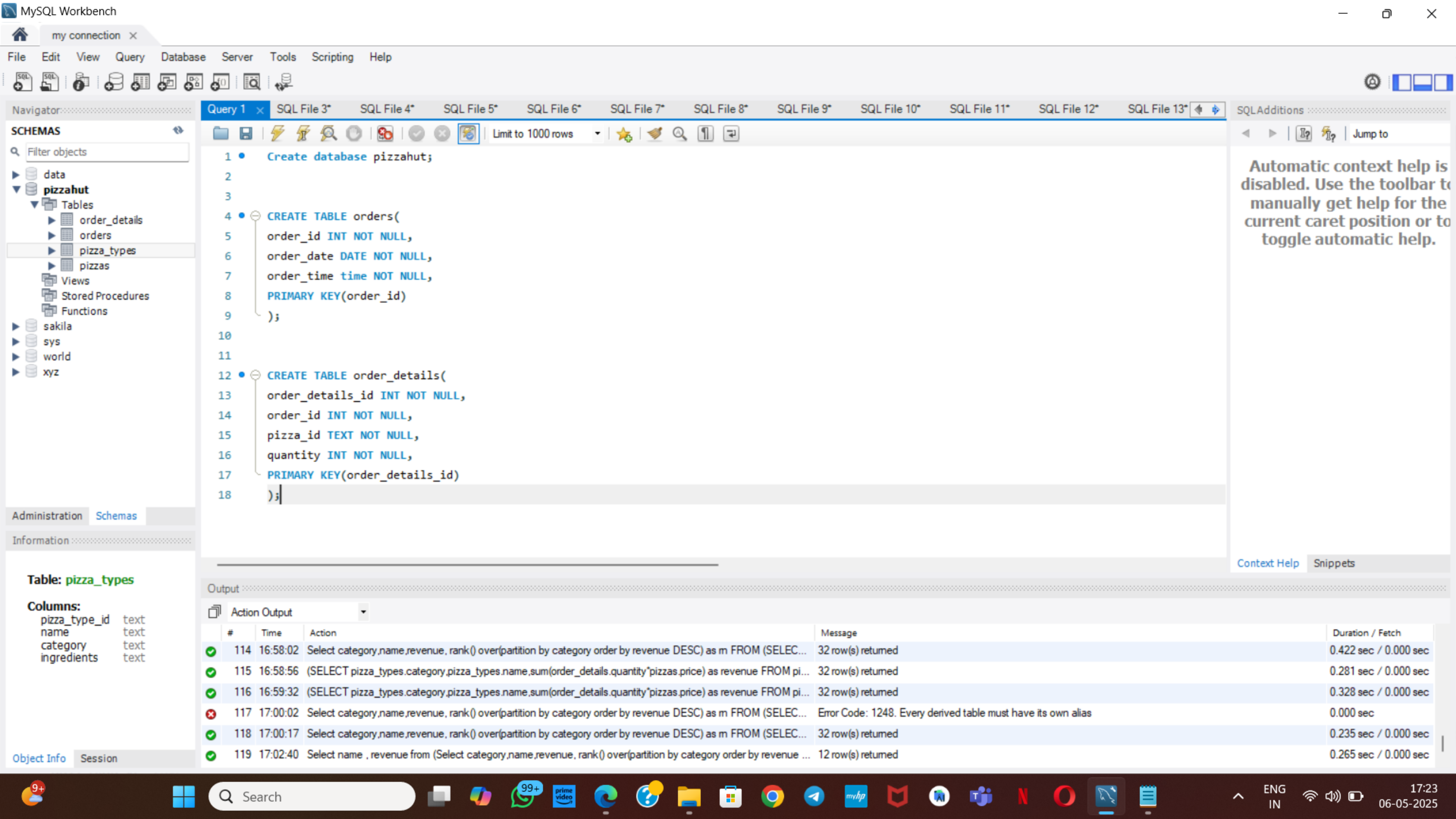


```
Create database pizzahut;


CREATE TABLE orders(
order_id INT NOT NULL,
order_date DATE NOT NULL,
order_time time NOT NULL,
PRIMARY KEY(order_id)
);


CREATE TABLE order_details(
order_details_id INT NOT NULL,
order_id INT NOT NULL,
pizza_id TEXT NOT NULL,
quantity INT NOT NULL,
PRIMARY KEY(order_details_id)
);
```

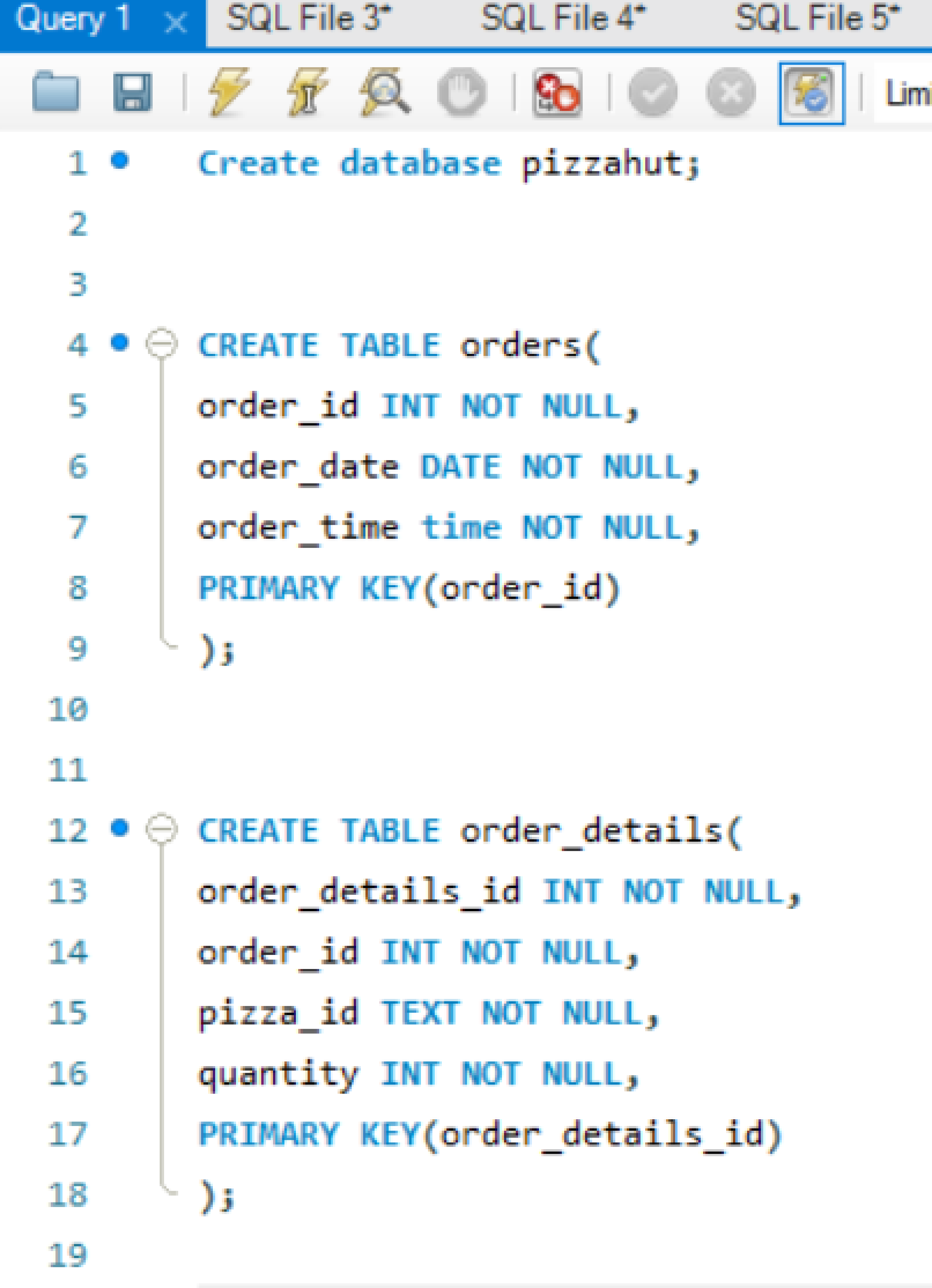
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Table: pizza_types

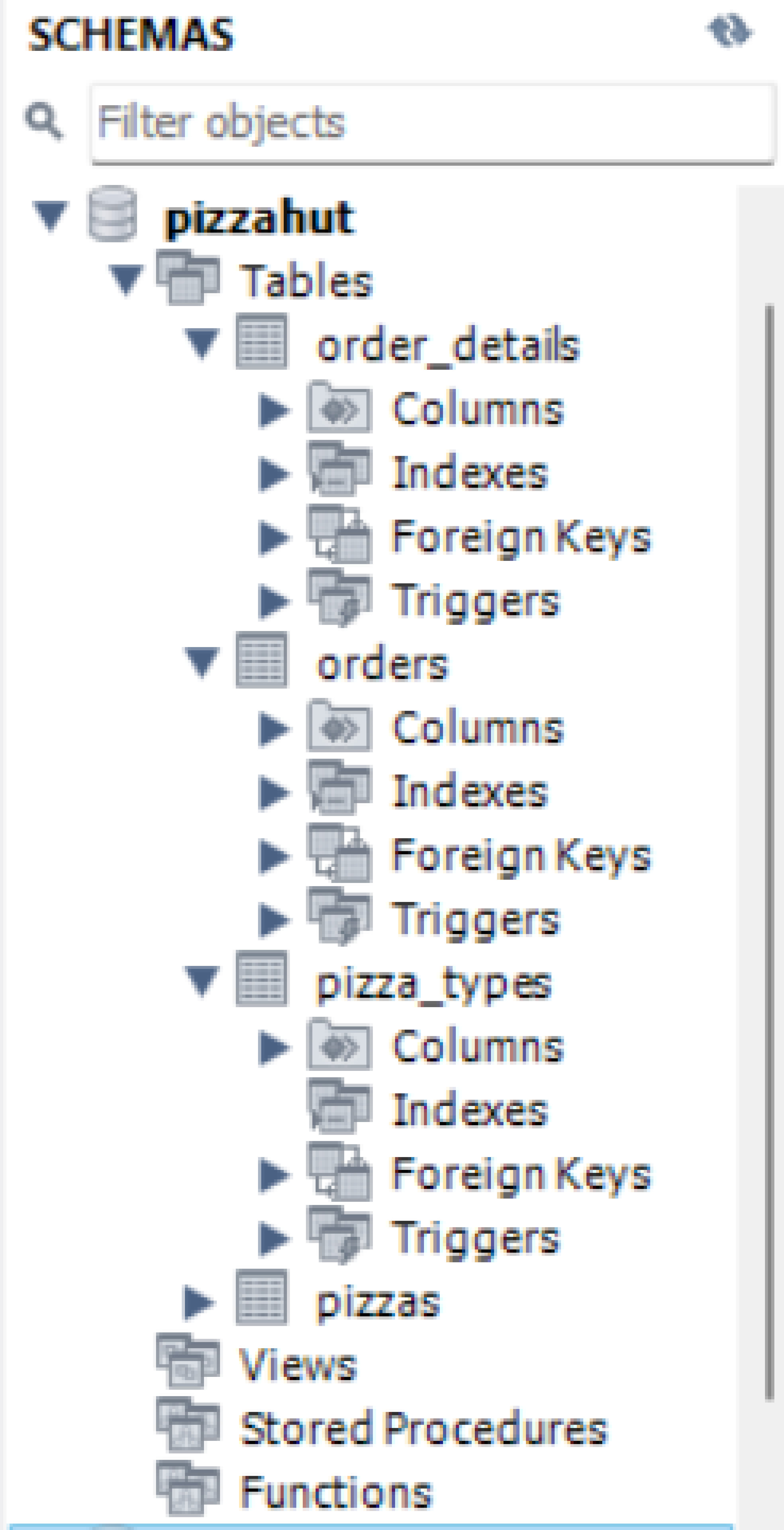
Columns:

pizza_type_id	text
name	text
category	text
ingredients	text

#	Time	Action	Message	Duration / Fetch
114	16:58:02	Select category,name,revenue, rank() over(partition by category order by revenue DESC) as rn FROM (SELEC...	32 row(s) returned	0.422 sec / 0.000 sec
115	16:58:56	(SELECT pizza_types.category,pizza_types.name,sum(order_details.quantity*pizzas.price) as revenue FROM pi...	32 row(s) returned	0.281 sec / 0.000 sec
116	16:59:32	(SELECT pizza_types.category,pizza_types.name,sum(order_details.quantity*pizzas.price) as revenue FROM pi...	32 row(s) returned	0.328 sec / 0.000 sec
117	17:00:02	Select category,name,revenue, rank() over(partition by category order by revenue DESC) as rn FROM (SELEC...	Error Code: 1248. Every derived table must have its own alias	0.000 sec
118	17:00:17	Select category,name,revenue, rank() over(partition by category order by revenue DESC) as rn FROM (SELEC...	32 row(s) returned	0.235 sec / 0.000 sec
119	17:02:40	Select name , revenue from (Select category,name,revenue, rank() over(partition by category order by revenue ...	12 row(s) returned	0.265 sec / 0.000 sec



