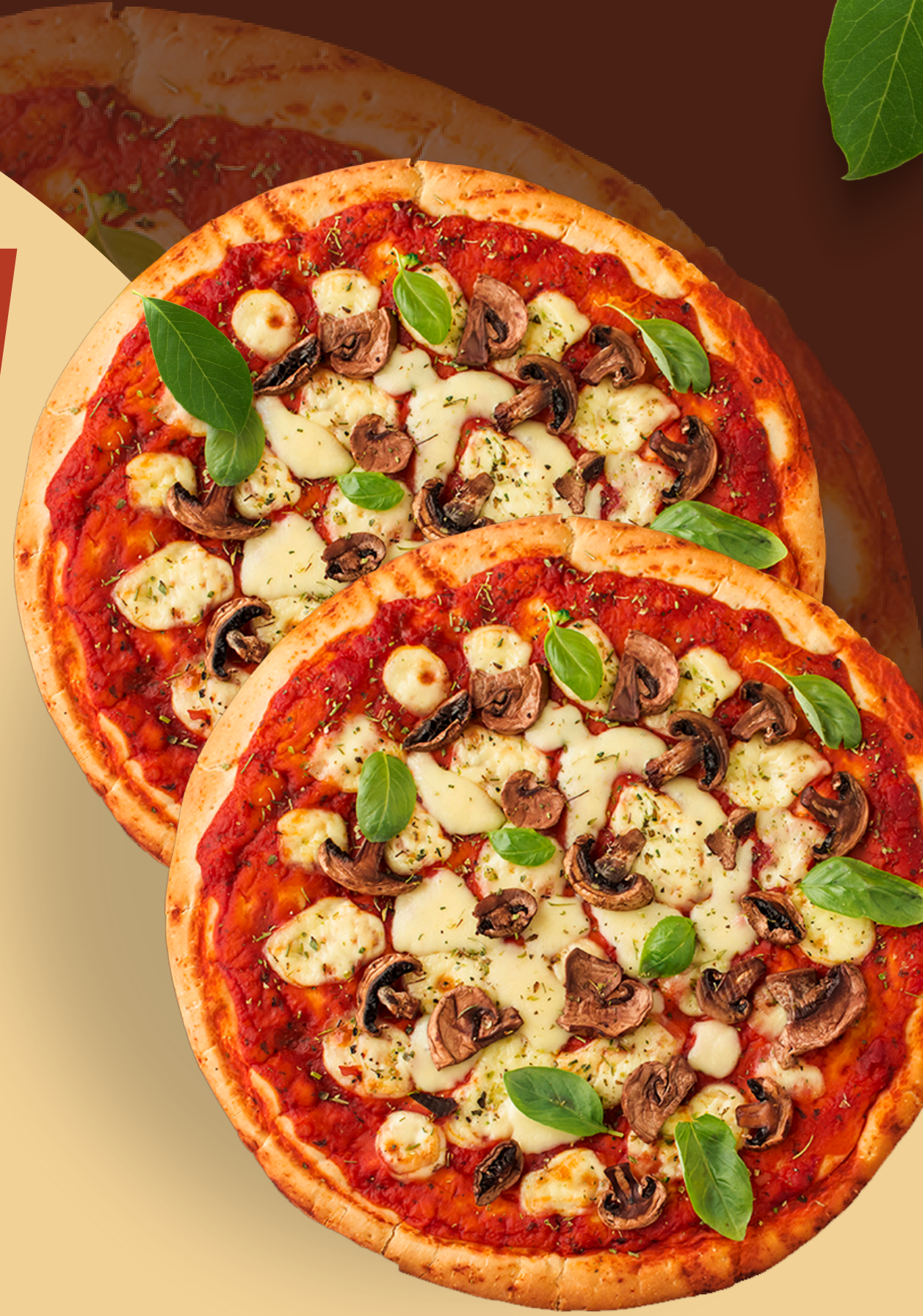
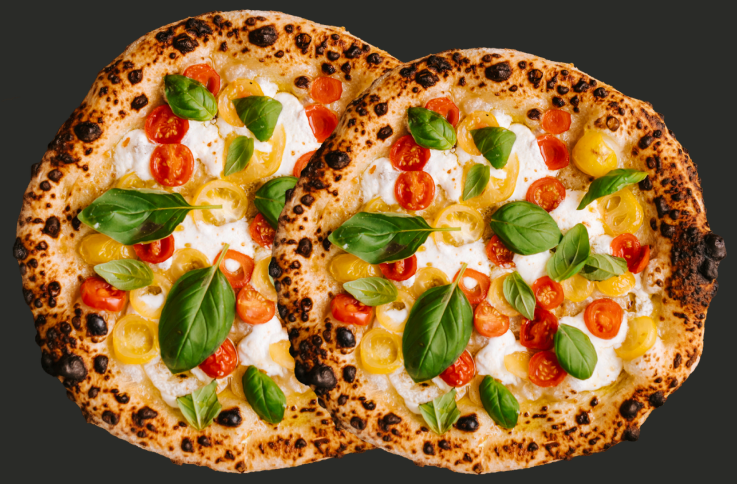
PIZZA

