Pizza Sales Analysis Using MySQL

Presented By Sanskar Choudhari

ABOUT PROJECT

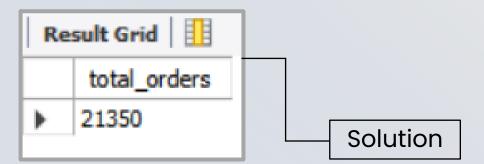
This project focuses on analyzing a pizza sales dataset using **MySQL** to extract meaningful business insights and support data-driven decision-making. The goal was to explore sales patterns, product performance, and customer behavior through SQL queries ranging from basic aggregations to advanced analytics.

The analysis was conducted in MySQL, and all results, including SQL query screenshots and visual summaries, have been compiled into this presentation to demonstrate technical proficiency and real-world data application.

Retrieve the total number of orders placed.

```
-- Retrieve the total number of orders placed.

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



Calculate the total revenue generated from pizza sales.

```
-- Calculate the total revenue generated from pizza sales.

SELECT

ROUND(SUM(order_details.quantity * pizzas.price),

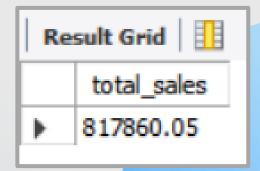
2) AS total_sales

FROM

order_details

JOIN

pizzas ON pizzas.pizza_id = order_details.pizza_id
```



Identify the highest-priced pizza.

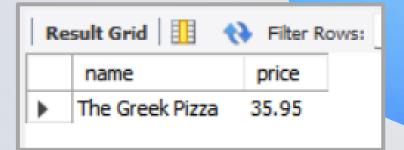
```
-- Identify the highest-priced pizza.

SELECT
    pizza_types.name, pizzas.price

FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id

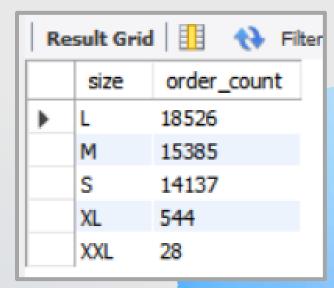
ORDER BY pizzas.price DESC

LIMIT 1;
```



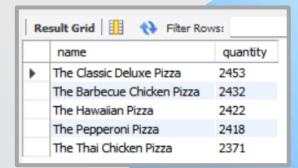
Identify the most common pizza size ordered.

```
-- Identify the most common pizza size ordered.
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;
```



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

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



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

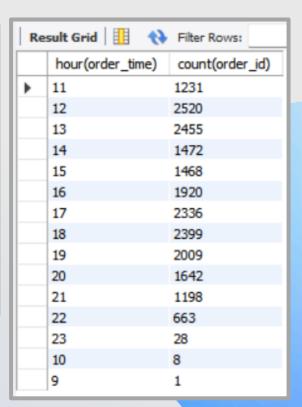
```
-- Join the necessary tables to find the total quantity of each pizza category ordered.
SELECT
    pizza types.category,
    SUM(order details.quantity) AS quantity
                                                                  FROM
                                                                             quantity
                                                                     category
    pizza types
                                                                    Classic
                                                                             14888
        JOIN
                                                                    Supreme
                                                                             11987
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
                                                                             11649
                                                                    Veggie
        JOIN
                                                                    Chicken
                                                                             11050
    order details ON order details.pizza id = pizzas.pizza id
GROUP BY pizza types.category
ORDER BY quantity DESC;
```

Determine the distribution of orders by hour of the day.

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

SELECT
   HOUR(order_time), COUNT(order_id)

FROM
   orders
GROUP BY HOUR(order_time);
```

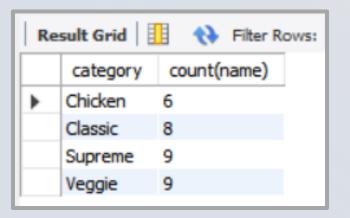


> Join relevant tables to find the category-wise distribution of pizzas.

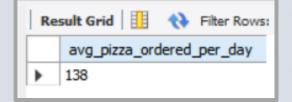
```
-- Join relevant tables to find the category-wise distribution of pizzas.

SELECT
category, COUNT(name)

FROM
pizza_types
GROUP BY category;_
```

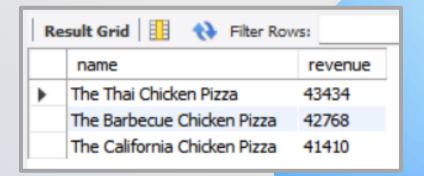


Group the orders by date and calculate the average number of pizzas ordered per day.



