DATA ANALYSIS OF SALES IN

PIZZA HUT

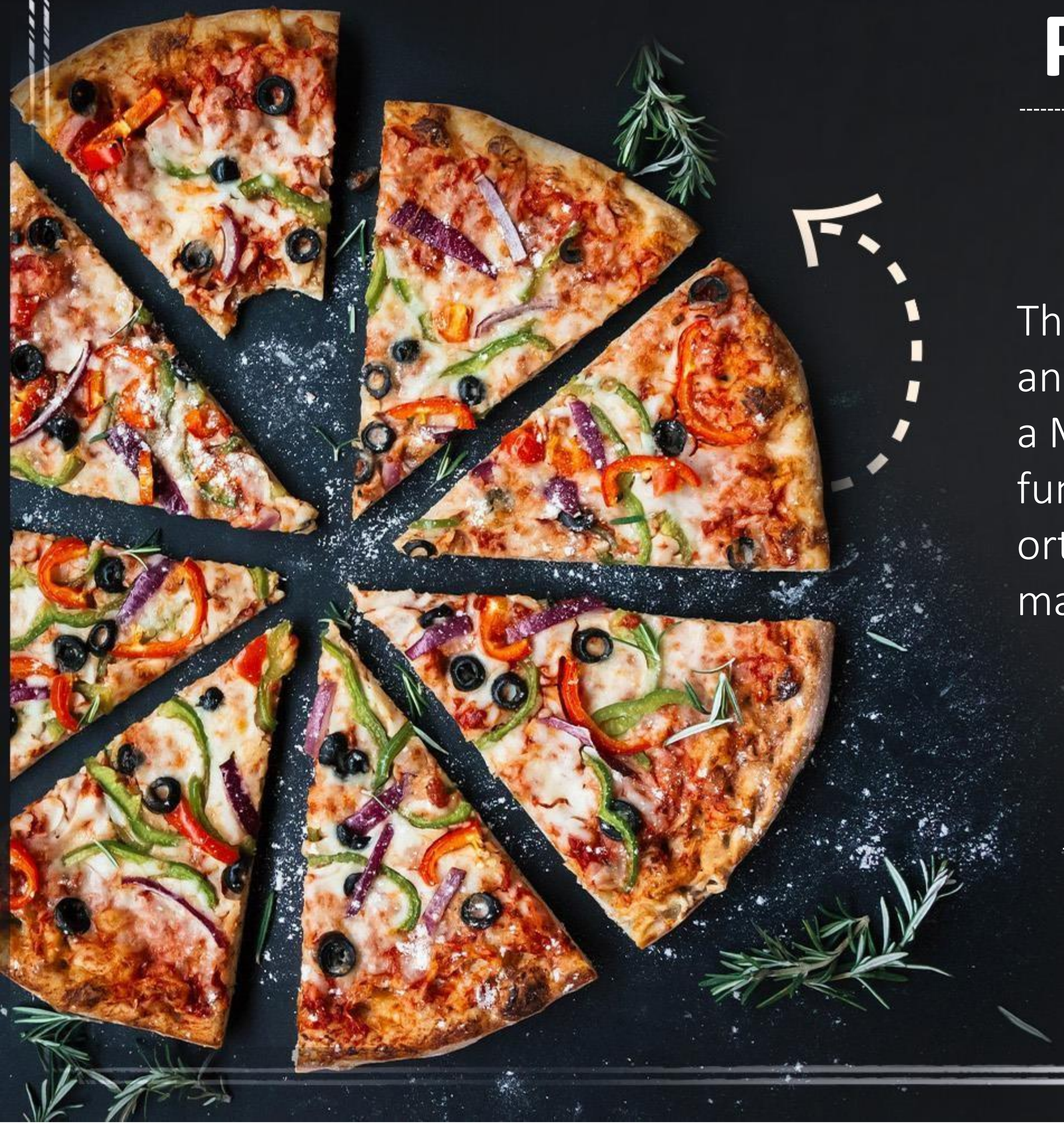
--by using SQL

A PROJECT REPORT BY AMBARISH YADAV

PROJECT OVERVIEW-

This project involves extracting, transforming, and analyzing the pizza sales data from various tables in a MySQL database. By employing SQL joins and functions, and aim to generate comprehensive rep orts and visualizations that will help Pizzahut Brand make data-driven decisions.

A PROJECT REPORT BY AMBARISH YADAV



PROJECT OBJECTIVE-

To analyze the sales data of PizzaHut Brand to unco ver insights into sales performance, customer prefe rences, and operational efficiency using MySQL join s and functions to solve some problem statements that they want from us.

A PROJECT REPORT BY AMBARISH YADAV

TABLES NAME

These are the tables we have in our database.

