THE PIZZA SALES ANALYSIS

PROJECT OVERVIEW : PIZZA SALES ANALYSIS USING SQL

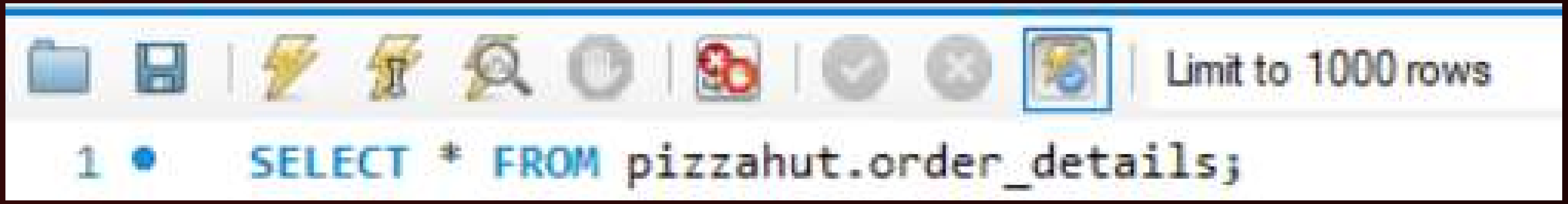
In this project, I carried out a comprehensive analysis of a Kaggle pizza sales dataset using SQL. The goal was to extract meaningful insights through a series of exploratory tasks ranging from basic summaries to more advanced analytics. I began by answering foundational questions such as the total number of orders, overall revenue, and identifying the most expensive pizza. Building on that, I explored deeper patterns—such as when customers are most likely to place orders during the day, how revenue accumulates over time, and which pizza categories contribute most significantly to overall sales. This end-to-end analysis highlighted key sales trends and customer behavior, offering a data-driven view of pizza performance.

🧾 About the Project

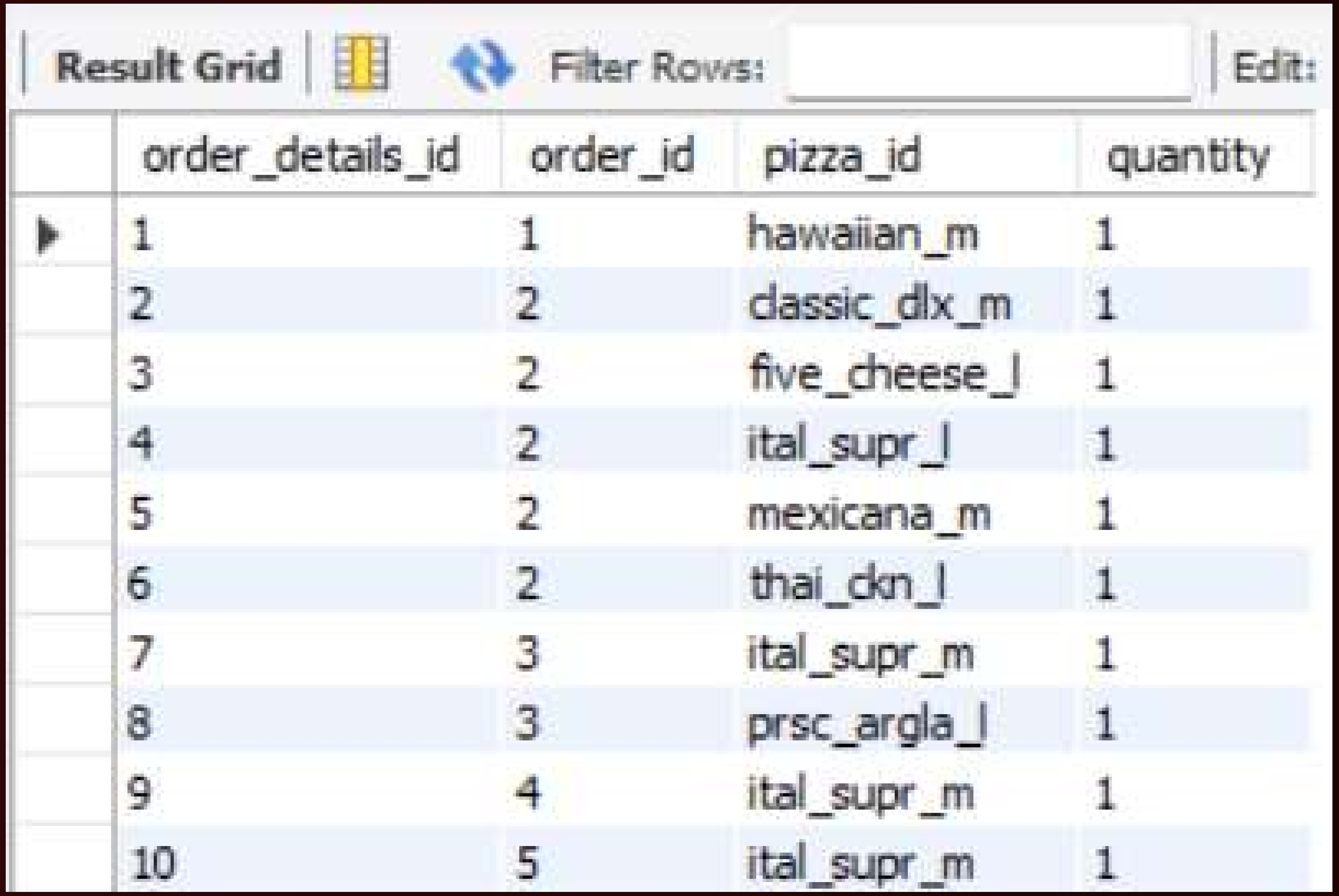
The objective of this project was to apply SQL for exploratory data analysis (EDA) on a pizza sales dataset. Using a combination of table joins and well-structured queries, I uncovered key insights and trends hidden within the data. Initial queries focused on capturing essential metrics such as total revenue and number of orders. As the analysis progressed, more complex queries revealed customer ordering habits, pizza category preferences, and revenue breakdowns over time. This project showcases how SQL can be effectively used to analyze structured data and uncover patterns that can inform sales and marketing decisions in the food industry.

🧾 Snapshot of the order_details Table

The table below shows a glimpse of the raw data used in the analysis. It contains information about individual pizza orders, including the pizza ID and quantity ordered, and serves as a key table for calculating sales metrics and trends.



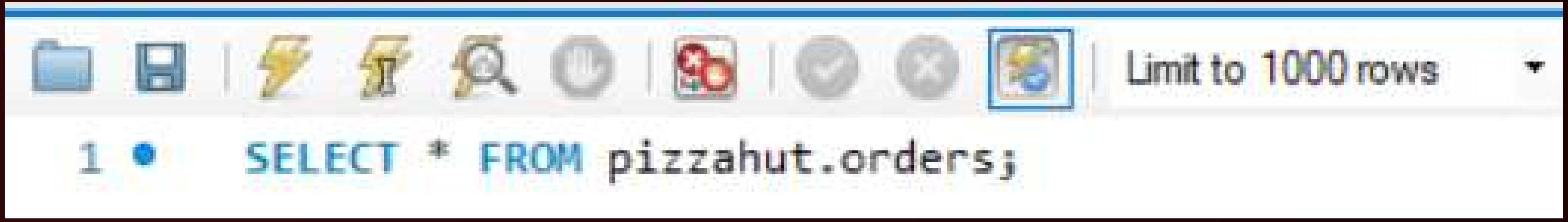
```sql
SELECT * FROM pizzahut.order_details;
```



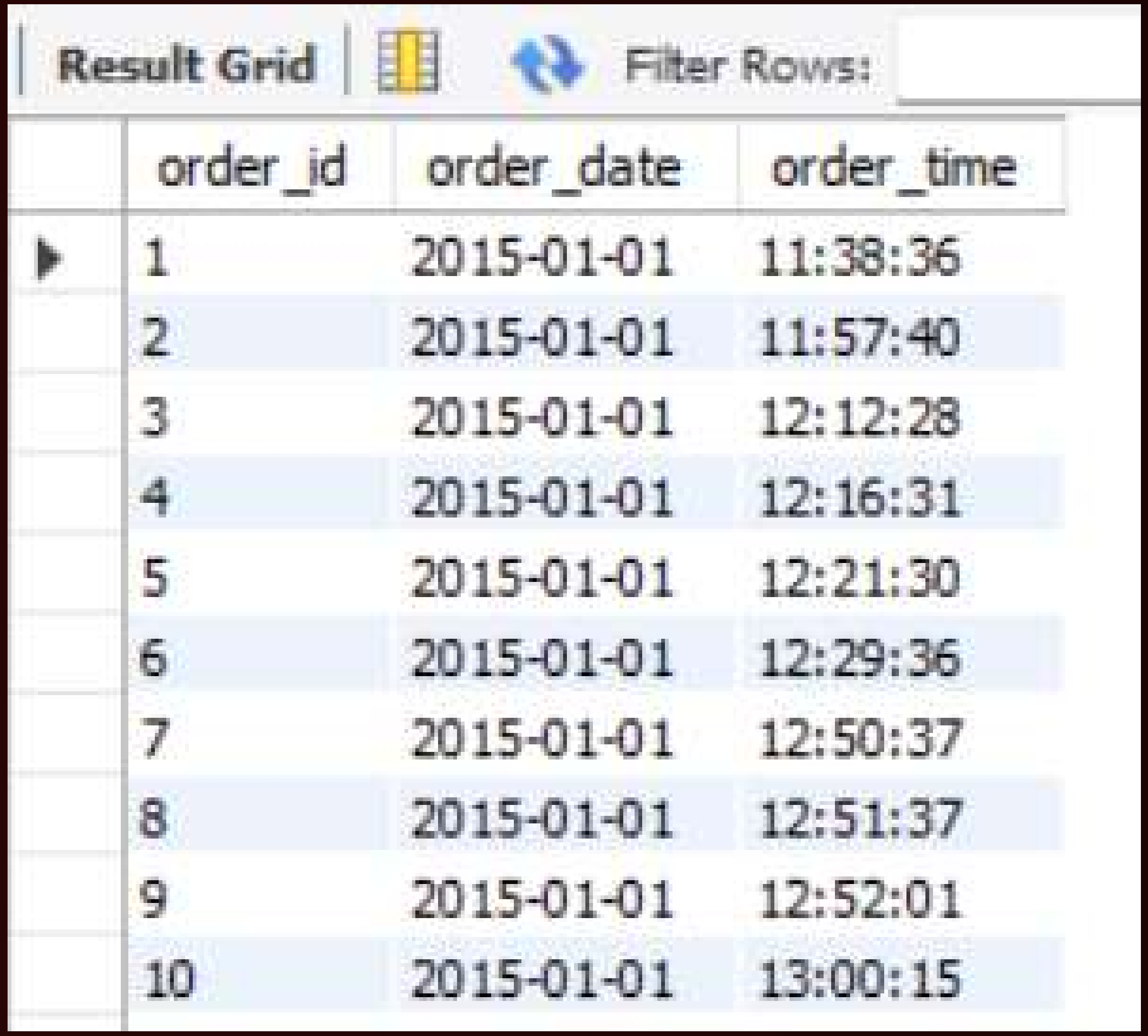
order_details_id	order_id	pizza_id	quantity
1	1	hawaiian_m	1
2	2	classic_dlx_m	1
3	2	five_cheese_l	1
4	2	ital_supr_l	1
5	2	mexicana_m	1
6	2	thai_ckn_l	1
7	3	ital_supr_m	1
8	3	prsc_argla_l	1
9	4	ital_supr_m	1
10	5	ital_supr_m	1

📋 Snapshot of the orders Table

This table provides details about each customer order, including the unique order ID and the corresponding date and time. It plays a crucial role in analyzing order volume, peak hours, and sales trends over time.



