PIZZAHUT SALES ANALYSIS USING SQL

RAVISHANKAR SINGH

Basic Questions:

1. Retrieve the total number of orders placed.

2. Calculate the total revenue generated from pizza sales.

3. Identify the highest-priced pizza.

4. Identify the most common pizza size ordered.

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

Intermediate Questions:

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

7. Determine the distribution of orders by hour of the day.

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

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

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

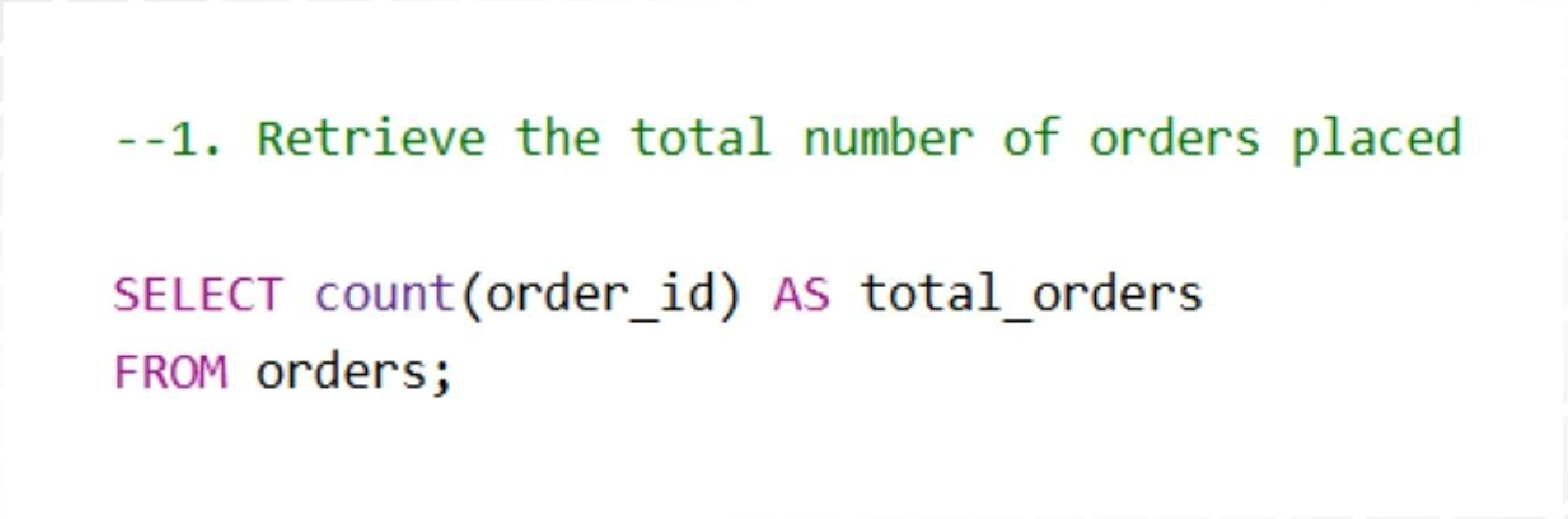
Advanced Questions:

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

12. Analyze the cumulative revenue generated over time.

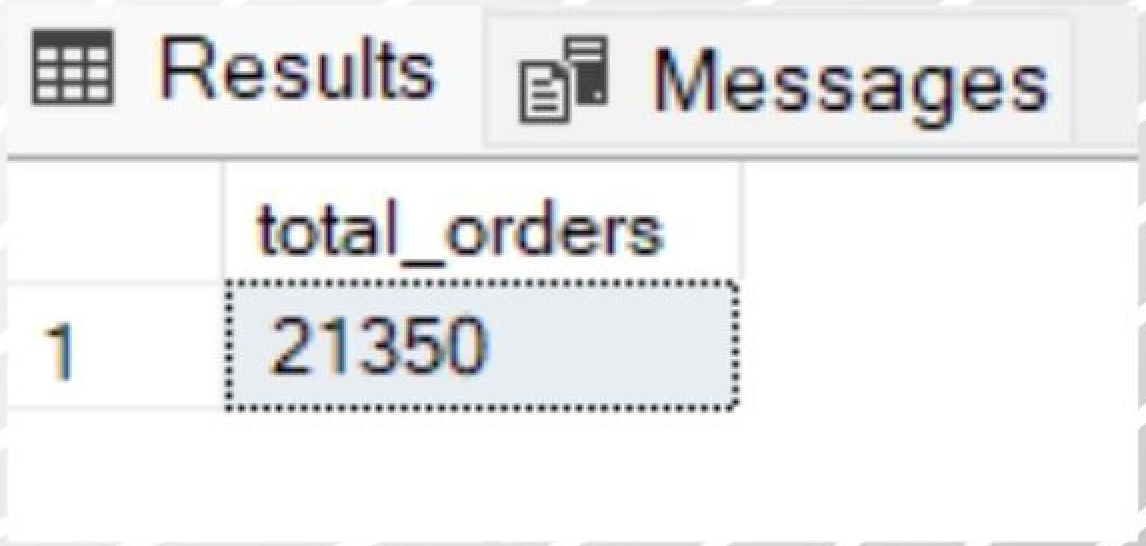
13. Determine the top 3 most ordered pizza types based on revenue for each pizza category.

code



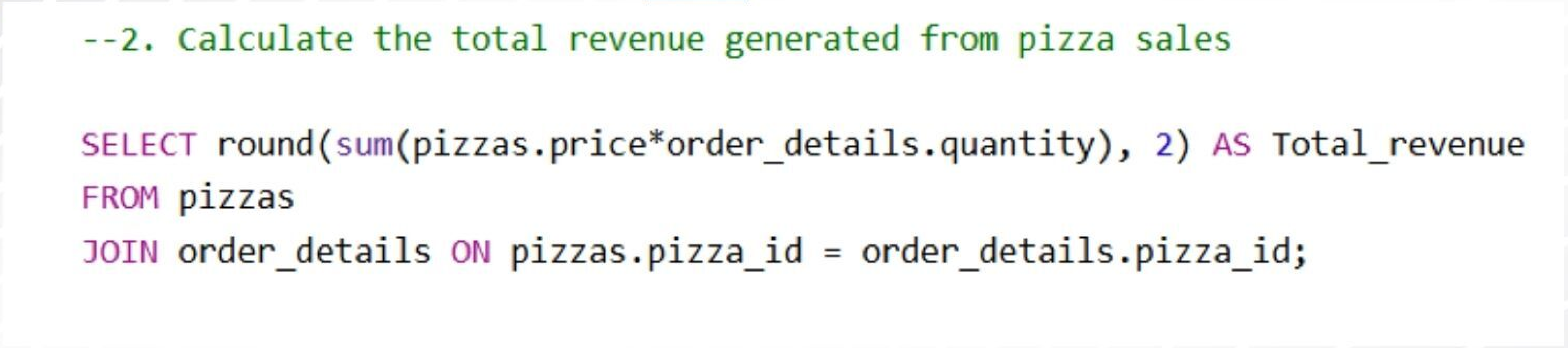
```sql
--1. Retrieve the total number of orders placed

SELECT count(order_id) AS total_orders
FROM orders;
```

