Identify groups of consecutive years of purchase – static technique

PUBLISHED MAY 29, 2023 BY FRANCESCO BERGAMASCHI

In order to take Marketing decisions, it is often necessary to identify the behavior of customers during the purchase phase. In other cases, purchase frequency is calculated, in others consecutive purchases, and so on.

In a recent small project for a client, a request arose to identify and count:

1. the groups of consecutive years in which a customer made at least one purchase;
2. the customers who have bought consecutively for a given number of years.

The request may seem, at first instance, not complex but this is not the case: as we will see, a fairly advanced DAX is needed.

In this article we will deal with solving the problem in a static way, i.e., using a calculated column – where most of the complexity will reside – and some measures. If the topic is of interest to the readers, we will also publish the dynamic version, where everything is solved with measures. The static version, in fact, has some limits that are overcome by the dynamic version.

The data model is visible in figure 1, a classic star schema.

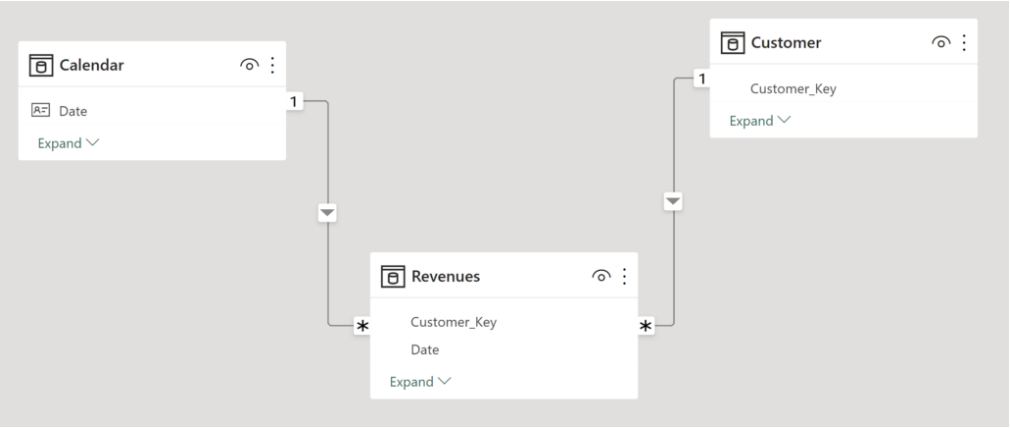

Figure 1

The *Revenues* table is a fact table, shown in figure 2. The other two tables (*Customer* and *Calendar*) are dimension tables with primary key.

Date	Customer_Key	Total_Revenue
01/gen/2020	1	134
01/gen/2021	1	102
01/gen/2022	1	124
01/gen/2024	1	190
01/gen/2025	1	100
01/gen/2023	2	166
01/gen/2024	2	151
01/gen/2021	3	118
01/gen/2023	3	123
01/gen/2025	3	154
01/gen/2027	1	134
01/gen/2028	1	102
01/gen/2029	1	124

Figure 2

The report to be obtained is shown in figure 3.

Customer_Key	Consecutive Years Col	Number of Events Static
1	3	2
1	2	1
2	2	1
3	1	3
Total		**7**

Consecutive Years Col	Number of Customers Static	Number of Customers Static Pct
1	3	50,0%
2	2	33,3%
3	1	16,7%
Total	**6**	**100,0%**

Figure 3

The matrix on the left in figure 3 shows, customer by customer (Customer_Key), the groups of consecutive years of purchase (Consecutive Years Col) and the number of each of these groups (Number of Events Static). For example, for customer 1, two lines highlighted in red are shown: in the first it can be seen that customer 1 has two groups of 3 consecutive years (in figure 2 it can be seen that they are 2020, 2021 and 2022 as the first group and 2027, 2028 and 2029 as the second), in the second it can be seen that customer 1 has a group of two consecutive years (again in figure 2 it can be seen that they are 2024 and 2025).

On the other hand, the matrix on the right in figure 3 shows, group of consecutive years by group of consecutive years (1, 2 or 3 given the data), how many customers, in absolute terms and as a percentage of the total, have had at least one group of years corresponding consecutive numbers (***Number of Customers Static*** and ***Number of Customers Static Pct***). For example, in the row highlighted in red, it is shown that two customers (in figure 2 it can be seen that this is customer 1 in the years 2024 and 2025 and customer 2 in the years 2023 and 2024) had at least one group of two years consecutive purchases. The total value of 6 for the ***Number of Customers Static*** measure needs clarification: the customer's request was explicitly to consider each ***customer - group of consecutive years*** pair as a customer. The number of distinct customers is actually 3.

