Learning DAX: How to Extract Month and Month Name from Dates in Power BI

Authored by

Mohammed looti

November 12, 2025

RECOMMENDED CITATION

Mohammed looti (2025). *Learning DAX: How to Extract Month and Month Name from Dates in Power BI*. PSYCHOLOGICAL STATISTICS. Retrieved from https://statistics.arabpsychology.com/?p=17352

Introduction to Date and Time Intelligence in Power BI

Date and time intelligence forms the analytical backbone of effective reporting within Power BI. Data professionals frequently encounter the need to disaggregate complex date fields to facilitate precise filtering, grouping, and visualization based on specific temporal hierarchies, such as year, quarter, or, most commonly, the **month**. Extracting the month component from a full date column is an essential prerequisite for performing critical analyses, including **seasonality analysis**, detailed month-over-month (M-o-M) comparisons, and establishing clean date hierarchies that drive user-friendly dashboards. This comprehensive guide details the two principal methods available in DAX (Data Analysis Expressions) for achieving this crucial data separation, providing you with the tools necessary to handle both numerical and textual month representations with precision.

A firm grasp of isolating the month component empowers analysts to construct robust and high-performing time intelligence models. Depending on the final use case--whether you require the month as a simple integer (1 through 12) for sorting, filtering, and internal calculations, or as a descriptive name (January, Feb, etc.) for enhanced dashboard display--the Power BI environment, powered by DAX, offers distinct and highly optimized functions to address these divergent analytical needs. We will focus specifically on how to implement these solutions using calculated columns. Utilizing calculated columns ensures that the extracted temporal data is permanently stored within your data model, ready for reuse across various reports and measures, thereby maintaining model cleanliness and scalability.

The subsequent methodologies utilize calculated columns to derive the necessary temporal components. This approach is preferred when the result must be integrated into the data model structure itself, allowing for consistent reference in visuals and calculations throughout the report. The choice between Method 1 and Method 2 rests entirely upon the required output format: numerical output is mandatory for correct chronological sorting and mathematical operations, while textual output is reserved strictly for aesthetic presentation.

Understanding DAX Functions for Time Intelligence

DAX serves as the proprietary expression language utilized across Microsoft's suite of analytical tools, including Power BI, Analysis Services, and Power Pivot. It provides an extensive library of functions for data manipulation, aggregation, and complex calculations. For processing date and time values, DAX treats dates as numerical values, which simplifies the extraction of individual components like the month or year. Although Power BI includes built-in date hierarchy features, best practice dictates creating explicit columns for year, month, and day. This strategy offers superior control over custom formatting, sorting behavior, and overall data model performance.

Our focus will center on two specialized DAX functions that directly address month extraction. First, the dedicated MONTH function provides a straightforward numerical output. Second, the incredibly

versatile FORMAT function is indispensable for generating custom text outputs based on specific formatting strings. The selection criteria between these two methods are clear: choose `MONTH` when the priority is mathematical calculation and chronological integrity, and choose `FORMAT` when the primary goal is aesthetic presentation for end-users.

To effectively demonstrate these techniques, we will assume a common data modeling scenario where you are working with a table named **my_data** that contains a standard date column. This hypothetical setup is used consistently throughout the examples below, illustrating how these simple yet powerful DAX expressions are implemented within the Power BI data modeling environment.

Here is a quick overview of the two primary DAX methods used to extract the month:

Method 1: Extract Month Number from Date

month = MONTH('my_data')

Method 2: Extract Month Name from Date

month = FORMAT('my_data', "MMM")

Method 1: Extracting the Month Number using the MONTH Function

The most robust and chronologically accurate method for separating the month from a date involves leveraging the dedicated MONTH function in DAX. This function is designed to accept a single argument--a date value or a reference to a column containing date values--and reliably returns an integer ranging from 1 (representing January) to 12 (representing December). This numerical output is critical because it ensures that the resulting column is treated as a measure of time, not merely a text label, which is vital for calculations, filtering date ranges, and guaranteeing correct **chronological sorting** in all tables and charts.

