

Pizza sales Analysis Using Sql

BY NISHANT GORE

Description Of Projet

Pizza Sales Analysis

This project analyzes pizza sales data to uncover trends, revenue insights, and customer preferences. Using four datasets, we explored order patterns, revenue generation, and pizza popularity. The findings

- Total orders, revenue, and popular pizza types.
 Distribution of orders by time and category.
- Top-performing pizzas by revenue.

Key SQL concepts applied:

- Data extraction with SELECT queries.
- Joins to combine multiple datasets.
- 3. Aggregations (SUM, COUNT, AVG) for insights.
- Filtering data using WHERE and HAVING clauses.
- Grouping data with GROUP BY.
- 6. Calculating percentages and rankings for advanced analysis.

Key insights support strategic decision-making for pricing, promotions, and operational efficiency.



TABLES FOR DATASETS

pizza.csv dataset

mysql> create table pizza(

- -> pizza_id varchar(30) primary key
- -> pizza_type_id varchar(20),
- -> size varchar(10),
- -> price float);

Query OK, 0 rows affected (0.11 sec)

order.csv dataset

mysql> create table orders(

- -> order_id int primary key,
- -> date date,
- -> time time

1:

Query OK, 0 rows affected (0.04 sec)

order_details .csv dataset

mysql> create table order_details(

- -> orders_details_id int primary key,
 - -> order_id int.
 - -> pizza_id varchar(30),
 - -> quantity int

1:

Query OK, 0 rows affected (0.04 sec)

pizza_types .csv dataset

mysql> create table pizza_types(

- -> pizza_type_id varchar(30) primary key,
- -> name varchar(50),
- -> category varchar(30),
- -> ingredient varchar(200));

Query OK, 0 rows affected (0.03 sec)



Retrieve the total number of orders placed.

QUERY:=>

select count(order_id) as total_orders from orders;

output:->

| total_orders | | +------+ | 21350 |



Calculate the total revenue generated from pizza sales

QUERY:=>

select round(sum(order_details.quantity* pizza.price),2)as total_revenue from order_details join pizza on order_details.pizza_id=pizza.pizza_id;

output:->



Identify the highest-priced pizza.

QUERY:=>

select pizza_types.name, pizza.price from pizza_types join pizza on pizza_types.pizza_type_id =pizza.pizza_type_id order by pizza.price desc limit 1;

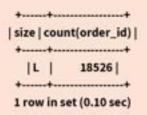
output:->

| name | price | | the Greek Pizza | 35.95 |

Identify the most common pizza size ordered.

Queriies:=>

select pizza.size,count(order_id) from pizza join order_details on pizza.pizza_id=order_details.pizza_id group by (pizza.size) order by count(order_id) desc limit 1;







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

QUERY:=>

select pizza_types .name,sum(order_details.quantity) from
pizza_types join pizza on
pizza_types.pizza_type_id=pizza.pizza_type_id
-> join order_details on order_details.pizza_id=pizza.pizza_id group
by pizza_types.name order by sum(order_details.quantity) desc limit 1;

output:->

+	***************************************	***************************************
name	sum(order_deta	ils.quantity)
+	+	+
The Classic Deluxe Pizza 2453		
t		

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

Queriies:=>

select pizza_types .category,sum(order_details.quantity) as quantity
from pizza_types join pizza on
pizza_types.pizza_type_id=pizza.pizza_type_id
-> join order_details on order_details.pizza_id=pizza.pizza_id
group by pizza_types.category order by sum(order_details.quantity)
desc;

Output:=>

category	quantity
category	quantity
Classic	13529
Supreme	11987
Chicken	11050
Veggie	10652
Mushroom	1359





Calculate the total revenue generated from pizza sales

QUERY:=>

select round(sum(order_details.quantity* pizza.price),2)as total_revenue from order_details join pizza on order_details.pizza_id=pizza.pizza_id;

output:->



Determine the distribution of orders by hour of the day

Queriies:=>

select hour(time),count(order_id) as orders from orders group by(hour(time)) order by hour(time) asc;

hour(time) orders			
1	9 1		
- 1	10 8		
- 1	11 1231		
1	12 2520		
13	13 2455		
1	14 1472		
- 10	15 1468		
- 1	15 1468 16 1920		
	17 2336		
1/3	18 2399		
13	19 2009		
100	20 1642		
1	21 1198		
1	22 663		
1	23 28		



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

Queriies:=>

select date,sum(quantity) from orders join order_details on orders.order_id=order_details.order_id group by(date) limit 5;

date sum	(quantity)
2015-01-01	162
2015-01-02	165
2015-01-03	158
2015-01-04	106
2015-01-05	125



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

Queriies:=>

select pizza_types.name
,round(sum(pizza.price*order_details.quantity),0) as revenue from
pizza_types join pizza on pizza_types.pizza_type_id=pizza.pizza_type_id
join order_details on pizza.pizza_id=order_details.pizza_id group
by(pizza_types.name) order by revenue desc limit 3;

Output:=>

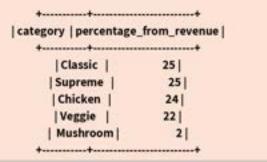
name	revenue
the Thai Chicken Pizza	43434
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41410



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

Queriies:=>

- select sum(pizza.price*order_details.quantity) into @total from pizza join order_details on pizza.pizza_id=order_details.pizza_id;
- 2] select pizza_types.category ,round[(sum(pizza.price*order_details.quantity)|@total[*100,0) as percentage_from_revenue from pizza_types join pizza on pizza_types.pizza_type_id=pizza_type_id join order_details on pizza_pizza_id=order_details.pizza_id group by(pizza_types.category) order by percentage_from_revenue desc;





Analyze the cumulative revenue generated over time

Queriies:=>

select sales.date,sum(revenue) over (order by sales.date) as commulative_revenue from(

-> select orders.date,sum(pizza.price*order_details.quantity) as revenue from orders join order_details on orders.order_id =order_details.order_id join pizza on order_details.pizza_id=pizza.pizza_id group by(date) order by sum(pizza.price*order_details.quantity) desc)as sales limit 5;

Output:=>



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

|category | name 1 Queriies:=> SELECT pizza_types.category, -> pizza_types.name, -> SUM(pizza.price * order_details.quantity) AS revenue -> pizza_types ⇒ pizza ON pizza_types.pizza_type_id = pizza.pizza_type_id -> JOIN order_details ON pizza.pizza_id = order_details.pizza_id -> GROUP BY -> pizza_types.category, pizza_types.name -> ORDER BY -> revenue DESC;



Thanks a lot