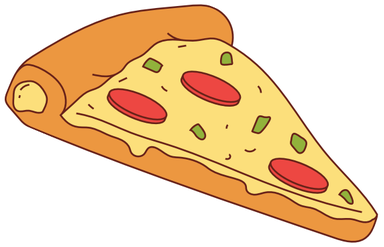
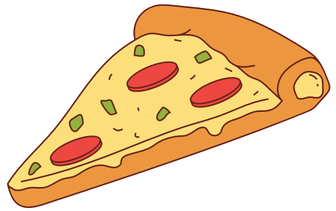
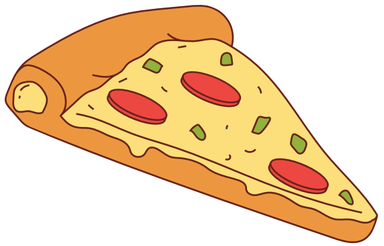
PIZZAHUT SALES ANALYSIS

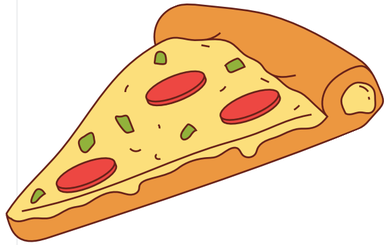
INTRODUCTION

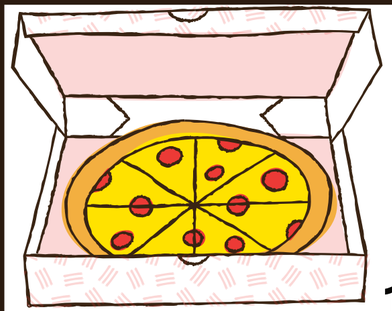


This presentation provides an overview of pizza sales analysis conducted for the PizzaHut project using SQL queries. The analysis focuses on key performance indicators such as total orders, revenue generation, and category-wise sales. It also highlights the top-selling and highest-priced pizzas, giving insight into which products drive the most revenue. By leveraging SQL queries, we extract valuable data to help guide decision-making, understand sales trends, and optimize pricing strategies. The following slides showcase the SQL queries used for this analysis, along with their results and interpretations.

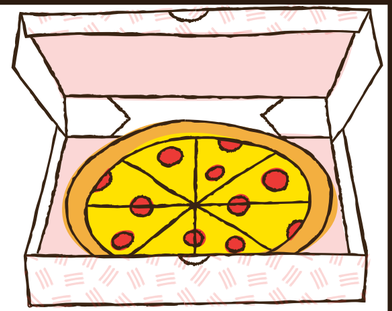








INDEX



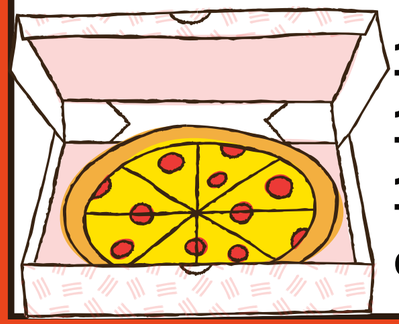
- **Basic:**

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.

