PIZZA SALES ANALYSIS

USING POSTGRE SQL

By : Ankit Kumar Rajak

Introduction

This project Focuses on analyzing a pizza dataset to solve business problems and extract insights on sales, orders and customer preferences. The data was processed using postgre SQl and the analysis helped provide recommendations for optimizing business operations

Data Collection

The Dataset was collected in csv format containing four main files:

- *Orders* : **information's about the orders placed, including order date and time.**
- *Orders_details*: **Detailed breakdown of each order, including pizza types and quantity.**
- Pizzas : **Information about pizzas, including price and size.**
- Pizza_types : **categorizes pizzas by ingredients, size, and type.**

Schemas

The Schemas design invovled creating relationship between these tables to ensure integrity and facilitate complex querying. foreign key were link orders with pizza types and sizes:

The key tables and their fields :

- orders : (order_id, order_date, order_time)
- prder_details : (Order_details_id, order_id, pizza_id, quantity)
- Pizzas : (pizza_id, pizza_type_id, price, size)
- pizza_types : (pizza_type_id, category, ingredients)

Questions

Basic

1. Retrieve total number of orders placed
2. Calculate the total revenue generated from pizza sales
3. Identify the highest price pizza
4. Identify the most common pizzas order

Intermediate

1. Join the necessary table to find the total quantity of each pizza category ordered
2. Determine the distribution of order by hours of th day
3. find categories wise distribution of pizzas
4. Group the order by date and calculate the average number of pizzas ordered per day
5. Determine the top 3 most ordered pizzas based on revenue

Advance

1. calculate teh percentage contribution of each pizzas to total revenue
2. Analyze the cumulative revenue genrated over time
3. determine the top 3 most ordered pizzas types based on revenue for each pizza category

Basic

1. Total number of orders placed

```sql
select  count(order_id) as total_orders from orders
```

Total_orders

21350

2. Total revenue generated from pizza sales

```sql
select
round(sum(cast(o.quantity as decimal(10,2)) * cast(p.price as decimal(10,2))),2)
as total_revenue from order_details o join
pizzas p on o.pizza_id =p.pizza_id

```

Total_revenue

817860.05

3. Highest priced pizza

```sql
select pizza_types.pizza_type_id, pizzas.pizza_id, pizzas.price from pizzas
join pizza_types   on pizzas.pizza_type_id=pizza_types.pizza_type_id
where price = (select max(price) from pizzas)
```

Name	Price
The Greek Pizza	35.95

4. Most common pizzas order

Query | Query History

```
1 select  pizza_types.pizza_name, sum(order_details.quantity) as total_quanity from pizza_types
2 join pizzas on pizza_types.pizza_type_id=pizzas.pizza_type_id
3 join order_details on pizzas.pizza_id=order_details.pizza_id
4 group by pizza_types.pizza_name
5 order by total_quanity desc
6 limit 5
7
```

Pizza_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

Intermediate

1. Total quantity of each pizza category ordered

Query | Query History

```
1  select  pizza_types.category , sum(quantity) from  pizza_types
2  join pizzas on pizza_types.pizza_type_id= pizzas.pizza_type_id
3  join  order_details on order_details.pizza_id=pizzas.pizza_id
4  group by pizza_types.category
5  order by sum(quantity) desc
6
```

Category	Total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

2. Distribution of order by hours of the day

Query | Query History

```
1  select extract(hour from order_time) as hour,
2  count(order_id) as order_count from  orders
3   group by  extract(hour from order_time)
4   order by count(order_id) desc
```

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

3. Categories wise distribution of pizzas

Query | Query History

```
1  select category,count(pizza_name) from public.pizza_types
2  group by category
3
```

Category	Count
Veggie	9
Chicken	6
Supreme	9
Classic	8

4. Group the order by date and Average number of pizzas ordered per day

Query | Query History

```
1  select avg(total_quantity) as Avg_pizza_per_day from
2   (select order_date, sum(quantity) as total_quantity  from  orders
3     join order_details on  order_details.order_id= orders.order_id
4     group by order_date) as ordered_quantity
5
```

Avg_pizza_per_day
138

5. Top 3 most ordered pizzas based on revenue

Query | Query History

```sql
select pizza_name, sum(cast(quantity as decimal) * cast(price as decimal)) 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_name
order by revenue desc
limit 3
```

pizza_name	revenue
The Thai Chicken pizza	43434.25
The Barbecue Chicken pizza	42768.00
The California Chicken pizza	41409.50

Advance

1.Percentage contribution of each pizzas to total revenue

Query | Query History

```sql
select category,
    cast(sum(quantity * price) / (select cast(sum(quantity*price) as decimal(10,2))
    from  order_details join pizzas on order_details.pizza_id = pizzas.pizza_id  )
    as decimal(10,2))* 100 as percentage
    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 category
    order by percentage desc

```

Category	Percentage
Classic	27.00
Supreme	25.00
Veggie	24.00
Chicken	24.00

2. Analyze the cumulative revenue genrated over time

Query | Query History

```sql
select order_date, sum(revenue) over(order by order_date) as cum_revenue
from
    (select  order_date, sum(quantity*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 order_date)
as sales

```

3. Top 3 most ordered pizzas types based on revenue for each pizza category

Query | Query History

```sql
select category,pizza_name, revenue,rn
from
    (select category,pizza_name,revenue, rank()
     over(partition by category order by revenue desc) as rn
        from
            (select  category,pizza_name , cast(sum(quantity*price)as decimal(10,2)) 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 category, pizza_name)
    )
where rn<=3

```

Chicken	The Thai Chicken Pizza	43434.25	1
Chicken	The Barbecue Chicken Pizza	42768.00	2
Chicken	The California Chicken Pizza	41409.50	3
Classic	The Classic Deluxe Pizza	38180.50	1
Classic	The Hawaiian Pizza	32273.25	2
Classic	The Pepperoni Pizza	30161.75	3
Supreme	The Spicy Italian Pizza	34831.25	1
Supreme	The Italian Supreme Pizza	33476.75	2
Supreme	The Sicilian Pizza	30940.50	3
Veggie	The Four Cheese Pizza	32265.70	1
Veggie	The Mexicana Pizza	26780.75	2
Veggie	The Five Cheese Pizza	26066.50	3

Recommendations

1. focus on classic pizzas
2. optimize inventory for peak hour (12-1 pm and 5-7pm)
3. upsell large sizes
4. Targeting Thai chicken & barbecue chicken to maximize profit.


Thank you