LARANA PIZZA

LARANA PIZZA

LARANA PIZZA

ANALYZE DATA WITH SQL

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:

6. Join the necessary tables to find the total quantity of each pizza category ordered.
7. Determine the distribution of orders by hour of the day.
   Join relevant tables to find the category-wise distribution of pizzas.
9. Group the orders by date and calculate the average number of pizzas ordered per day.
10. Determine the top 3 most ordered pizza types based on revenue.

Advanced:

11. Calculate the percentage contribution of each pizza type to total revenue.
12. Analyze the cumulative revenue generated over time.
13. Determine the top 3 most ordered pizza types based on revenue for each pizza category.

```sql
#Retrieve the total number of orders placed
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

Result Grid

	total_orders
▶	21350

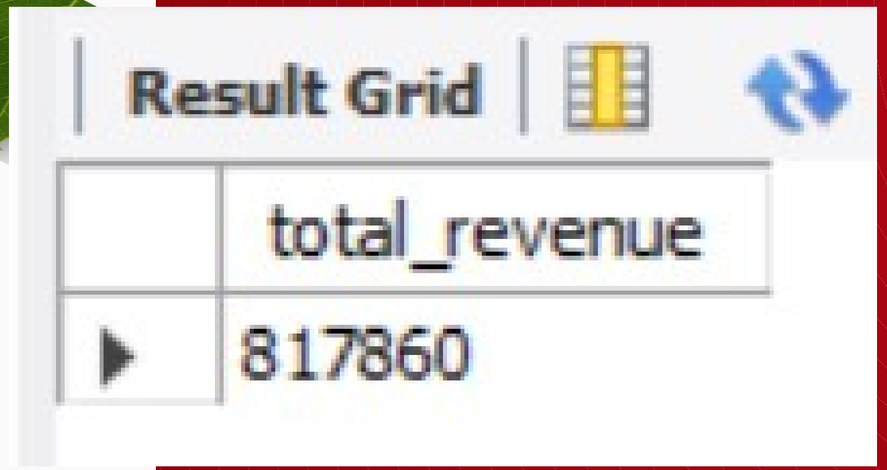
PURPOSE

The purpose is to count the total number of orders placed.

INSIGHT

The code retrieves the total number of orders placed by counting the ‘order_id’ from the `orders` table. The result shows that there are 21,350 total orders. This insight helps understand the volume of orders in the database.

```sql
#Calculate the total revenue generated from pizza sales
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price)) AS total_revenue
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id;
```



PURPOSE

Calculate the total revenue generated from pizza sales.

INSIGHT

The code calculates the total revenue generated from pizza sales by summing the product of the quantity of each pizza ordered and its price.
The result is rounded to the nearest whole number and displayed as `total_revenue`. This provides a clear insight into the total sales revenue from pizzas.

PURPOSE

Identify the highest-priced pizza.

INSIGHT

This SQL query efficiently identifies the highest-priced pizza from the database. It achieves this by joining relevant tables, ordering results based on price in descending order, and then limiting the output to the top-ranked row, which represents the most expensive pizza.

```sql
#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

PURPOSE

Identify the most common pizza size ordered.

