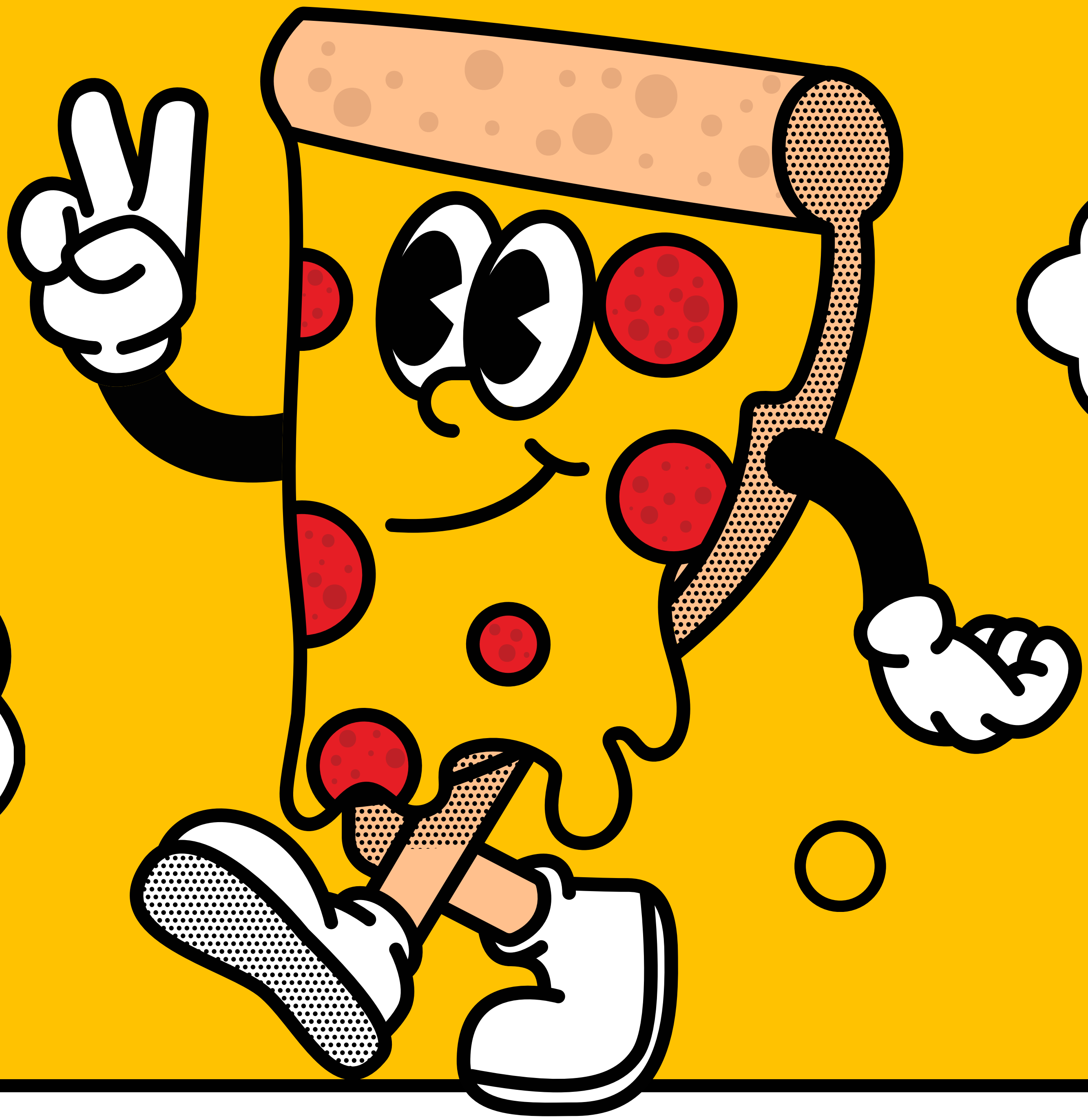
PIZZA SQL

SQL PROJECT

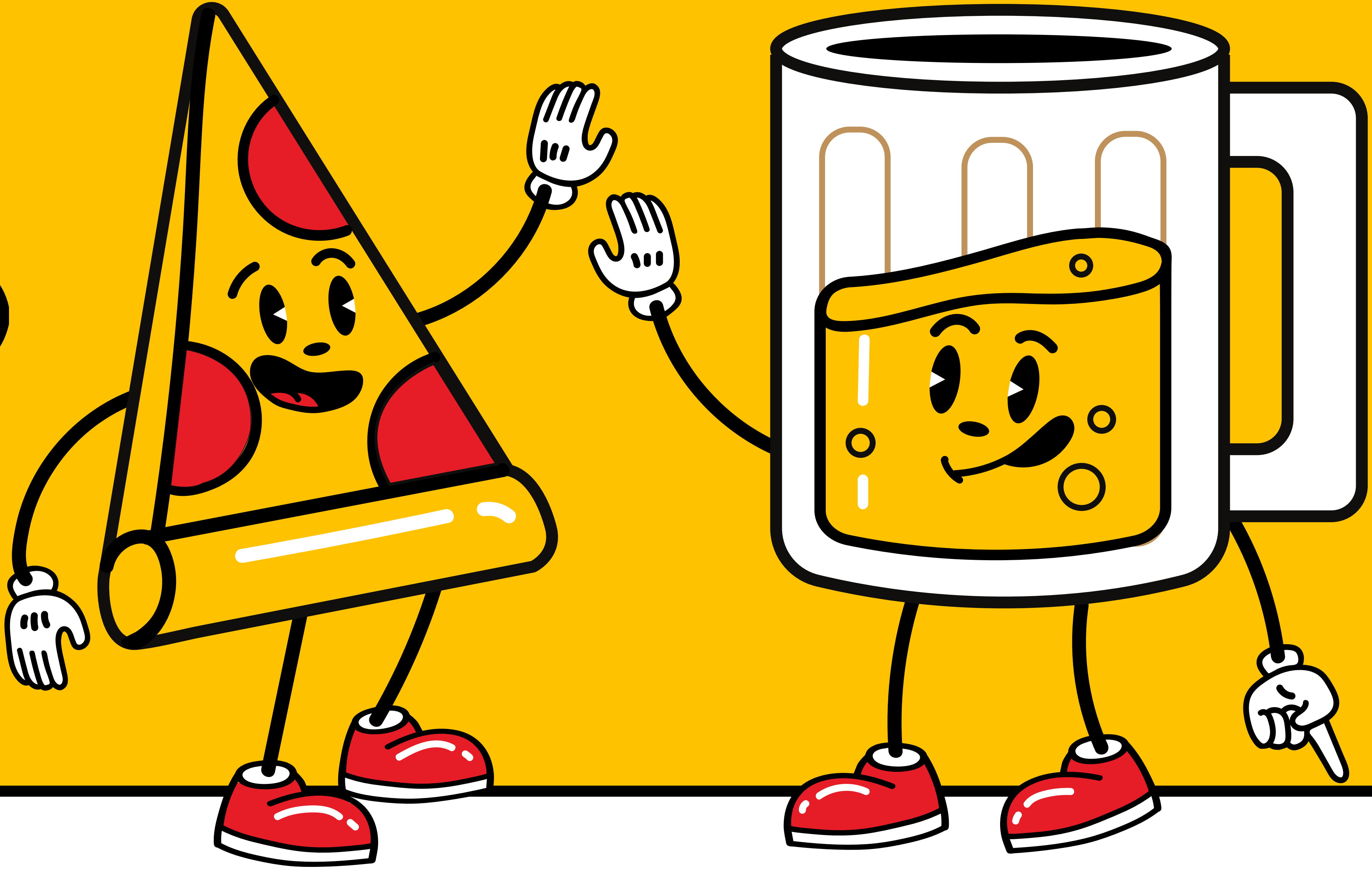


HELLO

HELLO EVERYONE,

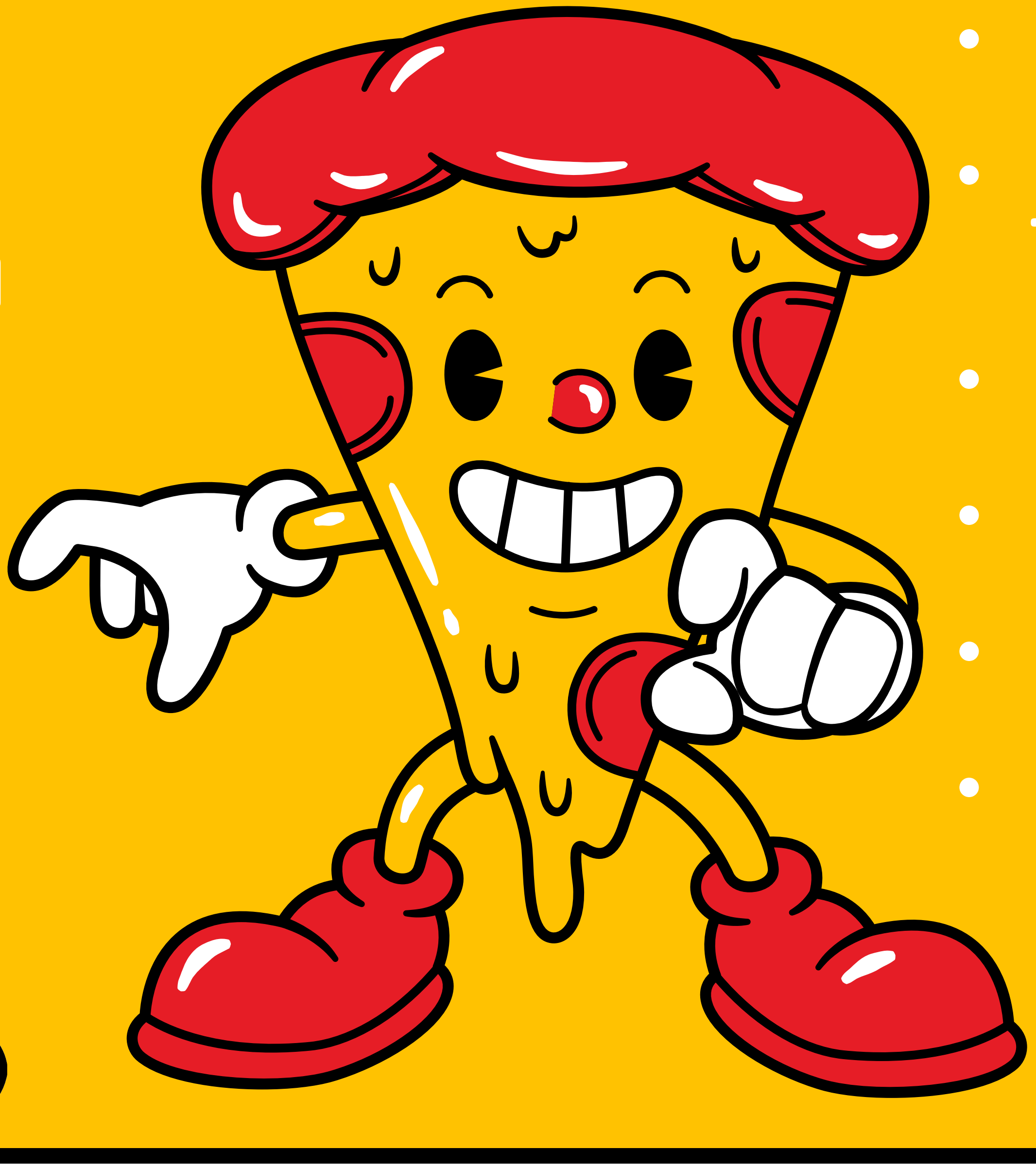
My name is Sana.
Today I will be presenting my SQL-based Pizza Sales Analysis project.
In this project, I utilized SQL queries to answer various analytical questions related to pizza sales data

THIS PROJECT HELPED ME UNDERSTAND HOW SQL CAN BE USED IN REAL-WORLD SCENARIOS TO SUPPORT DATA-DRIVEN DECISION MAKING.



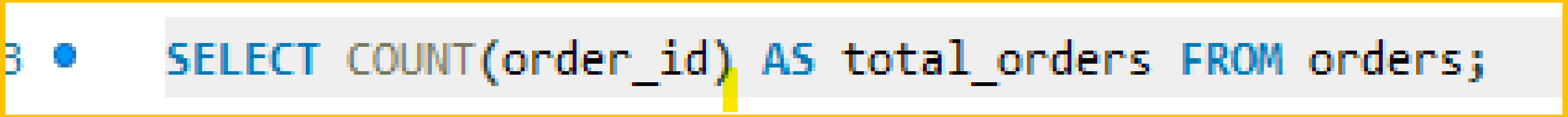
ANALYTICAL QUESTIONS

- 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.

