# Online Retail Case study

## Question 1:

## First analytical SQL:

```
---------------------------- First Analytical SQL ----------------------------
select to_date(to_char(to_date(invoicedate, 'mm/dd/yyyy hh24:mi'), 'mm/yyyy'), 'mm/yyyy' ) as InvoiceDate,
sum(price*quantity) as Revenue, rank() over(order by sum(price*quantity) desc) as Max_Revenue_Rank
from tableretail
group by to_date(to_char(to_date(invoicedate, 'mm/dd/yyyy hh24:mi'), 'mm/yyyy'), 'mm/yyyy')
order by Revenue desc;
```

We wanted to discover the total revenue per month to discover when the online retail gain the most. As the result, we notice the following:

the online retail gained the most in November, then in August.

Therefore, we need more data to have a clear insights about the reason behind this significant increase in revenue in those two months compared to other months.

We also need to make further analysis to discover the reason behind this drop in revenue in January to fix the problem and increase the revenue.

## Second analytical SQL:

```
---------------------------- Second Analytical SQL ----------------------------
select customer_id, InvoiceDate, Recent_Rank, Customer_max_payment, Customer_min_payment
from(
select customer_id, Invoice, StockCode, Quantity, InvoiceDate, Price, Country,
last_value(price*quantity) over(partition by customer_id order by price*quantity rows between unbounded
preceding and unbounded following) Customer_max_payment,
rank () over(partition by customer_id order by to_date(invoicedate, 'mm-dd-yyyy hh24:mi') desc) as Recent_Rank,
last_value(price*quantity) over(partition by customer_id order by price*quantity desc rows between unbounded
preceding and unbounded following) Customer_min_payment
from tableretail)
where Recent_Rank=1
group by customer_id, InvoiceDate, Recent_Rank, Customer_max_payment, Customer_min_payment
order by Customer_max_payment desc;
```

The online retail wants to discover who are the customers who have the most contribute in revenue by their purchases,

by that they can make a survey to discover what encourage those customers to make huge purchase like that to help them in their marketing plan.

In the result, the table shows the customer_id, and their recent purchase date to make sure that it was not from the long time, and their maximum payment to the retail.

## Third analytical SQL:

```
-------------------------- Third Analytical SQL ----------------------------
select to_date(to_char(to_date(invoicedate, 'mm/dd/yyyy hh24:mi'), 'mm/yyyy'), 'mm/yyyy') as InvoiceDate,
sum(price*quantity) as Paid, count(*) as Transaction_Count,
round((sum(price*quantity)/count(*)),2) as Avg_Transaction_Paid,
rank() over(order by count(*) desc) as Transactipn_Count_Rank,
rank() over(order by sum(price*quantity) desc) Paid_Rank
from tableretail
group by to_date(to_char(to_date(invoicedate, 'mm/dd/yyyy hh24:mi'), 'mm/yyyy'), 'mm/yyyy')
order by Paid desc;
```

We wanted to make a further a further analysis to discover if there a correlation between the months revenue and the amount of transaction the online retail had.

As a result, we can notice that there's a positive correlation.

In August, we can notice that the amount of transactions made is significantly less than other months , while it has the second biggest revenue compared to other months, which can be indicator that customers bought expensive stuff in this month.

## Fourth analytical SQL:

```
-------------------------- Fourth Analytical SQL ----------------------------
select stockcode, sum(quantity) as Quantity,
rank() over( order by sum(quantity) desc) as Quantity_Rank
from tableretail
group by stockcode;
```

We wanted to know which stock codes sold the most to discover what attracts our customers the most.

Therefore, we can increase our revenue in the future using what attracts the customers the most.

## Fifth analytical SQL:

```
------------------------- Fifth Analytical SQL ---------------------------
select invoicedate, Amount_sold,
lag(Amount_sold,1) over( order by invoicedate ) as lag_month_frequency,
lag(Amount_sold,2) over( order by invoicedate ) as lag_2months_frequency
from (
select to_date(to_char(to_date(invoicedate, 'mm/dd/yyyy hh24:mi'), 'mm/yyyy'), 'mm/yyyy') as invoicedate,
sum(quantity) as Amount_sold
from tableretail
group by to_date(to_char(to_date(invoicedate, 'mm/dd/yyyy hh24:mi'), 'mm/yyyy'), 'mm/yyyy')
)
order by invoicedate desc;
```

We want to compare the amount sold of this month with the previous 1month amount and 2 months amount.

That's to be prepared for the future amounts.

## Question 2: Monetary model:

```
------------------------------------------------ Q2 ------------------------------------------------------
---------------------------------- Case Study ----------------------------------
-- creating our customer table for a later use
create table customer as
select Customer_ID,
(select to_date(to_char(max(to_date(invoicedate, 'mm/dd/yyyy hh24:mi')), 'mm/dd/yyyy'), 'mm/dd/yyyy')from
tableretail) -
(to_date(to_char(max( to_date(invoicedate, 'mm-dd-yyyy hh24:mi')), 'dd-mm-yyyy'), 'dd-mm-yyyy') ) as recency,
sum(price*quantity) as Monetary, count(*) as Frequency, ntile(5) over (order by count(*)*avg(price*quantity)) as
FM_Score,
ntile(5) over (order by (select to_date(to_char(max(to_date(invoicedate, 'mm/dd/yyyy hh24:mi')), 'mm/dd/yyyy'),
'mm/dd/yyyy')from tableretail) -
 (to_date(to_char(max( to_date(invoicedate, 'mm-dd-yyyy hh24:mi')), 'dd-mm-yyyy'), 'dd-mm-yyyy'))) as R_Score
from tableretail
group by customer_id
order by Monetary;

-------------------------------------------------------------

select CUSTOMER_ID, RECENCY, FREQUENCY, MONETARY, R_SCORE, FM_SCORE,
(case when (R_SCORE=5 and FM_SCORE=5) or  (R_SCORE=5 and FM_SCORE=5) or (R_SCORE=5 and
FM_SCORE=5) then 'Champions'
when (R_SCORE=5 and FM_SCORE=2) or  (R_SCORE=4 and FM_SCORE=2) or (R_SCORE=4 and FM_SCORE=3)
or (R_SCORE=3 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 Customers'
when (R_SCORE=5 and FM_SCORE=1) then 'Recent Customers'
when (R_SCORE=4 and FM_SCORE=1) or  (R_SCORE=3 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 'Customers Needing 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)
or (R_SCORE=2 and FM_SCORE=1) 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'
when (R_SCORE=1 and FM_SCORE=1)then 'Lost'
else 'wrong data'
end) as Cust_Segment
from customer;
-------------------------------------------------------
```