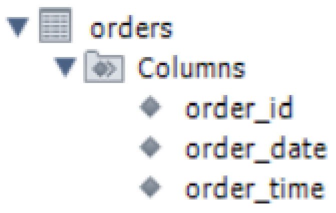
PIZZAHUT SALES - SQL PROJECT .

DATABASE NAME - PIZZAHUT

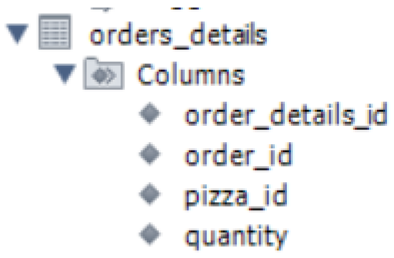
TABLES:

- ORDERS



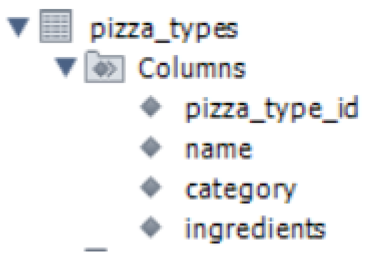
```
orders
  Columns
    order_id
    order_date
    order_time
```

- ORDERS_DETAILS



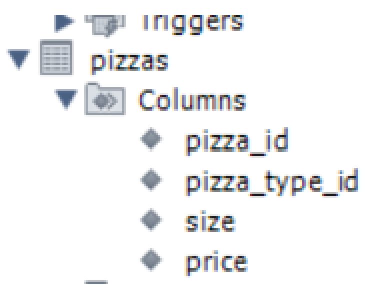
```
orders_details
  Columns
    order_details_id
    order_id
    pizza_id
    quantity
```

- PIZZA_TYPES



```
pizza_types
  Columns
    pizza_type_id
    name
    category
    ingredients
```

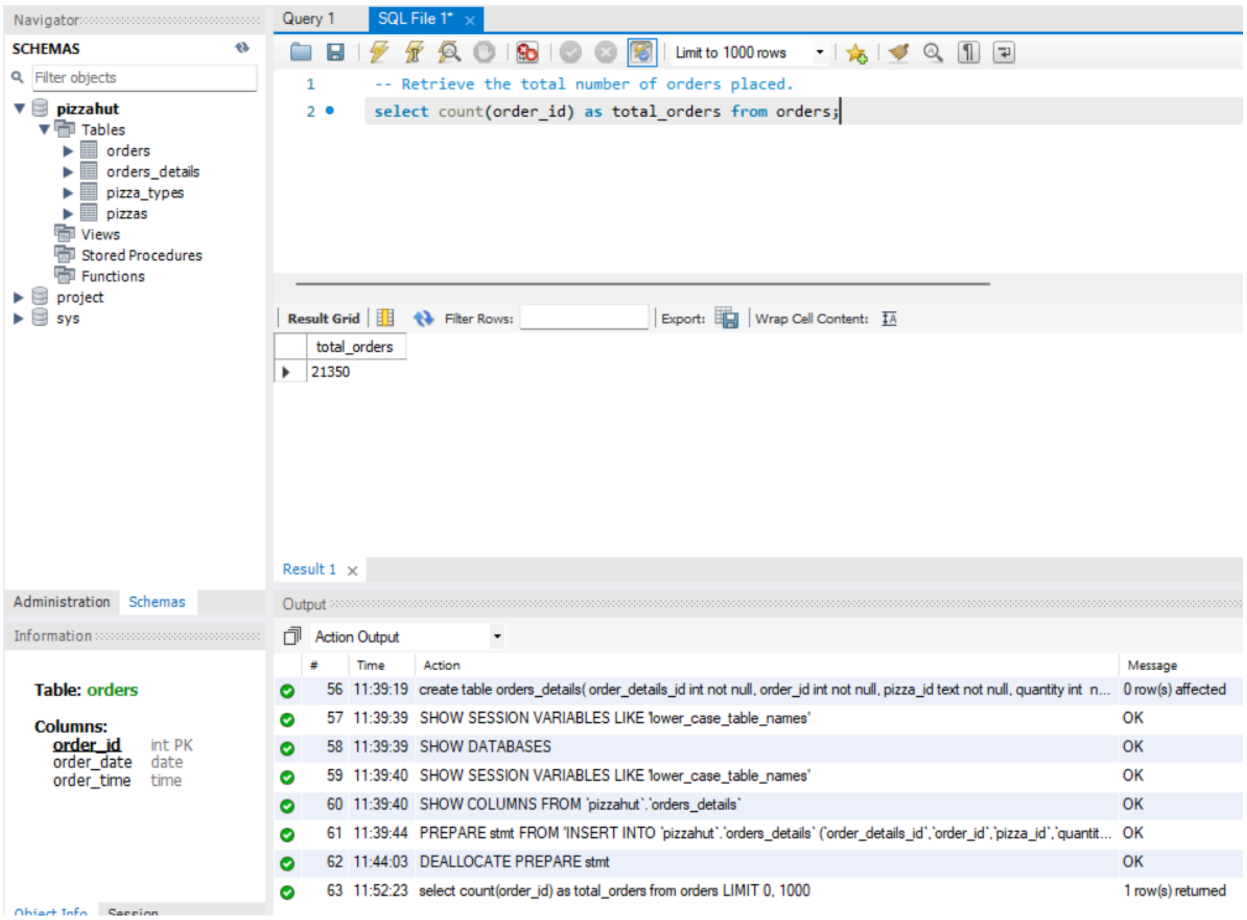
- PIZZAS



```
Triggers
pizzas
  Columns
    pizza_id
    pizza_type_id
    size
    price
```

