

Analysis of Pizza Sales Data

Insights and Findings from SQL Queries

Purpose of the Analysis

The objective of this analysis is to gain comprehensive insights into pizza sales patterns, identify top-selling products, understand customer preferences, and provide actionable recommendations for business strategy. By leveraging SQL queries, the analysis aims to address several key business questions ranging from basic sales metrics to advanced revenue contributions and trends. The detailed analysis will help in making informed decisions to enhance sales strategies, inventory management, and marketing efforts.

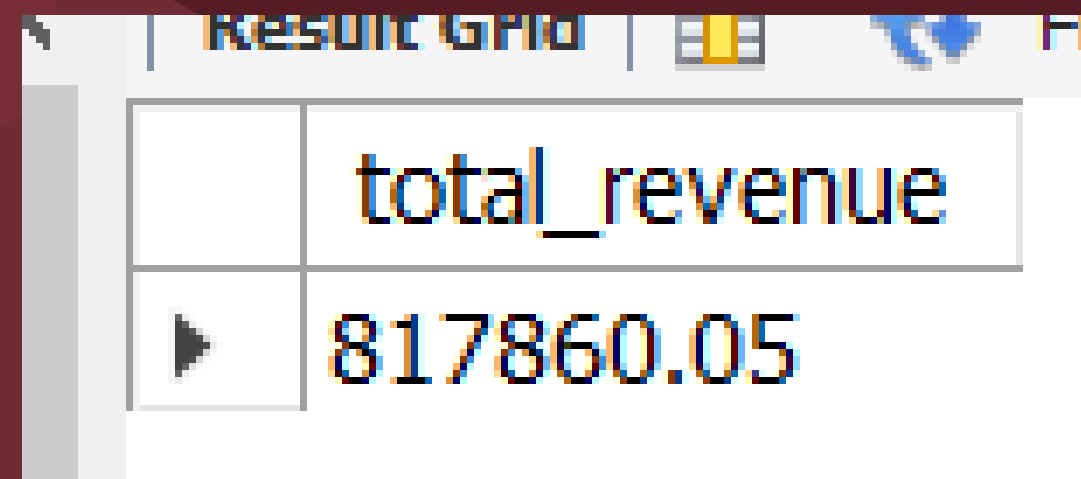
1. Retrieve the total number of orders placed.

```
select count(order_id) as total_orders from orders;
```

	total_orders
▶	21350

2. Calculate the total revenue generated from pizza sales.

```
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
          2) AS total_revenue
FROM
    order_details
    JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id
```



The screenshot shows a window titled "Result Grid" with a table containing one row of data. The column header is "total_revenue" and the value in the row is "817860.05".

	total_revenue
▶	817860.05

3. 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;
```

	name	price
▶	The Greek Pizza	35.95

4. Identify the most common pizza ordered.

```
SELECT
    pizza_types.name,
    COUNT(order_details.quantity) AS no_of_pizza_sales
FROM
    pizzas
    JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
    JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
GROUP BY pizza_types.name
ORDER BY no_of_pizza_sales DESC
LIMIT 1
```

	name	no_of_pizza_sales
▶	The Classic Deluxe Pizza	2416

5. Identify the most common pizza size ordered.

```
SELECT
    pizzas.size, COUNT(order_details.quantity) AS no_of_sales
FROM
    pizzas
    JOIN
        order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY no_of_sales DESC limit 1
```

	size	no_of_sales
▶	L	18526

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

```
SELECT
    pizza_types.name,
    sum(order_details.quantity) AS quantity
FROM
    pizzas
    JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
    JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC
LIMIT 5
```

	name	quantity
►	The Classic Deluxe Pizza	2453
	The Barbecue Chicken Pizza	2432
	The Hawaiian Pizza	2422
	The Pepperoni Pizza	2418

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

```
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS total_quantity
FROM
    pizzas
    JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
    JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category
ORDER BY total_quantity DESC
```

	category	total_quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

8. Determine the distribution of orders by hour of the day.

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

	hour	order_count
▶	11	1231
	12	2520
	13	2455
	14	1472
	15	1168

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

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

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

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

```
SELECT
    round(AVG(Quantity),0) AS Average_no_Pizza_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;
```

	Average_no_Pizza_per_day
▶	138

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

```
SELECT
    pizza_types.name,
    ROUND(SUM(pizzas.price * order_details.quantity),
          1) AS revenue
FROM
    pizzas
    JOIN
    pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
    JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3
```

	name	revenue
▶	The Barbecue Chicken Pa	43434.2
	The Barbecue Chicken Pizza	42768
	The California Chicken Pizza	41409.5

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

```
SELECT
    pizza_types.category,
    ROUND((SUM(order_details.quantity * pizzas.price) / (SELECT
        SUM(pizzas.price * order_details.quantity)
        FROM
            pizzas
            JOIN
                order_details ON pizzas.pizza_id = order_details.pizza_id))) * 100,
    2) AS revenue
FROM
    pizzas
    JOIN
        pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
    JOIN
        order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue DESC
```

	category	revenue
▶	Classic	26.91
	Supreme	Supreme
	Chicken	23.96
	Veggie	23.68

13. Analyze the cumulative revenue generated over time.

```
select order_date,  
sum(revenue) over(order by order_date) as cumm_revenue  
from  
(select orders.order_date, sum(order_details.quantity * pizzas.price) as revenue  
from order_details  
join pizzas on order_details.pizza_id = pizzas.pizza_id  
join orders on order_details.order_id = orders.order_id  
group by orders.order_date) as sales;
```

	order_date	cumm_revenue
►	2015-01-01	2713.850000000000004
	2015-01-02	5445.75
	2015-01-03	8108.15
	2015-01-04	9863.6
	2015-01-05	11929.55

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

```
select name, revenue
from
(select category, name, revenue,
rank() over(partition by category order by revenue desc) as rn
from
(select pizza_types.category, pizza_types.name,
sum(order_details.quantity * pizzas.price) as revenue
from pizzas
join pizza_types on pizzas.pizza_type_id = pizza_types.pizza_type_id
join order_details on pizzas.pizza_id = order_details.pizza_id
group by pizza_types.category, pizza_types.name) as a) as b
where rn <= 3
```

	category	name	revenue	rn
▶	Chicken	The Chicken Pesto Pizza	16701.75	1
	Chicken	The Chicken Alfredo Pizza	16900.25	2
	Chicken	The Southwest Chicken Pizza	34705.75	3
	Classic	The Pepperoni, Mushroom, and Pepp...	18834.5	1
	Classic	The Big Meat The Big Meat Pizza	22968	2
	Classic	The Meaty Meaty Pizza	24007	3

Result 10