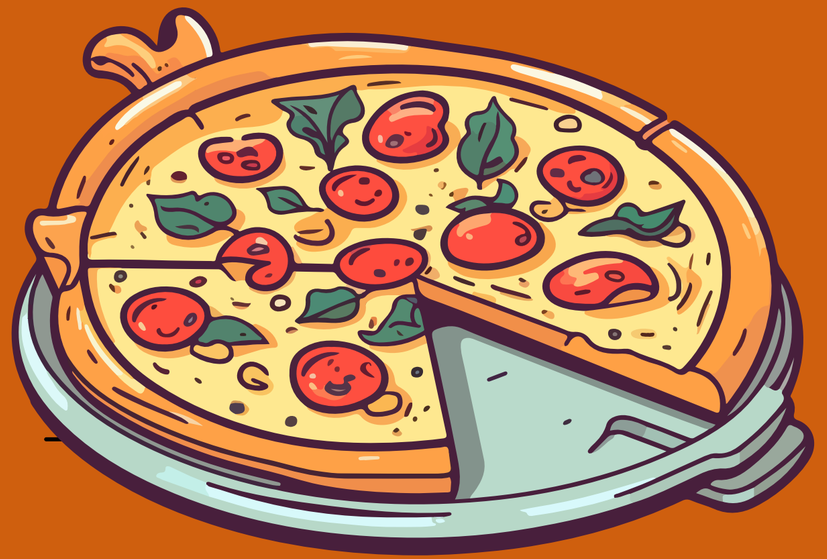
PIZZA SALES ANALYSIS USING SQL



UNCOVERING INSIGHTS FROM SALES DATA

presented by
Ishant Katiyar

INTRODUCTION

For this project, I analyzed pizza sales data using SQL to uncover key business insights. The goal was to understand sales trends, customer preferences, and revenue patterns to help optimize business decisions.

presented by
Ishant Katiyar

DATABASE STRUCTURE

ORDERS

- order_id (Primary Key)
- order_date
- order_time

ORDER_DETAILS

- order_details_id (Primary Key)
- pizza_id (Foreign Key from Pizzas)
- order_id (Foreign Key from Orders)
- quantity

PIZZAS

- pizza_id (Primary Key)
- pizza_type_id (Foreign Key from Pizza_Types)
- size
- price

PIZZA_TYPES

- pizza_type_id (Primary Key)
- name
- category
- ingredients

QUERY - CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

```
1 •  SELECT
2        ROUND(SUM(pizzas.price * orders_details.quantity),
3                2) AS total_sales
4    FROM
5        pizzas
6            JOIN
7        orders_details ON orders_details.pizza_id = pizzas.pizza_id
```

Result Grid

	total_sales
▶	817860.05

QUERY - IDENTIFY THE HIGHEST-PRICED PIZZA.

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

Result Grid | Filter Rows:

name	max_price_pizza
The Greek Pizza	35.95

QUERY - IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

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

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

```sql
SELECT
    pizza_types.name,
    SUM(orders_details.quantity) AS total_quantity
FROM pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    orders_details ON pizzas.pizza_id = orders_details.pizza_id
GROUP BY pizza_types.name
ORDER BY total_quantity DESC LIMIT 5;
```

Result Grid | Filter Rows: | Export:

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

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

```
Limit to 1000 rows

1  SELECT
2      pizza_types.category,
3      SUM(orders_details.quantity) AS total_quantity
4  FROM pizza_types
5          JOIN
6      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
7          JOIN
8      orders_details ON pizzas.pizza_id = orders_details.pizza_id
9  GROUP BY pizza_types.category
10 ORDER BY total_quantity DESC;
```

Result Grid | Filter Rows:

category	total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

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

```
Limit to 1000 rows

1 •  SELECT
2        HOUR(order_time) AS hour, COUNT(order_id) AS order_count
3    FROM
4        orders
5    GROUP BY hour
```

Result Grid | Filter Rows:

hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336

Result 2

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

```
Limit to 1000 rows

1  SELECT ROUND(AVG(total)) AS avg_pizzas_per_day
2  FROM
3      (SELECT
4          orders.order_date, COUNT(orders_details.quantity) AS total
5      FROM orders
6      JOIN orders_details ON orders.order_id = orders_details.order_id
7      GROUP BY orders.order_date) AS daily_counts;
```

Result Grid | Filter Rows:

avg_pizzas_per_day
136

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

```sql
SELECT pizza_types.name,
    SUM(pizzas.price * orders_details.quantity) AS revenue
FROM pizzahut.pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    orders_details ON pizzas.pizza_id = orders_details.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC LIMIT 3;
```

Result Grid | Filter Rows: | Export:

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

QUERY - CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH CATEGORY TO TOTAL REVENUE.

