Excel DAX - Time Intelligence

DAX has an important and powerful feature, referred to as **Time Intelligence**. Time intelligence enables you to write DAX formulas that refer to the time periods for use in the PivotTables.

DAX has 35 time-intelligence functions specifically for aggregating and comparing data over time. However, these DAX functions have some constraints on the data that you need to understand and work with caution to avoid errors.

Why Time Intelligence Makes DAX Powerful?

The time intelligence functions work with data that is constantly changing, depending on the context you select in PivotTables and Power View visualizations. As you are aware, most of the data analysis involves summarization of data over time periods, comparing data values across the time periods, understanding the trends and making decisions based on future projections.

For example, you might want to sum sales amounts for the past month product-wise and compare the totals with those of other months in the fiscal year. This means, you have to use the dates as a way to group and aggregate sales transactions for a particular period in time.

This is where you can observe the power of DAX. You can use DAX time intelligence functions to define calculated fields that help you in analyzing the data over time, without having to change the date selections in the pivot tables. This makes your job easier. Moreover, you can build PivotTables that would not be possible any other way.

Requirements for DAX Time Intelligence Functions

DAX time intelligence functions have certain requirements. If these requirements are not met, you might get errors or they may not work properly. Hence, you can refer to these requirements as rules or constraints as well. Following are certain DAX time intelligence functions requirements/rules/constraints −

- You need to have a date table in your Data Model.
- The date table must include a column considered to be the Date column by DAX. You can name the column the way you want, but it should comply with the following conditions: o The date column should contain a contiguous set of dates that covers every day in the time period you are analyzing the data.
  - Every date must exist once and only once in the date column.
  - You cannot skip any dates (For e.g. you cannot skip weekend dates).
- DAX time intelligence functions work only on a standard calendar and assume the start of the year as January 1 and the end of the year as December 31, with the months in the year and days in each month as of a calendar year.

However, you can customize a standard calendar for different financial years. It is a good practice to verify the above requirements before any time intelligence function is used.

For more details on date tables and their usage in DAX formulas, refer to the tutorial = Data Modeling with DAX in this tutorials library.

DAX Time Intelligence Functions – Categories

DAX Time Intelligence functions can be categorized as follows −

- DAX functions that return a single date.
- DAX functions that return a table of dates.
- DAX functions that evaluate expressions over a time period.

DAX Functions That Return a Single Date

DAX functions in this category return a single date.

There are 10 DAX functions in this category −

Sr.No.	DAX Function & Return Value
1	**FIRSTDATE (Date_Column)** Returns the first date in the Date_Column in the current context.
2	**LASTDATE (Date_Column)** Returns the last date in the Date_Column in the current context.
3	**FIRSTNONBLANK (Date_Column, Expression)** Returns the first date where an expression has a non-blank value.
4	**LASTNONBLANK (Date_Column, Expression)** Returns the last date where an expression has a non-blank value.
5	**STARTOFMONTH (Date_Column)** Returns the first date of a month in the current context.
6	**ENDOFMONTH (Date_Column)** Returns the last date of a month in the current context.
7	**STARTOFQUARTER (Date_Column)** Returns the first date of a quarter in the current context.
8	**ENDOFQUARTER (Date_Column)** Returns the last date of a quarter in the current context.
9	**STARTOFYEAR (Date_Column, [YE_Date])** Returns the first date of a year in the current context.
10	**ENDOFYEAR (Date_Column, [YE_Date])** Returns the last date of a year in the current context.

DAX Functions That Return a Table of Dates

DAX Functions in this category return a table of dates. These functions will be mostly used as a SetFilter argument to the DAX function - CALCULATE.

There are 16 DAX functions in this category. Eight (8) of these DAX functions are the "previous" and "next" functions.

- The "previous" and "next" functions start with a date column in the current context and calculate the previous or next day, month, quarter or year.
- The 'previous' functions work backward from the first date in the current context and the "next" functions move forward from the last date in the current context.
- The "previous" and "next" functions return the resulting dates in the form of a single column table.

Sr.No.	DAX Function & Return Value
1	**PREVIOUSDAY (Date_Column)** Returns a table that contains a column of all dates representing the day that is previous to the first date in the Date_Column in the current context.
2	**NEXTDAY (Date_Column)** Returns a table that contains a column of all dates from the next day, based on the first date specified in the Date_Column in the current context.
3	**PREVIOUSMONTH (Date_Column)** Returns a table that contains a column of all dates from the previous month, based on the first date in the Date_Column in the current context.
4	**NEXTMONTH (Date_Column)** Returns a table that contains a column of all dates from the next month, based on the first date in the Date_Column in the current context.
5	**PREVIOUSQUARTER (Date_Column)** Returns a table that contains a column of all dates from the previous quarter, based on the first date in the Date_Column in the current context.
6	**NEXTQUARTER (Date_Column)** Returns a table that contains a column of all dates in the next quarter, based on the first date specified in the Date_Column in the current context.
7	**PREVIOUSYEAR (Date_Column, [YE_Date])** Returns a table that contains a column of all dates from the previous year, given the last date in the Date_Column in the current context.
8	**NEXTYEAR (Date_Column, [YE_Date])** Returns a table that contains a column of all dates in the next year, based on the first date in the Date_Column in the current context.

