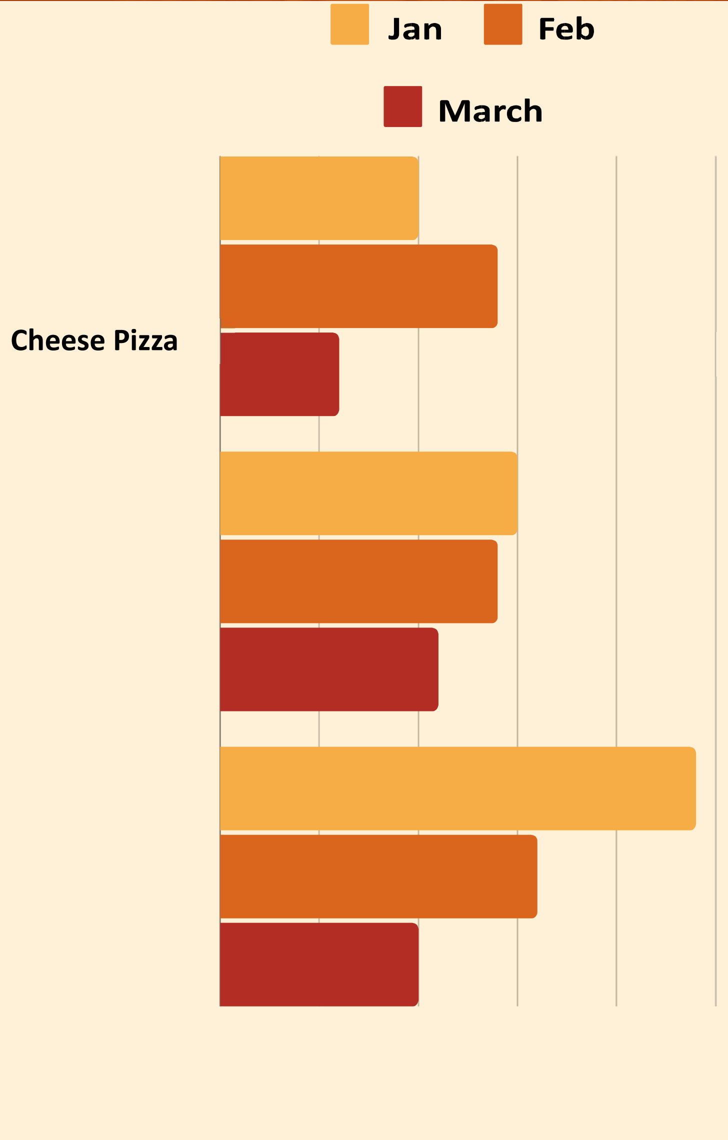
PIZZA SALES

SQL ANALYSIS

Solutions to Common Business Queries



PRESENTED BY

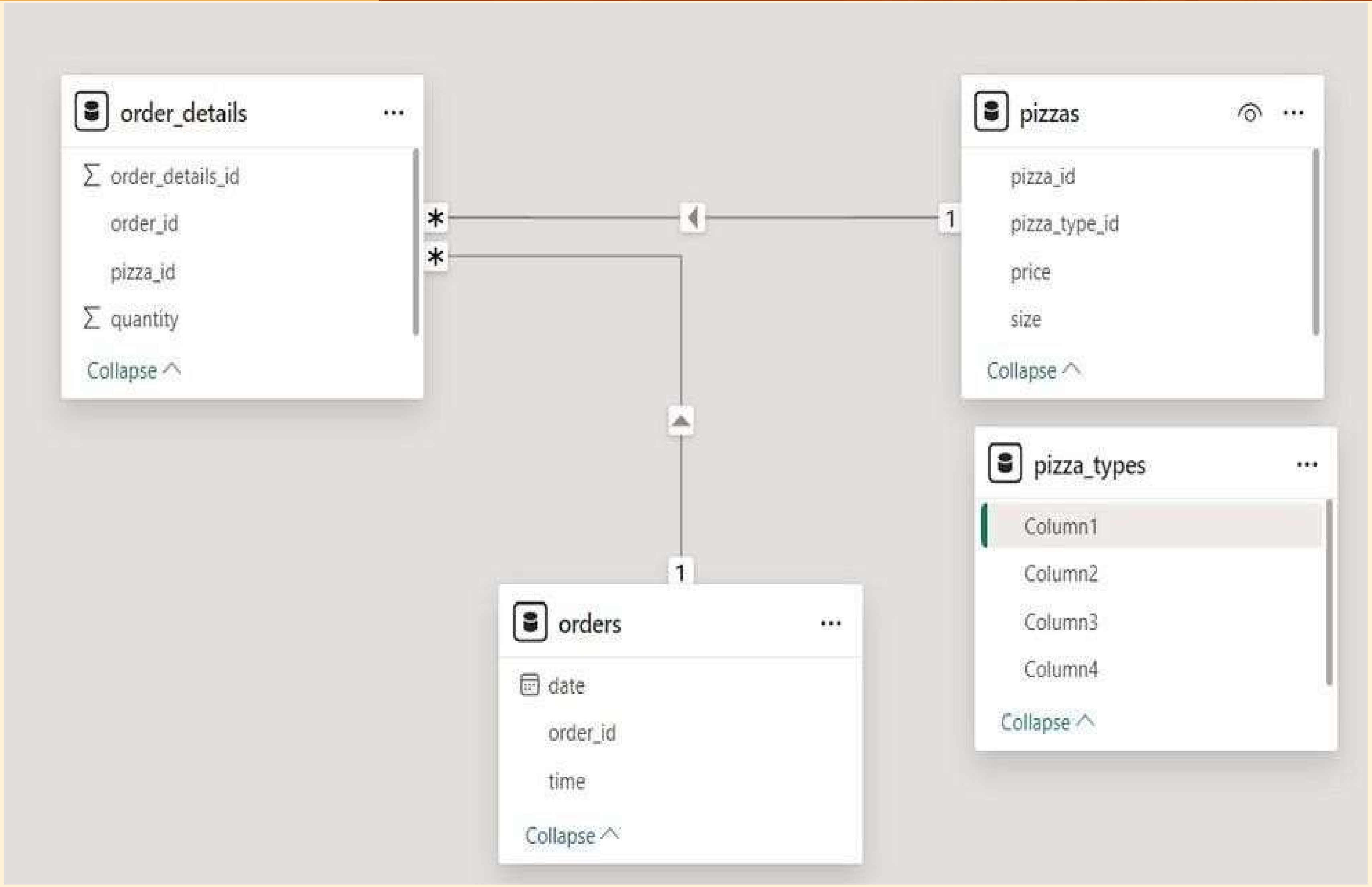
SNEHA CHOUDHRY

20 25

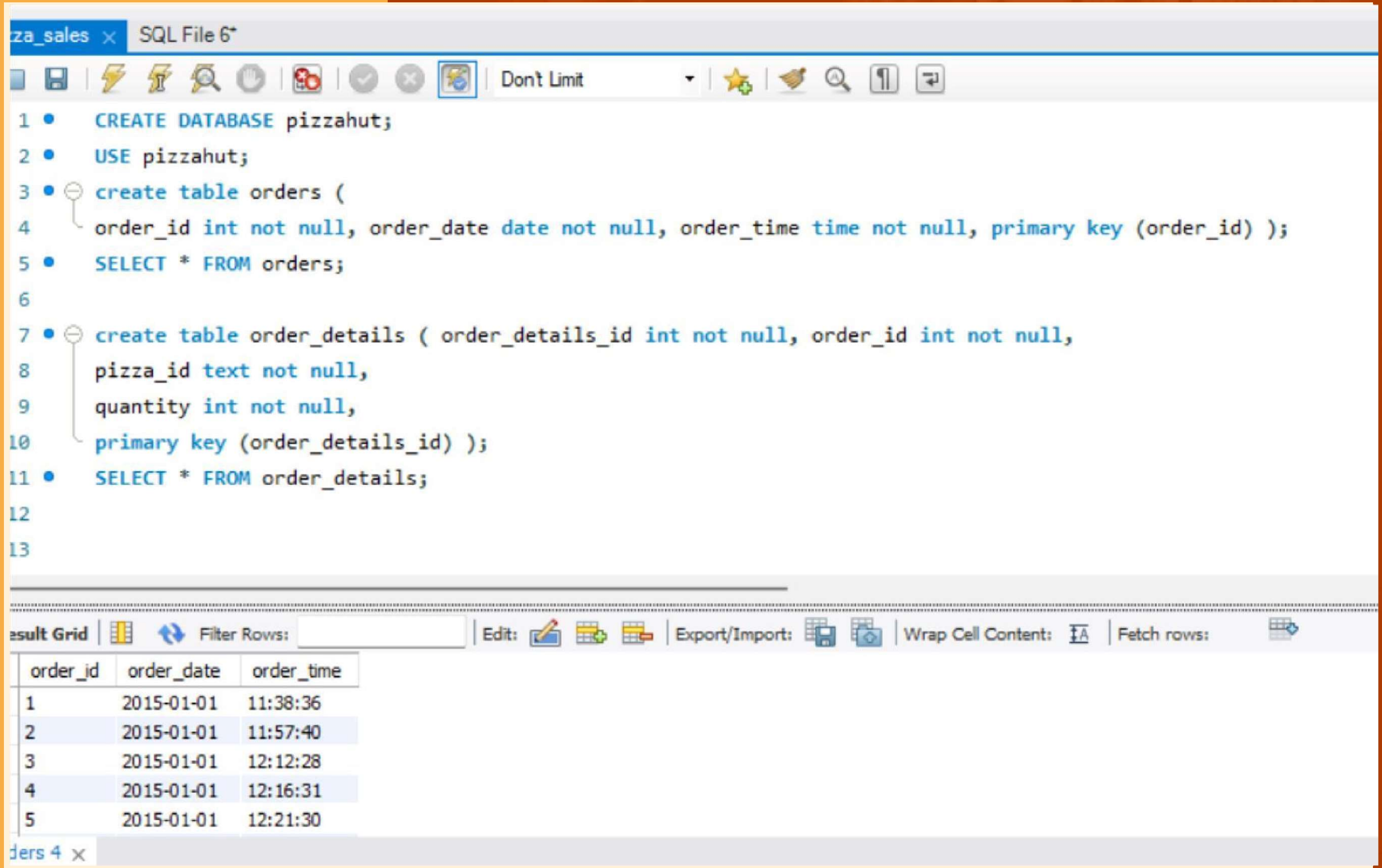
INTRODUCTION

- Analysis of pizza sales data using SQL.

- Addressing various business-related questions to derive meaningful insights.


order_details
order_details_id
order_id
pizza_id
quantity
Collapse
pizzas
pizza_id
pizza_type_id
price
size
Collapse
pizza_types
Column1
Column2
