SQL PROJECT

PIZZA_HUT

A PROJECT ON PIZZA HUT SALES ANALYSIS

PIZZA PPT PRESENTATIONS

PROJECT SUMMARY

01 OBJECTIVE

To analyse pizzas sales data :-

- Total number of orders placed
- Total revenue generated
- Highest-priced pizza
- Most common pizza size ordered
- Analyze the cumulative revenue

02 FEATURES ON TOPIC

- SUBQUERY
- GROUP BY
- HAVING
- AGGREGATIONS FUNCTION

03 ABOUT DATASET

To analyse pizzas sales data we have:-

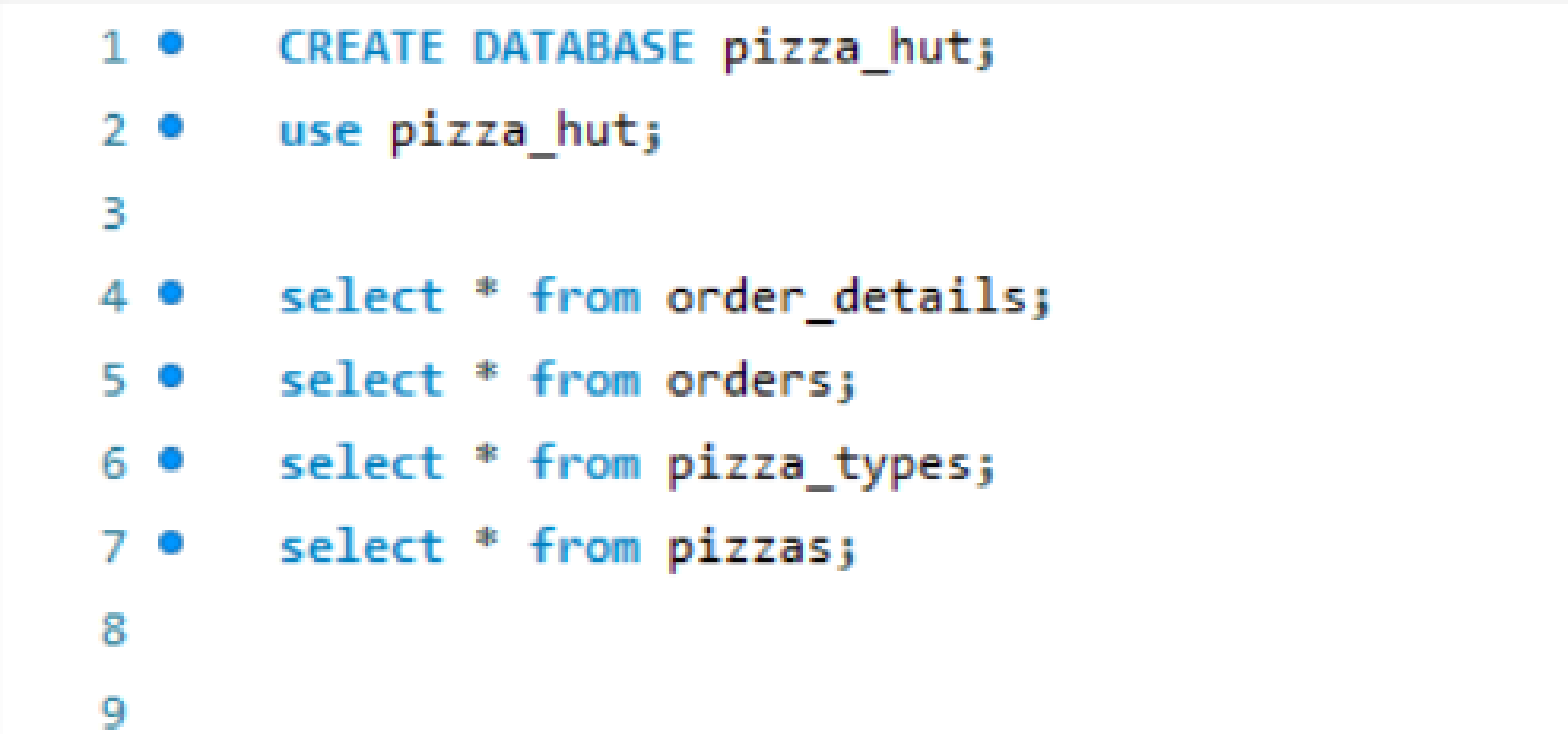
- ORDERS CSV FILE
- ORDERS_DETAILS CSV FILE
- PIZZA_TYPES CSV FILE
- PIZZAS

04 TOOLS USED

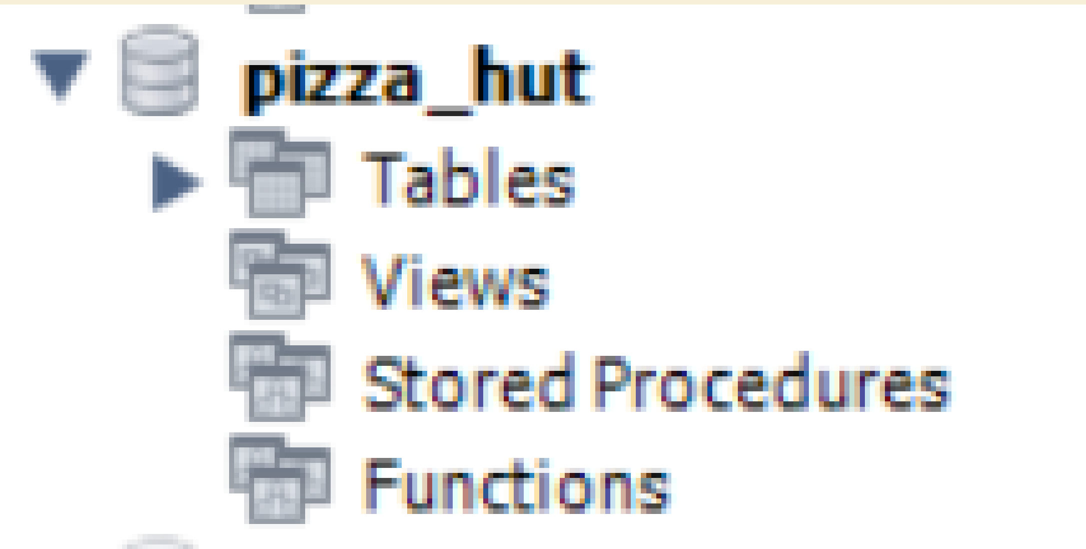
To analyse pizzas sales data we used:-

- **MYSQL** • FOR DATA QUERYING AND DATA MANIPULATING

DATABASE & ALL SALES DATA



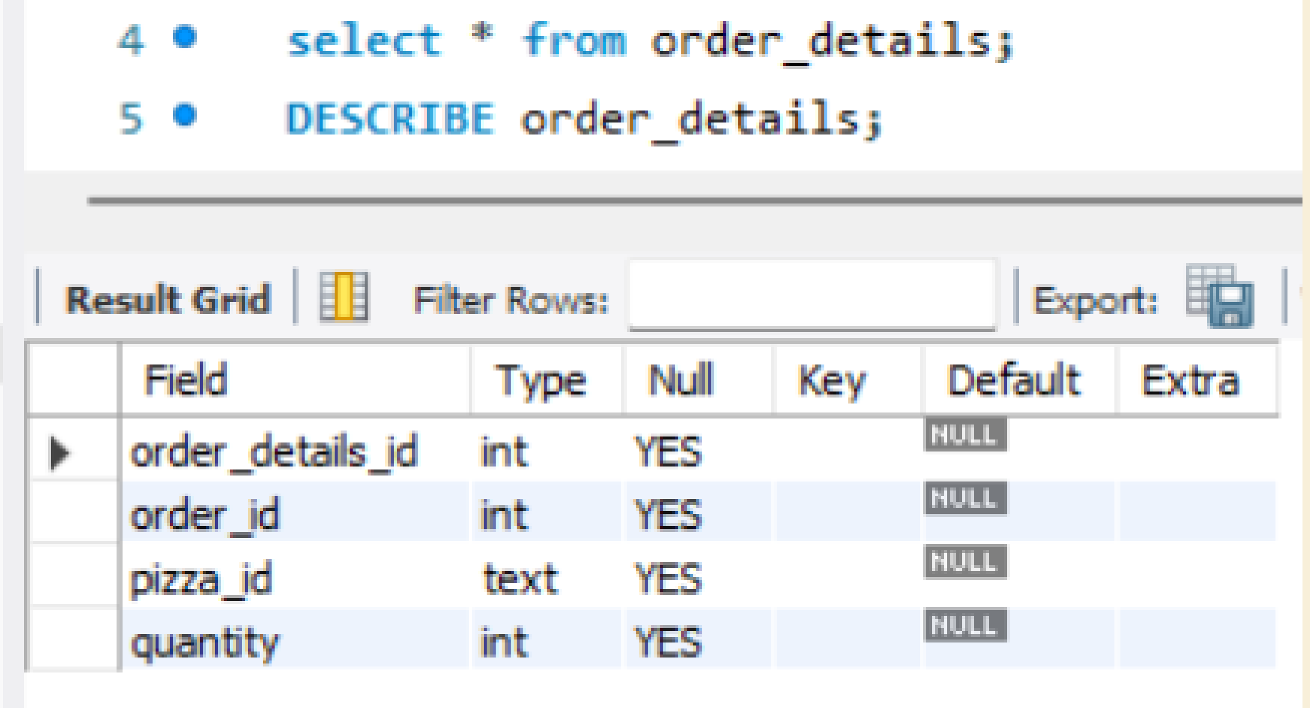
```sql
CREATE DATABASE pizza_hut;
use pizza_hut;

select * from order_details;
select * from orders;
select * from pizza_types;
select * from pizzas;


```



pizza_hut
Tables
Views
Stored Procedures
Functions

DATA SCHEMA

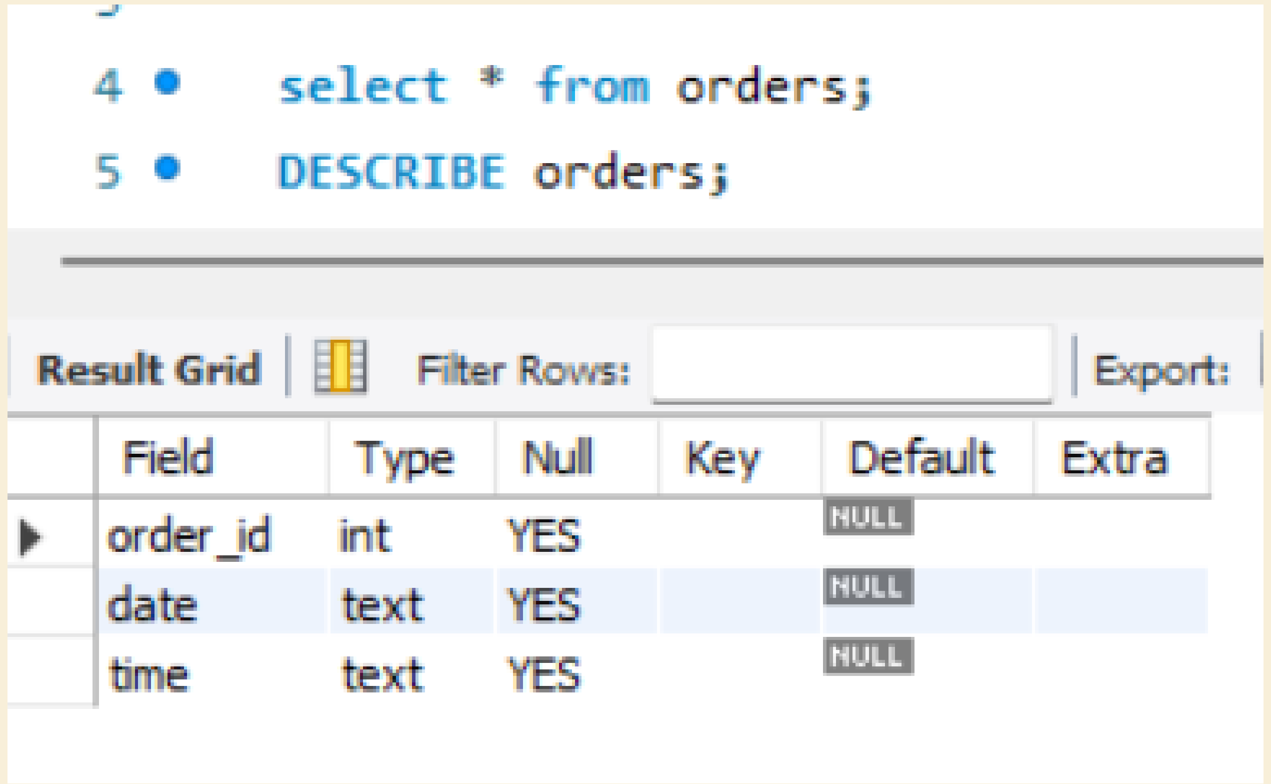
1.ORDER_DEATILS



```sql
select * from order_details;
DESCRIBE order_details;
```

Field	Type	Null	Key	Default	Extra
order_details_id	int	YES		NULL	
order_id	int	YES		NULL	
pizza_id	text	YES		NULL	
quantity	int	YES		NULL	

