Domino's Pizza Sales SQL Analysis

An Interactive Data Exploration Report
Made by Nandini

Table of Contents

Q1. Retrieve the total number of orders placed.

Q2. Calculate the total revenue generated from pizza sales.

Q3. Identify the highest-priced pizza.

Q4. Identify the most common pizza size ordered.

Q5. List the top 5 most ordered pizza types along with their quantities.

Q6. Total quantity ordered per pizza type.

Q7. Determine the distribution of orders by hour of the day.

Q8. Category-wise distribution of pizzas.

Q9. Average number of pizzas ordered per day.

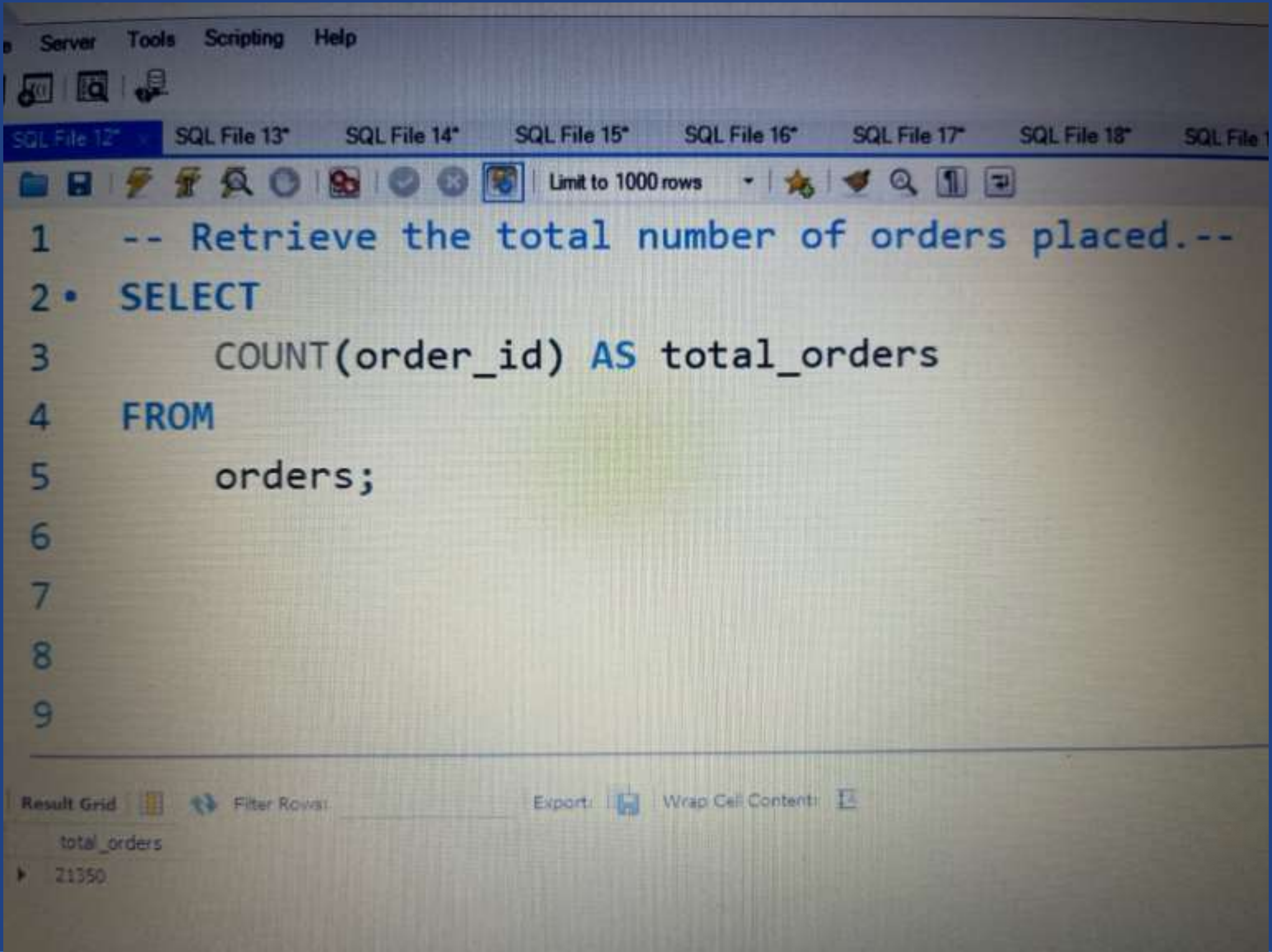
Q10. Top 3 pizza types by revenue.

Q11. Calculate the percentage contribution of each pizza type to total revenue.

Q12. Analyze the cumulative revenue generated over time

Q13. Determine the top 3 most ordered pizza types based on revenue for each pizza category.

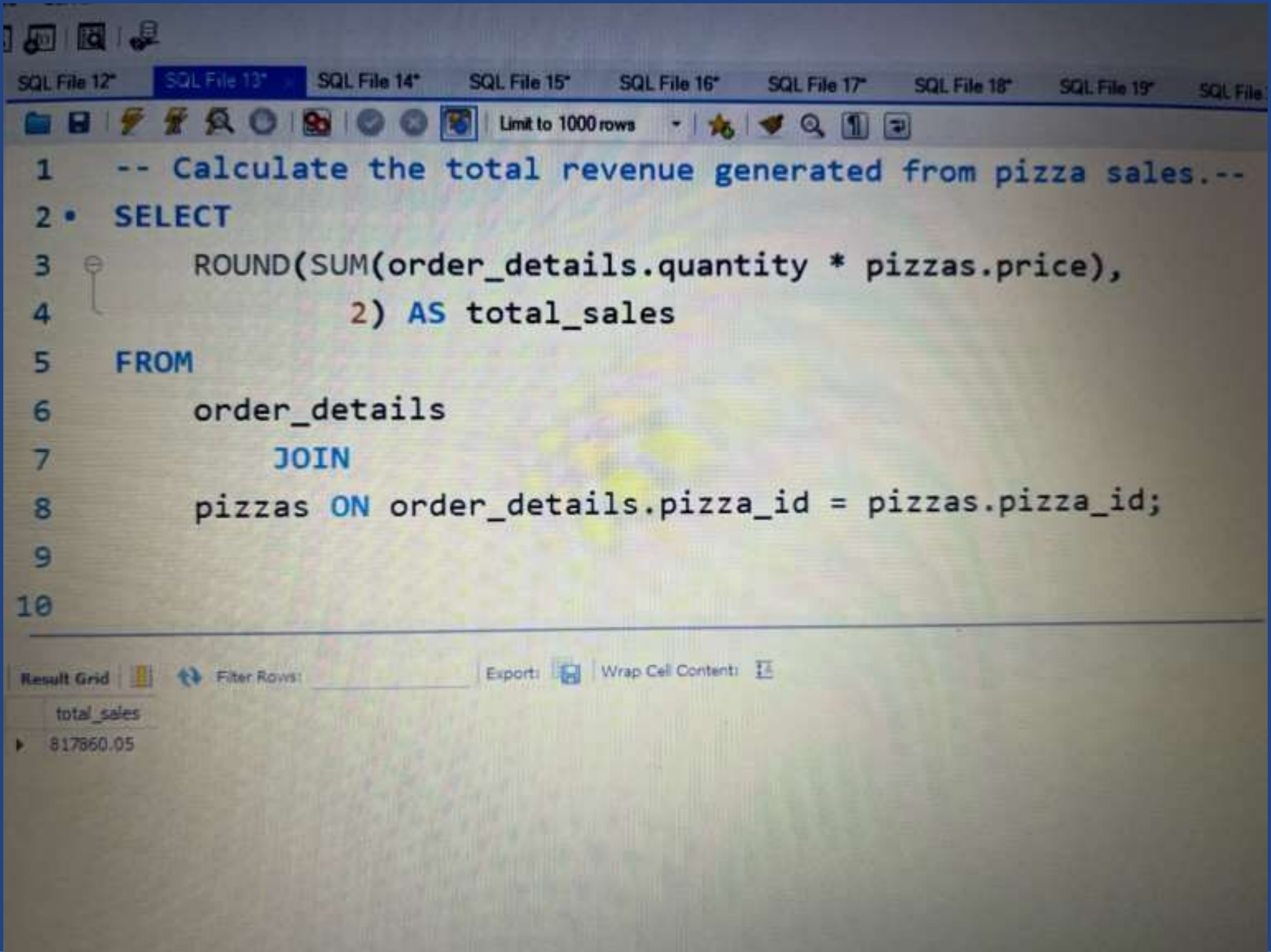
Q1. Retrieve the total number of orders placed.



```sql
-- Retrieve the total number of orders placed.--
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

