
SQL
PROJECT

INTRODUCTION

- In this project, I undertook a comprehensive analysis of a pizza sales dataset, which consisted of four distinct data sheets provided in an Excel file. By importing these data sheets into an SQL database, I was able to write and execute various queries to extract meaningful insights and answers to twelve specific questions related to pizza sales.

- The primary objective of this project was to leverage SQL to perform data analysis and uncover patterns and trends within the dataset. Each query was designed to address a particular question, ranging from calculating the total number of orders placed to identifying the top-performing pizza categories.

- The following presentation showcases the SQL queries used and their corresponding outputs, providing a clear and concise overview of the results obtained from this analysis. Each slide presents a query along with its result, illustrating the step-by-step approach taken to answer the twelve questions posed in this project.

QUESTIONS

RETRIEVE TOTAL NUMBERS OF ORDERS PLACED

```sql
select count(order_id) as Total_orders
from orders;
```

SQL Querry

Solution Table

Total_orders
21350

CALCULATE TOTAL REVENUE GENERATED FROM PIZZA SALES

```sql
select
round(sum(order_details.quantity * pizzas.price),2) as total_sales
from order_details join pizzas
on pizzas.pizza_id = order_details.pizza_id;
```

SQL Querry

Solution Table

total_sales
817860.05

IDENTIFY THE PIZZA WITH HIGHEST PRICE

```sql
select pizzas.price, pizza_types.name
from pizzas join pizza_types
on pizzas.pizza_type_id = pizza_types.pizza_type_id
order by pizzas.price desc limit 1
```

SQL Querry

Solution Table

price	name
35.95	The Greek Pizza

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED

```sql
select pizzas.size, count(order_details.order_details_id)
from pizzas join order_details
on pizzas.pizza_id = order_details.pizza_id
group by pizzas.size order by count(order_details.order_details_id) desc limit 1;
```

SQL Querry

Solution Table

size	count(order_details.order_details_id)
L	18526

LIST THE TOP 5 MOST ORDERED PIZZA TYPE ALONG WITH THEIR QUANTITIES

```sql
Select pizza_types.name, sum(order_details.quantity) as quantity
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 pizza_types.name order by quantity Desc limit 5
```

SQL Querry

Solution Table

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

FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED

```sql
select pizza_types.category, sum(order_details.quantity)
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  pizza_types.category
```

SQL Querry

Solution Table

category	sum(order_details.quantity
Classic	14888
Veggie	11649
Supreme	11987
Chicken	11050

DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

```sql
select hour(order_time), count(order_id)
from orders
group by hour(order_time)
```

SQL Querry

hour(order_time)	count(order_id)
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

Solution Table

FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.

```sql
select category, count(pizza_type_id) as num_of_pizzas
from pizza_types
group by category
```

SQL Querry

Solution Table

category	num_of_pizzas
Chicken	6
Classic	8
Supreme	9
Veggie	9

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

```sql
select avg(quantity) from
(select orders.order_date, count(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;
```

SQL Querry

Solution Table

	avg(quantity)
▶	135.8101

DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

```sql
select pizza_types.name, sum(order_details.quantity * pizzas.price)
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 sum(order_details.quantity * pizzas.price) desc limit 3;
```

SQL Querry

Solution Table

name	sum(order_details.quantity * pizzas.price)
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

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

```sql
select category, (revenue / total_revenue) * 100 as percentage_revenue
from (select pizza_types.category as category, sum(order_details.quantity * pizzas.price) as revenue,
(select sum(order_details.quantity * pizzas.price)
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) as total_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) as pizza_revenue
group by category, total_revenue;
```

SQL Querry

Solution Table

category	percentage_revenue
Classic	26.905960255669903
Veggie	23.682590927384783
Supreme	25.45631126009884
Chicken	23.955137556847493

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

```sql
select date, sum(revenue) over(order by date) as cum_revenue from
(select orders.order_date as 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 order_sum;
```

SQL Querry

The solution table exceeds the allowable size for display in this context.

Thank You