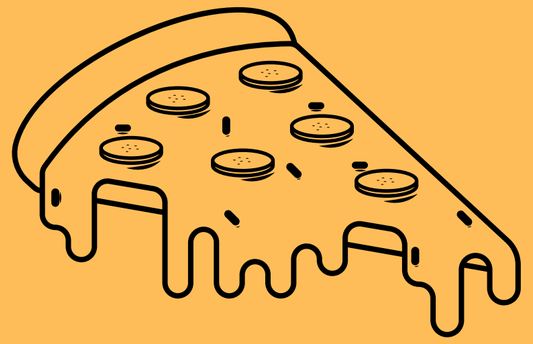


PIZZA SALES PROJECT USING SQL

DATABASE

QUESTIONS

SQL QUERIES

OUTPUTS

PRITPAL KESAR
pritpalkesar999@gmail.com

ABOUT

THIS IS A SQL PROJECT IN WHICH PIZZA SALES DATA IS TAKEN INTO MYSQL AND SEVERAL QUERIES WERE USED TO GATHER SOME IMPORTANT INFORMATION ABOUT THE PIZZA SALES, TO ANSWER THE QUESTIONS MENTIONED

QUESTION 1

Retrieve the total number of orders placed.

SOLUTION

```
select * from orders;
select count(order_id) as total_orders from orders;
```

OUTPUT

	total_orders
▶	21350

QUESTION 2

Calculate the total revenue generated from pizza sales.

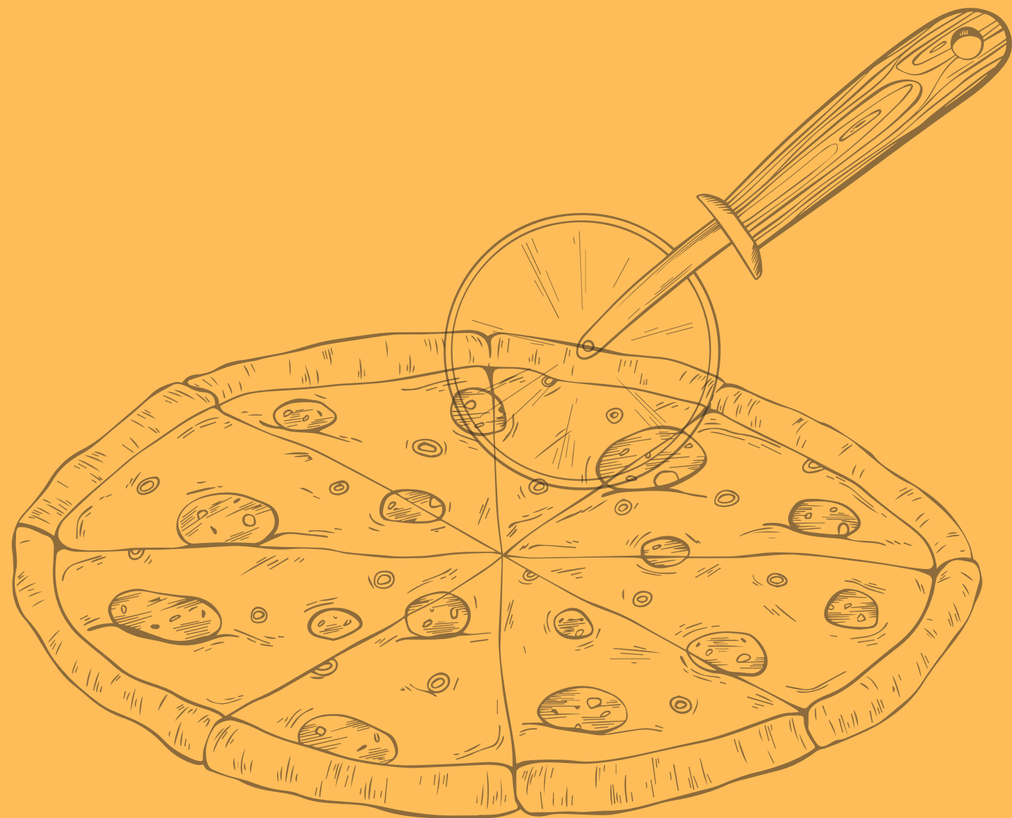
SOLUTION

```
select * from orders_details;
select * from pizzas;

select round(sum(orders_details.quantity* pizzas.price),2) as
total_revenue
from orders_details join pizzas
on pizzas.pizza_id = orders_details.pizza_id
```

