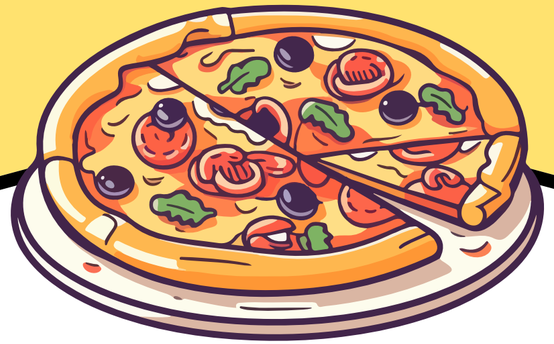

Pizza

SALES ANALYSIS

BY- HITESH GUPTA

Introduction

The **Pizza Sale Analysis** project is designed to explore and analyze sales data from a pizza store, using SQL queries to uncover valuable insights and identify emerging trends. The goal of this project is to demonstrate proficiency in SQL by answering key business questions through data analysis, leveraging MySQL as the database management system and MySQL Workbench as the integrated development environment.

The findings of this analysis will provide a comprehensive understanding of the business landscape, enabling data-driven decisions to optimize operations, enhance customer satisfaction, and drive overall business growth.

Project Overview

This project involves working with a dataset that has been meticulously cleaned and pre-processed to ensure the accuracy and relevance of the data. The dataset is structured in a relational format, with various tables representing different aspects of the business.

The project adopts a systematic approach, beginning with data exploration to understand the dataset's structure and contents. This is followed by targeted SQL queries aimed at answering specific business questions. Each query addresses a particular business aspect, and the results are interpreted to provide actionable recommendations.

Q1. Retrieve the total number of orders placed.

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

Result Grid

total_orders
21350

Q2. Calculate the total revenue generated from pizza sales.

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

total_sales
817860.05

Q3. Identify the highest-priced pizza.

```sql
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

Q4. Identify the different pizza sizes 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;
```

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

Q5. 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;
```

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

Q6. 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;
```

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Q7. Determine the distribution of orders by hour of the day.

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

HOUR(order_time)	order_count
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

Q8. Join relevant tables to find the category-wise distribution of pizzas.

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

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

Q9. Group the orders by date and calculate the average number of pizzas ordered per day.

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

	avg_order
▶	138

Q10. 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;
```

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

Q11. 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 pizzas.pizza_id = order_details.pizza_id) )*100,2) AS percent_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 percent_revenue DESC;
```

category	percent_revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Q12. Analyze the cumulative revenue generated over time.

```sql
SELECT
    order_date,
    SUM(revenue) OVER(ORDER BY order_date) AS cumul_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;
```

order_date	cumul_revenue
2015-01-01	2713.850000000000
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	23990.35000000000

2015-12-21	801288.65
2015-12-22	803171.6
2015-12-23	805415.9
2015-12-24	807553.75
2015-12-26	809196.8
2015-12-27	810615.8
2015-12-28	812253
2015-12-29	813606.25
2015-12-30	814944.05
2015-12-31	817860.05

Conclusion

The Pizza Sale Analysis project not only demonstrates the ability to work with SQL and MySQL but also highlights the importance of data-driven decision-making in a business context. By uncovering insights and trends from the sales data, the project provides a solid foundation for optimizing business strategies, improving customer satisfaction, and ultimately driving revenue growth.

THANK YOU