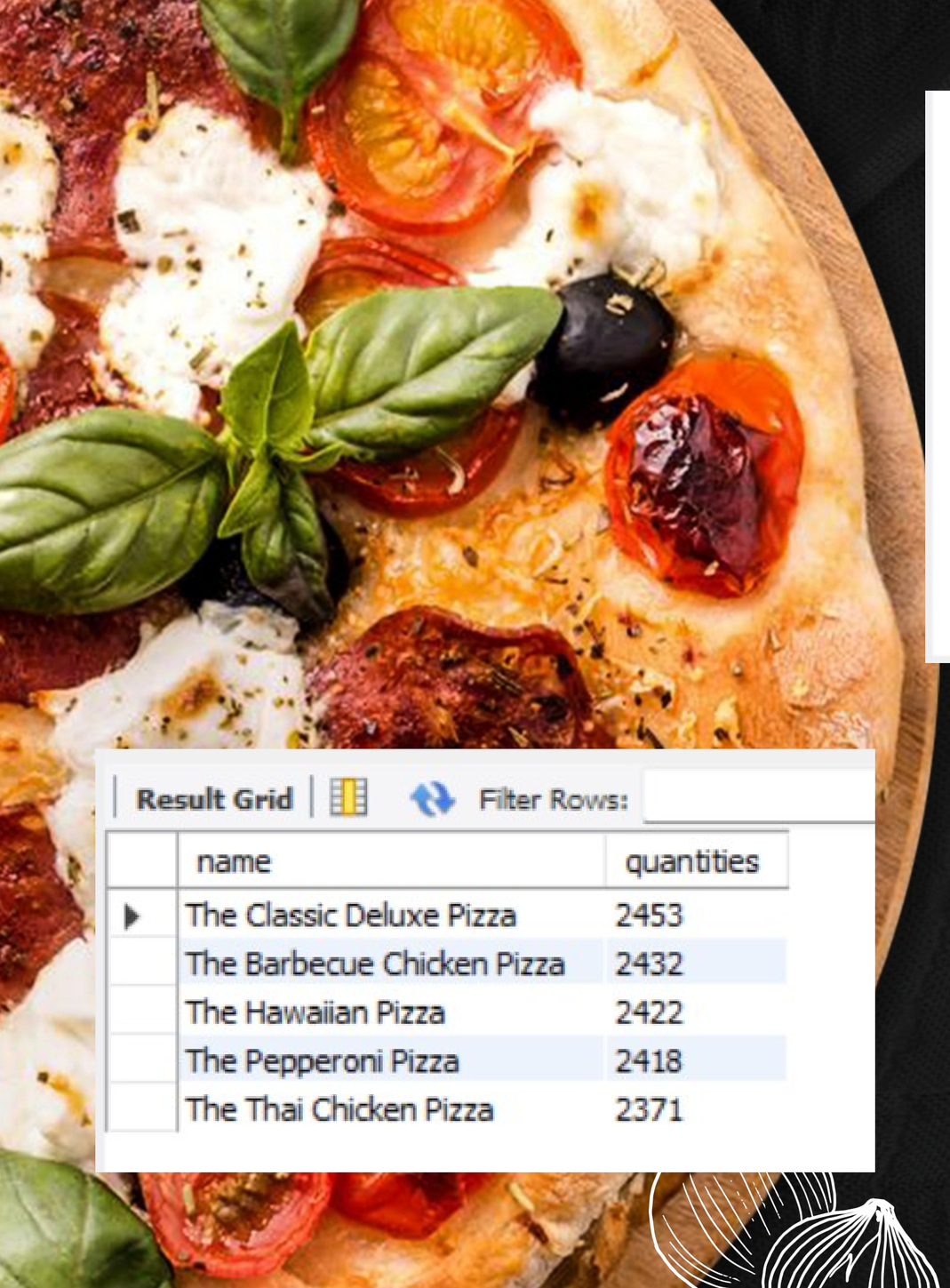
INSIGHT

This SQL query identifies the most frequently ordered pizza size. It does this by joining the 'pizzas' and 'order_details' tables, grouping the results by pizza size, counting the number of orders for each size, ordering the results by the count in descending order, and finally limiting the output to the top row, which represents the size with the highest order count.

```sql
#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
LIMIT 1;
```

Result Grid | Filter R

	size	order_count
▶	L	18526



```sql
#List the top 5 most ordered pizza types along with their quantities
SELECT
    pizza_types.name, SUM(order_details.quantity) AS quantities
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 quantities DESC
LIMIT 5;
```

Result Grid | Filter Rows:

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

PURPOSE

List the top 5 most ordered pizza types along with their quantities.

INSIGHT

This SQL query lists the top 5 most ordered pizza types along with their order quantities. It achieves this by joining three tables ('pizza_types', 'pizzas', and 'order_details'), grouping the results by pizza type, summing up the order quantities for each type, ordering the results in descending order based on the sum, and finally limiting the output to the top 5 rows.

PURPOSE

Join the necessary tables to find the total quantity of each pizza category ordered

INSIGHT

This SQL query calculates the total quantity ordered for each pizza category. It achieves this by joining three tables ('pizza_types', 'pizzas', and 'order_details'), grouping the results by pizza category, summing up the quantities for each category, and finally ordering the results in descending order based on the total quantity.

```sql
#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;
```

Result Grid | Filte

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

LARANA PIZZA

PURPOSE

Determine the distribution of orders by hour of the day.

