PIZZA SALES ANALYSIS

Meghna Chatterjee

28 May, 2024

INTRODUCTION

Hi, I am Meghna Chatterjee!

In this Project, I have utilized MySQL to solve queries related to PIzza Sales. Please go through these slides.

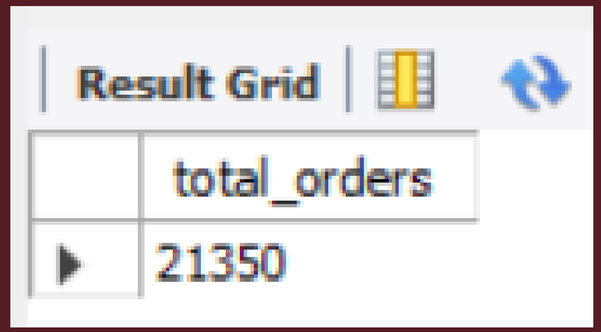
PROJECT OVERVIEW

In this project, we delve into the realm of pizza sales data analysis, aiming to unravel insights vital for optimizing business strategies in a pizza restaurant. By dissecting various facets of the dataset, we aim to empower decision-making processes, enhance customer experiences, and boost overall revenue generation. Through a multi-tiered analysis approach, we navigate from fundamental metrics like total orders and revenue to more intricate examinations such as hourly order distributions and revenue contributions of different pizza types. With these insights, we can tailor offerings, streamline operations, and ultimately foster growth in the competitive pizza market landscape.

1) RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

Input Command

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



Total no. of Orders placed

2) CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

Input Command

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

Result Grid

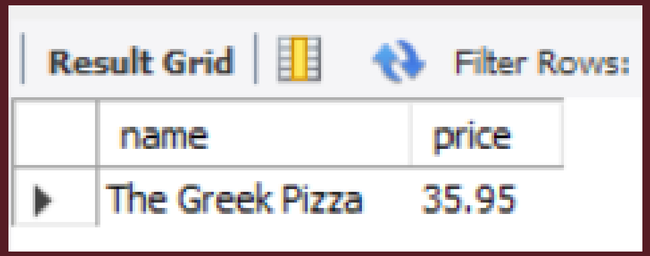
	total_sales
▶	817860.05

Total Revenue generated from Pizza Sales

3) IDENTIFY THE HIGHEST-PRICED PIZZA.

Input Command

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



The Highest-priced pizza

4) IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

Input Command

```
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 pizzas.size
ORDER BY order_count DESC;
```

Result Grid | Filter Rows

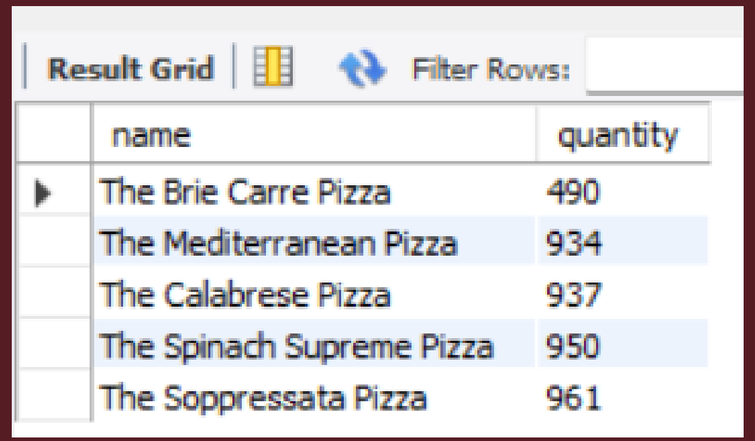
size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

The most Common Pizza size Ordered

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

Input Command

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



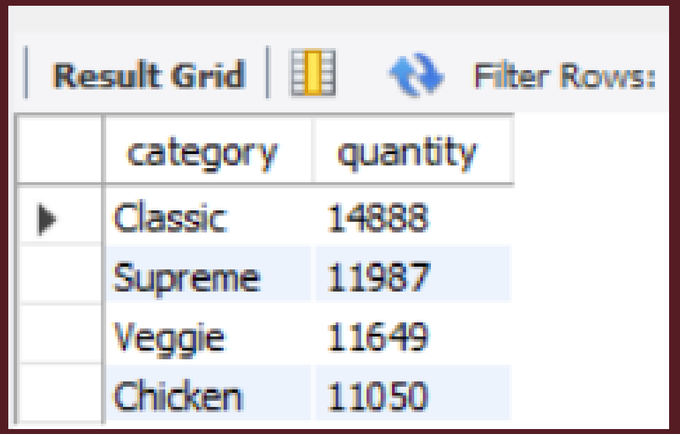
	name	quantity
▶	The Brie Carre Pizza	490
	The Mediterranean Pizza	934
	The Calabrese Pizza	937
	The Spinach Supreme Pizza	950
	The Soppressata Pizza	961

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.