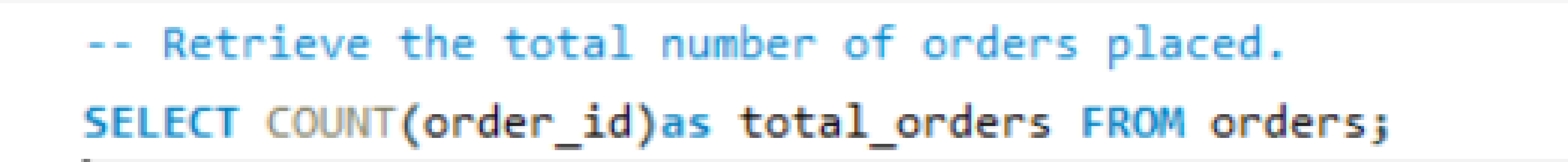
```sql
Create database pizzahut;


CREATE TABLE orders(
order_id INT NOT NULL,
order_date DATE NOT NULL,
order_time time NOT NULL,
PRIMARY KEY(order_id)
);


CREATE TABLE order_details(
order_details_id INT NOT NULL,
order_id INT NOT NULL,
pizza_id TEXT NOT NULL,
quantity INT NOT NULL,
PRIMARY KEY(order_details_id)
);

```



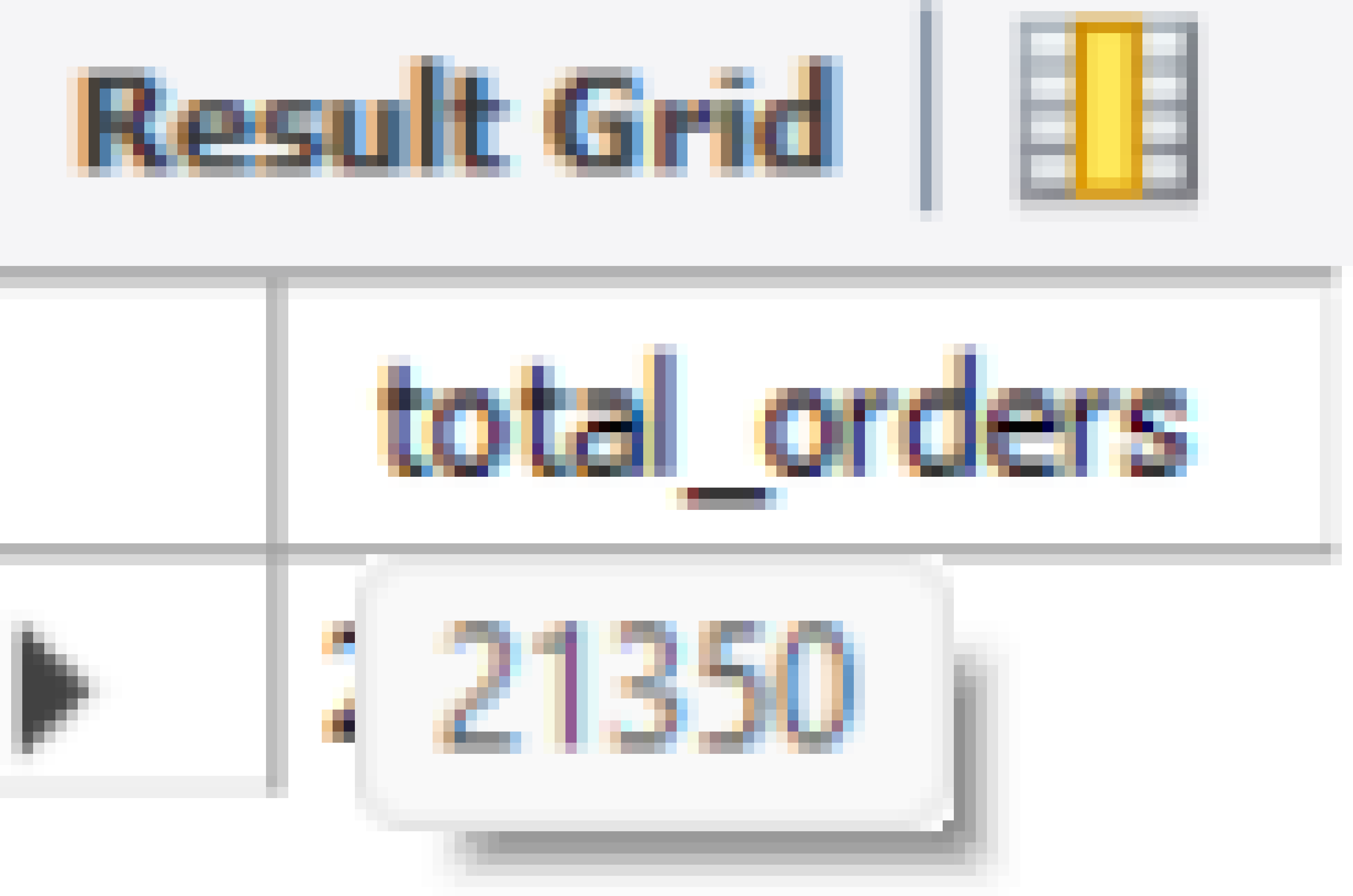
SCHEMAS

Filter objects

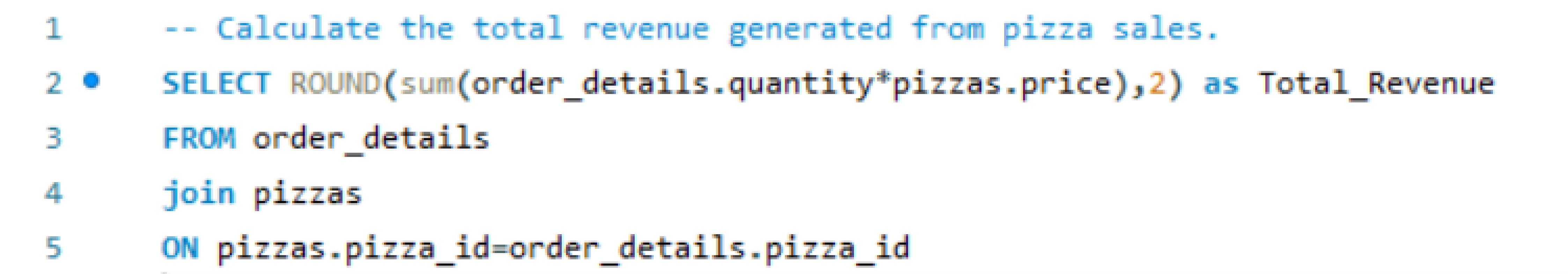
- pizzahut
  - Tables
    - order_details
      - Columns
      - Indexes
      - Foreign Keys
      - Triggers
    - orders
      - Columns
      - Indexes
      - Foreign Keys
      - Triggers
    - pizza_types
      - Columns
      - Indexes
      - Foreign Keys
      - Triggers
    - pizzas
  - Views
  - Stored Procedures
  - Functions



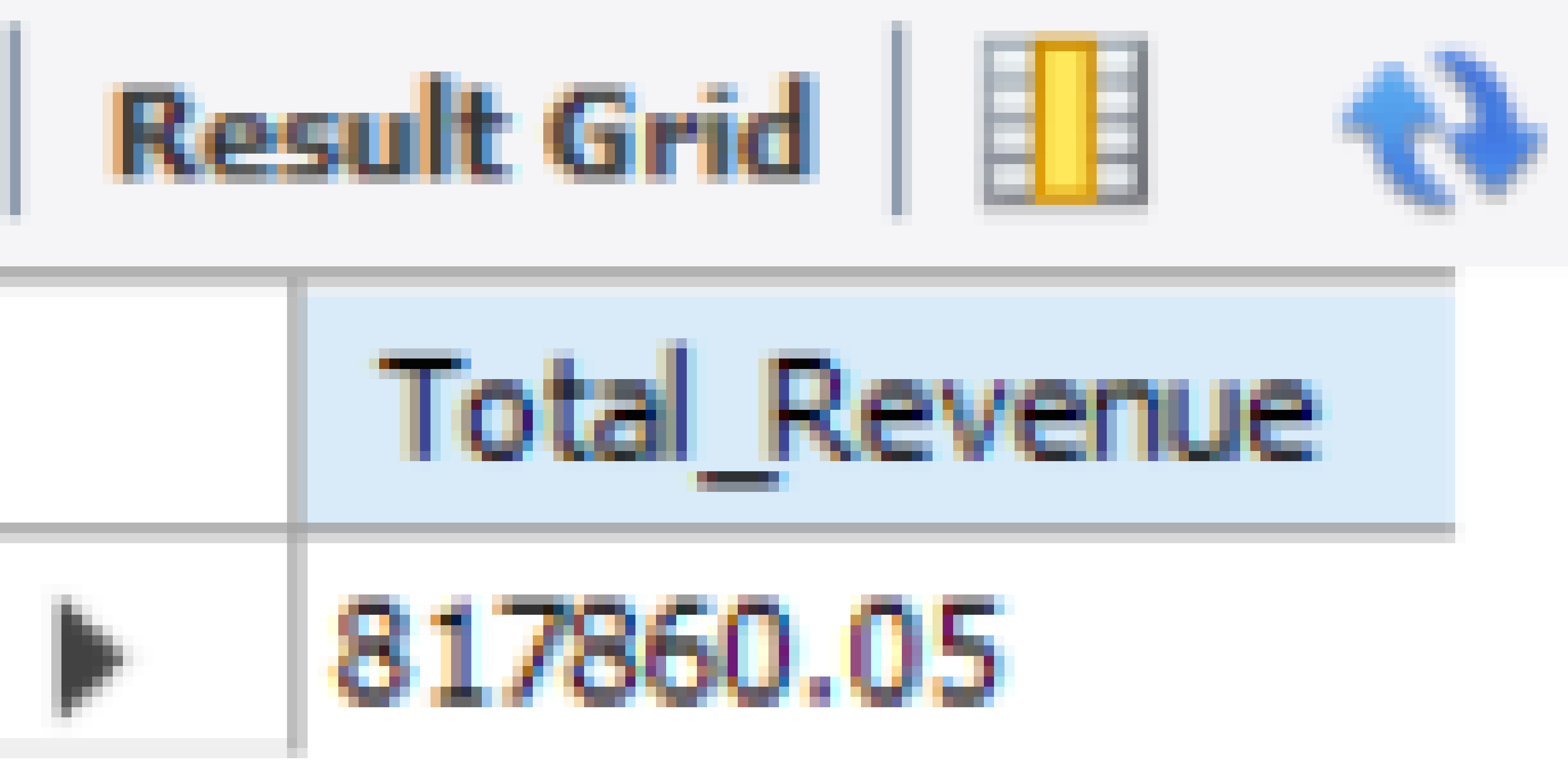
```sql
-- Retrieve the total number of orders placed.
SELECT COUNT(order_id)as total_orders FROM orders;
```