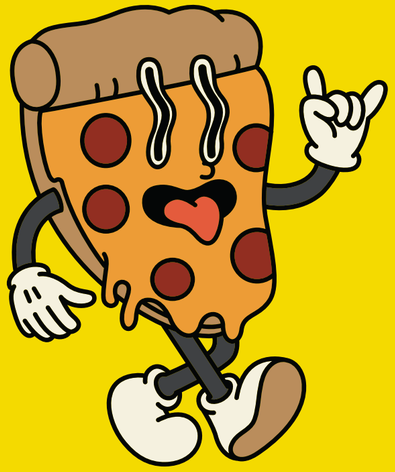
- **Advanced:**

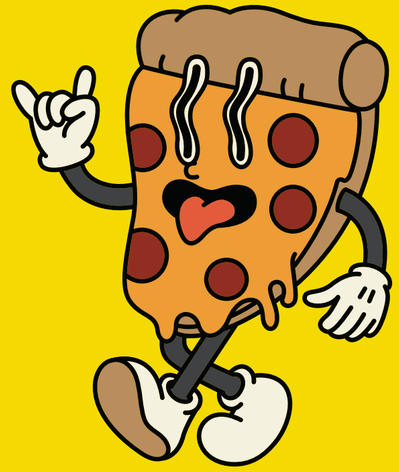


11.Calculate the percentage contribution of each pizza type to total revenue.
12.Analyze the cumulative revenue generated over time.
13.Determine the top 3 most ordered pizza types based on revenue for each pizza category.

1. Retrieve the total number of orders placed.

```
SELECT
    COUNT(Order_id) AS Total_Orders
FROM
    Orders;
```



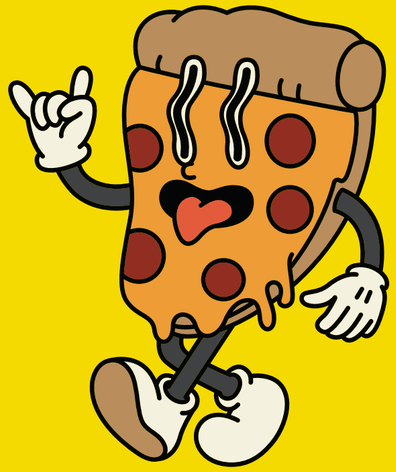


Result Grid

	Total_Orders
▶	21350

TOTAL ORDERS

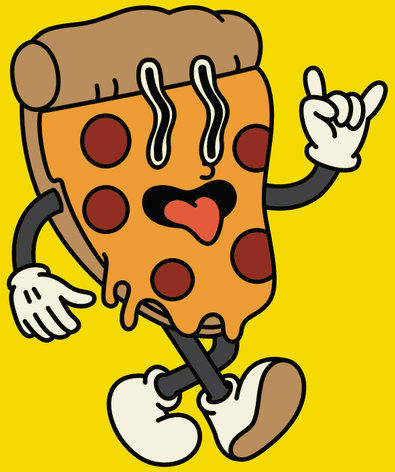
2. Calculate the total revenue generated from pizza sales.



```
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;
```

