**Building DATE TABLE**

. The sample files available for this pattern include four different scenarios for the *Date* table:

1. One row for each date based on the Gregorian calendar, using *Date* as the primary key. In this case, the behavior is close to the standard time intelligence calculations, with the noticeable difference that the formulas are faster.
2. One row for each month based on the Gregorian calendar, using *Year Month Number* as the primary key. This pattern is even better than the previous one, because the date table is significantly smaller.
3. One row for each month in an accounting calendar with 13 fiscal months, where the 13th fiscal month is projected as an additional month in the Gregorian calendar between the last month of a fiscal year and the first month of the following fiscal year. Performance is close to that of the second pattern.
4. One row for each month in an accounting calendar with 13 fiscal months, where the 13th fiscal month is projected in the last fiscal month on the Gregorian calendar. Performance and behavior are very close to what is observed in the third example.
5. Date =
6. VAR FirstFiscalMonth = 3  -- First month of the fiscal year
7. VAR MonthsInYear = 12     -- Must be 12 for GranularityByDate
8. -- can be different for GranularityByMonth
9. VAR CalendarFirstDate = MIN ( Sales[Order Date]  )
10. VAR CalendarLastDate = MAX ( Sales[Order Date]  )
11. VAR CalendarFirstYear = YEAR ( CalendarFirstDate )
12. VAR CalendarFirstMonth = MONTH ( CalendarFirstDate )
13. VAR CalendarLastYear = YEAR ( CalendarLastDate )
14. VAR CalendarLastMonth = MONTH ( CalendarLastDate )
16. -------------------------
17. -- Internal calculations
18. -------------------------
19. VAR GranularityByDate =
20. ADDCOLUMNS (
21. CALENDAR (
22. DATE ( CalendarFirstYear, CalendarFirstMonth, 1 ),
23. EOMONTH (
24. DATE ( CalendarLastYear, CalendarLastMonth, 1 ),
25. 0
26. )
27. ),
28. "Year Month Number", YEAR ( [Date] ) \* MonthsInYear
29. + MONTH ( [Date] ) - 1
30. )
31. VAR GranularityByMonth =
32. SELECTCOLUMNS (
33. GENERATESERIES (
34. CalendarFirstYear \* MonthsInYear + CalendarFirstMonth - 1
35. - (MonthsInYear - 12) \* (CalendarFirstMonth < FirstFiscalMonth),
36. CalendarLastYear \* MonthsInYear + CalendarLastMonth - 1
37. - (MonthsInYear - 12) \* (CalendarLastMonth < FirstFiscalMonth),
38. 1
39. ),
40. "Year Month Number", [Value]
41. )
42. RETURN GENERATE (
43. GranularityByDate,       -- Use GranularityByMonth to get one row for each month
44. VAR YearMonthNumber = [Year Month Number]
45. VAR FiscalMonthNumber =
46. MOD (
47. YearMonthNumber + 1
48. \* (FirstFiscalMonth > 1)
49. \* (MonthsInYear + 1 - FirstFiscalMonth),
50. MonthsInYear
51. ) + 1
52. VAR FiscalYearNumber =
53. QUOTIENT (
54. YearMonthNumber + 1
55. \* (FirstFiscalMonth > 1)
56. \* (MonthsInYear + 1 - FirstFiscalMonth),
57. MonthsInYear
58. )
59. VAR OffsetFiscalMonthNumber = MonthsInYear + 1 - (MonthsInYear - 12)
60. VAR MonthNumber =
61. IF (
62. FiscalMonthNumber <= 12 && FirstFiscalMonth > 1,
63. FiscalMonthNumber + FirstFiscalMonth
64. - IF (
65. FiscalMonthNumber > (OffsetFiscalMonthNumber - FirstFiscalMonth),
66. OffsetFiscalMonthNumber,
67. 1
68. ),
69. FiscalMonthNumber
70. )
71. VAR YearNumber = FiscalYearNumber - 1 \* (MonthNumber > FiscalMonthNumber)
72. VAR YearMonthKey = YearNumber \* 100 + MonthNumber
74. VAR MonthDate = DATE ( YearNumber, MonthNumber, 1 )
75. VAR FiscalQuarterNumber = MIN ( ROUNDUP ( FiscalMonthNumber / 3, 0 ), 4 )
76. VAR FiscalYearQuarterNumber = FiscalYearNumber \* 4 + FiscalQuarterNumber - 1
77. VAR FiscalMonthInQuarterNumber =
78. MOD ( FiscalMonthNumber - 1, 3 ) + 1 + 3 \* (MonthNumber > 12)
79. VAR MonthInQuarterNumber = MOD ( MonthNumber - 1, 3 ) + 1 + 3 \* (MonthNumber > 12)
80. VAR QuarterNumber = MIN ( ROUNDUP ( MonthNumber / 3, 0 ), 4 )
81. VAR YearQuarterNumber = YearNumber \* 4 + QuarterNumber - 1
82. RETURN ROW (
83. "Year Month Key", YearMonthKey,
84. "Year", YearNumber,
85. "Year Quarter", FORMAT ( QuarterNumber, "\Q0" )
86. & "-" & FORMAT ( YearNumber, "0000" ),
87. "Year Quarter Number", YearQuarterNumber,
88. "Quarter", FORMAT ( QuarterNumber, "\Q0" ),
89. "Year Month", IF (
90. MonthNumber > 12,
91. FORMAT ( MonthNumber, "\M00" ) & FORMAT ( YearNumber, " 0000" ),
92. FORMAT ( MonthDate, "mmm yyyy" )
93. ),
94. "Month", IF (
95. MonthNumber > 12,
96. FORMAT ( MonthNumber, "\M00" ),
97. FORMAT ( MonthDate, "mmm" )
98. ),
99. "Month Number", MonthNumber,
100. "Month In Quarter Number", MonthInQuarterNumber,
101. "Fiscal Year", FORMAT ( FiscalYearNumber, "\F\Y 0000" ),
102. "Fiscal Year Number", FiscalYearNumber,
103. "Fiscal Year Quarter", FORMAT ( FiscalQuarterNumber, "\F\Q0" ) & "-"
104. & FORMAT ( FiscalYearNumber, "0000" ),
105. "Fiscal Year Quarter Number", FiscalYearQuarterNumber,
106. "Fiscal Quarter", FORMAT ( FiscalQuarterNumber, "\F\Q0" ),
107. "Fiscal Month", IF (
108. MonthNumber > 12,
109. FORMAT ( MonthNumber, "\M00" ),
110. FORMAT ( MonthDate, "mmm" )
111. ),
112. "Fiscal Month Number", FiscalMonthNumber,
113. "Fiscal Month In Quarter Number", FiscalMonthInQuarterNumber
114. )
115. )