OUTPUT

total_revenue
817860.05



QUESTION 3

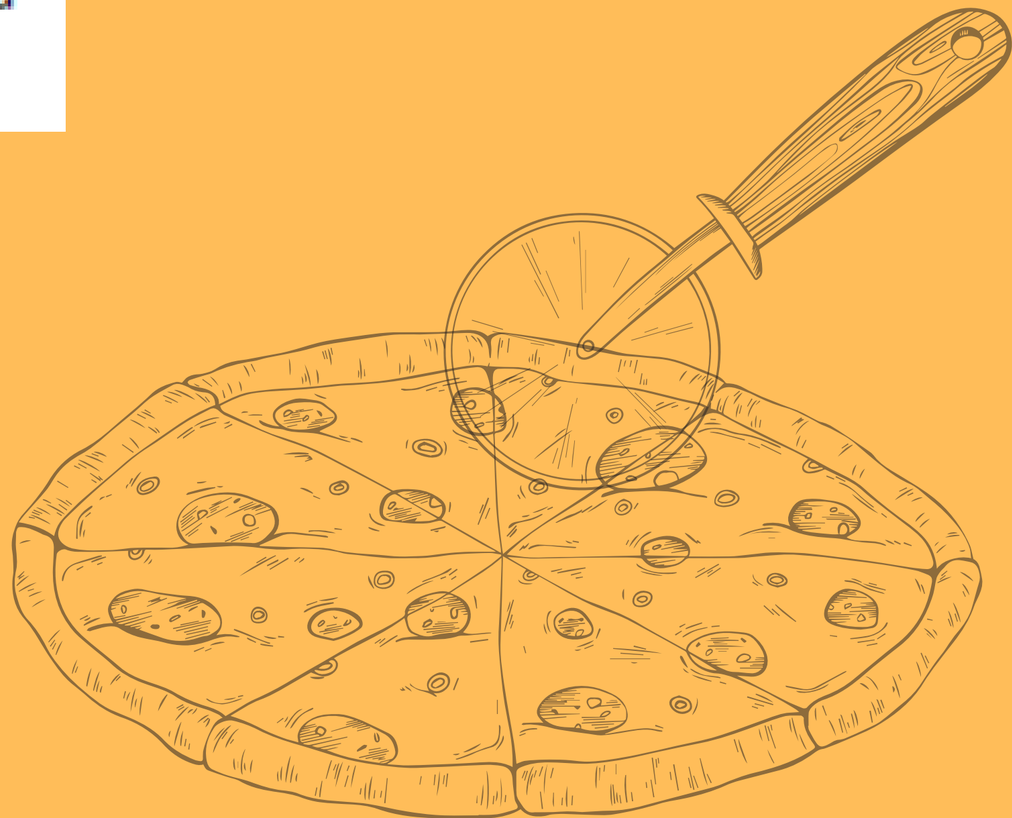
Identify the highest-priced pizza.

SOLUTION

```
select name, price from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
order by price desc limit 1
```

