

# Pizza sales report

*I recently worked on a small project where I explored pizza sales data using SQL. The main goal was to practice writing queries and get hands-on experience with real-world-style data.*



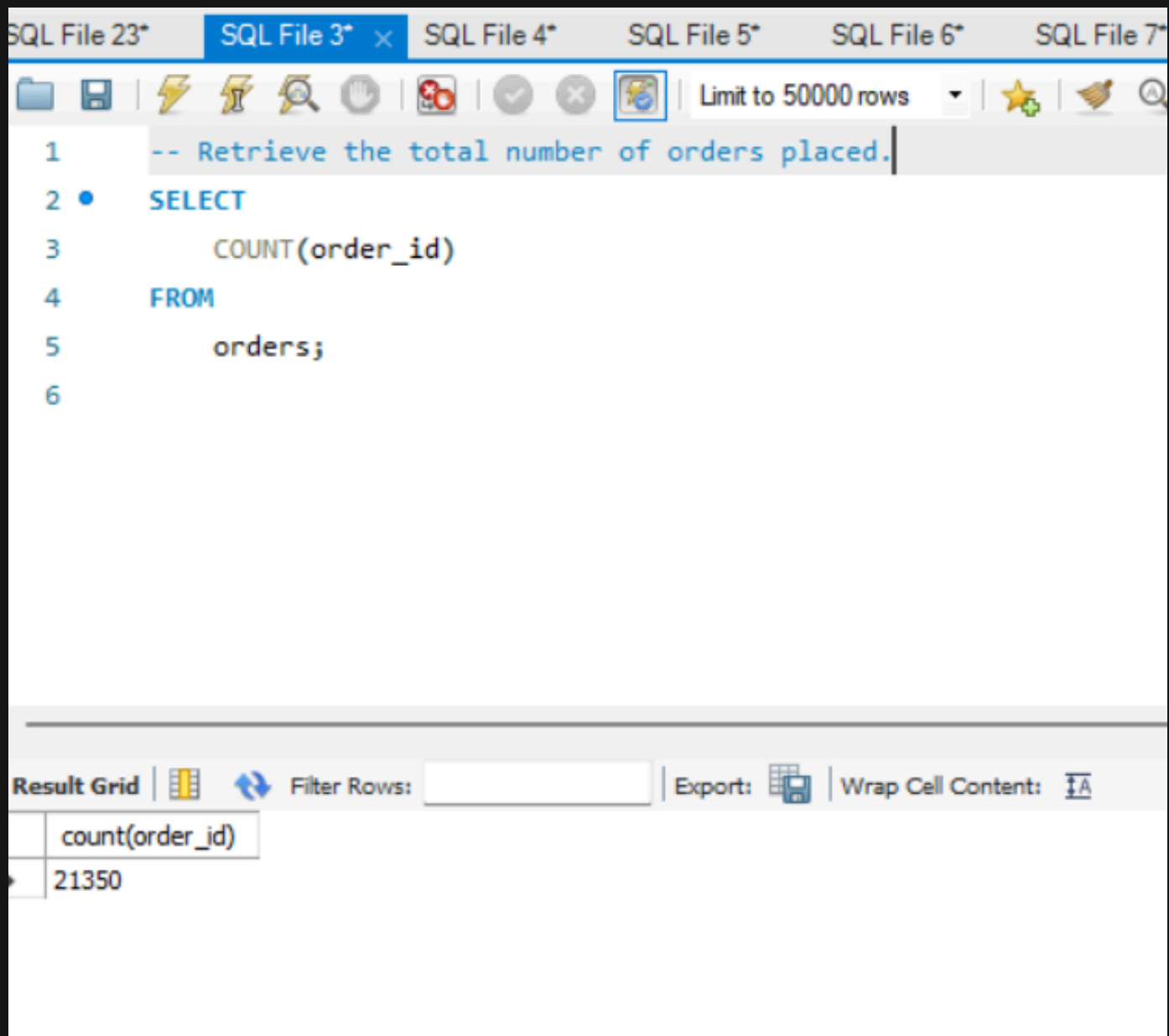
*In this project, I answered different business questions like:*

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
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.



*1. Retrieve the total number of orders placed.*



The screenshot shows a SQL IDE interface with multiple tabs at the top: "SQL File 23\*", "SQL File 3\*", "SQL File 4\*", "SQL File 5\*", "SQL File 6\*", and "SQL File 7\*". The "SQL File 3\*" tab is active. Below the tabs is a toolbar with various icons, including a "Limit to 50000 rows" dropdown. The main editor area contains the following SQL code:

```
1  -- Retrieve the total number of orders placed.  
2  •  SELECT  
3      COUNT(order_id)  
4  FROM  
5      orders;  
6
```