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

```
-- Determine the top 3 most ordered pizza types based on revenue.
SELECT
    pizza_types.name,
    ROUND(SUM(order_details.quantity * pizzas.price),
            0) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza type id = pizza types.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;
```



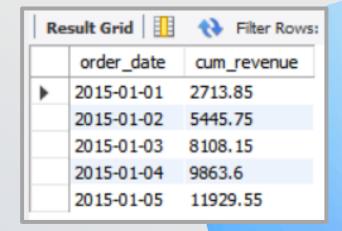
Calculate the percentage contribution of each pizza type to total revenue.

```
-- Calculate 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 pizzas.pizza_id = order_details.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		

Analyze the cumulative revenue generated over time.

```
-- Analyze the cumulative revenue generated over time.
SELECT
    order date,
    ROUND(SUM(revenue) OVER (ORDER BY order date), 2) AS cum revenue
FROM (
    SELECT
        o.order date,
        ROUND(SUM(od.quantity * p.price), 2) AS revenue
    FROM
        order details od
    JOIN
        pizzas p ON od.pizza_id = p.pizza_id
    JOIN
        orders o ON o.order_id = od.order_id
    GROUP BY
        o.order date
) AS sales;
```



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

```
-- Determine the top 3 most ordered pizza types based on revenue for each pizza category.
SELECT
    ROUND(revenue, 2) AS revenue
FROM (
    SELECT
        category,
        revenue,
        RANK() OVER (PARTITION BY category ORDER BY revenue DESC) AS rn
    FROM (
        SELECT
            pt.category,
            pt.name,
            SUM(od.quantity * p.price) AS revenue
        FROM
            pizza_types pt
        JOIN
            pizzas p ON pt.pizza type id = p.pizza type id
            order_details od ON od.pizza_id = p.pizza_id
            pt.category,
            pt.name
    ) AS a
) AS b
    rn <= 3;
```

Result Grid			
	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	
	The Italian Supreme Pizza	33476.75	
	The Sicilian Pizza	30940.5	
	The Four Cheese Pizza	32265.7	
	The Mexicana Pizza	26780.75	
	The Five Cheese Pizza	26066.5	

Strategies for growth

KEY STRATEGIES:

- Identify top-performing products to boost marketing focus.
- > Determine customer preferences by time and size.
- > Leverage high-revenue pizza types for bundled offers.
- Align kitchen operations with peak ordering hours.

Marketing strategies

Insights:

- Category-Wise Quantity Analysis helps decide what type of pizzas to scale up.
- Hourly Distribution of Orders guides expansion of delivery hours or promotions.
- Category Distribution informs inventory and staffing.

Opportunities:

- Promote popular categories during high order hours.
- > Expand offerings in underperforming categories.

Innovative solutions

Solutions:

- > Focus on high-revenue items for premium promotions.
- Reconsider stock for low-contribution pizzas.



Sales Targets

Key Metric:

Average Pizzas Ordered Per Day: **138**

Use Case:

Set weekly or monthly pizza sales targets based on the average daily sales to monitor store performance.

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

SELECT

ROUND(AVG(quantity), 0) AS avg_pizza_ordered_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;
```

Future initiatives

Cumulative Revenue Over Time

Top 3 Revenue Pizzas by Category:

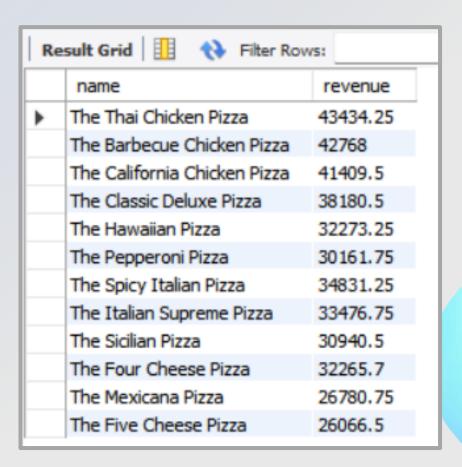
The Thai Chicken Pizza
The Barbecue Chicken Pizza
The California Chicken Pizza

Future Plans:

Dynamic pricing based on trends.

Automated dashboards for daily sales monitoring.

Geo-based performance tracking with branch comparison.



Thank you

- > Connect With Me:
- Github: https://github.com/sanskarchoudhari-1
- Email: sanskarchoudhari3@gmail.com
- Linkedin: www.linkedin.com/in/sanskar-choudhari-720729207