

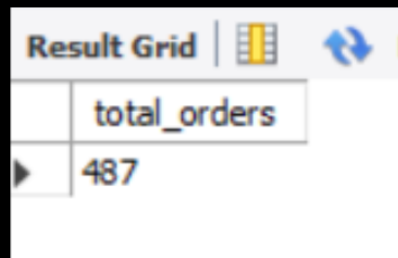
**PIZZA**  
**SALES**  
**SQL**  
**PROJECT**



- **This SQL project is used to extract valuable insights and conduct detailed data analysis ,facilitating informed decision -making and efficient management of pizza store's operations. In this project, we have to answer some questions to gain insights.**

# Q1 -Retrieve the total number of orders placed.

```
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

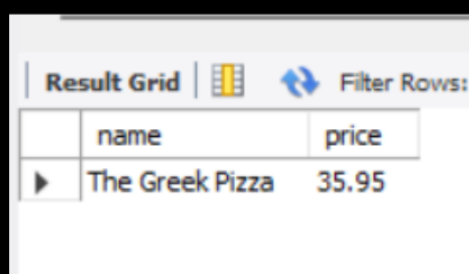


A screenshot of a database interface showing a 'Result Grid'. The grid has two columns: 'total\_orders' and a value '487'. The interface includes a 'Result Grid' header, a table icon, and a refresh icon.

|   | total_orders |
|---|--------------|
| ▶ | 487          |

# Q2- 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;
```

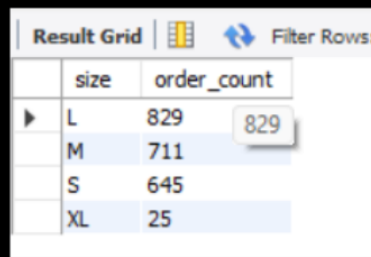


A screenshot of a database interface showing a 'Result Grid'. The grid has three columns: 'name', 'price', and a value 'The Greek Pizza 35.95'. The interface includes a 'Result Grid' header, a table icon, a refresh icon, and a 'Filter Rows:' label.

|   | name            | price |
|---|-----------------|-------|
| ▶ | The Greek Pizza | 35.95 |

## Q3- 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;
```

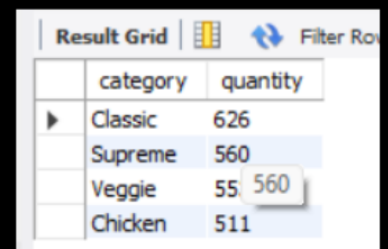


The screenshot shows a 'Result Grid' with two columns: 'size' and 'order\_count'. The data is sorted in descending order of 'order\_count'. The 'order\_count' for size 'L' is 829, which is highlighted with a tooltip.

| size | order_count |
|------|-------------|
| L    | 829         |
| M    | 711         |
| S    | 645         |
| XL   | 25          |

## Q4- 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;
```



The screenshot shows a 'Result Grid' with two columns: 'category' and 'quantity'. The data is sorted in descending order of 'quantity'. The 'quantity' for category 'Veggie' is 560, which is highlighted with a tooltip.

| category | quantity |
|----------|----------|
| Classic  | 626      |
| Supreme  | 560      |
| Veggie   | 55       |
| Chicken  | 511      |

## Q5- Determine the distribution of orders by hour of the day.

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

| Result Grid | Filter Rows:    |
|-------------|-----------------|
| hour        | count(order_id) |
| 19          | 42              |
| 20          | 38              |
| 21          | 23              |
| 13          | 54              |
| 12          | 56              |
| 17          | 62              |
| 16          | 35              |
| 18          | 51              |
| 22          | 18              |
| 14          | 46              |
| 15          | 32              |
| 11          | 30              |

## Q6- 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            |

# Q7- Group the orders by date and calculate the average number of pizzas ordered per day.

```
SELECT
    ROUND(AVG(quantity), 0) AS avg_pizza_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;
```

| Result Grid |                           | Filter Rows: |
|-------------|---------------------------|--------------|
|             | avg_pizza_ordered_per_day |              |
| ▶           | 20                        |              |

# Q8- 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       | 1918.75      |
|             | The Barbecue Chicken Pizza   | 1913         |
|             | The California Chicken Pizza | 1884.5       |
|             | The California Chicken Pizza |              |