LARANA PIZZA

PIZZA ANALYSIS

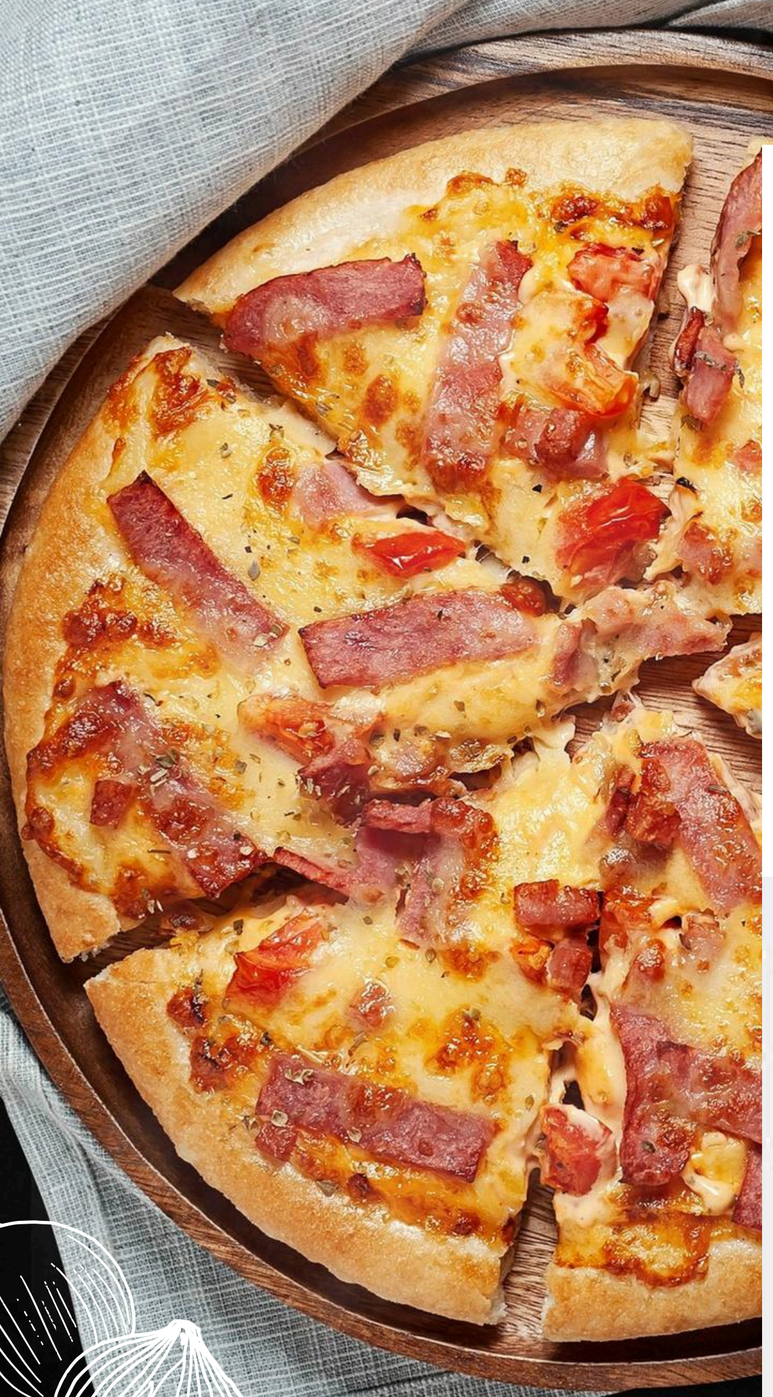
LARANA PIZZA

BUSINESS INSIGHTS USING SQL.

Key Objectives:

- Understand customer ordering behavior
- Identify top-performing pizzas and categories
- Analyze revenue generation over time
- Provide actionable business insights for strategic decisions

