

# PIZZA SALES ANALYSIS REPORT

THE PIZZERIA

USING MYSQL



Prepared By :  
Anasua Sarkar

AUGUST 2024

# Introduction

---

In the competitive food industry, data-driven decision-making is essential for businesses to stay ahead. This project analyses pizza sales data to uncover valuable insights that can enhance a pizzeria's operational efficiency and profitability. By examining various aspects of sales, such as order frequency, revenue generation, and customer preferences, this analysis will provide a detailed understanding of the factors driving sales performance.

The project is structured into three levels of analysis: Basic, Intermediate, and Advanced. The Basic analysis addresses foundational questions such as total orders, revenue, and popular pizza types. The Intermediate analysis delves deeper into sales patterns, including hourly order distribution and category-wise pizza preferences. Finally, the Advanced analysis provides a more granular view of revenue contributions and the performance of pizza types within specific categories. Through this structured approach, the project aims to equip stakeholders with actionable insights that can guide menu optimisation, marketing strategies, and operational improvements, ultimately driving the business toward greater success.

# Problem statement

---

The primary goal of this project is to conduct a comprehensive analysis of pizza sales data to gain insights into customer preferences, sales trends, and revenue generation. By leveraging SQL queries in MySQL, the project aims to address key questions that can inform business decisions, such as understanding which pizzas are most popular, identifying peak sales periods, and analyzing the revenue contribution of different pizza types. The insights derived from this analysis will help the business optimize its menu offerings, pricing strategy, and marketing efforts to maximize profitability and customer satisfaction.

## Project Objectives:

---

The primary objectives of this project are to analyse the pizza sales data to achieve the following:

- Understand the overall sales performance and revenue generation.
- Identify customer preferences in terms of pizza types and sizes.
- Determine the distribution of sales over different times and categories.
- Analyse revenue contributions of various pizza types and categories.

# Analysis Questions:

---

To achieve these objectives, the following specific questions will be addressed:

## Basic Analysis:

1. Order Volume: What is the total number of orders placed? Revenue Calculation: How much total revenue has been generated from pizza sales?
2. Pricing Insight: Which pizza is the highest-priced?
3. Size Popularity: What is the most commonly ordered pizza size?
4. Top Pizza Types: Which are the top 5 most ordered pizza types and their respective quantities

## Intermediate Analysis:

1. Category-Wise Sales: What is the total quantity of each pizza category ordered?
2. Hourly Sales Distribution: How are orders distributed by the hour of the day?
3. Category Distribution: How do pizza orders distribute across different categories?
4. Daily Order Trends: What is the average number of pizzas ordered per day, grouped by date?
5. Top Pizza Revenue: Which are the top 3 most ordered pizza types based on revenue.

## Advanced Analysis:

1. Revenue Contribution: What is the percentage contribution of each pizza type to total revenue?
2. Cumulative Revenue: How has the cumulative revenue generated from pizza sales evolved over time?
3. Category-Specific Top Pizzas: What are the top 3 most ordered pizza types based on revenue within each pizza category?

# Setting Up the Database for Pizza Sales Analysis

---

In this project, I initiated the process of analyzing pizza sales by setting up a MySQL database to store and organize the relevant data. The steps taken are as follows:

## 1. Creating the Database:

I started by creating a new database named PIZZERIA to hold all the tables and data required for the analysis.

```
1 • CREATE DATABASE PIZZERIA;  
2 • USE PIZZERIA;
```

## 2. Verifying Data Import:

After importing the data, I performed an initial check by selecting all records from the pizzas and pizza\_types tables to ensure that the data was correctly loaded.

```
SELECT * FROM pizzas;  
SELECT * FROM pizza_types;
```

## 3. Creating the orders Table:

Next, I created a table named orders to store details of each pizza order. This table includes columns for order\_id, order\_date, and order\_time. The order\_id column is set as the primary key to uniquely identify each order.

```
• CREATE TABLE orders(  
  order_id int NOT NULL,  
  order_date DATE NOT NULL,  
  order_time TIME NOT NULL,  
  PRIMARY KEY(order_id)  
);
```

## 5 Creating the order\_details Table:

```
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)  
);
```

To capture the details of each order, I created the order\_details table. This table stores information such as order\_details\_id, order\_id, pizza\_id, and quantity. The order\_details\_id is the primary key, ensuring each record is unique, while order\_id acts as a foreign key to link to the orders table

## 5. Verification of Data Structure:

Finally, I verified the structure of the order\_details and orders table by selecting all its records to ensure that the tables are correctly created and ready to store the data.