Column3
Column4
Collapse
orders
date
order_id
time
Collapse
*
*
1
1

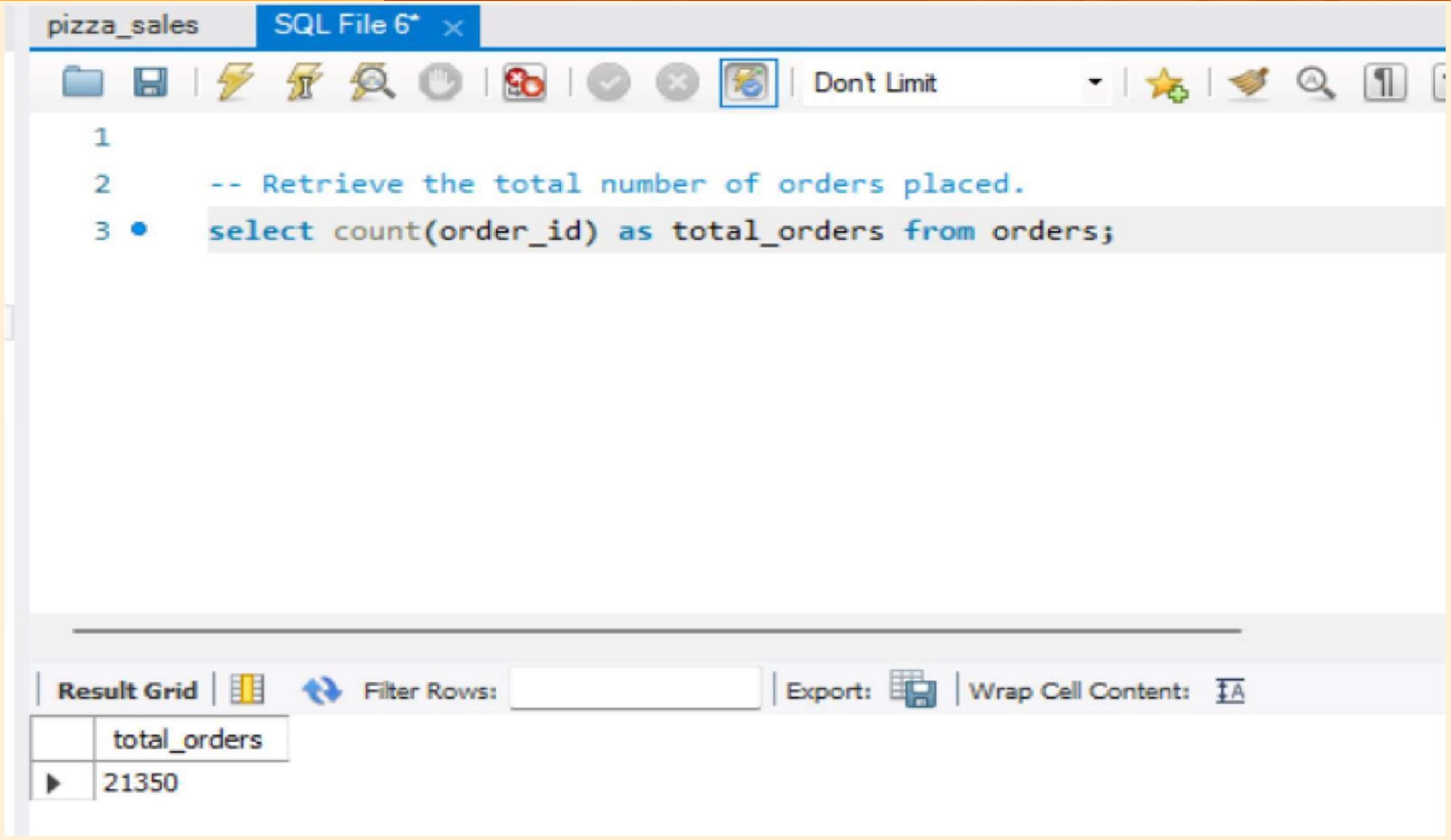
SCHEMA



```
CREATE DATABASE pizzahut;
USE pizzahut;
create table orders (
order_id int not null, order_date date not null, order_time time not null, primary key (order_id) );
SELECT * FROM orders;

create table order_details ( order_details_id int not null, order_id int not null,
pizza_id text not null,
quantity int not null,
primary key (order_details_id) );
SELECT * FROM order_details;
```