YOGESH KUMAR ATAL

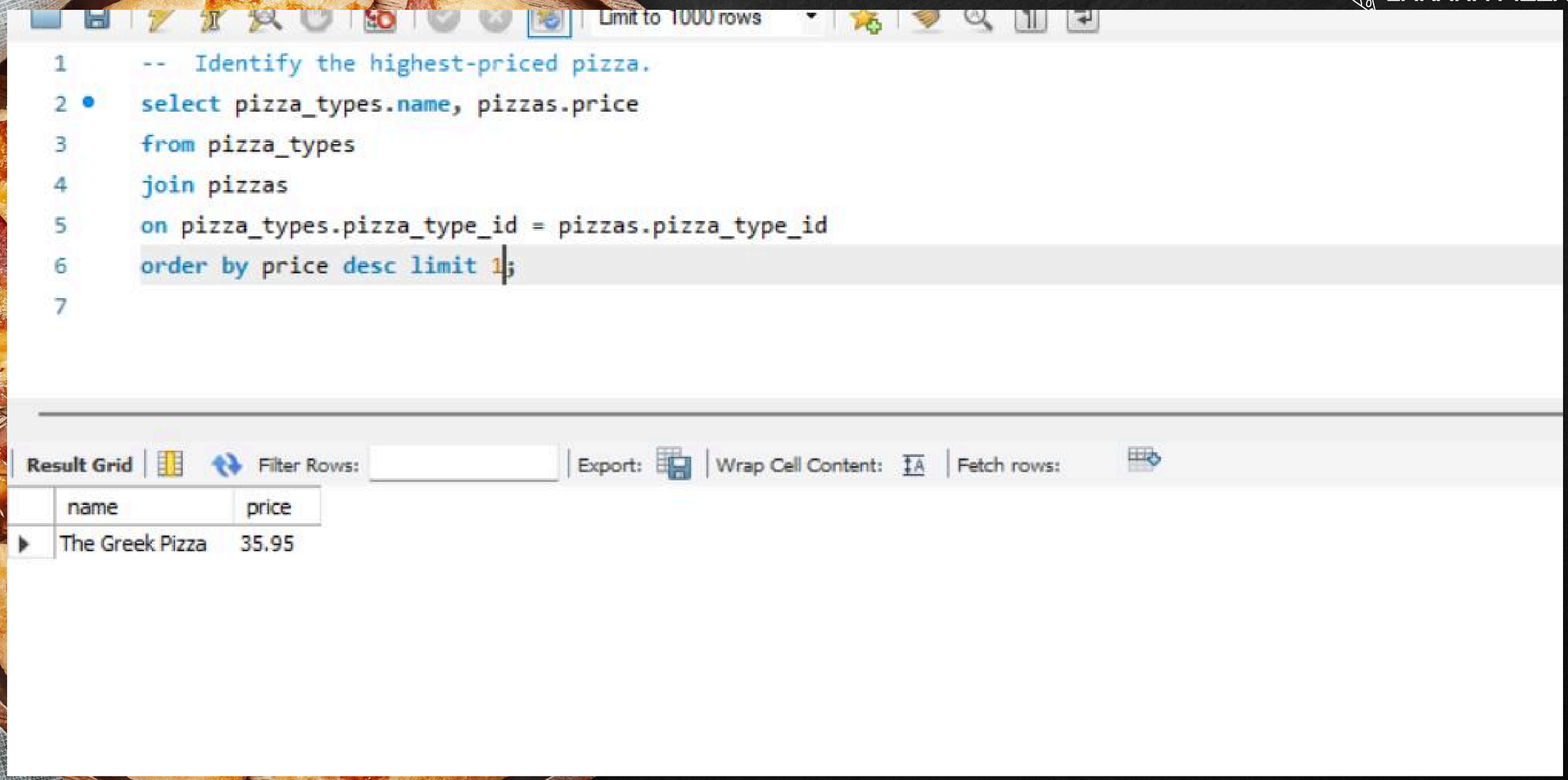


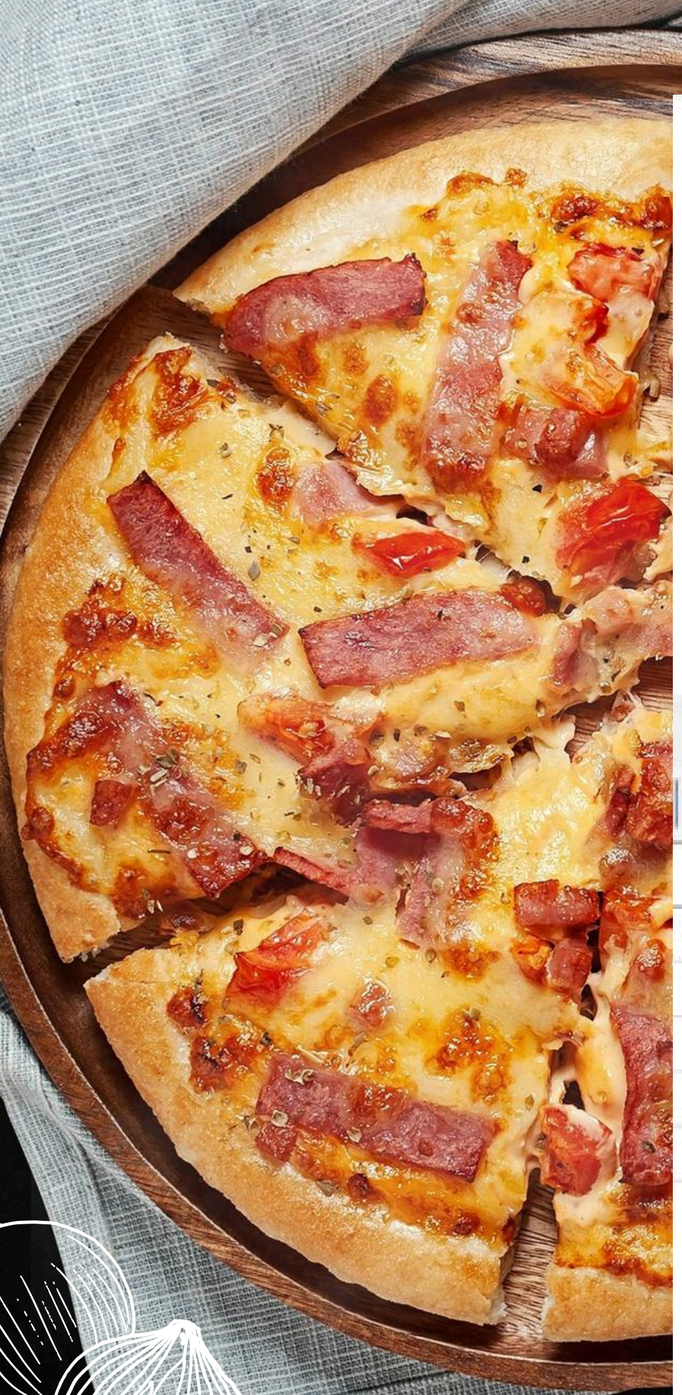
```sql
-- Retrieve the total number of orders placed.
select count(order_id) as total_number  from orders;
 -- Calculate the total revenue generated from pizza sales.

 select sum(order_details.quantity * pizzas.price) as total_revenue
 from order_details
 join pizzas
 on order_details.order_id = pizzas.order_id;
```