OUTPUT

name	price
The Greek Pizza	35.95



QUESTION 4

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

SOLUTION

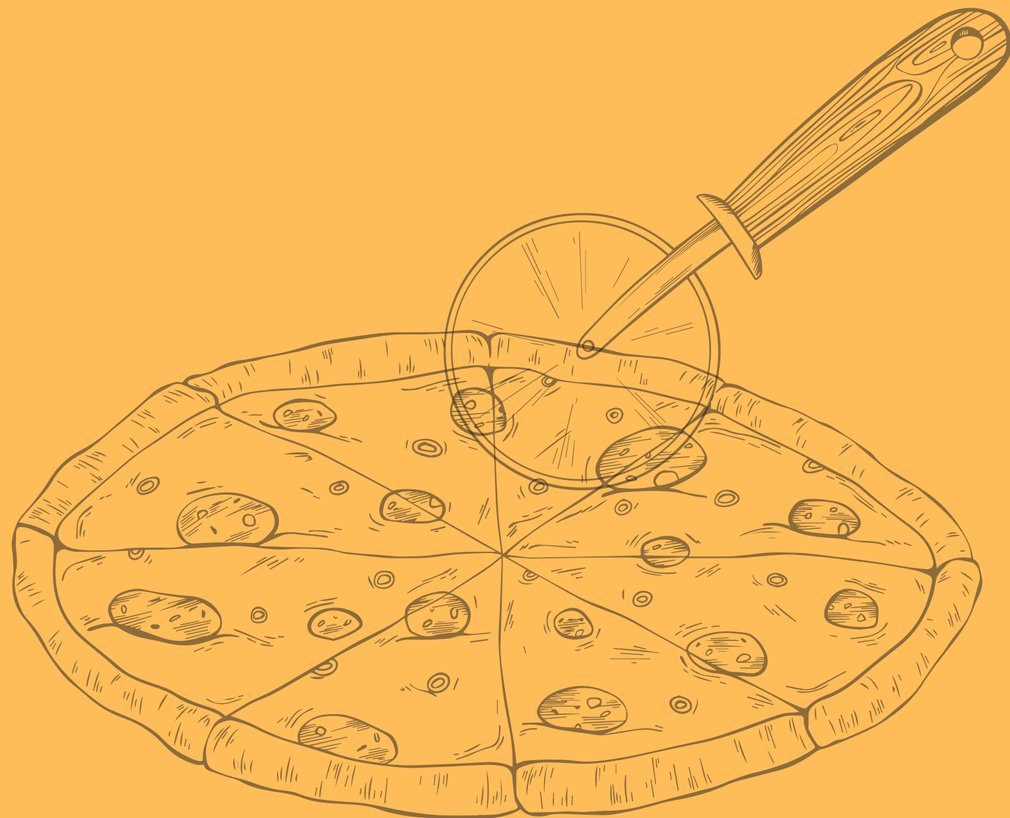
```
select name , sum(orders_details.quantity) as total_quantity
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id

join orders_details
on orders_details.pizza_id = pizzas.pizza_id

group by name
order by total_quantity desc limit 5
```

OUTPUT

name	total_quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371



QUESTION 5

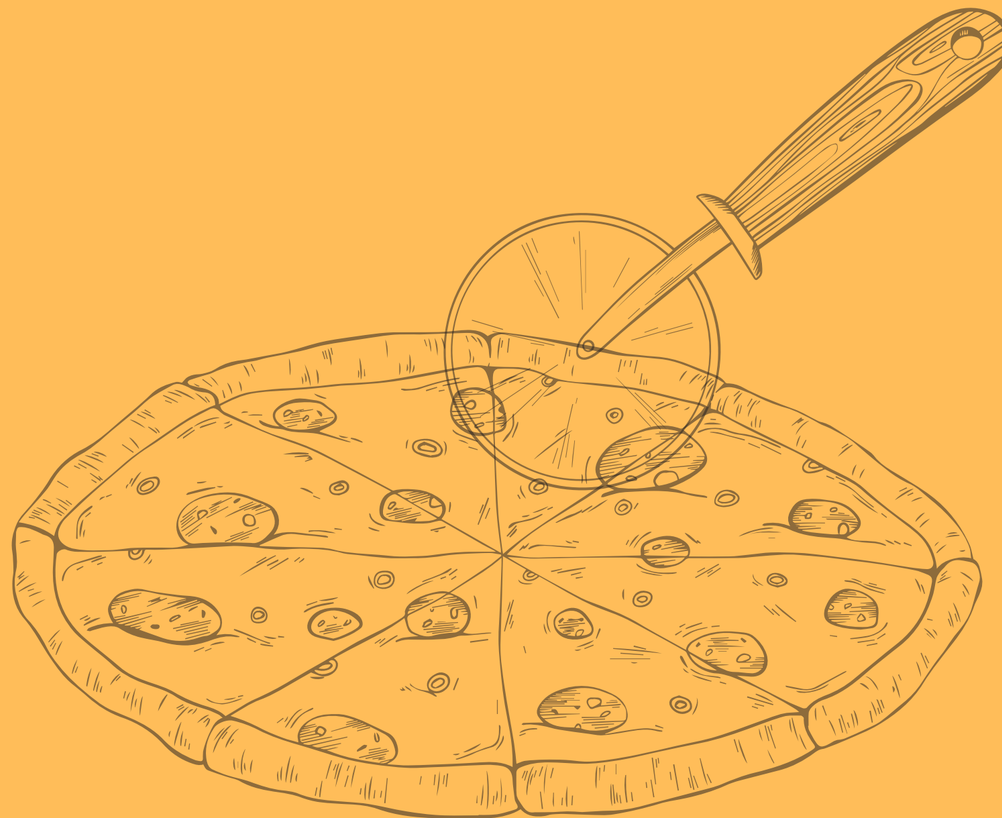
find the total quantity of each pizza category ordered.