output



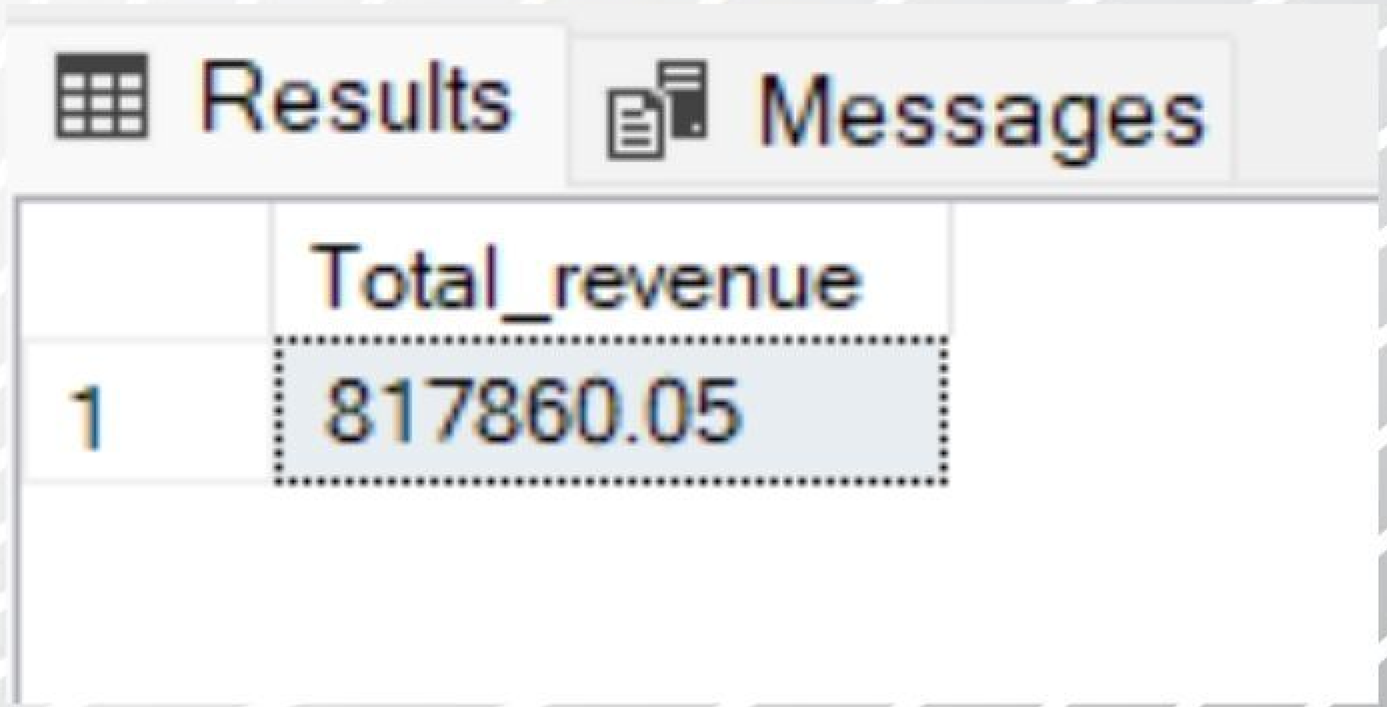
```
   total_orders
1  21350
```

code



```sql
--2. Calculate the total revenue generated from pizza sales

SELECT round(sum(pizzas.price*order_details.quantity), 2) AS Total_revenue
FROM pizzas
JOIN order_details ON pizzas.pizza_id = order_details.pizza_id;
```

output

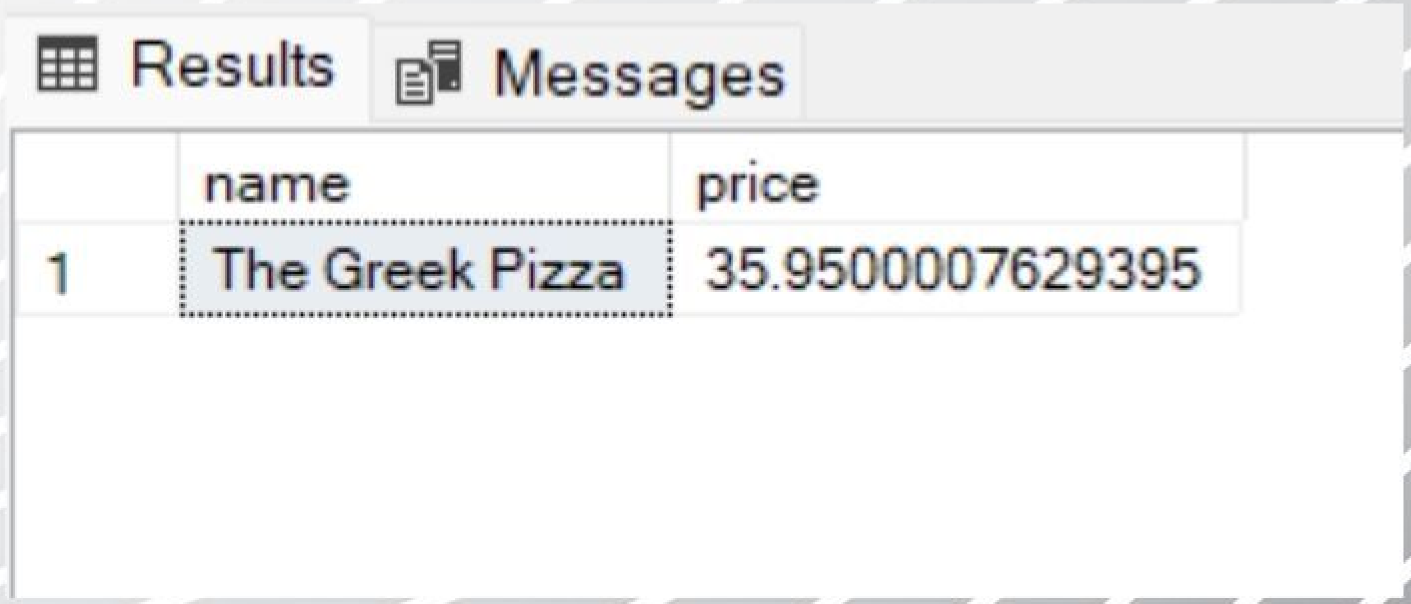


```
   Total_revenue
1  817860.05
```

code

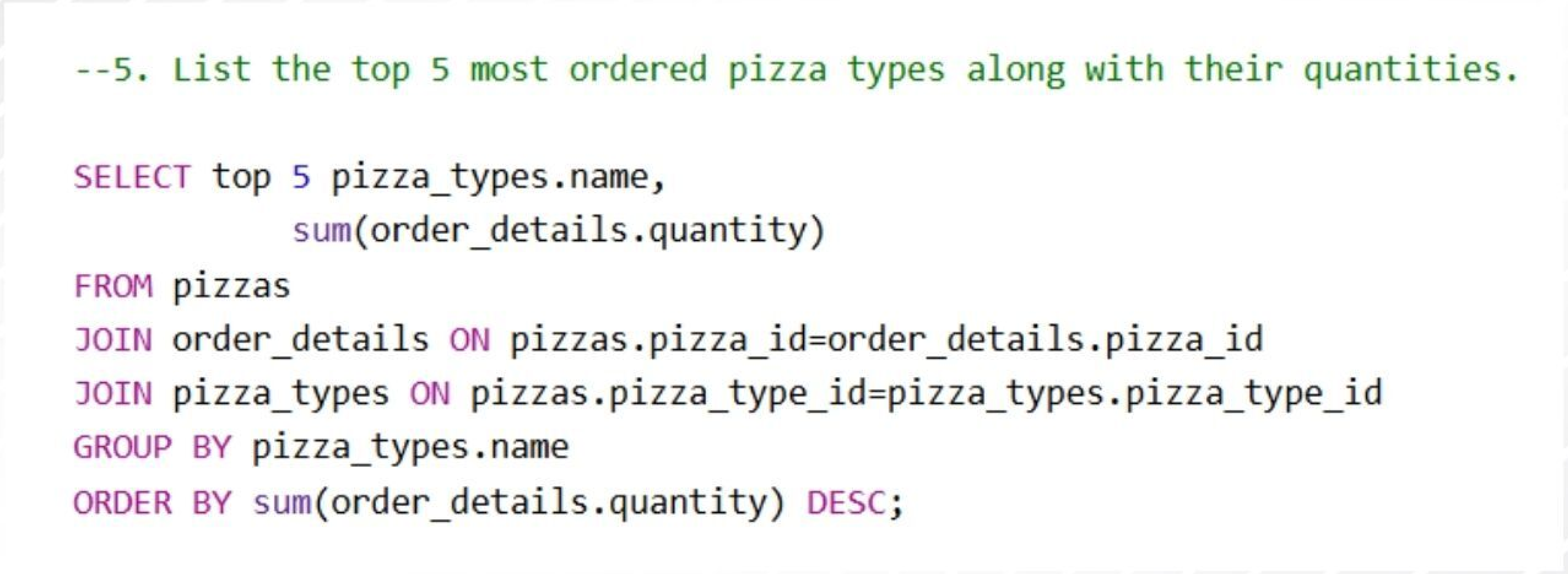
```
--3.Identify the highest-priced pizza.

SELECT top 1 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;
```