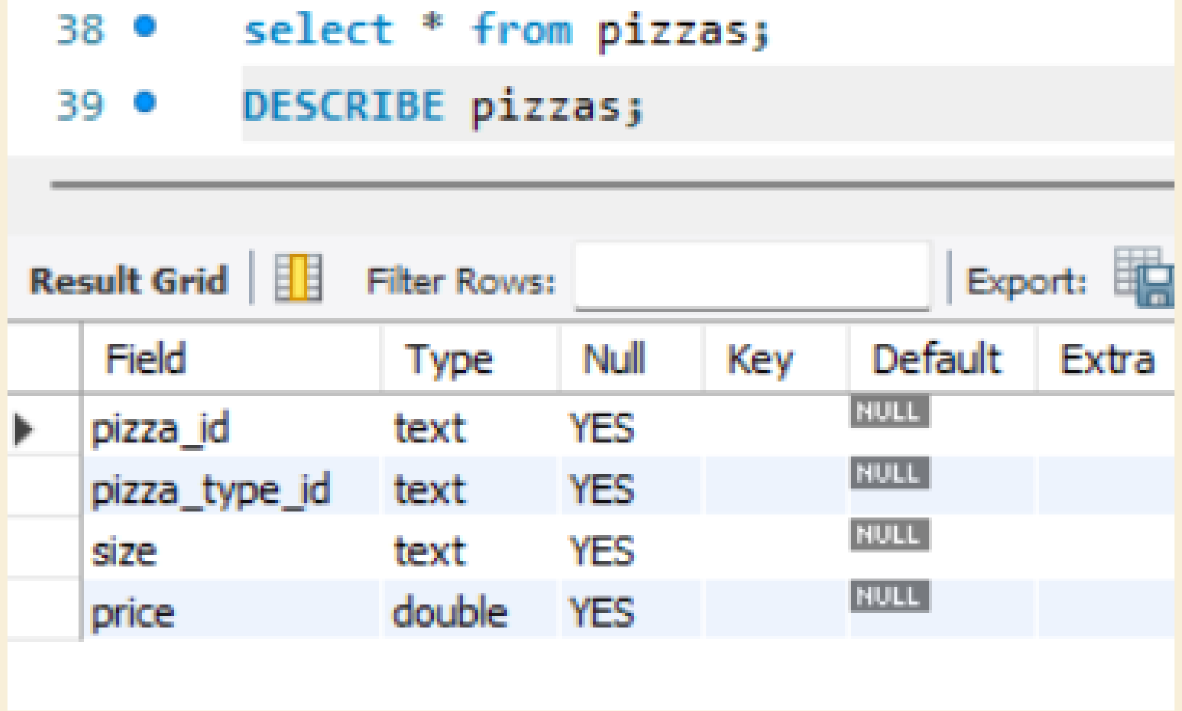
2. ORDERS



```sql
select * from orders;
DESCRIBE orders;
```

Field	Type	Null	Key	Default	Extra
order_id	int	YES		NULL	
date	text	YES		NULL	
time	text	YES		NULL	

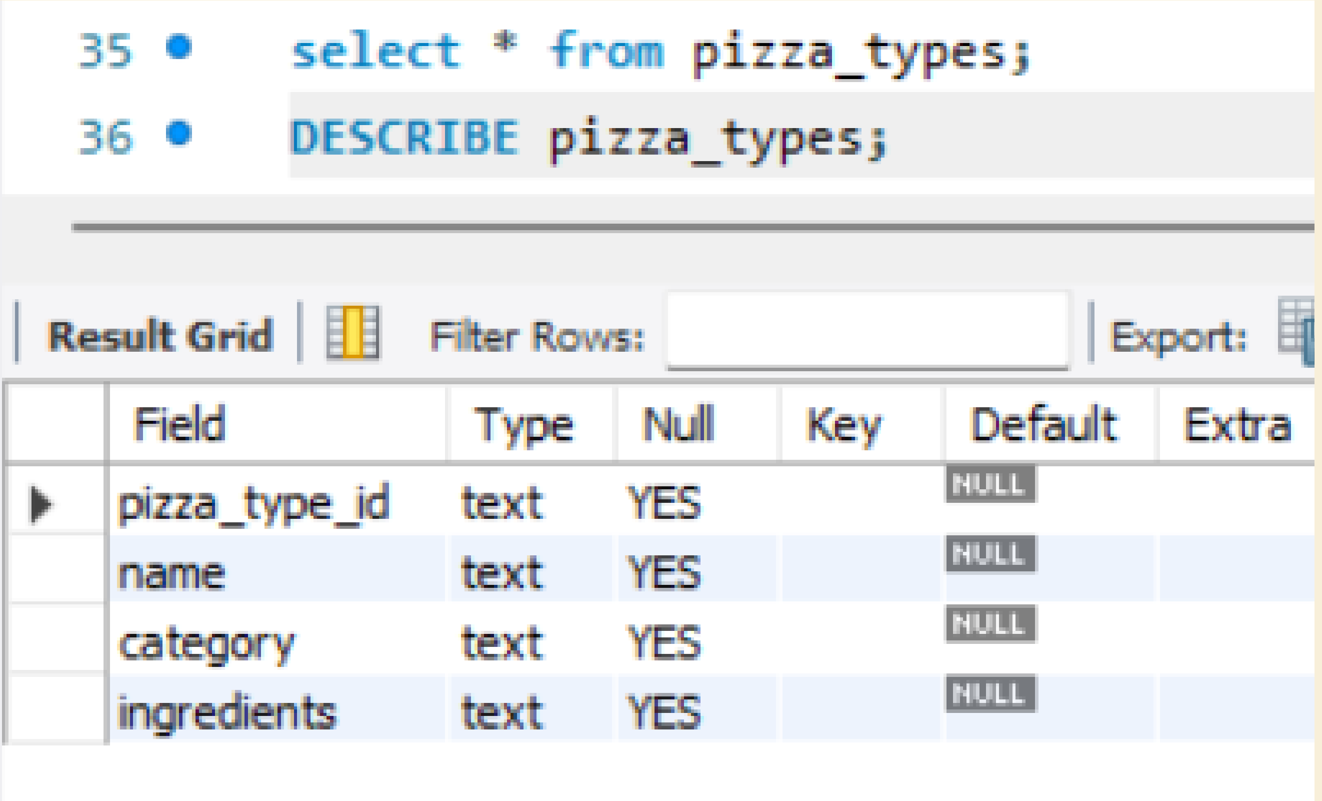
3. PIZZAS



```sql
select * from pizzas;
DESCRIBE pizzas;
```

Field	Type	Null	Key	Default	Extra
pizza_id	text	YES		NULL	
pizza_type_id	text	YES		NULL	
size	text	YES		NULL	
price	double	YES		NULL	

4. PIZZAS_TYPES

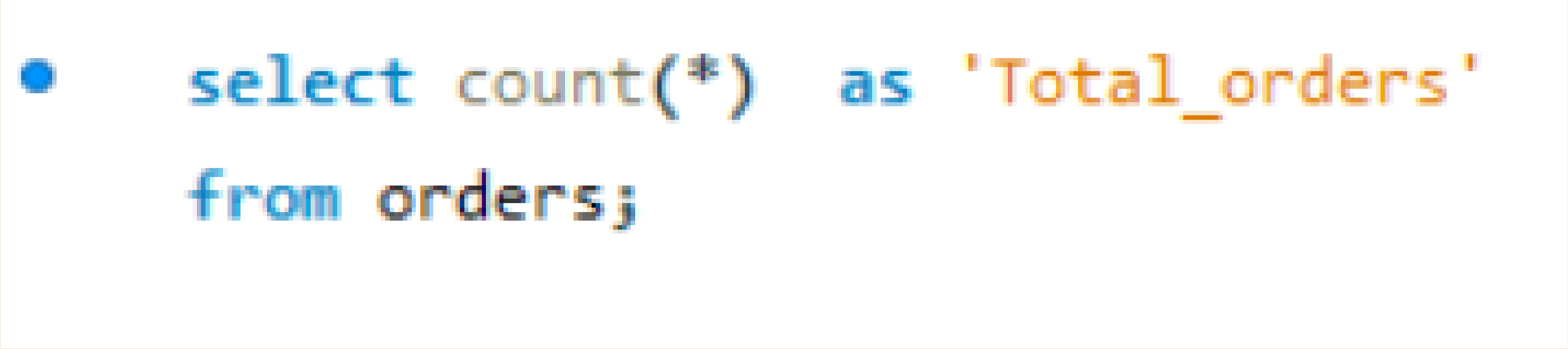


```sql
select * from pizza_types;
DESCRIBE pizza_types;
```

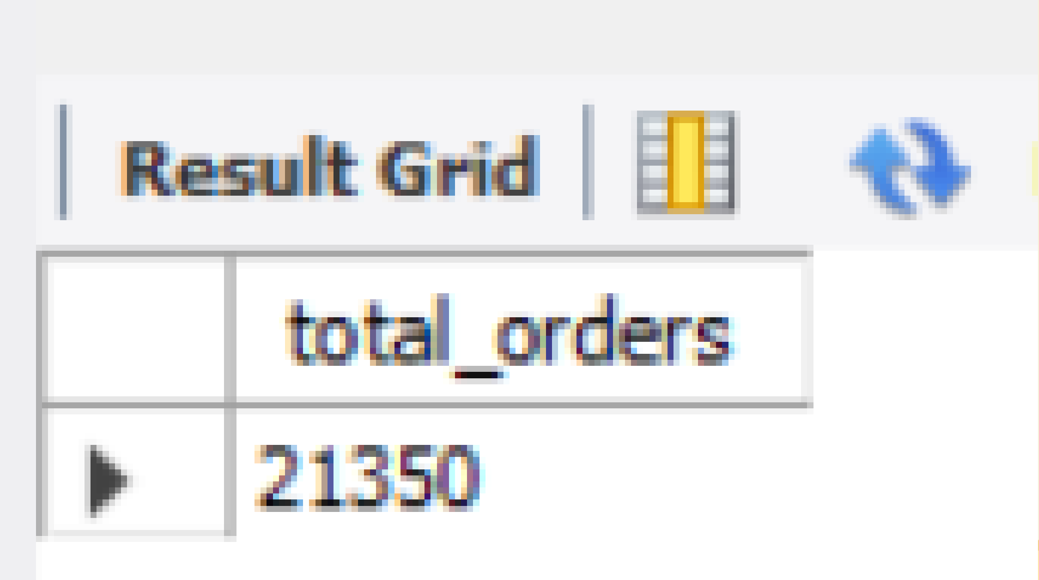
Field	Type	Null	Key	Default	Extra
pizza_type_id	text	YES		NULL	
name	text	YES		NULL	
category	text	YES		NULL	
ingredients	text	YES		NULL	

BASIC QUESTIONS

Retrieve the total number of orders placed.

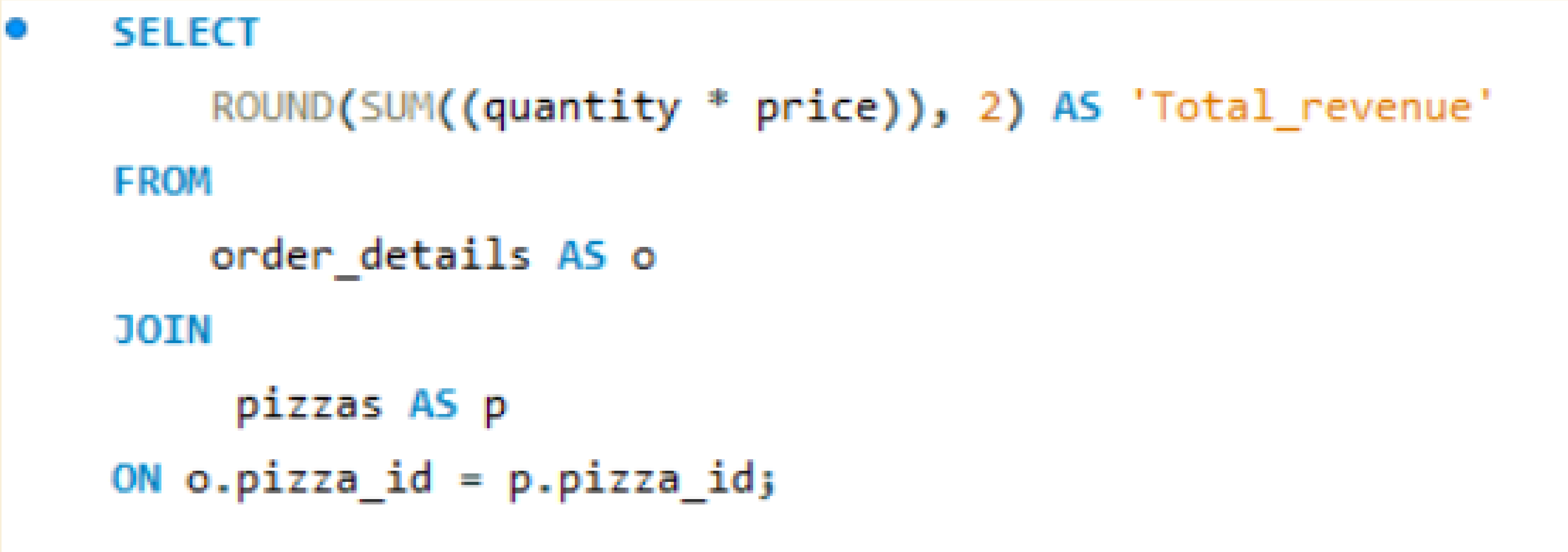


```
select count(*)  as 'Total_orders'
from orders;
```

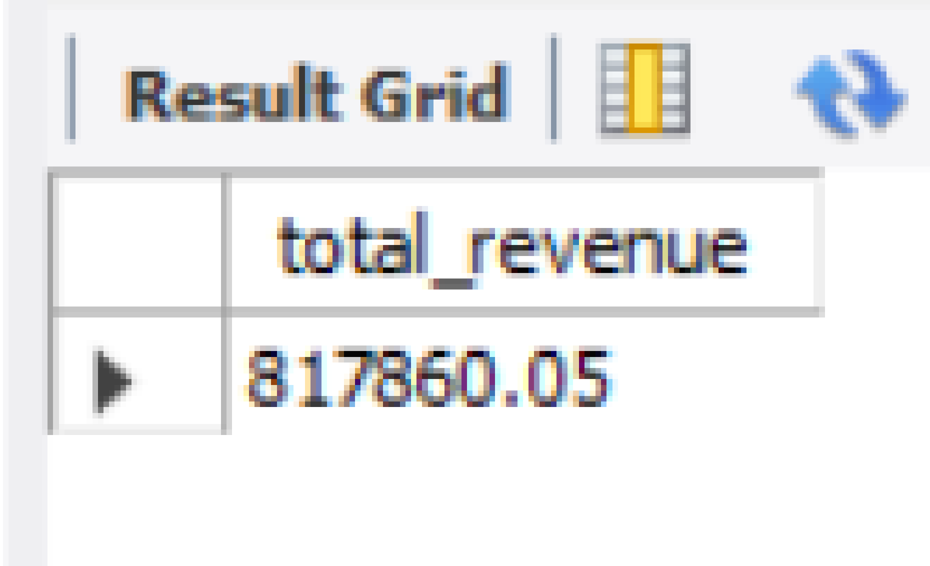


```
total_orders
21350
```

Calculate the total revenue generated from pizza sales.

