

PIZZA SALES ANALYSIS USING SQL



BILLU'S PIZZA HUB



INTRODUCTION

Hello! I am Karuna Rathore an Aspiring Data Analyst and Economics professional.

- Objective: Used SQL to analyze a comprehensive pizza sales dataset. Used MYSQL for this Project. The dataset includes four tables:
1. Orders 2. Order_Detail 3. Pizzas 4. Pizza_Types
- Key Activities: Executed SQL queries to answer critical business questions, focusing on uncovering patterns in orders, understanding customer preferences, evaluating revenue trends, and assessing product performance.
- Outcome: This project showcases proficiency in working with relational databases and converting unprocessed data into actionable insights and strategic business intelligence.
- Business Impact: The analysis underscores the significance of data-driven decision-making in understanding sales patterns and optimizing inventory management, particularly in the dynamic food industry.



QUESTIONS

Basic:

- 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.

Intermediate:

- 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.

Advanced:

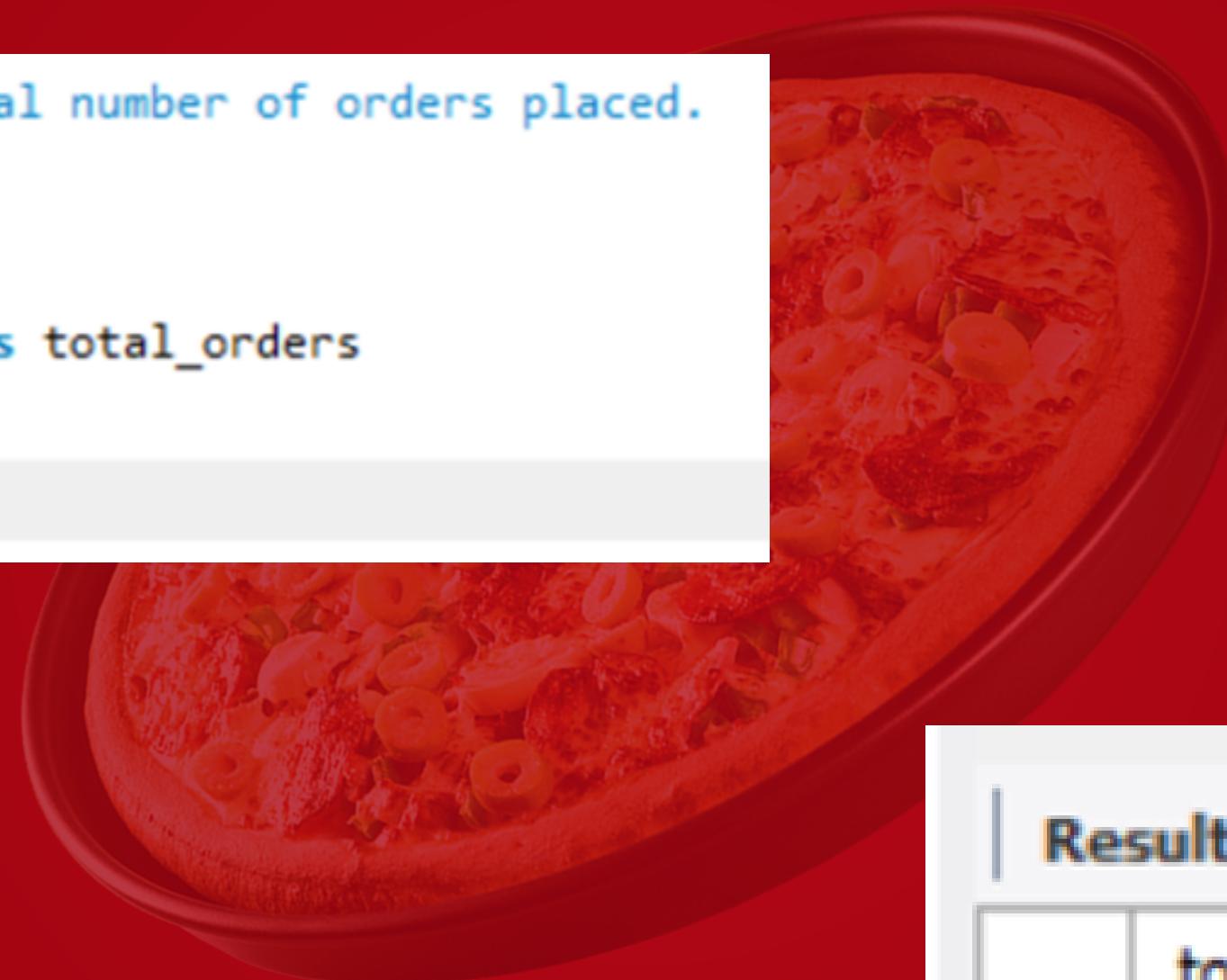
- Calculate the percentage contribution of each pizza type to total revenue.
- Analyze the cumulative revenue generated over time.
- Determine the top 3 most ordered pizza types based on revenue for each pizza category.