Result Grid

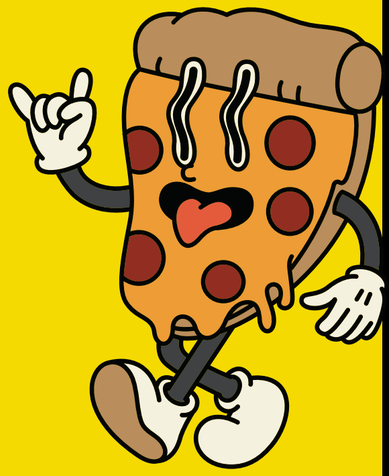
	Total_sales
▶	74208.45

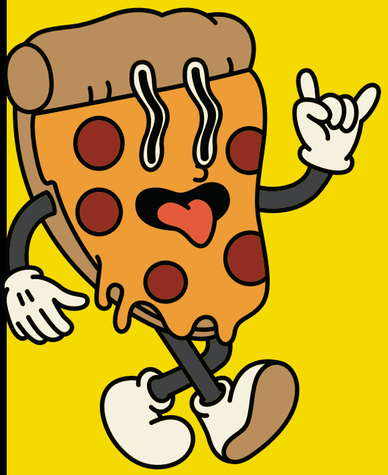


TOTAL SALES

3. 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;
```





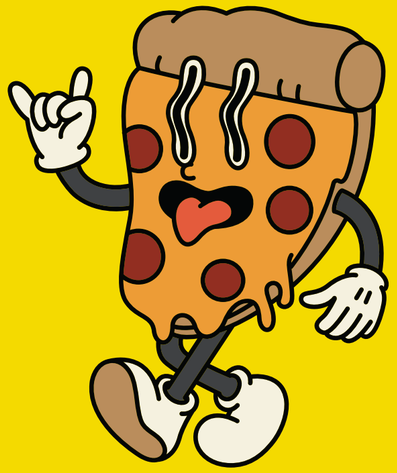
Result Grid | Filter Ro

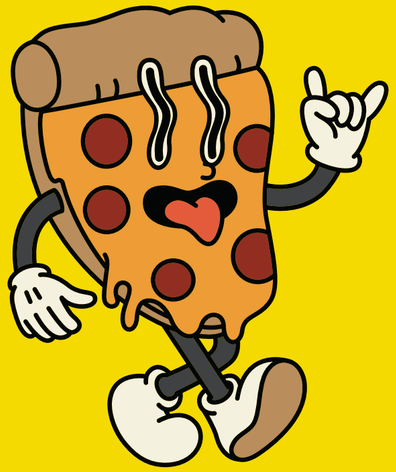
	name	price
▶	The Greek Pizza	35.95

Highest Priced Pizza

4. Identify the most common pizza size ordered.

```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
LIMIT 1;
```



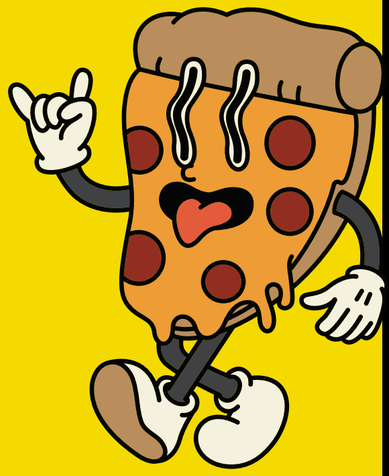


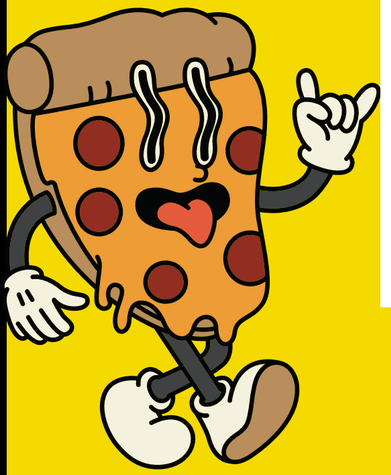
Result Grid | Filter

size	Order_count
L	1703

Most commom size pizza ordered

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





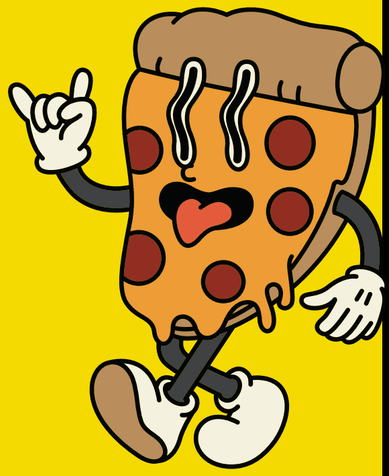
```
SELECT
    Pizza_Types.name, SUM(order_details.Quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.Pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC
LIMIT 5;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

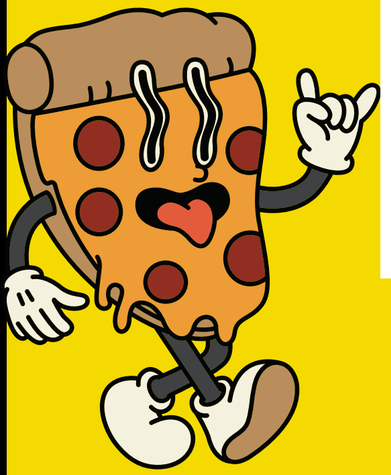
	name	quantity
▶	The Pepperoni Pizza	255
	The Barbecue Chicken Pizza	225
	The California Chicken Pizza	222
	The Thai Chicken Pizza	214
	The Classic Deluxe Pizza	204

