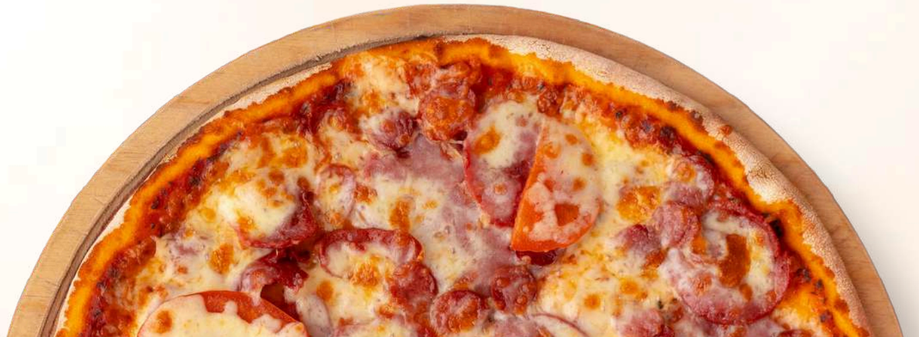
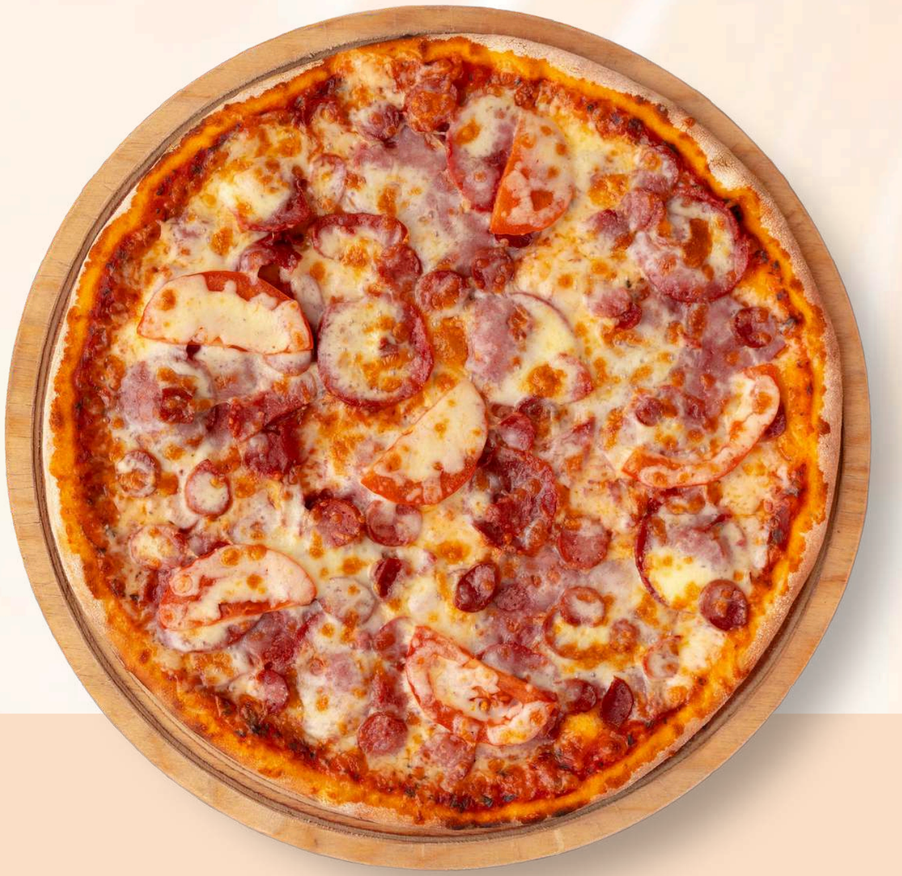
Pizza Sales Analysis

Using MySQL®



Introduction

- **Analyzed pizza sales data to extract meaningful business insights.**
- **Performed end-to-end analysis from basic metrics to advanced trends.**



Components Used

- Joins
- Aggregate Functions
- Window Functions
- Grouping Clause

Dataset Overview

Tables Involved

pizzas

pizza_types

orders

order_details

Key Fields

pizza_id, order_id, order_date,
size, category, quantity, price

Basic Analysis

Total Number of Orders Placed.....
Total Revenue Generated.....
Getting Highest Priced Pizza.....
Most Common Pizza Size.....
Top 5 Most Ordered Pizzas.....

Total Number of Orders Placed

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

total_orders
21350

Total Revenue Generated from Pizza Sales

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

Filter

	total_revenue
▸	817860.05

Highest Priced Pizza

```sql
SELECT
    name AS highest_priced_pizza, price
FROM
    pizzas
        INNER JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
ORDER BY price DESC
LIMIT 1;
```

lter Rows:

highest_priced_pizza	price
The Greek Pizza	35.95

Most Common Pizza Size Ordered

```sql
SELECT
    size, COUNT(quantity) AS pizzas_ordered
FROM
    order_details
        JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id
GROUP BY size
ORDER BY COUNT(quantity) DESC
LIMIT 3;
```

Filter Rows:

size	pizzas_ordered
L	18526
M	15385
S	14137

Top 5 most Ordered Pizza Types (with Quantities)

```sql
SELECT
    pizza_types.name AS most_ordered_pizza,
    SUM(order_details.quantity) AS ordered_quantity
FROM
    pizzas
        JOIN
    pizza_types ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY name
ORDER BY SUM(order_details.quantity) DESC
LIMIT 5;
```

Export: Wrap Cell Con

most_ordered_pizza	ordered_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 Analysis

Total quantity of each pizza category ordered
Distribution of orders by hour of the day
Category-wise distribution of pizzas
Average number of pizzas ordered per day
Top 3 most ordered pizza types by revenue

