Analytical SQL Project

Name: Mohamed Ahmed Fathy

Q1- Using OnlineRetail dataset

write at least 5 analytical SQL queries that tells a story about the data

write small description about the business meaning behind each query

1- Query1:

```
select customer_id, sum(quantity * price) as total_revenue,
rank() over (order by sum(quantity * price) desc) as rank
from tableretail
group by customer_id
order by total_revenue desc;
```

CUSTOMER_ID	TOTAL_REVENUE	RANK
12931	42055.96	1
12748	33719.73	2
12901	17654.54	3
12921	16587.09	4
12939	11581.8	5
12830	6814.64	6

The company wanted to know the highest customers that makes revenue to company, to make some offers to them depending on percentage of their total_revenue, so we used rank function to rank customer by sales

2- Query2

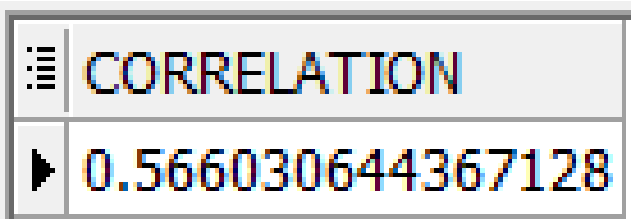
```
with monthly_revenue as (
  select customer_id, to_char(to_date(invoicedate, 'mm/dd/yyyy
hh24:mi'), 'yyyy') as year, to_char(to_date(invoicedate, 'mm/dd/yyyy
hh24:mi'), 'fmmm') as month, sum(quantity * price) as revenue
  from tableretail
  group by customer_id, to_char(to_date(invoicedate, 'mm/dd/yyyy
hh24:mi'), 'yyyy'), to_char(to_date(invoicedate, 'mm/dd/yyyy
hh24:mi'), 'fmmm')
)
select customer_id, year, month, revenue,
       round(avg(revenue) over (partition by customer_id order by
year, month rows between 2 preceding and current row)) as
rolling_avg
from monthly_revenue
order by (rolling_avg - revenue) desc;
```

CUSTOMER_ID	YEAR	MONTH	REVENUE	ROLLING_AVG
12748	2011	2	389.64	4033
12748	2011	12	1070.27	4667
12931	2011	3	62.5	3125
12931	2011	2	1696.4	3741
12748	2011	1	418.77	2323
12939	2011	7	761.76	2019
12921	2011	3	1063.87	2310
12830	2011	9	1210.32	2138
12901	2011	7	1301.03	2074

the company want to know the most customer in danger of churn, which means who are the highest customers that buy products and suddenly they stopped or selling rate largly decreased, so first we made a cte that has the monthly revnue, then from that calculated the avg between current month, and the last 2 months, as a new column, and we select that column + the montly revenue and compare them together by ordering descing of the difference between avg and revenue, so now those customers are the most important to care of because their sales decreased.

3-Query3

```
with cte as (
  select customer_id, count(distinct stockcode) as num_products,
sum(quantity * price) as revenue
  from tableretail
  group by customer_id
)
select corr(num_products, revenue) as correlation
from cte;
```



CORRELATION
0.566030644367128

the company need to decide a very important decision, they want to improve and they got to choose 1o f 2 choices, either to increase the number of products to increase sales or improve the product quality and offers on them, also there will be a marketing campaign , so what to focus on? , so the BI team wanted to study the correlation between the number of products and revenue, and what is the relation between that each customer buy different product, and revenue he makes? And the result was that there is the decent relation between increasing the number of product and sales but yet it’s not so high, but worth focusing on.

4-Query4

```
with cte as (
select customer_id,
to_char(to_date(invoicedate, 'mm/dd/yyyy hh24:mi'), 'yyyy') as year,
sum(quantity * price) as revenue,
sum(quantity * price) / lag( sum(quantity * price)) over (partition by
customer_id order by to_char(to_date(invoicedate, 'mm/dd/yyyy hh24:mi'),
'yyyy'))  as change_pct,
lag( sum(quantity * price)) over (partition by customer_id order by
to_char(to_date(invoicedate, 'mm/dd/yyyy hh24:mi'), 'yyyy'))  as
last_year_revenue
  from tableretail
  group by customer_id, to_char(to_date(invoicedate, 'mm/dd/yyyy
hh24:mi'), 'yyyy')
)
select customer_id, year,revenue, last_year_revenue, round(Change_Pct)
as rate
from cte
where change_pct is not null
order by change_pct desc ;
```

