MARATHA, Inc.

ANIRUDDHA GHOGARE SQL DATA ANALYST PROJECT

+91 9359698594
aniruddhaghogareda@gmail.com
123 Anywhere St., Any City, ST 12345

Introduction

Pizza Sales Data Analysis Project

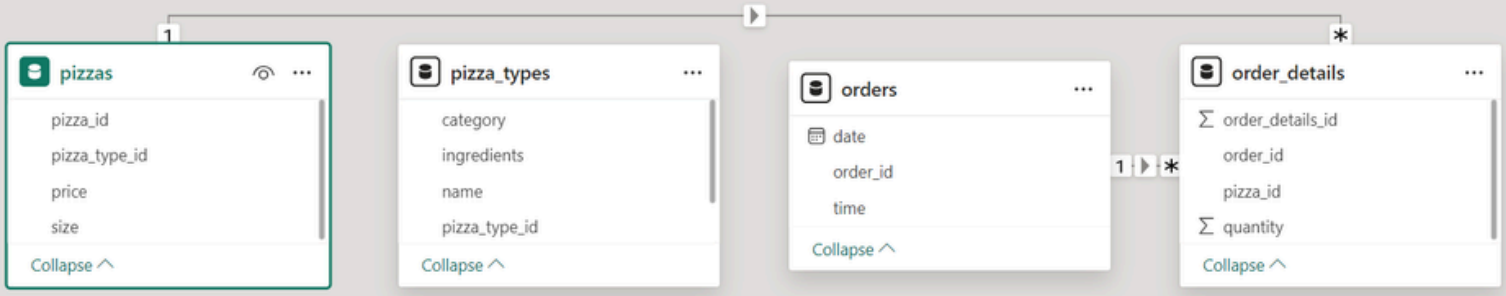
In this project, I analyzed a fictional dataset from a pizza restaurant to gain actionable insights into its sales performance, customer preferences, and operational efficiency. Using SQL, I worked with structured data stored in a relational database, exploring various aspects such as:

The project involved:

- Retrieving and summarizing key metrics, such as the total number of orders and revenue generated.
- Identifying top-performing pizza types, sizes,and categories to understand customer preferences.
- Analyzing sales trends by time to uncover patterns in customer ordering behavior.
- Leveraging advanced SQL techniques, including joins, window functions, and aggregations, to calculate revenue contributions, category-wise performance, and cumulative revenue trends.

Data Tables

here we use 4 csv files



Questions

Basic:

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

Intermediate:

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

Advanced:

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

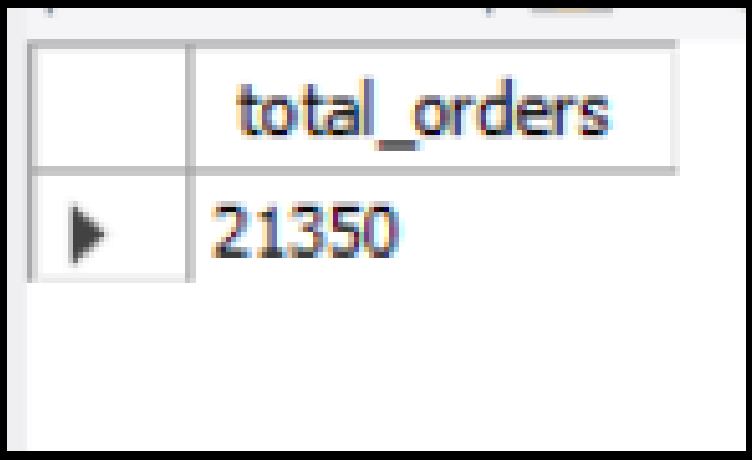
create database & tables

```sql
create database pizzahut;

create table orders (
order_id int not null ,
order_date date not null ,
order_time time not null ,
primary key (order_id) );



create table order_details (
order_details_id int not null,
order_id int not null ,
pizza_id text not null ,
quantity int not null ,
primary key (order_details_id) );
```

BASICS

```sql
-- Retrieve the total number of orders placed.

select * from orders;
select count(order_id)as total_orders from orders;
```

