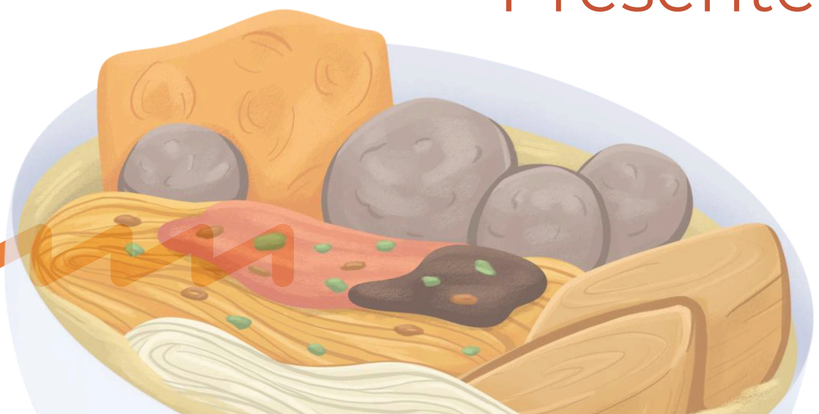
PIZZA SALES ANALYSIS

Presented by Anuj Sharma



PROJECT OVERVIEW

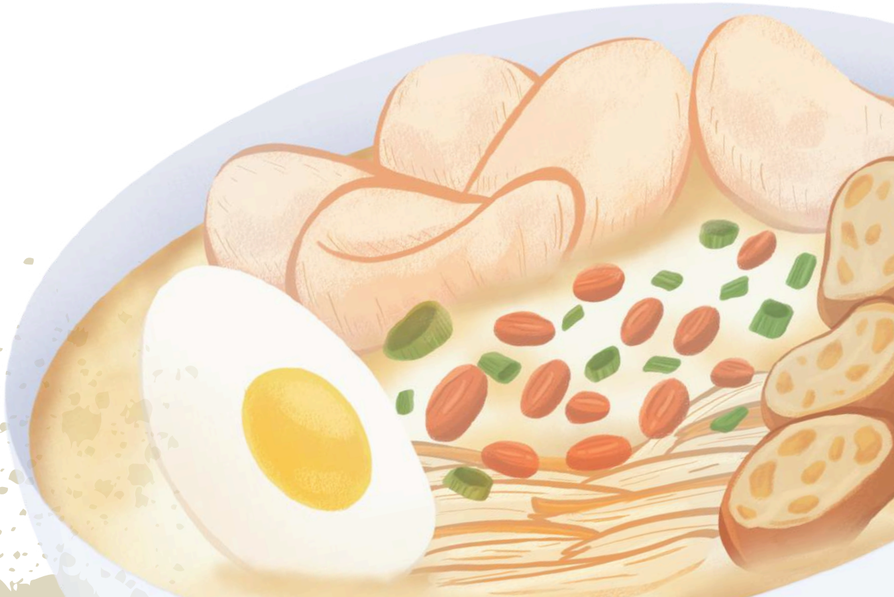
Analyzing sales data is crucial for any business to identify trends, improve operations, and enhance profitability. This project leverages SQL to analyze a pizza sales dataset, offering key insights into sales, revenue, and customer preferences.

Understanding sales trends, customer preferences, and revenue drivers is essential for any business. This project utilizes SQL queries to analyze a pizza sales dataset, uncover key insights, and support data-driven decision-making.

Business Requirements

The objective of this analysis is to:

- Identify the most and least popular pizza types.
- Analyze revenue trends and sales distribution.
- Optimize inventory and operational efficiency.
- Understand customer preferences based on order patterns.



Dataset Overview

The dataset consists of four key tables:

- orders: Contains order details with timestamps.
- orders_details: Includes the quantity and type of pizzas sold.
- pizzas: Provides size and pricing information.
- pizza_types: Describes pizza names and categories.

ANALYSIS & SQL QUERIES

The project is divided into three sections based on complexity: **Basic, Intermediate, and Advanced.**

Basic:

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

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

Result Grid

	total_orders
▶	21350

Basic: 2

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

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

Result Grid

total_sales
817860.05

Basic: 3

```sql
-- 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;
```

Result Grid | Filter R

name	price
The Greek Pizza	35.95

Basic: 4

```sql
-- 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;
```

Result Grid | Filt

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

Basic: 5

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

SELECT
    pizza_types.name, SUM(orders_details.quantity) AS quantity
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 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

Intermediate: 1

```sql
-- Find the total quantity of each pizza category ordered.

SELECT
    pizza_types.category,
    SUM(orders_details.quantity) AS quantity
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 quantity DESC;
```

Result Grid

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Intermediate: 2

```sql
-- 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;
```