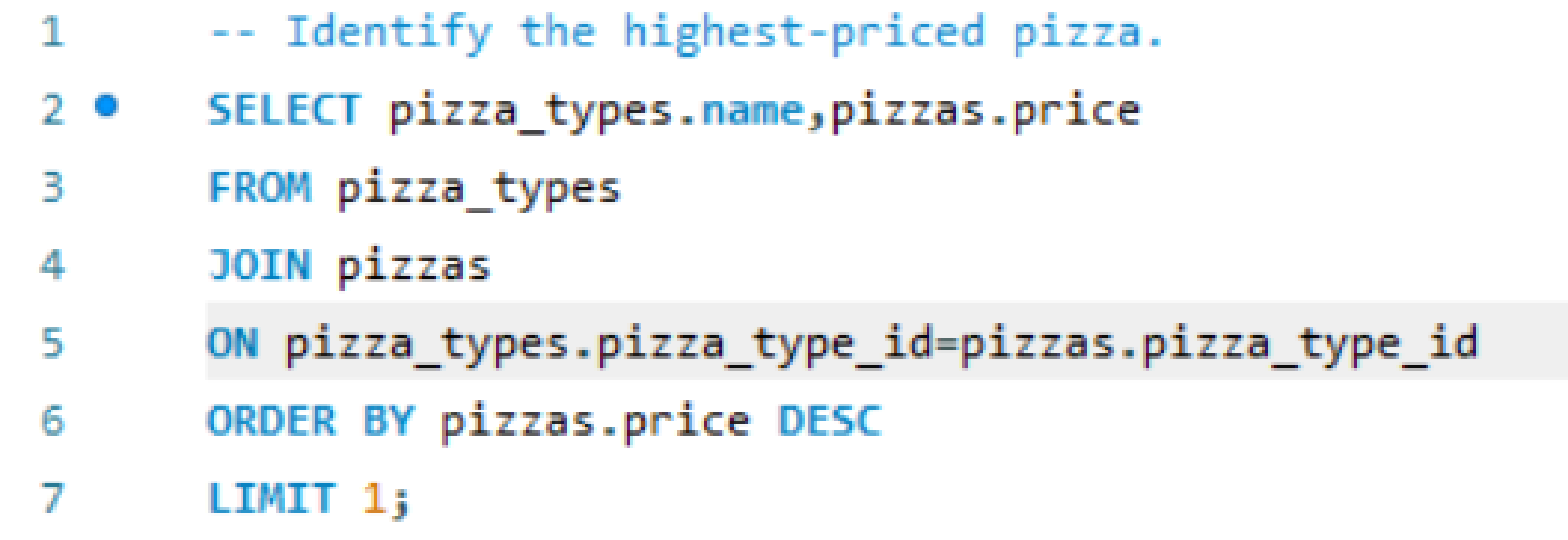
	total_orders
▶	21350



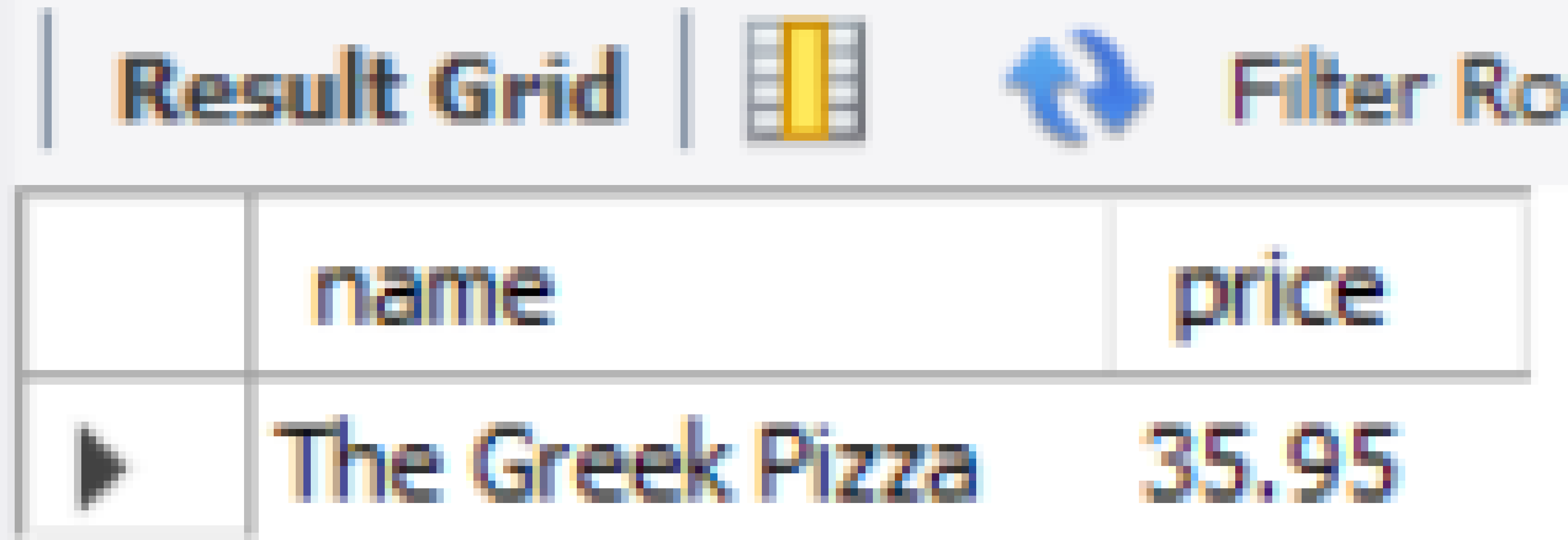
```sql
-- Calculate the total revenue generated from pizza sales.
SELECT ROUND(sum(order_details.quantity*pizzas.price),2) as Total_Revenue
FROM order_details
join pizzas
ON pizzas.pizza_id=order_details.pizza_id
```