Total Quantity of Each Pizza Category Ordered

```sql
SELECT
    pizza_types.category AS Category,
    SUM(order_details.quantity) AS Quantity
FROM
    pizzas
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY category
ORDER BY SUM(quantity) DESC;
```

Filter Rows:

Category	Quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Hourly Distribution

```sql
SELECT
    HOUR(orders.order_time) AS Order_time,
    COUNT(DISTINCT orders.order_id) AS total_orders_placed,
    SUM(order_details.quantity) AS total_quantity
FROM
    orders
        JOIN
    order_details ON orders.order_id = order_details.order_id
GROUP BY HOUR(orders.order_time)
ORDER BY HOUR(orders.order_time);
```

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

Order_time	total_orders_placed	total_quantity
9	1	4
10	8	18
11	1231	2728
12	2520	6776
13	2455	6413
14	1472	3613
15	1468	3216
16	1920	4239
17	2336	5211
18	2399	5417
19	2009	4406
20	1642	3534
21	1198	2545
22	663	1386

Result 6

Category Wise Distribution

```sql
SELECT category,count(name) AS types_of_pizzas
FROM pizza_types
GROUP BY category;
```

Filter Rows:

category	types_of_pizzas
Chicken	6
Classic	8
Supreme	9
Veggie	9

Pizzas Ordered / Day

```sql
SELECT
        ROUND((COUNT(DISTINCT orders.order_id)) / COUNT(DISTINCT orders.order_date),
            0) AS orders_per_day,
        ROUND(SUM(order_details.quantity) / COUNT(DISTINCT orders.order_date),
            0) AS quantity_per_day
FROM
    orders
        JOIN
    order_details ON orders.order_id = order_details.order_id;
```

Expor

	orders_per_day	quantity_per_day
▸	60	138

Top 3 Most Ordered Pizza (Revenue)

```sql
SELECT
    pizza_types.name,
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS revenue
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3;
```

Export:

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

Advanced Analysis

% contribution of each pizza type .....
Cumulative revenue over time .....
Top 3 pizza types (by category).....

% Contribution of Each Pizza Category

```sql
SELECT
    pizza_types.category,
    ROUND(SUM(pizzas.price * order_details.quantity),
            2) AS revenue,
    ROUND((ROUND(SUM(pizzas.price * order_details.quantity),
                    2)) * 100 / (SELECT
                    ROUND(SUM(order_details.quantity * pizzas.price),
                                2) AS total_revenue
                FROM
                    pizzas
                        JOIN
                    order_details ON pizzas.pizza_id = order_details.pizza_id
                        JOIN
                    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id),
            2) AS percentage_revenue
FROM
    pizzas
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY category
ORDER BY percentage_revenue DESC;
```

ult Grid | Filter Rows: | Export: | Wrap C

category	revenue	percentage_revenue
Classic	220053.1	26.91
Supreme	208197	25.46
Chicken	195919.5	23.96
Veggie	193690.45	23.68

Cumulative Revenue Generated

```sql
SELECT order_date,
       Round(Sum(pizzas.price * order_details.quantity), 2)
       AS daily_revenue,
       Round(Sum(Sum(pizzas.price * order_details.quantity))
               OVER (
                 ORDER BY order_date), 2)
                 AS cum_revenue
FROM   order_details
       JOIN orders
         ON order_details.order_id = orders.order_id
       JOIN pizzas
         ON pizzas.pizza_id = order_details.pizza_id
GROUP  BY order_date;
```

sult Grid | Filter Rows: | Export: | Wrap C

order_date	daily_revenue	cum_revenue
2015-01-01	2713.85	2713.85
2015-01-02	2731.9	5445.75
2015-01-03	2662.4	8108.15
2015-01-04	1755.45	9863.6
2015-01-05	2065.95	11929.55
2015-01-06	2428.95	14358.5
2015-01-07	2202.2	16560.7
2015-01-08	2838.35	19399.05
2015-01-09	2127.35	21526.4
2015-01-10	2463.95	23990.35

Top 3 Most Ordered Pizza (Category Wise)

```sql
SELECT
  category, name, revenue, rn
FROM
  (SELECT
      category, name, revenue,
      RANK() OVER
      (PARTITION BY category ORDER BY revenue DESC) AS rn
    FROM
      (SELECT
          pizza_types.category AS category,
          pizza_types.name AS name,
          SUM(order_details.quantity * pizzas.price) AS revenue
        FROM pizzas
          JOIN order_details ON order_details.pizza_id = pizzas.pizza_id
          JOIN pizza_types ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        GROUP BY category,
          name) AS a) AS b
WHERE rn <= 3;
```

category	name	revenue
Chicken	The Thai Chicken Pizza	43434.25
Chicken	The Barbecue Chicken Pizza	42768
Chicken	The California Chicken Pizza	41409.5
Classic	The Classic Deluxe Pizza	38180.5
Classic	The Hawaiian Pizza	32273.25
Classic	The Pepperoni Pizza	30161.75
Supreme	The Spicy Italian Pizza	34831.25
Supreme	The Italian Supreme Pizza	33476.75
Supreme	The Sicilian Pizza	30940.5
Veggie	The Four Cheese Pizza	32265.70
Veggie	The Mexicana Pizza	26780.75
Veggie	The Five Cheese Pizza	26066.5

Conclusions

Skills Demonstrated

- Real-world SQL problem solving
- Data joining and aggregation
- Analytical thinking using queries
- Use of window functions for business analysis

Thank You!

Links

Github : https://github.com/KhumeshSonkar/Pizza-Sales-Report

LinkedIn : https://www.linkedin.com/in/khumeshsonkar2003