

How to Extract the Month from a Date in Power BI: A Step-by-Step Guide

Authored by
stats writer

January 29, 2026

RECOMMENDED CITATION

stats writer (2026). *How to Extract the Month from a Date in Power BI: A Step-by-Step Guide*. PSYCHOLOGICAL SCALES. Retrieved from <https://scales.arabpsychology.com/?p=128424>

To extract the month from a date in Power BI, you will need to use the DAX function called "MONTH". This function allows you to extract the month from a given date and returns a number representing the month. For example, if the date is January 1st, the function will return the number 1. This allows you to easily sort and analyze your data by month. To use this function, you will need to create a new column in your data table and use the MONTH function to reference the date column. This will then display the month for each corresponding date in your data. You can also use other DAX functions such as "FORMAT" to display the month as a text value instead of a number. This feature in Power BI is useful for organizing and visualizing data by month, allowing for better insights and analysis.

Power BI: Extract Month from Date

Introduction: The Necessity of Date Extraction in Power BI

Effective temporal analysis is fundamental to business intelligence, and in Power BI, extracting specific components from a date column--such as the month--is a frequent and critical requirement. Analyzing sales trends, resource allocation, or usage patterns often necessitates grouping data solely by month, irrespective of the specific year or day. To achieve this separation from a full date timestamp, you must utilize the powerful calculation capabilities provided by the Data Analysis Expressions (DAX) language. This process converts a complex date value into a simple, manageable monthly identifier, whether numerical or textual.

The primary tool for numerical extraction is the MONTH function in DAX. This function is specifically designed to isolate the month index from any given date field, returning an integer between 1 (January) and 12 (December). Utilizing this integer output allows for straightforward sorting, filtering, and aggregation of your dataset, laying the groundwork for robust time-series visualizations. For instance, if your data spans multiple years, extracting the month number enables you to compare January performance across all years seamlessly, providing immediate insight into seasonal behaviors without being skewed by yearly variance.

Implementing this extraction involves creating a new calculated column within your data model. This approach ensures that the original date column remains intact while providing a dedicated column solely for monthly analysis. By applying the chosen DAX formula within this new column, Power BI dynamically calculates and stores the month value for every corresponding date entry in your table. While the numeric output is often optimal for calculations and precise sorting, we will also explore methods, such as using the FORMAT function, to display the month as readable text (e.g., "Jan" or "January"), catering to different reporting needs and enhancing user experience.

Understanding the DAX Engine for Date Manipulation

Data Analysis Expressions (DAX) serves as the functional language backend for Power BI, enabling the creation of complex metrics, measures, and, crucially for this task, calculated columns required for advanced data modeling. When dealing with temporal data, DAX provides a rich set of built-in Time Intelligence functions. For simple extraction tasks like isolating the month, we rely on straightforward scalar functions that operate row by row within the context of a calculated column. Understanding this context is vital: since the month value is static for each row's date, we use a calculated column rather than a measure.

A calculated column computes its result based on the specific row context; this means that for every row in your data table, the formula is executed individually. The calculation pulls the date value from that row, performs the extraction, and returns the corresponding month. When using the **MONTH** or **FORMAT** functions, the primary input required is a date type column reference. If the date column is named 'my_data', the DAX formula works by iterating through the my_data table, applying the extraction logic to each date entry, and populating the newly created column with the resulting month value. This persistence makes the extracted month available instantly for filtering, grouping, and establishing relationships within your data model.

Power BI offers important flexibility in how you choose to represent the extracted month. Choosing between a numerical output (1 through 12) and a textual output (Jan, February) depends entirely on the downstream visualization or analytical requirements. Numerical output is highly beneficial for guaranteed correct sorting order and integration into mathematical operations, whereas textual output provides immediate readability for end-users on reports. The following sections detail both primary DAX methods for achieving month extraction, ensuring you have the appropriate tool for any reporting scenario.

Overview of Date Extraction Methods in DAX

Within the DAX environment, there are two primary and highly efficient methods for isolating and presenting the month component of a date field. These methods leverage specialized DAX functions that handle the intricacies of date formats internally, allowing the user to focus solely on the desired output format. Both techniques are critical skills for any Power BI developer focused on time-series analysis and both involve the creation of a new calculated column.

The first method utilizes the intrinsic MONTH function. This is the most direct way to retrieve the numerical index of the month. It is exceptionally simple to implement and results in an integer column (1, 2, 3, etc.) that guarantees correct chronological sorting by default, which is a major, often decisive, advantage when constructing line charts or matrix tables intended to show temporal progression. If performance and sorting stability are paramount considerations for your data model, this numerical method is the preferred standard.

