Pizza Sales

Introduction

Hi, my name is Swastik, in this project I have delved into the analysis of pizza sales data, focusing on understanding the most popular pizza types ordered and examining the ordering details associated with these transactions. Leveraging MySQL Workbench, I have employed SQL queries to extract specific insights from the dataset provided by WSCube Tech.

Problem Statement

The primary objective of this project is to extract actionable insights from the pizza sales dataset, with a focus on determining the most frequently ordered pizza types and analyzing the associated ordering details. By doing so, I have aimed to gain a deeper understanding of customer preferences and behaviors within the context of pizza consumption.

Approach and Methodology

My approach involves leveraging SQL queries within MySQL Workbench to extract, manipulate, and analyze the dataset. I employed various SQL techniques to aggregate data, calculate metrics, and derive insights pertinent to our research objectives. Throughout the project, I adhere to best practices in database querying and data analysis to ensure accuracy and reliability of our findings.

1. Retrieve the total number of orders placed.

```
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.

```
SELECT
    pizza_id, price
FROM
    pizzas
ORDER BY price DESC
LIMIT 1;
```

Result Grid | Filter Rov

	pizza_id	price
▶	the_greek_xxl	35.95

4. Identify the highest-priced pizza.

```sql
SELECT
    pizzas.size, count(order_details.quantity) AS commonly_ordered
FROM
    pizzas
        JOIN
    order_details on pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
order by commonly_ordered desc
LIMIT 1;
```

Result Grid | Filter Rows:

size	commonly_ordered
L	18526

5. List the top 5 most ordered pizza types along with their quantities..

```sql
SELECT
    pizza_types.name,
    SUM(order_details.quantity) AS total_ordered
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
    total_ordered DESC
    limit 5;
```

Result Grid Filter Rows:

name	total_ordered
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 ordered.

```sql
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 order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY total_quantity DESC;
```

Result Grid | Filter Rows

category	total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

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

```sql
SELECT
    HOUR(order_time) AS order_hour,
    COUNT(order_id) AS order_count
FROM
    orders
GROUP BY HOUR(order_time)
```

Result Grid | Filter Rows:

order_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

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

```sql
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

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

```sql
SELECT
    ROUND(AVG(total_quantity), 0) as avg_order_per_day
FROM
    (SELECT
        orders.order_date,
            SUM(order_details.quantity) AS total_quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS order_quantity
```

Result Grid | Filte

avg_order_per_day
138

10.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:

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

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

```
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 category
ORDER BY revenue DESC
```

Result Grid | Filter

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

12. Analyze the cumulative revenue generated over time.

```
select order_date,
sum(Revenue) over(order by order_date) as cumm_revenue
from
(Select orders.order_date, Sum(order_details.quantity * pizzas.price) as Revenue
from order_details join pizzas
on pizzas.pizza_id = order_details.pizza_id
join orders
on order_details.order_id = orders.order_id
group by orders.order_date) as Sales
```

Result Grid | Filter Rows:

order_date	cumm_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

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

```sql
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 pizzas.pizza_type_id = pizza_types.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 b
where rn <= 3;
```

Result Grid | Filter Rows:

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