

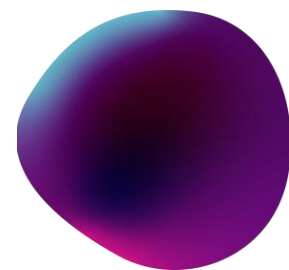
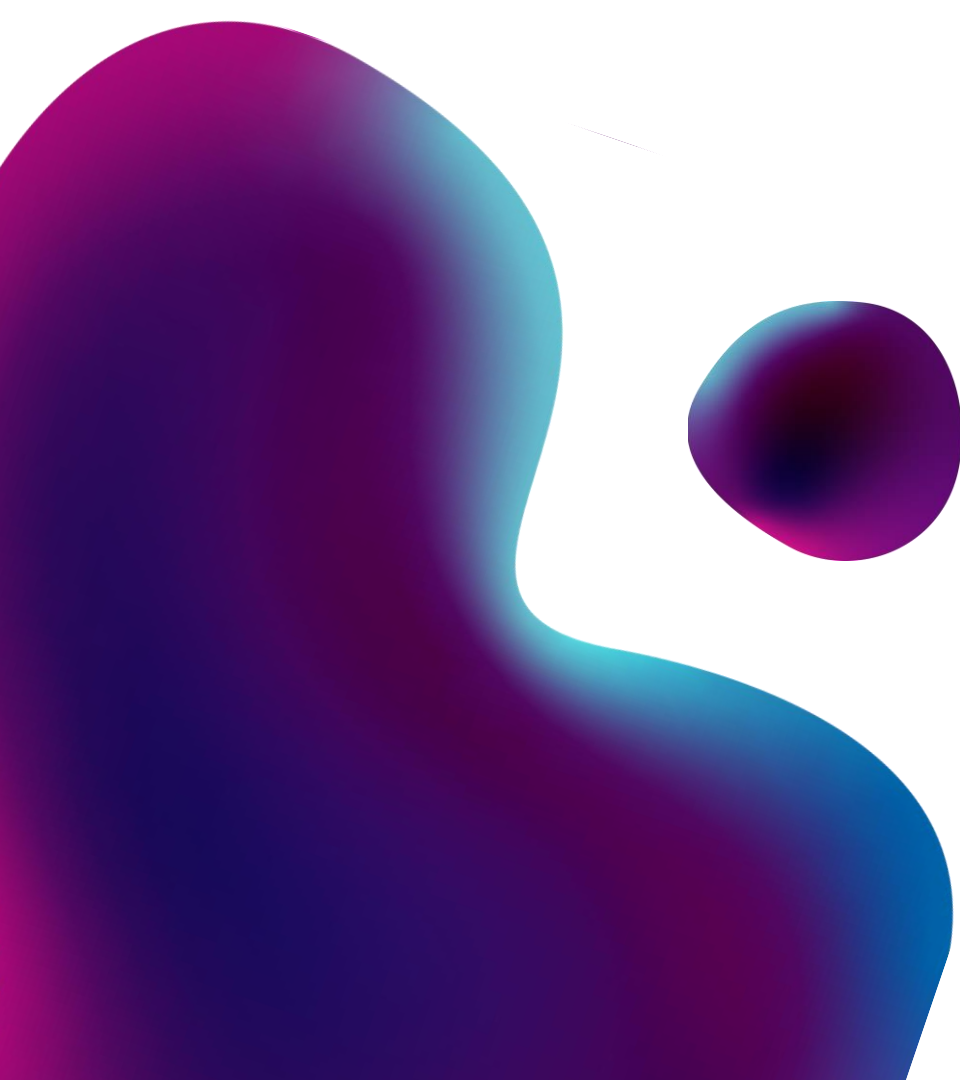


