Name: Salma Khaled Mohamed

Track : Data Management

Analytical Sql case study

My assumptions to solve cases

Case 1:

In this case: first I need get daily revenue: calculate that by recharge date of day minus recharge date of next day divided by amount recharge

But when do that: show last value of each customer is equal to zero because no value minus of it no lead date so, must calculate last value

Then get avg and stddev of revenue

Then show graph and need to compare lastvalue to avg and stddvv

To equation : y= avg+2\*stddev

A drawing on a piece of paper

Description automatically generated with medium confidence

Then calculate: Y=lastvalue-avg / stddev

But problem faced in this cases: stddev has zero value and cannot divide by it so do case when if greater than zero calculate y

Then calculate cases of graph

Results show repeated so, use choose first value of results graph then do ranking to get first row only .

Code formatted :

SELECT calendar\_dt,  
       cust\_id,  
       value  
FROM   (SELECT calendar\_dt,  
               cust\_id,  
               value,  
               **Row\_number**()  
                 over(  
                   PARTITION BY calendar\_dt, cust\_id, mvm\_status  
                   ORDER BY cust\_id) AS row1  
        FROM   (SELECT calendar\_dt,  
                       cust\_id,  
                       mvm\_status,  
                       **First\_value**(mvm\_status ignore nulls)  
                         over(  
                           PARTITION BY calendar\_dt, cust\_id, mvm\_status  
                           ORDER BY cust\_id) AS value  
                FROM   (SELECT calendar\_dt,  
                               cust\_id,  
                               y,  
                               CASE  
                                 WHEN y = 0THEN 'N'  
                                 WHEN y > 0  
                                      AND y < 1 THEN 'U'  
                                 WHEN y > 1 THEN 'HU'  
                                 WHEN y < 0  
                                      AND y >- 1 THEN 'R'  
                                 WHEN y >- 2  
                                      AND y <- 1 THEN 'HR'  
                               END AS MVM\_STATUS  
                        FROM   (SELECT calendar\_dt,  
                                       cust\_id,  
                                       CASE  
                                         WHEN stdd > 0 THEN ( lastvalue - avg )  
                                                            / stdd  
                                       END AS y  
                                FROM   (SELECT calendar\_dt,  
                                               cust\_id,  
                                               recharge\_dt,  
                                               recharge\_amt\_num,  
                                               lastvalue,  
                                               **Round**(**Avg**(daily\_revenue)  
                                                       over(  
                                                         PARTITION BY cust\_id),  
                                               2) AS  
                                               avg,  
                                               **Round**(**STDDEV**(daily\_revenue)  
                                                       over(  
                                                         PARTITION BY cust\_id),  
                                               2) AS  
                                               STDD  
                                        FROM   (SELECT calendar\_dt,  
                                                       cust\_id,  
                                                       recharge\_dt,  
                                                       recharge\_amt\_num,  
                                                       recharge\_amt\_num / (  
                                                       **Lead**(recharge\_dt, 1)  
                                                       over(  
                                                         PARTITION BY cust\_id  
                                                         ORDER BY cust\_id,  
                                                       recharge\_dt  
                                                       )  - recharge\_dt ) AS  
       daily\_revenue,  
       recharge\_amt\_num / ( calendar\_dt -  
       **Last\_value**(recharge\_dt  
       ignore  
       nulls)  
       over(  
       PARTITION BY cust\_id  
       ORDER BY cust\_id ) )  
       AS lastvalue  
       FROM   case1))))))  
WHERE  row1 = 1

Results :

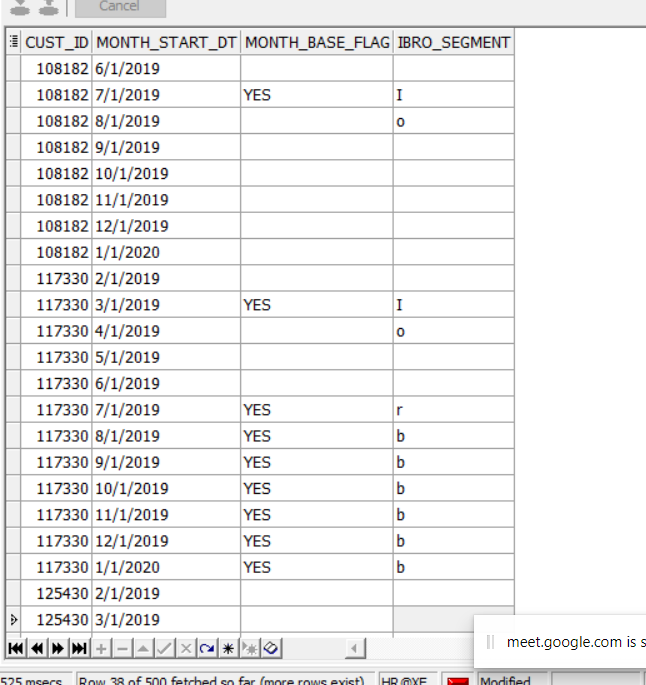
Table

Description automatically generated

Case 2:

SELECT cust\_id,  
       month\_start\_dt,  
       month\_base\_flag,  
       CASE  
         WHEN **SUM**(countno)  
                over(  
                  PARTITION BY cust\_id  
                  ORDER BY cust\_id ROWS BETWEEN 1 preceding AND CURRENT ROW) = 2  
       THEN  
         'b'  
         WHEN **SUM**(countno)  
                over(  
                  PARTITION BY cust\_id  
                  ORDER BY cust\_id ROWS BETWEEN unbounded preceding AND 1  
                preceding ) =  
              1  
              AND **SUM**(countno)  
                    over(  
                      PARTITION BY cust\_id  
                      ORDER BY cust\_id ROWS BETWEEN CURRENT ROW AND CURRENT ROW  
                    ) = 1  
       THEN 'r'  
         WHEN **SUM**(countno)  
                over(  
                  PARTITION BY cust\_id  
                  ORDER BY cust\_id ROWS BETWEEN 1 preceding AND 1 preceding ) =  
              1 THEN  
         'o'  
         WHEN firstval IS NOT NULL THEN 'I'  
       END AS IBRO\_SEGMENT  
FROM   (SELECT cust\_id,  
               month\_start\_dt,  
               month\_base\_flag,  
               **First\_value**(month\_base\_flag ignore nulls)  
                 over (  
                   PARTITION BY cust\_id  
                   ORDER BY cust\_id ROWS BETWEEN CURRENT ROW AND CURRENT ROW )  
               AS  
                      firstval,  
               **Count**(month\_base\_flag)  
                 over (  
                   PARTITION BY cust\_id, month\_start\_dt  
                   ORDER BY cust\_id, month\_start\_dt ROWS BETWEEN CURRENT ROW AND  
                 1  
                 following )  
               AS  
                      countno  
        FROM   case2  
        ORDER  BY cust\_id,  
                  month\_start\_dt)

Results :



Case 3 (a):

Sol:1

1)My assumptions

First : when solve problem get date difference between currentdate and before date then filter where diff equal 0ne that indicate each day then count difference and get max

Results :

Table

Description automatically generated

But , I know problem case sum all consecutive days not only diff 1

2) code formatted

SELECT cust\_id,  
       max\_consecutive\_days  
FROM   (SELECT cust\_id,  
               max\_consecutive\_days,  
               **Row\_number**()  
                 OVER(  
                   partition BY cust\_id, max\_consecutive\_days  
                   ORDER BY cust\_id) AS SELECT\_ONE\_ROW  
        FROM  (SELECT cust\_id,  
                      *Max*(c)  
                        OVER (  
                          partition BY cust\_id  
                          ORDER BY cust\_id ) AS Max\_Consecutive\_Days  
               FROM  (SELECT cust\_id,  
                             calendar\_dt,  
                             *Count*(diff)  
                               OVER (  
                                 partition BY cust\_id  
                                 ORDER BY calendar\_dt, cust\_id ) AS c  
                      FROM  (SELECT cust\_id,  
                                    calendar\_dt,  
                                    diff  
                             FROM   (SELECT cust\_id,  
                                            calendar\_dt,  
                                            calendar\_dt - **Lag** (calendar\_dt)  
                                            OVER (  
                                              partition BY cust\_id  
                                              ORDER BY calendar\_dt ) AS  
                                            diff  
                                     FROM   case3))  
                      WHERE  diff = 1)))  
WHERE  select\_one\_row = 1

Sol2: but it fit in first case but wrong in other cases , use flag to indicate not 1 day diff but not fit when data is null