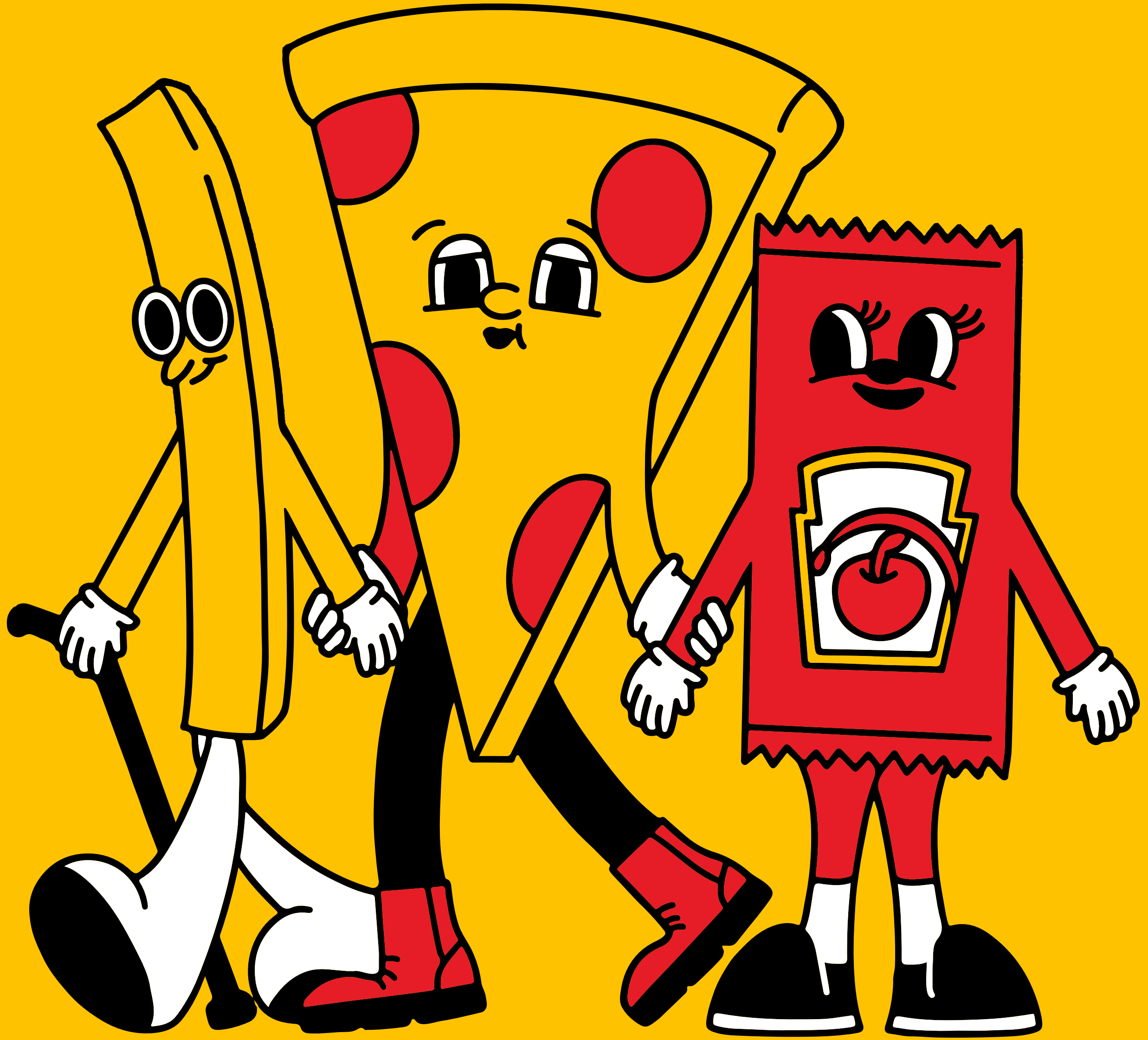


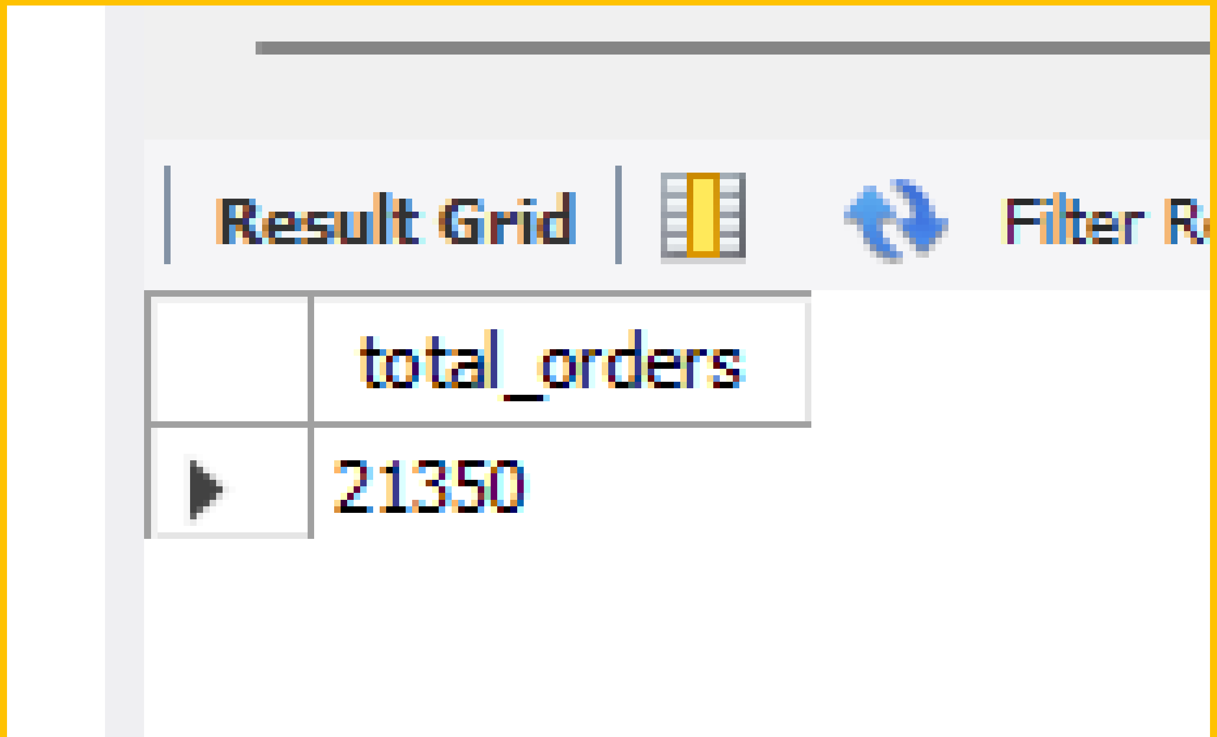
- JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.
- DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.
- JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS
- GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.
- DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.
- 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.



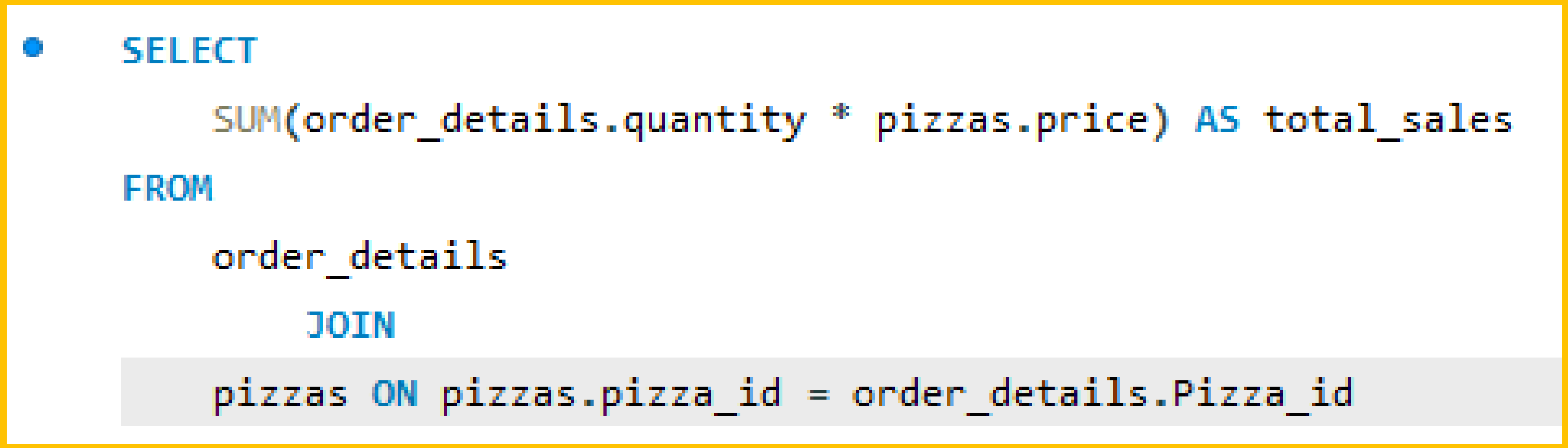
```sql
SELECT COUNT(order_id) AS total_orders FROM orders;
```



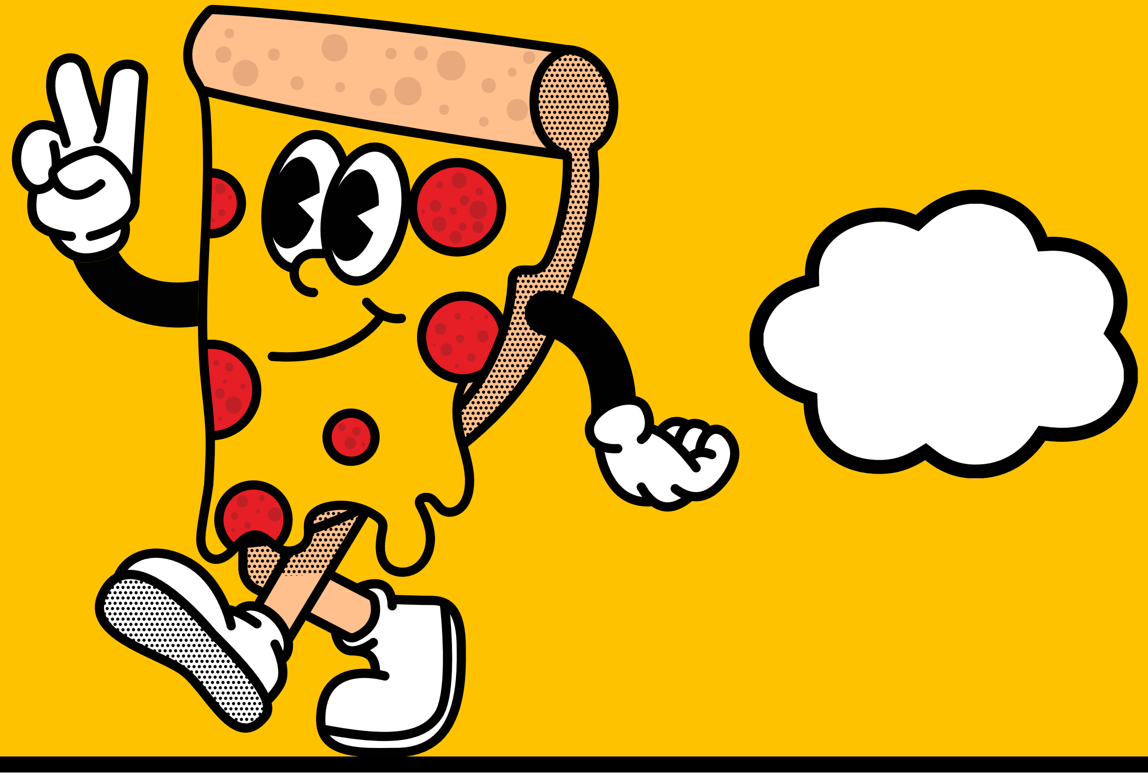


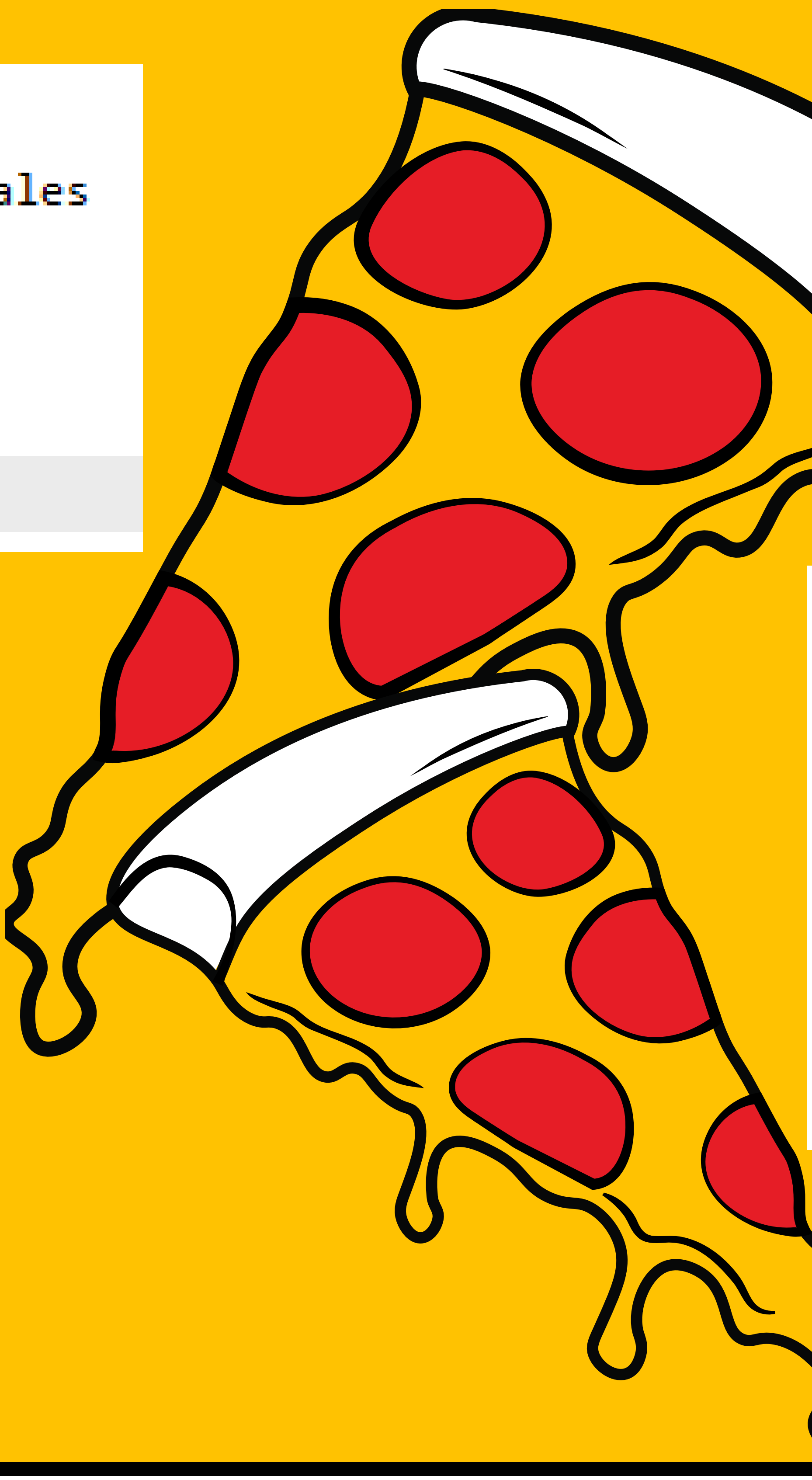
```
total_orders
21350
```

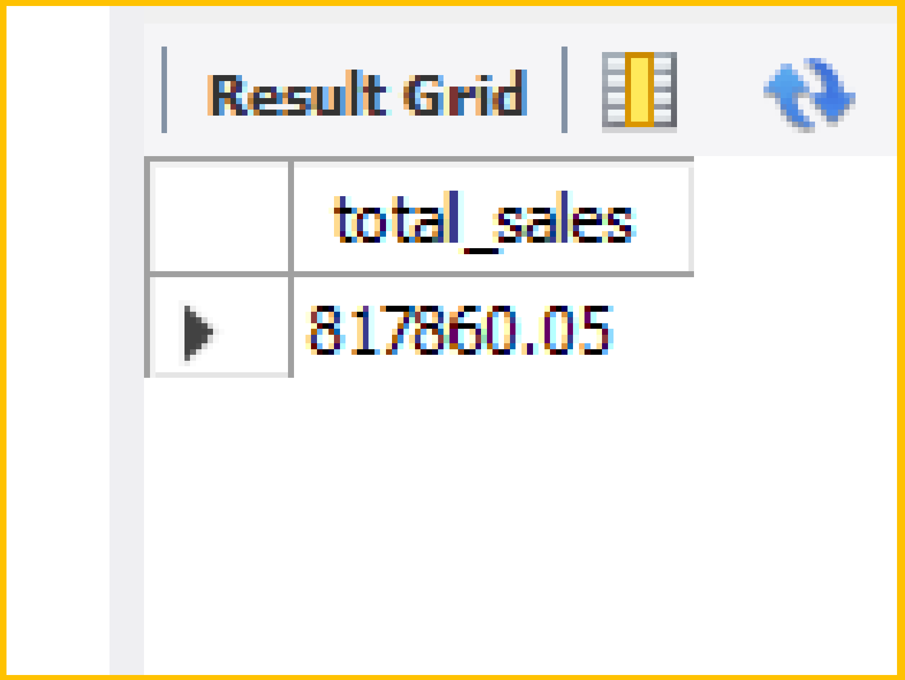
CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.



```sql
SELECT
    SUM(order_details.quantity * pizzas.price) AS total_sales
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.Pizza_id
```