SALES ANALYSIS OF A RESTAURANT USING SQL

BY NIHARIKA SARKER NITU



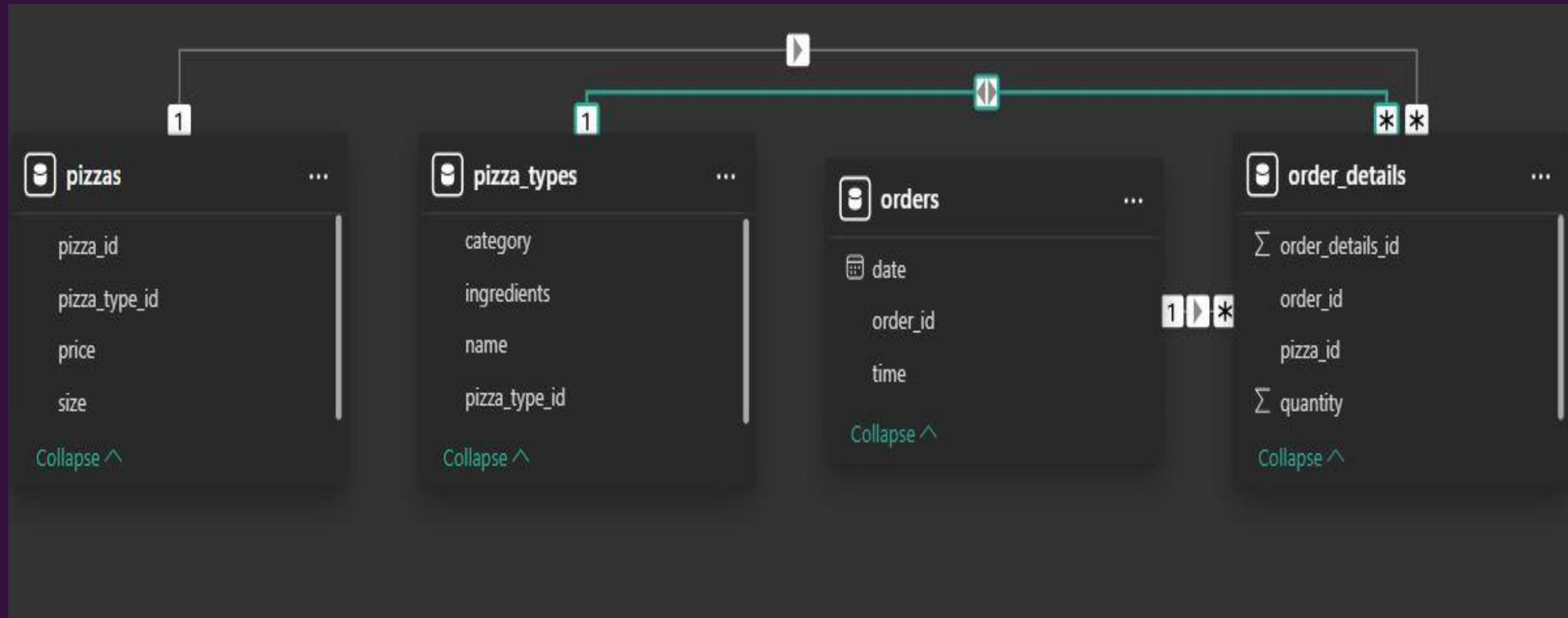
PROJECT OBJECTIVES



This project concentrates on sales analyzing a restaurant using SQL. The main purpose of this project is to analyze and identify the trend in sales key objectives, methodologies and anticipated outcomes for strategic execution. To ensure a structured path to success, the analysis is categorized according to a number of components that are intended to give a clear and thorough understanding of the dynamics of the project.



Schema of the Datasets

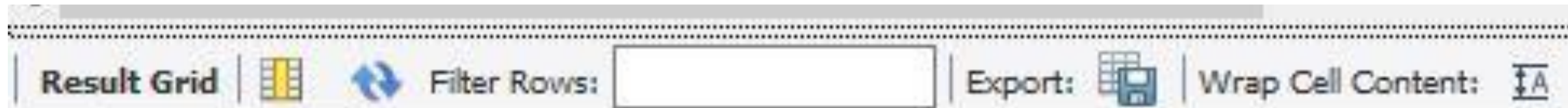


Retrieve the total number of orders placed.

01



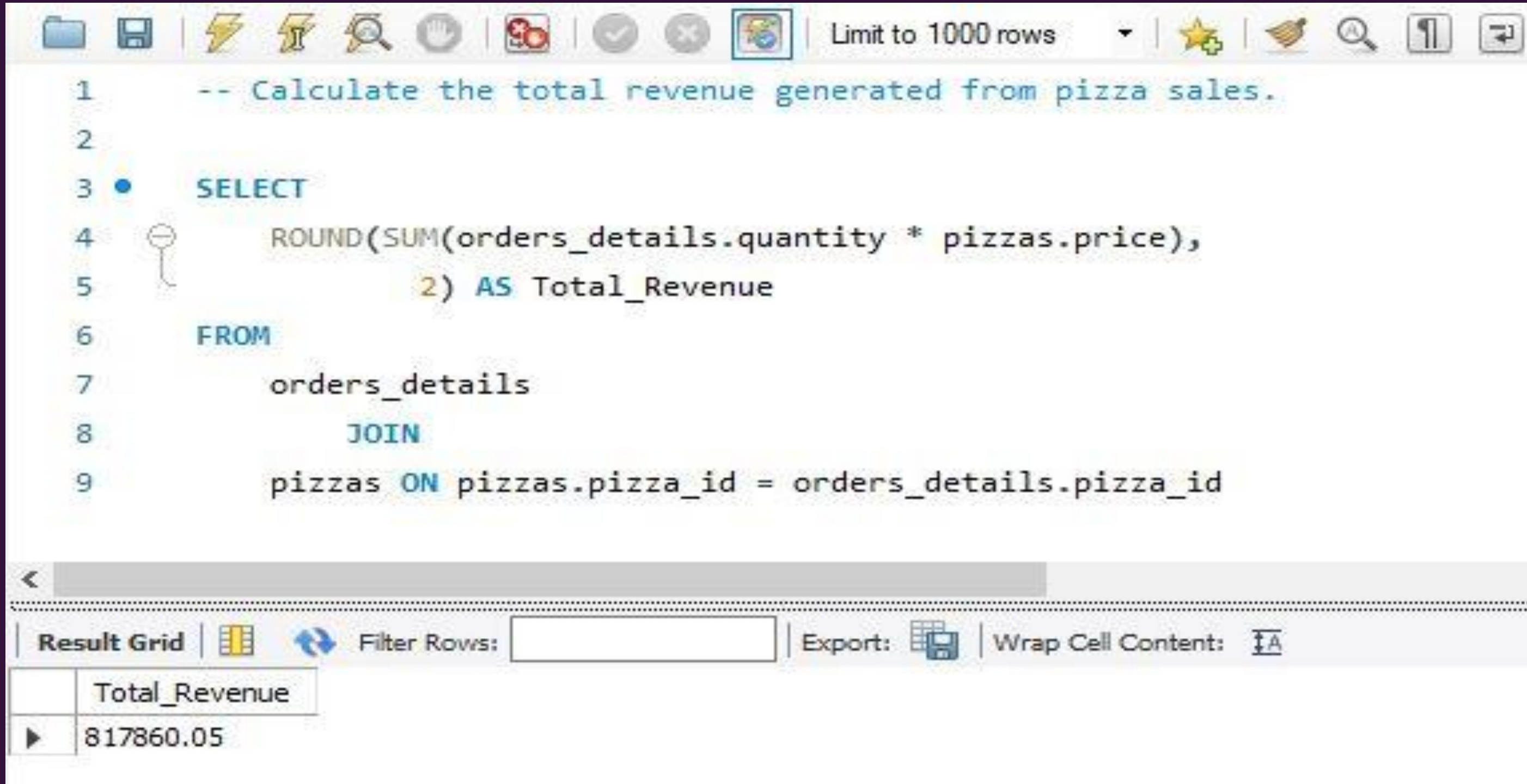
```
1  -- Basic:
2  -- Retrieve the total number of orders placed.
3
4  ● SELECT count(order_id) as Total_Orders from orders;
5
```



	Total_Orders
▶	21350



2. Calculate the total revenue generated from pizza sales.



The screenshot shows a SQL query editor with a toolbar at the top containing icons for file operations, execution, and settings. The query text is as follows:

```
1  -- Calculate the total revenue generated from pizza sales.
2
3  SELECT
4      ROUND(SUM(orders_details.quantity * pizzas.price),
5             2) AS Total_Revenue
6  FROM
7      orders_details
8      JOIN
9      pizzas ON pizzas.pizza_id = orders_details.pizza_id
```

Below the query editor, there is a horizontal scrollbar and a control bar with options: "Result Grid", a grid icon, "Filter Rows:" with an input field, "Export:" with a download icon, and "Wrap Cell Content:" with a text icon.

	Total_Revenue
▶	817860.05

03

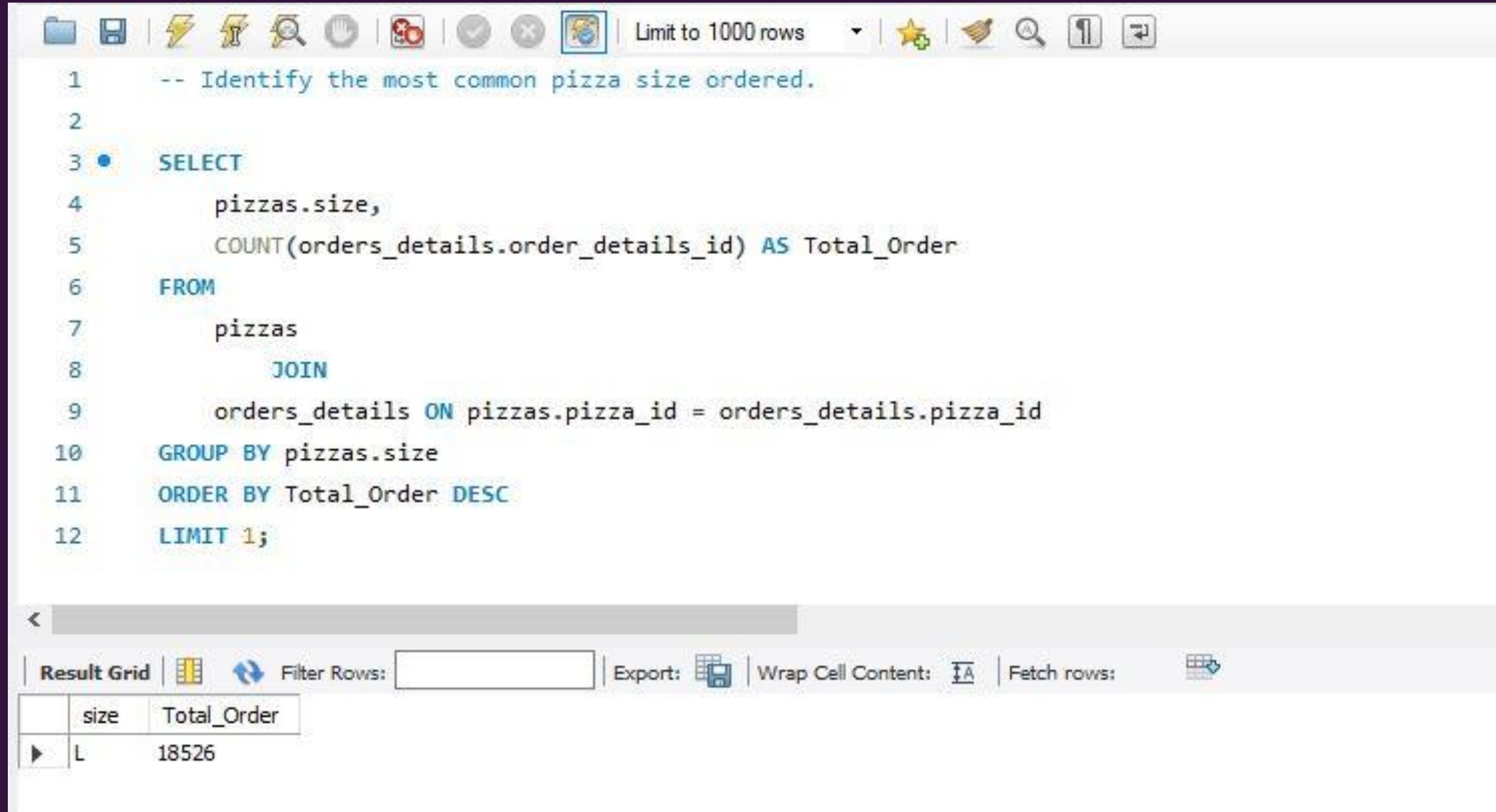
Identify the highest-priced pizza.

```
1  -- Identify the highest-priced pizza.
2
3  SELECT
4      pizza_types.name, pizzas.price
5  FROM
6      pizza_types
7      JOIN
8      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
9  ORDER BY pizzas.price DESC
10 LIMIT 1;
```