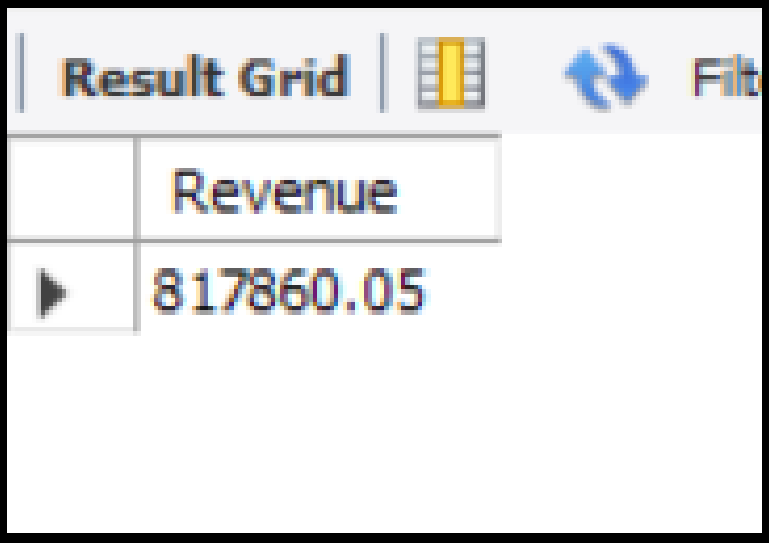
OUTPUT:-



	total_orders
▶	21350

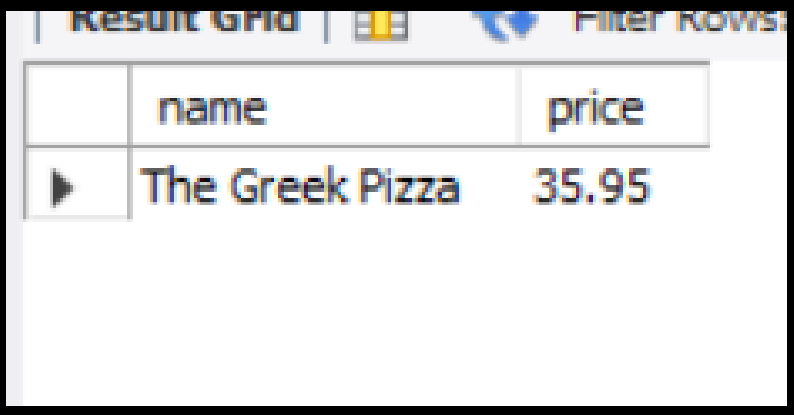
```sql
-- Calculate the total revenue generated from pizza sales

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

OUTPUT:-



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


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;
```

OUTPUT:-



	name	price
▶	The Greek Pizza	35.95

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

select quantity, count(order_details_id)
from order_details group by quantity ;


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
LIMIT 3 ;
```

OUTPUT:-

	size	order_count
▶	L	18526
	M	15385
	S	14137

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

select pizza_types.name,
sum(order_details.quantity) as most_ordered
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 most_ordered desc LIMIT 5 ;
```

OUTPUT:-

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

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;
```

OUTPUT:-

Result Grid | Filter Rows

category	Quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

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

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

```

OUTPUT:-

HOUR	OrderCount
11	1231
12	2520
13	2455
14	1472
15	1468

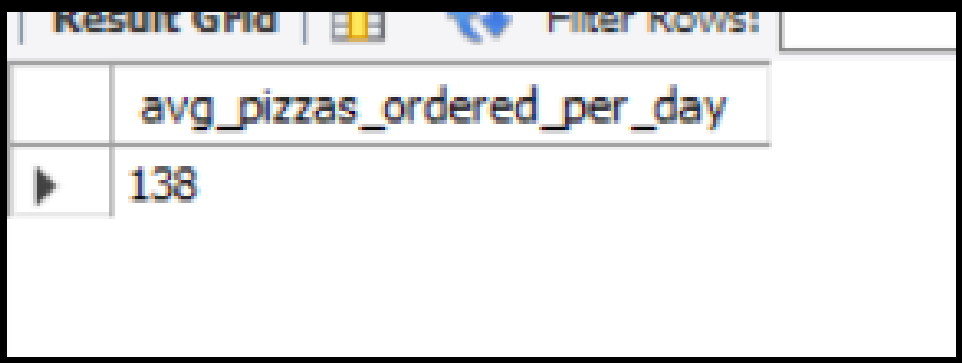
```sql
-- Join relevant tables to find the category-wise distribution of pizzas.

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

OUTPUT:-

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

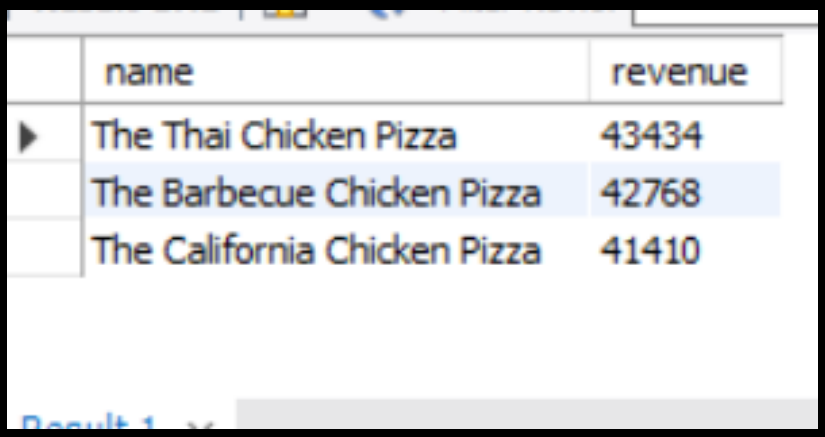
```sql
-- Group the orders by date and calculate the average number of pizzas ordered per day.
SELECT
    ROUND(AVG(quantity), 0) as avg_pizzas_ordered_per_day
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