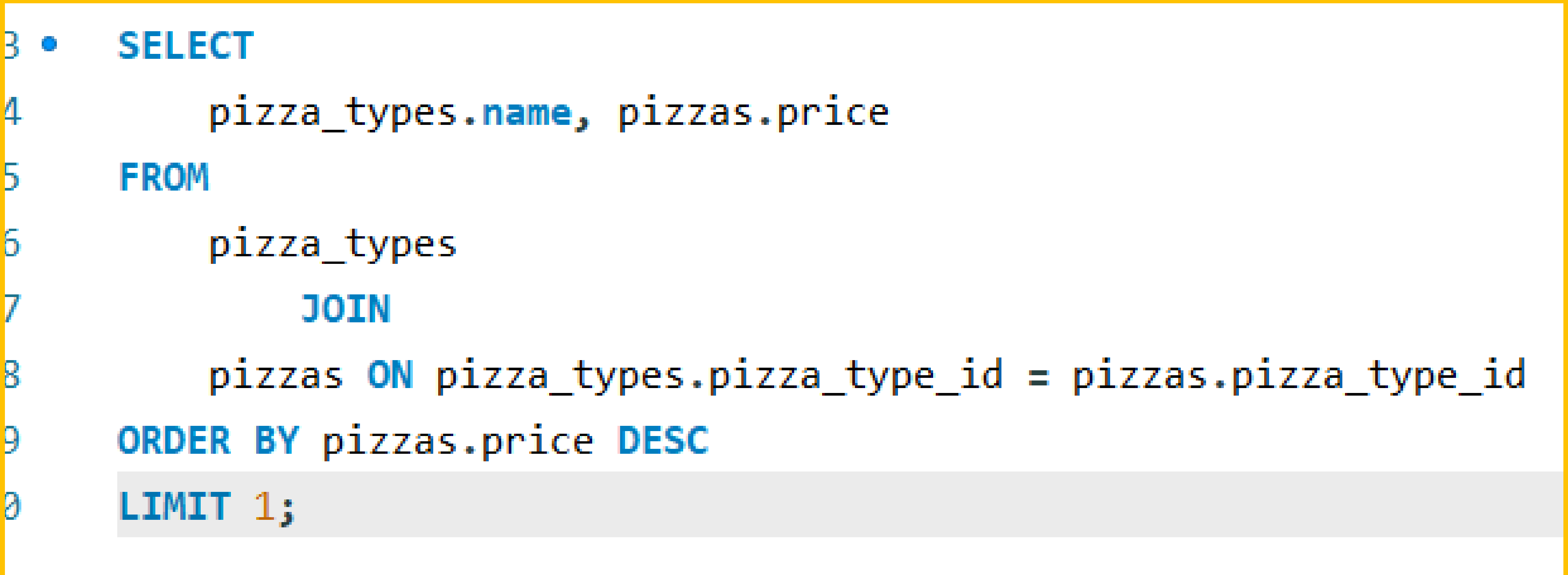




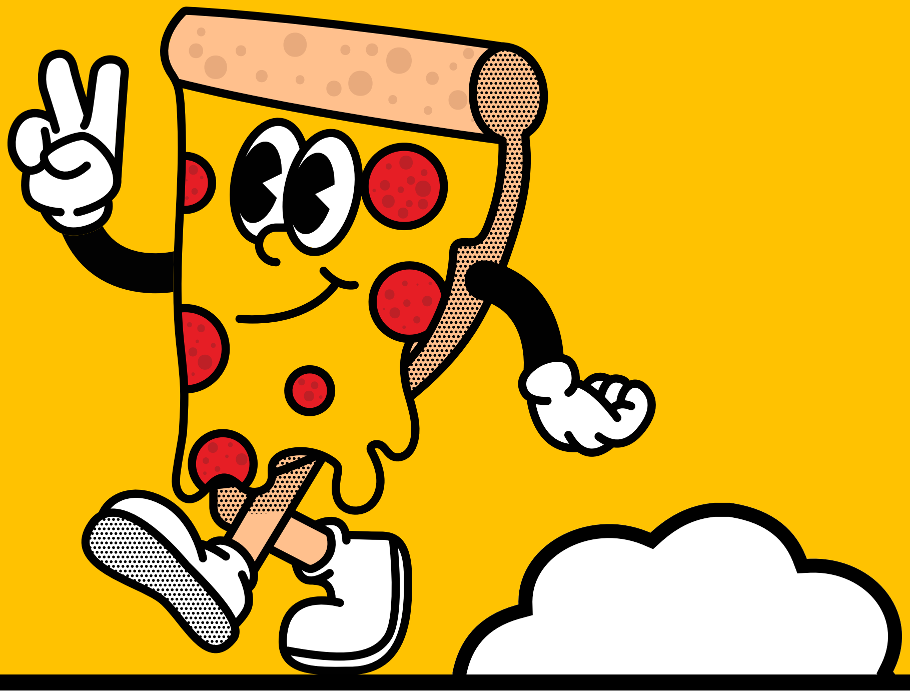


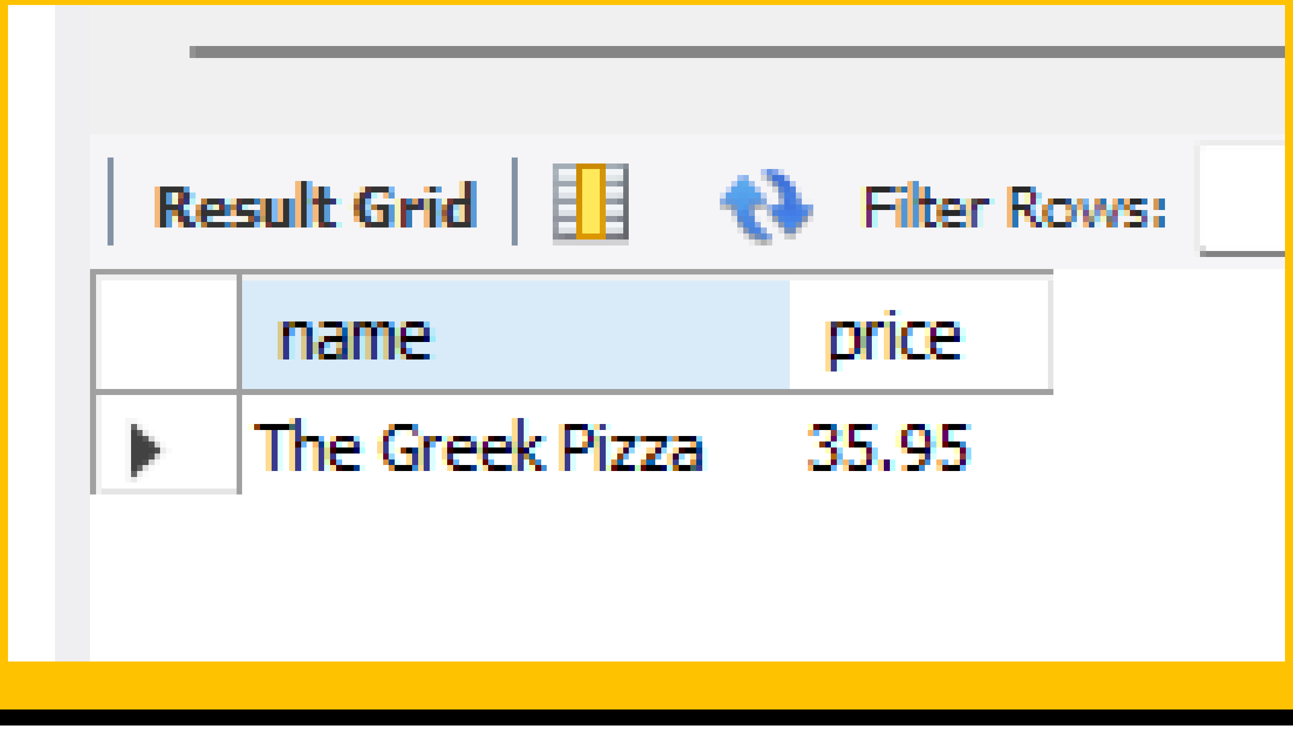
```
total_sales
817860.05
```

IDENTIFY THE HIGHEST-PRICED PIZZA.