Result Grid | Filte

hour	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

Intermediate: 3

```sql
-- Find the category-wise distribution of pizzas.

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

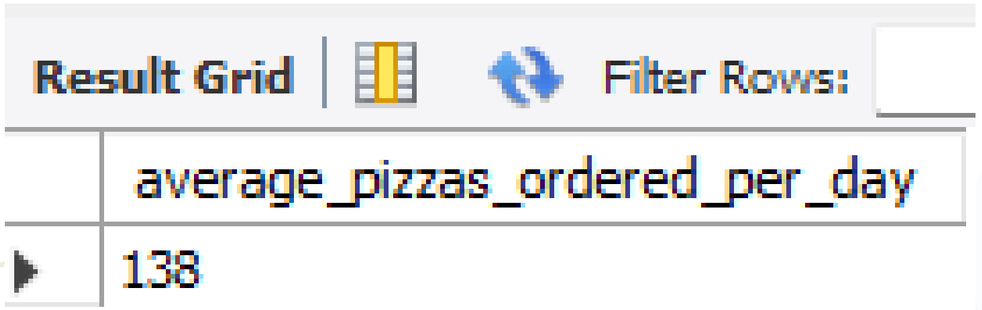
Result Grid | Filter Row:

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

Intermediate: 4

```sql
-- calculate the average number of pizzas ordered per day.

SELECT
    ROUND(AVG(quantity), 0) as average_pizzas_ordered_per_day
FROM
    (SELECT
        orders.order_date, SUM(orders_details.quantity) AS quantity
    FROM
        orders
    JOIN orders_details ON orders.order_id = orders_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```



Intermediate: 5

```sql
-- 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;
```

Result Grid | Filter Rows:

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

Advanced: 1

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

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

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Advanced: 2

```sql
-- Analyze the cumulative revenue generated over time.

SELECT order_date, revenue,
    SUM(revenue) OVER (ORDER BY order_date) AS com_revenue
FROM
    (SELECT orders.order_date,
        ROUND(SUM(orders_details.quantity * pizzas.price),0) AS revenue
        FROM
            orders_details
            JOIN
            pizzas ON orders_details.pizza_id = pizzas.pizza_id
            JOIN
            orders ON orders.order_id = orders_details.order_id
    GROUP BY orders.order_date) AS sales;
```

Result Grid | Filter Rows:

order_date	revenue	com_revenue
2015-01-01	2714	2714
2015-01-02	2732	5446
2015-01-03	2662	8108
2015-01-04	1755	9863
2015-01-05	2066	11929
2015-01-06	2429	14358
2015-01-07	2202	16560
2015-01-08	2838	19398
2015-01-09	2127	21525
2015-01-10	2464	23989
2015-01-11	1872	25861
2015-01-12	1919	27780
2015-01-13	2050	29830
2015-01-14	2527	32357
2015-01-15	1985	34342
2015-01-16	2594	36936
2015-01-17	2064	39000
2015-01-18	1977	40977
2015-01-19	2387	43364
2015-01-20	2398	45762
2015-01-21	2041	47803
2015-01-22	2497	50300
2015-01-23	2424	52724
2015-01-24	2289	55013
2015-01-25	1618	56631
2015-01-26	1884	58515
2015-01-27	2528	61043
2015-01-28	2016	63059
2015-01-29	2045	65104
2015-01-30	2270	67374
2015-01-31	2418	69792

