**Business Case: SQL Analysis of Pizza Sales**

**Project Overview:** This project involves an in-depth analysis of pizza sales data using SQL. The primary objective is to derive meaningful insights that can aid in decision-making and strategy formulation.

**Key Analyses Conducted:**

1. **Revenue Breakdown:**
   * Calculated the total revenue generated by each pizza type.
   * Analysed the percentage contribution of each pizza type to the overall revenue.
2. **Cumulative Revenue:**
   * Tracked the cumulative revenue generated over time to understand sales trends and growth.
3. **Top-Performing Pizzas:**
   * Identified the top 3 most ordered pizza types based on revenue within each pizza category.

**SQL Techniques and Functions Used:**

* Window functions for cumulative calculations.
* Join operations to combine multiple tables.
* Grouping and aggregation to summarize data.
* Subqueries for advanced data manipulation.

**Insights:**

* The analysis revealed the most popular and profitable pizza types, providing actionable insights for marketing and inventory management.
* Cumulative revenue tracking highlighted key sales periods and trends.

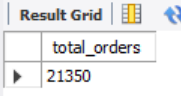
**Basic Level:**

**1.Retrieve the total number of orders placed.**

SELECT

DISTINCT COUNT(order\_id) as total\_orders

FROM mydatabase.orders;



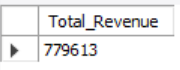
**2.Calculate the total revenue generated from pizza sales.**

SELECT

ROUND(SUM(od.quantity \* p.price),0) as Total\_Revenue

FROM order\_details od

JOIN pizzas p ON od.pizza\_id = p.pizza\_id;



**3.Identify the highest-priced pizza.**

SELECT

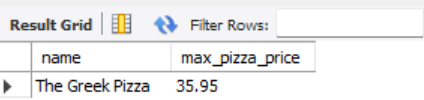
pt.name,

p.price as max\_pizza\_price

FROM pizza\_types pt

JOIN pizzas p ON pt.pizza\_type\_id = p.pizza\_type\_id

WHERE p.price = (SELECT MAX(price) FROM pizzas)



**4.Identify the most common pizza size ordered.**

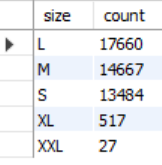
SELECT p.size, count(od.order\_details\_id) as count

FROM order\_details od

JOIN pizzas p ON od.pizza\_id = p.pizza\_id

GROUP BY 1

ORDER BY 2 DESC;



**5.List the top 5 most ordered pizza types along with their quantities.**

WITH RankedOrders AS (

SELECT pt.name,

SUM(od.quantity) AS Quantity,

ROW\_NUMBER() OVER (ORDER BY SUM(od.quantity) DESC) AS OrdersRank

FROM pizza\_types pt

JOIN pizzas p ON pt.pizza\_type\_id = p.pizza\_type\_id

JOIN order\_details od ON p.pizza\_id = od.pizza\_id

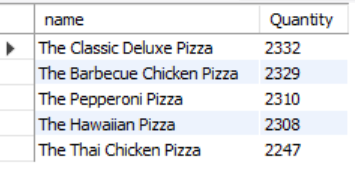
GROUP BY pt.name

)

SELECT name, Quantity FROM RankedOrders

WHERE OrdersRank <= 5

ORDER BY Quantity DESC;



**Intermediate Level :**

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

SELECT pt.category, SUM(od.quantity) AS Quantity

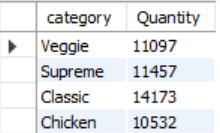
FROM pizza\_types pt

JOIN pizzas p ON pt.pizza\_type\_id = p.pizza\_type\_id

JOIN order\_details od ON od.pizza\_id = p.pizza\_id

GROUP BY pt.category

ORDER BY pt.category DESC;



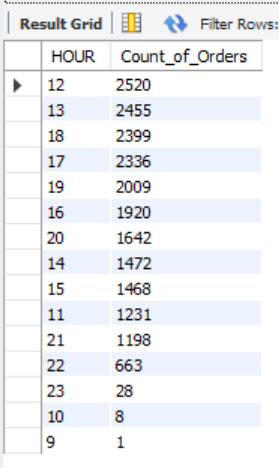
**7.Determine the distribution of orders by hour of the day and arrange by orders.**

SELECT HOUR(O.time) as HOUR, COUNT(order\_id) AS Count\_of\_Orders

FROM orders O

GROUP BY 1

ORDER BY 2 DESC;



**8.Join relevant tables to find the category-wise distribution of pizzas.**

SELECT

PT.category,

PT.name,

COUNT(OD.order\_details\_id) as Count\_of\_Orders,

SUM(OD.quantity) as Quantity

FROM pizza\_types PT

JOIN pizzas P

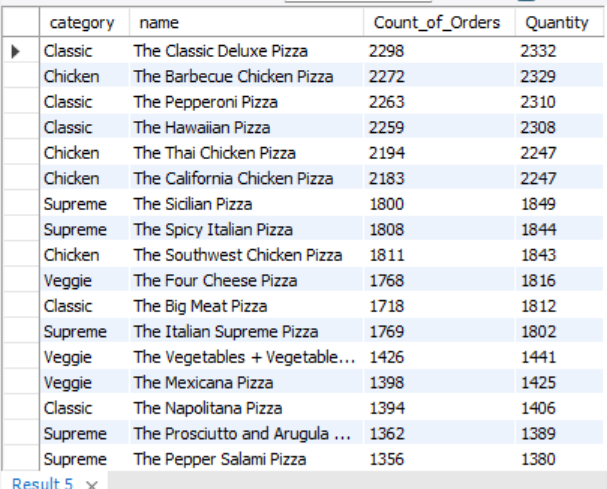
ON PT.pizza\_type\_id = P.pizza\_type\_id