When you define a new column using the MONTH function, the resulting data type is automatically set to a **whole number**. This characteristic is fundamental to building a reliable data model. If, conversely, you extract the month name (as shown in Method 2), the resulting column is a text field, which introduces a significant challenge: text fields sort alphabetically by default (e.g., April before January). Utilizing the numerical month column eliminates this sorting complexity entirely when the primary objective is to maintain chronological order in your visualizations.

The syntax for this function is straightforward and highly optimized for performance. You only need to specify the column containing the date value you intend to process. For instance, if your date column is named and resides within a table named `'my_data'`, the concise DAX expression is

constructed as follows:

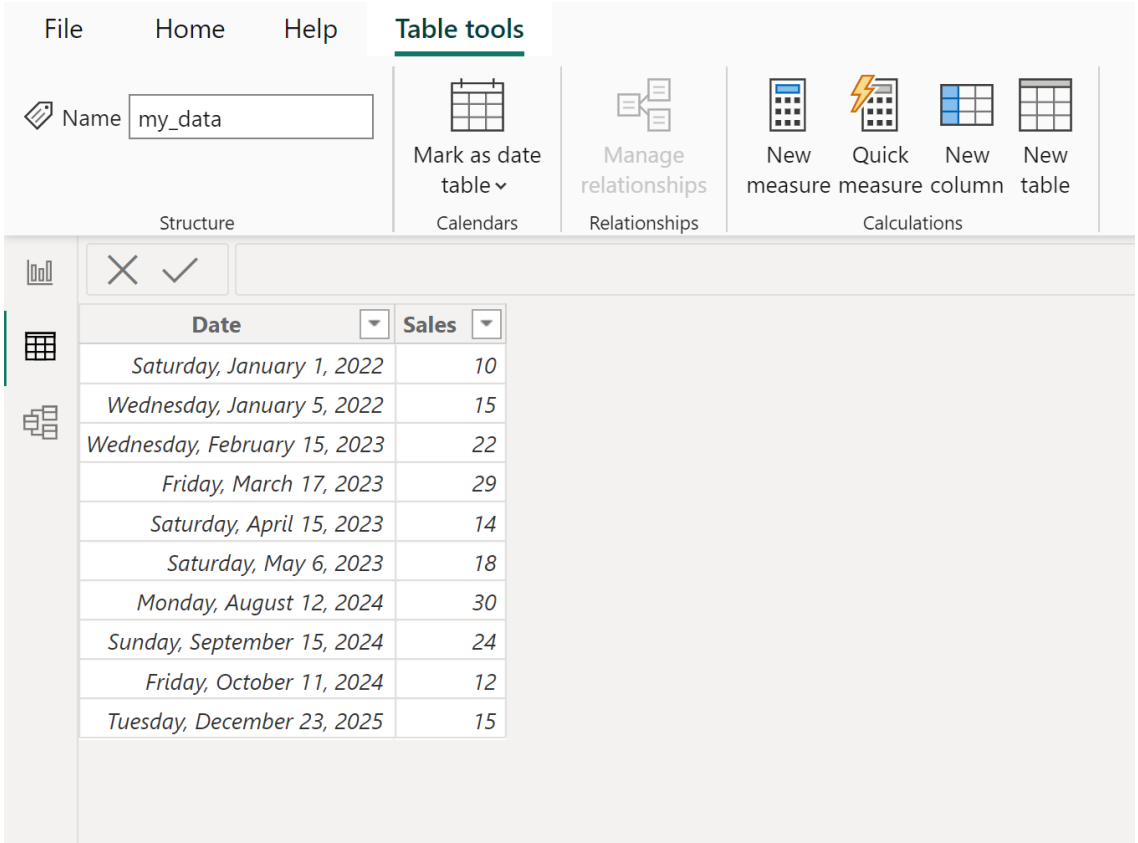
month = MONTH('my_data')

This approach is highly recommended whenever the extracted month value is intended for back-end data processing, the creation of calculated measures, or as the necessary sorting column for the textual month names.

Practical Application: Implementing Month Number Extraction (Example 1)

Let us proceed with the implementation steps using our sample table, **my_data**, which contains various date entries. Our goal is to augment this table by deriving a new column that holds only the numerical representation of the month for each corresponding record, ready for analysis and sorting.