OUTPUT:-



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

select pizza_types.name,
round(sum(order_details.quantity * pizzas.price),0) 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.name order by revenue desc limit 3 ;
```

OUTPUT:-

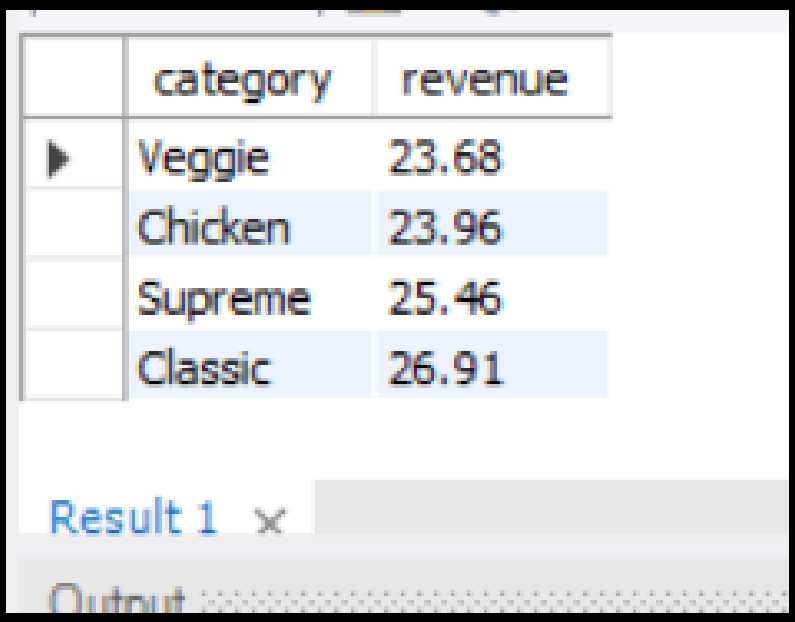


name	revenue
The Thai Chicken Pizza	43434
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41410

ADVANCED

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

select pizza_types.category,
round((sum(order_details.quantity* pizzas.price) / (
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS Revenue
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;
```

OUTPUT:-

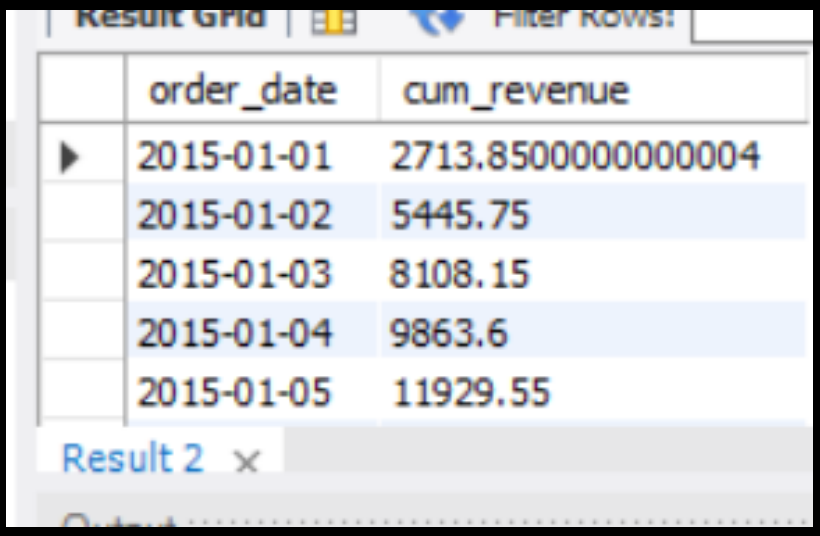


category	revenue
Veggie	23.68
Chicken	23.96
Supreme	25.46
Classic	26.91

Result 1

```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(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 ;
```

OUTPUT:-



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

Result 2

```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((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 ;
```

OUTPUT:-

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

Result 1