Home About Service Contact

Data analysis on

PIZZA SALES

using SQL

Mumbai 410-218

Home About Service Contact

HELLO !!!

I'm Khushi Singh, and in this project, I used SQL queries to analyze pizza sales and solved problems related it.

QUESTIONS

Basic:

- Retrieve the total number of orders placed.
- Calculate the total revenue generated from pizza sales.
- Identify the highest-priced pizza.
- Identify the most common pizza size ordered.
- List the top 5 most ordered pizza types along with their quantities.

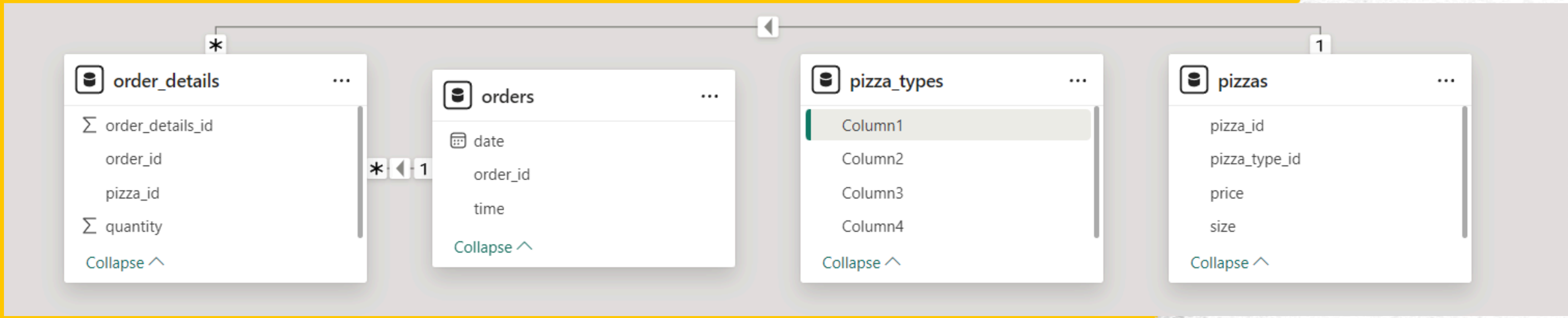
Intermediate:

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

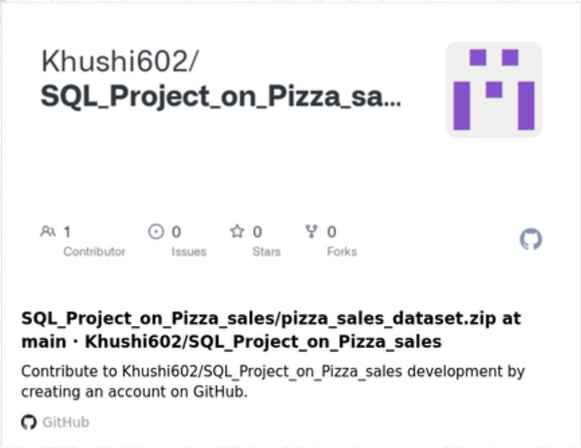
Advanced:

- Calculate the percentage contribution of each pizza type to total revenue.
- Analyze the cumulative revenue generated over time.

ABOUT DATASET



https://github.com/Khushi602/SQL_Project_on_Pizza_sales/blob/main/pizza_sales_dataset.zip



RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

```sql
-- Retrieve the total number of orders placed.

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

Query

Result Grid | Filter Rows:

	total_orders
▶	21350

Output

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

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

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
```

Query

Result Grid

	Total_Revenue
▶	817860.05

Output

IDENTIFY THE HIGHEST-PRICED PIZZA.

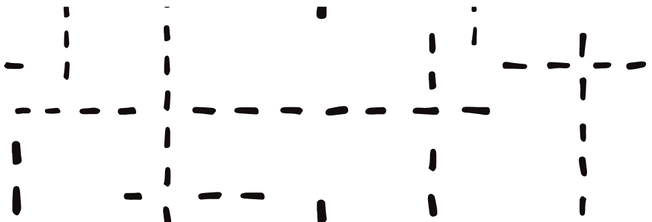
```sql
-- Identify the highest-priced pizza.

SELECT
    pizza_types.name as Name, pizzas.price as Price
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```

Query

Result Grid | Filter Rows:

	Name	Price
▶	The Greek Pizza	35.95

Output



IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

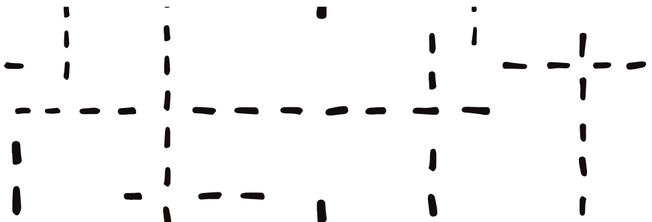
```sql
-- Identify the most common pizza size ordered.