Advanced: 3

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

SELECT category, name, revenue,Rnk
FROM
(   SELECT category, name, revenue,
    RANK() OVER(PARTITION BY category ORDER BY revenue DESC) AS Rnk
    FROM
    (   SELECT
            pizza_types.category,
            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.category , name) AS A) AS B
WHERE Rnk <=3;
```

Result Grid | Filter Rows: | Export: | Wra

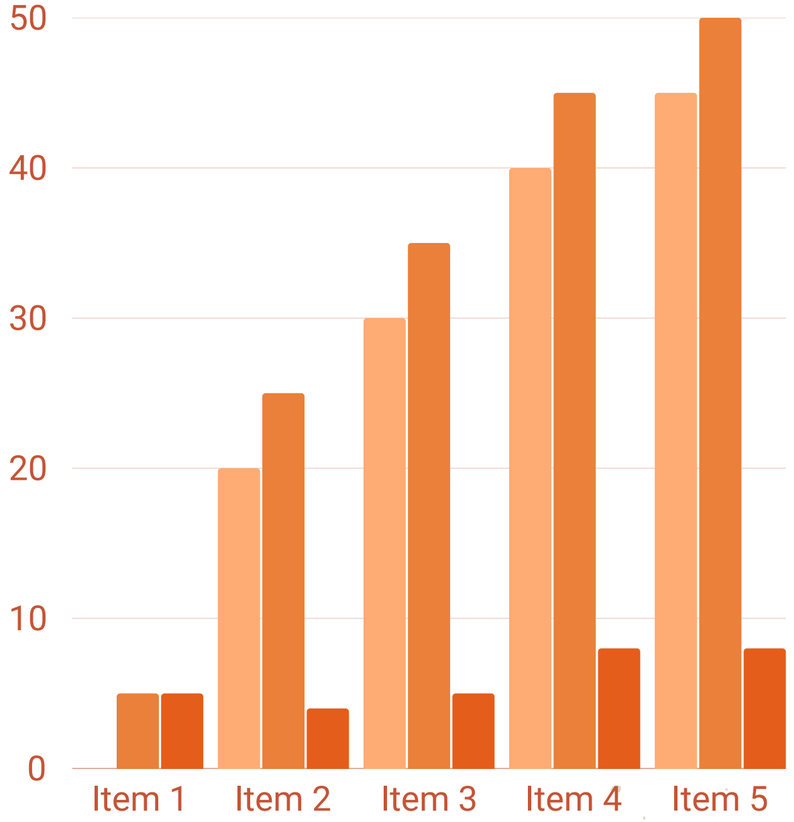
category	name	revenue	Rnk
Chicken	The Thai Chicken Pizza	43434.25	1
Chicken	The Barbecue Chicken Pizza	42768	2
Chicken	The California Chicken Pizza	41409.5	3
Classic	The Classic Deluxe Pizza	38180.5	1
Classic	The Hawaiian Pizza	32273.25	2
Classic	The Pepperoni Pizza	30161.75	3
Supreme	The Spicy Italian 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

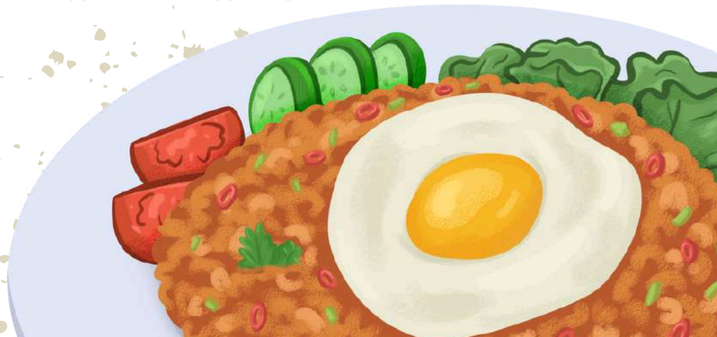
GOALS

- Gain insights into pizza sales trends.
- Determine peak ordering times.
- Optimize business strategies based on sales performance.
- Improve revenue forecasting and inventory planning.

RESULT

- Revenue Drivers: Identified the top-selling pizzas and the highest revenue-generating categories.
- Customer Preferences: Found the most common pizza sizes and peak ordering times.
- Operational Efficiency: Insights into demand trends help optimize inventory and staffing.





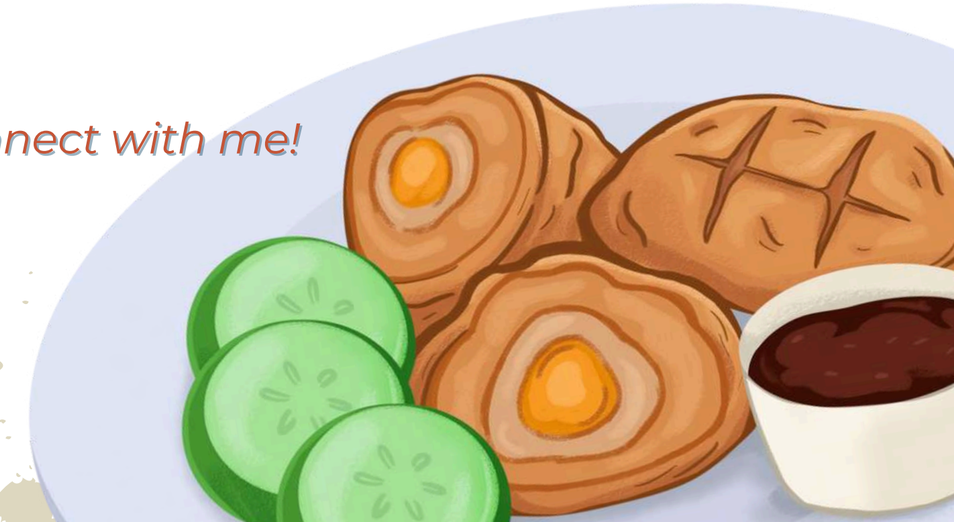
CONCLUSION

This analysis provides a comprehensive view of pizza sales trends, helping businesses make data-driven decisions. By leveraging SQL queries, we identified key revenue contributors, customer ordering behaviours, and operational efficiencies. The findings enable better resource allocation, improved forecasting, and enhanced business performance.

ABOUT ME

I am a data analyst with expertise in SQL, Excel, Power BI. Passionate about uncovering business insights through data-driven approaches, I specialize in transforming raw data into meaningful stories that drive strategic decisions.

For collaboration or discussion, feel free to connect with me!



DOES ANYONE HAVE QUESTIONS?

THANK YOU