SCREENSHOTS OF QUERIES

1. Retrieving total orders .



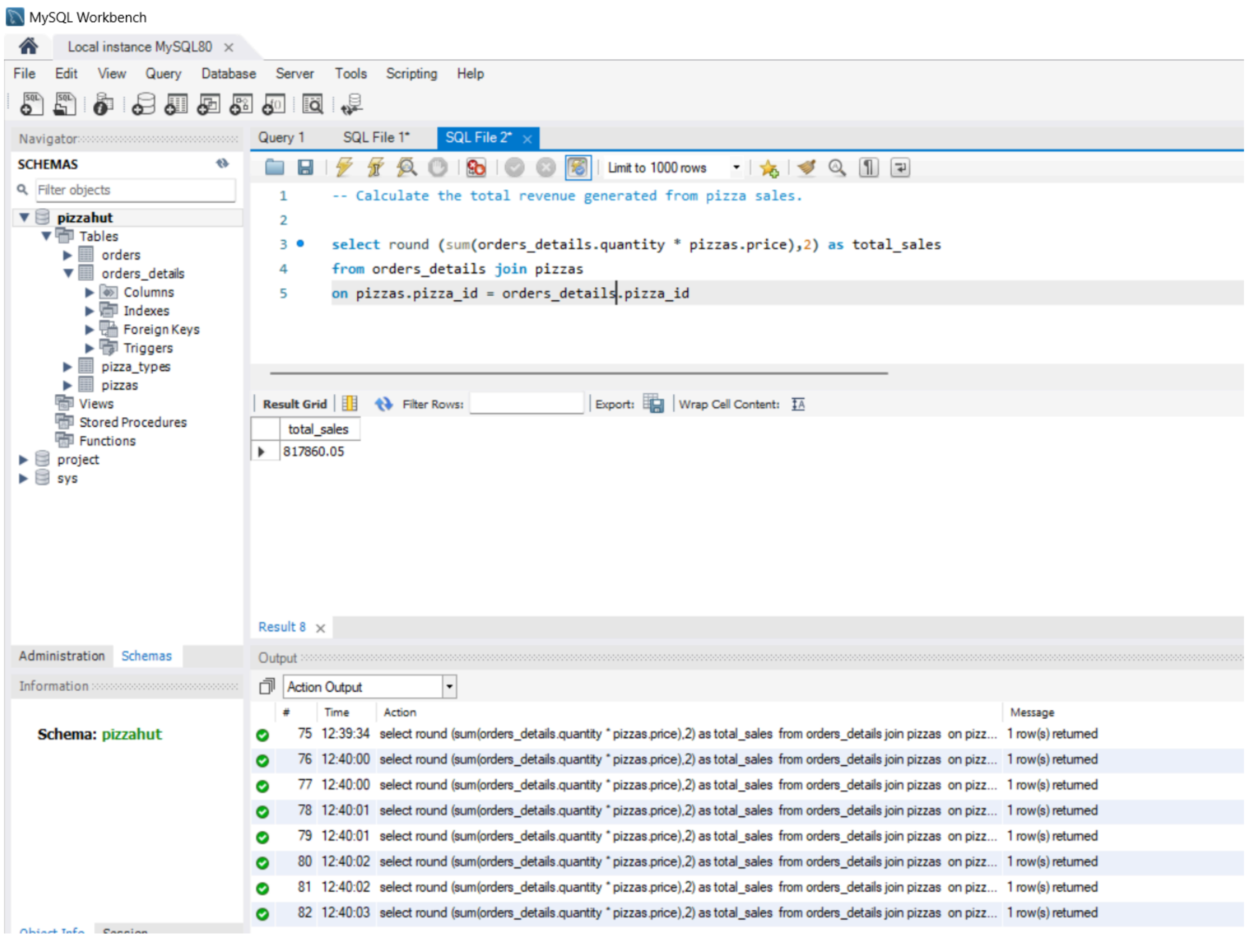
```sql
-- Retrieve the total number of orders placed.
select count(order_id) as total_orders from orders;
```

```
total_orders
21350
```

```
#   Time      Action                                                                                                   Message
56  11:39:19  create table orders_details( order_details_id int not null, order_id int not null, pizza_id text not null, quantity int  n...   0 row(s) affected
57  11:39:39  SHOW SESSION VARIABLES LIKE 'lower_case_table_names'                                                    OK
58  11:39:39  SHOW DATABASES                                                                                           OK
59  11:39:40  SHOW SESSION VARIABLES LIKE 'lower_case_table_names'                                                    OK
60  11:39:40  SHOW COLUMNS FROM `pizzahut`.`orders_details`                                                           OK
61  11:39:44  PREPARE stmt FROM 'INSERT INTO `pizzahut`.`orders_details` (`order_details_id`,`order_id`,`pizza_id`,`quantit...   OK
62  11:44:03  DEALLOCATE PREPARE stmt                                                                                  OK
63  11:52:23  select count(order_id) as total_orders from orders LIMIT 0, 1000                                         1 row(s) returned
```

```
Table: orders
Columns:
  order_id     int PK
  order_date   date
  order_time   time
```

2.Calculate the total revenue generated from pizza sales.



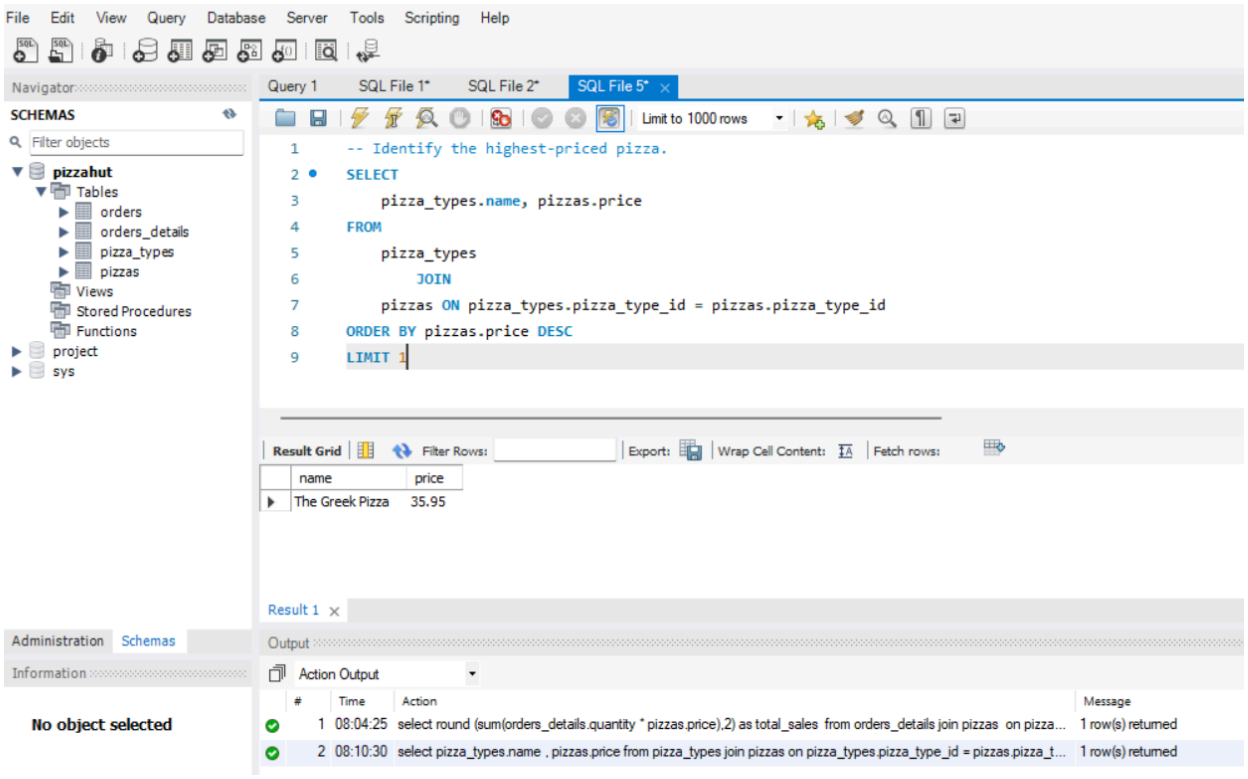
```sql
-- Calculate the total revenue generated from pizza sales.

select round (sum(orders_details.quantity * pizzas.price),2) as total_sales
from orders_details join pizzas
on pizzas.pizza_id = orders_details.pizza_id
```

```
total_sales
817860.05
```