CUSTOMER_ID	YEAR	REVENUE	LAST_YEAR_REVENUE	RATE
12931	2011	41878.96	177	237
12921	2011	15899.29	687.8	23
12971	2011	4754.55	436.19	11
12867	2011	3674.03	362.79	10
12877	2011	1385.71	150.06	9
12826	2011	1319.72	155	9

here the business wanted to make a new category for loyal customer, which is calculated by how much did this customer improve from the last year, as the company sales are only in 2010 and 2011, so if a customer was in 2010, and stayed with us in 2011, and made much more sales in 2011, then that's a very good progress and loyalty we ned to keep using it and make him offers to even add more of friends.

Note: rate is this year revenue over the last year revenue.

5-Query5

```
with cte as (
  select  invoice, stockcode
  from tableretail
)
select
  c1.stockcode as item1,
  c2.stockcode as item2,
  count(*) as freq,
  dense_rank() over (order by count(*) desc) as freq_rank
from cte c1
join cte c2 on c1.invoice = c2.invoice and c1.stockcode < c2.stockcode
group by c1.stockcode, c2.stockcode
having count(*) >= 5
order by freq desc;
```

ITEM1	ITEM2	FREQ	FREQ_RANK
20724	22355	23	1
20725	20728	22	2
82482	82494L	21	3
20719	22355	21	3
20725	22382	21	3
20725	22384	21	3

Might be kinda complicated, but its so fun, the most famous business case everyone heard of about BI, which is what is the most 2 product bought together? Or can put offers for them together or put them close to each others, so in this case we want to rank every 2 products sold together, to have priority for redistribution of the store, and also this report will be submitted to marketing service, to work on making offers for those products together, with cte and rank, this logic became much easier than sub query.

Q2:

Note: the highest user has very high data comparing to others, so I think to have better analysis , we should have discluded him specially in frequency and monetary with customer ID(12748)

Note: first the monetary was calculated in the given pdf as sum(price) and I made it sum of quantity*price since that is more relevant

Note: the rules weren't given so I assumed that for recency will round the max recency – current / max , because its inverse relation, the customer with lower recency should get higher score

Note: for score of fm_score, I assumed we do ceiling instead of round, because for me not logic to have 0 as a score with money, since he already paid so started it from 1

Note: I wanted to create table for score and join it with this query, but as it wasn't given to do that , so made it with case

Query:

```
with customer_data as (
  select
    customer_id,
    to_date('2011-12-09', 'yyyy-mm-dd') - max(to_date(invoicedate, 'mm/dd/yyyy
hh24:mi')) as recency,
    count(distinct invoice) as frequency,
    sum( quantity * price) as monetary
  from
    tableretail
  group by
    customer_id
),
scoring_data as (
  select
    customer_id,
    ceil(recency) as recency_days,
    frequency,
    round(monetary) as monetary,
    round(((select max(recency) from customer_data) - recency) / (select
max(recency) from customer_data) * 5 ) as r_score,
```

```
 ceil(( (frequency / (select max(frequency) from customer_data))  * 5  + (monetary
/ (select max(monetary) from customer_data))  * 5) /2 ) as fm_score
  from
    customer_data
)
select
  customer_id, recency_days, frequency, monetary, r_score,fm_score,
  case
    when r_score = 5 and fm_score >= 4 or r_score = 4 and fm_score = 5 then
'champion'
    when r_score >= 4 and fm_score = 2 or r_score in (3, 4) and fm_score = 3 then
'potential_loyalists'
    when r_score = 5 and fm_score = 3 or r_score = 4 and fm_score = 4 or r_score
= 3 and fm_score = 5 or r_score = 3 and fm_score = 4 then 'loyal customer'
    when r_score = 5 and fm_score = 1 then 'recent customer'
    when r_score in(3,4)  and fm_score = 1 then 'promising'
    when r_score = 3 and fm_score = 2 or r_score = 2 and fm_score = 3 or r_score
= 2 and fm_score = 2  then 'customer need attention'
    when r_score = 2 and fm_score = 5 or r_score = 2 and fm_score = 4 or r_score
= 1 and fm_score = 3  then 'at risk'
    when r_score = 1 and fm_score = 5 or r_score = 1 and fm_score = 4   then
'cant lose them'
    when r_score = 1 and fm_score = 2   then 'hibernating'
    else 'lost' end as segment
from
  scoring_data
order by
  fm_score desc,  r_score desc;
```