SOLUTION

```
select category, sum(orders_details.quantity) as total_quantity
from pizza_types join pizzas
on pizzas.pizza_type_id = pizza_types.pizza_type_id
join orders_details
on orders_details.pizza_id= pizzas.pizza_id
group by category
order by total_quantity desc
```

OUTPUT

category	total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050



QUESTION 6

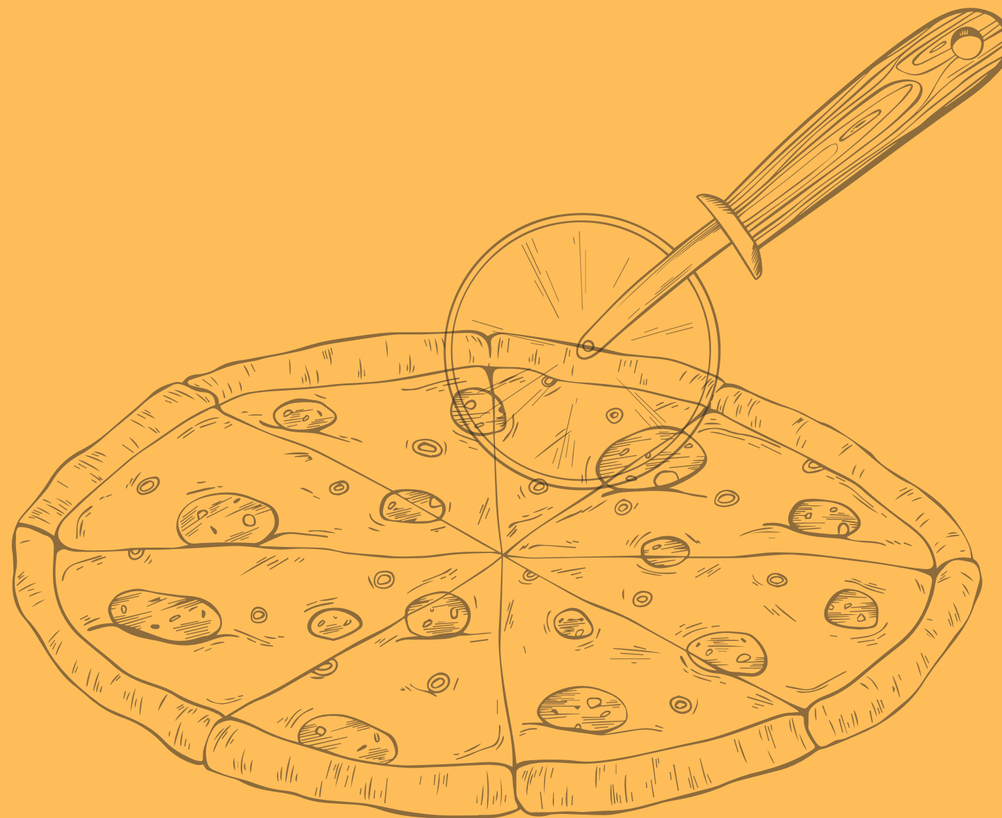
Determine the distribution of orders by hour of the day.

SOLUTION

```
select hour(orders.order_time),
count(orders_details.order_details_id) as orders_count
from orders_details join orders
on orders_details.order_id=orders.order_id
group by hour(order_time)
order by hour(order_time) asc
```

OUTPUT

hour(orders.order_time)	orders_count
9	4
10	17
11	2672



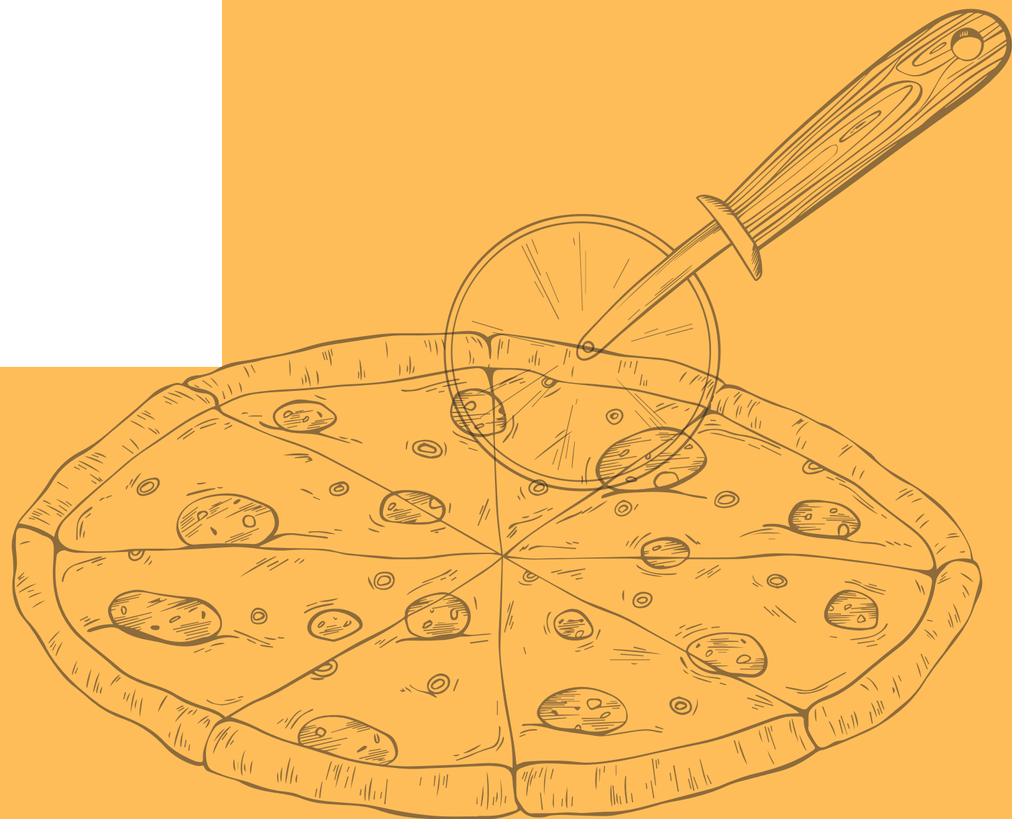
QUESTION 7

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

SOLUTION

```
select round(avg(orders_by_date),0) as average_orders_per_day
from (select orders.order_date,
sum(orders_details.quantity) as orders_by_date
from orders_details join orders
on orders.order_id=orders_details.order_id

group by order_date) as order_data
```



OUTPUT

average_orders_per_day
138

QUESTION 8

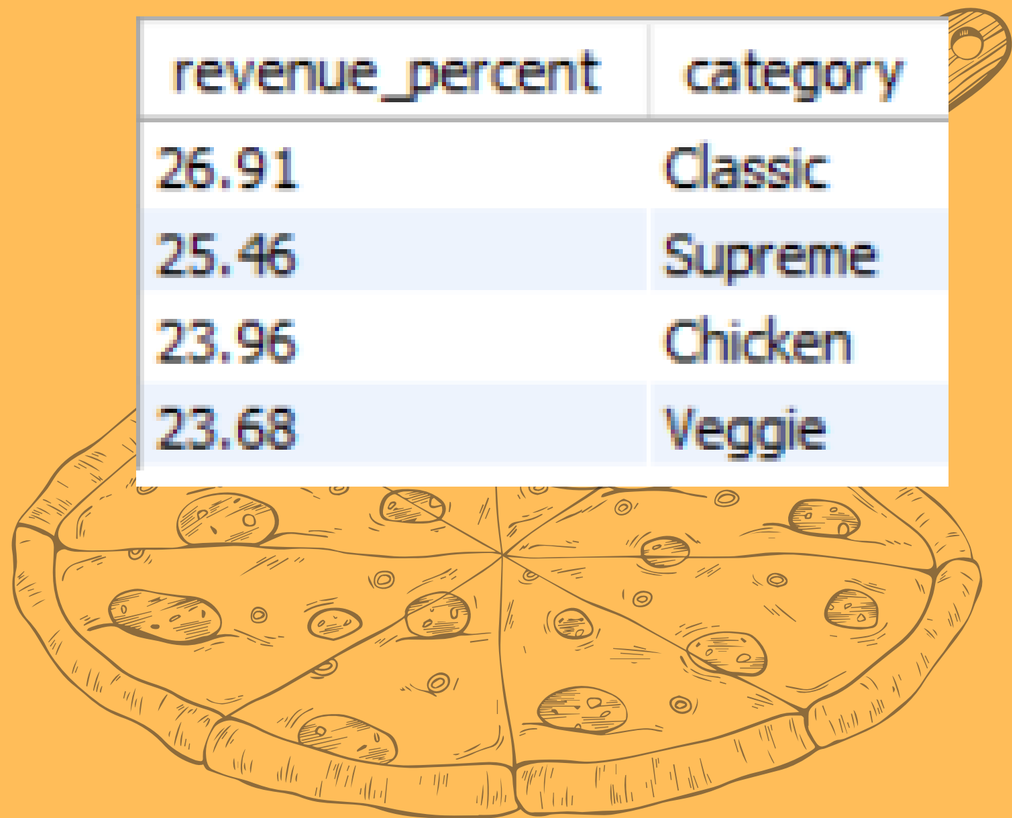
Calculate the percentage contribution of each pizza type to total revenue.

SOLUTION

```
select round(((sum(quantity*price)/(select sum(revenue)
from (select round(sum(quantity*price),2) as revenue, pizza_types.name
from orders_details join pizzas
on orders_details.pizza_id = pizzas.pizza_id
join pizza_types
on pizza_types.pizza_type_id= pizzas.pizza_type_id
group by pizza_types.name
order by revenue desc) as revenue_details))*100),2)
as revenue_percent, pizza_types.category
from orders_details join pizzas
on orders_details.pizza_id = pizzas.pizza_id
join pizza_types
on pizza_types.pizza_type_id= pizzas.pizza_type_id
group by pizza_types.category
order by revenue_percent desc
```

OUTPUT

revenue_percent	category
26.91	Classic
25.46	Supreme
23.96	Chicken
23.68	Veggie



QUESTION 9

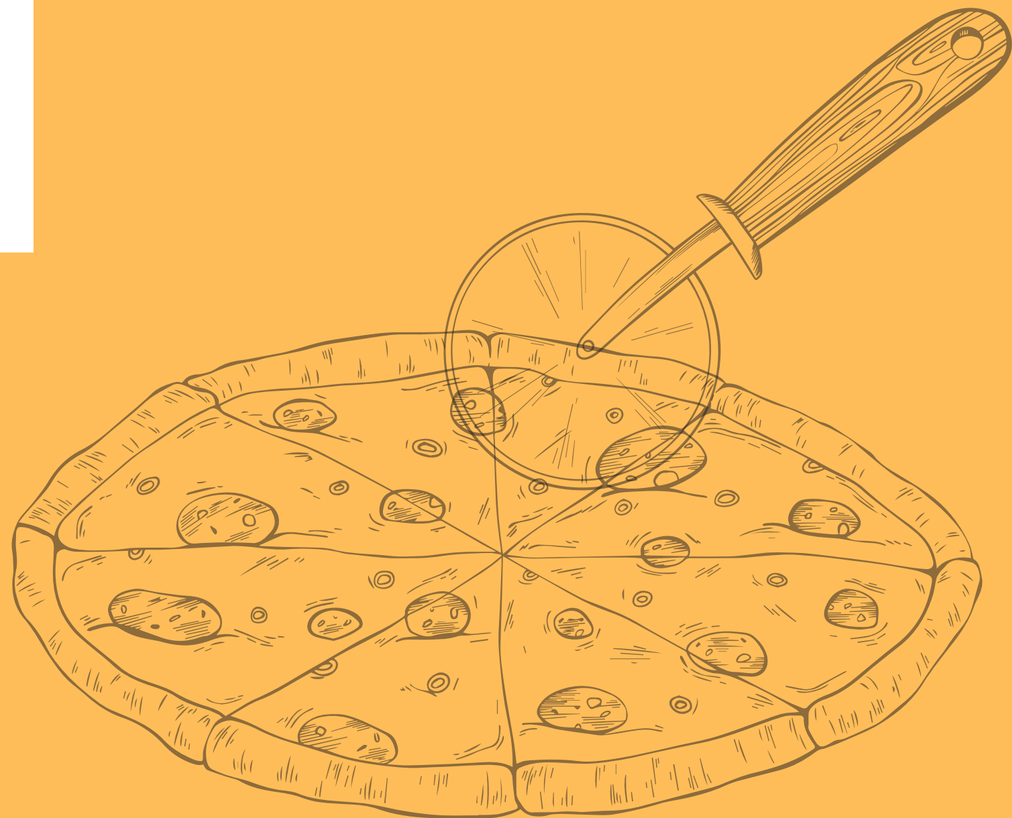
Analyze the cumulative revenue generated over time.

