**Case Study**

**Question One**

I Created a view called **Retail\_view** and added a new calculated column in this view which is **Sales (Quantity \* Price).**



**#1**

I calculated the profit share of the top 20% customer to have a good understanding of where are the majority of our sales come from and the result was surprisingly beneficial as the top 20% customer are responsible for around **73% of the total sales** of the company.

**SELECT**

**(SELECT SUM(sum\_per\_cust)**

**FROM (**

**SELECT SUM(sales) AS sum\_per\_cust, PERCENT\_RANK() OVER (ORDER BY SUM(sales) DESC) \* 100 AS Sales\_perce**

**FROM Retail\_view**

**GROUP BY Customer\_ID**

**) rs**

**WHERE Sales\_perce <= 20**

**) /**

**(SELECT SUM(sales) FROM Retail\_view) \* 100 AS sales\_percentage**

**FROM dual;**

****

**#2**

The Second Insight is even more interesting, Only 5% of the products (StockCode) are responsible for more than 52% of the total sales, this is very important as almost 95% of our product lines has serious issues when it comes to sales, But we have to take into consideration that this number isn’t totally honest as we don’t have here the cost of these product to be able to calculate The revenue which would be a better indicator, but still the GAP is too huge!!

**SELECT**

**(SELECT SUM(sum\_per\_STOCK)**

**FROM (**

**SELECT SUM(sales) AS sum\_per\_STOCK, PERCENT\_RANK() OVER (ORDER BY SUM(sales) DESC) \* 100 AS Sales\_perce**

**FROM Retail\_view**

**GROUP BY STOCKCODE**

**) rs**

**WHERE Sales\_perce <= 5**

**) /**

**(SELECT SUM(sales) FROM Retail\_view) \* 100 AS sales\_percentage**

**FROM dual;**



**#3**

The previous Insight could mean that those 5% of our product are more expensive than the others and doesn’t have to mean that they are the most popular products, and this make us ask an important question to understand the nature of our customer behavior, In other words, Is the higher the price the higher the sum of sales?   
It seems Not, the correlation between the price of the product and the sum of sales per product indicates that there’s no strong relation between those two factors.  
  
**select corr ( price , sum\_per\_prod ) from (  
select StockCode , price , sum(price) over(PARTITION BY StockCode) as sum\_per\_prod from retail\_view)**



**#4**

This is some of our least favorable products since that only 5% of our product that gets more than half of our sales, I believe that considering either stop producing those products or at least have a serious look on what is wrong with them.

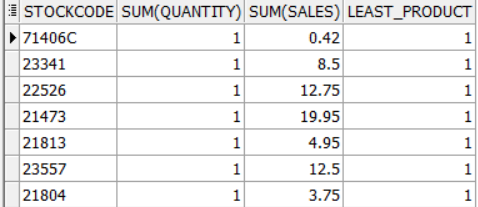
**select \* from (**

**select StockCode , sum(QUANTITY) , sum(Sales) , Dense\_Rank() OVER (ORDER BY SUM(QUANTITY) ) AS least\_product**

**From Retail\_view**

**group by StockCode )**

**where least\_product < 50**



**#5**To Monitor Our sales variation during those years I had to get the difference between each two consecutive months to get the negative difference in sales which means the months that we faced a drop in sales to investigate what happened back then

**WITH monthly\_sales AS (**

**SELECT**

**TO\_CHAR(TO\_DATE(INVOICEDATE, 'DD-MON-YY'), 'YYYY') AS YEAR,**

**TO\_CHAR(TO\_DATE(INVOICEDATE, 'DD-MON-YY'), 'MONTH') AS MONTH\_NAME,**

**SUM(Sales) AS current\_Month\_Sales,**

**LAG(SUM(sales)) OVER (ORDER BY TO\_CHAR(TO\_DATE(INVOICEDATE, 'DD-MON-YY'), 'YY'), TO\_CHAR(TO\_DATE(INVOICEDATE, 'DD-MON-YY'), 'MM')) AS Prev\_Month,**

