Pizza Sales Analysis using MySQL

Description :

The Pizza Analysis Project is designed to showcase how SQL can be used to analyze sales data for a pizza business. The project focuses on implementing database management techniques and executing queries to extract valuable insights. By analyzing customer orders, pizza preferences, and revenue trends, the project provides a practical demonstration of data analytics using relational databases .

Tables in Dataset

- Pizza_Types
- Order_details
- Orders
- Pizzas

Table_Name : Pizza_Types

Columns in Table :

- Pizza_type_id (text) (Primary Key)
- Name (text)
- Category (text)
- Ingridients (text)

	Field	Type	Null	Key	Default	Extra
▶	pizza_type_id	text	YES		NULL	
	name	text	YES		NULL	
	category	text	YES		NULL	
	ingredients	text	YES		NULL	

Table_Name : Order_details

Columns in Table :

- order-details_id (int) (Primary Key)
- order_id (int) (Foreign Key)
- pizza_id (text) (Foreign Key)
- quantity (int)

	Field	Type	Null	Key	Default	Extra
▶	order_details_id	int	YES		NULL	
	order_id	int	YES		NULL	
	pizza_id	text	YES		NULL	
	quantity	int	YES		NULL	

Table_Name : Orders

Columns in Table :

- order_id (int) (Primary Key)
- Order_time (time)
- Order_date (date)

	Field	Type	Null	Key	Default	Extra
▶	order_id	int	YES		NULL	
	date	text	YES		NULL	
	time	text	YES		NULL	

Table_Name : Pizzas

Columns in Table :

- Pizza_id (text) (Primary Key)
- Pizza_type_id (text)
- Size (text)
- Price (double)

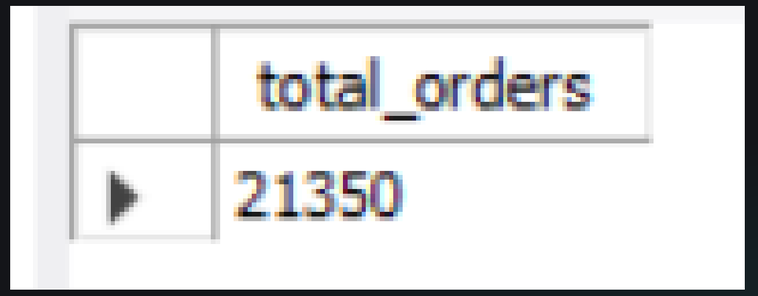
	Field	Type	Null	Key	Default	Extra
▶	pizza_type_id	text	YES		NULL	
	name	text	YES		NULL	
	category	text	YES		NULL	
	ingredients	text	YES		NULL	

Retrive total number of order placed

Solution :

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

MySQL Workbench Output :



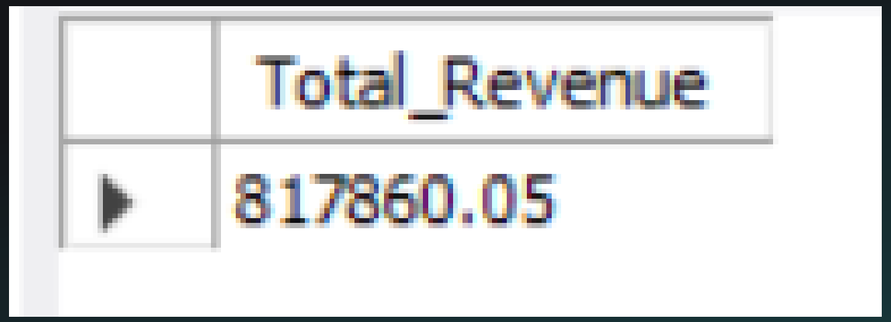
total_orders
21350

Calculate total revenue genrated from pizza sale

Solution :

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

MySQL Workbench Output :



	Total_Revenue
▶	817860.05

Identify the highest price pizza

Solution :

```
SELECT pizza_types.name, pizzas.price AS price
FROM  pizza_types
JOIN  pizzas
ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY price DESC
LIMIT 5;
```

MySQL Workbench Output :

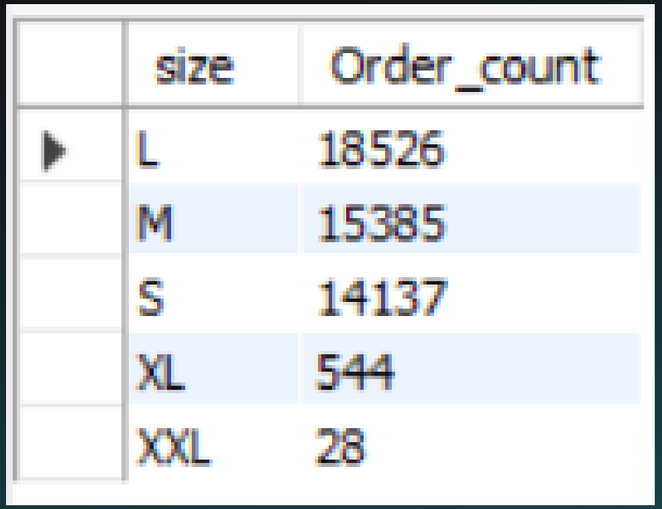
name	price
The Greek Pizza	35.95
The Greek Pizza	25.5
The Brie Carre Pizza	23.65
The Italian Vegetables Pizza	21
The Barbecue Chicken Pizza	20.75