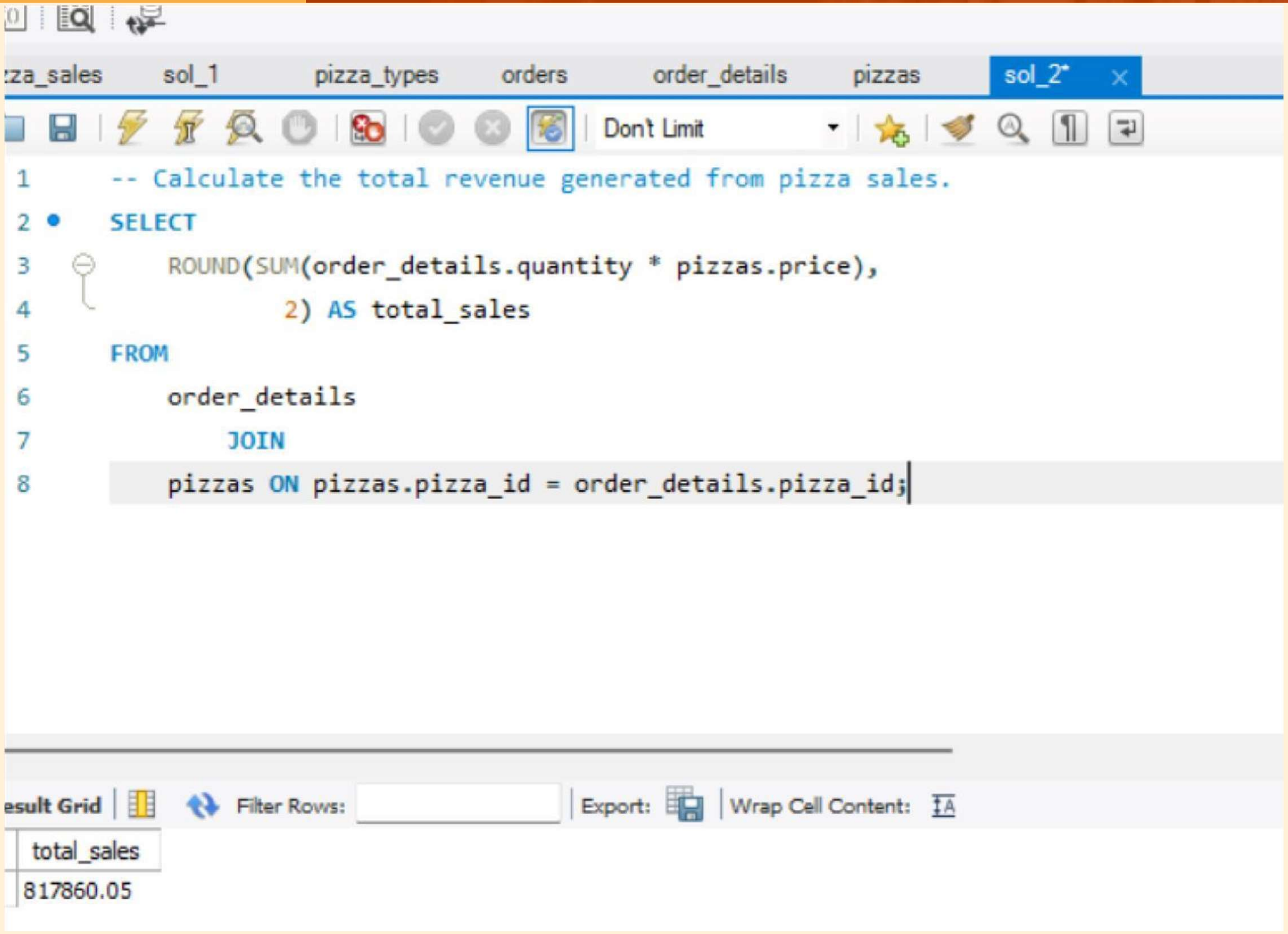
order_id	order_date	order_time
1	2015-01-01	11:38:36
2	2015-01-01	11:57:40
3	2015-01-01	12:12:28
4	2015-01-01	12:16:31
5	2015-01-01	12:21:30



```sql
-- Retrieve the total number of orders placed.
select count(order_id) as total_orders from orders;
```

```
total_orders
21350
```

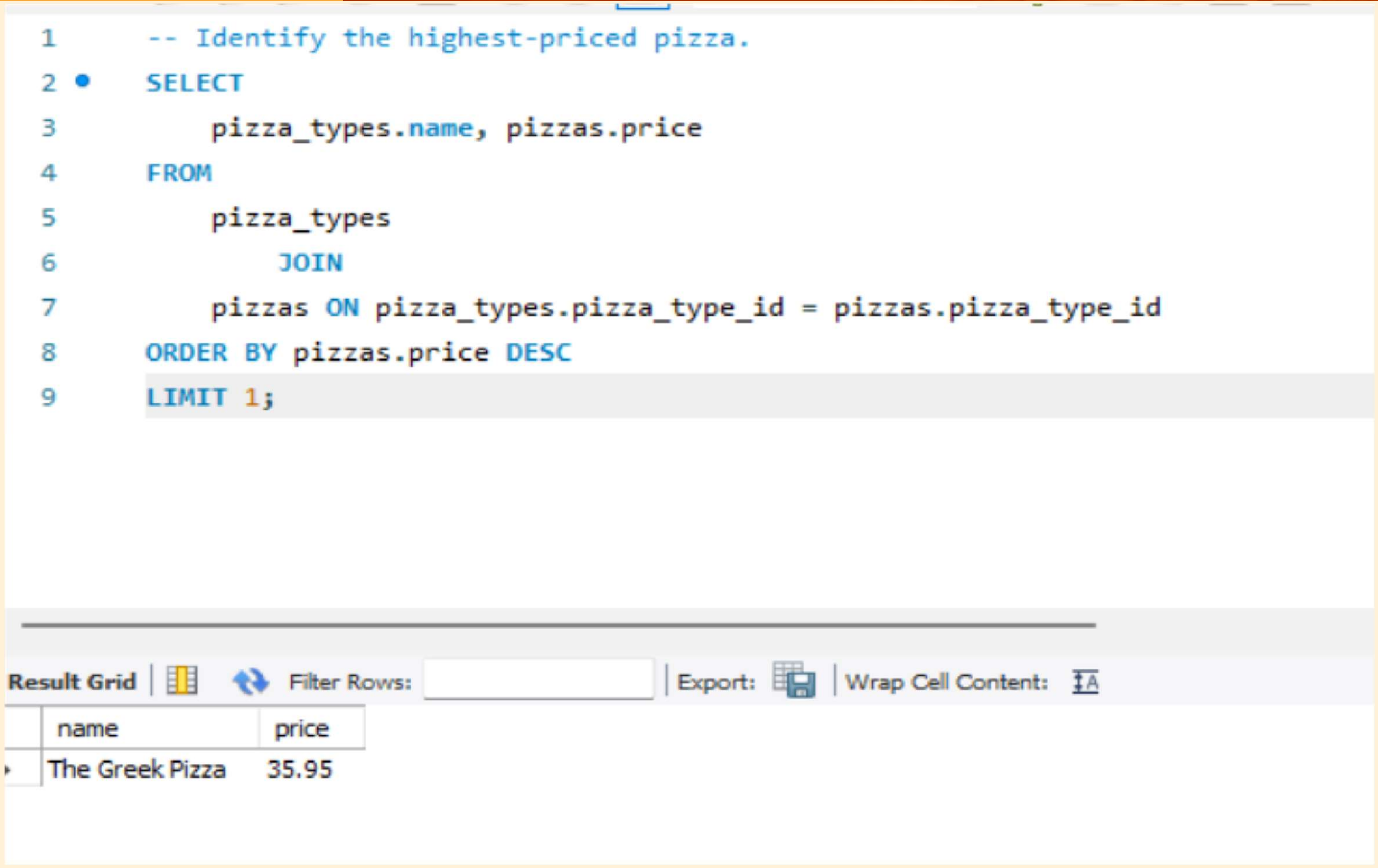
QUERY 1



```sql
-- 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 pizzas.pizza_id = order_details.pizza_id;
```

```
total_sales
817860.05
```