Below the editor is a "Result Grid" section. It includes a "Filter Rows:" input field, an "Export:" button, and a "Wrap Cell Content:" checkbox. The result grid displays the following data:

count(order_id)
21350



*2. Calculate the total revenue generated from pizza sales.*

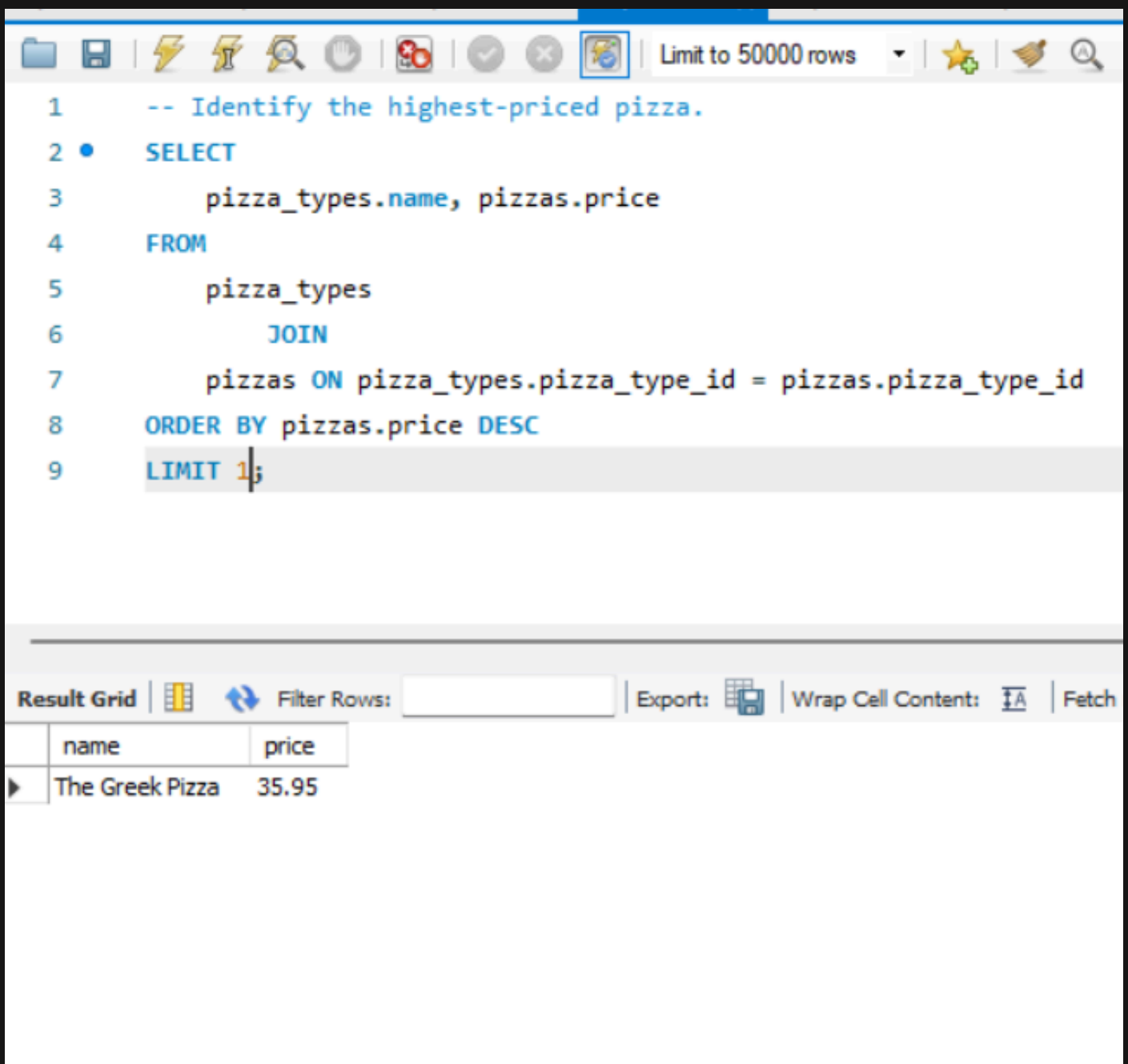
The screenshot shows a SQL IDE with multiple tabs. The active tab is 'SQL File 4\*', which contains the following SQL query:

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

Below the query editor, the 'Result Grid' is visible, showing the output of the query:

Total_sales
817860.05

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



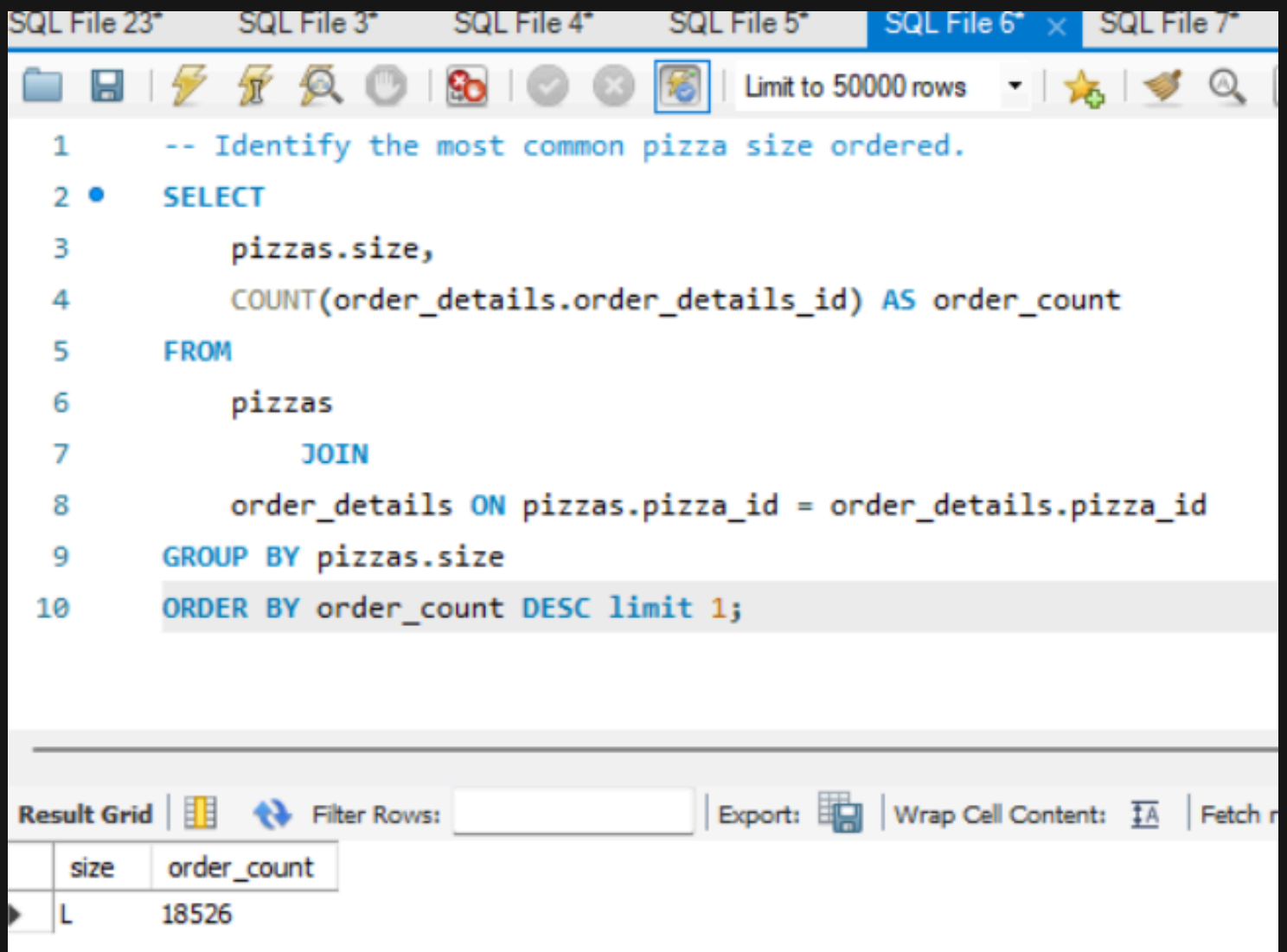
The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and a search icon. A dropdown menu indicates 'Limit to 50000 rows'. The SQL editor contains the following query:

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

The bottom toolbar includes 'Result Grid', 'Filter Rows', 'Export', 'Wrap Cell Content', and 'Fetch'. The result grid displays the following data:

	name	price
▶	The Greek Pizza	35.95

*4. Identify the most common pizza size ordered.*



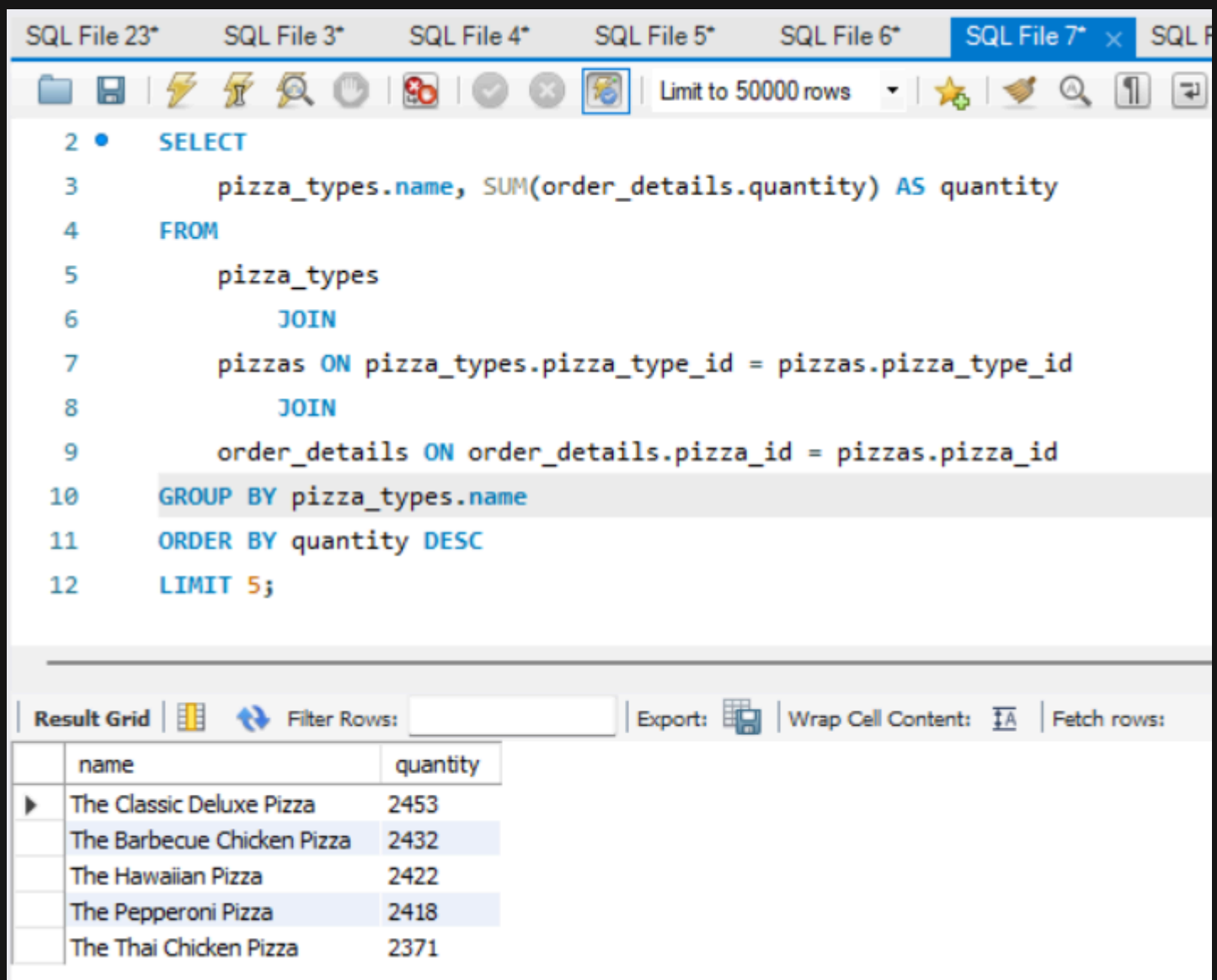
The screenshot shows a SQL IDE with multiple tabs. The active tab is 'SQL File 6\*'. The query editor contains the following SQL code:

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

Below the query editor, the 'Result Grid' is visible. It shows a table with two columns: 'size' and 'order\_count'. The first row of data shows 'L' with an 'order\_count' of 18526.

size	order_count
L	18526

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



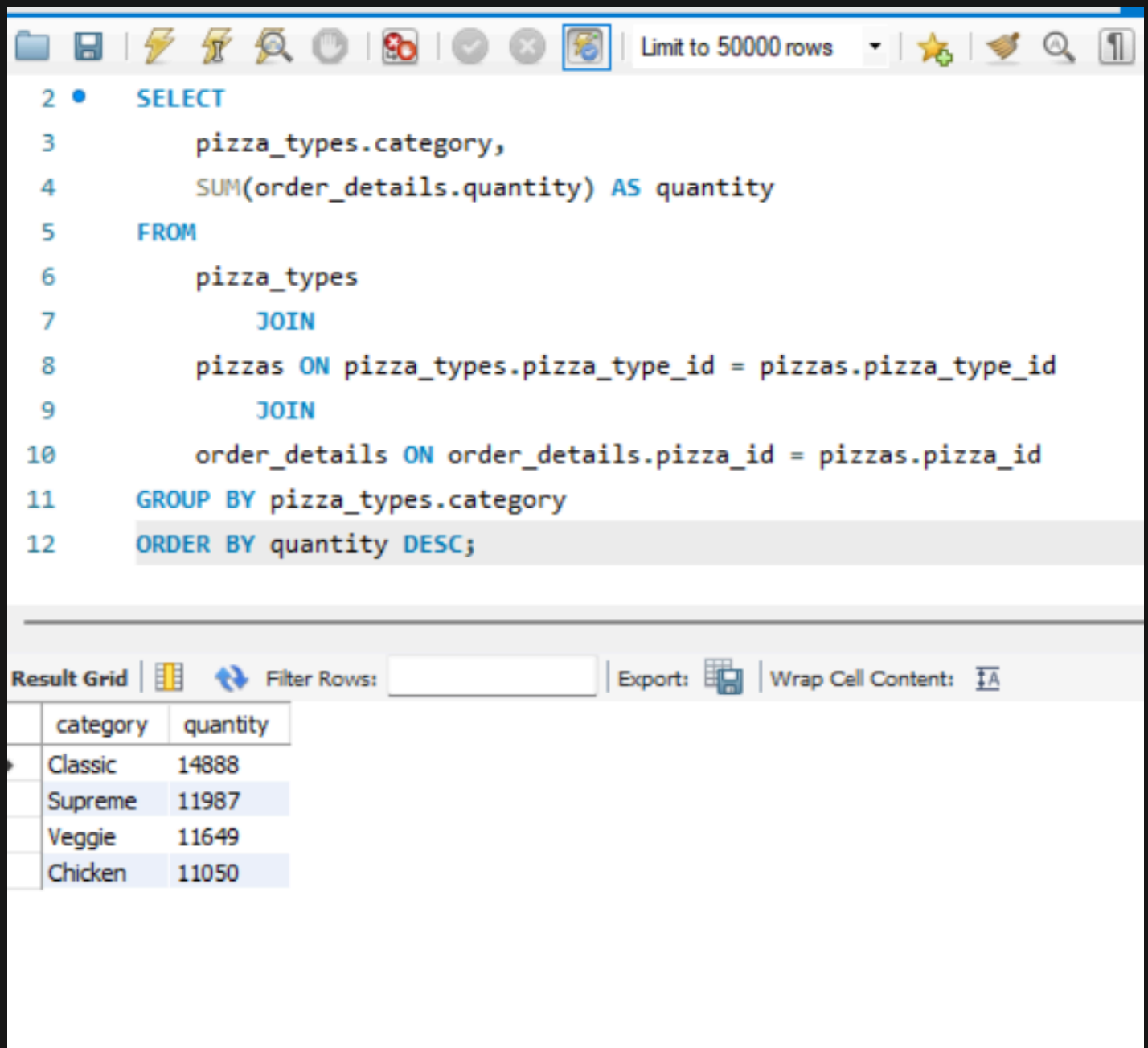
The screenshot shows a SQL IDE interface with multiple tabs at the top labeled 'SQL File 23\*', 'SQL File 3\*', 'SQL File 4\*', 'SQL File 5\*', 'SQL File 6\*', 'SQL File 7\*' (active), and 'SQL File 8\*'. The active tab contains the following SQL query:

```
2 • SELECT
3     pizza_types.name, SUM(order_details.quantity) AS quantity
4 FROM
5     pizza_types
6     JOIN
7     pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
8     JOIN
9     order_details ON order_details.pizza_id = pizzas.pizza_id
10 GROUP BY pizza_types.name
11 ORDER BY quantity DESC
12 LIMIT 5;
```