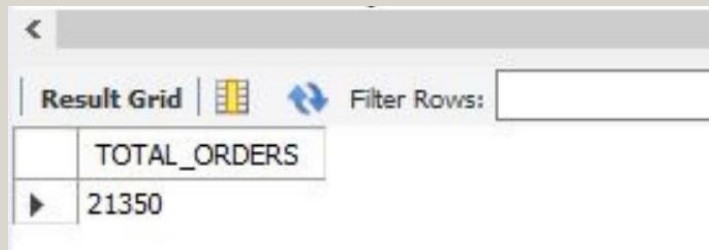
```
• SELECT * FROM orders;  
• SELECT * FROM order_details;  
|
```

# Basic Analysis

---

1. Order Volume: What is the total number of orders placed? Revenue Calculation: How much total revenue has been generated from pizza sales?

```
-- Retrieve the total number of orders placed. --  
SELECT (count(*))AS TOTAL_ORDERS FROM orders;
```



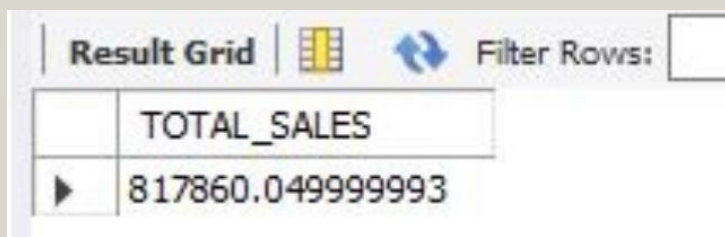
The screenshot shows a database interface with a 'Result Grid' tab. It contains a single row with the column header 'TOTAL\_ORDERS' and the value '21350'. There is a 'Filter Rows' input field to the right of the grid.

TOTAL_ORDERS
21350

Total Order: 21350

2. Calculate the total revenue generated from pizza sales.

```
SELECT SUM(order_details.quantity * pizzas.price) AS TOTAL_SALES  
FROM order_details  
JOIN  
pizzas  
ON order_details.pizza_id =pizzas.pizza_id;
```



The screenshot shows a database interface with a 'Result Grid' tab. It contains a single row with the column header 'TOTAL\_SALES' and the value '817860.049999993'. There is a 'Filter Rows' input field to the right of the grid.

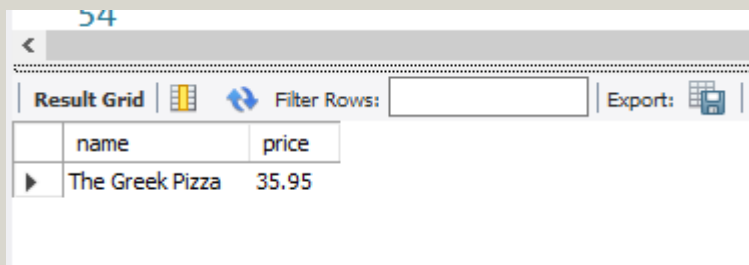
TOTAL_SALES
817860.049999993

Total Revenue: 817860 pounds



### 3. Pricing Insight: Which pizza is the highest-priced?

```
47      -- Identify the highest-priced pizza. --
48 •    SELECT pizza_types.name, pizzas.price
49      FROM pizza_types
50      JOIN
51      pizzas
52      ON pizza_types.pizza_type_id = pizzas.pizza_type_id
53      ORDER BY pizzas.price DESC LIMIT 1;
54
```



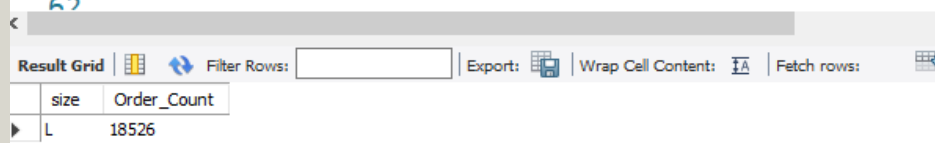
The screenshot shows a database interface with a 'Result Grid' tab selected. The grid displays two columns: 'name' and 'price'. The first row shows 'The Greek Pizza' with a price of 35.95. Above the grid, there are controls for 'Filter Rows' and 'Export'.

	name	price
▶	The Greek Pizza	35.95

The highest priced pizza is: "The Greek Pizza"

### 4. Size Popularity: What is the most commonly ordered pizza size?

```
54      -- Identify the most common pizza size ordered.--
55 •    SELECT pizzas.size, COUNT(order_details.pizza_id) AS Order_Count
56      FROM pizzas
57      JOIN
58      order_details
59      ON pizzas.pizza_id = order_details.pizza_id
60      GROUP BY pizzas.size
61      ORDER BY Order_Count DESC LIMIT 1;
62
```



The screenshot shows a database interface with a 'Result Grid' tab selected. The grid displays two columns: 'size' and 'Order\_Count'. The first row shows 'L' with an order count of 18526. Above the grid, there are controls for 'Filter Rows', 'Export', 'Wrap Cell Content', and 'Fetch rows'.

	size	Order_Count
▶	L	18526



## Most popular pizza size is "Large"

5. Top Pizza Types: Which are the top 5 most ordered pizza types and their respective quantities

```
63  -- List the top 5 most ordered pizza
64  -- types along with their quantities. --
65  • SELECT pizza_types.name AS Pizza_Name,
66      SUM(order_details.quantity) AS Total_order
67  FROM pizza_types
68  JOIN pizzas
69  ON
70      pizza_types.pizza_type_id = pizzas.pizza_type_id
71  JOIN order_details
72  ON
73      order_details.pizza_id = pizzas.pizza_id
74  GROUP BY pizza_types.name
75  ORDER BY Total_order DESC LIMIT 5;
76
```