Result Grid | Filter Rows:

total_revenue
817860.05

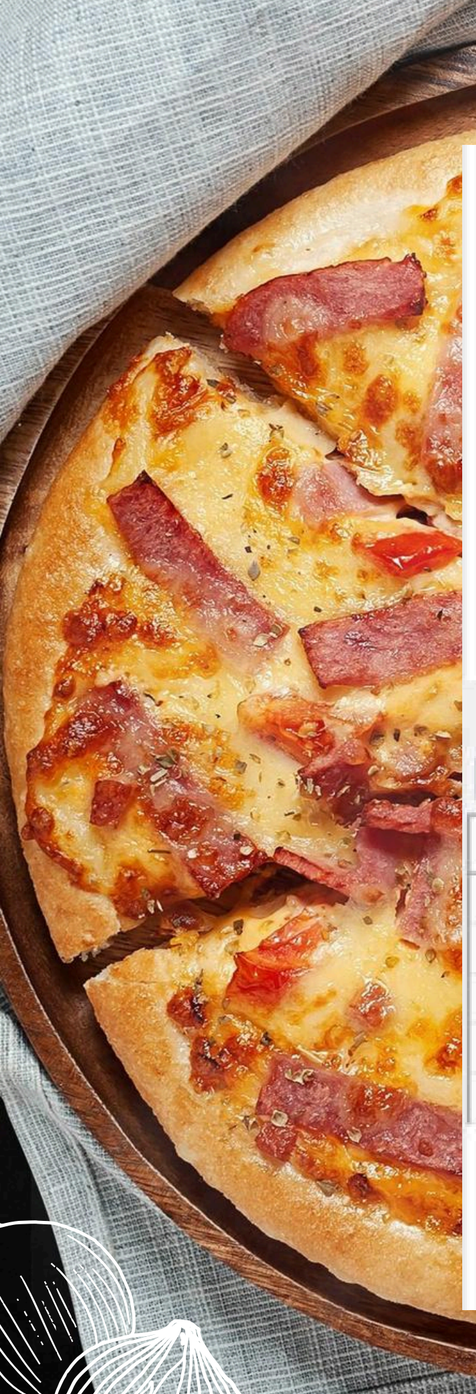

Limit to 1000 rows
1 -- Identify the highest-priced pizza.
2 select pizza_types.name, pizzas.price
3 from pizza_types
4 join pizzas
5 on pizza_types.pizza_type_id = pizzas.pizza_type_id
6 order by price desc limit 1;
7
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
Fetch rows:
name
price
The Greek Pizza
35.95



```sql
-- Identify the most common pizza size ordered.
SELECT pizzas.size,
       COUNT(*) AS total_orders
FROM order_details
JOIN pizzas
  ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizzas.size
ORDER BY total_orders DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

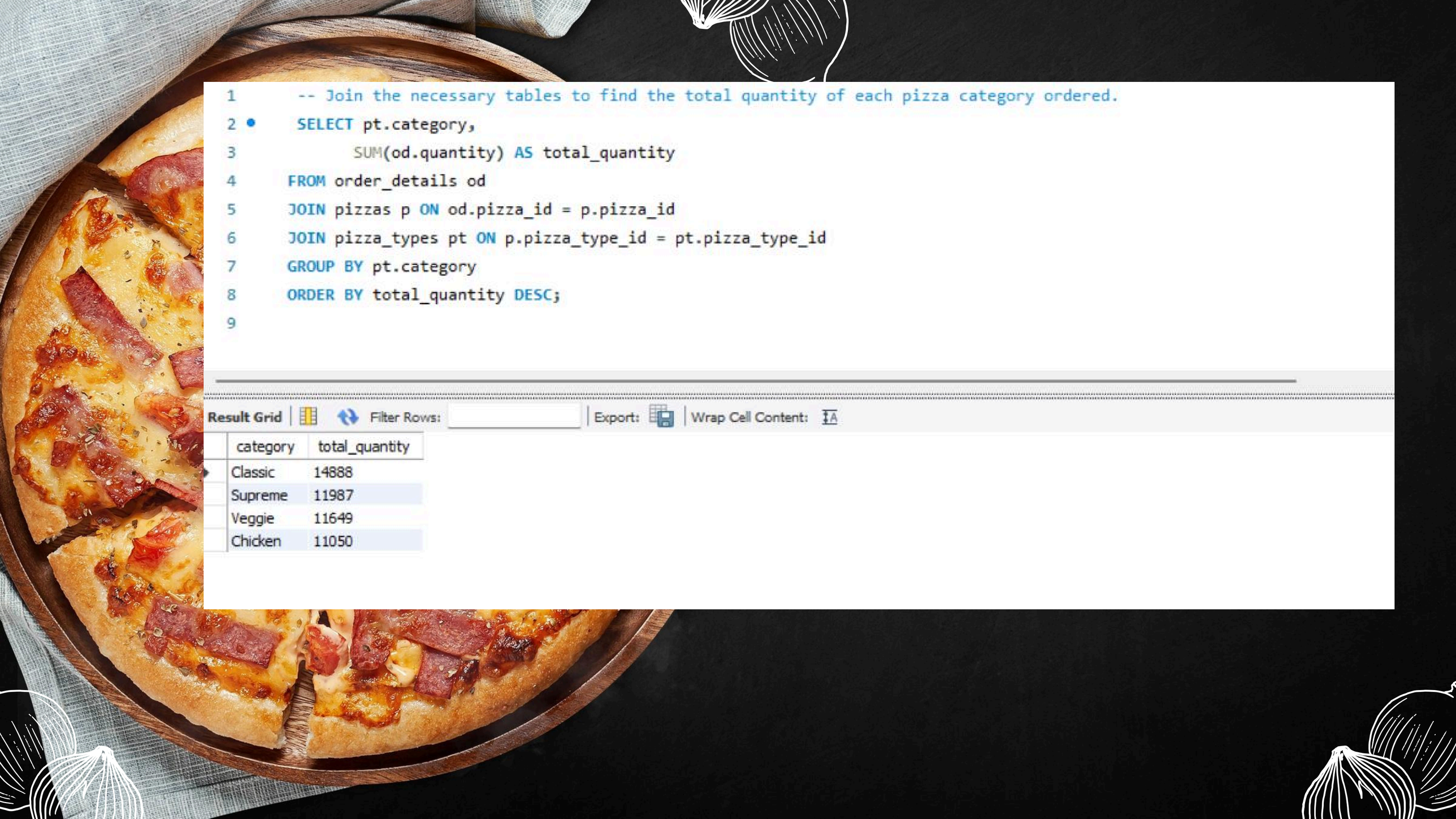
size	total_orders
L	18526
M	15385
S	14137
XL	544
XXL	28