output



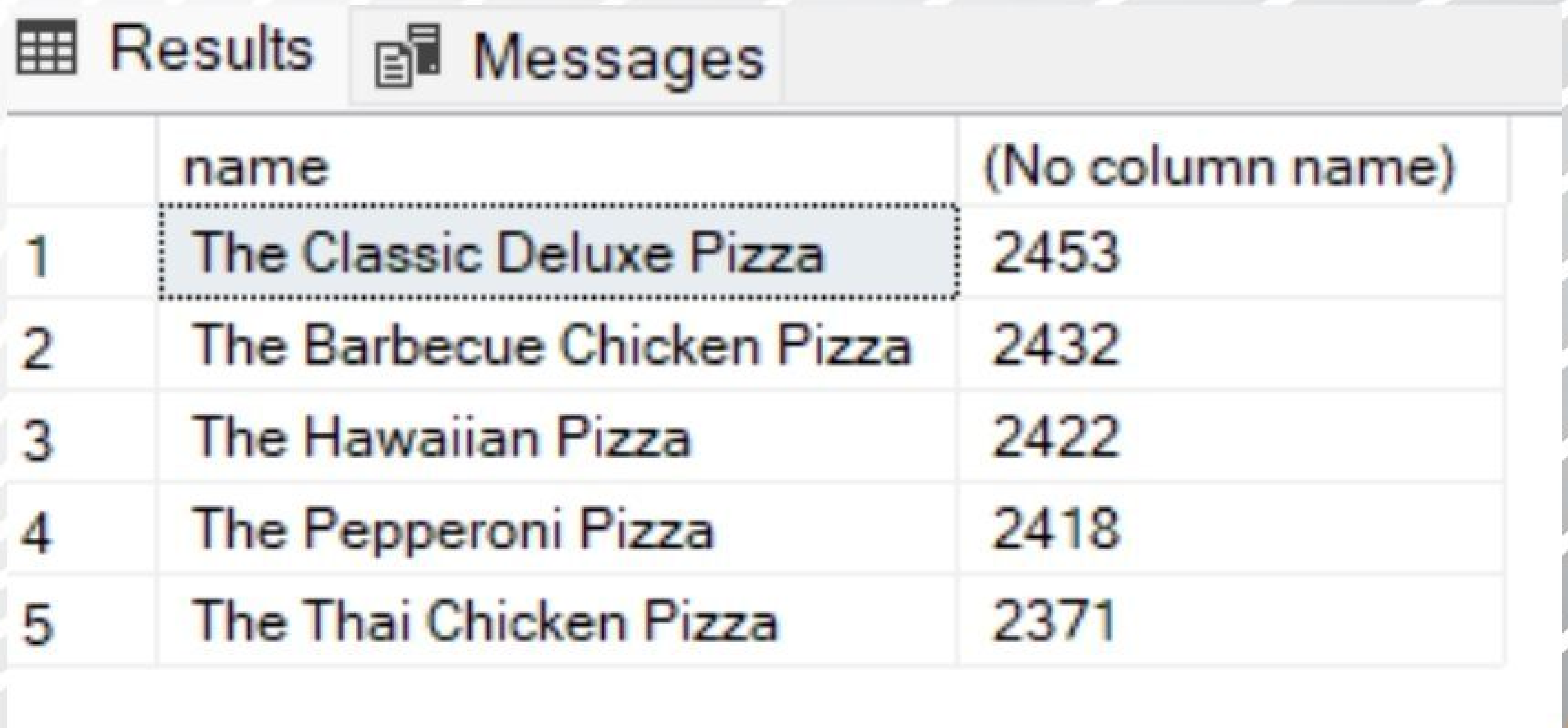
```
   name             price
1  The Greek Pizza  35.9500007629395
```