01: orders

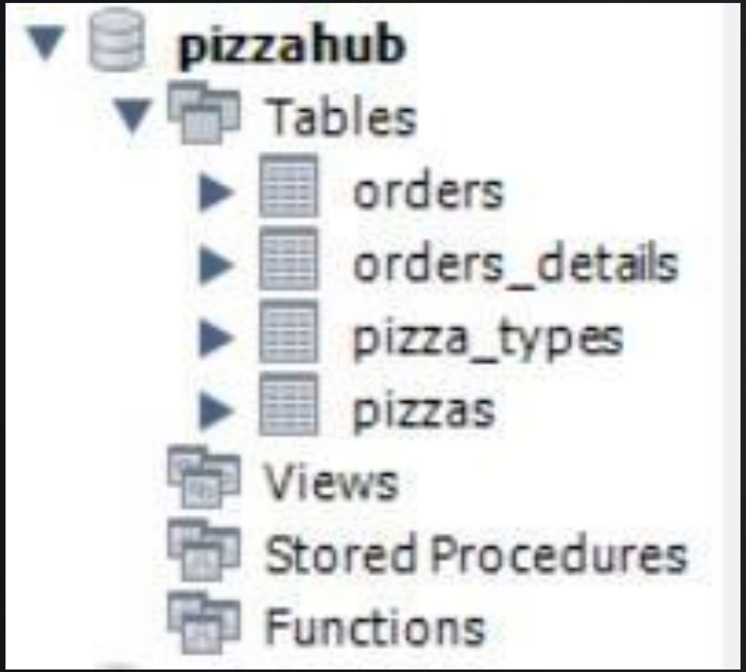
02: orders_details

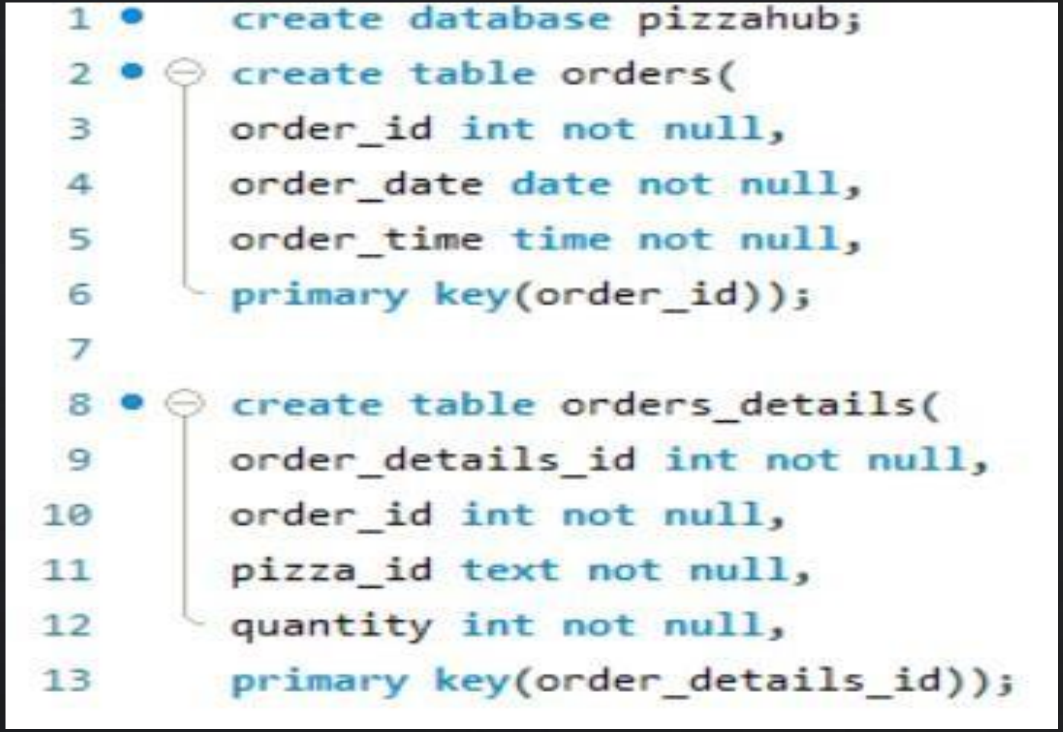
03: pizza_types

04: pizzas

A PROJECT REPORT BY AMBARISH YADAV

BASIC STRUCTURE AND SCHEMA:

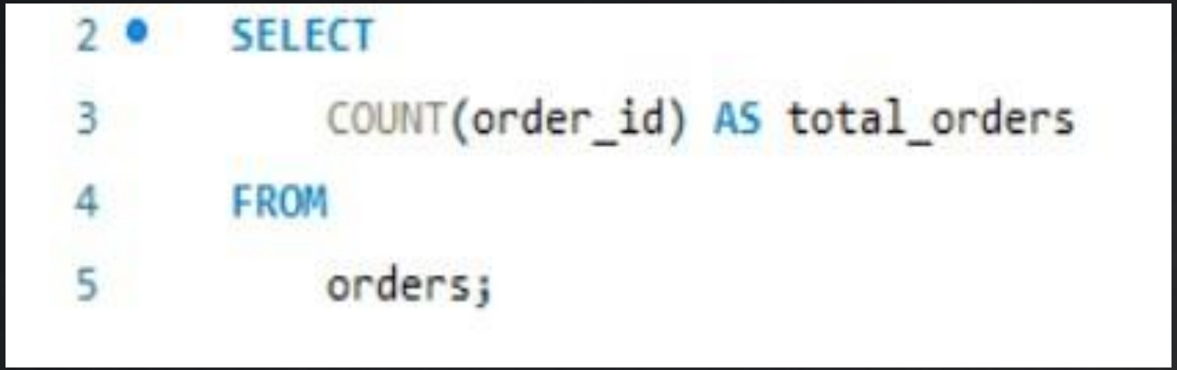




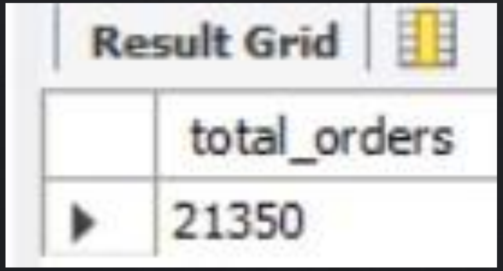
```
create database pizzahub;
create table orders(
order_id int not null,
order_date date not null,
order_time time not null,
primary key(order_id));

create table orders_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));
```

