# Introduction to the Pizza Sales With SQL

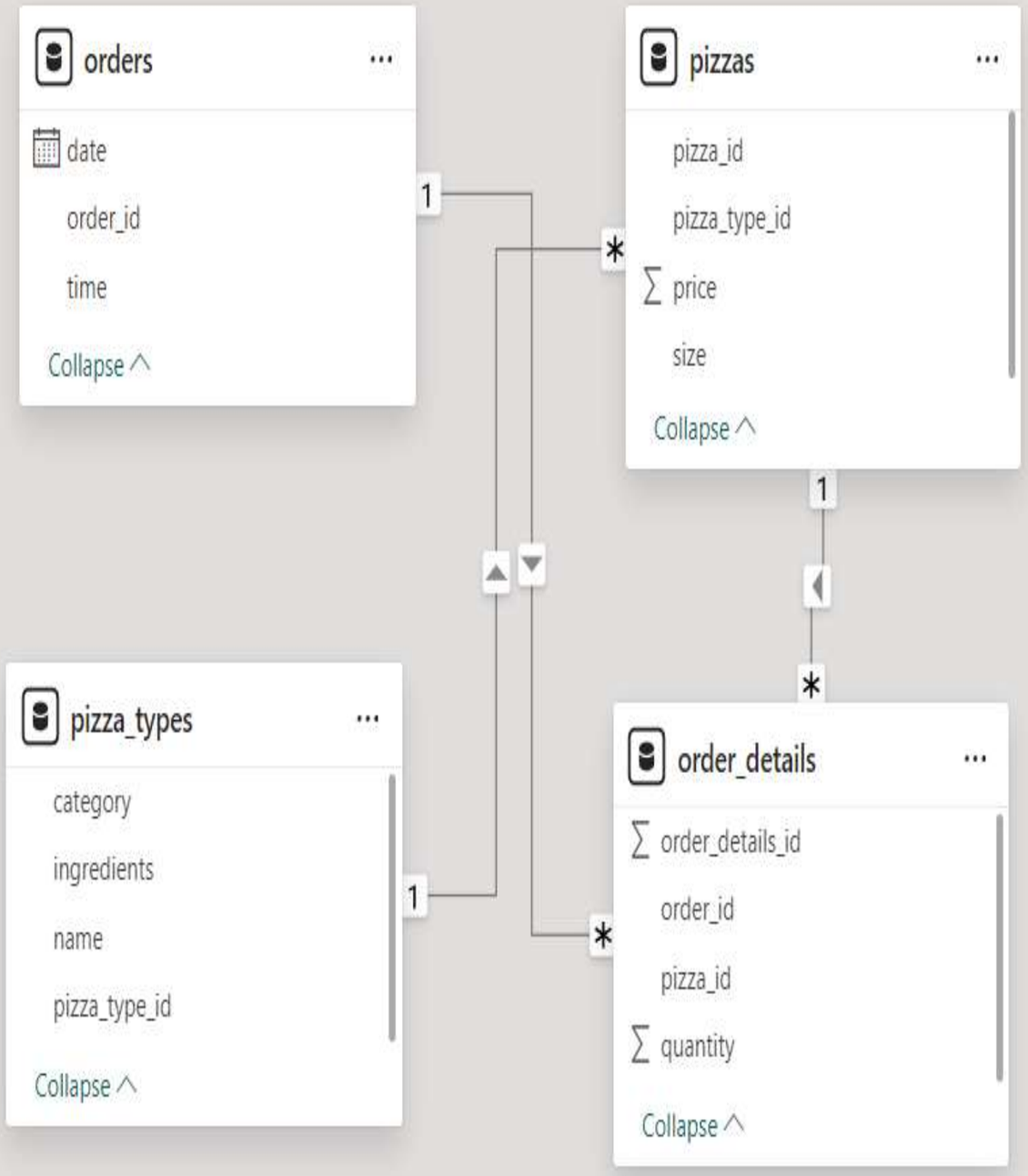
by Saurabh Shah

# Project Overview

The SQL project on pizza sales analysis involves a comprehensive examination of a dataset containing details about pizza orders, including information on order quantities, pizza types, sizes, and revenue. The project aims to extract valuable insights and key performance indicators (KPIs) to support data-driven decision-making for the pizza business. I solved and performed queries on a range of questions, from easy to advanced levels, including identifying the busiest days and times, best-selling pizzas, and average order value. The analysis covered various aspects of the pizza store's operations, such as sales trends, customer preferences, and operational efficiency. By leveraging SQL and Power BI, the project provides actionable insights and recommendations to optimize operations, improve customer satisfaction, and enhance profitability.