SOLUTION

```
 select order_date, sum(revenue) over(order by order_date) from(
 select order_date, round(sum(quantity*price),2) as revenue
from orders_details join pizzas
on orders_details.pizza_id = pizzas.pizza_id
join orders
on orders.order_id=orders_details.order_id
group by order_date) as sales_data
```

OUTPUT

order_date	sum(revenue) over(order by order_date)
2015-01-01	2713.85
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6



QUESTION 10

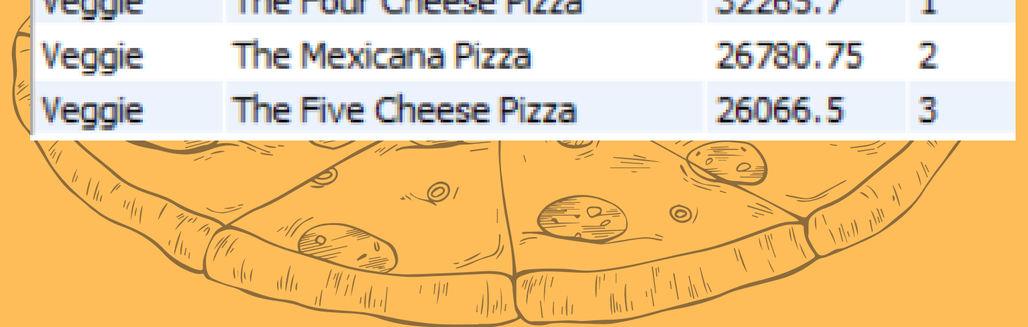
Analyze the cumulative revenue generated over time.

SOLUTION

```
select category, name, revenue, ranks
from(select category, name, revenue, rank()
over (partition by category order by revenue desc) as ranks
from(select name, round(sum(quantity*price),2) as revenue, category
from orders_details join pizzas
on orders_details.pizza_id = pizzas.pizza_id
join pizza_types
on pizza_types.pizza_type_id=pizzas.pizza_type_id
group by name, category) as apple) as ball
where ranks<=3
```

OUTPUT

category	name	revenue	ranks
Chicken	The Thai Chicken Pizza	43434.25	1
Chicken	The Barbecue Chicken Pizza	42768	2
Chicken	The California Chicken Pizza	41409.5	3
Classic	The Classic Deluxe Pizza	38180.5	1
Classic	The Hawaiian Pizza	32273.25	2
Classic	The Pepperoni Pizza	30161.75	3
Supreme	The Spicy Italian Pizza	34831.25	1
Supreme	The Italian Supreme Pizza	33476.75	2
Supreme	The Sicilian Pizza	30940.5	3
Veggie	The Four Cheese Pizza	32265.7	1
Veggie	The Mexicana Pizza	26780.75	2
Veggie	The Five Cheese Pizza	26066.5	3



THANKYOU