

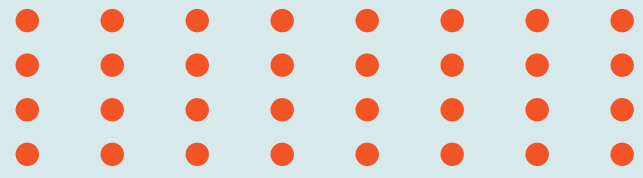


SQL
Pizza
Analysis

PIZZA SALES_ANALYSIS REPORT

-- Using SQL
by Venkatesh





Introduction

**Welcome to my Sales Report
Presentation. This project done by
Venkatesh Mandhapalli, to analyse
pizza sales using SQL to uncover key
insights.**



DATA SET



File	Columns
Order_details.csv	1.order_details_id 2.order_id 3.pizza_id 4.Quantity
Orders.csv	1.order_id 2.order_date 3.order_time
Pizzas	1.Pizza_id 2.pizza_type_id 3.size 4.price
Pizza_types	1.pizza_type_id 2.name 3.category 4.ingredients

Objective



01 Retrieve Total orders placed

02 Total revenue generated

03 Distribution of orders by hour

04 Category wise distribution of orders

05 Highest priced pizza

06 Most common pizza size ordered

07 Top 5 most ordered pizza types along with quantities

08 Total quantity of each pizza category ordered

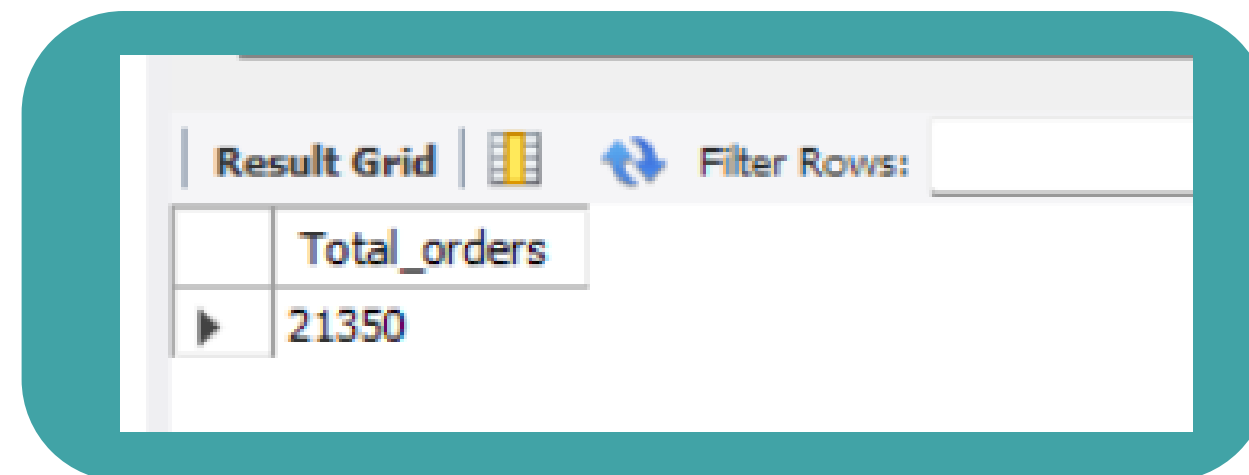
09 Average no. of pizzas ordered per day

10 Top 3 most ordered pizza types based on revenue



1) Retrieve the total number of orders placed.

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



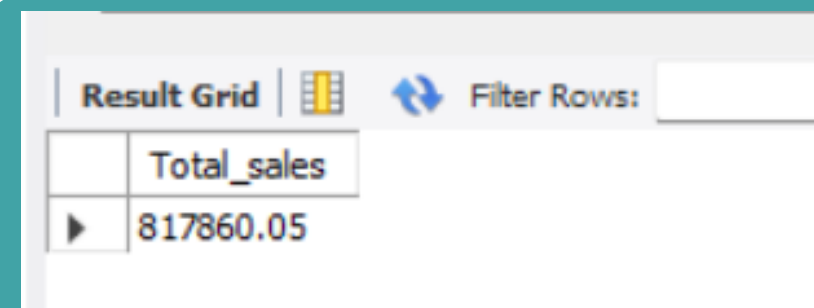
The screenshot shows a database query result grid. The header row is labeled 'Total_orders'. The first data row shows the value '21350'. The grid has a 'Result Grid' tab and a 'Filter Rows' button.