BUSINESS INSIGHTS

- Large pizzas were the most commonly ordered size around 18,526 units, indicating a preference for larger portions among customers.
- The classic Deluxe Pizza is the most ordered (Quantity sold) pizza type followed by others.
- Classic Category has highest Total quantity of pizza ordered in comparison to Supreme, Veggie, and Chicken.
- The busiest hours for orders were in the evening (5–7 PM) and afternoon (12–1 PM), reflecting mealtime peaks for the Billu's Pizza Hub.
- in terms of revenue, the Chicken category dominated, with its top three pizza types contributing significantly. Contribute around 23.96% in total revenue.
- Total ordered placed are around 21,350 and total revenue generated is around \$8,17,860.
- Average pizza ordered per day is 138.

Q1. RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED

```
20    -- Retrieve the total number of orders placed.  
21  
22 • select  
23     count(order_id) as total_orders  
24   from  
25     orders;
```



A large pizza with various toppings like olives and cheese is shown on a metal tray, positioned behind the code editor window.

Result Grid |  Filter R

total_orders
21350

Q2. CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES

```
25      -- Calculate the total revenue generated from pizza sales.  
26  
27 •   select  
28     round(  
29       sum(  
30         order_details.quantity * pizzas.price  
31       ),  
32       2  
33     ) as total_sales  
34   from  
35     order_details  
36   join pizzas on pizzas.pizza_id = order_details.pizza_id;  
37
```



	total_sales
▶	817860.05

Q3. IDENTIFY THE HIGHEST-PRICED PIZZA

```
33    -- Identify the highest-priced pizza.  
34  
35 • select  
36     pizza_types.name,  
37     pizzas.price  
38   from  
39     pizza_types  
40   join pizzas on pizza_types.pizza_type_id = pizzas.pizza_type_id  
41   order by  
42     pizzas.price desc  
43   limit 1;
```



Result Grid | Filter Rows:

	name	price
▶	The Greek Pizza	35.95

Q4. IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED

```
40    -- Identify the most common pizza size ordered.  
41  
42 • select  
43     pizzas.size,  
44     count(order_details.order_details_id) as order_count  
45   from  
46     pizzas  
47   join order_details on pizzas.pizza_id = order_details.pizza_id  
48   group by  
49     pizzas.size  
50   order by  
51     order_count desc;
```

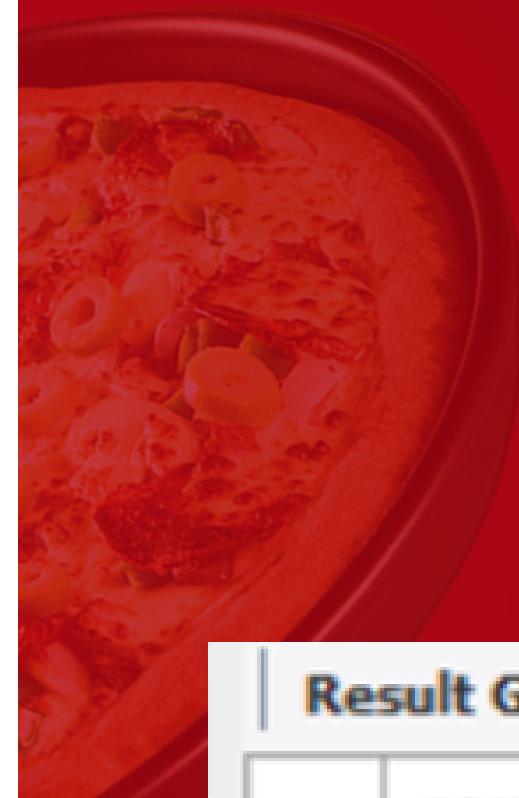


Result Grid | Filter Rows:

	size	order_count
▶	L	18526
	M	15385
	S	14137
	XL	544
	XXL	28

Q5. LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES

```
66      -- List the top 5 most ordered pizza types along with their quantities.  
67  
68 •  select  
69      pizza_types.name,  
70      sum(order_details.quantity) as quantity  
71  from  
72      pizza_types  
73  join pizzas on pizza_types.pizza_type_id = pizzas.pizza_type_id  
74  join order_details on order_details.pizza_id = pizzas.pizza_id  
75  group by  
76      pizza_types.name  
77  order by  
78      quantity desc  
79  limit 5;
```