Schema: pizzahut

```
#   Time      Action                                                                                                                  Message
75  12:39:34  select round (sum(orders_details.quantity * pizzas.price),2) as total_sales  from orders_details join pizzas  on pizz...  1 row(s) returned
76  12:40:00  select round (sum(orders_details.quantity * pizzas.price),2) as total_sales  from orders_details join pizzas  on pizz...  1 row(s) returned
77  12:40:00  select round (sum(orders_details.quantity * pizzas.price),2) as total_sales  from orders_details join pizzas  on pizz...  1 row(s) returned
78  12:40:01  select round (sum(orders_details.quantity * pizzas.price),2) as total_sales  from orders_details join pizzas  on pizz...  1 row(s) returned
79  12:40:01  select round (sum(orders_details.quantity * pizzas.price),2) as total_sales  from orders_details join pizzas  on pizz...  1 row(s) returned
80  12:40:02  select round (sum(orders_details.quantity * pizzas.price),2) as total_sales  from orders_details join pizzas  on pizz...  1 row(s) returned
81  12:40:02  select round (sum(orders_details.quantity * pizzas.price),2) as total_sales  from orders_details join pizzas  on pizz...  1 row(s) returned
82  12:40:03  select round (sum(orders_details.quantity * pizzas.price),2) as total_sales  from orders_details join pizzas  on pizz...  1 row(s) returned
```

3. To calculate highest pricing of pizzas

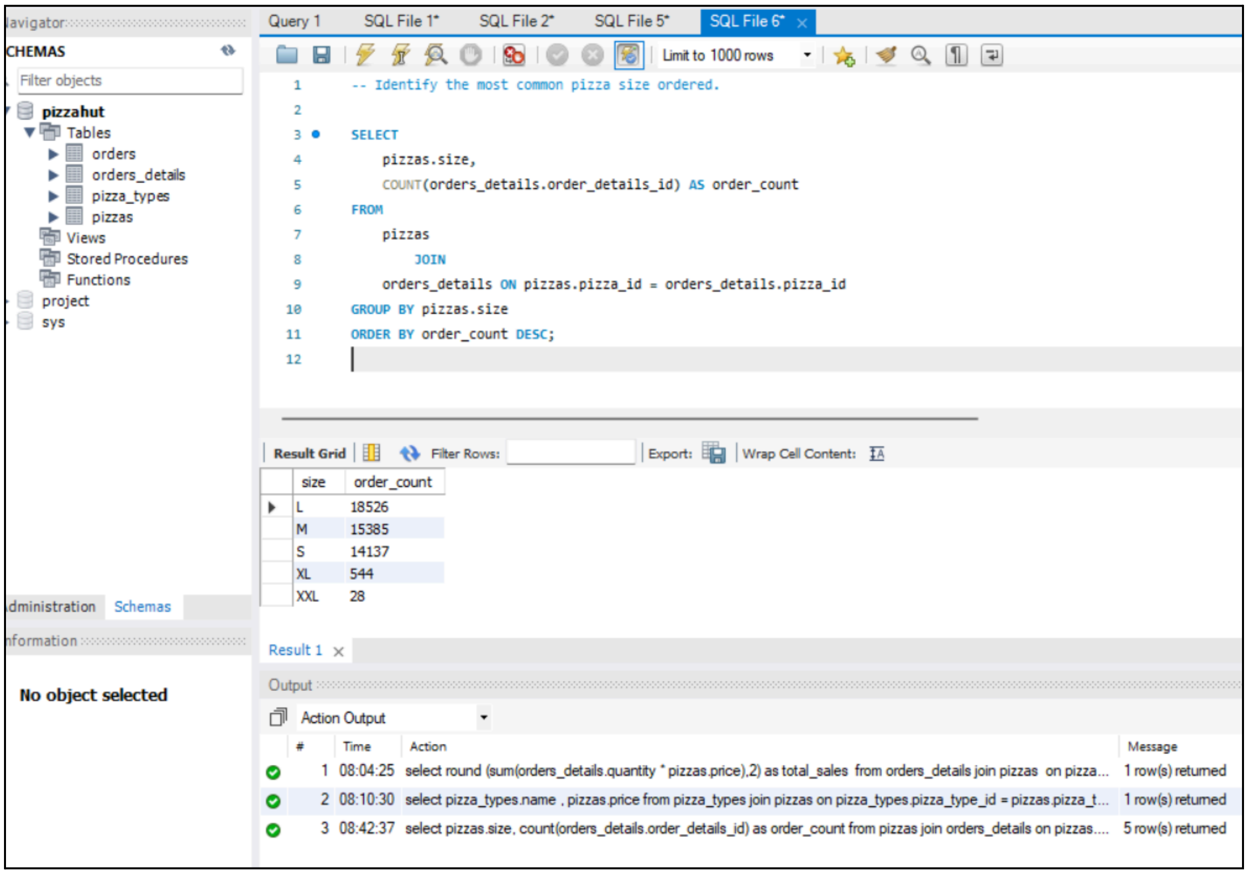


```sql
-- Identify the highest-priced pizza.
SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1
```

name	price
The Greek Pizza	35.95

#	Time	Action	Message
1	08:04:25	select round (sum(orders_details.quantity * pizzas.price),2) as total_sales from orders_details join pizzas on pizza...	1 row(s) returned
2	08:10:30	select pizza_types.name , pizzas.price from pizza_types join pizzas on pizza_types.pizza_type_id = pizzas.pizza_t...	1 row(s) returned

4.Identify the most common pizza size ordered.

