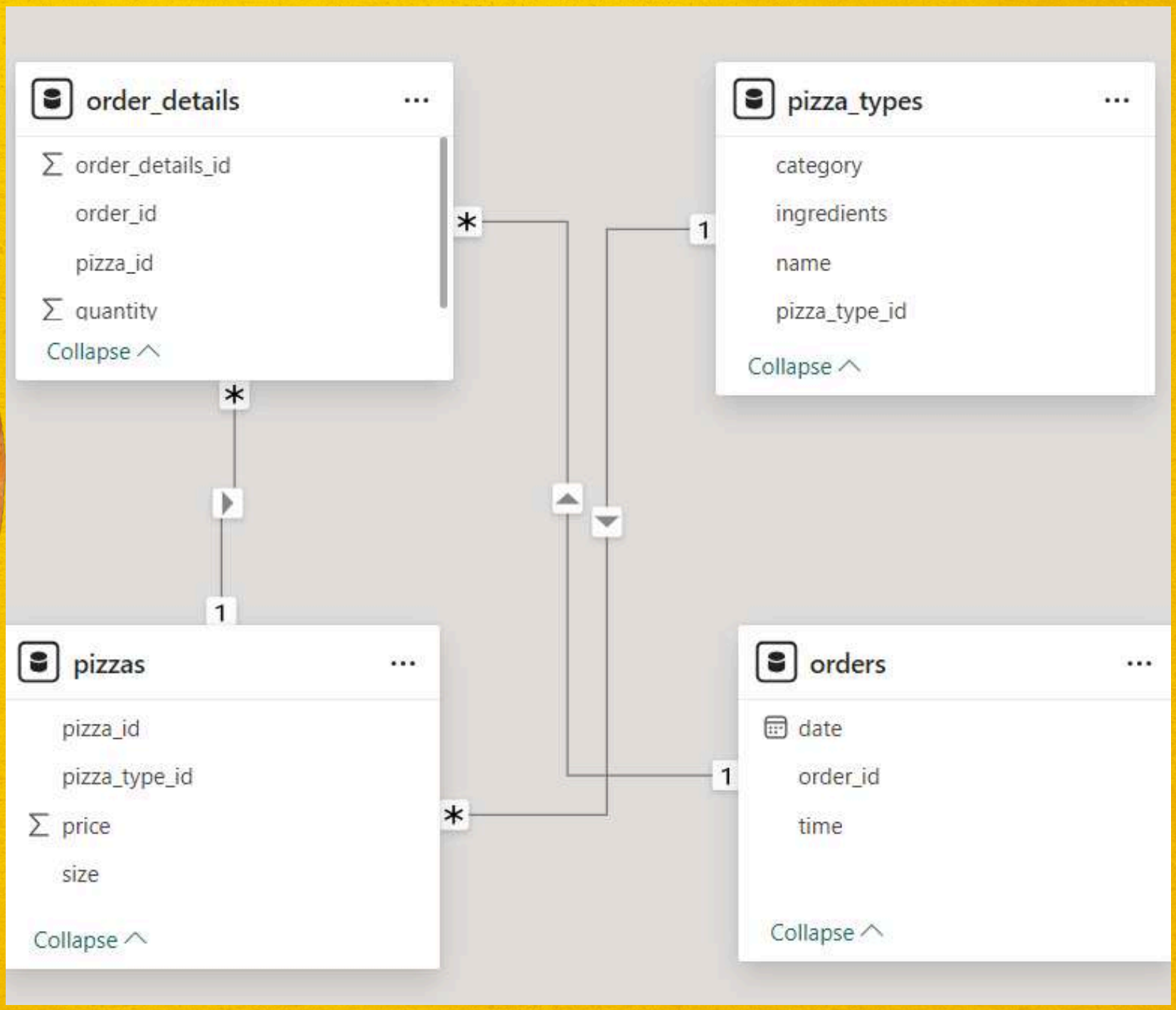
Pizza Time

This SQL project explored pizza sales data by writing queries to answer various business questions. The queries extracted insights such as top-selling pizzas, cumulative revenue over time, and popular pizza categories, utilizing techniques like joins, subqueries, and window functions to analyze the data. The project highlights my ability to work with SQL for data analysis and reporting.