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

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

category	revenue	percentage_contribution
Classic	220053.1000000001	26.91
Supreme	208196.99999999822	25.46
Chicken	195919.5	23.96
Veggie	193690.45000000298	23.68

QUERY - ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

```sql
SELECT order_date, total_price,
       SUM(total_price) OVER (ORDER BY order_date) AS cumulative_sum
FROM (
    SELECT orders.order_date,
        round(SUM(orders_details.quantity * pizzas.price),2) AS total_price
    FROM pizzahut.orders
    JOIN orders_details ON orders.order_id = orders_details.order_id
    JOIN pizzas ON orders_details.pizza_id = pizzas.pizza_id
    GROUP BY orders.order_date) AS subquery;
```

Result Grid | Filter Rows: | Export: | Wrap

order_date	total_price	cumulative_sum
2015-01-01	2713.85	2713.85
2015-01-02	2731.9	5445.75
2015-01-03	2662.4	8108.15
2015-01-04	1755.45	9863.6
2015-01-05	2065.95	11929.55
2015-01-06	2428.95	14358.5
2015-01-07	2202.2	16560.7

Result 1

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

```
Limit to 1000 rows

1  select category, name, revenue from
2  (select category, name, revenue,
3  rank() over(partition by category order by revenue desc) as rn
4  from
5  (select pizza_types.category, pizza_types.name,
6  sum(orders_details.quantity * pizzas.price) as revenue
7  from pizza_types join pizzas
8  on pizza_types.pizza_type_id = pizzas.pizza_type_id
9  join orders_details on orders_details.pizza_id = pizzas.pizza_id
10 group by pizza_types.category, pizza_types.name) as a) as b
11 where rn<=3
```

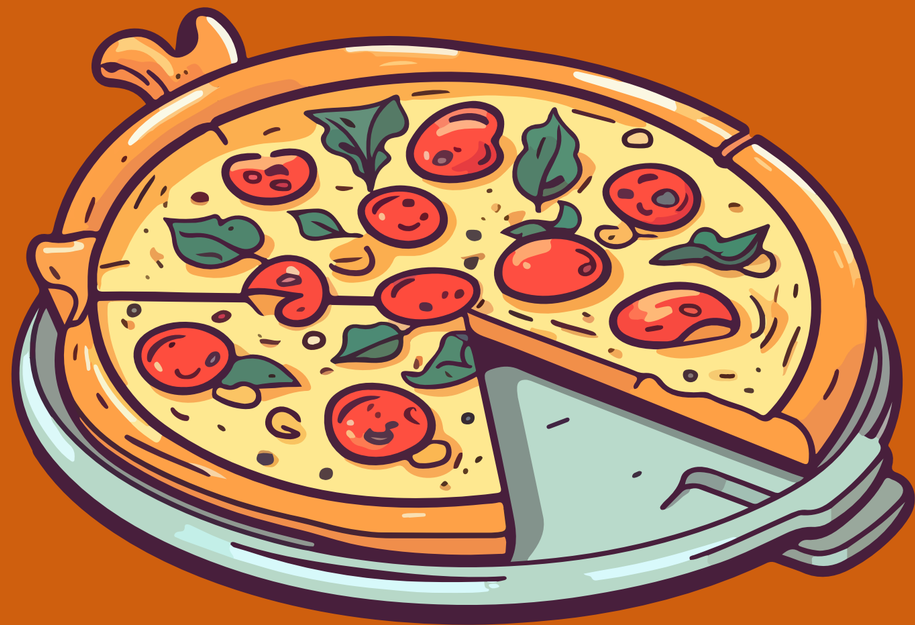
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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
Supreme	The Spicy Italian Pizza	34831.25

ishantkatiyar68@gmail.com

THANK YOU

Thank you for reviewing my project! I would love to hear your feedback.



LINKEDIN -
https://www.linkedin.com/in/ishantkatiyar/