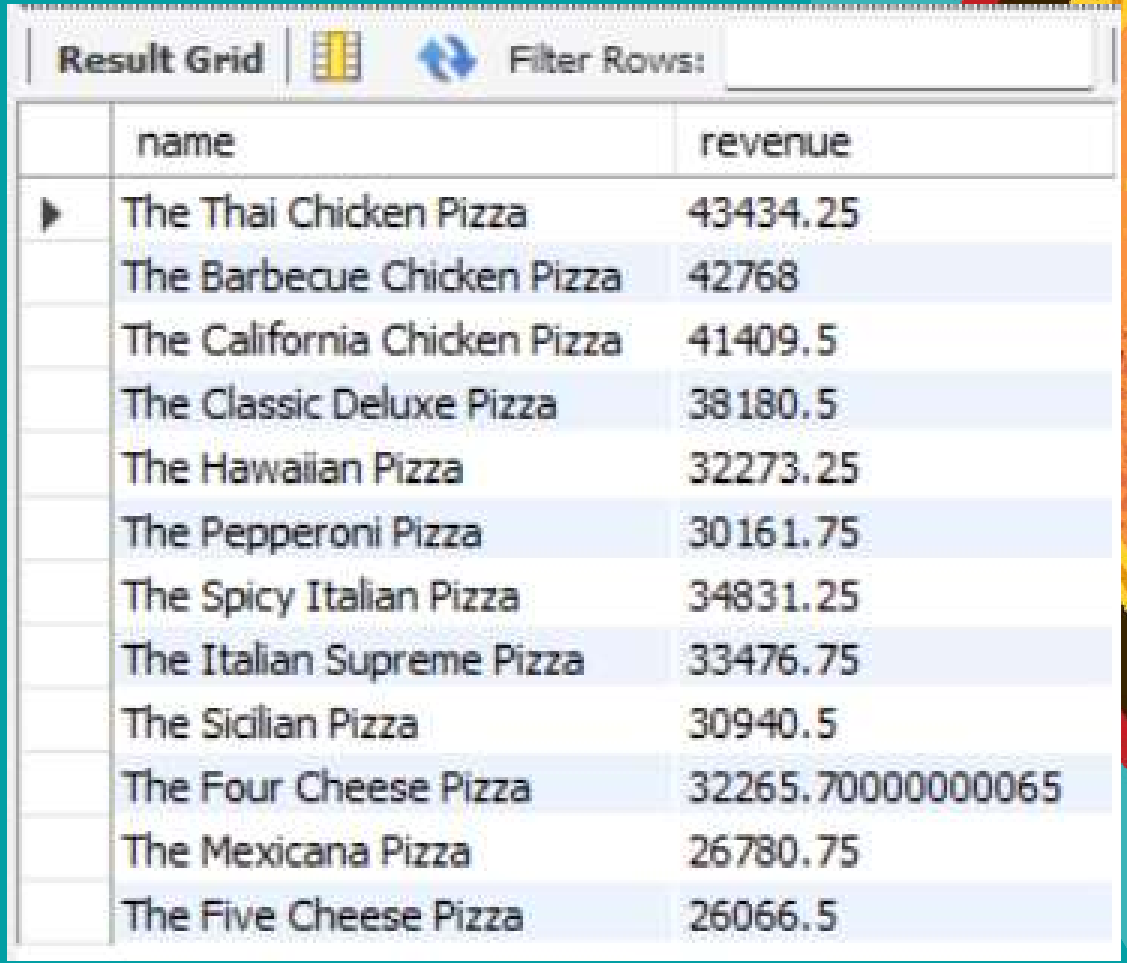
order_date	cum_revenue
2015-01-01	2713.8500000000004
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55
2015-01-06	14358.5
2015-01-07	16560.7
2015-01-08	19399.05



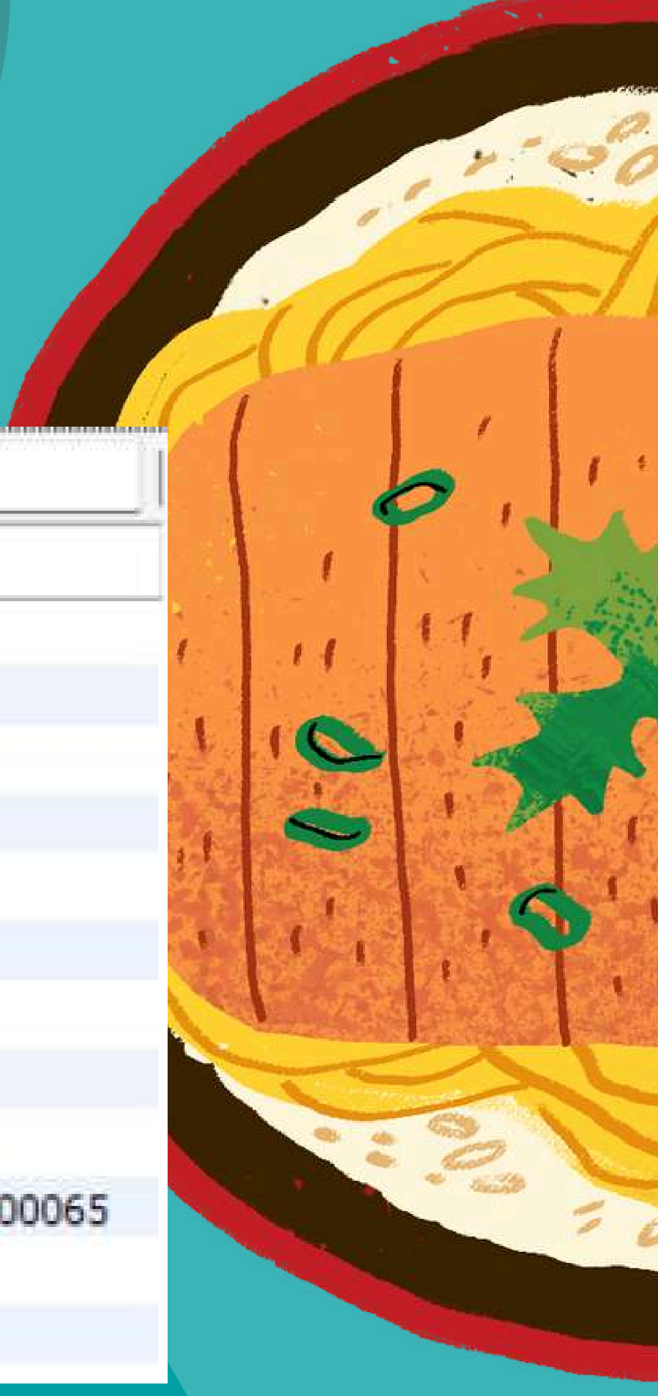
DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY



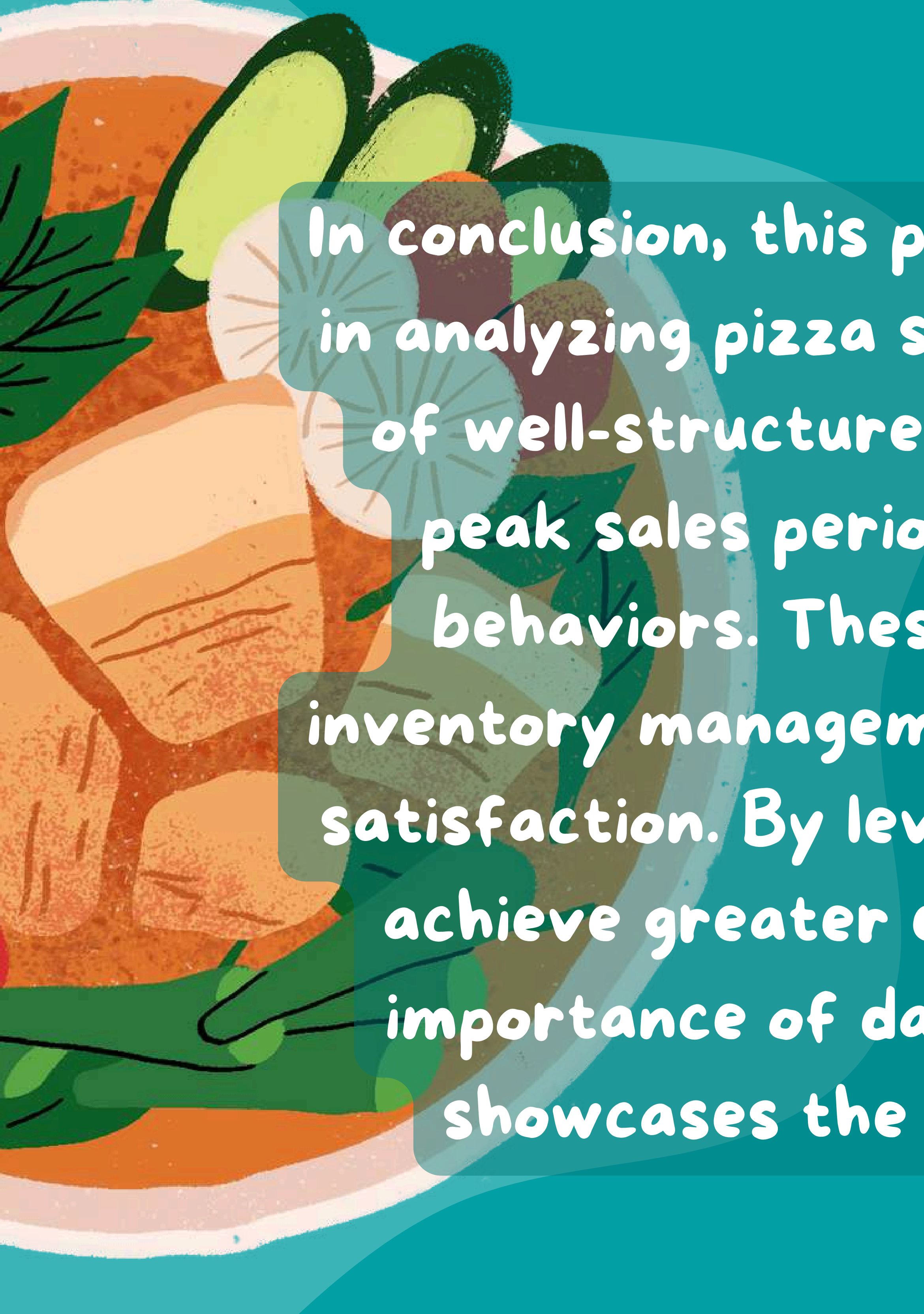
```
select name,revenue
from
(select category,name, revenue,
rank() over(partition by category order by revenue desc) as rn
from
(select pizza_types.category, pizza_types.name,
sum(order_details.quantity*pizzas.price) as revenue
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id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



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



CONCLUSION



In conclusion, this project has successfully demonstrated the power of SQL in analyzing pizza sales data to uncover valuable insights. Through a series of well-structured queries, we were able to identify key trends such as peak sales periods, popular pizza varieties, and customer purchasing behaviors. These insights can inform strategic decisions to optimize inventory management, tailor marketing campaigns, and enhance customer satisfaction. By leveraging data-driven approaches, the pizza business can achieve greater efficiency and profitability. This project highlights the importance of data analysis in making informed business decisions and showcases the versatility of SQL as a tool for comprehensive data examination.

Thanks for watching!

END