Schema


order_details
order_details_id
order_id
pizza_id
quantity
Collapse
pizza_types
category
ingredients
name
pizza_type_id
Collapse
pizzas
pizza_id
pizza_type_id
price
size
Collapse
orders
date
order_id
time
Collapse
*
1
*
1
*
1

Queries

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

Retrieve the total number of orders placed.

```sql
SELECT
    COUNT(order_id) AS Total_Orders
FROM
    orders;
```

Result Grid

	Total_Orders
▶	21350

Calculate the total revenue generated from pizza sales.

```sql
SELECT
    ROUND(SUM(orders_details.quantity * pizzas.price),
            2) AS total_sales
FROM
    orders_details
        JOIN
    pizzas USING (pizza_id);
```

Result Grid

	total_sales
▶	817860.05

Identify the highest-priced pizza.

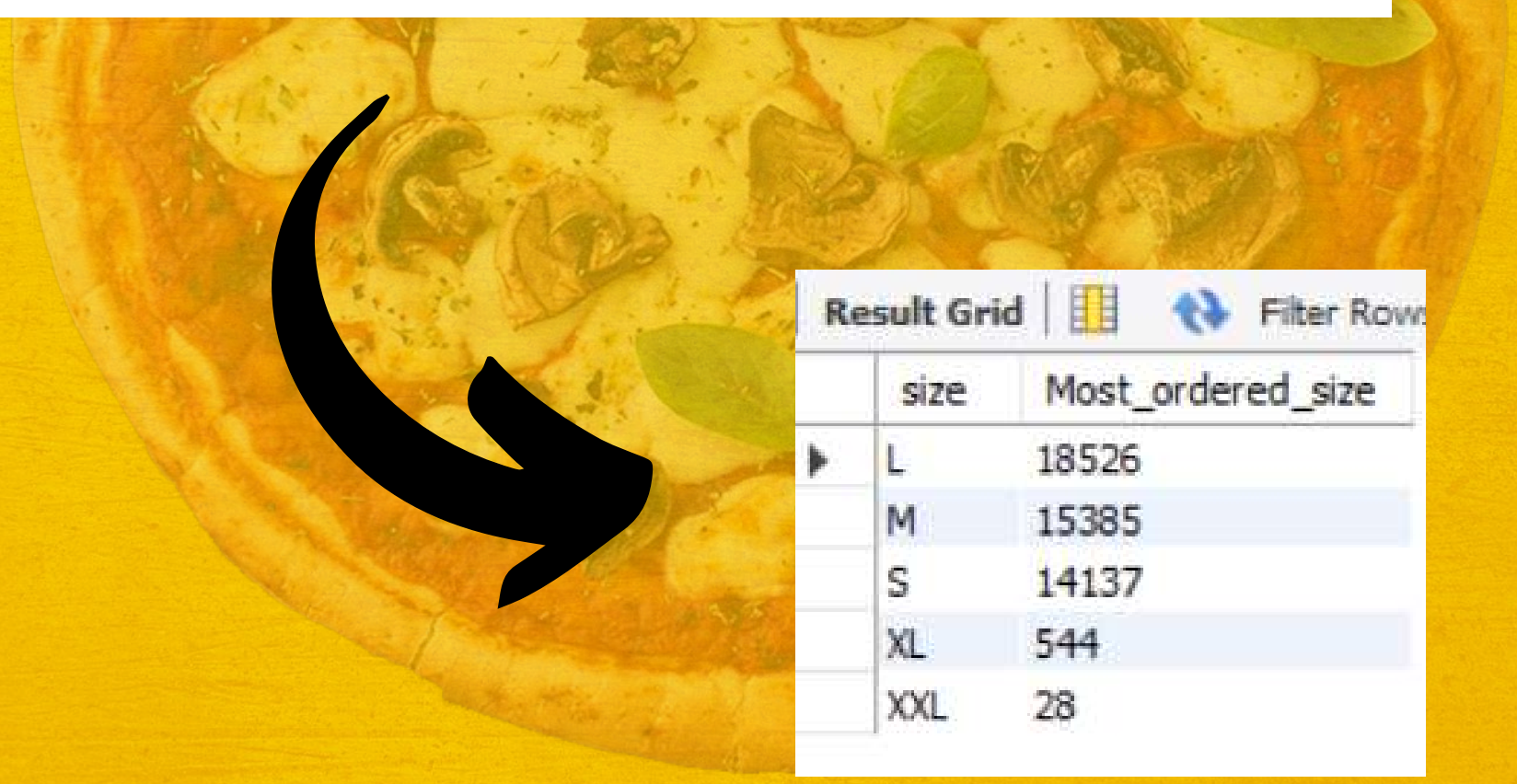
```
SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
        JOIN
    pizzas USING (pizza_type_id)
ORDER BY pizzas.price DESC
LIMIT 1;
```