The limits of the static technique that we will show here reside in the non-reactivity of the report in figure 3 to any applied filters.

Discussion

To solve the problem in a static way, we resort to a calculated column of the *Revenues* table: *Revenues[Consecutive Years Col]*. For convenience, we also created a *Revenues[Year]* calculated column.

The *Revenues[Consecutive Years Col]* column must indicate, at the end of any consecutive block of purchase years for a customer, the number of consecutive years (figure 4, which highlights the blocks of customer 1).

Date	Customer_Key	Total_Revenue	Year	Consecutive Years Col
01/gen/2020	1	134	2020	
01/gen/2021	1	102	2021	
01/gen/2022	1	124	2022	3
01/gen/2024	1	190	2024	
01/gen/2025	1	100	2025	2
01/gen/2023	2	166	2023	
01/gen/2024	2	151	2024	2
01/gen/2021	3	118	2021	1
01/gen/2023	3	123	2023	1
01/gen/2025	3	154	2025	1
01/gen/2027	1	134	2027	
01/gen/2028	1	102	2028	
01/gen/2029	1	124	2029	3

Figure 4

Here is the code for the *Revenues[Year]* calculated column and the most important calculated column for the static solution, *Revenues[Consecutive Years Col]*:

```
Year =
YEAR ( Revenues[Date] )

Consecutive Years Col =
VAR CurrentCust = Revenues[Customer_Key]
VAR CurrentDate = Revenues[Date]
VAR ConsYearTab =
   ADDCOLUMNS (
      ADDCOLUMNS (
         FILTER ( Revenues, Revenues[Customer_Key] = CurrentCust ),
         "@NearestLowerHole",
            VAR RefYear = Revenues[Year]
            VAR AllYears =
               DISTINCT (
                  UNION (
                     ALL ( Revenues[Year] ),
                     GENERATESERIES ( MIN ( Revenues[Year] ) – 1, MAX ( Revenues[Ye
```

```
ar] ) + 1, 1 )
                )
            )
        VAR YearsWithSales =
            CALCULATETABLE (
                VALUES ( Revenues[Year] ),
                Revenues[Customer_Key] = CurrentCust,
                REMOVEFILTERS ( Revenues )
            )
        VAR YearsWithNoSales =
            EXCEPT ( AllYears, YearsWithSales )
        VAR NearestHole =
            MAXX ( FILTER ( YearsWithNoSales, Revenues[Year] <= RefYear ), Reven
ues[Year] )
        RETURN
            NearestHole,
    "@NearestHigherHole",
        VAR RefYear = Revenues[Year]
        VAR AllYears =
            DISTINCT (
                UNION (
                    ALL ( Revenues[Year] ),
                    GENERATESERIES ( MIN ( Revenues[Year] ) - 1, MAX ( Revenues[Ye
ar] ) + 1, 1 )
                )
            )
        VAR YearsWithSales =
            CALCULATETABLE (
                VALUES ( Revenues[Year] ),
                Revenues[Customer_Key] = CurrentCust,
                REMOVEFILTERS ( Revenues )
            )
        VAR YearsWithNoSales =
            EXCEPT ( AllYears, YearsWithSales )
        VAR NearestHole =
            MINX ( FILTER ( YearsWithNoSales, Revenues[Year] > RefYear ), Revenu
es[Year] )
        RETURN
            NearestHole
    ),
    "@ConsYears", Revenues[Year] - [@NearestLowerHole]
)
```

```
VAR ConsYearTabWithMax =
   ADDCOLUMNS (
      ConsYearTab,
      "@MaxConsYears",
         VAR CurrentLowerHole = [@NearestLowerHole]
         VAR CurrentHigherHole = [@NearestHigherHole]
         RETURN
            MAXX (
               FILTER (
                  ConsYearTab,
                  Revenues[Year] >= CurrentLowerHole
                     && Revenues[Year] < CurrentHigherHole
               ),
               [@ConsYears]
            )
   )
VAR CurrentCustDateConsYears =
   SELECTCOLUMNS (
      FILTER ( ConsYearTabWithMax, Revenues[Date] = CurrentDate ),
      "@@ConsYears", [@ConsYears]
   )
VAR MaxConsYearsCurrentCust =
   SELECTCOLUMNS (
      FILTER ( ConsYearTabWithMax, Revenues[Date] = CurrentDate ),
      "@@MaxConsYears", [@MaxConsYears]
   )
RETURN
   IF (
      CurrentCustDateConsYears = MaxConsYearsCurrentCust,
      CurrentCustDateConsYears
   )
```

As can be seen, the *Revenues[Consecutive Years Col]* code is far from trivial.