```sql
-- List the top 5 most ordered pizza types along with their quantities.
SELECT pizzas.pizza_type_id,
       SUM(order_details.quantity) AS total_quantity
FROM order_details
JOIN pizzas
  ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizzas.pizza_type_id
ORDER BY total_quantity DESC
LIMIT 5;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

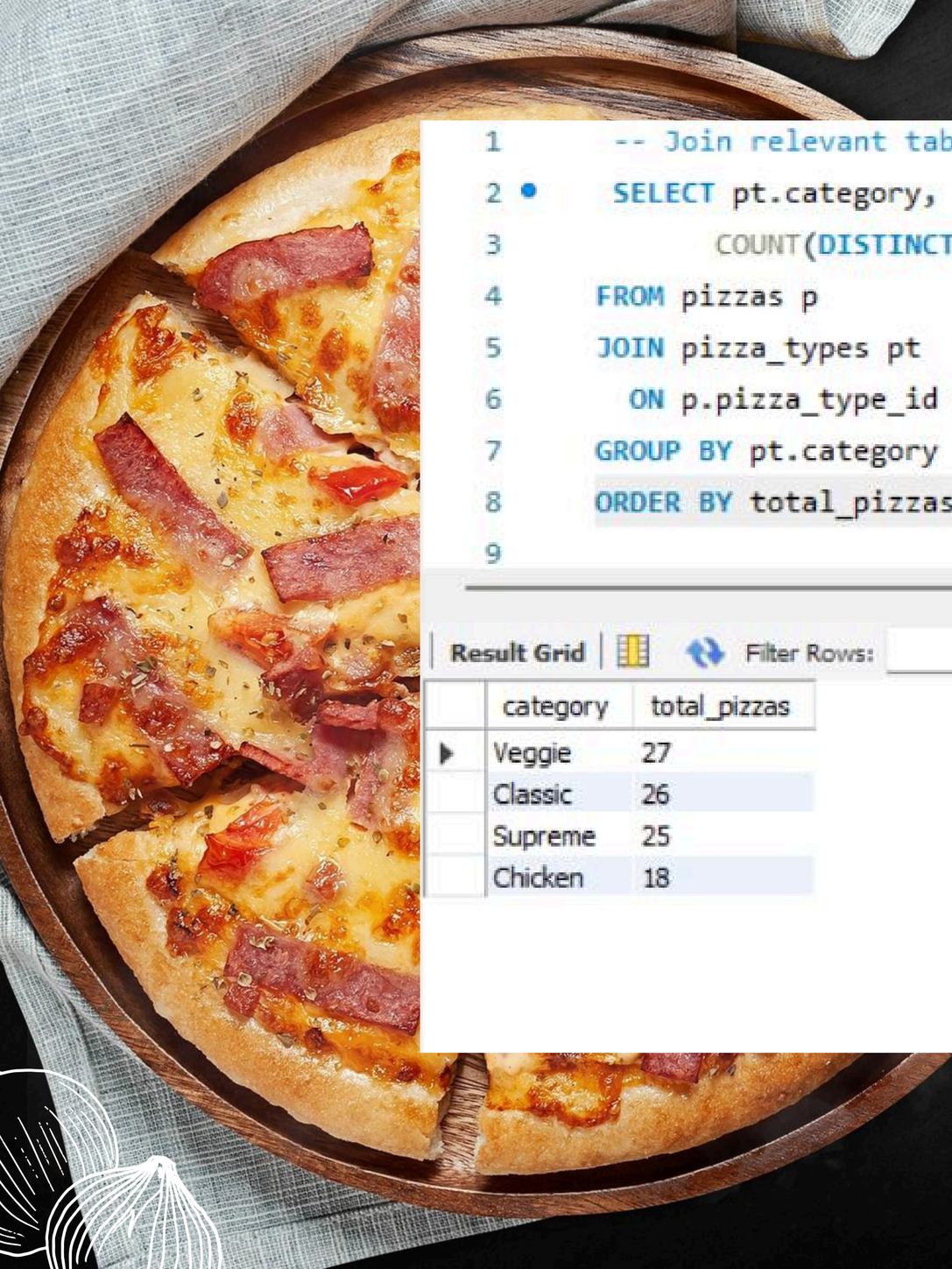
pizza_type_id	total_quantity
classic_dlx	2453
bbq_ckn	2432
hawaiian	2422
pepperoni	2418
thai_ckn	2371



```sql
-- Join the necessary tables to find the total quantity of each pizza category ordered.
SELECT pt.category,
       SUM(od.quantity) AS total_quantity
