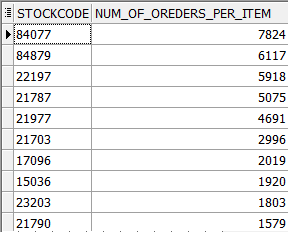
**Q1/Questions asked to get insights:**

1. What is the total number of orders by item, identifying the most popular products, we can focus on their promotion, ensure their availability, and explore opportunities for cross-selling or upselling.

select distinct stockcode ,sum(quantity) over(partition by stockcode ) as num\_of\_oreders\_per\_item

from retail

order by num\_of\_oreders\_per\_item desc;



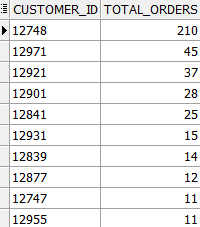
1. What is the total number of orders by customer. This information can be used to identify high-value customers and to develop targeted marketing campaigns to encourage them to continue shopping.

select distinct customer\_id,

count(distinct invoice) over(partition by customer\_id ) AS total\_orders

from retail

order by total\_orders desc;



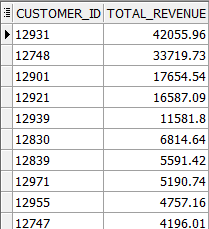
1. What is the total revenue generated by each customer. This query helps us understand the contribution of each customer to the total revenue. By identifying high-revenue customers, we can prioritize them for personalized offers and retention strategies.

select distinct customer\_id,

sum(quantity \* price) over (partition by customer\_id )AS total\_revenue

from retail

order by total\_revenue desc;



1. What is the monthly revenue trend over time? By analyzing the revenue trend over time, we can identify seasonal patterns, detect growth, or decline periods, and plan marketing campaigns or inventory management accordingly.

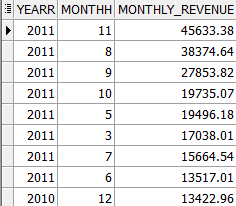
select distinct extract(year from to\_date(invoicedate, 'MM/DD/YYYY HH24:MI')) as yearr,

extract(month from to\_date(invoicedate, 'MM/DD/YYYY HH24:MI')) as monthh,

sum(quantity \* price) over (partition by extract(year from to\_date(invoicedate, 'MM/DD/YYYY HH24:MI')), extract(month from to\_date(invoicedate, 'MM/DD/YYYY HH24:MI'))) as monthly\_revenue

from retail

order by monthly\_revenue desc;



1. What is the customer churn rate. Description: Churn rate helps us understand the percentage of customers who have stopped purchasing. By tracking churn rate, we can identify at-risk customers and implement customer retention strategies to reduce churn.

with last\_purchase\_dates as (

select distinct customer\_id,

last\_value(to\_date(invoicedate, 'MM/DD/YYYY HH24:MI')) over (partition by customer\_id) as last\_pur\_date

from retail

),

churn\_metrics as (

select count(distinct customer\_id) as total\_customers,

sum(

case

when last\_pur\_date < to\_date('12/30/2011 11:59:59 PM', 'MM/DD/YYYY HH12:MI:SS AM') - interval '2' month then 1

else 0

end

) AS churned\_customers

from last\_purchase\_dates

)

select total\_customers,churned\_customers,

round((churned\_customers / total\_customers)\*100,2) as churn\_rate

from churn\_metrics;

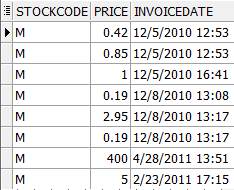


1. The price of the stock is changing over time, the provided data is not sufficient to get insight, but I think it is very important to know why prices are changing and what is affecting it.

select stockcode,price,invoicedate

from retail

where stockcode='M';



