

```
-- Determine the distribution of orders by hour of the day.

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

sult Gri	d 📗 🥠	Filter Rows:	Export:	Wrap Cell Content:	I
hour	orders				
11	1231				
12	2520				
13	2455				
14	1472				
15	1468				
16	1920				
17	2336				
18	2399				
140	mann				

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

SELECT

ROUND(AVG(quantity), 2) as Average_pizza_ordered_perday

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;
```



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

SELECT
category, COUNT(pizza_type_id) AS Distributed_pizzas

FROM
pizza_types
ROUP BY category
```

1 -			In the law of the second
100	esult Grid		Export: Wrap Cell Content: IA
	category Distributed_pizzas		
	Chicken	6	
	Classic	8	
	Supreme	9	

Veggie

```
-- Determine the top 3 most ordered pizza types based on revenue for each pizza category.
 2
 3
4 .
       select category, name, revenue , rank_r
 5
       from
    (select category,name,revenue,rank() over(partition by category order by revenue desc) as rank_r
 7
   (select pizza_types.category, pizza_types.name, sum(order_details.quantity * pizzas.price) as revenue
      from order_details
9
10
      join pizzas
11
       on pizzas.pizza_id = order_details.pizza_id
12
      join pizza_types
13
       on pizza_types.pizza_type_id = pizzas.pizza_type_id
      group by pizza_types.category, pizza_types.name) as a) as b
14
15
       where rank_r <= 3
16
```

- 1			
Result Grid	Fiter Rows:	Expo	ort: Wrap O
catego	ry name	revenue	rank_r
Chicken	The Barbecue Chicken Pizza	22756.25	1
Chicken	The Thai Chicken Pizza	21638.5	2
Chicken	The California Chicken Pizza	21303.25	3
Classic	The Classic Deluxe Pizza	19062.5	1
Classic	The Hawaiian Pizza	16354.75	2
Classic	The Pepperoni Pizza	15401.5	3

```
2
         SELECT
  3.
             pizza_types.name, pizzas.price
  4
         FROM
  5
  6
             pizzas
  7.
                 JOIN
             pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
 8
        ORDER BY pizzas.price DESC
  9
        LIMIT 1
 10
 11
                                          Export: Wrap Cell Content: IA | Fetch rows:
                                                                                       110
Result Grid !!! (*) Filter Rows:
   name
                 price
  The Greek Pizza
                 35.95
```

-- Identify the highest-priced pizza.

```
-- Determine the top 3 most ordered pizza types based on revenue for each pizza category.
1
2
       select category, name, revenue , rank_r
5
    (select category,name,revenue,rank() over(partition by category order by revenue desc) as rank_r
6
       from
    (select pizza_types.category, pizza_types.name, sum(order_details.quantity * pizzas.price) as revenue
8
9
       from order_details
       join pizzas
10
       on pizzas.pizza_id = order_details.pizza_id
11
      join pizza_types
12
       on pizza_types.pizza_type_id = pizzas.pizza_type_id
13
       group by pizza_types.category, pizza_types.name) as a) as b
14
       where rank_r <= 3
15
16
```

Fiter Rows:	Ехро	orti Wrap C	Content: 17
name	revenue	rank_r	
The Barbecue Chicken Pizza	22756.25	1	
The Thai Chicken Pizza	21638.5	2	
The California Chicken Pizza	21303.25	3	
The Classic Deluxe Pizza	19062.5	1	
The Hawaiian Pizza	16354.75	2	
The Pepperoni Pizza	15401.5	3	
	name The Barbecue Chicken Pizza The Thai Chicken Pizza The California Chicken Pizza The Classic Deluxe Pizza The Hawaiian Pizza	name revenue The Barbecue Chicken Pizza 22756.25 The Thai Chicken Pizza 21638.5 The California Chicken Pizza 21303.25 The Classic Deluxe Pizza 19062.5 The Hawaiian Pizza 16354.75	name revenue rank_r The Barbecue Chicken Pizza 22756.25 1 The Thai Chicken Pizza 21638.5 2 The California Chicken Pizza 21303.25 3 The Classic Deluxe Pizza 19062.5 1 The Hawaiian Pizza 16354.75 2