```
total_orders
21350
```

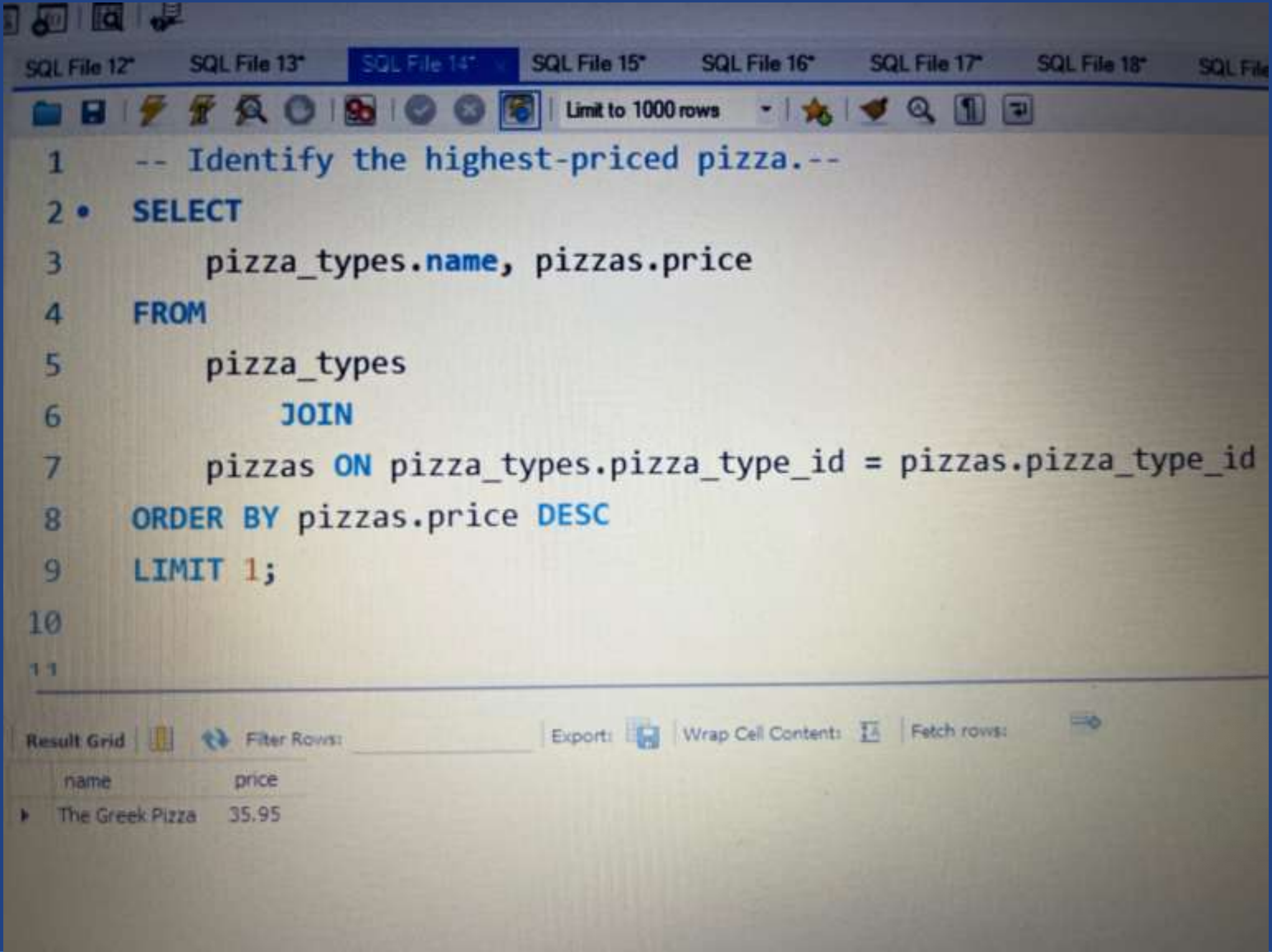
Q2. Calculate the total revenue generated from pizza sales.



```
-- Calculate the total revenue generated from pizza sales.--
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS total_sales
FROM
    order_details
        JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id;
```

