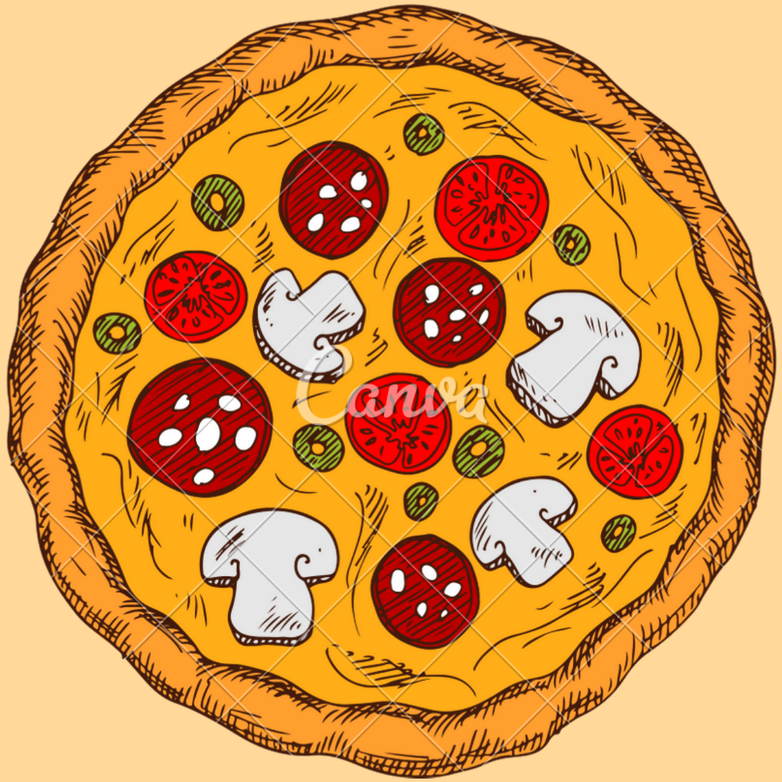
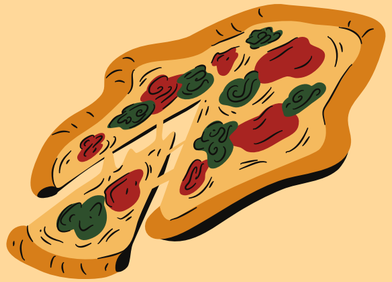


Delicious Pizza for Everyone!