INSIGHT

This SQL query determines the distribution of orders throughout the day by counting the number of orders placed in each hour. It achieves this by extracting the hour from the order time and counting the corresponding order IDs, grouping the results by hour. The output reveals the peak and off-peak hours for order activity.

```
64      #Determine the distribution of orders by hour of the day
65  •   SELECT
66          HOUR(order_time) AS hour, COUNT(order_id) AS order_count
67      FROM
68          orders
69      GROUP BY hour;
```

Result Grid | Filter

	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

```sql
#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

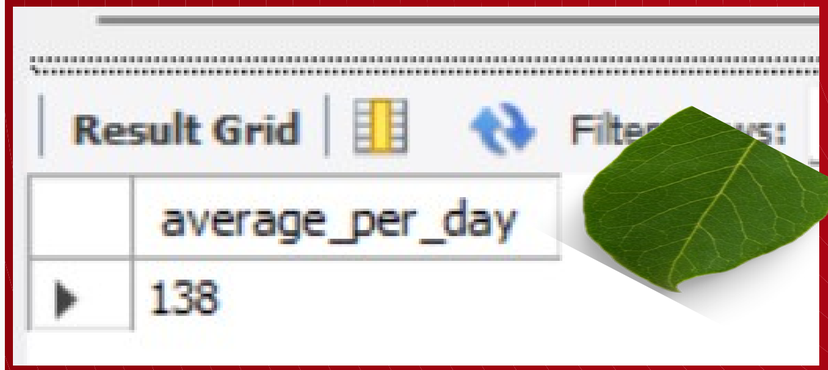
PURPOSE

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

INSIGHT

This SQL query provides a category-wise distribution of pizzas. It does so by selecting the category and counting the number of pizza names within each category from the `pizza_types` table. The results are then grouped by category to display the count of pizzas for each category.

```sql
#Group the orders by date and calculate the average number of pizzas ordered per day.
SELECT
    ROUND(AVG(quantity)) AS average_per_day
FROM
    (SELECT
        orders.order_date AS date,
            SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY date) AS order_quantity;
```



PURPOSE

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

INSIGHT

This SQL query calculates the average number of pizzas ordered per day. It achieves this by first grouping the orders by date and summing the quantities for each date. Then, it calculates the average of these summed quantities across all days. The final result provides a single value representing the average number of pizzas ordered per day.

PURPOSE

Determine the top 3 most ordered pizza types based on revenue.

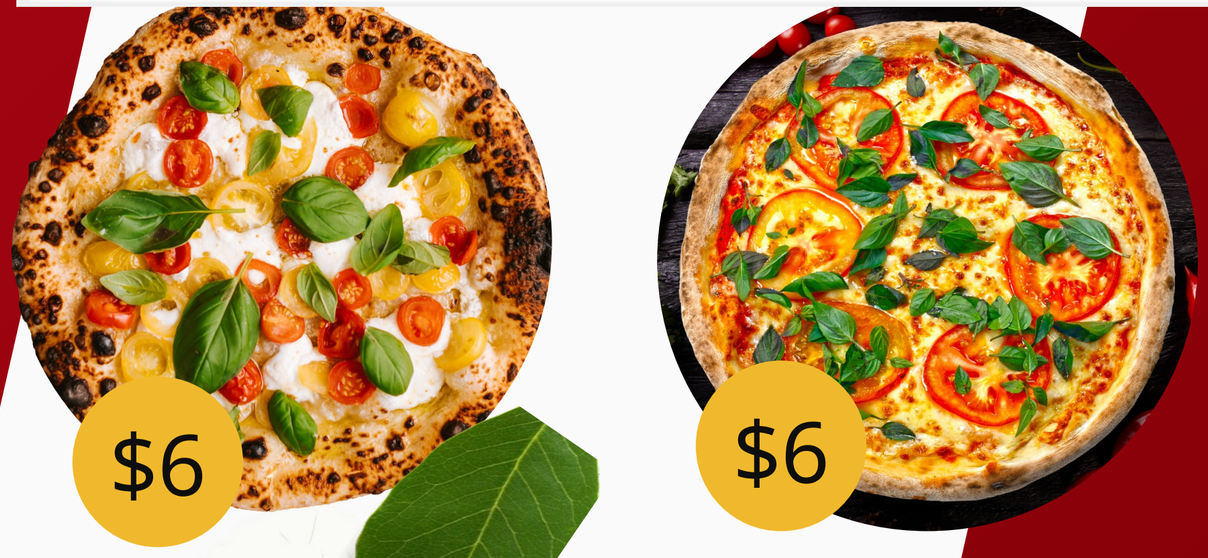
INSIGHT

The code determines the top 3 most ordered pizza types based on the total revenue generated. It calculates the revenue for each pizza type by multiplying the price of each pizza with the quantity ordered and then summing up the values for each type. Finally, it orders the pizza types by revenue in descending order and displays the top 3.

