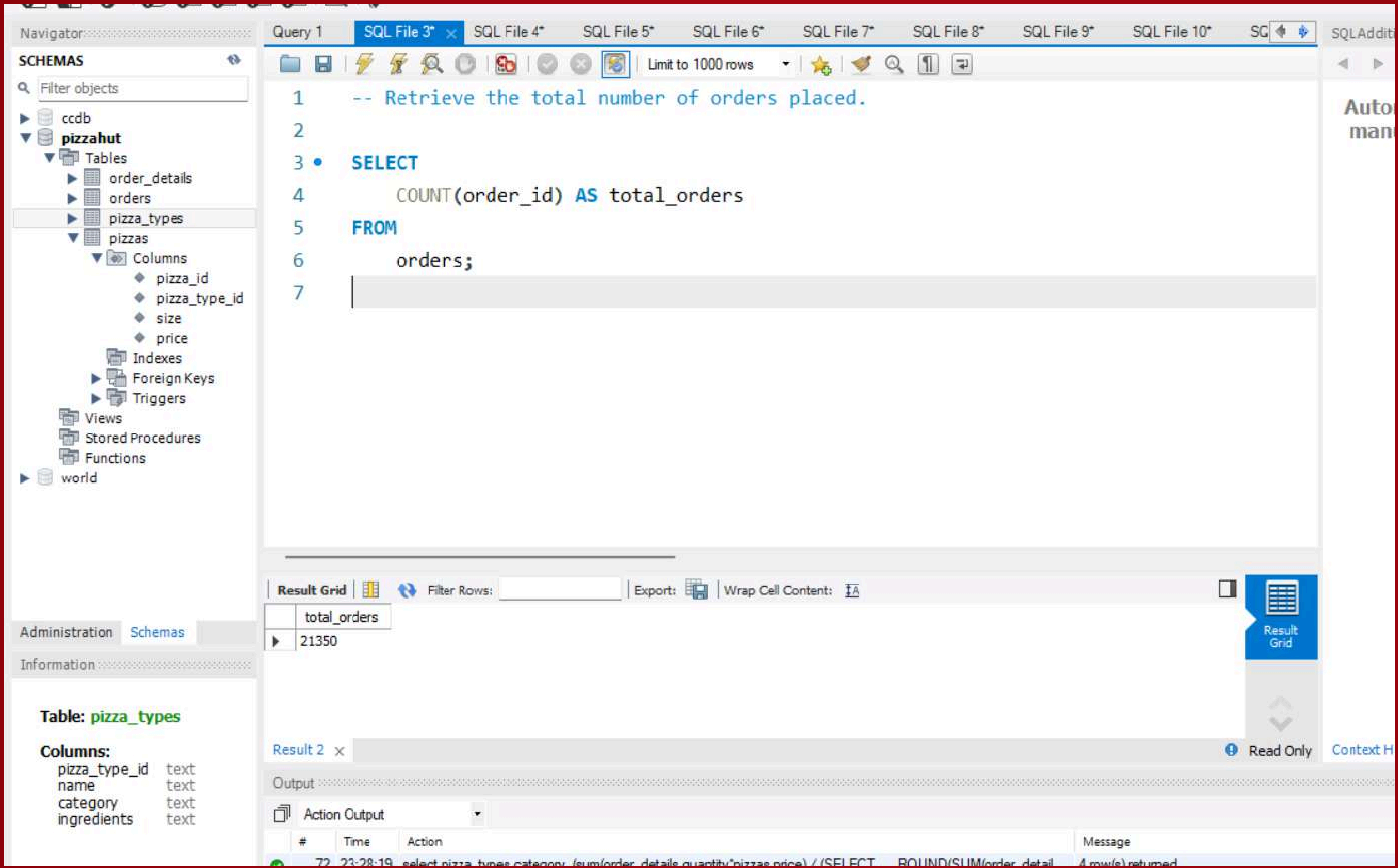
PIZZAHUT SALES ANALYSIS (USING MY SQL)

INTRODUCTION :

This project presents a comprehensive sales analysis of a pizza business using SQL querying techniques on transactional and product data. The analysis involves joining multiple related tables such as pizza types, individual pizzas, and order details to calculate key performance metrics like revenue by pizza category and size. Advanced SQL functions including aggregation and window functions are utilized to rank pizza categories based on sales performance.

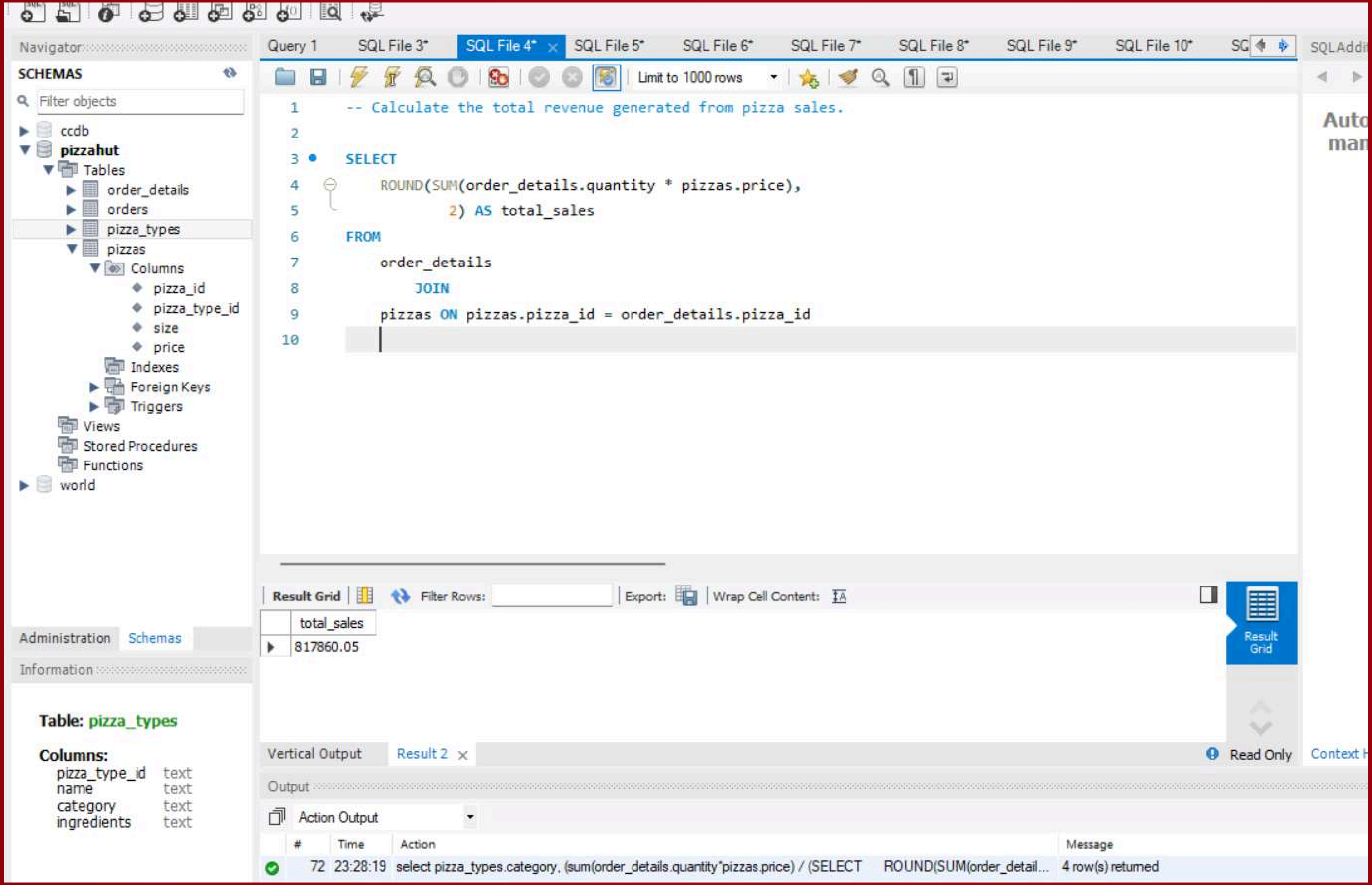
Q)1 - RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED



```
-- 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 pizzas.pizza_id = order_details.pizza_id