```
total_sales
817860.05
```

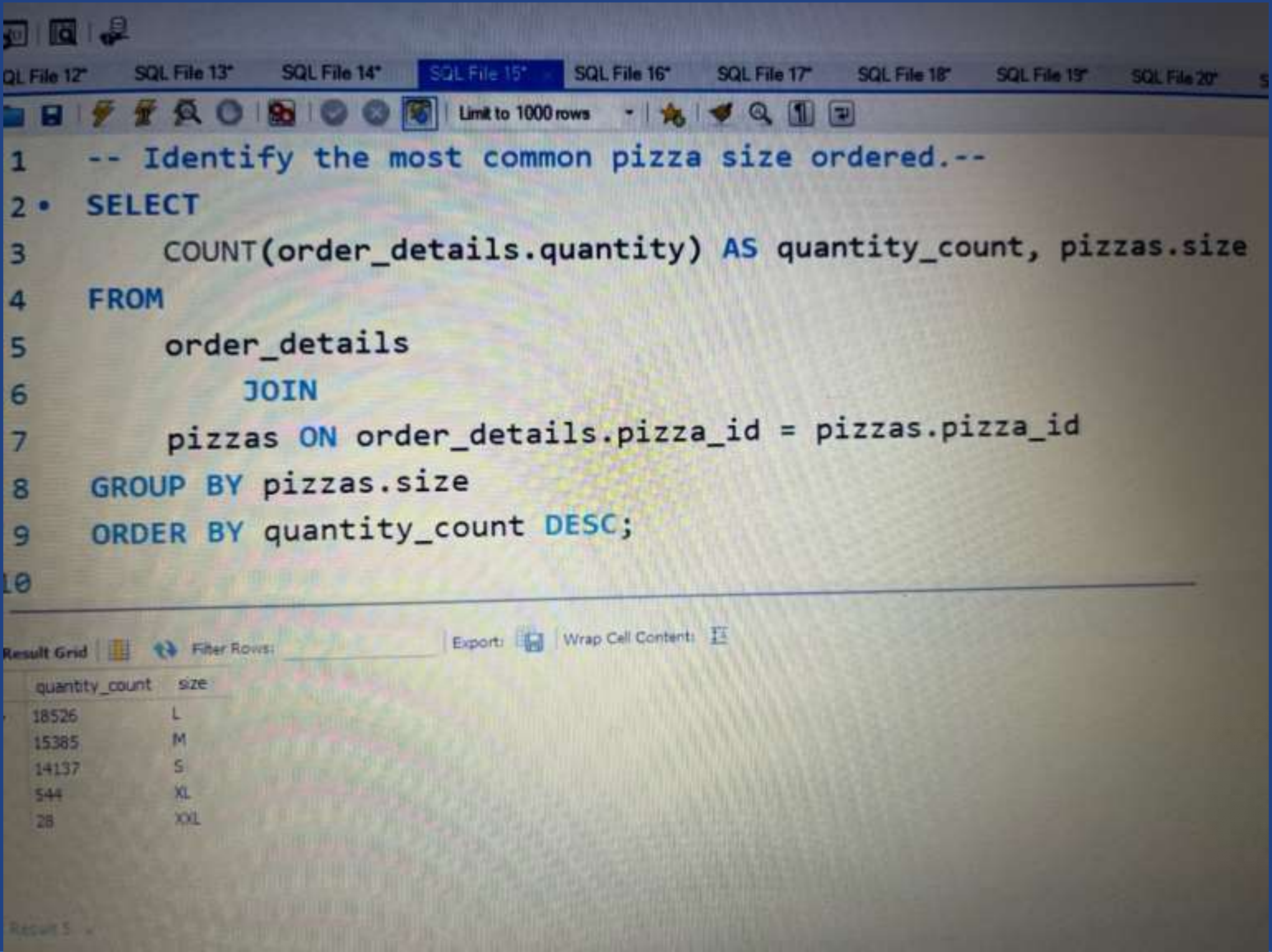
Q3. Identify the highest-priced pizza.



```sql
-- 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;
```

```
name             price
The Greek Pizza  35.95
```

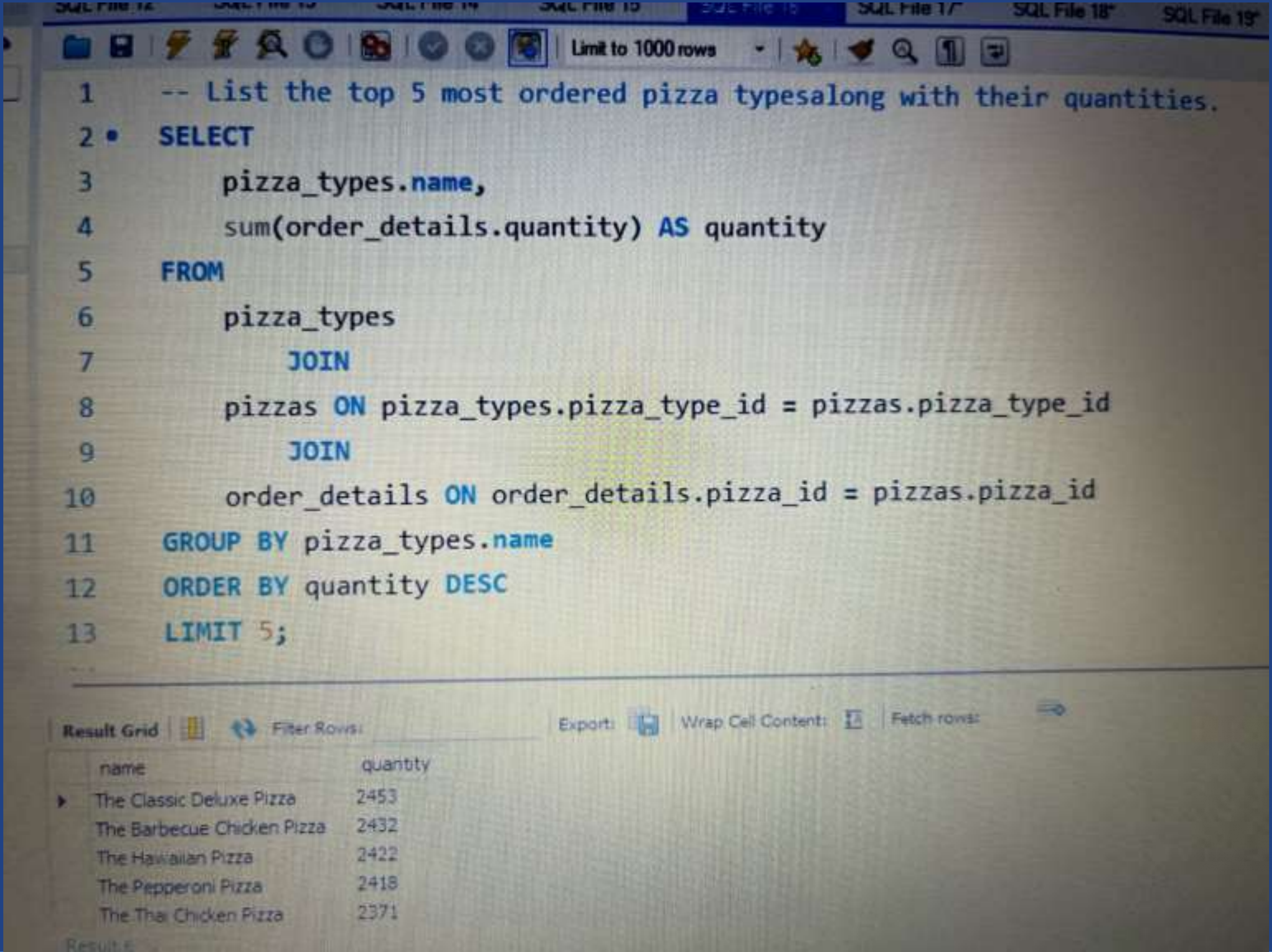
Q4. Identify the most common pizza size ordered.



