

Pizza Sales Analysis

Using SQL



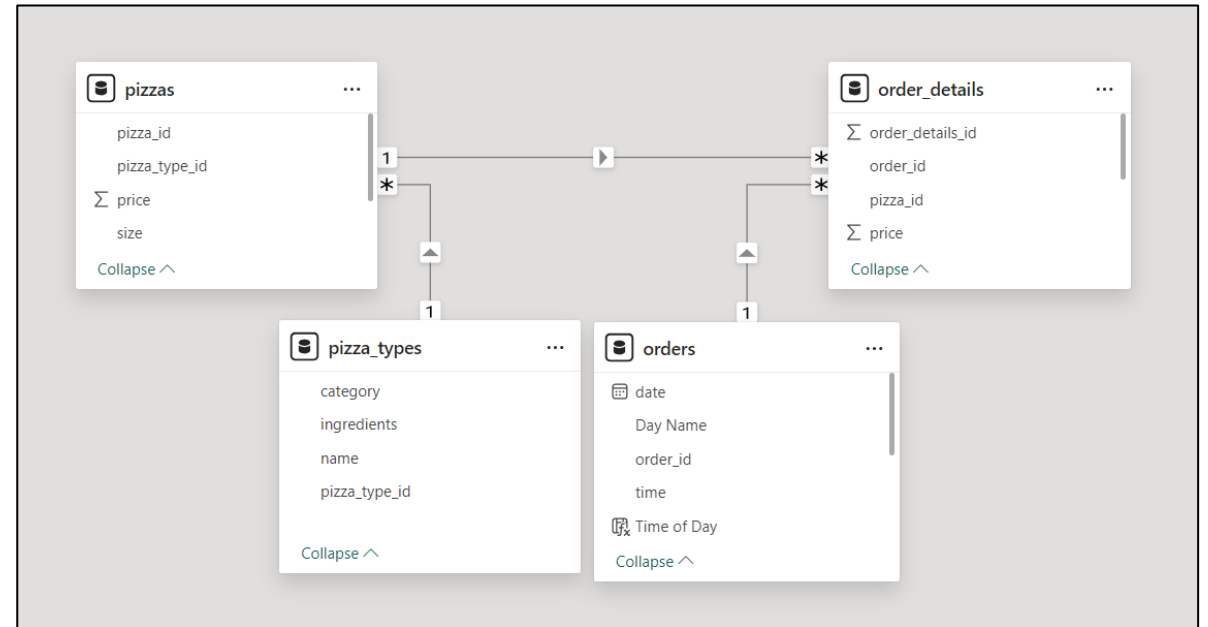
By Shaun Mia



Hello Everyone

My name is Shaun Mia, and I have developed a MySQL project in which I utilized MySQL Server Management Studio to analyze pizza sales data. This analysis is based on four datasets that provide valuable insights into various aspects of pizza sales performance. To uncover these insights, I addressed 13 key questions, which helped in understanding sales trends, customer behavior, and overall business performance.

Schema



All 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.

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.

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.



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

Retrieve the total number of orders placed.



Result Grid		Filter Rows:
	total_orders	
▶	21350	



Calculate the total revenue generated from pizza sales

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



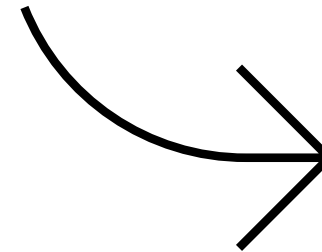
Result Grid			
	total_sales		
▶	817860.05		



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	



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;
```

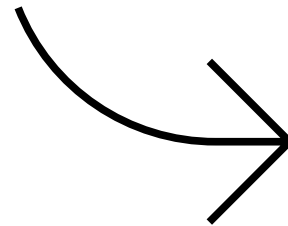


Result Grid			Filter Rows:
	size	order_count	
▶	L	18526	
	M	15385	
	S	14137	
	XL	544	
	XXL	28	



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

```
SELECT
    pizza_types.name, 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.name
ORDER BY quantity DESC
LIMIT 5;
```



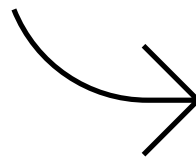
Result Grid			Filter Rows:
	name	quantity	
▶	The Classic Deluxe Pizza	2453	
	The Barbecue Chicken Pizza	2432	
	The Hawaiian Pizza	2422	
	The Pepperoni Pizza	2418	
	The Thai Chicken Pizza	2371	



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;
```