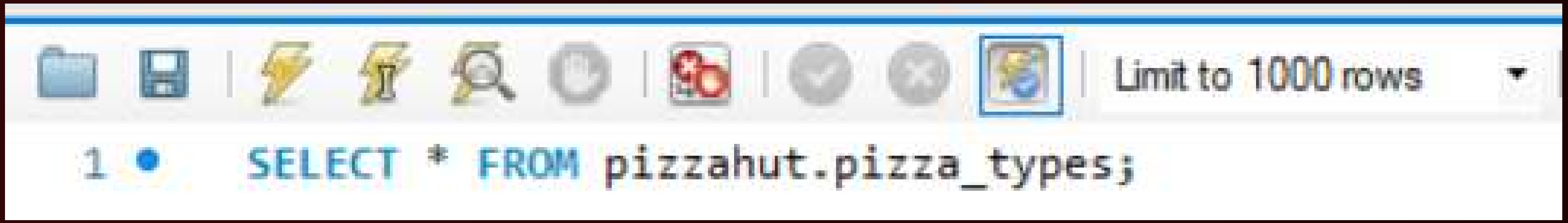
```
SELECT * FROM pizzahut.orders;
```



order_id	order_date	order_time
1	2015-01-01	11:38:36
2	2015-01-01	11:57:40
3	2015-01-01	12:12:28
4	2015-01-01	12:16:31
5	2015-01-01	12:21:30
6	2015-01-01	12:29:36
7	2015-01-01	12:50:37
8	2015-01-01	12:51:37
9	2015-01-01	12:52:01
10	2015-01-01	13:00:15

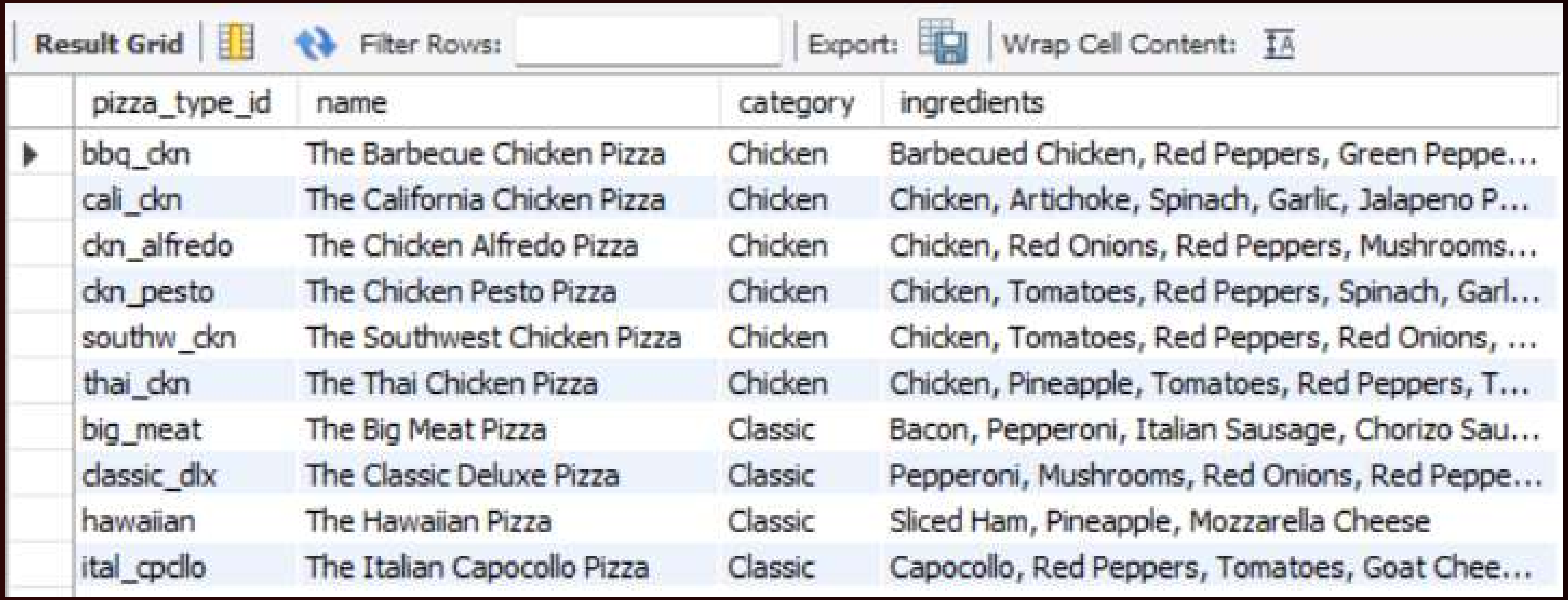
🍕 Snapshot of the pizza_types Table

This table contains descriptive information about each type of pizza, including its name, category (such as classic, veggie, or chicken), and a list of ingredients. It helps in categorizing pizzas and understanding customer preferences based on type and composition.



Limit to 1000 rows

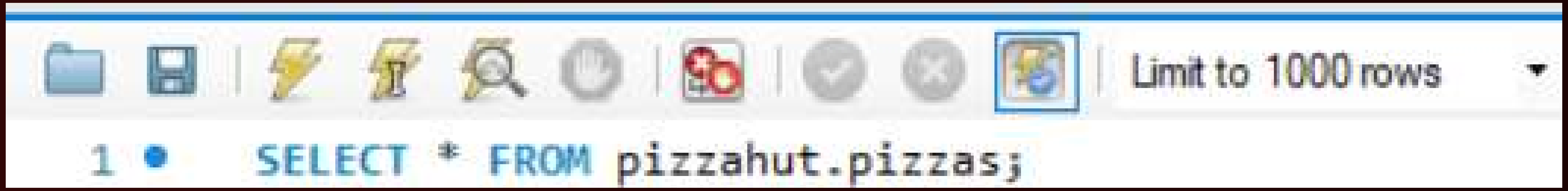
```
SELECT * FROM pizzahut.pizza_types;
```



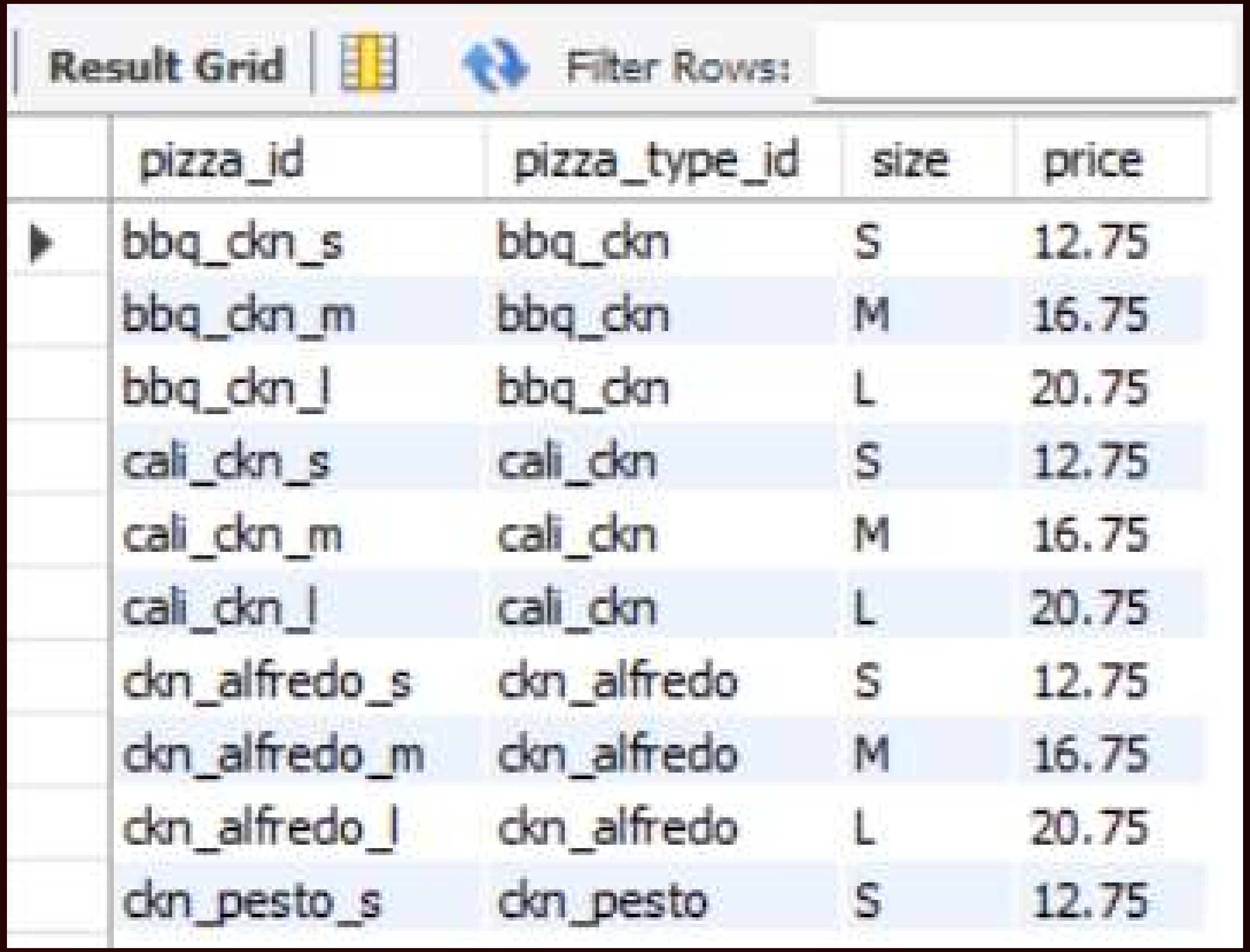
pizza_type_id	name	category	ingredients
bbq_ckn	The Barbecue Chicken Pizza	Chicken	Barbecued Chicken, Red Peppers, Green Peppe...
cali_ckn	The California Chicken Pizza	Chicken	Chicken, Artichoke, Spinach, Garlic, Jalapeno P...
ckn_alfredo	The Chicken Alfredo Pizza	Chicken	Chicken, Red Onions, Red Peppers, Mushrooms...
ckn_pesto	The Chicken Pesto Pizza	Chicken	Chicken, Tomatoes, Red Peppers, Spinach, Garl...
southw_ckn	The Southwest Chicken Pizza	Chicken	Chicken, Tomatoes, Red Peppers, Red Onions, ...
thai_ckn	The Thai Chicken Pizza	Chicken	Chicken, Pineapple, Tomatoes, Red Peppers, T...
big_meat	The Big Meat Pizza	Classic	Bacon, Pepperoni, Italian Sausage, Chorizo Sau...
classic_dlx	The Classic Deluxe Pizza	Classic	Pepperoni, Mushrooms, Red Onions, Red Peppe...
hawaiian	The Hawaiian Pizza	Classic	Sliced Ham, Pineapple, Mozzarella Cheese
ital_cpdlo	The Italian Capocollo Pizza	Classic	Capocollo, Red Peppers, Tomatoes, Goat Chee...

Snapshot of the pizzas Table

The pizzas table links each pizza to its type and includes pricing details for various sizes. This table is essential for revenue calculations and understanding how pricing varies across different pizza types and sizes.