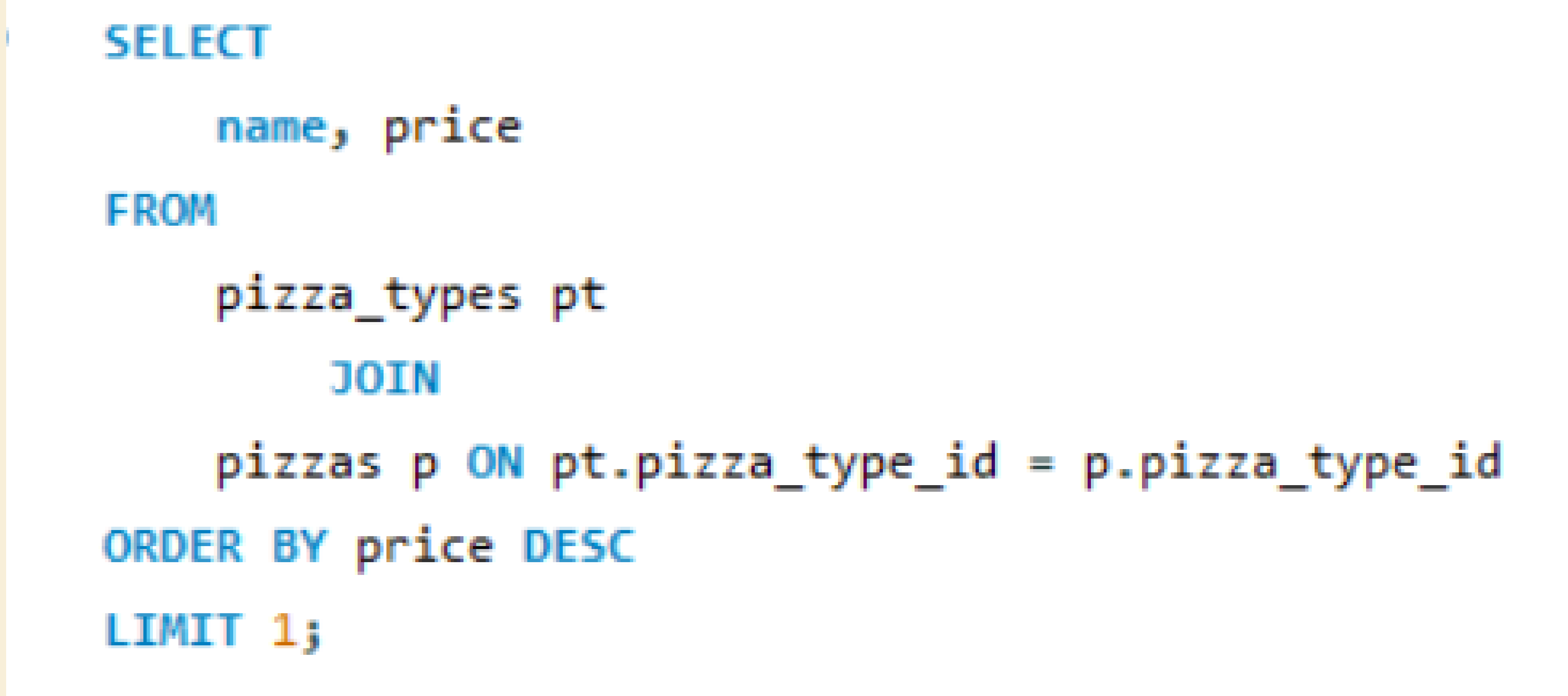


```
SELECT
    ROUND(SUM((quantity * price)), 2) AS 'Total_revenue'
FROM
    order_details AS o
JOIN
     pizzas AS p
ON o.pizza_id = p.pizza_id;
```

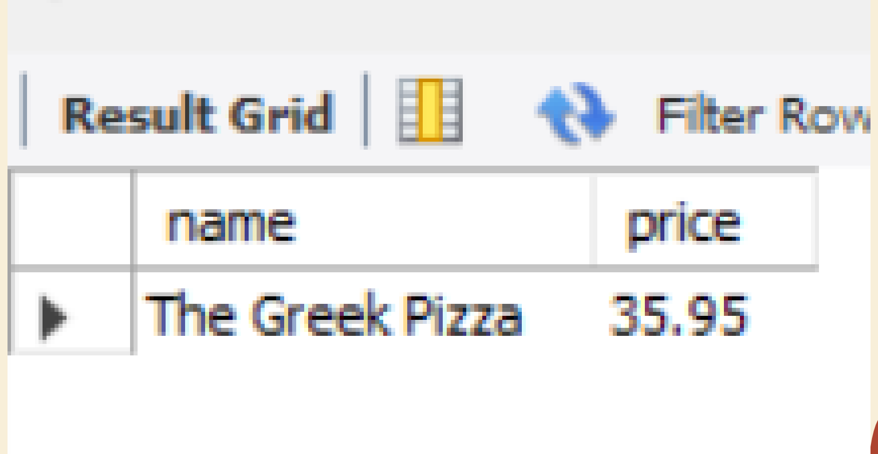


```
   total_revenue
▶  817860.05
```

Identify the highest-priced pizza.

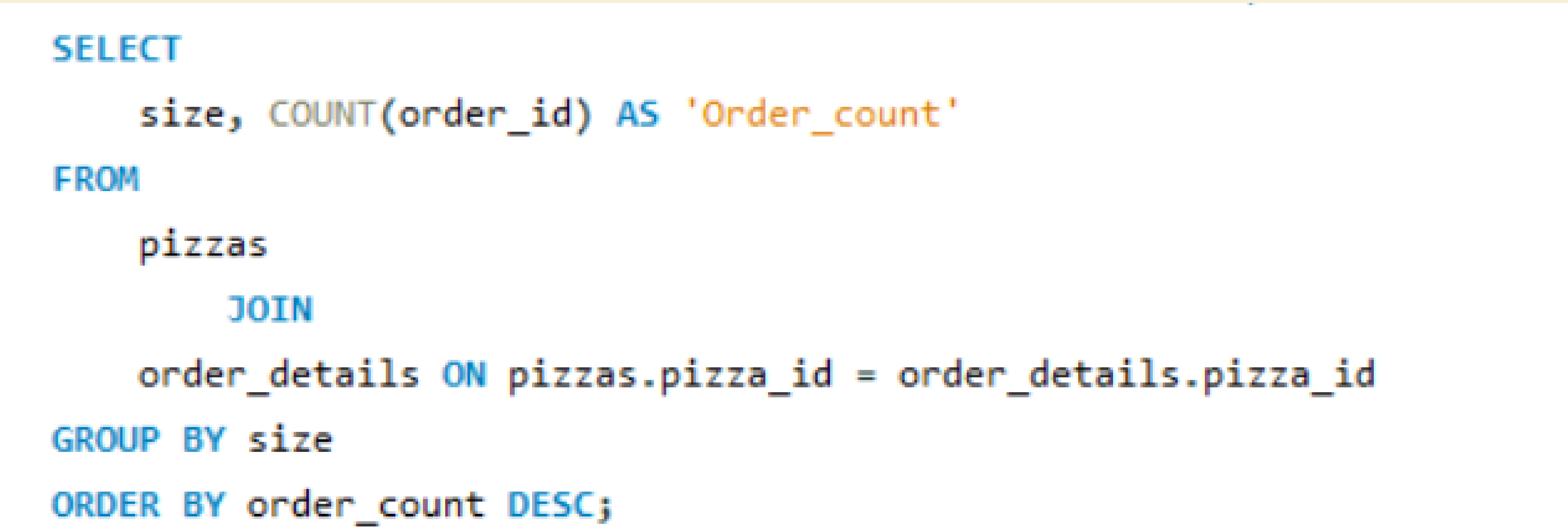


```
SELECT
    name, price
FROM
    pizza_types pt
        JOIN
    pizzas p ON pt.pizza_type_id = p.pizza_type_id
ORDER BY price DESC
LIMIT 1;
```

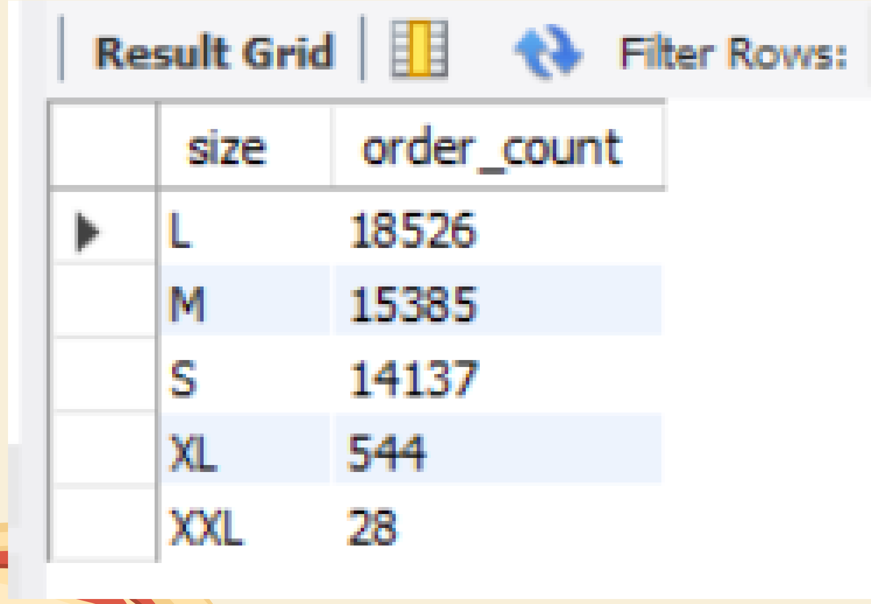


name	price
The Greek Pizza	35.95

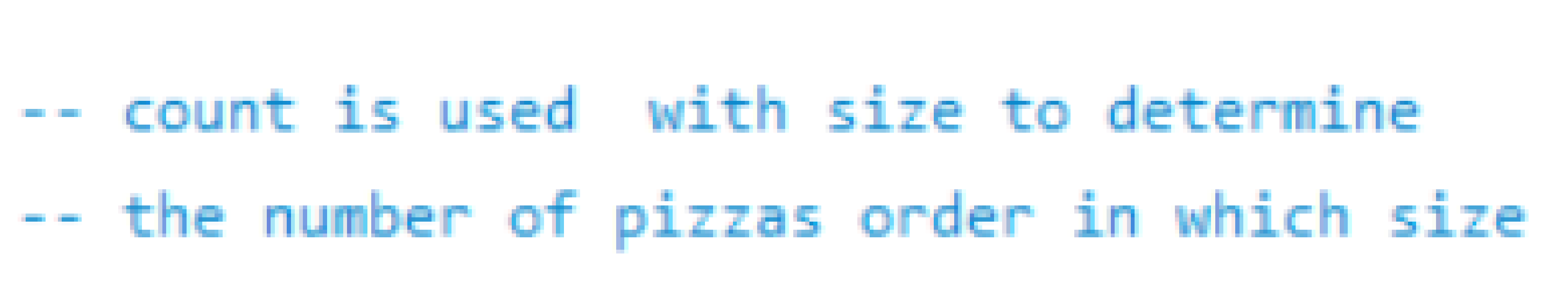
Identify the most common pizza size ordered.