	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
    INNER JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id;
```



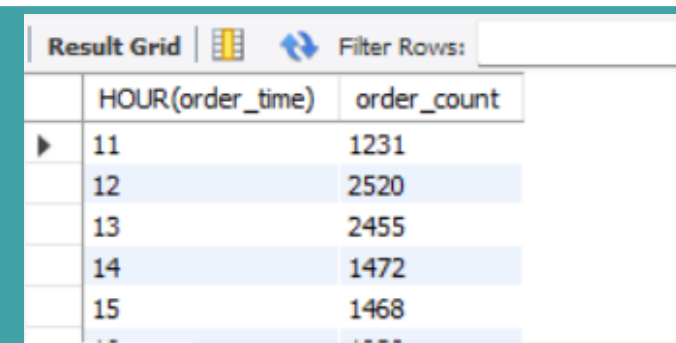
The screenshot shows a 'Result Grid' window with a 'Filter Rows' button. The grid contains one column named 'Total_sales' and one row with the value '817860.05'.

Total_sales
817860.05



3) Determine the distribution of orders by hour of the day.

- ```
SELECT
 HOUR(order_time), COUNT(order_id) AS order_count
FROM
 orders
GROUP BY HOUR(order_time);
```



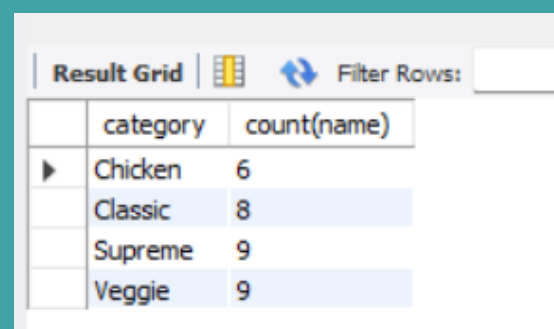
The screenshot shows a database interface with a 'Result Grid' tab. It displays a table with two columns: 'HOUR(order\_time)' and 'order\_count'. The data is as follows:

| HOUR(order_time) | order_count |
|------------------|-------------|
| 11               | 1231        |
| 12               | 2520        |
| 13               | 2455        |
| 14               | 1472        |
| 15               | 1468        |
| ...              | ....        |



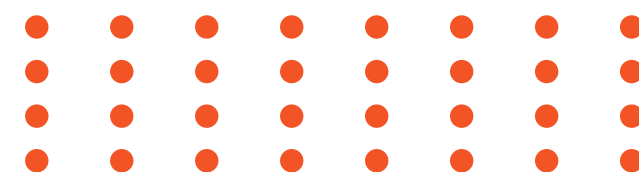
4)Join relevant tables to find the category-wise distribution of pizzas.

- `select category, count(name) from pizza_types  
group by category;`



The screenshot shows a database interface with a 'Result Grid' tab. It displays the output of the SQL query, showing four categories: Chicken, Classic, Supreme, and Veggie, each with a corresponding count of pizzas. The 'Classic' category has the highest count at 8, followed by 'Supreme' and 'Veggie' at 9 each, and 'Chicken' at 6.

|   | category | count(name) |
|---|----------|-------------|
| ▶ | Chicken  | 6           |
|   | Classic  | 8           |
|   | Supreme  | 9           |
|   | Veggie   | 9           |





## 5) Identify the highest-priced pizza.

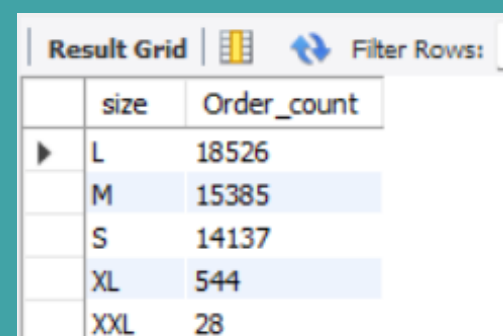
- ```
SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
    INNER JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```

Result Grid			Filter Rows:	
	name	price		
▶	The Greek Pizza	35.95		



6) Identify the most common pizza size ordered.

