

How to Add an Index Column to a Table in Power BI

Authored by
stats writer

January 30, 2026

RECOMMENDED CITATION

stats writer (2026). *How to Add an Index Column to a Table in Power BI*. PSYCHOLOGICAL SCALES. Retrieved from <https://scales.arabpsychology.com/?p=128583>

The ability to manipulate and structure data efficiently is paramount in modern business intelligence. One common requirement when preparing data for robust analysis in Power BI is the need for a sequential, unique identifier for each record. This is precisely the function of an Index Column. Rather than relying on complex formulas or external tooling, Power BI simplifies this process significantly through the built-in functionality available within the Power Query Editor. Using the "Index Column" feature, located conveniently in the "Add Column" tab, you can automatically generate a series of sequential numbers for every row in your dataset. This capability is not just about aesthetics; it provides a crucial foundation for effective data modeling, facilitating sophisticated data sorting, filtering operations, and the establishment of reliable, one-to-many relationships between disparate tables within your analytical model. By mastering this straightforward process, you ensure your data is optimally structured for high-performance reporting.

This comprehensive guide details the exact methodology for incorporating an Index Column into any table imported into your Power BI environment. We will utilize the native tools available in the Power Query interface, which is the primary staging area for all data transformation tasks. The generation of a sequential index column serves as a powerful means to ensure data integrity and traceability, especially when dealing with data sources that lack a native primary key or unique identifier. The resulting column provides a stable reference point that remains consistent even if the underlying data source changes or is refreshed, provided the transformation steps within Power Query are maintained. Understanding how this simple feature works is a fundamental skill for any data analyst working within the Microsoft Power Platform ecosystem, enabling more precise and repeatable data preparation workflows.

Power BI: Adding a Sequential Index Column to Your Data Tables

The addition of an index column is performed exclusively within the Power Query Editor, which acts as the Extract, Transform, Load (ETL) layer of Power BI Desktop. The editor provides a graphical interface (GUI) to apply complex transformations without needing to manually write the underlying M language code. Leveraging the dedicated **Index Column** feature significantly simplifies the process compared to writing custom functions, making it accessible even for users who are new to data transformation. This method ensures that the indexing is applied consistently across the entire dataset as a discrete transformation step, which is logged and repeatable every time the dataset is refreshed. This transformation is applied at the source level, meaning the index exists before the data is loaded into the final data model for reporting.

Prerequisites: Accessing the Power Query Editor

Before any transformation can occur, you must first navigate from the standard Report View or Data View in Power BI Desktop into the data preparation workspace. This workspace is the **Power Query Editor**, where all data ingestion and cleaning activities take place. Initiating this transition is straightforward and is always performed from the main application ribbon. To begin, ensure you have loaded the target table into your Power BI file. Once loaded, click on the **Home** tab located in the top ribbon of the Power BI Desktop application. This tab houses the primary navigation and control features for data management, including the key entry point for transformations.