Below the query editor, the 'Result Grid' tab is active, displaying the results of the query in a table format. The table has two columns: 'name' and 'quantity'. The results are as follows:

	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

*6. find the total quantity of each pizza category ordered.*



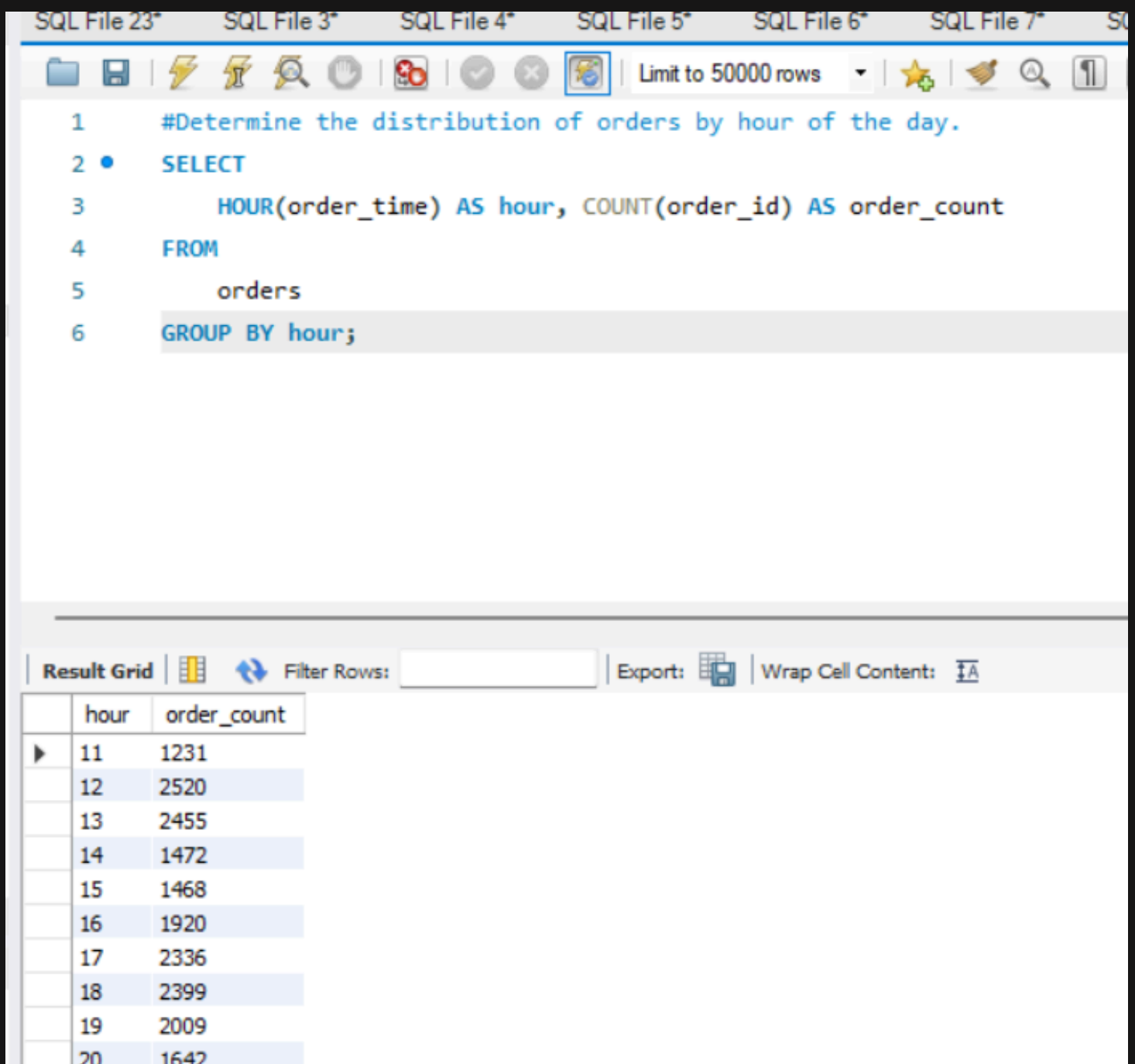
The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and search, along with a 'Limit to 50000 rows' dropdown. The SQL editor contains the following query:

```
2 • SELECT
3     pizza_types.category,
4     SUM(order_details.quantity) AS quantity
5 FROM
6     pizza_types
7     JOIN
8     pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
9     JOIN
10    order_details ON order_details.pizza_id = pizzas.pizza_id
11 GROUP BY pizza_types.category
12 ORDER BY quantity DESC;
```