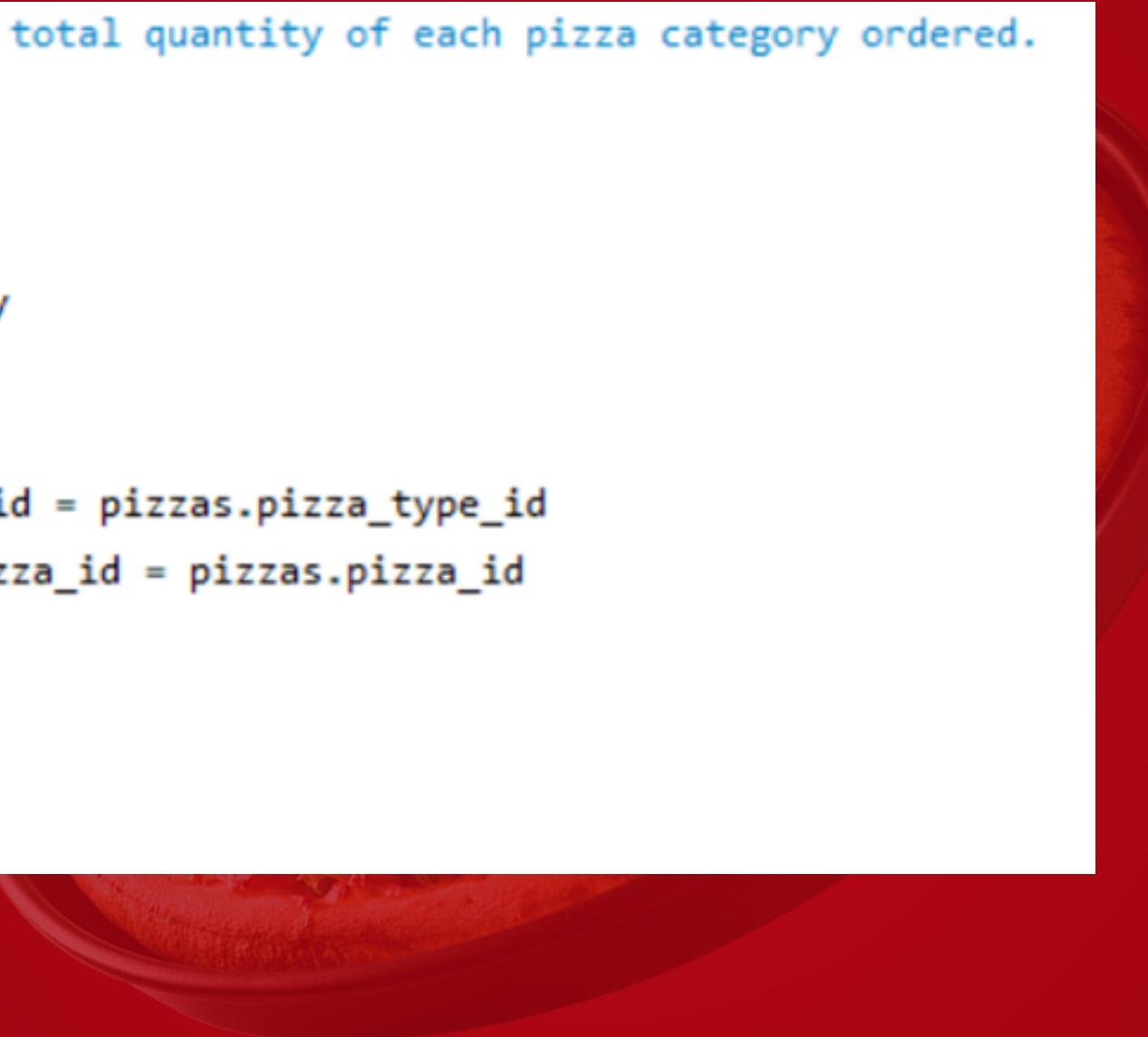


Result Grid | Filter Rows:

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

Q6. JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED

```
81    -- Join the necessary tables to find the total quantity of each pizza category ordered.  
82  
83 • select  
84     pizza_types.category,  
85     sum(order_details.quantity) as quantity  
86   from  
87     pizza_types  
88   join pizzas on pizza_types.pizza_type_id = pizzas.pizza_type_id  
89   join order_details on order_details.pizza_id = pizzas.pizza_id  
90   group by  
91     pizza_types.category  
92   order by  
93     quantity desc;
```



A screenshot of a database query results grid. The grid has a header row with columns for 'category' and 'quantity'. Below the header, there are four data rows. The first row is highlighted in light blue. The data is as follows:

	category	quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

Q7. DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY

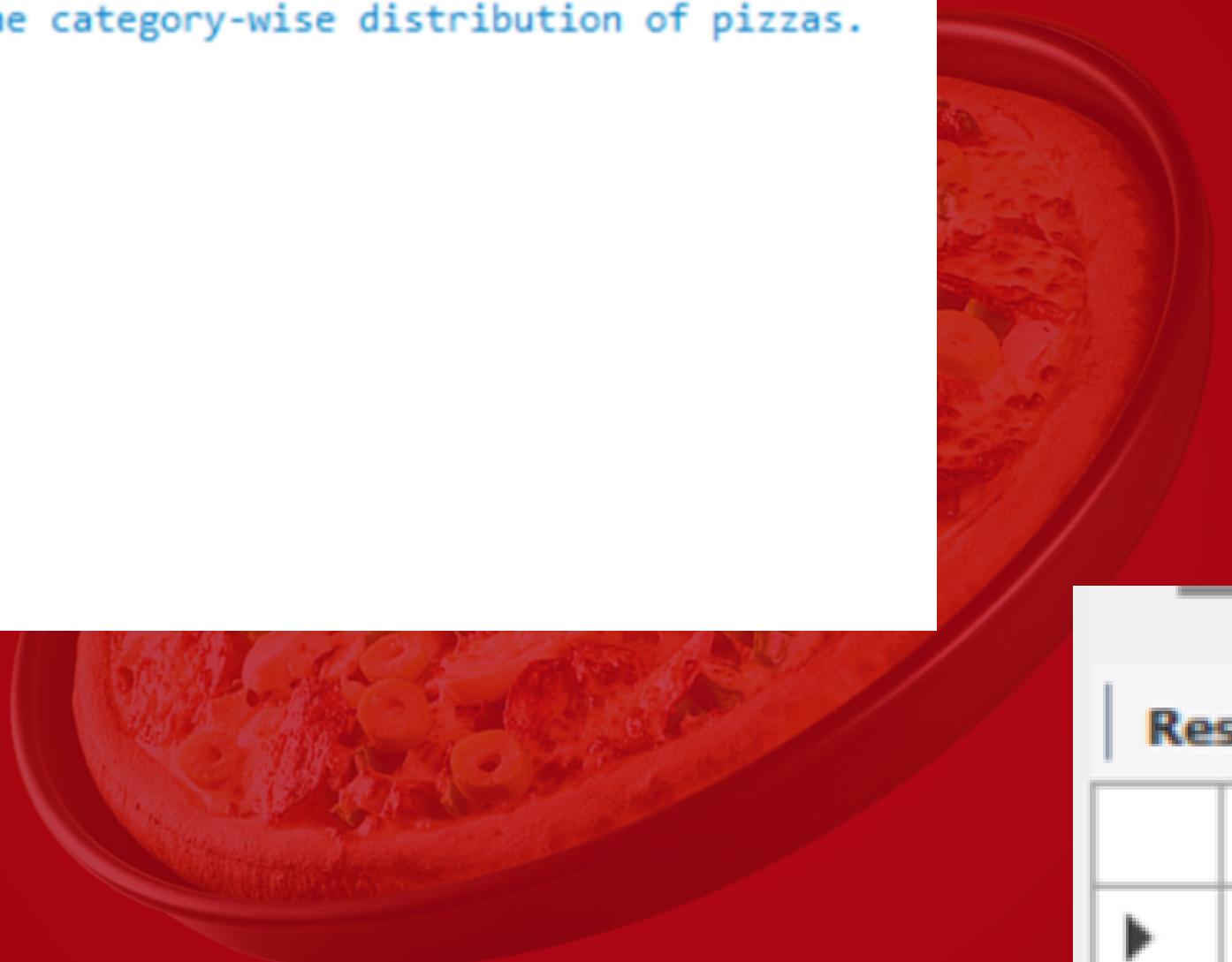
```
95    -- Determine the Distribution of orders by hour of the Day.  
96  
97 • select  
98     hour(order_time) as hour,  
99     count(order_id) as order_count  
100    from  
101    orders  
102    group by  
103    hour;  
104
```



