



WELCOME TO

PIZZA HUT SALES PROJECT

This project focuses on analyzing Pizza Hut sales data using SQL to uncover key business insights. The dataset contains details such as order date, pizza category, pizza size, quantity sold, revenue, and customer details.

Queries Used: Joins, Window Functions, Aggregations, Group By, Order By, Subquerie





VISSION & MISSION

VISSION

To build a data-driven sales analysis system that helps Pizza Hut gain clear visibility into their sales performance, customer preferences, and market trends, enabling smarter decisions for growth and customer satisfaction.

MISSION

- To analyze Pizza Hut sales data using SQL and uncover meaningful insights.
- To provide actionable reports on revenue, product performance, and customer behavior.
- optimization, pricing strategies, and promotional offers.





PIZZA HUT CATEGORY







'CHICKEN'

'CLASSIC'

'SUPREME'
'VEGGIE'



QUESTION

- Retrieve the total number of orders placed.
- Calculate the total revenue generated from pizza sales.
- Identify the highest-priced pizza.
- Identify the most common pizza size ordered.
- List the top 5 most ordered pizza types along with their quantities.
- Join the necessary tables to find the total quantity of each pizza category ordered.
- Determine the distribution of orders by hour of the day.
- Join relevant tables to find the category-wise distribution of pizzas.
- Group the orders by date and calculate the average number of pizzas ordered per day.
- Determine the top 3 most ordered pizza types based on revenue.
- Calculate the percentage contribution of each pizza type to total revenue.
- Analyze the cumulative revenue generated over time.
- Determine the top 3 most ordered pizza types based on revenue for each pizza category.





USING SQL FOR INSIGHTS ANALYSIS

"Using SQL to analyze Pizza Hut sales data and deliver business-desired insights for better decision-making."

4 Types of data sets - pizzas, order, pizza_type, order_details









RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

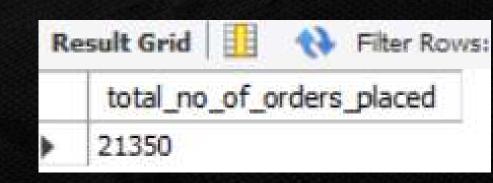
SELECT

COUNT(order_id) AS total_no_of_orders_placed

FROM

pizzahut.orders;











CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

SELECT

```
ROUND(SUM(pizzas.price * order_details.quantity),

2) AS total_revenue
```

FROM

```
order_details
JOIN
```

pizzas ON pizzas.pizza_id = order_details.pizza_id;









IDENTIFY THE HIGHEST-PRICED PIZZA.

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



	pizza_type_id	name	price
Þ	the_greek	The Greek Pizza	35.95







IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

```
pizzas.size, COUNT(order_details.order_details_id) AS orders_counts

FROM

pizzas

JOIN

order_details ON order_details.pizza_id = pizzas.pizza_id

GROUP BY pizzas.size

ORDER BY orders_counts DESC

LIMIT 1
```



	size	orders_counts	
b	L	18526	







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

```
SELECT

pizza_types.name,

SUM(order_details.quantity) AS total_quantity

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 total_quantity DESC

LIMIT 5;
```



	name	total_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.

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



	category	quantity
>	Classic	14888
	Veggie	11649
	Supreme	11987
	Chicken	11050







DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

SELECT

HOUR(order_time) AS hour, COUNT(order_id) A5 count_of_distribution

FROM

orders

GROUP BY hour;











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

SELECT

category, COUNT(pizza_type_id) AS counts

FROM

pizza_types

GROUP BY category;



category	counts
Chicken	6
Classic	8
Supreme	9
Veggie	9







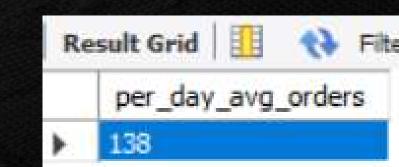
GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

```
SELECT
   ROUND(AVG(total_quantity), 0) AS per_day_avg_orders
FROM

(SELECT
          orders.order_date,
                SUM(order_details.quantity) AS total_quantity
FROM
          orders

JOIN order_details ON orders.order_id = order_details.order_id
GROUP BY orders.order_date) AS date_by_quantity_order;
```











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

```
SELECT

pizza_types.name,

SUM(pizzas.price * order_details.quantity) 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 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.

```
SELECT

pizza_types.category,

ROUND(SUM(pizzas.price * order_details.quantity) / (SELECT

ROUND(SUM(pizzas.price * order_details.quantity),

2)

FROM

order_details

JOIN

pizzas ON pizzas.pizza_id = order_details.pizza_id) * 100,

2) AS revenue_by_percentage

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 category;
```



category	revenue_by_percentage
Classic	26.91
Veggie	23.68
Supreme	25.46
Chicken	23.96







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(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 sale;
```

order_date	cum_revenue
2015-01-01	2713.85000000000004
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