QUESTION 1: -- Retrieve the total number of orders placed.



```
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```



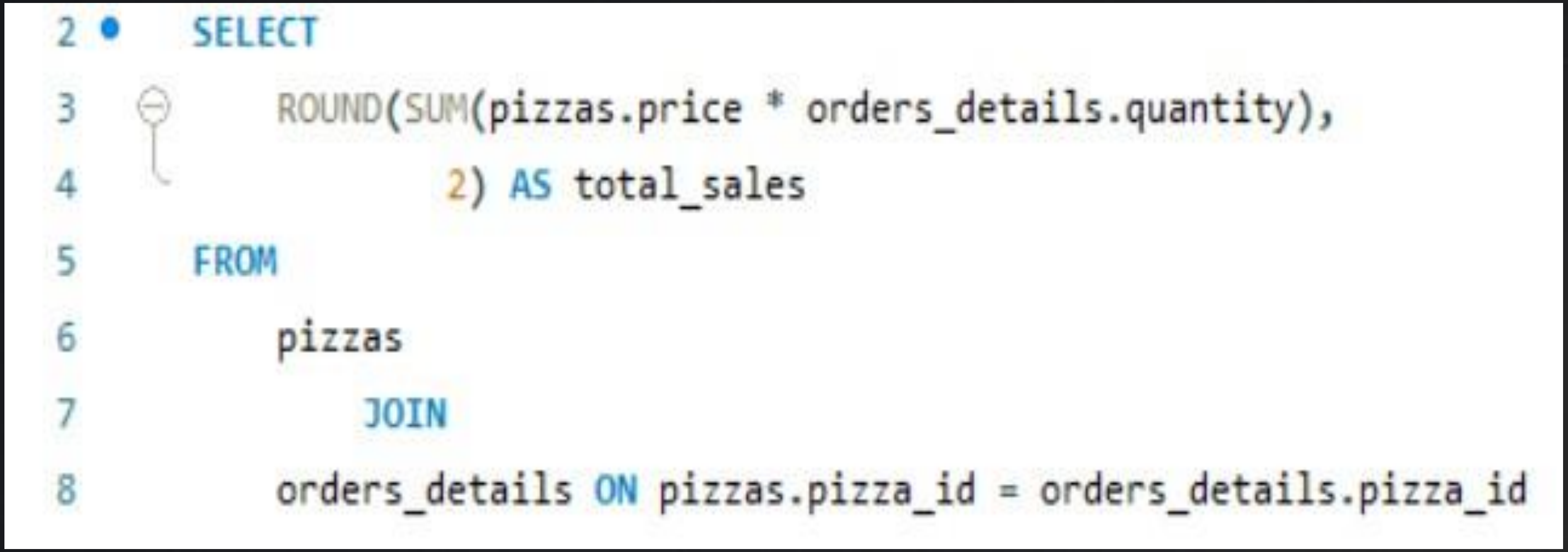
Result Grid

	total_orders
▶	21350

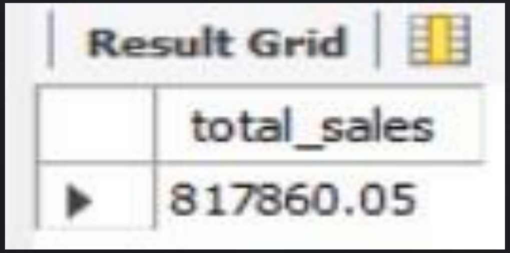
A PROJECT REPORT BY AMBARISH YADAV

QUESTION 2:

-- Calculate the total revenue generated from pizza sales.



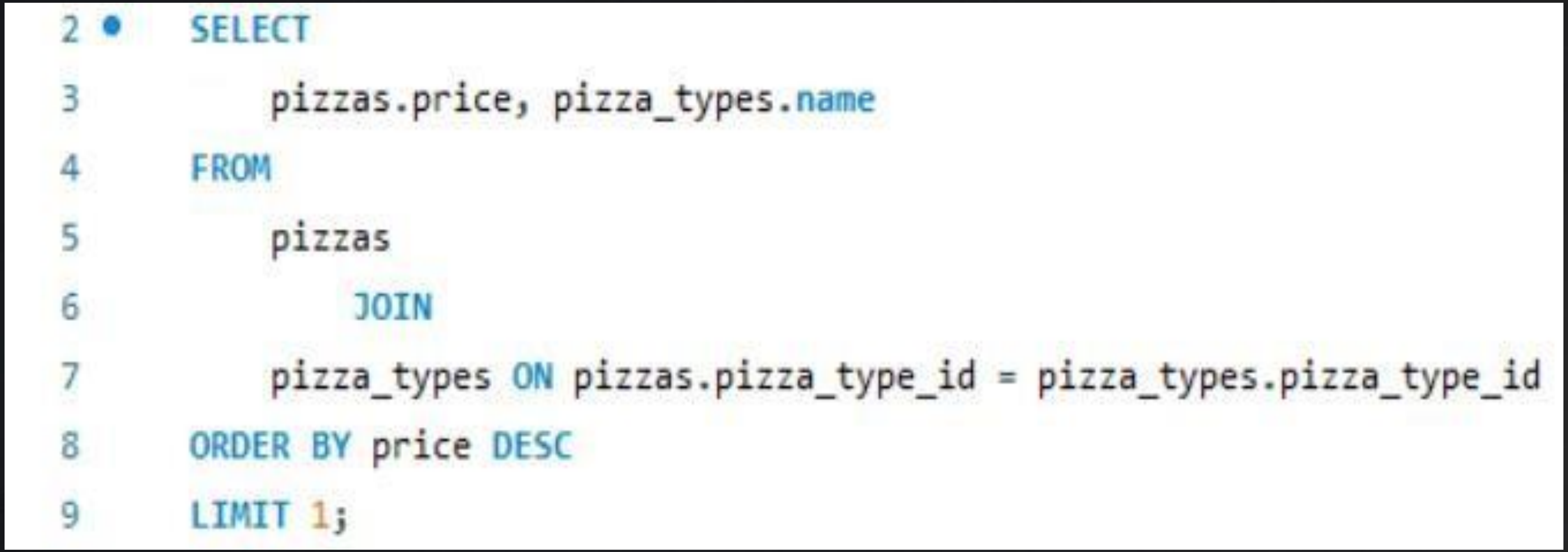
```
SELECT
    ROUND(SUM(pizzas.price * orders_details.quantity),
            2) AS total_sales
FROM
    pizzas
        JOIN
    orders_details ON pizzas.pizza_id = orders_details.pizza_id
```



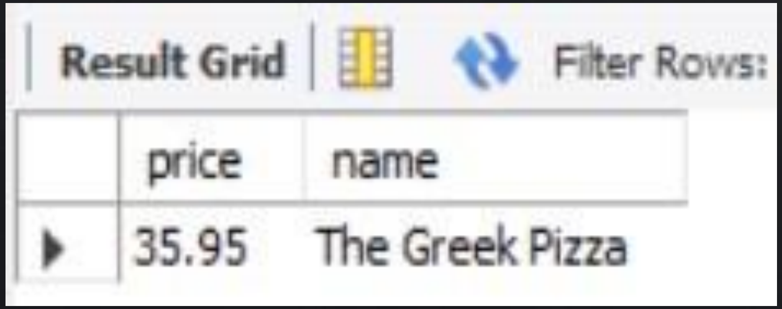
```
total_sales
817860.05
```

QUESTION 3:

-- Identify the highest-priced pizza.