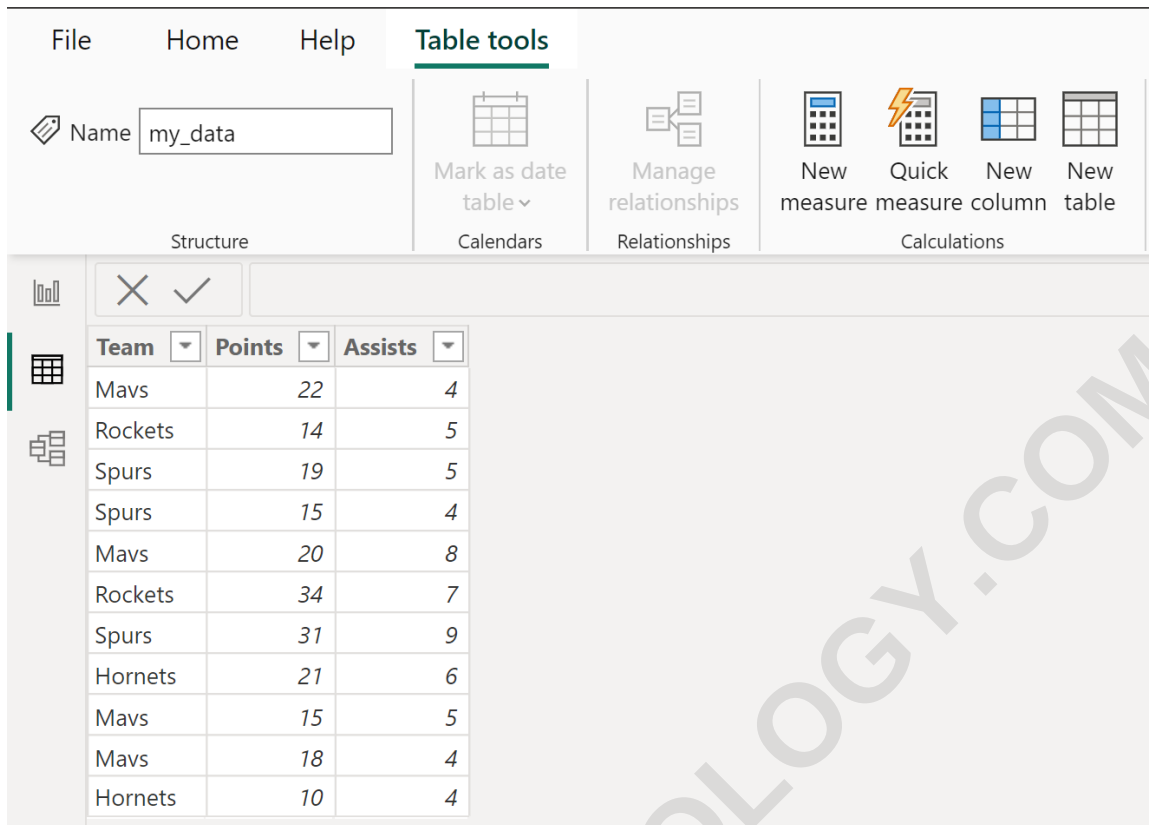
Within the **Home** tab, you will find the critical button labeled **Transform data**. Clicking this icon serves as the gateway to the ETL environment. It is important to note that sometimes this button appears as a small icon depicting a transformation or funnel, but its name remains consistent: **Transform data**. Upon clicking this button, the entire Power Query Editor window will launch, presenting a dedicated interface separate from the main Power BI application. This separation allows for focused data manipulation without interfering with the visualization or modeling layers of the report.

The Power Query Editor will display a list of all current queries (tables) on the left pane. Select the specific table to which you intend to add the index column. For the purpose of our demonstration, we will assume we are working with a table designated as **my_data**, which represents a typical set of raw data imported from a spreadsheet or database source. Understanding the structure of your data before applying an index is crucial, especially if the data contains duplicates or requires pre-sorting based on specific criteria before the index is applied. If sorting is needed, that step should be performed immediately before the index column is added in the list of Applied Steps.

Practical Demonstration Setup

For illustrative purposes, consider the following example table loaded into the Power Query Editor, named **my_data**. This table currently lacks a unique row identifier, which can complicate complex analytical tasks, especially those involving granular record tracking or row-level security implementation. The data consists of several columns, but no single column guarantees uniqueness across all rows, making the artificial creation of an index necessary for robust data modeling.

The visual representation below shows the initial state of our sample data before any transformations have been applied within the Power Query environment. Notice the absence of a dedicated sequential column on the left side of the table, confirming the need for the indexing operation.



Our objective is to augment this table by adding an index column whose numerical sequence starts at 1 and increments sequentially to the total number of records, denoted as N (where N represents the count of rows in the table). This transformation is considered an essential data preparation step that enhances the utility of the dataset for subsequent analysis using tools like DAX calculations or visual filtering within Power BI reports. This operation is fundamentally about enhancing metadata to support better analytical outcomes.