	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

Q8. JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.

```
105    -- Join relevant tables to find the category-wise distribution of pizzas.  
106  
107 •   select  
108     category,  
109     count(name)  
110   from  
111     pizza_types  
112   group by  
113     category;  
114
```



The image shows a large, round pizza with various toppings like pepperoni and cheese, served in a metal tray. It is positioned in the background, partially obscured by the code window and the results grid.

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

QS. GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY

```
115      -- Group the orders by date and calculate the average number of pizzas ordered per day.  
116  
117 •   select  
118     round(  
119       avg(quantity),  
120       0  
121     ) as avg_pizza_ordered_per_day  
122   from  
123   (  
124     select  
125       orders.order_date,  
126       sum(order_details.quantity) as quantity  
127     from  
128       orders  
129       join order_details on orders.order_id = order_details.order_id  
130     group by  
131       orders.order_date  
132   ) as order_quantity;  
133
```



The screenshot shows a database query results window. At the top, there are buttons for "Result Grid" (highlighted in blue), "Filter Rows:" (with a dropdown menu), and other options like "Copy" and "Print". Below the buttons is a table with two rows. The first row contains a header cell "avg_pizza_ordered_per_day" and an empty data cell. The second row contains a right-pointing arrow icon and the value "138".

avg_pizza_ordered_per_day	
▶	138

Q10. DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

```
134      -- Determine the top 3 most ordered pizza types based on revenue.  
135  
136 •  select  
137      pizza_types.name,  
138      sum(  
139          order_details.quantity * pizzas.price  
140      ) as revenue  
141  from  
142      pizza_types  
143      join pizzas on pizza_types.pizza_type_id = pizzas.pizza_type_id  
144      join order_details on order_details.pizza_id = pizzas.pizza_id  
145  group by  
146      pizza_types.name  
147  order by  
148      revenue desc  
149  limit 3;
```



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

Q11. CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE

```
151    -- Calculate the percentage contribution of each pizza type to total revenue.  
152  
153 • select  
154     pizza_types.category,  
155     round((sum(order_details.quantity * pizzas.price) / (select round(sum(order_details.quantity * pizzas.price),  
156         2) as total_sales  
157     from  
158     order_details  
159     join pizzas on pizzas.pizza_id = order_details.pizza_id)) * 100,2) as revenue  
160     from  
161     pizza_types  
162     join pizzas on pizza_types.pizza_type_id = pizzas.pizza_type_id  
163     join order_details on order_details.pizza_id = pizzas.pizza_id  
164     group by  
165     pizza_types.category  
166     order by revenue desc;
```

