

Pizza Sales Project Using SQL



Here I have analysed various trends from pizza sales data such as total sales, average sales per day and so on.

Introduction

- The Pizza Sales Data Analysis project focuses on examining sales performance and customer purchasing patterns for a fictional pizza restaurant using Structured Query Language (SQL). The primary objective of this analysis is to extract valuable business insights that can guide decision-making in areas such as inventory management, marketing strategies, and menu optimization.
- This project demonstrates how SQL can be effectively used to clean, manipulate, and analyze relational data stored in a database, offering a practical example of how data analytics can drive operational improvements in the food and beverage industry.

Q-1) Retrieve the total number of orders placed.

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

Result Grid			
	total_orders		
1	21350		

Q-2) Calculate the total revenue generated from pizza sales.

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

Result Grid	
	total_revenue
▶	817860.05

Q-3) Identify the highest-priced pizza.

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

Result Grid			Filter Rows:	
	name	price		
▶	The Greek Pizza	35.95		

Q-4) Identify the most common pizza size ordered.

```
SELECT
    pizzas.size,
    COUNT(order_details.order_details_id) AS order_count
FROM
    pizzas
    INNER JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC;
```

Result Grid			Filter Rows:
	size	order_count	
	L	18526	
	M	15385	
	S	14137	
	XL	544	
	XXL	28	

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

```
SELECT
    pizza_types.name, SUM(order_details.quantity) AS quantity
FROM
    pizza_types
    INNER JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
    INNER JOIN
    order_details ON pizzas.pizza_id = order_details.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	

Q-6) Join the necessary tables to find the total quantity of each pizza category ordered.

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

Result Grid			Filter
	category	quantity	
▶	Classic	14888	
	Supreme	11987	
	Veggie	11649	
	Chicken	11050	



Q-7) Determine the distribution of orders by hour of the day.

```
SELECT
    HOUR(order_time) AS hour, COUNT(order_id) AS order_count
FROM
    orders
GROUP BY hour;
```

Result Grid			Filter
	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	



Q-8) Join relevant tables to find the category-wise distribution of pizzas.

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

Result Grid   Filter Rows:		
	category	pizza_count
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9

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

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

Result Grid			 Filter Rows:
	pizza_ordered_per_day_avg		
	138		

Q-10) Determine the top 3 most ordered pizza types based on revenue.

```
SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
    INNER JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
    INNER 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	

Q-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
        INNER JOIN
        pizzas ON order_details.pizza_id = pizzas.pizza_id) * 100,
    2) AS percent_revenue
FROM
    pizza_types
    INNER JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
    INNER JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY percent_revenue DESC;
```

Result Grid





Filter Rows:

	category	percent_revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68



Q-12) Analyze the cumulative revenue generated over time.

```
select order_date,  
sum(revenue) over(order by order_date) as cumulative_revenue  
from  
(select orders.order_date,  
round(sum(order_details.quantity * pizzas.price), 2) as revenue  
from order_details inner join pizzas  
on order_details.pizza_id = pizzas.pizza_id  
inner join orders  
on orders.order_id = order_details.order_id  
group by orders.order_date  
order by revenue desc) as sales
```

Result Grid				 Filter Rows: <input type="text"/>
	order_date	cumulative_revenue		
▶	2015-01-01	2713.85		
	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	21526.399999999998		
	2015-01-10	23990.35		
	2015-01-11	25862.649999999998		
	2015-01-12	27781.699999999997		
	2015-01-13	29831.299999999996		
	2015-01-14	32358.699999999997		
	2015-01-15	34343.5		
	2015-01-16	36937.65		
	2015-01-17	39001.75		

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

```
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 inner join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
inner 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: <input data-bbox="1324 406 1605 471" type="text"/> Export		
	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	32265.700000000065
	The Mexicana Pizza	26780.75
	The Five Cheese Pizza	26066.5