code



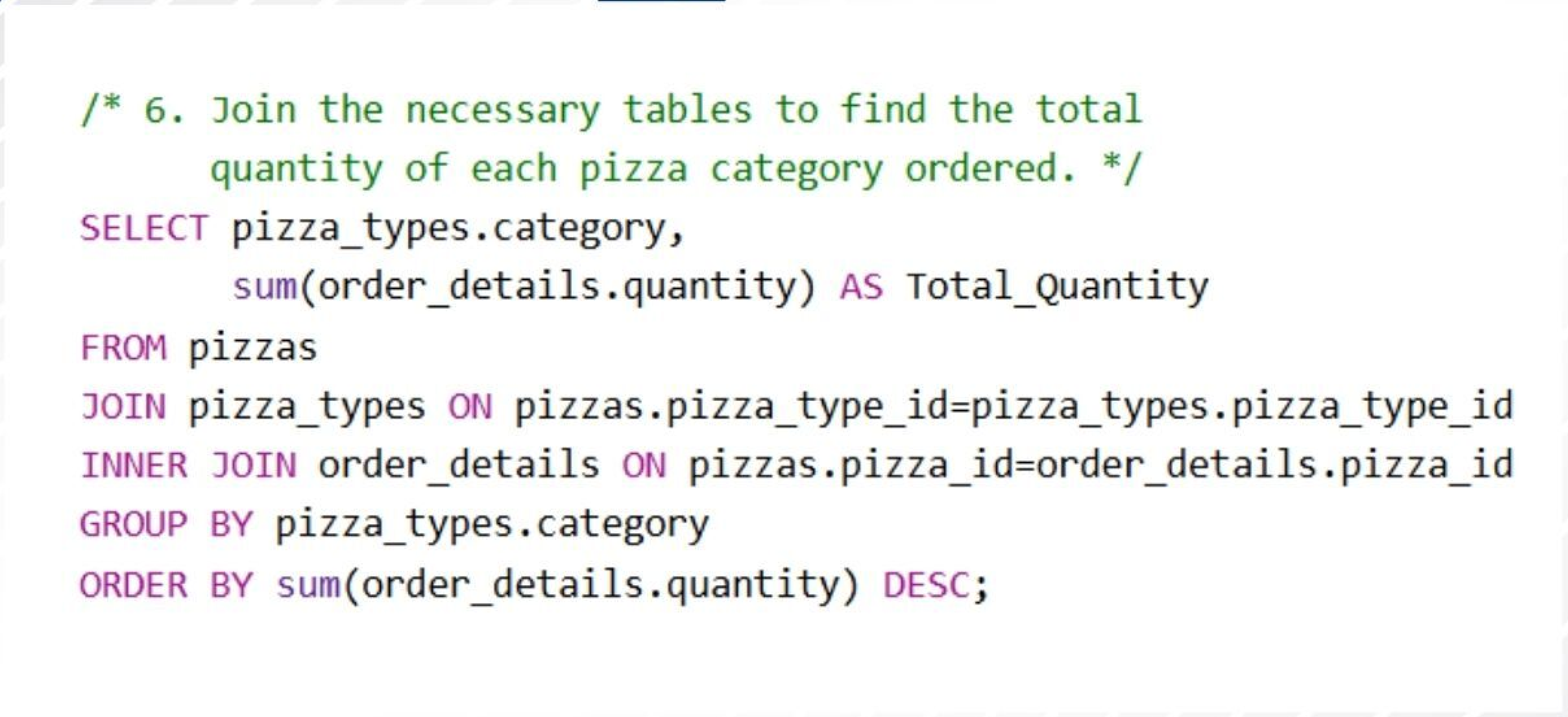
```sql
--5. List the top 5 most ordered pizza types along with their quantities.

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

output



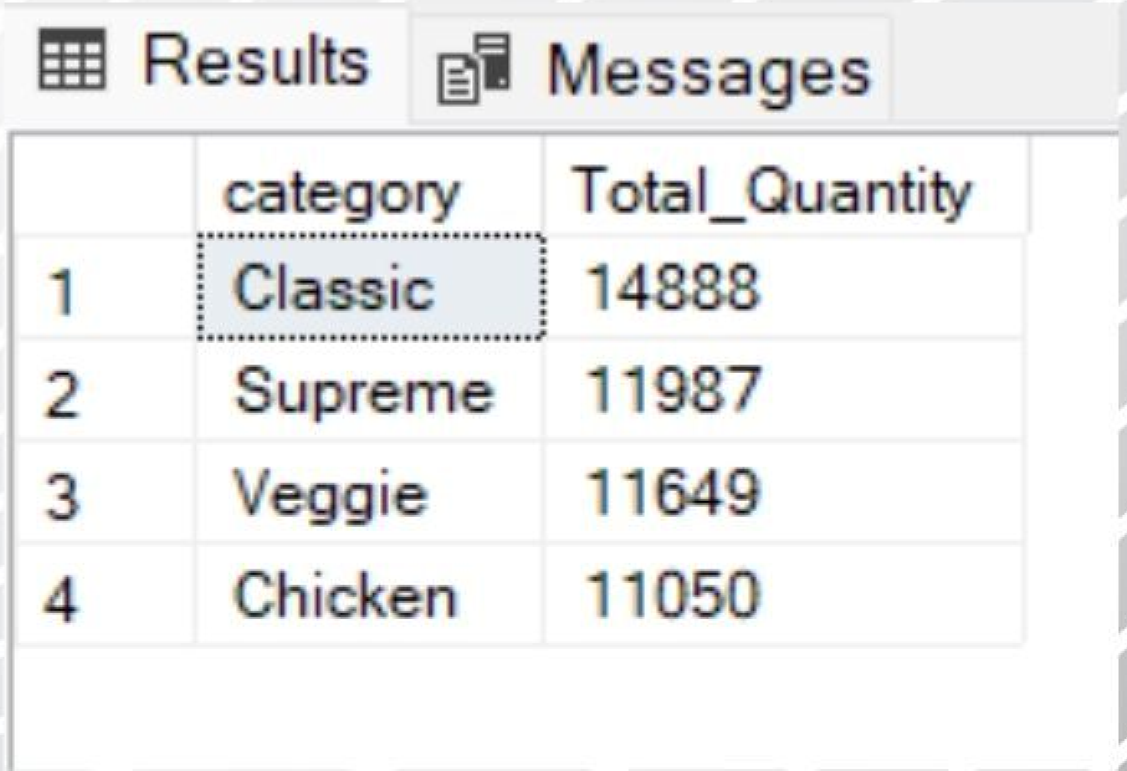
	name	(No column name)
1	The Classic Deluxe Pizza	2453
2	The Barbecue Chicken Pizza	2432
3	The Hawaiian Pizza	2422
4	The Pepperoni Pizza	2418
5	The Thai Chicken Pizza	2371

code



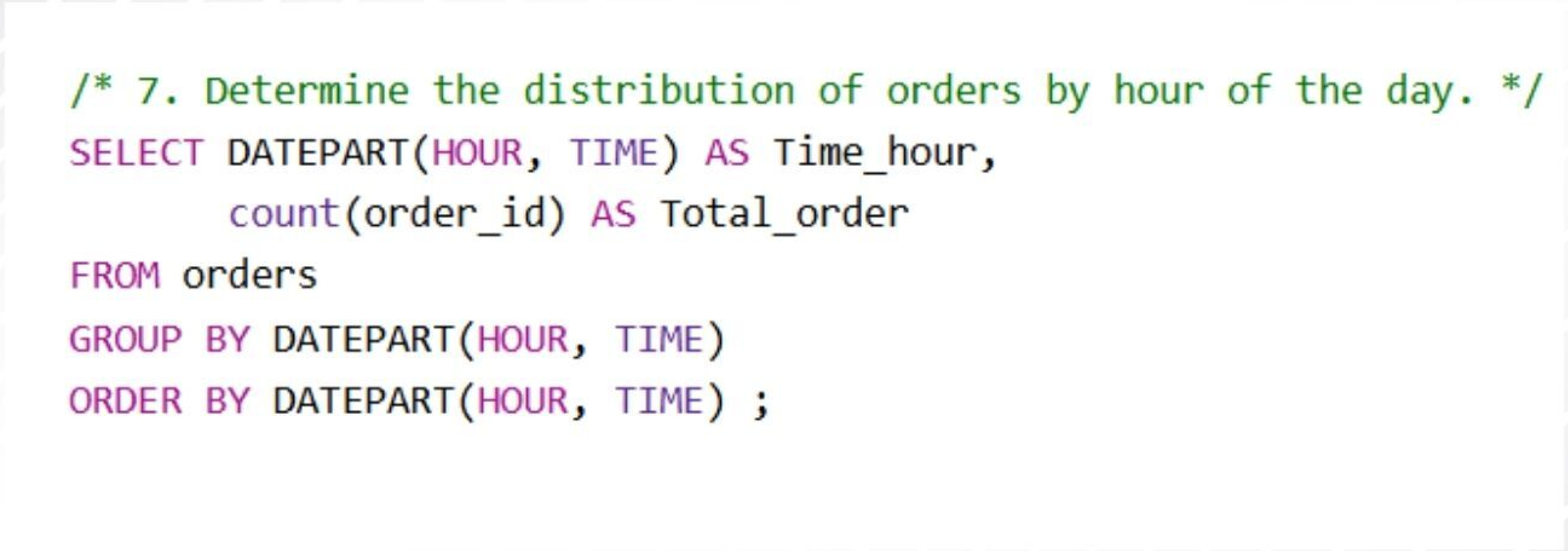
```
/* 6. Join the necessary tables to find the total
      quantity of each pizza category ordered. */
SELECT pizza_types.category,
       sum(order_details.quantity) AS Total_Quantity
FROM pizzas
JOIN pizza_types ON pizzas.pizza_type_id=pizza_types.pizza_type_id
INNER JOIN order_details ON pizzas.pizza_id=order_details.pizza_id
GROUP BY pizza_types.category
ORDER BY sum(order_details.quantity) DESC;
```

output



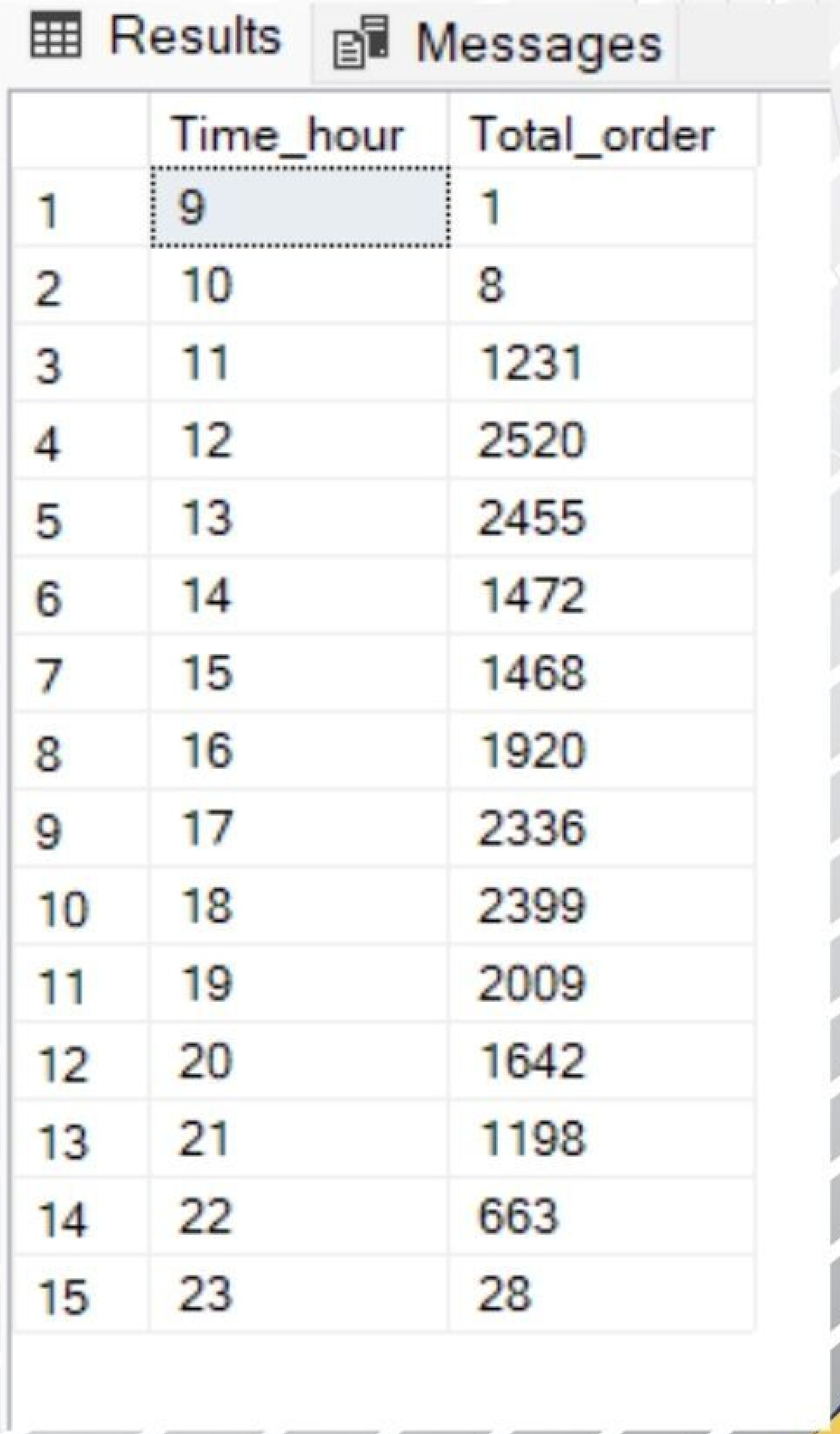
```
   category  Total_Quantity
1  Classic   14888
2  Supreme   11987
3  Veggie    11649
4  Chicken   11050
```

code