Top 5 most ordered pizza

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

```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;
```



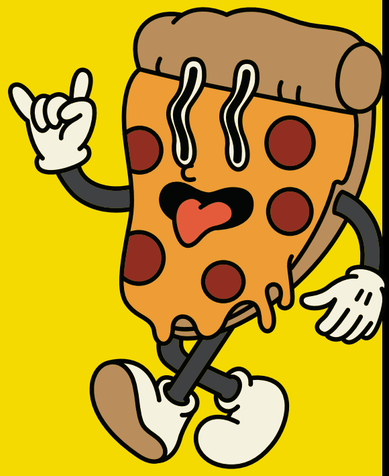


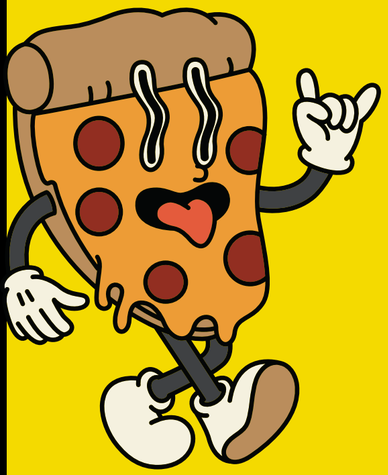
	category	Quantity
▶	Classic	1333
	Supreme	1111
	Veggie	1074
	Chicken	980

Total Quantity of each pizza ordered

7. Determine the distribution of orders by hour of the day.

```sql
SELECT
    HOUR(Order_time) AS Hour, COUNT(order_id) AS Order_Count
FROM
    orders
GROUP BY HOUR(Order_time);
```





Hour	Order_Count
11	1231
12	2520
13	2455
14	147
15	1468

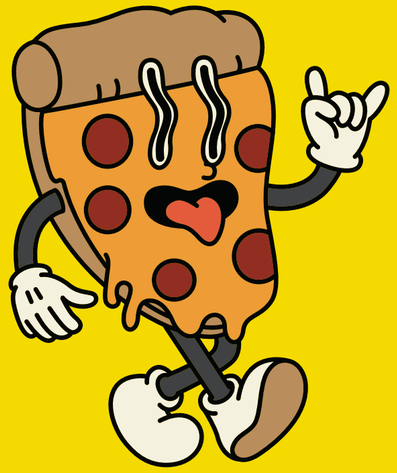
2455

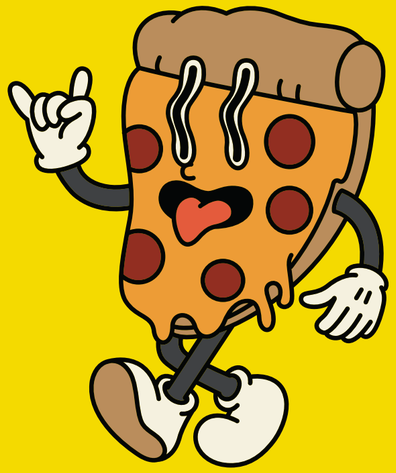
Distribution of Orders by hour of the Day

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

```
Select Category, count(name) from pizza_types
group by category;
```

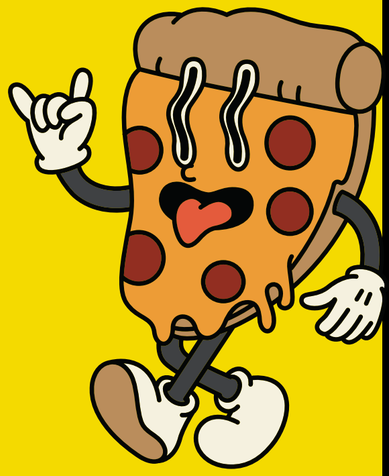
	Category	count(name)
▶	Chicken	6
	Class	Chicken
	Supreme	9
	Veggie	9