SALES

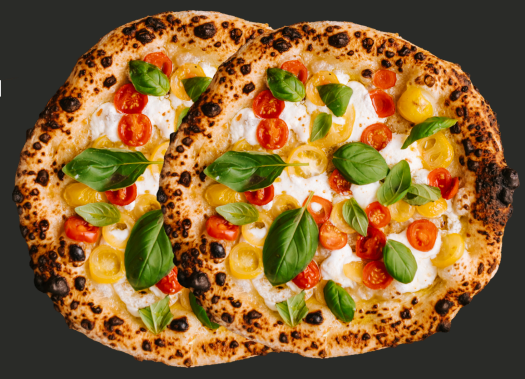


INTRODUCTION



Hello everyone, my name is Prince Yadav. In this project, I have utilized SQL queries to analyze and solve various questions related to pizza sales. Through this analysis, I aim to uncover key insights and trends within the data, which can help inform decision-making processes and optimize sales strategies. By leveraging the power of SQL, I have been able to efficiently extract, manipulate, and interpret the data, providing a comprehensive overview of the pizza sales dynamics.

OVERVIEW OF PROJECT



In this project, I utilized SQL queries to perform a comprehensive analysis of pizza sales data. The analysis is structured into three levels of complexity:

Basic Analysis:

Retrieve the total number of orders placed.
Calculate the total revenue generated from pizza sales.
Identify the highest-priced pizza.
Determine the most common pizza size ordered.
List the top 5 most ordered pizza types along with their quantities.
Intermediate Analysis:

Join the necessary tables to find the total quantity of each pizza category ordered.
Analyze the distribution of orders by hour of the day.
Find the category-wise distribution of pizzas.
Group orders by date and calculate the average number of pizzas ordered per day.
Determine the top 3 most ordered pizza types based on revenue.
Advanced Analysis:

Calculate the percentage contribution of each pizza type to total revenue.
Analyze the cumulative revenue generated over time.
Identify the top 3 most ordered pizza types based on revenue for each pizza category.
Through these analyses, the project aims to uncover key insights into sales trends, customer preferences, and revenue distribution, providing valuable data-driven recommendations for optimizing sales and marketing strategies.

PIZZA SALES

Retrieve the total number of orders placed.

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

Result Grid | Filter Rows:

total_order
21350

PIZZA SALES

Calculate the total revenue generated from pizza sales.

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

Result Grid | Filter Rows:

total_sales
817860.05

PIZZA SALES

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;
```

Result Grid | Filter Rows:

name	price
The Greek Pizza	35.95

PIZZA SALES

Identify the most common pizza size ordered.

```sql
SELECT
    pizzas.size, COUNT(details.order_details_id) AS order_count
FROM
    pizzas
        JOIN
    details ON pizzas.pizza_id = 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

SALES PIZZA

determine the distribution of orders by hours of the day.

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

Result Grid | Filter Rows:

hour	order_count
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663

PIZZA SALES

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

```sql
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

PIZZA SALES

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

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

Result Grid | Filter Rows:

avg_pizza_ordered_per_day
138

PIZZA SALES

Determine top 3 pizza types based on revenue

```sql
SELECT
    pizza_types.name,
    SUM(details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    details ON 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

PIZZA SALES

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

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

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

PIZZA SALES

Analyze the cumulative revenue generated over time.

```
select order_date,
sum(revenue) over(order by order_date) as cum_revenue
from
(select orders.order_date,
sum(details. quantity * pizzas . price) as revenue
from details join pizzas
on details.pizza_id = pizzas.pizza_id
join orders
on orders.order_id = details.order_id
group by orders.order_date) as sales ;
```

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
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526.4
2015-01-10	23990.350000000002

PIZZA SALES

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

```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((details.quantity) * pizzas.price)  as revenue
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
join details
on details . pizza_id = pizzas.pizza_id
group by pizza_types. category, pizza_types.name) as a ) as b
where rn <=3;
```

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	32265.70000000065