**Q2/segmenting customers**

with cu\_info as (

select distinct customer\_id,

round(to\_date('12/09/2011 12:20:00 PM', 'MM/DD/YYYY HH12:MI:SS AM') - last\_value(to\_date(InvoiceDate, 'MM/DD/YYYY HH24:MI')) over (partition by customer\_id)) as recency,

count(distinct invoice) over (partition by customer\_id) as frequency,

sum(quantity \* price) over (partition by customer\_id) as monetary

from retail

),

scores as (

select distinct customer\_id, recency,monetary,frequency,

ntile(5) over (order by recency) as rscore,

ntile(5) over (order by frequency) as fscore,

ntile(5) over (order by monetary) as mscore

from cu\_info

),

fm as (

select distinct customer\_id, round((fscore + mscore) / 2) as fm\_score

from scores

),

finall as (

select distinct s.customer\_id,s.rscore ,f.fm\_score,

case

when s.rscore = 5 and f.fm\_score in (4, 5) then 'Champions'

when s.rscore = 4 and f.fm\_score = 5 then 'Champions'

when s.rscore = 5 and f.fm\_score = 2 then 'Potential Loyalists'

when s.rscore = 4 and f.fm\_score in (2, 3) then 'Loyal Customers'

when s.rscore = 3 and f.fm\_score = 3 then 'Loyal Customers'

when s.rscore = 5 and f.fm\_score = 3 then 'Loyal Customers'

when s.rscore = 4 and f.fm\_score = 4 then 'Loyal Customers'

when s.rscore = 3 and f.fm\_score in (5, 4) then 'Loyal Customers'

when s.rscore = 5 and f.fm\_score = 1 then 'Recent Customers'

when s.rscore = 4 and f.fm\_score = 1 then 'Promising'

when s.rscore = 3 and f.fm\_score = 1 then 'Promising'

when s.rscore = 3 and f.fm\_score = 2 then 'Customers Needing Attention'

when s.rscore = 2 and f.fm\_score = 3 then 'Customers Needing Attention'

when s.rscore = 2 and f.fm\_score = 2 then 'Customers Needing Attention'

when s.rscore = 2 and f.fm\_score in (4, 5) then 'At Risk'

when s.rscore = 1 and f.fm\_score = 3 then 'At Risk'

when s.rscore = 1 and f.fm\_score in (4, 5) then 'Cant Lose Them'

when s.rscore = 1 and f.fm\_score = 2 THEN 'Cant Lose Them'

when s.rscore = 1 and f.fm\_score = 1 THEN 'Cant Lose Them'

else 'LOST'

end as segmentation

from fm f ,scores s

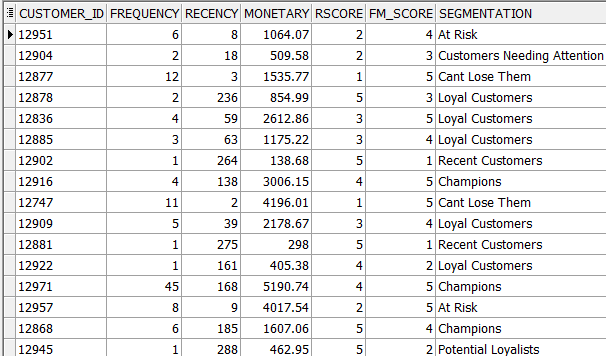
where f.customer\_id = s.customer\_id

)

select distinct fi.customer\_id,c.frequency, c.recency,c.monetary,fi.rscore,fi.fm\_score,segmentation

from finall fi ,cu\_info c

where fi.customer\_id = c.customer\_id



**Q3A/maximum number of consecutive days a customer made purchases.**

with cust\_transactions as (

select cust\_id, dt, lead(dt) over (partition by cust\_id order by dt) as next\_dt

from transactions

),

consecutive\_days as (

select distinct Cust\_id,count(\*) over(partition by cust\_id) as days

from cust\_transactions

where next\_dt - dt = '1'

)

select Cust\_id, max(days) over(partition by cust\_id) as maximum\_consecutive\_days

from consecutive\_days;

**A screenshot of a data

Description automatically generated with low confidence**

**Q3B/How many days/transactions does it take a customer to reach a spent threshold of 250 L.E**

with ordered\_transactions as (

select cust\_id, dt, amt,

sum(amt) over (partition by cust\_id order by dt) as total\_amount,

row\_number() over (partition by cust\_id order by dt) as rn

from transactions

)

select avg(days\_to\_250) as average\_days\_to\_thresh

from (

select cust\_id, min(rn) over(partition by cust\_id) as days\_to\_250

from ordered\_transactions

where total\_amount >= 250

)

