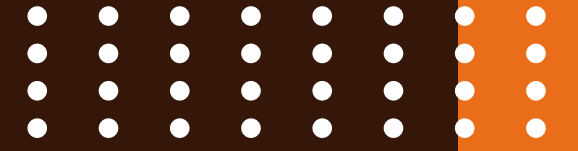
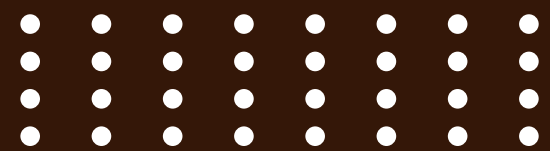


# SQL PROJECT ON PIZZA POINT

**ORDER  
NOW**







# Hello!

My name is Geetika Sahu, and I have worked on a project titled "Pizza Point", where I analyzed pizza sales data, using SQL techniques such as GROUP BY, ORDER BY, LIMIT, JOINS, ROUND, SUM, and more to uncover meaningful insights.



# QUESTIONS



- Retrieve the total number of orders placed.
- Calculate the total revenue generated from pizza sales.
- Identify the highest-priced pizza.
- Identify the most common pizza size ordered.
- List the top 5 most ordered pizza types along with their quantities.
- Join the necessary tables to find the total quantity of each pizza category ordered.
- Determine the distribution of orders by hour of the day.
- Join relevant tables to find the category-wise distribution of pizzas.
- Group the orders by date and calculate the average number of pizzas ordered per day.
- Determine the top 3 most ordered pizza types based on revenue.





# 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 grid has a header row with the column name 'total\_orders' and a data row with the value '21350'. The grid is titled 'Result Grid' and has a 'Filter Rows' button. The background of the slide features a faint, repeating pattern of coffee beans.

	total_orders
▶	21350

# 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 Grid		Filter F
	total_sales	
▶	817860.05	



# IDENTIFY THE HIGHEST-PRICED PIZZA.

```
select pizza_types.name, pizzas.price
from pizza_types 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	

# 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 ;
```

A screenshot of a database query result grid. The grid has two columns: 'size' and 'order\_count'. The rows are ordered by 'order\_count' in descending order. The first row is 'L' with a count of 18526. The second row is 'M' with a count of 15385. The third row is 'S' with a count of 14137. The fourth row is 'XL' with a count of 544. The fifth row is 'XXL' with a count of 28. The grid is titled 'Result Grid' and has a 'Filter Rows' button.


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




# 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 Grid





Filter Rows:

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.

```
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;
```

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050



# 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 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	



# JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.



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

Result Grid |   Filter Rows:

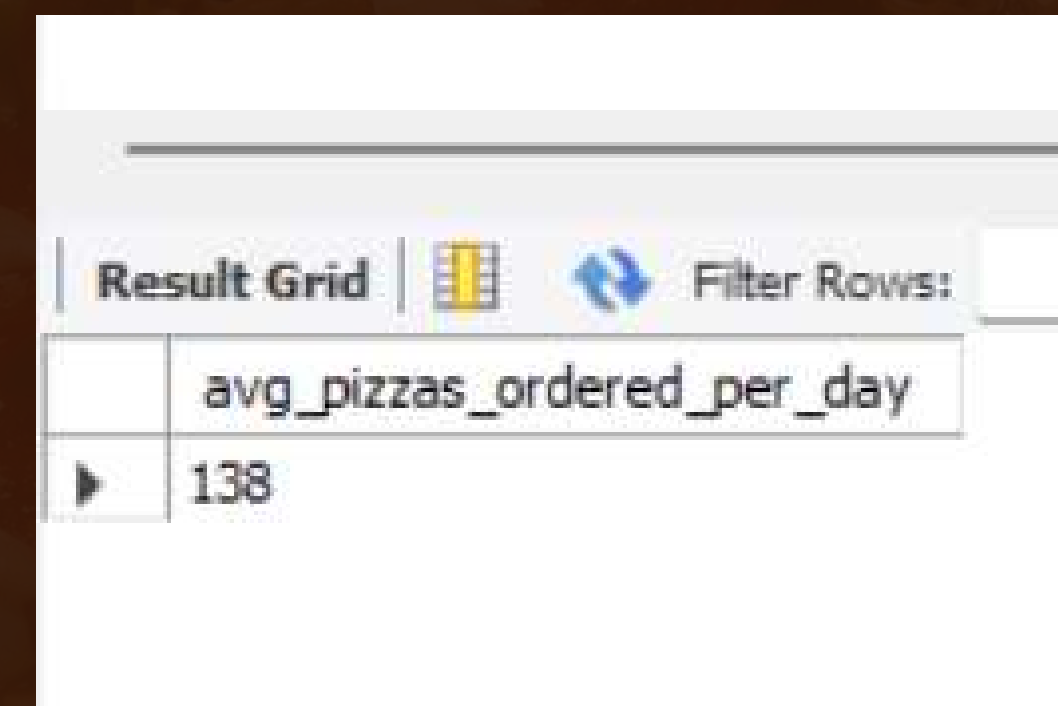
	category	count(name)
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9





# 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 ;
```



The screenshot shows a database interface with a 'Result Grid' tab. The grid contains a single column named 'avg\_pizzas\_ordered\_per\_day' and a single row with the value '138'. Above the grid, there are icons for a grid, a refresh button, and a 'Filter Rows:' input field.

	avg_pizzas_ordered_per_day
▶	138



# 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 Grid			Filter Rows:
	name	revenue	
▶	The Thai Chicken Pizza	43434.25	
	The Barbecue Chicken Pizza	42768	
	The California Chicken Pizza	41409.5	



**THANK YOU  
FOR ATTENTION**

