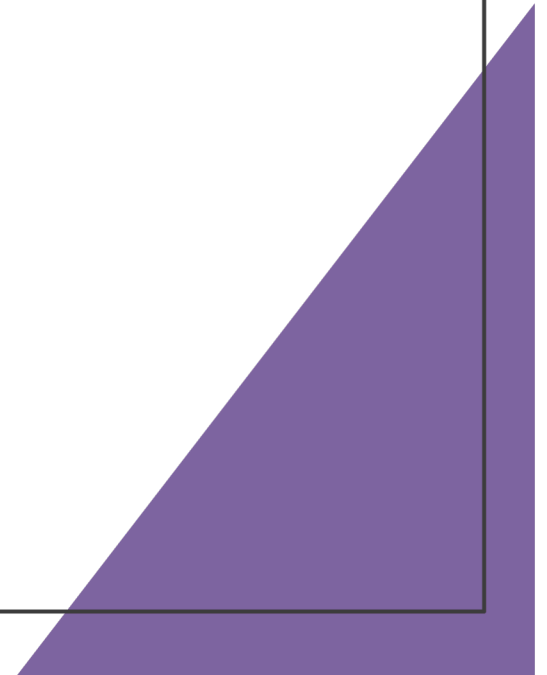
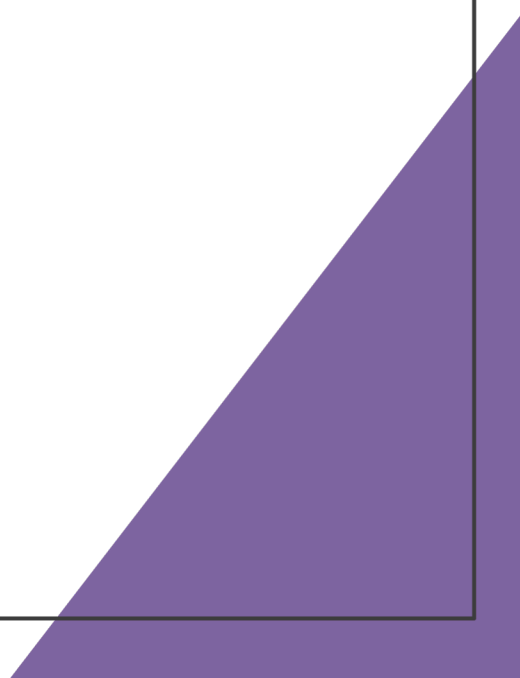


MY-SQL PIZZA SALES ANALYSIS PROJECT

Sales - MySQL Analysis



Project Overview

- Objective:
 - Analyze sales trends, percentage contribution and generate cumulative revenue.
 - Datasets:
 - -Pizza_Types
 - -Order_Details
 - -Orders
 - - Pizzas
- 

Database Design



Schema Definition:



Pizza_Types : PizzaTypeID, Name, Category, Ingredients



Order_Details : OrderDetailsID, OrderID, PizzaID Quantity,



Orders : OrderID, OrderDate, OrderTime



Pizzas : PizzaID, PizzaTypeID, Sales, Price



Relationships:



-Foreign keys link Orders to Order_ID and Order_Details.

SQL QUERIES

1. Retrieve the total number of orders placed :

```
SELECT count(order_id) AS Total_Orders FROM orders;
```

Result:

Result Grid	
	Total_Orders
▶	21350

2. Calculate the total revenue generated from pizza sales :

```
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;
```



Result:

Result Grid	
	Total_Sales
▶	817860.05

3. 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;
```


Result:

Result Grid   Filter Rows:		
	Size	Order_Count
▶	L	18526
	M	15385
	S	14137
	XL	544
	XXL	28

4. 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
    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 Quantity DESC
LIMIT 5;
```



Result:

Result Grid   Filter Rows: <input type="text"/>		
	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

5. 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
    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.category
ORDER BY Quantity DESC;
```



Result:

Result Grid   Filter Rows:		
	Category	Quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

6. 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(order_time);
```




Result:

Result Grid   Filter Rows:		
	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

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

```
SELECT
    ROUND(AVG(quantity), 0) AS AVG_Pizzas_ordered_per_day
FROM
    (SELECT
        orders.order_date, SUM(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;
```



Result:

Result Grid   Filter Rows:	
	AVG_Pizzas_ordered_per_day
	138

8. 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
    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.Name
ORDER BY Revenue DESC
LIMIT 3;
```



Result:

Result Grid   Filter Rows: <input type="text"/>		
	Name	Revenue
▶	The Thai Chicken Pizza	43434.25
	The Barbecue Chicken Pizza	42768
	The California Chicken Pizza	41409.5

9. 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_Sales  
FROM order_details  
       JOIN pizzas ON pizzas.pizza_id = order_details.pizza_id) * 100,2) AS Revenue  
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.Category  
ORDER BY Revenue DESC;
```



Result:

Result Grid   Filter Rows:		
	Category	Revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

10. Analyze the cumulative revenue generated over time :

```
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 Sales;
```

Result:

Result Grid   Filter Rows: <input type="text"/>		
	Order_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
	2015-01-11	25862.65
	2015-01-12	27781.7
	2015-01-13	29831.300000000003
	2015-01-14	32358.700000000004
	2015-01-15	34343.500000000001

THANK YOU