Conversely, the second method leverages the versatile FORMAT function. While the primary function of FORMAT is to convert values into specific text strings based on a defined pattern, it is perfectly suited for extracting date components. By passing the date column and a specific format string (such as "MMM" or "MMMM"), we can instruct DAX to return the abbreviated or full name of the month (e.g., "Mar" or "March"). This text representation is crucial for highly readable reports but, as a text field, it introduces a potential sorting challenge that must be manually corrected later using the "Sort by Column" feature in the data model.

You can use the following methods in DAX to extract the month from a date in Power BI:

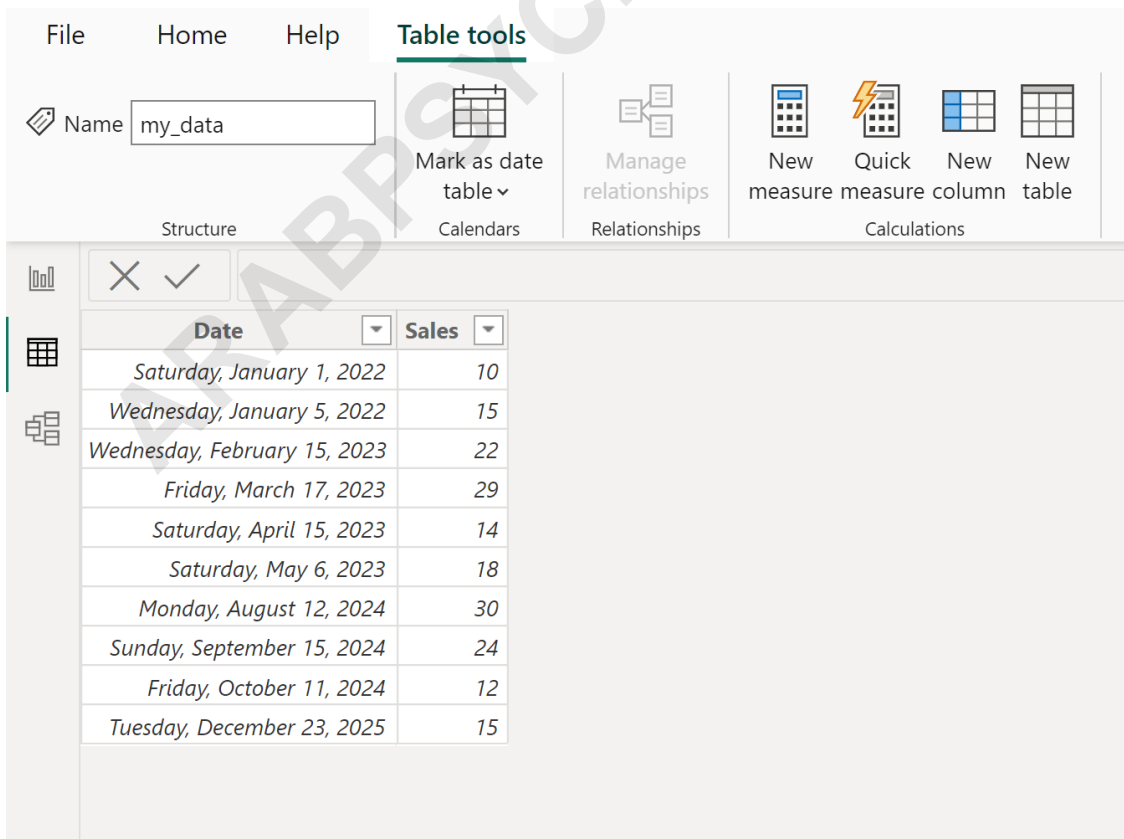
Method 1: Extract Month Number from Date

```
month = MONTH('my_data')
```

Method 2: Extract Month Name from Date

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

The following examples show how to use each method in practice with the following table named **my_data** in Power BI:



The screenshot displays the Power BI interface with the 'Table tools' ribbon active. The ribbon includes options for 'Name' (set to 'my_data'), 'Mark as date table', 'Manage relationships', and 'Calculations' (with sub-options for 'New measure', 'Quick measure', 'New column', and 'New table'). Below the ribbon, a data table is visible with the following content:

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

Example 1: Extract Month Number from Date in Power BI

Step-by-Step Implementation of Method 1: Extracting the Month Number

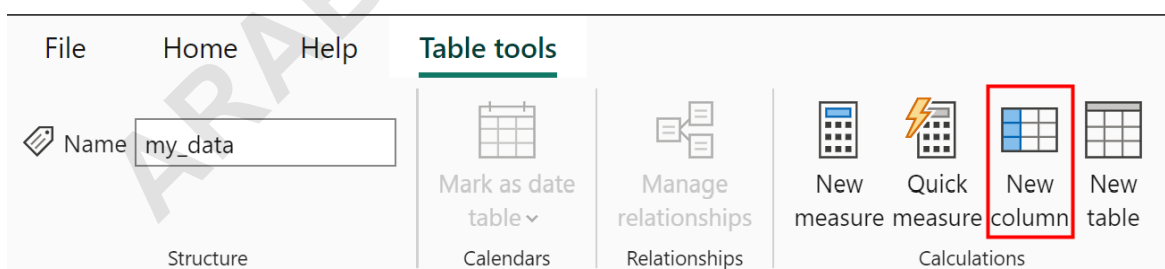
The first and most robust method for date component extraction in Power BI utilizes the dedicated MONTH function. This approach guarantees that the resulting column is an integer data type, which is critical for numerical comparisons and precise chronological sorting in visuals. To begin this process, ensure you are either in the Data view or Report view where your target table (in this comprehensive example, named **my_data**) is currently selected and visible.

The process starts by instructing Power BI to create a new container for the derived data. You must navigate to the **Table tools** tab located in the Power BI ribbon interface. Within this ribbon, locate and click the dedicated icon labeled **New column**. This crucial action opens the formula bar, allowing you to input your specific DAX logic. The key to successful implementation lies in correctly referencing the table and the column containing the source date information, ensuring the expression operates on the correct data field.

We want the newly created column, which we will assign the name `month`, to contain the simple numerical representation of the date's month. We achieve this by assigning the result of the **MONTH** function to the column name, referencing the date column (e.g., `'my_data'`) as the function's sole argument. The specific syntax shown below is straightforward, highly optimized, and instantly creates the desired numeric column in your data model.

Suppose that we would like to extract the month number from each date in the **Date** column.

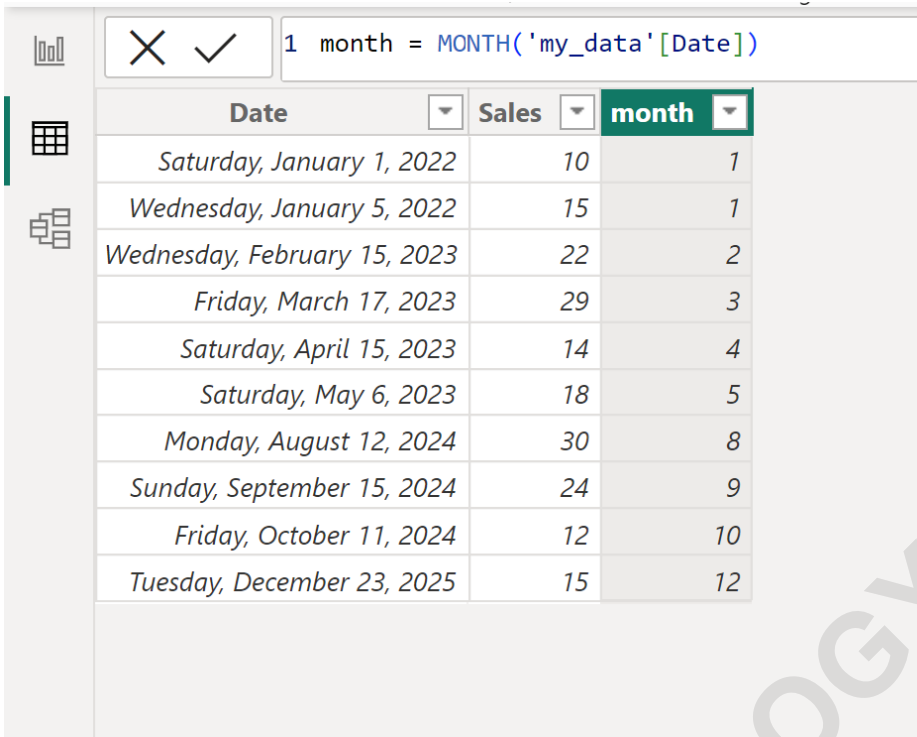
To do so, click the **Table tools** tab, then click the icon called **New column**:



Then type the following formula into the formula bar:

```
month = MONTH('my_data')
```

This will create a new column named **month** that contains only the month number from the the corresponding date in the **Date** column:



The screenshot shows the Power BI interface with a DAX formula bar at the top containing the formula: `1 month = MONTH('my_data'[Date])`. Below the formula bar is a table with three columns: Date, Sales, and month. The table contains the following data:

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

Note: You can find the complete documentation for the [MONTH function](#) in [DAX](#).

