SQL Project : Pizza Sales Data Analysis

By Muskanpreet

PROJECT OVERVIEW

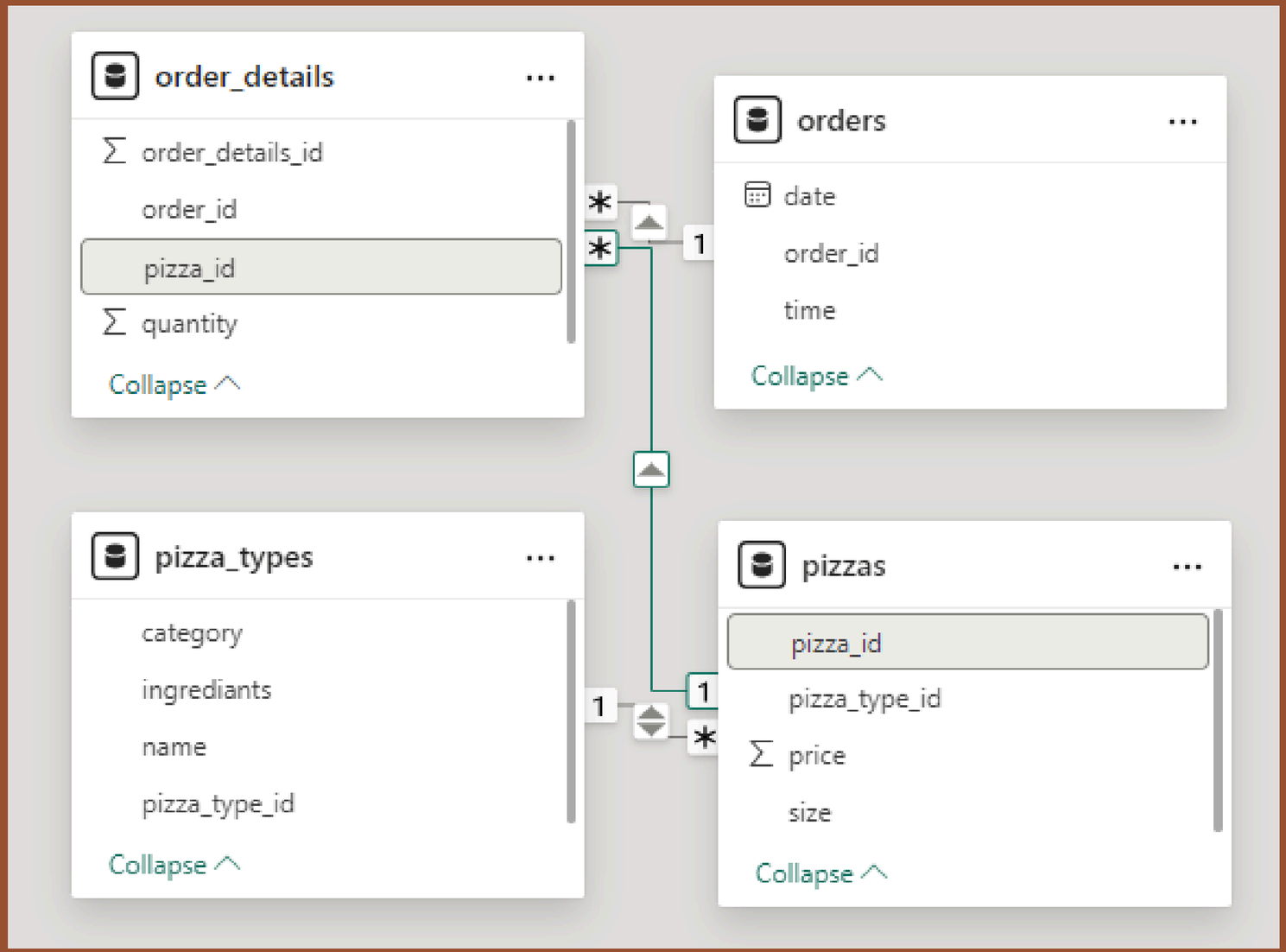
The Pizza Sales Data Analysis project aims to leverage MySQL to analyze data related to pizza sales. This project involves setting up a database to executing SQL queries to extract insights, and solving various questions related to pizza sales to help understand sales trends.

