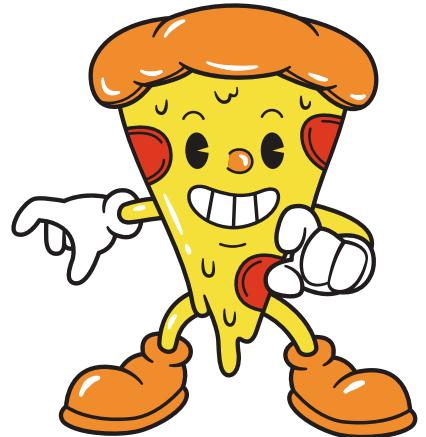
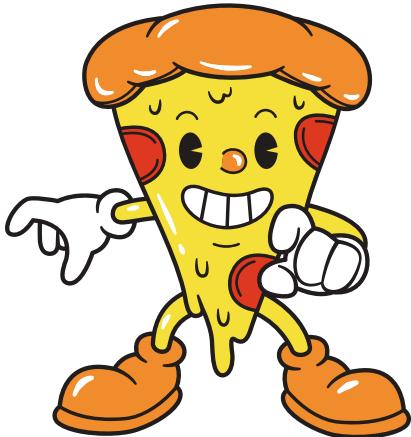


PIZZA SALES REPORT - SQL



SNEHAL DANGE

QUESTIONNAIR

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.

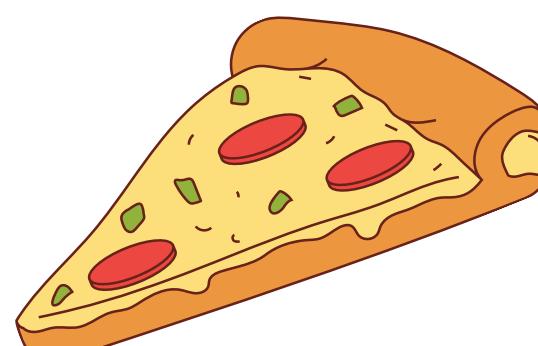
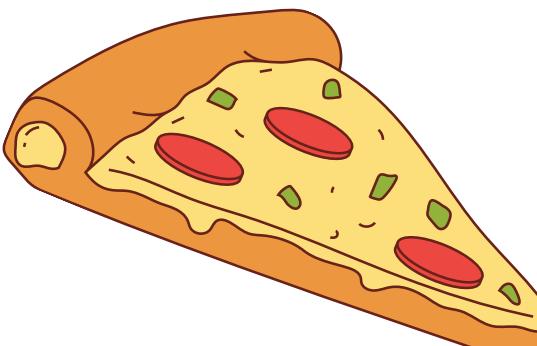
Analyze the cumulative revenue generated over time.

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



Retrieve the total number of orders placed.

```
SELECT  
    COUNT(order_id) AS total_orders  
FROM  
    orders;
```



	total_orders
▶	21350

Calculate the total revenue generated from pizza sales.

```
SELECT
```

```
    ROUND(SUM(order_details.quantity * pizzas.price),
```

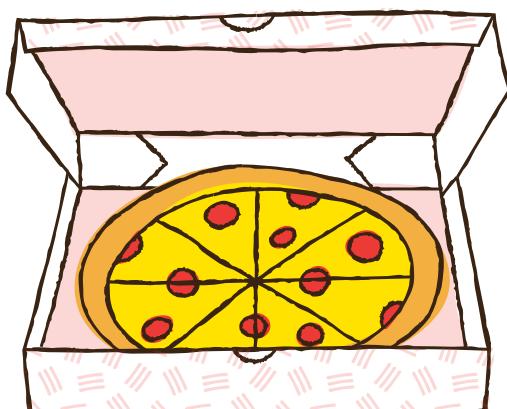
```
        2) AS total_sales
```

```
FROM
```

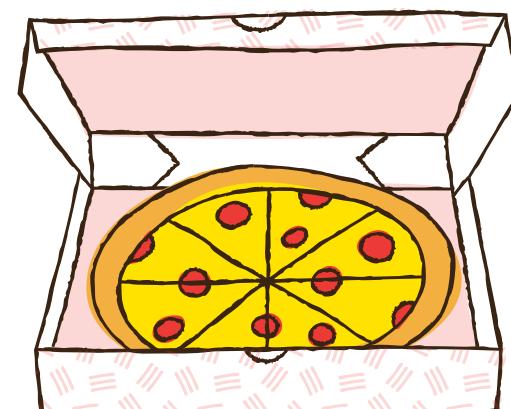
```
order_details
```

```
JOIN
```

```
pizzas ON pizzas.pizza_id = order_details.pizza_id;
```



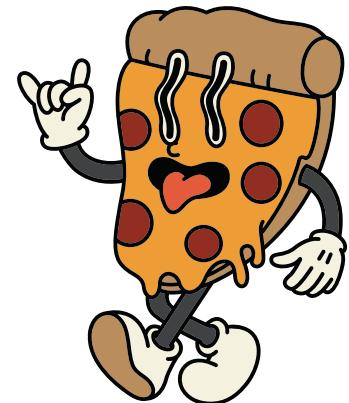
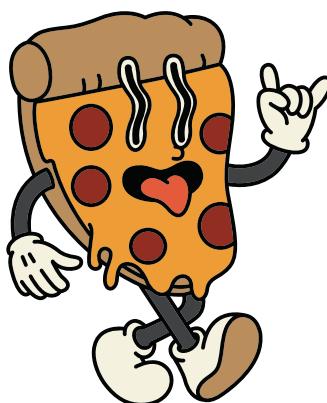
	total_sales
▶	817860.05



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;
```

	name	price
▶	The Greek Pizza	35.95



Identify the most common pizza size ordered.

SELECT

pizzas.size, COUNT(order_details.order_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

LIMIT 1;

	size	order_count
▶	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;



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;
```

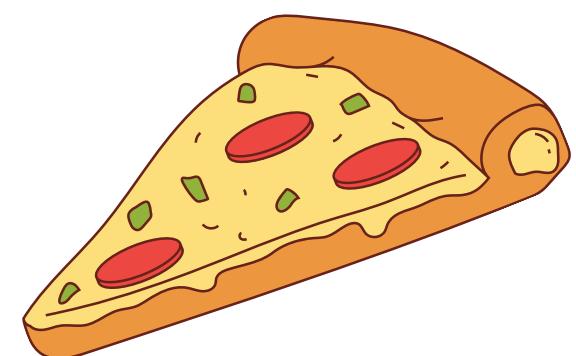
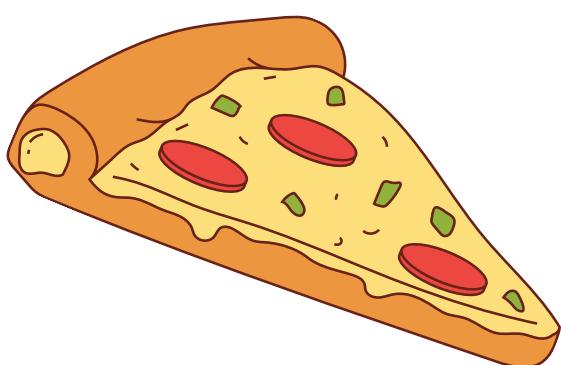
category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050



Determine the distribution of orders by hour of the day.

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

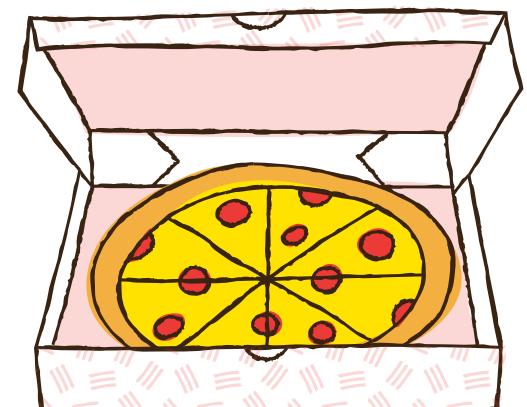
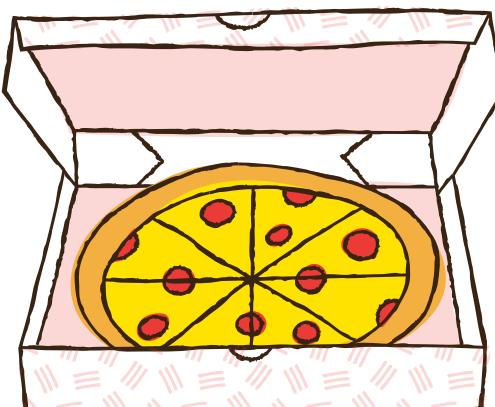
hour	COUNT(order_id)
11	2
11	1
12	1
12	3
12	1
12	1
12	1
12	1