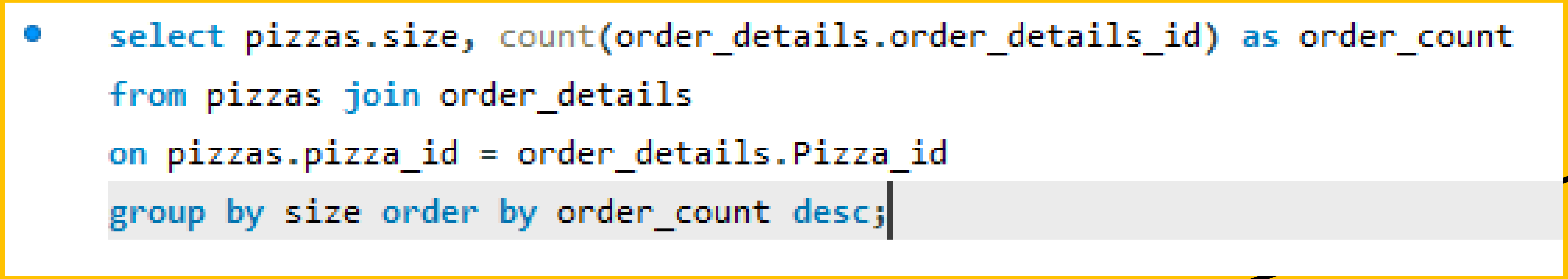
```sql
SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```



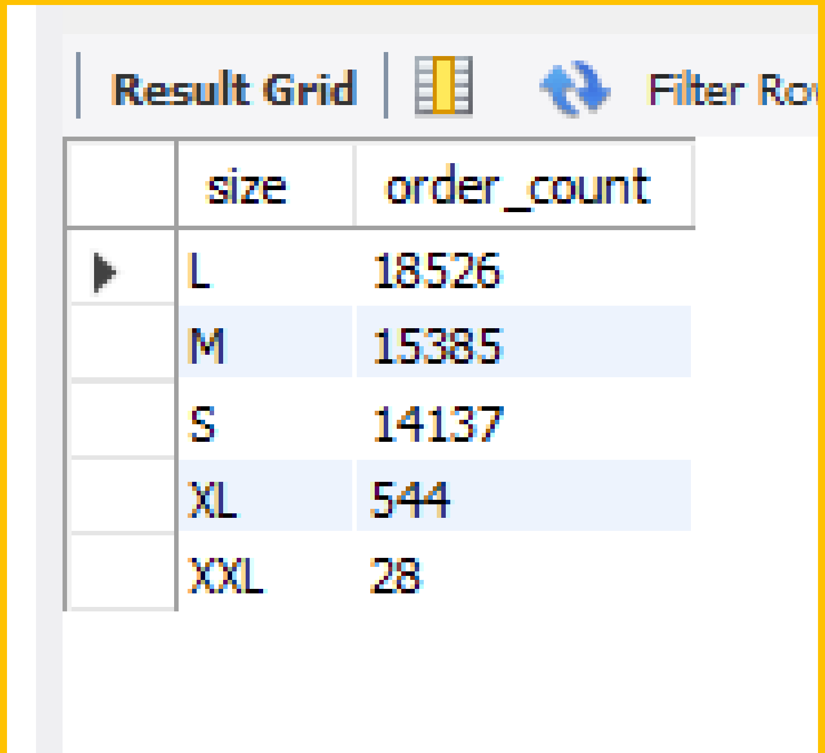


name	price
The Greek Pizza	35.95

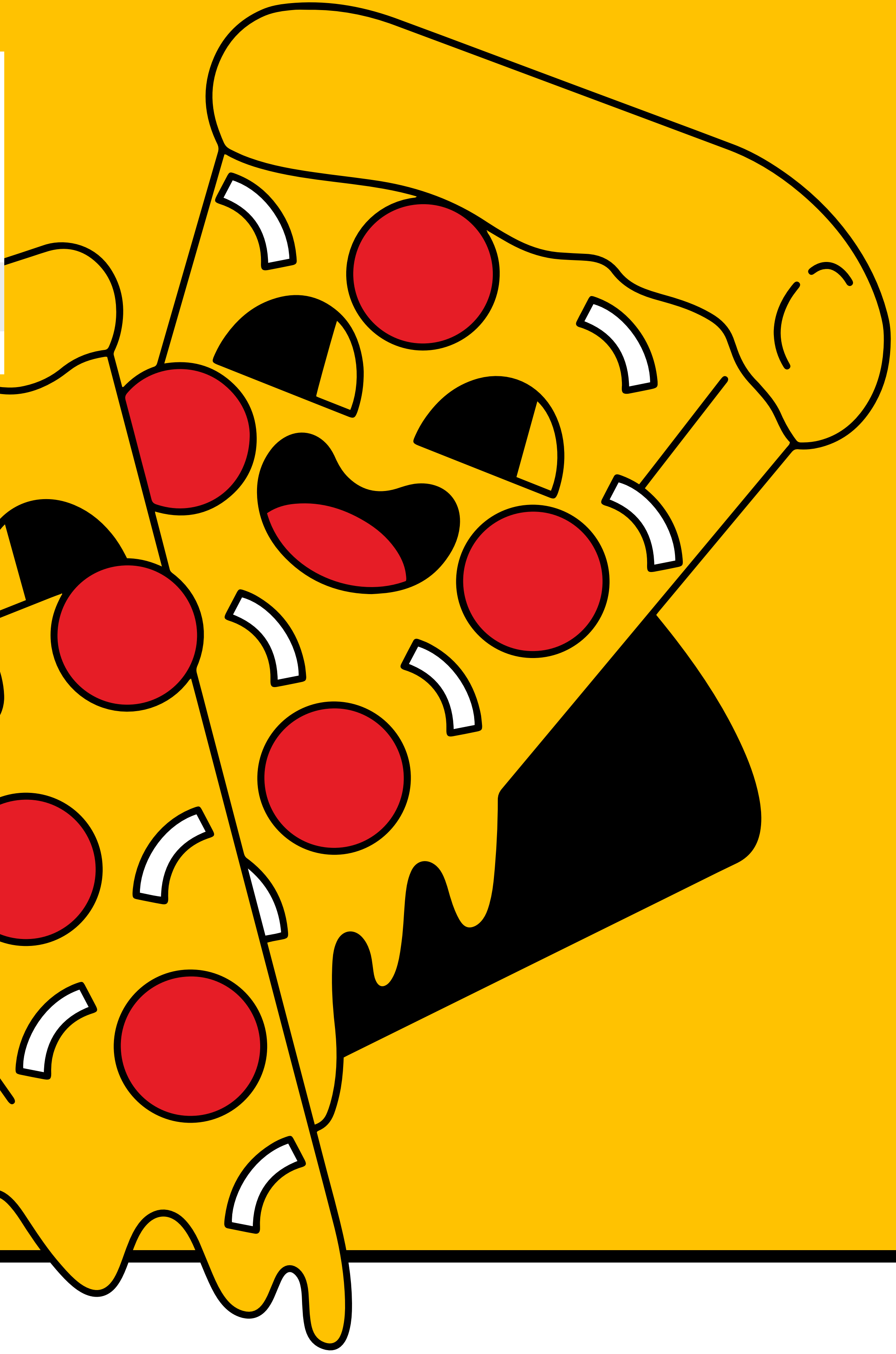
IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.



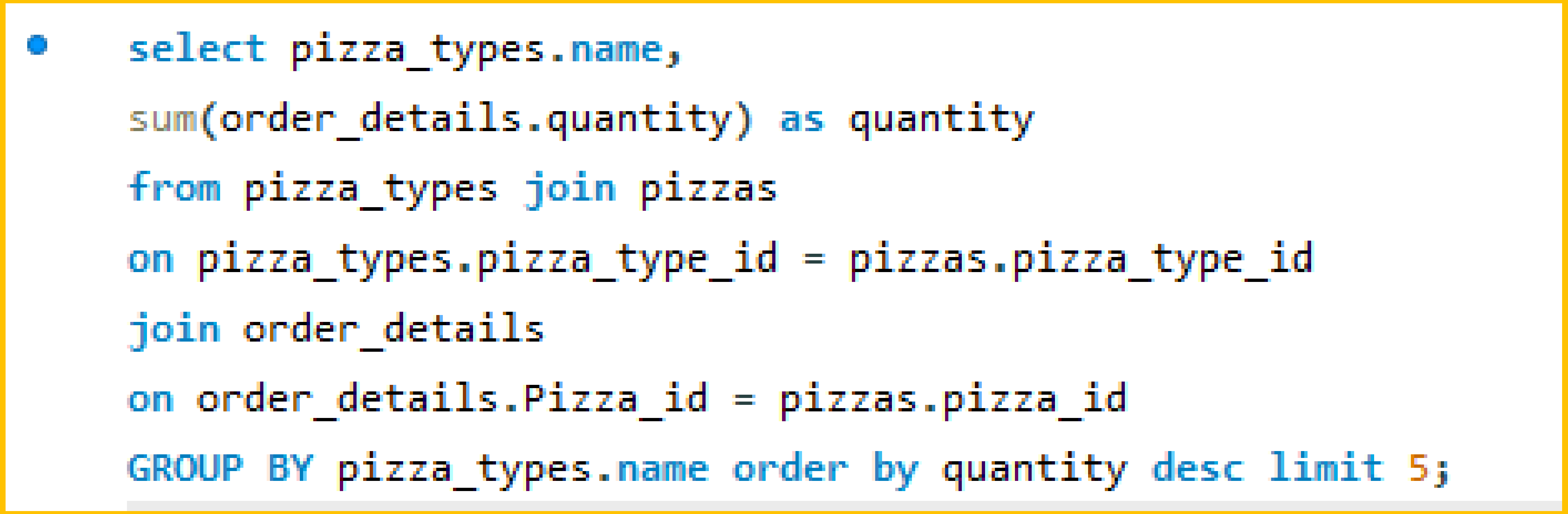
```sql
select pizzas.size, count(order_details.order_details_id) as order_count
from pizzas join order_details
on pizzas.pizza_id = order_details.Pizza_id
group by size order by order_count desc;
```



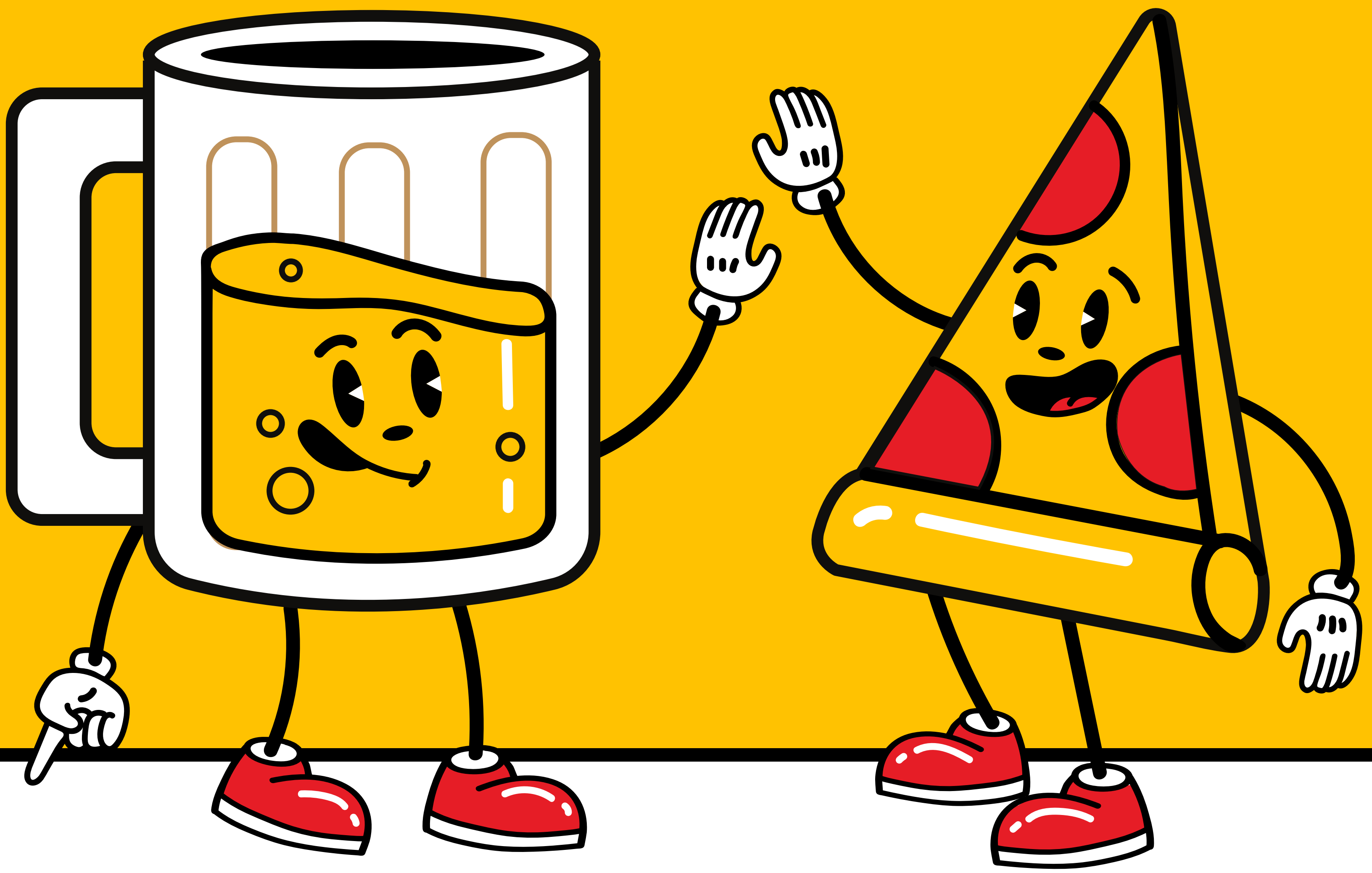
size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

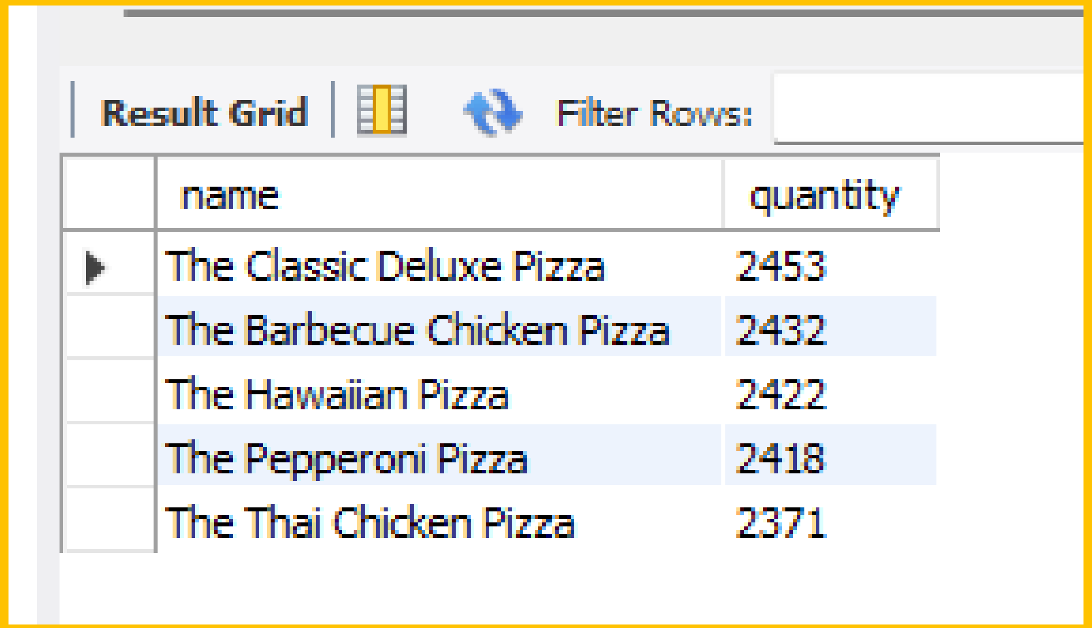


LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.



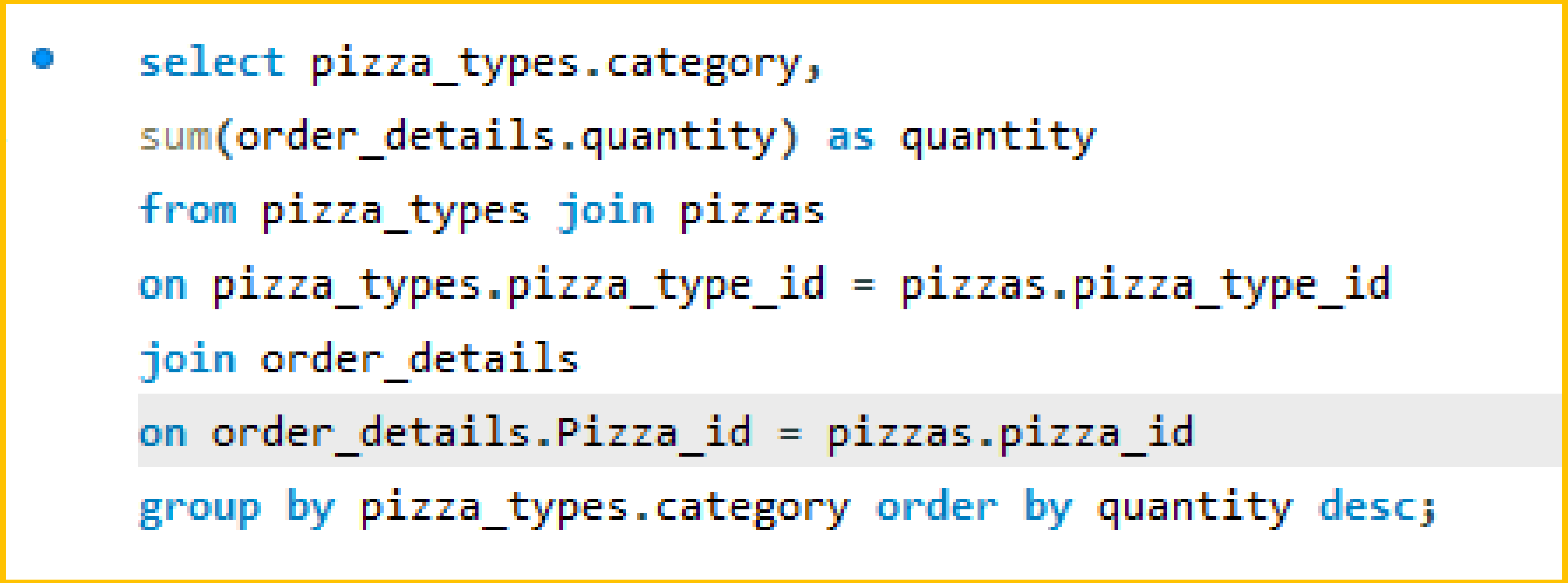
```sql
select pizza_types.name,
sum(order_details.quantity) as quantity
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
join order_details
on order_details.Pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name order by quantity desc limit 5;
```



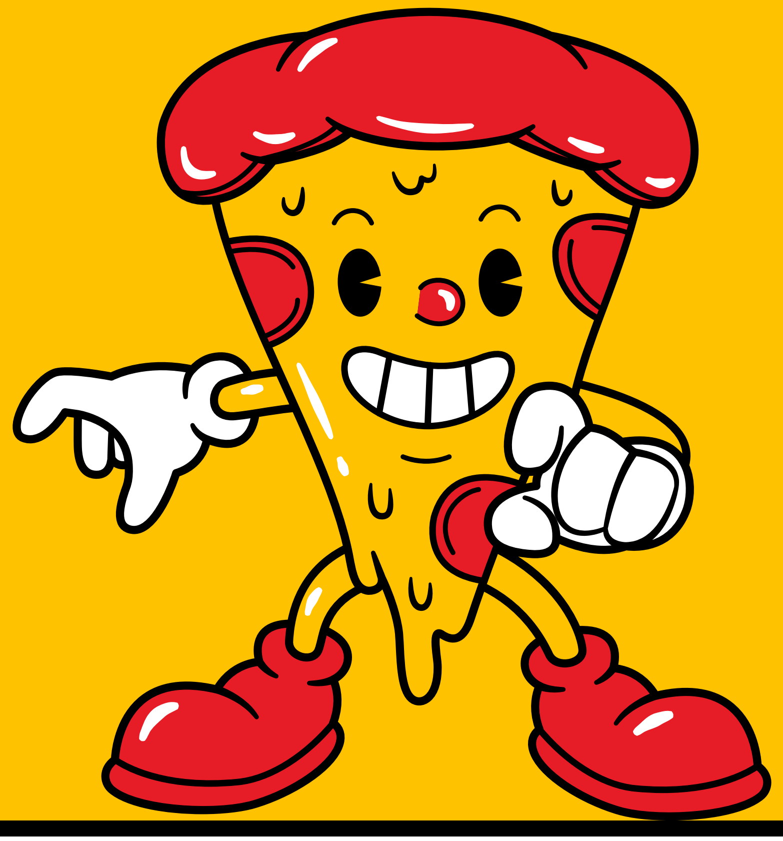


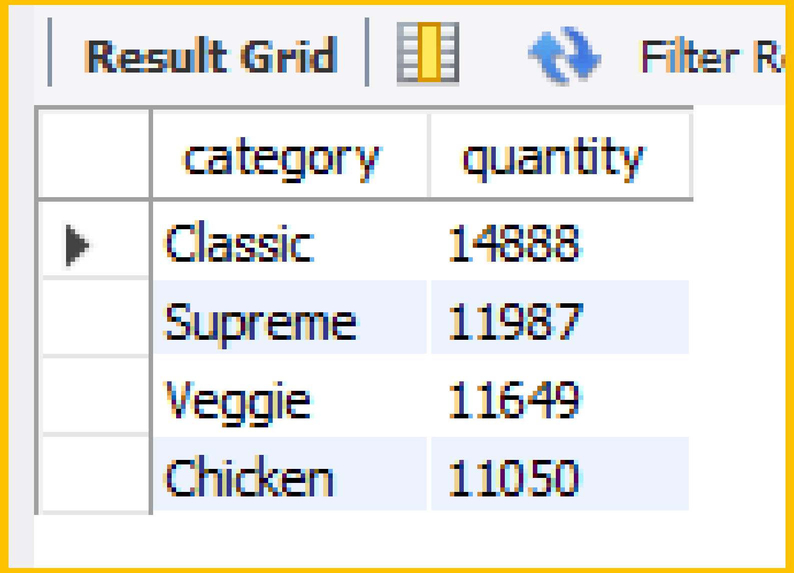
name	quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