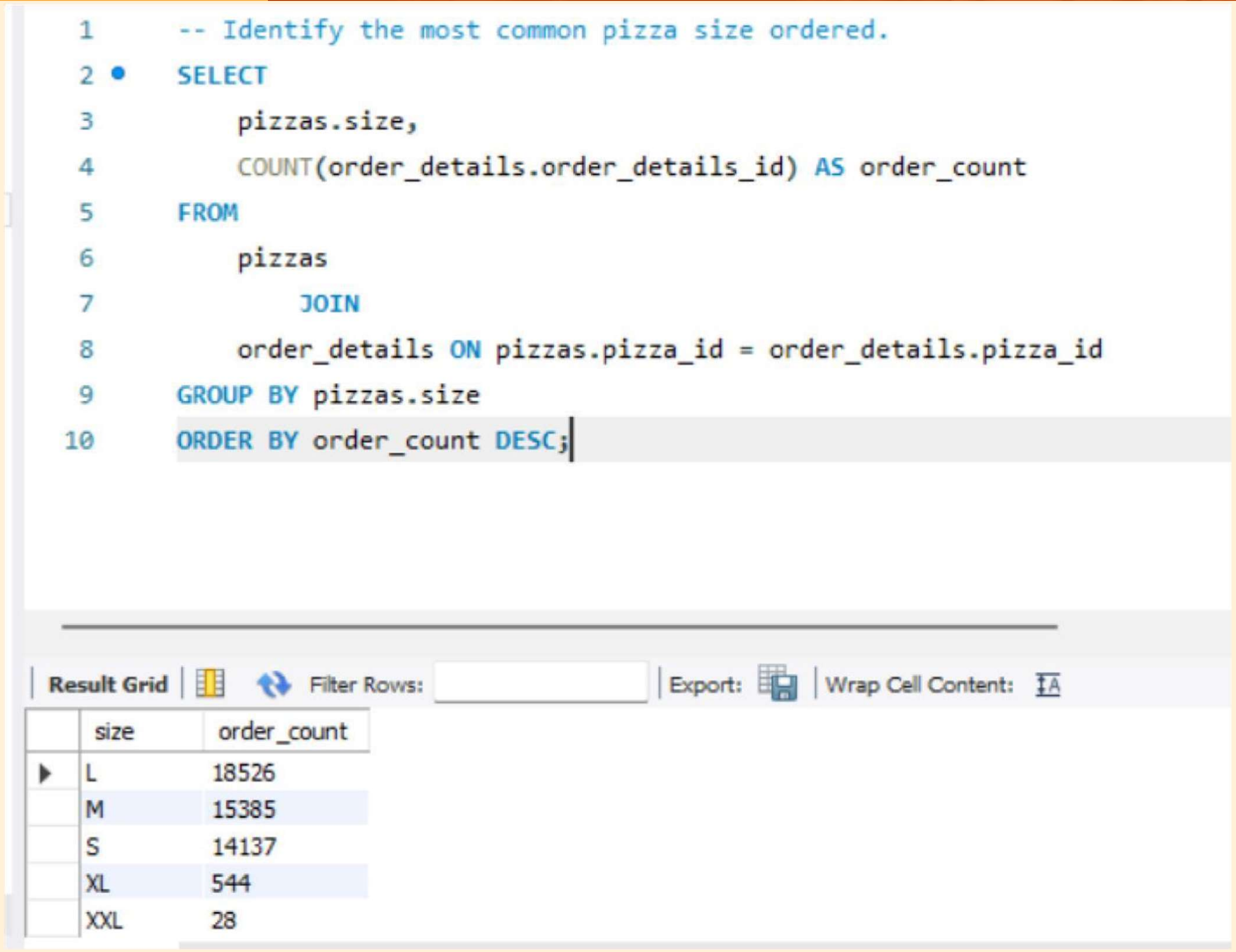
QUERY 2:



```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
```

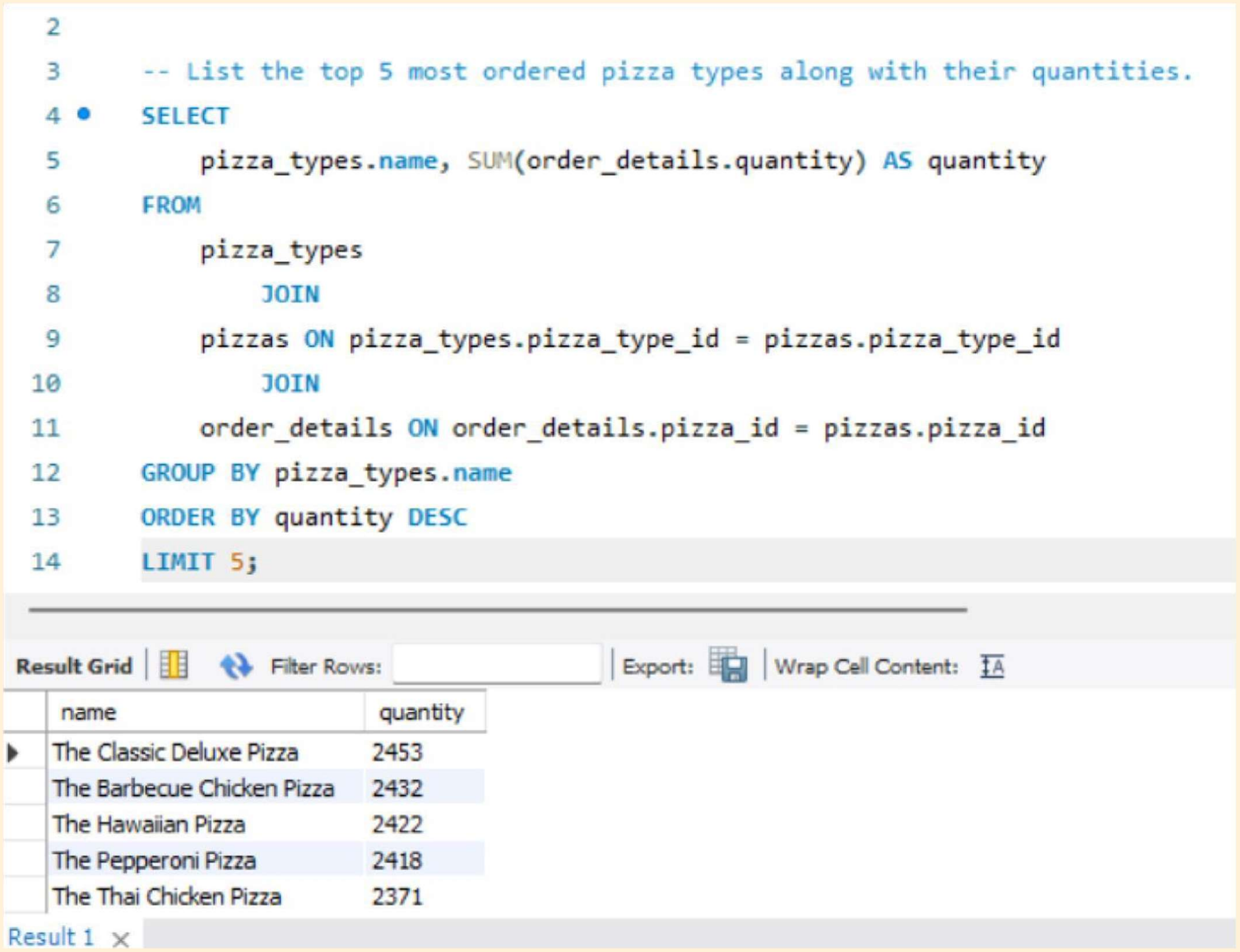
QUERY 3:



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

size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

QUERY 4:

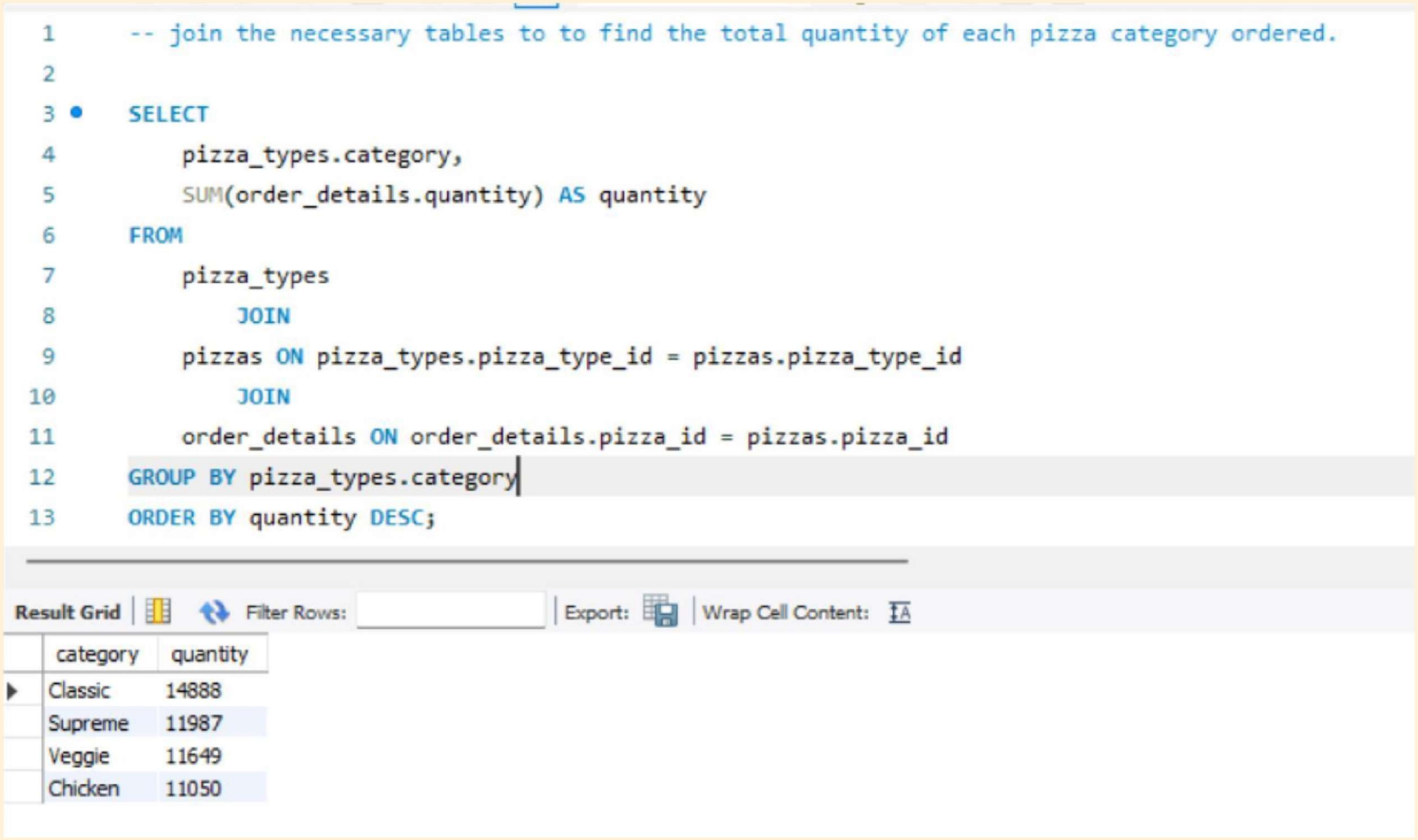


```sql

-- List the top 5 most ordered pizza types along 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

QUERY 5:

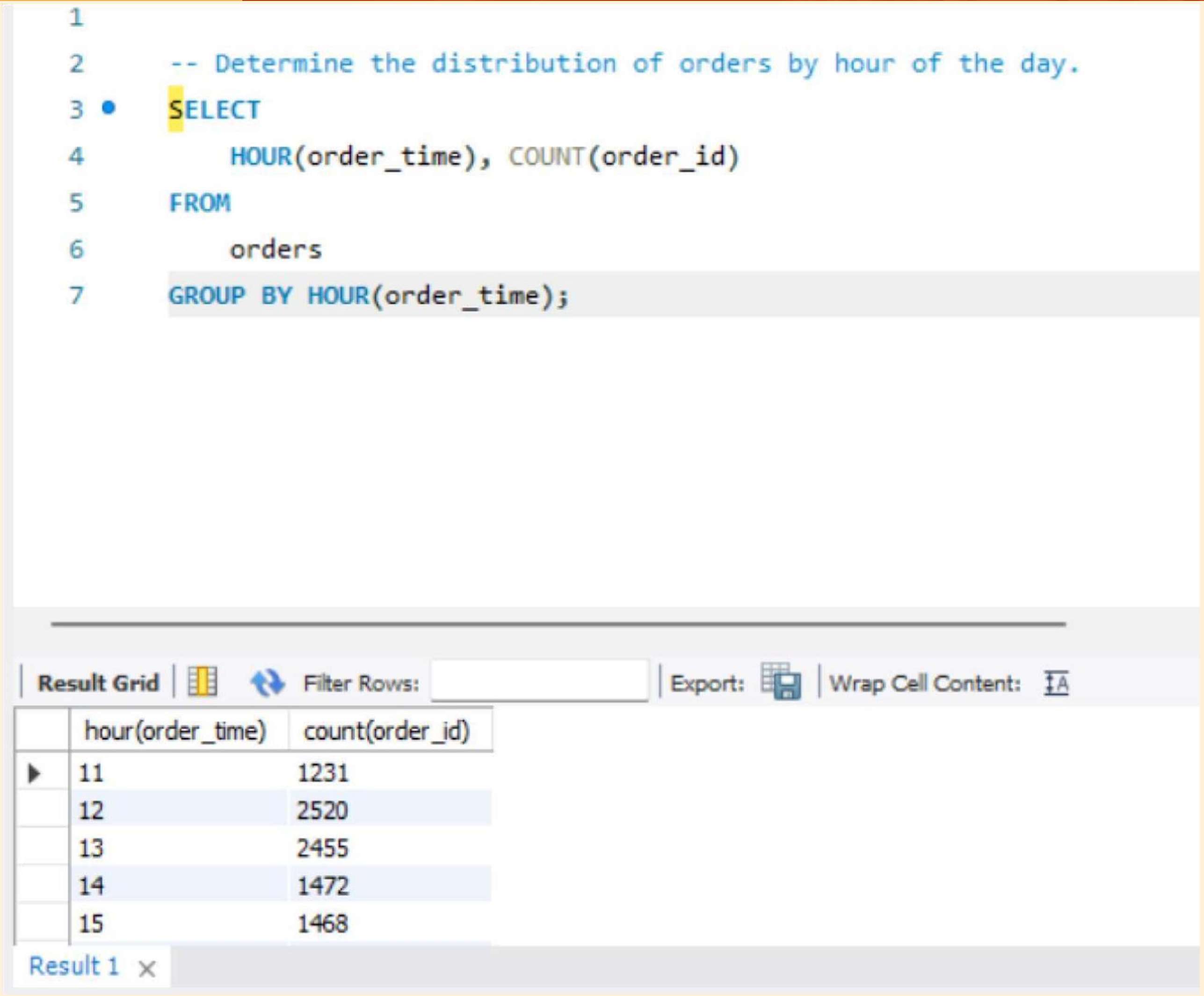


```sql
-- join the necessary tables to to find the total quantity of each pizza category ordered.

