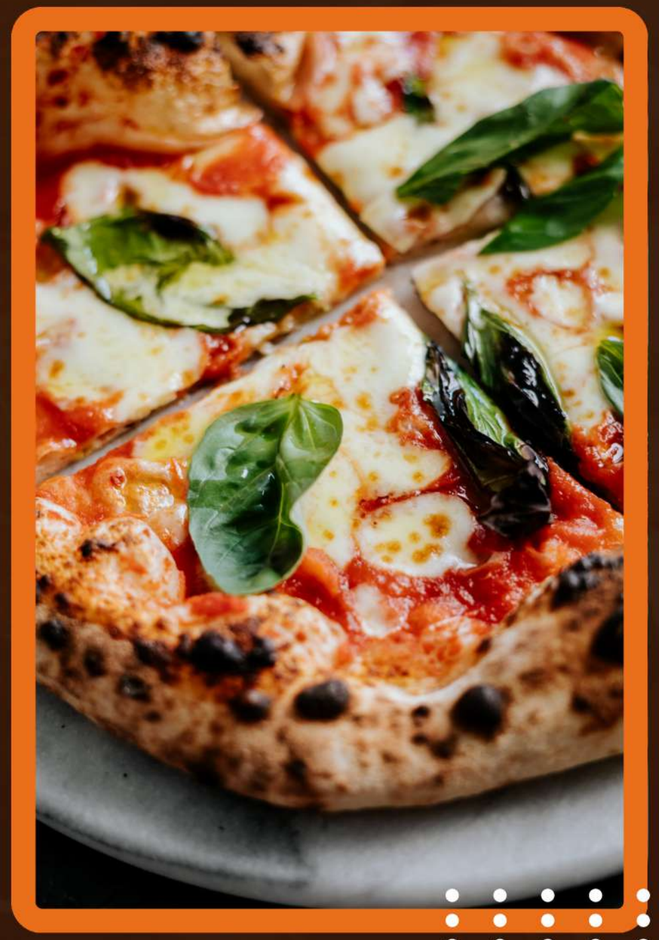
Pizza Sales Data Analysis using SQL

WELCOME TO PIZZAMANIC



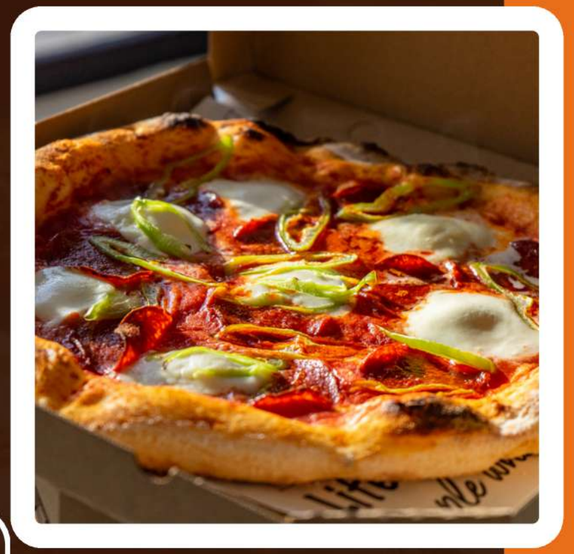
ABOUT THIS PROJECT

The dataset used in this project simulates a real-world pizza sales scenario with multiple interrelated tables, enabling the use of advanced SQL concepts like joins, groupings, aggregate functions, and filtering to answer relevant business questions.

DATASET OVERVIEW

1 PIZZAS

- pizza_id
- pizza_type_id
- size
- price



2 PIZZA_TYPES

- pizza_type_id
- name
- category
- ingredients

3 ORDERS

- order_id
- order_date
- order_time

4 ORDER_DETAILS

- order_id
- pizza_id
- quantity

1.RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

QUERY :

```
SELECT COUNT(order_id) AS Total_Orders_Placed
 FROM orders
```

RESULT:

Result Grid | Filter Rows

Total_Orders_Placed
21350

2. CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

--

QUERY :

```
USE pizzaHut;
SELECT ROUND(SUM((p.price * o.quantity)), 2) AS Revenue
FROM pizzas p
JOIN order_details o
ON p.pizza_id = o.pizza_id
```

RESULT:

Result Grid | Filter Rows

Revenue
817860.05

3. IDENTIFY THE HIGHEST-PRICED PIZZA.

QUERY :

SQL File 12* | SQL File 15* | SQL File 16* | SQL File 7

Limit to

```sql
USE pizzahut ;
SELECT pt.name,p.price
FROM pizzas p
JOIN pizza_types pt
ON p.pizza_type_id = pt.pizza_type_id
ORDER BY p.price DESC
LIMIT 1 ;
```