```sql
SELECT
    pizzas.price, pizza_types.name
FROM
    pizzas
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
ORDER BY price DESC
LIMIT 1;
```

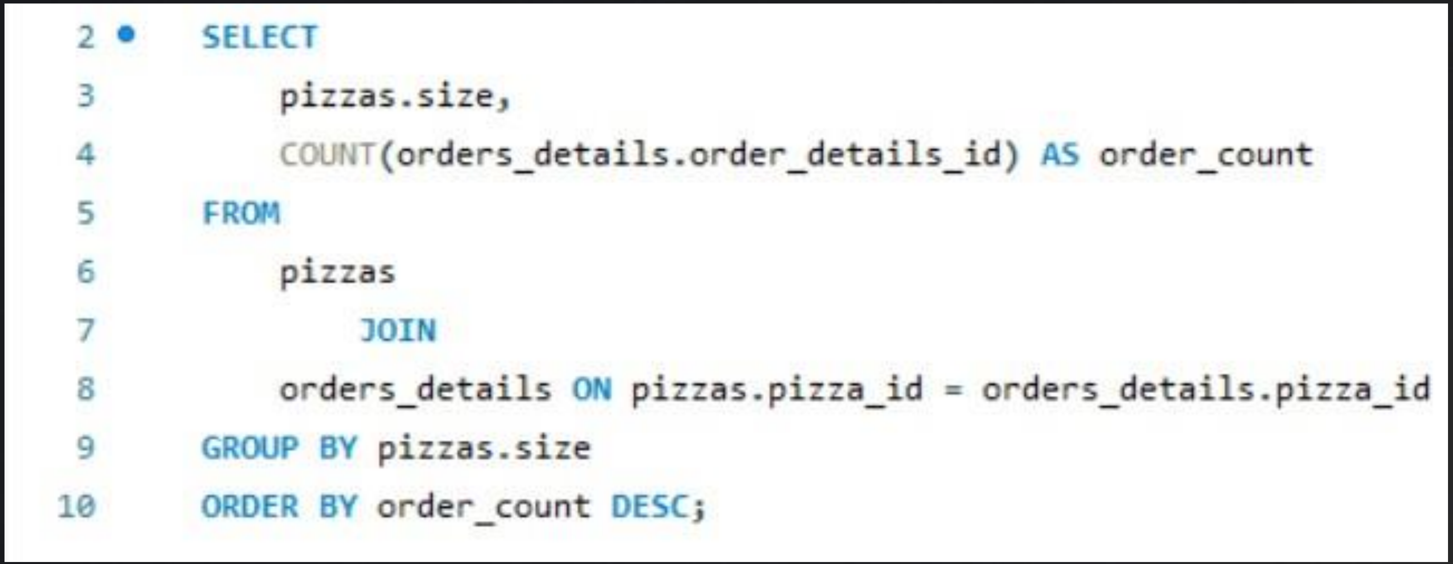


Result Grid | Filter Rows:

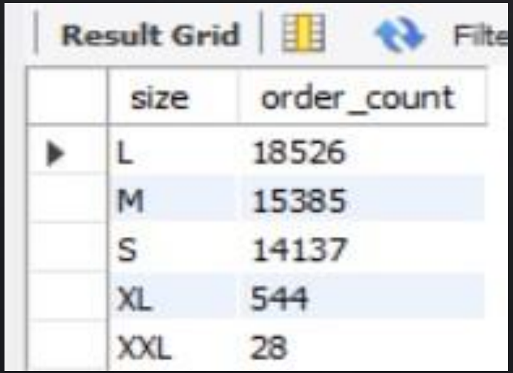
price	name
35.95	The Greek Pizza

QUESTION 4:

-- Identify the most common pizza size ordered.



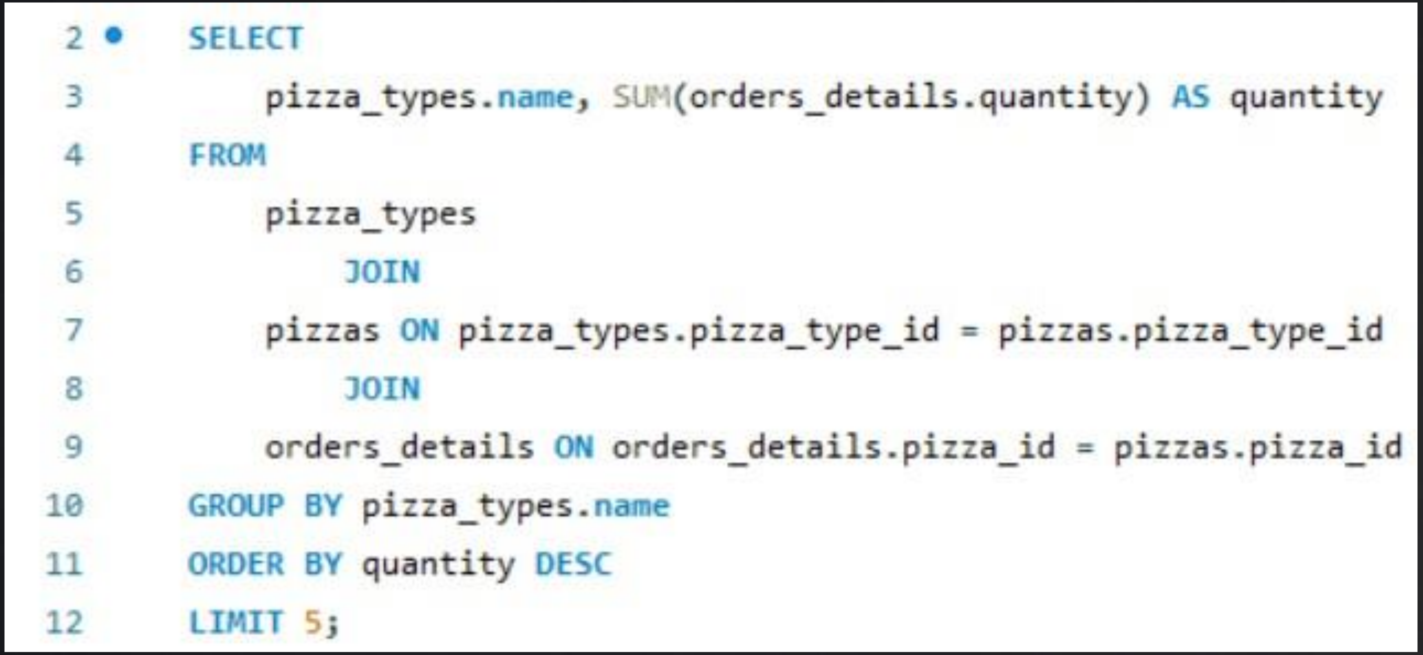
```
SELECT
    pizzas.size,
    COUNT(orders_details.order_details_id) AS order_count
FROM
    pizzas
        JOIN
    orders_details ON pizzas.pizza_id = orders_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC;
```



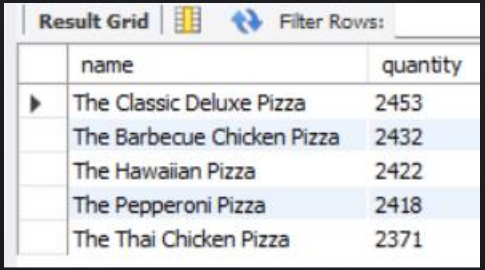
size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

QUESTION 5:

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



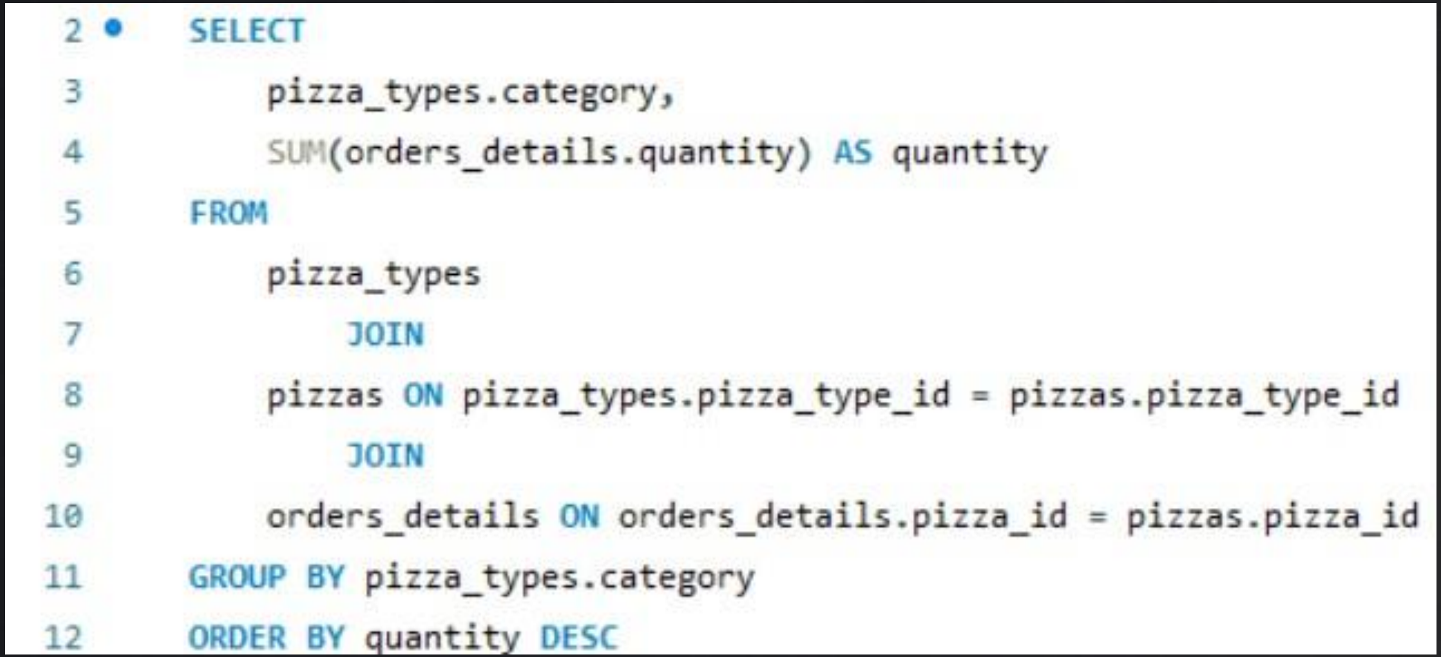
```sql
SELECT
    pizza_types.name, SUM(orders_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC
LIMIT 5;
```



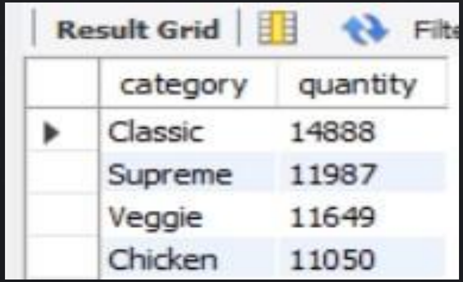
name	quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

QUESTION 6:

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



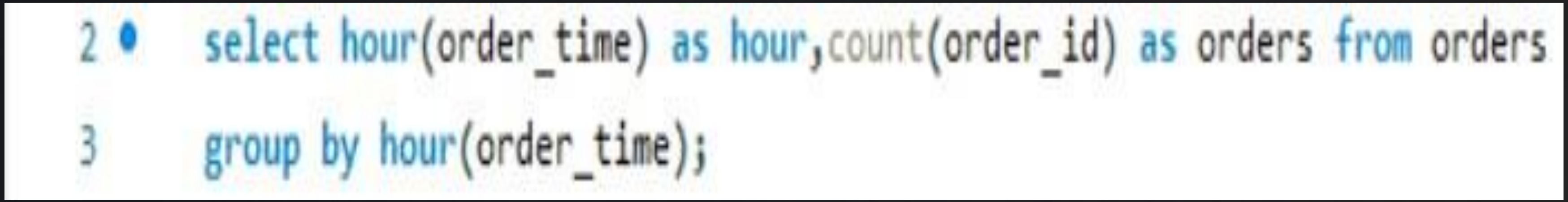
```
SELECT
    pizza_types.category,
    SUM(orders_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY quantity DESC
```



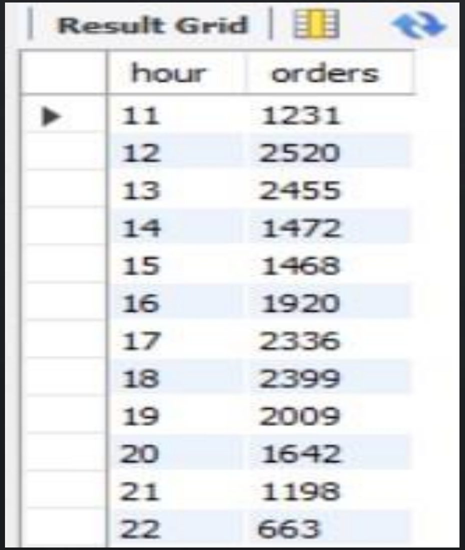
category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

QUESTION 7:

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