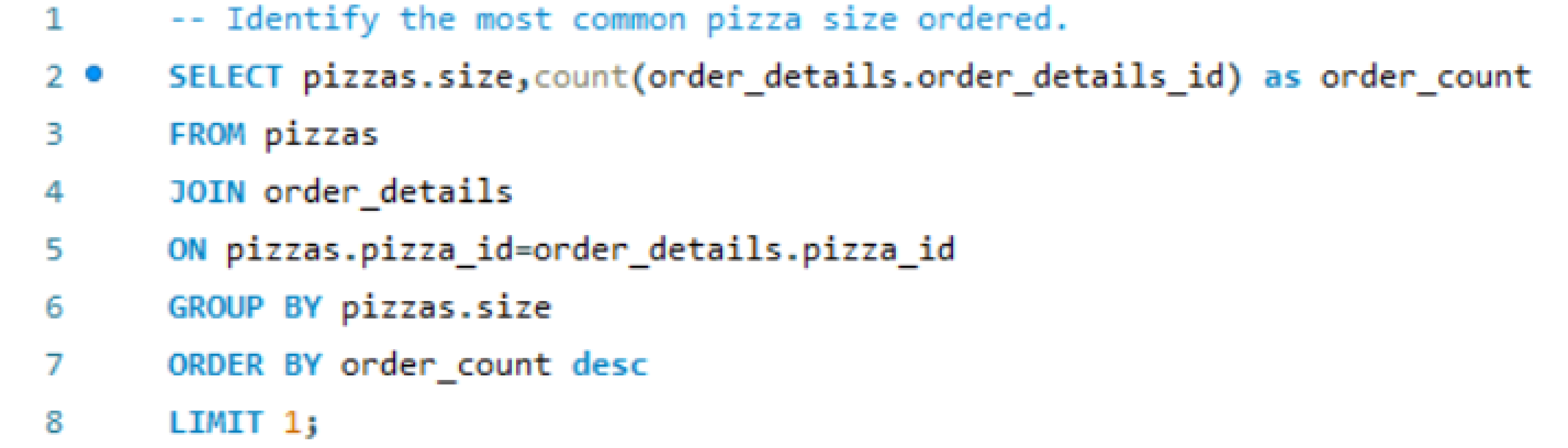
Total_Revenue
817860.05



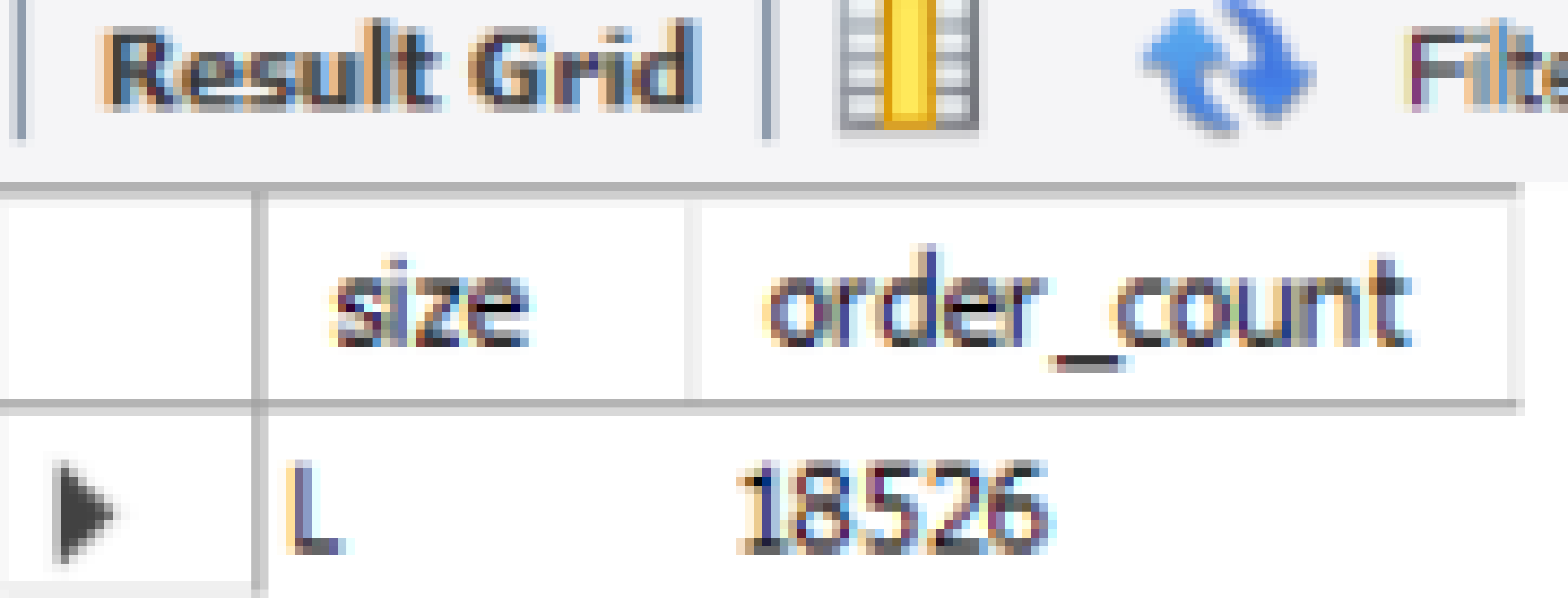
```sql
-- Identify the highest-priced pizza.
SELECT pizza_types.name,pizzas.price
FROM pizza_types
JOIN pizzas
ON pizza_types.pizza_type_id=pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```



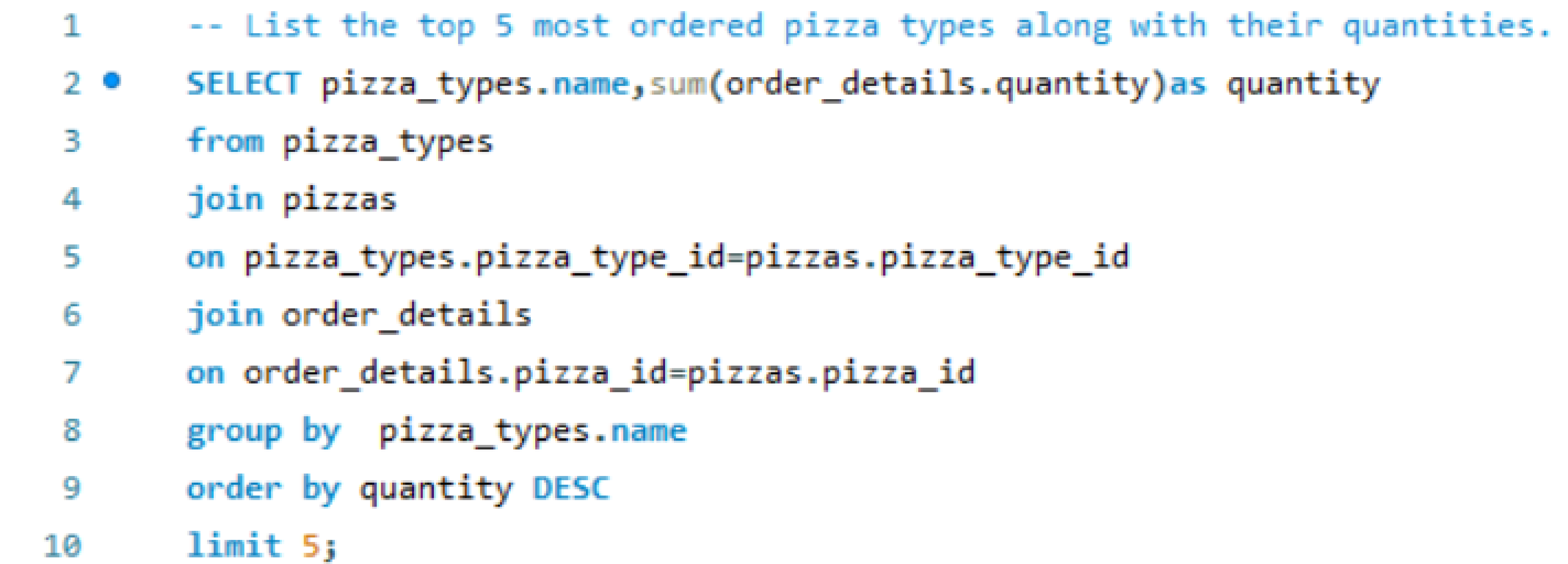
	name	price
▶	The Greek Pizza	35.95



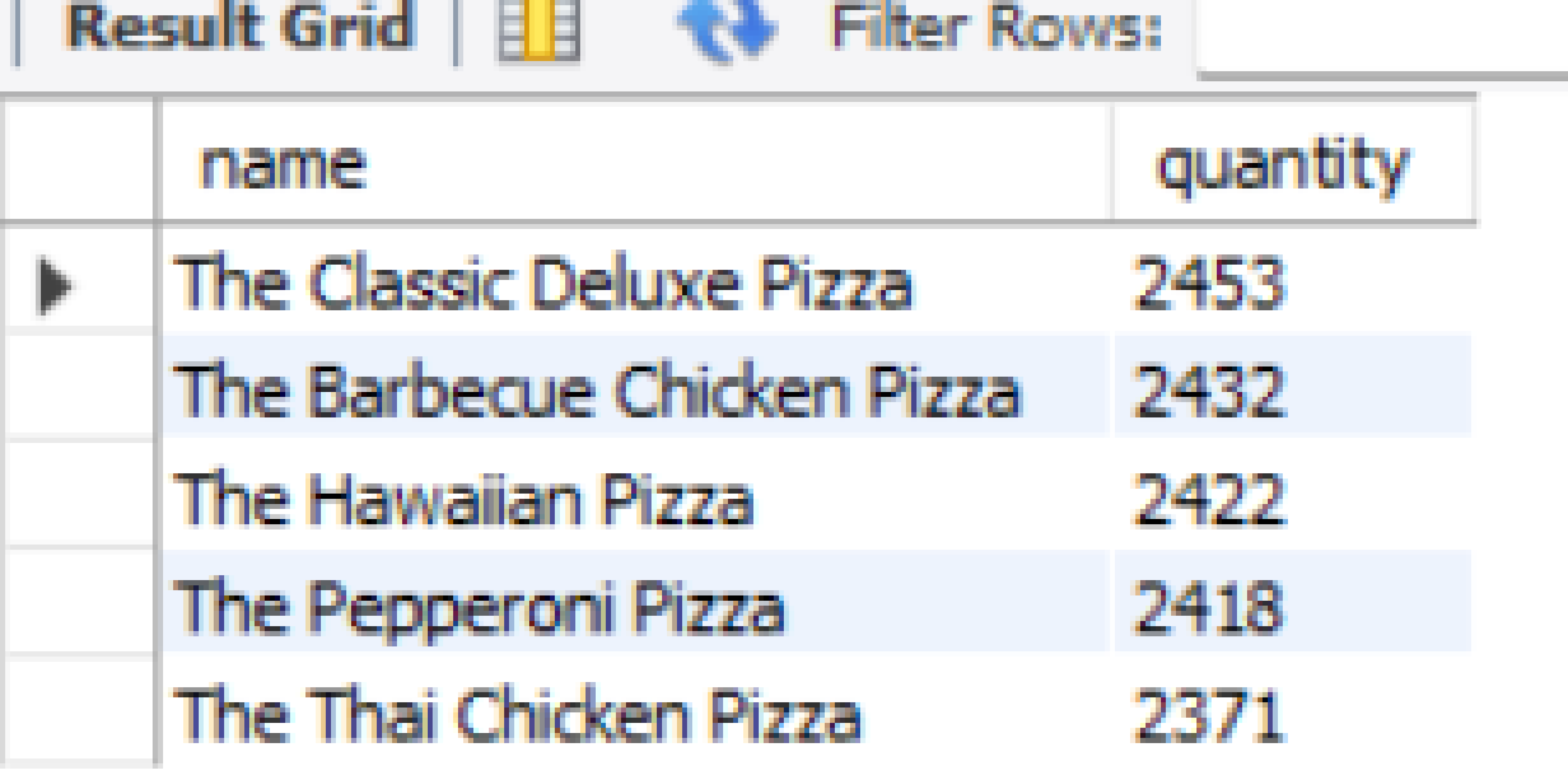
```sql
-- Identify the most common pizza size ordered.
SELECT pizzas.size,count(order_details.order_details_id) as order_count
FROM pizzas
JOIN order_details
ON pizzas.pizza_id=order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count desc
LIMIT 1;
```



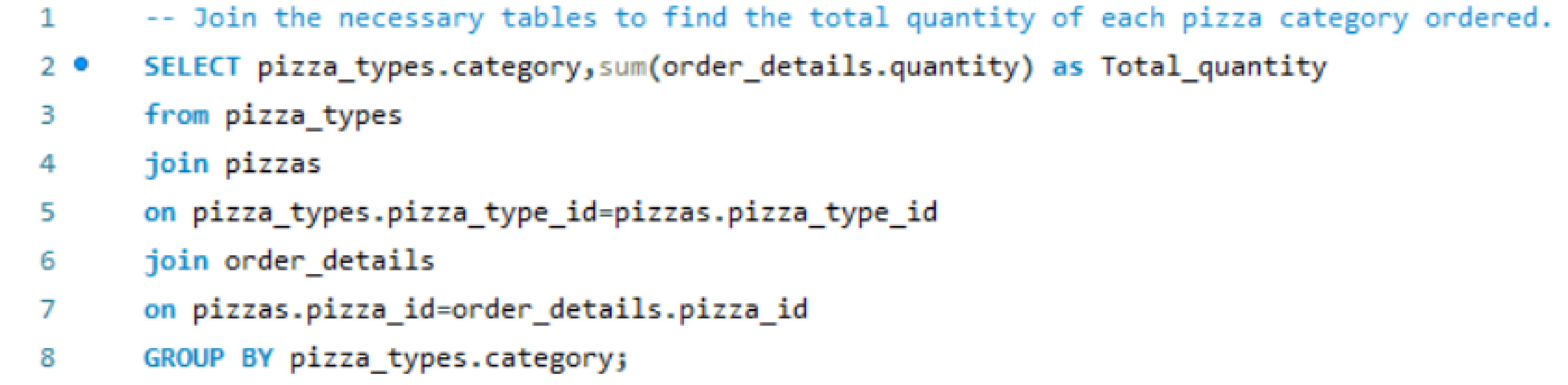
size	order_count
L	18526



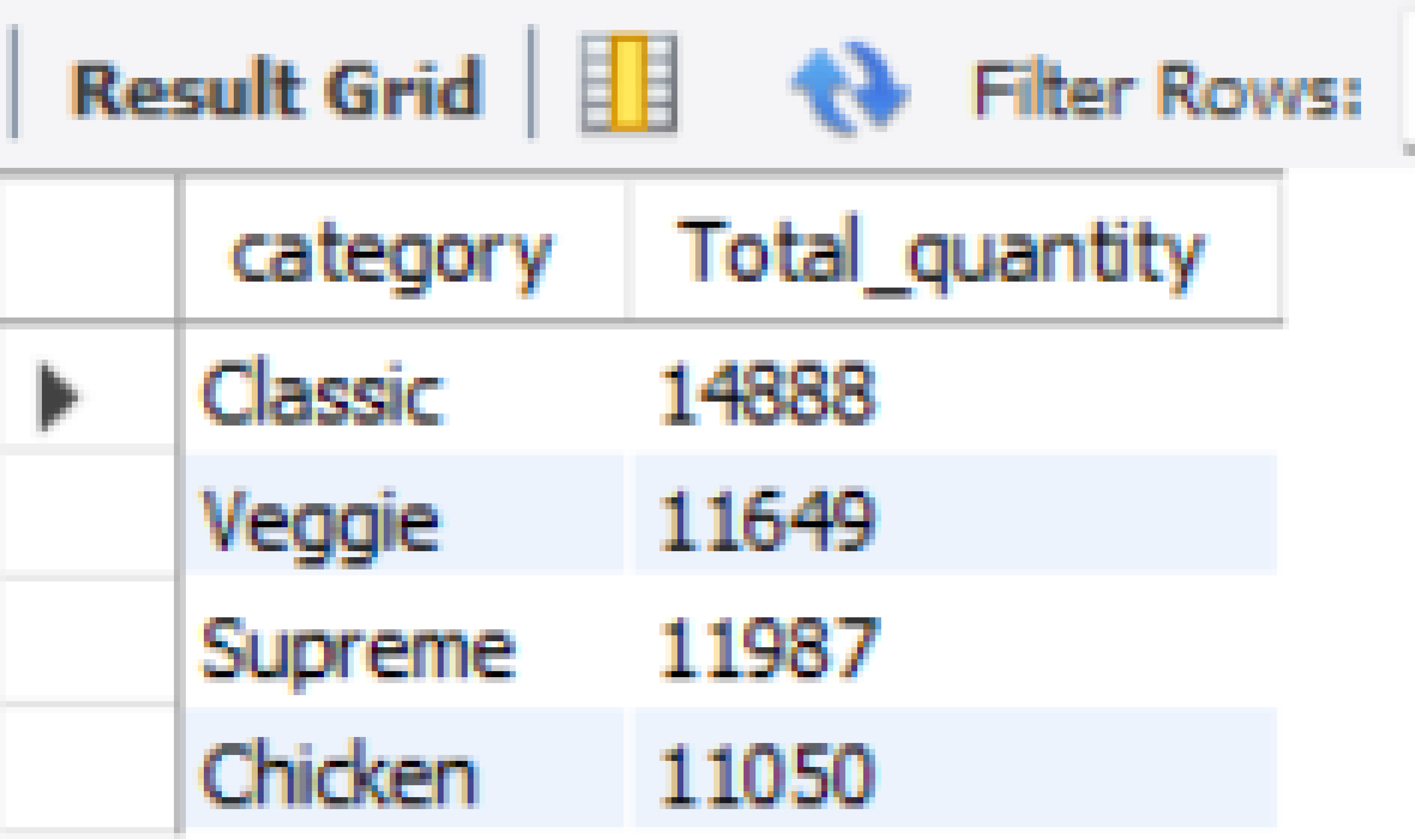
```sql
-- List the top 5 most ordered pizza types along with their quantities.
SELECT pizza_types.name,sum(order_details.quantity)as quantity
from pizza_types
join pizzas
on pizza_types.pizza_type_id=pizzas.pizza_type_id
join order_details
on order_details.pizza_id=pizzas.pizza_id
group by  pizza_types.name