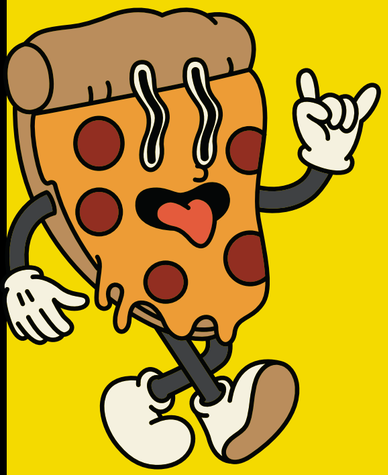


Category-wise Distribution of pizzas

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



```
SELECT
    ROUND(AVG(Avg_Pizza_Ordered_Perday), 0)
FROM
    (SELECT
        Orders.Order_date,
            SUM(order_details.Quantity) AS Avg_Pizza_Ordered_Perday
    FROM
        orders
    JOIN order_details ON orders.Order_id = order_details.Order_id
    GROUP BY Orders.Order_date) AS Order_Quantity;
```

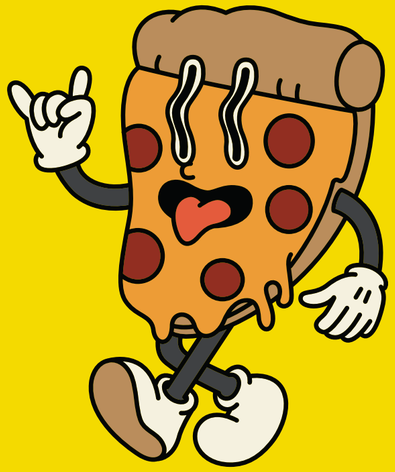


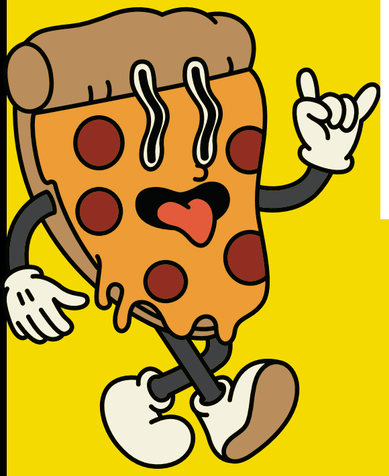
	ROUND(AVG(Avg_Pizza_Ordered_Perday), 0)
▶	136

Average number of pizzas ordered per day.

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

```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;
```



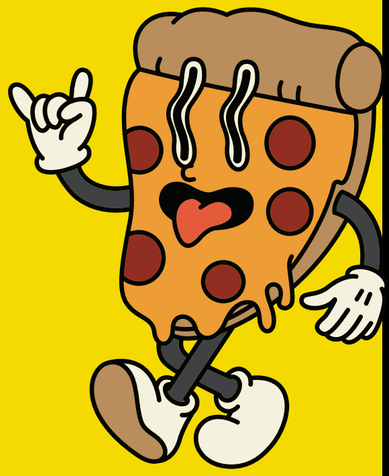


name	Revenue
The Barbecue Chicken Pizza	4020.75
The Thai Chicken Pizza	3936.5
The California Chicken Pizza	3822.5

Top 3 most ordered pizza types based on revenues

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

```sql
SELECT pt.category,
       ROUND(SUM(od.quantity * p.price) /
       (SELECT ROUND(SUM(od2.quantity * p2.price), 2) AS Total_sales
        FROM order_details od2
        JOIN pizzas p2 ON p2.pizza_id = od2.pizza_id) * 100, 2) AS Revenue
FROM pizza_types pt
JOIN pizzas p ON pt.pizza_type_id = p.pizza_type_id
JOIN order_details od ON p.pizza_id = od.pizza_id
GROUP BY pt.category
ORDER BY Revenue DESC
LIMIT 1000;
```