Result Grid

	name	price
▶	The Greek Pizza	35.95

4. Identify the most common pizza size ordered.



The screenshot shows a SQL query editor with a toolbar at the top containing icons for file operations, execution, and settings. The query text is as follows:

```
1  -- Identify the most common pizza size ordered.
2
3  SELECT
4      pizzas.size,
5      COUNT(orders_details.order_details_id) AS Total_Order
6  FROM
7      pizzas
8      JOIN
9      orders_details ON pizzas.pizza_id = orders_details.pizza_id
10 GROUP BY pizzas.size
11 ORDER BY Total_Order DESC
12 LIMIT 1;
```

Below the query editor, there is a "Result Grid" section with a table showing the results of the query. The table has two columns: "size" and "Total_Order". The first row shows the result for the most common pizza size.

size	Total_Order
L	18526

05

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

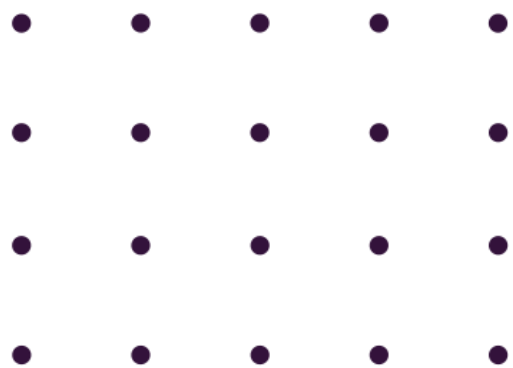


```
1  -- List the top 5 most ordered pizza types along with their quantities.
2  • SELECT
3      pizza_types.name,
4      SUM(orders_details.quantity) AS Total_Quantity_Ordered
5  FROM
6      pizza_types
7      JOIN
8      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
9      JOIN
10     orders_details ON orders_details.pizza_id = pizzas.pizza_id
11 GROUP BY pizza_types.name
12 ORDER BY Total_Quantity_Ordered DESC
13 LIMIT 5;
```

