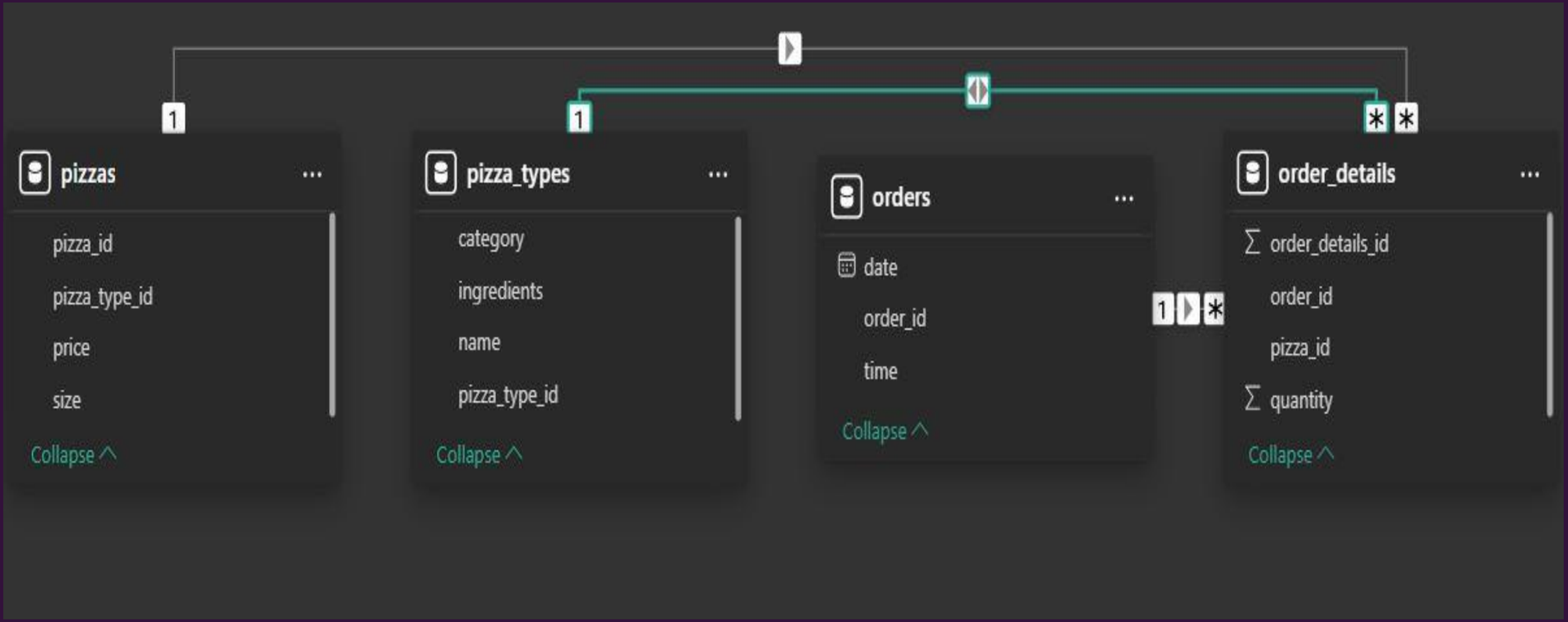
# SALES ANALYSIS OF A RESTAURANT USING SQL

BY NIHARIKA SARKER NITU

# PROJECT OBJECTIVES

This project concentrates on sales analyzing a restaurant using SQL. The main purpose of this project is to analyze and identify the trend in sales key objectives, methodologies and anticipated outcomes for strategic execution. To ensure a structured path to success, the analysis is categorized according to a number of components that are intended to give a clear and thorough understanding of the dynamics of the project.

# Schema of the Datasets



# Retrieve the total number of orders placed.

01

```
-- Basic:
-- Retrieve the total number of orders placed.

SELECT count(order_id) as Total_Orders from orders;
```

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

| Total_Orders |
|---|
| 21350 |

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

Limit to 1000 rows

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

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

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

| Total_Revenue |
|---|
| 817860.05 |

03

# Identify the highest-priced pizza.

```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;
```

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

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

# 4.Identify the most common pizza size ordered.

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

SELECT
    pizzas.size,
    COUNT(orders_details.order_details_id) AS Total_Order
FROM
    pizzas
        JOIN
    orders_details ON pizzas.pizza_id = orders_details.pizza_id
GROUP BY pizzas.size
ORDER BY Total_Order DESC
LIMIT 1;
```

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

| size | Total_Order |
|---|---|
| L | 18526 |

05

## 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 Total_Quantity_Ordered
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 Total_Quantity_Ordered DESC
LIMIT 5;
```

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

| name | Total_Quantity_Ordered |
|---|---|
| The Classic Deluxe Pizza | 2453 |
| The Barbecue Chicken Pizza | 2432 |
| The Hawaiian Pizza | 2422 |
| The Pepperoni Pizza | 2418 |
| The Thai Chicken Pizza | 2371 |

Result 1

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

06

```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: | Export: | Wrap Cell Content:

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

Result 1

07

## 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 |
| 21 | 1198 |
| 22 | 663 |
| 23 | 28 |
| 10 | 8 |
| 9 | 1 |

Result 1

## 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: | Export: | Wrap Cell Content:

| category | COUNT(name) |
|---|---|
| Chicken | 6 |
| Classic | 8 |
| Supreme | 9 |
| Veggie | 9 |

08

# 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 average_pizza_ordered_per_day
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: | Wrap Cell Content:

| average_pizza_ordered_per_day |
|---|
| 138 |

09

10

# 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,
    SUM(orders_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    orders_details ON orders_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: | Fetch rows:

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

# 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.category,
    ROUND(SUM(orders_details.quantity * pizzas.price) / (SELECT
                    ROUND(SUM(orders_details.quantity * pizzas.price),
                                2) AS Total_Revenue
                FROM
                    orders_details
                        JOIN
                    pizzas ON pizzas.pizza_id = orders_details.pizza_id) * 100,
            2) 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
ORDER BY revenue DESC;
```

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

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

# Analyze the cumulative revenue generated over time



```sql
-- Analyze the cumulative revenue generated over time.
SELECT order_date,
ROUND(SUM(revenue) over(order by order_date),2) as Cumulative_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 generated_revenue_per_day;
```

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

| order_date | Cumulative_Revenue |
|---|---|
| 2015-01-01 | 2713.85 |
| 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 | 23990.35 |
| 2015-01-11 | 25862.65 |
| 2015-01-12 | 27781.7 |
| 2015-01-13 | 29831.3 |
| 2015-01-14 | 32358.7 |

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



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

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,
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: | Export: | Wrap Cell Content:

| category | name | revenue |
|---|---|---|
| Chicken | The Thai Chicken Pizza | 43434.25 |
| Chicken | The Barbecue Chicken Pizza | 42768 |
| Chicken | The California Chicken Pizza | 41409.5 |
| Classic | The Classic Deluxe Pizza | 38180.5 |
| Classic | The Hawaiian Pizza | 32273.25 |
| Classic | The Pepperoni Pizza | 30161.75 |
| Supreme | The Spicy Italian Pizza | 34831.25 |
| Supreme | The Italian Supreme Pizza | 33476.75 |
| Supreme | The Sicilian Pizza | 30940.5 |
| Veggie | The Four Cheese Pizza | 32265.70000000065 |
| Veggie | The Mexicana Pizza | 26780.75 |
| Veggie | The Five Cheese Pizza | 26066.5 |

# THANK YOU