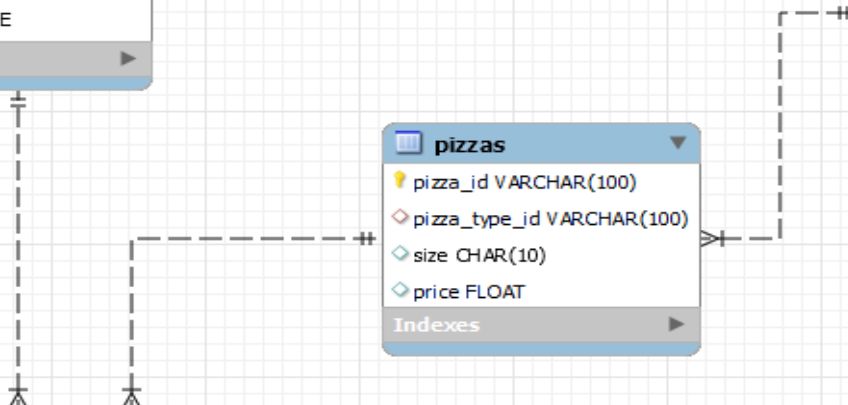
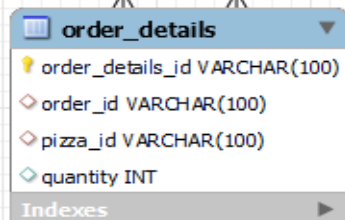
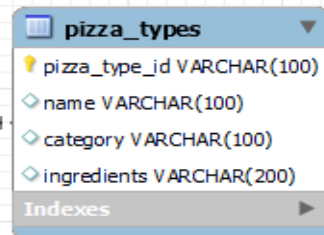
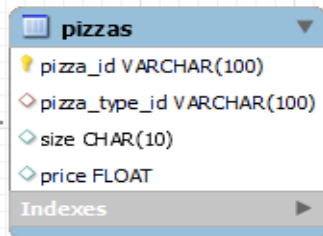
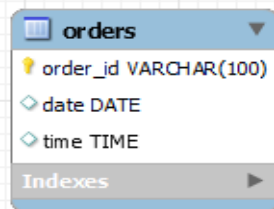




Data Analysis on Pizza Sales

by Yuva Teja



SQL Questions

➤ Basic:

1. Retrieve the total number of orders placed.
2. Calculate the total revenue generated from pizza sales.
3. Identify the highest-priced pizza.
4. Identify the most common pizza size ordered.
5. List the top 5 most ordered pizza types along with their quantities.

➤ Intermediate:

1. Join the necessary tables to find the total quantity of each pizza category ordered.
2. Determine the distribution of orders by hour of the day.





3. Join relevant tables to find the category-wise distribution of pizzas.
4. Determine the top 3 most ordered pizza types based on revenue.

➤ Advanced:

1. Calculate the percentage contribution of each pizza type to total revenue.
2. Analyze the cumulative revenue generated over time.





1. RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED ??

- `SELECT COUNT(order_id) as Totalorders FROM orders ;`

Result Grid 		 Filter Rows: <input type="text"/>		Export: 	Wrap Cell Content: 
	Totalorders				
▶	21350				

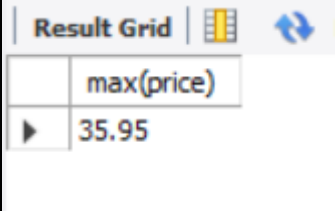
2. CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES ??

- `SELECT SUM(price*quantity)as TotalRevenue FROM order_details as od
JOIN pizzas as p ON od.pizza_id=p.pizza_id;`

Result Grid			Filter Rows: <input type="text"/>	Export: 	Wrap Cell Content: 
	TotalRevenue				
▶	817860.049999993				

3. IDENTIFY THE HIGHEST PRICED PIZZA ?

- `SELECT MAX(price) FROM pizzas;`



The screenshot shows a 'Result Grid' window with a single row of data. The header row contains the text 'max(price)'. The data row contains the value '35.95'. There are icons for a grid and a refresh button in the top right corner of the window.