```

```
total_sales
817860.05
```

```
#   Time      Action                                                                                          Message
72  23:28:19  select pizza_types.category, (sum(order_details.quantity*pizzas.price) / (SELECT   ROUND(SUM(order_detail...   4 row(s) returned
```

Table: pizza_types

Columns:

pizza_type_id	text
name	text
category	text
ingredients	text

Q3) IDENTIFY THE HIGHEST-PRICED PIZZA.

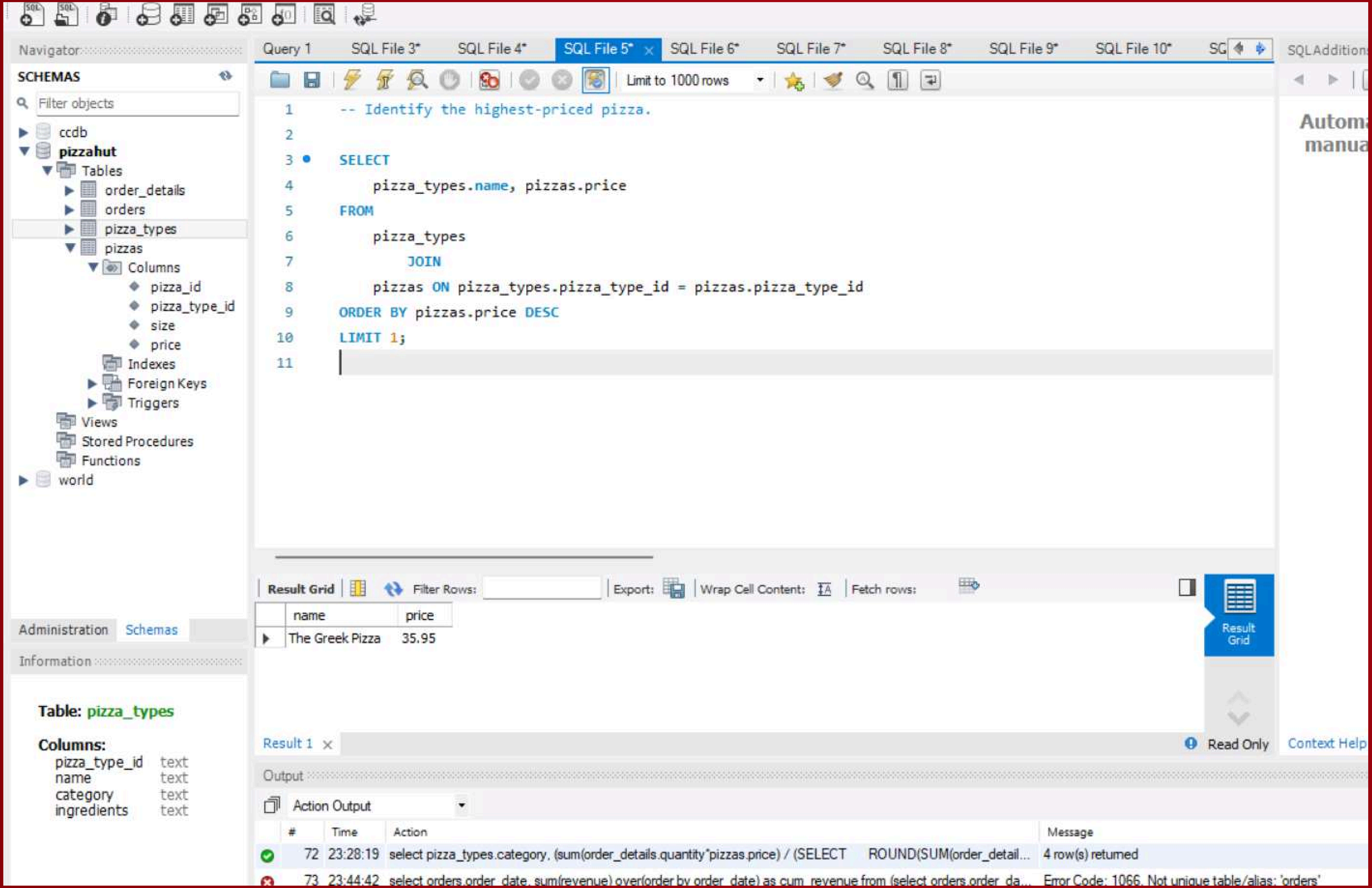


Table: pizza_types

Columns:

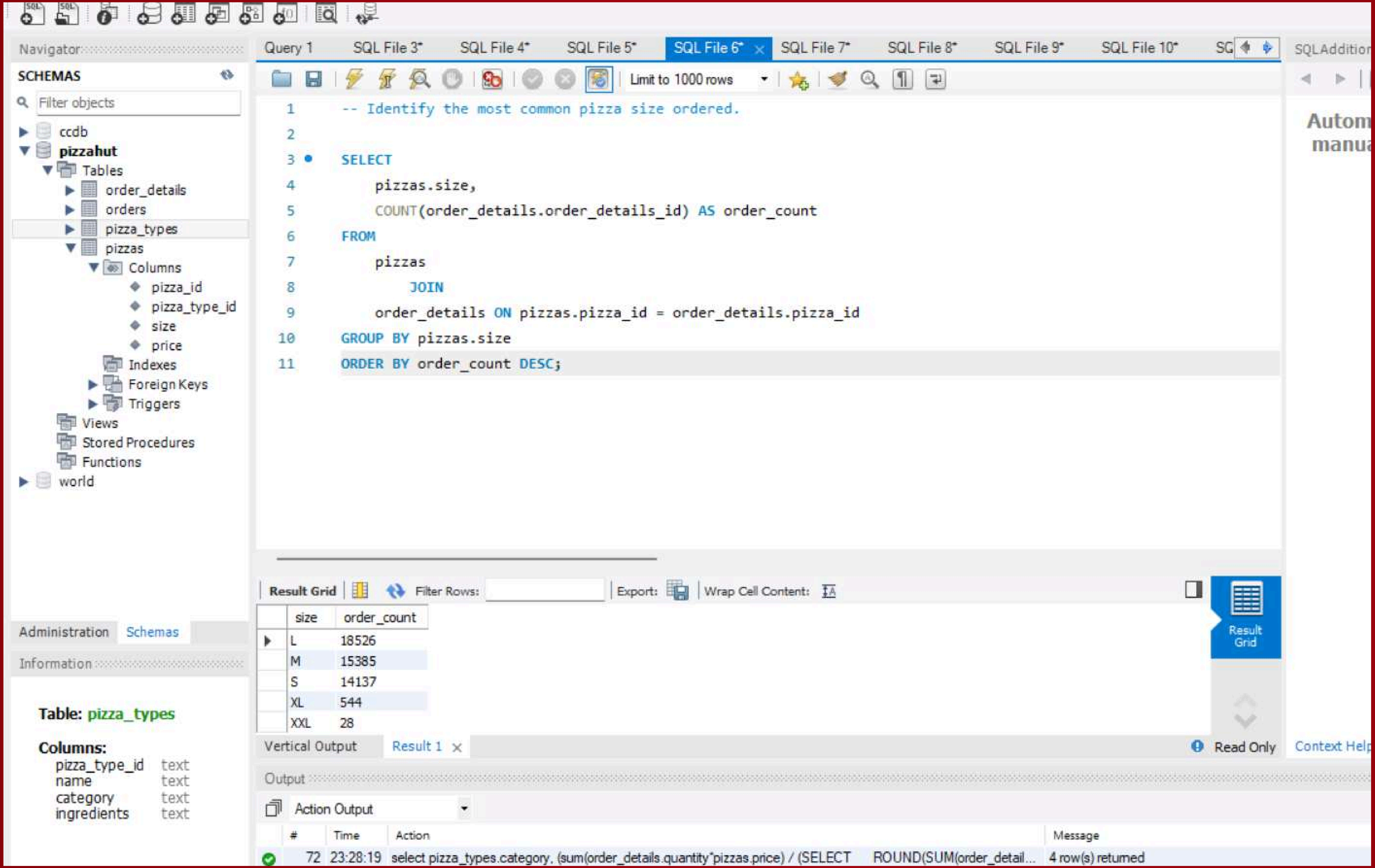
pizza_type_id	text
name	text
category	text
ingredients	text

```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

```
#   Time      Action                                                                                       Message
72  23:28:19  select pizza_types.category, (sum(order_details.quantity*pizzas.price) / (SELECT     ROUND(SUM(order_detail...   4 row(s) returned
73  23:44:42  select orders.order_date, sum(revenue) over(order by order_date) as cum_revenue from (select orders.order_da...   Error Code: 1066. Not unique table/alias: 'orders'
```

Q4) IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.



```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

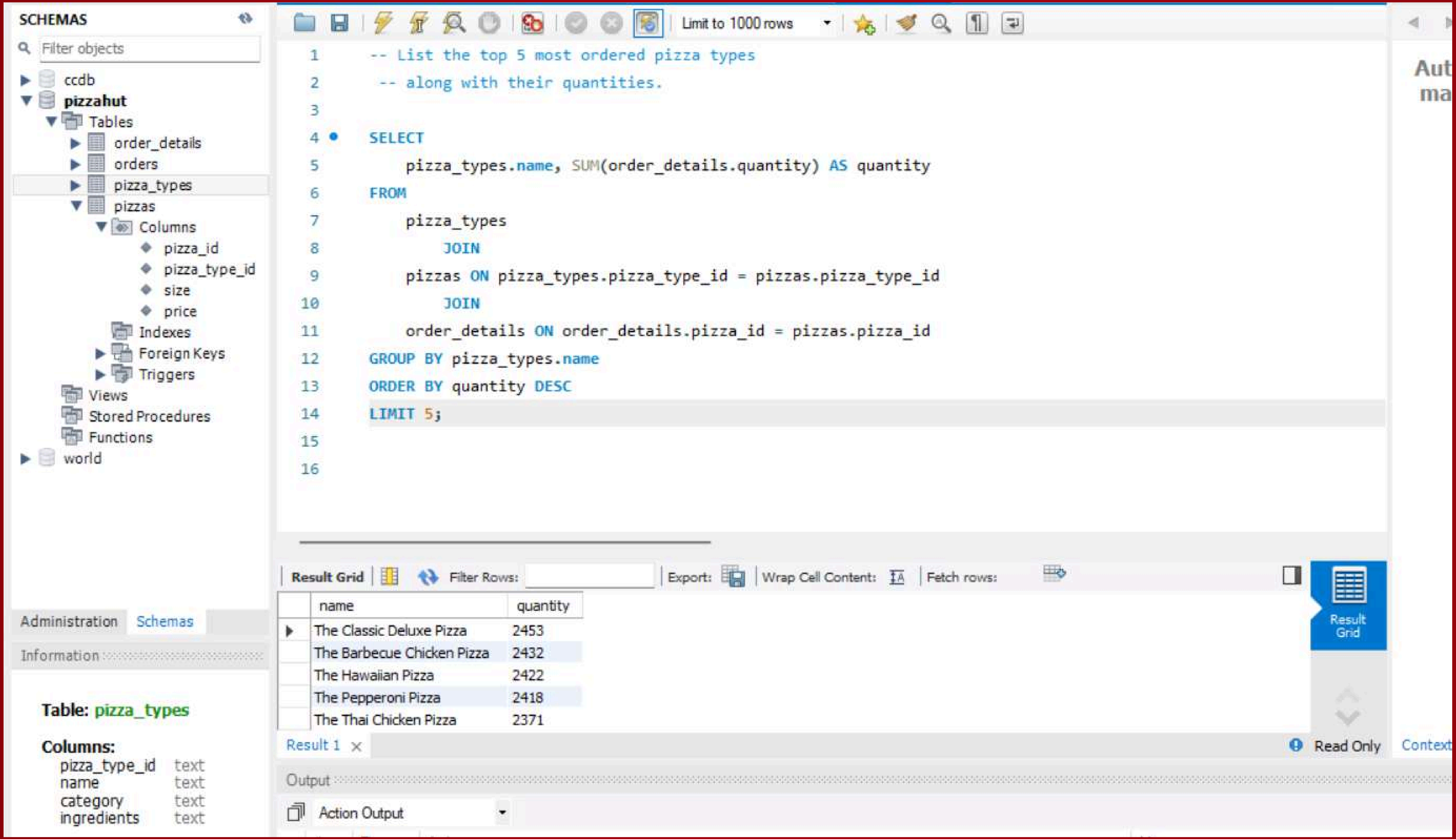
Table: pizza_types

Columns:

pizza_type_id	text
name	text
category	text
ingredients	text

#	Time	Action	Message
72	23:28:19	select pizza_types.category, (sum(order_details.quantity*pizzas.price) / (SELECT ROUND(SUM(order_detail...	4 row(s) returned