In order to obtain the correct visualization, the **calendar columns** must be configured in the data model as follows. For each column we show the data type followed by a sample value assuming a fiscal month starting in March where there are 12 months in the fiscal year:

* *Date*: Date, Hidden (8/14/2007), used only for the first scenario
* *Year Month Key*: Whole Number, Hidden (200708), used to define relationships
* *Year Month*: Text (Aug 2007)
* *Year Quarter*: Text (Q3-2007)
* *Year Quarter Number*: Whole Number, Hidden (8030)
* *Quarter*: Text (Q3)
* *Year Month Number*: Whole Number, Hidden (24091)
* *Month*: Text (Aug)
* *Month Number*: Whole Number, Hidden (8)
* *Month In Quarter Number*: Whole Number, Hidden (2)
* *Fiscal Month*: Text (Aug)
* *Fiscal Month Number*: Whole Number, Hidden (6)
* *Fiscal Month in Quarter Number*: Whole Number, Hidden (3)
* *Fiscal Year*: Text (FY 2008)
* *Fiscal Year Number*: Whole Number, Hidden (2008)
* *Fiscal Year Quarter*: Text (FQ2-2008)
* *Fiscal Year Quarter* Number: Whole Number, Hidden (8033)
* *Fiscal Quarter*: Text (FQ2)

The *Date* table in this pattern has four hierarchies:

* Fiscal Year-Quarter: Year (*Fiscal Year*), Quarter (*Fiscal Year Quarter*), Month (*Year Month*)
* Fiscal Year-Month: Year (*Fiscal Year*), Month (*Year Month*)
* Year-Quarter: Year (*Year*), Quarter (*Year Quarter*), Month (*Year Month*)
* Year-Month: Year (*Year*), Month (*Year Month*)
* Several columns serve the only purpose of simplifying the formulas used in custom time-related calculations. The Year Month Key column is only used to define a relationship with the Sales table using an integer in the format YYYYMM. This numeric format to identify a month is common in many data sources that manage data at the month granularity.
* The Date table has only the range of months required by the data available. For example, in the example the Date table includes only the months from March 2007 to August 2009. This pattern does not come with the constraint of including all the months in one year. For this reason, there is no need for additional calculated columns like the DateWithSales used in the [**Standard time-related calculations**](https://www.daxpatterns.com/standard-time-related-calculations/) pattern.