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

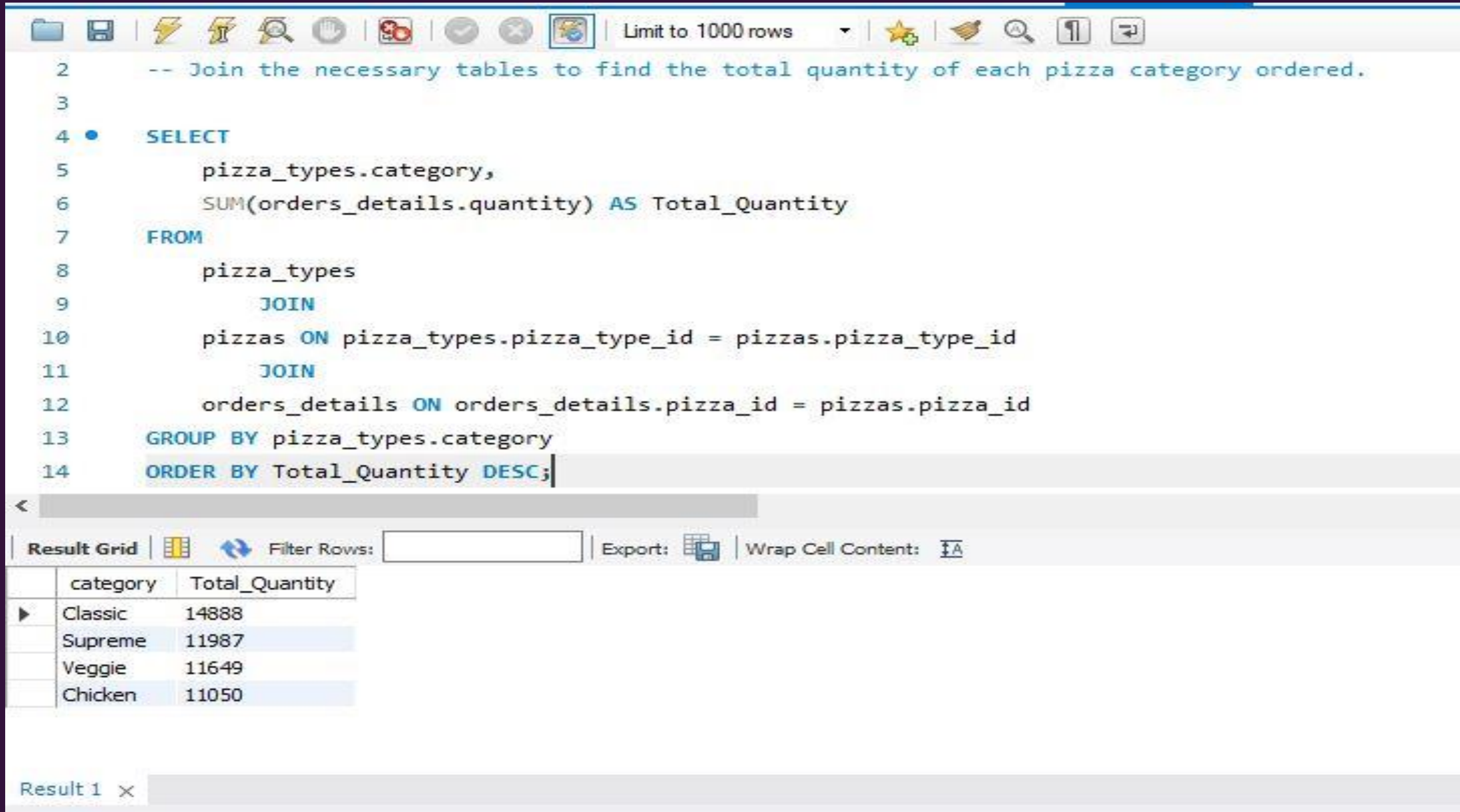
	name	Total_Quantity_Ordered
▶	The Classic Deluxe Pizza	2453
	The Barbecue Chicken Pizza	2432
	The Hawaiian Pizza	2422
	The Pepperoni Pizza	2418
	The Thai Chicken Pizza	2371

Result 1 x



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

06



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

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

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

	category	Total_Quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

Result 1 x

07

-
-
-
-
-
-
-
-
-
-

Determine the distribution of orders by hour of the day

The screenshot shows a SQL query editor window with a toolbar at the top. The query is as follows:

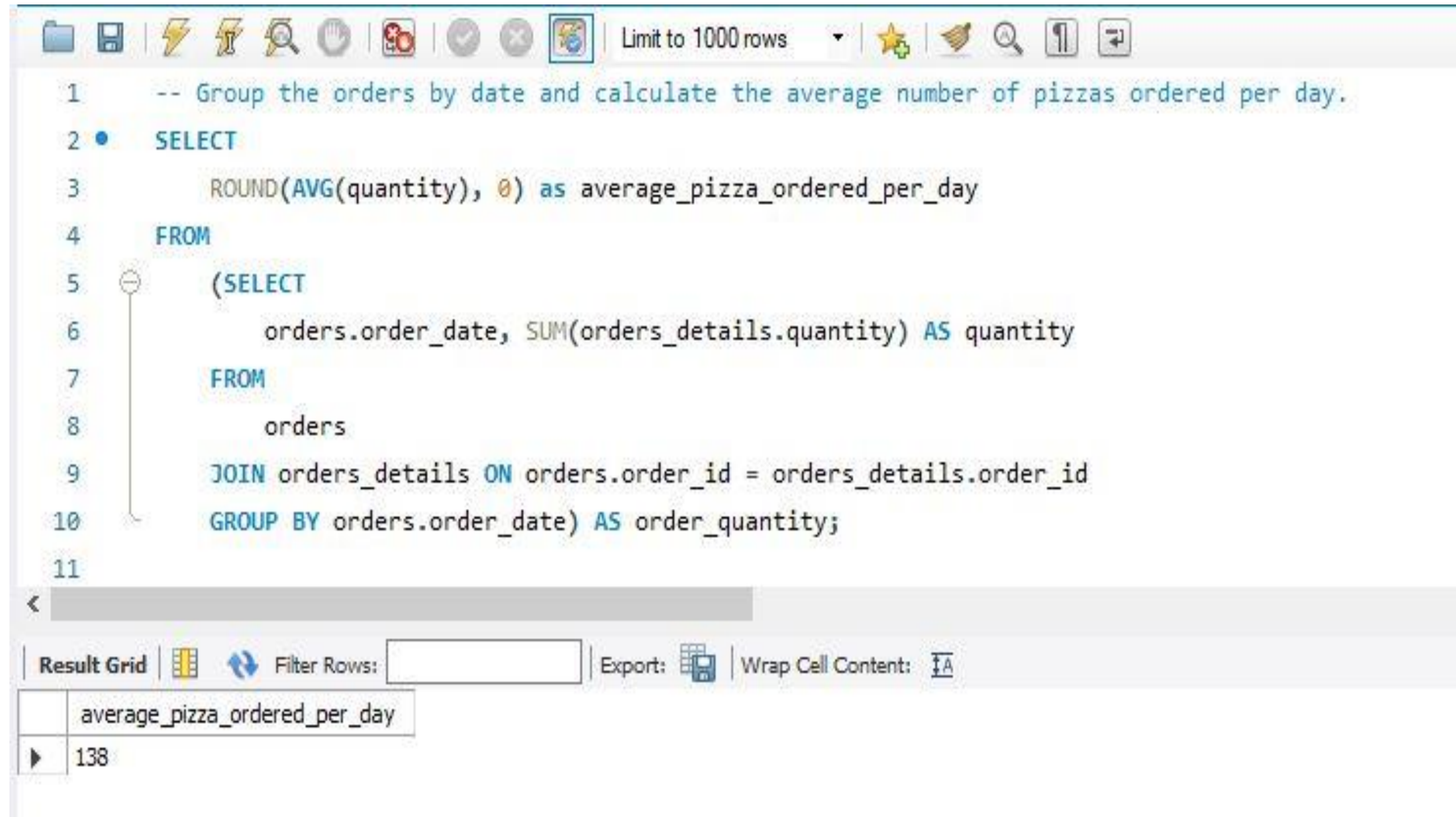
```
1  -- Determine the distribution of orders by hour of the day.
2  •  SELECT
3      HOUR(order_time) AS hour, COUNT(order_id) AS order_count
4  FROM
5      orders
6  GROUP BY HOUR(order_time);
```

Below the query editor is a 'Result Grid' section. It includes a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The results are displayed in a table with two columns: 'hour' and 'order_count'.

	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

At the bottom of the window, there is a tab labeled 'Result 1' with a close button (X).