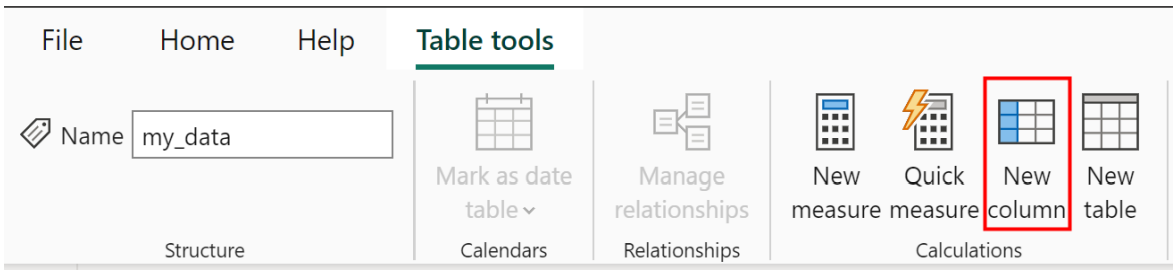
The initial data table **my_data**, illustrating the raw date format, is shown here:



Date	Sales
Saturday, January 1, 2022	10
Wednesday, January 5, 2022	15
Wednesday, February 15, 2023	22
Friday, March 17, 2023	29
Saturday, April 15, 2023	14
Saturday, May 6, 2023	18
Monday, August 12, 2024	30
Sunday, September 15, 2024	24
Friday, October 11, 2024	12
Tuesday, December 23, 2025	15

To initiate the extraction process, navigate to the Power BI Desktop interface. Ensure you are viewing the Data View or Model View. Locate the **Table tools** tab in the ribbon menu. Within this tab, click the icon labeled **New column**. This action promptly opens the formula bar, enabling the

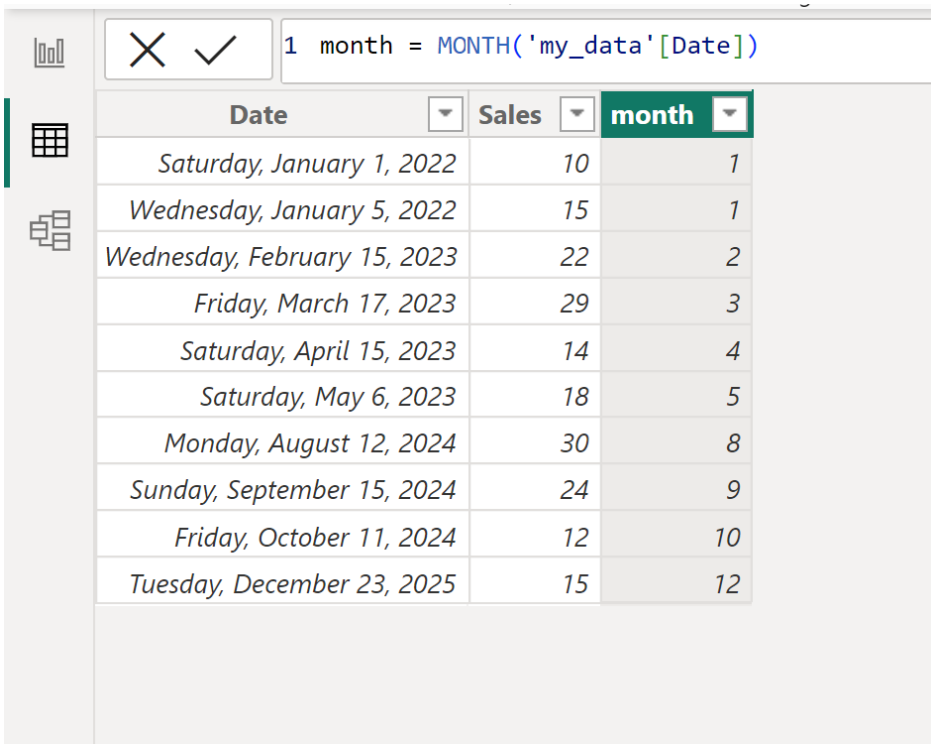
input of the DAX expression for your new calculated column.