SELECT
    pizzas.size as 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;
```

Query

Result Grid | Filter Rc

Size	Order_count
L	18526

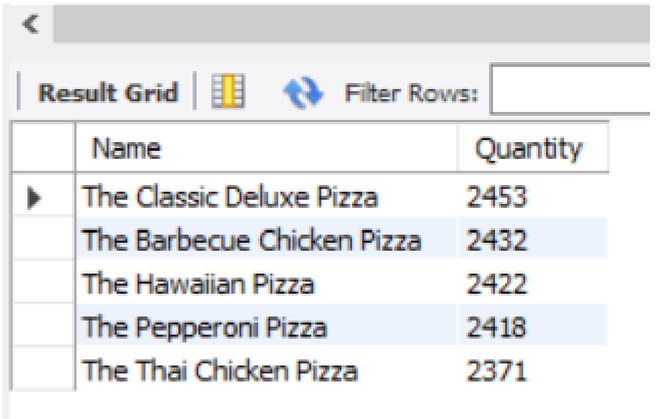
Output



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 as 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;
```

Query



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

Output

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

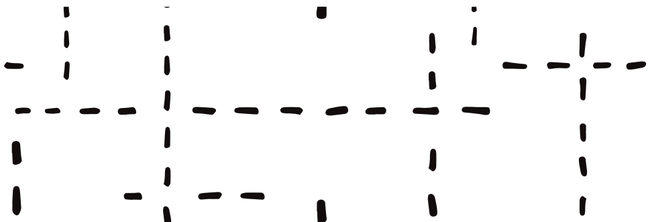
```sql
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY quantity DESC;
```

Query

Result Grid | Filter Rows

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Output



DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

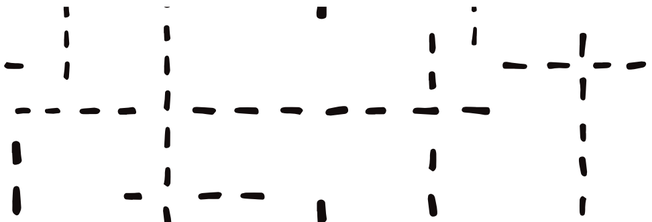
```sql
-- Determine the distribution of orders by hour of the day.

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

Query

Result Grid | Filter Rows:

Hours	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

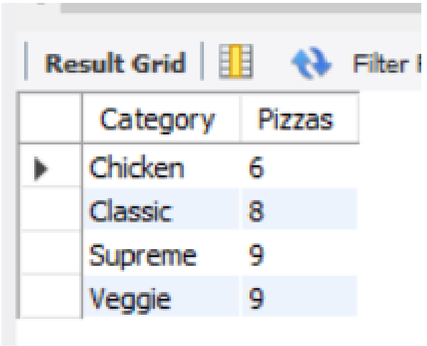
Output



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

```
-- Join relevant tables to find the category-wise distribution of pizzas.

SELECT
    pizza_types.category as Category, COUNT(name) AS Pizzas
FROM
    pizza_types
GROUP BY category
ORDER BY category ASC;
```

Query

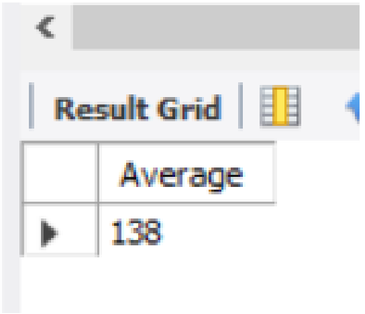


Output

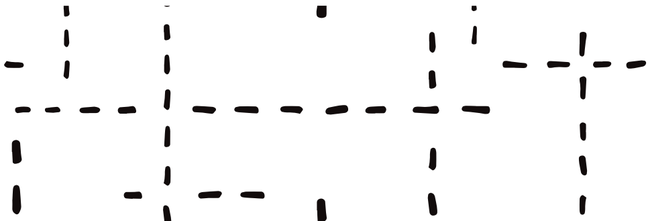
GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

```sql
-- Group the orders by date and calculate the average number of pizzas ordered per day.

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

Query



Output



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 as 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.name
ORDER BY revenue DESC
LIMIT 3;
```

Query

Result Grid | Filter Rows:

Name	Revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

Output

CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE.

```sql
SELECT
    pizza_types.category,
    ROUND(SUM(order_details.quantity * pizzas.price) / (SELECT
                    SUM(order_details.quantity * pizzas.price)
                FROM
                    order_details
                        JOIN
                    pizzas ON pizzas.pizza_id = order_details.pizza_id) * 100,
            2) AS revenue_percentage
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_percentage DESC;
```

Query

Result Grid | Filter Rows:

category	Revenue_percentage
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Output

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

```sql
-- 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;
```

Query

Result Grid | Filter Rows: | Export:

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
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

Output

THANK YOU