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



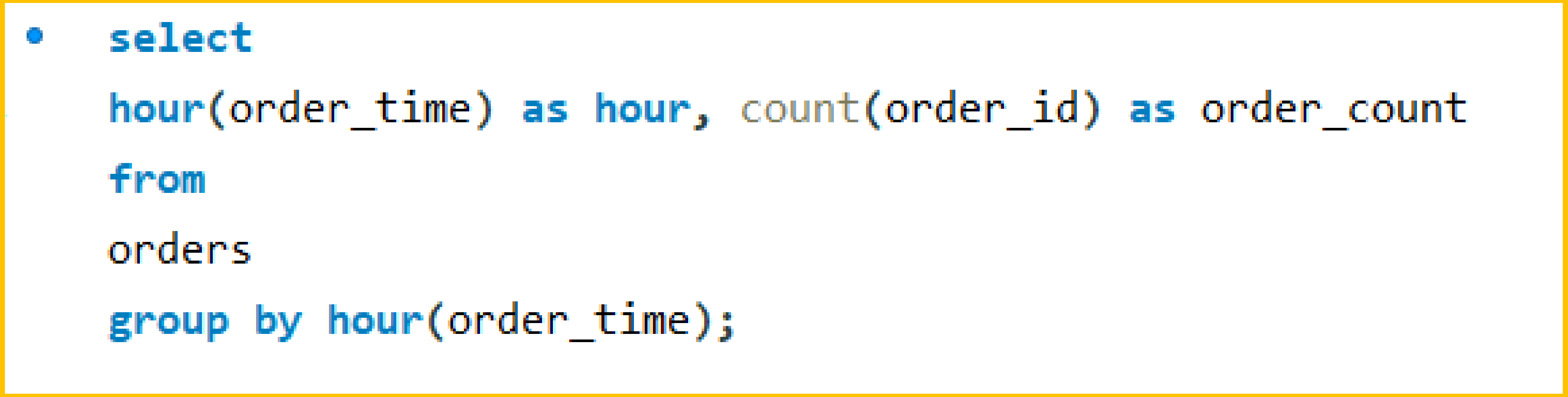
```
select pizza_types.category,
sum(order_details.quantity) as quantity
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
join order_details
on order_details.Pizza_id = pizzas.pizza_id
group by pizza_types.category order by quantity desc;
```



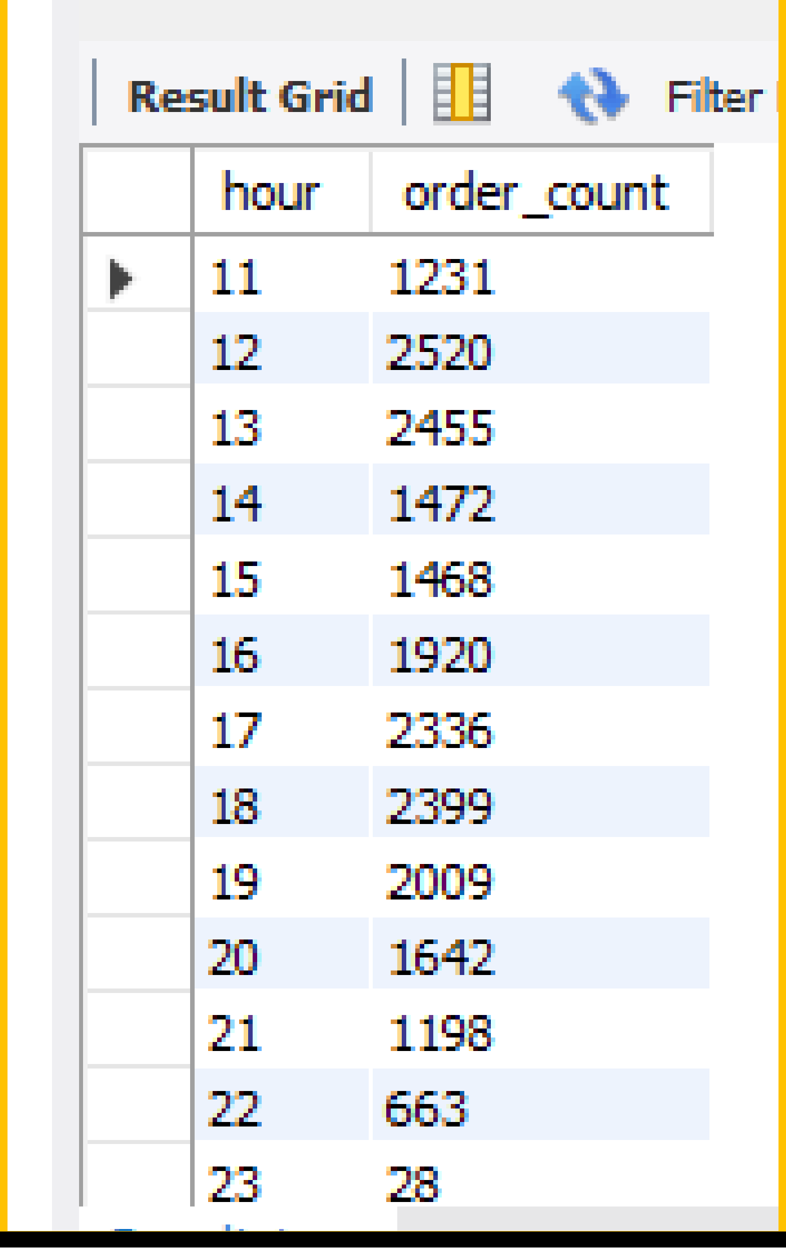


category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

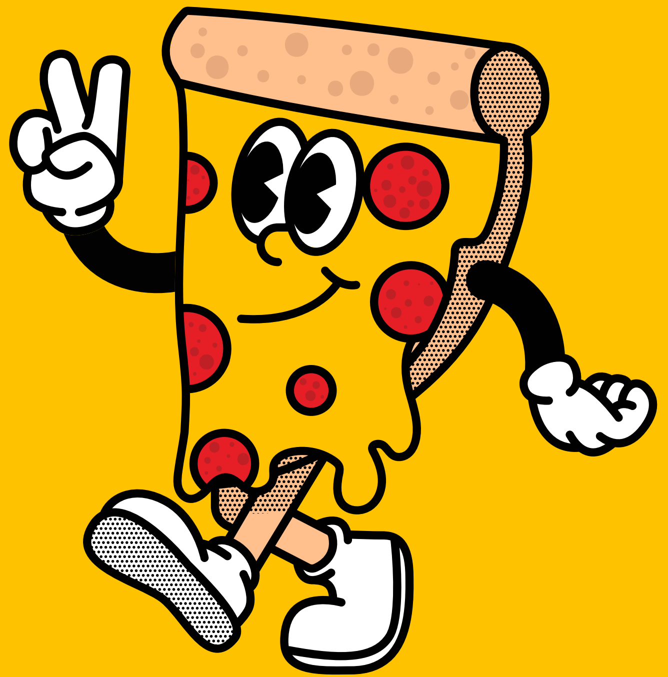
DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.



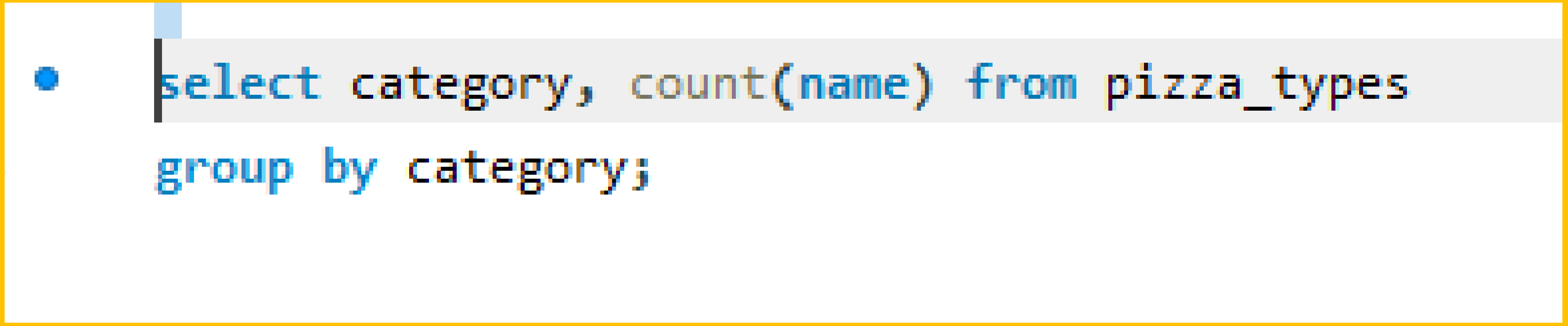
```
select
hour(order_time) as hour, count(order_id) as order_count
from
orders
group by hour(order_time);
```



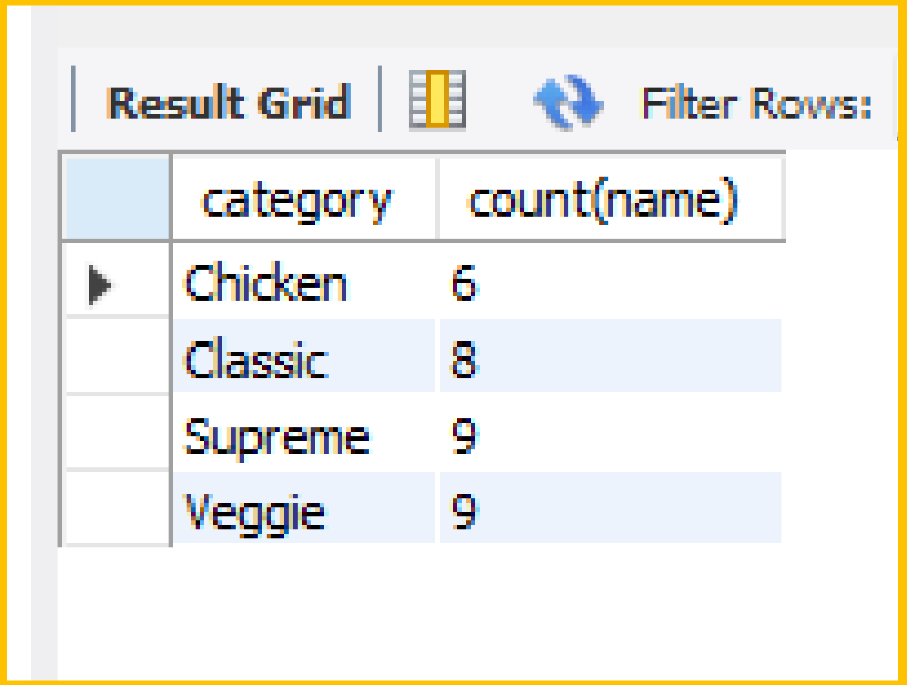
hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28



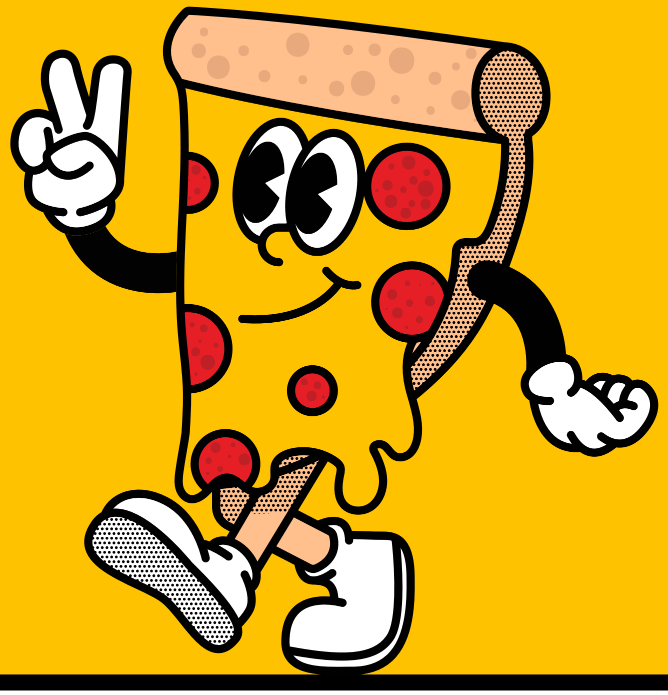
JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.



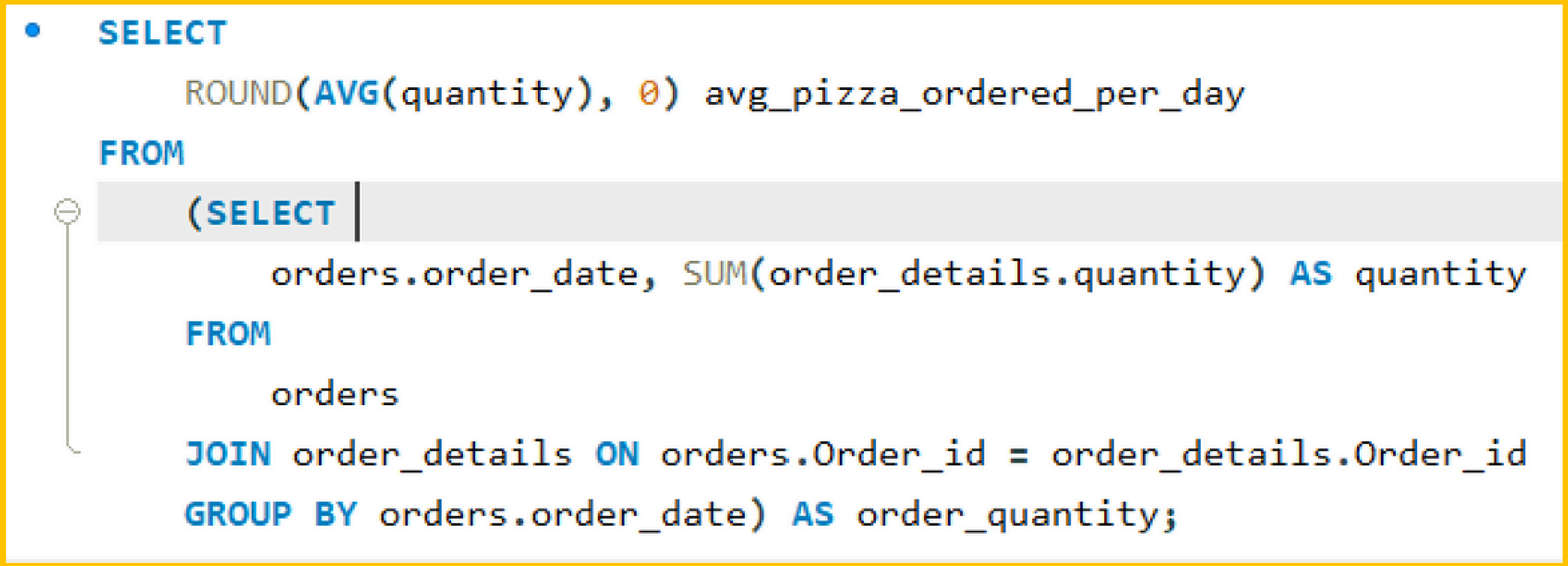
```
select category, count(name) from pizza_types
group by category;
```



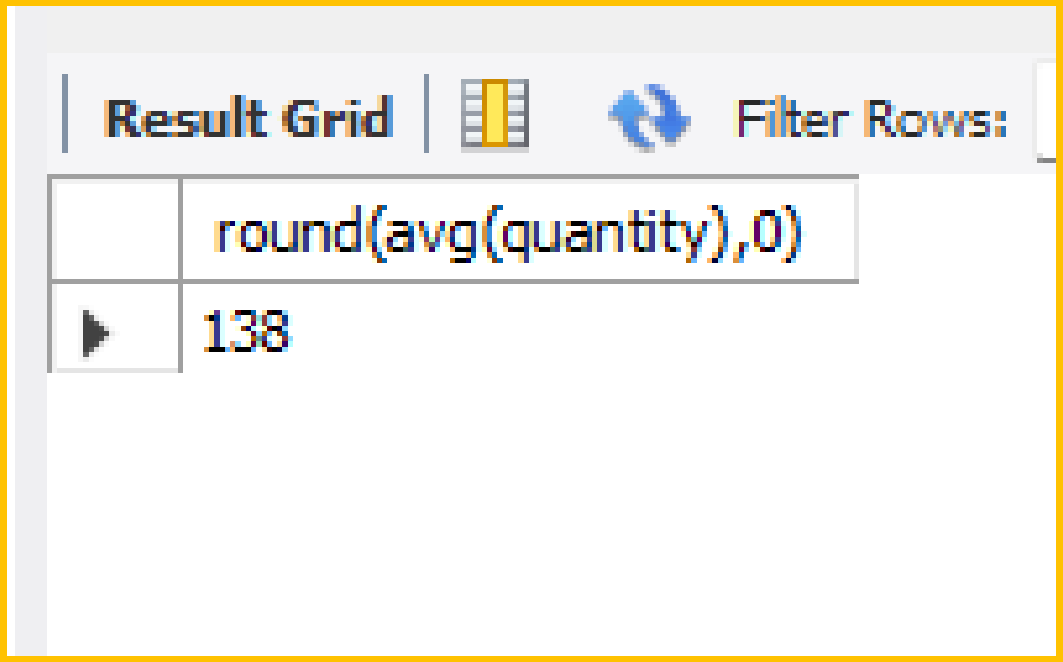
category	count(name)
Chicken	6
Classic	8
Supreme	9
