# PIZZA HUT SALES ANALYSIS USING SQL

2024

presented by
Kushagra Mishra

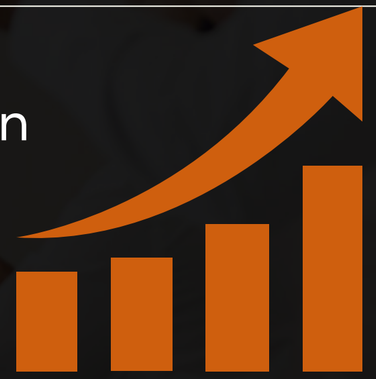
# INTRODUCTION

2024

This project involves a sales analysis for Pizza Hut using SQL to gain insights into sales performance and customer preferences.

The analysis is divided into basic, intermediate, and advanced queries.

In the Basic section, I retrieved key data, such as the total number of orders, revenue, and popular pizza types and sizes. The Intermediate level focused on deeper insights through SQL joins, including category distribution and order patterns by time. Finally, in the Advanced section, I analyzed cumulative revenue and calculated the contribution of each pizza type to overall sales.



This analysis provides valuable insights to optimize product offerings and improve customer satisfaction.

presented by
**Kushagra Mishra**

# DATA

2024

The data used for this project was gathered from an online source and provided in CSV format. The dataset contains detailed sales records for Pizza Hut, including order details, pizza types, prices, and timestamps.

# SCHEMA

2024

The schema consists of the following key tables:

1. Orders: contains order_id, date, time.
2. Pizzas: pizza_type_id, pizza_id, size, price.
3. Pizza_types: pizza_type_id, ingredients, category, name
4. Order_details: order_id, order_details_id, Pizza_id, quantity

This structured schema allowed for seamless querying and analysis, enabling insights into customer preferences, sales patterns, and revenue trends.

# QUESTIONS

2024

**Basic:**

1.Retrieve the total number of orders placed.
2.Calculate the total revenue generated from pizza sales.
3.Identify the highest-priced pizza.
4.Identify the most common pizza size ordered.
5.List the top 5 most ordered pizza types along with their quantities.

**Intermediate:**

6.Join the necessary tables to find the total quantity of each pizza category ordered.
7.Determine the distribution of orders by hour of the day.
8.Join relevant tables to find the category-wise distribution of pizzas.
9.Group the orders by date and calculate the average number of pizzas ordered per day.
10.Determine the top 3 most ordered pizza types based on revenue.

**Advanced:**

11.Calculate the percentage contribution of each pizza type to total revenue.
12.Analyze the cumulative revenue generated over time.
13.Determine the top 3 most ordered pizza types based on revenue for each pizza category.

# RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

2024

```sql
select count(order_id) as  total_order from orders;
```

Result Grid

| | total_order |
|---|---|
| ▶ | 21350 |

# CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

2024

```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;
```

Result Grid

| | total_sales |
|---|---|
| ▶ | 817860.05 |

# IDENTIFY THE HIGHEST-PRICED PIZZA.

2024

```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;
```

Result Grid | Filter Rows

| | name | price |
|---|---|---|
| ▶ | The Greek Pizza | 35.95 |

# IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

2024

```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 Rows:

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

# LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.

2024

```
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 |

# JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.

2024

```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 | Filter

| category | quantity |
|---|---|
| Classic | 14888 |
| Supreme | 11987 |
| Veggie | 11649 |
| Chicken | 11050 |

# DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

2024

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

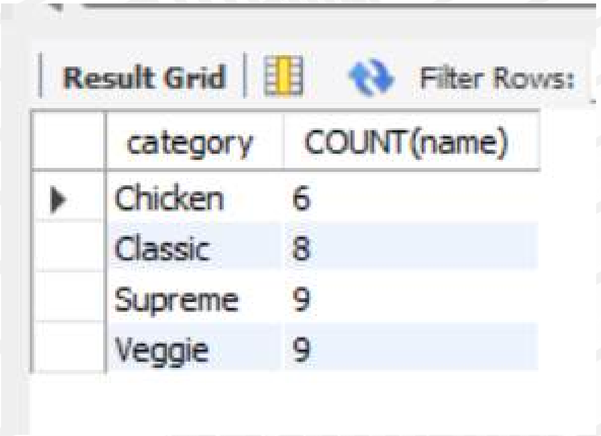
Result Grid | Filter

| hour | order_count |
|---|---|
| 11 | 1231 |
| 12 | 2520 |
| 13 | 2455 |
| 14 | 1472 |
| 15 | 1468 |
| 16 | 1920 |

# JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.

2024

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



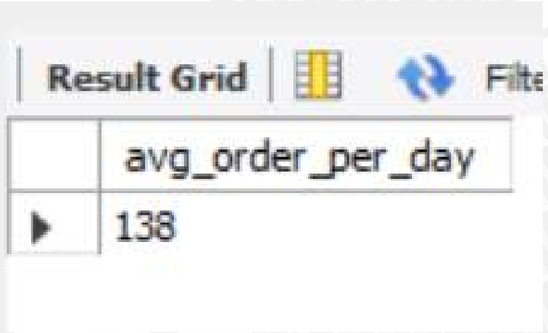
Result Grid | Filter Rows:

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

# GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

2024

```sql
SELECT
    ROUND(AVG(quantity), 0)as avg_order_per_day
FROM
    (SELECT
        orders.order_date,
            ROUND(SUM(order_details.quantity), 2) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```



# DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

2024

```sql
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) 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;
```

Result Grid | Filter Rows:

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

# CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE.

2024

```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 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.category
ORDER BY revenue DESC;
```

Result Grid | Filter R

| category | revenue |
|---|---|
| Classic | 26.91 |
| Supreme | 25.46 |
| Chicken | 23.96 |
| Veggie | 23.68 |

# ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

2024

```
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;
```

Result Grid | Filter Rows:

| order_date | cum_revenue |
|---|---|
| 2015-01-01 | 2713.850000000004 |
| 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 |

# DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY.

2024

```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: | Exp

| 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 |

2024

# INSIGHTS & RECOMMENDATIONS:

- **Popular Pizza Types & Sizes**: Regular and large-sized pizzas are the most frequently ordered.
- **Peak Ordering Times**: Sales peak during lunch (12-2 PM) and dinner (6-8 PM).
- **High-Revenue Contributors**: Specialty pizzas and larger sizes generate the most revenue.
- **Recommendations**: Focus on promoting high-revenue pizzas during peak times to increase sales.

2024

# CONCLUSION:

This project aimed to analyze Pizza Hut’s sales data to uncover key trends in customer preferences and revenue. SQL proved essential for extracting meaningful insights, driving data-driven decisions. Future analysis could explore customer demographics and seasonal sales patterns to further refine marketing strategies and enhance business performance.

+91 8718922643

kushagra20mishra@gmail.com

# THANK YOU!

Thank you for your attention to our sales analysis presentation. If you have any questions or would like to discuss the findings in more detail, please don't hesitate to reach out to me.