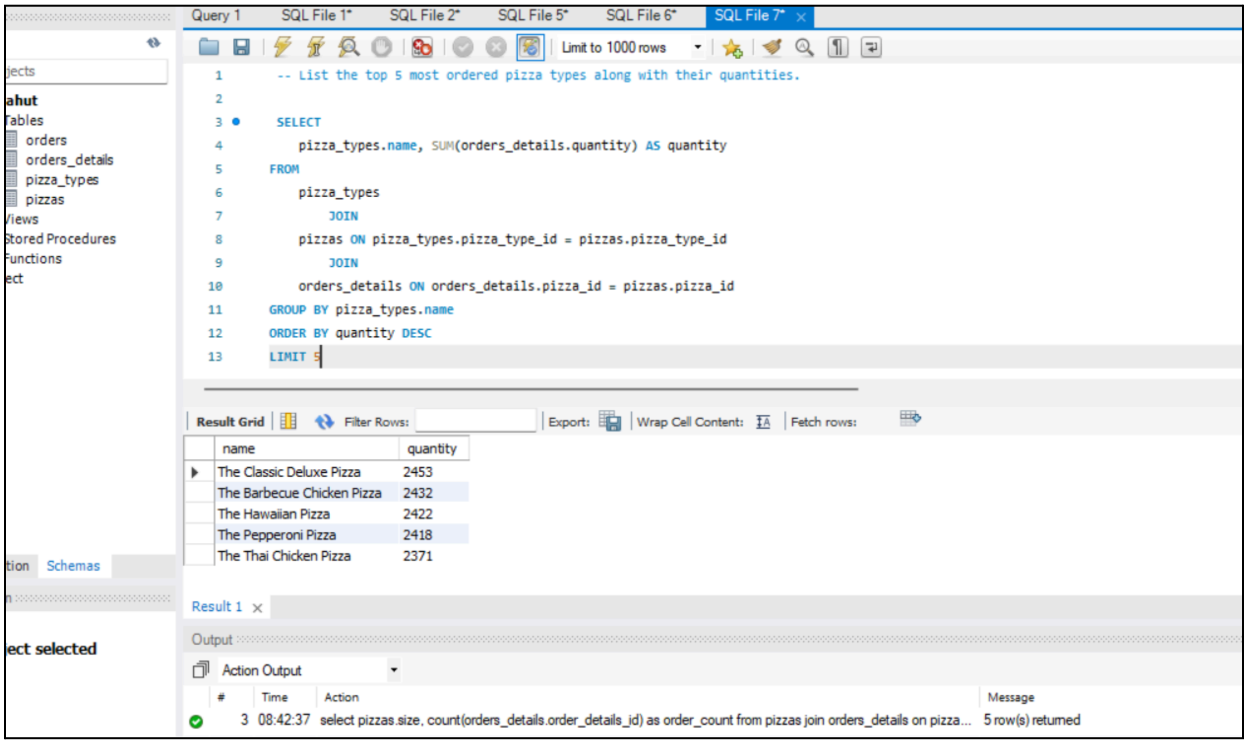


```sql
-- Identify the most common pizza size ordered.

SELECT
    pizzas.size,
    COUNT(orders_details.order_details_id) AS order_count
FROM
    pizzas
        JOIN
    orders_details ON pizzas.pizza_id = orders_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

#	Time	Action	Message
1	08:04:25	select round (sum(orders_details.quantity * pizzas.price),2) as total_sales from orders_details join pizzas on pizza...	1 row(s) returned
2	08:10:30	select pizza_types.name , pizzas.price from pizza_types join pizzas on pizza_types.pizza_type_id = pizzas.pizza_t...	1 row(s) returned
3	08:42:37	select pizzas.size, count(orders_details.order_details_id) as order_count from pizzas join orders_details on pizzas....	5 row(s) returned

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

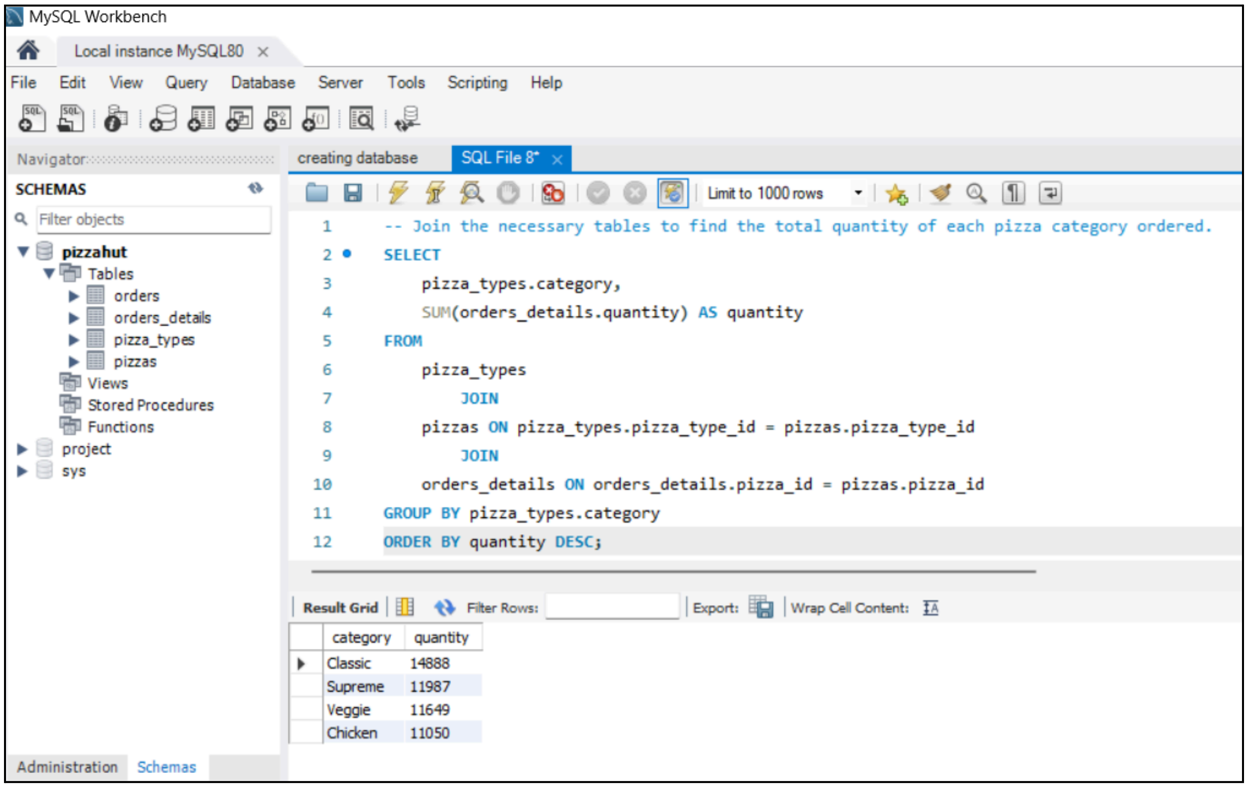


```sql
-- List the top 5 most ordered pizza types along with their quantities.