Below the editor is the 'Result Grid' section, which 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: 'category' and 'quantity'.

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

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



The screenshot shows a SQL IDE with multiple tabs. The active tab is 'SQL File 5'. The query editor contains the following SQL code:

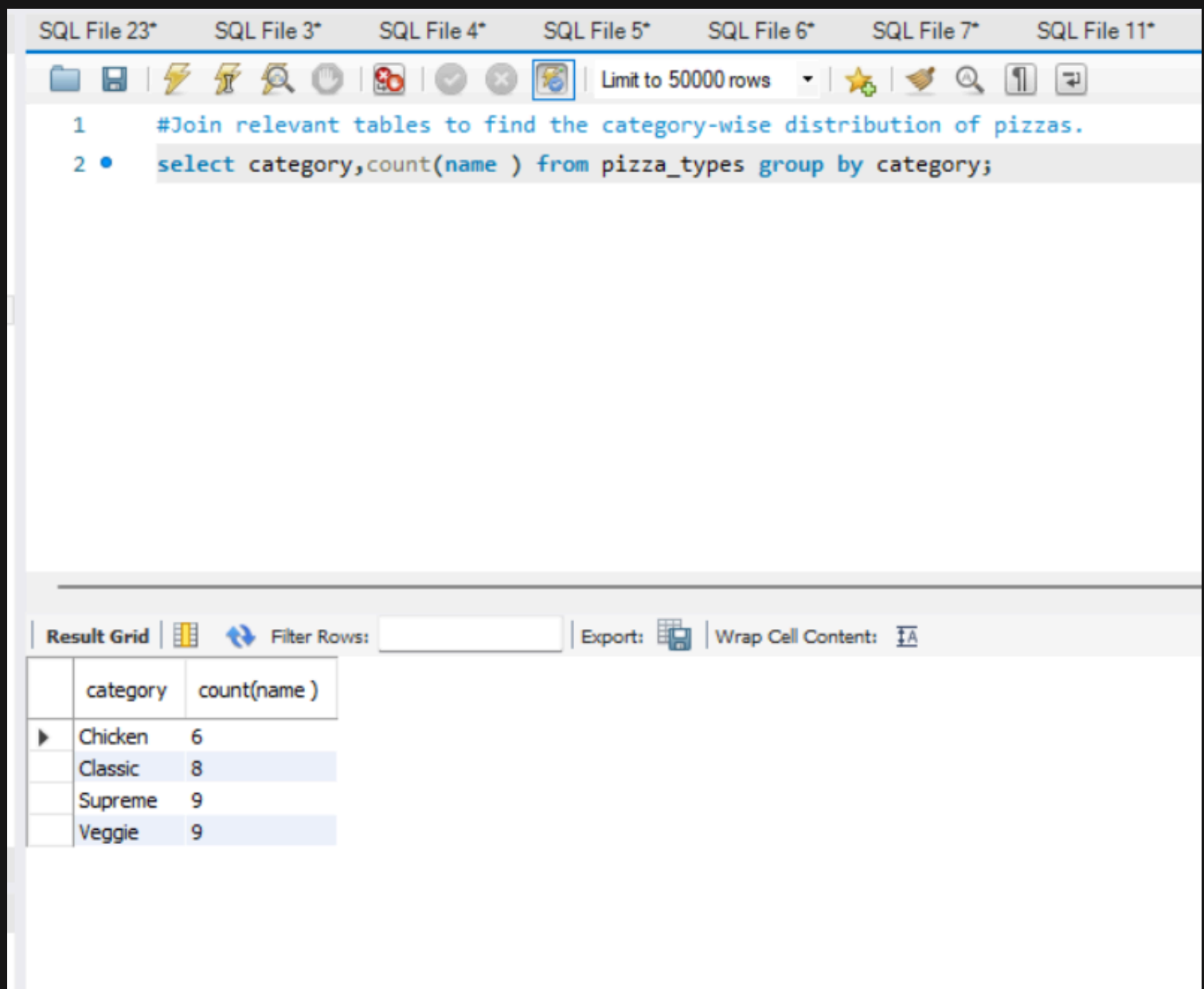
```
1 #Determine the distribution of orders by hour of the day.
2 • SELECT
3     HOUR(order_time) AS hour, COUNT(order_id) AS order_count
4 FROM
5     orders
6 GROUP BY hour;
```

