PIZZA SALES ANALYSIS

USING SQL

PRESENTATION BY

KHUSHBOO

CONTENTS

- overview
- DATA SOURCE AND METHODOLOGY
- UPLOADING CSVS INTO sql server
- DATA MODELLING (ER DIAGRAM)
- INSIGHT SLIDES (SQL QUERIES & THEIR SIGNIFICANCE)
- CONCLUSION

OVERVIEW

- *Diving into the world of pizza sales to analyze customer behavior.*
- *Studying sales data to identify key metrics and patterns.*
- Understanding how menu items and promotions influence customer choices.
- *Providing insights to help Pizza Hut enhance its sales strategies.*
- *Embarking on an exciting journey of decoding pizza sale behavior together!*

DATASETS DESCRIPTIONS

Orders

Field	Type	Null	Key	Default	Description
order_id	int	NO	PRI	NULL	Unique identifier for each order.
order_date	date	NO		NULL	Date when the order was placed.
order_time	time	NO		NULL	Time at which the order was placed.

Orders_Details

Field	Type	Null	Key	Default	Description
order_details_id	int	NO	PRI	NULL	Unique identifier for each order detail.
order_id	int	NO		NULL	Unique identifier for each order.
pizza_id	text	NO		NULL	Identifier for the type of pizza ordered.
quantity	int	YES		NULL	Quantity of the specific pizza ordered.

Pizza_Types1

Field	Type	Null	Key	Default	Description
pizza_type_id	text	YES		NULL	Identifier for the type of pizza.
name	text	YES		NULL	Name of the pizza type.
category	text	YES		NULL	Category of the pizza (e.g., vegetarian).
ingredients	text	YES		NULL	Ingredients used in the pizza.

Pizzas

Field	Type	Null	Key	Default	Description
pizza_id	text	YES		NULL	Unique identifier for each pizza.
pizza_type_id	text	YES		NULL	Identifier for the type of pizza.
size	text	YES		NULL	Size of the pizza (e.g., small, medium).
price	double	YES		NULL	Price of the pizza.

01

RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

```
select count(order_id)
as total_order from
orders;
```

OUTPUT

Total_Orders
21350

02 CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

QUERY

```
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
```

OUTPUT

Total_Sales
817860.05

Significance: Total pizza sales revenue measures financial performance, guiding strategic decisions for Pizza Hut.

03

IDENTIFY THE HIGHEST-PRICED PIZZA.

OUTPUT

Name	Total
The Greek Pizza	109.95

```
select TOP 1 pizza_types1.name,pizzas.price
from pizza_types1
join
pizzas
on
pizza_types1.pizza_type_id=pizzas.pizza_type_id
order by pizzas.price desc ;
```

04 IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

```
select pizzas.size,
count(order_details.order_details_id) 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;
```

OUTPUT

Size	Total
L	18956
M	15635
S	14403
XL	552
XXL	28

05 LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.

PREPARATION

```
select top 5 pizza_types1.name,
sum(order_details.quantity) as quantity
from pizza_types1 join pizzas
on
pizza_types1.pizza_type_id=pizzas.pizza_type_id
join order_details
on order_details.pizza_id=pizzas.pizza_id
group by
pizza_types1.name order by quantity desc;
```

Name	Total_Quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

06 JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.

```
select pizza_types1.category,
sum(order_details.quantity ) as  quantity from
pizza_types1 join pizzas
on
pizza_types1.pizza_type_id=pizzas.pizza_type_id
join order_details on
order_details.pizza_id=pizzas.pizza_id
group by pizza_types1.category order by quantity
desc;
```

OUTPUT

Category	Total_Quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

07 DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

```
select datepart(HOUR,[order_time]) as
hourtime,count(order_id) as order_count
from orders
group by datepart(hh,[order_time])
order by order_count desc;
```

OUTPUT

Hours	Orders
12	2520
13	2455
18	2399
17	2336
19	2009
16	1920
20	1642
14	1472
15	1468
11	1231
21	1198
22	663
23	28
10	8
9	1

08 JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.

```
SELECT
pizza_types1.CATEGORY,count(order_details.order_id)as
num_ordrs from
pizza_types1 join pizzas
on
pizza_types1.pizza_type_id=pizzas.pizza_type_id
join  order_details on
order_details.pizza_id=pizzas.pizza_id
group by pizza_types1.category order by num_ordrs desc
```

OUTPUT

Category	Orders
Classic	14579
Supreme	11777
Veggie	11449
Chicken	10815

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

QUERY

```
select round(avg(quantity),0)as
Avg_pizza_Perday from
(select orders.order_date,
sum(order_details.quantity )as quantity
from orders join order_details
on
orders.order_id=order_details.ord
er_id
group by orders.order_date) as
order_quantity ;
```

OUTPUT

Avg_Pizza_Ordered_Per_Day
138

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

PREPARATION

```
select top 3 pizza_types1.name as top_3,
sum(order_details.quantity*pizzas.price)as revenue
from pizza_types1 join pizzas
on
pizzas.pizza_type_id=pizza_types1.pizza_type_id
join order_details
on
order_details.pizza_id=pizzas.pizza_id
group by pizza_types1.name order by revenue desc;
```

OUTPUT

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.

QUERY

```
select pizza_types1.category,
round(sum(order_details.quantity*pizzas.price)/
(select round(sum(order_details.quantity*pizzas.price),2)as
total
from
order_details
join
pizzas on pizzas.pizza_id=order_details.pizza_id)*100,2)as
revenue
from pizza_types1 join pizzas on
pizza_types1.pizza_type_id=pizzas.pizza_type_id
join order_details on
order_details.pizza_id=pizzas.pizza_id
group by pizza_types1.category order by revenue desc;
```

OUTPUT

Category	Revenue	Percentage_Revenue
Classic	220053.1	26.91
Supreme	208197	25.46
Chicken	195919.5	23.96
Veggie	193690.45	23.68

12 ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

PREPARATION

QUERY

```
select order_date,
sum(revenue) over (order by order_date) as
cum_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 revenue;
```

OUTPUT

order_date	Revenue	Cum_Revenue
01-01-2015	2713.85	2713.85
02-01-2015	2731.9	5445.75
03-01-2015	2662.4	8108.15
04-01-2015	1755.45	9863.6

13

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

```
select category,name ,revenue from
(select category,name,revenue,
rank() over ( partition by category order
by revenue desc)
as top_rank
from
(select
pizza_types1.name,pizza_types1.category,
sum((order_details.quantity) *
pizzas.price) as revenue
from
pizza_types1 join pizzas
on
pizza_types1.pizza_type_id=pizzas.pizza_type
_id
join  order_details
on order_details.pizza_id=pizzas.pizza_id
group by
pizza_types1.name,pizza_types1.category)as
a)as b
where top_rank<=3;
```

OUTPUT

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

CONCLUSION

Our project utilized Pizza Sales Analysis data, harnessing SQL for efficient database management. Through meticulous data preparation and SQL analysis, we addressed key inquiries, revealing essential insights into pizza sales behavior.

These insights, ranging from popular pizza types to revenue trends, provide actionable implications for menu optimization and marketing strategies. Our project highlights the versatility of SQL in handling complex datasets, emphasizing the importance of systematic analysis in shaping Pizza Hut's strategies.

The outcomes of this project have the potential to drive decision-making within Pizza Hut, showcasing the value of rigorous data analysis within MySQL environments for the food industry.