Once the formula bar is active, input the precise DAX expression utilizing the MONTH function, ensuring correct reference to the original date column:

month = MONTH('my_data')

Executing this formula successfully generates a new column named **month**. This column exclusively displays the numerical month (1 through 12) corresponding to the date in the **Date** column. Observe the transformation where the complex date information is cleanly parsed into its fundamental numerical month component:



Date	Sales	month
Saturday, January 1, 2022	10	1
Wednesday, January 5, 2022	15	1
Wednesday, February 15, 2023	22	2
Friday, March 17, 2023	29	3
Saturday, April 15, 2023	14	4
Saturday, May 6, 2023	18	5
Monday, August 12, 2024	30	8
Sunday, September 15, 2024	24	9
Friday, October 11, 2024	12	10
Tuesday, December 23, 2025	15	12

Conclusion for Method 1: This technique is the definitive standard for obtaining a numerical month value, a prerequisite for accurate chronological sorting and all date-based mathematical

operations within your data model.

Method 2: Extracting the Month Name using the FORMAT Function

While numerical months are essential for back-end processing, effective data visualization frequently demands the display of the full or abbreviated month name (e.g., "Jan" or "January") to significantly enhance readability and user comprehension. To convert a numerical date value into an easily readable text string, we must utilize the highly flexible FORMAT function. The `FORMAT` function requires two core arguments: the input value (the date column) and a **format string** that dictates the desired output structure.

Mastering this method depends on understanding the specific date format codes used within the function. For month names, analysts typically rely on two primary codes:

"MMM": This returns the abbreviated month name (e.g., "Jan", "Feb", "Dec"). This format is typically favored when space is limited on visual axes or within tables.

"MMMM": This returns the full, unabbreviated month name (e.g., "January", "February", "December"). This format offers maximum clarity but naturally requires more visual real estate.

The key benefit of the FORMAT function lies in its granular control over the output appearance. However, a critical consideration is that the result of the `FORMAT` function is always a text string. Because text strings are sorted alphabetically by default, a column containing month names will sort incorrectly ("April" will appear before "January"). Therefore, if you use this method for display, you must subsequently link this new month name column to the numerical month column (created using Method 1) and configure the **Sort by column** property to ensure correct chronological sequencing.

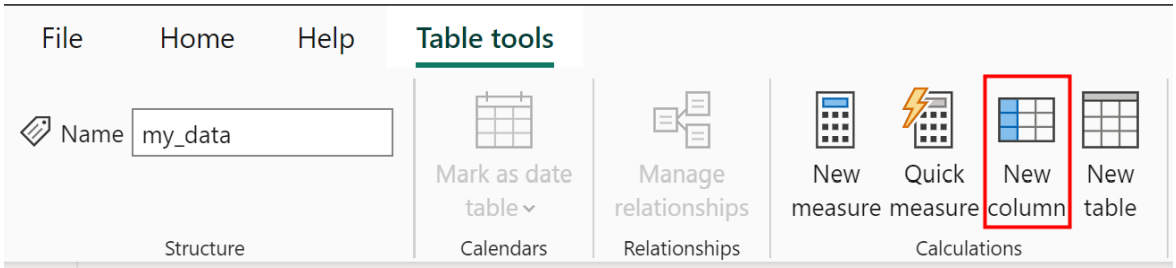
The DAX expression required for extracting the abbreviated month name is structured as follows:

month = FORMAT('my_data', "MMM")

Practical Application: Implementing Month Name Extraction (Example 2)

If the analytical requirement specifies displaying the month name for reporting purposes, the implementation process mirrors the numerical extraction, but we substitute the `MONTH` function with the FORMAT function and the appropriate format string.

