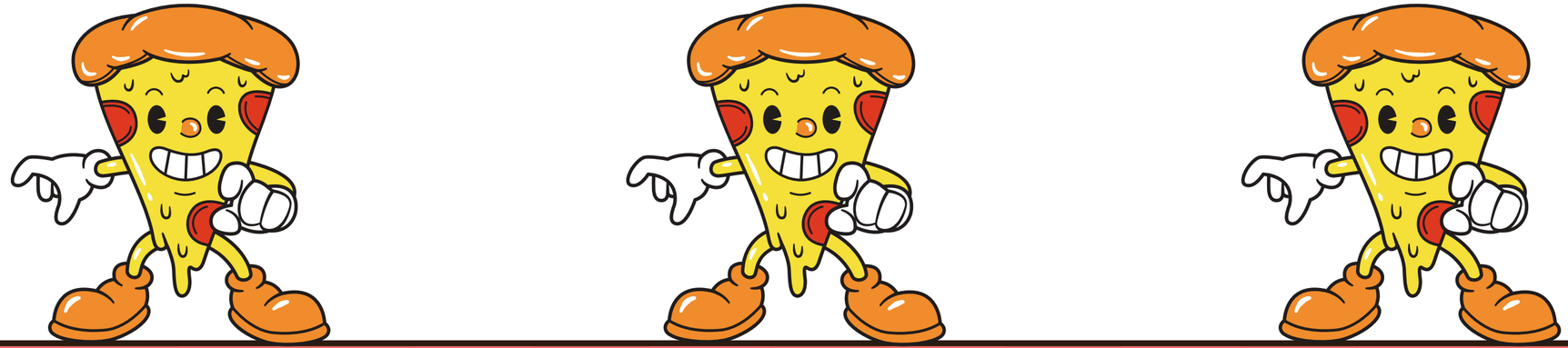
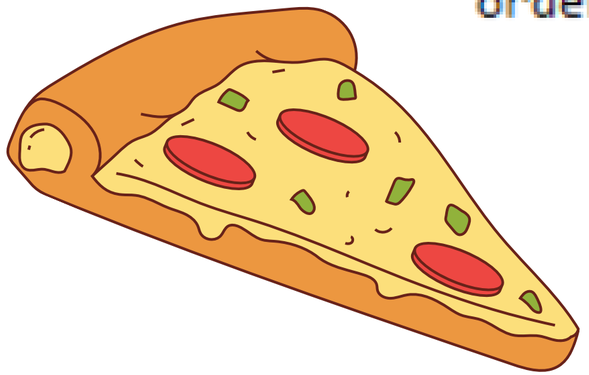
PIZZA PARTY!



A MySQL project analyzing pizza sales data to gain insights into business performance.

