



# Title: **SQL Project Analysis**

Subtitle: Comprehensive Analysis of Pizza Sales Data

Presenter Name: [Ashish Pal]



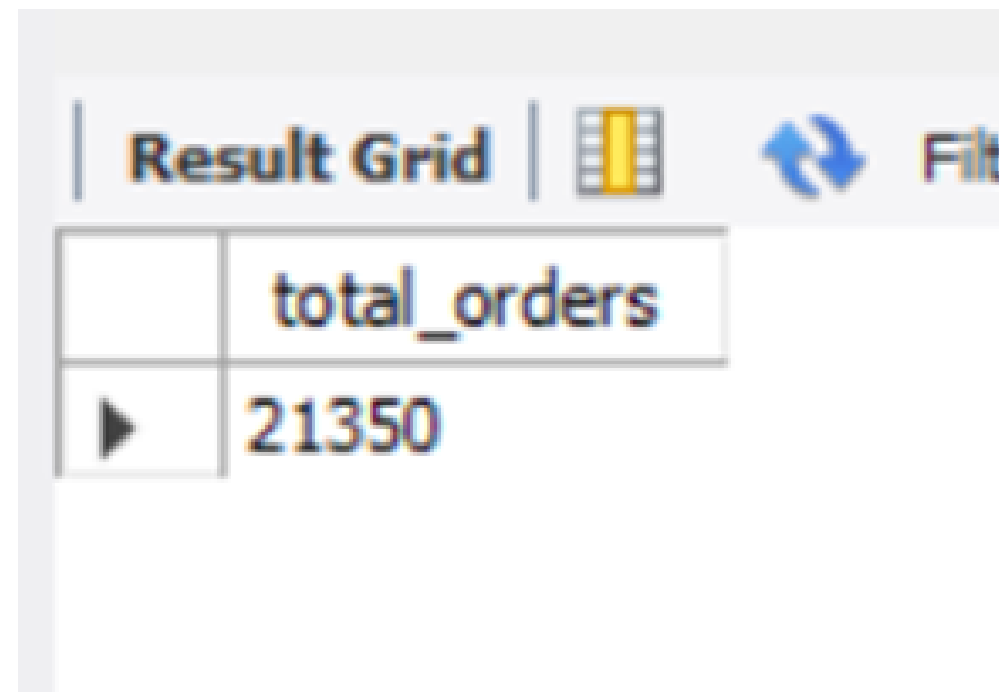
# INTRODUCTION

- Overview of the pizza sales data
  - Purpose of the analysis
  - Key metrics and insights



-- Retrieve the total number of orders placed

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

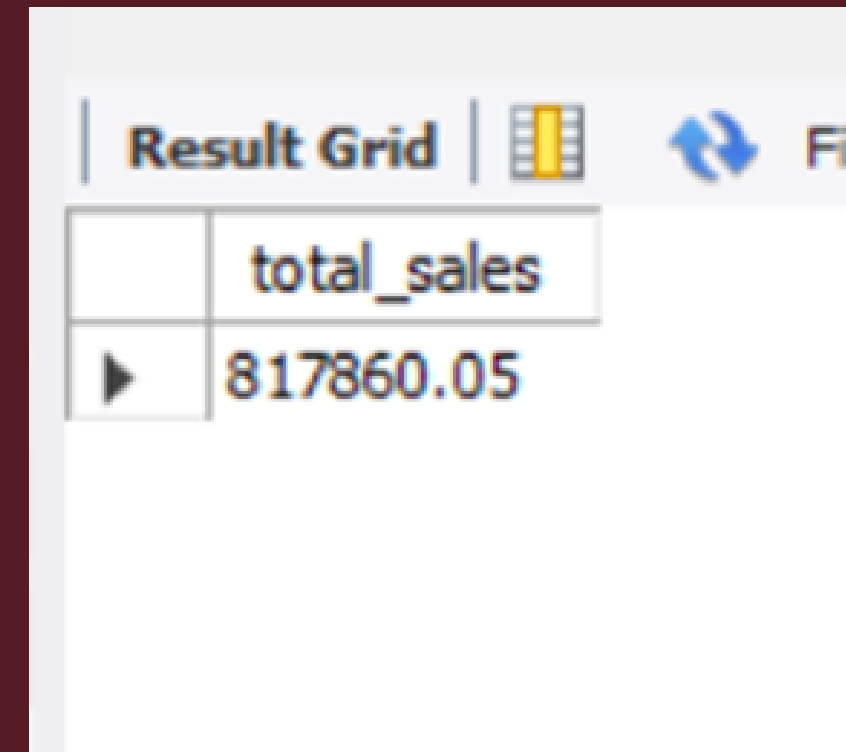


The screenshot shows a database interface with a 'Result Grid' header. Below the header, there is a table with one column named 'total\_orders' and one row containing the value '21350'. The table is displayed in a simple grid format with a light gray background.