Q5) LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.



```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;
```

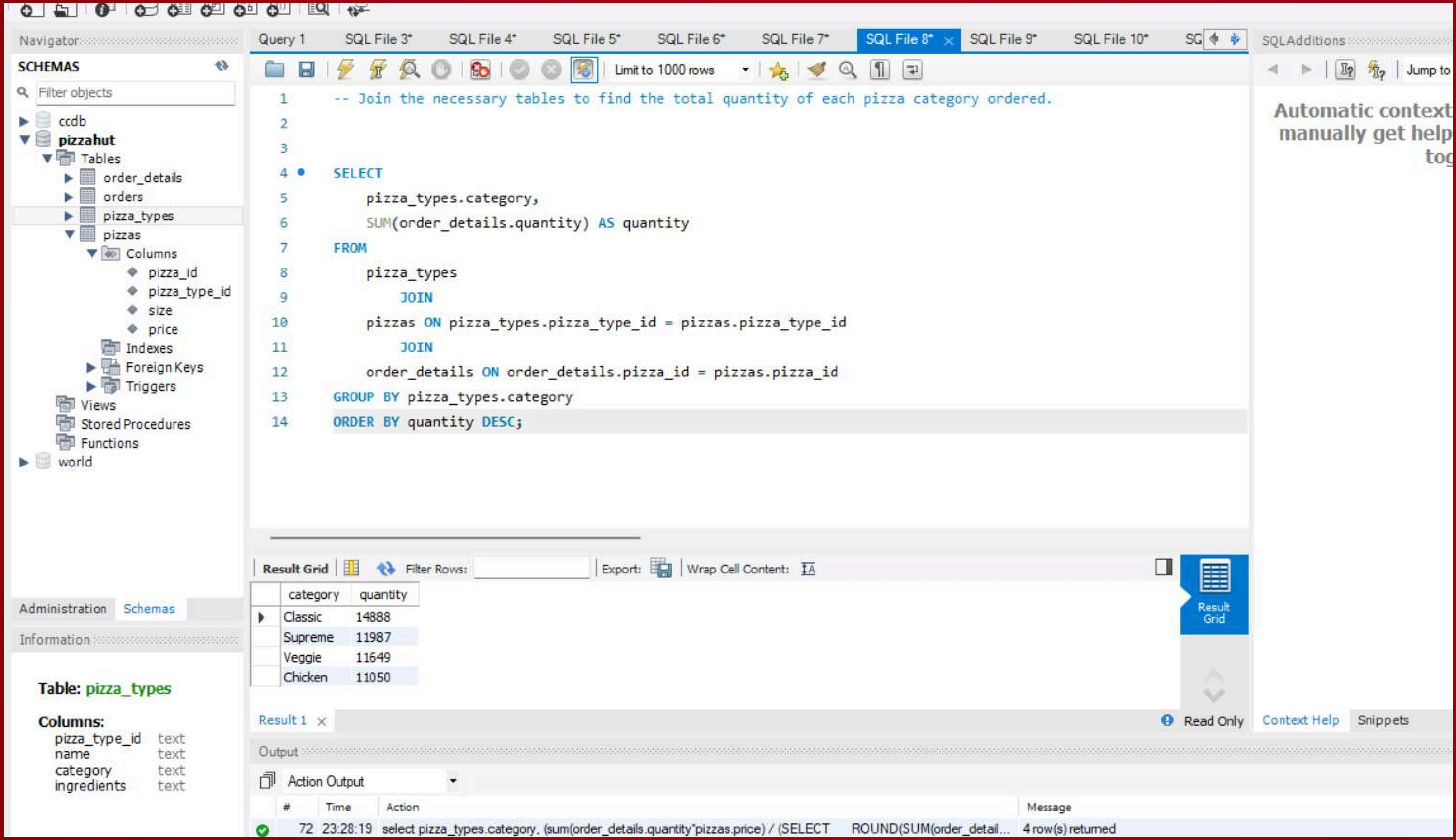
name	quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

Table: pizza_types

Columns:

pizza_type_id	text
name	text
category	text
ingredients	text

Q8) JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.



```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;
```

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

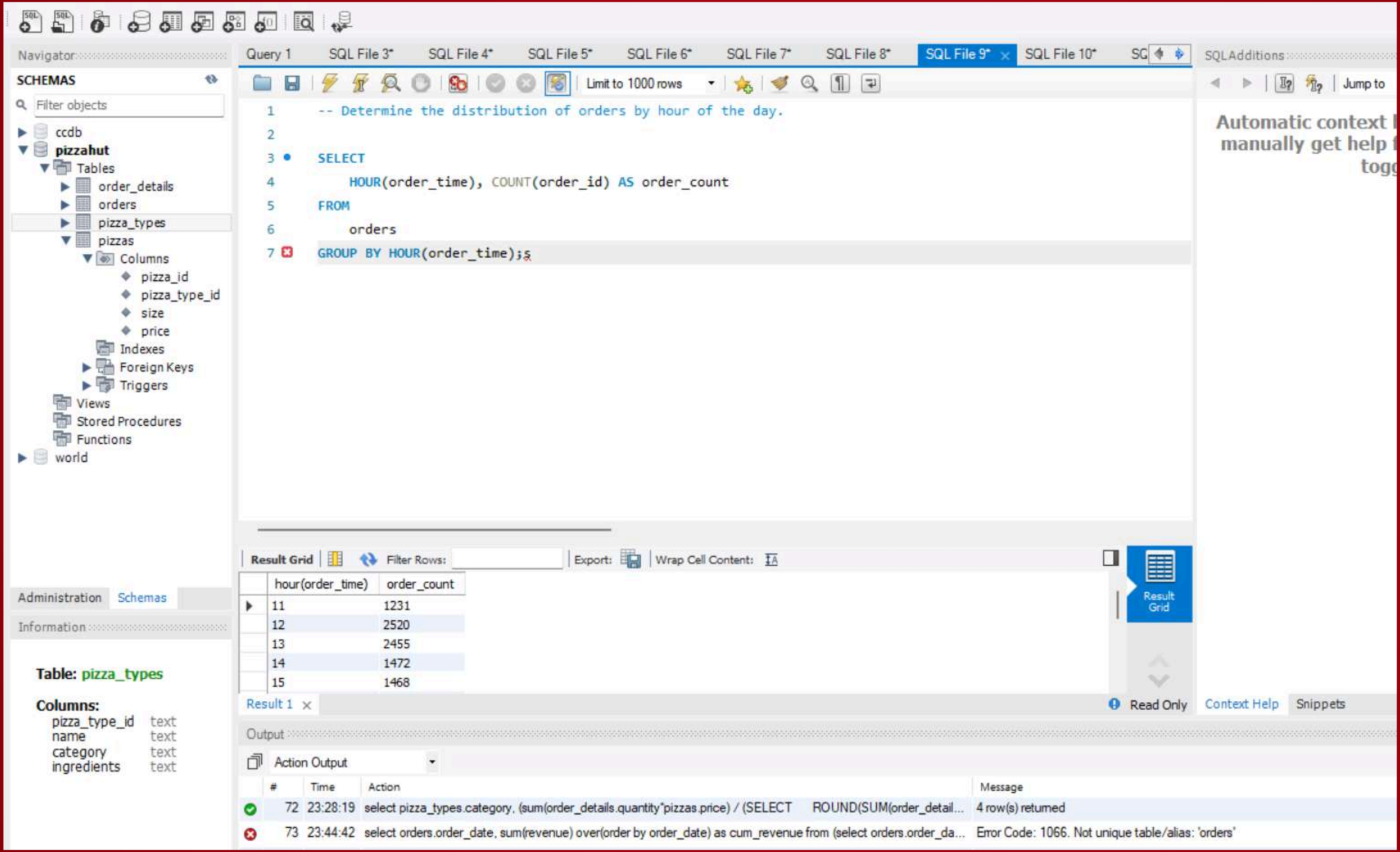
Table: pizza_types

Columns:

pizza_type_id	text
name	text
category	text
ingredients	text

```
#   Time      Action                                                                                        Message
72  23:28:19  select pizza_types.category, (sum(order_details.quantity*pizzas.price) / (SELECT   ROUND(SUM(order_detail...  4 row(s) returned
```

Q7) DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.



```sql
-- Determine the distribution of orders by hour of the day.

SELECT
    HOUR(order_time), COUNT(order_id) AS order_count
FROM
    orders
GROUP BY HOUR(order_time);s
```

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

Table: pizza_types

Columns:

pizza_type_id	text
name	text
category	text
ingredients	text

#	Time	Action	Message
72	23:28:19	select pizza_types.category, (sum(order_details.quantity*pizzas.price) / (SELECT ROUND(SUM(order_detail...	4 row(s) returned
73	23:44:42	select orders.order_date, sum(revenue) over(order by order_date) as cum_revenue from (select orders.order_da...	Error Code: 1066. Not unique table/alias: 'orders'

Q8) JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.

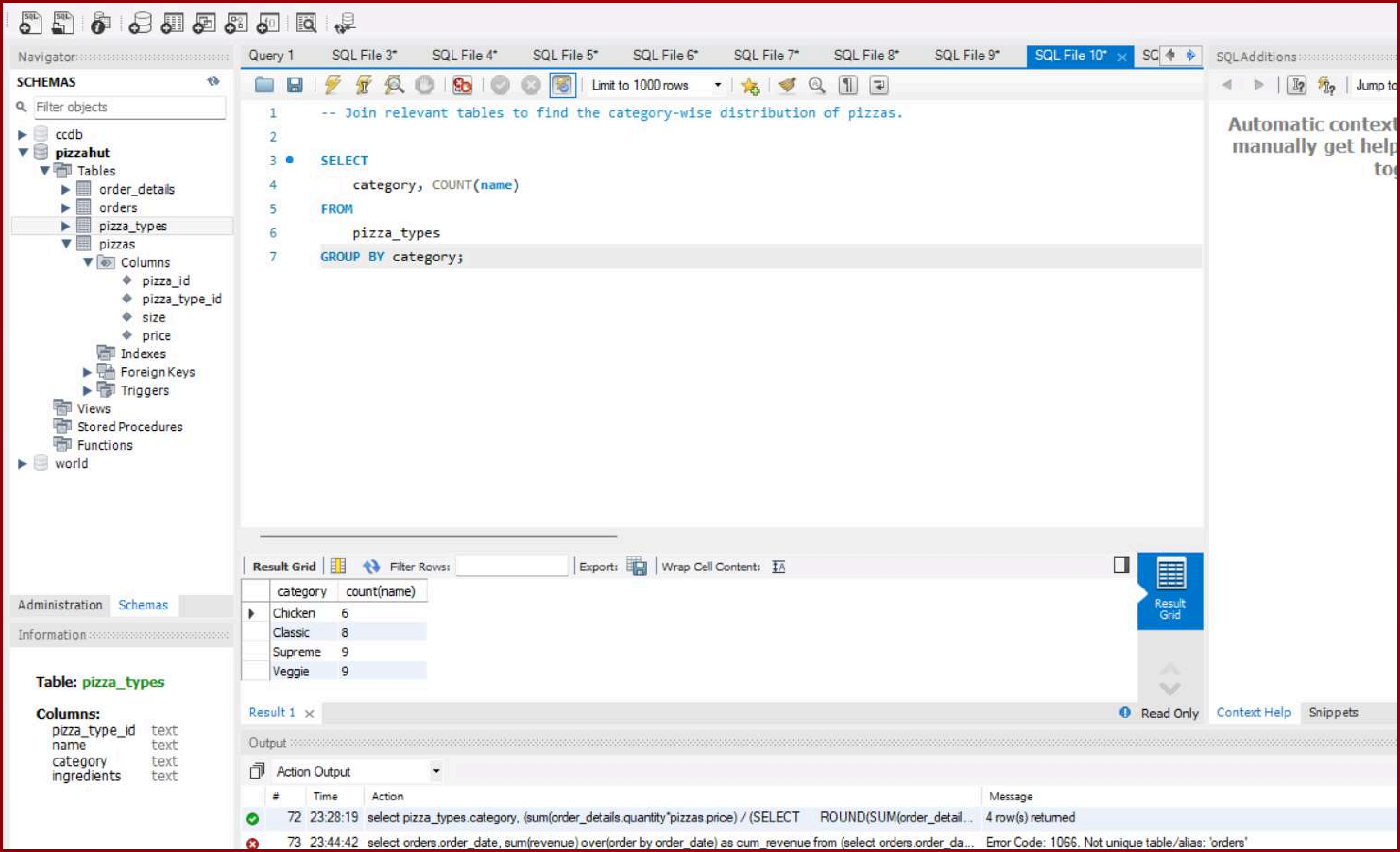


Table: pizza_types

Columns:

pizza_type_id	text
name	text
category	text
ingredients	text

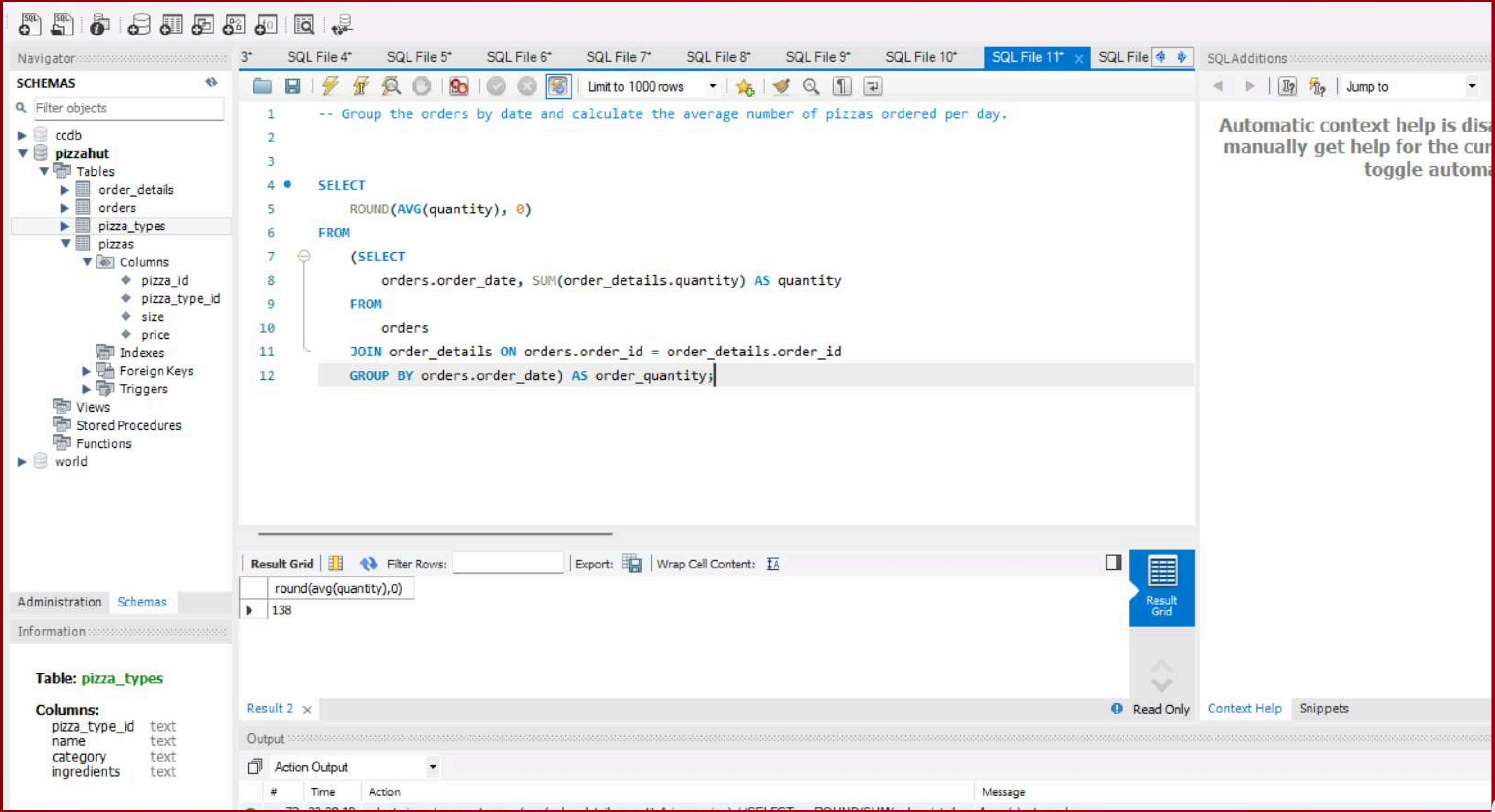
```sql
-- Join relevant tables to find the category-wise distribution of pizzas.

SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```

Automatic context manually get help to

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

#	Time	Action	Message
72	23:28:19	select pizza_types.category, (sum(order_details.quantity*pizzas.price) / (SELECT ROUND(SUM(order_detail...	4 row(s) returned
73	23:44:42	select orders.order_date, sum(revenue) over(order by order_date) as cum_revenue from (select orders.order_da...	Error Code: 1066. Not unique table/alias: 'orders'

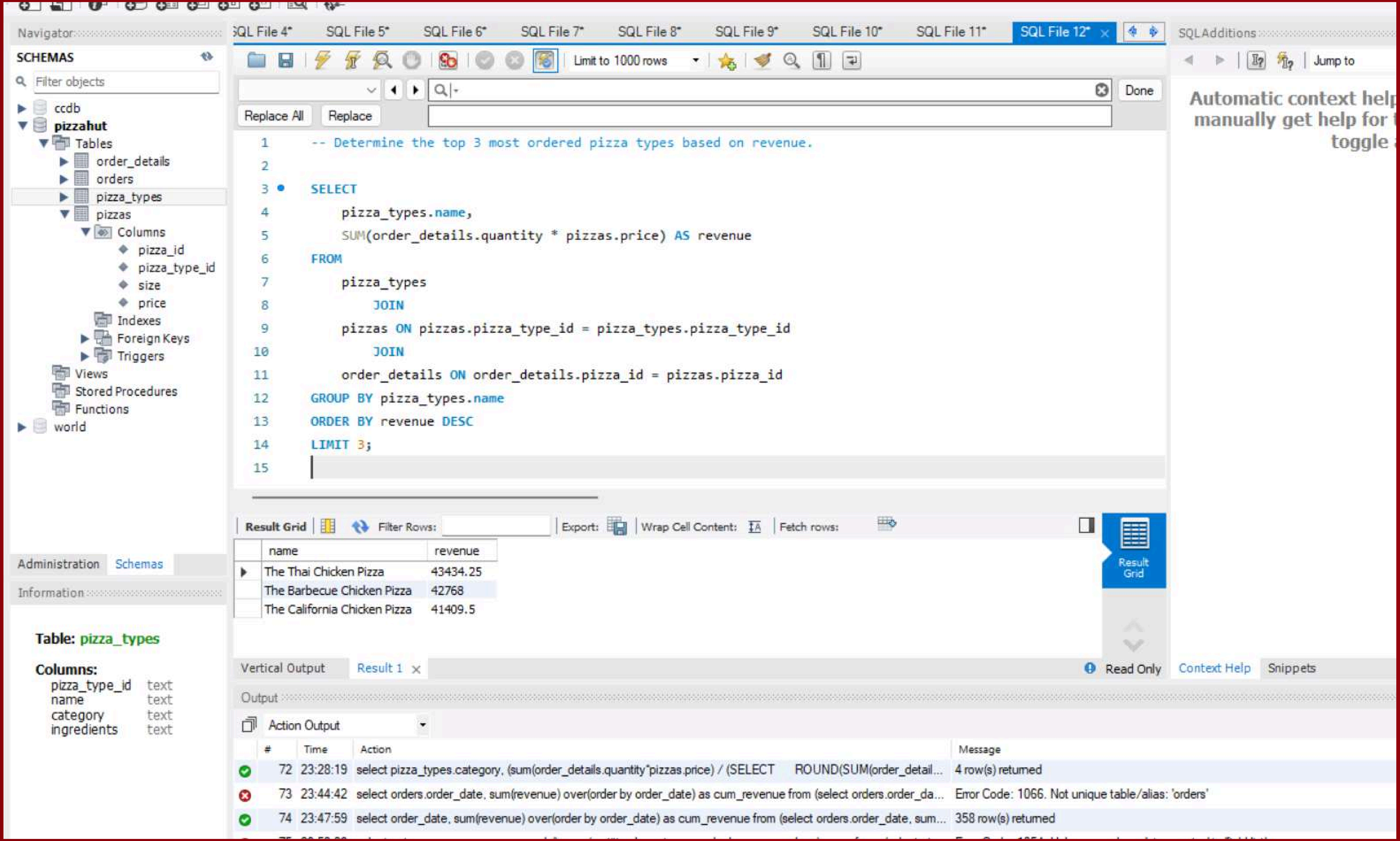
Q9) GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.



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


SELECT
    ROUND(AVG(quantity), 0)
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;
```

```
round(avg(quantity),0)
138
```