Veggie	9



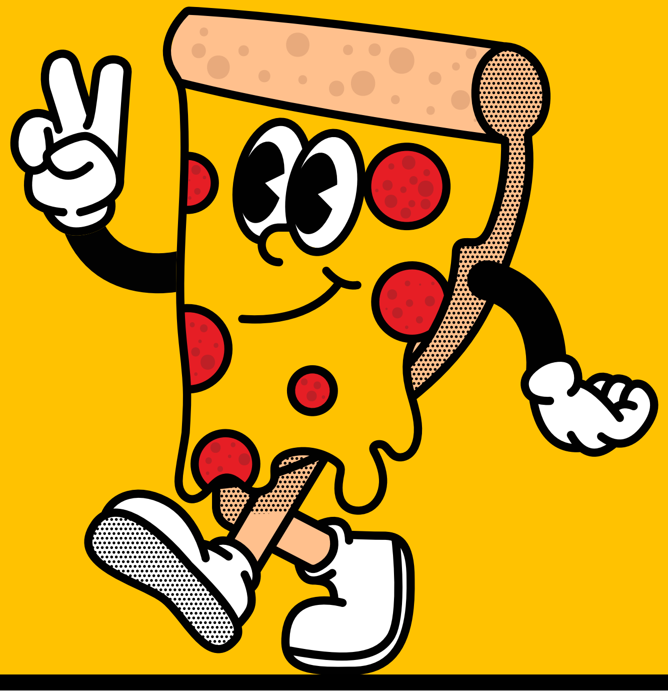
GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.



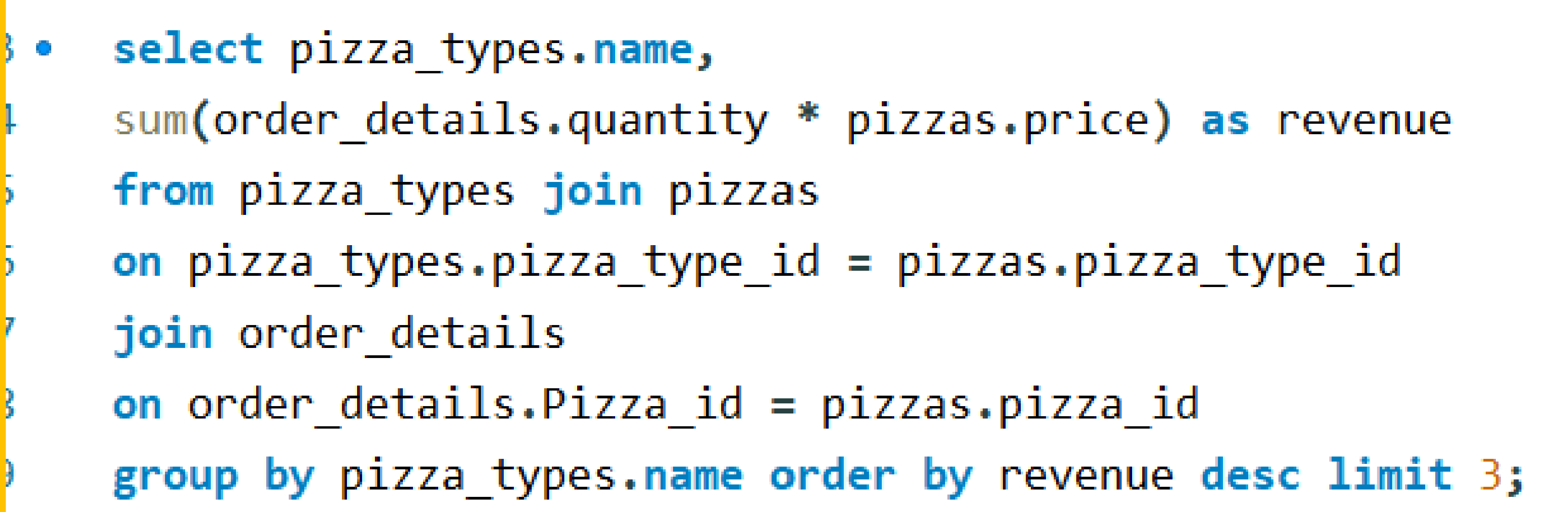
```sql
SELECT
    ROUND(AVG(quantity), 0) avg_pizza_ordered_per_day
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.Order_id = order_details.Order_id
    GROUP BY orders.order_date) AS order_quantity;
```



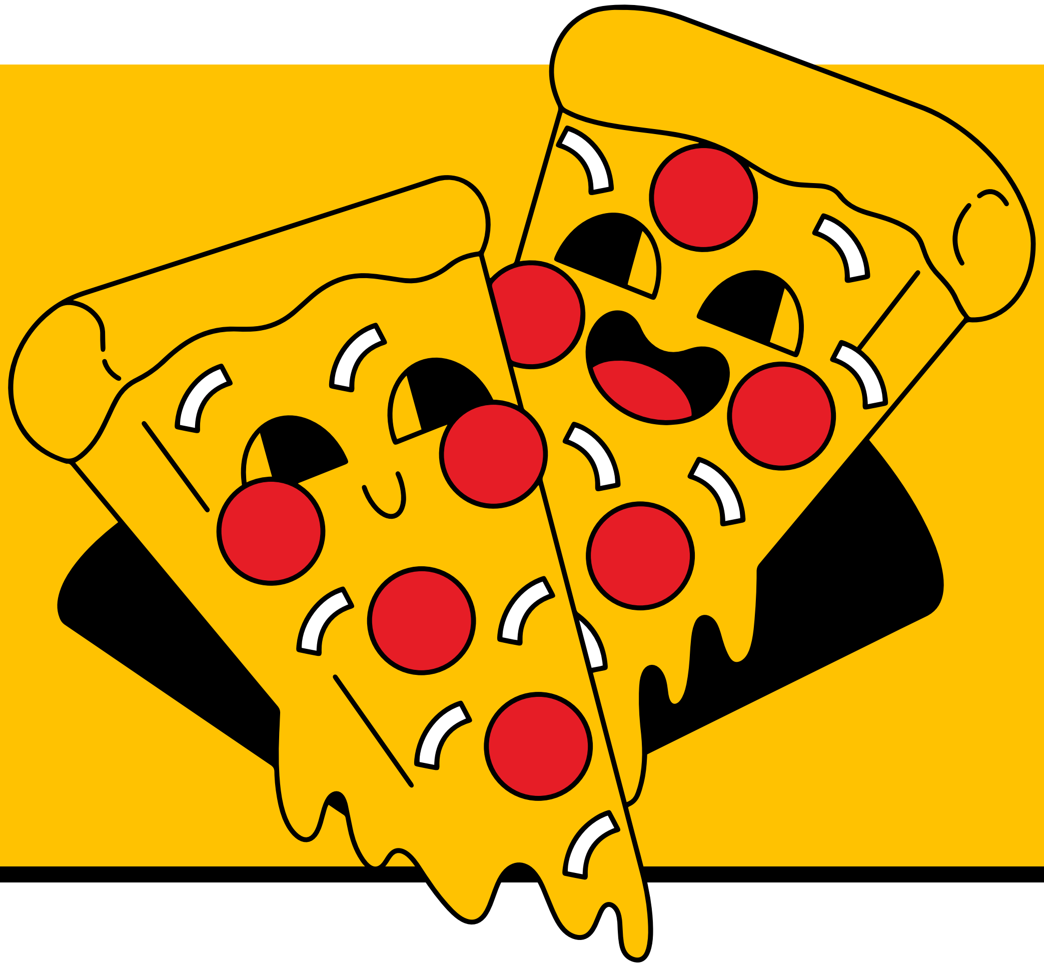
```
round(avg(quantity),0)
138
```

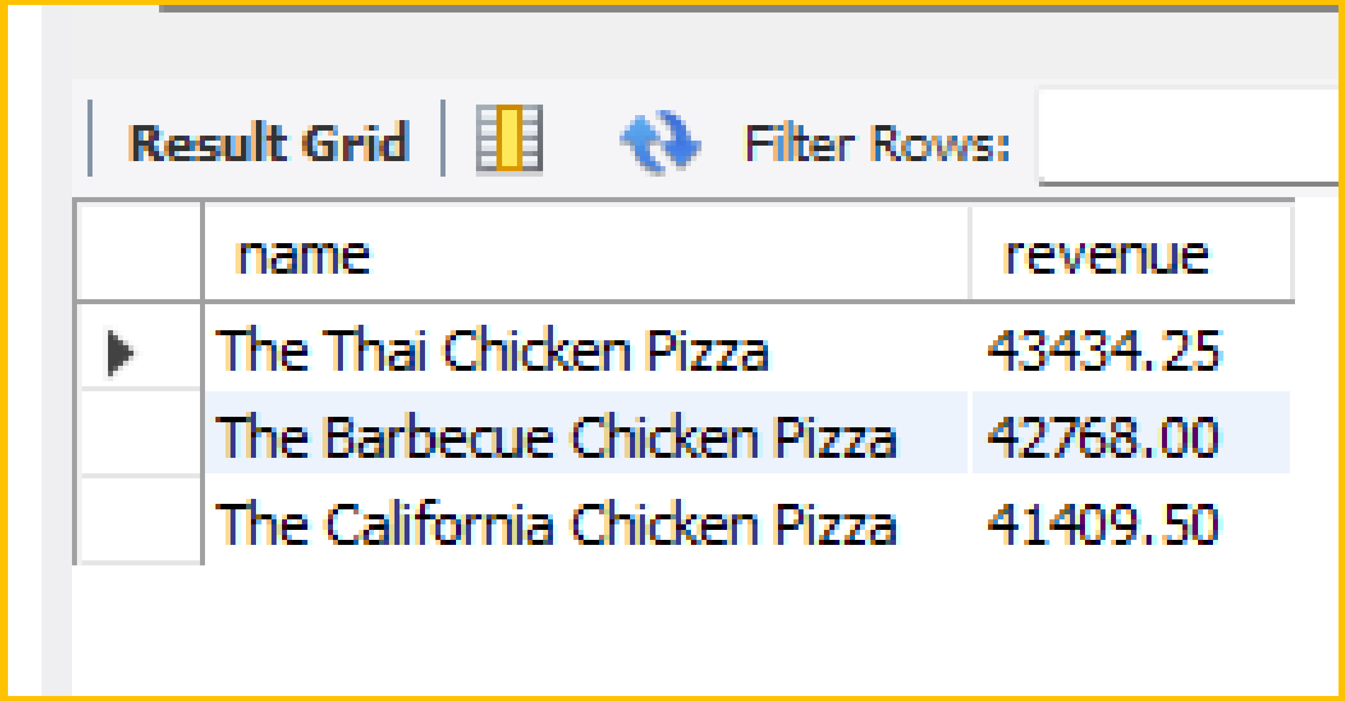


DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.



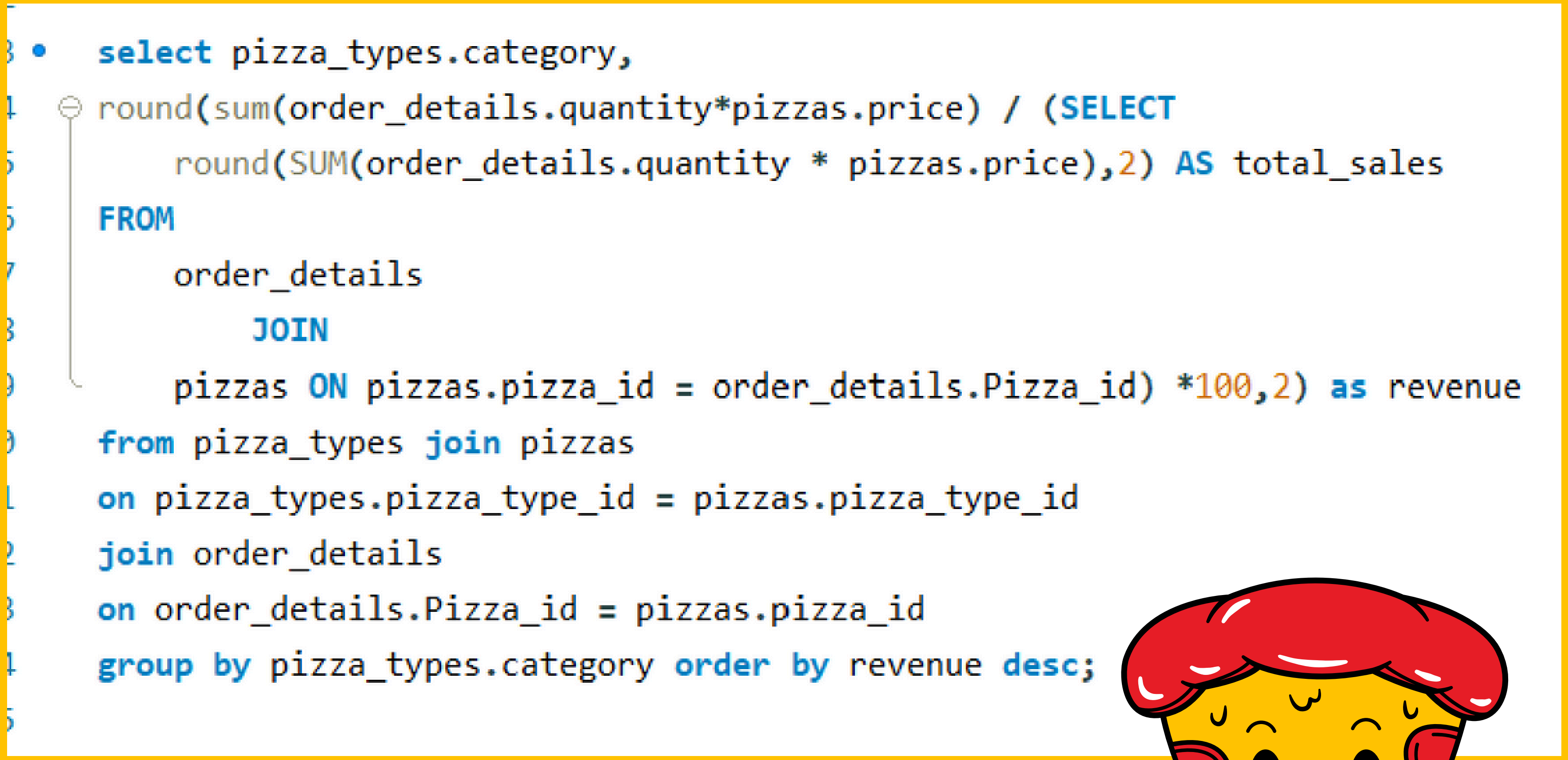
```
select pizza_types.name,
sum(order_details.quantity * pizzas.price) as revenue
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
join order_details
on order_details.Pizza_id = pizzas.pizza_id
group by pizza_types.name order by revenue desc limit 3;
```



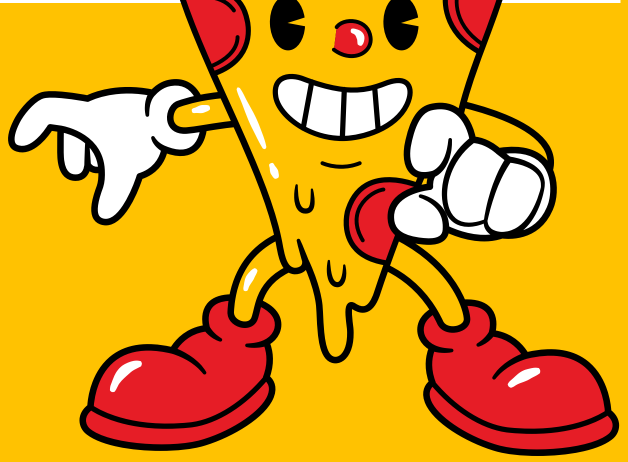


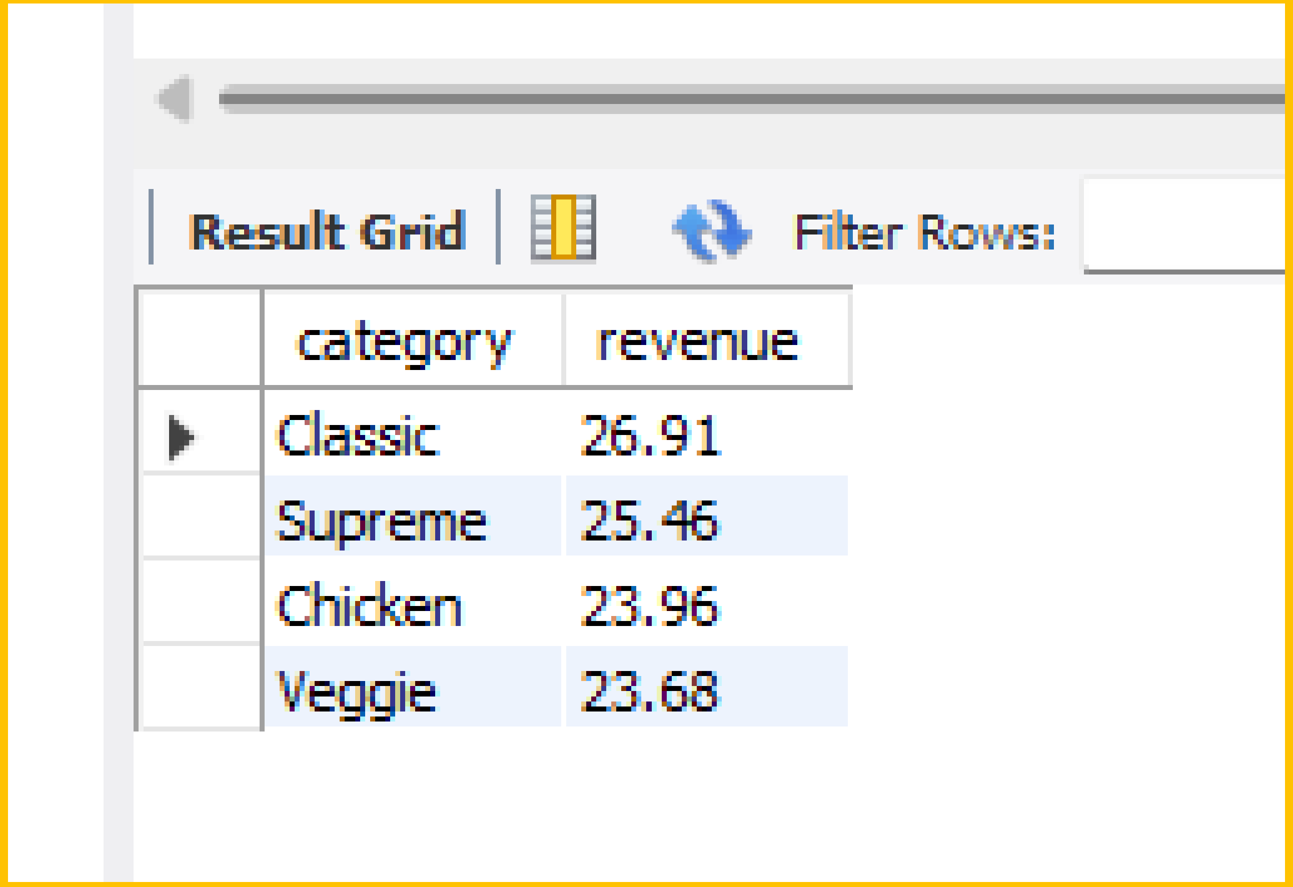
name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768.00
The California Chicken Pizza	41409.50

CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE.



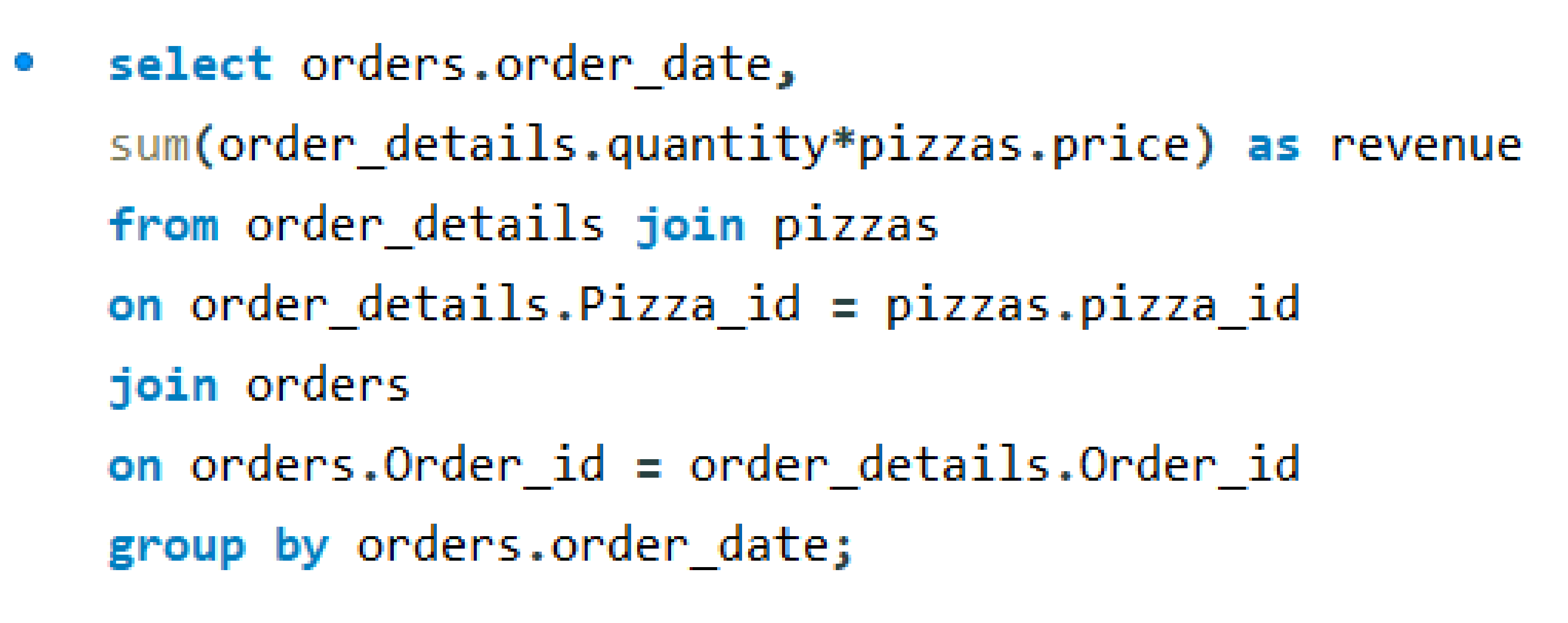
```