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;
```

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

QUERY 6

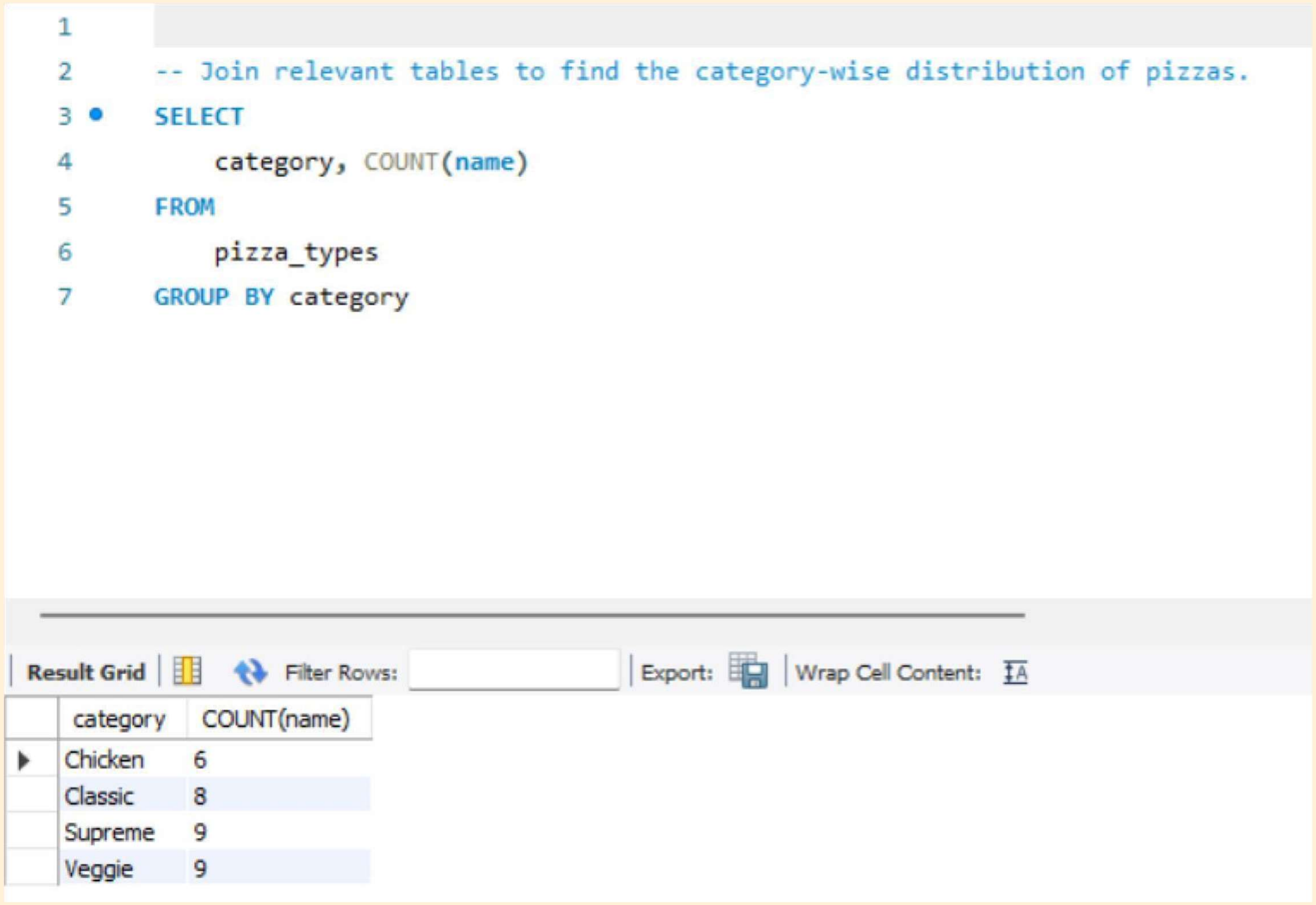


```sql

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

hour(order_time)	count(order_id)
11	1231
12	2520
13	2455
14	1472
15	1468

QUERY 7

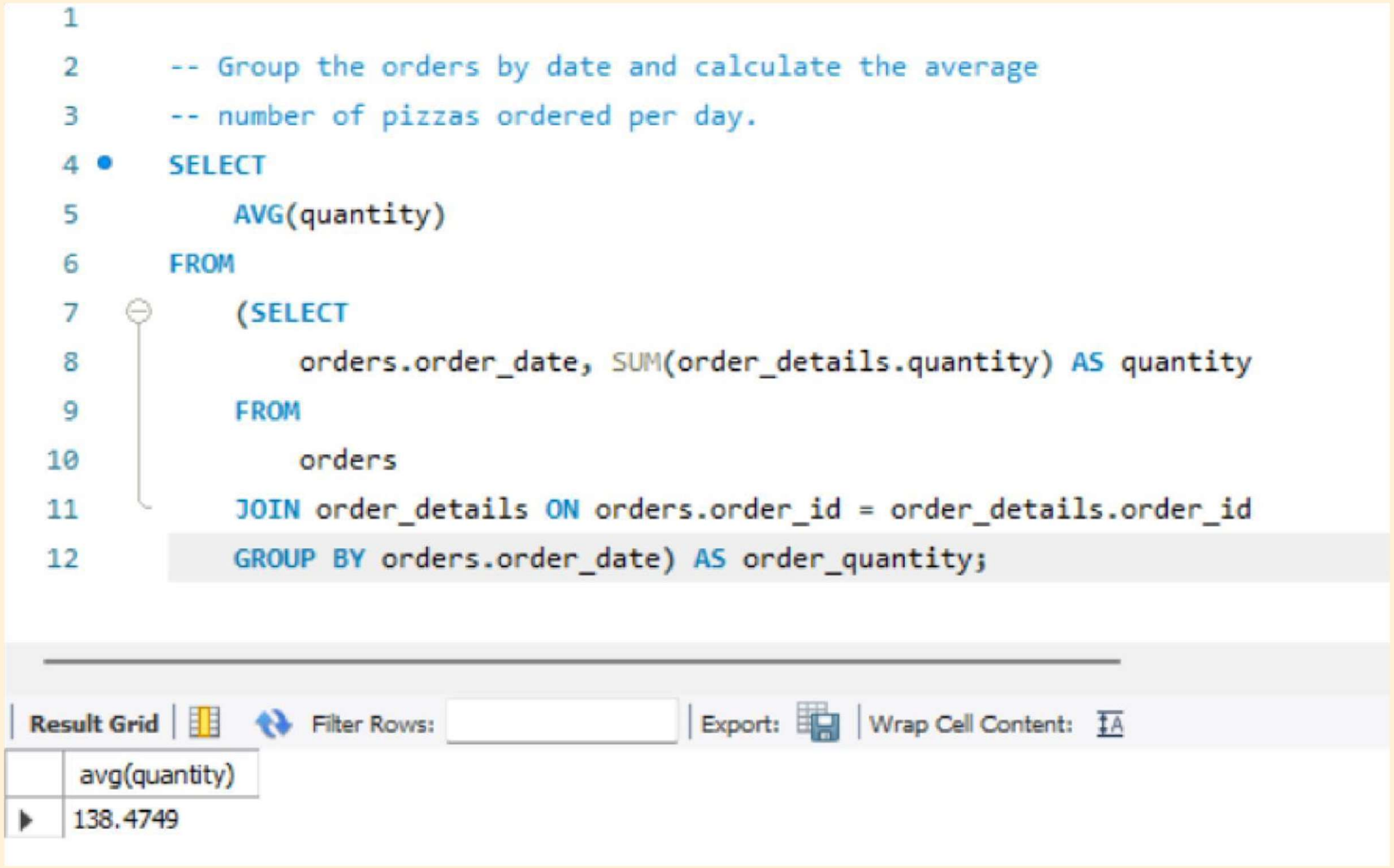


```sql

-- Join relevant tables to find the category-wise distribution of pizzas.
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category
```

