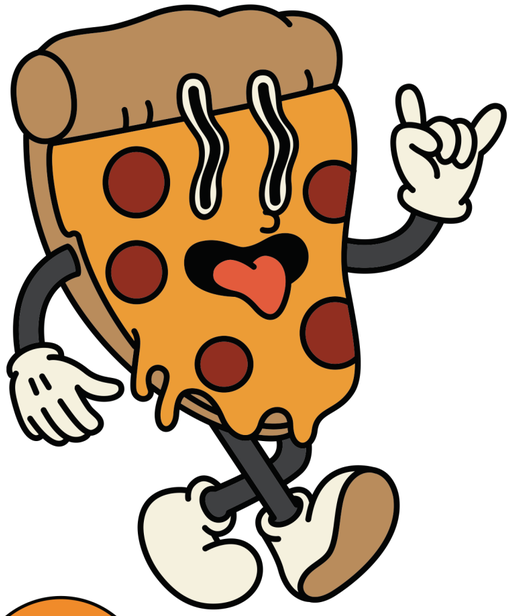
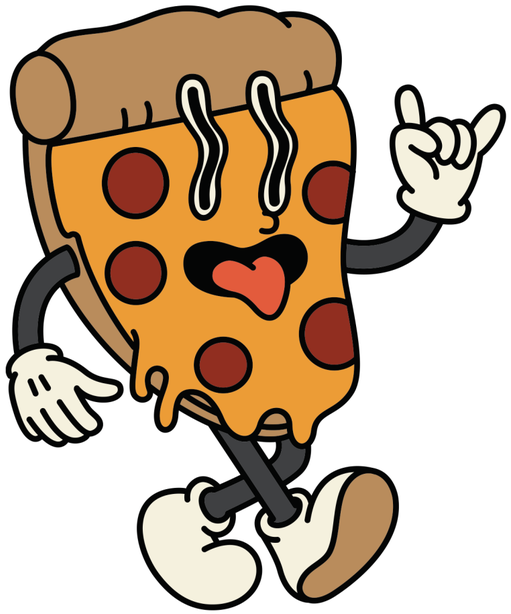
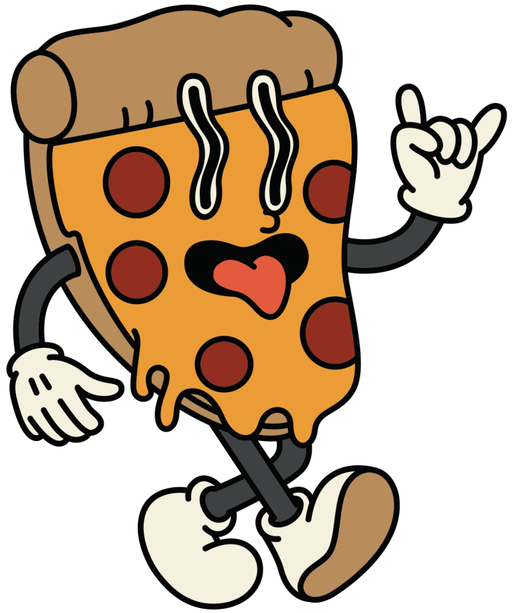
SQL PROJECT ON PIZZA SALES

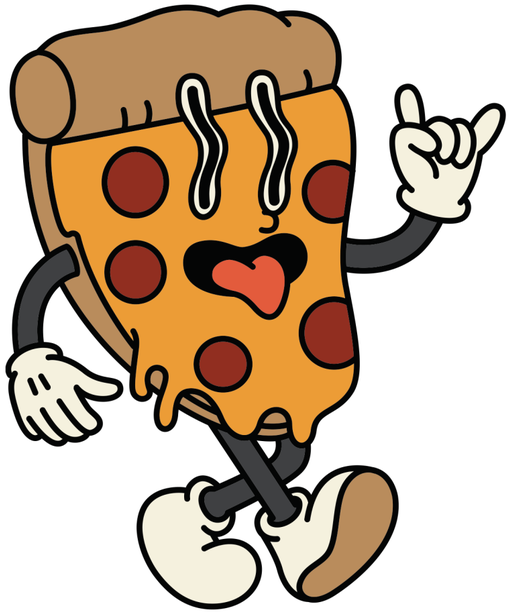
Hello, I'm Nikhil Kumar

I leverage SQL queries to analyze and resolve complex questions related to pizza sales. My expertise lies in utilizing advanced SQL techniques to extract valuable insights