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



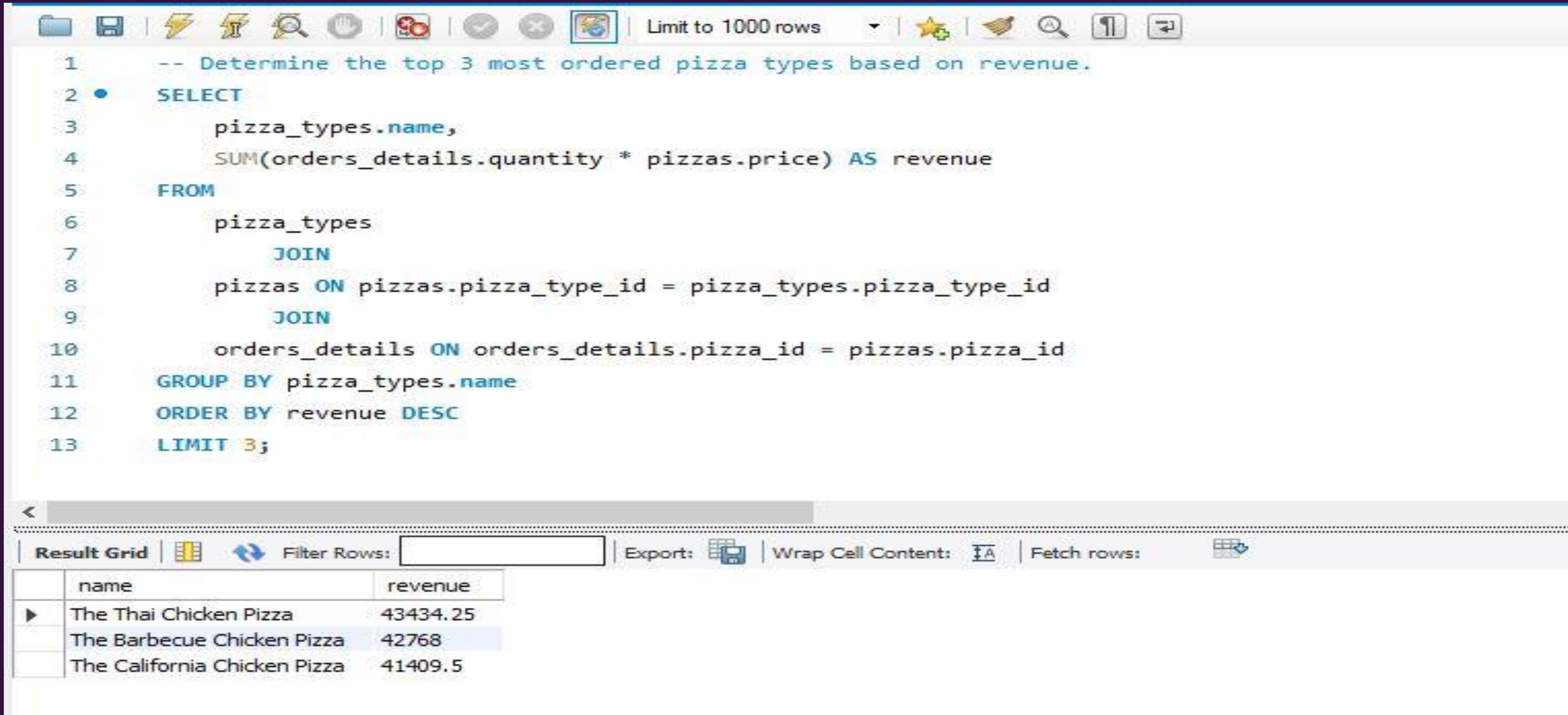
```
1  -- Group the orders by date and calculate the average number of pizzas ordered per day.
2  SELECT
3      ROUND(AVG(quantity), 0) as average_pizza_ordered_per_day
4  FROM
5      (SELECT
6          orders.order_date, SUM(orders_details.quantity) AS quantity
7      FROM
8          orders
9      JOIN orders_details ON orders.order_id = orders_details.order_id
10     GROUP BY orders.order_date) AS order_quantity;
```