```
-- Identify the most common pizza size ordered.
1
2
 3 .
       SELECT
           pi.size, COUNT(od.order details id)
4
5
       FROM
           pizzas AS pi
 6
7
               JOIN
           order_details A5 od ON pi.pizza_id = od.pizza_id
8
       GROUP BY pi.size
9
       ORDER BY COUNT(od.order_details_id) DESC
10
11
       LIMIT 1;
12
       -- why we count order_details here?
13
```

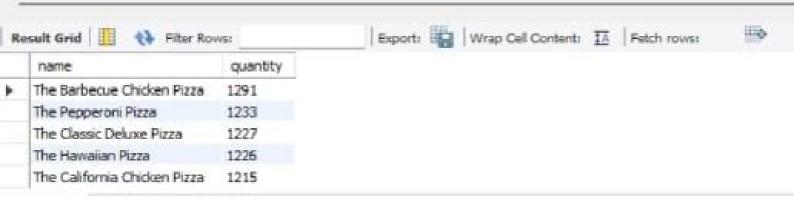


```
1
        -- Calculate the percentage contribution of each pizza type to total revenue.
 2
3
 4 .
       SELECT
           ROUND(SUM(order_details.quantity * pizzas.price) / (SELECT
 5
                           ROUND(SUM(order_details.quantity * pizzas.price),
 6
7
                                       1) AS reveue
                       FROM
8
9
                           order_details
                               JOIN
10
                           pizzas ON pizzas.pizza_id = order_details.pizza_id) * 100,
11
                   2) AS revenue,
12
13
           pizza_types.category
14
       FROM
15
           order_details
16
           pizzas ON pizzas.pizza_id = order_details.pizza_id
17
               JOIN
18
           pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id
19
       GROUP BY pizza_types.category
28
       ORDER BY revenue DESC
21
22
23
```

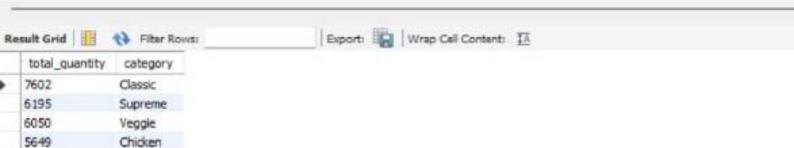
```
-- Analyze the cumulative revenue generated over time.
1
2
 3 .
       select order_date, sum(revenue) over(order by order_date) as cummulative_revenue
    (select orders.order_date, round(SUM(order_details.quantity * pizzas.price),1) as revenue
5
       from orders
6
7
       join order_details
       on orders.order_id = order_details.order_id
8
       join pizzas
9
       on order_details.pizza_id = pizzas.pizza_id
10
       group by orders.order_date) as sales
11
```

R	esult Grid	♦ Filter Rows:	Export: Wrap Cell Content: IX
	order_date	cummulative_revenue	
	2015-01-01	2713.9	
	2015-01-02	5445.8	
	2015-01-03	8108.200000000001	
	2015-01-04	9863.7	
	2015-01-05	11929.7	
	2015-01-06	14358.7	
	2015-01-07	16560.9	

```
1.
       -- List the top 5 most ordered pizza types along with their quantities.
 2
       SELECT
 3 .
           pizza_types.name, SUM(od.quantity) AS quantity
 4
 5
       FROM
           pizzas AS pi
 6
 7
                COIN
8
           order details AS od ON od.pizza id = pi.pizza id
9
           pizza_types ON pizza_types.pizza_type_id = pi.pizza_type_id
10
11
       GROUP BY pizza_types.name
       ORDER BY SUM(od.quantity) DESC
12
13
       LIMIT 5
14
15
```



```
-- Join the necessary tables to find the total quantity of each pizza category ordered.
 1
 2
       SELECT
           sum(order_details.quantity) AS total_quantity,
 5
           pizza_types.category
       FROM
           order_details
7
               JOIN
8
           pizzas ON pizzas.pizza_id = order_details.pizza_id
10
               JOIN
           pizza_types ON pizza_types.pizza_type_id = pizzas.pizza_type_id
11
       GROUP BY pizza_types.category
12
13
       order by total quantity desc
14
```



```
-- Calculate the total revenue generated from pizza sales.

SELECT

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

1) AS reveue

FROM

order_details

JOIN

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





-- Retrieve the total number of orders placed.