Pizza Sales Management System

Bhuvana yelubandi

Agenda

1. Aim
2. Data overview
3. schema
4. Queries execution
5. conclusion

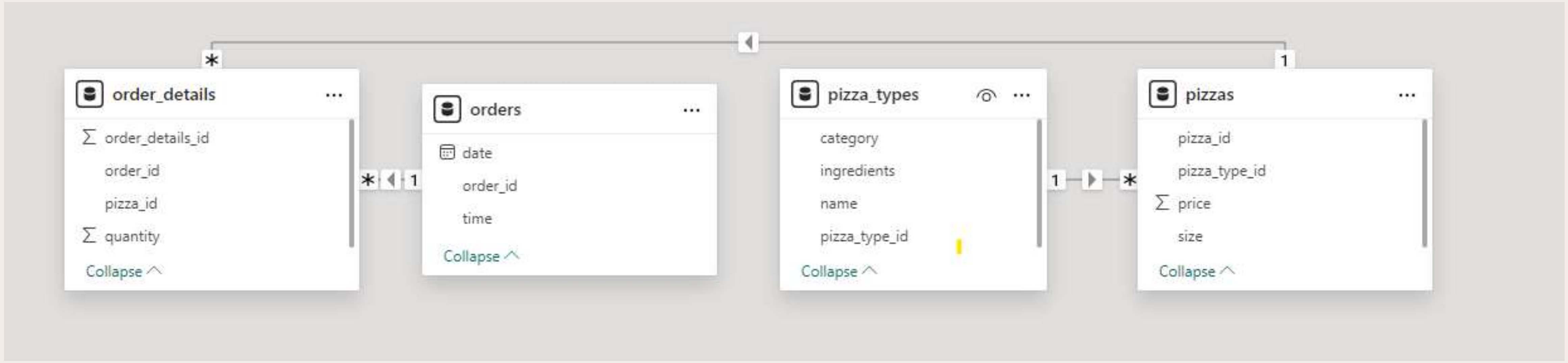
AIM

The "Pizza Sales Management System" project aims to develop a database-driven application to manage and analyze sales data for a pizza restaurant chain. Using MySQL, the system will store and manipulate data related to pizza orders, revenues, profits, and other relevant details to provide valuable insights for business decision-making.

Data overview

•**Database Overview:**
Name: Dominos
Purpose: To manage and analyze pizza orders, details, and types within the Domino's Pizza chain.
Tables in the Database:
a. orders:
Key Fields:
order_id
order_date
order_time
b. orders_details:
Key Fields:
order_details_id
order_id
pizza_id
quantity
c. pizzas:
Key Fields:
pizza_type_id
name
category
ingredients
d. pizza_types:
Key Fields:
pizza_id
pizza_type_id
size
price

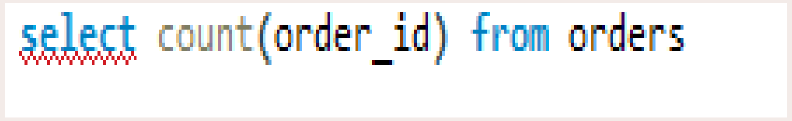
Schema



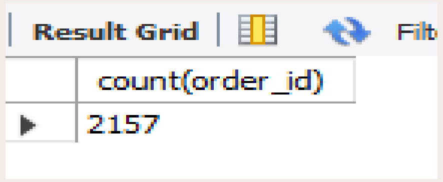
order_details
order_details_id
order_id
pizza_id
quantity
Collapse
orders
date
order_id
time
Collapse
pizza_types
category
ingredients
name
pizza_type_id
Collapse
pizzas
pizza_id
pizza_type_id
price
size
Collapse
*
1
*
1
1
*

Queries Execution

1. Retrieve the total number of orders placed

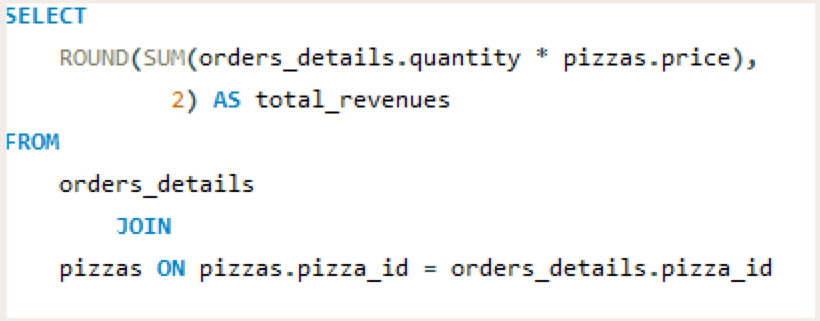


```
select count(order_id) from orders
```