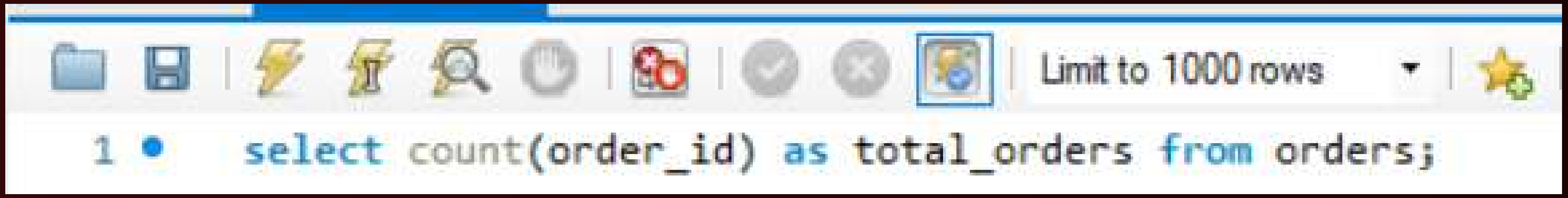


```sql
SELECT * FROM pizzahut.pizzas;
```


Result Grid
Filter Rows:

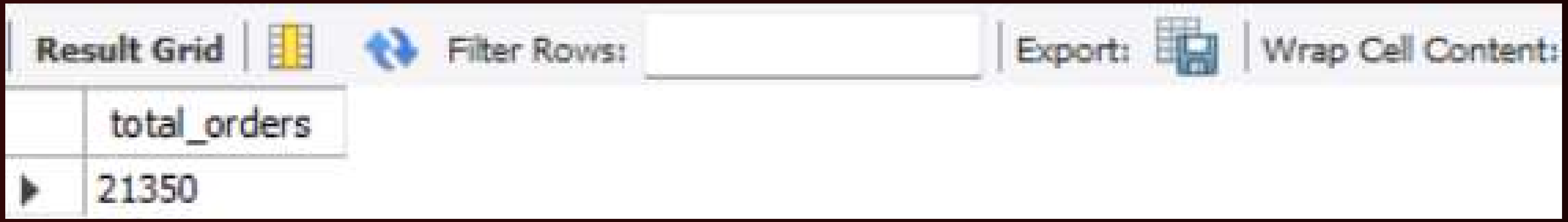
pizza_id	pizza_type_id	size	price
bbq_ckn_s	bbq_ckn	S	12.75
bbq_ckn_m	bbq_ckn	M	16.75
bbq_ckn_l	bbq_ckn	L	20.75
cali_ckn_s	cali_ckn	S	12.75
cali_ckn_m	cali_ckn	M	16.75
cali_ckn_l	cali_ckn	L	20.75
ckn_alfredo_s	ckn_alfredo	S	12.75
ckn_alfredo_m	ckn_alfredo	M	16.75
ckn_alfredo_l	ckn_alfredo	L	20.75
ckn_pesto_s	ckn_pesto	S	12.75

Retrieve the total number of orders placed.



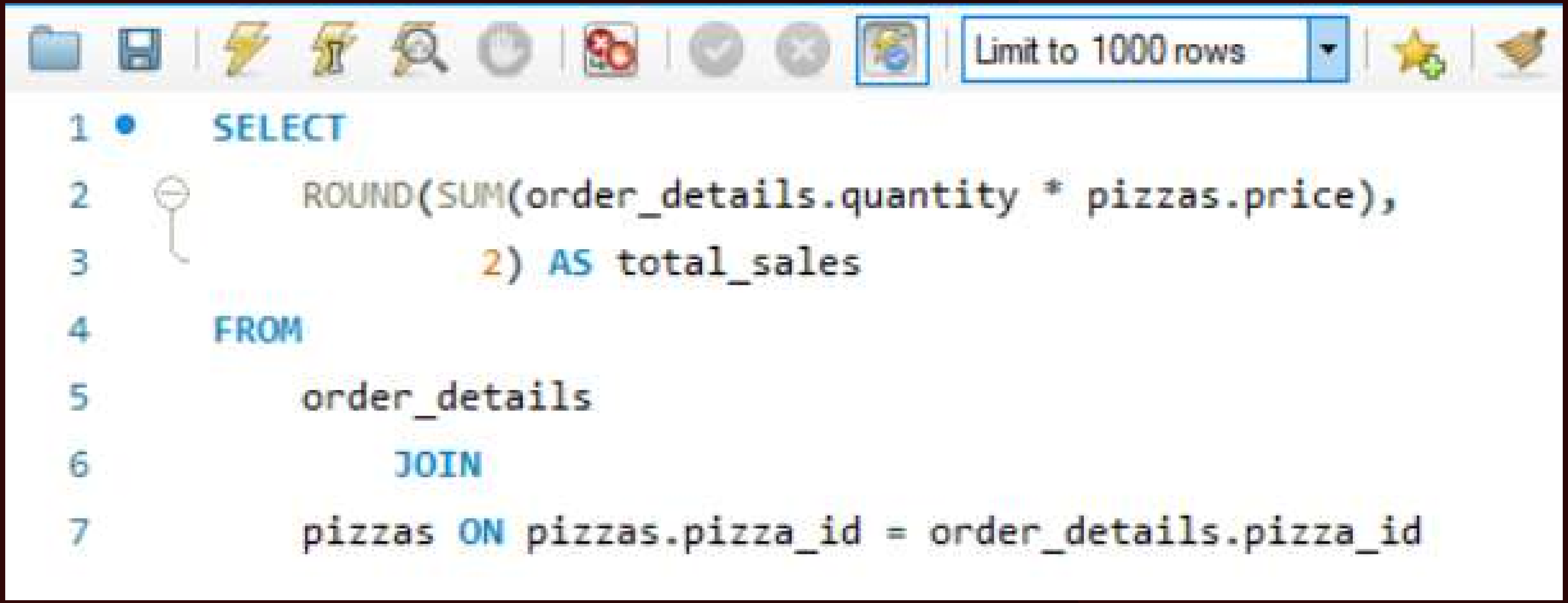
```
select count(order_id) as total_orders from orders;
```

OUTPUT AS:



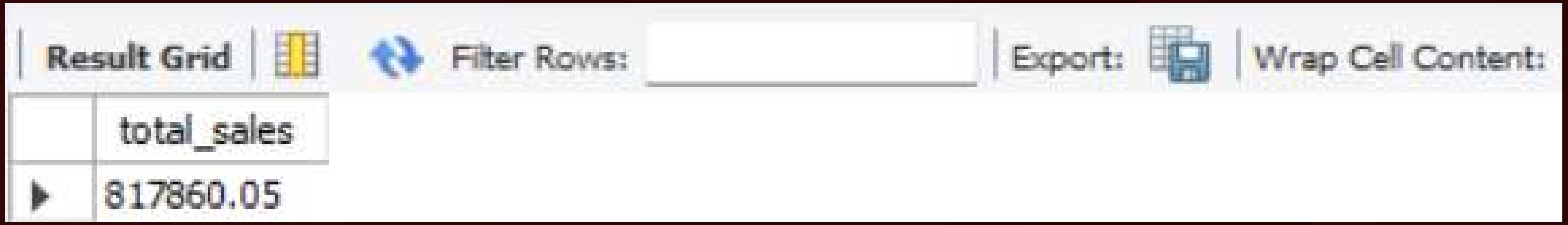
```
total_orders
21350
```

Calculate the total revenue generated from pizza sales.



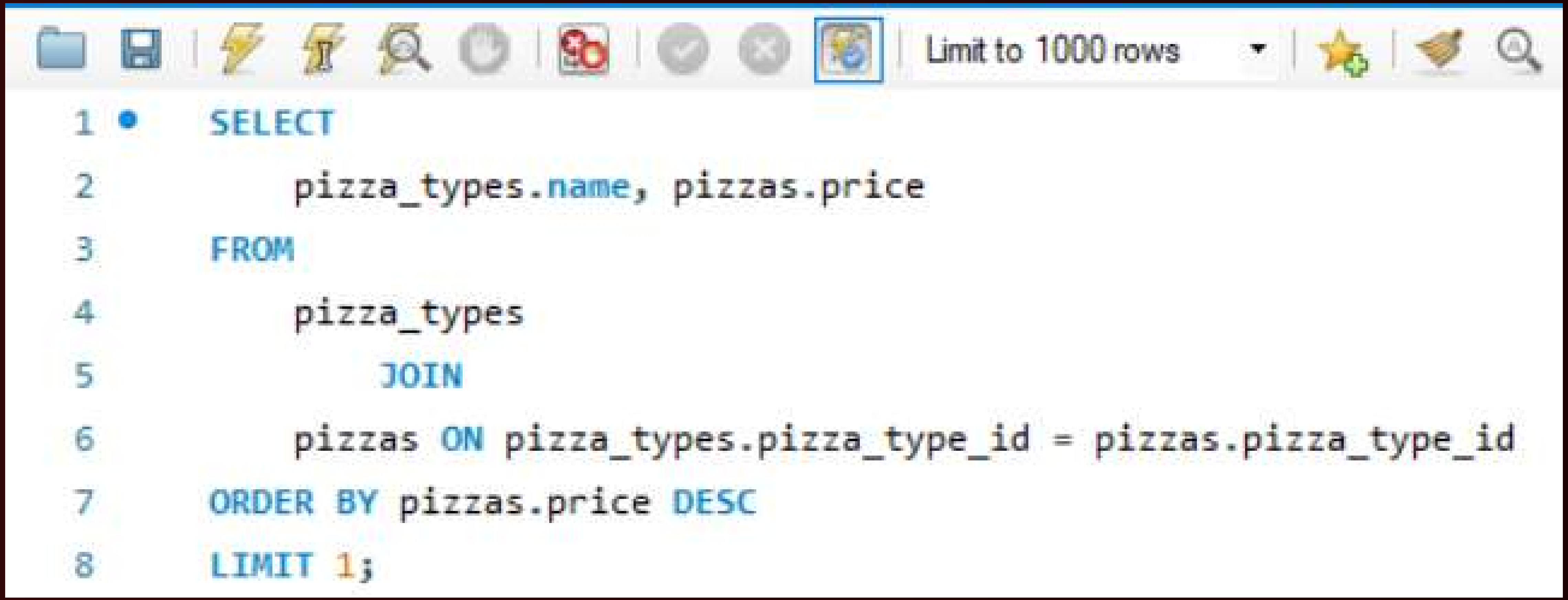
```sql
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS total_sales
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id
```

OUTPUT AS:



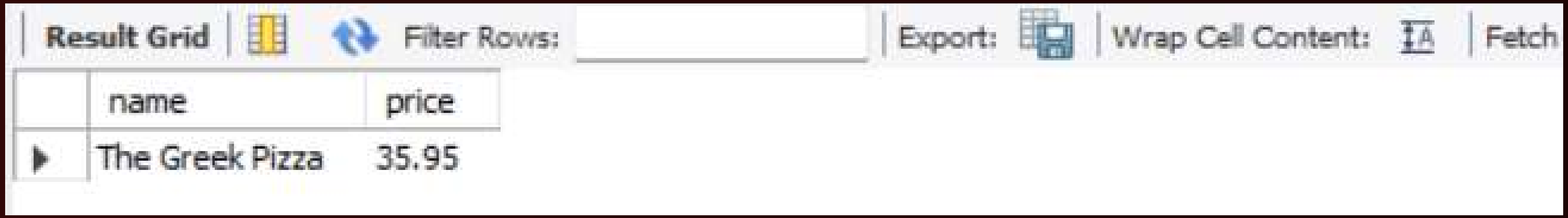
```
total_sales
817860.05
```

Identify the highest-priced pizza.