Result Grid			Filter Rows:	Export:	Wrap Cell Content:	Fetch rows:
	Pizza_Name	Total_order				
▶	The Classic Deluxe Pizza	2453				
	The Barbecue Chicken Pizza	2432				
	The Hawaiian Pizza	2422				
	The Pepperoni Pizza	2418				
	The Thai Chicken Pizza	2371				

## The top five pizza type ordered is-

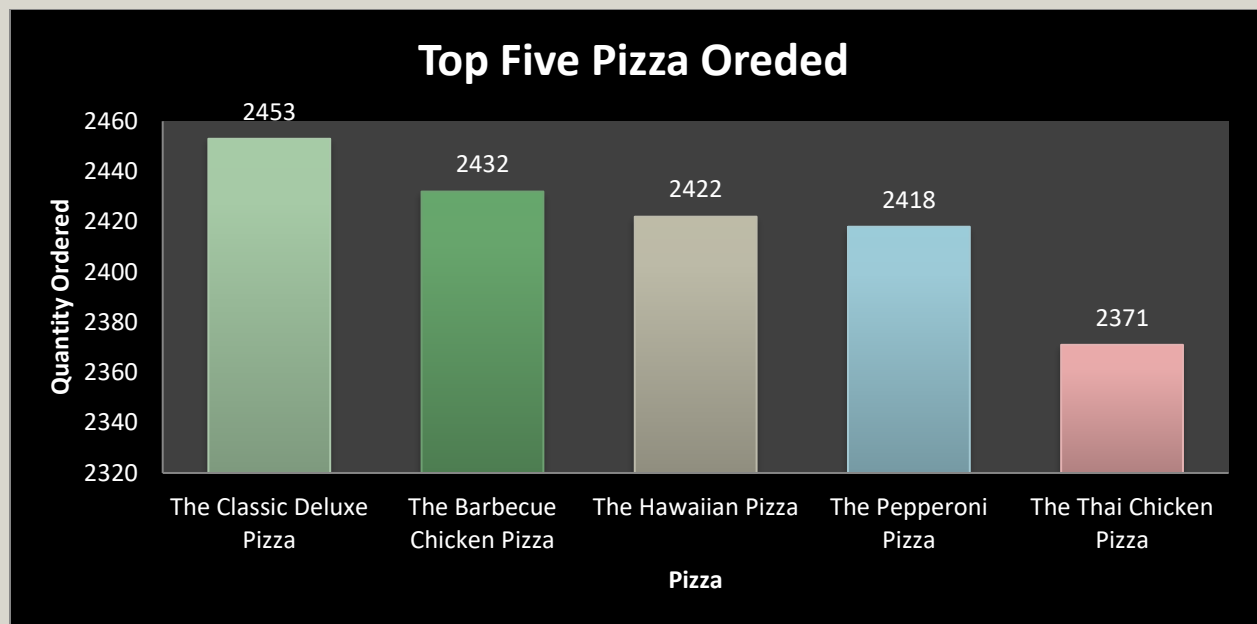
First is "The Classic Deluxe Pizza" with total number of order - 2453 pizzas.

Second "The Barbecue Chicken Pizza" with total number of order - 2432 pizzas.

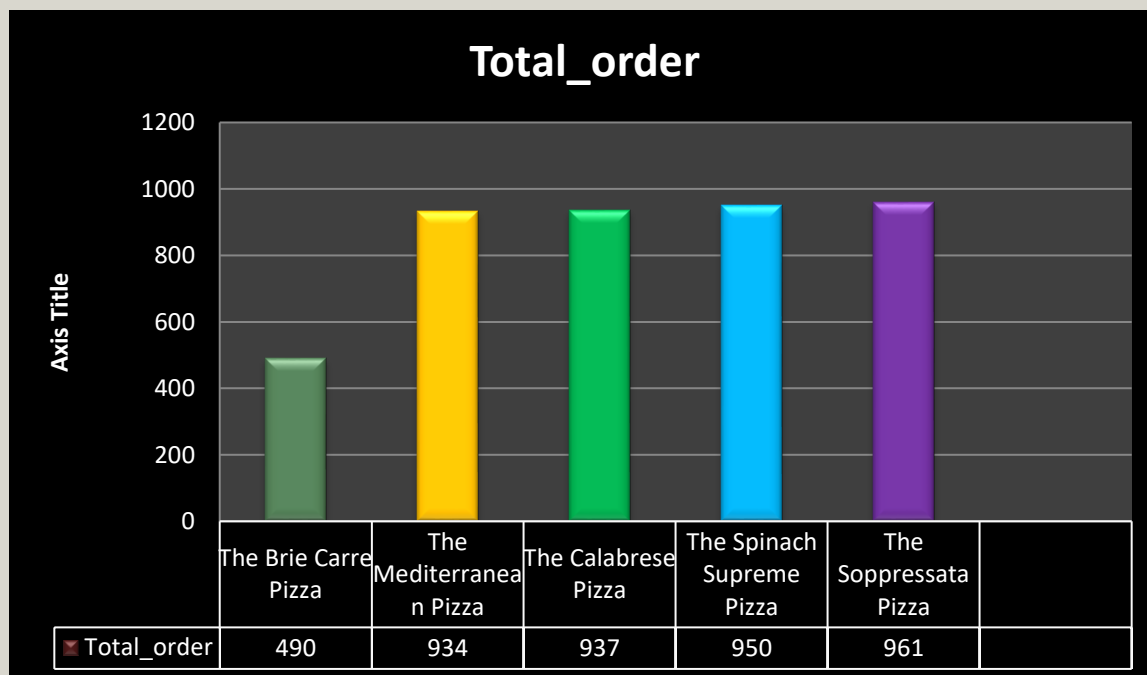
Third "The Hawaiian Pizza" with total number of order - 2422 pizzas.

