PIZZA SALES ANALYSIS USING SQL

Insights & Analytics from Pizza Sales Dataset

ABOUT ME & PROJECT OVERVIEW

About Me

- Name: Ritu Mishra
- Background: BCA student with specialization in AI & Data Science at K.R. Mangalam University
- Experience:
  Interned in Data Science at Thinkian Infotech
  Completed multiple tech internships and certifications
  Hands-on skills in SQL, Python, Power BI, and Web Development
  Actively participated in project competitions and technical events

Project Description: Pizza Sales Analysis

This project focuses on exploring a pizza sales dataset using structured SQL queries.
The objective is to extract business insights like:

- Order trends across time
- Revenue breakdowns by pizza type and size
- Customer preferences
- Performance of different pizza categories

It demonstrates how SQL can be used to convert raw sales data into actionable insights for business decisions.

DATASET OVERVIEW

Files Used:

- orders.csv: Contains order IDs and timestamps.
- order_details.csv: Links orders with pizza IDs and quantity.
- pizzas.csv: Links pizza IDs with size and type.
- pizza_types.csv: Contains pizza name, category, and ingredients.

BASIC SQL ANALYSIS

Retrieve the total number of orders placed

```
1      -- Retrieve the total number of orders placed.
2
3 •    Select count(order_id) as total_orders from orders;
4
```

Result Grid | Filter Rows:

	total_orders
▶	21350

BASIC SQL ANALYSIS

Calculate the total revenue generated from pizza sales.

```sql
-- Calculate the total revenue generated from pizza sales.

SELECT
round(sum(orders_details.quantity*pizzas.price) , 2) as total_sales
from orders_details join pizzas
on pizzas.pizza_id = orders_details.pizza_id;
```

Result Grid | Filter Rows:

total_sales
817860.05

BASIC SQL ANALYSIS

Identify the highest-priced pizza.

```sql
-- Identify the highest-priced pizza

Select pizza_types.name , pizzas.price
from pizza_types JOIN pizzas
on pizzas.pizza_type_id = pizza_types.pizza_type_id
order by pizzas.price desc limit 1;
```

Result Grid | Filter Rows:

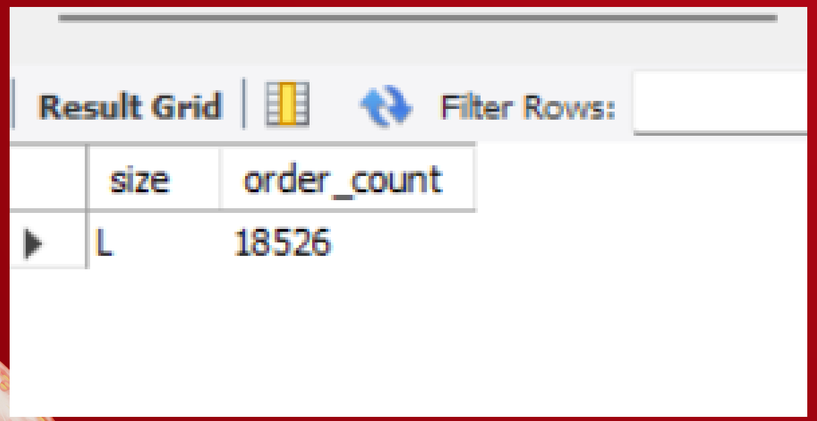
name	price
The Greek Pizza	35.95

BASIC SQL ANALYSIS

Identify the most common pizza size ordered.

```sql
-- Identify the most common pizza size ordered.

SELECT pizzas.size, count(orders_details.order_detail_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 limit 1;
```



Result Grid | Filter Rows:

size	order_count
L	18526

BASIC SQL ANALYSIS

List the top 5 most ordered pizza types along with their quantities

```sql
-- List the top 5 most ordered pizza types along with their quantities.

Select pizza_types.name , 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.name order by quantity desc limit 5;
```

Result Grid | Filter Rows:

name	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 SQL ANALYSIS

Join the necessary tables to find the total quantity of each pizza category ordered.

```sql
-- Join the necessary tables to find the total quantity of each pizza category ordered.

SELECT pizza_types.category , 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 orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category order by total_quantity desc;

```

Result Grid | Filter Rows:

category	total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

INTERMEDIATE SQL ANALYSIS

Determine the distribution of orders by hour of the day.

```sql
-- Determine the distribution of orders by hour of the day.

SELECT hour(order_time) as hour , count(order_id) as order_count from orders
GROUP BY hour(order_time);
```

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

hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642

INTERMEDIATE SQL ANALYSIS

Join relevant tables to find the category-wise distribution of pizzas.

```sql
-- Join relevant tables to find the category-wise distribution of pizzas

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

INTERMEDIATE SQL ANALYSIS

Group the orders by date and calculate the average number of pizzas ordered per day

```sql
-- Group the orders by date and calculate the average number of pizzas ordered per day.

select round(avg(quantity) , 0) as avg_pizza_orders 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: | Export:

avg_pizza_orders
138

INTERMEDIATE SQL ANALYSIS

Determine the top 3 most ordered pizza types based on revenue.

```sql
-- Determine the top 3 most ordered pizza types based on revenue.

SELECT pizza_types.name , round(sum(orders_details.quantity * pizzas.price ), 0)as revenu
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 revenue desc limit 3;
```

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

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

ADVANCED SQL ANALYSIS

Calculate the percentage contribution of each pizza type to total revenue.

```sql
-- Calculate the percentage contribution of each pizza type to total revenue.

SELECT
    pizza_types.name,
    (SUM(orders_details.quantity * pizzas.price) / (SELECT
            ROUND(SUM(orders_details.quantity * pizzas.price),
                    2) AS total_sales
        FROM
            orders_details
                JOIN
            pizzas ON pizzas.pizza_id = orders_details.pizza_id)) * 100 AS revenue
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 revenue DESC;
```

Result Grid | Filter Rows:

name	revenue
The Thai Chicken Pizza	5.310719113863062
The Barbecue Chicken Pizza	5.229256521332714
The California Chicken Pizza	5.063152308270833
The Classic Deluxe Pizza	4.66834148458529
The Spicy Italian Pizza	4.258827656394269
The Southwest Chicken Pizza	4.243482732773168
The Italian Supreme Pizza	4.093212524563341
The Hawaiian Pizza	3.946060209200828
The Four Cheese Pizza	3.945137068377487
The Sicilian Pizza	3.783104456563198

ADVANCED SQL ANALYSIS

Analyze the cumulative revenue generated over time.

```sql
-- Analyze the cumulative revenue generated over time.


select order_date,
sum(revenue) over (order by order_date) as cum_revenue
from
(SELECT orders.Order_date , sum(orders_details.quantity * pizzas.price) 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	cum_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
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526.4
2015-01-10	23990.350000000002

ADVANCED SQL ANALYSIS

Determine the top 3 most ordered pizza types based on revenue for each pizza category

```sql
-- 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 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

THANK YOU!