One slice is never enough

SQL PROJECT ON PIZZA SALES

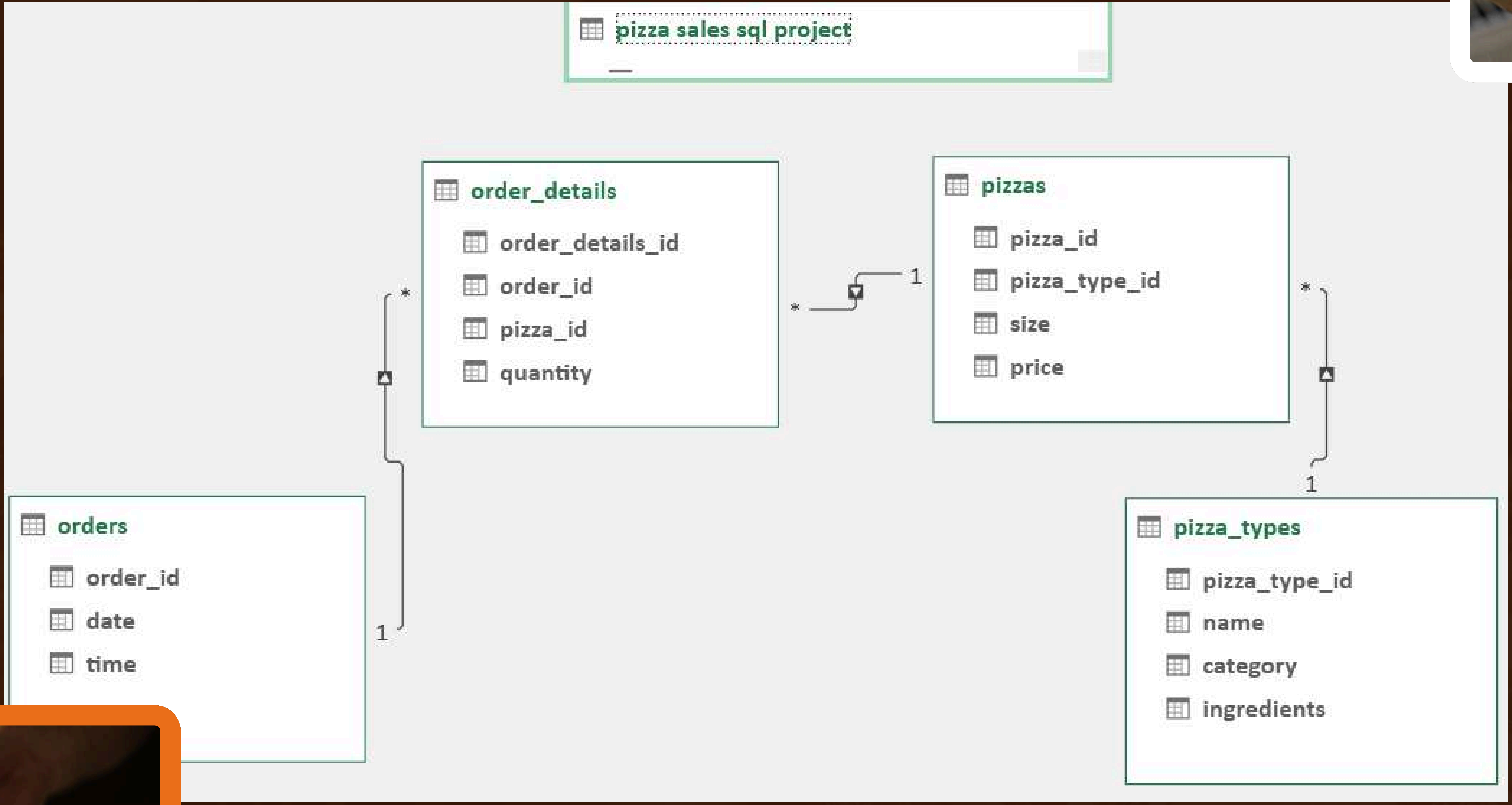
ORDER NOW

@firdousrahmani

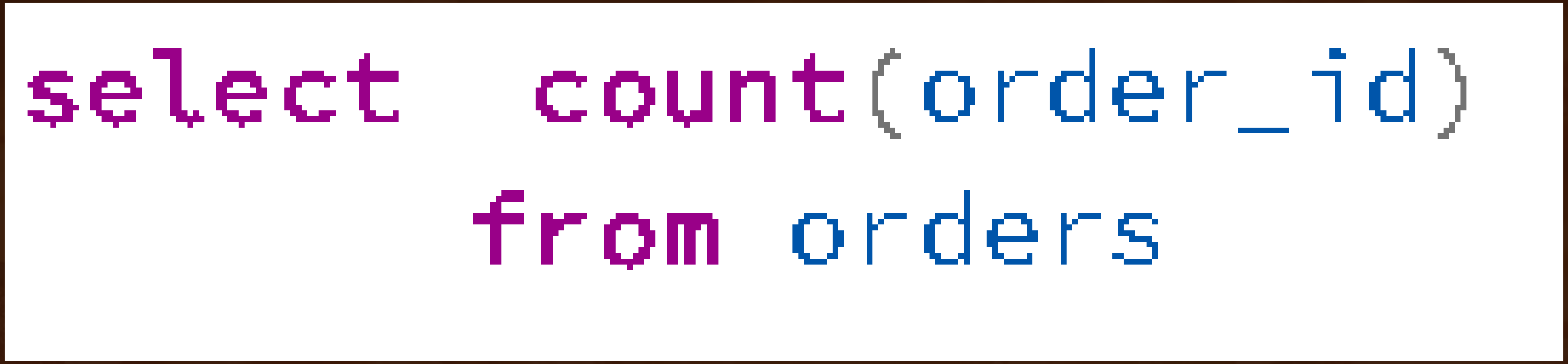
HELLO!

My name is Firdous Rahmani. In this project, I have used SQL queries to analyze and solve key business challenges related to pizza sales. The focus of this analysis is to extract meaningful insights from sales data, identify trends, and support data-driven decision-making. Through structured queries, I have explored aspects such as sales performance, customer preferences, and operational efficiency which provides actionable insights for business growth.

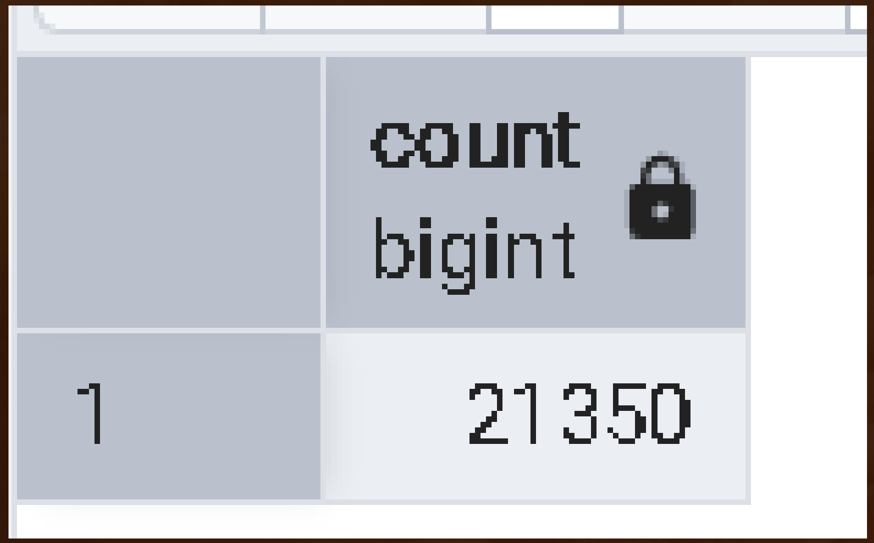
DATABASE SCHEMA



RETRIEVE THE TOTLA NUMBER OF ORDER PLACED ?