SELECT
    pizza_types.name, SUM(orders_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    orders_details ON orders_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

#	Time	Action	Message
3	08:42:37	select pizzas.size, count(orders_details.order_details_id) as order_count from pizzas join orders_details on pizza...	5 row(s) returned

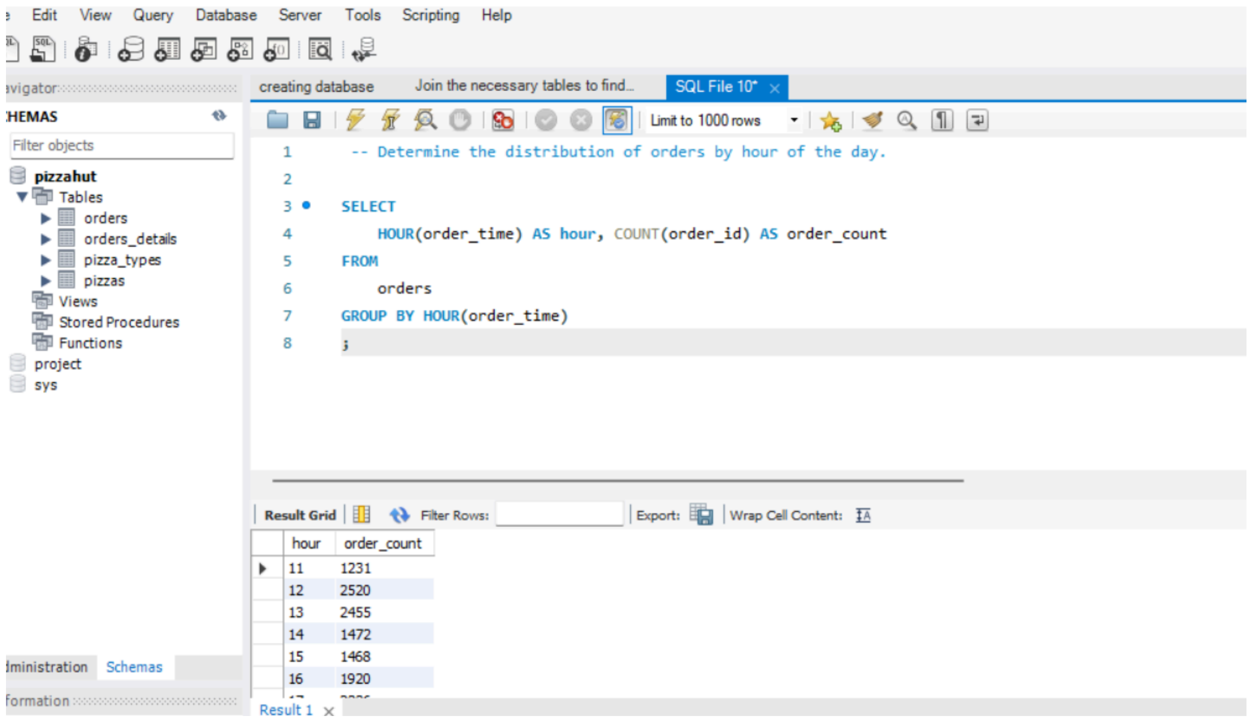
6. Join the necessary tables to find the total quantity of each pizza category ordered.



```sql
-- Join the necessary tables to find the total quantity of each pizza category ordered.
SELECT
    pizza_types.category,
    SUM(orders_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    orders_details ON orders_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

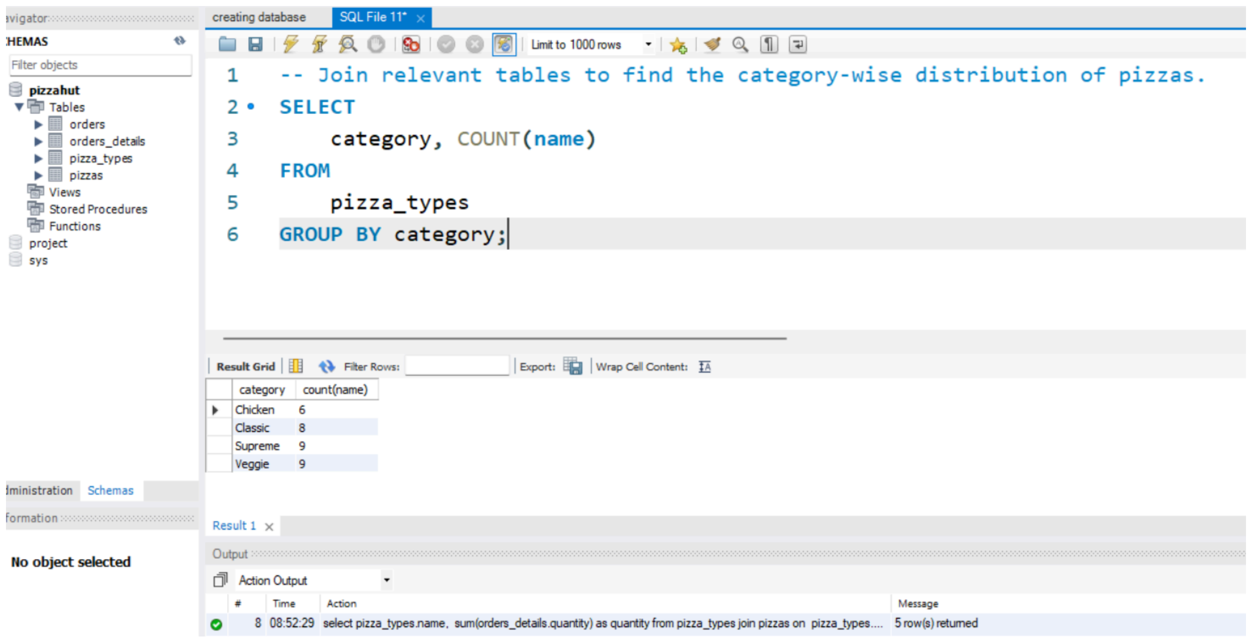
7. Determine the distribution of orders by hour of the day.



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

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