Result Grid | Filter Ro

	name	price
▶	The Greek Pizza	35.95

Identify the most common pizza size ordered.

```sql
SELECT
    pizzas.size,
    COUNT(orders_details.order_details_id) AS Most_ordered_size
FROM
    pizzas
        JOIN
    orders_details USING (pizza_id)
GROUP BY pizzas.size
ORDER BY Most_ordered_size DESC;
```



Result Grid | Filter Row

size	Most_ordered_size
L	18526
M	15385
S	14137
XL	544
XXL	28

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

```sql
SELECT
    pizza_types.name,
    SUM(orders_details.quantity) AS Total_Quantity
FROM
    pizza_types
        JOIN
    pizzas USING (pizza_type_id)
        JOIN
    orders_details USING (pizza_id)
GROUP BY pizza_types.name
ORDER BY total_Quantity DESC
LIMIT 5;
```

Result Grid | Filter Rows:

name	Total_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.

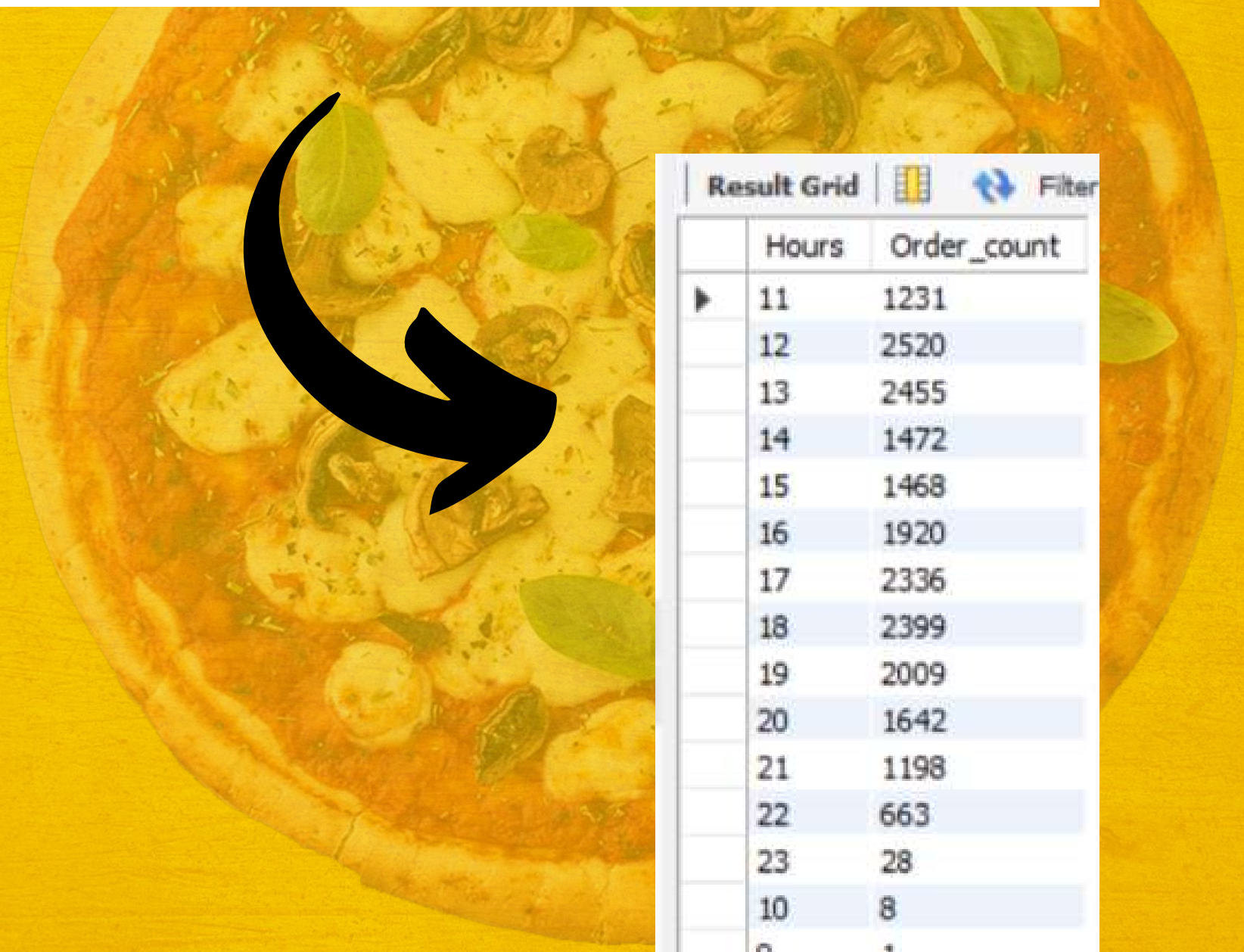
```
SELECT
    pizza_types.category,
    SUM(orders_details.quantity) AS quantity
FROM
    pizza_types
        JOIN
    pizzas USING (pizza_type_id)
        JOIN
    orders_details USING (pizza_id)
GROUP BY pizza_types.category
ORDER BY quantity DESC;
```