Database Overview

Our database, pizza_sales, contains several tables that store various aspects of the pizza sales operations.

1. Pizzas : This Table stores details about different types of pizzas avialable, thier prices and sizes.
2. Pizza Types : This table stores various type id's of pizzas, thier names, categories and ingrediants.
3. Orders : This Table Stores information about order id's, date and time in which order is placed.
4. Order Details : This table Stores information regarding order details, which pizza is ordered and in how much quantity.

Schema of Dataset



1. RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

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

Result Grid

	total_orders
▶	21350

2. CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

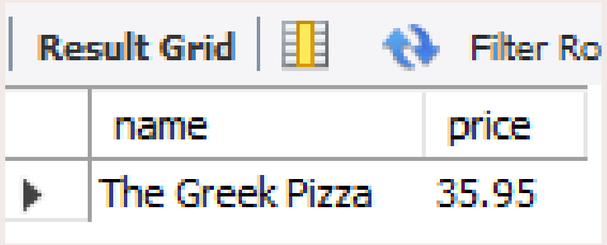
```sql
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS total_revenue
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id
```

Result Grid

	total_revenue
▶	817860.05

3. IDENTIFY THE HIGHEST-PRICED 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;
```



	name	price
▶	The Greek Pizza	35.95

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
LIMIT 1;
```

Result Grid | Filt

size	order_count
L	18526

5. LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.

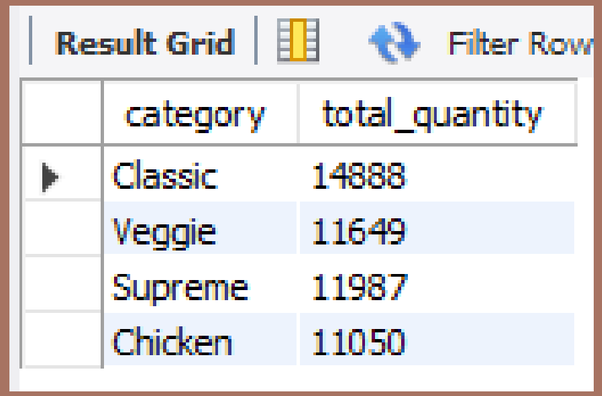
```sql
SELECT
    pizzas.pizza_type_id,
    sum(order_details.quantity) AS total_quantity
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.pizza_type_id
ORDER BY total_quantity DESC
LIMIT 5;
```

Result Grid | Filter Rows:

pizza_type_id	total_quantity
classic_dlx	2453
bbq_ckn	2432
hawaiian	2422
pepperoni	2418
thai_ckn	2371

6. JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.

```
SELECT
    pizza_types.category,
    sum(order_details.quantity) AS total_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.category;
```


Result Grid | Filter Row

category	total_quantity
Classic	14888
Veggie	11649
Supreme	11987
Chicken	11050

7. 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)
ORDER BY HOUR(order_time) ASC;
```

Result Grid | Filt

hour	order_count
9	1
10	8
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

8. JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.

```
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```

Result Grid | Filter Row

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

9. GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

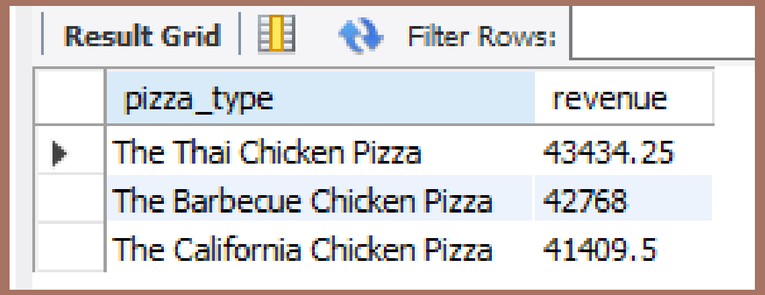
```sql
SELECT
    ROUND(AVG(quantity), 0) AS average_orders
FROM
    (SELECT
        orders.order_date, COUNT(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

Result Grid

	average_orders
▶	136

10. DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

```sql
SELECT
    pizza_types.name AS pizza_type,
    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
LIMIT 3;
```



pizza_type	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

11. 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;
```

Result Grid | Filter Rows:

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
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

THANK YOU