```
hour  order_count
11    1231
12    2520
13    2455
14    1472
15    1468
16    1920
```

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

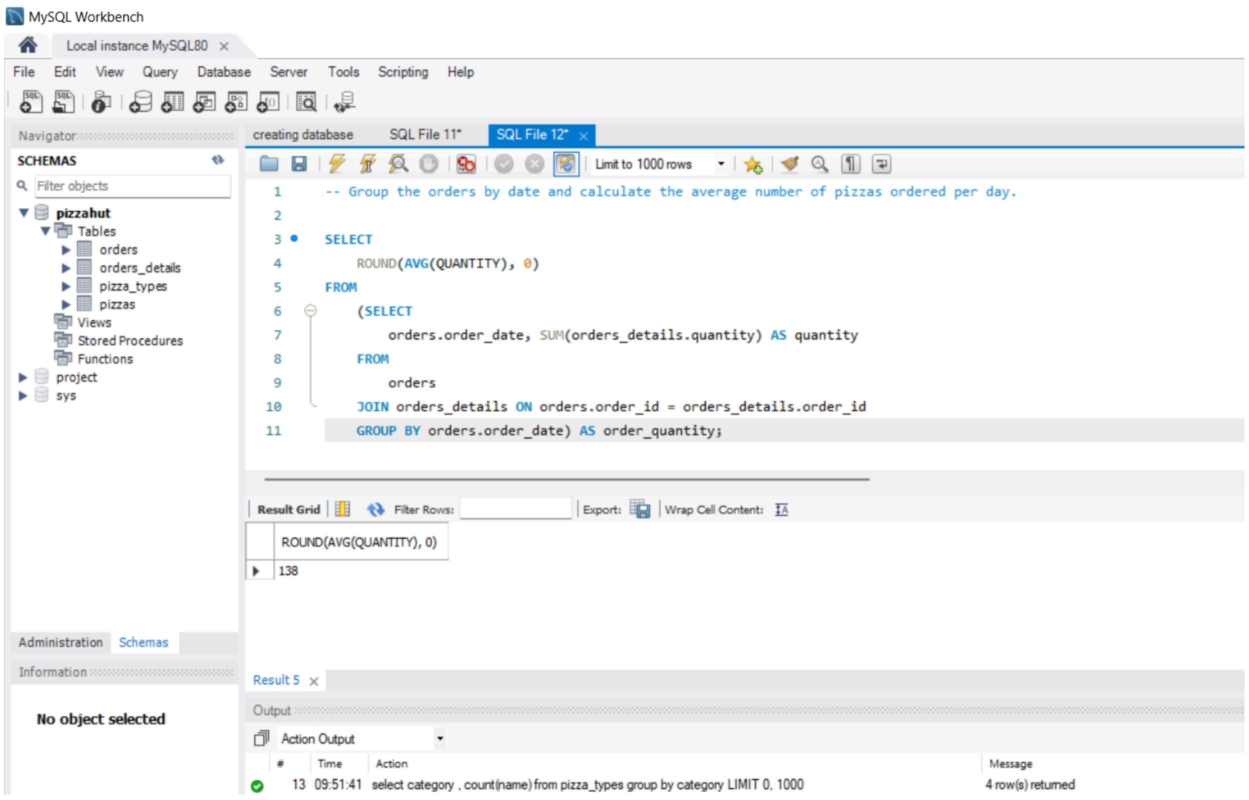


```
-- Join relevant tables to find the category-wise distribution of pizzas.
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```

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

```
#  Time      Action                                                                                                   Message
8  08:52:29  select pizza_types.name, sum(orders_details.quantity) as quantity from pizza_types join pizzas on pizza_types....  5 row(s) returned
```

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

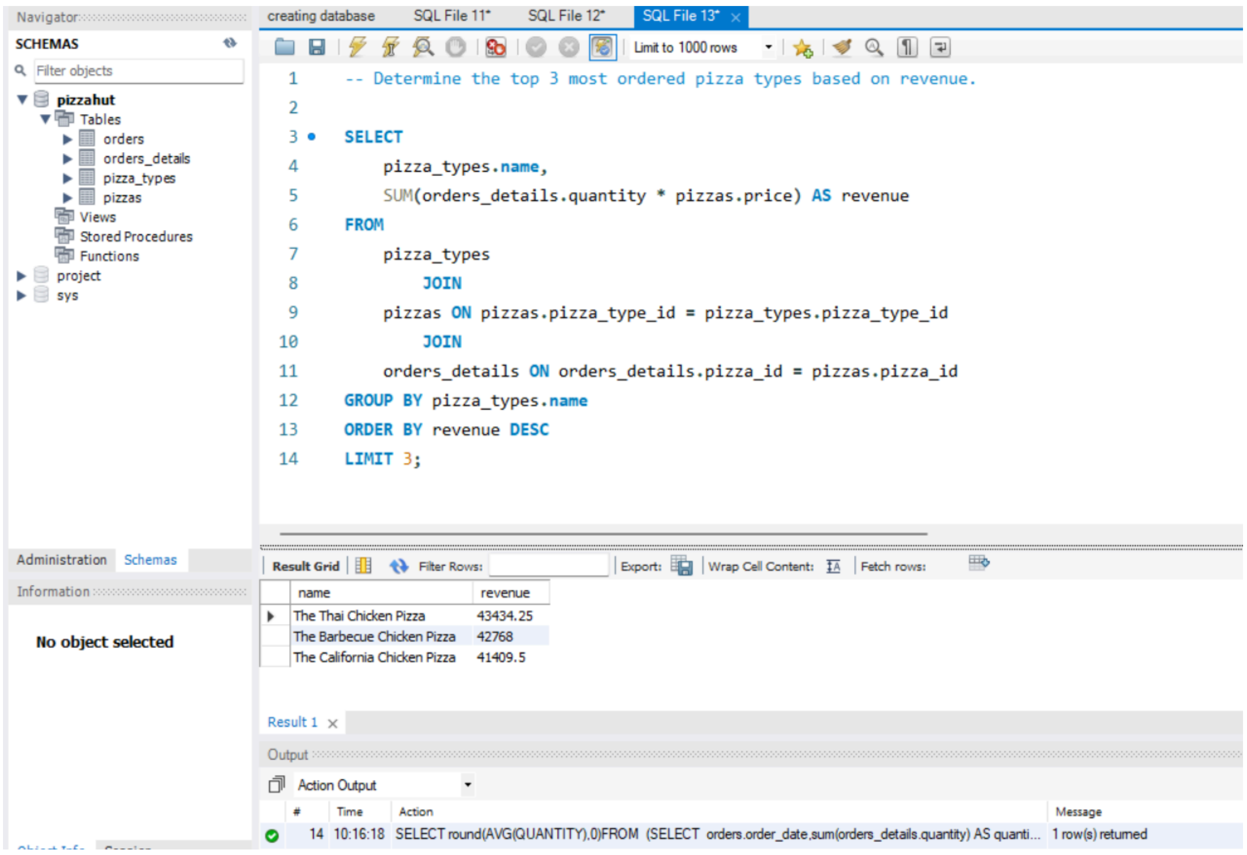


```
-- Group the orders by date and calculate the average number of pizzas ordered per day.

SELECT
    ROUND(AVG(QUANTITY), 0)
FROM
    (SELECT
        orders.order_date, SUM(orders_details.quantity) AS quantity
    FROM
        orders
    JOIN orders_details ON orders.order_id = orders_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