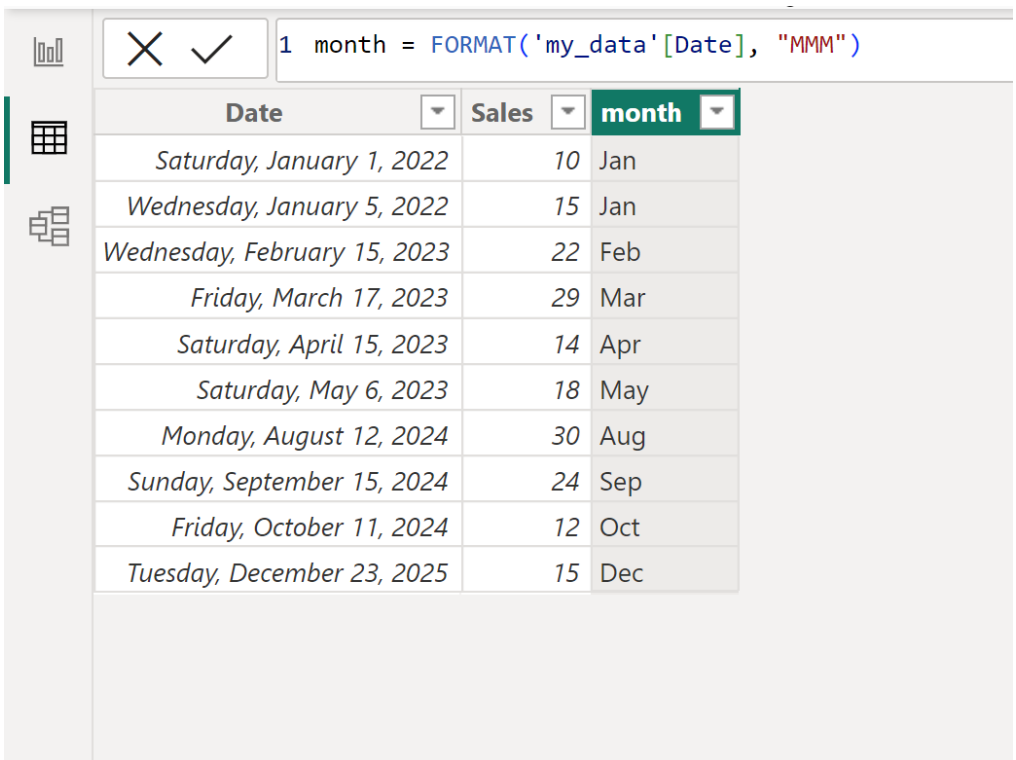
Begin by navigating to the **Table tools** tab in Power BI Desktop and clicking the **New column** icon, which prepares the formula bar for expression input:



Next, input the following formula into the active bar to extract the abbreviated month name, using the "MMM" format string:

month = FORMAT('my_data', "MMM")

Executing this calculation creates a new column named **month** that populates with the three-letter abbreviation corresponding to the month in the source **Date** column. Note carefully that the output now consists of text values, which are optimized for visual consumption:



Date	Sales	month
Saturday, January 1, 2022	10	Jan
Wednesday, January 5, 2022	15	Jan
Wednesday, February 15, 2023	22	Feb
Friday, March 17, 2023	29	Mar
Saturday, April 15, 2023	14	Apr
Saturday, May 6, 2023	18	May
Monday, August 12, 2024	30	Aug
Sunday, September 15, 2024	24	Sep
Friday, October 11, 2024	12	Oct
Tuesday, December 23, 2025	15	Dec

Important Note on Sorting: Since this new month column is a text field, it will inherently sort alphabetically. To enforce the correct chronological order (January, February, March, etc.), you must perform a crucial post-extraction step: select this new text month column in the Data View, navigate to the **Column tools** tab, and set the "Sort by column" property to reference the

numerical month column that you previously created using the `MONTH` function in Method 1. This linkage ensures that your visualizations display months in the correct time sequence, regardless of the text format utilized.

Summary of Best Practices and Additional Resources

Effective management of time-series data is foundational to generating high-quality analytical reports in Power BI. By mastering the two methods detailed above--employing the MONTH function for numerical integrity and the FORMAT function for textual presentation--you ensure that your data model can support both highly accurate calculations and aesthetically refined visualizations simultaneously. The most robust data models consistently incorporate both a numerical month column (essential for sorting) and a textual month column (essential for display).

The definitive takeaway is to always determine the end goal before choosing the function: if the month value is used for chronological sorting, filtering, or internal calculations, the numerical output of `MONTH` is mandatory. If the month is used purely for presentation on a chart axis or slicer, the textual output of `FORMAT` is appropriate, provided you establish the correct sorting relationship immediately afterward. This dual approach maximizes the analytical capability and visual clarity of your reports, leading to superior data insights.

To continue advancing your expertise in data manipulation and temporal transformations within Power BI, explore the following resources which detail other common date and data transformation tasks:

The following tutorials explain how to perform other common tasks in Power BI: