PIZZA HUT SALES ANALYSIS

Deriving business insights through SQL

Presented by
Sahil Thakur

Understanding the Sales Data

To analyze pizza sales data and extract actionable insights using SQL techniques.

Tools Used: SQL (PostgreSQL/MySQL/SQLite), Excel (for viewing), GitHub

Dataset source : Public sample dataset used for learning purposes

Tables Involved:

- orders
- order_details
- pizzas
- pizza_types

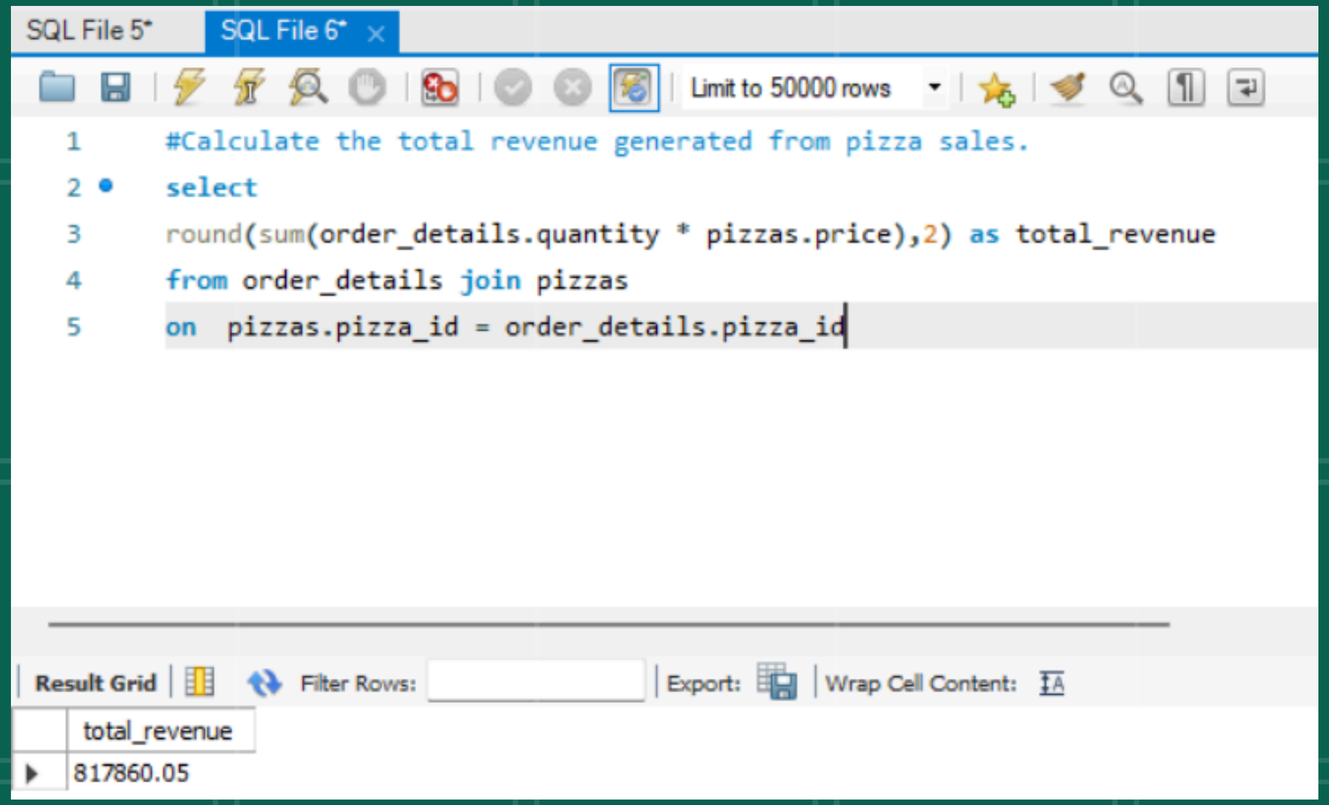
Retrieve the total number of orders placed.

```
1    #Retrieve the total number of orders placed.
2 •  select count(order_id) as Total_orders from pizzahut.orders;
```

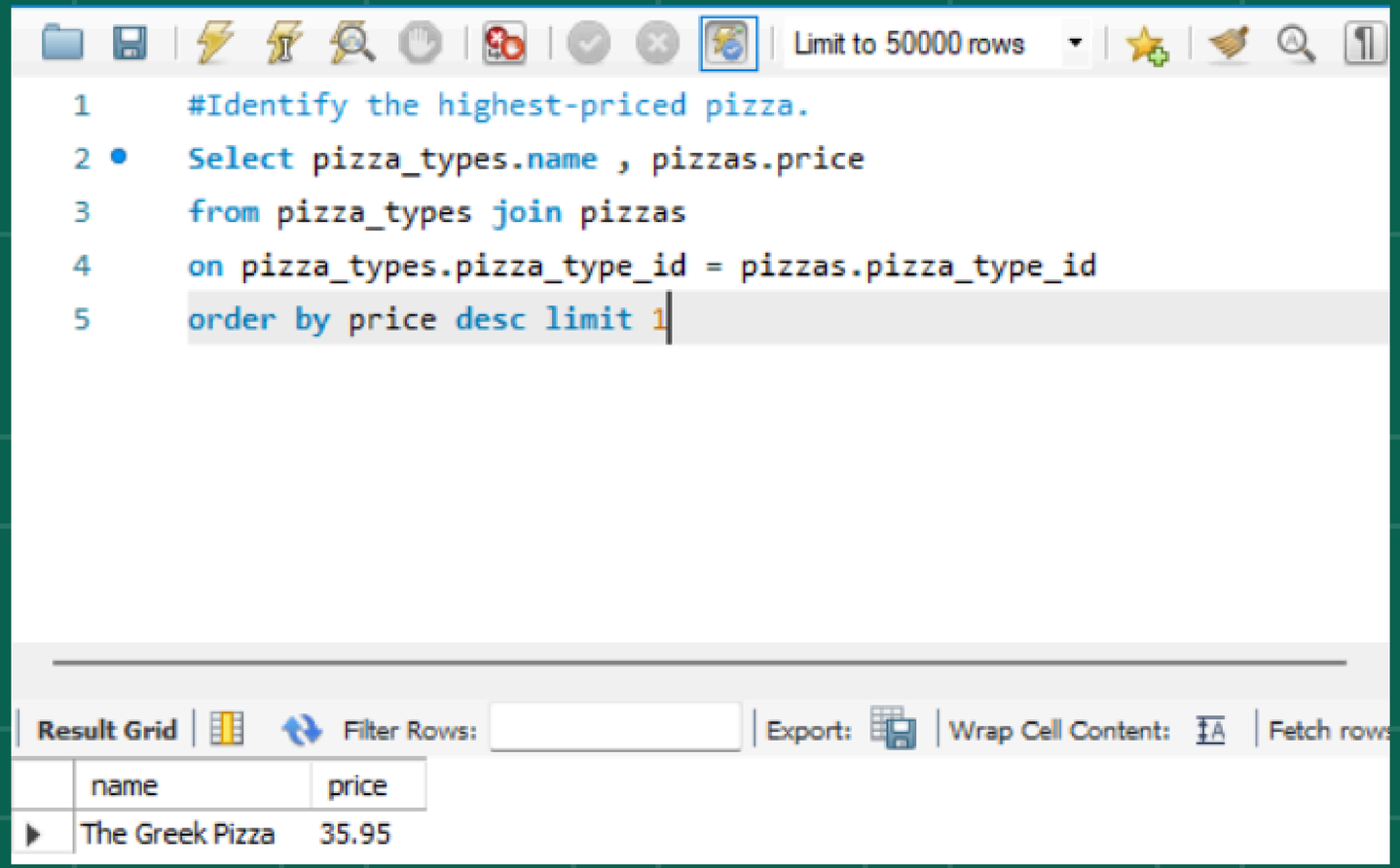
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	Total_orders
▶	21350

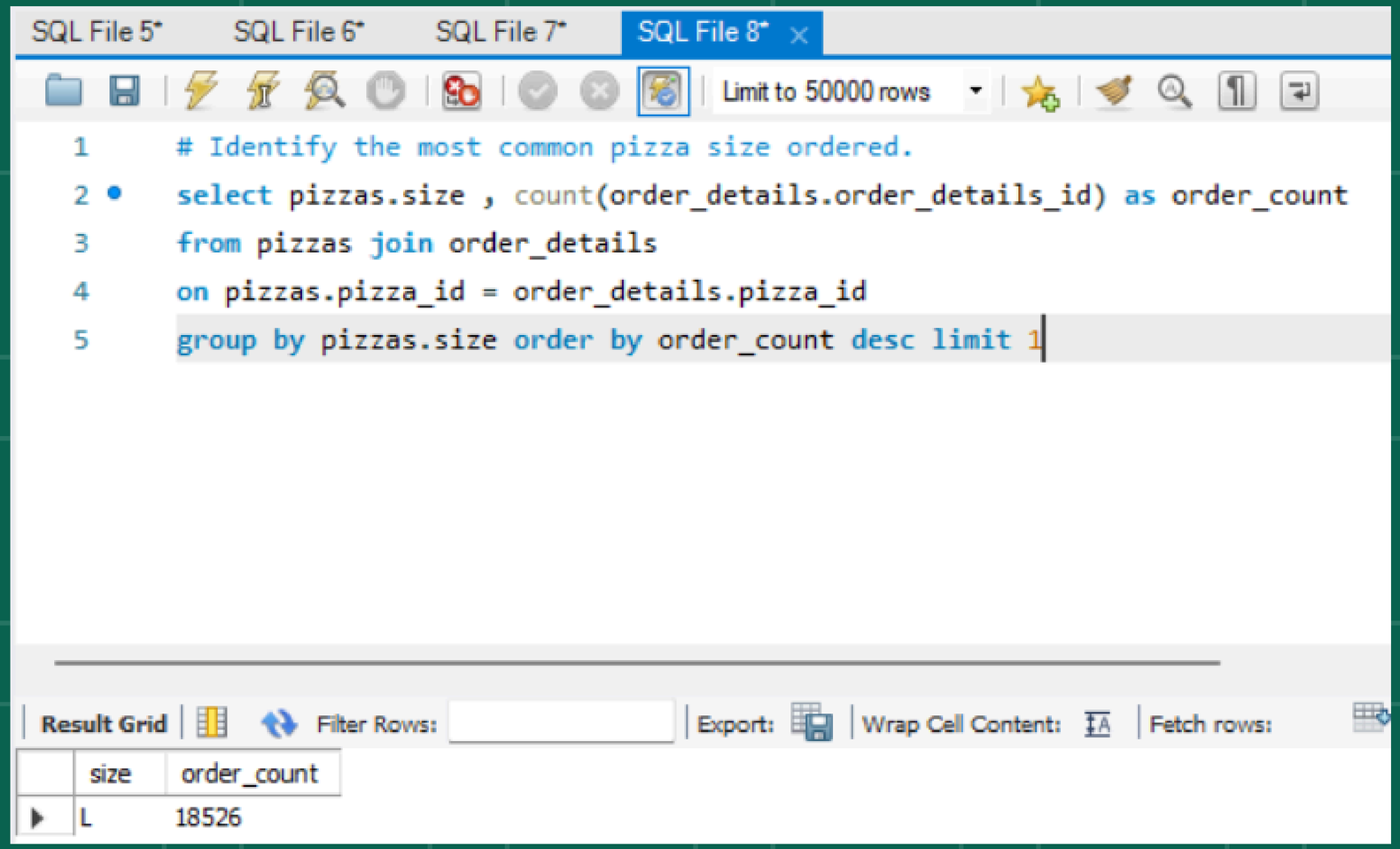
Calculate the total revenue generated from pizza sales.