```
select hour(order_time) as hour,count(order_id) as orders from orders
group by hour(order_time);
```

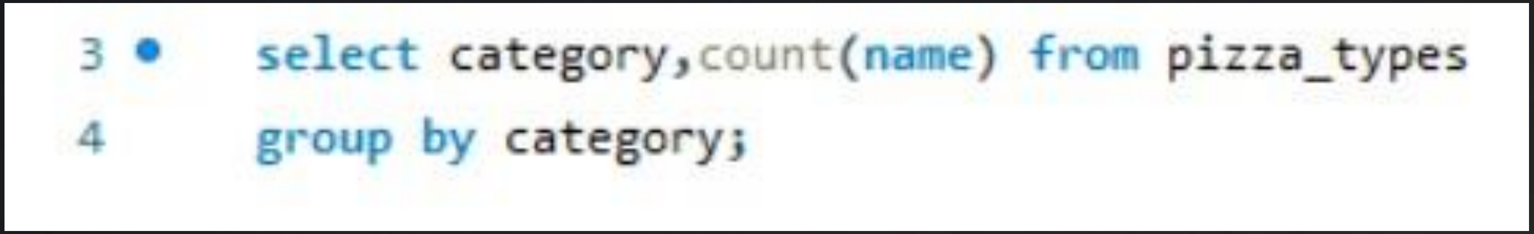


Result Grid

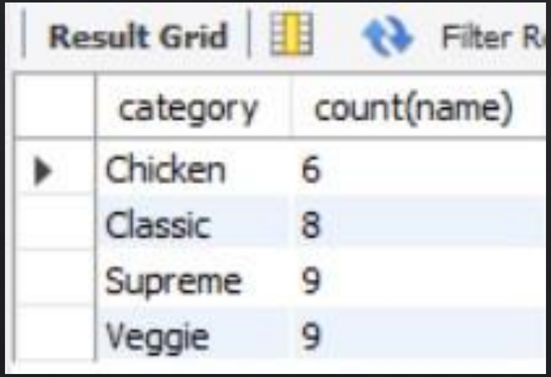
hour	orders
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663

QUESTION 8:

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