```sql
-- Identify the most common pizza size ordered.--
SELECT
    COUNT(order_details.quantity) AS quantity_count, pizzas.size
FROM
    order_details
        JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizzas.size
ORDER BY quantity_count DESC;
```

```
quantity_count   size
18526            L
15385            M
14137            S
544              XL
28               XXL
```

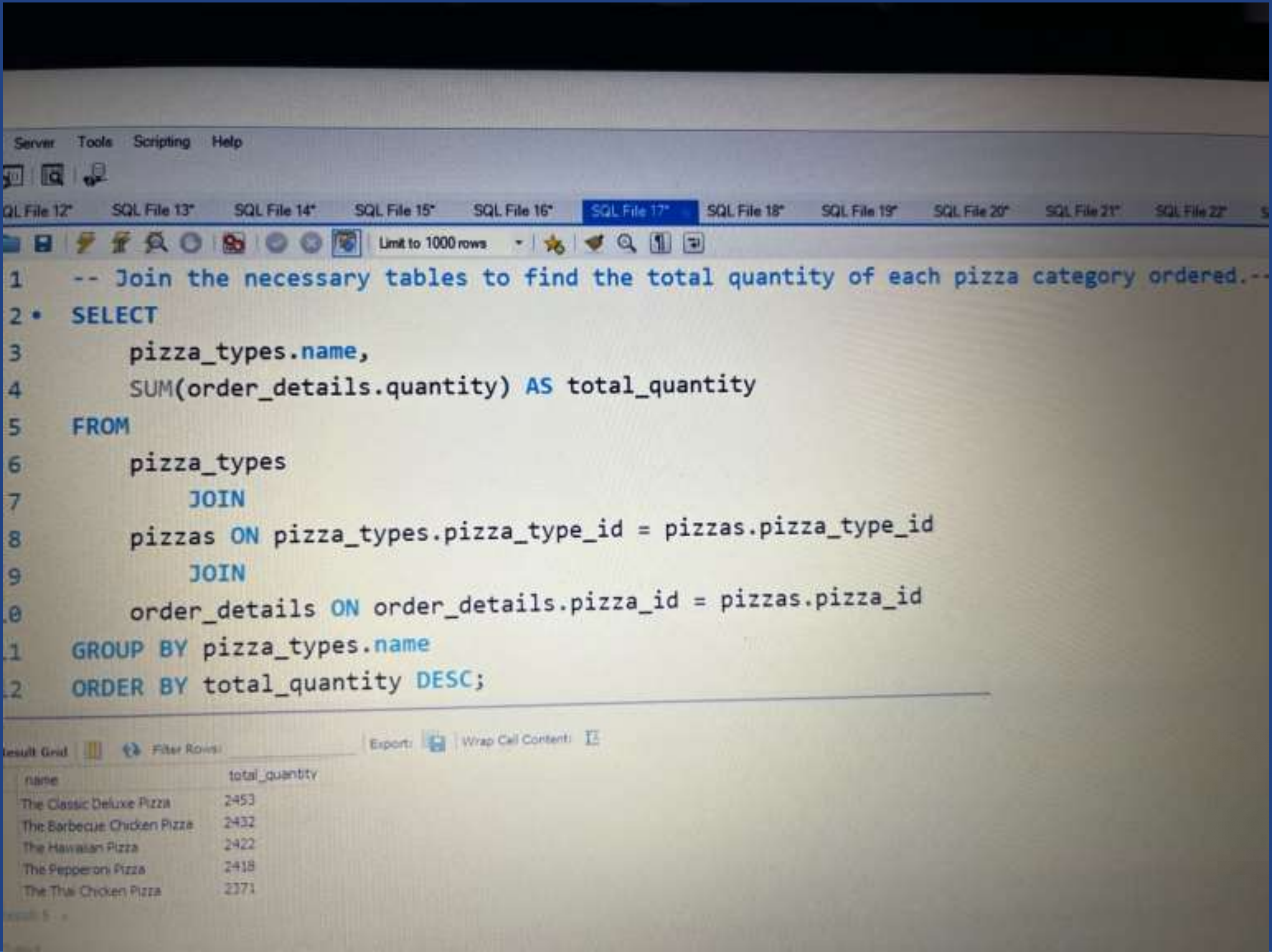
Q5. List the top 5 most ordered pizza types along with their quantities.



```sql
-- List the top 5 most ordered pizza typesalong 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;
```

```
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
```

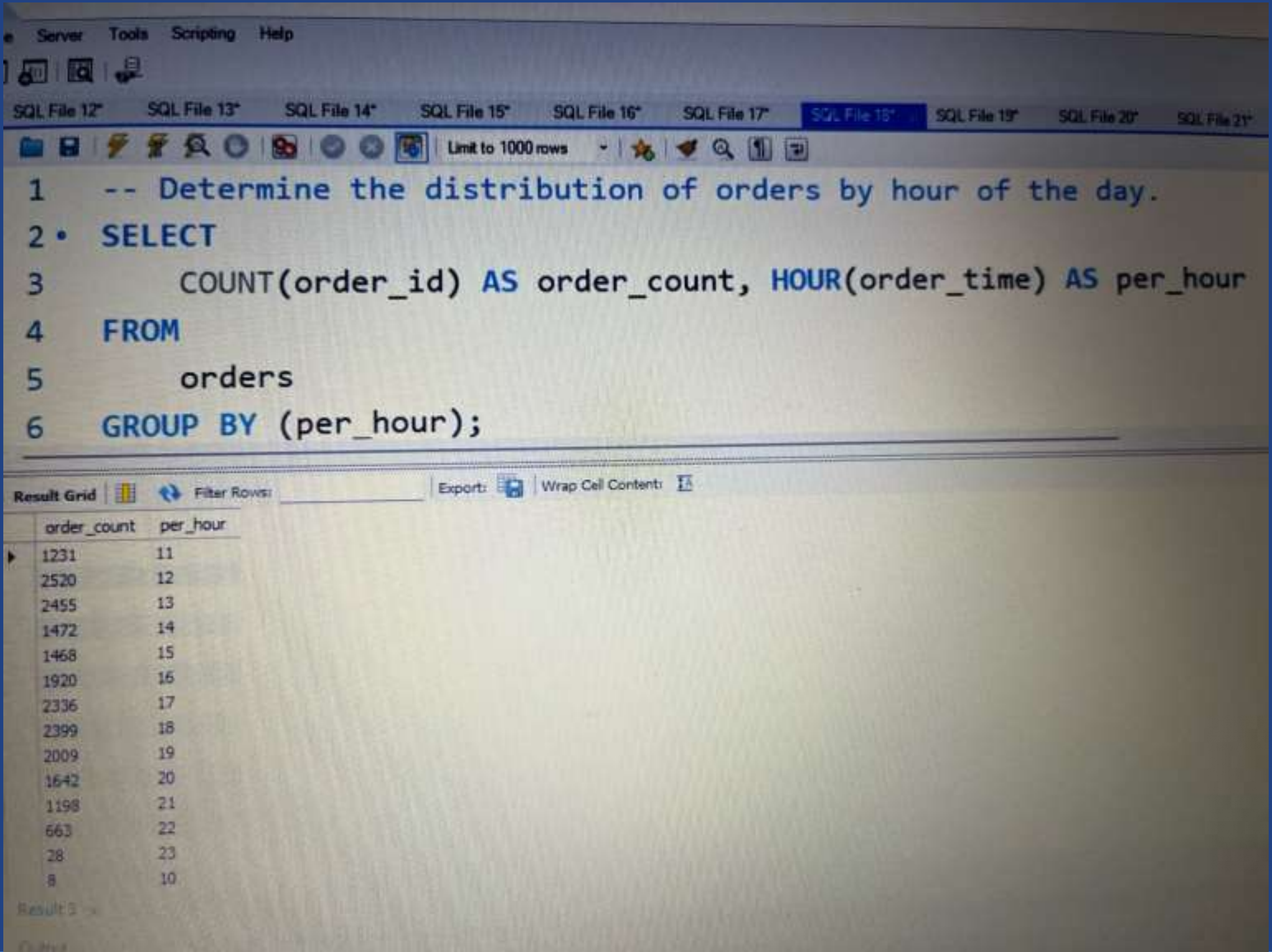
Q6. Total quantity ordered per pizza type.