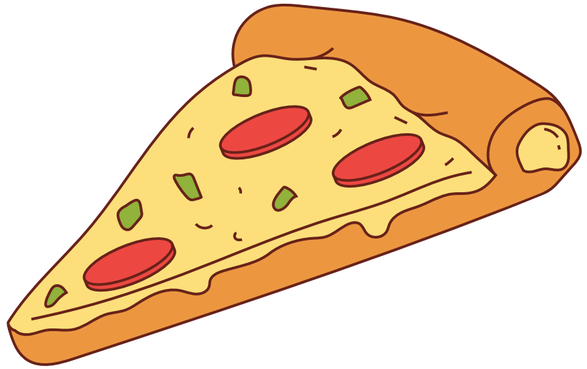
Retrieve total number of orders placed

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



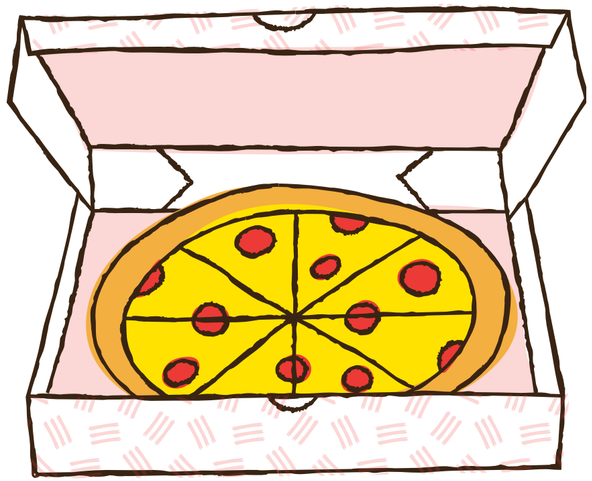
Result Grid | Filte

	Total_Orders_Placed
▶	21350



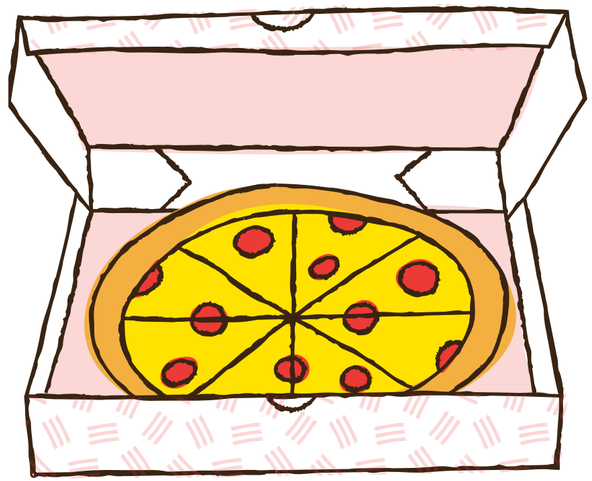
This pizza is amazing!

Calculate total revenue generated from pizza sales



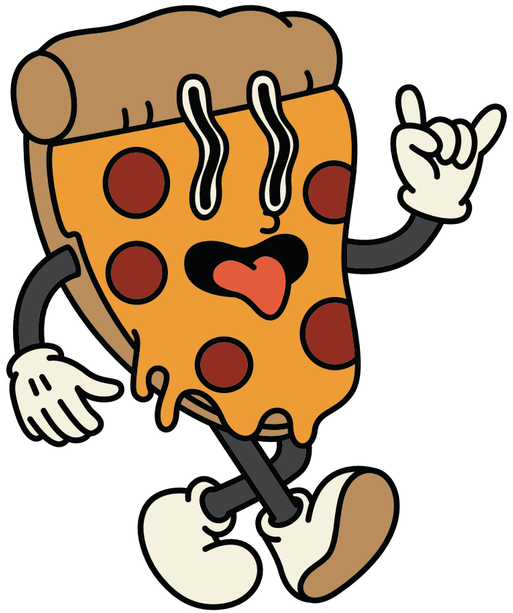
```
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS Total_Sales
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id
```

Total_Sales
817860.05



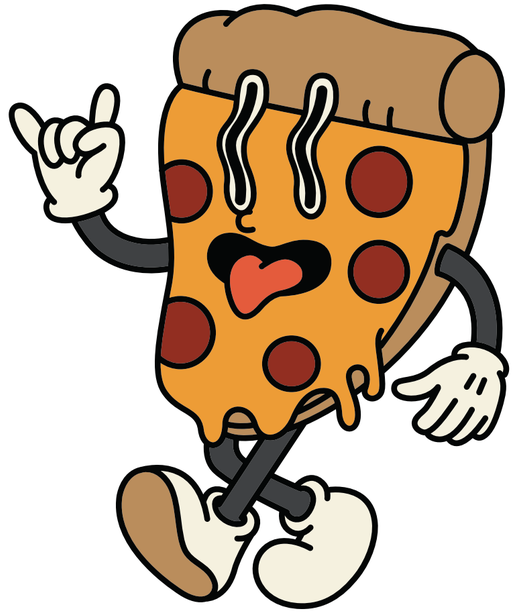
Thank you for the pizza.