The results are displayed in a table with the following data:

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



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



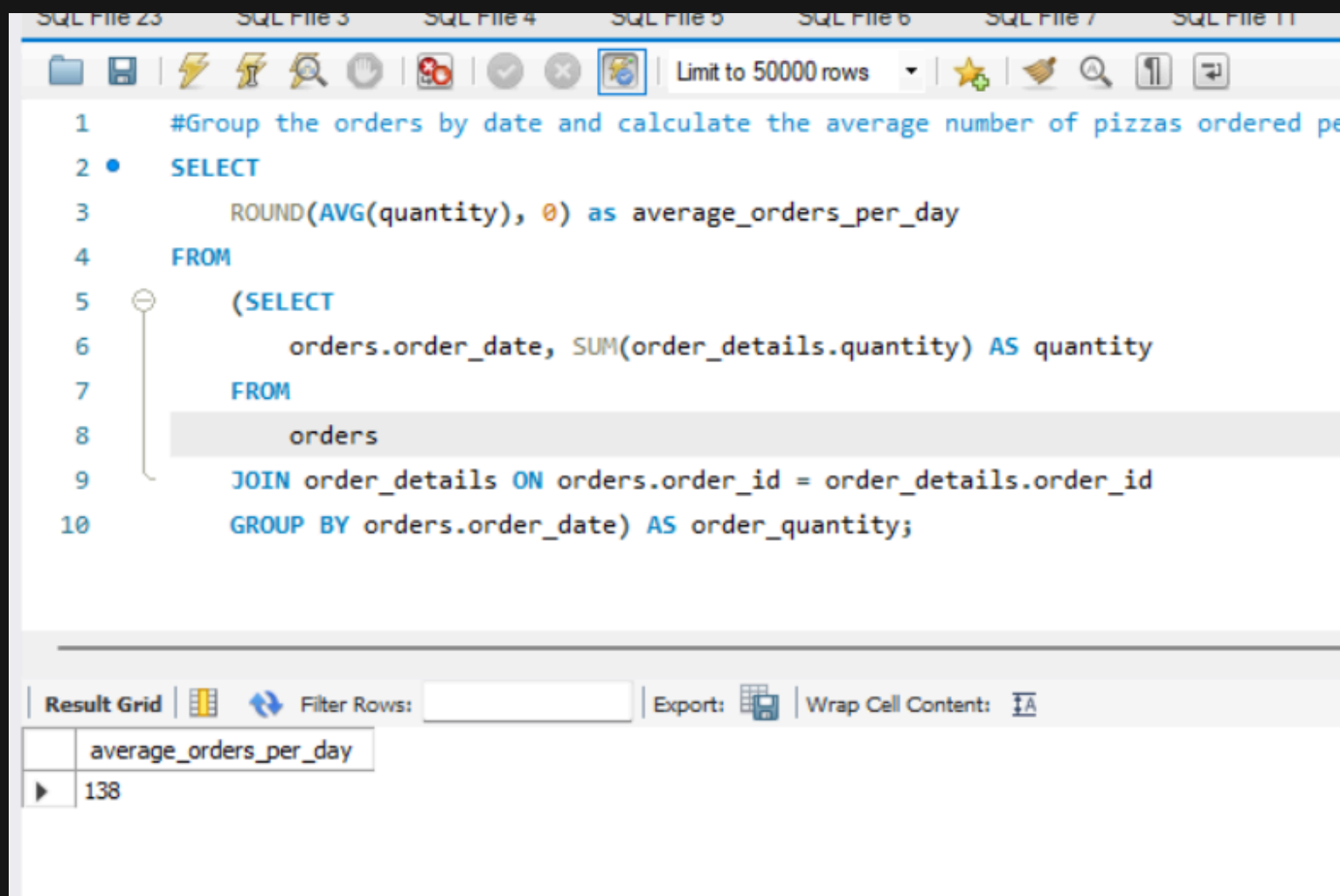
The screenshot shows a SQL IDE interface with multiple tabs at the top: SQL File 23\*, SQL File 3\*, SQL File 4\*, SQL File 5\*, SQL File 6\*, SQL File 7\*, and SQL File 11\*. The active tab is SQL File 5\*. The toolbar includes icons for file operations, a search icon, a limit dropdown set to 'Limit to 50000 rows', and other utility icons. The SQL editor contains the following text:

```
1 #Join relevant tables to find the category-wise distribution of pizzas.  
2 • select category,count(name ) from pizza_types group by category;
```

Below the editor, the 'Result Grid' tab is active, displaying the query results in a table. The table has two columns: 'category' and 'count(name)'. The results are as follows:

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

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



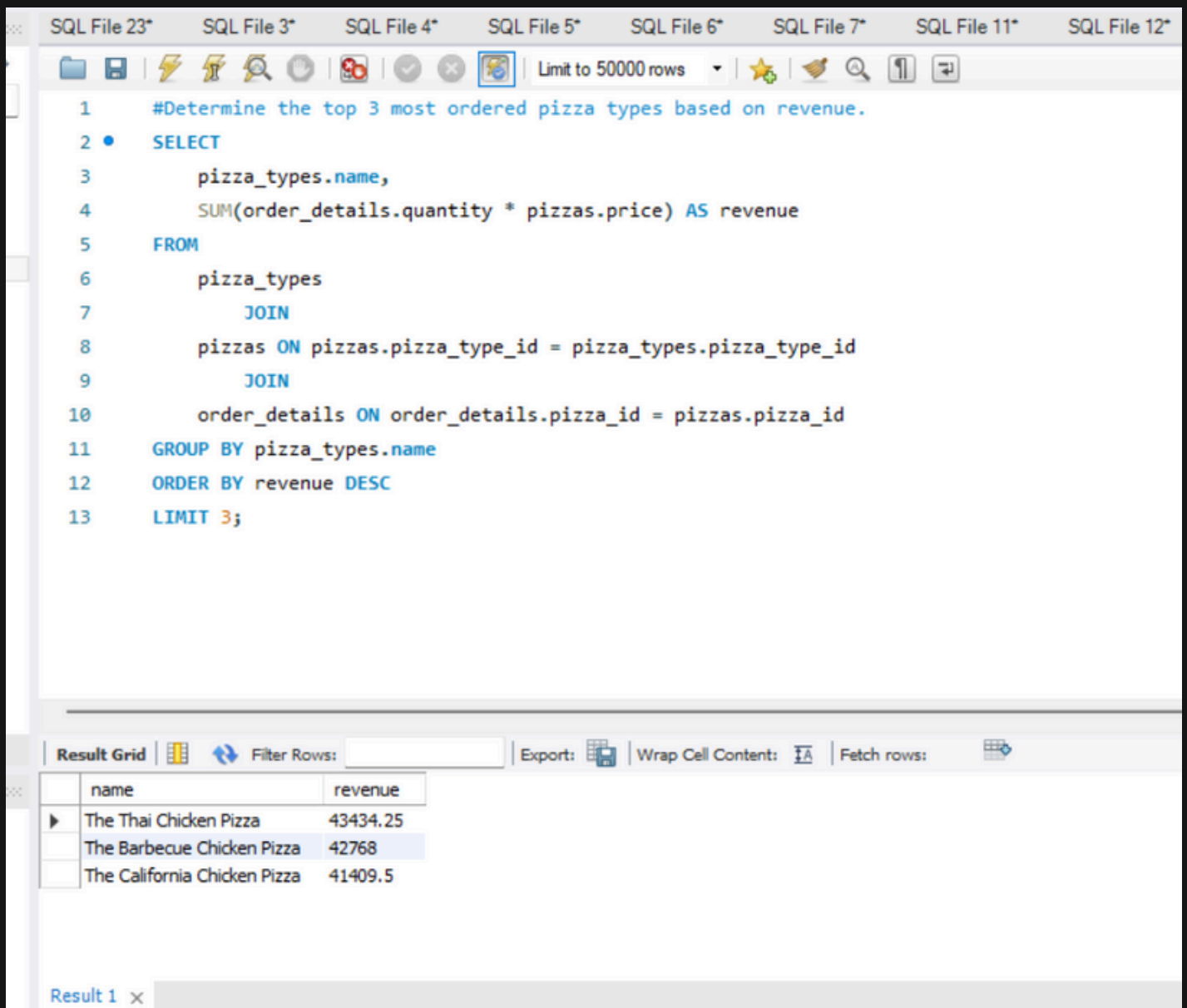
The screenshot shows a SQL IDE interface with a query editor and a result grid. The query editor contains the following SQL code:

```
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_orders_per_day
4 FROM
5     (SELECT
6         orders.order_date, SUM(order_details.quantity) AS quantity
7     FROM
8         orders
9     JOIN order_details ON orders.order_id = order_details.order_id
10    GROUP BY orders.order_date) AS order_quantity;
```

The result grid at the bottom shows the following data:

average_orders_per_day
138

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



The screenshot shows a SQL IDE with multiple tabs. The active tab is 'SQL File 5\*', which contains the following SQL query:

```
1 #Determine the top 3 most ordered pizza types based on revenue.
2 • SELECT
3     pizza_types.name,
4     SUM(order_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    order_details ON order_details.pizza_id = pizzas.pizza_id
11 GROUP BY pizza_types.name
12 ORDER BY revenue DESC
13 LIMIT 3;
```

Below the query editor, the 'Result Grid' is displayed, showing the results of the query. The grid has two columns: 'name' and 'revenue'. The results are as follows:

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

The IDE interface includes various toolbars at the top and bottom, and a status bar at the bottom left indicating 'Result 1'.