Example 2: Extract Month Name from Date in Power BI

Detailed Breakdown of Method 2: Using the FORMAT Function for Month Names

While numerical month extraction is excellent for back-end analysis, presenting months as recognizable names (e.g., "Feb," "October") significantly improves the interpretability and user experience of reports and dashboards. This goal requires the use of the [FORMAT function](#), which is one of the most versatile text manipulation tools available in [Power BI](#). The FORMAT function takes two main arguments: the initial value you wish to format (which will be the date column) and a mandatory string specifying the desired output format.

For month extraction specifically, the crucial format specifiers are "MMM" and "MMMM". Using "**MMM**" tells the [FORMAT function](#) to return the abbreviated month name (e.g., "Jan," "Jul," "Dec"). This compact representation is often favored in visualizations where space is limited, such as axis labels or dense matrix tables. If clarity and completeness are preferred by the report consumers, using the format string "**MMMM**" will return the full, unabbreviated month name (e.g., "January," "July," "December"). It is crucial to remember that the output of the FORMAT function is always a text string, which fundamentally impacts how visuals attempt to sort the resulting data.

When implementing this method, it is absolutely vital to address the resulting sorting issue. Because the output is text, Power BI defaults to alphabetical sorting ("Apr" comes before "Feb"), which is chronologically incorrect. To ensure your visuals display months in the correct order (January, February, March...), you must implement a two-step process: first, create the numerical month column (using Method 1), and then, in the Data View, set the text-based Month Name column to "Sort by Column" the numerical month column. This configuration links the display order of the highly readable text field to the underlying chronological order of the numbers, effectively resolving the inherent sorting challenge of textual date components.

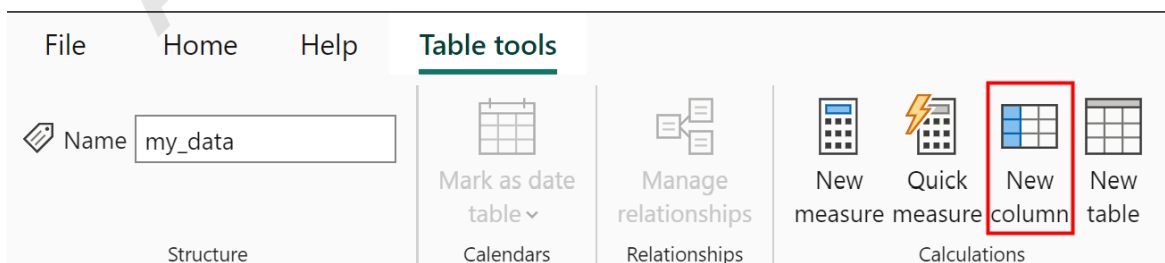
Step-by-Step Implementation of Method 2: Extracting the Month Name

To successfully demonstrate the extraction of the month name, we follow the same initial setup steps as Method 1, focusing on creating a new calculated column container. This approach is highly recommended for any user-facing reports where immediate readability is prioritized over purely numerical analysis or calculations. In the example below, we will use the format string "MMM" to return the abbreviated month name.

Begin by ensuring the target table (**my_data**) is selected in your Power BI environment. Navigate once again to the **Table tools** tab located in the main ribbon. Click the **New column** icon to instantly open the formula editor at the top of the interface. This action prepares the environment for inputting the DAX expression that utilizes the FORMAT function to transform the raw date column's output into a textual month identifier.

The syntax for the FORMAT function requires the date column reference followed by the specific format string, which must be enclosed in double quotes. For instance, the expression `FORMAT('my_data', "MMM")` instructs Power BI to look at the date in each row of the `Date` column and return the three-letter abbreviation of the corresponding month. Naming this new column `month` allows for easy recognition and seamless integration into report visualizations.