Four (4) DAX functions calculate a set of dates in a period. These functions perform the calculations using the last date in the current context.

Sr.No.	DAX Function & Return Value
1	**DATESMTD (Date_Column)** Returns a table that contains a column of the dates for the month to date, in the current context.
2	**DATESQTD (Date_Column)** Returns a table that contains a column of the dates for the quarter to date, in the current context.
3	**DATESYTD (Date_Column, [YE_Date])** Returns a table that contains a column of the dates for the year to date, in the current context.
4	**SAMEPERIODLASTYEAR (Date_Column)** Returns a table that contains a column of dates shifted one year back in time from the dates in the specified Date_Column, in the current context. **Note**– SAMEPERIODLASTYEAR requires that the current context contains a contiguous set of dates. If the current context is not a contiguous set of dates, then SAMEPERIODLASTYEAR will return an error.

- Four (4) DAX functions are used to shift from the set of dates that are in the current context to a new set of dates.

These DAX functions are more powerful than the previous ones.

- DAX functions – DATEADD, DATESINPERIOD and PARALLELPERIOD shift some number of time intervals from the current context. The interval can be day, month, quarter or year, represented by the key words – DAY, MONTH, QUARTER and YEAR respectively.

  For example:

- Shift backward by 2 days.
- Move forward by 5 months.
- Move forward by one month from today.
- Go back to same quarter in the last year.

If the function argument - number of intervals (integer value) is positive, shift is forward and if it is negative, shift is backward.

- DAX function – DATESBETWEEN calculates the set of dates between the specified start date and the end date.

Sr.No.	DAX Function & Return Value
1	**DATEADD (Date_Column, Number_of_Intervals, Interval)** Returns a table that contains a column of dates, shifted either forward or backward in time by the specified number of intervals from the dates in the current context.
2	**DATESINPERIOD (Date_Column, Start_Date, Number_of_Intervals, Interval)** Returns a table that contains a column of dates that begins with the start_date and continues for the specified number_of_intervals.
3	**PARALLELPERIOD (Date_Column, Number_of_Intervals, Interval)** Returns a table that contains a column of dates that represents a period parallel to the dates in the specified Date_Column in the current context, with the dates shifted a number of intervals either forward or backward in time.
4	**DATESBETWEEN (Date_Column, Start_Date, End_Date)** Returns a table that contains a column of dates that begins with the start_date and continues until the end_date.

DAX Functions that Evaluate Expressions Over a Time Period

DAX Functions in this category evaluate an expression over a specified time period.

There are nine (9) DAX functions in this category −

- Three (3) DAX functions in this category can be used to evaluate any given expression over a specified time period.

Sr.No.	DAX Function & Return Value
1	**TOTALMTD (Expression, Date_Column, [SetFilter])** Evaluates the value of the expression for the dates in the month to date, in the current context.
2	**TOTALQTD (Expression, Date_Column, [SetFilter])** Evaluates the value of the expression for the dates in the quarter to date, in the current context.
3	**TOTALYTD (Expression, Date_Column, [SetFilter], [YE_Date])** Evaluates the value of the expression for the dates in the year to date, in the current context

- Six (6) DAX functions in this category can be used to calculate the opening and the closing balances.
  - The opening balance for any period is the same as the closing balance for the previous period.
  - The closing balance includes all data through the end of the period, while the opening balance does not include any data from within the current period.
  - These DAX functions always return the value of an expression evaluated for a specific point in time.
- The point in time we care about is always the last possible date value in a calendar period.
- The opening balance is based on the last date of the previous period, while the closing balance is based on the last date in the current period.
- The current period is always determined by the last date in the current date context.

Sr.No.	DAX Function & Return Value
1	**OPENINGBALANCEMONTH (Expression, Date_Column, [SetFilter])** Evaluates the expression at the first date of the month in the current context.
2	**CLOSINGBALANCEMONTH (Expression, Date_Column, [SetFilter])** Evaluates the expression at the last date of the month in the current context.
3	**OPENINGBALANCEQUARTER (Expression, Date_Column, [SetFilter])** Evaluates the expression at the first date of the quarter, in the current context.
4	**CLOSINGBALANCEQUARTER (Expression, Date_Column, [SetFilter])** Evaluates the expression at the last date of the quarter in the current context.
5	**OPENINGBALANCEYEAR (Expression, Date_Column, [SetFilter], [YE_Date])** Evaluates the expression at the first date of the year in the current context.
6	**CLOSINGBALANCEYEAR (Expression, Date_Column, [SetFilter], [YE_Date])** Evaluates the expression at the last date of the year in the current context.