SELECT cust\_id,  
       calendar\_dt,  
       NEW  
FROM  (SELECT cust\_id,  
              calendar\_dt,  
              **Max**(pp)  
                over (  
                  PARTITION BY cust\_id  
                  ORDER BY calendar\_dt, cust\_id )AS NEW  
       FROM  (SELECT cust\_id,  
                     calendar\_dt,  
                     a,  
                     pp,  
                     CASE  
                       WHEN **Lead**(a)  
                              over (  
                                PARTITION BY cust\_id  
                                ORDER BY calendar\_dt, cust\_id ) IS NULL THEN  
                       'yes'  
                     END AS firstlead  
              FROM  (SELECT cust\_id,  
                            calendar\_dt,  
                            a,  
                            **Count**(e)  
                              over (  
                                PARTITION BY cust\_id  
                                ORDER BY calendar\_dt, cust\_id ROWS BETWEEN  
                              CURRENT ROW  
                              AND  
                              unbounded  
                              following ) AS pp  
                     FROM  (SELECT cust\_id,  
                                   calendar\_dt,  
                                   a,  
                                   **Lead**(w, 1)  
                                     over (  
                                       PARTITION BY cust\_id  
                                       ORDER BY calendar\_dt, cust\_id ) AS e  
                            FROM  (SELECT cust\_id,  
                                          calendar\_dt,  
                                          a,  
                                          **Max**(a)  
                                            over (  
                                              PARTITION BY cust\_id  
                                              ORDER BY calendar\_dt, cust\_id ) AS  
                                          w  
                                   FROM   (SELECT cust\_id,  
                                                  calendar\_dt,  
                                                  CASE  
                                                    WHEN diff = 1 THEN **Count**(  
                                                    diff)  
                                                    over (  
                                                      PARTITION BY cust\_id  
                                                      ORDER BY calendar\_dt,  
                                                    cust\_id )  
                                                  END AS a  
                                           FROM   (SELECT cust\_id,  
                                                          calendar\_dt,  
                                                          calendar\_dt - **Lag** (  
                                                          calendar\_dt)  
                                                          over (  
                                                            PARTITION BY cust\_id  
                                                            ORDER BY calendar\_dt  
                                                          )  
                                                          AS diff  
                                                   FROM   case3))))))  
       WHERE  firstlead IS NOT NULL)  
GROUP  BY cust\_id,  
          calendar\_dt,  
          NEW  
ORDER  BY cust\_id

Sol 3 : to solve my problem in first sol and second sol , think about ranking to rank value then get diff between current value and no in row number such as : my current row no 5 and date 5/10/2019 then diff equal 1/10/2019 , I do that in all days , results all days have 1 difference = 1/1/2019 is first value but diff greater than 1 results not equal first date then count and get max , results is correct now(right sol)

SELECT cust\_id,  
       max\_consecutive\_days  
FROM   (SELECT cust\_id,  
               max\_consecutive\_days,  
               **Row\_number**()  
                 OVER(  
                   partition BY cust\_id, max\_consecutive\_days  
                   ORDER BY cust\_id) AS row1  
        FROM   (SELECT cust\_id,  
                       *Max*(max1)  
                         OVER (  
                           partition BY cust\_id  
                           ORDER BY cust\_id ) AS Max\_Consecutive\_Days  
                FROM   (SELECT cust\_id,  
                               calendar\_dt,  
                               *Count*(val)  
                                 OVER (  
                                   partition BY cust\_id, val  
                                   ORDER BY cust\_id, calendar\_dt ) AS max1  
                        FROM   (SELECT cust\_id,  
                                       calendar\_dt,  
                                       ( calendar\_dt - ranking ) AS val  
                                FROM   (SELECT cust\_id,  
                                               calendar\_dt,  
                                               **Row\_number**()  
                                                 OVER (  
                                                   partition BY cust\_id  
                                                   ORDER BY cust\_id, calendar\_dt  
                                                 ) AS  
                                               ranking  
                                        FROM   case3)))))  
WHERE  row1 = 1

Case 3 b:

SELECT cust\_id,  
       calendar\_dt,  
       amt\_le,  
       total,  
       mycase,  
       daydiff,  
       **Avg**(daydiff)  
         over(  
           PARTITION BY cust\_id)  
FROM  (SELECT cust\_id,  
              calendar\_dt,  
              amt\_le,  
              total,  
              mycase,  
              **SUM**(CASE  
                    WHEN mycase IS NULL THEN 1  
                    ELSE 0  
                  END)  
                over(  
                  PARTITION BY cust\_id) AS daydiff  
       FROM  (SELECT cust\_id,  
                     calendar\_dt,  
                     amt\_le,  
                     total,  
                     CASE  
                       WHEN total > 250 THEN 1  
                     END AS mycase  
              FROM   (SELECT cust\_id,  
                             calendar\_dt,  
                             amt\_le,  
                             **SUM**(amt\_le)  
                               over (  
                                 PARTITION BY cust\_id  
                                 ORDER BY calendar\_dt ROWS BETWEEN unbounded  
                               preceding  
                               AND  
                               CURRENT  
                               ROW) AS  
                             total  
                      FROM   case3)))

My assumptions : get sum of amount then calculate cumulative sum of amount then cases if greater 250 threshold then is number of days I need sum all 0 not 1 to reach to threshold : ,then calc avg but I didn’t reach correct avg value or median value