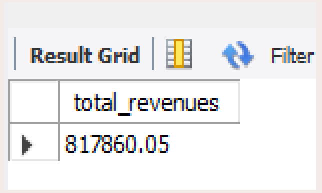


```
count(order_id)
2157
```

2. Calculate the total revenue generated from pizza sales

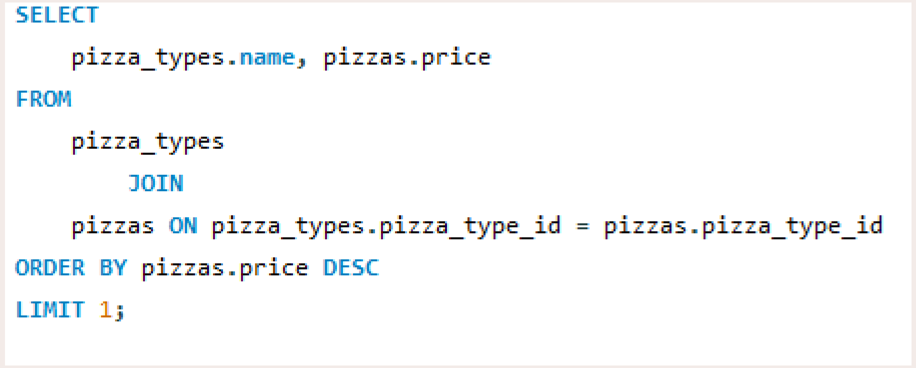


```
SELECT
    ROUND(SUM(orders_details.quantity * pizzas.price),
            2) AS total_revenues
FROM
    orders_details
        JOIN
    pizzas ON pizzas.pizza_id = orders_details.pizza_id
```

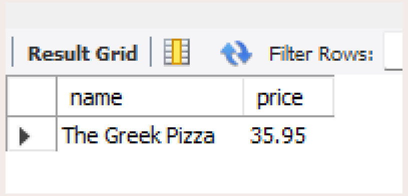


```
Result Grid    Filter
   total_revenues
▶  817860.05
```

3. Identify the highest-priced pizza.

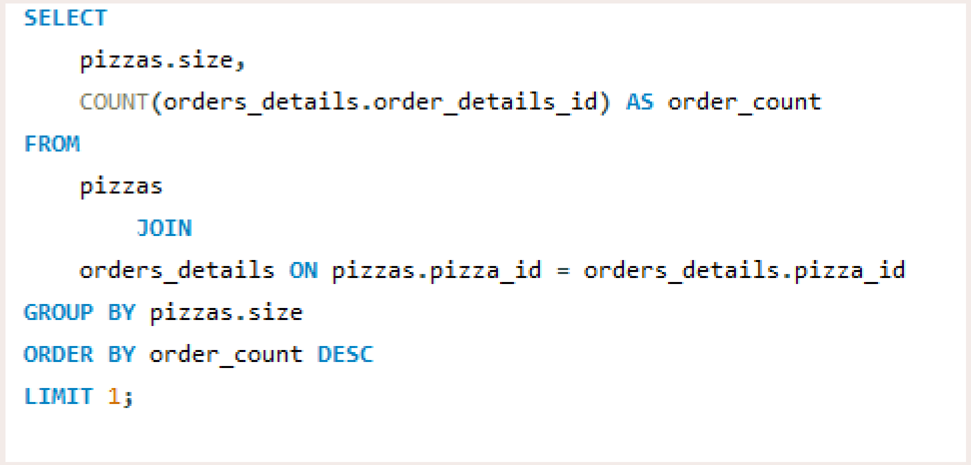


```
SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```

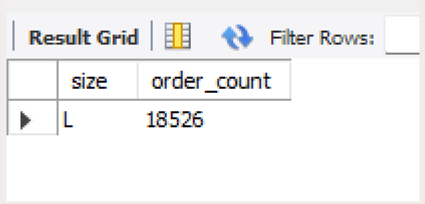


```
  name              price
▶ The Greek Pizza   35.95
```

4. Identify the most common pizza size ordered.

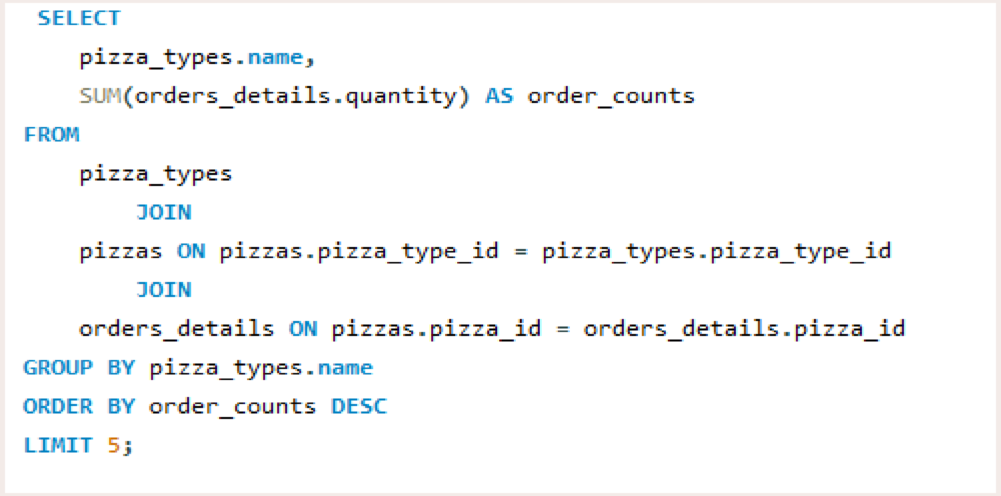


```
SELECT
    pizzas.size,
    COUNT(orders_details.order_details_id) AS order_count
FROM
    pizzas
        JOIN
    orders_details ON pizzas.pizza_id = orders_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC
LIMIT 1;
```

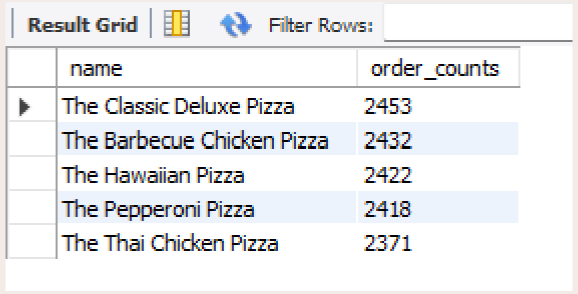


```
size   order_count
L      18526
```

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



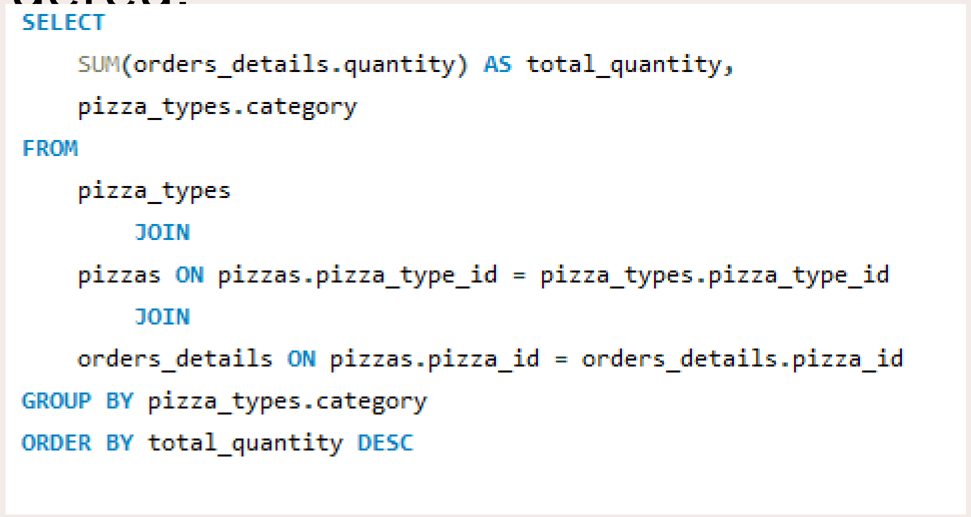
```
 SELECT
    pizza_types.name,
    SUM(orders_details.quantity) AS order_counts
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    orders_details ON pizzas.pizza_id = orders_details.pizza_id
GROUP BY pizza_types.name
ORDER BY order_counts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



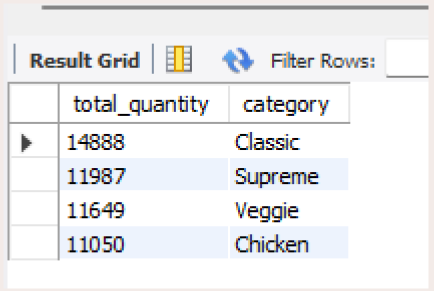
Result Grid | Filter Rows:

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

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

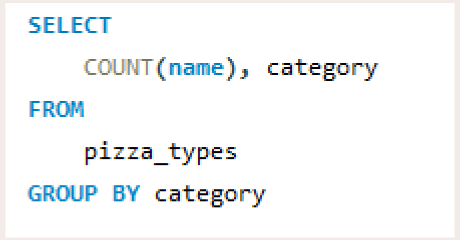


```sql
SELECT
    SUM(orders_details.quantity) AS total_quantity,
    pizza_types.category
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    orders_details ON pizzas.pizza_id = orders_details.pizza_id
GROUP BY pizza_types.category
ORDER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

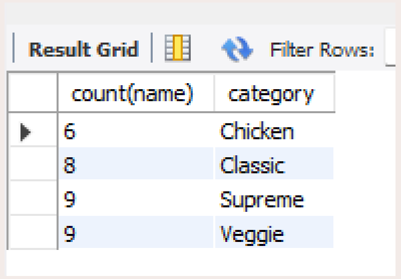


total_quantity	category
14888	Classic
11987	Supreme
11649	Veggie
11050	Chicken

7.Retrieve the total number of orders placed

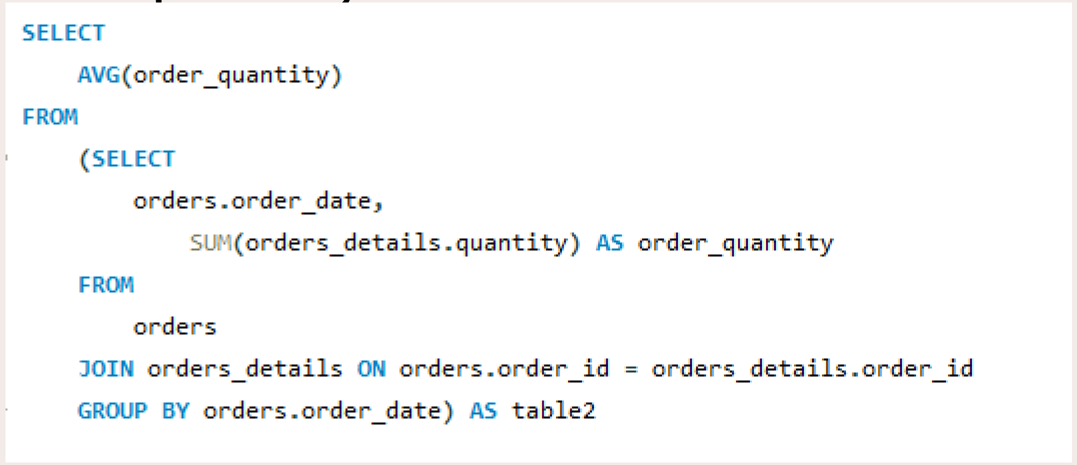


```
SELECT
    COUNT(name), category
FROM
    pizza_types
GROUP BY category
```

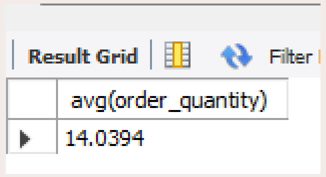