```sql
-- Join the necessary tables to find the total quantity of each pizza category ordered.--
SELECT
    pizza_types.name,
    SUM(order_details.quantity) AS total_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 total_quantity DESC;
```

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

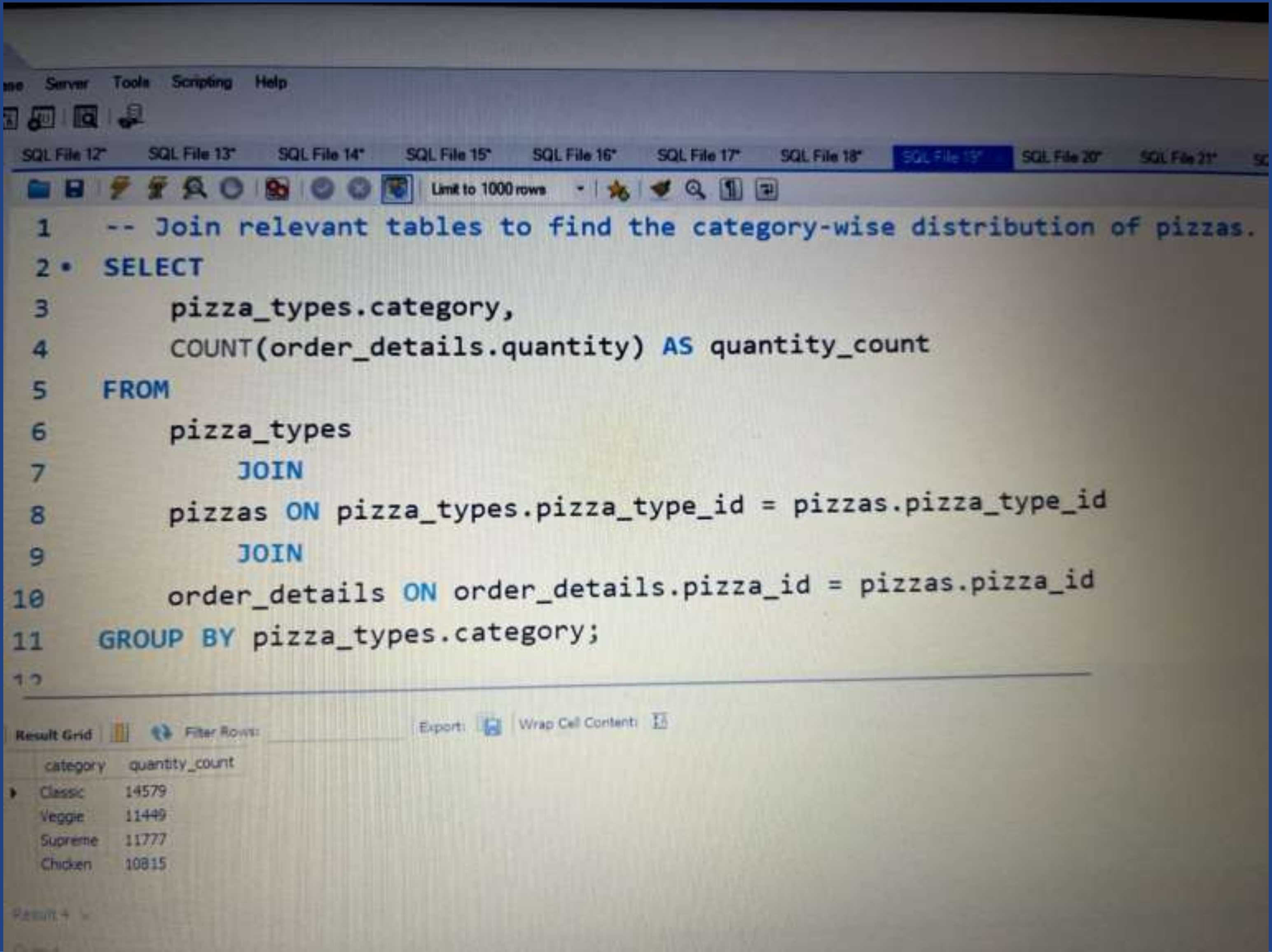
Q7. Determine the distribution of orders by hour of the day.