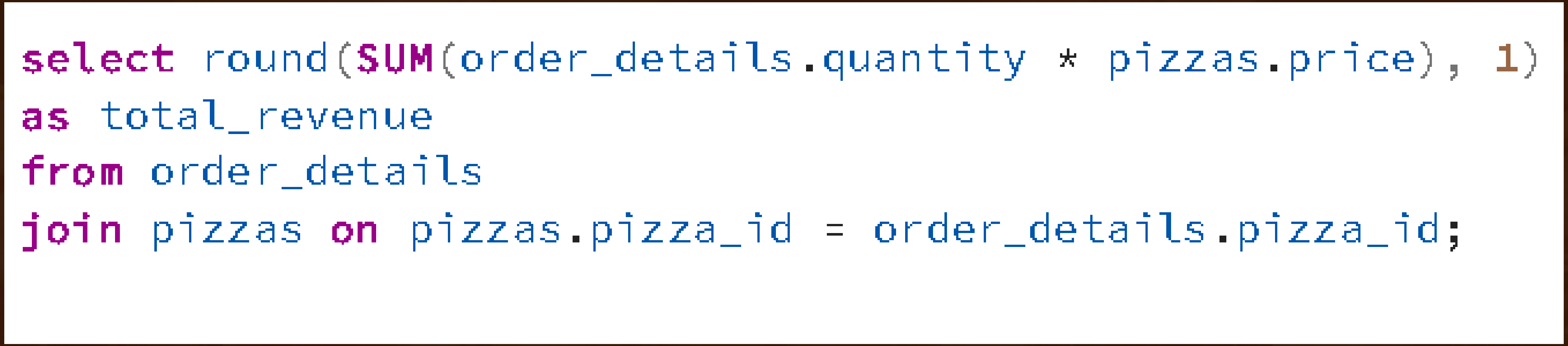


```
select  count(order_id)
    from orders
```

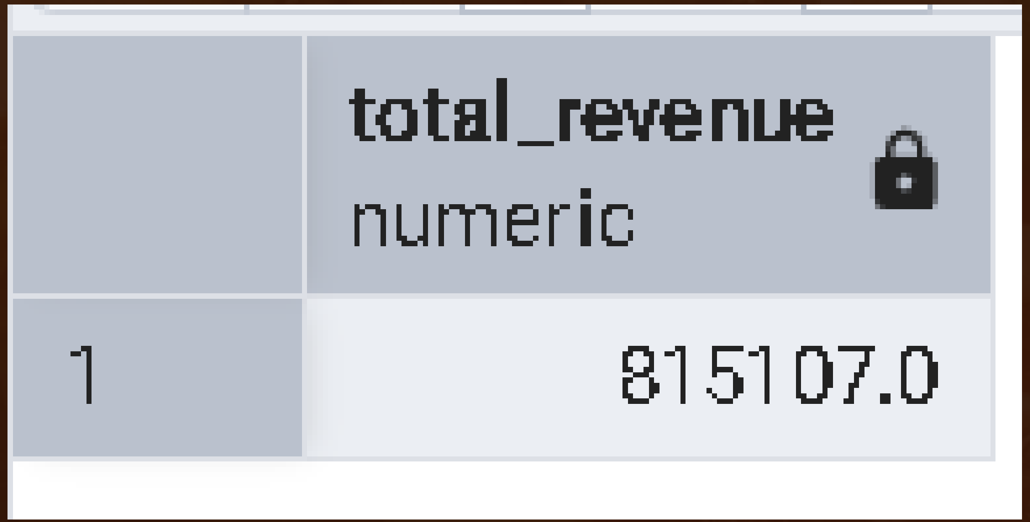


```
     count
     bigint
1    21350
```

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES?



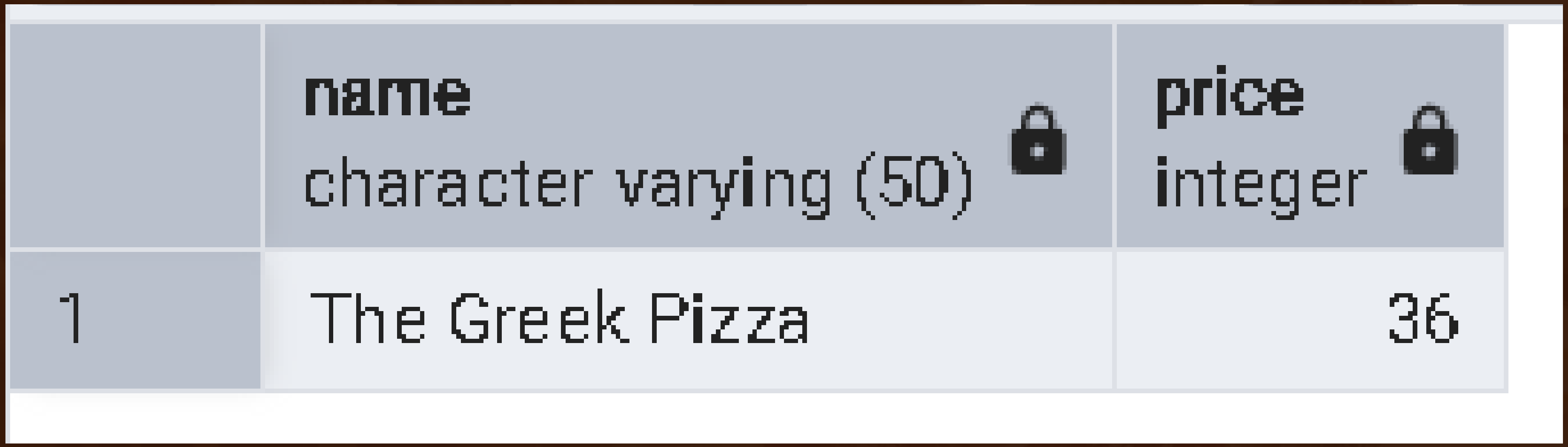
```sql
select round(SUM(order_details.quantity * pizzas.price), 1)
as total_revenue
from order_details
join pizzas on pizzas.pizza_id = order_details.pizza_id;
```



	total_revenue numeric
1	815107.0

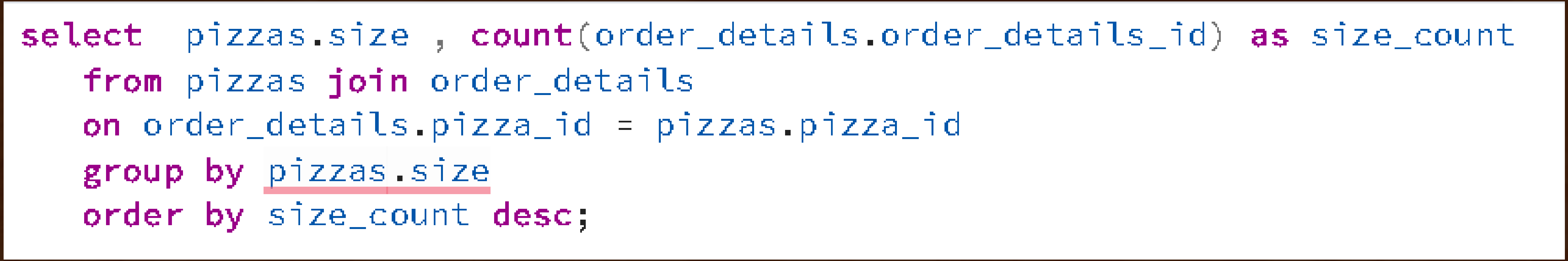
IDENTIFY THE HIGHEST PRICED PIZZA?

```sql
select pizza_types.name, pizzas.price
     from pizza_types
     join pizzas on pizza_types.pizza_type_id = pizzas.pizza_type_id
     order by price desc
     limit 1
```