```sql
/* 7. Determine the distribution of orders by hour of the day. */
SELECT DATEPART(HOUR, TIME) AS Time_hour,
       count(order_id) AS Total_order
FROM orders
GROUP BY DATEPART(HOUR, TIME)
ORDER BY DATEPART(HOUR, TIME) ;
```

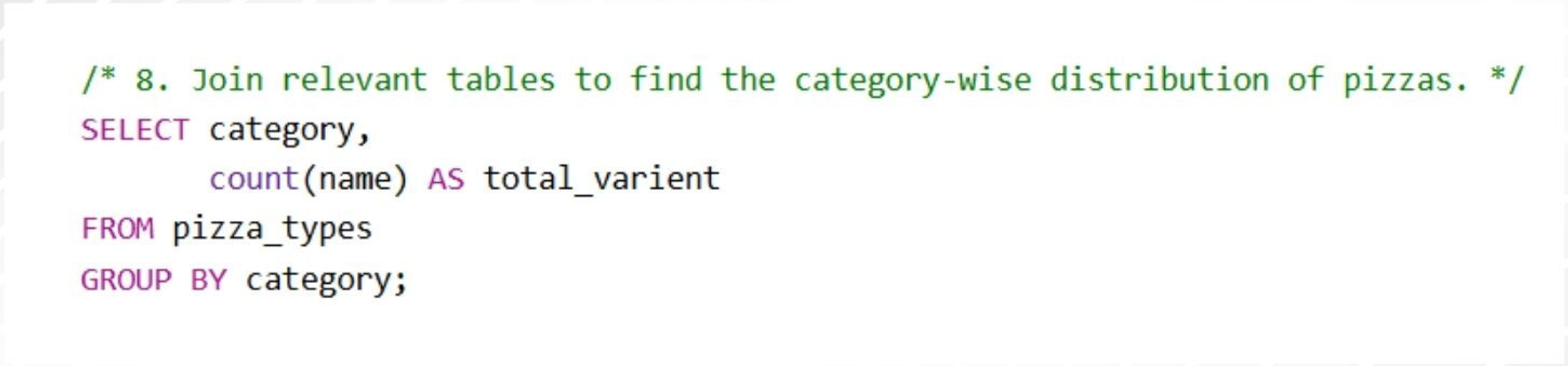
output



Results Messages

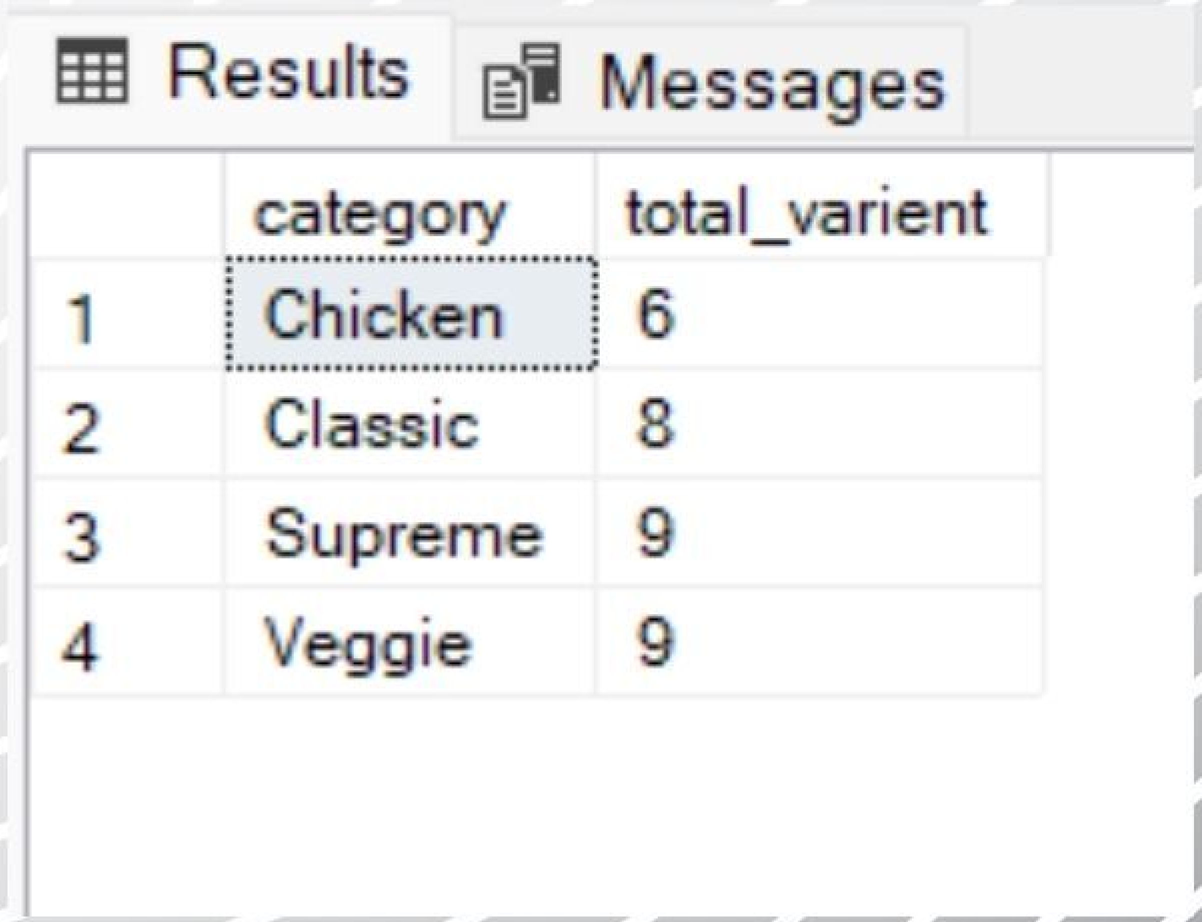
	Time_hour	Total_order
1	9	1
2	10	8
3	11	1231
4	12	2520
5	13	2455
6	14	1472
7	15	1468
8	16	1920
9	17	2336
10	18	2399
11	19	2009
12	20	1642
13	21	1198
14	22	663
15	23	28

code



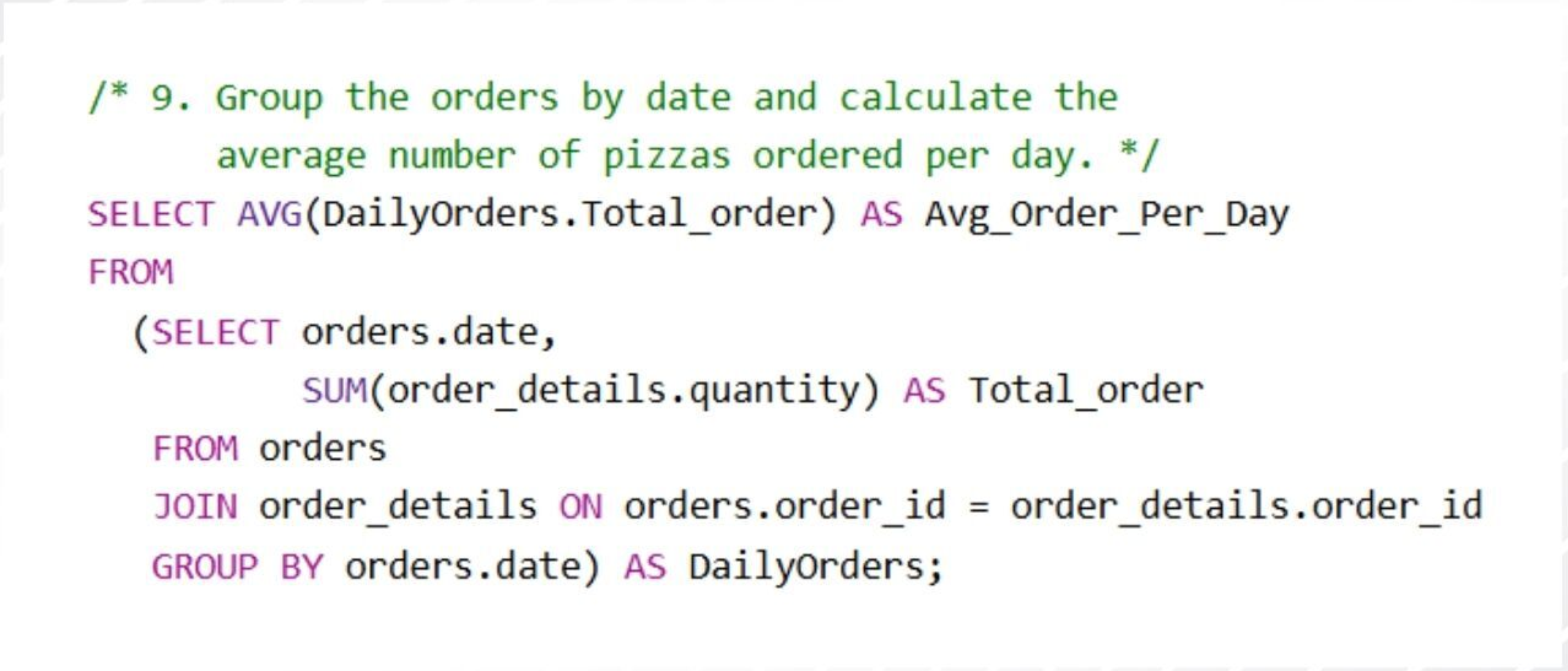
```
/* 8. Join relevant tables to find the category-wise distribution of pizzas. */
SELECT category,
       count(name) AS total_varient
FROM pizza_types
GROUP BY category;
```

output



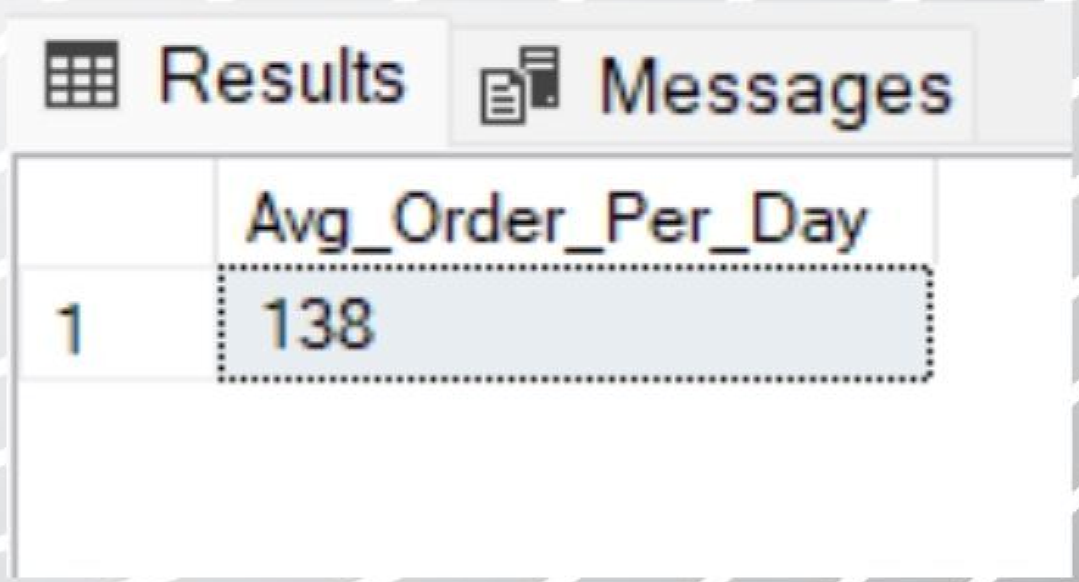
	category	total_varient
1	Chicken	6
2	Classic	8
3	Supreme	9
4	Veggie	9

code