Result Grid | Filter Rows

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

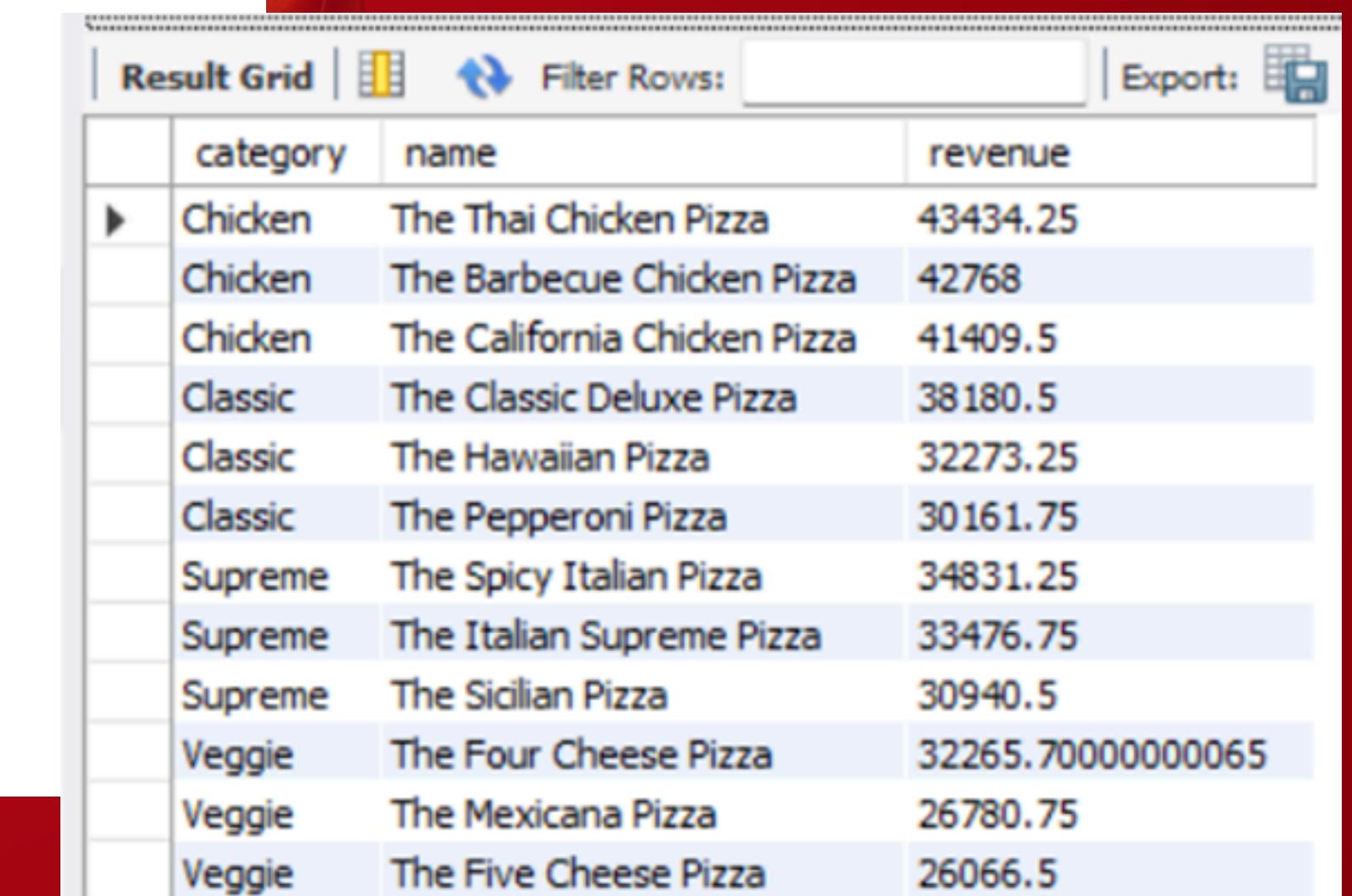
Q12. ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME

```
168    -- Analyze the cumulative revenue generated over time.  
169  
170 •  select  
171      order_date,  
172      sum(revenue) over (order by order_date  
173      ) as Cum_Revenue  
174  from  
175  (select  
176      orders.order_date,  
177      sum( order_details.quantity * pizzas.price  
178      ) as revenue  
179  from  
180      order_details  
181      join pizzas on order_details.pizza_id = pizzas.pizza_id  
182      join orders on orders.order_id = order_details.order_id  
183      group by  
184      orders.order_date  
185  ) as sales;
```

	order_date	Cum_Revenue
▶	2015-01-01	2713.8500000000004
	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	23990.350000000002
	2015-01-11	25862.65
	2015-01-12	27781.7

Q13. DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY

```
188    -- Determine the top 3 most ordered pizza types based on revenue for each pizza category.  
189  
190 • select category, name, revenue  
191   from  
192     (select category, name, revenue,  
193       rank() over (partition by category order by revenue desc  
194         ) as rn  
195   from  
196     (select pizza_types.category, pizza_types.name,  
197       sum(order_details.quantity * pizzas.price  
198         ) as revenue  
199       from  
200         pizzas  
201         join pizza_types on pizzas.pizza_type_id = pizza_types.pizza_type_id  
202         join order_details on order_details.pizza_id = pizzas.pizza_id  
203       group by pizza_types.category, pizza_types.name  
204         ) as a  
205     ) as b  
206   where rn <= 3;  
207
```



The screenshot shows a database query results grid titled "Result Grid". The grid has columns for "category", "name", and "revenue". The data is grouped by category, and for each category, the top 3 most ordered pizza types are listed. The results are as follows:

	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



BILLU'S PIZZA HUB

THANK YOU!