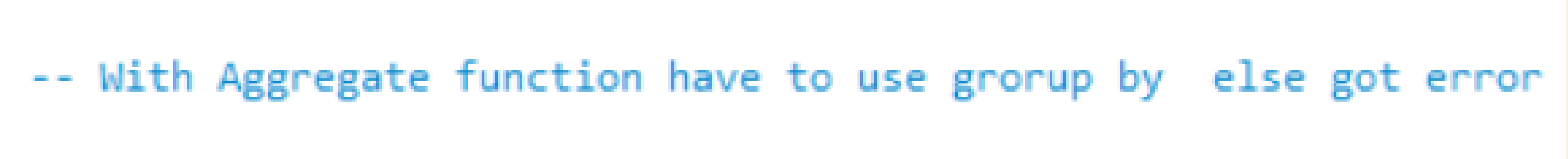
```
SELECT
    size, COUNT(order_id) AS 'Order_count'
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY size
ORDER BY order_count DESC;
```



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

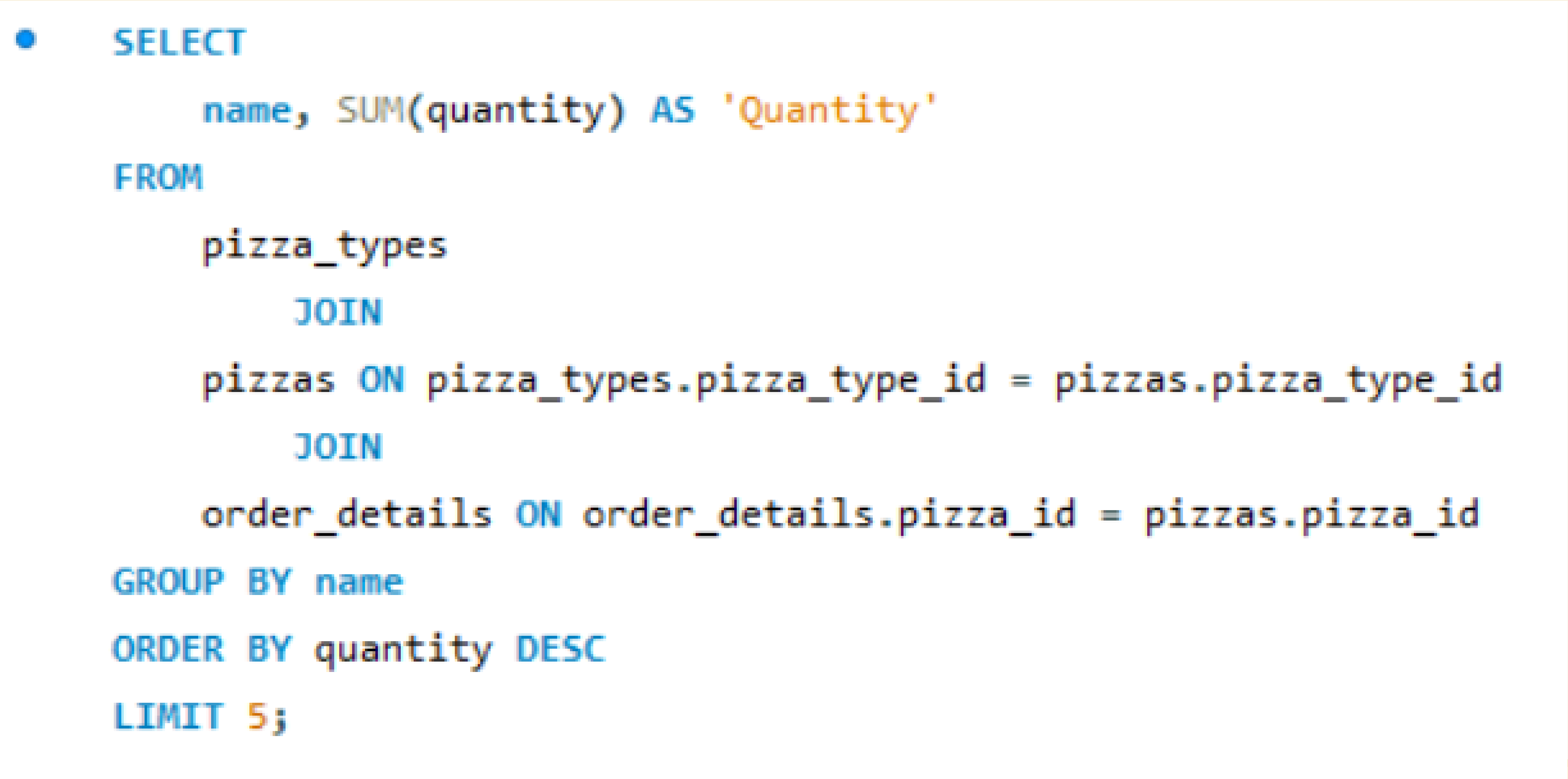


```
-- count is used  with size to determine
-- the number of pizzas order in which size
```

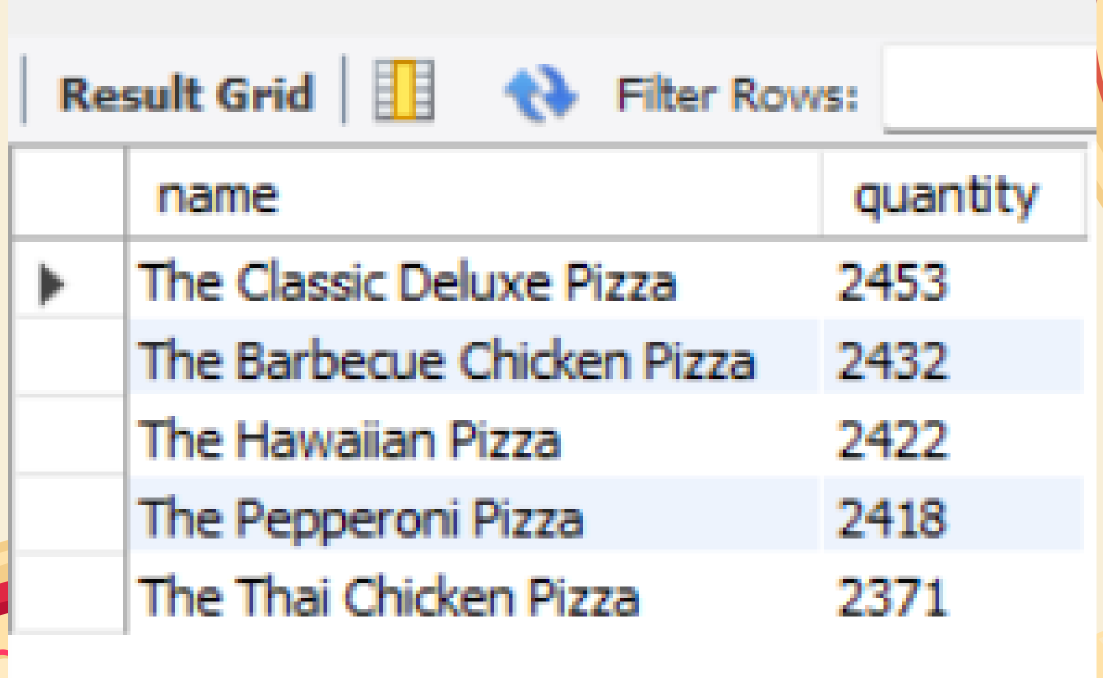


```
-- With Aggregate function have to use grorup by  else got error
```

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



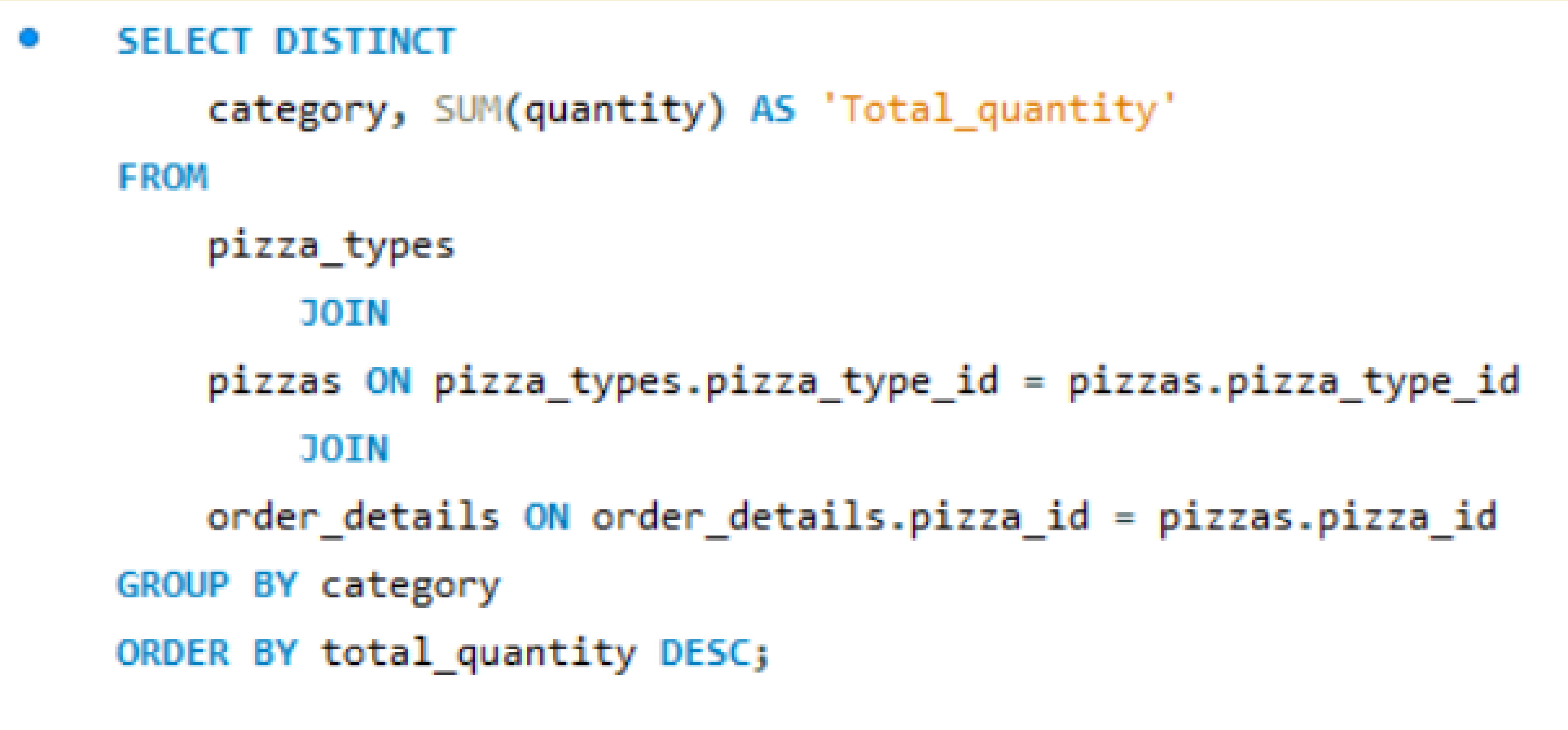
```
SELECT
    name, SUM(quantity) AS 'Quantity'
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY name
ORDER BY quantity DESC
LIMIT 5;
```



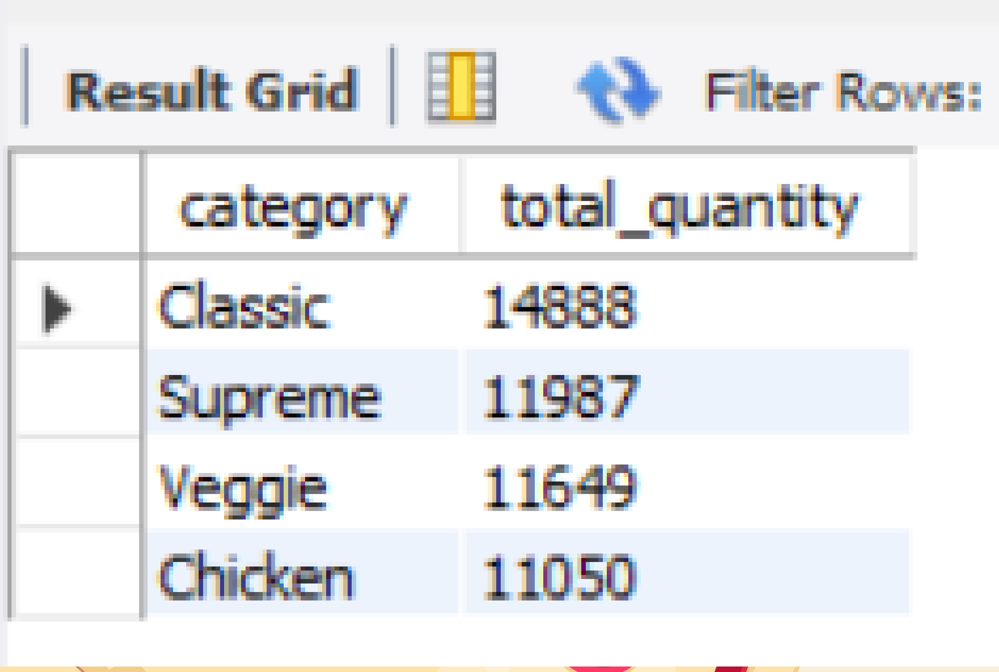
name	quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

INTERMEDIATE QUESTIONS

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

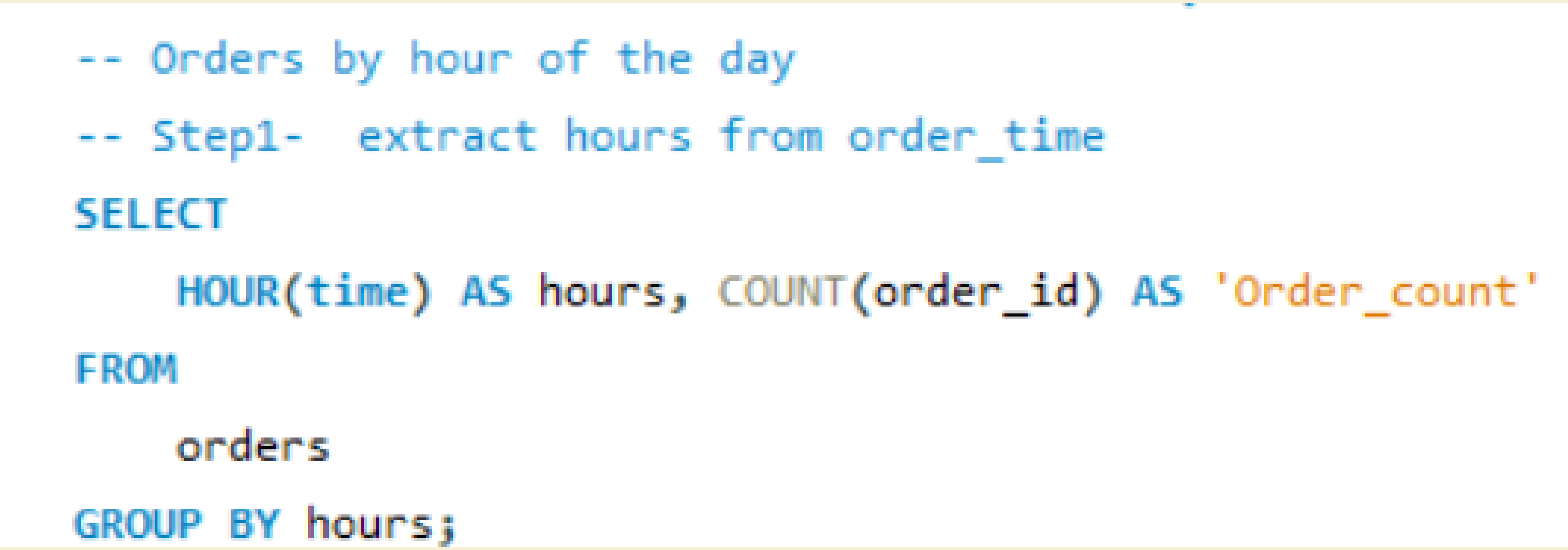


```
SELECT DISTINCT
    category, SUM(quantity) AS 'Total_quantity'
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY category
ORDER BY total_quantity DESC;
```

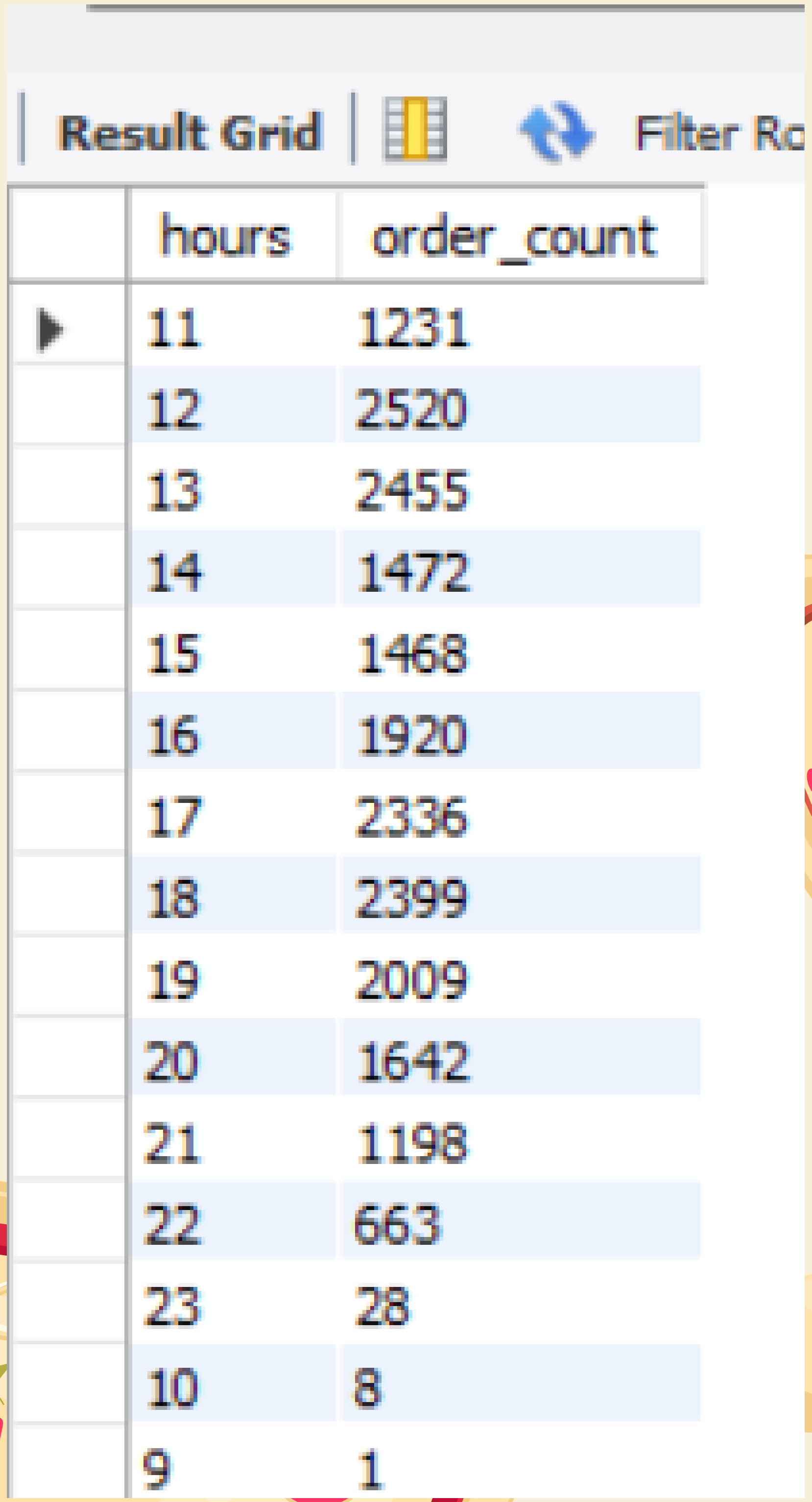


category	total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Determine the distribution of orders by hour of the day.

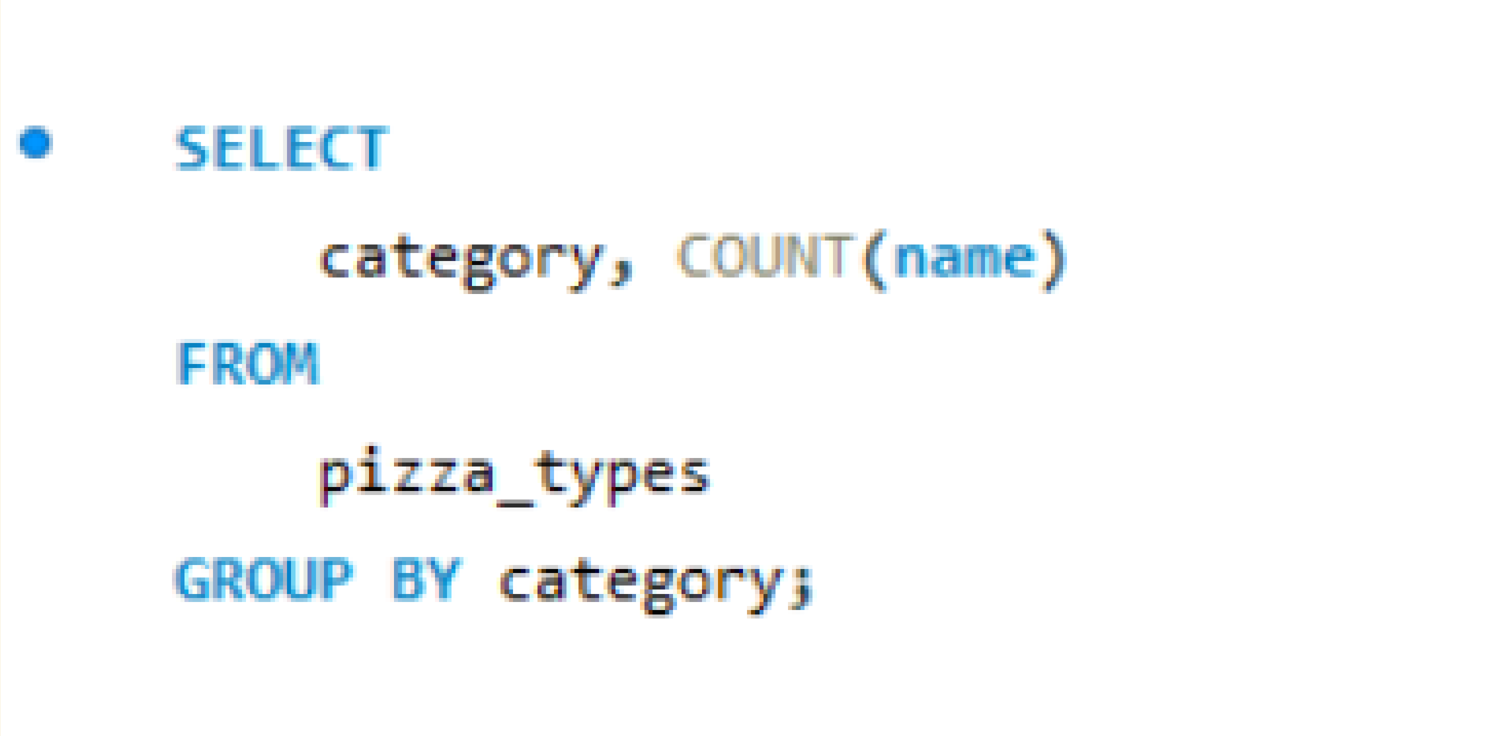


```
-- Orders by hour of the day
-- Step1-  extract hours from order_time
SELECT
    HOUR(time) AS hours, COUNT(order_id) AS 'Order_count'
FROM
    orders
GROUP BY hours;
```

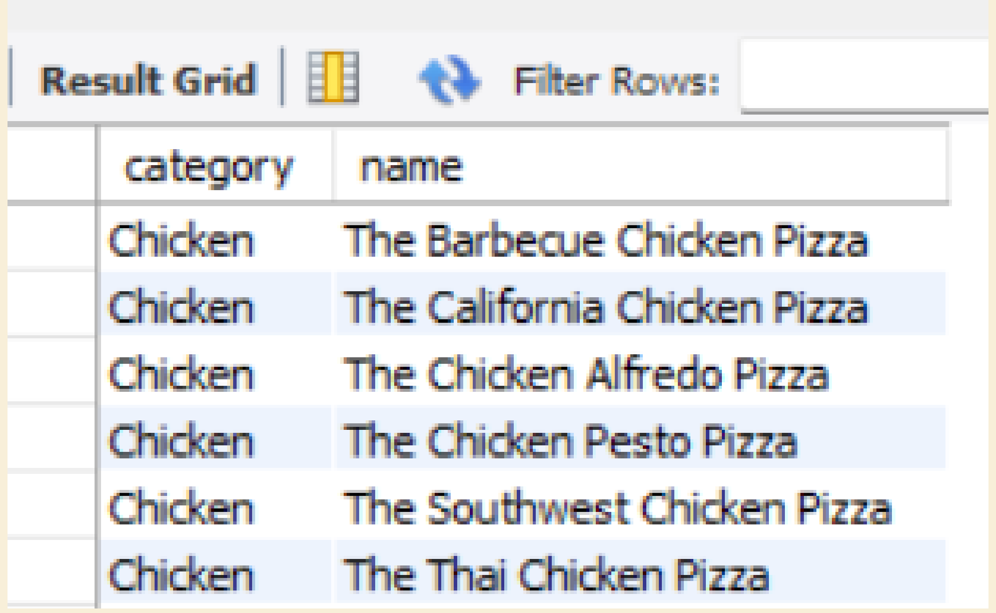


hours	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

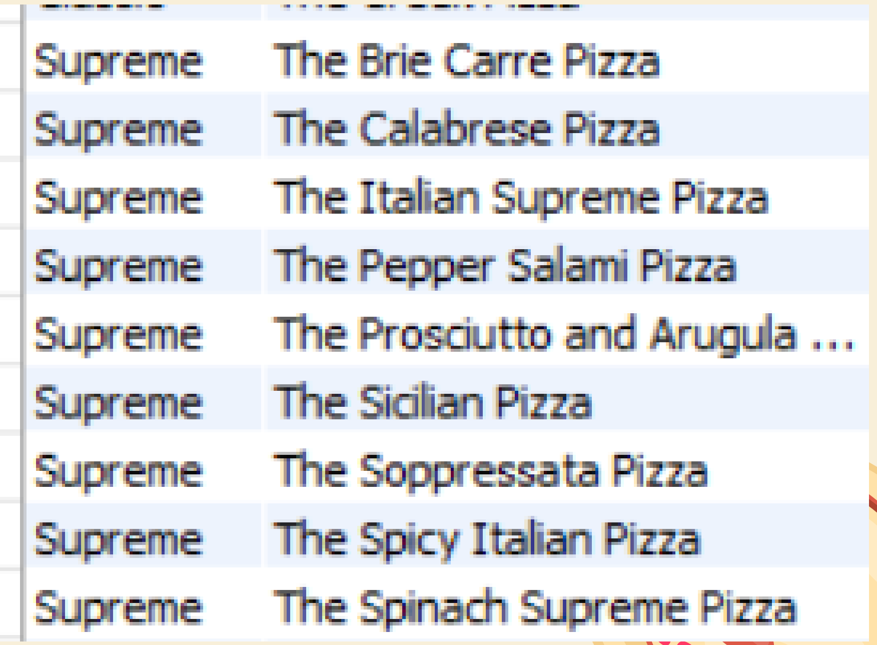
Join relevant tables to find the category-wise distribution of pizzas.



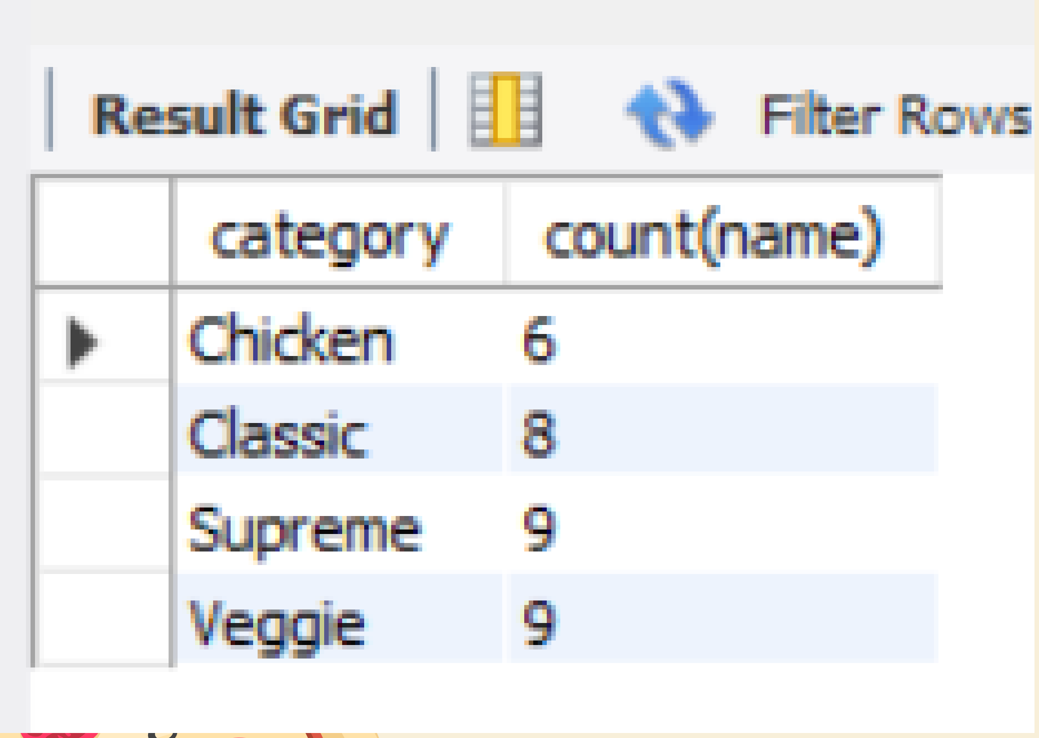
```
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```



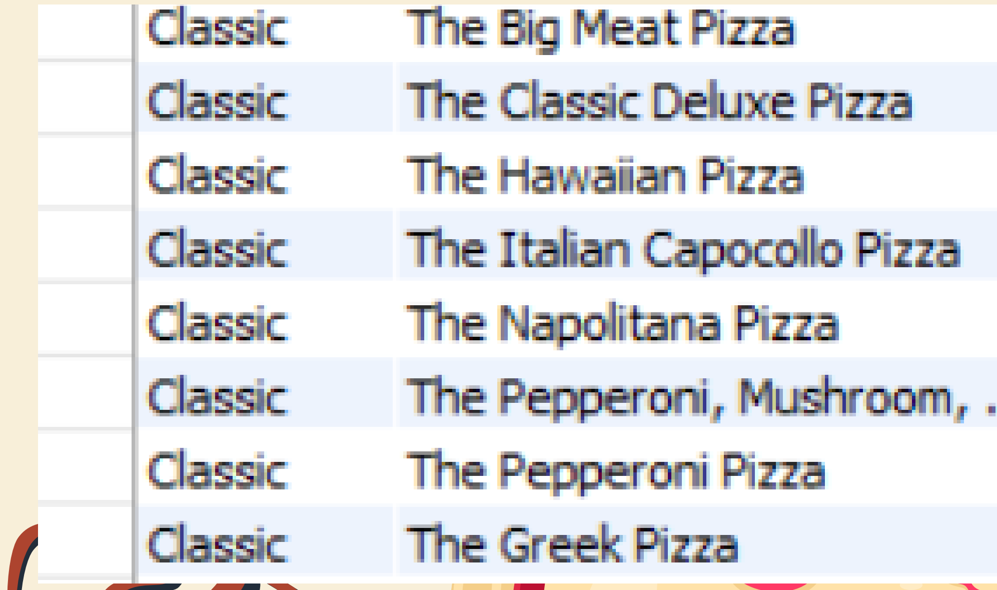
category	name
Chicken	The Barbecue Chicken Pizza
Chicken	The California Chicken Pizza
Chicken	The Chicken Alfredo Pizza
Chicken	The Chicken Pesto Pizza
Chicken	The Southwest Chicken Pizza
Chicken	The Thai Chicken Pizza



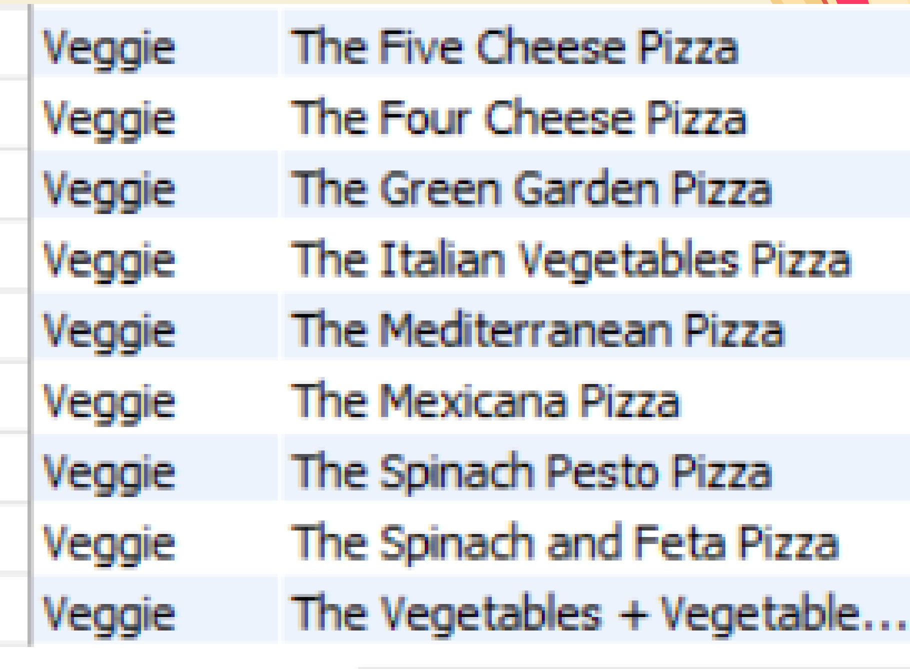
Supreme	The Brie Carre Pizza
Supreme	The Calabrese Pizza
Supreme	The Italian Supreme Pizza
Supreme	The Pepper Salami Pizza
Supreme	The Prosciutto and Arugula ...
Supreme	The Sicilian Pizza
Supreme	The Soppressata Pizza
Supreme	The Spicy Italian Pizza
Supreme	The Spinach Supreme Pizza



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

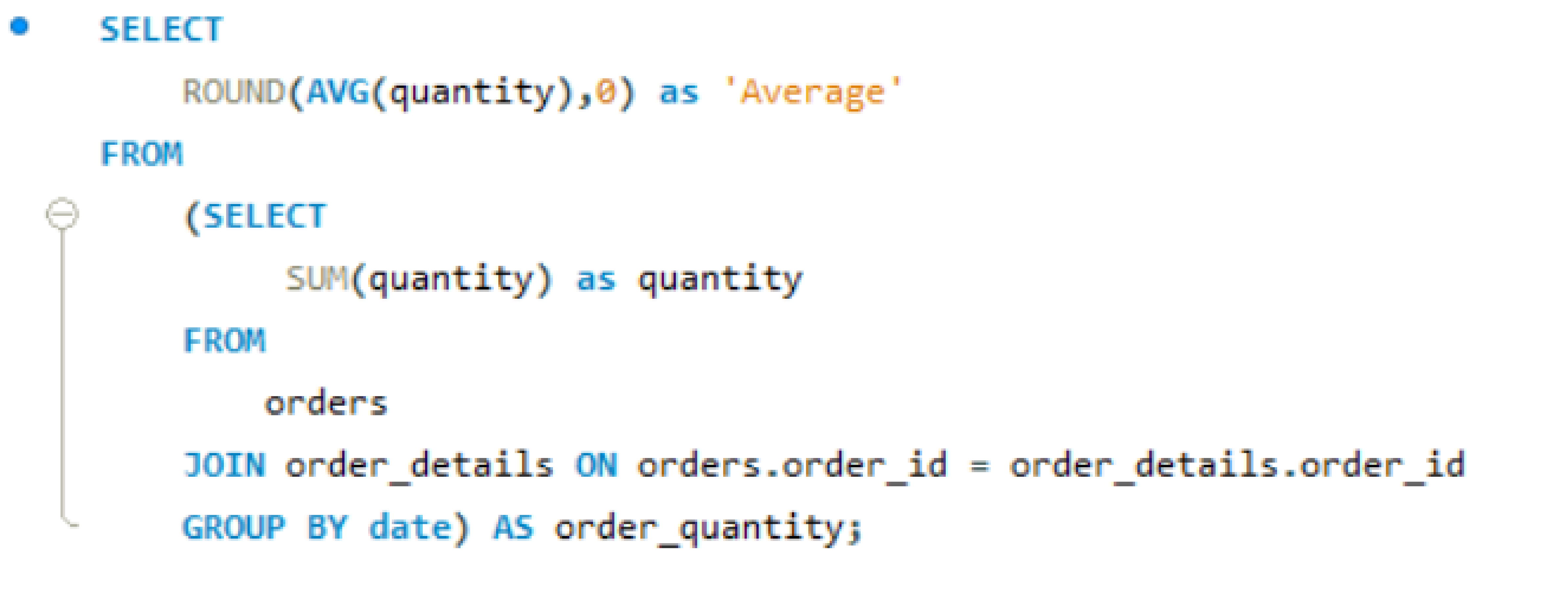


Classic	The Big Meat Pizza
Classic	The Classic Deluxe Pizza
Classic	The Hawaiian Pizza
Classic	The Italian Capocollo Pizza
Classic	The Napolitana Pizza
Classic	The Pepperoni, Mushroom, .
Classic	The Pepperoni Pizza
Classic	The Greek Pizza

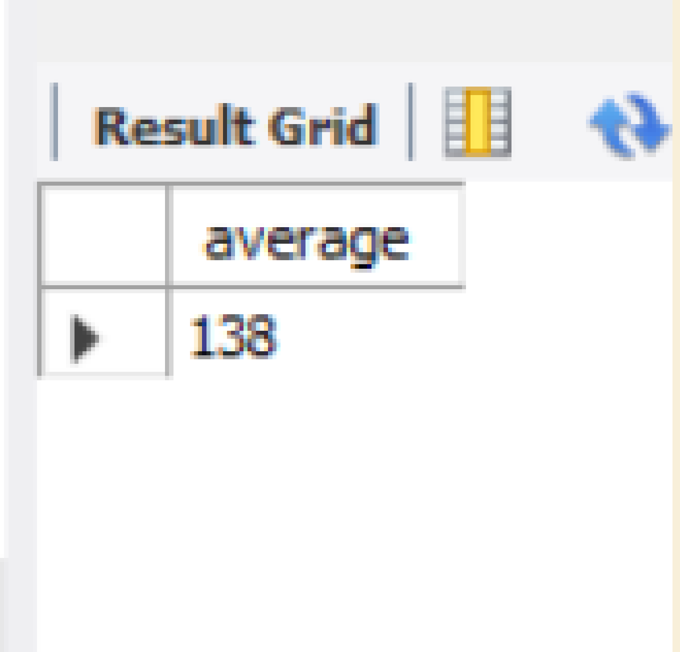


Veggie	The Five Cheese Pizza
Veggie	The Four Cheese Pizza
Veggie	The Green Garden Pizza
Veggie	The Italian Vegetables Pizza
Veggie	The Mediterranean Pizza
Veggie	The Mexicana Pizza
Veggie	The Spinach Pesto Pizza
Veggie	The Spinach and Feta Pizza
Veggie	The Vegetables + Vegetable...

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

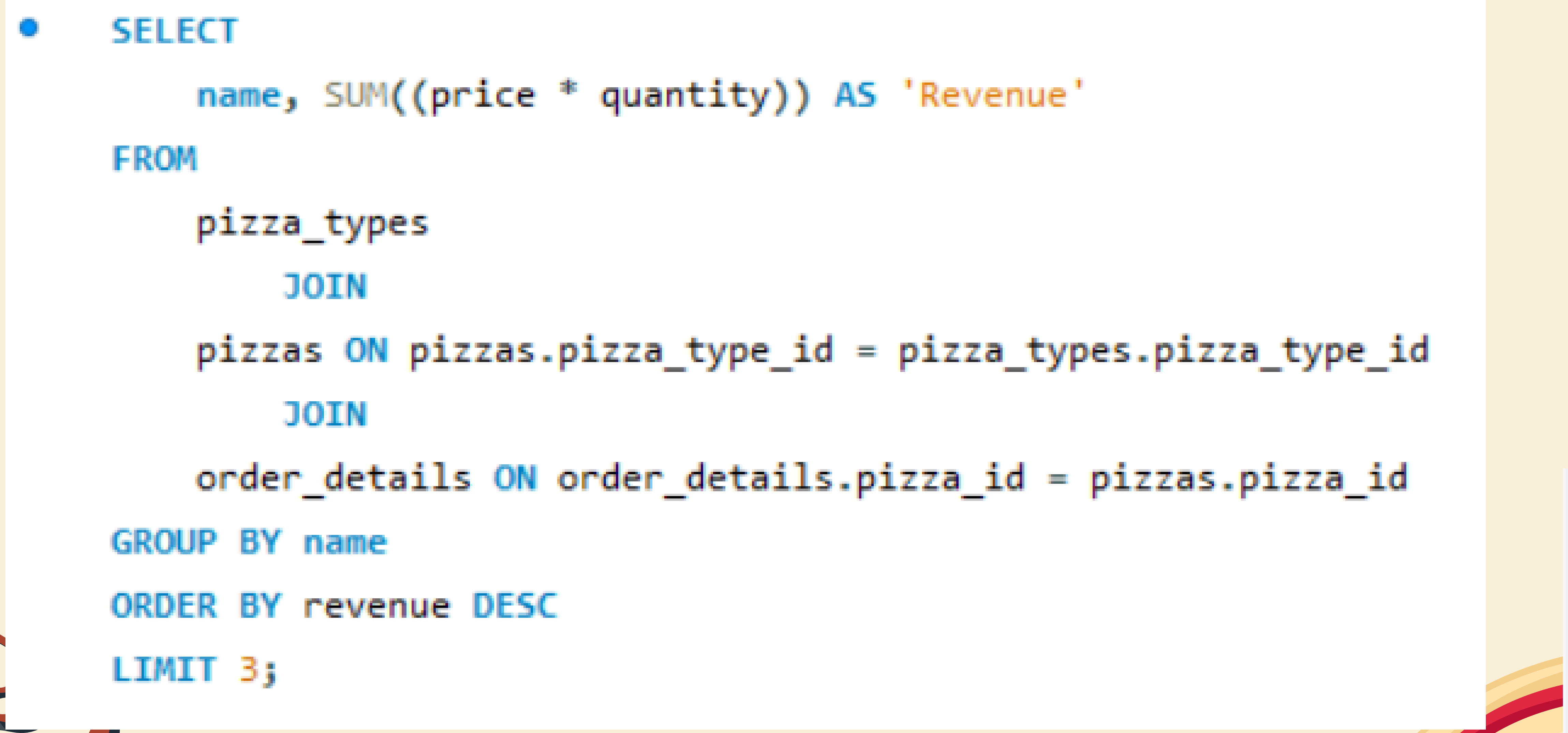