**TO\_CHAR(TO\_DATE(INVOICEDATE, 'DD-MON-YY'), 'MM') AS Prev\_Month\_Number,**

**(SUM(Sales) - LAG(SUM(sales)) OVER (ORDER BY TO\_CHAR(TO\_DATE(INVOICEDATE, 'DD-MON-YY'), 'YY'), TO\_CHAR(TO\_DATE(INVOICEDATE, 'DD-MON-YY'), 'MM'))) AS Difference**

**FROM Retail\_view**

**GROUP BY TO\_CHAR(TO\_DATE(INVOICEDATE, 'DD-MON-YY'), 'YY'), TO\_CHAR(TO\_DATE(INVOICEDATE, 'DD-MON-YY'), 'MM'), TO\_CHAR(TO\_DATE(INVOICEDATE, 'DD-MON-YY'), 'YYYY'), TO\_CHAR(TO\_DATE(INVOICEDATE, 'DD-MON-YY'), 'MONTH')**

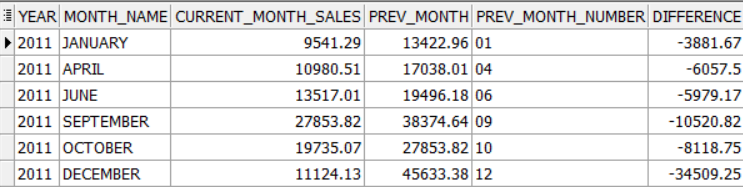
**)**

**SELECT \***

**FROM monthly\_sales**

**WHERE Difference < 0;**

**#6**



Although this is not an analytical function but it gives a great insight that the highest 4 months of sales were at the last quarter of 2011 which gives us a good indication that whatever we were doing at this period it has a great effect in our sales.

**SELECT**

**TO\_CHAR(TO\_DATE(INVOICEDATE, 'DD-MON-YY'), 'YYYY') AS YEAR,**

**TO\_CHAR(TO\_DATE(INVOICEDATE, 'DD-MON-YY'), 'MONTH') AS MONTH\_NAME,**

**SUM(sales) / (SELECT SUM(sales) FROM Retail\_view) \* 100 AS PERCENTAGE\_SALES**

**FROM**

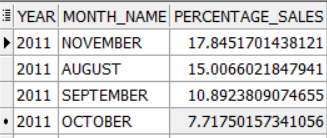
**Retail\_view**

**GROUP BY**

**TO\_CHAR(TO\_DATE(INVOICEDATE, 'DD-MON-YY'), 'MONTH'),**

**TO\_CHAR(TO\_DATE(INVOICEDATE, 'DD-MON-YY'), 'YYYY')**

**order by PERCENTAGE\_SALES desc**



**Question Two**

I choose to divide the queries into simple views to enhance the readability and the maintenance as well in case of increasing any other business requirements.

**Segment\_base**   
This View Calculates Recency, Frequency and Monetary, and here we can add any extra business requirements.

**Create View segment\_base as**

**select Customer\_ID ,**

**( select Max(to\_date( INVOICEDATE)) from tableretail ) - Max( to\_date(Invoicedate) ) as recency ,**

**count(distinct Invoice) as frequency ,**

**sum(Quantity \* Price) as monetary**

**FROM tableretail**

**group by Customer\_ID**

**Segment\_Score**

In this view we calculate the scores for each KPI we have.

**create View segment\_score as**

**select Customer\_ID , recency , frequency , monetary ,**

**NTILE (5) Over (Order by Recency desc) as r\_score ,**

**NTILE (5) Over (Order by monetary ) as m\_score ,**

**NTILE (5) Over (Order by frequency ) as f\_score**

**FROM segment\_base**

**Segment\_Scor\_FM**

Since fm\_score is calculated from averaging two other scores I preferred to calculate it in a sperate view to enhance readability and clarity.

**create View segment\_score\_fm as**

**select Customer\_ID , recency , frequency , monetary , r\_score , m\_score , f\_score ,**

**NTILE (5) Over (Order by (m\_score + f\_score ) /2 ) as fm\_score**

**from segment\_score**

**Final\_Segment**

It’s the view that applies the business segment naming to the customers, and it can be easy changed or modified since it’s in a sperate view.