N

TO GAIN DEEPER INSIGHTS, WE SHOULD THOROUGHLY ANALYZE THE DETAILED PIZZA SALES DATA

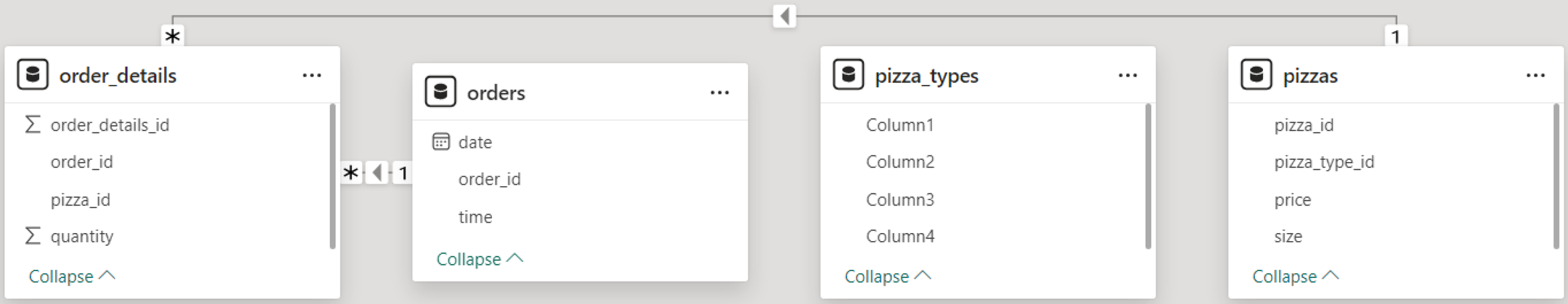
```
Basic:
Retrieve the total number of orders placed.
Calculate the total revenue generated from pizza sales.
Identify the highest-priced pizza.
Identify the most common pizza size ordered.
List the top 5 most ordered pizza types along with their quantities.


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.
```

N

schema


*
1
order_details
order_details_id
order_id
pizza_id
quantity
Collapse
*
1
orders
date
order_id
time
Collapse
pizza_types
Column1
Column2
Column3
Column4
Collapse
1
pizzas
pizza_id
pizza_type_id
price
size
Collapse

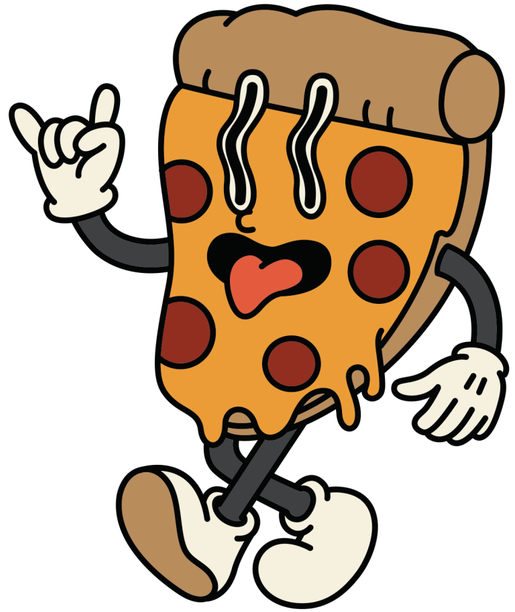
N

Retrieve the total number of orders placed.

```sql
SELECT
    COUNT(order_id) AS total_orders_placed
FROM
    orders;
```

Result Grid | Filter

	total_orders_placed
▶	21350



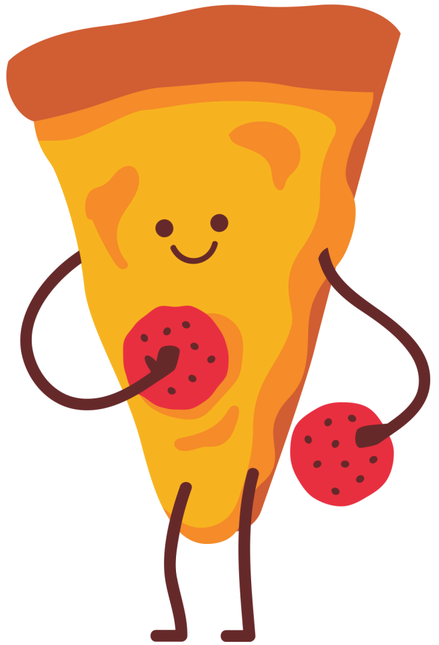
N

Calculate the total revenue generated from pizza sales.