Identify highest priced pizza



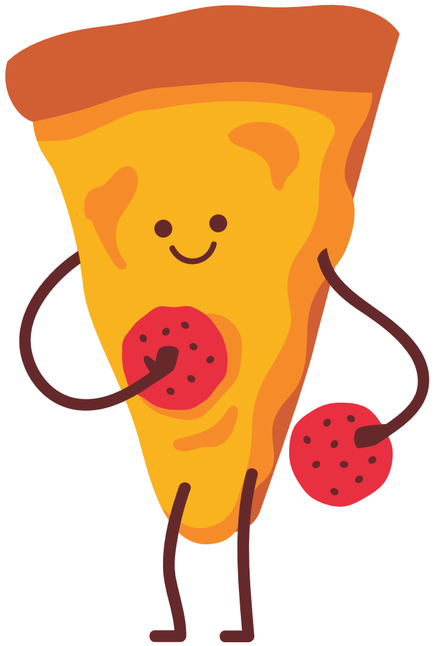
```sql
SELECT
    pizza_types.name as Name, pizzas.price as Price
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```

Name	Price
The Greek Pizza	35.95



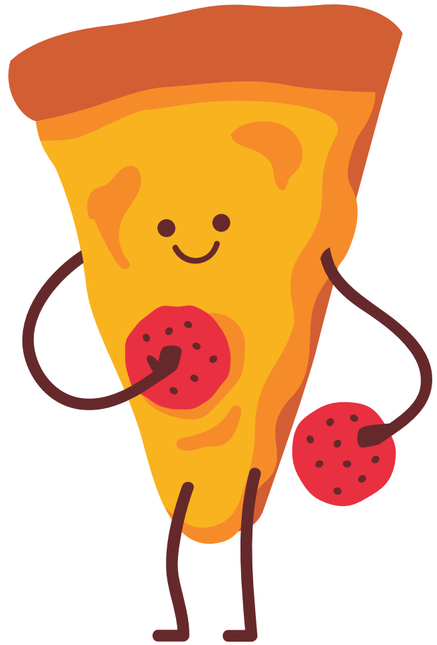
Pizza Palace is where I buy pizza.

Identify the most common pizza size ordered



```sql
SELECT
    pizzas.size as 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
LIMIT 1;
```

Size	Order_Count
L	18526



Would you like some pizza?

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

```sql
SELECT
    pizza_types.name AS 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;
```

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

I need cheese, tomato and olives.

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

```sql
SELECT
    pizza_types.category as 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;
```

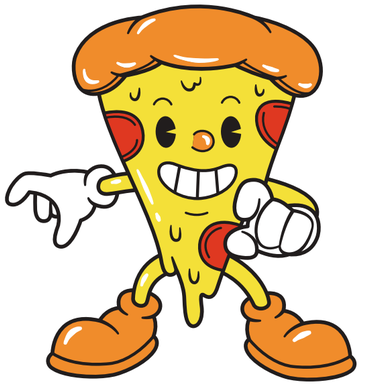
Category	Quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

"This pizza is mine," said Dad.

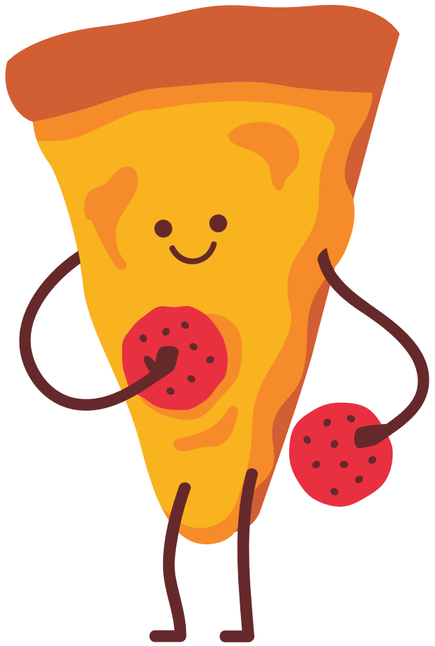
Determine the distribution of orders by hour of the day