```sql
#Determine the top 3 most ordered pizza types based on revenue.
SELECT
    pizza_types.name,
    SUM(pizzas.price * order_details.quantity) 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.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



PURPOSE

Calculate the percentage contribution of each pizza type to total revenue

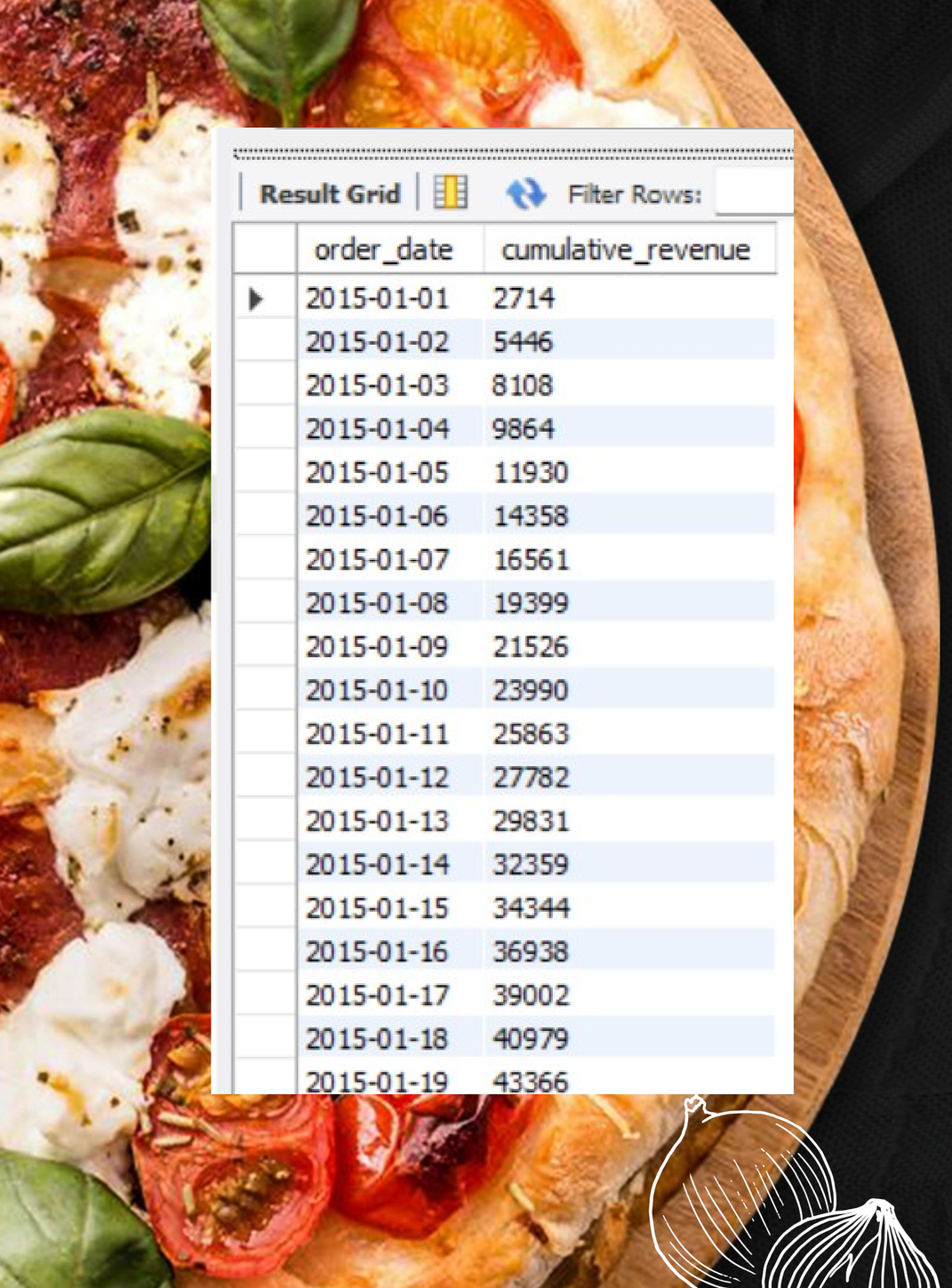
INSIGHT

The code calculates the percentage contribution of each pizza category to the total revenue. It does this by summing the revenue for each category, dividing it by the total revenue, and then multiplying by 100 to get a percentage. Finally, it orders the categories by their percentage contribution in descending order.

```sql
#Calculate the percentage contribution of each pizza type to total revenue
SELECT
    pizza_types.category,
    ROUND(SUM(pizzas.price * order_details.quantity) / (SELECT
                    SUM(pizzas.price * order_details.quantity)
                FROM
                    pizzas
                        JOIN
                    order_details ON pizzas.pizza_id = order_details.pizza_id) * 100,
            2) AS revenue_percentage
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_percentage DESC;
```

Result Grid | Filter Rows:

category	revenue_percentage
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68



Result Grid | Filter Rows:

order_date	cumulative_revenue
2015-01-01	2714
2015-01-02	5446
2015-01-03	8108
2015-01-04	9864
2015-01-05	11930
2015-01-06	14358
2015-01-07	16561
2015-01-08	19399
2015-01-09	21526
2015-01-10	23990
2015-01-11	25863
2015-01-12	27782
2015-01-13	29831
2015-01-14	32359
2015-01-15	34344
2015-01-16	36938
2015-01-17	39002
2015-01-18	40979
2015-01-19	43366

```sql
#Analyze the cumulative revenue generated over time.
SELECT order_date,
ROUND(sum(revenue) OVER(ORDER BY order_date)) AS cumulative_revenue
FROM
(SELECT orders.order_date,