select pizza_types.category,
round(sum(order_details.quantity*pizzas.price) / (SELECT
    round(SUM(order_details.quantity * pizzas.price),2) AS total_sales
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.Pizza_id) *100,2) as revenue
from pizza_types join pizzas
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;
```



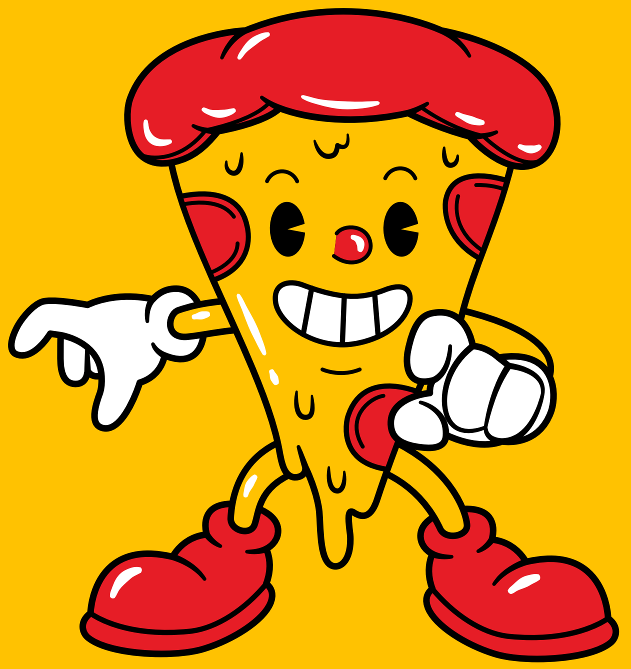


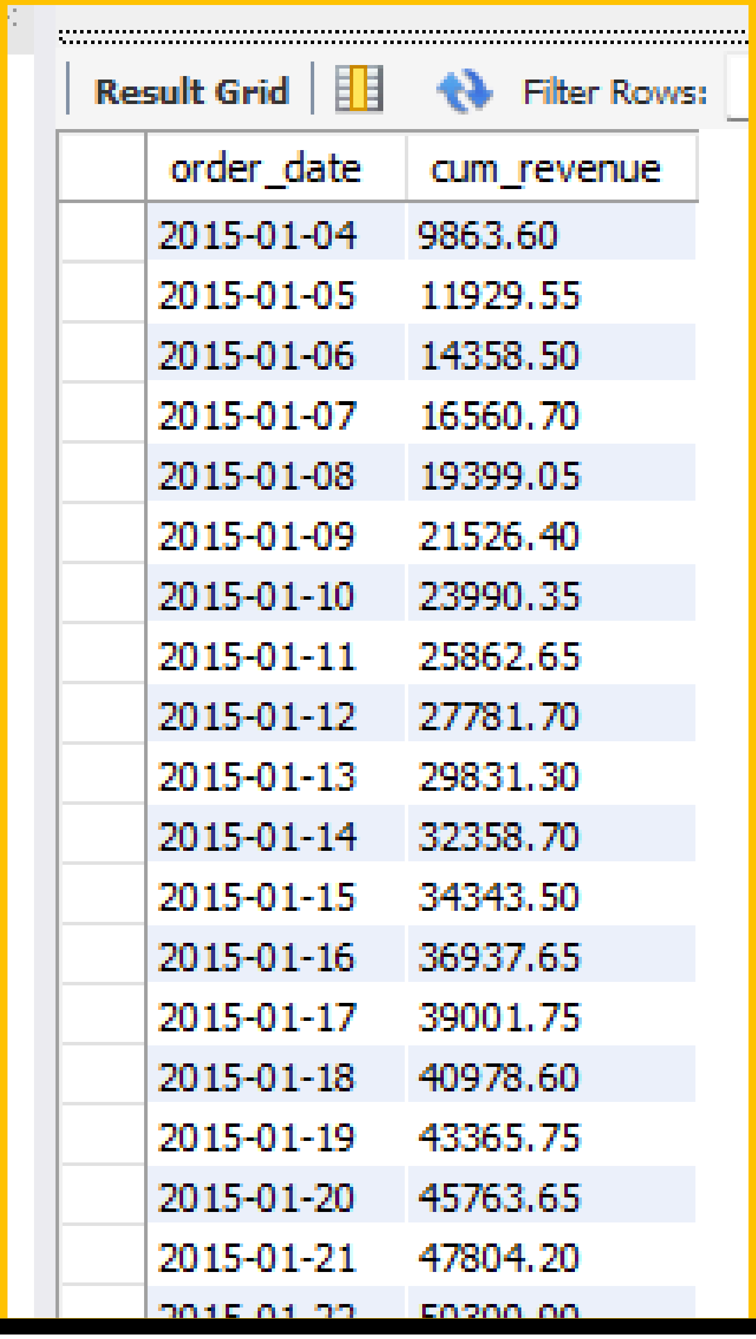
category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.



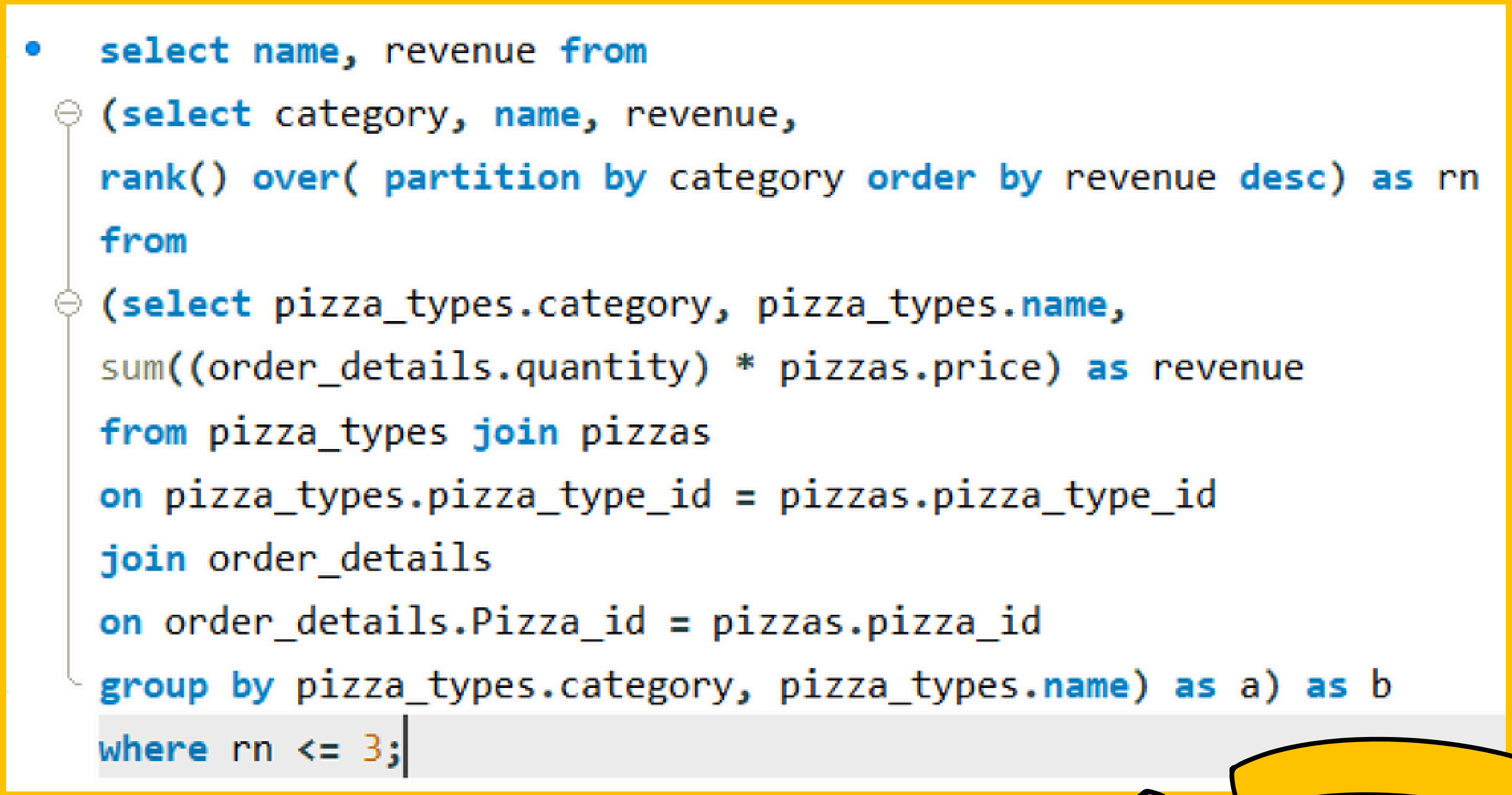
```sql
select orders.order_date,
sum(order_details.quantity*pizzas.price) as revenue
from order_details join pizzas
on order_details.Pizza_id = pizzas.pizza_id
join orders
on orders.Order_id = order_details.Order_id
group by orders.order_date;
```



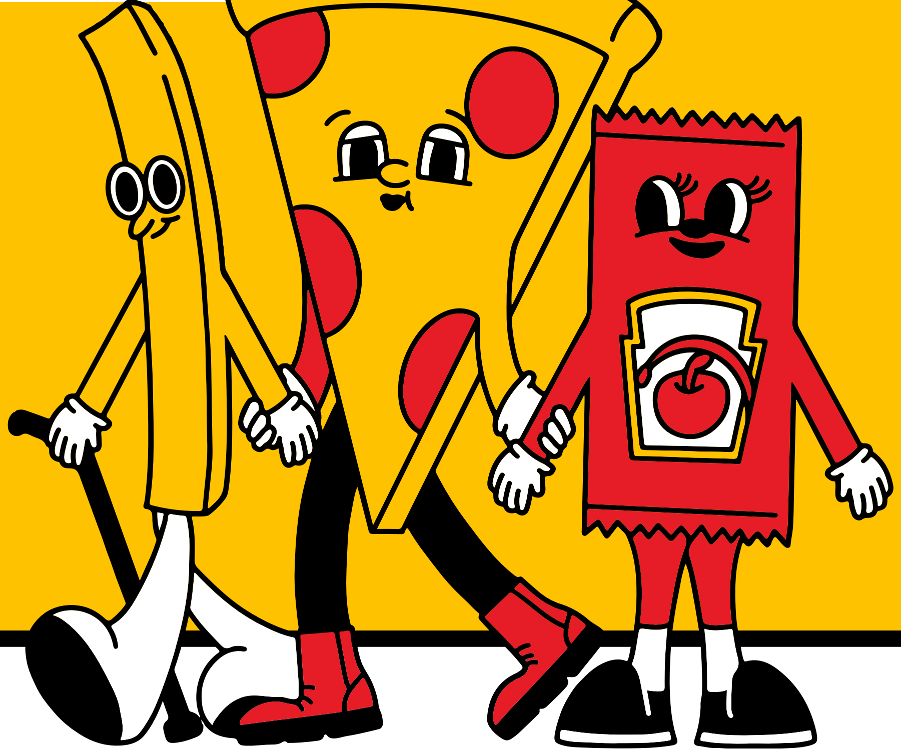


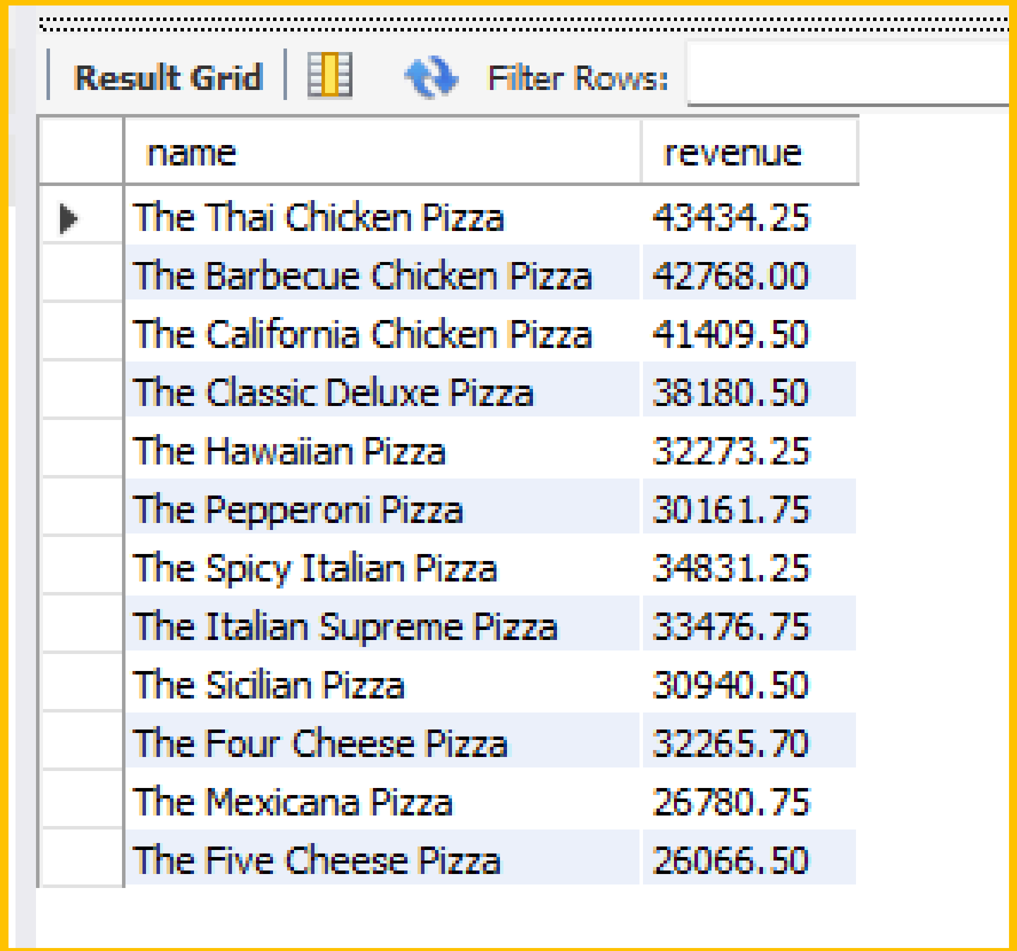
order_date	cum_revenue
2015-01-04	9863.60
2015-01-05	11929.55
2015-01-06	14358.50
2015-01-07	16560.70
2015-01-08	19399.05
2015-01-09	21526.40
2015-01-10	23990.35
2015-01-11	25862.65
2015-01-12	27781.70
2015-01-13	29831.30
2015-01-14	32358.70
2015-01-15	34343.50
2015-01-16	36937.65
2015-01-17	39001.75
