Pizza Sales Analysis Using SQL

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.**

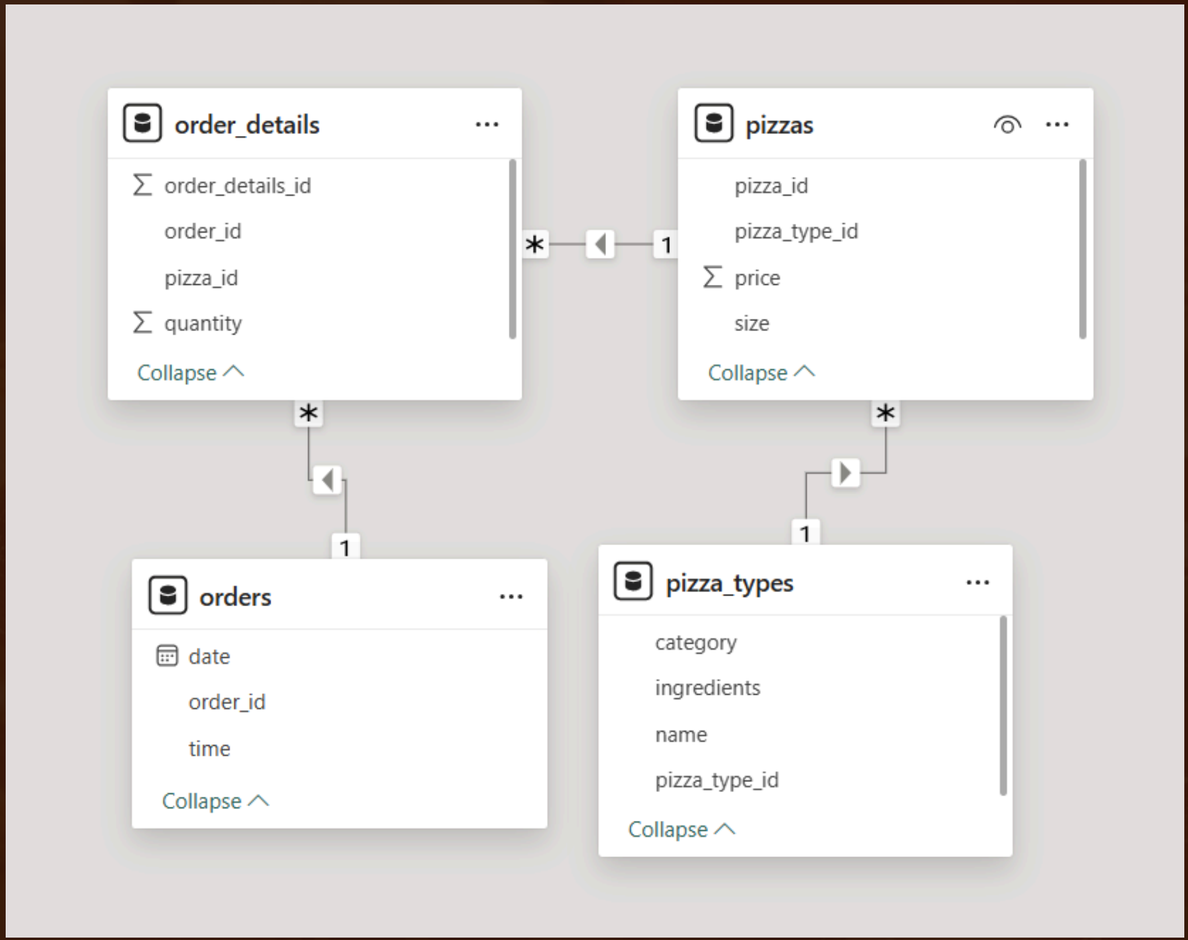
Intermediate:

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

Advanced:

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

Dataset & Schema Overview

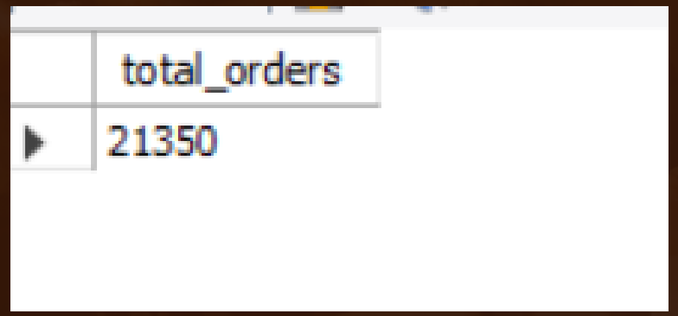


These tables are connected through primary and foreign keys, enabling analytical queries related to sales trends, pricing, Pizza's performance.

1. Retrive the total number of orders.

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

Output:



	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 order_details.pizza_id = pizzas.pizza_id;
```

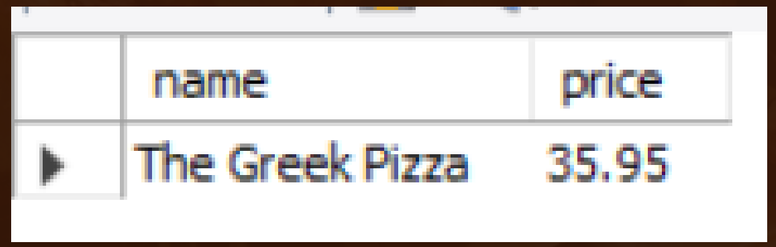
Output:

	Total_sales
▶	817860.05

3.Identify the Highest price of 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;
```

Output:

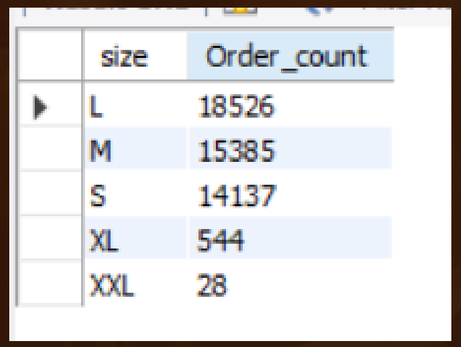


	name	price
▶	The Greek Pizza	35.95

4. Identify the most common ordered pizza size

```sql
SELECT
    pizzas.size, COUNT(order_details.order_id) AS Order_count
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY COUNT(order_details.order_id) DESC;
```

Output:



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 quantity

```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 pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY Quantity DESC
LIMIT 5;
```

Output:

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 order.

```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;
```

Output:

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

7.Determine the distribution of orders by hours of the day.

```sql
SELECT
    HOUR(order_time) AS Hours, COUNT(order_id) AS order_count
FROM
    orders
GROUP BY HOUR(order_time)
ORDER BY order_count DESC
LIMIT 5;
```

Output:

Hours	order_count
12	2520
13	2455
18	2399
17	2336
19	2009

8. Join relevant tables to find the category - wise distribution of pizza

```sql
SELECT
    category AS Pizza_Category, count(name) Total_Pizza
FROM
    pizza_types
GROUP BY category;
```

Output:

	Pizza_Category	Total_Pizza
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9

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

```sql
SELECT
    ROUND(AVG(quantity), 0) AS AVG_Number_of_Pizzas
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_quanity;
```

Output:

	AVG_Number_of_Pizzas
▶	138

10. Determine the distribution of orders by hours of the day.

```sql
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

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

```
(SELECT
    pizza_types.category,
    ROUND(SUM(order_details.quantity * pizzas.price) / (SELECT
                    SUM(pizzas.price * order_details.quantity)
                FROM
                    pizzas
                        JOIN
                    order_details ON pizzas.pizza_id = order_details.pizza_id) * 100,
            2) AS Percentage_of_Contribution_In_Revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category);
```

Output:

category	Percentage_of_Contribution_In_Revenue
Classic	26.91
Veggie	23.68
Supreme	25.46
Chicken	23.96