CUSTOMER_ID	RECENCY_DAYS	FREQUENCY	MONETARY	R_SCORE	FM_SCORE	SEGMENT
12748	0	210	33720	5	5	Champion
12931	21	15	42056	5	3	Loyal Customer
12901	8	28	17655	5	2	Potential_Loyalists
12921	9	37	16587	5	2	Potential_Loyalists
12828	2	6	1019	5	1	Recent Customer
12841	4	25	4022	5	1	Recent Customer
12844	29	2	326	5	1	Recent Customer
12856	7	6	2180	5	1	Recent Customer

Q3:

a- What is the maximum number of consecutive days a customer made purchases?

```
with consecutive_days as (
  select
    cust_id,
    calendar_dt,
    amt_le,
    calendar_dt - interval '1' day * (row_number() over(order by cust_id ,
calendar_dt)) as base
  from
    customer_transactions
  where
    amt_le > 0
),
consecutive_groups as (
  select
    cust_id,base,
   count(base) as consective_days
  from
    consecutive_days
      group by base, cust_id

)
select cust_id,max(consective_days) as max_consective_days
from consecutive_groups
group by cust_id ;
```

CUST_ID	MAX_CONSECTIVE_DAYS
36269445	9
36467785	7
38348625	26
39503745	5
39871481	8
40202120	5
40581557	7

Explaining:

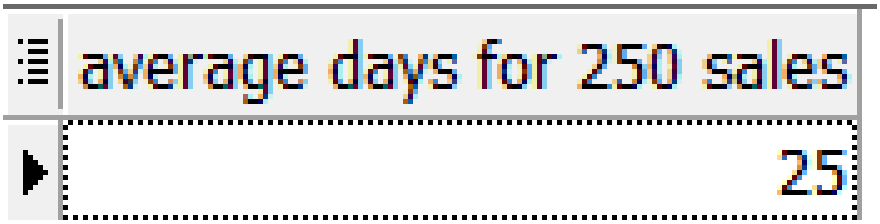
First made a CTE that selects all the data normally and adds a row number, in case we subtract the date from row number, if there are consective (difference between them is a day) then the output will be same base date, if not it will be different, then another CTE to count the equivalent base date together and group by them, then the output will be that for every customer, will have several consecutive dates, then in the last select, will select his maximum consecutive dates, and tests the values.

b- On average, How many days/transactions does it take a customer to reach a spent threshold of 250 L.E?

not sure if you want for each customer how many days/transactions, or just 1 row as average days for all the data, so brought both,

first average for all data:

```
with sales_calc as (select cust_id , amt_le, calendar_dt, sum(amt_LE) over
(partition by cust_id order by cust_id rows between unbounded preceding
and current row) as sales ,
row_number() over(partition by cust_id order by cust_id ) as days_count
from customer_transactions
order by cust_id
),
 rank_count as (
select cust_id, amt_le, calendar_dt, sales,days_count, RANK() OVER
(partition BY cust_id ORDER BY sales) AS sales_rank
from sales_calc
where sales > 250
)
select  round(avg(days_count)) as "average days for 250 sales"
from rank_count ;
```



second for each customer:

```
with sales_calc as (select cust_id , amt_le, calendar_dt, sum(amt_LE) over
(partition by cust_id order by cust_id rows between unbounded preceding
and current row) as sales ,
row_number() over(partition by cust_id order by cust_id ) as days_count
from customer_transactions
order by cust_id
),
 rank_count as (
select cust_id, amt_le, calendar_dt, sales,days_count, RANK() OVER
(partition BY cust_id ORDER BY sales) AS sales_rank
from sales_calc
where sales > 250
)
select cust_id , sales, days_count
from rank_count
where sales_rank = 1;
```

CUST_ID	SALES	DAYS_COUNT
26592	396.37	6
45234	262.53	9
60045	301.12	7
66688	339.88	4
113502	275.72	12
151293	442.07	6
217534	468.75	8
232210	275.22	8
259866	422.36	7