Identify most comman pizza size ordered

Solution :

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

MySQL Workbench Output :



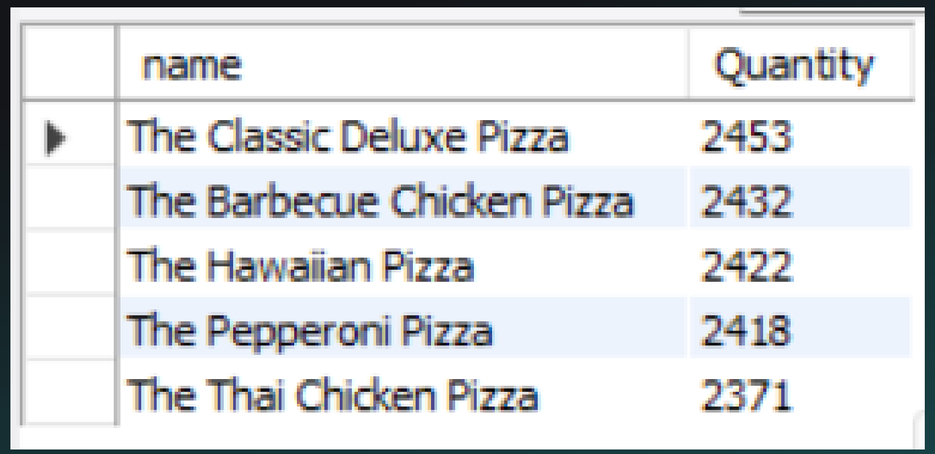
size	Order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

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

MySQL Workbench Output :



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

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

Solution :

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

MySQL Workbench Output :

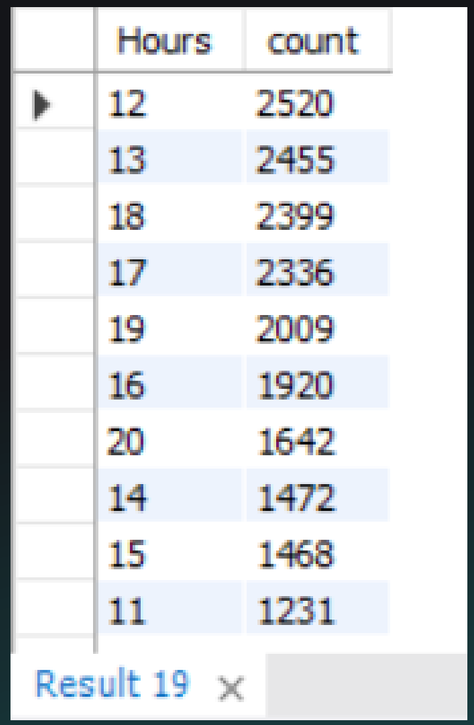
Category	Quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Determine the distribution of orders by hours of the day

Solution :

```
SELECT  HOUR(orders.time) AS Hours, COUNT(order_id) AS count
FROM  orders
GROUP BY Hours
ORDER BY count DESC;
```

MySQL Workbench Output :



Hours	count
12	2520
13	2455
18	2399
17	2336
19	2009
16	1920
20	1642
14	1472
15	1468
11	1231

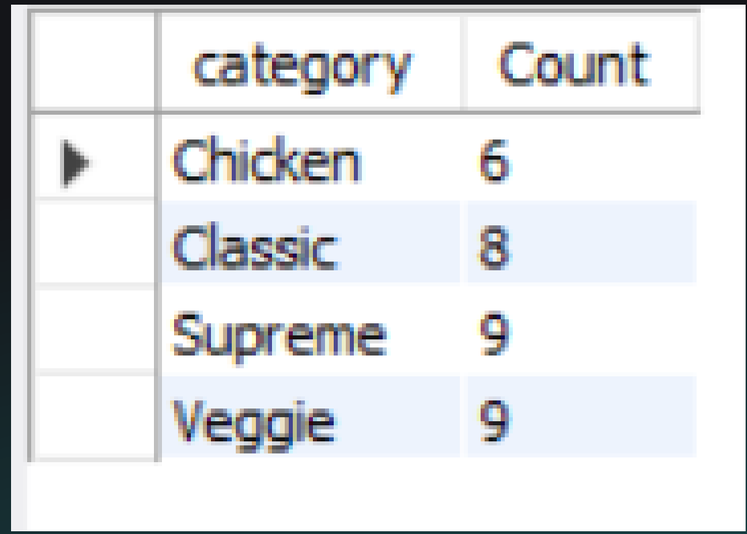
Result 19

Join relevant tables to find the category wise distribution of pizzas

Solution :

```
SELECT category, COUNT(name) as Count
FROM  pizza_types
GROUP BY category;
```

MySQL Workbench Output :