FROM order_details od
JOIN pizzas p ON od.pizza_id = p.pizza_id
JOIN pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
GROUP BY pt.category
ORDER BY total_quantity DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

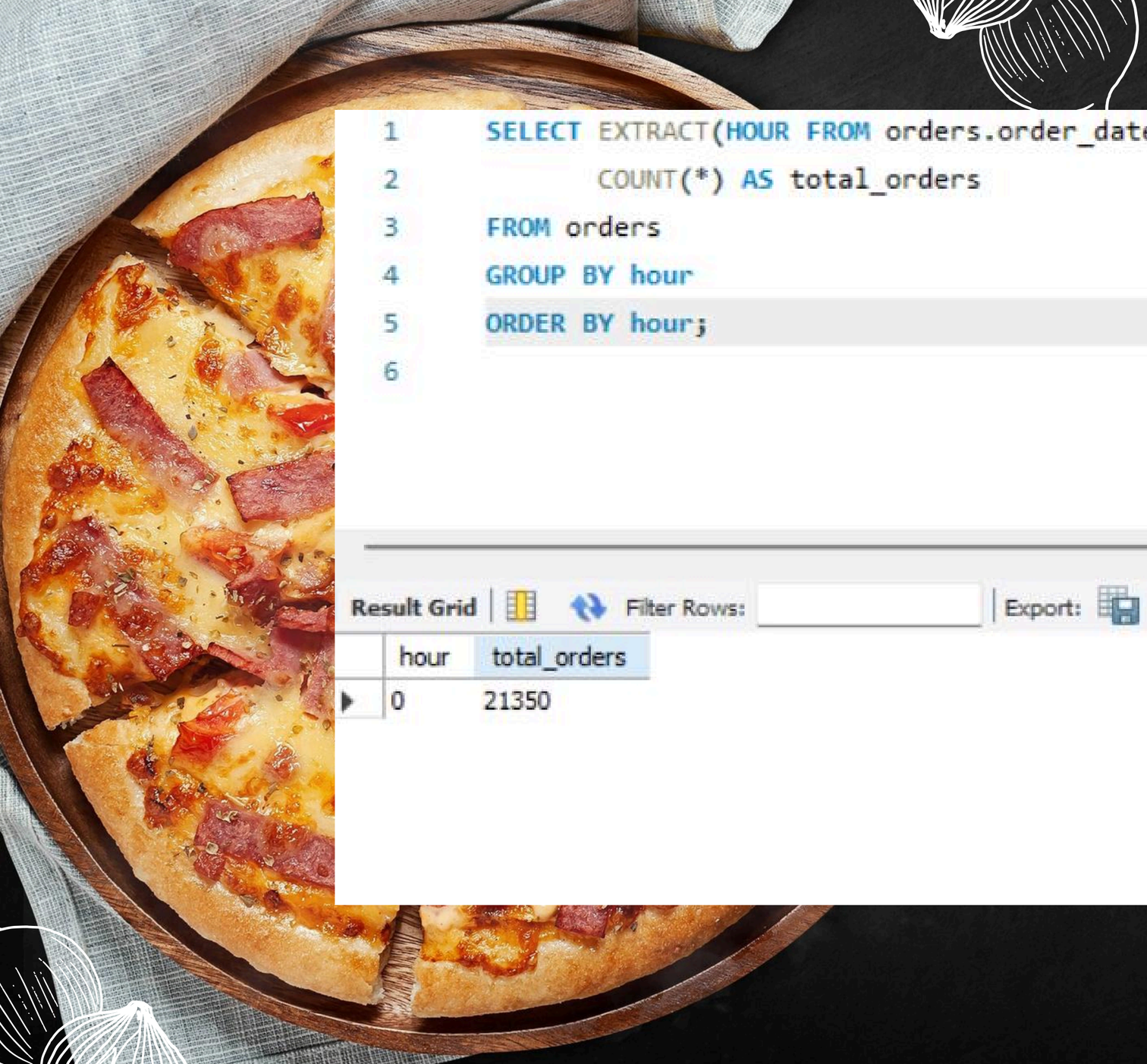
category	total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050



```sql
-- Join relevant tables to find the category-wise distribution of pizzas.
SELECT pt.category,
       COUNT(DISTINCT p.pizza_id) AS total_pizzas
FROM pizzas p
JOIN pizza_types pt
  ON p.pizza_type_id = pt.pizza_type_id
GROUP BY pt.category
ORDER BY total_pizzas DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