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

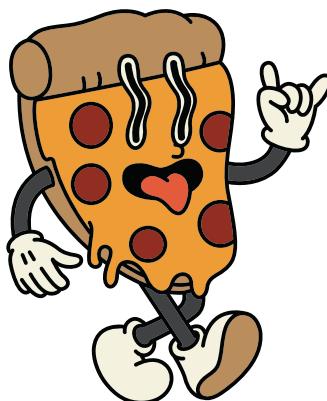
```
SELECT  
    category, COUNT(name)  
FROM  
    pizza_types  
GROUP BY category;
```

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

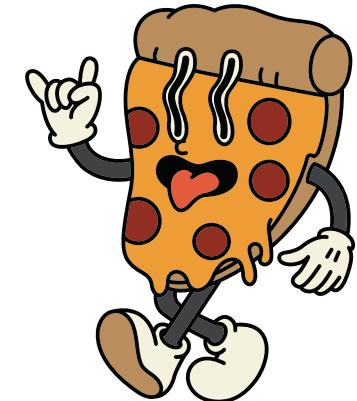


Group the orders by date and calculate the average number of pizzas ordered per day.

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



avg_number_pizzas
138



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

```
SELECT
```

```
    pizza_types.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 pizzas.pizza_id = order_details.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



Analyze the cumulative revenue generated over time.

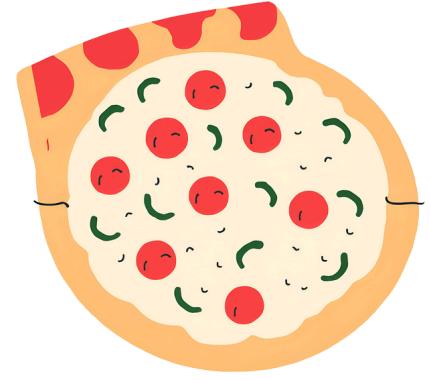
```
select order_date,  
       sum(revenue) over(order by order_date) as cum_revenue  
  from  
    (select orders.order_date, sum(order_details.quantity * pizzas.price) as revenue  
     from order_details join pizzas on order_details.pizza_id = pizzas.pizza_id  
                      join orders on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

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



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

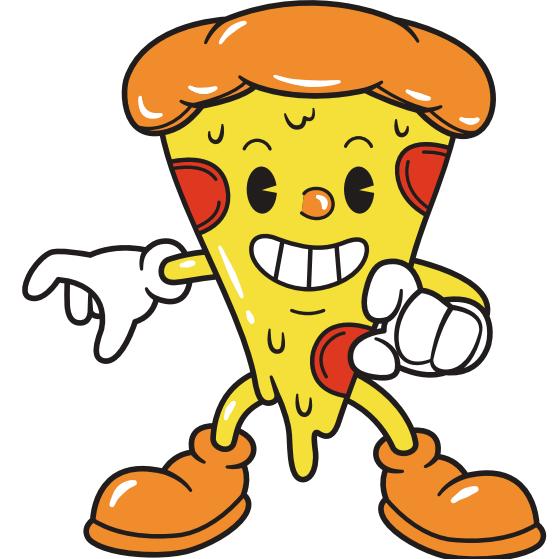
```
select name, revenue from
(select category, name, revenue,
rank() over(partition by category order by revenue desc ) as rn
from
(select pizza_types.category, pizza_types.name,
sum(order_details.quantity * pizzas.price) as revenue
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
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;
```



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

SQL functions used in this project

- **Joins**
- **Aggregate**
- **Group By**
- **Subqueries**
- **Window**



THANK YOU!