```
count(name)  category
6            Chicken
8            Classic
9            Supreme
9            Veggie
```

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

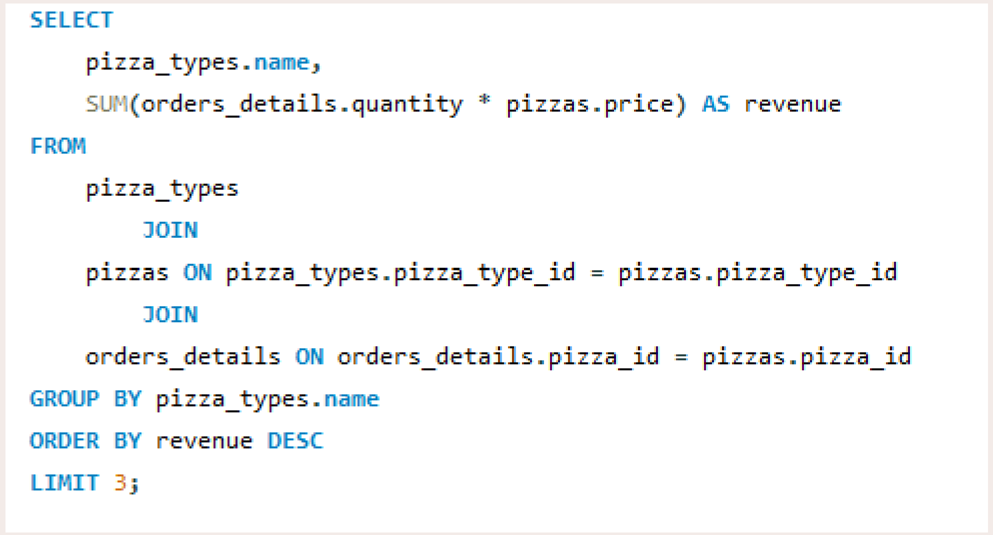


```sql
SELECT
    AVG(order_quantity)
FROM
    (SELECT
        orders.order_date,
            SUM(orders_details.quantity) AS order_quantity
    FROM
        orders
    JOIN orders_details ON orders.order_id = orders_details.order_id
    GROUP BY orders.order_date) AS table2
```

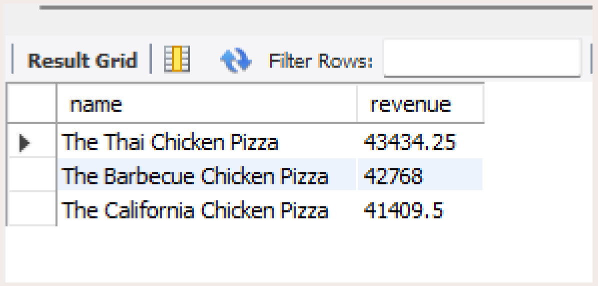


```
avg(order_quantity)
14.0394
```

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

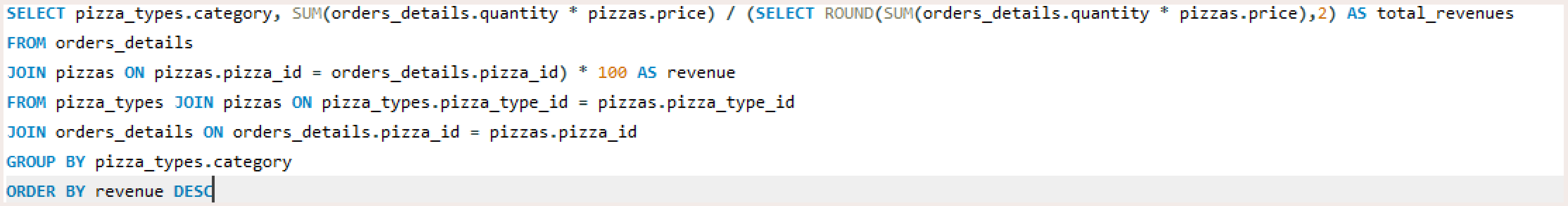


```
SELECT
    pizza_types.name,
    SUM(orders_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY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

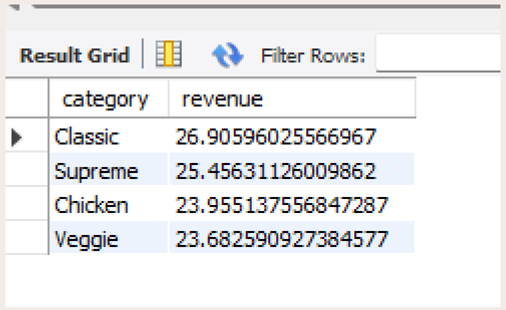


name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

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



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



Result Grid | Filter Rows:

category	revenue
Classic	26.90596025566967
Supreme	25.45631126009862
Chicken	23.955137556847287
Veggie	23.682590927384577

Conclusion

The "Pizza Sales Management System" leverages MySQL to create a robust platform for managing and analyzing sales data in a pizza restaurant chain. By implementing efficient database management and query execution, the system facilitates informed decision-making and operational efficiency, ultimately contributing to enhanced business performance and customer satisfaction.

Thank you

Bhuvana yelubandi

yelubandibhuvana@gmail.com