```sql
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

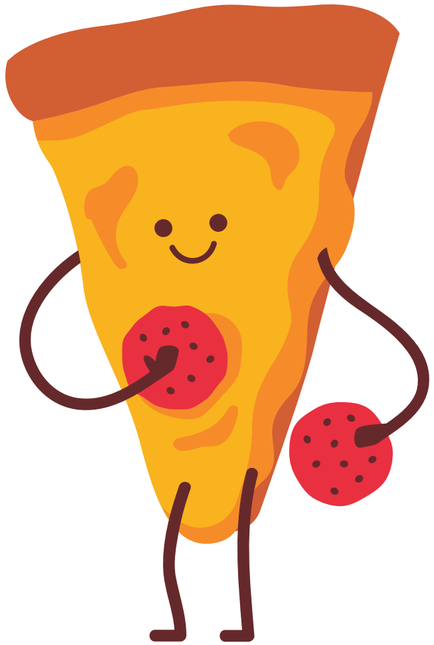


Join relevant tables to find the category-wise distribution of pizzas



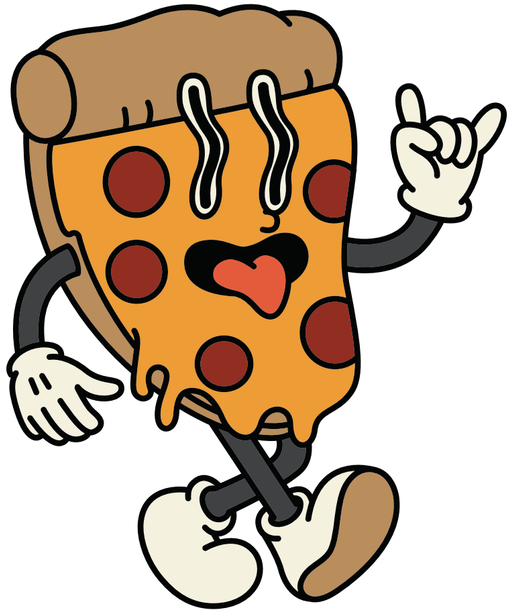
```
SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```

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



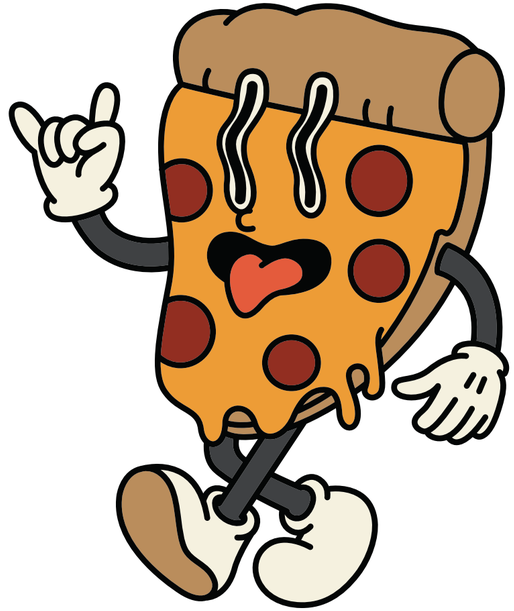
Would you like some pizza?

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



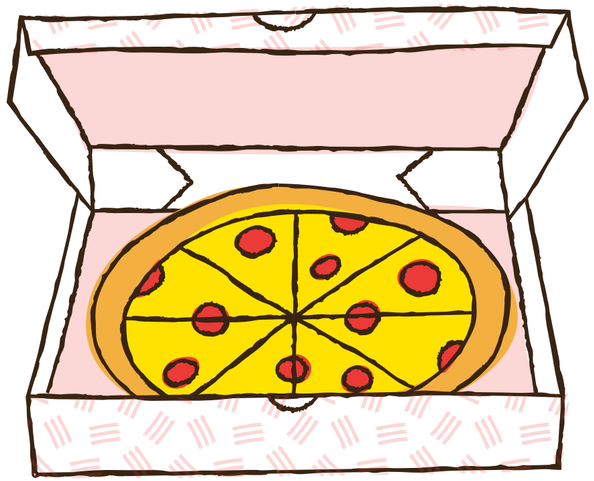
```sql
SELECT
    ROUND(AVG(quantity), 0) as Avg_Pizzas_Ordered
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS Quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS Order_Quantity;
```

Avg_Pizzas_Ordered
138



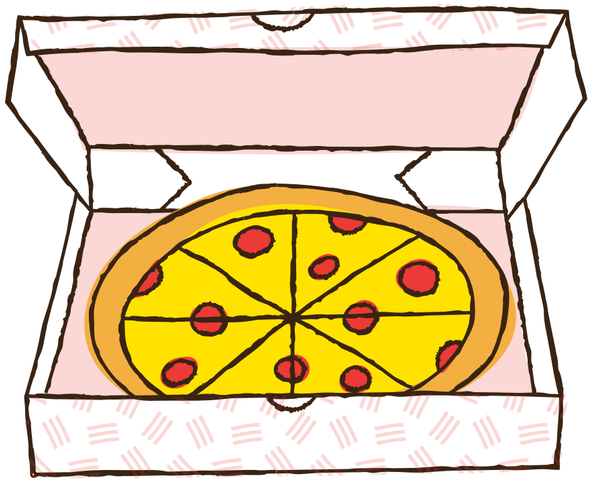
Pizza Palace is where I buy pizza.

Determine top 3 most ordered pizza types based on revenue