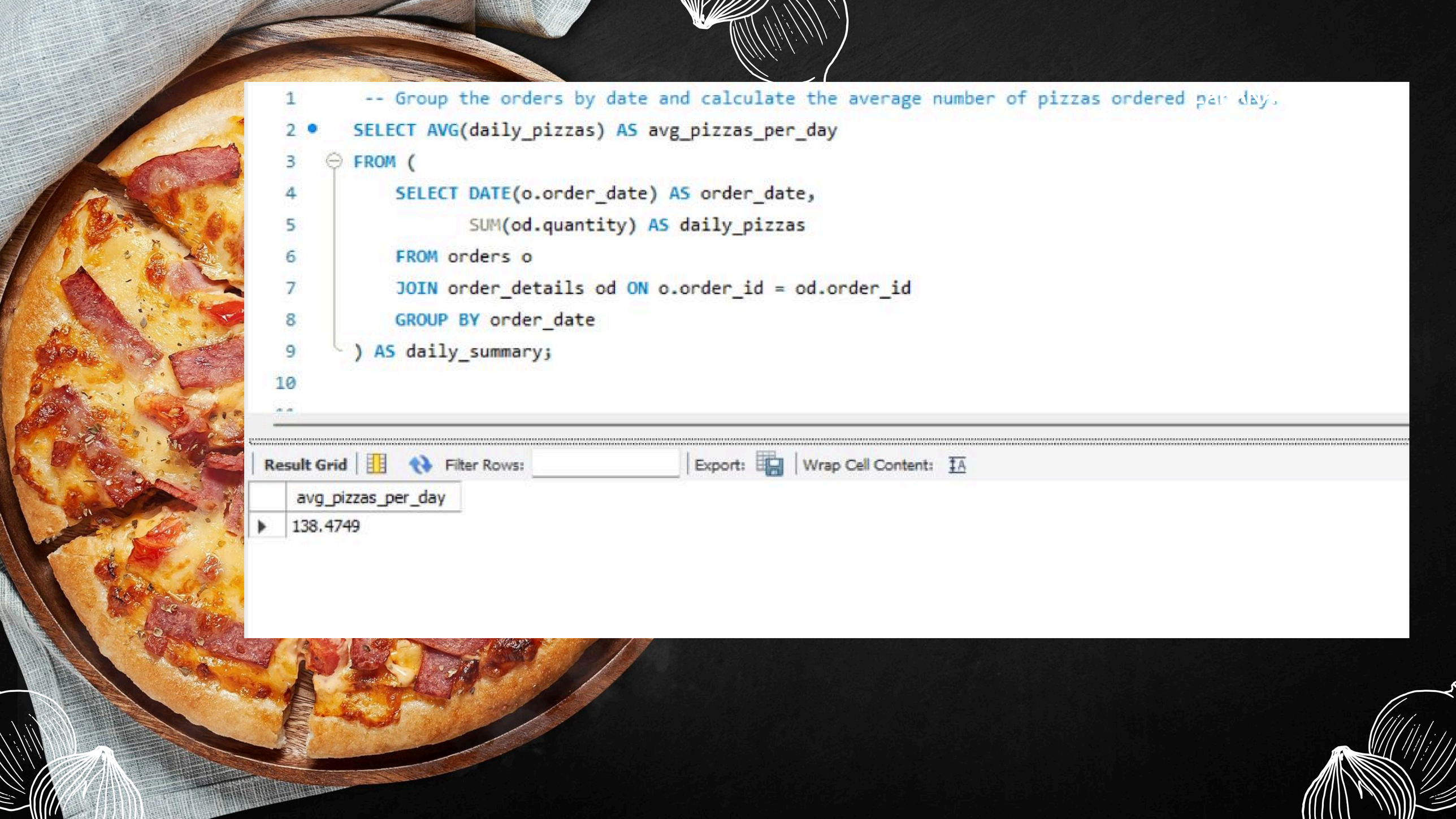
category	total_pizzas
Veggie	27
Classic	26
Supreme	25
Chicken	18



```sql
SELECT EXTRACT(HOUR FROM orders.order_date) AS hour,
       COUNT(*) AS total_orders
FROM orders
GROUP BY hour
ORDER BY hour;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

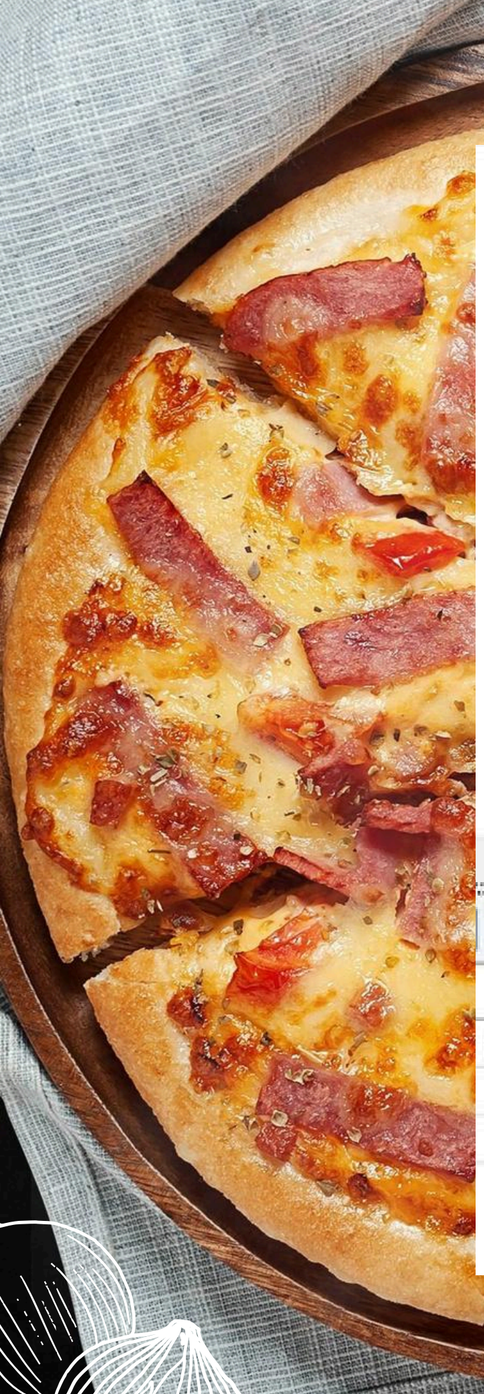
hour	total_orders
0	21350