sum(pizzas.price * order_details.quantity) AS revenue
FROM orders
JOIN order_details
ON orders.order_id = order_details.order_id
JOIN pizzas
ON pizzas.pizza_id = order_details.pizza_id
GROUP BY orders.order_date) AS sales ;
```

PURPOSE

Analyze the cumulative revenue generated over time.

INSIGHT

The code analyzes the cumulative revenue generated over time. It calculates the total revenue for each order date and then uses a window function to sum up the revenue for all previous order dates. This allows you to see how the total revenue has grown over time.

PURPOSE

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

INSIGHT

The code determines the top 3 most ordered pizza types based on revenue for each pizza category. It calculates the revenue for each pizza type and then uses a window function to rank the pizza types within each category by their revenue. Finally, it filters the results to only include the top 3 pizza types for each category.

```sql
#Determine the top 3 most ordered pizza types based on revenue for each pizza category
SELECT category, name, revenue FROM
(SELECT category, name, revenue, RANK() OVER(partition by category order by revenue desc) as rnk
FROM
(SELECT pizza_types.category, pizza_types.name, sum(pizzas.price * order_details.quantity) 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.category, pizza_types.name) AS a) AS b
WHERE rnk<=3;
```

Result Grid | Filter Rows: | Export:

category	name	revenue
Chicken	The Thai Chicken Pizza	43434.25
Chicken	The Barbecue Chicken Pizza	42768
Chicken	The California Chicken Pizza	41409.5
Classic	The Classic Deluxe Pizza	38180.5
Classic	The Hawaiian Pizza	32273.25
Classic	The Pepperoni Pizza	30161.75
Supreme	The Spicy Italian Pizza	34831.25
Supreme	The Italian Supreme Pizza	33476.75
Supreme	The Sicilian Pizza	30940.5
Veggie	The Four Cheese Pizza	32265.70000000065
Veggie	The Mexicana Pizza	26780.75
Veggie	The Five Cheese Pizza	26066.5