```
select category,count(name) from pizza_types
group by category;
```



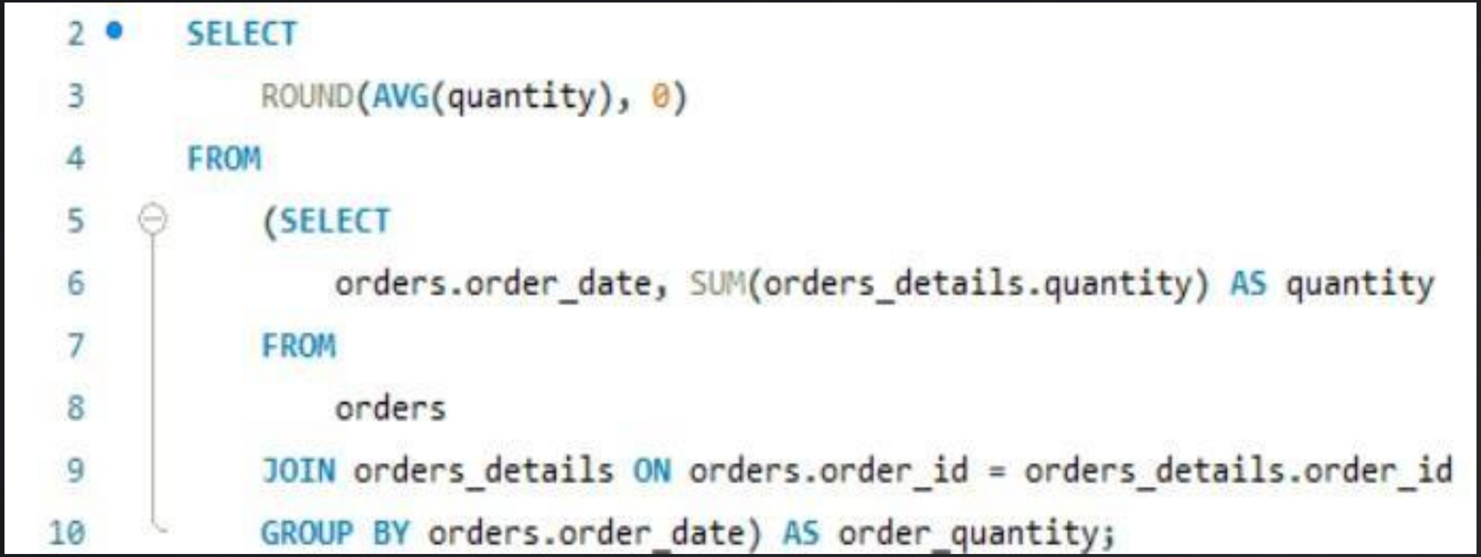
Result Grid | Filter R

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

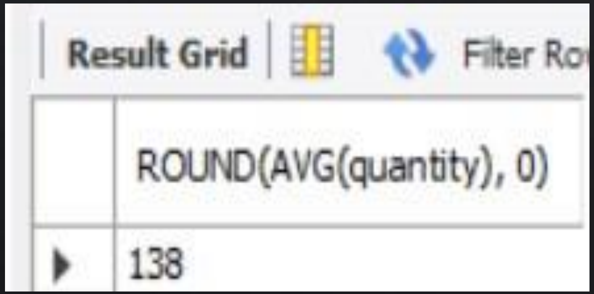
A PROJECT REPORT BY AMBARISH YADAV

QUESTION 9:

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

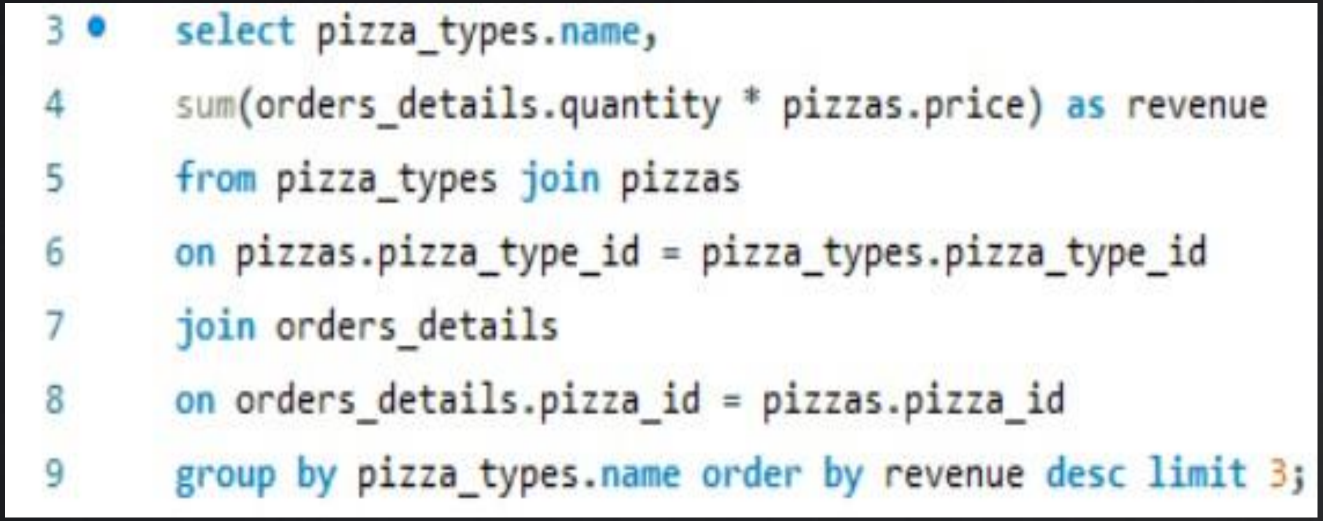