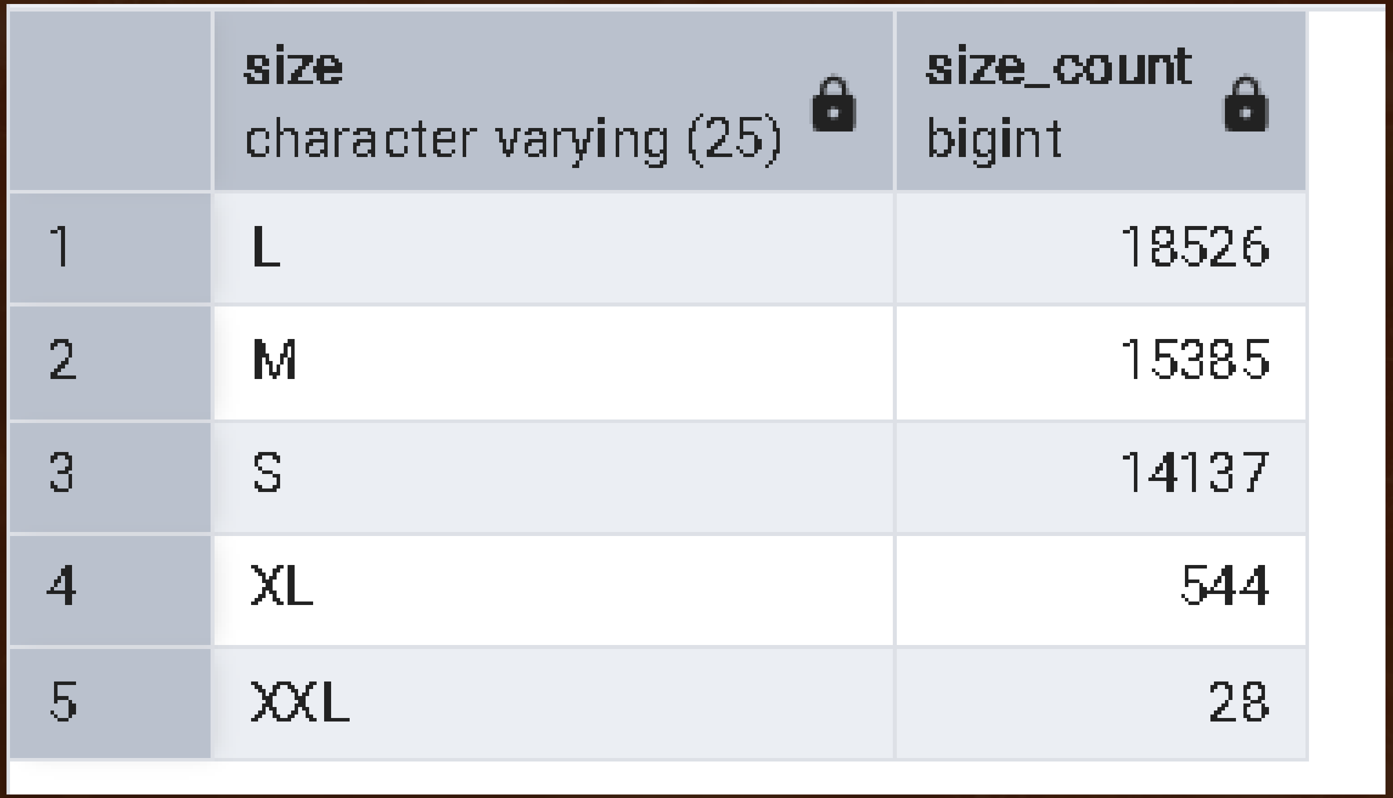


	name character varying (50)	price integer
1	The Greek Pizza	36

IDENTIFY THE MOST COMMON PIZZA SIZE THAT ORDERED?

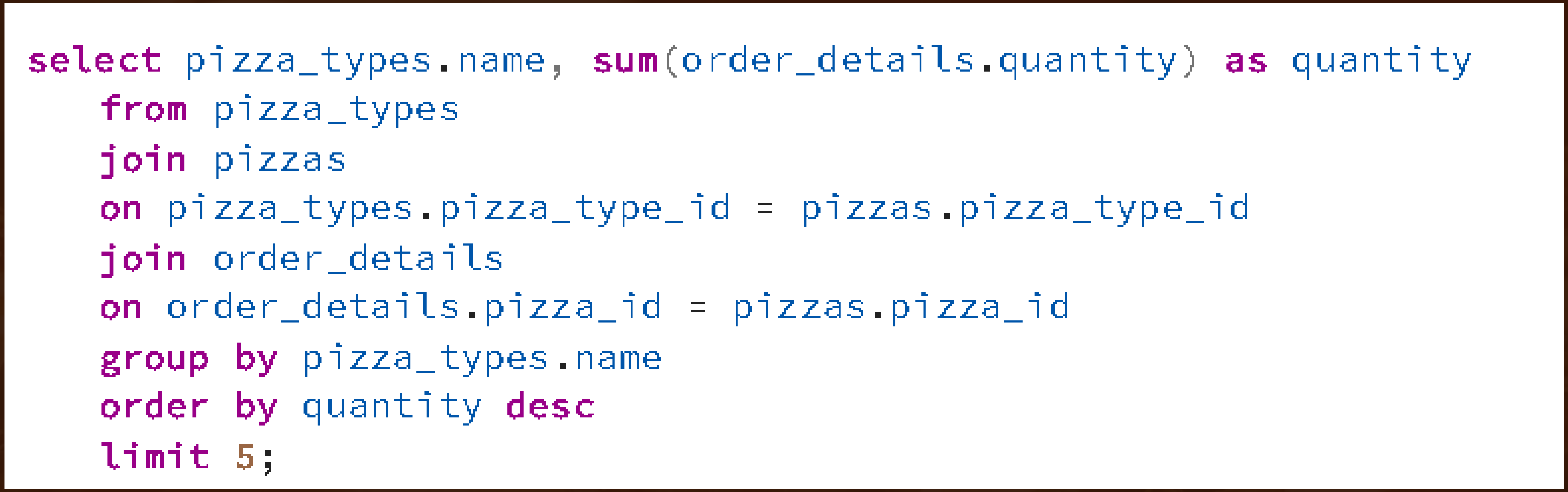


```
select  pizzas.size , count(order_details.order_details_id) as size_count
   from pizzas join order_details
   on order_details.pizza_id = pizzas.pizza_id
   group by pizzas.size
   order by size_count desc;
```

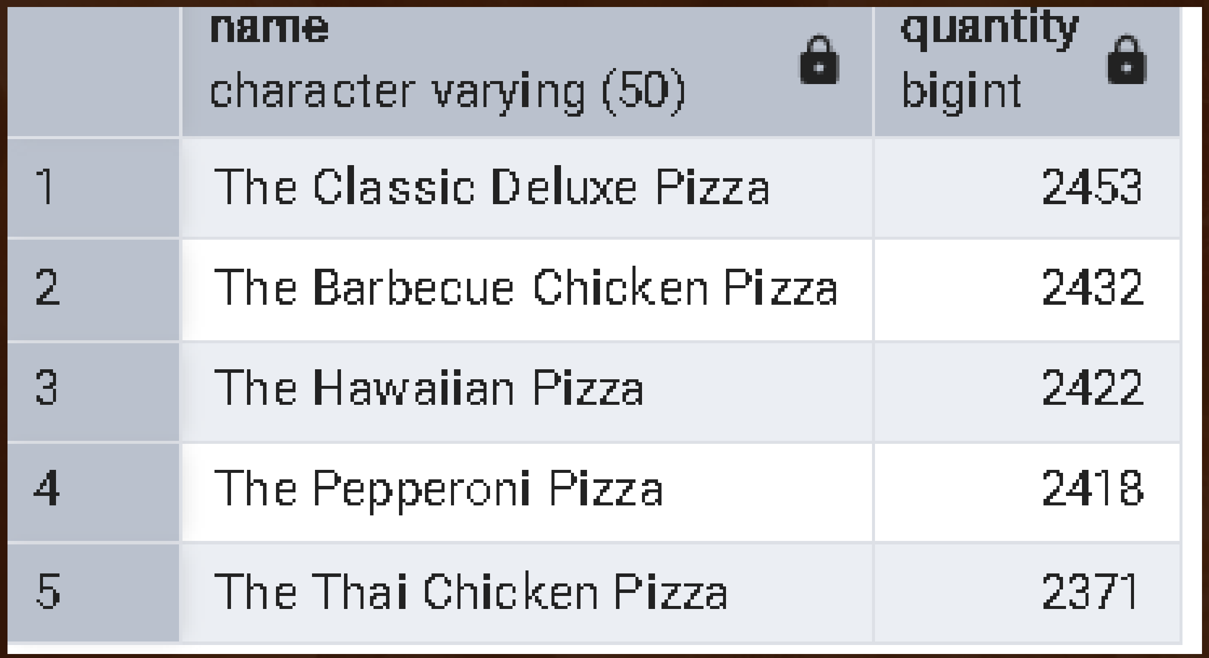


	size character varying (25)	size_count bigint
1	L	18526
2	M	15385
3	S	14137
4	XL	544
5	XXL	28

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

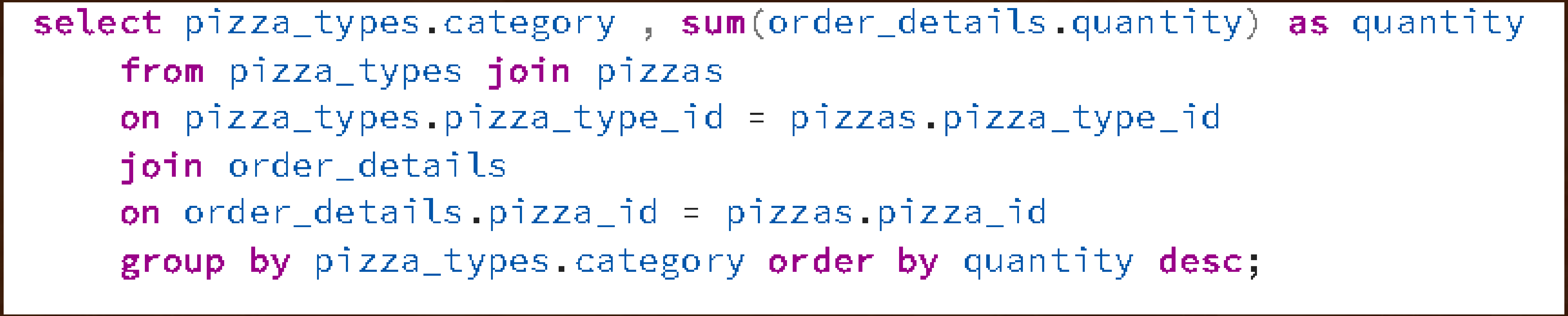


```sql
select pizza_types.name, sum(order_details.quantity) as quantity
   from pizza_types
   join pizzas
   on pizza_types.pizza_type_id = pizzas.pizza_type_id
   join order_details
   on order_details.pizza_id = pizzas.pizza_id
   group by pizza_types.name
   order by quantity desc
   limit 5;
```