Q10) DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.



```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

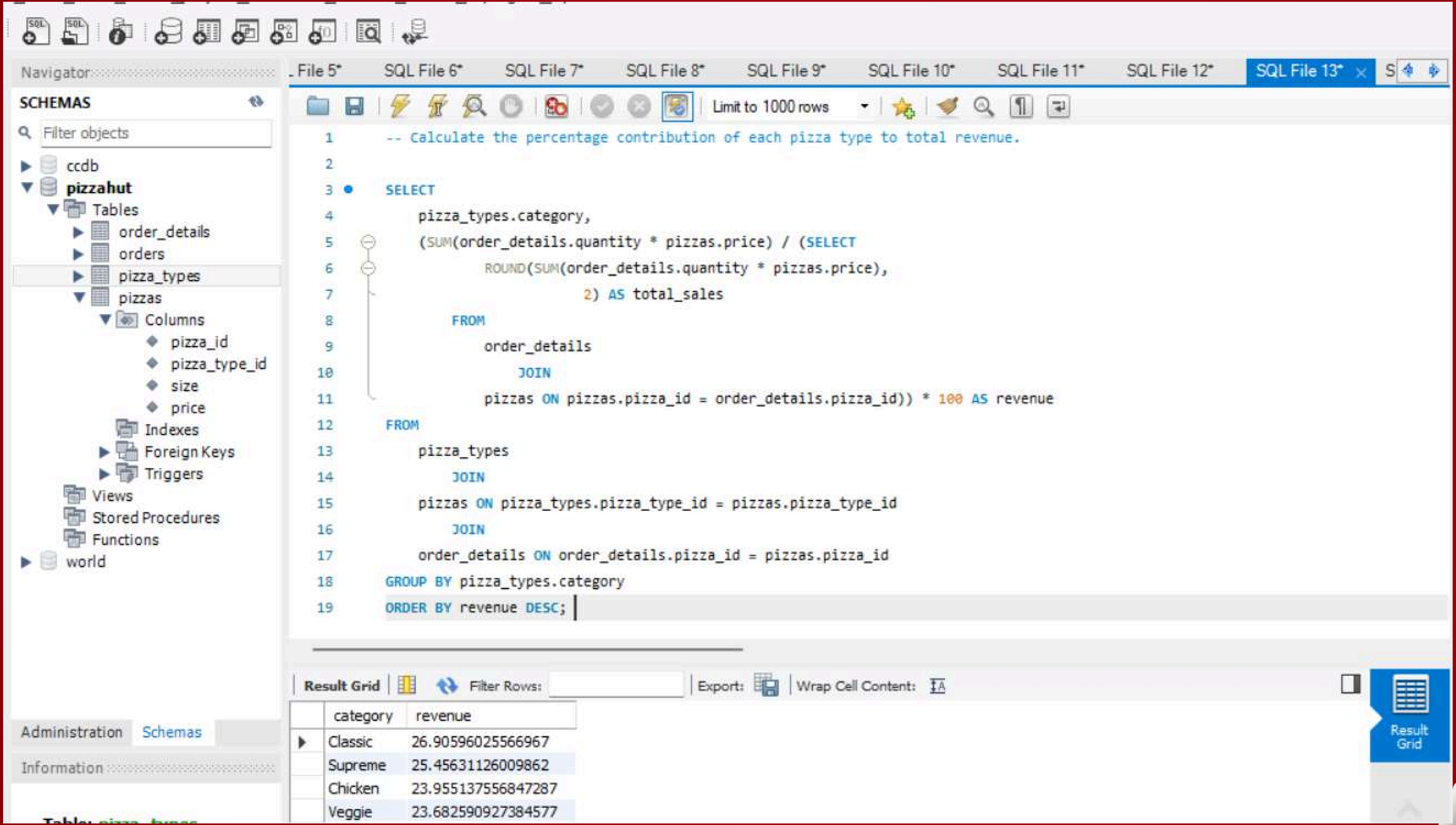
Table: pizza_types

Columns:

pizza_type_id	text
name	text
category	text
ingredients	text