```
SELECT
    ROUND(AVG(quantity), 0)
FROM
    (SELECT
        orders.order_date, SUM(orders_details.quantity) AS quantity
    FROM
        orders
    JOIN orders_details ON orders.order_id = orders_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

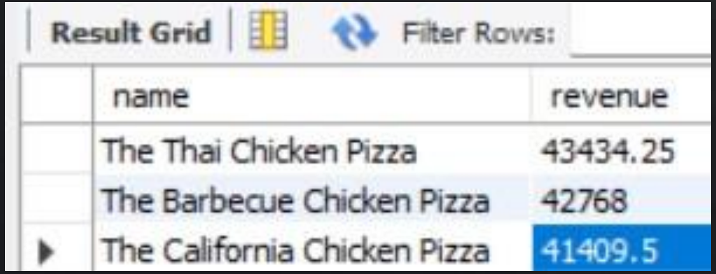


```
ROUND(AVG(quantity), 0)
138
```

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

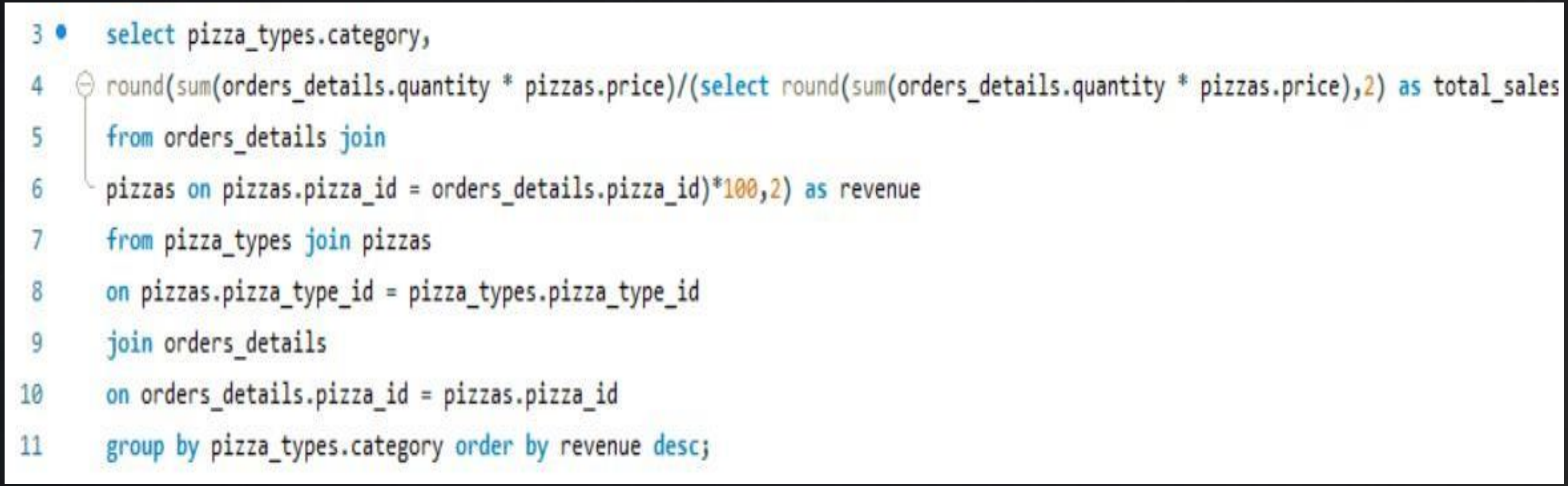


```
select pizza_types.name,
sum(orders_details.quantity * pizzas.price) as revenue
from pizza_types join pizzas
on pizzas.pizza_type_id = pizza_types.pizza_type_id
join orders_details
on orders_details.pizza_id = pizzas.pizza_id
group by pizza_types.name order by revenue desc limit 3;
```