Result Grid	
	max(price)
▶	35.95

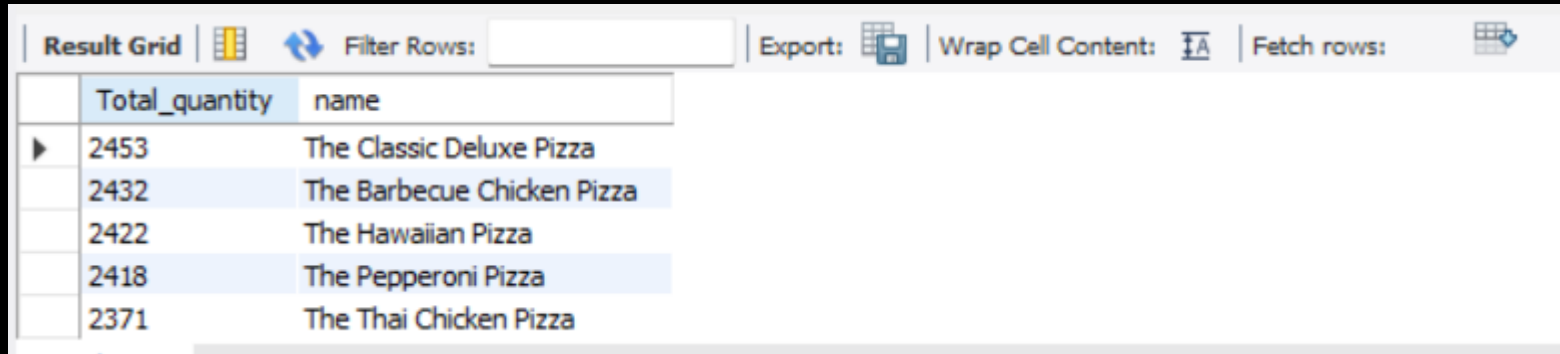
4. IDENTIFY THE COMMON SIZE ORDERED PIZZA ?

- SELECT COUNT(quantity) as Total_quantity,size FROM pizzas as p
JOIN order_details as od ON od.pizza_id=p.pizza_id
GROUP BY size
ORDER BY size;

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	Total_quantity	size			
▶	18526	L			
	15385	M			
	14137	S			
	544	XL			
	28	XXL			

5. LIST THE TOP 5 ORDERED PIZZA TYPES ALONG WITH QUANTITY ?

- `SELECT SUM(quantity)as Total_quantity,name from order_details as od
JOIN pizzas as p ON p.pizza_id=od.pizza_id
JOIN pro.pizza_types as pt ON pt.pizza_type_id=p.pizza_type_id
GROUP BY name
ORDER BY Total_quantity desc
LIMIT 5;`



The screenshot shows a database interface with a 'Result Grid' tab. The grid displays the results of a SQL query. The columns are 'Total_quantity' and 'name'. The data is sorted in descending order of total quantity. The top 5 results are:

Total_quantity	name
2453	The Classic Deluxe Pizza
2432	The Barbecue Chicken Pizza
2422	The Hawaiian Pizza
2418	The Pepperoni Pizza
2371	The Thai Chicken Pizza

6.Total quantity of each pizza ordered ?

- SELECT SUM(quantity) as TotalQuantity,category from order_details as od
JOIN pizzas as p ON p.pizza_id=od.pizza_id
JOIN pizza_types as pt ON pt.pizza_type_id=p.pizza_type_id
GROUP BY category ;

	TotalQuantity	category
▶	14888	Classic
	11649	Veggie
	11987	Supreme
	11050	Chicken

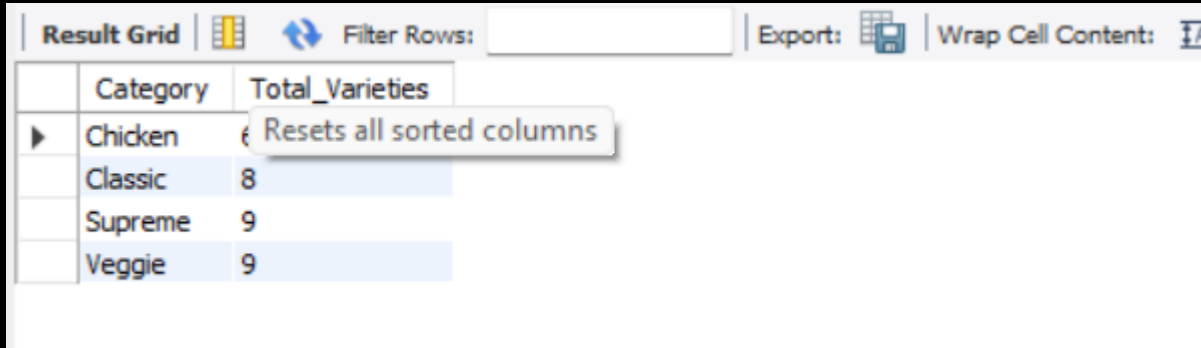
7. Distribution of orders by an hour of the day ?