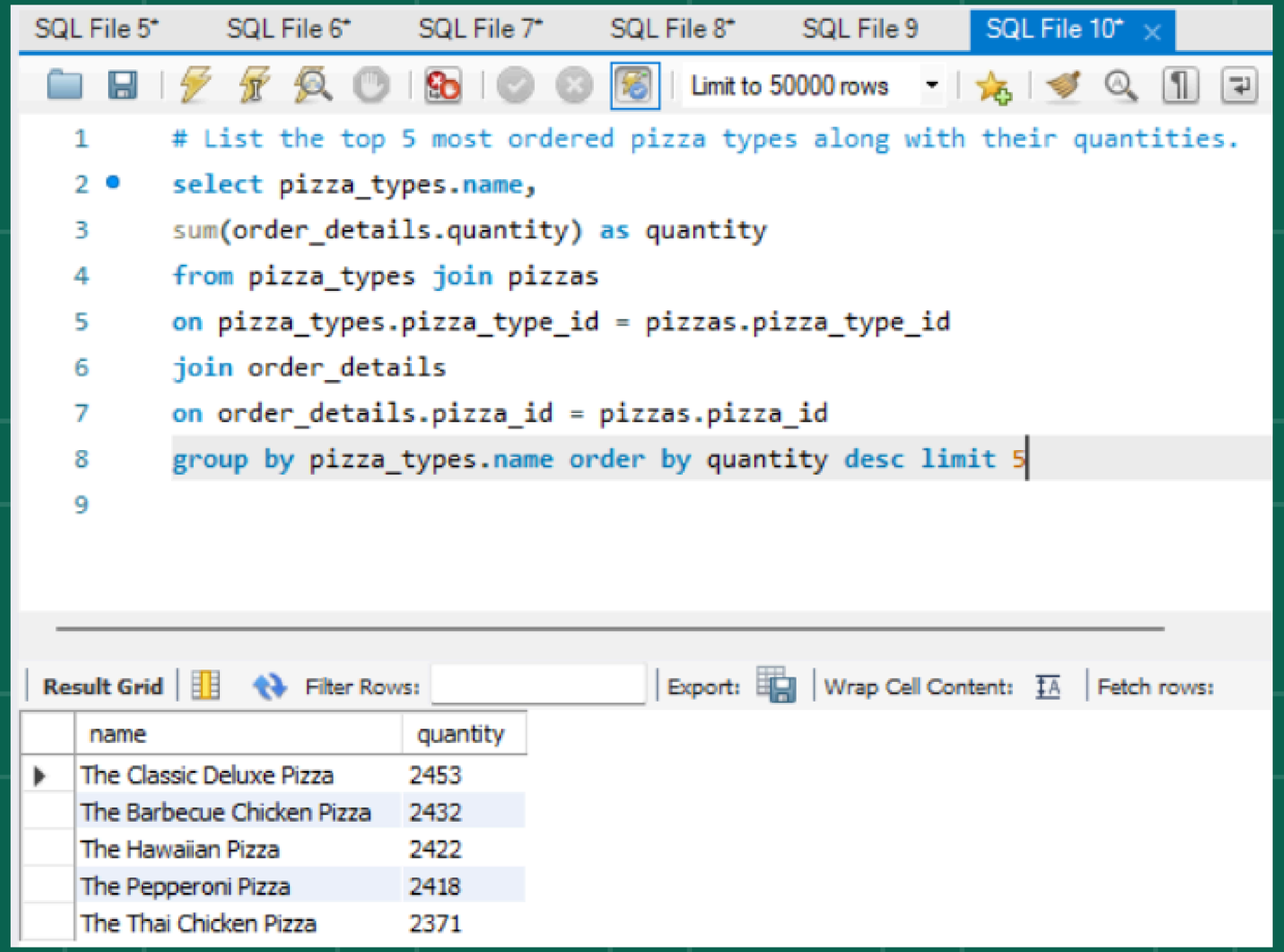
Identify the highest-priced .



Identify the most common pizza size ordered.



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



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

SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8* | SQL File 9 | SQL File 10* | SQL File 11*