RESULT:

Result Grid | Filter Rows:

name	price
The Greek Pizza	35.95

4. IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

QUERY :

```
SELECT  p.size ,COUNT(*) AS Orders_placed
FROM order_details o
JOIN pizzas p
ON o.pizza_id=p.pizza_id
GROUP BY p.size
ORDER BY Orders_placed DESC
LIMIT 1 ;
```

RESULT:

Result Grid | Filter Rows

size	Orders_placed
L	18526

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

QUERY :

```
SELECT pt.name , COUNT(o.quantity) AS Quantity
FROM pizza_types pt
JOIN pizzas p
ON pt.pizza_type_id = p.pizza_type_id
JOIN order_details o
ON p.pizza_id=o.pizza_id
GROUP BY pt.name
ORDER BY Quantity DESC
LIMIT 5 ;
```

RESULT:

esult Grid | Filter Rows:

name	Quantity
The Classic Deluxe Pizza	2416
The Barbecue Chicken Pizza	2372
The Hawaiian Pizza	2370
The Pepperoni Pizza	2369
The Thai Chicken Pizza	2315

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

QUERY :

```
SELECT pt.category , SUM(o.quantity) AS Quantity_Ordered
FROM order_details o
JOIN pizzas p
ON o.pizza_id = p.pizza_id
JOIN pizza_types pt
ON p.pizza_type_id= pt.pizza_type_id
GROUP BY pt.category ;
```

RESULT:

Result Grid | Filter Rows:

category	Quantity_Ordered
Classic	14888
Veggie	11649
Supreme	11987
Chicken	11050

7. DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY. --

QUERY :

```
SELECT HOUR(order_time) AS HOUR ,COUNT(order_id) AS ORDERS_PLACED
FROM orders
GROUP BY HOUR(order_time);
```

RESULT:

HOUR	ORDERS_PLACED
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920

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

QUERY :

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

RESULT:

Result Grid | Filter

category	COUNT(*)
Chicken	6
Classic	8
Supreme	9
Veggie	9

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

QUERY :

```
SELECT ROUND(AVG(Quantity),0) AS AVG_Orders  FROM
(SELECT o.order_date , SUM(quantity) AS Quantity
FROM orders o
JOIN order_details od
ON o.order_id = od.order_id
GROUP BY o.order_date ) AS Order_Sales;
```

RESULT:

Result Grid

	AVG_Orders
▶	138

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

QUERY :

```sql
SELECT pt.name ,SUM(p.price*od.quantity) 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.name
ORDER BY REVENUE DESC
LIMIT 3 ;
```

RESULT:

Result Grid | Filter Rows:

name	REVENUE
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

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

QUERY :

```sql
SELECT Pizza_Type ,ROUND((Pizza_Revenue/SUM(Pizza_Revenue) OVER ()*100),2) AS Percentage
FROM
(SELECT pt.category AS Pizza_Type,SUM(p.price*od.quantity) AS Pizza_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 ) AS REVENUE_TABLE
ORDER BY Percentage DESC ;
```

RESULT:

Result Grid | Filter Row

Pizza_Type	Percentage
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

12. ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

QUERY :

```
SELECT order_date , SUM(revenue) OVER(ORDER BY order_date) AS Cumulative_Revenue
FROM
(SELECT o.order_date,
SUM(p.price*od.quantity) as Revenue
FROM orders o
JOIN order_details od
ON o.order_id = od.order_id
JOIN pizzas p
ON od.pizza_id = p.pizza_id
GROUP BY o.order_date) AS Sales
```

RESULT:

Result Grid | Filter Rows:

order_date	Cumulative_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

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

QUERY :

```sql
SELECT category , name , Revenue
FROM
(SELECT category ,name ,Revenue,
RANK() OVER(PARTITION BY category ORDER BY Revenue DESC) AS Ran
FROM
(SELECT pt.category ,pt.name,
SUM(p.price*od.quantity) as Revenue
FROM pizza_types as pt
JOIN pizzas as 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,pt.name) AS A)AS B
WHERE Ran<=3;
```

RESULT:

Result Grid | Filter Rows:

category	name	Revenue
Chicken	The Thai Chicken Pizza	43434.25
Chicken	The Barbecue Chicken Pizza	42768
Chicken	The California Chicken Pizza	41409.5
Classic	The Classic Deluxe Pizza	38180.5
Classic	The Hawaiian Pizza	32273.25
Classic	The Pepperoni Pizza	30161.75

THANK YOU

ANY QUERIES PLEASE CONTACT :

avantichinchone276@gmail.com