```
/* 9. Group the orders by date and calculate the
      average number of pizzas ordered per day. */
SELECT AVG(DailyOrders.Total_order) AS Avg_Order_Per_Day
FROM
  (SELECT orders.date,
          SUM(order_details.quantity) AS Total_order
   FROM orders
   JOIN order_details ON orders.order_id = order_details.order_id
   GROUP BY orders.date) AS DailyOrders;
```

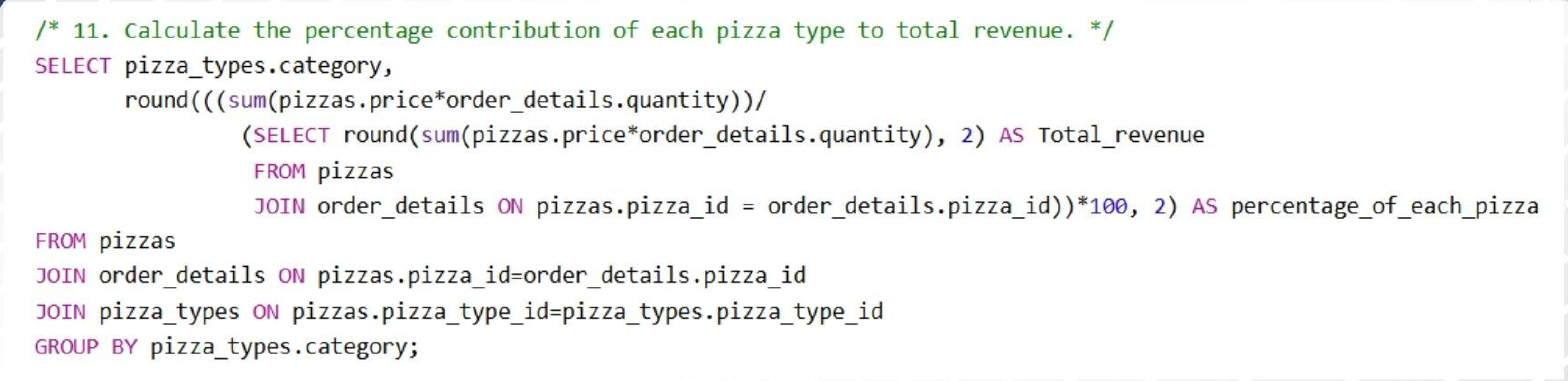
output



Results | Messages

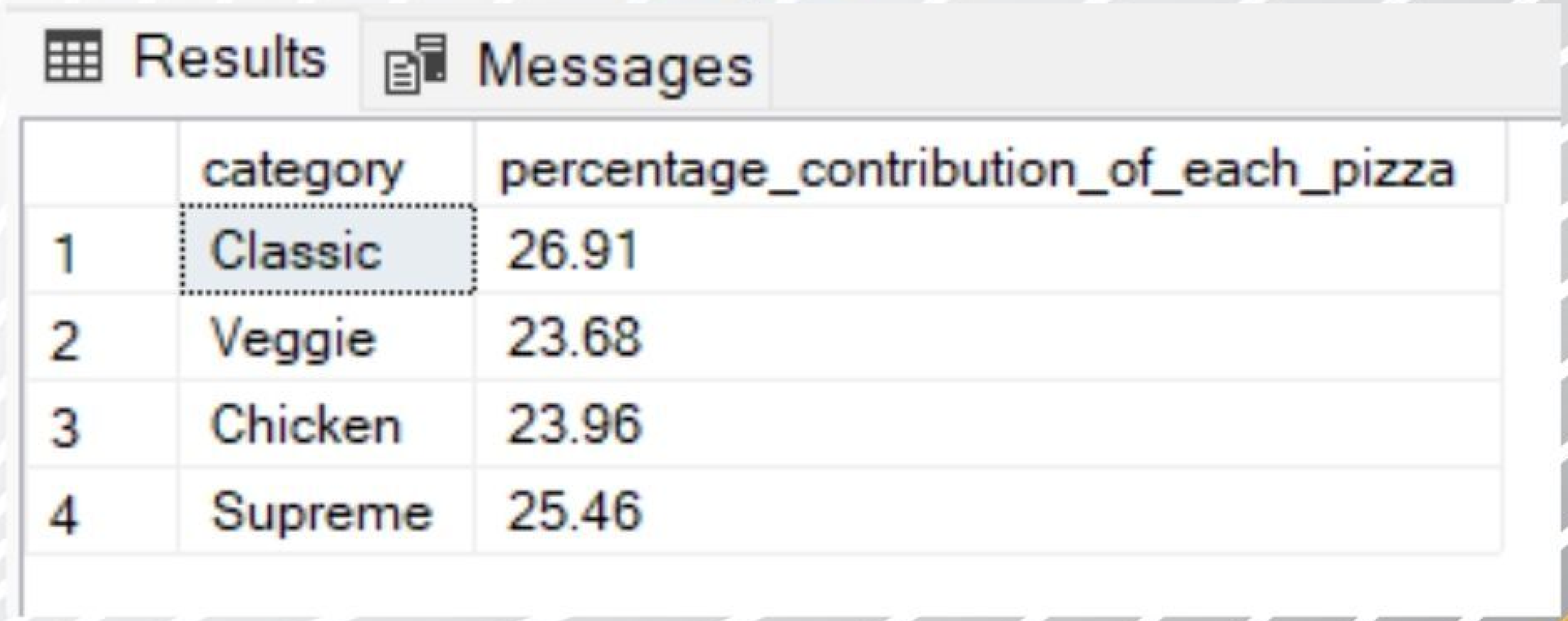
```
   Avg_Order_Per_Day
1  138
```

code



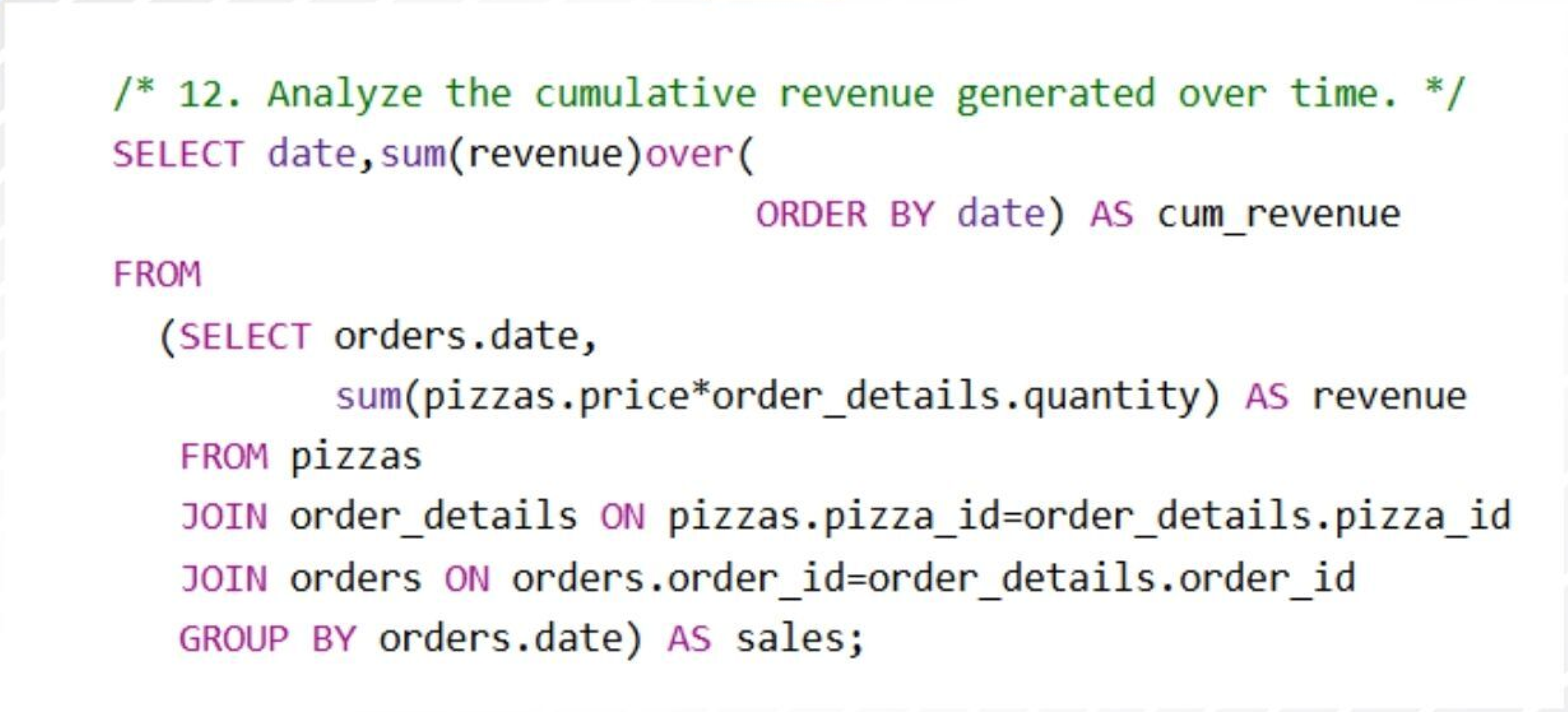
```sql
/* 11. Calculate the percentage contribution of each pizza type to total revenue. */
SELECT pizza_types.category,
       round(((sum(pizzas.price*order_details.quantity))/
                (SELECT round(sum(pizzas.price*order_details.quantity), 2) AS Total_revenue
                 FROM pizzas
                 JOIN order_details ON pizzas.pizza_id = order_details.pizza_id))*100, 2) AS percentage_of_each_pizza
FROM pizzas
JOIN order_details ON pizzas.pizza_id=order_details.pizza_id
JOIN pizza_types ON pizzas.pizza_type_id=pizza_types.pizza_type_id
GROUP BY pizza_types.category;
```

output



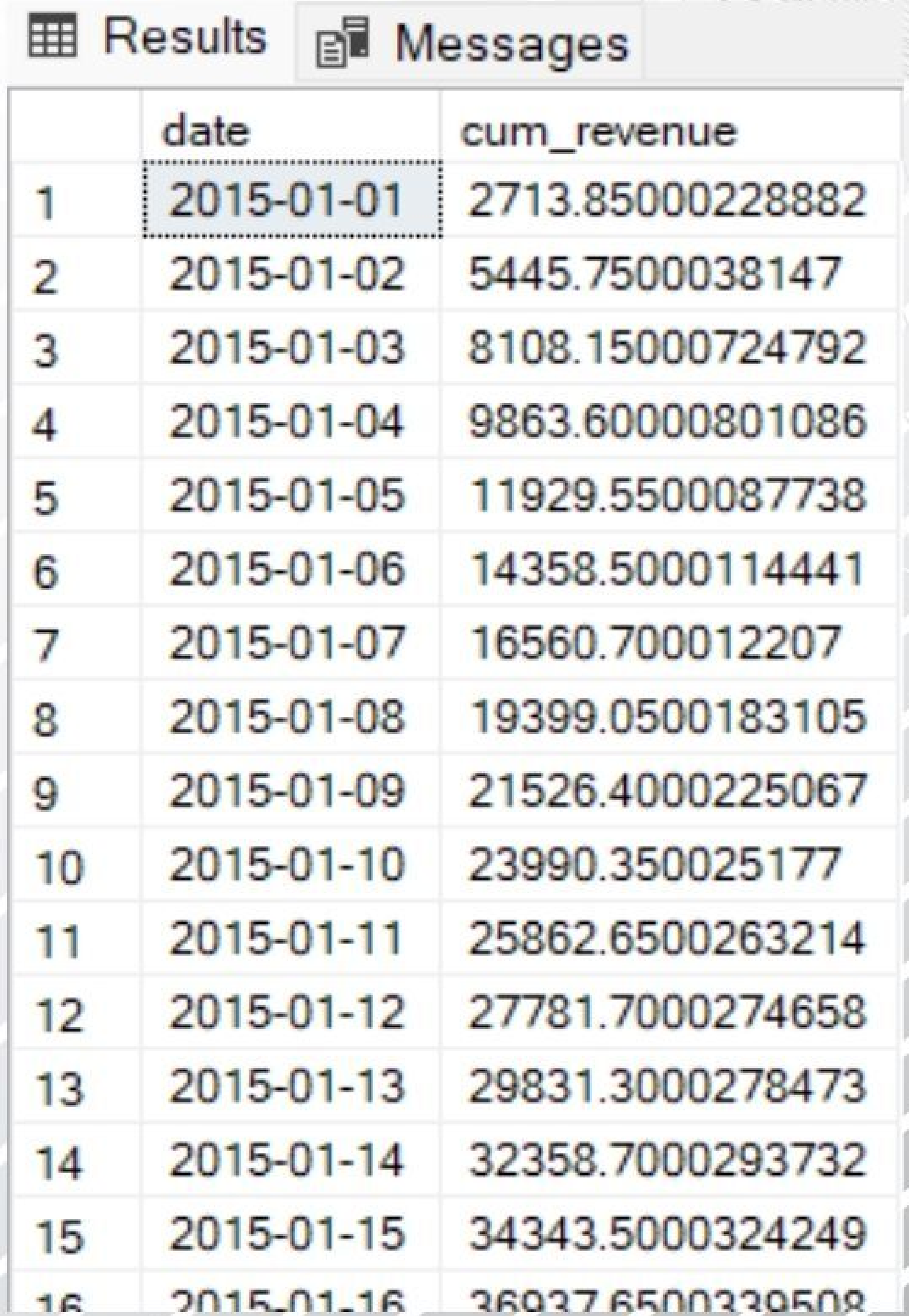
	category	percentage_contribution_of_each_pizza
1	Classic	26.91
2	Veggie	23.68
3	Chicken	23.96
4	Supreme	25.46

code



```
/* 12. Analyze the cumulative revenue generated over time. */
SELECT date,sum(revenue)over(
                            ORDER BY date) AS cum_revenue
FROM
  (SELECT orders.date,
          sum(pizzas.price*order_details.quantity) AS revenue
   FROM pizzas
   JOIN order_details ON pizzas.pizza_id=order_details.pizza_id
   JOIN orders ON orders.order_id=order_details.order_id
   GROUP BY orders.date) AS sales;
```

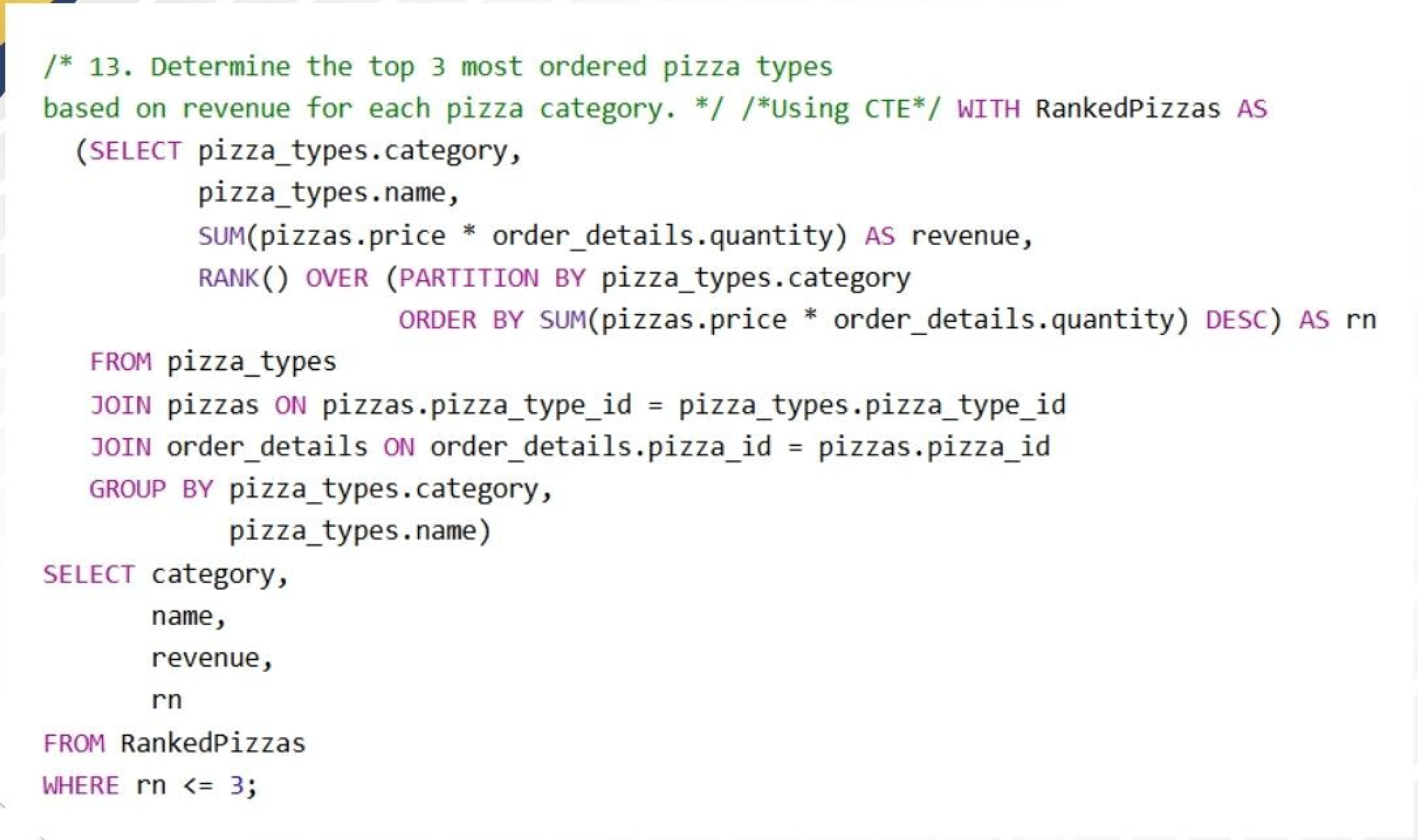
output



Results | Messages

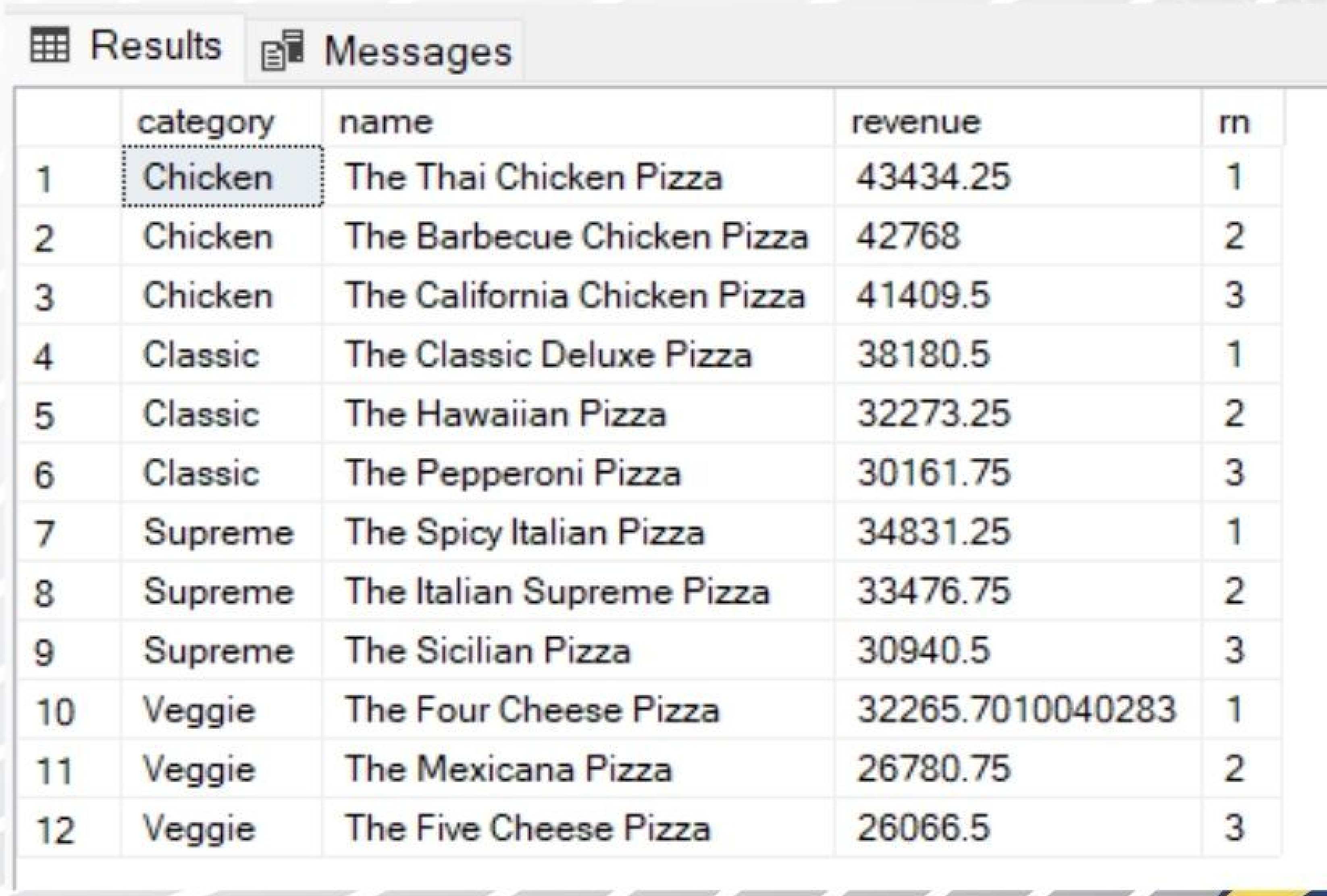
	date	cum_revenue
1	2015-01-01	2713.8500228882
2	2015-01-02	5445.7500038147
3	2015-01-03	8108.15000724792
4	2015-01-04	9863.60000801086
5	2015-01-05	11929.5500087738
6	2015-01-06	14358.5000114441
7	2015-01-07	16560.700012207
8	2015-01-08	19399.0500183105
9	2015-01-09	21526.4000225067
10	2015-01-10	23990.350025177
11	2015-01-11	25862.6500263214
12	2015-01-12	27781.7000274658
13	2015-01-13	29831.3000278473
14	2015-01-14	32358.7000293732
15	2015-01-15	34343.5000324249
16	2015-01-16	36937.6500339508

code



```sql
/* 13. Determine the top 3 most ordered pizza types
based on revenue for each pizza category. */ /*Using CTE*/ WITH RankedPizzas AS
  (SELECT pizza_types.category,
          pizza_types.name,
          SUM(pizzas.price * order_details.quantity) AS revenue,
          RANK() OVER (PARTITION BY pizza_types.category
                       ORDER BY SUM(pizzas.price * order_details.quantity) DESC) AS rn
   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,
            pizza_types.name)
SELECT category,
      name,
      revenue,
      rn
FROM RankedPizzas
WHERE rn <= 3;
```

output



	category	name	revenue	rn
1	Chicken	The Thai Chicken Pizza	43434.25	1
2	Chicken	The Barbecue Chicken Pizza	42768	2
3	Chicken	The California Chicken Pizza	41409.5	3
4	Classic	The Classic Deluxe Pizza	38180.5	1
5	Classic	The Hawaiian Pizza	32273.25	2
6	Classic	The Pepperoni Pizza	30161.75	3
7	Supreme	The Spicy Italian Pizza	34831.25	1
8	Supreme	The Italian Supreme Pizza	33476.75	2
9	Supreme	The Sicilian Pizza	30940.5	3
10	Veggie	The Four Cheese Pizza	32265.7010040283	1
11	Veggie	The Mexicana Pizza	26780.75	2
12	Veggie	The Five Cheese Pizza	26066.5	3

THANK YOU