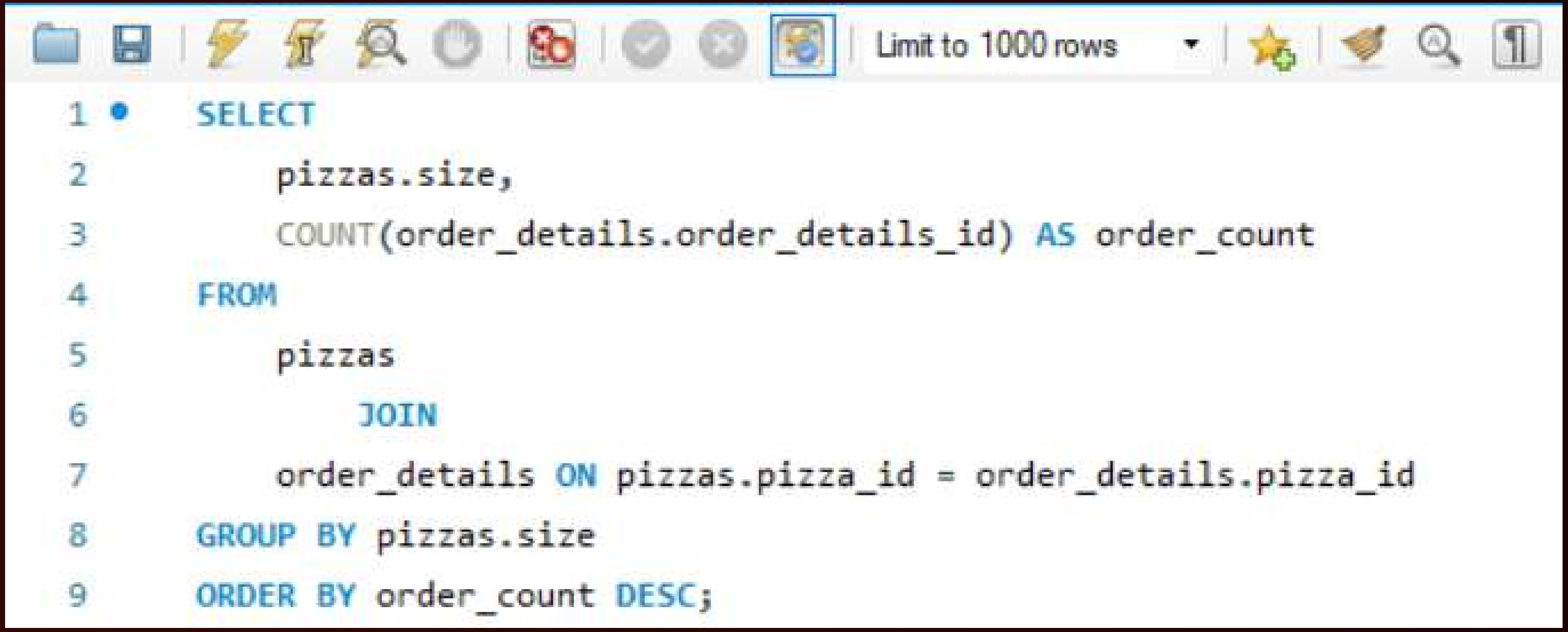
```
SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```

OUTPUT AS:



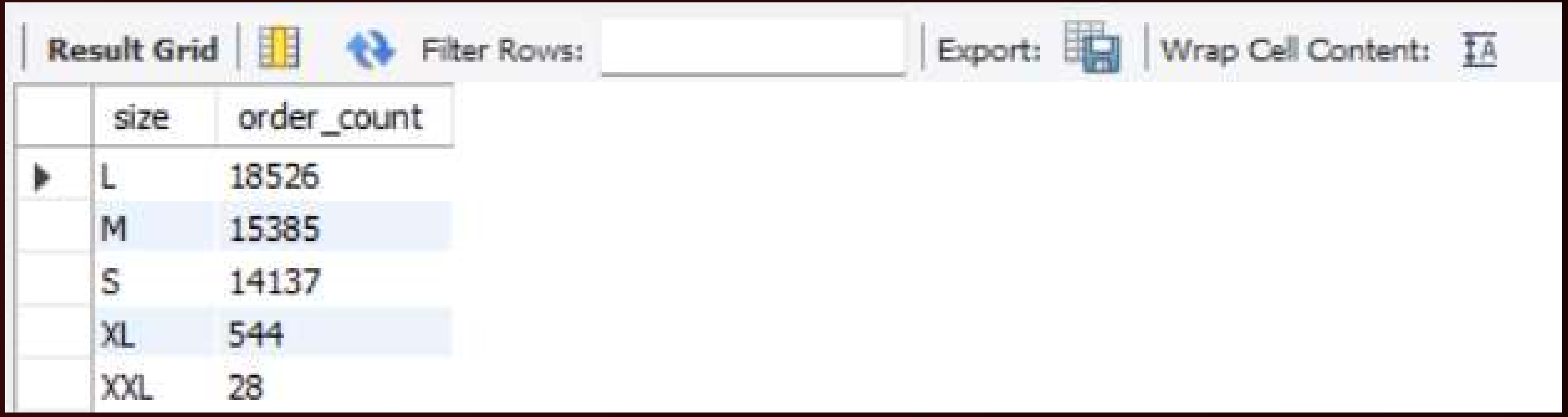
name	price
The Greek Pizza	35.95

Identify the most common pizza size ordered.

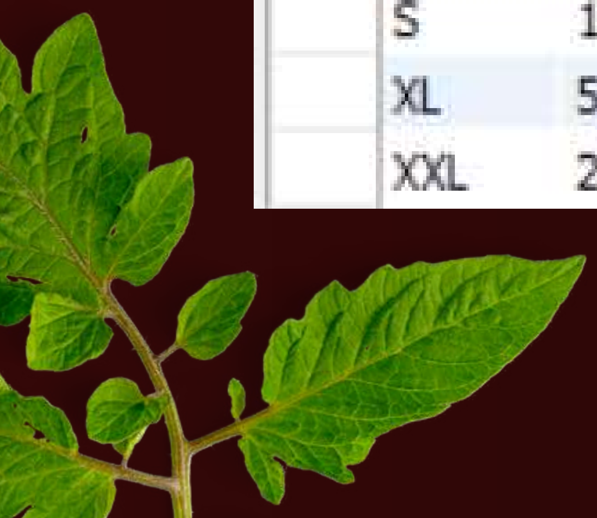


```sql
SELECT
    pizzas.size,
    COUNT(order_details.order_details_id) AS order_count
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC;
```

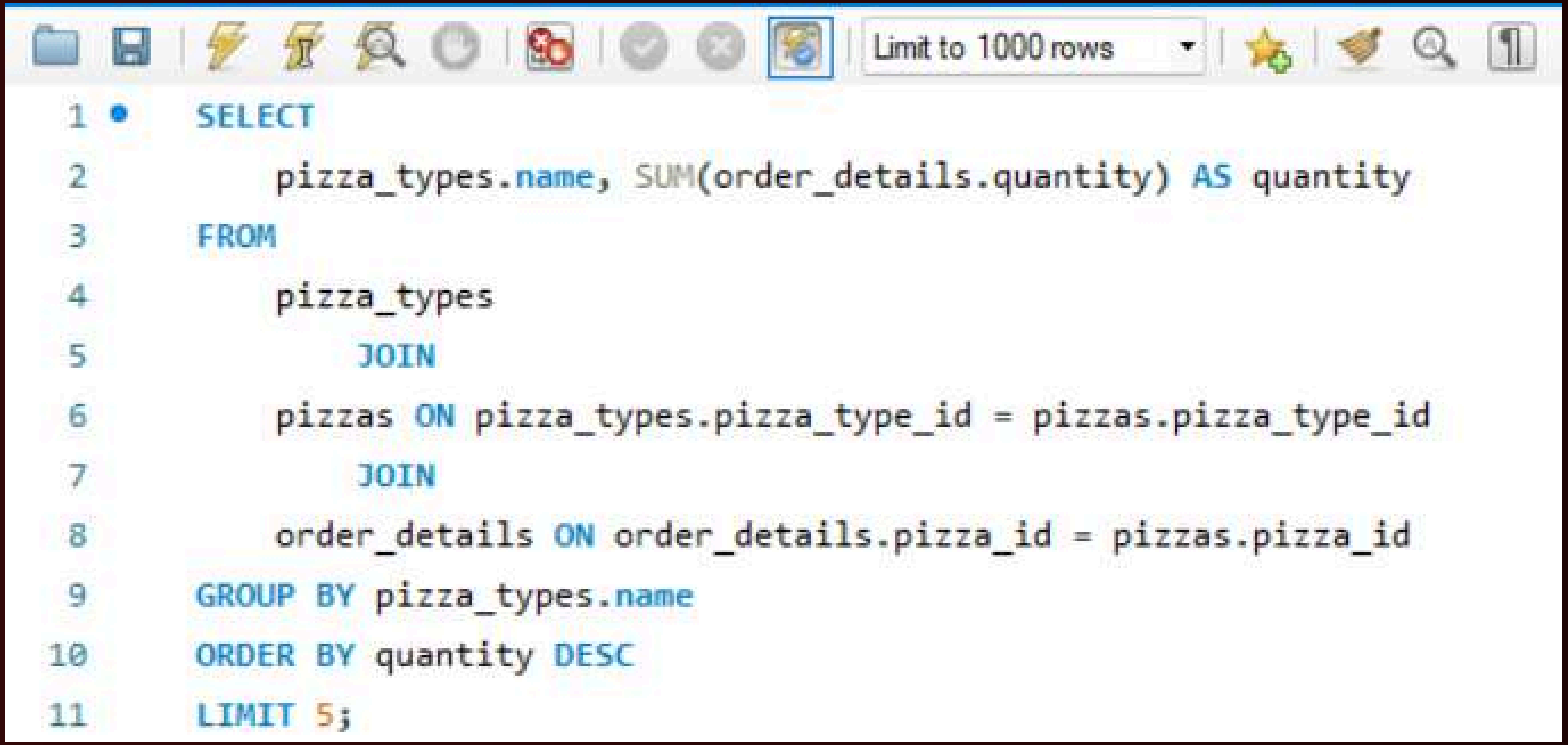
OUTPUT AS:



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

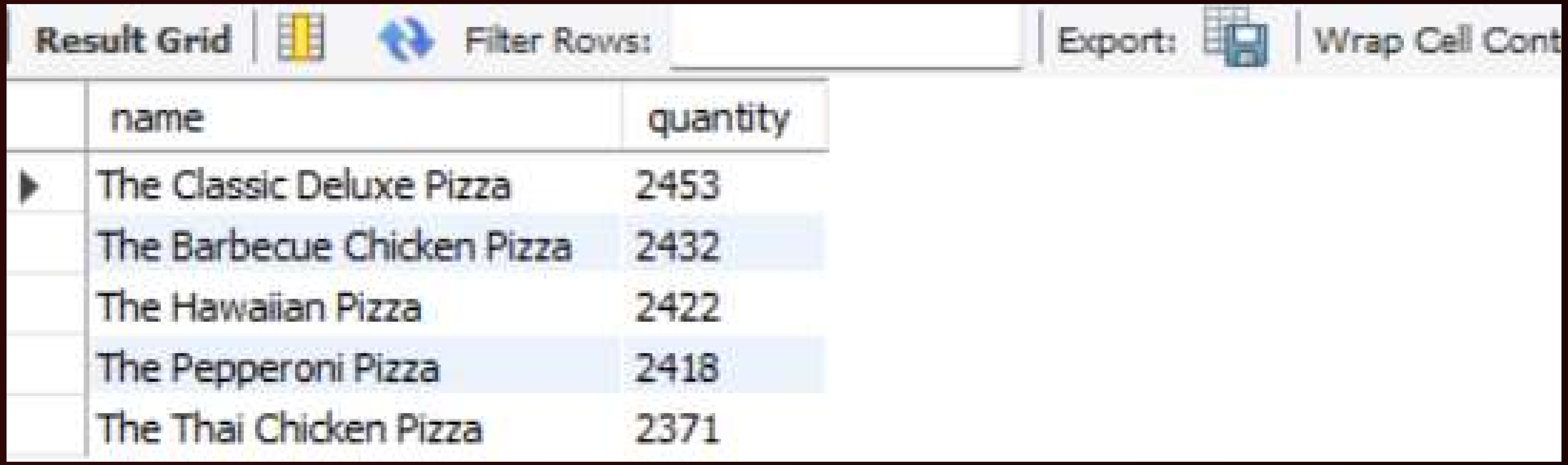


List the top 5 most ordered pizza types along with their quantities.



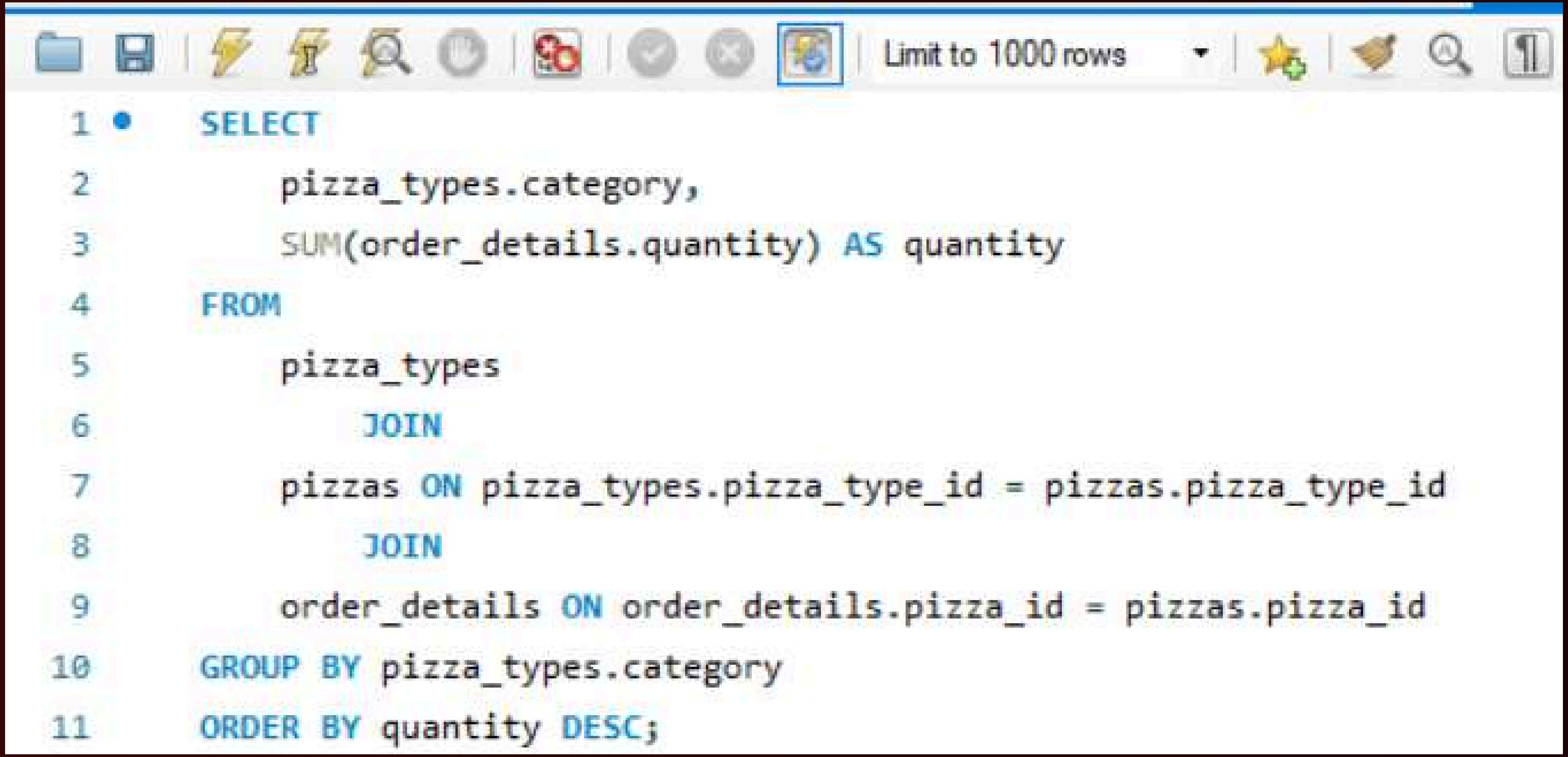
```
SELECT
    pizza_types.name, SUM(order_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC
LIMIT 5;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

OUTPUT AS:



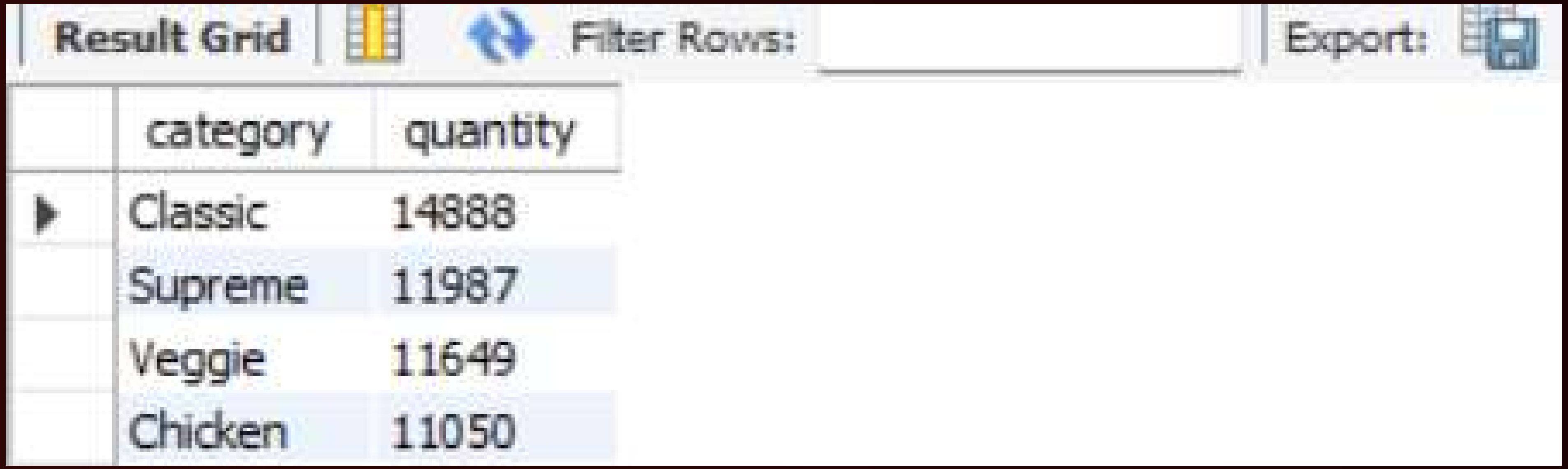
name	quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

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



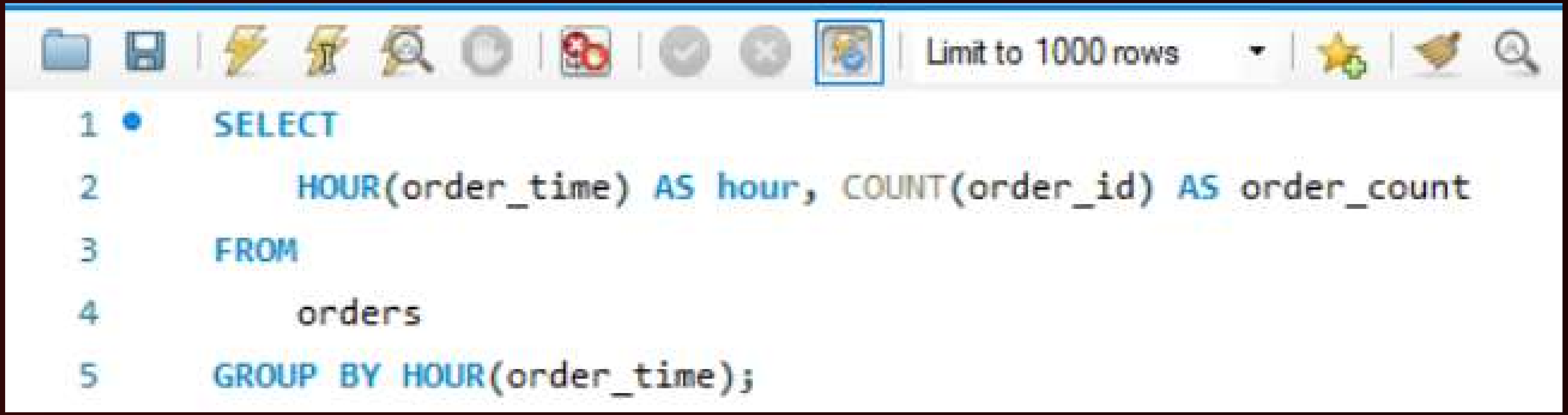
```sql
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY quantity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

OUTPUT AS:



category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

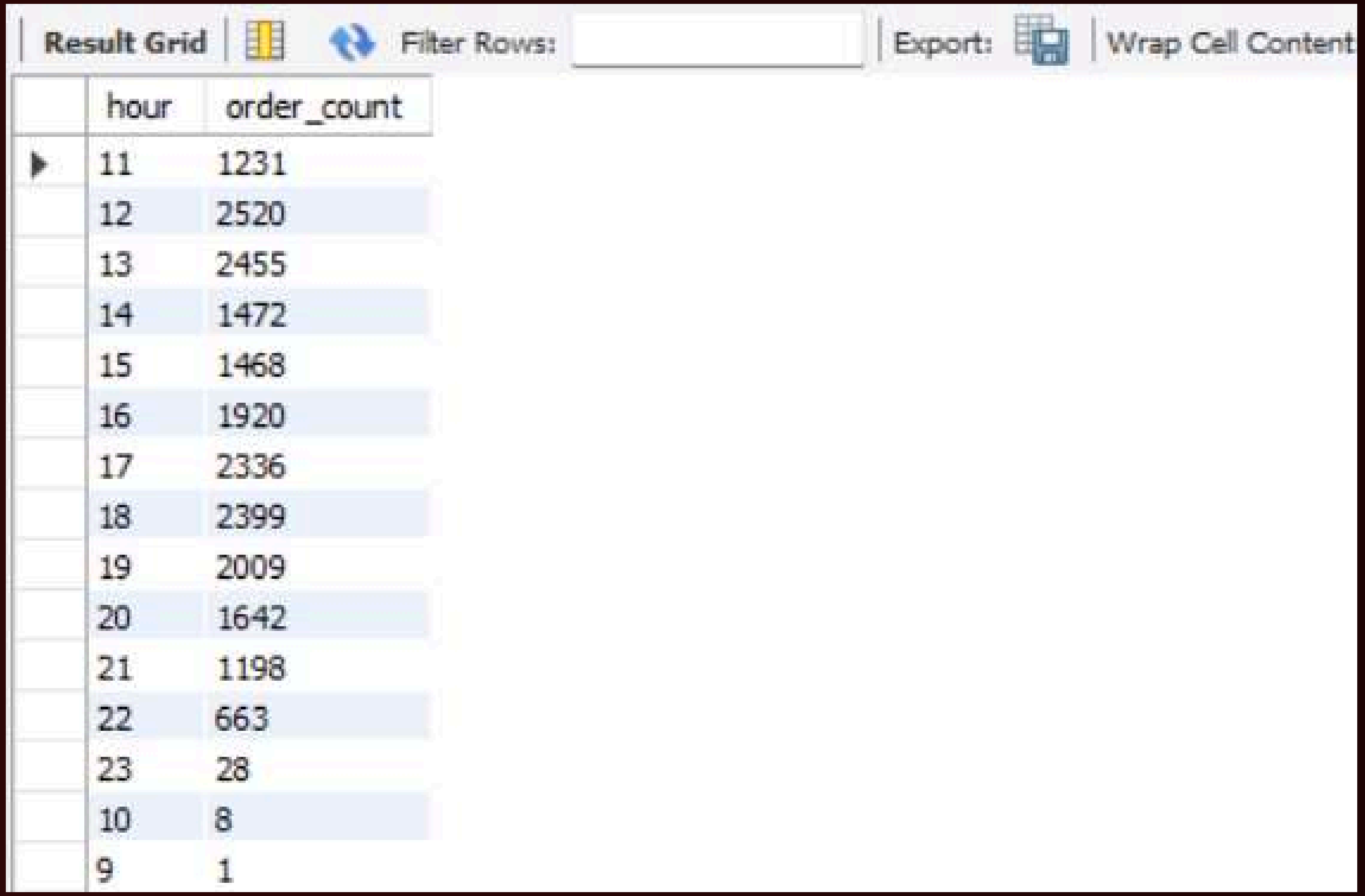
Determine the distribution of orders by hour of the day.



Limit to 1000 rows

```sql
SELECT
    HOUR(order_time) AS hour, COUNT(order_id) AS order_count
FROM
    orders
GROUP BY HOUR(order_time);
```

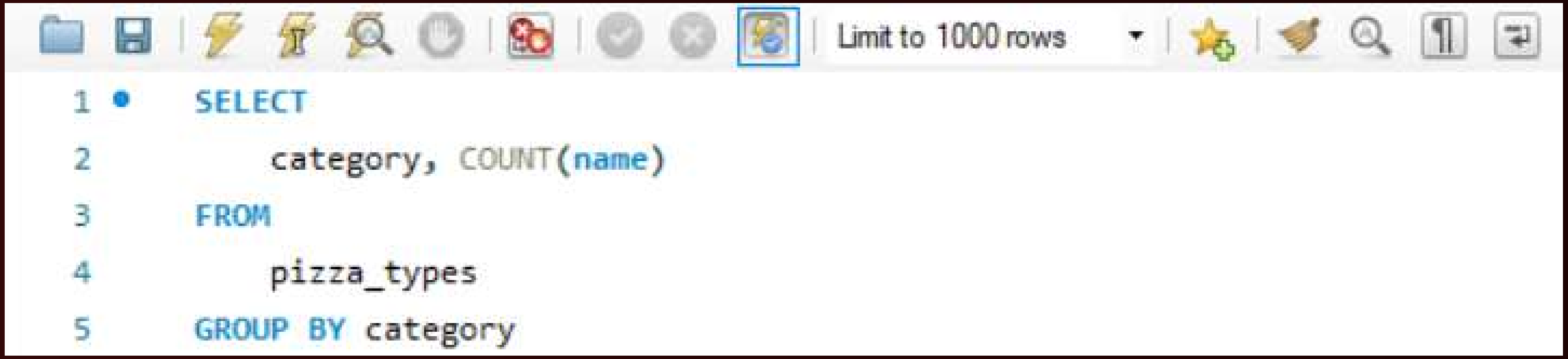
OUTPUT AS:



Result Grid | Filter Rows: | Export: | Wrap Cell Content

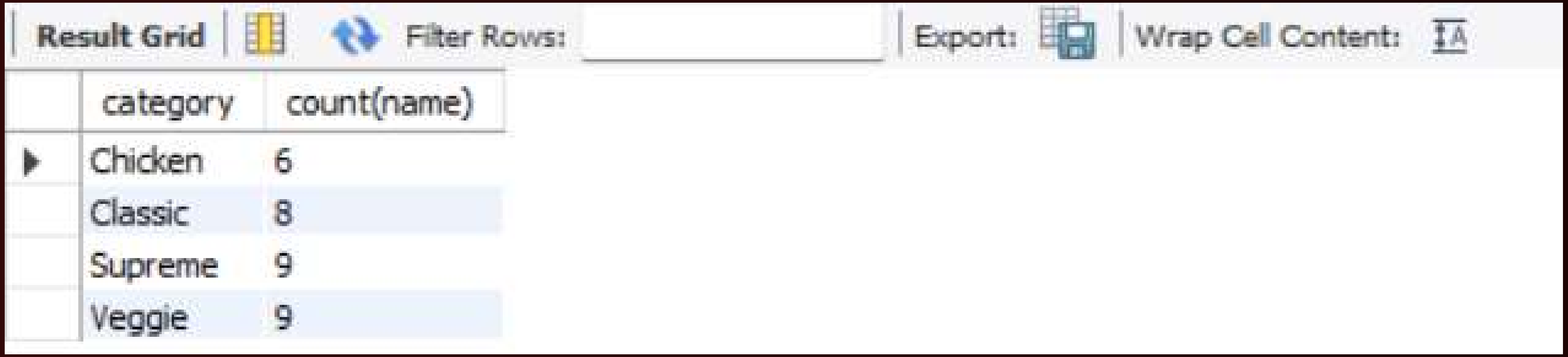
hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

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



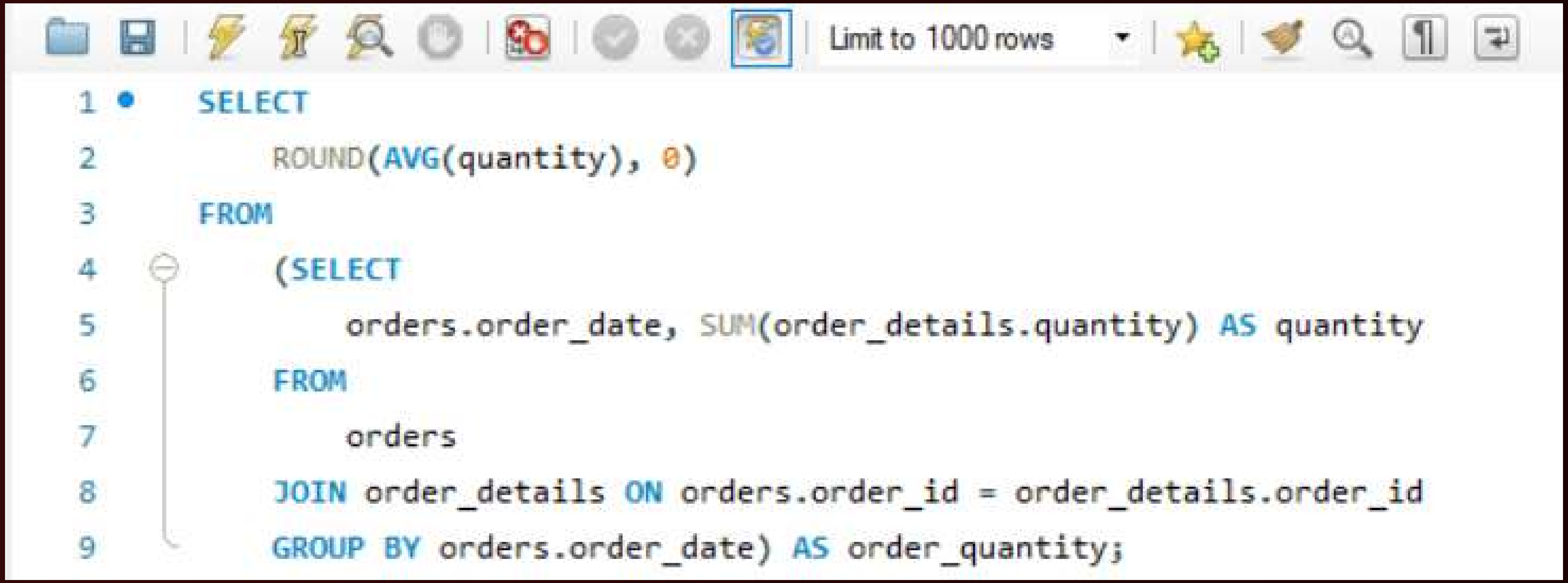
```
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category
```

OUTPUT AS:



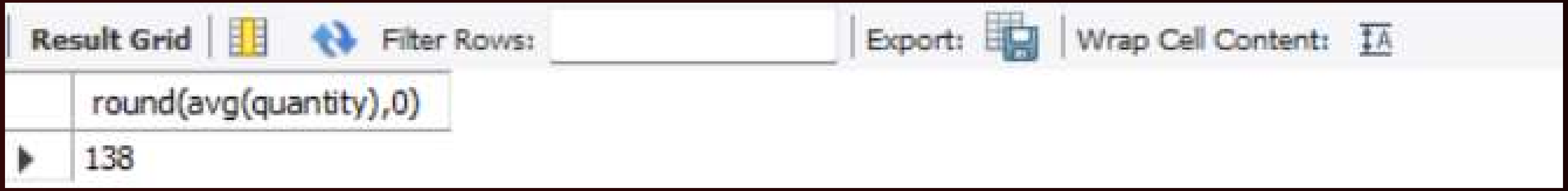
category	count(name)
Chicken	6
Classic	8
Supreme	9
Veggie	9

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



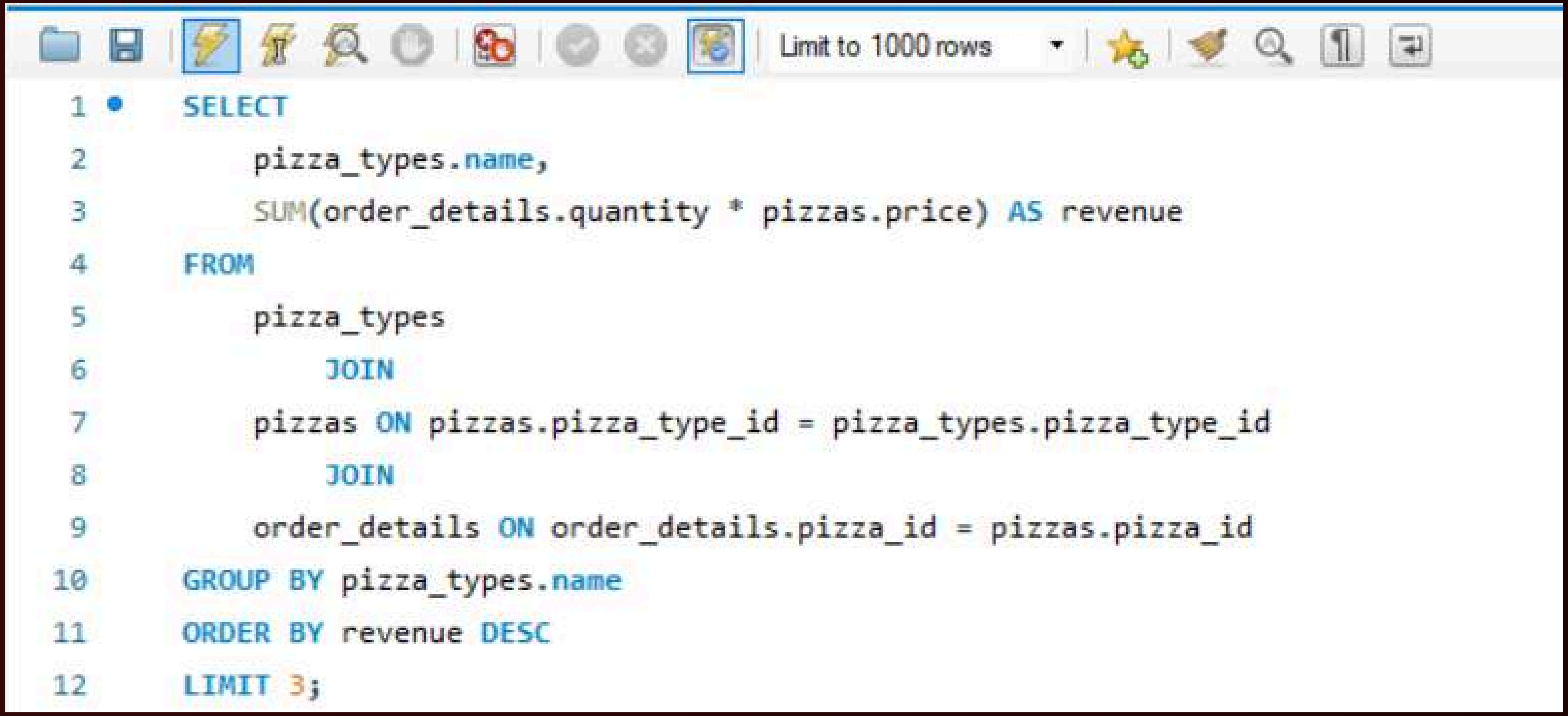
```
SELECT
    ROUND(AVG(quantity), 0)
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

OUTPUT AS:



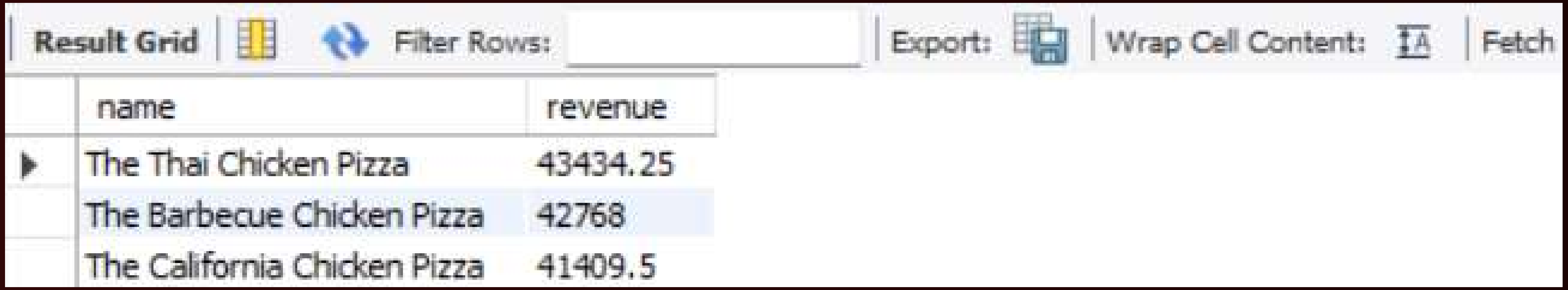
```
round(avg(quantity),0)
138
```

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

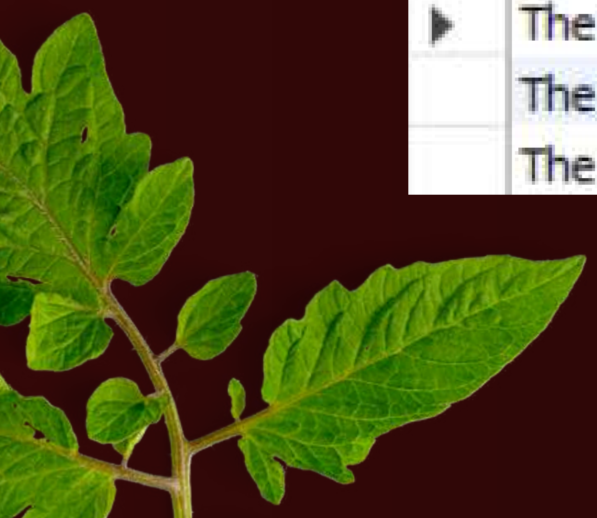


```
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue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

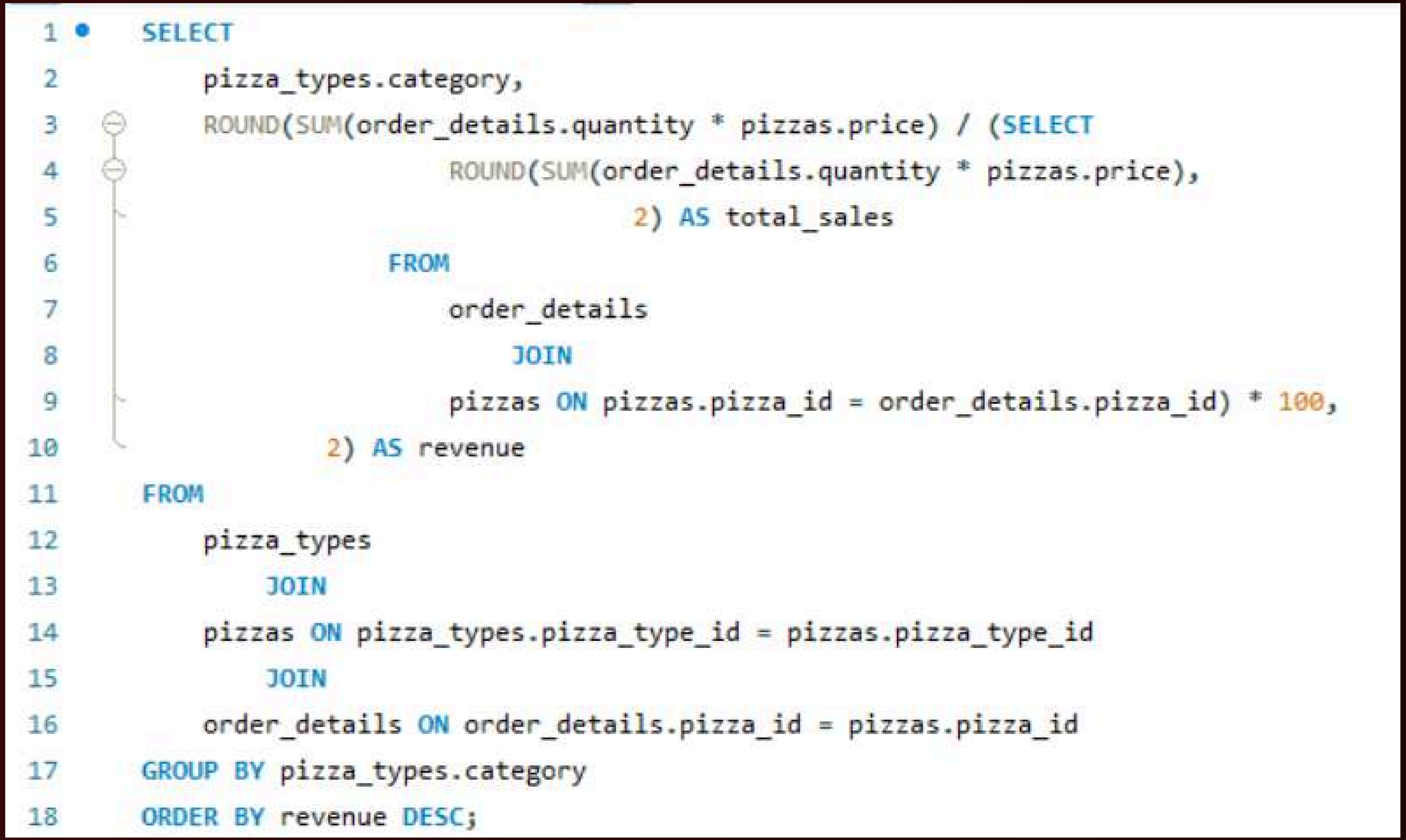
OUTPUT AS:



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

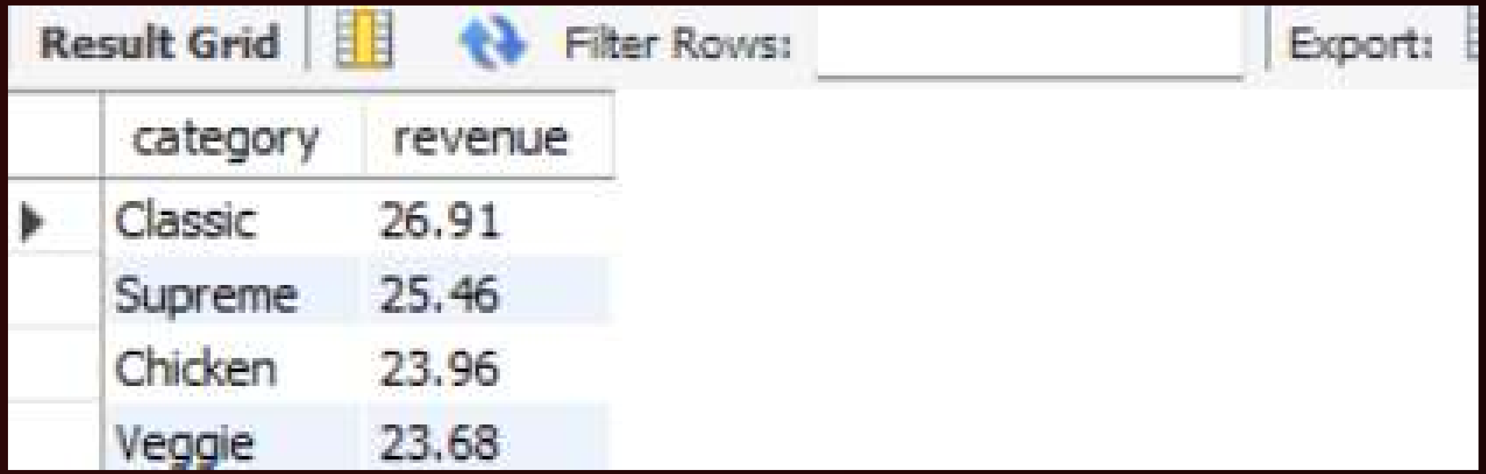


Calculate the percentage contribution of each pizza type to total revenue.



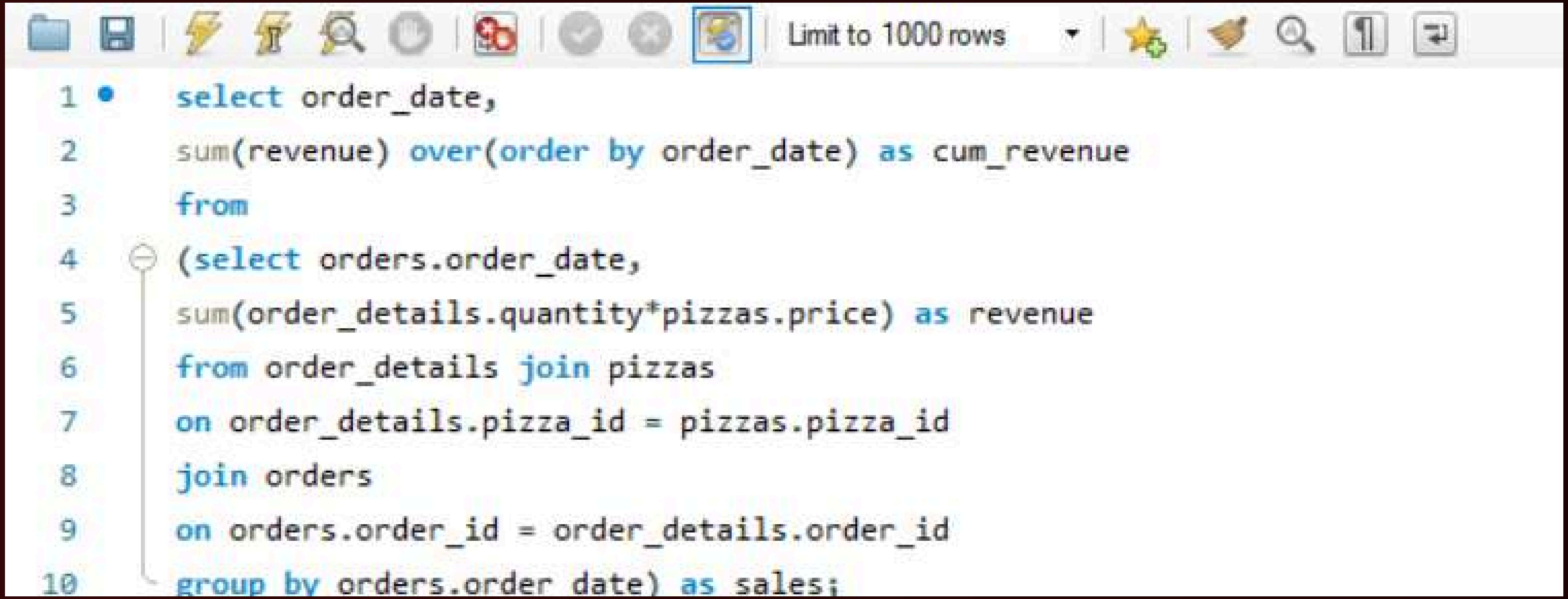
```
SELECT
    pizza_types.category,
    ROUND(SUM(order_details.quantity * pizzas.price) / (SELECT
                ROUND(SUM(order_details.quantity * pizzas.price),
                            2) AS total_sales
            FROM
                order_details
                    JOIN
                pizzas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