- `SELECT COUNT(order_id) as
Total_id, hour(time) as Hours FROM
orders GROUP BY Hours;`

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	Total_ids	Hours			
▶	1231	11			
	2520	12			
	2455	13			
	1472	14			
	1468	15			
	1920	16			

8. Find the category wise distribution of pizzas ?

- `SELECT Category, count(name) as Total_Varieties
FROM pizza_types GROUP BY Category;`



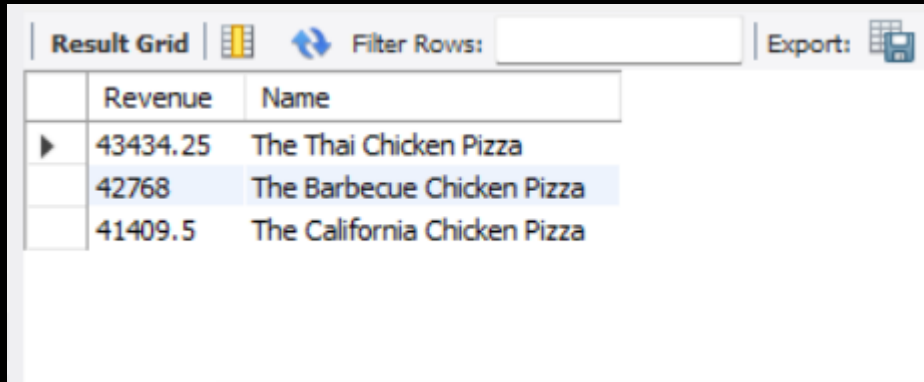
The screenshot shows a database query result grid with the following data:

	Category	Total_Varieties
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9

The interface includes a toolbar with options: Result Grid, Filter Rows, Export, and Wrap Cell Content. A tooltip "Resets all sorted columns" is visible over the Chicken row.

9.Determine top 3 ordered pizzas type based on revenue ??

- `SELECT SUM(price*quantity) as Revenue,name as Name FROM order_details as od JOIN pizzas as p ON p.pizza_id=od.pizza_id JOIN pizza_types as pt ON pt.pizza_type_id=p.pizza_type_id GROUP BY name ORDER BY revenue DESC LIMIT 3;`

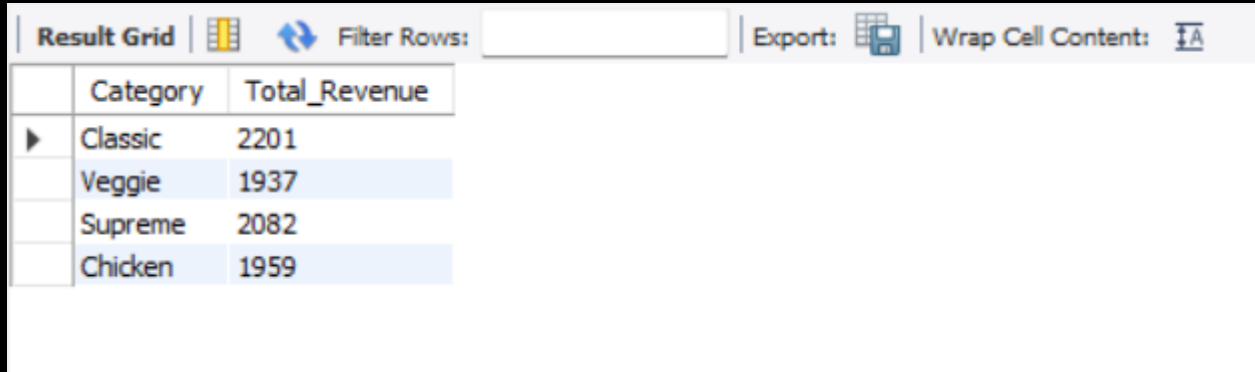


The screenshot shows a database query result grid with the following data:

	Revenue	Name
▶	43434.25	The Thai Chicken Pizza
	42768	The Barbecue Chicken Pizza
	41409.5	The California Chicken Pizza

10. Calculating % contribution of each pizza to total revenue ??

- SELECT category as Category, ROUND(SUM(price*quantity)/100) as Total_Revenue
FROM order_details as od JOIN pizzas as p ON p.pizza_id=od.pizza_id
JOIN pizza_types as pt ON pt.pizza_type_id=p.pizza_type_id
GROUP BY category;



The screenshot shows a database query result grid with a toolbar at the top. The toolbar includes a 'Result Grid' tab, a 'Filter Rows' button, an 'Export' button, and a 'Wrap Cell Content' button. The result grid displays a table with two columns: 'Category' and 'Total_Revenue'. The data is as follows:

	Category	Total_Revenue
▶	Classic	2201
	Veggie	1937
	Supreme	2082
	Chicken	1959

11. Analyse cumulative revenue ??

- SELECT date ,SUM(revenue) over(ORDER BY date) as Cumulative_Revenue
FROM(SELECT SUM(price*quantity) as revenue,date FROM order_details as od JOIN
pizzas as p ON p.pizza_id=od.pizza_id JOIN pizza_types as pt ON
pt.pizza_type_id=p.pizza_type_id JOIN orders as o ON od.order_id=o.order_id
GROUP BY date) as sales ;

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	date	Cumulative_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			