order by quantity DESC
limit 5;
```



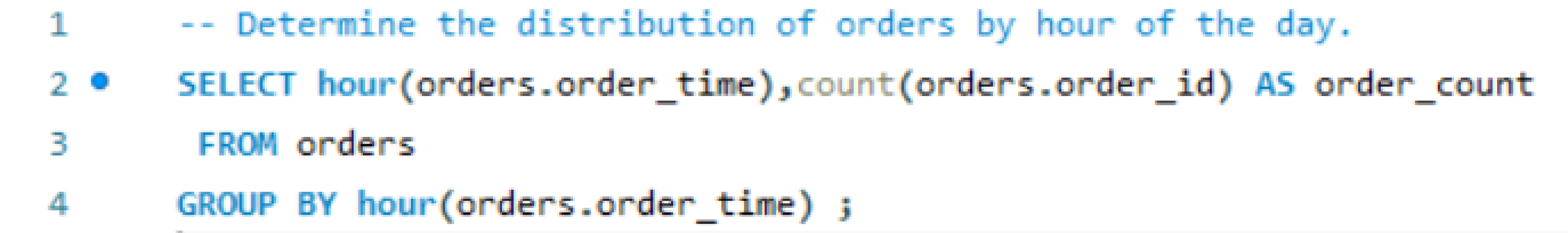
name	quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371



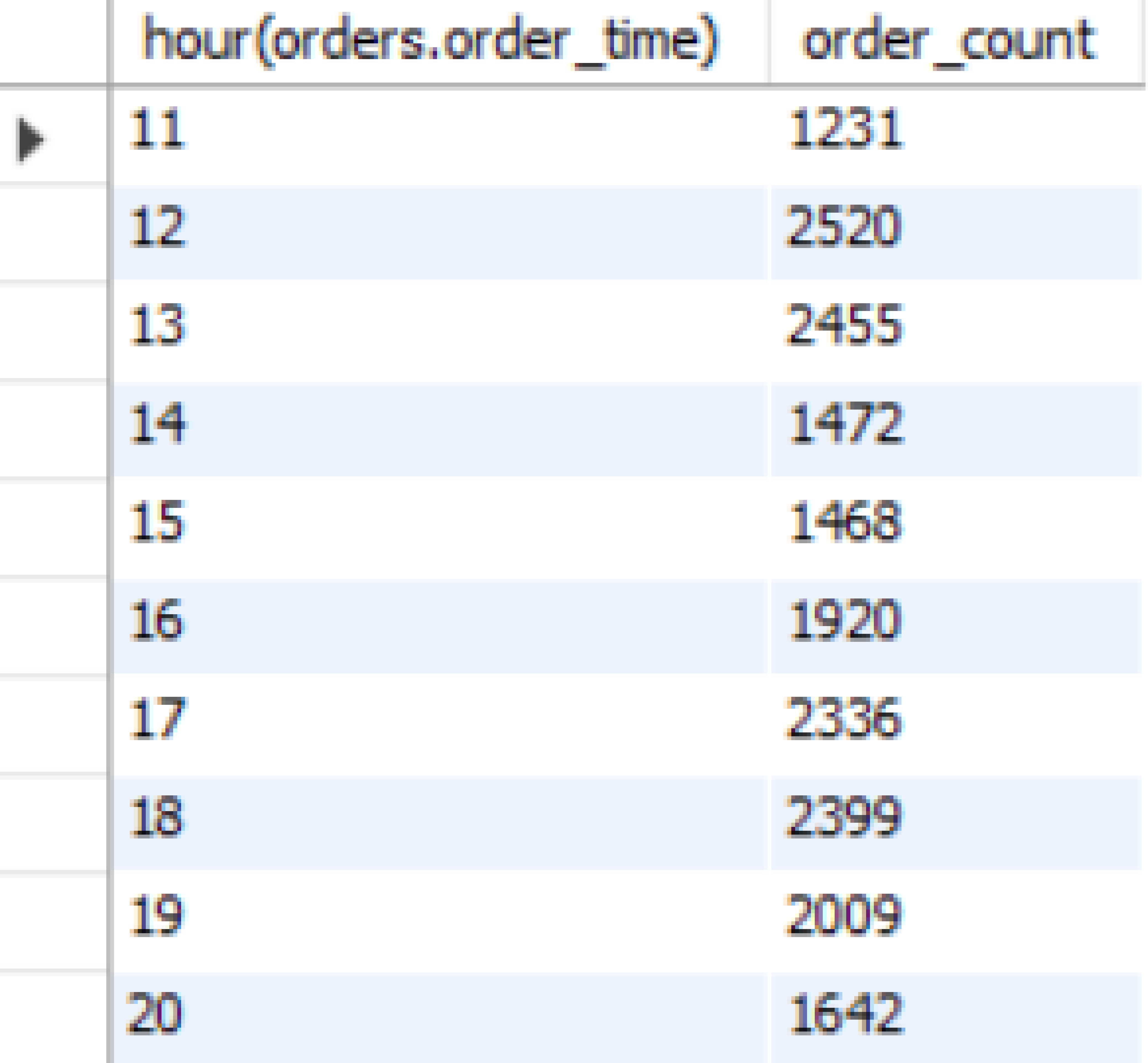
```sql
-- Join the necessary tables to find the total quantity of each pizza category ordered.
SELECT pizza_types.category,sum(order_details.quantity) as Total_quantity
from pizza_types
join pizzas
on pizza_types.pizza_type_id=pizzas.pizza_type_id
join order_details
on pizzas.pizza_id=order_details.pizza_id
GROUP BY pizza_types.category;
```



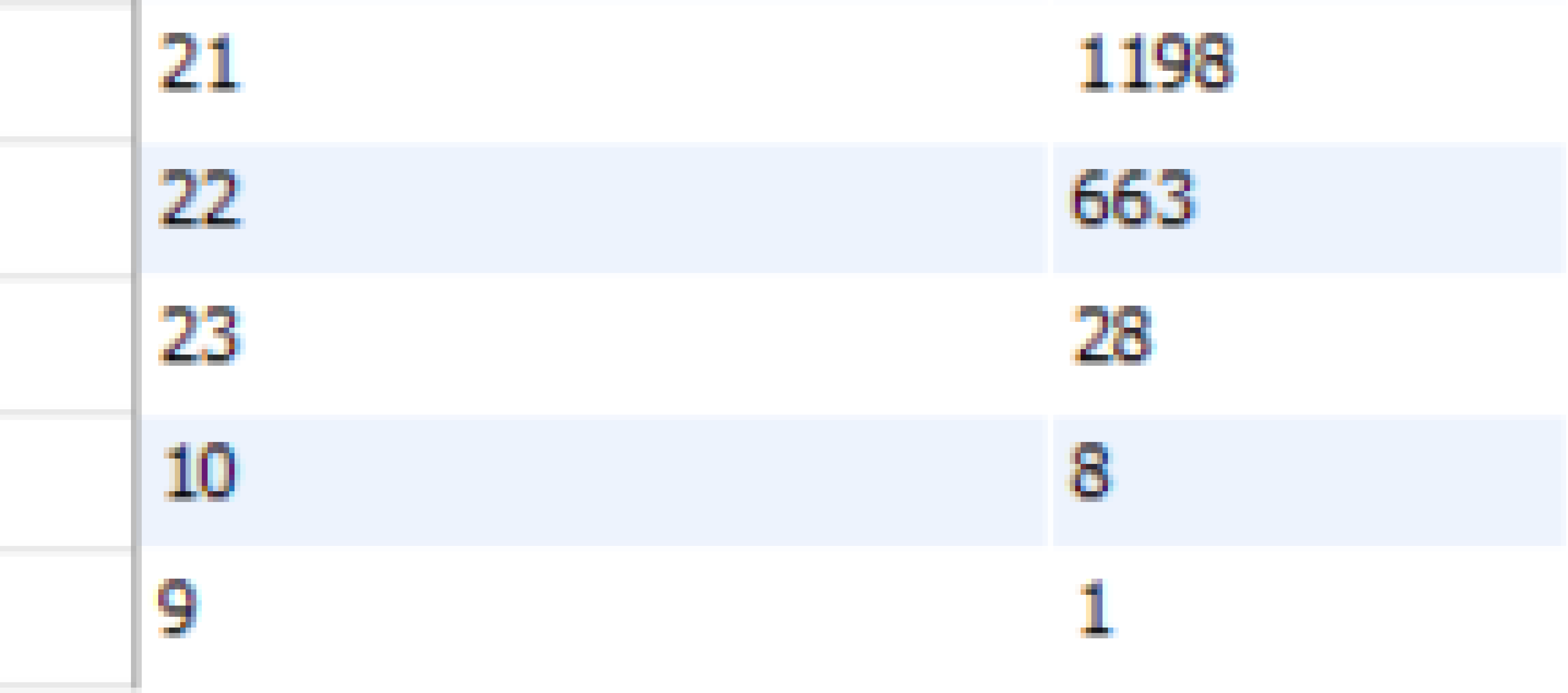
category	Total_quantity
Classic	14888
Veggie	11649
Supreme	11987
Chicken	11050



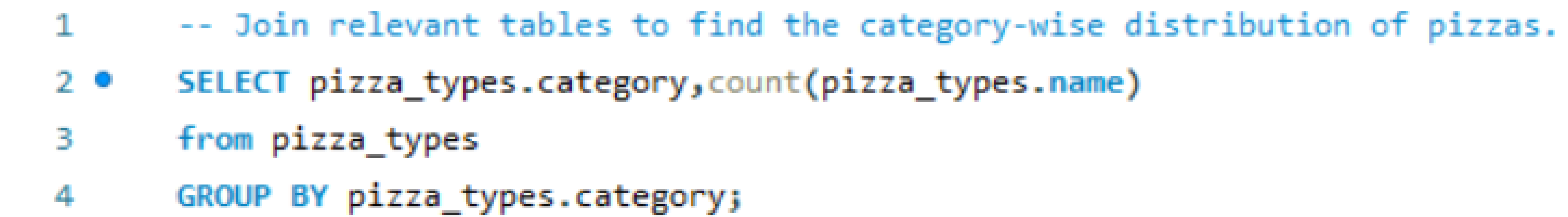
```sql
-- Determine the distribution of orders by hour of the day.
SELECT hour(orders.order_time),count(orders.order_id) AS order_count
 FROM orders
GROUP BY hour(orders.order_time) ;
```



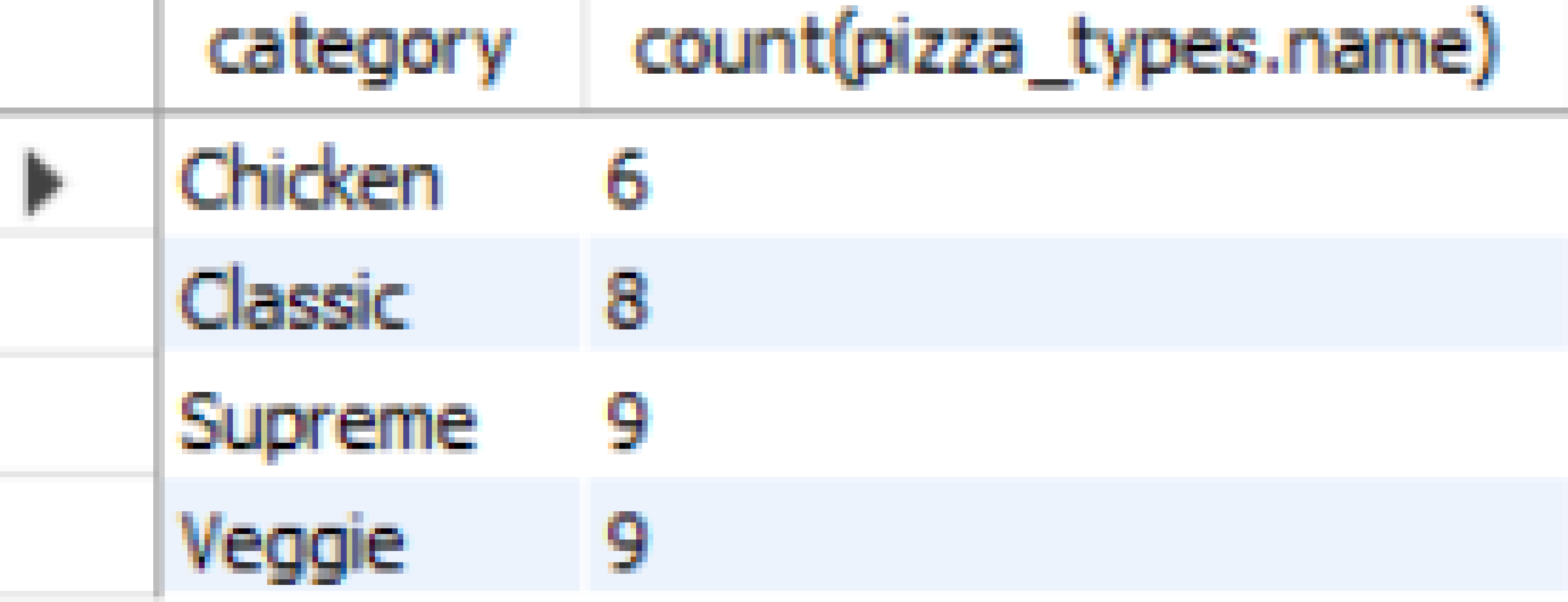
hour(orders.order_time)	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642



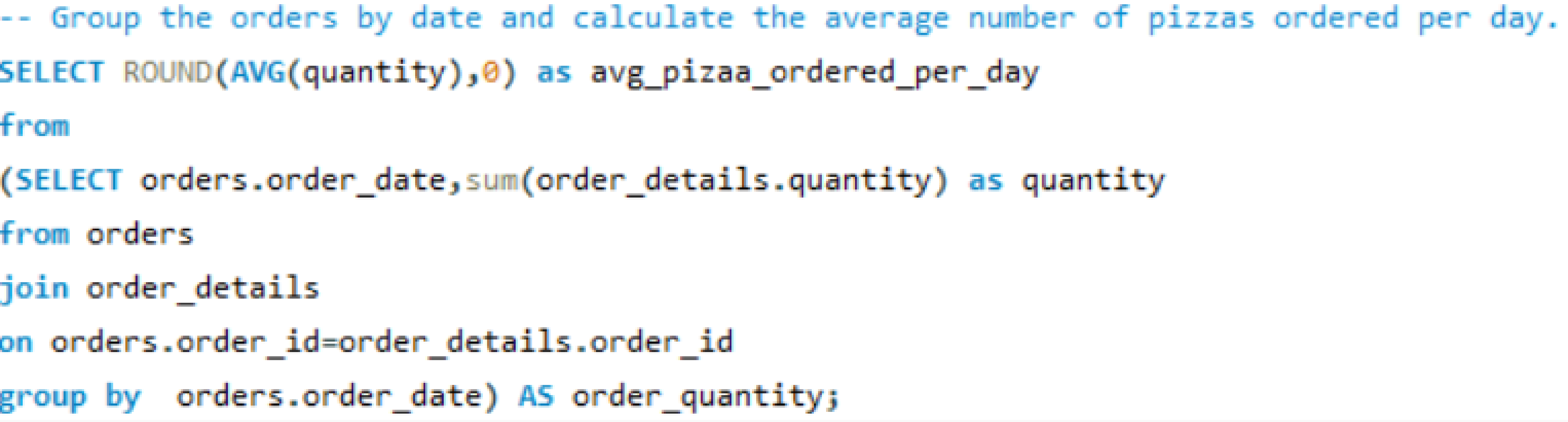
	21	1198
	22	663
	23	28
	10	8
	9	1