```sql
-- Determine the distribution of orders by hour of the day.
SELECT
      COUNT(order_id) AS order_count, HOUR(order_time) AS per_hour
FROM
      orders
GROUP BY (per_hour);
```

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

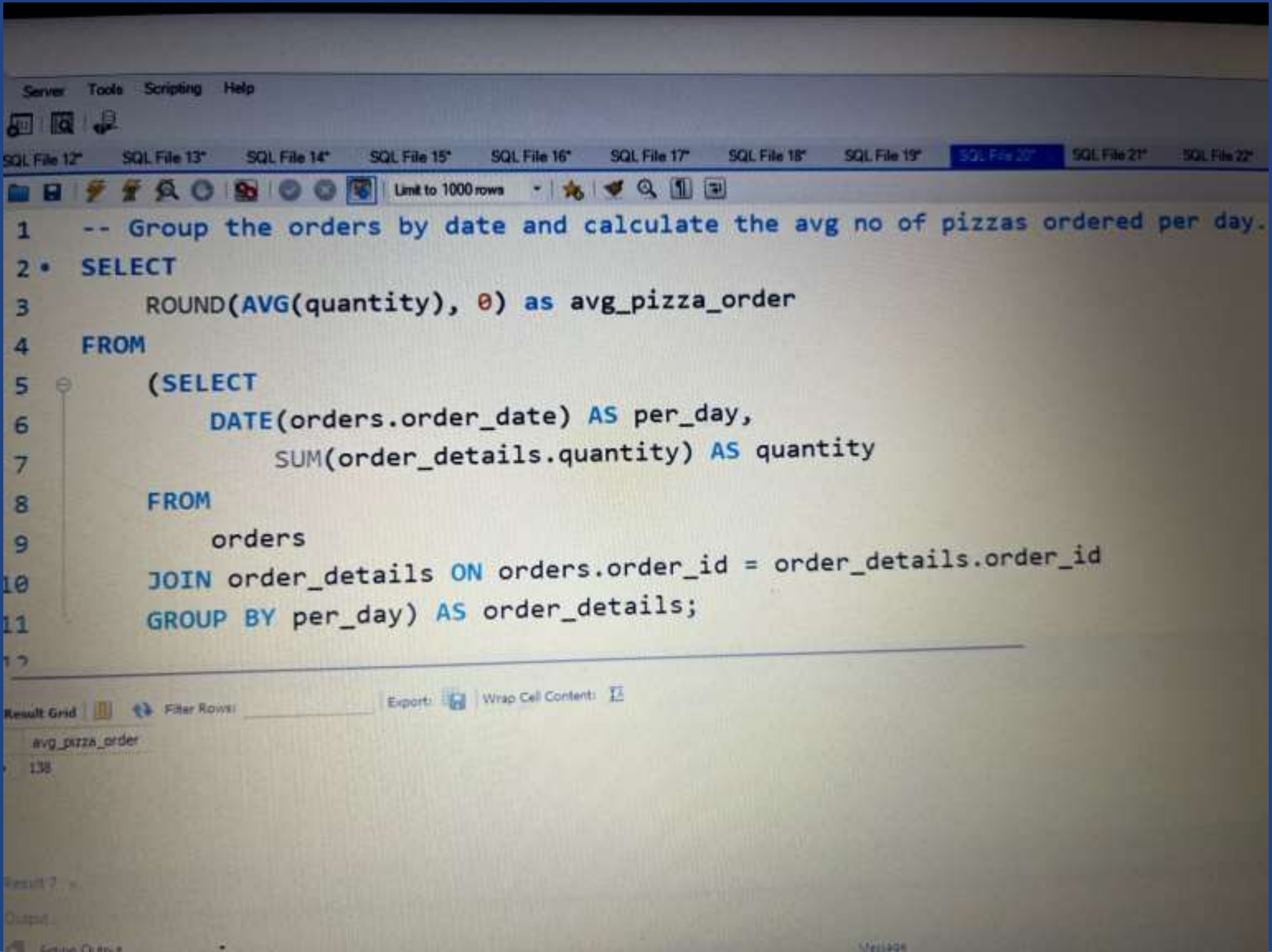
Q8. Category-wise distribution of pizzas.



```sql
-- Join relevant tables to find the category-wise distribution of pizzas.
SELECT
    pizza_types.category,
    COUNT(order_details.quantity) AS quantity_count
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;
```

category	quantity_count
Classic	14579
Veggie	11449
Supreme	11777
Chicken	10815

Q9. Average number of pizzas ordered per day.

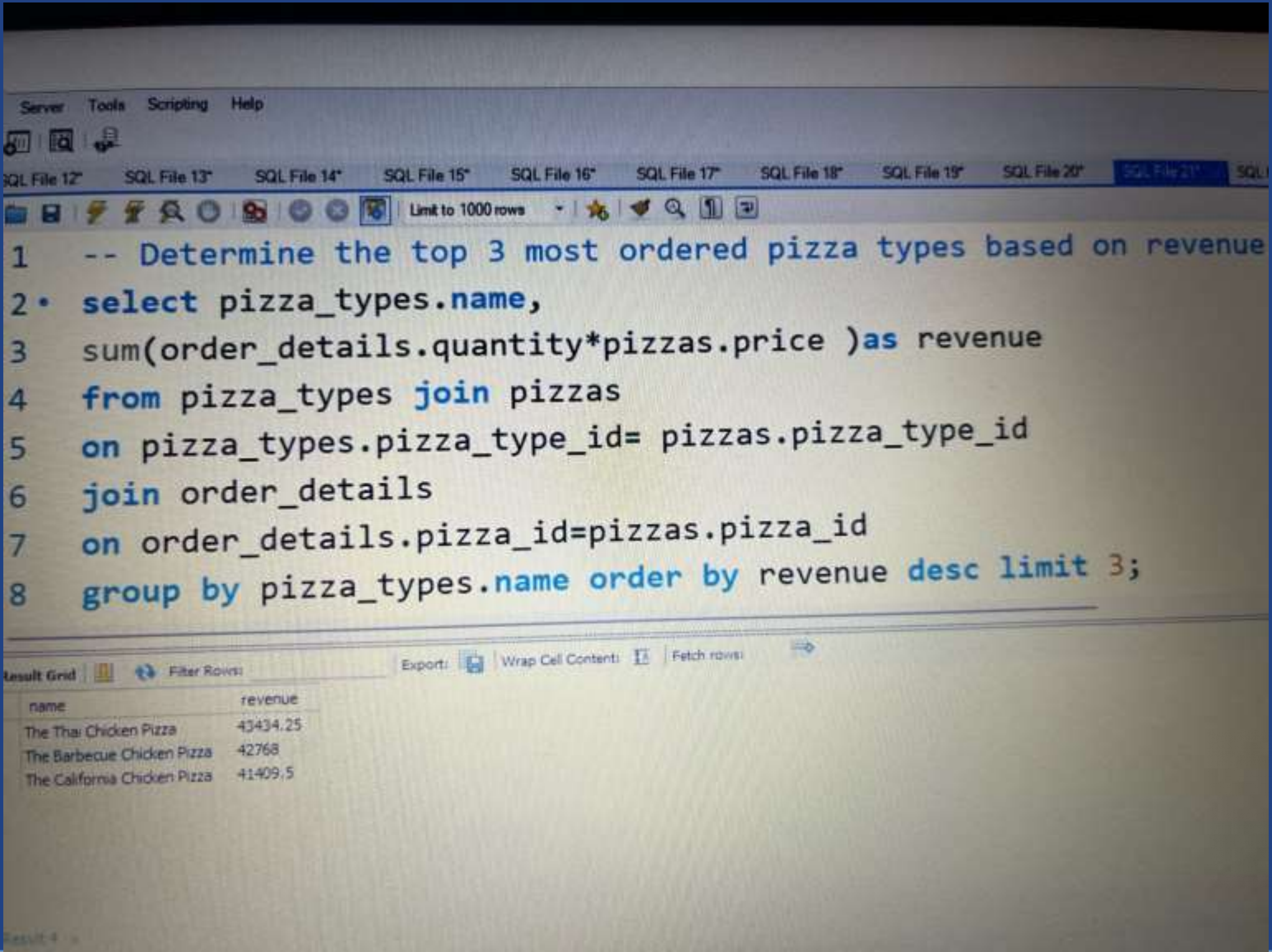


Server Tools Scripting Help
SQL File 12* SQL File 13* SQL File 14* SQL File 15* SQL File 16* SQL File 17* SQL File 18* SQL File 19* SQL File 20* SQL File 21* SQL File 22*
Limit to 1000 rows
Result Grid Filter Rows: Export: Wrap Cell Content:

```
-- Group the orders by date and calculate the avg no of pizzas ordered per day.
SELECT
    ROUND(AVG(quantity), 0) as avg_pizza_order
FROM
    (SELECT
        DATE(orders.order_date) AS per_day,
            SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY per_day) AS order_details;
```

```
avg_pizza_order
138
```

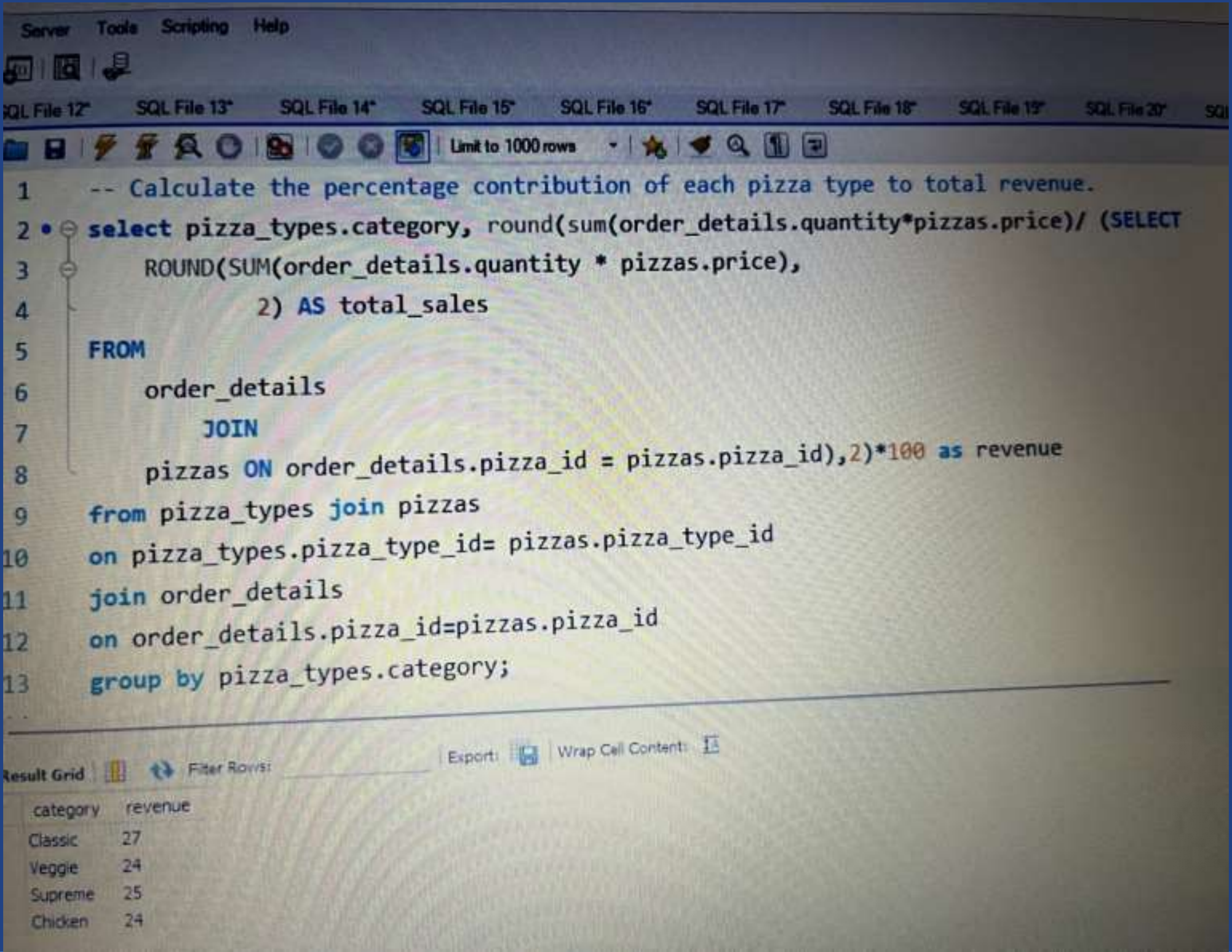
Q10. Top 3 pizza types by revenue.


Server Tools Scripting Help
SQL File 12* SQL File 13* SQL File 14* SQL File 15* SQL File 16* SQL File 17* SQL File 18* SQL File 19* SQL File 20* SQL File 21*
Limit to 1000 rows
Result Grid Filter Rows: Export: Wrap Cell Content: Fetch rows:

```sql
-- 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 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 revenue desc limit 3;
```

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

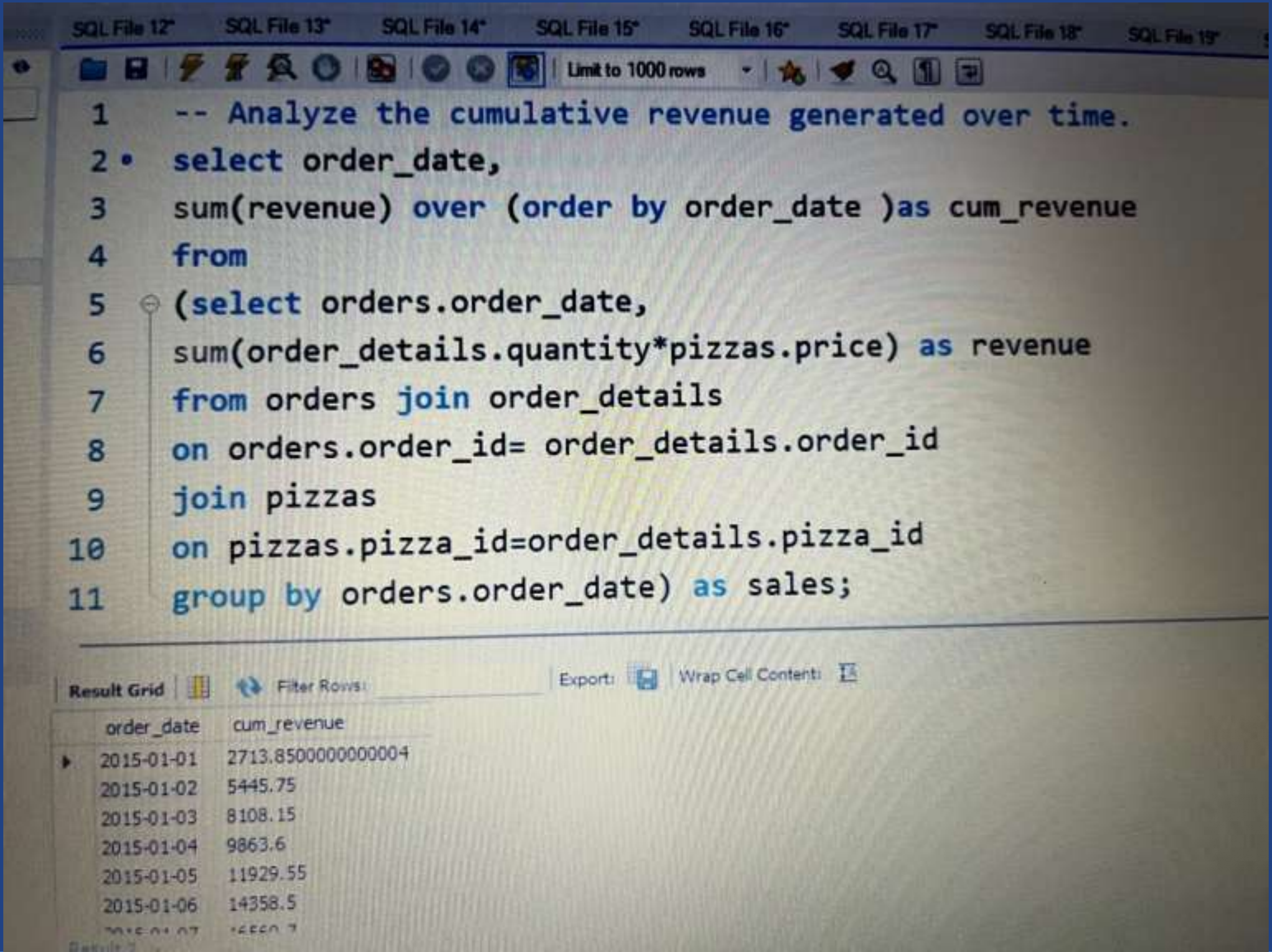
Q11. Calculate the percentage contribution of each pizza type to total revenue.


Server Tools Scripting Help
SQL File 12* SQL File 13* SQL File 14* SQL File 15* SQL File 16* SQL File 17* SQL File 18* SQL File 19* SQL File 20*
Limit to 1000 rows