```
ROUND(AVG(QUANTITY), 0)
138
```

```
#   Time      Action                                                                    Message
13  09:51:41  select category , count(name) from pizza_types group by category LIMIT 0, 1000   4 row(s) returned
```

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



Navigator
SCHEMAS
pizzahut
Tables
orders
orders_details
pizza_types
pizzas
Views
Stored Procedures
Functions
project
sys

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

SELECT
    pizza_types.name,
    SUM(orders_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3;
```

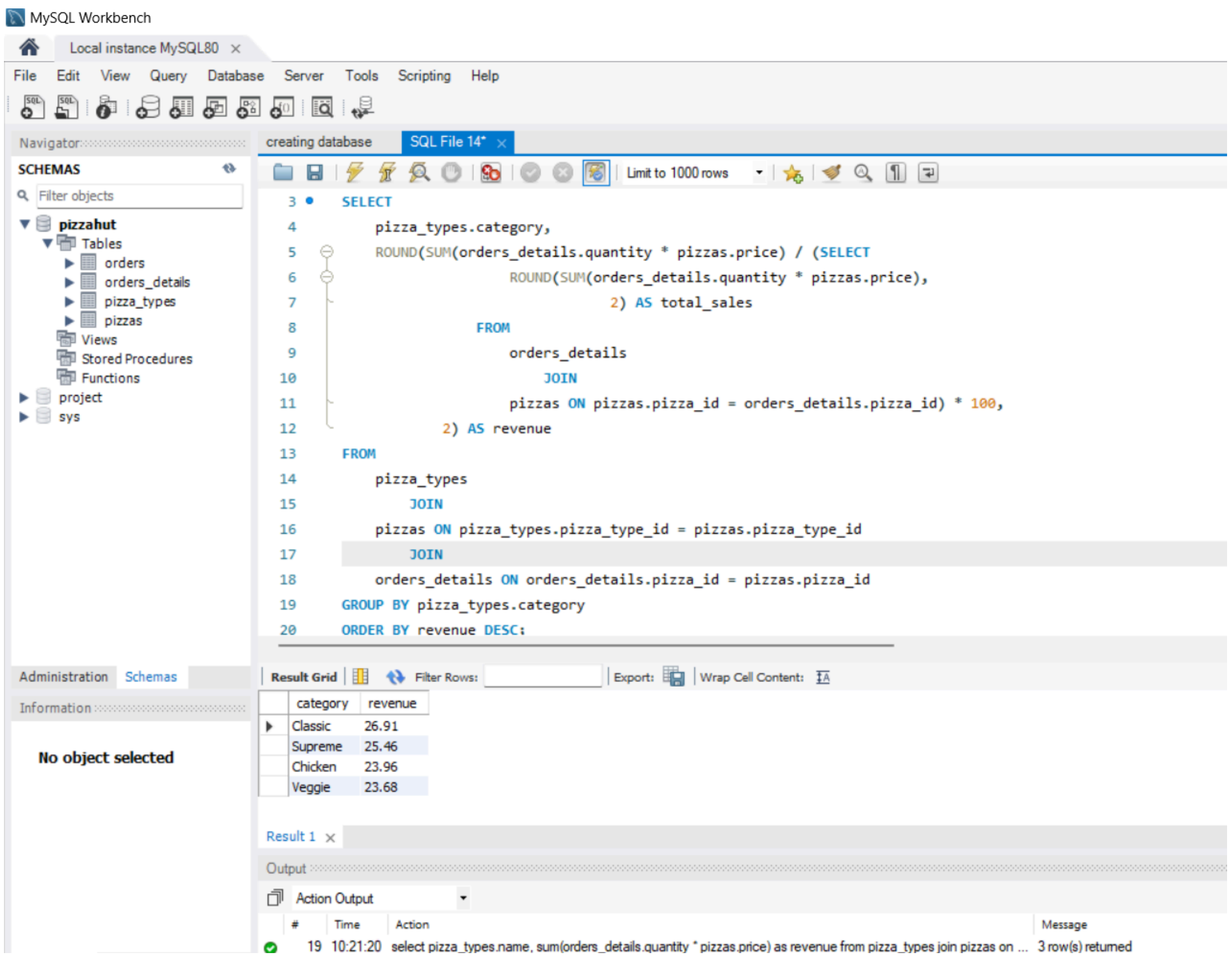
name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

Action Output

#	Time	Action	Message
14	10:16:18	SELECT round(AVG(QUANTITY),0)FROM (SELECT orders.order_date,sum(orders_details.quantity) AS quanti...	1 row(s) returned

No object selected

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

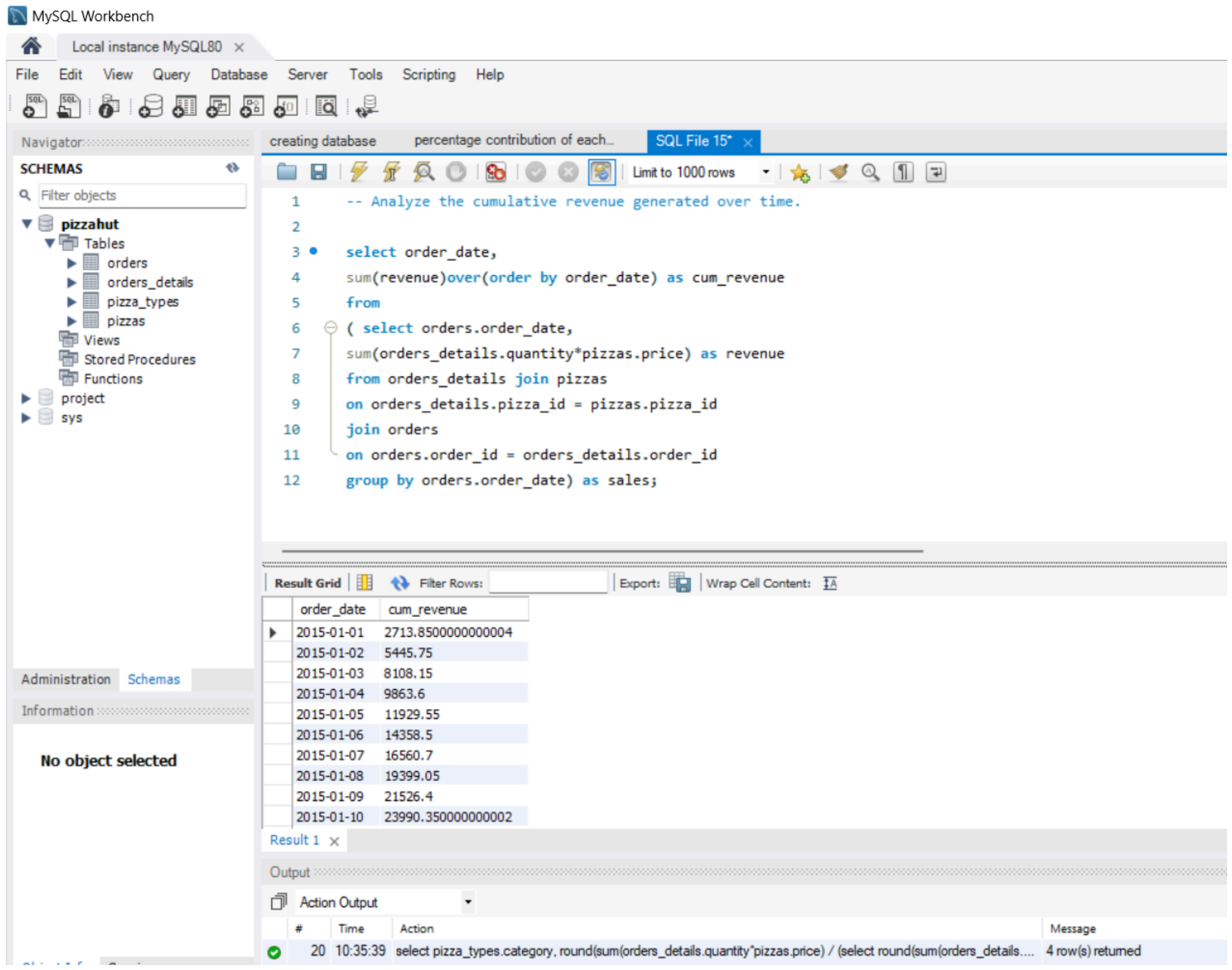


```sql
SELECT
    pizza_types.category,
    ROUND(SUM(orders_details.quantity * pizzas.price) / (SELECT
                    ROUND(SUM(orders_details.quantity * pizzas.price),
                                2) AS total_sales
                FROM
                    orders_details
                        JOIN
                    pizzas ON pizzas.pizza_id = orders_details.pizza_id) * 100,
            2) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    orders_details ON orders_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