	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
```



The screenshot shows a 'Result Grid' window with a single row of data. The column header is 'total\_sales' and the value in the row is '817860.05'. There is a small play button icon in the first column of the data row.

	total_sales
▶	817860.05

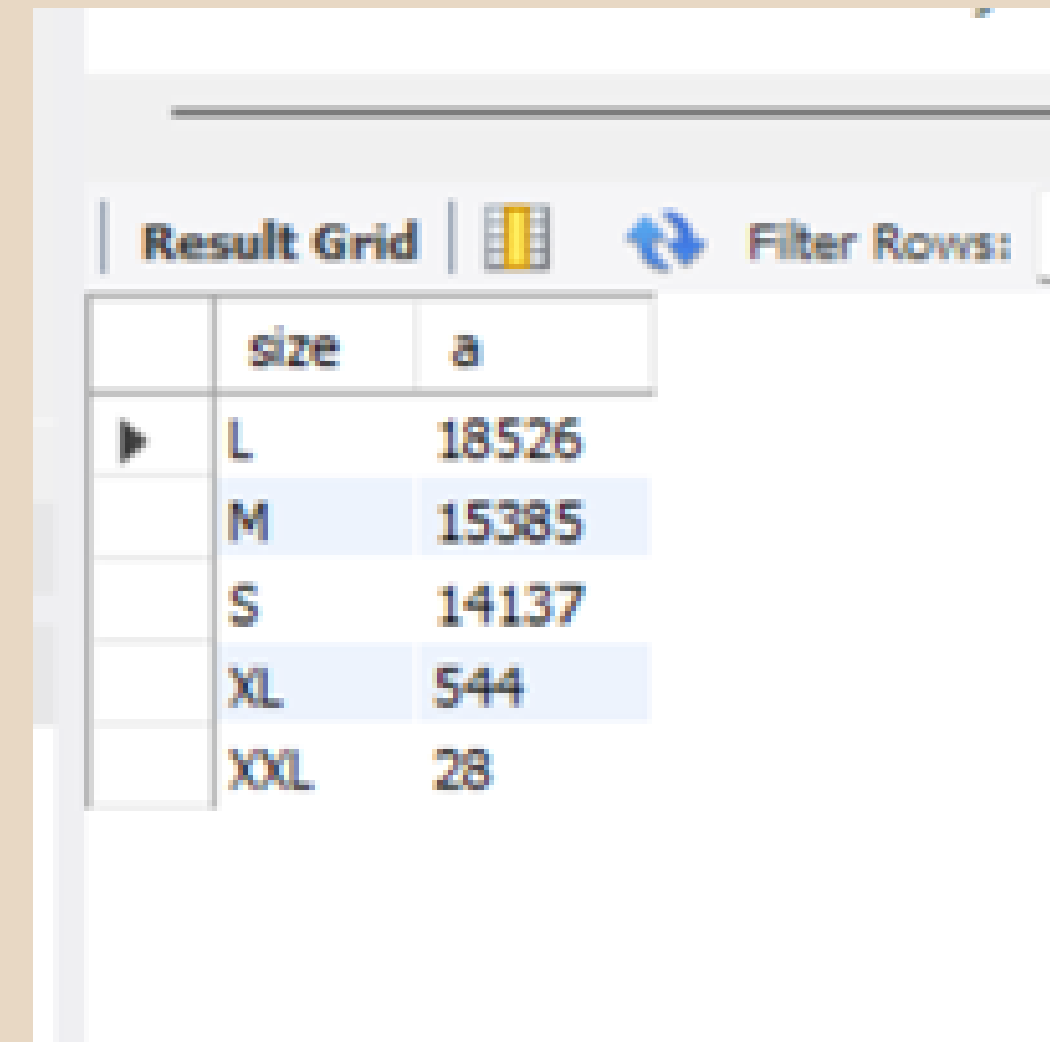
-- Identify the highest-priced pizza.