OUTPUT AS:



category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

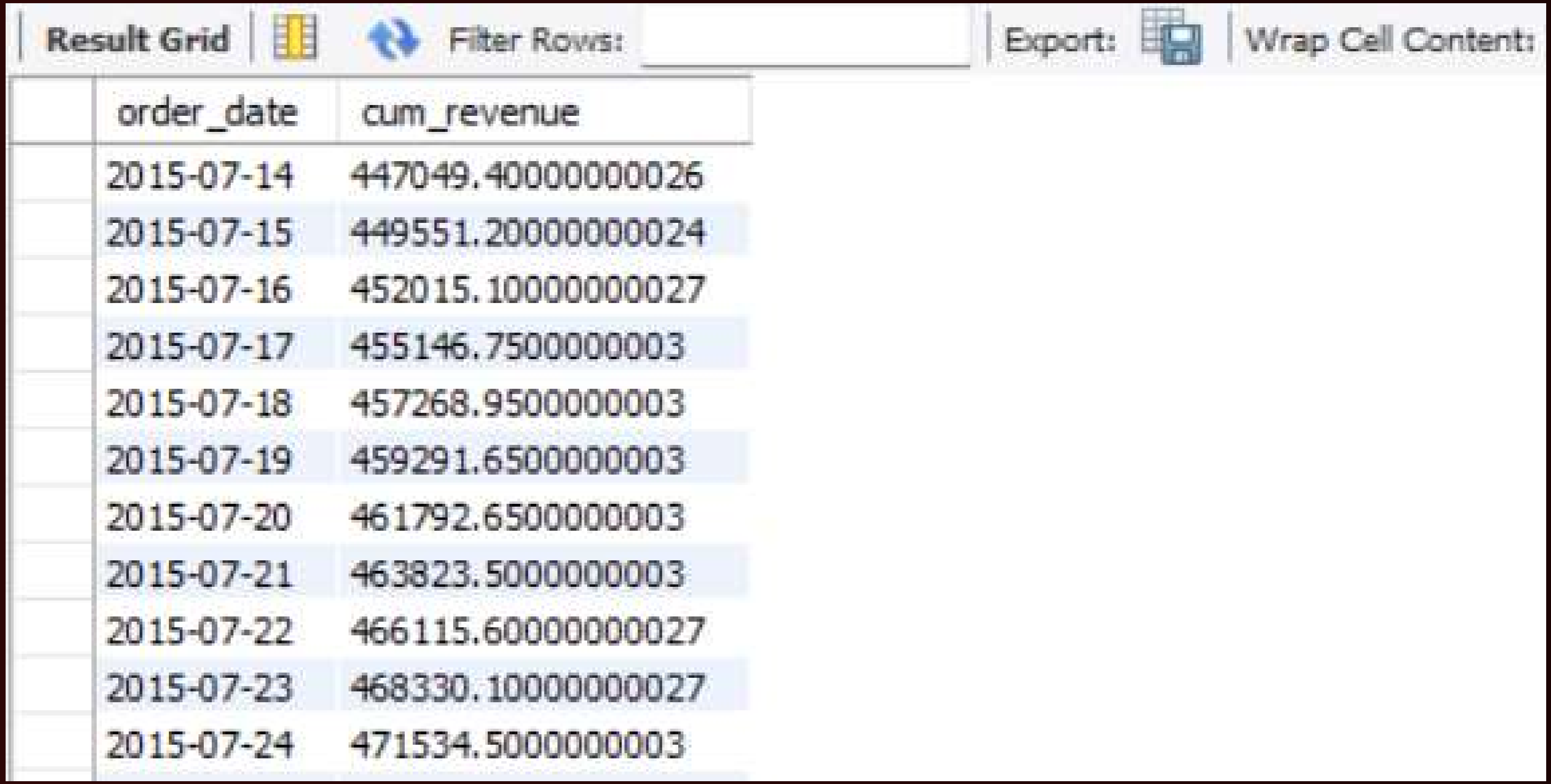
Analyze the cumulative revenue generated over time.



Limit to 1000 rows

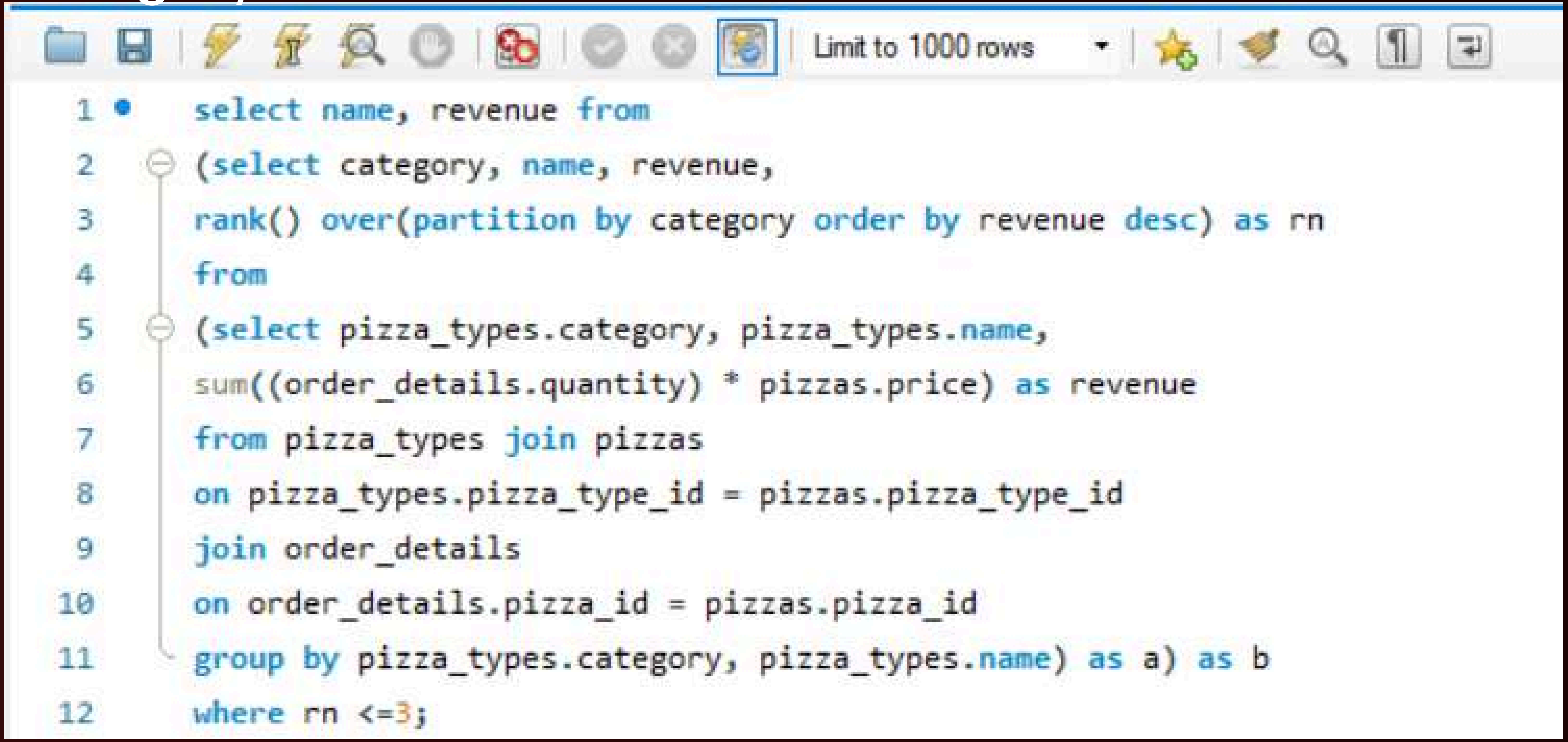
```sql
select order_date,
sum(revenue) over(order by order_date) as cum_revenue
from
(select orders.order_date,
sum(order_details.quantity*pizzas.price) as revenue
from order_details join pizzas
on order_details.pizza_id = pizzas.pizza_id
join orders
on orders.order_id = order_details.order_id
group by orders.order_date) as sales;
```

OUTPUT AS:



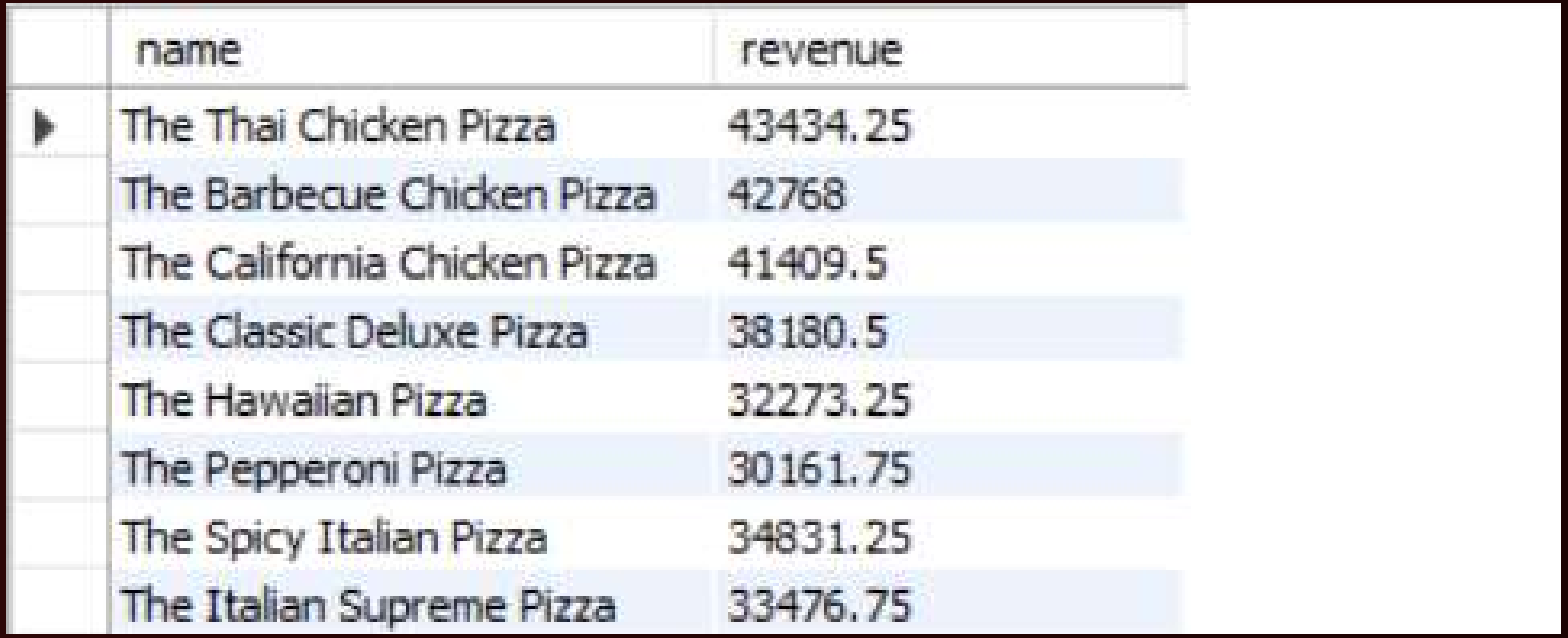
order_date	cum_revenue
2015-07-14	447049.40000000026
2015-07-15	449551.20000000024
2015-07-16	452015.10000000027
2015-07-17	455146.7500000003
2015-07-18	457268.9500000003
2015-07-19	459291.6500000003
2015-07-20	461792.6500000003
2015-07-21	463823.5000000003
2015-07-22	466115.60000000027
2015-07-23	468330.10000000027
2015-07-24	471534.5000000003

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;
```

OUTPUT AS:



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

THANK YOU