About Contact

PIZZA SALES DATA ANALYSIS PORTFOLIO PROJECT

BY NITIN SHELLE

PROJECT OVERVIEW

This project showcases the use of SQL to analyze pizza sales data by importing raw data into a database and writing queries to uncover key business insights. Through joining tables and applying aggregations, I identified top-selling pizzas, popular sizes, and revenue trends. The queries helped transform data into actionable information to support better sales and menu decisions. To illustrate my approach, I've included screenshots of the key SQL queries I wrote and their results, highlighting practical skills in real-world data analysis.

1. RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

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

Result Grid

	Total_orders
▶	99441

CALCULATE THE TOTAL REVENUE(SALES) GENERATED FROM PIZZA SALES.

```
SELECT
    ROUND(SUM(orders_details.quantity * pizzas.price),
            2) AS total_sales
FROM
    orders_details
        JOIN
    pizzas ON orders_details.pizza_id = pizzas.pizza_id;
```

Result Grid

	total_sales
▶	817860.05

IDENTIFY THE HIGHEST-PRICED PIZZA


```sql
SELECT
    pizza_types.name, pizzas.price
FROM
    pizzas
        JOIN
    pizza_types 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(orders_details.order_detail_id)
FROM
    orders_details
        JOIN
    pizzas ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizzas.size
ORDER BY COUNT(order_detail_id) DESC;
```

Result Grid | Filter Rows:

size	Count_Orders
L	18526
M	15385
S	14137
XL	544
XXL	28

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

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

Result Grid | Filter Rows:

name	Most_Orders_Types
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(orders_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category;
```

Result Grid | Filter Rows:

category	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 HOUR(order_time);
```

Result Grid | Filter Rows:

hours	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

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

```
SELECT
    category, COUNT(name) as Categorywise_Distribution
FROM
    pizza_types
GROUP BY category;
```

Result Grid | Filter Rows:

category	Categorywise_Distribution
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_pizza_ordered
FROM
    (SELECT
        orders.order_date, SUM(orders_details.quantity) AS quantity
    FROM
        orders
    JOIN orders_details ON orders.order_id = orders_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

Result Grid | Filter Rows:

avg_pizza_ordered
138

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

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

Result Grid | Filter Rows:

pizza_name	revenue
The Thai Chicken Pizza	43434
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41410

CALCULATE THE PERCENTAGE CONTRIBUTION(EACH PIZZA TYPE'S CONTRIBUTION IN PERCENT) OF EACH PIZZA TYPE TO TOTAL REVENUE.

```
select pizza_types.category as pizza_type,
round((sum(pizzas.price*orders_details.quantity) / (SELECT
    ROUND(SUM(orders_details.quantity * pizzas.price),
            2) AS total_sales
FROM
    orders_details
        JOIN
    pizzas ON orders_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 orders_details on orders_details.pizza_id = pizzas.pizza_id
group by pizza_type order by revenue desc;
```

Result Grid | Filter Rows:

pizza_type	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 cumulative_revenue
from
(select orders.order_date,
round(sum(orders_details.quantity*pizzas.price), 2) as revenue
from orders_details join pizzas
on orders_details.pizza_id = pizzas.pizza_id
join orders on orders.order_id = orders_details.order_id
group by orders.order_date) as sales;
```

Result Grid | Filter Rows:

order_date	cumulative_revenue
2015-04-07	222146.19999999998
2015-04-08	224440.15
2015-04-09	226487.44999999998
2015-04-10	228912.4
2015-04-11	231456.15
2015-04-12	233450.44999999998
2015-04-13	235946.65
2015-04-14	238452.35
2015-04-15	241031.2

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

```
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(orders_details.quantity * pizzas.price) as revenue
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
join orders_details
on orders_details.pizza_id = pizzas.pizza_id
group by pizza_types.category, pizza_types.name) as sales) as sales_data
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
The Spicy Italian Pizza	34831.25
The Italian Supreme Pizza	33476.75
The Sicilian Pizza	30940.5
The Four Cheese Pizza	32265.70000000065
The Mexicana Pizza	26780.75
The Five Cheese Pizza	26066.5

CONCLUSION

This project provided valuable hands-on experience in applying SQL to real business data. By importing and querying the pizza sales dataset, I extracted insights such as top-selling products, customer preferences, and revenue patterns. Adding a variety of queries—from simple filters to complex joins and aggregations—not only enabled me to answer specific business questions but also deepened my understanding of data relationships and analytical methods. Including these queries, along with their results, highlights my practical skills in translating raw data into actionable strategies. Overall, this project demonstrates my ability to leverage SQL queries for data-driven decision-making and lays a strong foundation for further analytics work

THANK YOU

FOR ATTENTION

- CREATED AND ANALYZED BY NITIN SHELLE | DATA ANALYST