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

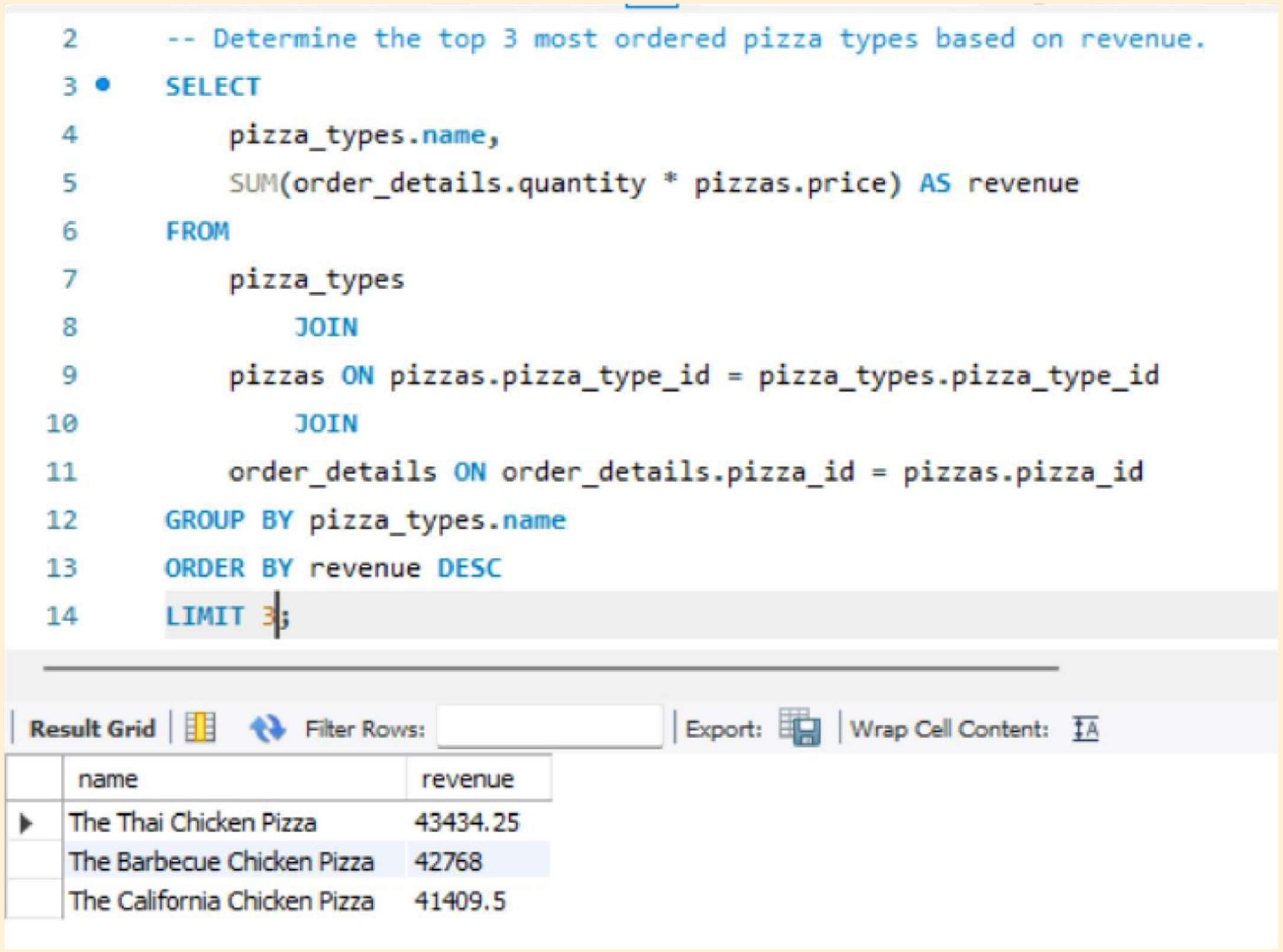
QUERY 8



```
-- Group the orders by date and calculate the average
-- number of pizzas ordered per day.
SELECT
    AVG(quantity)
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

```
avg(quantity)
138.4749
```

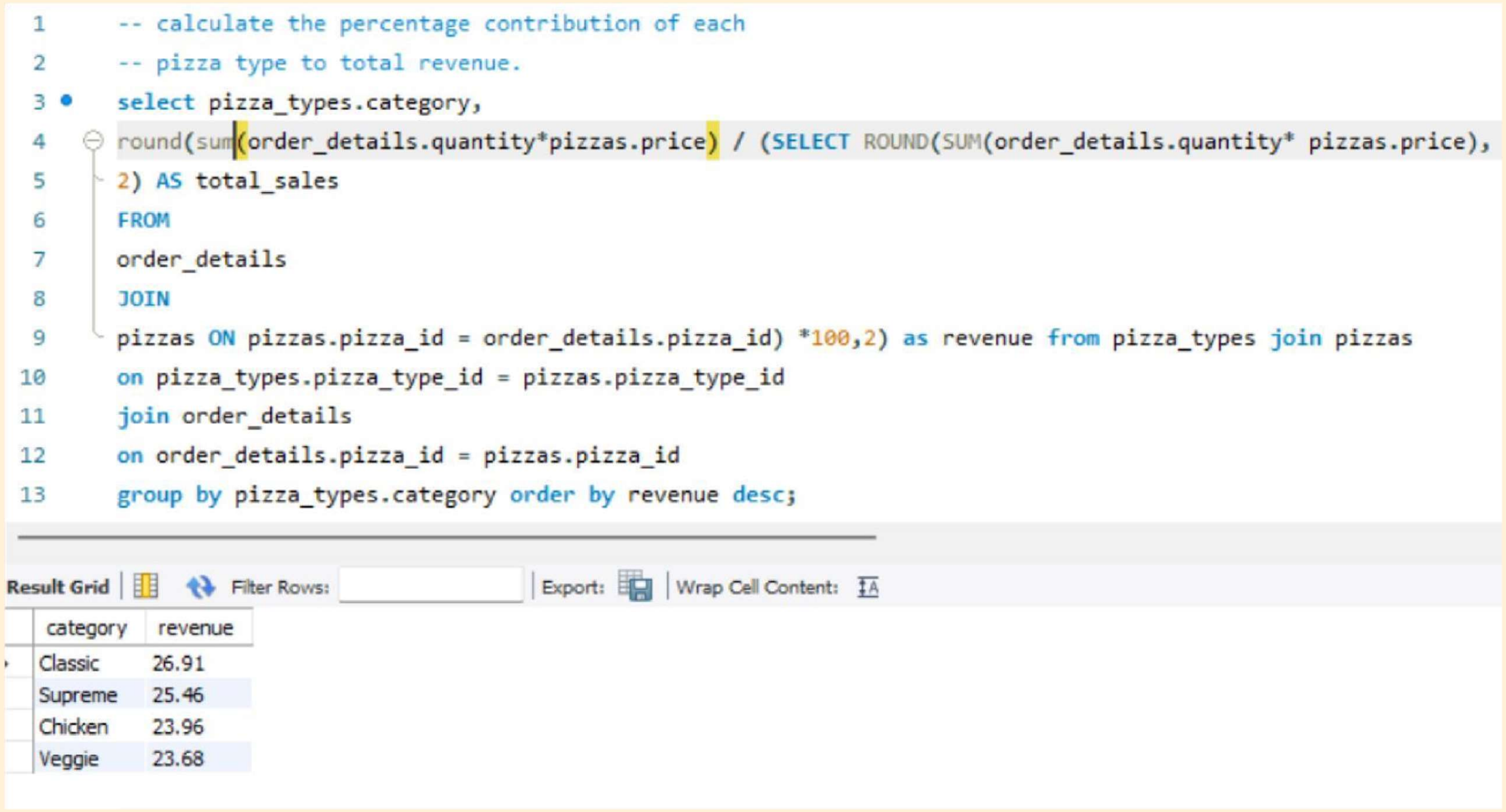
QUERY 9



```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 pizzas.pizza_type_id = pizza_types.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