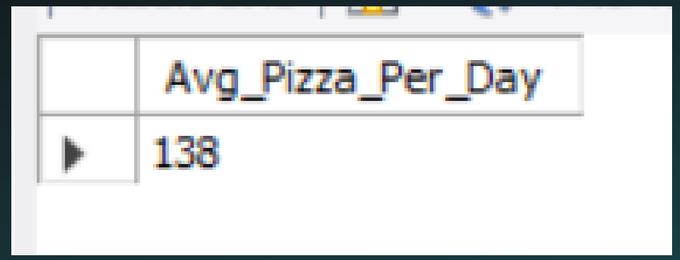
category	Count
Chicken	6
Classic	8
Supreme	9
Veggie	9

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

Solution :

```
SELECT  ROUND(AVG(Quantity), 0) as Avg_Pizza_Per_Day
FROM
(SELECT orders.date, SUM(order_details.quantity) AS Quantity
FROM orders
JOIN order_details
ON order_details.order_id = orders.order_id
GROUP BY orders.date) AS Order_Quantity;
```

MySQL Workbench Output :



	Avg_Pizza_Per_Day
▶	138

Determine the top 3 most ordered pizzas based on revenue

Solution :

```
SELECT pizza_types.name, SUM(order_details.quantity * pizzas.price) AS Revenue
FROM  pizza_types
JOIN  pizzas
ON pizzas.pizza_type_id = pizza_types.pizza_type_id
JOIN  order_details
ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY Revenue DESC
LIMIT 3;
```

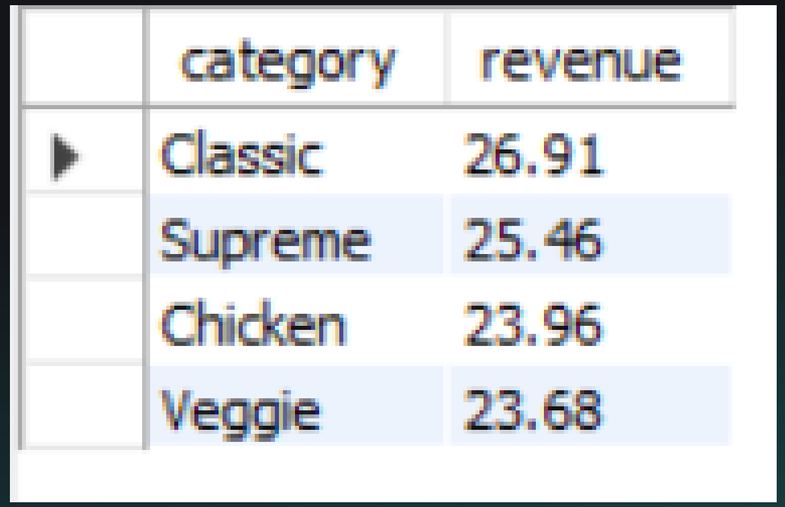
MySQL Workbench Output :

name	Revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

Calclulate 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_sale
FROM  order_details
JOIN    pizzas
ON pizzas.pizza_id = order_details.pizza_id)
* 100,2) AS revenue
FROM pizza_types
JOIN pizzas
ON pizzas.pizza_type_id = pizza_types.pizza_type_id
JOIN order_details
ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue DESC;
```



category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Analyse the cummulative revenue genrated over time

Solution :

```
Select date,sum(revenue) over (order by date) as cum_revenue
From
(select orders.date,sum(order_details.quantity*pizzas.price) as revenue
From order_details
Join pizzas
On pizzas.pizza_id = order_details.pizza_id
Join orders
On orders.order_id=order_details.order_id
Group By orders.date ) as Sales;
```

MySQL Workbench Output :

	date	cum_revenue
▶	2015-01-01	2713.8500000000004
	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	21526.4
	2015-01-10	23990.350000000002

Result 3 ×

Determine the top 3 most orderd pizzas types based on revenue for each pizza category

Solution :

```
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 pizzas.pizza_type_id = pizza_types.pizza_type_id
JOIN  order_details
ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category ,pizza_types.name) as a ) as b
where rn>=3;
```

MySQL Workbench Output :

category	name	revenue
Chicken	The California Chicken Pizza	41409.5
Chicken	The Southwest Chicken Pizza	34705.75
Chicken	The Chicken Alfredo Pizza	16900.25
Chicken	The Chicken Pesto Pizza	16701.75
Classic	The Pepperoni Pizza	30161.75
Classic	The Greek Pizza	28454.100000000013
Classic	The Italian Capocollo Pizza	25094
Classic	The Napolitana Pizza	24087
Classic	The Big Meat Pizza	22968
Classic	The Pepperoni, Mushroom, and Peppers Pizza	18834.5

Result 4

Project Conclusion :

The Pizza Analysis Project successfully demonstrates the power of SQL in analyzing business data to drive informed decision-making. By exploring pizza sales data, customer preferences, and revenue trends .

This project highlights several key outcomes :

- Business Insights
- Data-Driven Decision-Making
- Skill Development

Thank You For Visit

Pune

maliharsha0927@gmail.com