PROJECT

Home About Contact

PIZZA SALES

HELLO !

ABOUT THE PROJECT:

My name is Abhijeet Patel, and in this project, I have used SQL queries to analyze and solve various business problems related to pizza sales.

The goal of this project is to extract meaningful insights from the sales data — such as identifying top-performing pizzas, customer preferences, and sales trends — to help improve decision-making and business strategy.

DB_name : pizza_hut

SALES SCHEMA

I

Pizzas

pizza_id	TEXT
pizza_type	TEXT
size	TEXT
price	DOUBLE

II

Pizza_types

pizza_type_id	TEXT
name	TEXT
category	TEXT
ingredients	TEXT

III

Orders

order_id	int
order_date	date
order_time	time

IV

Order_details

order_details_id	int
order_id	int
pizza_id	text
quantity	int

BUSINESS PROBLEMS

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.

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.

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

```
SELECT
    COUNT(order_id)
FROM
    orders;
```

Result Grid | Filter Rows:

	COUNT(order_id)
▶	21350

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES

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

Result Grid | Filter Rows:

total_revenue
817860

IDENTIFY THE HIGHEST-PRICED PIZZA

```sql
SELECT
    pizza_types.name, 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

```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 size
ORDER BY order_count DESC
LIMIT 1;
```

Result Grid | Filter Rows:

size	order_count
L	18526

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

```sql
SELECT
    pizza_types.name,
    SUM(order_details.quantity) AS ordered_pizza
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 ordered_pizza DESC
LIMIT 5;
```

Result Grid | Filter Rows:

name	ordered_pizza
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

```sql
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS Total_Quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY category
```

Result Grid | Filter Rov

category	Total_Quantity
Classic	14888
Veggie	11649
Supreme	11987
Chicken	11050

DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY

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

Result Grid | Filter Rows:

hours	order_count
9	1
10	8
11	1231
12	2520
13	2455
14	1472
15	1468

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

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

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

Result Grid | Filter Rows:

	avg_quantity_perday
▶	138

DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE

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

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

Result Grid | Filter Rows

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME

```sql
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 orders join order_details
on orders.order_id = order_details.order_id
join pizzas on order_details.pizza_id = pizzas.pizza_id
group by orders.order_date) as sales;
```

Result Grid | Filter Rows:

order_date	cum_revenue
2015-01-01	2713.8500000000004
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

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((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:

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

THANK YOU!

This project gave me valuable hands-on experience in using SQL to solve real-world business problems. By analyzing pizza sales data, I learned how to write queries, manage table relationships, and generate meaningful insights from raw data.

Tools Used: MySQL Workbench, Canva

LET'S CONNECT

👨‍💻 Abhijeet Patel

📧 AbhijeetPatel4444@gmail.com

📱 Instagram: @CODEWITHABHIJEET