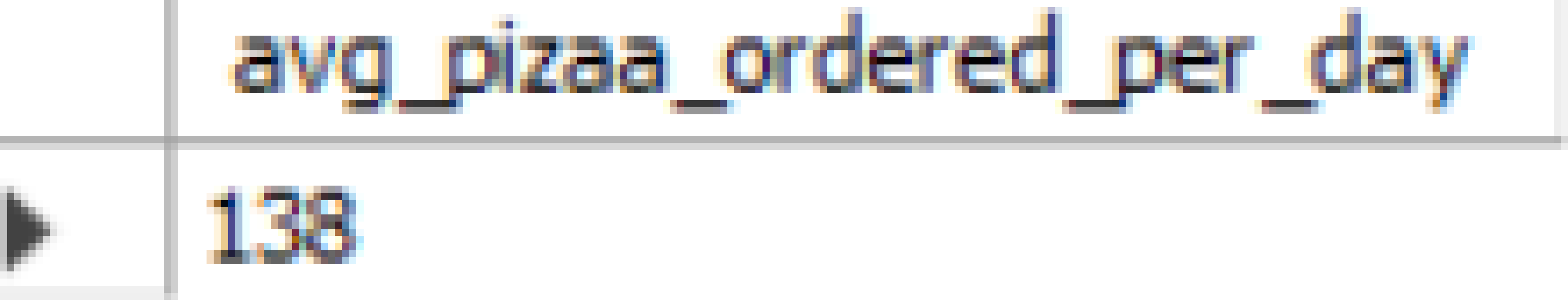
```sql
-- Join relevant tables to find the category-wise distribution of pizzas.
SELECT pizza_types.category,count(pizza_types.name)
from pizza_types
GROUP BY pizza_types.category;
```



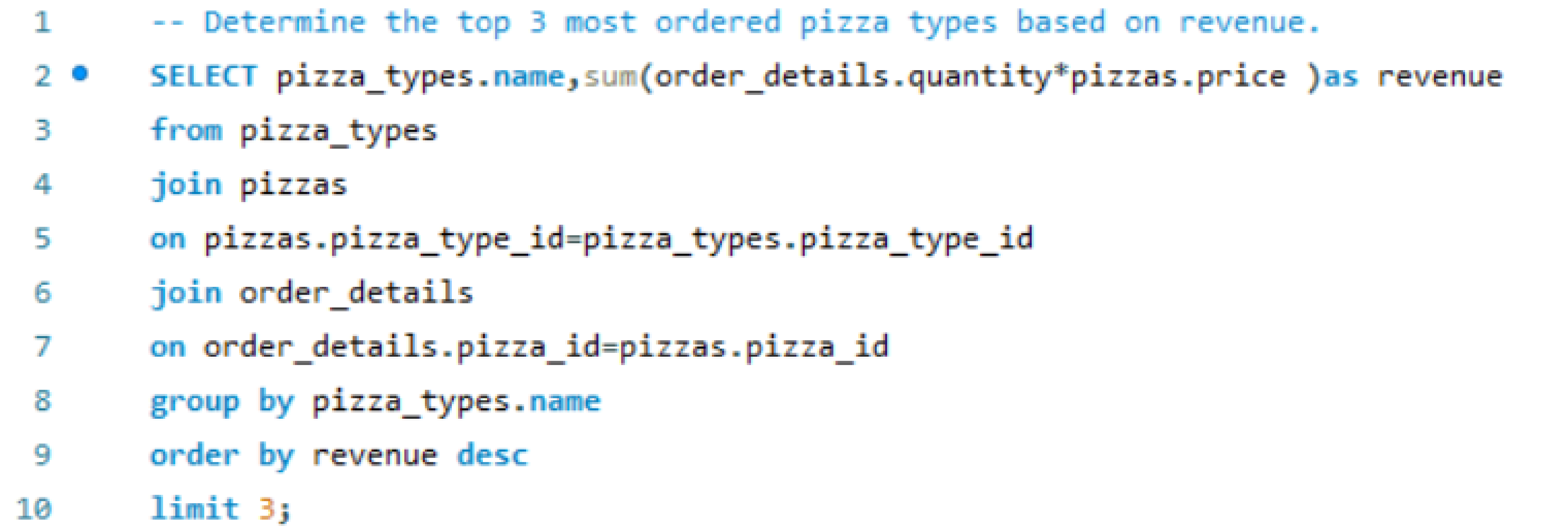
category	count(pizza_types.name)
Chicken	6
Classic	8
Supreme	9
Veggie	9



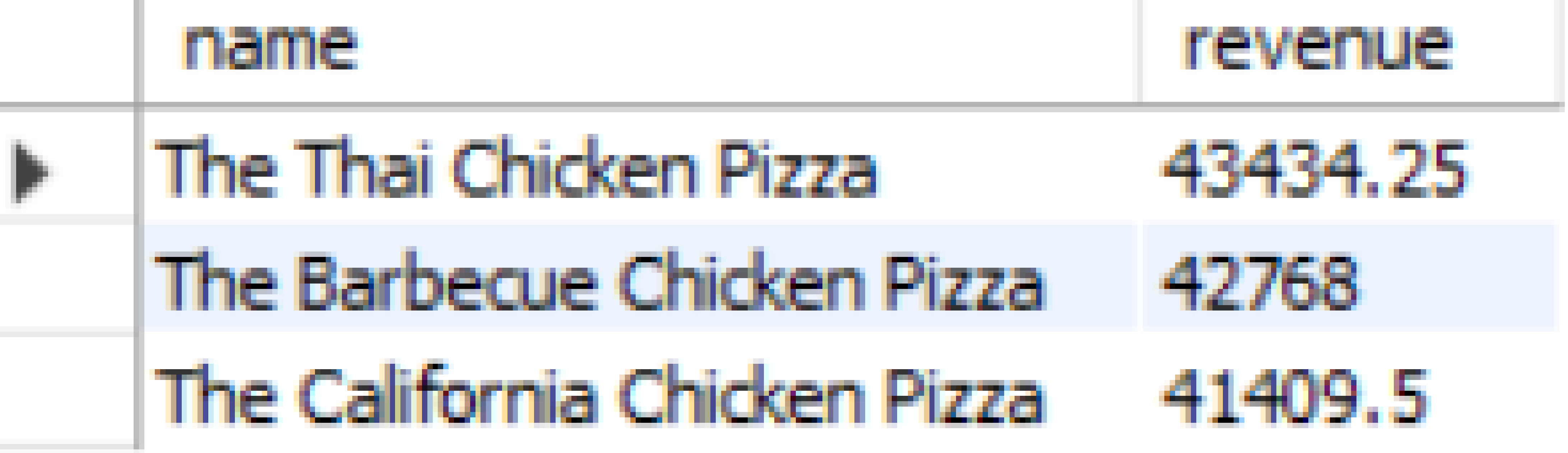
```sql
-- Group the orders by date and calculate the average number of pizzas ordered per day.
SELECT ROUND(AVG(quantity),0) as avg_pizaa_ordered_per_day
from
(SELECT orders.order_date,sum(order_details.quantity) as quantity
from orders
join order_details
on orders.order_id=order_details.order_id
group by  orders.order_date) AS order_quantity;
```



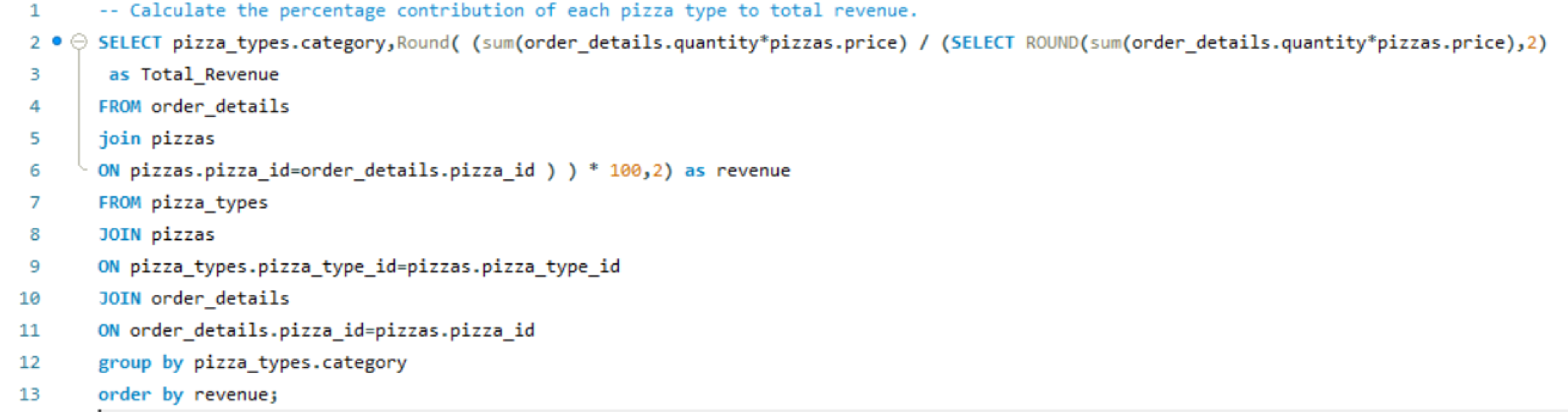
avg_pizaa_ordered_per_day
138



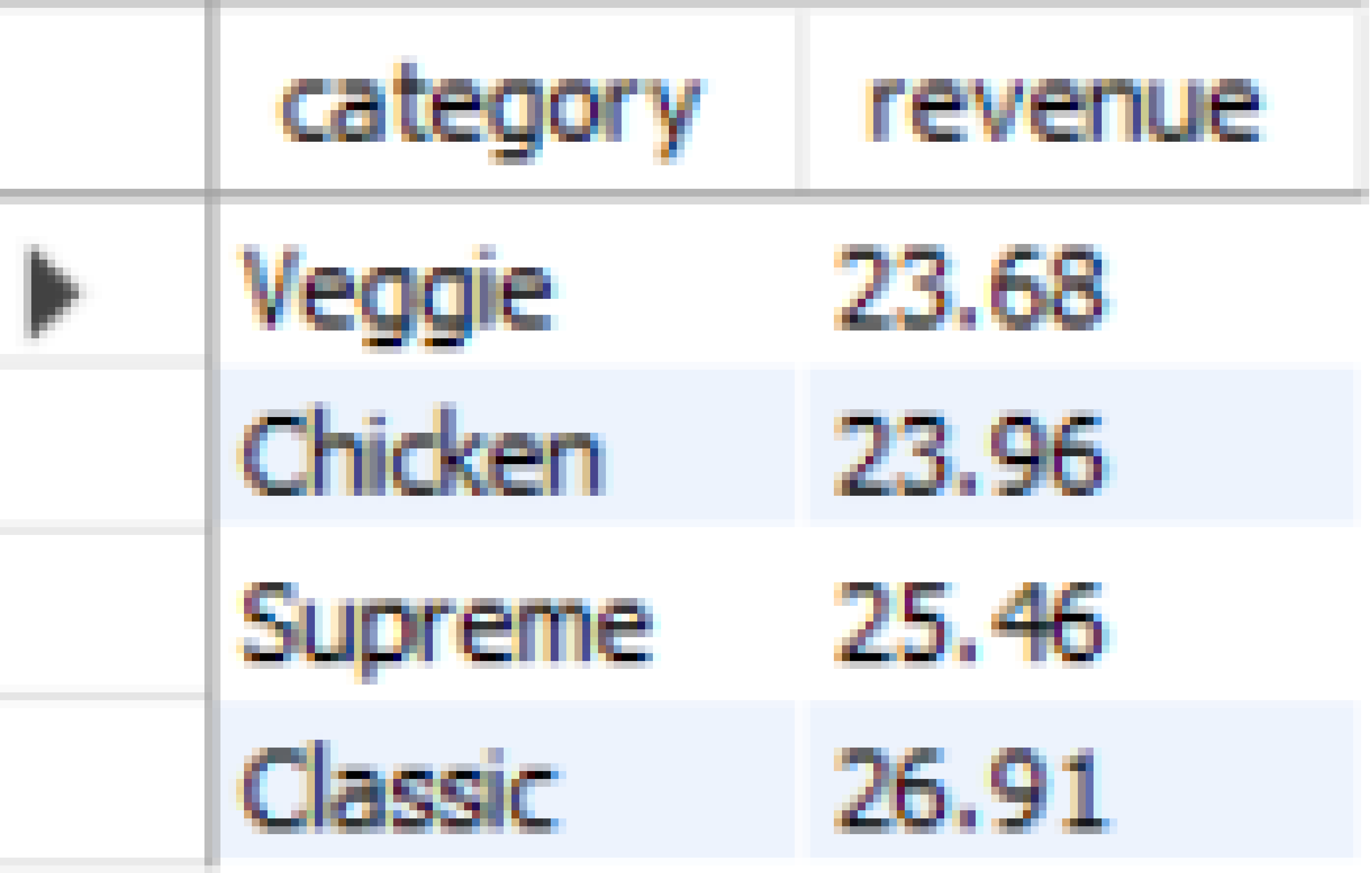
```sql
-- Determine the top 3 most ordered pizza types based on revenue.
SELECT pizza_types.name,sum(order_details.quantity*pizzas.price )as revenue
from pizza_types
join pizzas
on pizzas.pizza_type_id=pizza_types.pizza_type_id
join order_details
on order_details.pizza_id=pizzas.pizza_id
