SQL PROJECT

# Pizza Sales Analysis

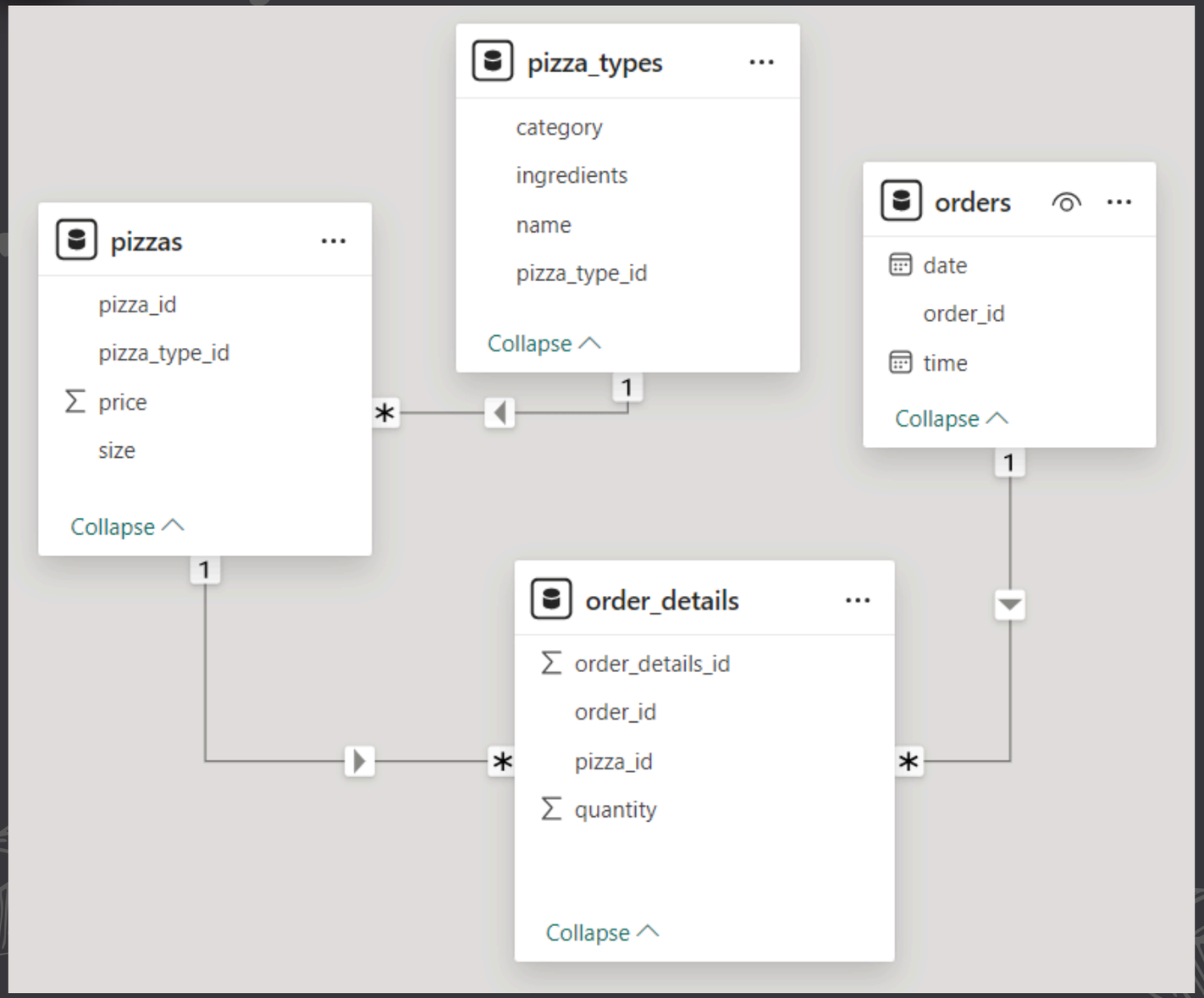
-Sanskruti Chavan

# Objectives

✤ The primary objective of the Pizza Sales Analysis is to achieve the sustainable business growth while addressing  existing challenges

✤ We need to examine the dataset with SQL and help the Pizza Sales understand it's business growth by answering simple questions.

# Pizza Sales Database Schema




# Division Of Questions

BASIC

INTERMEDIATE

ADVANCE

# BASIC

# Question - 1

✤ Retrieve the total number of orders placed.

```
SELECT
    COUNT(order_id) AS Total_Orders
FROM
    orders;
```

Result Grid | Filter Rows:

| | Total_Orders |
|---|---|
| ▶ | 21350 |

# Question - 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;
```

Result Grid | Filter Rows:

| Total_Revenue |
| --- |
| 817860.05 |

# Question - 3

✤ 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;
```

Result Grid | Filter Rows:

| name | price |
|---|---|
| The Greek Pizza | 35.95 |

# Question - 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 |

# Question - 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: | Export:

| 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

## Question - 1

✤ Join the necessary tables to find thetotal 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 category
ORDER BY quantity DESC;
```

Result Grid | Filter Rows:

| category | Quantity |
|---|---|
| Classic | 14888 |
| Supreme | 11987 |
| Veggie | 11649 |
| Chicken | 11050 |

# Question - 2

✤ Determine the distribution of orders by hour of the day.

```sql
SELECT
    HOUR(order_time) AS Hour, COUNT(order_id) AS Order_Count
FROM
    orders
GROUP BY HOUR(order_time);
```

Result Grid | Filter Rows:

| 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 |

# Question - 3

✤ 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 |

# Question - 4

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

```
SELECT
    ROUND(AVG(quantity), 0) AS Avg_Pizza_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;
```

Result Grid | Filter Rows: | Exp

| Avg_Pizza_Ordered_per_day |
|---|
| 138 |

# Question - 5

✤ 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: | Export: | Wrap Cell Content:

| name | Revenue |
|---|---|
| The Thai Chicken Pizza | 43434.25 |
| The Barbecue Chicken Pizza | 42768 |
| The California Chicken Pizza | 41409.5 |

# ADVANCED

# Question - 1

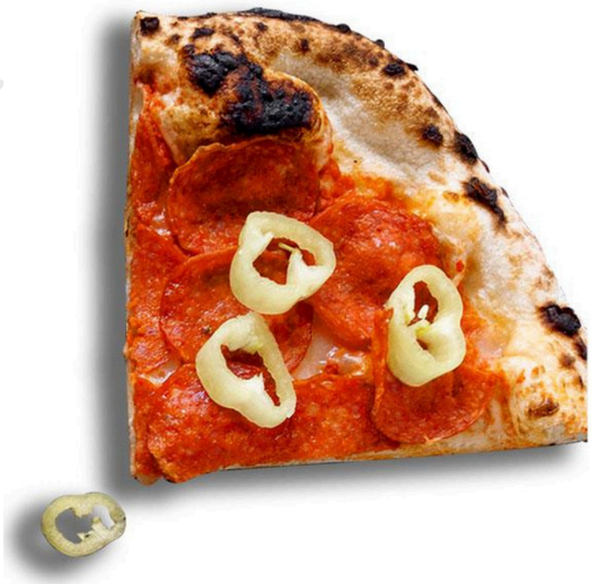
✤ Calculate the percentage contribution of each pizza type to total revenue.

```sql
SELECT
    pizza_types.category,
    ROUND(SUM(order_details.quantity * pizzas.price) / (SELECT
                ROUND(SUM(order_details.quantity * pizzas.price),
                        2) AS Total_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;
```

# OUTPUT

Result Grid | Filter Rows:

| category | Revenue |
|---|---|
| Classic | 26.91 |
| Supreme | 25.46 |
| Chicken | 23.96 |
| Veggie | 23.68 |



# Question - 2

✤ Analyze the cumulative revenue generated over time.

```sql
SELECT order_date,
sum(Revenue) OVER(ORDER BY order_date) AS Cummulative_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

Result Grid | Filter Rows: | Expo

| order_date | Cummulative_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 |
| 2015-01-14 | 32358.700000000004 |
| 2015-01-15 | 34343.50000000001 |
| 2015-01-16 | 36937.65000000001 |
| 2015-01-17 | 39001.75000000001 |
| 2015-01-18 | 40978.600000000006 |
| 2015-01-19 | 43365.75000000001 |
| 2015-01-20 | 45763.65000000001 |
| 2015-01-21 | 47804.20000000001 |
| 2015-01-22 | 50300.90000000001 |
| 2015-01-23 | 52724.600000000006 |
| 2015-01-24 | 55013.850000000006 |
| 2015-01-25 | 56631.40000000001 |
| 2015-01-26 | 58515.80000000001 |
| 2015-01-27 | 61043.85000000001 |
| 2015-01-28 | 63059.85000000001 |
| 2015-01-29 | 65105.150000000016 |
| 2015-01-30 | 67375.45000000001 |
| 2015-01-31 | 69793.30000000002 |
| 2015-02-01 | 72982.50000000001 |
| 2015-02-02 | 75311.10000000002 |
| 2015-02-03 | 77925.90000000002 |
| 2015-02-04 | 80159.80000000002 |
| 2015-02-05 | 82375.60000000002 |
| 2015-02-06 | 84885.55000000002 |
| 2015-02-07 | 87123.20000000001 |
| 2015-02-08 | 89158.20000000001 |
| 2015-02-09 | 91353.55000000002 |
| 2015-02-10 | 93410.05000000002 |
| 2015-02-11 | 95870.05000000002 |
| 2015-02-12 | 98028.85000000002 |
| 2015-02-13 | 100783.35000000002 |
| 2015-02-14 | 103102.50000000001 |
| 2015-02-15 | 105243.75000000001 |
| 2015-02-16 | 107212.55000000002 |
| 2015-02-17 | 109334.45000000001 |
| 2015-02-18 | 111977.30000000002 |
| 2015-02-19 | 114007.55000000002 |
| 2015-02-20 | 116898.70000000001 |

# Question - 3

✤Determine the top 3 most ordered pizza type 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 AA
WHERE RN <= 3;
```

# OUTPUT

Result Grid | Filter Rows: | Export: | Wrap C

| 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 |
| The Mexicana Pizza | 26780.75 |
| The Five Cheese Pizza | 26066.5 |

# THANK YOU!!!