**create View final\_segment as**

**select Customer\_ID , recency , frequency , monetary , r\_score , m\_score , f\_score , fm\_score ,**

**CASE**

**when r\_score in (5,4 ) and fm\_score in (5,4) then 'Champions'**

**when r\_score in (5,4,3 ) and fm\_score in (2,3) then 'Potential Loyalists'**

**when r\_score in (5,4,3 ) and fm\_score in (3,4,5) then 'Loyal Customers'**

**when r\_score in (5 ) and fm\_score in (1) then 'Recent Customers'**

**when r\_score in (4,3 ) and fm\_score in (1) then 'Promising'**

**when r\_score in (2,3 ) and fm\_score in (2,3) then 'Customers needing Attention'**

**when r\_score in (1,2 ) and fm\_score in (3,4,5) then 'At Risk'**

**when r\_score in (1 ) and fm\_score in (4,5) then 'Cant Lose Them'**

**when r\_score in (1 ) and fm\_score in (2) then 'Hibernating'**

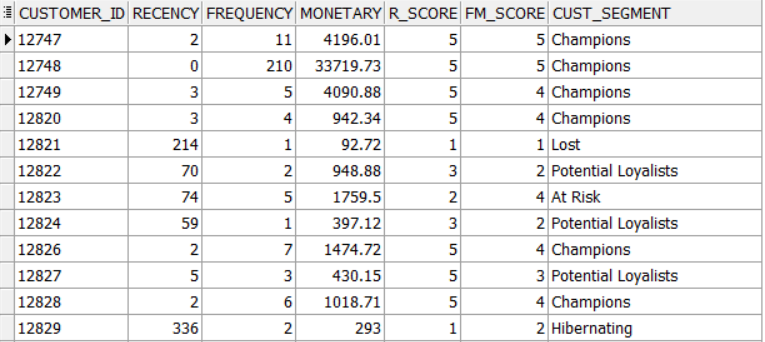
**when r\_score in (1 ) and fm\_score in (1) then 'Lost'**

**END AS Cust\_segment**

**FROM segment\_score\_fm**

**select Customer\_ID , Recency , Frequency , Monetary, R\_score , FM\_score , Cust\_Segment from final\_segment**

**order by customer\_id**



**Question Three (A)**

**select cust\_id , max(count) as max from (**

**select cust\_id , count(date) as count , grp from**

**(select cust\_id , date , row\_number () over(partition by cust\_id order by date) as d , DATE\_ADD(date, INTERVAL -(ROW\_NUMBER() OVER (partition by cust\_id ORDER BY date)) DAY) as grp**

**from daily\_purchase**

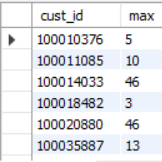
**group by cust\_id , date ) as x**

**group by grp , cust\_id**

**order by cust\_id**

**) as y**

**group by cust\_id**

****

**The Maximum number of consecutive days is 61 Days.**

**Question Three (B)**

I assumed that every row in the table is considered a transaction so to get the average number of transactions to reach the threshold of 250$ I need to get the average number of rows.

First I created a view to exclude all the customers that didn’t get the sum of 250$ in all of their transactions, this will help me in the join later.

**create view More\_than\_250 as (**

**select cust\_id , sum(amount) as total\_sum**

**from daily\_purchase**

**group by cust\_id**

**having sum(amount) >=250**

**)**

Then I created another view to calculate the running total for each customer.

**create view running as (**

**SELECT cust\_id,**

**date,**

**amount,**

**SUM(amount) OVER (PARTITION BY cust\_id ORDER BY date) AS sum\_cust**

**FROM daily\_purchase )**

Finally I joined both of the views in the inner query to have the running total for only the customers who reached the threshold then count the number of rows in the middle query, and calculate the average in the outer one.

**SELECT avg( t.num\_days )**

**FROM (**

**SELECT cust\_id, MIN(date) AS start\_date, COUNT(\*) + 1 AS num\_days**

**FROM (**

**select r.cust\_id , m.total\_sum , sum\_cust , r.date**

**from running r**

**join More\_than\_250 m**

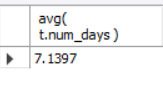
**on r.cust\_id = m.cust\_id**

**) t**

**WHERE sum\_cust < 250**

**GROUP BY cust\_id**

**) t**

****