Let's deal with a brief code illustration of the calculated column *Revenues[Consecutive Years Col]*:

- first, the code stores the date and customer of the current row in two variables (*CurrentCust* and *CurrentDate*);
- subsequently a table is created in memory, *ConsYearTab*, in which, row by row on the table itself, pre-selected for the current customer

via *FILTER*, three values are calculated: the closest year without purchases, both in the past and in the future, compared to the considered transaction year (*[@NearestLowerHole]* and *[@NearestHigherHole]*) and the number of consecutive years (*[@ConsYears]*) as difference between *Revenues[Year]* and *[@NearestLowerHole]*;

- subsequently another table is created in memory, *ConsYearTabWithMax*, which is none other than the *ConsYearTab* table with the addition of a calculated column (*[@MaxConsYears]*) which calculates the maximum values of the column *[@ConsYears]* with the constraint of consider only *Revenues[Year]* values that are greater than or equal to *[@NearestLowerHole]* and less than *[@NearestHigherHole]*;
- subsequently the *[@ConsYears]* column of *ConsYearTabWithMax* is selected, binding the date to be that of the currently iterated row, to have the value of years consecutive to the current row. This column has only one row – remember that the customer has already been selected in the *ConsYearTab* – and thus we obtain a scalar value, *CurrentCustDateConsYears*;
- subsequently the *[@MaxConsYears]* column of *ConsYearTabWithMax* is selected, still constraining the date to be that of the currently iterated row, to have the maximum value of the consecutive years of the rows belonging to the group to which the current row belongs – identified by the current customer and by values of *[@NearestLowerHole]* and *[@NearestHigherHole]*. Again, this column has only one row – remember that the customer has already been selected in the ConsYearTab – resulting in a scalar value, *MaxConsYearsCurrentCust*;
- finally, if *CurrentCustDateConsYears* is equal to *MaxConsYearsCurrentCust*, *CurrentCustDateConsYears* is returned, so that we have the value only at the end of the current block of consecutive years for the current customer. Having the values also in the other rows would have caused grouping problems in the report.

Finally, the measures needed to create the report are (as shown in figure 3):

```
Number of Events Static =
COUNTROWS ( Revenues )
```

```
Number of Customers Static =
SUMX (
    ADDCOLUMNS (
        VALUES ( Revenues[Consecutive Years Col] ),
        "@Num",
            VAR ConsYears = Revenues[Consecutive Years Col]
            RETURN
                CALCULATE (
                    DISTINCTCOUNT ( Revenues[Customer_Key] ),
                    Revenues[Consecutive Years Col] >= ConsYears
                )
    ),
    [@Num]
)

Number of Customers Static Pct =
DIVIDE (
    [Number of Customers Static],
    CALCULATE (
        [Number of Customers Static],
        ALLSELECTED ( Revenues[Consecutive Years Col] )
    )
)
```

Let us now illustrate the limits of this solution, which is perfect until filters are introduced in the years to be considered. Look at figure 5 where the calendar year 2020 has been excluded. One would expect to see the number of customer 1's three-year groups drop from 2 (figure 3) to 1 because the group of years 2020, 2021 and 2022 should now consist of two years (2021 and 2022) and the number of customer 1's two-year groups grow from 1 (figure 3, again) to 2. However, this being a static solution, nothing changes in the report.

Customer _Key	Consecutive Years Col	Number of Events Static
1	3	2
1	2	1
2	2	1
3	1	3
Total		**7**

Consecutive Years Col	Number of Customers Static	Number of Customers Static Pct
1	3	50,0%
2	2	33,3%
3	1	16,7%
Total	**6**	**100,0%**

Figure 5

Conclusions

The report is complex to produce, although the request that generated it may seem simple at first glance. The calculated column *Revenues[Consecutive Years Col]*, which represents the key to the problem, allows grouping by number of consecutive years but, being a calculated column, it is static and its values are not sensitive to report filters. If you want to get interaction with the report you have to go towards a dynamic solution, based exclusively on measures. If we receive a request, we will also show this solution in a future article.

.pbix file Download