PIZZA SELES ANALYSIS

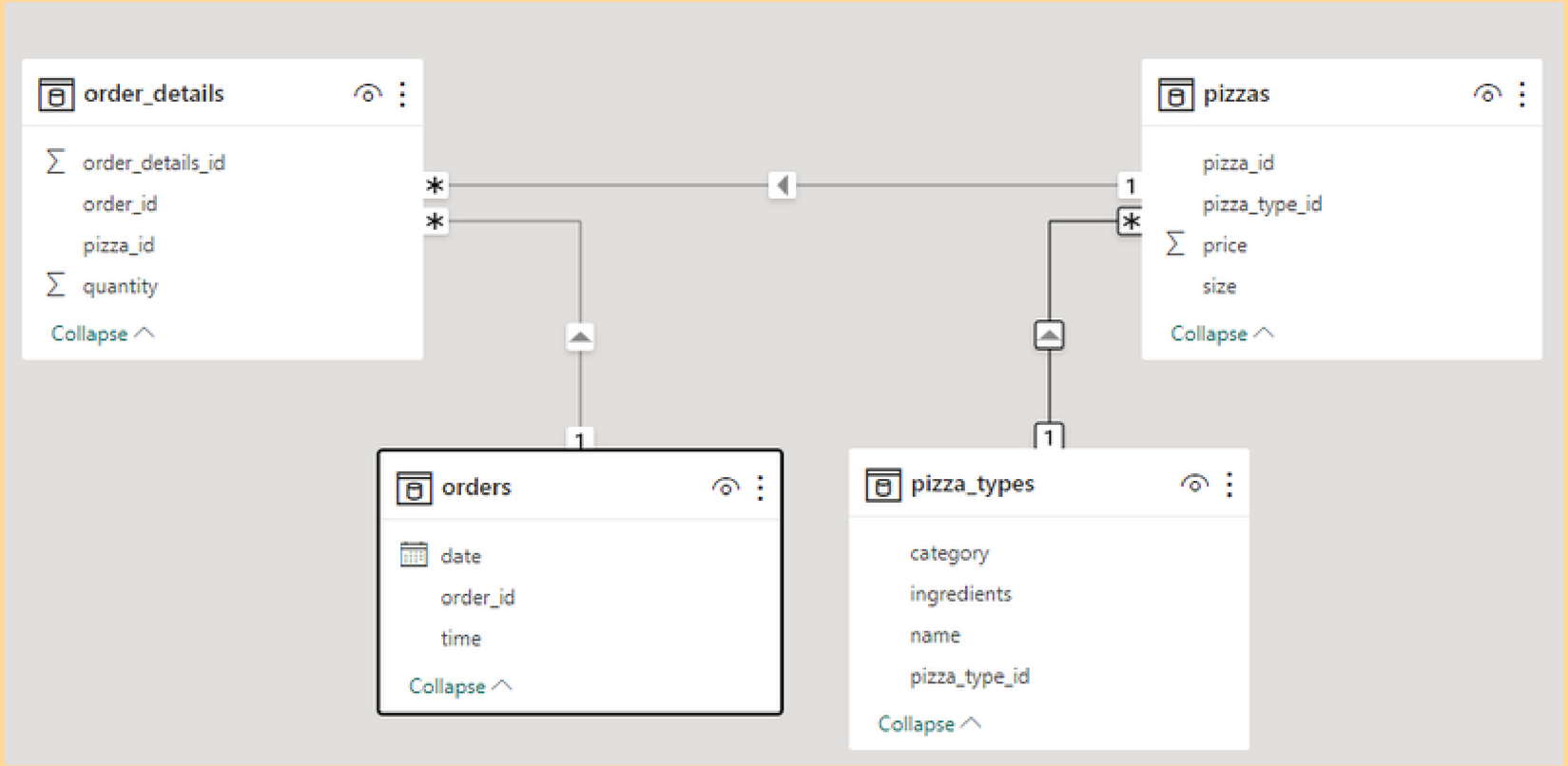
Linkedin



Hello!

My Name Is Shreejay and in this project i have utilize SQL quarry and Solve Questions that where related to Pizza Sales

Data Module


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

Q1. Retrieve the total number of orders placed.

```
2    SELECT
3        COUNT(order_id) AS total_orders
4    FROM
5        orders;
```

OUT PUT :

	total_orders
▶	21350

Q2. Calculate the total revenue generated from pizza sales.

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

OUT PUT :

	Total_sales
▶	817860.05

Q3. Identify the highest-priced pizza.

```sql
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;
```

OUT PUT :

	name	price
▶	The Greek Pizza	35.95

Q4. Identify the most common pizza size ordered.

```
SELECT
    pizzas.size,
    COUNT(orders_details.order_details_id) AS Order_Count
FROM
    pizzas
        JOIN
    orders_details ON pizzas.pizza_id = orders_details.pizza_id
GROUP BY pizzas.size
ORDER BY Order_Count DESC;
```

OUT PUT :

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.

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

OUT PUT :

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.

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

OUT PUT :

	category	quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

Q7. 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);
```

OUT PUT :

Hour	Order_Count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642

Q8. Join relevant tables to find the category-wise distribution of pizzas.

```
select category , count(name)
from pizza_types
group by category;
```

OUT PUT :

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

6

Q9. Join relevant tables to find the category-wise distribution of pizzas.

```
select round(avg(quantity),0) from
(select orders.oeder_date, sum(orders_details.quantity) as quantity
from orders join orders_details on
orders.order_id = orders_details.oeder_id
group by orders.oeder_date)as order_quantity;
```

OUT PUT :

	round(avg(quantity),0)
▶	138

Q10. Determine the top 3 most ordered pizza types based on revenue.

```
select pizza_types.name ,
sum(orders_details.quantity * pizzas.price )as revenue
from pizza_types join pizzas on
 pizza_types.pizza_type_id = pizzas.pizza_type_id
 join orders_details on
 orders_details.pizza_id = pizzas.pizza_id
 group by pizza_types.name
 order by revenue desc limit 3;
```

OUT PUT :

	name	revenue
▶	The Thai Chicken Pizza	43434.25
	The Barbecue Chicken Pizza	42768
	The Califo	

The California Chicken Pizza

Q11. Calculate the percentage contribution of each pizza type to total revenue.

```
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 on
pizzas.pizza_id = orders_details.pizza_id) )*100,2 )as revenue

from pizza_types join pizzas on
pizza_types.pizza_type_id = pizzas.pizza_type_id
join orders_details on
orders_details.pizza_id = pizzas.pizza_id
group by pizza_types.category order by revenue desc;
```

OUT PUT :

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

Q12. Analyze the cumulative revenue generated over time.

```
select oeder_date,
sum(revenue) over (order by oeder_date )
as cum_revenue
from

(select orders.oeder_date,
sum(orders_details.quantity * pizzas.price) as revenue
from orders_details join pizzas
on orders_details.pizza_id = pizzas.pizza_id
join orders
on orders.order_id = orders_details.oeder_id
group by orders.oeder_date) as sales ;
```

OUT PUT :

	oeder_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