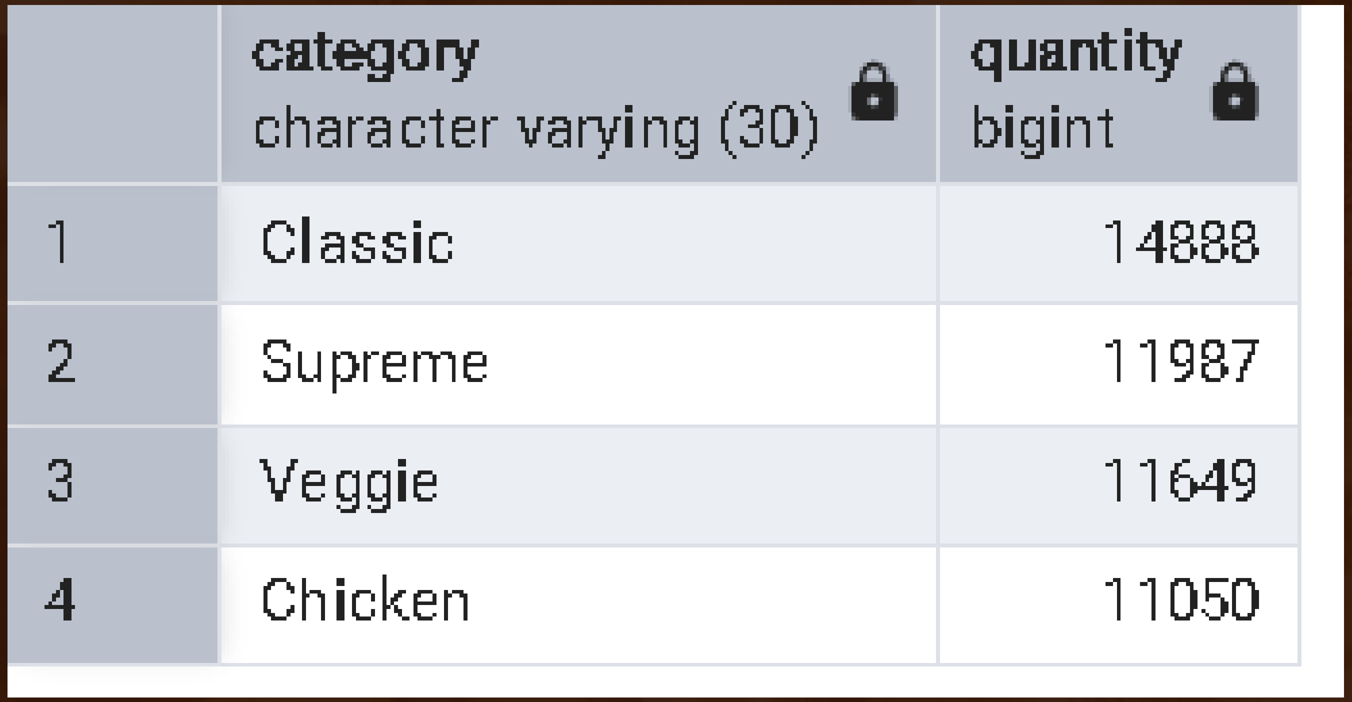


	name character varying (50)	quantity bigint
1	The Classic Deluxe Pizza	2453
2	The Barbecue Chicken Pizza	2432
3	The Hawaiian Pizza	2422
4	The Pepperoni Pizza	2418
5	The Thai Chicken Pizza	2371

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

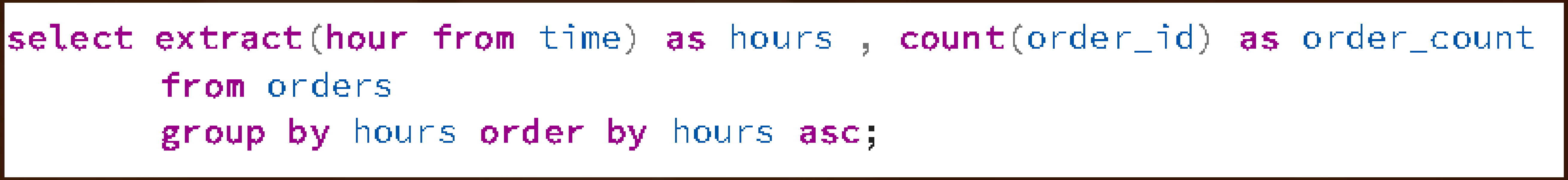


```
select pizza_types.category , sum(order_details.quantity) as quantity
    from pizza_types join pizzas
    on pizza_types.pizza_type_id = pizzas.pizza_type_id
    join order_details
    on order_details.pizza_id = pizzas.pizza_id
    group by pizza_types.category order by quantity desc;
```

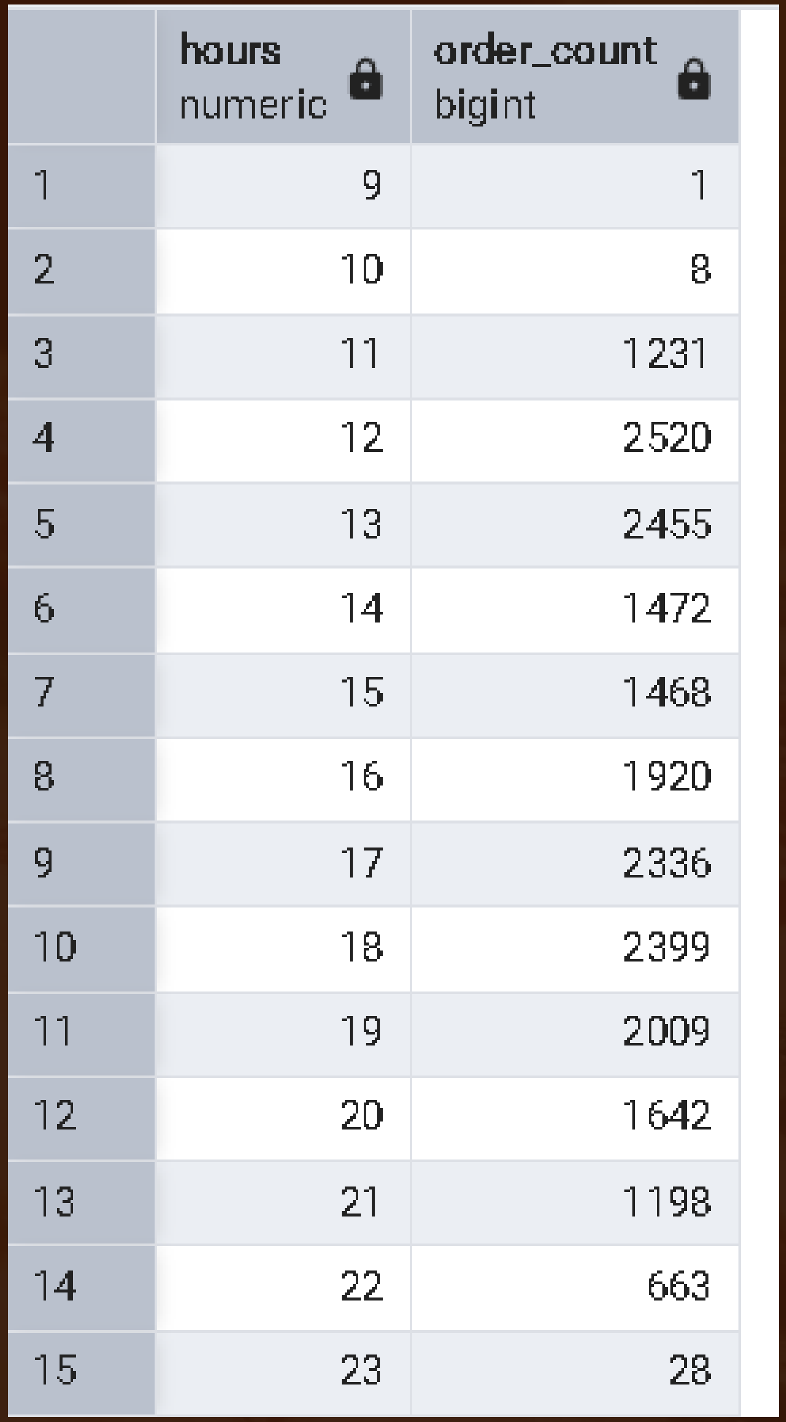


	category character varying (30)	quantity bigint
1	Classic	14888
2	Supreme	11987
3	Veggie	11649
4	Chicken	11050

DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

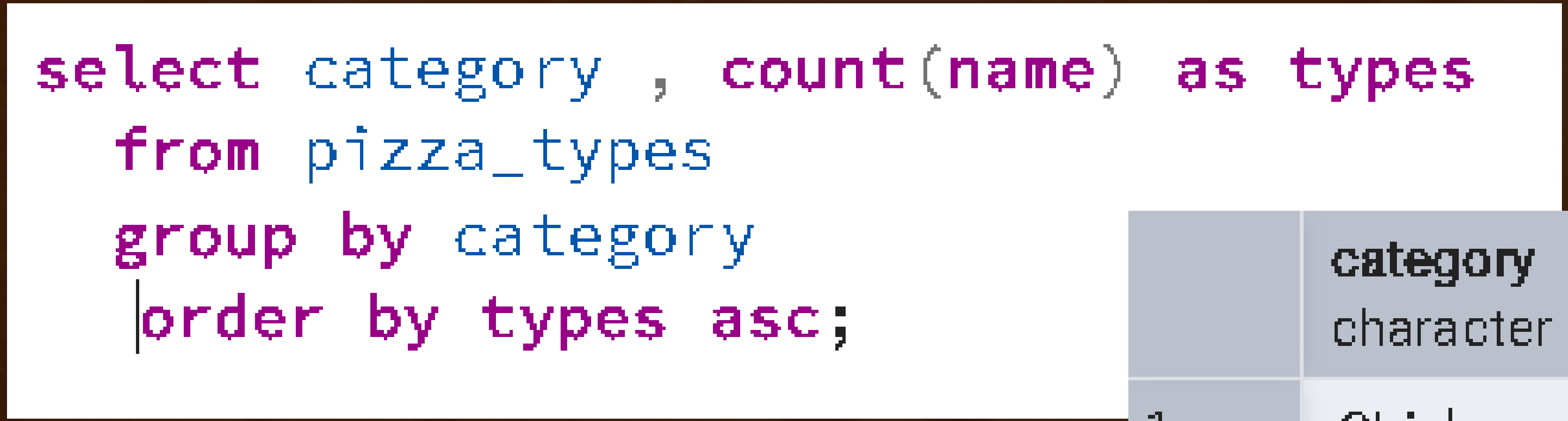


```
select extract(hour from time) as hours , count(order_id) as order_count
       from orders
       group by hours order by hours asc;
```

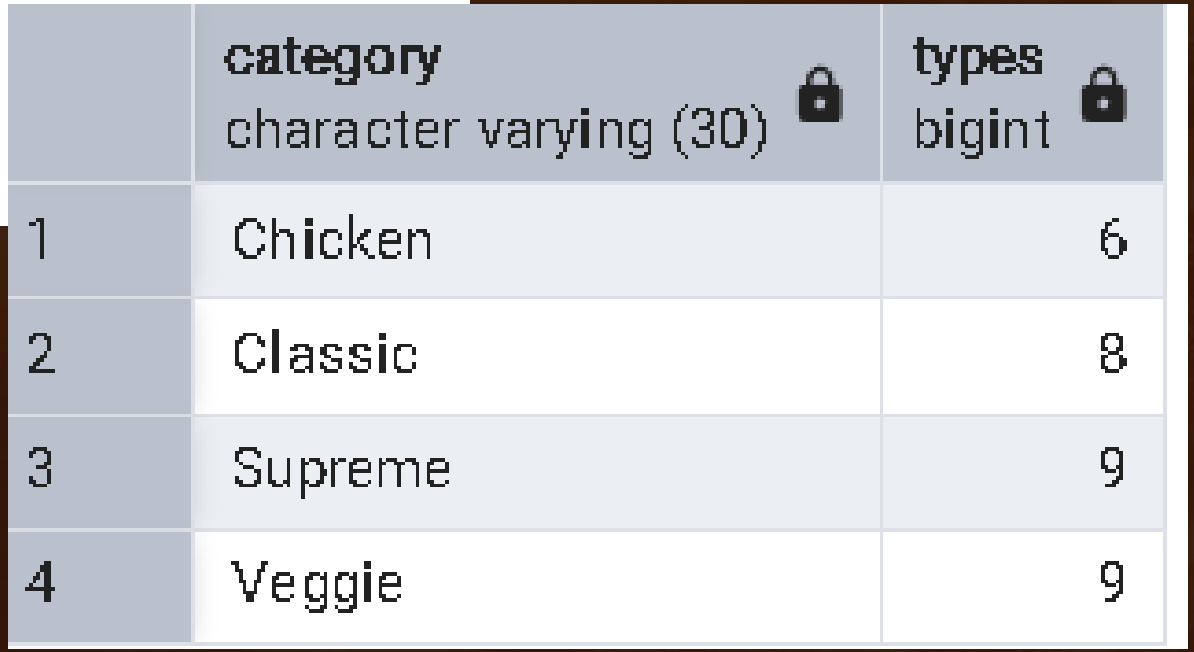


	hours numeric	order_count bigint
1	9	1
2	10	8
3	11	1231
4	12	2520
5	13	2455
6	14	1472
7	15	1468
8	16	1920
9	17	2336
10	18	2399
11	19	2009
12	20	1642
13	21	1198
14	22	663
15	23	28

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

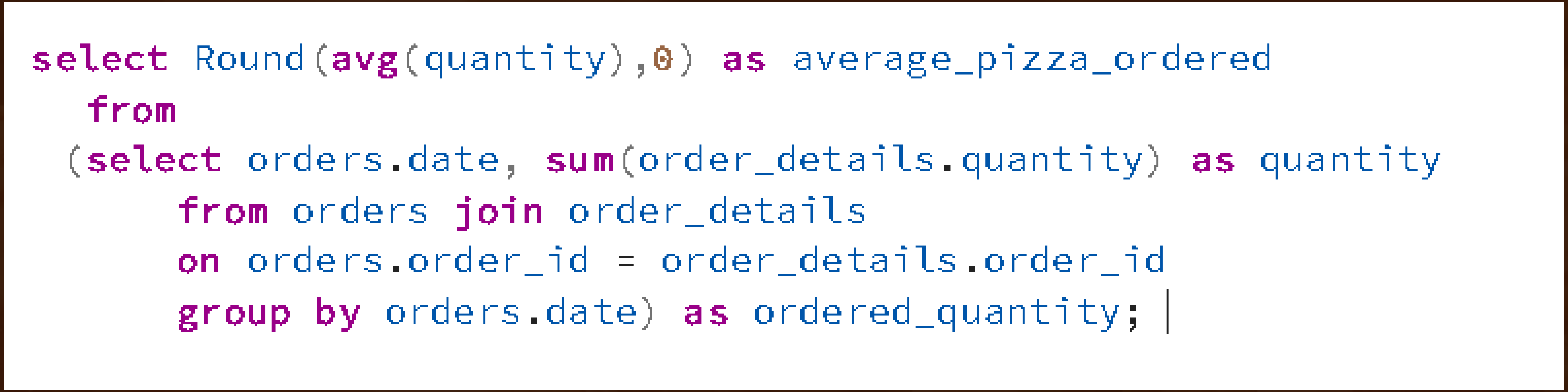


```
select category , count(name) as types
  from pizza_types
  group by category
  order by types asc;
```

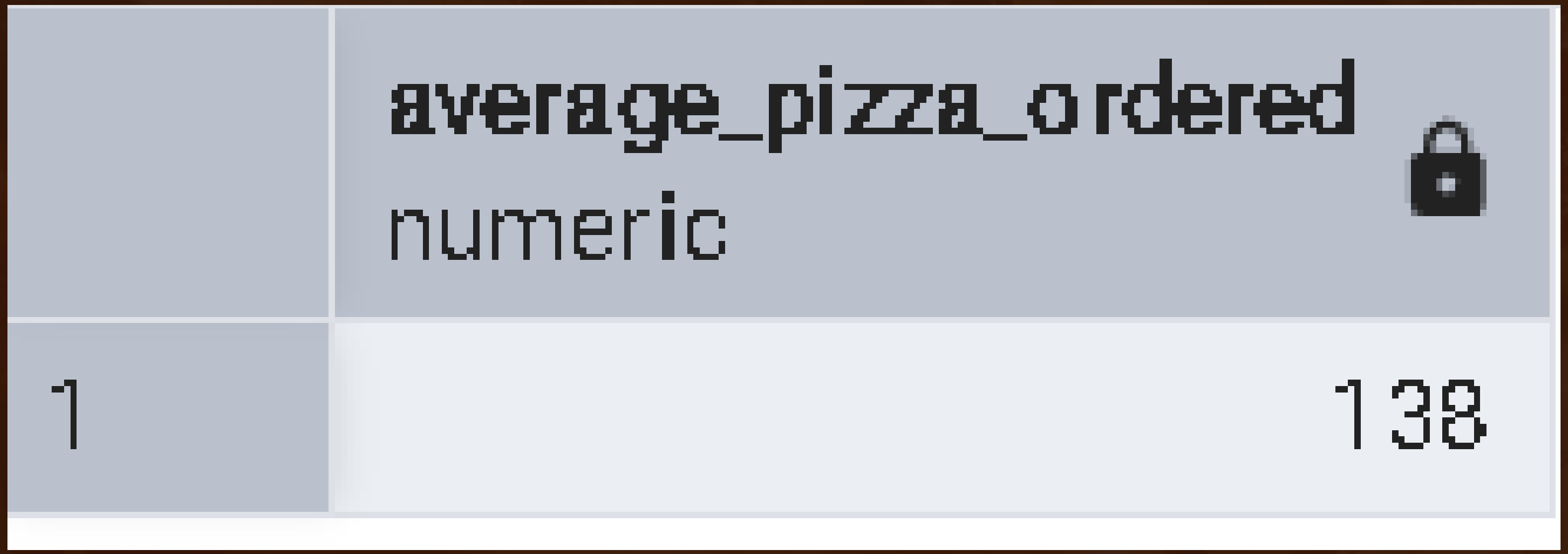


	category character varying (30)	types bigint
1	Chicken	6
2	Classic	8
3	Supreme	9
4	Veggie	9