```sql
SELECT
    ROUND(AVG(quantity),0) as 'Average'
FROM
    (SELECT
        SUM(quantity) as quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY date) AS order_quantity;
```

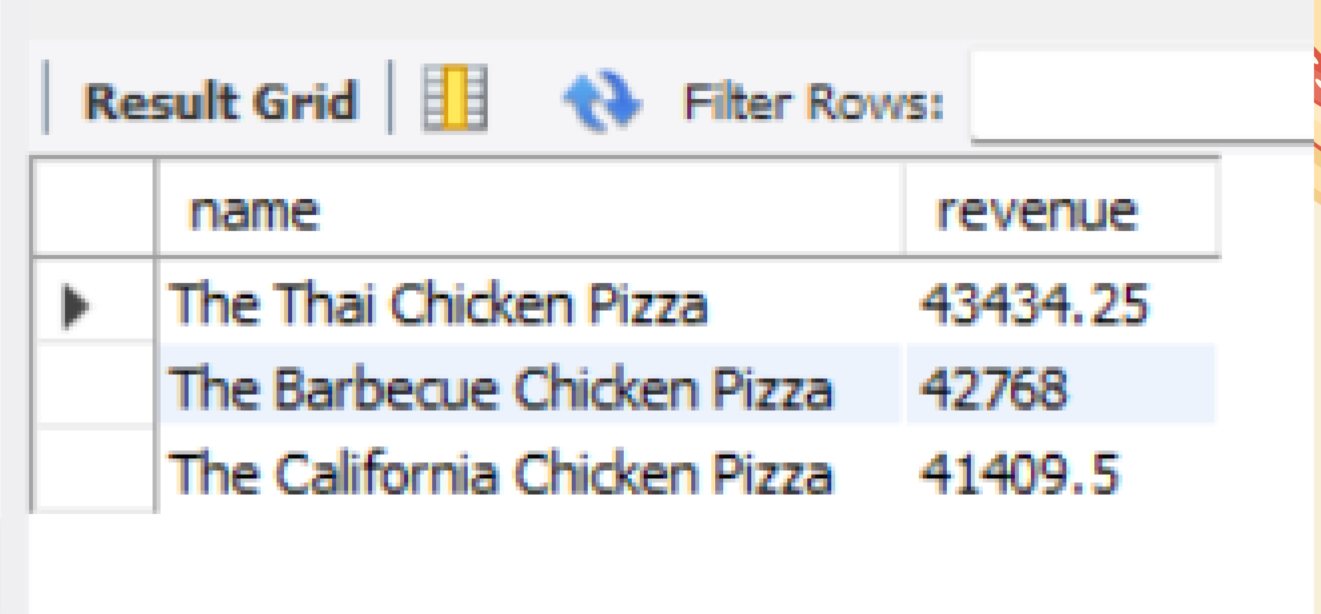


```
average
138
```

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



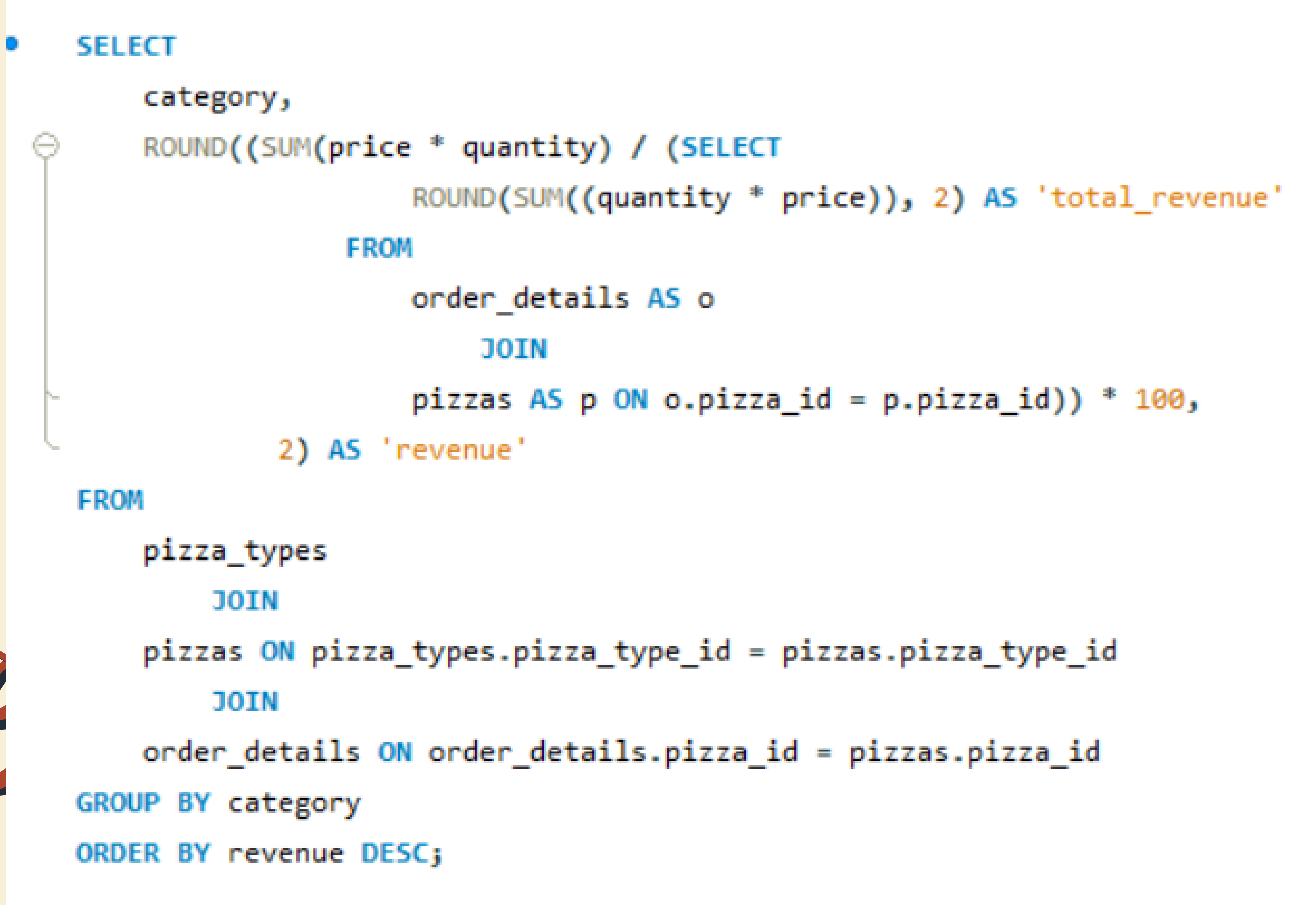
```
SELECT
    name, SUM((price * quantity)) AS 'Revenue'
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY name
ORDER BY revenue DESC
LIMIT 3;
```



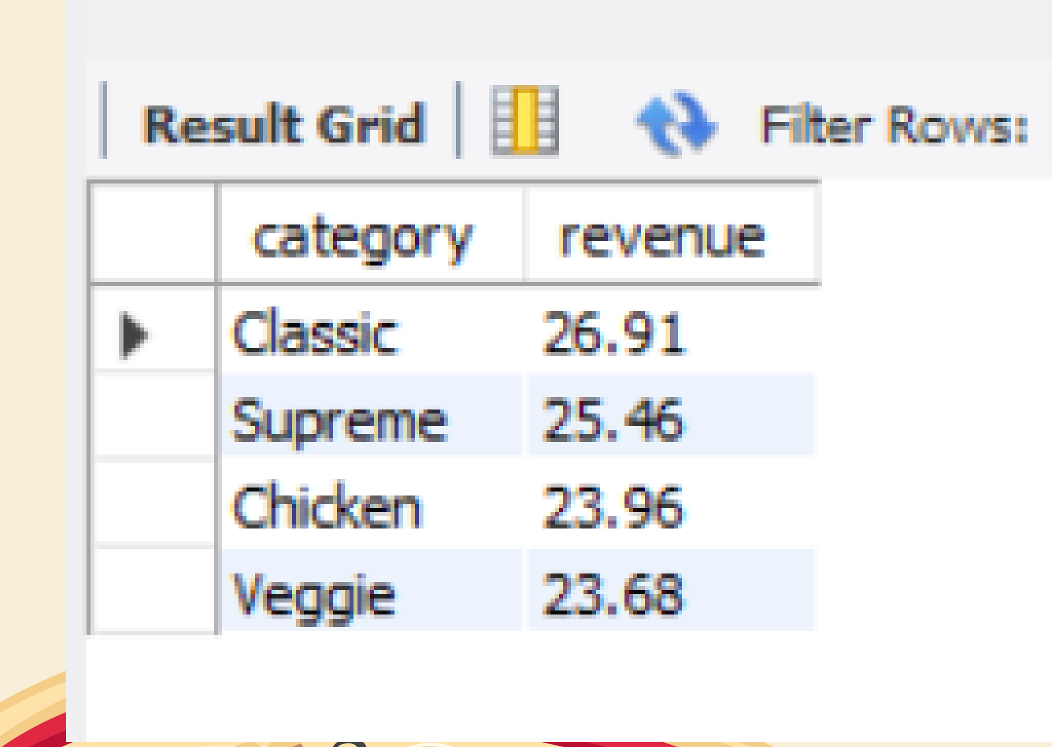
name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

ADVANCE QUESTIONS

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

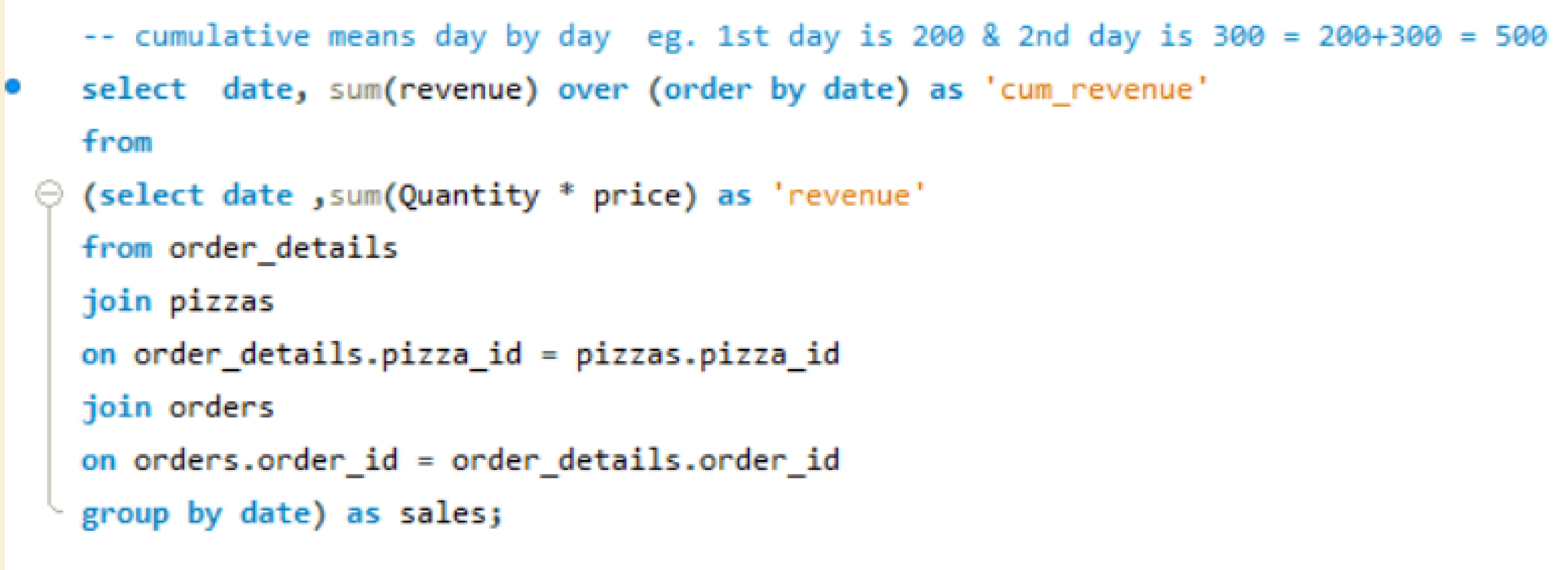