```
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 a
FROM
  pizzas
JOIN
order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY a DESC;
```



The screenshot shows a database interface with a 'Result Grid' tab. The grid displays the results of a SQL query, showing pizza sizes and their corresponding counts. The columns are 'size' and 'a'. The rows are ordered by count in descending order.

	size	a
▶	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 s
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 s DESC
LIMIT 5;
```

Result Grid			Filter Rows:	
	name	s		
▶	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 s
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 s DESC;
```

Result Grid			Filter Rows
	category	s	
▶	Classic	14888	
	Supreme	11987	
	Veggie	11649	
	Chicken	11050	



-- Determine the distribution of orders by hour of the day.

```
SELECT
  HOUR(time) AS hour, COUNT(order_id)
FROM
  orders
GROUP BY HOUR(time);
```

	hour	COUNT(order_id)
▶	11	1231
	12	2520
	13	2455
	14	1472
	15	1468
	16	1920
	17	2336
	18	2399
	19	2009
	20	1647

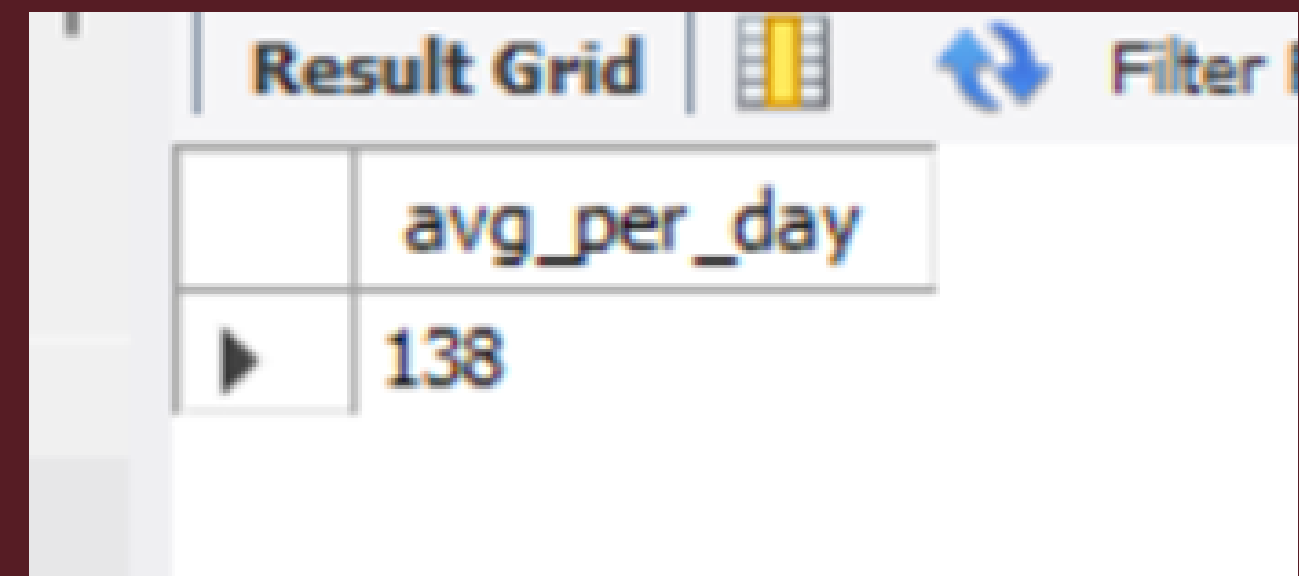
-- Join relevant tables to find the category-wise distribution of pizzas.

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

	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_per_day  
FROM  
    (SELECT  
        orders.date, SUM(order_details.quantity) AS quantity  
    FROM  
        orders  
    JOIN order_details ON orders.order_id = order_details.order_id  
    GROUP BY orders.date) AS order_quantity;
```