JOIN order\_details OD

ON P.pizza\_id = OD.pizza\_id

GROUP BY 1,2

ORDER BY 4 DESC;



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

SELECT

ROUND(AVG(Count\_of\_Orders),0) as Avg\_Orders\_per\_day,

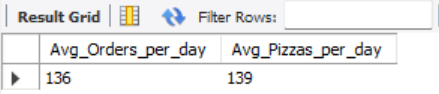
ROUND(AVG(Quantity),0) as Avg\_Pizzas\_per\_day

FROM (SELECT DATE(O.date) as Date, COUNT(O.order\_id) as Count\_of\_Orders, SUM(OD.quantity) as Quantity

FROM orders O

JOIN order\_details OD ON O.order\_id = OD.order\_id

GROUP BY 1) as Date\_wise\_Orders;



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

SELECT PT.name, COUNT(OD.order\_details\_id) as Count\_of\_Orders, ROUND(SUM(OD.quantity \* p.price),0) as Revenue\_by\_pizza\_name

FROM pizza\_types PT JOIN pizzas P ON PT.pizza\_type\_id = P.pizza\_type\_id

JOIN order\_details OD ON P.pizza\_id = OD.pizza\_id

GROUP BY 1 ORDER BY 3 DESC;



**Advanced Level:**

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

WITH RevenuePerCategory AS (

SELECT

PT.category,

ROUND(SUM(OD.quantity \* P.price),0)AS Revenue

FROM pizza\_types PT

JOIN pizzas P ON PT.pizza\_type\_id = P.pizza\_type\_id

JOIN order\_details OD ON P.pizza\_id = OD.pizza\_id

GROUP BY PT.category

),

TotalRevenue AS (

SELECT

SUM(Revenue) AS TotalRevenue

FROM RevenuePerCategory

)

SELECT

RPC.category,

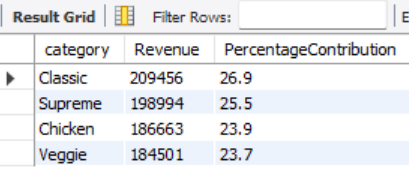
RPC.Revenue,

ROUND((RPC.Revenue / TR.TotalRevenue) \* 100, 1) AS PercentageContribution

FROM RevenuePerCategory RPC

CROSS JOIN TotalRevenue TR

ORDER BY RPC.Revenue DESC;



**12.Analyse the cumulative revenue generated over time.**

SELECT

O.date AS Date,

ROUND(SUM(P.price \* OD.quantity), 0) AS Revenue,

ROUND(SUM(SUM(P.price \* OD.quantity)) OVER (ORDER BY O.date), 0) AS CumulativeRevenue

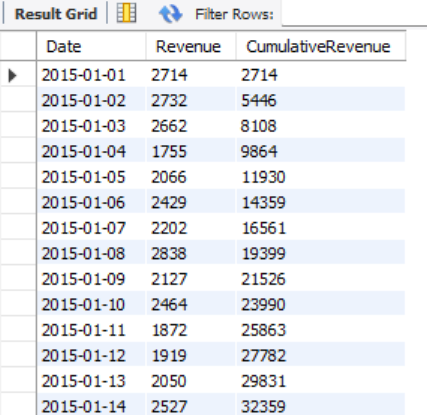
FROM orders O

JOIN order\_details OD ON O.order\_id = OD.order\_id

JOIN pizzas P ON OD.pizza\_id = P.pizza\_id

GROUP BY O.date

ORDER BY O.date;



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

WITH RevenuePerPizzaType AS (

SELECT

PT.category,

PT.name,

ROUND(SUM(OD.quantity \* P.price),0)AS Revenue

FROM pizza\_types PT

JOIN pizzas P ON PT.pizza\_type\_id = P.pizza\_type\_id

JOIN order\_details OD ON P.pizza\_id = OD.pizza\_id

GROUP BY PT.category, PT.name

)

SELECT \* FROM (

SELECT

RPPT.category,

RPPT.name,

RPPT.Revenue,

RANK() OVER(PARTITION BY category ORDER BY Revenue DESC) as Rank\_by\_Category

FROM RevenuePerPizzaType RPPT

ORDER BY RPPT.category, RPPT.Revenue DESC) as ABC

WHERE Rank\_by\_Category <= 3;



**14.Determine the Revenue for each month and cumulative revenue over each month.**

SELECT

EXTRACT(YEAR\_MONTH FROM O.date) AS Month,

ROUND(SUM(P.price \* OD.quantity), 0) AS Revenue,

ROUND(SUM(SUM(P.price \* OD.quantity)) OVER (ORDER BY EXTRACT(YEAR\_MONTH FROM O.date)), 0) AS CumulativeRevenue

FROM orders O

JOIN order\_details OD ON O.order\_id = OD.order\_id

JOIN pizzas P ON OD.pizza\_id = P.pizza\_id

GROUP BY EXTRACT(YEAR\_MONTH FROM O.date)

ORDER BY EXTRACT(YEAR\_MONTH FROM O.date);