The screenshot shows a SQL query editor with a toolbar at the top containing icons for file operations, execution, and search. The query is written in a multi-line editor with line numbers 1 through 11. The query calculates the average number of pizzas ordered per day by grouping orders by date and summing the quantities of pizzas ordered. The result is rounded to the nearest integer. Below the query editor, there is a 'Result Grid' section with a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The result grid shows a single row with the value 138 for the column 'average_pizza_ordered_per_day'.

average_pizza_ordered_per_day
138

10

Determine the top 3 most ordered pizza types based on revenue



The screenshot shows a SQL IDE window with a query editor and a result grid. The query is as follows:

```
1  -- Determine the top 3 most ordered pizza types based on revenue.
2  SELECT
3      pizza_types.name,
4      SUM(orders_details.quantity * pizzas.price) AS revenue
5  FROM
6      pizza_types
7      JOIN
8      pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
9      JOIN
10     orders_details ON orders_details.pizza_id = pizzas.pizza_id
11  GROUP BY pizza_types.name
12  ORDER BY revenue DESC
13  LIMIT 3;
```

The result grid displays the following data:

	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

11

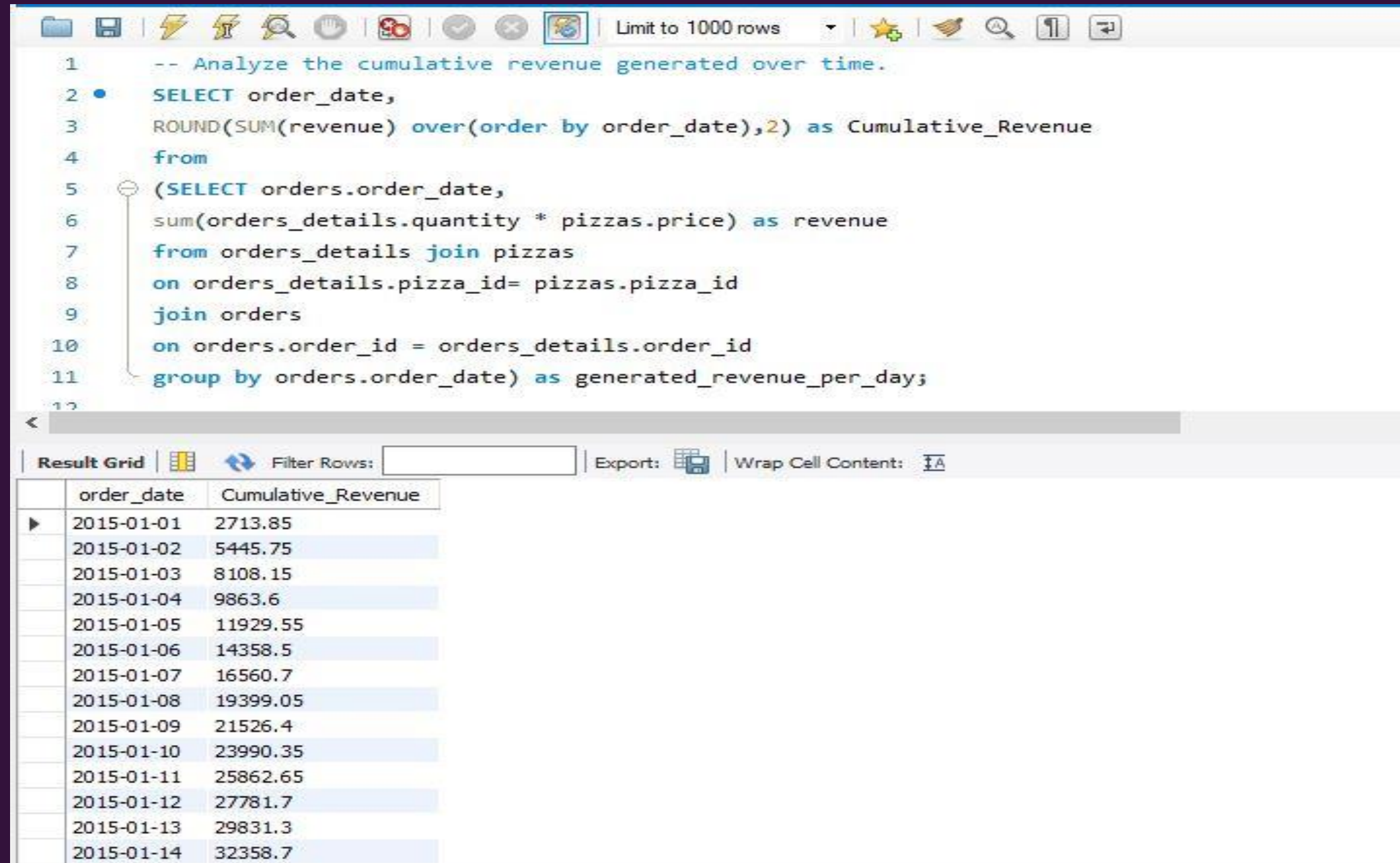
```
2  -- Calculate the percentage contribution of each pizza type to total revenue.
3  SELECT
4      pizza_types.category,
5      ROUND(SUM(orders_details.quantity * pizzas.price) / (SELECT
6          ROUND(SUM(orders_details.quantity * pizzas.price),
7              2) AS Total_Revenue
8          FROM
9              orders_details
10             JOIN
11                 pizzas ON pizzas.pizza_id = orders_details.pizza_id) * 100,
12          2) AS revenue
13  FROM
14      pizza_types
15      JOIN
16      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
17      JOIN
18      orders_details ON orders_details.pizza_id = pizzas.pizza_id
19  GROUP BY pizza_types.category
20  ORDER BY revenue DESC;
```

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

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

Analyze the cumulative revenue generated over time

12



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and search, along with a 'Limit to 1000 rows' dropdown. The SQL editor contains a query to calculate cumulative revenue. The query is as follows:

```
1  -- Analyze the cumulative revenue generated over time.
2  SELECT order_date,
3  ROUND(SUM(revenue) over(order by order_date),2) as Cumulative_Revenue
4  from
5  (SELECT orders.order_date,
6   sum(orders_details.quantity * pizzas.price) as revenue
7   from orders_details join pizzas
8   on orders_details.pizza_id= pizzas.pizza_id
9   join orders
10  on orders.order_id = orders_details.order_id
11  group by orders.order_date) as generated_revenue_per_day;
```

Below the editor, the 'Result Grid' tab is active, displaying the query results in a table. The table has two columns: 'order_date' and 'Cumulative_Revenue'. The results show the cumulative revenue for each day from 2015-01-01 to 2015-01-14.

order_date	Cumulative_Revenue
2015-01-01	2713.85
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	23990.35
2015-01-11	25862.65
2015-01-12	27781.7
2015-01-13	29831.3
2015-01-14	32358.7

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

13

```
1  -- Determine the top 3 most ordered pizza types based on revenue for each pizza category.
2
3  • SELECT category, name, revenue from
4
5  (SELECT category, name, revenue,
6   rank() over(partition by category order by revenue desc) as rn
7   from
8   (Select pizza_types.category, pizza_types.name,
9    SUM((orders_details.quantity) *pizzas.price) as revenue
10   from pizza_types join pizzas
11   ON pizza_types.pizza_type_id = pizzas.pizza_type_id
12   join orders_details
13   on orders_details.pizza_id = pizzas.pizza_id
14   Group by pizza_types.category, pizza_types.name) as A) as B
15  where rn<=3;
```

Result Grid			
Filter Rows:			
Export:			
Wrap Cell Content:			
category	name	revenue	
Chicken	The Thai Chicken Pizza	43434.25	
Chicken	The Barbecue Chicken Pizza	42768	
Chicken	The California Chicken Pizza	41409.5	
Classic	The Classic Deluxe Pizza	38180.5	
Classic	The Hawaiian Pizza	32273.25	
Classic	The Pepperoni Pizza	30161.75	
Supreme	The Spicy Italian Pizza	34831.25	
Supreme	The Italian Supreme Pizza	33476.75	
Supreme	The Sicilian Pizza	30940.5	
Veggie	The Four Cheese Pizza	32265.700000000065	
Veggie	The Mexicana Pizza	26780.75	
Veggie	The Five Cheese Pizza	26066.5	



THANK YOU