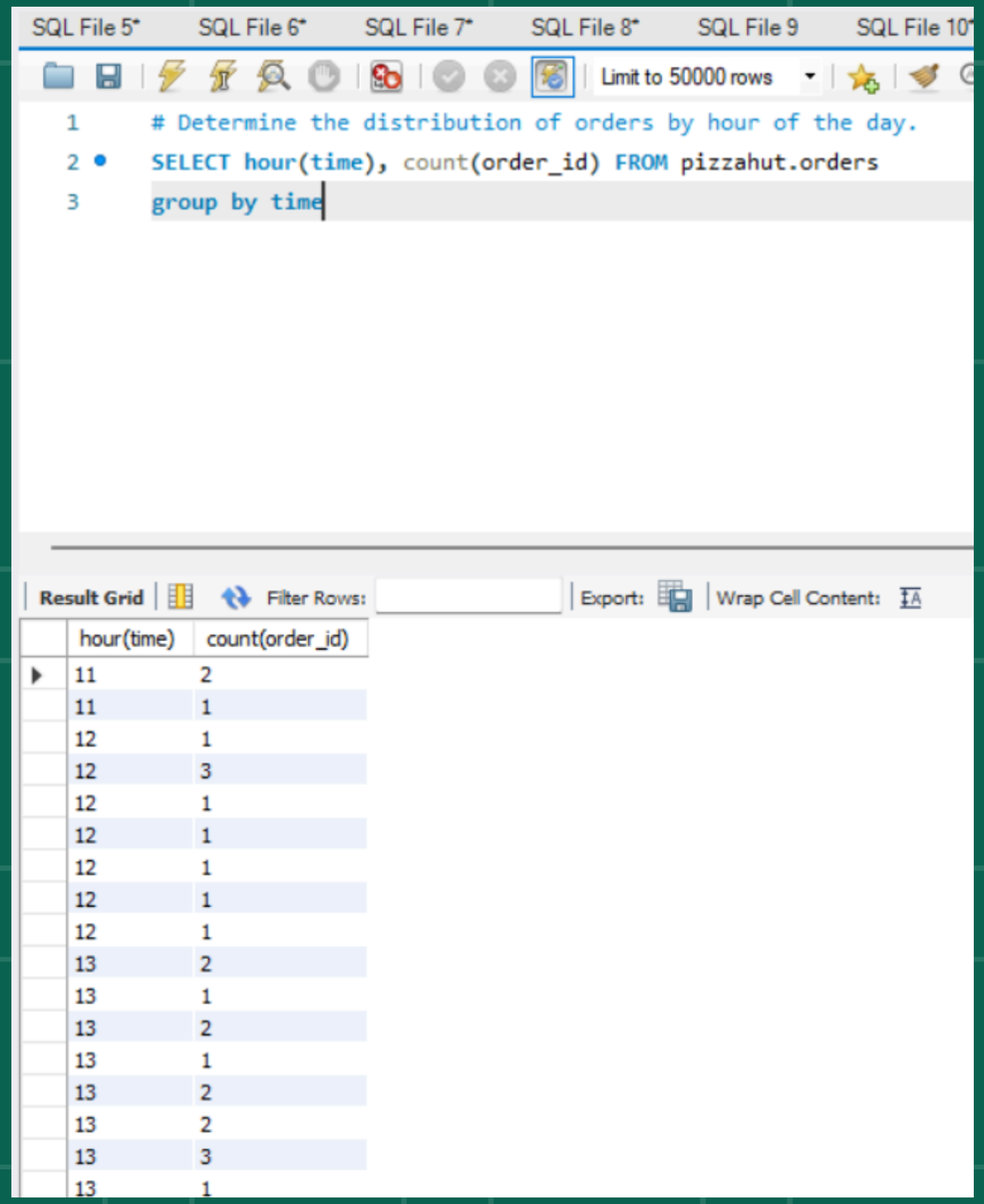
Limit to 50000 rows

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

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

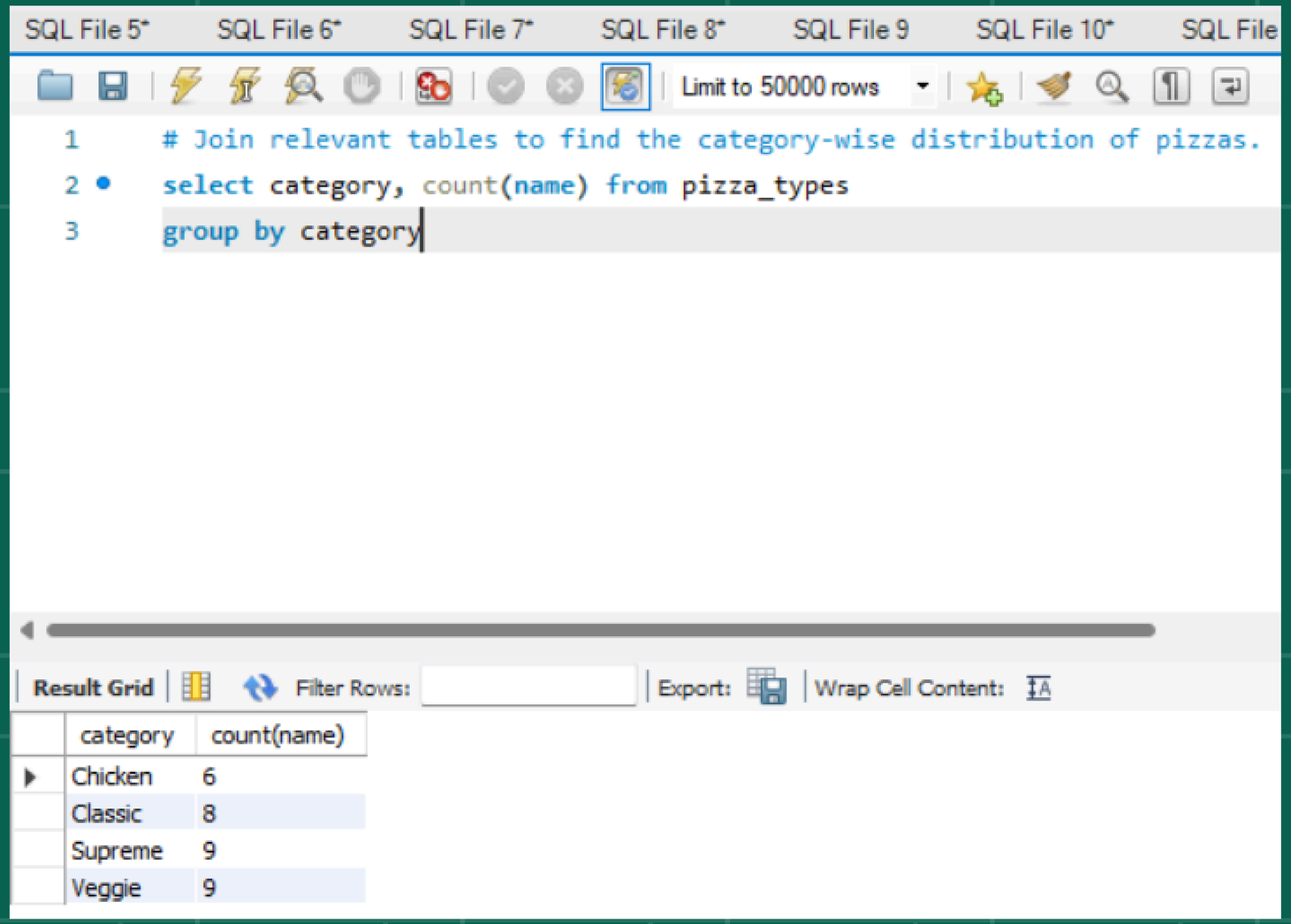
category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Determine the distribution of orders by hour of the day.



hour(time)	count(order_id)
11	2
11	1
12	1
12	3
12	1
12	1
12	1
12	1
12	1
13	2
13	1
13	2
13	1
13	2
13	2
13	3
13	1

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



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

SQL File 5* SQL File 6* SQL File 7* SQL File 8* SQL File 9 SQL File 10* SQL File 11* SQL File 12