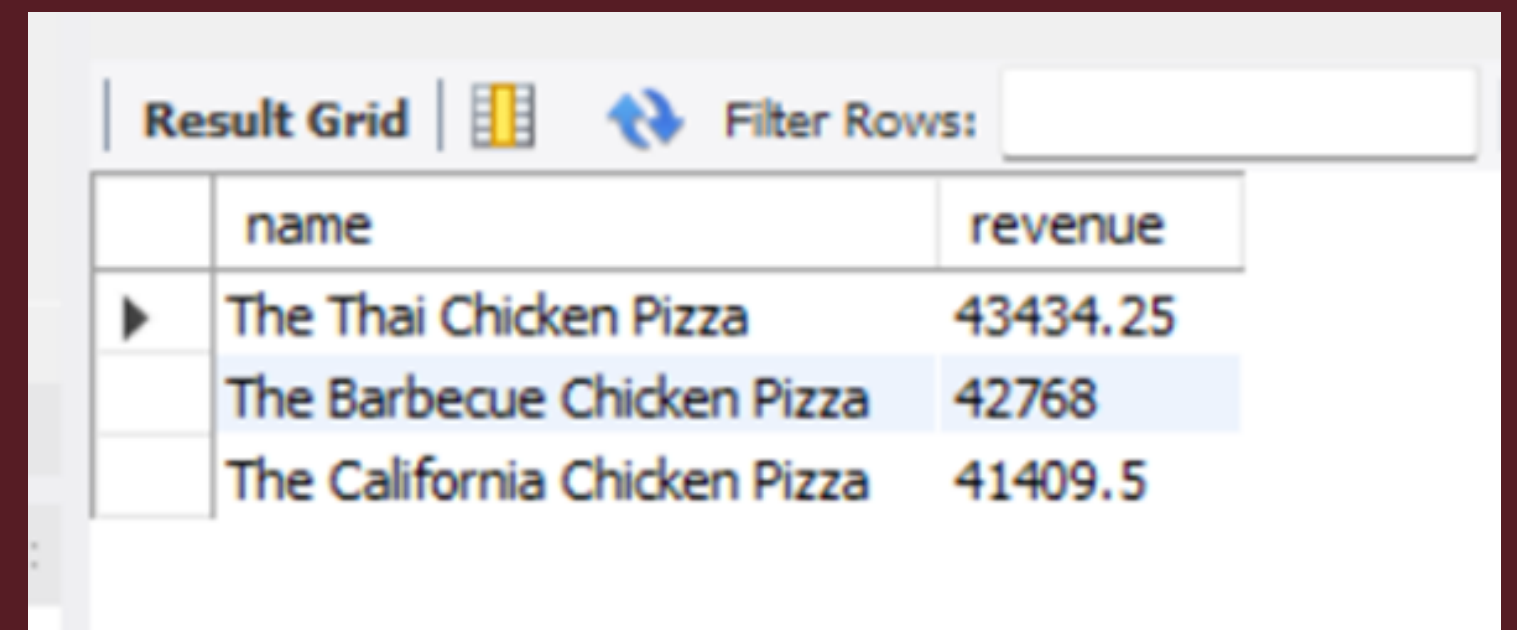


The screenshot shows a database interface with a 'Result Grid' tab. The grid contains one column named 'avg\_per\_day' and one row with the value '138'. There are icons for refreshing and filtering the results.

	avg_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;
```



The screenshot shows a database interface with a 'Result Grid' tab. The grid displays the results of a query, showing the top 3 most ordered pizza types based on revenue. The columns are 'name' and 'revenue'. The rows are:

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

-- 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 Grid			Filter R
	category	revenue	
▶	Classic	26.91	
	Supreme	25.46	
	Chicken	23.96	
	Veggie	23.68	

-- Analyze the cumulative revenue generated over time.

```
select date,sum(revenue) over (order by date) as
      cum_revenue
      from
      (select orders.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.date) as sales;
```

Result Grid			Filter Rows:
	date	cum_revenue	
▶	2015-01-01	2713.85000000000004	
	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	23000.3500000000000000	

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

```
select name,revenue from
(select category,name,revenue,
rank()over (partition by category order by revenue desc) as rn
from
(select pizza_types.category,pizza_types.name,
sum((order_details.quantity)*pizzas.price) 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,pizza_types.name) as a) as b
where rn <=3;
```

	name	revenue
▶	The Thai Chicken Pizza	43434.25
	The Barbecue Chicken Pizza	42768
	The California Chicken Pizza	41409.5
	The Classic Deluxe Pizza	38180.5
	The Hawaiian Pizza	32273.25
	The Pepperoni Pizza	30161.75
	The Spicy Italian Pizza	34831.25
	The Italian Supreme Pizza	33476.75
	The Sicilian Pizza	30940.5
	The Four Cheese Pizza	32265.700000000065

The image features a dark maroon background with several overlapping, semi-transparent hexagonal shapes of varying sizes and positions. These shapes create a layered, geometric effect. Scattered across the background are several small, solid maroon hexagons. The text "THANK YOU" is centered in a bold, white, sans-serif font.

**THANK YOU**