```
-- 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 order_details.pizza_id = pizzas.pizza_id),2)*100 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;
```

```
category   revenue
Classic    27
Veggie     24
Supreme    25
Chicken    24
```

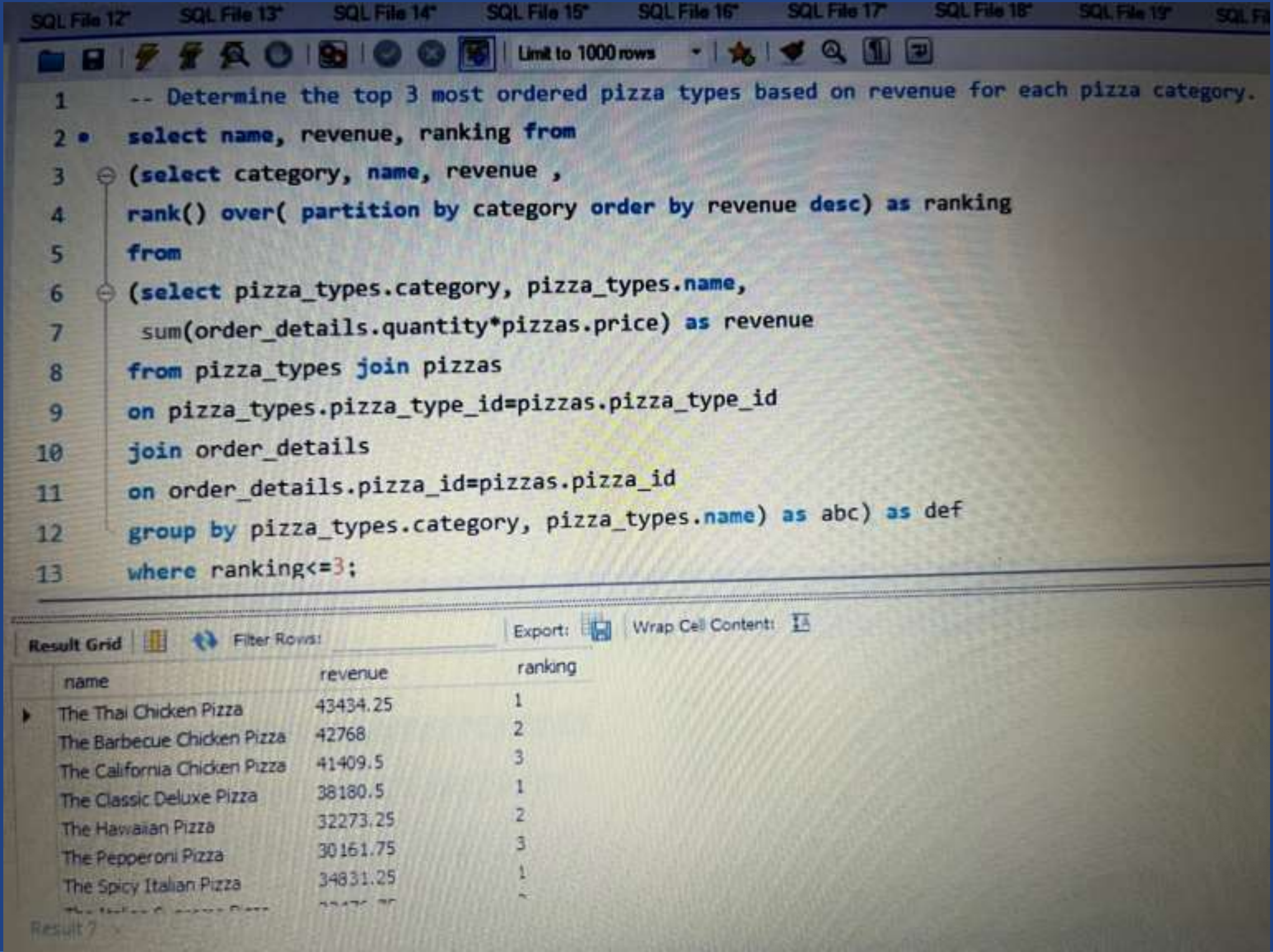
Q12. Analyze the cumulative revenue generated over time.



```
-- Analyze the cumulative revenue generated over time.
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 orders join order_details
on orders.order_id= order_details.order_id
join pizzas
on pizzas.pizza_id=order_details.pizza_id
group by orders.order_date) as sales;
```

```
order_date   cum_revenue
2015-01-01   2713.8500000000004
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
```

Q13. Determine the top 3 most ordered pizza types based on revenue for each pizza category.



```sql
-- Determine the top 3 most ordered pizza types based on revenue for each pizza category.
select name, revenue, ranking from
(select category, name, revenue ,
rank() over( partition by category order by revenue desc) as ranking
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 abc) as def
where ranking<=3;
```

name	revenue	ranking
The Thai Chicken Pizza	43434.25	1
The Barbecue Chicken Pizza	42768	2
The California Chicken Pizza	41409.5	3
The Classic Deluxe Pizza	38180.5	1
The Hawaiian Pizza	32273.25	2
The Pepperoni Pizza	30161.75	3
The Spicy Italian Pizza	34831.25	1

Thank You!

This Domino's Pizza SQL Analysis was crafted with data insights.