group by pizza_types.name
order by revenue desc
limit 3;
```



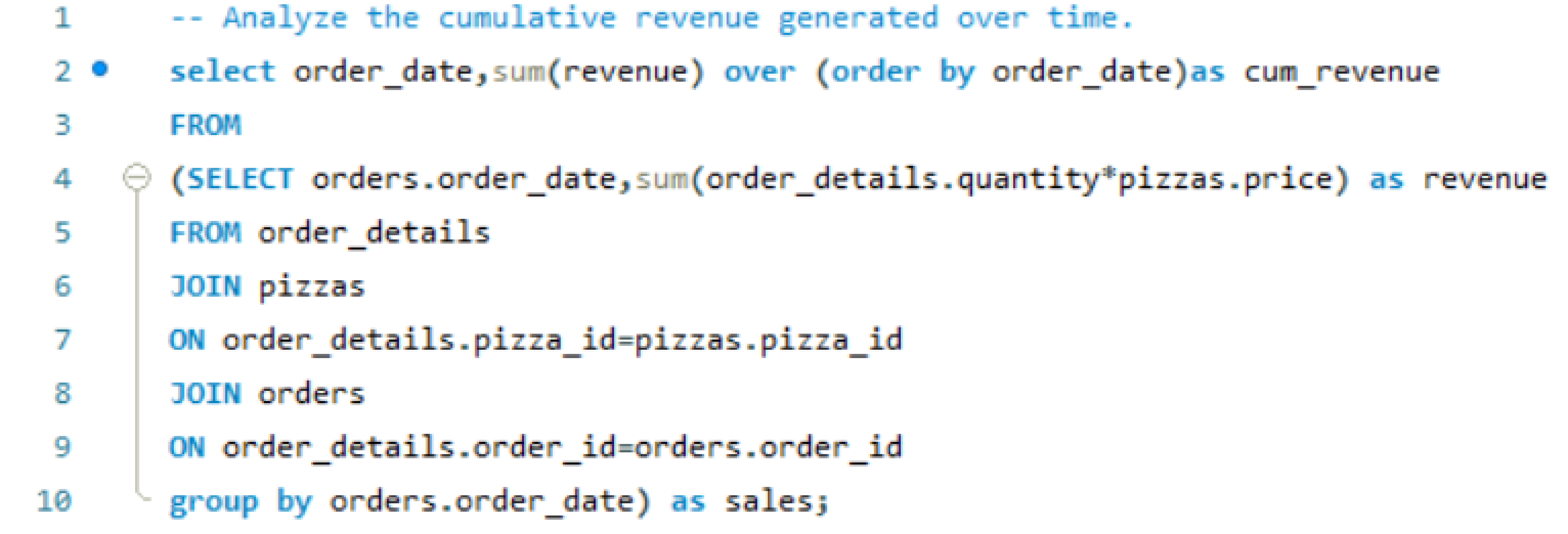
name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5



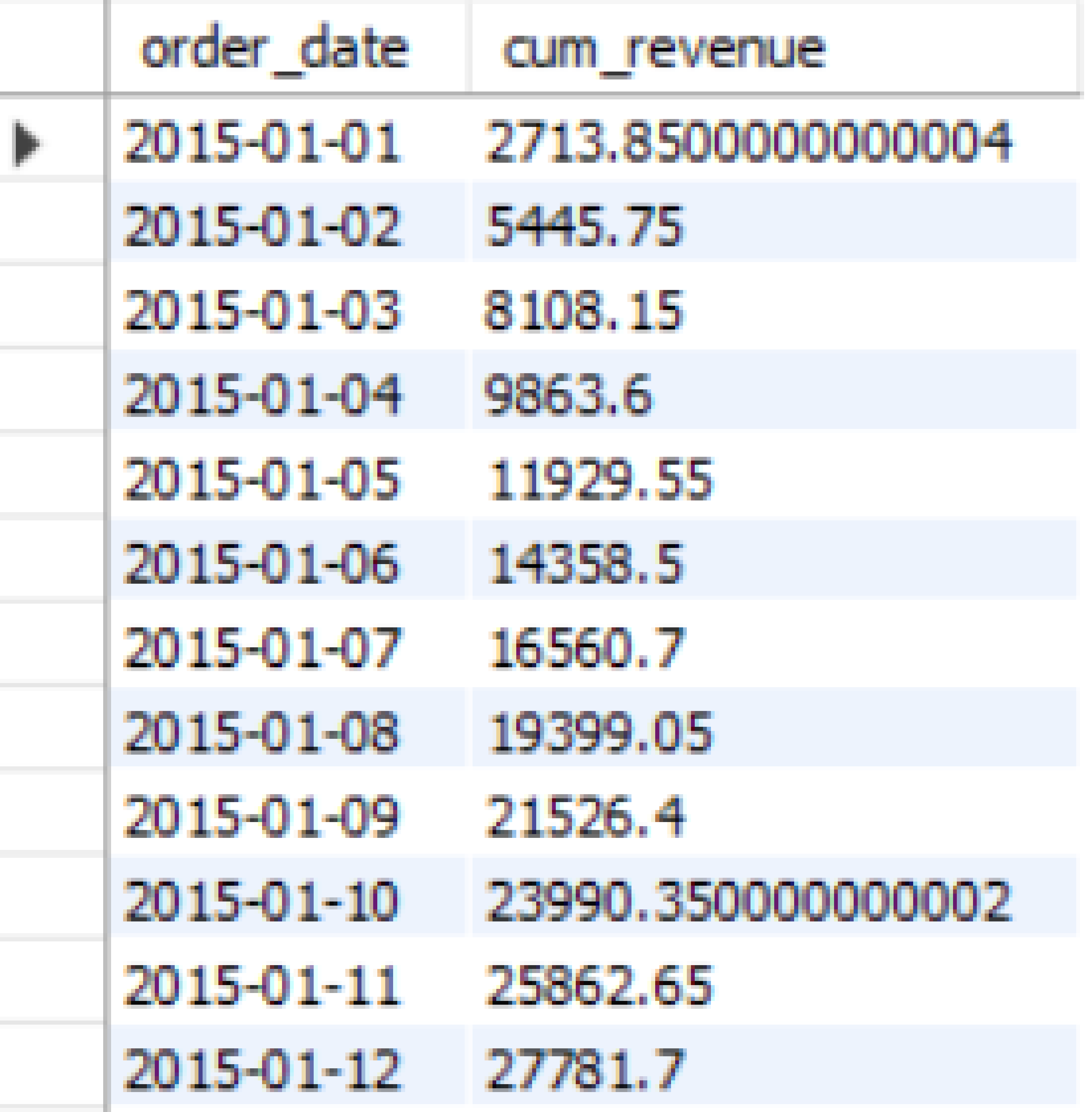
```
-- Calculate the percentage contribution of each pizza type to total revenue.
SELECT pizza_types.category,Round( (sum(order_details.quantity*pizzas.price) / (SELECT ROUND(sum(order_details.quantity*pizzas.price),2)
 as Total_Revenue
FROM order_details
join pizzas
ON pizzas.pizza_id=order_details.pizza_id ) ) * 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;
```



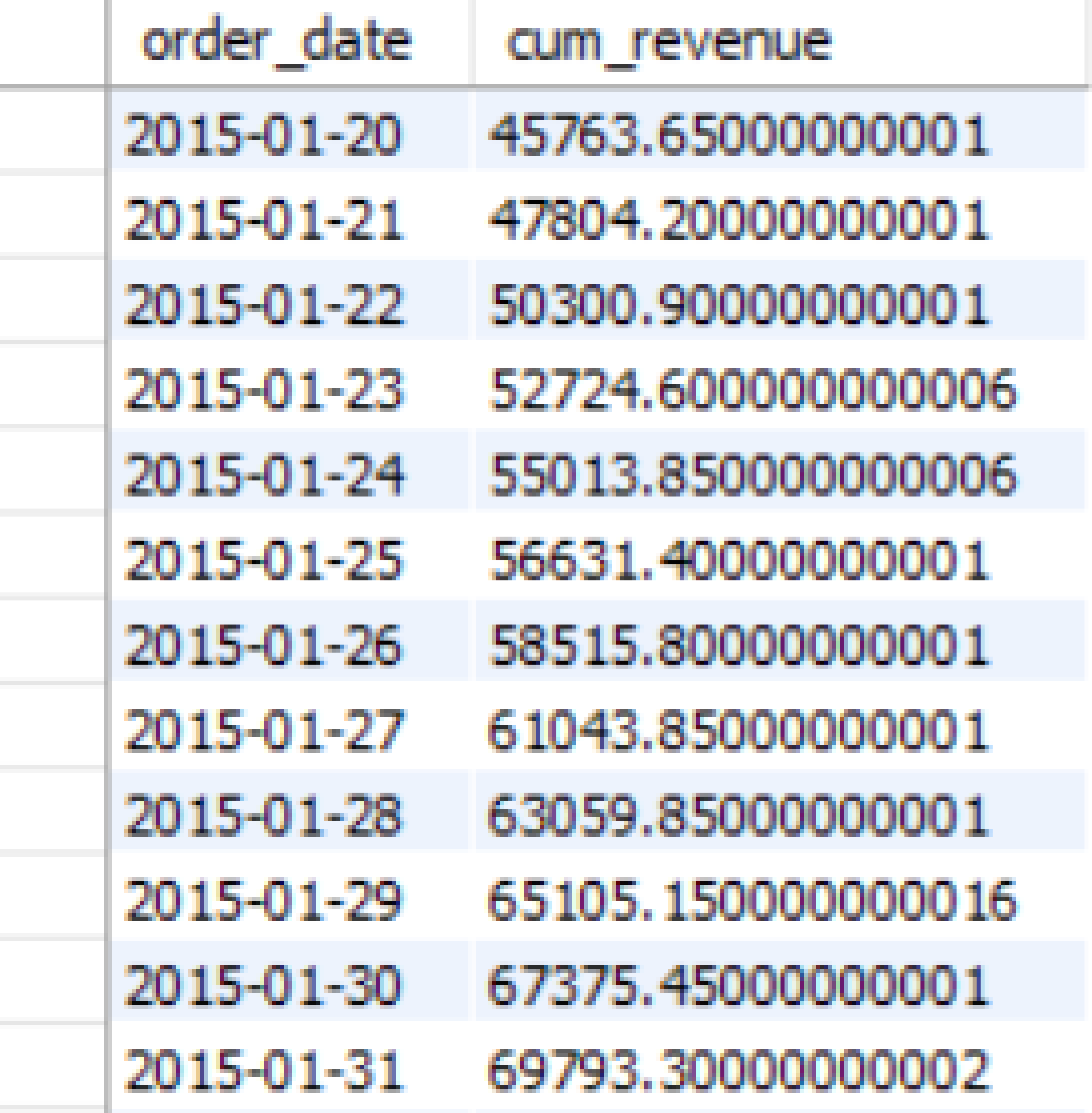
	category	revenue
▶	Veggie	23.68
	Chicken	23.96
	Supreme	25.46
	Classic	26.91



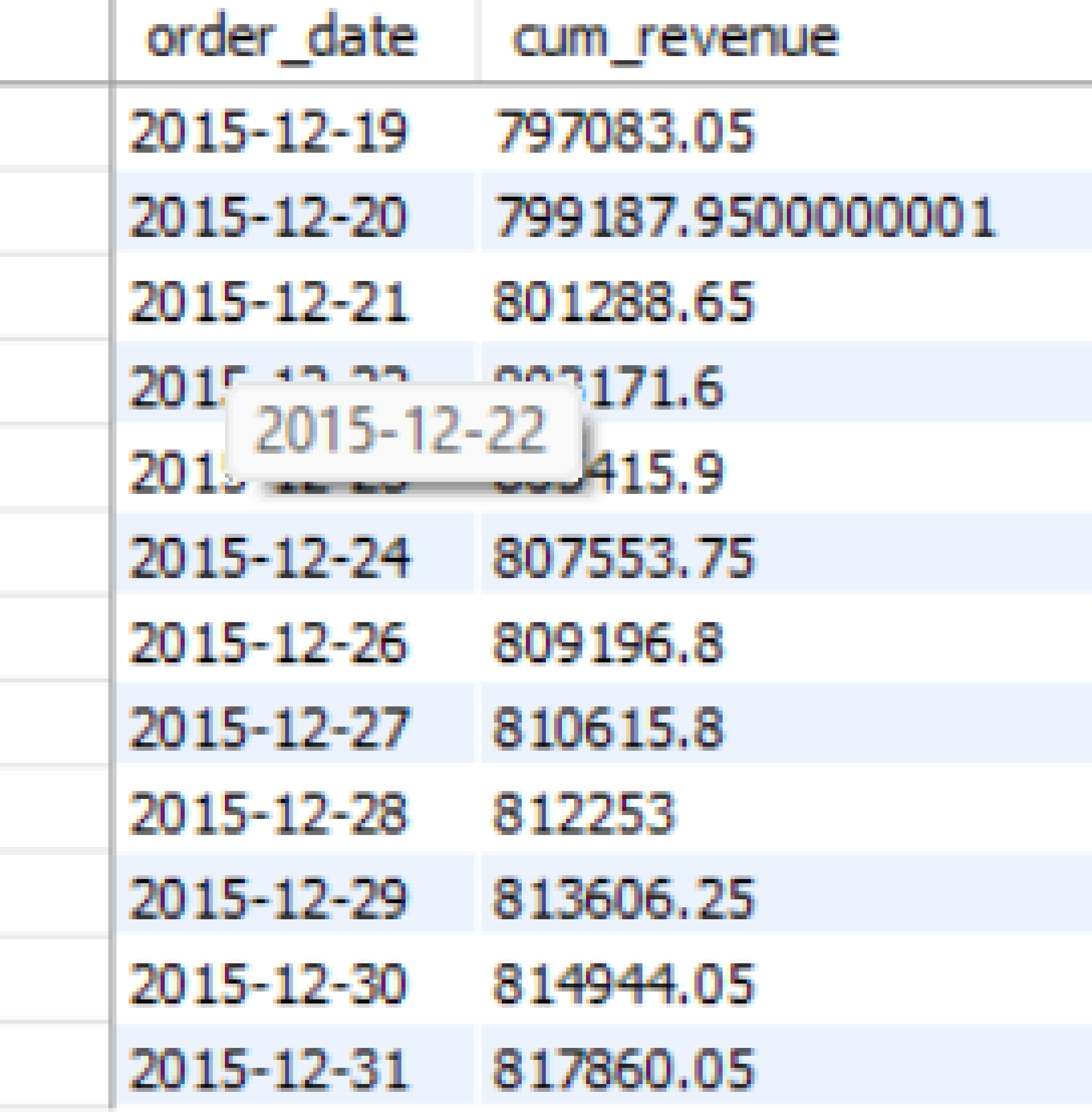
```sql
-- Analyze the cumulative revenue generated over time.
select order_date,sum(revenue) over (order by order_date)as cum_revenue
FROM
(SELECT orders.order_date,sum(order_details.quantity*pizzas.price) as revenue
FROM order_details
JOIN pizzas
ON order_details.pizza_id=pizzas.pizza_id
JOIN orders
ON order_details.order_id=orders.order_id