Result Grid | Filter

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Determine the distribution of orders by hour of the day.

```sql
SELECT
    HOUR(order_time) AS Hours, COUNT(order_id) AS Order_count
FROM
    orders
GROUP BY HOUR(order_time);
```



Result Grid | Filter

Hours	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.

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

Result Grid | Filter Ro

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

Group the orders by date and calculate the average number of daily pizzas.

```sql
SELECT
    ROUND(AVG(total_quantity), 0) as Average_pizza_per_day
FROM
    (SELECT
        orders.order_date,
            SUM(orders_details.quantity) AS total_quantity
    FROM
        orders
    JOIN orders_details USING (order_id)
    GROUP BY orders.order_date) AS order_quantity;
```

Result Grid | Filter Ro

	Average_pizza_per_day
▶	138

Determine the top 5 most ordered pizza types based on revenue.

```sql
SELECT
    pizza_types.name,
    SUM(orders_details.quantity * pizzas.price) AS Total_revenue
FROM
    pizza_types
        JOIN
    pizzas USING (pizza_type_id)
        JOIN
    orders_details USING (pizza_id)
GROUP BY pizza_types.name
ORDER BY Total_revenue DESC
LIMIT 5;
```

Result Grid | Filter Rows:

name	Total_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 Spicy Italian Pizza	34831.25

Calculate the percentage contribution of each pizza type to total revenue.

```sql
SELECT
    pizza_types.category,
    ROUND(SUM(orders_details.quantity * pizzas.price) / (SELECT
                    ROUND(SUM(orders_details.quantity * pizzas.price),
                                2) AS total_sales
                FROM
                    orders_details
                        JOIN
                    pizzas USING (pizza_id)) * 100,
            2) AS Revenue
FROM
    pizza_types
        JOIN
    pizzas USING (pizza_type_id)
        JOIN
    orders_details USING (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.

```sql
SELECT order_date,
       round(SUM(revenue) OVER (ORDER BY order_date),2) AS Cum_revenue
FROM (
    SELECT orders.order_date,
           ROUND(SUM(orders_details.quantity * pizzas.price), 2) AS revenue
    FROM orders_details
    JOIN pizzas USING (pizza_id)
    JOIN orders USING (order_id)
    GROUP BY orders.order_date
) AS revenue_by_date;
```

Result Grid | Filter Rows:

order_date	Cum_revenue
2015-01-01	2713.85
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	23990.35
2015-01-11	25862.65
2015-01-12	27781.7
2015-01-13	29831.3
2015-01-14	32358.7
2015-01-15	34343.5
2015-01-16	36937.65
2015-01-17	39001.75
2015-01-18	40978.6
2015-01-19	43365.75
2015-01-20	45763.65

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

```sql
SELECT category, name, revenue
FROM (
    SELECT category, name, revenue,
           RANK() OVER (PARTITION BY category ORDER BY revenue DESC) AS Reve
    FROM (
        SELECT pizza_types.category, pizza_types.name,
               SUM(orders_details.quantity * pizzas.price) AS revenue
        FROM pizza_types
        JOIN pizzas USING (pizza_type_id)
        JOIN orders_details USING (pizza_id)
        GROUP BY pizza_types.category, pizza_types.name
    ) AS Data
) AS Data_2
WHERE Reve <= 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