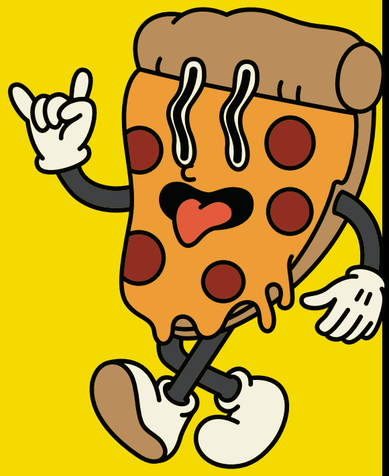
category	Revenue
Classic	26.56
Supreme	25.76
Veggie	23.68
Chicken	23.45

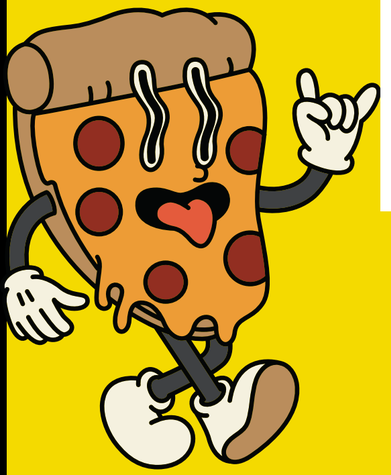
Supreme

Percentage Contribution by pizza type to total revenue

12. Analyze the cumulative revenue generated over time.

```sql
SELECT Sales.Order_Date,
       SUM(Sales.Revenue) OVER (ORDER BY Sales.Order_Date) AS Cum_Revenue
FROM (
    SELECT o.Order_Date,
           SUM(od.Quantity * p.price) 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;
```



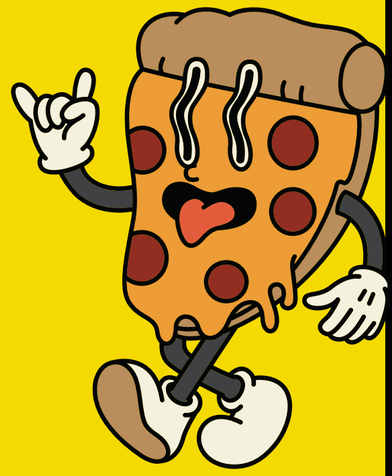


Order_Date	Cum_Revenue
2015-01-01	2713.850000000004
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55

Revenue generated over time

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

```sql
WITH RankedPizzas AS (
    SELECT pt.category, pt.name,
           SUM(od.Quantity * p.price) AS Revenue,
           RANK() OVER (PARTITION BY pt.category ORDER BY SUM(od.Quantity * p.price) DESC) AS RN
    FROM pizza_types pt
    JOIN pizzas p ON pt.pizza_type_id = p.pizza_type_id
    JOIN order_details od ON od.pizza_id = p.pizza_id
    GROUP BY pt.category, pt.name
)
SELECT category, name, Revenue
FROM RankedPizzas
WHERE RN <= 3;
```



category	name	Revenue
Chicken	The Barbecue Chicken Pizza	4020.75
Chicken	The Thai Chicken Pizza	3936.5
Chicken	The California Chicken Pizza	3822.5
Classic	The Pepperoni Pizza	3184.75
Classic	The Classic Deluxe Pizza	3158.5

Top 3 most pizza ordered for each category

CONCLUSION

This sales analysis for the PizzaHut project reveals key insights into product performance, customer preferences, and revenue drivers. By identifying top-selling pizzas and category-wise revenue, the analysis highlights opportunities to optimize pricing and enhance profitability. Leveraging SQL queries, we gained a clear understanding of sales trends and customer demand, providing a solid foundation for data-driven decisions. These insights will help improve marketing strategies, product offerings, and overall business growth.