2015-01-18	40978.60
2015-01-19	43365.75
2015-01-20	45763.65
2015-01-21	47804.20
2015-01-22	50300.00

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



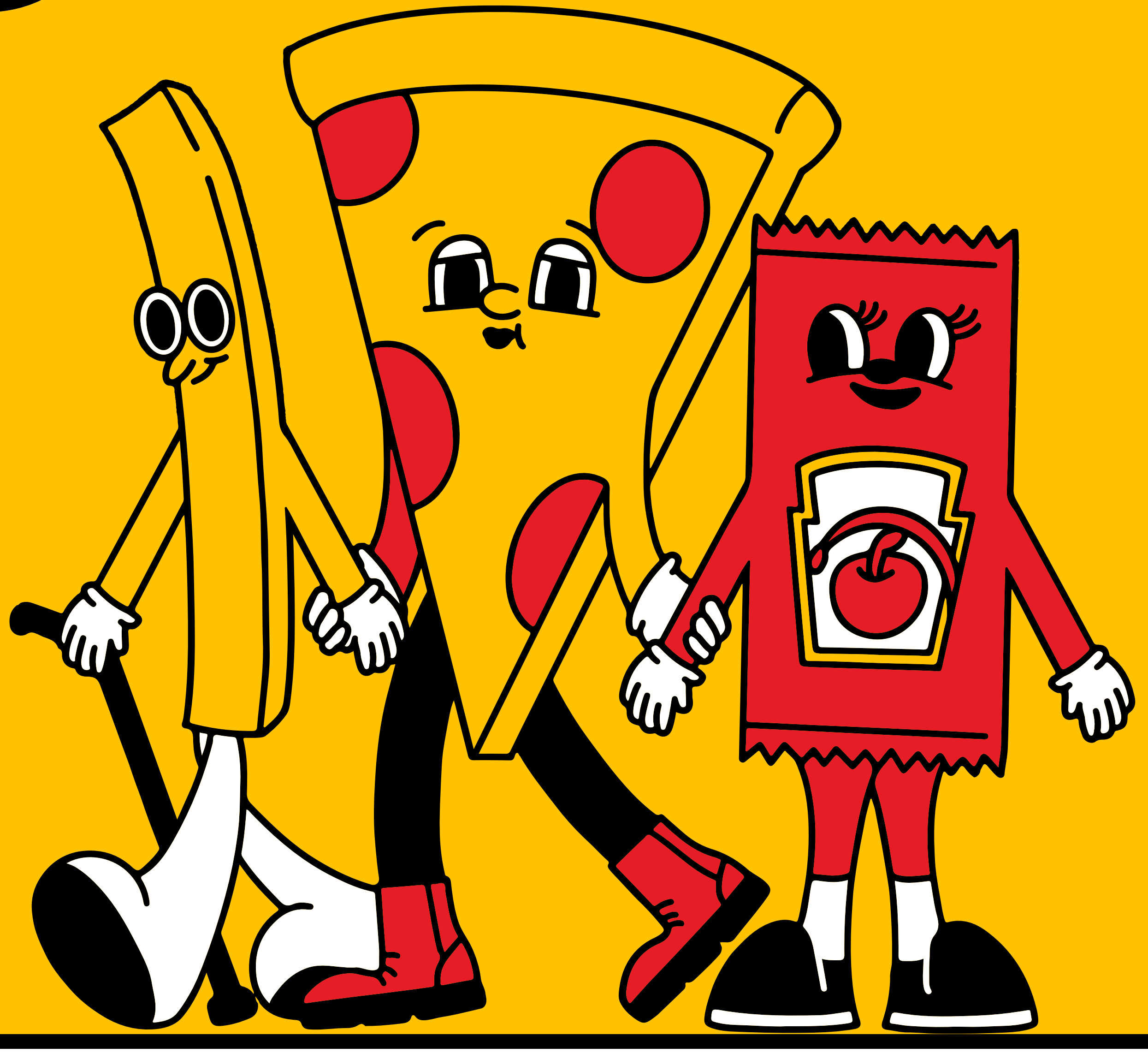
```
select name, revenue from
(select category, name, revenue,
rank() over( partition by category order by revenue desc) as rn
from
(select pizza_types.category, pizza_types.name,
sum((order_details.quantity) * pizzas.price) as revenue
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
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	42768.00
The California Chicken Pizza	41409.50
The Classic Deluxe Pizza	38180.50
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	30940.50
The Four Cheese Pizza	32265.70
The Mexicana Pizza	26780.75
The Five Cheese Pizza	26066.50

THANK YOU FOR YOUR TIME.!



IN THIS PROJECT, I ANALYZED PIZZA SALES DATA USING SQL TO ANSWER KEY BUSINESS QUESTIONS.

BY APPLYING CONCEPTS SUCH AS JOINS, AGGREGATE FUNCTIONS, GROUPING, I WAS ABLE TO EXTRACT MEANINGFUL INSIGHTS FROM THE DATASET.

THIS PROJECT STRENGTHENED MY UNDERSTANDING OF SQL AND DEMONSTRATED HOW DATA ANALYSIS CAN SUPPORT REAL-WORLD BUSINESS DECISION-MAKING.