Input Command

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



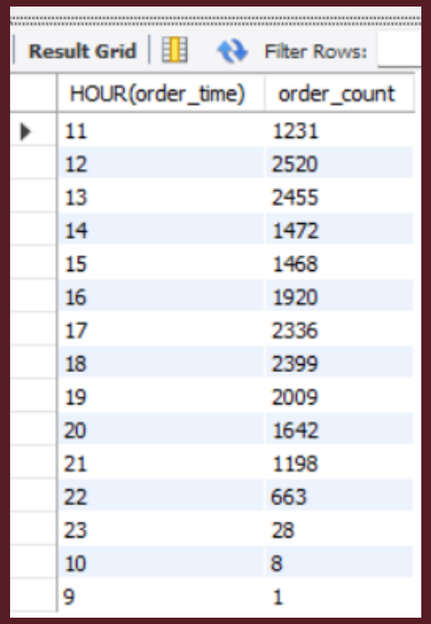
	category	quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

The Total Quantity of each Pizza category Ordered

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

Input Command

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



Result Grid | Filter Rows:

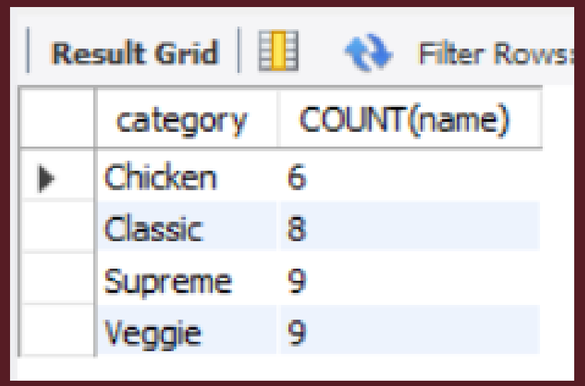
HOUR(order_time)	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

Distribution of Orders by Hour of the Day

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

Input Command

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



Result Grid | Filter Rows:

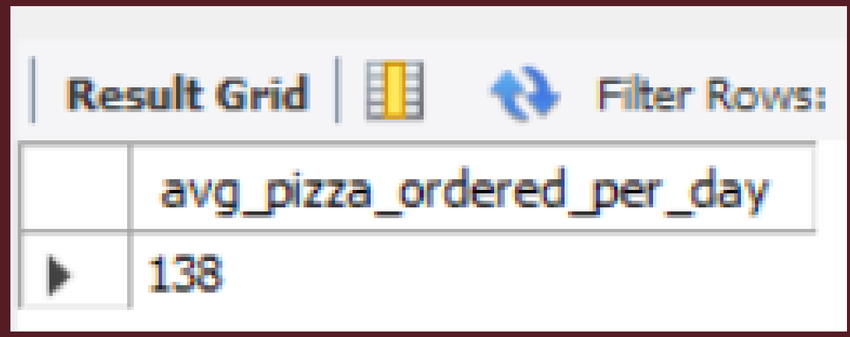
	category	COUNT(name)
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9

Category-wise distribution of Pizzas

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

Input Command

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

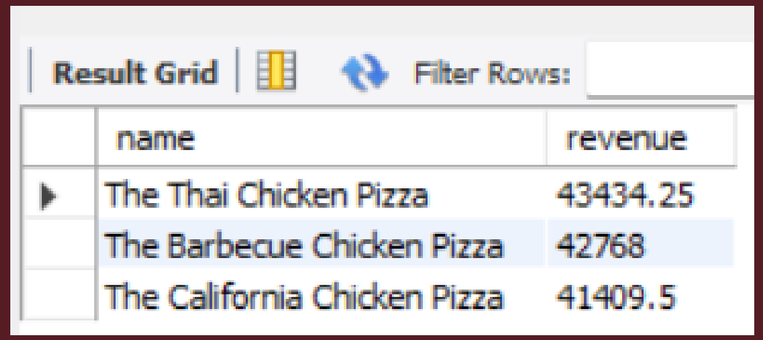


Average number of Pizzas Ordered per Day

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

Input Command

```
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) 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

Top 3 most Ordered Pizza types

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

Input Command

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

Result Grid | Filter Rows:

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Percentage Contribution of Each Pizza

12) ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

Input Command

```
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 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
2015-01-08	19399.05
2015-01-09	21526.4
2015-01-10	23990.350000000002
2015-01-11	25862.65
2015-01-12	27781.7
2015-01-13	29831.300000000003
2015-01-14	32358.700000000004
2015-01-15	34343.50000000001

Cumulative Revenue generated of First Year

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

Input Command

```
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
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

Top 3 most Ordered Pizza type based on Revenue

THANK YOU!