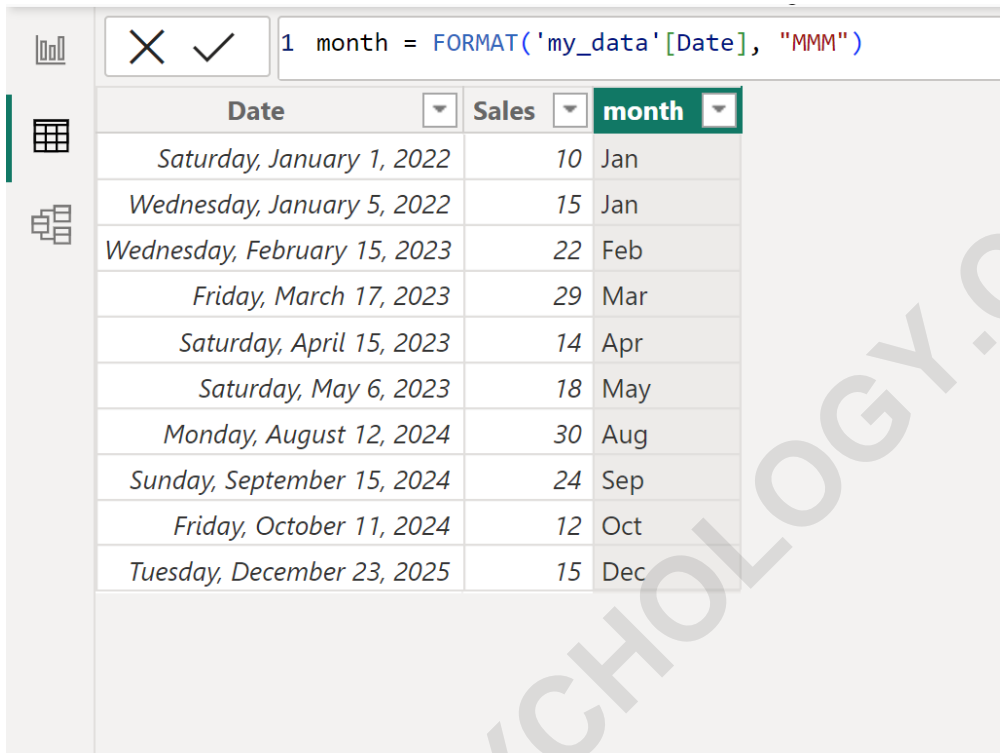
To do so, click the **Table tools** tab, then click the icon called **New column**:



Then type the following formula into the formula bar:

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

This will create a new column named **month** that contains only the month name from the the corresponding date in the **Date** column:



The screenshot shows the DAX editor interface. The formula bar contains the following DAX expression:

```
1 month = FORMAT('my_data'[Date], "MMM")
```

Below the formula bar, a table is displayed with the following data:

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

Note #1: If you would rather extract the full month name, then you should use **"MMMM"** in the [FORMAT function](#) instead.

Note #2: You can find the complete documentation for the [FORMAT function](#) in [DAX](#).

Advanced Applications and Alternatives for Date Analysis

Once the month has been successfully extracted using either the numerical **MONTH** function or the textual **FORMAT** function, data analysts can begin exploring more complex time intelligence features in [Power BI](#). One crucial advanced step is the development of a dedicated Date table, often referred to as a [date dimension](#). While extracting the month directly into your fact table (like `my_data`) is convenient for quick analysis, creating a standalone, comprehensive date table linked to your data provides unparalleled flexibility for complex filtering, hierarchical slicing, and utilizing native time intelligence calculations built into DAX.

When choosing to work with the full or abbreviated month name (using the **"MMMM"** or **"MMM"** format strings), it is highly recommended to immediately perform the **"Sort by Column"**

configuration discussed earlier. Failing to do so will result in confusing, alphabetically ordered charts and visuals that fundamentally misrepresent chronological trends. Furthermore, if your analysis requires the ability to cleanly separate months by year (e.g., comparing Jan 2023 sales versus Jan 2024 sales), you must also create a separate column using the **YEAR** function and potentially combine the month and year columns into a single, chronologically sortable identifier (e.g., "2023-01") for proper comparison across multi-year data sets.

Finally, for users who prefer to streamline their data preparation process before it even reaches the DAX data model, it is worth exploring the Power Query M language within the query editor environment. Power Query offers alternative, extract-transform-load (ETL) methods for date extraction. Functions such as `Date.Month` and `Date.ToText` provide similar capabilities to their DAX counterparts but operate at the data ingestion level. Ultimately, whether you choose the simple `MONTH` function for robust numerical analysis or the powerful `FORMAT` function for highly readable reports, mastery of date component extraction is a foundational skill for building high-quality business intelligence dashboards in Power BI.

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