Fourth "The Pepperoni Pizza" with total number of order - 2418 pizzas.

Fifth "The Thai chicken Pizza" with total number of order - 2371 pizzas.



## List Ordered Pizzas



The list orderd pizzas are –

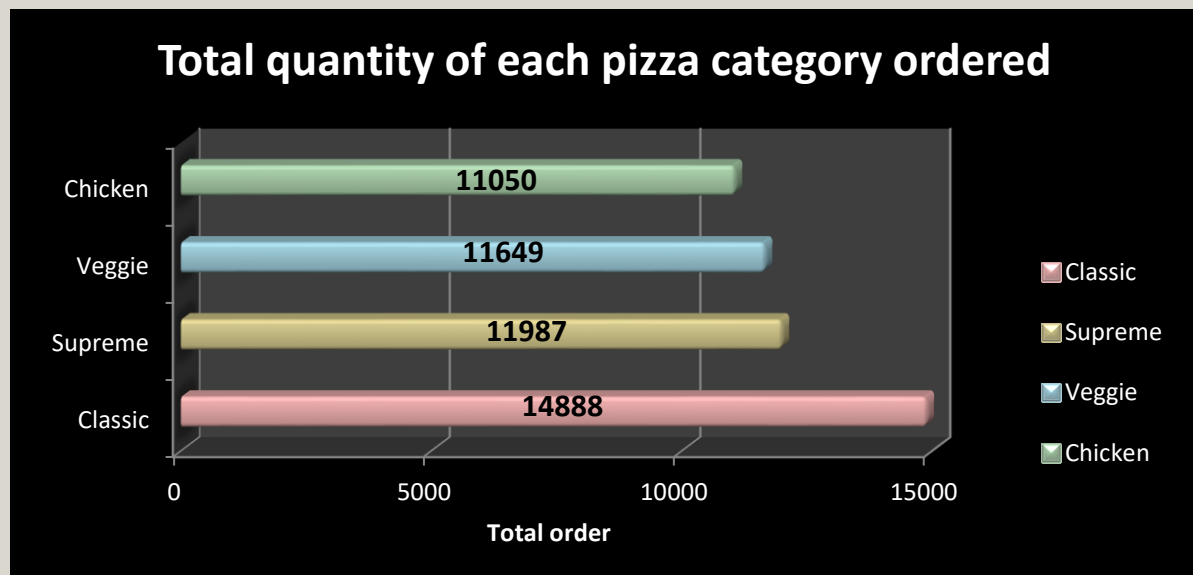
The Brie Carre Pizza with 490 orders only, The Mediterran Pizza with 934 orders only, The Calabrese Pizza with 937 orders only, The Spinach Supreme Pizza with 950 orders onl and The Soppressat Pizza with 961 orders only.

# Intermediate Analysis

1. What is the total quantity of each pizza category ordered?

```
92 • SELECT pizza_types.category ,
93      SUM(order_details.quantity) AS order_quantity
94 FROM   pizza_types
95 JOIN   pizzas
96 ON     pizza_types.pizza_type_id = pizzas.pizza_type_id
97 JOIN   order_details
98 ON     order_details.pizza_id = pizzas.pizza_id
99 GROUP BY pizza_types.category
100 ORDER BY order_quantity DESC;
101
```