```
19  10:21:20  select pizza_types.name, sum(orders_details.quantity * pizzas.price) as revenue from pizza_types join pizzas on ...  3 row(s) returned
```

12. Analyze the cumulative revenue generated over time.

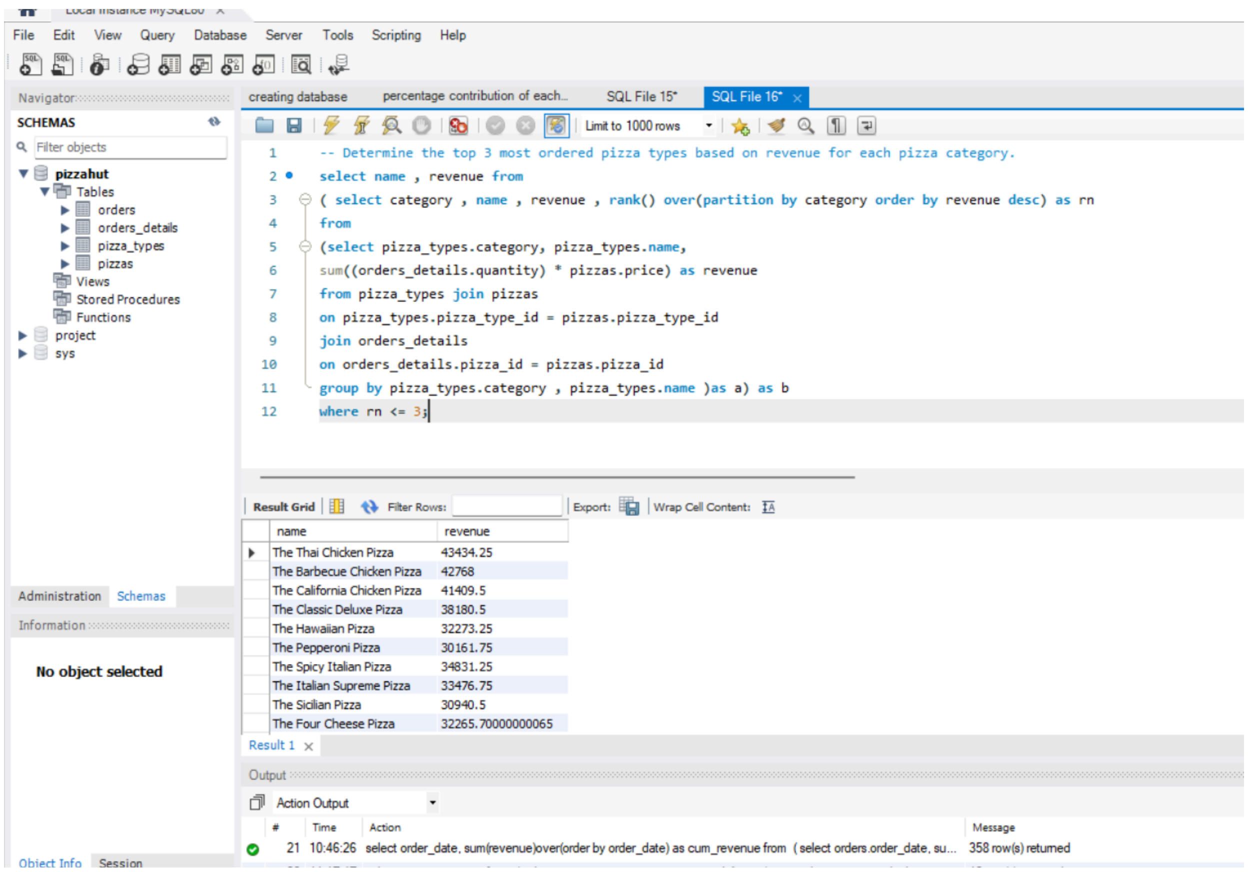


```sql
-- Analyze the cumulative revenue generated over time.

select order_date,
sum(revenue)over(order by order_date) as cum_revenue
from
( select orders.order_date,
sum(orders_details.quantity*pizzas.price) as revenue
from orders_details join pizzas
on orders_details.pizza_id = pizzas.pizza_id
join orders
on orders.order_id = orders_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
2015-01-06	14358.5
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526.4
2015-01-10	23990.350000000002

#	Time	Action	Message
20	10:35:39	select pizza_types.category, round(sum(orders_details.quantity*pizzas.price) / (select round(sum(orders_details....	4 row(s) returned

13. 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((orders_details.quantity) * pizzas.price) as revenue
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
join orders_details
on orders_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
The Sicilian Pizza	30940.5
The Four Cheese Pizza	32265.70000000065

```
#   Time      Action                                                                                              Message
21  10:46:26  select order_date, sum(revenue)over(order by order_date) as cum_revenue from  ( select orders.order_date, su...   358 row(s) returned
```

THANK YOU