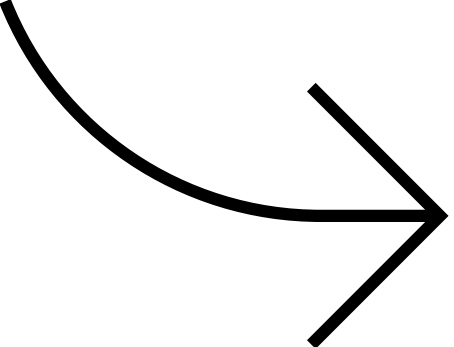


Result Grid		Filter Rows:
	category	quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050



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);
```

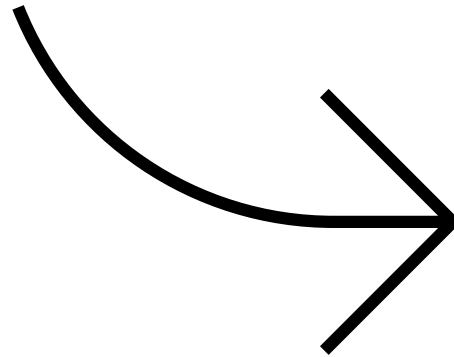


	hour	order_count
▶	11	1231
	12	2520
	13	2455
	14	1472
	15	1468
	16	1920
	17	2336
	18	2399
	19	2009
	20	1642
	21	1198
	22	663
	23	28
	10	8
	9	1



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

```
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	



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

```
SELECT
    ROUND(AVG(quantity), 0) as avg_pizza_ordered_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;
```

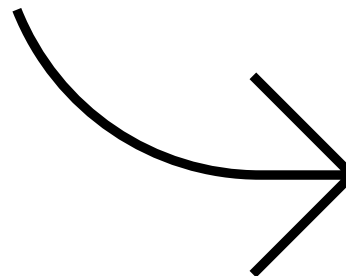


Result Grid		Filter Rows:
	round(avg(quantity),0)	
▶	138	



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

```
SELECT
    pizza_types.name,
    ROUND(SUM(order_details.quantity * pizzas.price),
          0) 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.name
ORDER BY revenue DESC
LIMIT 3;
```



Result Grid			Filter Rows:
	name	revenue	
▶	The Thai Chicken Pizza	43434	
	The Barbecue Chicken Pizza	42768	
	The California Chicken Pizza	41410	



Calculate the percentage contribution of each pizza type to total revenue



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


Result Grid			Filter Rows
	category	revenue	
▶	Classic	26.91	
	Supreme	25.46	
	Chicken	23.96	
	Veggie	23.68	





Analyze the cumulative revenue generated over time

```
select order_date, sum(revenue) over( order by order_date) as cum_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 orders.order_id = order_details.order_id
group by orders.order_date) as sales ;
```

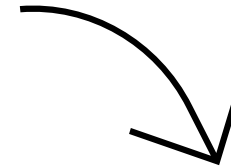


Result Grid			Filter Rows:
	order_date	cum_revenue	
▶	2015-01-01	2713.8500000000004	
	2015-01-02	5445.75	
	2015-01-03	8108.15	
	2015-01-04	9863.6	
	2015-01-05	11929.55	
	2015-01-06	14358.5	
	2015-01-07	16560.7	
	2015-01-08	19399.05	
	2015-01-09	21526.4	
	2015-01-10	23990.350000000002	
	2015-01-11	25862.65	
	2015-01-12	27781.7	
	2015-01-13	29831.300000000003	
	2015-01-14	32358.700000000004	
	2015-01-15	34343.50000000001	



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 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, pizza_types.name) as a) as b
where rn <=3;
```



Result Grid			Filter Rows:
	name	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 Hawaiian Pizza	32273.25	
	The Pepperoni Pizza	30161.75	
	The Spicy Italian Pizza	34831.25	
	The Italian Supreme Pizza	33476.75	
	The Sicilian Pizza	30940.5	
	The Four Cheese Pizza	32265.70000000065	
	The Mexicana Pizza	26780.75	
	The Five Cheese Pizza	26066.5	

Thank
You