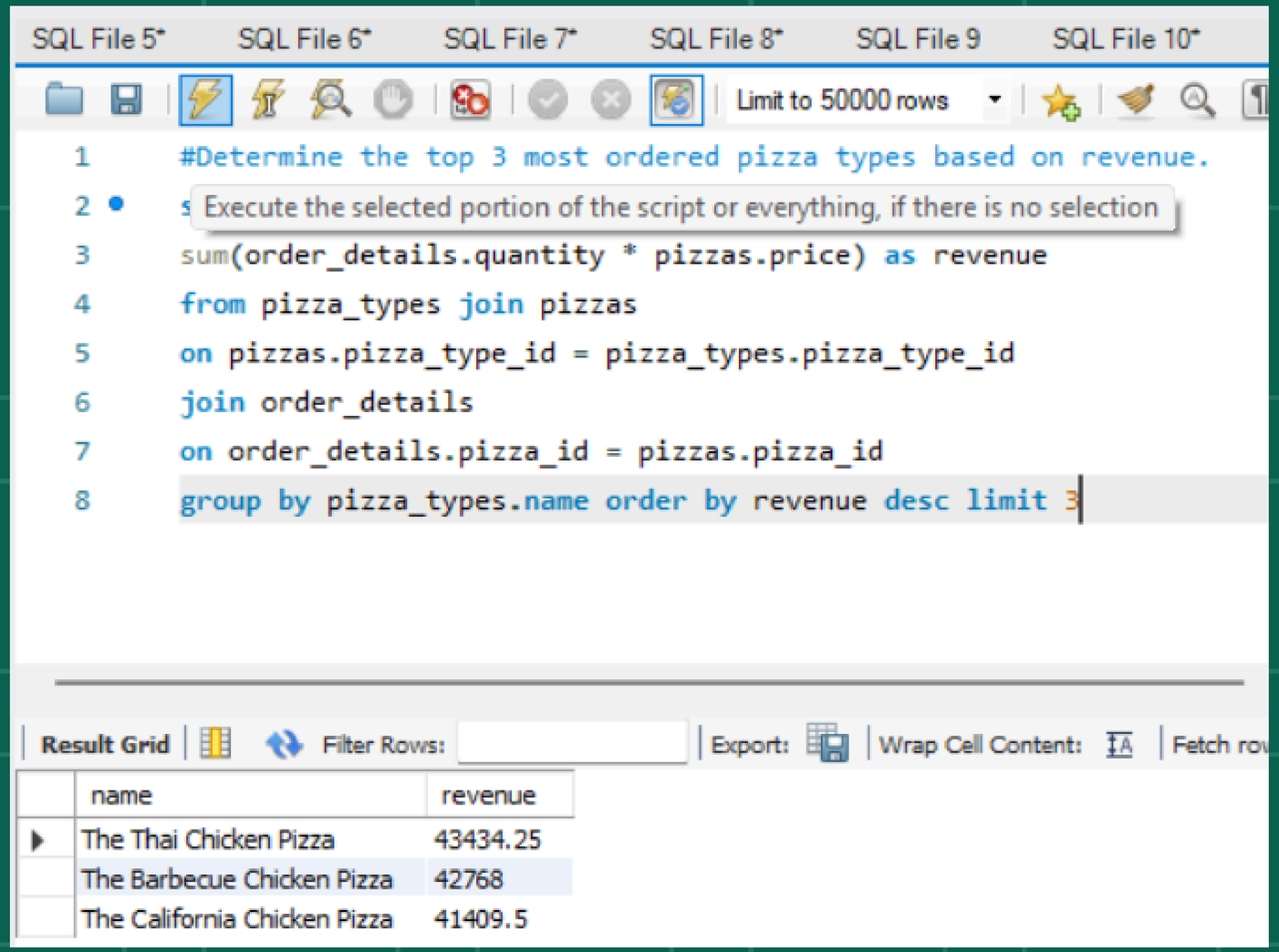
Limit to 50000 rows

```
1   # Group the orders by date and calculate the average number of pizzas ordered per day.
2   select round(avg(quantity),0) from
3   (select orders.date, sum(order_details.quantity) as quantity
4   from orders join order_details
5   on orders.order_id = order_details.order_id
6   group by orders.date) as order_quantity
```

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

round(avg(quantity),0)
138

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



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

SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8* | SQL File 9 | SQL File 10* | SQL File 11* | SQL File 12* | orders | SQL File 13* | SQL File 14

Limit to 50000 rows

```sql
# 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
```

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

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

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

SQL File 7* | SQL File 8* | SQL File 9 | SQL File 10* | SQL File 11* | SQL File 12* | orders | SQL File 13* | SQL File

Limit to 50000 rows

```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
```

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

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

Conclusion & Key Takeaways

- Learned how to retrieve aggregated data using COUNT(), SUM(), and MAX() to answer simple business queries like total orders, revenue, and highest-priced items.
- Understood how to identify most frequent values using GROUP BY and ORDER BY, helping in customer preference analysis.
- Gained experience in using joins across multiple tables to combine relevant data for deeper insights.
- Practiced using date/time functions (EXTRACT, HOUR, etc.) to analyze order patterns across hours and days.
- Learned how to group data effectively to understand category-wise distributions and average daily performance, enabling better sales trend identification.
- Understood the use of percentage calculations and subqueries to analyze revenue contribution per item.
- Practiced cumulative analysis using window functions or running totals to see business growth over time.
- Mastered the logic to apply multi-level grouping and filtering (e.g., top 3 pizzas by category revenue), helping simulate real-world business reporting.

Thank You

sahilt018@gmail.com