```
SELECT
    category,
    ROUND((SUM(price * quantity) / (SELECT
                    ROUND(SUM((quantity * price)), 2) AS 'total_revenue'
                FROM
                    order_details AS o
                        JOIN
                    pizzas AS p ON o.pizza_id = p.pizza_id)) * 100,
            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 category
ORDER BY revenue DESC;
```

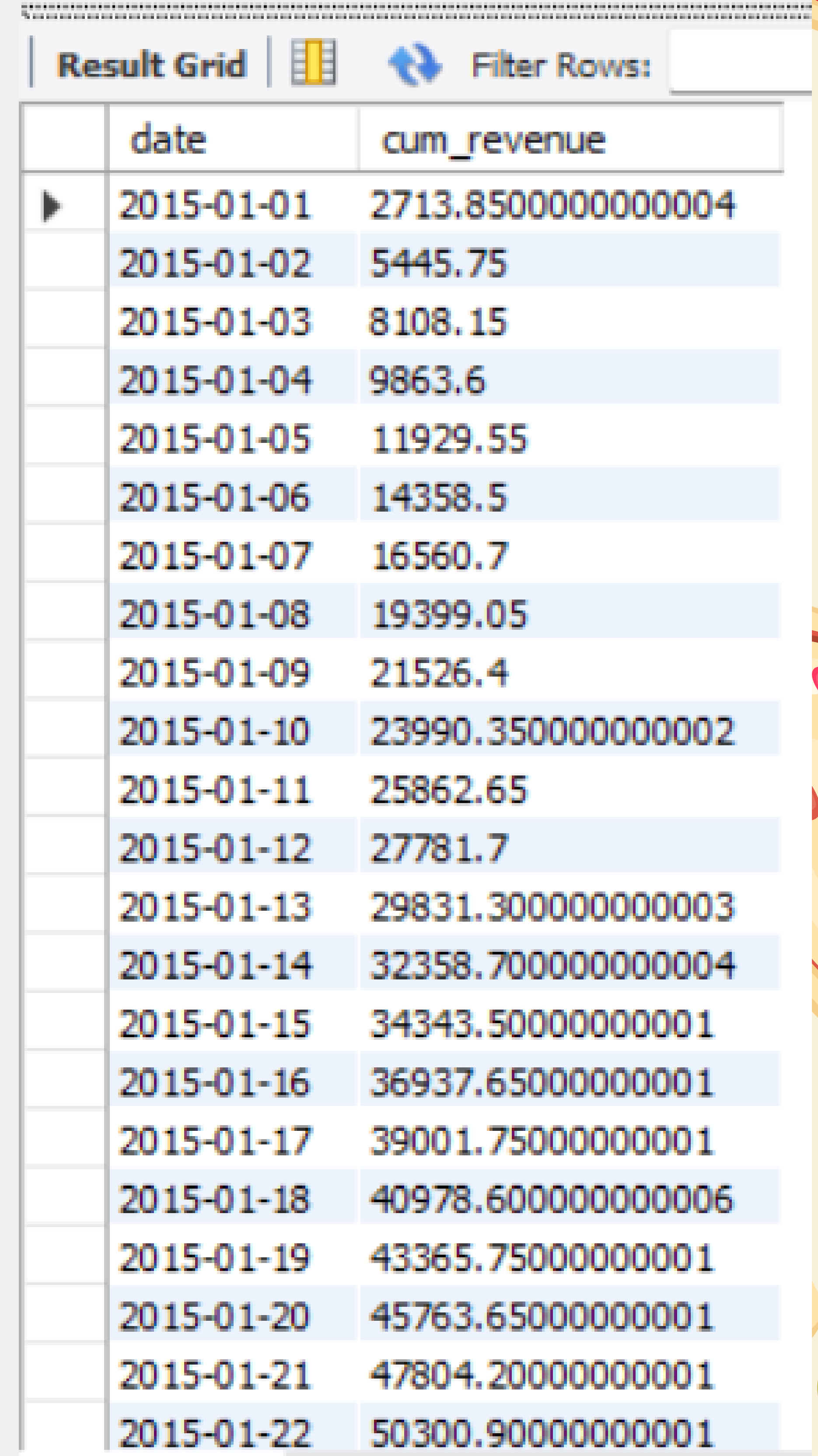


category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Analyze the cumulative revenue generated over time.

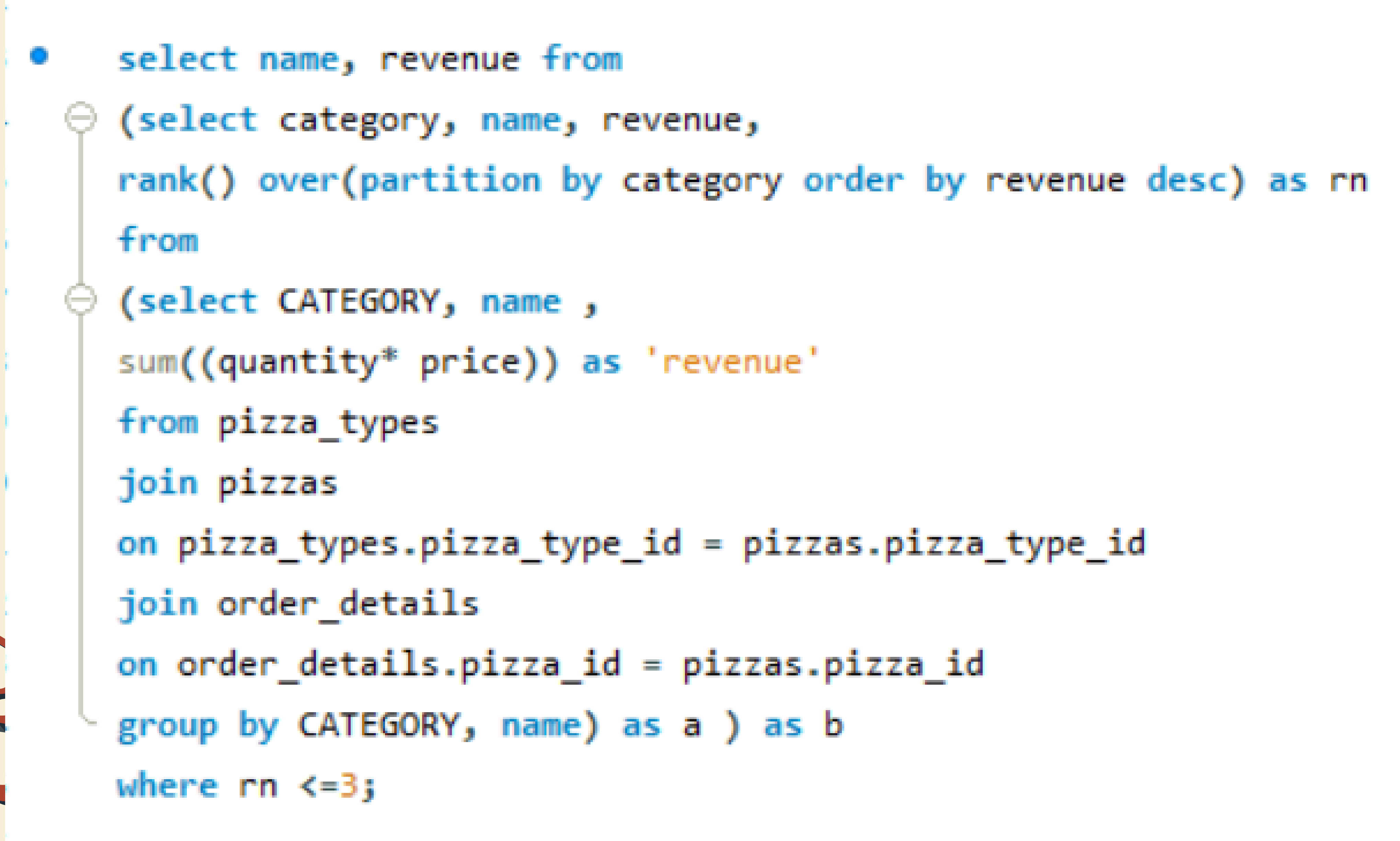


```
-- cumulative means day by day  eg. 1st day is 200 & 2nd day is 300 = 200+300 = 500
select  date, sum(revenue) over (order by date) as 'cum_revenue'
from
(select date ,sum(Quantity * price) as 'revenue'
from order_details
join pizzas
on order_details.pizza_id = pizzas.pizza_id
join orders
on orders.order_id = order_details.order_id
group by date) as sales;
```

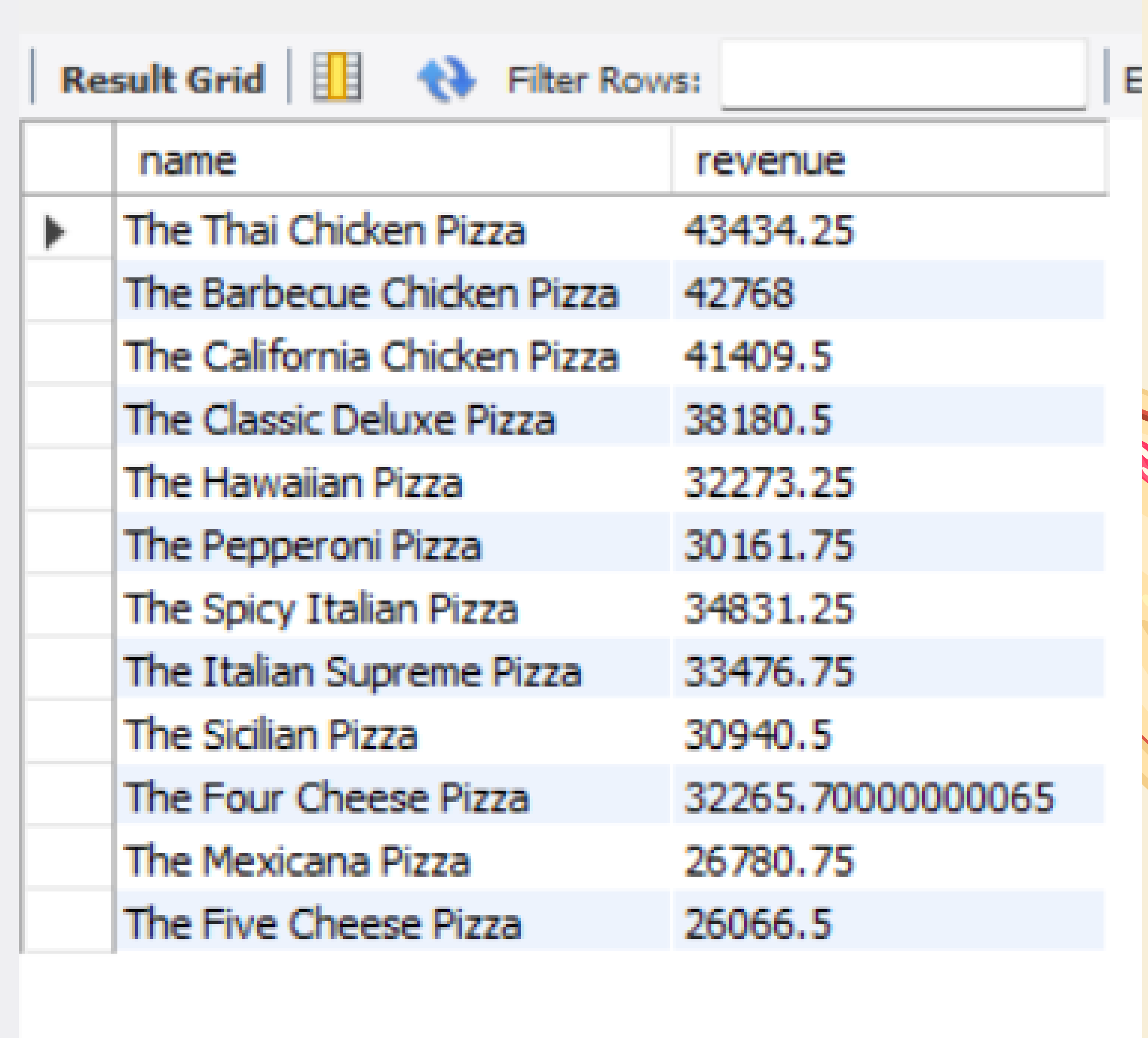


date	cum_revenue
2015-01-01	2713.8500000000004
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55
2015-01-06	14358.5
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526.4
2015-01-10	23990.350000000002
2015-01-11	25862.65
2015-01-12	27781.7
2015-01-13	29831.300000000003
2015-01-14	32358.700000000004
2015-01-15	34343.50000000001
2015-01-16	36937.65000000001
2015-01-17	39001.75000000001
2015-01-18	40978.600000000006
2015-01-19	43365.75000000001
2015-01-20	45763.65000000001
2015-01-21	47804.20000000001
2015-01-22	50300.90000000001

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 CATEGORY, name ,
sum((quantity* 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 CATEGORY, 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

WHAT pizza sales status?

count of pizza_sale

23150

Total_r evenue

817860.05

avg no of pizza orderd

138

AWESOME SQL QUESTIONS COVERD

23150

COUNT OF TOTAL PIZZA SALES

THANKS!

Do you have any questions?

kavitakamat90@gmail.com