```
#   Time      Action                                                                                          Message
72  23:28:19  select pizza_types.category, (sum(order_details.quantity*pizzas.price) / (SELECT   ROUND(SUM(order_detail...  4 row(s) returned
73  23:44:42  select orders.order_date, sum(revenue) over(order by order_date) as cum_revenue from (select orders.order_da...  Error Code: 1066. Not unique table/alias: 'orders'
74  23:47:59  select order_date, sum(revenue) over(order by order_date) as cum_revenue from (select orders.order_date, sum...  358 row(s) returned
```

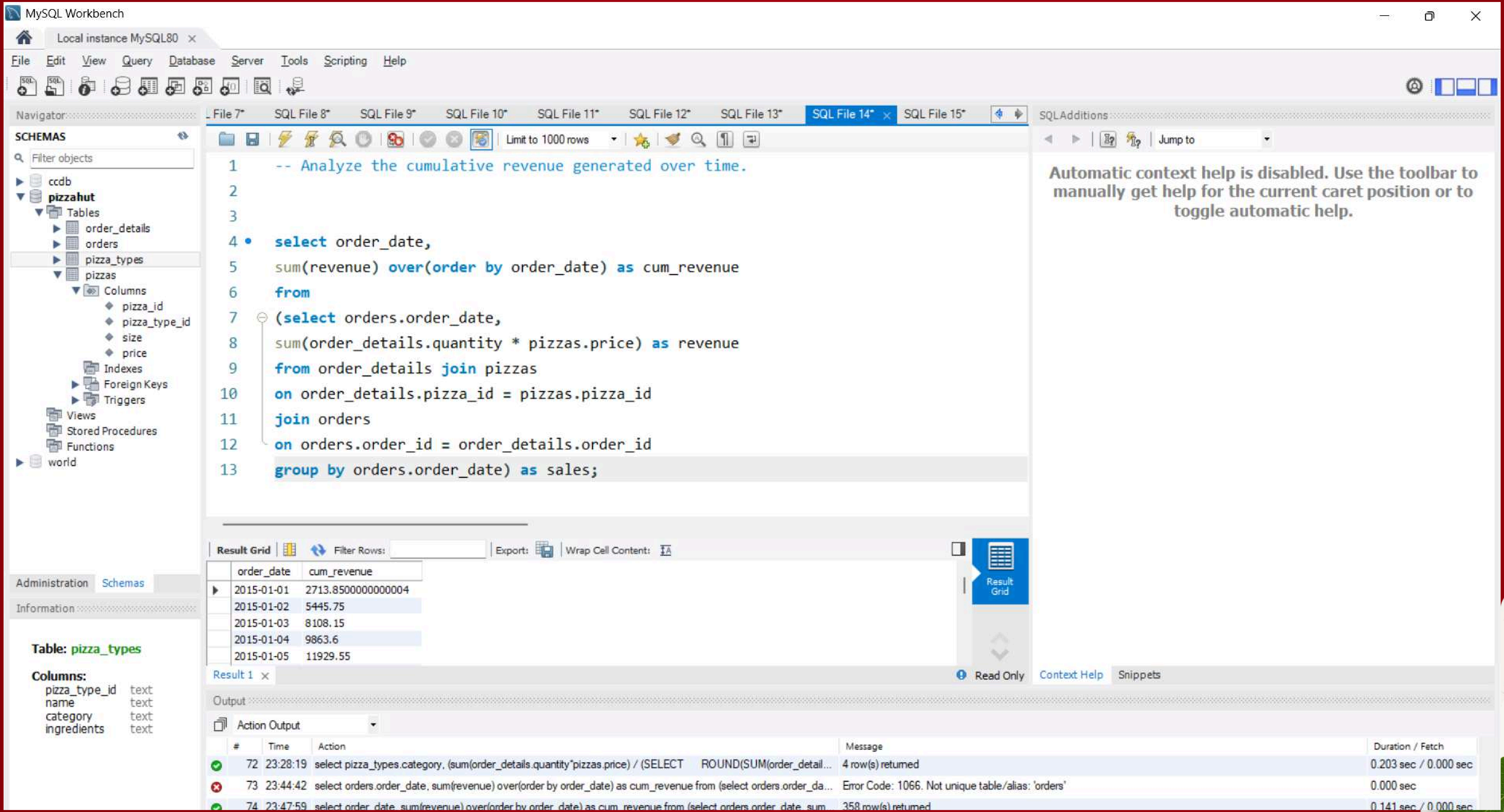
Q 11) CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE.



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

SELECT
    pizza_types.category,
    (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 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	26.90596025566967
Supreme	25.45631126009862
Chicken	23.955137556847287
Veggie	23.682590927384577

Q12) ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

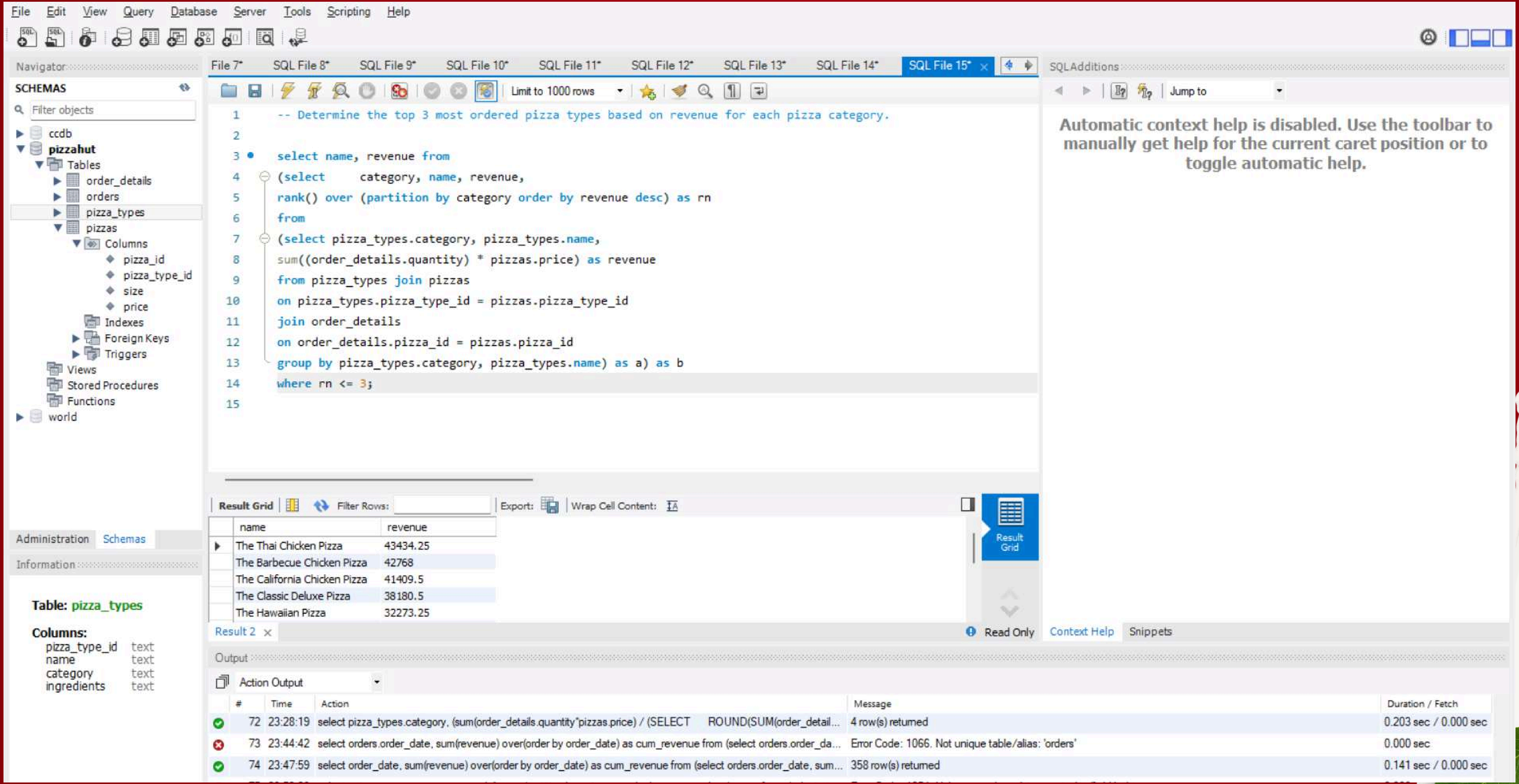


```sql
-- Analyze the cumulative revenue generated over time.


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

```
#   Time      Action                                                                                      Message                                              Duration / Fetch
72  23:28:19  select pizza_types.category, (sum(order_details.quantity*pizzas.price) / (SELECT    ROUND(SUM(order_detail...  4 row(s) returned                                    0.203 sec / 0.000 sec
73  23:44:42  select orders.order_date, sum(revenue) over(order by order_date) as cum_revenue from (select orders.order_da...  Error Code: 1066. Not unique table/alias: 'orders'   0.000 sec
74  23:47:59  select order_date, sum(revenue) over(order by order_date) as cum_revenue from (select orders.order_date, sum...  358 row(s) returned                                  0.141 sec / 0.000 sec
```

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 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;
```

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

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

Table: pizza_types

Columns:

pizza_type_id	text
name	text
category	text
ingredients	text

#	Time	Action	Message	Duration / Fetch
72	23:28:19	select pizza_types.category, (sum(order_details.quantity*pizzas.price) / (SELECT ROUND(SUM(order_detail...	4 row(s) returned	0.203 sec / 0.000 sec
73	23:44:42	select orders.order_date, sum(revenue) over(order by order_date) as cum_revenue from (select orders.order_da...	Error Code: 1066. Not unique table/alias: 'orders'	0.000 sec
74	23:47:59	select order_date, sum(revenue) over(order by order_date) as cum_revenue from (select orders.order_date, sum...	358 row(s) returned	0.141 sec / 0.000 sec

PIZZAHUT

THANK YOU!