Or:

Date =

VAR FirstFiscalMonth = 1      -- First month of the fiscal year

VAR FirstDayOfWeek = 0        -- 0 = Sunday, 1 = Monday, ...

VAR FirstSalesDate = MIN ( Sales[Order Date] )

VAR LastSalesDate = MAX ( Sales[Order Date] )

VAR TypeStartFiscalYear = 1    -- Fiscal year as Calendar Year of :

                                -- 0 - First day of fiscal year

                                -- 1 - Last day of fiscal year

VAR QuarterWeekType = "445" -- Supports only "445", "454", and "544"

VAR WeeklyType = "Last" -- Use: "Nearest" or "Last"

-- The remaining code of the calculated table is included in the sample file

* *ate*: Date, m/dd/yyyy (8/14/2007), used as a column to mark as date table, which is optional
* *Sequential Day Number*: Whole Number, Hidden (40040) , same value of Date as integer
* *Fiscal Year*: Text (FY 2007)
* *Fiscal Year Number*: Whole Number, Hidden (2007)
* *Fiscal Quarter*: Text (FQ3)
* *Fiscal Quarter Number*: Whole Number, Hidden (3)
* *Fiscal Year Quarter*: Text (FQ3-2007)
* *Fiscal Year Quarter* *Number*: Whole Number, Hidden (8030)
* *Fiscal Week*: Text (FW33)
* *Fiscal Week Number*: Whole Number, Hidden (33)
* *Fiscal Year Week*: Text (FW33-2007)
* *Fiscal Year Week Number*: Whole Number, Hidden (5564)
* *Fiscal Month*: Text (FM Aug)
* *Fiscal Month Number*: Whole Number, Hidden (8)
* *Fiscal Year Month*: Text (FM Aug 2007)
* *Fiscal Year Month Number*: Whole Number, Hidden (24091)
* *Day of Fiscal Month Number*: Whole Number, Hidden (17)
* *Day of Fiscal Quarter Number*: Whole Number, Hidden (45)
* *Day of Fiscal Year Number*: Whole Number, Hidden (227)

We want to introduce the concept of **filter-safe columns**. In a table, there are columns whose filters need to be preserved. The filters over filter-safe columns are not altered by the time intelligence calculations. They will be affecting the calculations presented in this pattern. The filter-safe columns in our sample table are the following:

* *Day of Week*: ddd (Tue)
* *Day of Week Number*: Whole Number, Hidden (6)
* *Working Day*: Text (Working Day)

We provide a more in-depth description of the behavior of filter-safe columns in the next section.

The *Date* table in this pattern contains several hierarchies:

* Year-Month-Week: Year (*Fiscal Year*), Month (*Fiscal Year Month*), Week (*Fiscal Year Week*)
* Year-Quarter-Month-Week: Year (*Fiscal Year*), Quarter (*Fiscal Year Quarter*), Month (*Fiscal Year Month*), Week (*Fiscal Year Week*)
* Year-Quarter-Week: Year (*Fiscal Year*), Quarter (*Fiscal Year Quarter*), Week (*Fiscal Year Week*)
* Year- Week: Year (*Fiscal Year*), Week (*Fiscal Year Week*)

The columns are designed to simplify the formulas. For example, the *Day of Fiscal Year Number* column contains the number of days since the beginning of the fiscal year; this number makes it easier to find a corresponding range of dates in the previous year.

The *Date* table must also include a hidden *DateWithSales* calculated column, used by some of the formulas of this pattern:

DateWithSales =

'Date'[Date] <= MAX ( Sales[Order Date] )

The *Date[DateWithSales]* column is *TRUE* if the date is on or before the last date with sales; it is *FALSE* otherwise. In other words, *DateWithSales* is *TRUE* for “past” dates and *FALSE* for “future” dates, where “past” and “future” are relative to the last date with sales.

In case you import a *Date* table, you want to create columns that are similar to the ones we describe in this pattern, in that they should behave the same way.