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

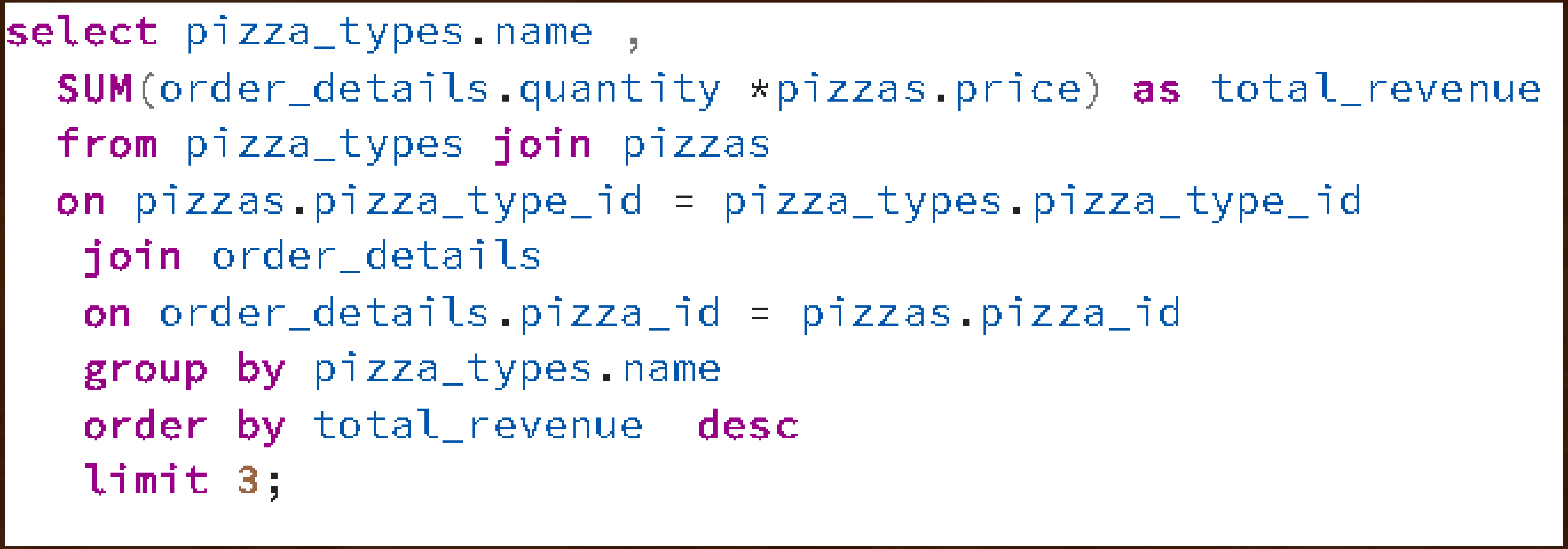


```sql
select Round(avg(quantity),0) as average_pizza_ordered
   from
  (select orders.date, sum(order_details.quantity) as quantity
       from orders join order_details
       on orders.order_id = order_details.order_id
       group by orders.date) as ordered_quantity;
```