group by orders.order_date) as sales;
```



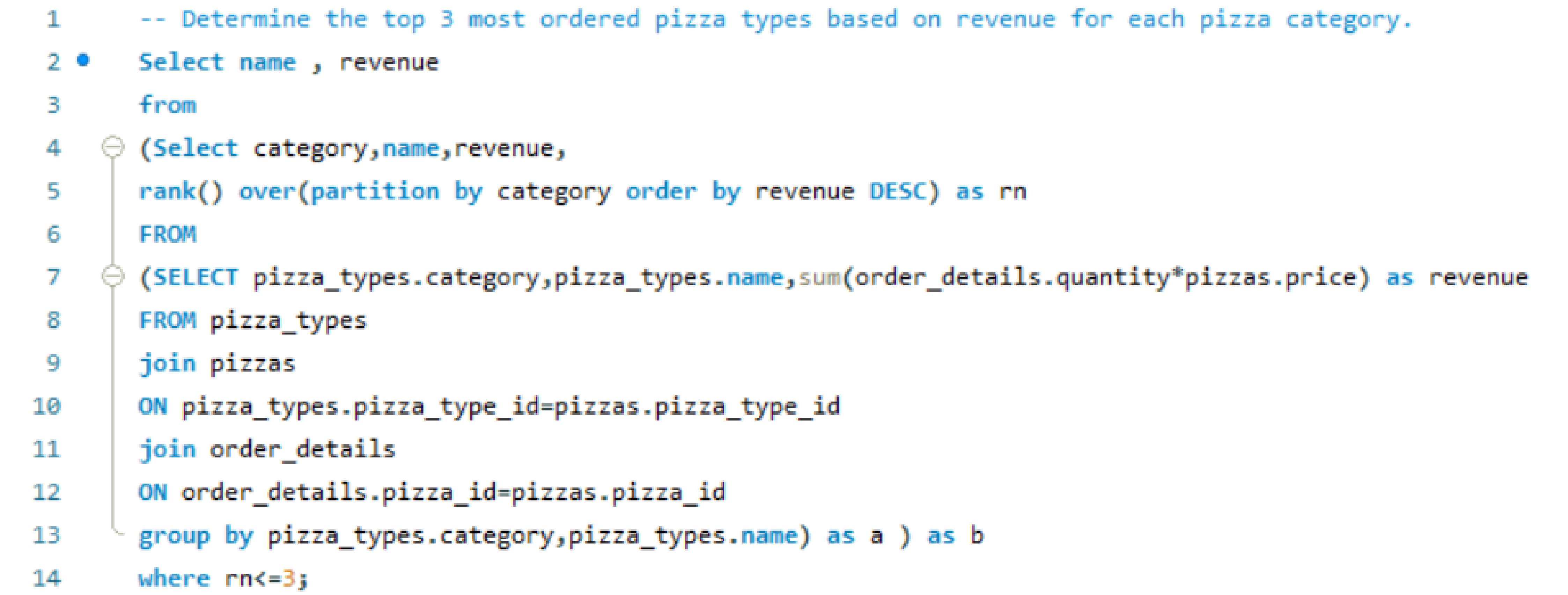
order_date	cum_revenue
2015-01-01	2713.8500000000004
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55
2015-01-06	14358.5
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526.4
2015-01-10	23990.350000000002
2015-01-11	25862.65
2015-01-12	27781.7



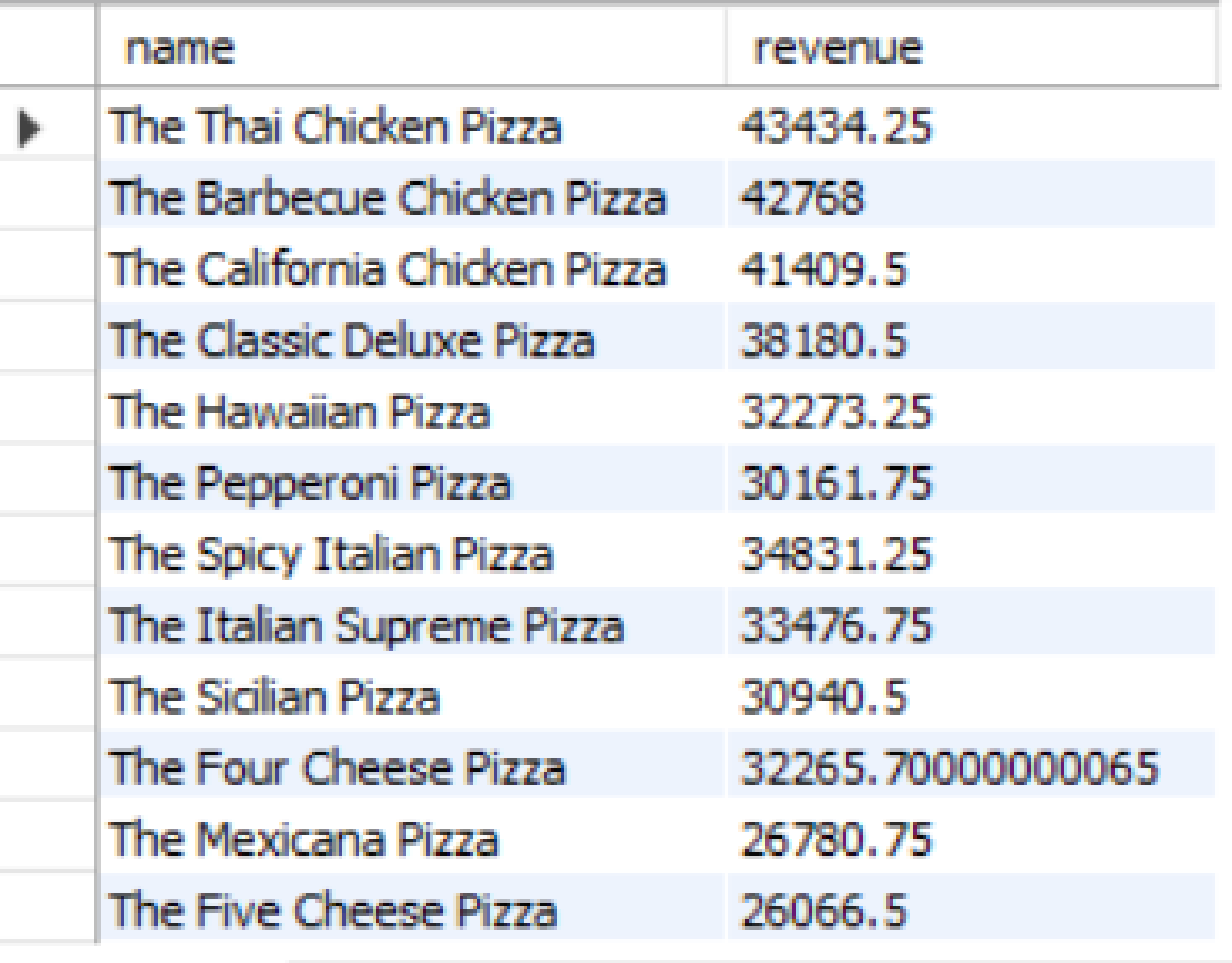
order_date	cum_revenue
2015-01-20	45763.65000000001
2015-01-21	47804.20000000001
2015-01-22	50300.90000000001
2015-01-23	52724.600000000006
2015-01-24	55013.850000000006
2015-01-25	56631.40000000001
2015-01-26	58515.80000000001
2015-01-27	61043.85000000001
2015-01-28	63059.85000000001
2015-01-29	65105.150000000016
2015-01-30	67375.45000000001
2015-01-31	69793.30000000002



order_date	cum_revenue
2015-12-19	797083.05
2015-12-20	799187.9500000001
2015-12-21	801288.65
2015-12-22	803171.6
2015-12-23	80[illegible]415.9
2015-12-24	807553.75
2015-12-26	809196.8
2015-12-27	810615.8
2015-12-28	812253
2015-12-29	813606.25
2015-12-30	814944.05
2015-12-31	817860.05



```sql
-- 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