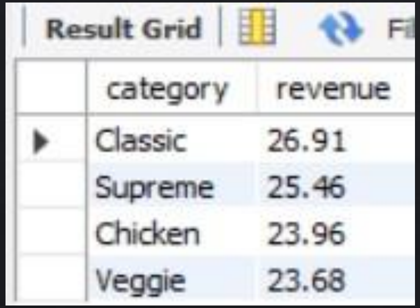


name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

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



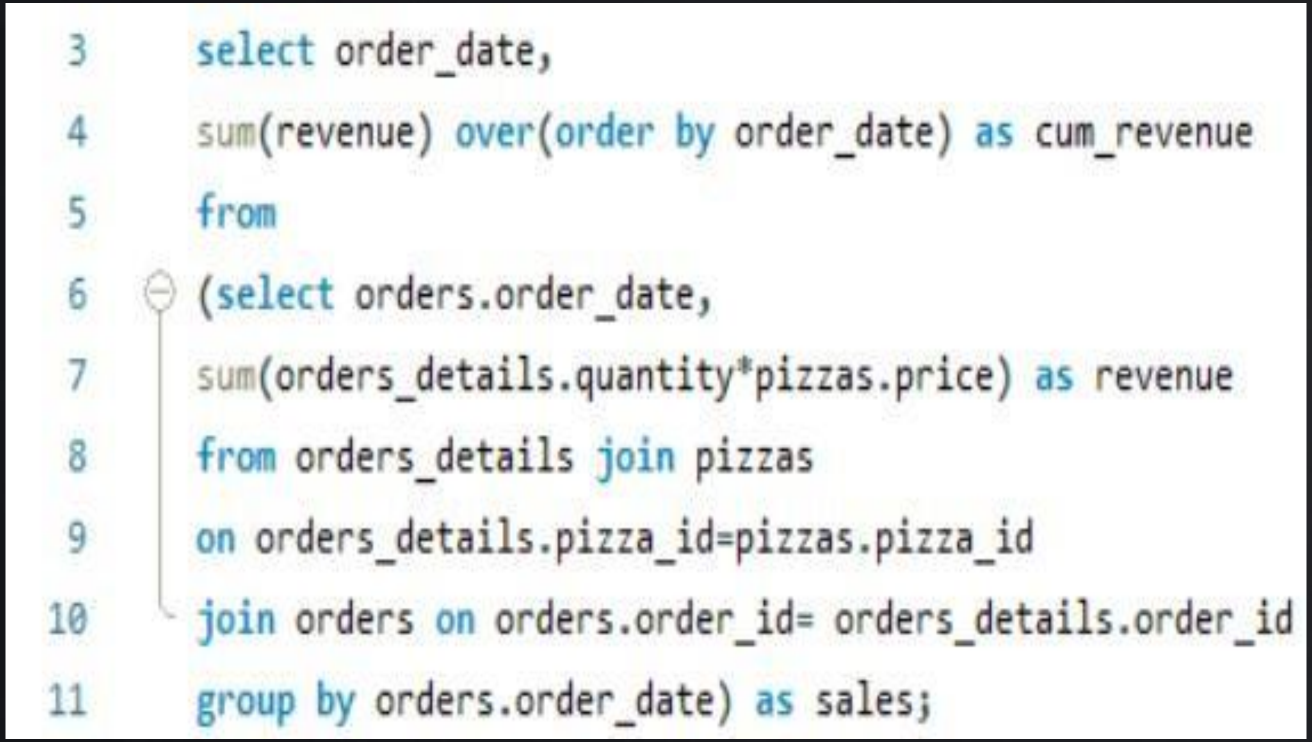
```
select pizza_types.category,
round(sum(orders_details.quantity * pizzas.price)/(select round(sum(orders_details.quantity * pizzas.price),2) as total_sales
from orders_details join
pizzas on pizzas.pizza_id = orders_details.pizza_id)*100,2) as revenue
from pizza_types join pizzas
on pizzas.pizza_type_id = pizza_types.pizza_type_id
join orders_details
on orders_details.pizza_id = pizzas.pizza_id
group by pizza_types.category order by revenue desc;
```



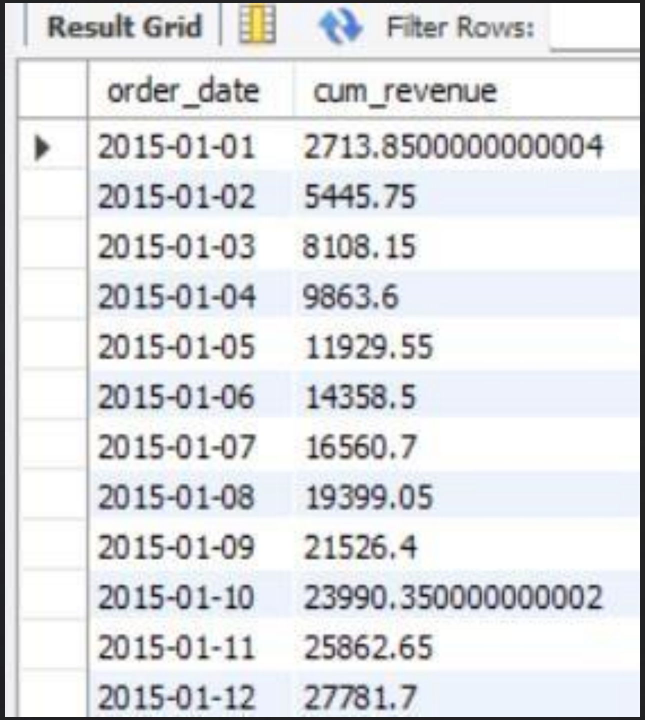
Result Grid

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

QUESTION 12:-- 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
2015-01-11   25862.65
2015-01-12   27781.7
```

THANK YOU

A PROJECT REPORT BY AMBARISH YADAV