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

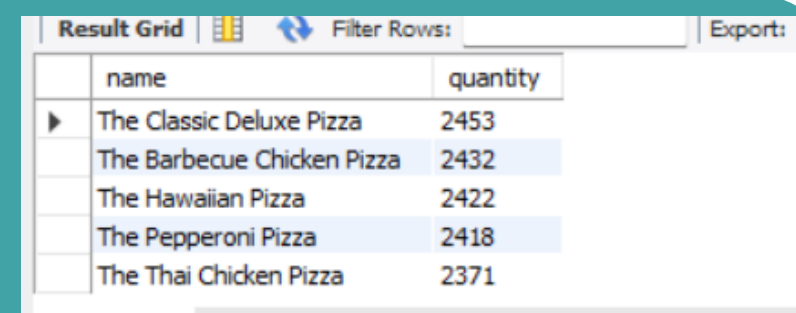


	size	Order_count
▶	L	18526
	M	15385
	S	14137
	XL	544
	XXL	28



7) 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
    INNER JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
    INNER JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC
LIMIT 5;
```



The screenshot shows a database interface with a 'Result Grid' tab. It displays the results of the SQL query, showing the top 5 most ordered pizza types by quantity. The table has two columns: 'name' and 'quantity'. The rows are: 'The Classic Deluxe Pizza' (2453), 'The Barbecue Chicken Pizza' (2432), 'The Hawaiian Pizza' (2422), 'The Pepperoni Pizza' (2418), and 'The Thai Chicken Pizza' (2371). The first four rows are highlighted with a light blue background.

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



8)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
    INNER JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
    INNER JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY quantity DESC;
```

	category	quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050



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

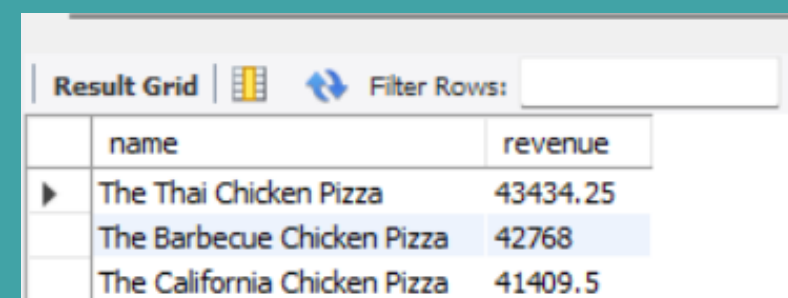
```
• select round(avg(quantity),0) from  
  (select orders.order_date,sum(order_details.quantity)as quantity from orders  
   inner join order_details  
   on orders.Order_id=order_details.order_id  
   group by orders.order_date) as order_quantity;
```

Result Grid		Filter Rows:
	round(avg(quantity),0)	
▶	138	



10) 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
INNER JOIN pizzas
ON pizza_types.pizza_type_id = pizzas.pizza_type_id
INNER JOIN
order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3;
```



The screenshot shows a database interface with a 'Result Grid' tab. It displays a table with two columns: 'name' and 'revenue'. The table contains three rows of data, sorted by revenue in descending order. The first row is 'The Thai Chicken Pizza' with a revenue of 43434.25. The second row is 'The Barbecue Chicken Pizza' with a revenue of 42768. The third row is 'The California Chicken Pizza' with a revenue of 41409.5. A 'Filter Rows' input field is visible at the top right of the grid.

|   | name                         | revenue  |
|---|------------------------------|----------|
| ▶ | The Thai Chicken Pizza       | 43434.25 |
|   | The Barbecue Chicken Pizza   | 42768    |
|   | The California Chicken Pizza | 41409.5  |





Thank You