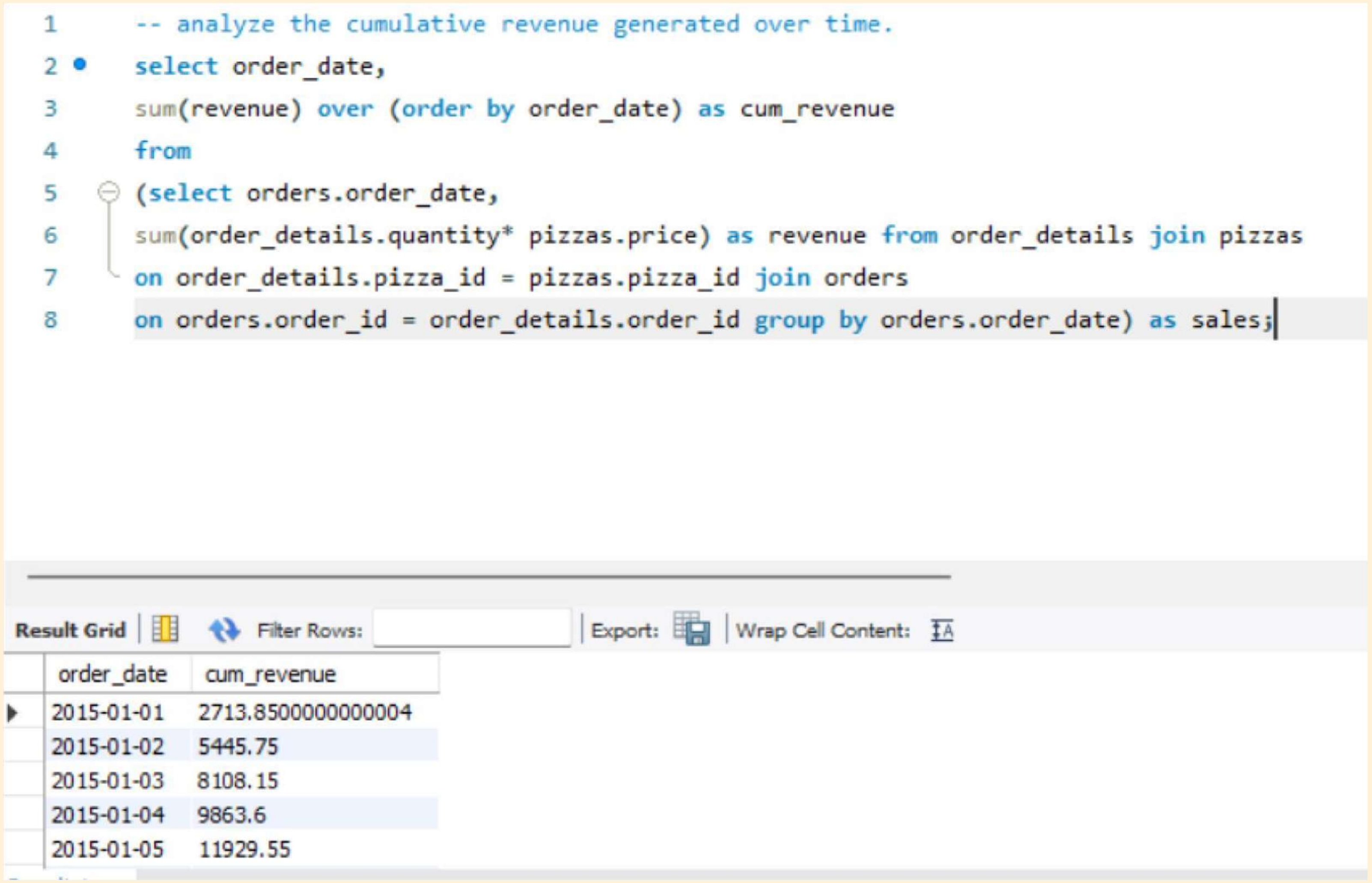
QUERY 10



```
-- 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;
```

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

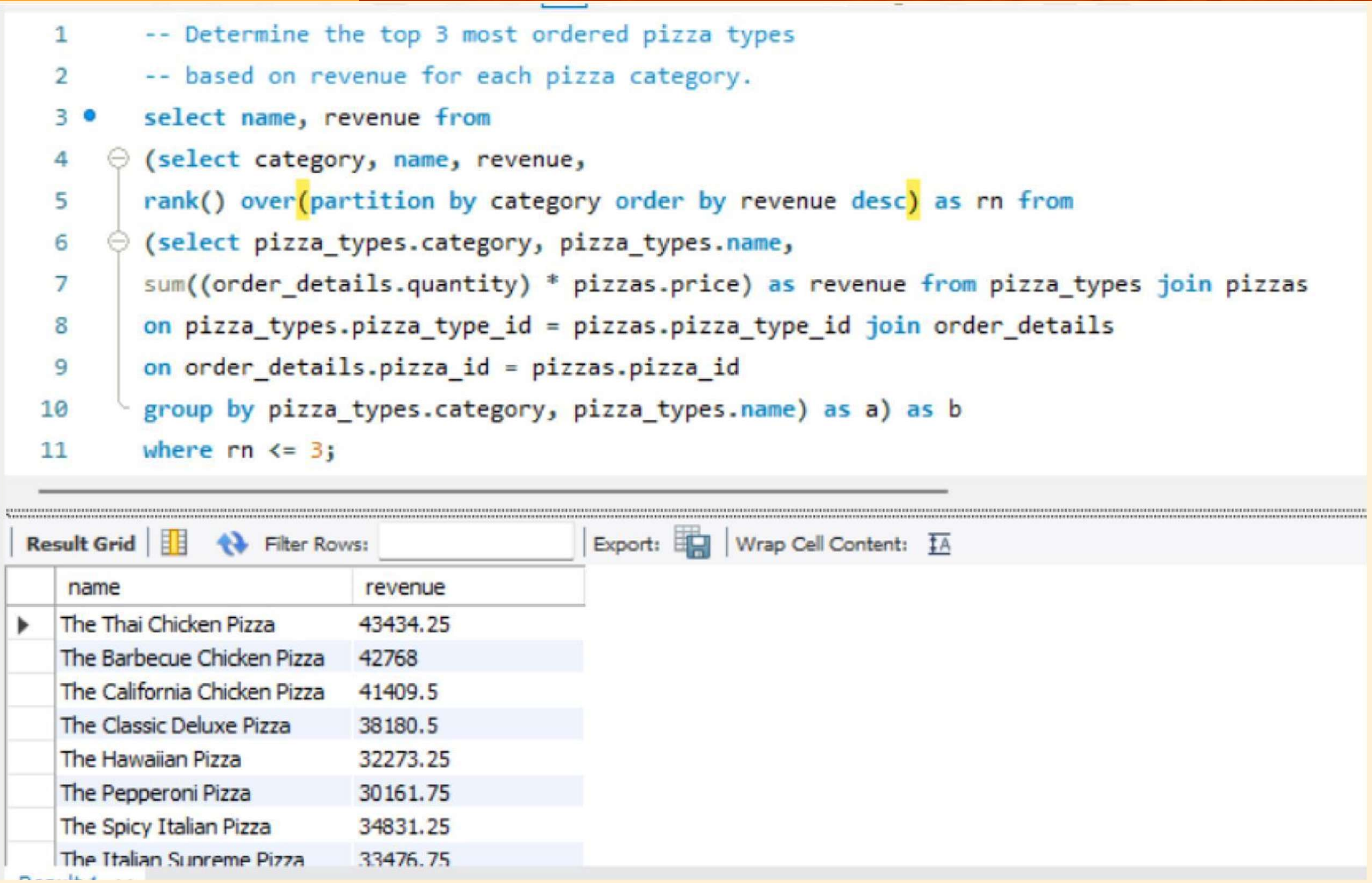
QUERY 11



```
-- 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 order_details join pizzas
on order_details.pizza_id = pizzas.pizza_id join orders
on orders.order_id = order_details.order_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
```

QUERY 12



```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;
```

```
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
```

QUERY 13

THANKYOU!!