LARANA PIZZA

PIZZA SALES

LARANA PIZZA

WELCOME TO

SALES ANALYSIS

Hi, I'm Maliha, and in this Pizza Sales Analysis project, I utilized SQL to explore and extract key business insights from the sales data. I applied queries using JOIN, ORDER BY, and GROUP BY clauses to analyze daily trends, top-selling pizzas, and revenue breakdowns. This project helped me strengthen my SQL skills while showcasing my ability to turn raw data into meaningful insights for decision-making.

QUESTIONS

Basic:

1. Retrieve the total number of orders placed.
2. Calculate the total revenue generated from pizza sales.
3. Identify the highest-priced pizza.
4. Identify the most common pizza size ordered.
5. List the top 5 most ordered pizza types along with their quantities.

Intermediate:

6. Join the necessary tables to find the total quantity of each pizza category ordered.
7. Determine the distribution of orders by hour of the day.
8. Join relevant tables to find the category-wise distribution of pizzas.
9. Group the orders by date and calculate the average number of pizzas ordered per day.
10. Determine the top 3 most ordered pizza types based on revenue.

Advanced:

11. Calculate the percentage contribution of each pizza type to total revenue.
12. Analyze the cumulative revenue generated over time.
13. Determine the top 3 most ordered pizza types based on revenue for each pizza category.

1. RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

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

Result Grid

total_orders
21350

2. CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

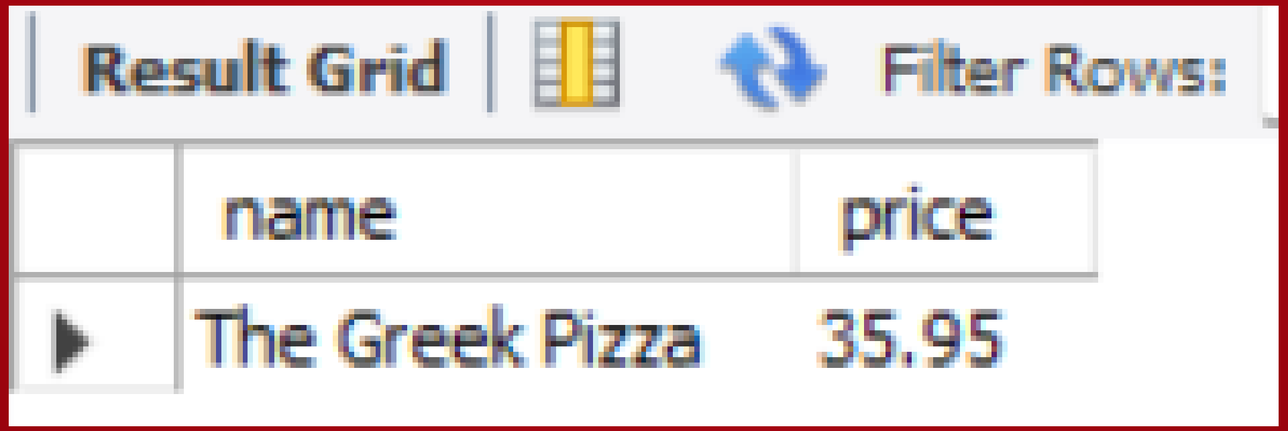
```
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS total_sales
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id;
```

Result Grid

total_sales
817860.05

3. IDENTIFY THE HIGHEST-PRICED PIZZA.

```sql
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;
```



4. IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

```sql
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

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

```sql
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 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

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

```sql
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;
```

Result Grid | Filter Rows:

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

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

```sql
SELECT
    HOUR(orders.time) AS hour, COUNT(order_id) AS order_count
FROM
    orders
GROUP BY HOUR(orders.time);
```

Result Grid | Filter Rov

hour	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

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

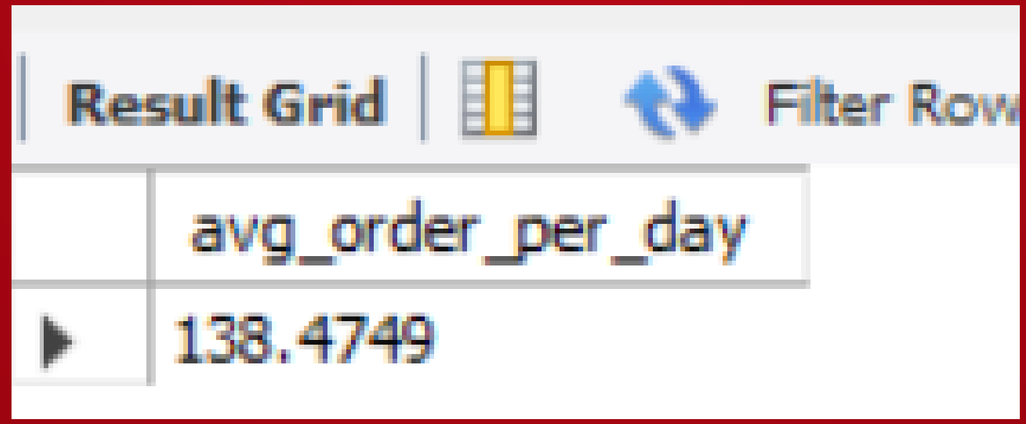
```
select category, count(name) from pizza_types
group by category;
```

Result Grid | Filter Rows:

category	count(name)
Chicken	6
Classic	8
Supreme	9
Veggie	9

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

```
SELECT
    AVG(order_quantity.quantity) AS avg_order_per_day
FROM
    (SELECT
        SUM(order_details.quantity) AS quantity, orders.date
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.date) AS order_quantity;
```



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

```sql
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;
```

Result Grid | Filter Rows:

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

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

```sql
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 Row

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

12. ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

```sql
select date,
sum(revenue) over(order by date) as cum_revenue
from
(Select orders.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.date) as sales;
```

Result Grid | Filter Rows:

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

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

```sql
Select pizza_types.category, round(sum(pizza_order.price * pizza_order.quantity), 0) as revenue
From
    pizza_types
Join
    (Select order_details.order_id, order_details.pizza_id, order_details.quantity, pizzas.price, pizzas.pizza_type_id
    From order_details
    Join pizzas
    on order_details.pizza_id = pizzas.pizza_id) as pizza_order

on  pizza_order.pizza_type_id = pizza_types.pizza_type_id
Group by pizza_types.category
Order by revenue desc
limit 3;
```

Result Grid | Filter R

category	revenue
Classic	220053
Supreme	208197
Chicken	195920

LARANA PIZZA

THANK YOU!