PIZZA SALES ANALYSIS

Introduction

This project is based on a Pizza Sales Dataset and focuses on analyzing sales performance using SQL. The goal was to explore business insights from raw transactional data by solving queries ranging from basic to advanced levels.

Problem 1

Retrieve the total number of orders placed.

Solution

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

Result Grid

total_orders
21350

Problem 2

Calculate the total revenue generated from pizza sales.

Solution

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

Result Grid

Total_Revenue
817860.05

Problem 3

Identify the highest-priced pizza.

Solution

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

Result Grid | Filter Rows:

name	price
The Greek Pizza	35.95

Problem 4 Identify the most common pizza size ordered.

Solution

```sql
SELECT
    pizzas.size, COUNT(order_details.quantity) 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 Filter

size	order_count
L	18526

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

Solution

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

Result Grid | Filter Rows:

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

Problem 6 Join the necessary tables to find the total quantity of each pizza category ordered.

Solution

```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;
```

Result Grid | Filter Rov

category	Quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

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

Solution

```sql
SELECT
    HOUR(order_time), COUNT(order_id)
FROM
    orders
GROUP BY HOUR(order_time);
```

Result Grid | Filter Rows:

HOUR(order_time)	COUNT(order_id)
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920

1231

Problem 8

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

Solution

```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

Problem 9

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

Solution

```sql
SELECT
    ROUND(AVG(quantity), 0)
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_quantity;
```

Result Grid | Filter Rows:

ROUND(AVG(quantity), 0)
138

Problem 10

Determine the top 3 most ordered pizza types based on revenue.

Solution

```sql
SELECT
    pizza_types.name,
    SUM(pizzas.price * order_details.quantity) AS Revenue
FROM
    order_details
        JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id
        JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_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

Problem 11

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

Solution

```
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 order_details.pizza_id = pizzas.pizza_id) * 100,
            2) AS Revenue
FROM
    pizzas
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
        JOIN
    pizza_types ON pizza_types.pizza_type_id = pizzas.pizza_type_id
GROUP BY category;
```

Result Grid | Filter F

category	Revenue
Classic	26.91
Veggie	23.68
Supreme	25.46
Chicken	23.96

Problem 12

Analyze the cumulative revenue generated over time.

Solution

```
select order_date,sum(revenue) over(order by order_date) as cumulative_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	cumulative_revenue
2015-01-01	2713.850000000004
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

Problem 13

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

Solution

Result Grid | Filter Rows: | Export:

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

```
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((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.category,pizza_types.name) as a) as b
where rn <=3;
```

THANK YOU

Project By:
Sunil Shetty