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

Result Grid | Filter Rows:

total_revenue
817860.05



N

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;
```

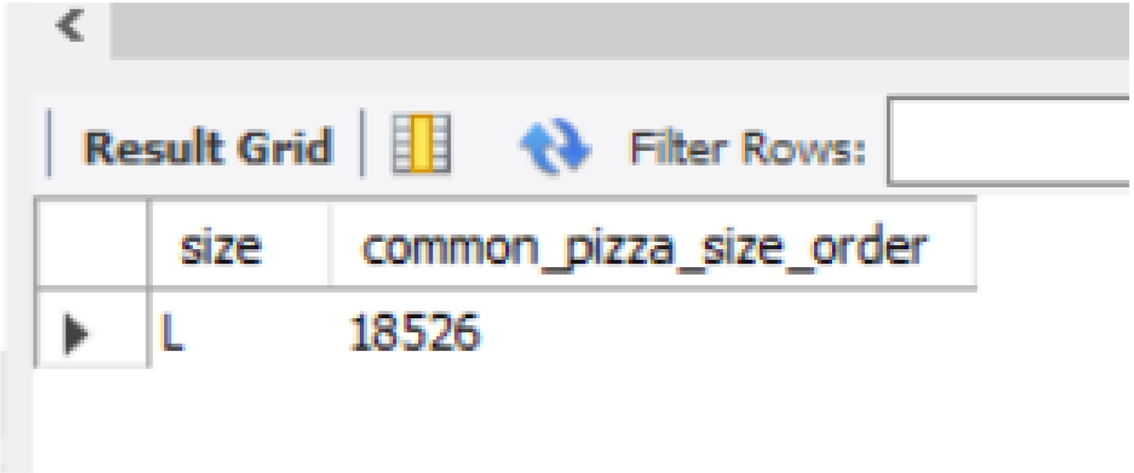
Result Grid | Filter Rows:

name	price
The Greek Pizza	35.95

N

Identify the most common pizza size ordered.

```sql
SELECT
    pizzas.size, COUNT(orders_details.order_details_id) as common_pizza_size_order
FROM
    pizzas
        JOIN
    orders_details ON pizzas.pizza_id = orders_details.pizza_id
GROUP BY pizzas.size
ORDER BY common_pizza_size_order  DESC
LIMIT 1;
```



size	common_pizza_size_order
L	18526

N

DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

```
SELECT
    HOUR(orders.order_time) AS hour_,
    COUNT(orders.order_id) AS orders_count
FROM
    orders
GROUP BY hour_;
```

Result Grid | Filter Rows:

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

N

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;
```

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

N

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

```sql
SELECT
    pizza_types.category,
    SUM(orders_details.quantity) AS total_quatity
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 total_quatity DESC;
```

Result Grid | Filter Rows:

category	total_quatity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

N

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

```sql
SELECT
    category, COUNT(name) as pizza_count
FROM
    pizza_types
GROUP BY category;
```

Result Grid | Filter Rows:

category	pizza_count
Chicken	6
Classic	8
Supreme	9
Veggie	9

N

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

```sql
SELECT
    ROUND(AVG(quantity), 0) AS avg_pizza_order_per_day
FROM
    (SELECT
        SUM(orders_details.quantity) AS quantity,
            orders.order_date AS order_date
    FROM
        orders_details
    JOIN orders ON orders_details.oredr_id = orders.order_id
    GROUP BY order_date) AS order_quantity;
```

Result Grid | Filter Rows:

avg_pizza_order_per_day
138

N

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

```sql
SELECT
    pizza_types.name AS pizza_name,
    SUM(pizzas.price * orders_details.quantity) 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_name
ORDER BY revenue DESC
LIMIT 3;
```

Result Grid | Filter Rows:

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

N

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

```
SELECT
    pizza_types.category,
    round(SUM(pizzas.price * orders_details.quantity) / (SELECT
            ROUND(SUM(pizzas.price * orders_details.quantity),
                        2)
        FROM
            orders_details
                JOIN
            pizzas ON orders_details.pizza_id = pizzas.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;
```

Result Grid | Filter Rows

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

N

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

```sql
select order_date , sum(revenue) over (order by order_date) as cum_revenue
from
(select orders.order_date , round(sum(pizzas.price * orders_details.quantity) ,0)as revenue
from orders join orders_details
on orders.order_id = orders_details.oredr_id
join pizzas
on orders_details.pizza_id = pizzas.pizza_id
group by orders.order_date order by revenue desc) as sales;
```

Result Grid | Filter Rows:

order_date	cum_revenue
2015-01-01	2714
2015-01-02	5446
2015-01-03	8108
2015-01-04	9863
2015-01-05	11929
2015-01-06	14358
2015-01-07	16560
2015-01-08	19398
2015-01-09	21525
2015-01-10	23989
2015-01-11	25861
2015-01-12	27780
2015-01-13	29830
2015-01-14	32357
2015-01-15	34342
2015-01-16	36936

N

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

```sql
select name,revenue from
(select category , name , revenue,
rank() over(partition by category order by revenue desc) as rank_
from
(select pizza_types.category , 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.category,pizza_types.name) as a) as b
where rank_ <=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

N

THANK YOU