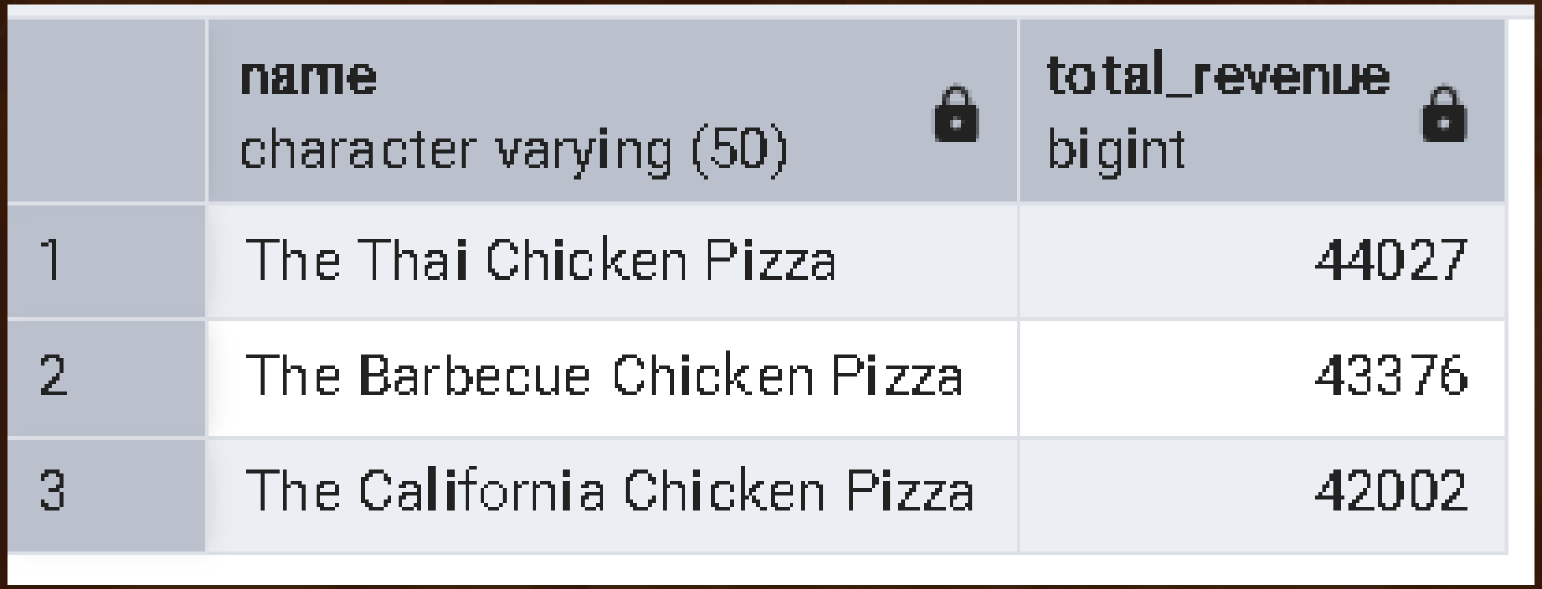


	average_pizza_ordered numeric
1	138

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

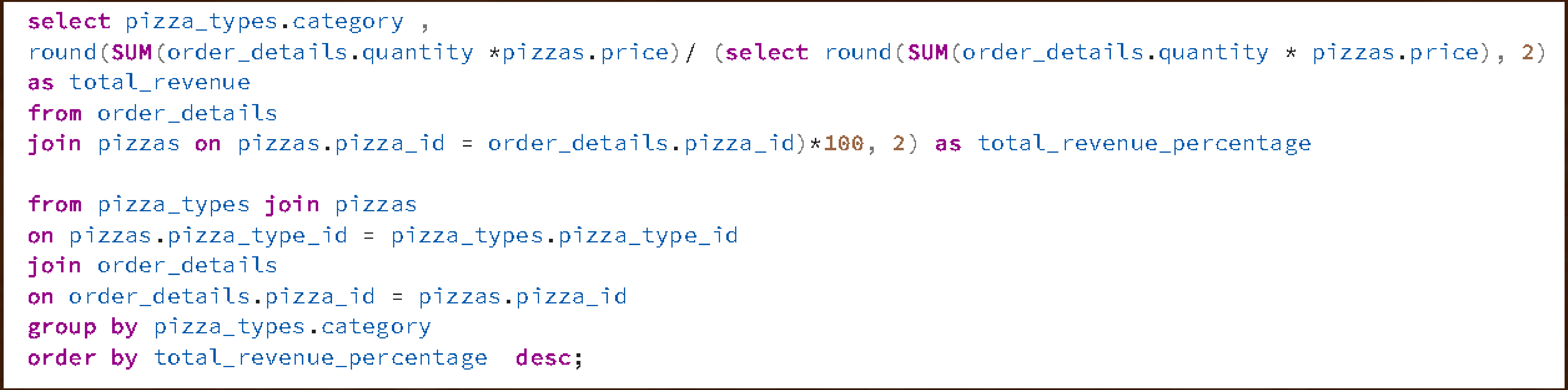


```
select pizza_types.name ,
  SUM(order_details.quantity *pizzas.price) as total_revenue
  from pizza_types join pizzas
  on pizzas.pizza_type_id = pizza_types.pizza_type_id
   join order_details
   on order_details.pizza_id = pizzas.pizza_id
   group by pizza_types.name
   order by total_revenue 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



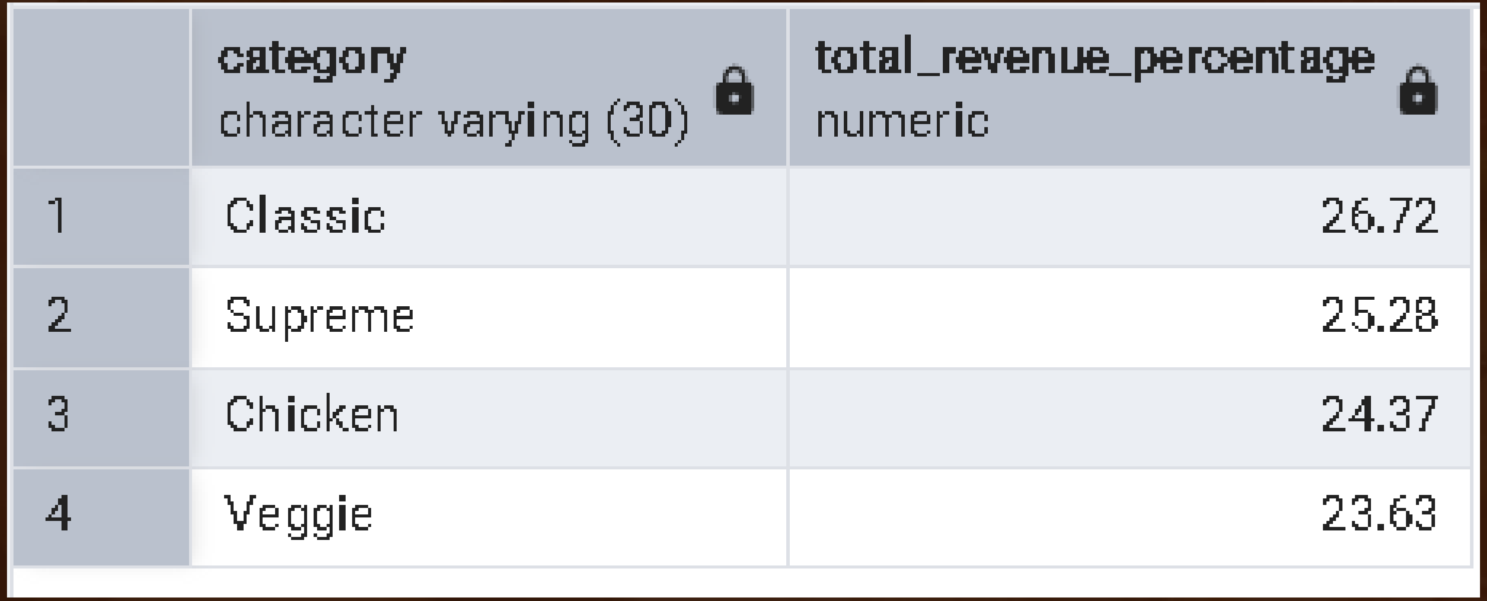
	name character varying (50)	total_revenue bigint
1	The Thai Chicken Pizza	44027
2	The Barbecue Chicken Pizza	43376
3	The California Chicken Pizza	42002

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



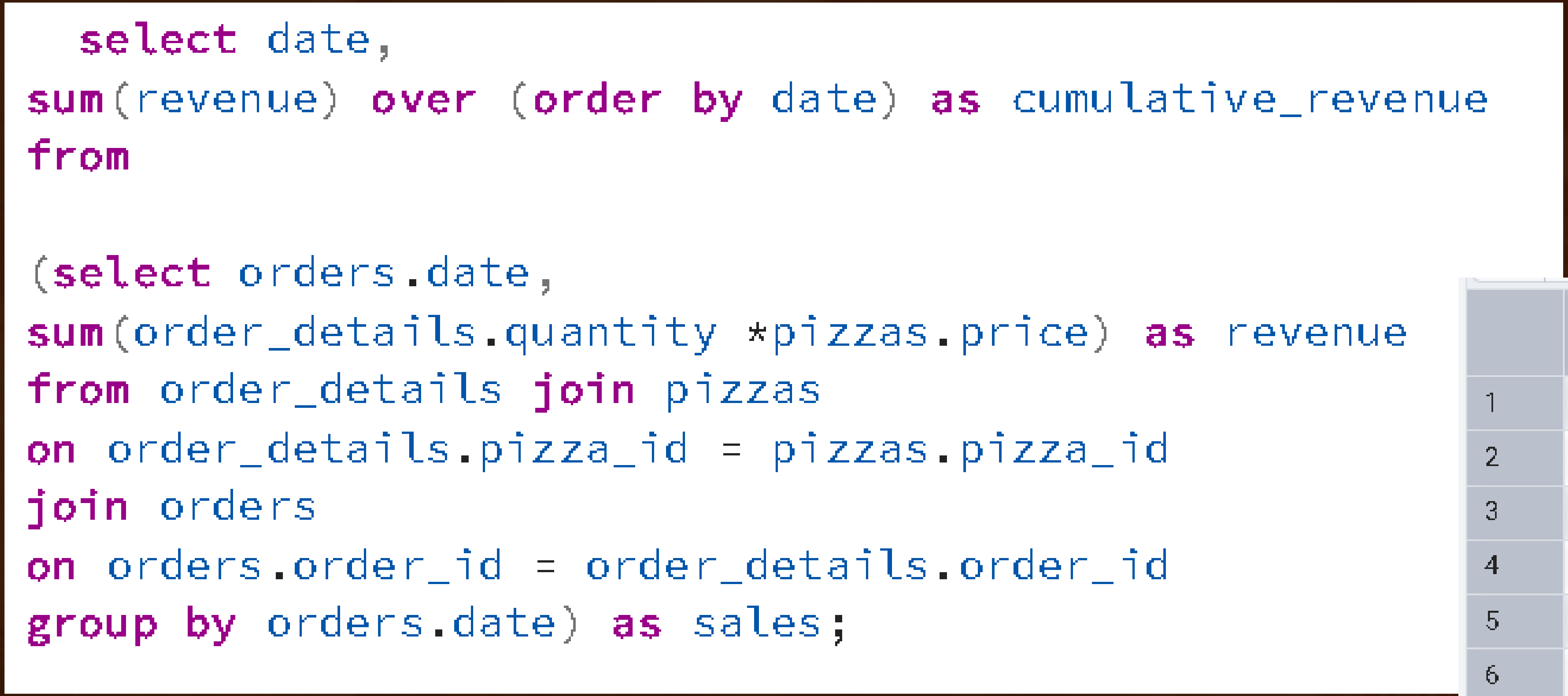
```
select pizza_types.category ,
round(SUM(order_details.quantity *pizzas.price)/ (select round(SUM(order_details.quantity * pizzas.price), 2)
as total_revenue
from order_details
join pizzas on pizzas.pizza_id = order_details.pizza_id)*100, 2) as total_revenue_percentage