category	order_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050



2. How are orders distributed by the hour of the day?

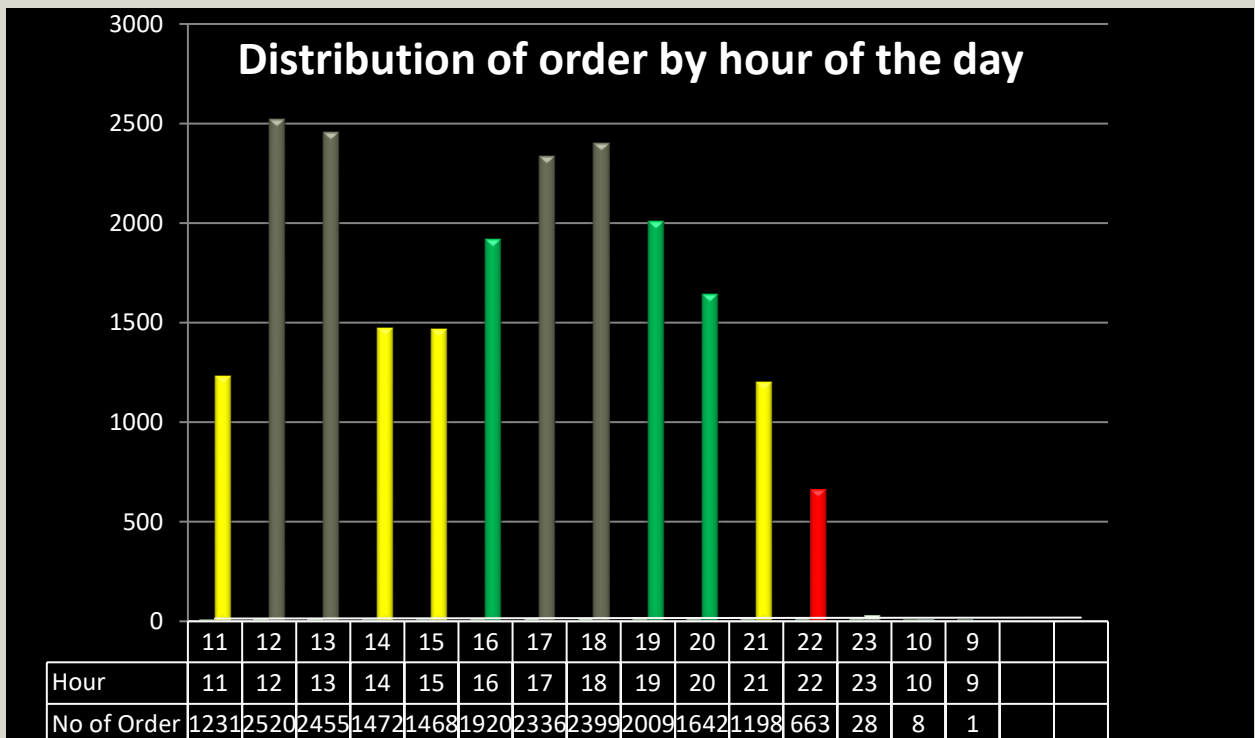
```

101  -- Determine the distribution of
102  -- orders by hour of the day. --
103  • SELECT hour(order_time) AS Hour,
104      count(order_id) AS Number_of_Order
105  FROM orders

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |





	Hour	Number_of_Order
▶	11	1231
	12	2520
	13	2455
	14	1472
	15	1468
	16	1920
	17	2336
	18	2399
	19	2009
	20	1642
	21	1198
	22	663
	23	28
	10	8
	9	1



The busiest hours of the day are 12 pm to 2pm in the afternoon and 4 pm to 9pm.

1. Category Distribution: How do pizza orders distribute across different categories?





```
108 -- categoriwise distribution of pizza --
109 • SELECT category,
110     COUNT(name) As Number_of_Diffrent_pizza_in_this_Category
111     FROM pizza_types
112     GROUP BY category;
113
```

<   Filter Rows:  | Export:  | Wrap Cell Content: 

	category	Number_of_Diffrent_pizza_in_this_Category
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9

2. Daily Order Trends: What is the average number of pizzas ordered per day, grouped by date?

```
107 -- Group the orders by date and calculate
108 -- the avarage number of pizza ordered by date --
109 • SELECT round(avg(quantity),0) AS avarage_order_per_day FROM
110     (SELECT orders.order_date,sum(order_details.quantity) AS quantity
111     FROM orders JOIN order_details
112     ON orders.order_id = order_details.order_id
113     GROUP BY
114     orders.order_date) AS order_quantity;
```

<   Filter Rows:  | Export:  | Wrap Cell Content: 

	avarage_order_per_day
▶	138

Average Per day order is 138.

3. Pizza Revenue: Which are the top 3 most ordered pizza types based on revenue.

```
115 -- Determine top 3 most ordered pizza type base on revenue --
116 • SELECT pizza_types.name AS Pizza_Name,
117    sum(order_details.quantity*pizzas.price) AS revenue
118 from pizza_types
119 JOIN
120 pizzas
121 ON pizza_types.pizza_type_id=pizzas.pizza_type_id
122 JOIN order_details
123 ON order_details.pizza_id = pizzas.pizza_id
124 GROUP BY pizza_types.name
125 ORDER BY revenue DESC LIMIT 3;
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:	Fetch rows:
	Pizza_Name	revenue			
▶	The Thai Chicken Pizza	43434.25			
	The Barbecue Chicken Pizza	42768			
	The California Chicken Pizza	41409.5			