# Data Overview

- **Order Table :** It Contains the data related to order like date, time & order_id .
- **Pizzas Table :** It Contains Essential data like pizza_id, pizza_type_id, price & size.
- **Pizza_types Table :** It Contains data related to pizza like category, ingredients, name & pizza_type_id .
- **Order_details Table :** It Contains essential data like order_details_id, order_id, pizza_id & quantity.



# Query 1: Retrieve the total number of orders placed.

```sql
select count(order_id) as Total_orders
from
    orders;
```

| | Total_orders |
|---|---|
| 1 | 21350 |

# Query 2: Calculate the total revenue generated from pizza sales.

```sql
select round(SUM(order_details.quantity * pizzas.price),2)as Total_Revenue
from
    order_details join pizzas
on
    pizzas.pizza_id = order_details.pizza_id;
```

| | Total_Revenue |
|---|---|
| 1 | 817860.05 |

# Query 3: Identify the highest-priced pizza.

```sql
Select Top 1 pizza_types.name, round(pizzas.price ,2) as Price
from
    pizza_types join pizzas
on
    pizza_types.pizza_type_id = pizzas.pizza_type_id
order by
    pizzas.price desc ;
```

| | name | Price |
|---|---|---|
| 1 | The Greek Pizza | 35.95 |

# Query 4: Identify the most common pizza size ordered.

```sql
select top 1 pizzas.size, SUM(order_details.quantity) as order_quantity
from
    pizzas join order_details
on
    pizzas.pizza_id = order_details.pizza_id
group by
    pizzas.size
order by
    order_quantity desc;
```

| | size | order_quantity |
|---|---|---|
| 1 | L | 18956 |

# Query 5: List the top 5 most ordered pizza types along with their quantities.

```sql
select top 5 pizza_types.name, SUM(order_details.quantity) as order_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
    order_quantity desc;
```

| | name | order_quantity |
|---|---|---|
| 1 | The Classic Deluxe Pizza | 2453 |
| 2 | The Barbecue Chicken Pizza | 2432 |
| 3 | The Hawaiian Pizza | 2422 |
| 4 | The Pepperoni Pizza | 2418 |
| 5 | The Thai Chicken Pizza | 2371 |

# Query 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 order_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
    order_quantity desc;
```

| | category | order_quantity |
|---|---|---|
| 1 | Classic | 14888 |
| 2 | Supreme | 11987 |
| 3 | Veggie | 11649 |
| 4 | Chicken | 11050 |

# Query 7: Determine the distribution of orders by hour of the day.

```
Select DATEPART(HOUR, time) as e_Hour , COUNT(order_id) as T_Orders
from
    orders
group by
    DATEPART(HOUR, time)
order by
    DATEPART(HOUR, time);
```

| | e_Hour | T_Orders |
|---|---|---|
| 1 | 9 | 1 |
| 2 | 10 | 8 |
| 3 | 11 | 1231 |
| 4 | 12 | 2520 |
| 5 | 13 | 2455 |
| 6 | 14 | 1472 |
| 7 | 15 | 1468 |
| 8 | 16 | 1920 |
| 9 | 17 | 2336 |
| 10 | 18 | 2399 |
| 11 | 19 | 2009 |
| 12 | 20 | 1642 |
| 13 | 21 | 1198 |
| 14 | 22 | 663 |
| 15 | 23 | 28 |

# Query 8: Join relevant tables to find the category-wise distribution of pizzas.

```
select category, COUNT(name) as Distribution
from
    pizza_types
group by
    category;
```

| | category | Distribution |
|---|---|---|
| 1 | Chicken | 6 |
| 2 | Classic | 8 |
| 3 | Supreme | 9 |
| 4 | Veggie | 9 |

# Query 9: Group the orders by date and calculate the average number of pizzas ered per

```sql
select round(avg(Total_Orders),0) as Average_Order_PerDay
from
    (select orders.date as Date, SUM(order_details.quantity) as Total_Orders
from
    order_details join orders
on
    order_details.order_id = orders.order_id
group by Date) as Order_Quantity;
```

| | Average_Order_PerDay |
|---|---|
| 1 | 138 |

# Query 10: Determine the top 3 most ordered pizza types based on revenue.

```sql
select top 3 pizza_types.name as Pizza, 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.name
order by
    Revenue desc;
```

| | Pizza | Revenue |
|---|---|---|
| 1 | The Thai Chicken Pizza | 43434.25 |
| 2 | The Barbecue Chicken Pizza | 42768 |
| 3 | The California Chicken Pizza | 41409.5 |

# Query 11: Calculate the percentage contribution of each pizza type to total revenue

```sql
select pizza_types.category as Pizza,
round((SUM (order_details.quantity * pizzas.price) /
(select round(SUM(order_details.quantity * pizzas.price),2)as Total_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 desc;
```

| | Pizza | Revenue |
|---|---|---|
| 1 | Classic | 26.91 |
| 2 | Supreme | 25.46 |
| 3 | Chicken | 23.96 |
| 4 | Veggie | 23.68 |

# Query 12: Analyze the cumulative revenue generated over time.

```sql
select date, Revenue,
round(sum(Revenue) over (order by date),2) as Cum_Revenue
from
    (select orders.date,
round(sum(order_details. quantity * pizzas.price),2) 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;
```

| | date | Revenue | Cum_Revenue |
|---|---|---|---|
| 1 | 2015-01-01 | 2713.85 | 2713.85 |
| 2 | 2015-01-02 | 2731.9 | 5445.75 |
| 3 | 2015-01-03 | 2662.4 | 8108.15 |
| 4 | 2015-01-04 | 1755.45 | 9863.6 |
| 5 | 2015-01-05 | 2065.95 | 11929.55 |
| 6 | 2015-01-06 | 2428.95 | 14358.5 |
| 7 | 2015-01-07 | 2202.2 | 16560.7 |
| 8 | 2015-01-08 | 2838.35 | 19399.05 |
| 9 | 2015-01-09 | 2127.35 | 21526.4 |
| 10 | 2015-01-10 | 2463.95 | 23990.35 |
| 11 | 2015-01-11 | 1872.3 | 25862.65 |
| 12 | 2015-01-12 | 1919.05 | 27781.7 |
| 13 | 2015-01-13 | 2049.6 | 29831.3 |
| 14 | 2015-01-14 | 2527.4 | 32358.7 |
| 15 | 2015-01-15 | 1984.8 | 34343.5 |
| 344 | 2015-12-16 | 2234.8 | 790011.8 |
| 345 | 2015-12-17 | 1880.75 | 791892.55 |
| 346 | 2015-12-18 | 2886.3 | 794778.85 |
| 347 | 2015-12-19 | 2304.2 | 797083.05 |
| 348 | 2015-12-20 | 2104.9 | 799187.95 |
| 349 | 2015-12-21 | 2100.7 | 801288.65 |
| 350 | 2015-12-22 | 1882.95 | 803171.6 |
| 351 | 2015-12-23 | 2244.3 | 805415.9 |
| 352 | 2015-12-24 | 2137.85 | 807553.75 |
| 353 | 2015-12-26 | 1643.05 | 809196.8 |
| 354 | 2015-12-27 | 1419 | 810615.8 |
| 355 | 2015-12-28 | 1637.2 | 812253 |
| 356 | 2015-12-29 | 1353.25 | 813606.25 |
| 357 | 2015-12-30 | 1337.8 | 814944.05 |
| 358 | 2015-12-31 | 2916 | 817860.05 |

# Query 13: Determine the top 3 most ordered pizza types based on revenue for each pizza category

```sql
select category, 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,
    round(sum((order_details.quantity) * pizzas.price),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, pizza_types.name) as a) as b
where rn <= 3;
```

| | category | name | revenue |
|---|---|---|---|
| 1 | Chicken | The Thai Chicken Pizza | 43434.25 |
| 2 | Chicken | The Barbecue Chicken Pizza | 42768 |
| 3 | Chicken | The California Chicken Pizza | 41409.5 |
| 4 | Classic | The Classic Deluxe Pizza | 38180.5 |
| 5 | Classic | The Hawaiian Pizza | 32273.25 |
| 6 | Classic | The Pepperoni Pizza | 30161.75 |
| 7 | Supreme | The Spicy Italian Pizza | 34831.25 |
| 8 | Supreme | The Italian Supreme Pizza | 33476.75 |
| 9 | Supreme | The Sicilian Pizza | 30940.5 |
| 10 | Veggie | The Four Cheese Pizza | 32265.7 |
| 11 | Veggie | The Mexicana Pizza | 26780.75 |
| 12 | Veggie | The Five Cheese Pizza | 26066.5 |

# THANK YOU