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.category
order by total_revenue_percentage  desc;
```



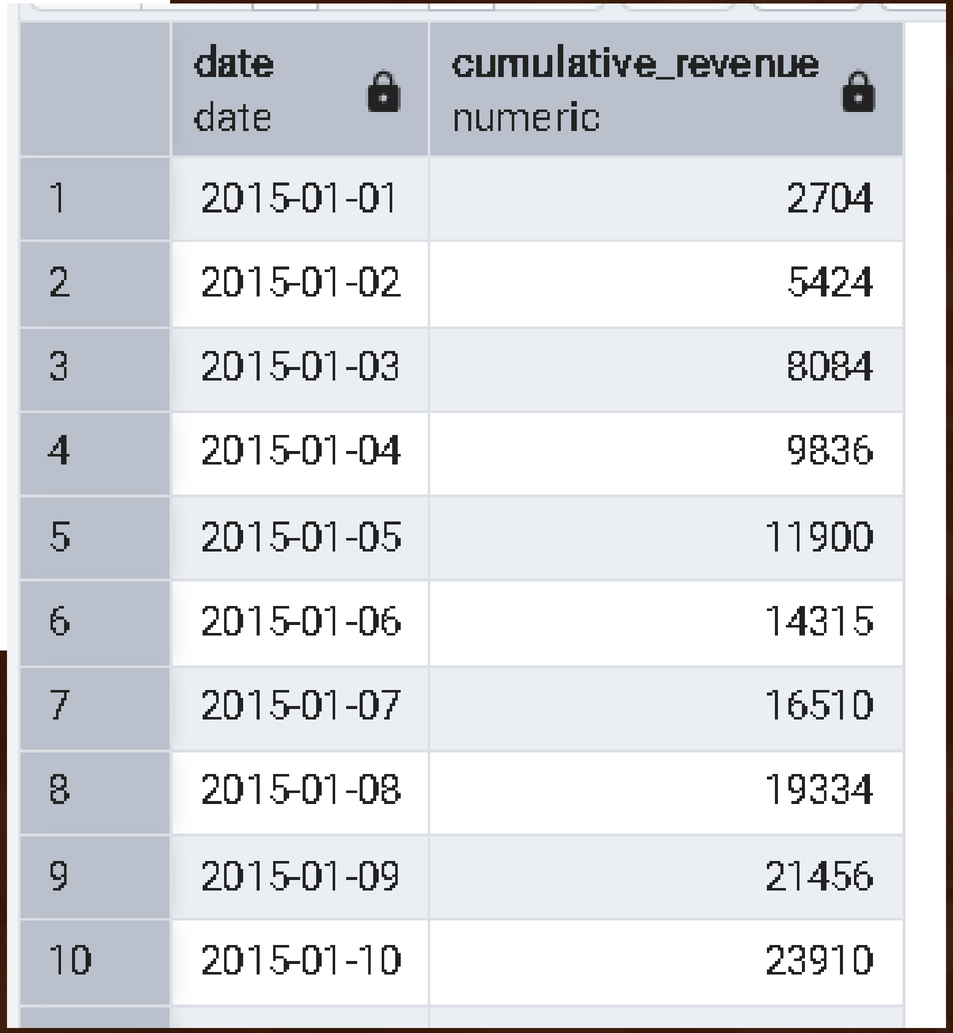
	category character varying (30)	total_revenue_percentage numeric
1	Classic	26.72
2	Supreme	25.28
3	Chicken	24.37
4	Veggie	23.63

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.



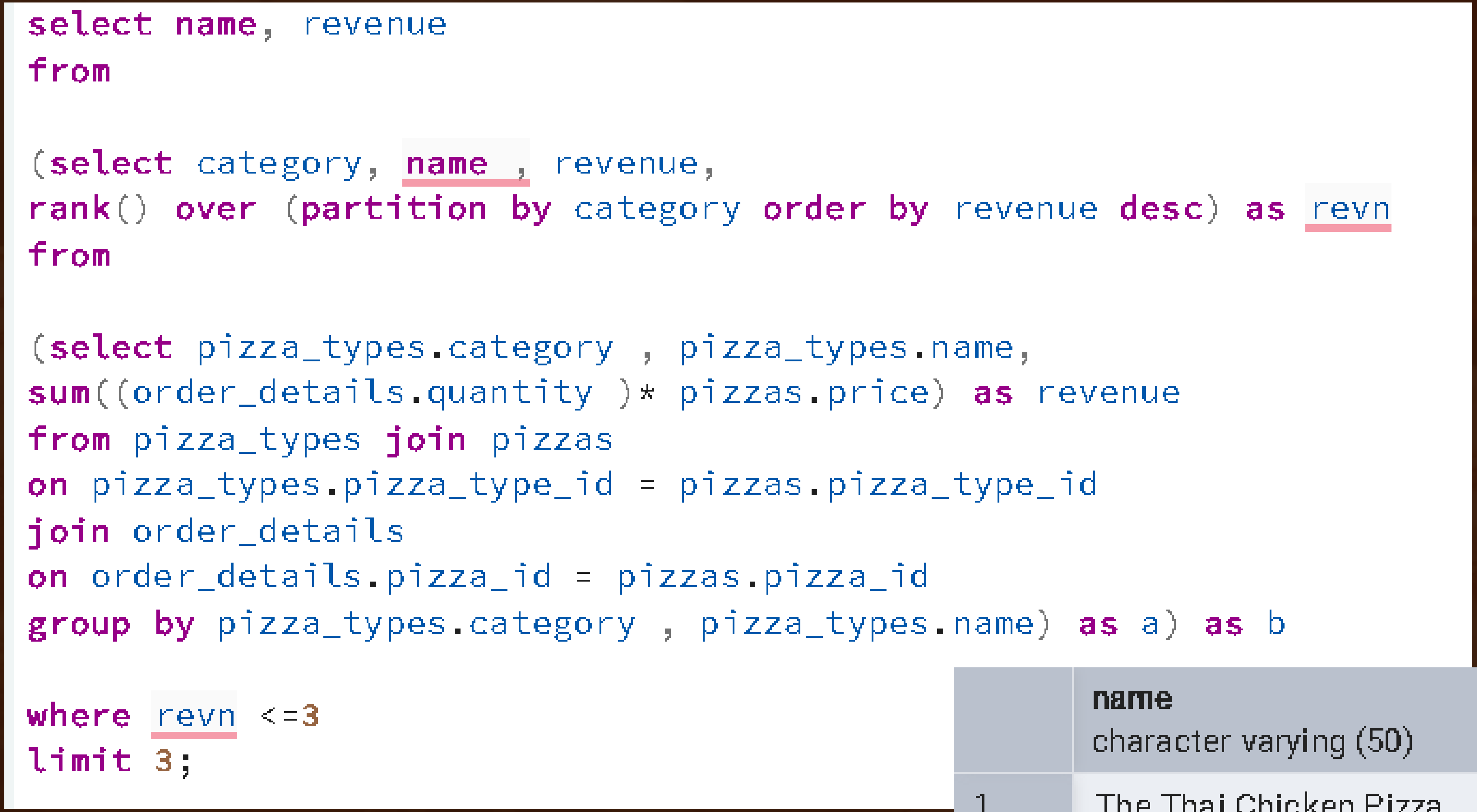
```
  select date,
sum(revenue) over (order by date) as cumulative_revenue
from

(select orders.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.date) as sales;
```



	date date	cumulative_revenue numeric
1	2015-01-01	2704
2	2015-01-02	5424
3	2015-01-03	8084
4	2015-01-04	9836
5	2015-01-05	11900
6	2015-01-06	14315
7	2015-01-07	16510
8	2015-01-08	19334
9	2015-01-09	21456
10	2015-01-10	23910

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

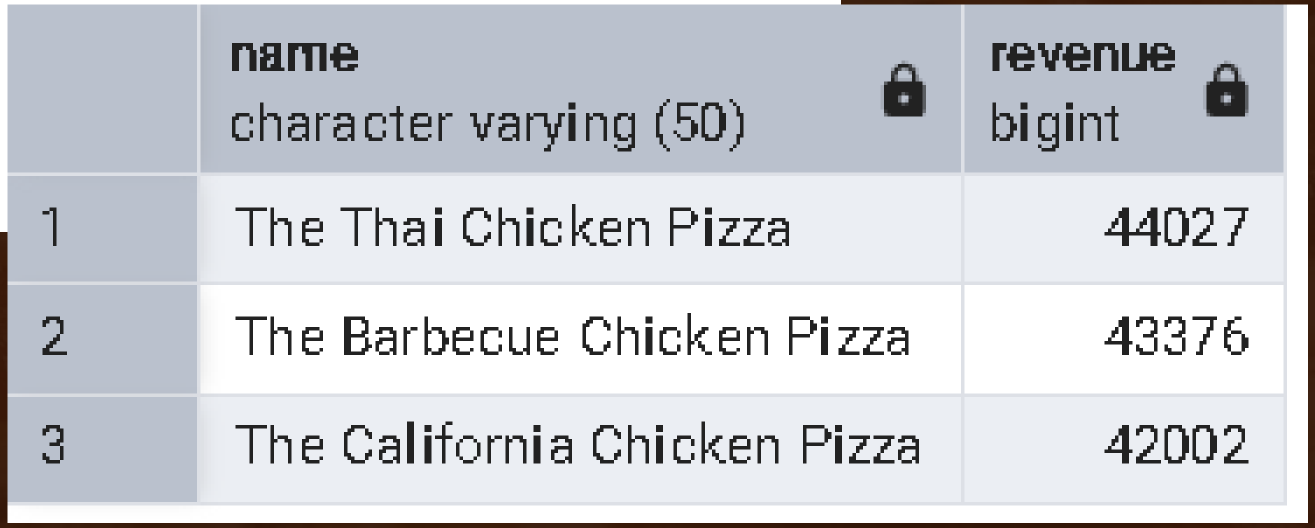


```
select name, revenue
from

(select category, name , revenue,
rank() over (partition by category order by revenue desc) as revn
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 revn <=3
limit 3;
```



	name character varying (50)	revenue bigint
1	The Thai Chicken Pizza	44027
2	The Barbecue Chicken Pizza	43376
3	The California Chicken Pizza	42002

Pizza Sales Presentation

THANK YOU FOR ATTENTION

My LinkedIN

My Github