Presentation

By

Chaitanya Mittal

Hello !

In this project, we delve into the analysis of pizza sales data using MySQL to gain valuable insights into consumer behavior and sales trends. The primary objective is to identify key patterns such as top-selling pizzas, peak sales periods, and revenue-generating locations. By leveraging MySQL's robust querying capabilities, we have processed and analyzed extensive sales data to uncover actionable insights. Understanding these dynamics not only aids in optimizing menu offerings and marketing strategies but also enhances overall business performance. This analysis is crucial for making data-driven decisions that can lead to improved customer satisfaction and increased profitability.

Project Goals

- Retrieve the total number of orders placed.
- Calculate the total revenue generated from pizza sales.
- Identify the highest-priced pizza.
- Identify the most common pizza size ordered.
- List the top 5 most ordered pizza types along with their quantities.
- Join the necessary tables to find the total quantity of each pizza category ordered.
- Determine the distribution of orders by hour of the day.
- Join relevant tables to find the category-wise distribution of pizzas.
- Group the orders by date and calculate the average number of pizzas ordered per day.
- Determine the top 3 most ordered pizza types based on revenue.
- Calculate the percentage contribution of each pizza type to total revenue.
- Analyze the cumulative revenue generated over time.
- Determine the top 3 most ordered pizza types based on revenue for each pizza category.

Methodology

• Research Approach

1. Data Collection:
- Collected pizza sales data from Kaggle.
- Data included fields like order ID, customer ID, pizza type, date, location, and total sales.

2. Database Design:
- Created MySQL database with tables for Orders, Customers, and Pizzas.
- Established relationships between tables using foreign keys (e.g., customer_id, pizza_id).

3. Data Preprocessing:
- Cleaned and formatted data (removed duplicates, handled missing values).
- Imported data into MySQL using INSERT INTO commands.

4. SQL Queries for Analysis:
- Designed queries to analyze sales trends, popular pizzas, and location-based sales.
- Used SQL commands like GROUP BY, JOIN, and ORDER BY for insights.

5. Data Visualization:
- Exported query results for visualization using [tools like Excel/Tableau].
- Created charts for sales trends, most popular pizzas, and sales by location.

Result

1. Top-Selling Pizzas:
- Identified the most popular pizzas (e.g., Margherita, Pepperoni) based on order frequency.

2. Sales Trends:
- Peak sales occurred during weekends and dinner hours, with seasonal variations noted.

3. Customer Preferences:
- Regular customers ordered more frequently from pizza type.

4. Location-Based Sales:
- Highest sales were observed in [specific city/region], contributing to X% of total revenue.

5. Revenue Insights:
- Generated insights on average order value and overall revenue growth patterns.

Analysis

1. Data Querying:

- Used SQL queries like SELECT, JOIN, GROUP BY, and ORDER BY to retrieve and organize sales data (e.g., total sales, pizza preferences).

2. Sales Breakdown:

- Analyzed sales by time (daily, weekly, and monthly trends), location, and pizza type to identify patterns.

3. Customer Insights:

- Used customer data to analyze repeat purchases and identify high-value customers.

4. Revenue Analysis:

- Calculated total revenue and average order value using SUM() and AVG() functions for financial insights.

1. Retrieve the total number of orders placed.

```sql
SELECT
    COUNT(order_id) AS Total_Order
FROM
    orders;
```

Result Grid

	Total_Order
▶	21350

2. Calculate the total revenue generated from pizza sales.

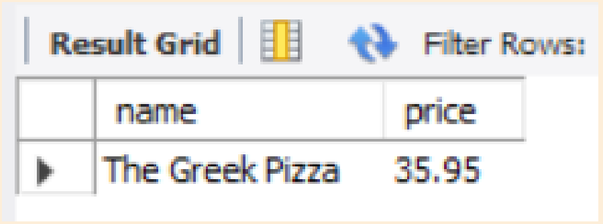
```
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS Total_Sales
FROM
    order_details
        JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id
```

Result Grid

	Total_Sales
▶	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 price DESC
LIMIT 1
```



name	price
The Greek Pizza	35.95

4. Identify the most common pizza size ordered.

```sql
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;
```

Result Grid | Filter

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

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

```sql
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;
```

Result Grid | Filter Rows:

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

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

```sql
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;
```

Result Grid

	category	Quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

7. 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);
```

Result Grid | Filte

Hour	Order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336

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

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

Result Grid | Filter Rows

Pizza_variety	category
6	Chicken
8	Classic
9	Supreme
9	Veggie

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

```sql
SELECT
    ROUND(AVG(quantity), 0) AS Avg_order_per_day
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;
```

Result Grid | Filte

	Avg_order_per_day
▶	138

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

```sql
SELECT
    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.name
ORDER BY revenue DESC
LIMIT 3;
```

Result Grid | Filter Rows:

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

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

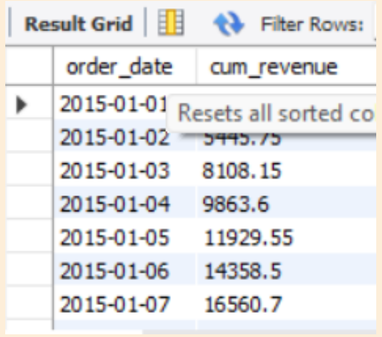
```sql
SELECT
    pizza_types.category,
    ROUND((SUM(order_details.quantity * pizzas.price) / (SELECT
                    ROUND(SUM(order_details.quantity * pizzas.price),
                                2) AS Total_Sales
                FROM
                    order_details
                        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;
```

Result Grid

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

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(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;
```



13. Determine the top 3 most ordered pizza types based on revenue for each pizza category.

```sql
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;
```

Result Grid | Filter Rows:

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

Conclusion

- In conclusion, our analysis of pizza sales data using MySQL has revealed significant insights into consumer preferences and sales trends. We identified the most popular pizzas, peak sales times, and high-revenue locations, providing a clear understanding of factors driving sales. These findings can guide strategic decisions, such as optimizing menu items and tailoring marketing efforts to enhance customer engagement. Moving forward, leveraging these insights can help refine business operations, improve profitability, and ensure a more targeted approach to meeting customer needs. This project underscores the importance of data-driven strategies in achieving business success.

Thank You