```sql
SELECT
    pizza_types.name as Name,
    SUM(order_details.quantity * pizzas.price) 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;
```

Name	Revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5



Thank you for the pizza.

Calculate the percentage contribution of each pizza type to total revenue

```sql
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;
```

category	Revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

"This pizza is mine," said Dad.

Analyze cumulative revenue generated over time

```sql
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;
```

Order_Date	Cumulative_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
2015-01-16	36937.65000000001
2015-01-17	39001.75000000001
2015-01-18	40978.600000000006

Order_Date	Cumulative_revenue
2015-01-19	43365.75000000001
2015-01-20	45763.65000000001
2015-01-21	47804.20000000001
2015-01-22	50300.90000000001
2015-01-23	52724.600000000006
2015-01-24	55013.850000000006
2015-01-25	56631.40000000001
2015-01-26	58515.80000000001
2015-01-27	61043.85000000001
2015-01-28	63059.85000000001
2015-01-29	65105.150000000016
2015-01-30	67375.45000000001
2015-01-31	69793.30000000002
2015-02-01	72982.50000000001
2015-02-02	75311.10000000002
2015-02-03	77925.90000000002
2015-02-04	80159.80000000002
2015-02-05	82375.60000000002

Order_Date	Cumulative_revenue
2015-02-06	84885.55000000002
2015-02-07	87123.20000000001
2015-02-08	89158.20000000001
2015-02-09	91353.55000000002
2015-02-10	93410.05000000002
2015-02-11	95870.05000000002
2015-02-12	98028.85000000002
2015-02-13	100783.35000000002
2015-02-14	103102.50000000001
2015-02-15	105243.75000000001
2015-02-16	107212.55000000002
2015-02-17	109334.45000000001
2015-02-18	111977.30000000002
2015-02-19	114007.55000000002
2015-02-20	116898.70000000001
2015-02-21	119009.70000000001
2015-02-22	120589.65000000001
2015-02-23	122758.20000000001

I need cheese, tomato and olives.

Determine 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 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;
```

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