```sql
-- Group the orders by date and calculate the average number of pizzas ordered per day.
SELECT AVG(daily_pizzas) AS avg_pizzas_per_day
FROM (
    SELECT DATE(o.order_date) AS order_date,
           SUM(od.quantity) AS daily_pizzas
    FROM orders o
    JOIN order_details od ON o.order_id = od.order_id
    GROUP BY order_date
) AS daily_summary;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

avg_pizzas_per_day
138.4749



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

SELECT pt.name AS pizza_name,
       SUM(od.quantity * p.price) AS total_revenue
FROM order_details od
JOIN pizzas p ON od.pizza_id = p.pizza_id
JOIN pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
GROUP BY pt.name
ORDER BY total_revenue DESC
LIMIT 3;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

pizza_name	total_revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

LARANA PIZZA

```sql
-- Calculate the percentage contribution of each pizza type to total revenue.
SELECT pt.name AS pizza_name,
       SUM(od.quantity * p.price) AS total_revenue
FROM order_details od
JOIN pizzas p ON od.pizza_id = p.pizza_id
JOIN pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
GROUP BY pt.name
ORDER BY total_revenue DESC
LIMIT 3;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

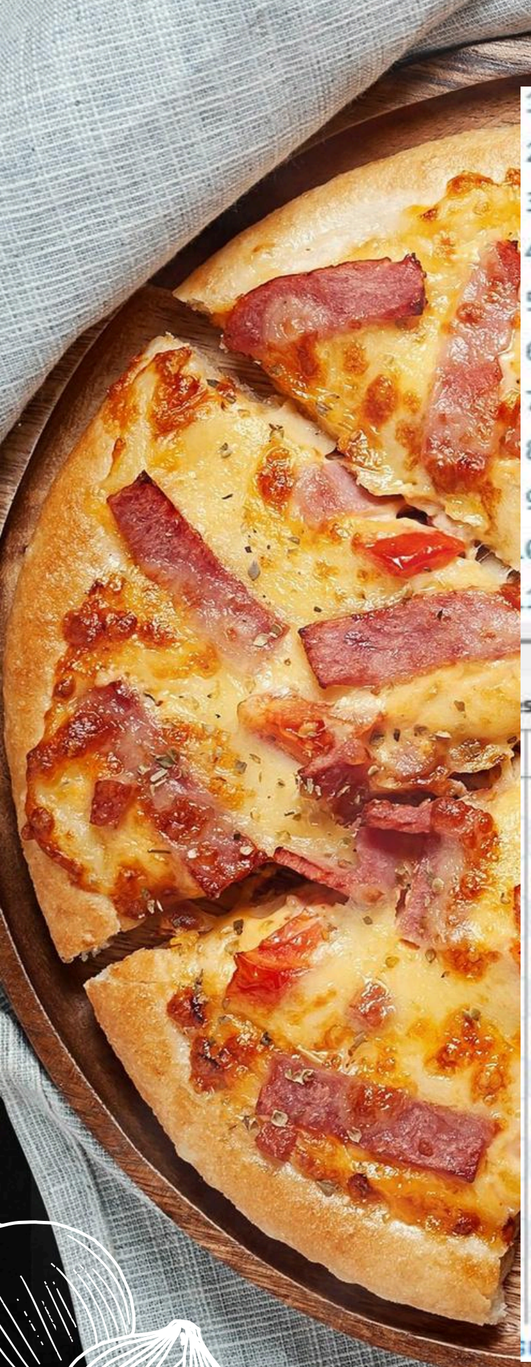
pizza_name	total_revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

```sql
-- Calculate the percentage contribution of each pizza type to total revenue.
SELECT
    pt.name AS pizza_name,
    ROUND(SUM(od.quantity * p.price), 2) AS revenue,
    ROUND(
        (SUM(od.quantity * p.price) /
        (SELECT SUM(od2.quantity * p2.price)
         FROM order_details od2
         JOIN pizzas p2 ON od2.pizza_id = p2.pizza_id)
        ) * 100, 2
    ) AS percentage_contribution
FROM order_details od
JOIN pizzas p ON od.pizza_id = p.pizza_id
JOIN pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
GROUP BY pt.name
ORDER BY percentage_contribution DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

pizza_name	revenue	percentage_contribution
The Thai Chicken Pizza	43434.25	5.31
The Barbecue Chicken Pizza	42768	5.23
The California Chicken Pizza	41409.5	5.06
The Classic Deluxe Pizza	38180.5	4.67
The Spicy Italian Pizza	34831.25	4.26
The Southwest Chicken Pizza	34705.75	4.24
The Italian Supreme Pizza	33476.75	4.09