Top three revenue generating pizza is

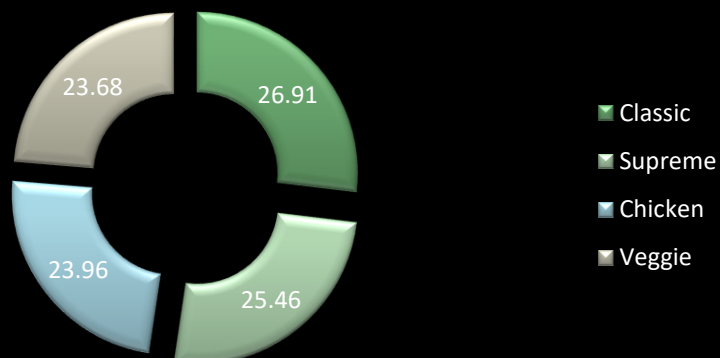
1. The Thai Chicken Pizza
2. The Barbecue Chicken Pizza
3. The California Chicken Pizza

# Advanced Analysis:

1. Revenue Contribution: What is 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 order_details.pizza_id =pizzas.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;
```

percentage contribution of each pizza  
type to total revenue







2. Cumulative Revenue: How has the cumulative revenue generated from pizza sales evolved over time?

Step -1

Revenue by day:

```
153 • SELECT orders.order_date,  
154      sum(order_details.quantity * pizzas.price)as revenue  
155 FROM order_details JOIN pizzas  
156 ON order_details.pizza_id = pizzas.pizza_id  
157 JOIN  
158 orders  
159 ON orders.order_id = order_details.order_id  
160 GROUP BY orders.order_date;
```

<		
Result Grid		
Filter Rows: <input type="text"/>		
Export: 		
Wrap Cell Content: 		
	order_date	revenue
▶	2015-01-01	2713.8500000000004
	2015-01-02	2731.8999999999996
	2015-01-03	2662.3999999999996
	2015-01-04	1755.4500000000003
	2015-01-05	2065.95
	2015-01-06	2428.95
	2015-01-07	2202.2000000000003
	2015-01-08	2838.3499999999995
	2015-01-09	2127.3500000000004
	2015-01-10	2463.95

## Step -2

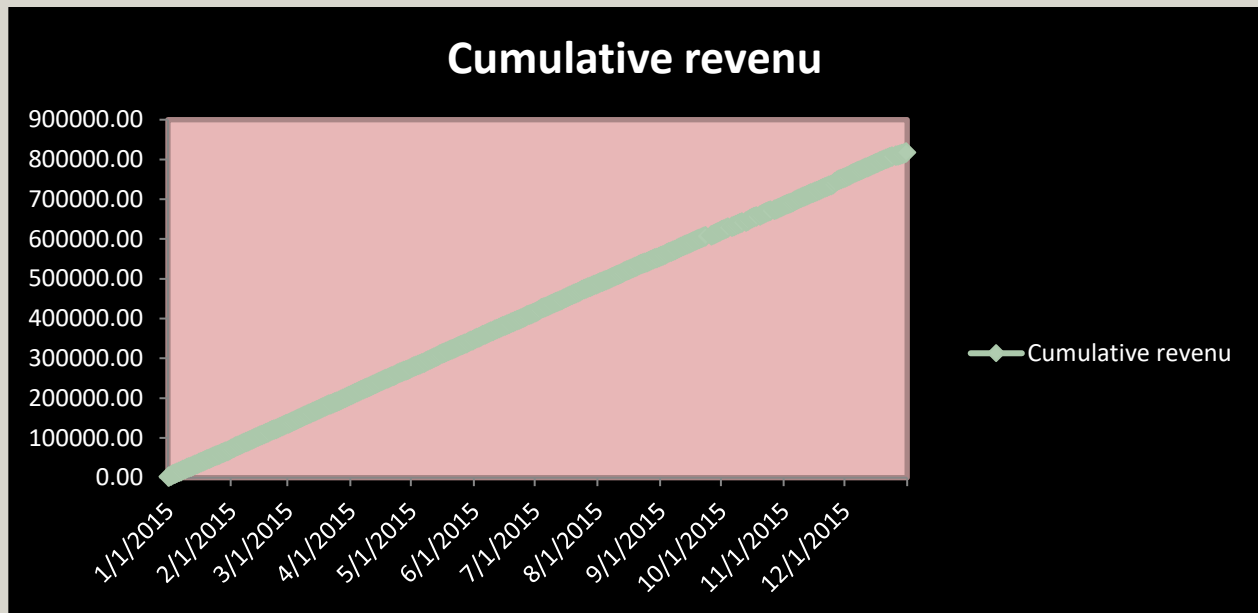
### Cumulative Revenue:

```
161  -- step 2--
162 • SELECT order_date,
163      SUM(revenue) OVER(ORDER BY order_date) as cumulative_revenue
164  FROM
165      (SELECT orders.order_date,
166         sum(order_details.quantity * pizzas.price)as revenue
167        FROM order_details JOIN pizzas
168        ON order_details.pizza_id = pizzas.pizza_id
169        JOIN
170         orders
171        ON orders.order_id = order_details.order_id
172       GROUP BY orders.order_date) AS Sales;
173
```

Result Grid	Filter Rows:	E
order_date	cumulative_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	
2015-01-13	29831.300000000003	
2015-01-14	32358.700000000004	
2015-01-15	34343.500000000001	
2015-01-16	36937.650000000001	
2015-01-17	39001.750000000001	
2015-01-18	40978.600000000006	
2015-01-19	43365.750000000001	

The cumulative revenue data tracks the total revenue generated by the business over a specific period. This metric is crucial for understanding the overall financial performance and growth trajectory of the business.

The data indicates a consistent increase in cumulative revenue over the specified period. This upward trend suggests a steady flow of sales, which is a positive sign for the business's financial health.



2. Category-Specific Top Pizzas: What are the top 3 most ordered pizza types based on revenue within each pizza category?

### Step-1: **Calculating Revenue for Each Pizza Type**

In this step, the revenue for each pizza type was calculated by summing the product of the quantity ordered and the price of each pizza. This was done for each pizza type within its respective category. The SQL query for this step is as follows:

```

175 -- Determine the top 3 most ordered pizza type
176 -- Based on revenue for each pizza category
177 -- Step - 1
178 • SELECT pizza_types.category,pizza_types.name,|
179      sum((order_details.quantity)*pizzas.price) AS revenue
180 FROM pizza_types JOIN pizzas
181 ON pizza_types.pizza_type_id = pizzas.pizza_type_id
182 JOIN order_details
183 ON order_details.pizza_id = pizzas.pizza_id
184 GROUP BY pizza_types.category,pizza_types.name;

```

Result Grid			
Filter Rows:		Export:	Wrap Cell Content:
	category	name	revenue
▶	Classic	The Hawaiian Pizza	32273.25
	Classic	The Classic Deluxe Pizza	38180.5
	Veggie	The Five Cheese Pizza	26066.5
	Supreme	The Italian Supreme Pizza	33476.75
	Veggie	The Mexicana Pizza	26780.75
	Chicken	The Thai Chicken Pizza	43434.25
	Supreme	The Prosciutto and Arugula Pizza	24193.25
	Chicken	The Barbecue Chicken Pizza	42768
	Classic	The Greek Pizza	28454.100000000013
	Supreme	The Spinach Supreme Pizza	15277.75
	Veggie	The Green Garden Pizza	13955.75
	Classic	The Italian Capocollo Pizza	25094
	Supreme	The Spicy Italian Pizza	34831.25
	Veggie	The Spinach Pesto Pizza	15596
	Veggie	The Vegetables + Vegetables Pi...	24374.75
	Chicken	The Southwest Chicken Pizza	34705.75
	Chicken	The California Chicken Pizza	41409.5
	Classic	The Pepperoni Pizza	30161.75
	Chicken	The Chicken Pesto Pizza	16701.75

## Step-2: Ranking Pizzas by Revenue Within Each Category



After calculating the revenue, pizzas were ranked within their categories based on the revenue generated. This was done using the RANK() function,

which assigns a rank to each pizza type within its category according to its revenue. The SQL query for this step is:

```

184  -- Step -2
185 •  SELECT category,name, revenue,
186      RANK() OVER(PARTITION BY category ORDER BY revenue DESC) AS revn
187  FROM
188  (SELECT pizza_types.category,pizza_types.name,
189      sum((order_details.quantity)*pizzas.price) AS revenue
190      FROM pizza_types JOIN pizzas
191      ON pizza_types.pizza_type_id = pizzas.pizza_type_id
192      JOIN order_details
193      ON order_details.pizza_id = pizzas.pizza_id
194      GROUP BY pizza_types.category,pizza_types.name) AS table_a;

```

Result Grid				
		Filter Rows:		Export:  Wrap Cell Content: 
	category	name	revenue	revn
▶	Chicken	The Thai Chicken Pizza	43434.25	1
	Chicken	The Barbecue Chicken Pizza	42768	2
	Chicken	The California Chicken Pizza	41409.5	3
	Chicken	The Southwest Chicken Pizza	34705.75	4
	Chicken	The Chicken Alfredo Pizza	16900.25	5
	Chicken	The Chicken Pesto Pizza	16701.75	6
	Classic	The Classic Deluxe Pizza	38180.5	1
	Classic	The Hawaiian Pizza	32273.25	2
	Classic	The Pepperoni Pizza	30161.75	3
	Classic	The Greek Pizza	28454.100000000013	4
	Classic	The Italian Capocollo Pizza	25094	5
	Classic	The Napolitana Pizza	24087	6
	Classic	The Big Meat Pizza	22968	7
	Classic	The Pepperoni, Mushroom, ...	18834.5	8
	Supreme	The Spicy Italian Pizza	34831.25	1
	Supreme	The Italian Supreme Pizza	33476.75	2
	Supreme	The Sicilian Pizza	30940.5	3
	Supreme	The Pepper Salami Pizza	25529	4
	Supreme	The Prosciutto and Arugula ...	24193.25	5
	Supreme	The Soppressata Pizza	16425.75	6

### Step: 3: **Selecting the Top 3 Pizzas per Category**

Finally, the top 3 pizzas in terms of revenue for each category were selected. This step filtered the ranked pizzas to include only those with a rank of 1, 2, or 3. The SQL query used is:

```

196 • SELECT category,name, revenue,pizza_rank
197 FROM
198 (SELECT category,name, revenue,
199 RANK() OVER(PARTITION BY category ORDER BY revenue DESC) AS pizza_rank
200 FROM
201 (SELECT pizza_types.category,pizza_types.name,
202 sum((order_details.quantity)*pizzas.price) AS revenue
203 FROM pizza_types JOIN pizzas
204 ON pizza_types.pizza_type_id = pizzas.pizza_type_id
205 JOIN order_details
206 ON order_details.pizza_id = pizzas.pizza_id
207 GROUP BY pizza_types.category,pizza_types.name) AS table_a) AS table_b
208 WHERE pizza_rank<=3;

```

category	name	revenue	pizza_rank
Chicken	The Thai Chicken Pizza	43434.25	1
Chicken	The Barbecue Chicken Pizza	42768	2
Chicken	The California Chicken Pizza	41409.5	3
Classic	The Classic Deluxe Pizza	38180.5	1
Classic	The Hawaiian Pizza	32273.25	2
Classic	The Pepperoni Pizza	30161.75	3
Supreme	The Spicy Italian Pizza	34831.25	1
Supreme	The Italian Supreme Pizza	33476.75	2
Supreme	The Sicilian Pizza	30940.5	3
Veggie	The Four Cheese Pizza	32265.70000000065	1
Veggie	The Mexicana Pizza	26780.75	2
Veggie	The Five Cheese Pizza	26066.5	3

The query successfully returned the top 3 most ordered pizza types based on revenue for each category. These results can now be used to gain insights into which pizzas are driving the most revenue and could help in making informed business decisions such as menu adjustments, targeted promotions, or inventory prioritization.

The analysis identified the top 3 most ordered pizza types based on revenue for each category. Below is a summary of the findings:

#### Chicken Category:

1. The Thai Chicken Pizza - \$43,434.25
2. The Barbecue Chicken Pizza - \$42,768.00
3. The California Chicken Pizza - \$41,409.50

#### Classic Category:

1. The Classic Deluxe Pizza - \$38,180.50
2. The Hawaiian Pizza - \$32,273.25
3. The Pepperoni Pizza - \$30,161.75

#### Supreme Category:

1. The Spicy Italian Pizza - \$34,831.25
2. The Italian Supreme Pizza - \$33,476.75
3. The Sicilian Pizza - \$30,940.50

#### Veggie Category:

1. The Four Cheese Pizza - \$32,265.70
2. The Mexicana Pizza - \$26,780.75
3. The Five Cheese Pizza - \$26,066.50

#### Explanation

The results show that each category has a distinct set of top performers. For example, in the Chicken category, The Thai Chicken Pizza leads with the highest revenue, followed closely by The Barbecue Chicken Pizza and The California Chicken Pizza. Similarly, in the Classic category, The Classic Deluxe Pizza generates the most revenue, while in the Supreme category,



The Spicy Italian Pizza ranks first. For the Veggie category, The Four Cheese Pizza is the top-seller.

The query successfully returned the top 3 most ordered pizza types based on revenue for each category. These results can now be used to gain insights into which pizzas are driving the most revenue and could help in making informed business decisions such as menu adjustments, targeted promotions, or inventory prioritization.

## Conclusion

---

The pizzeria sales analysis provided insightful data that can significantly inform business strategies and decision-making processes.

**Order Volume & Revenue:** The pizzeria has achieved a substantial total order volume of 21,350 orders, resulting in total revenue of £817,860. These figures demonstrate a strong market demand and effective sales strategies.

**Top Performers:** The analysis revealed that "The Greek Pizza" is the highest-priced item on the menu, while "The Classic Deluxe Pizza" is the most popular, with 2,453 orders. Additionally, the "Large" size is the most commonly ordered, indicating customer preference for larger portions. The top five pizzas, led by "The Classic Deluxe Pizza," collectively dominate the menu, highlighting key products that drive sales.

**Category Insights:** The sales data indicates a fairly balanced distribution across different pizza categories, with the "Classic" category

leading in total orders (14,888 pizzas). The chicken-based pizzas, particularly "The Thai Chicken Pizza," have emerged as significant contributors to revenue, underlining the popularity of this flavor profile among customers.

Hourly and Daily Trends: Peak order times were identified between 12 PM to 2 PM and 4 PM to 9 PM, aligning with typical lunch and dinner hours. The average daily order volume stands at 138, providing a baseline for operational planning and staffing.

Revenue Contribution: Each pizza category contributes significantly to the total revenue, with "Classic" pizzas contributing the most (26.91%), followed closely by "Supreme," "Chicken," and "Veggie" categories. This distribution suggests that while all categories are important, focusing on the top-performing categories could yield greater financial benefits.

## Strategic Implications

The findings highlight the need to possibly prioritize inventory for top-selling pizzas and consider promotional strategies for the less popular ones. Additionally, the distinct peak order hours suggest a targeted approach in staffing and resource allocation during these times to optimize customer service and operational efficiency.

## Final Thoughts

This analysis has provided a clear picture of the pizzeria's sales dynamics, revealing both strengths and opportunities for growth. By leveraging these insights, the pizzeria can enhance its product offerings, optimize operations, and ultimately increase profitability.