esult 1

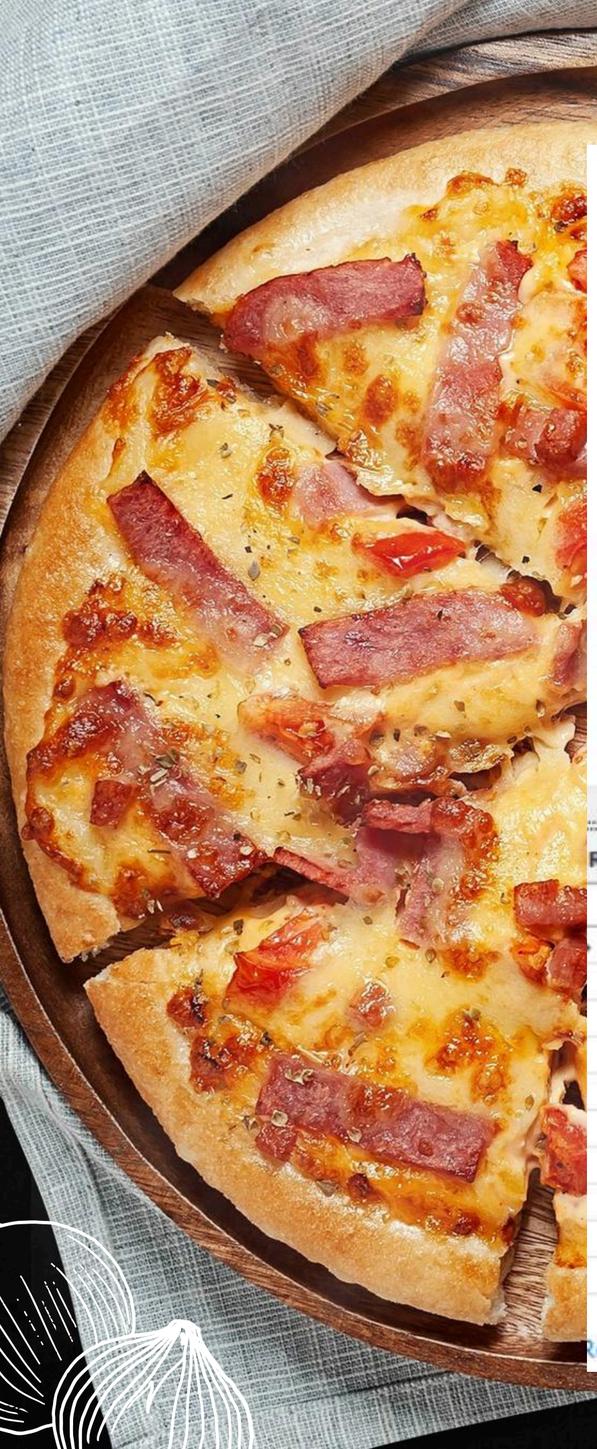


```sql
-- Analyze the cumulative revenue generated over time.
SELECT
    DATE(o.order_date) AS order_date,
    ROUND(SUM(od.quantity * p.price), 2) AS daily_revenue,
    ROUND(SUM(SUM(od.quantity * p.price)) OVER (ORDER BY DATE(o.order_date)), 2) AS cumulative_revenue
FROM orders o
JOIN order_details od ON o.order_id = od.order_id
JOIN pizzas p ON od.pizza_id = p.pizza_id
GROUP BY order_date
ORDER BY order_date;
```

sult Grid | Filter Rows: | Export: | Wrap Cell Content:

order_date	daily_revenue	cumulative_revenue
2015-02-18	2642.85	111977.3
2015-02-19	2030.25	114007.55
2015-02-20	2891.15	116898.7
2015-02-21	2111	119009.7
2015-02-22	1579.95	120589.65
2015-02-23	2168.55	122758.2
2015-02-24	2194.55	124952.75
2015-02-25	2341.3	127294.05
2015-02-26	2261.3	129555.35
2015-02-27	2857.95	132413.3
2015-02-28	2539.6	134952.9
2015-03-01	1598.55	136551.45
2015-03-02	2379.05	138930.5
2015-03-03	2287.9	141218.4

sult 1



```sql
-- Determine the top 3 most ordered pizza types based on revenue for each pizza category.
SELECT *
FROM (
    SELECT
        pt.category,
        pt.name AS pizza_name,
        ROUND(SUM(od.quantity * p.price), 2) AS revenue,
        RANK() OVER (PARTITION BY pt.category ORDER BY SUM(od.quantity * p.price) DESC) AS rank_in_category
    FROM order_details od
    JOIN pizzas p ON od.pizza_id = p.pizza_id
    JOIN pizza_types pt ON p.pizza_type_id = pt.pizza_type_id
    GROUP BY pt.category, pt.name
) ranked
WHERE rank_in_category <= 3
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

category	pizza_name	revenue	rank_in_category
Chicken	The Thai Chicken Pizza	43434.25	1
Chicken	The Barbecue Chicken Pizza	42768	2
Chicken	The California Chicken Pizza	41409.5	3
Classic	The Classic Deluxe Pizza	38180.5	1
Classic	The Hawaiian Pizza	32273.25	2
Classic	The Pepperoni Pizza	30161.75	3
Supreme	The Spicy Italian Pizza	34831.25	1
Supreme	The Italian Supreme Pizza	33476.75	2
Supreme	The Sicilian Pizza	30940.5	3
Veggie	The Four Cheese Pizza	32265.7	1
Veggie	The Mexicana Pizza	26780.75	2

Result 1

THANK YOU!