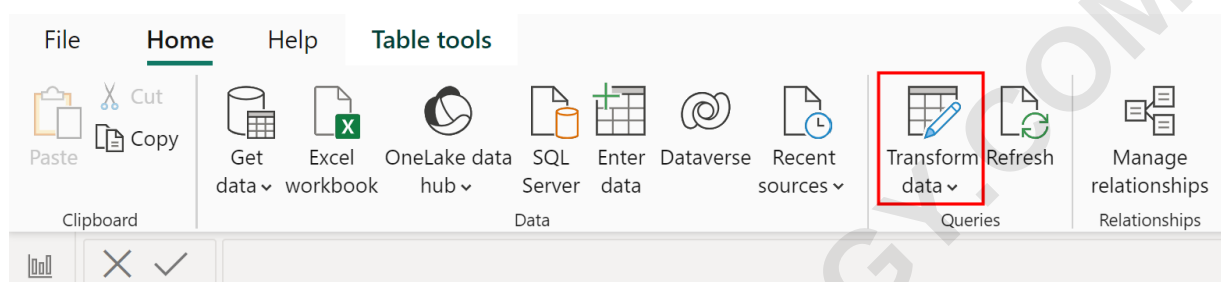
Executing the Index Column Transformation

Once you are within the Power Query Editor and the target table is selected, the next step involves navigating to the specific tab designed for creating new columns based on existing data or structured sequences. This is the **Add Column** tab, located prominently along the top ribbon, adjacent to the Home and Transform tabs. The organizational structure of the Power Query Editor is designed to group similar operations, making the process intuitive for data professionals.

Within the **Add Column** tab, locate the group labeled "General" or simply find the icon labeled **Index Column**. This feature is provided specifically to generate the required sequential identifiers. Instead of immediately clicking the button, which often defaults to starting the count at zero, you should click the small dropdown arrow associated with the **Index Column** button. This action reveals critical initialization options, allowing the user to specify the starting point for the index

sequence, which is a crucial detail for ensuring alignment with business requirements or standard database practices.

To initiate the transformation, you first navigate to the **Home** tab along the top ribbon of the Power BI Desktop application, and subsequently click the **Transform data** icon, as illustrated below. This confirms the transition into the Power Query Editor environment where column creation and manipulation occur. This step ensures that the transformation is recorded as part of the overall data load script.



Selecting the Index Start Point: From 0 vs. From 1

The choice of where the index begins--at 0 or 1--is a subtle yet important decision that depends largely on the intended use case or established conventions within your organization. The Power Query Editor provides three distinct options under the **Index Column** dropdown menu: **From 0**, **From 1**, and **Custom...**

Selecting **From 0** means the first row of your table will be assigned the value 0, the second row 1, and so on. This zero-based indexing is traditional in many programming languages and computing contexts, particularly when working with array offsets or system-level identifiers. If you anticipate exporting this data for processing by software that expects zero-based indices, this option is generally preferred.

Conversely, selecting **From 1** assigns the value 1 to the first row, 2 to the second, and continues incrementing by one. This one-based indexing is far more common in relational database systems and is often preferred for human-readable reports and user-facing identifiers, as it aligns more naturally with typical counting practices. For most standard analytical tasks within Power BI, especially when the index is used for sorting or simple lookups, starting **From 1** is the standard recommendation. Finally, the **Custom...** option allows for advanced configurations, where you can specify both the starting value and the increment step, offering maximum flexibility for unique sequencing requirements.

The following visual demonstrates the menu where you select the index starting position after

clicking the dropdown associated with the Index Column tool:

The screenshot shows the Power Query Editor interface. The 'Add Column' tab is selected, and the 'Index Column' dropdown is highlighted with a red box. A tooltip is visible, stating: 'Create a new column with an index starting at 1.' The data table below shows columns for Team, Points, and Assists.

	Team	Points	Assists
1	Mavs	22	4
2	Rockets	14	5
3	Spurs	19	5
4	Spurs	15	4
5	Mavs	20	8
6	Rockets	34	7
7	Spurs	31	9
8	Hornets	21	6
9	Mavs	15	5
10	Mavs	18	4
11	Hornets	10	4

Implementing the Index and Reviewing Results

For the purposes of this walkthrough, and aligning with typical business requirements for sequential counting, we will choose the **From 1** option. Upon selecting this option, the transformation is applied immediately to the query within the Power Query Editor, and a new column, typically named "Index," will appear as the leftmost column in your table preview. This action automatically generates the underlying M language code, specifically utilizing the `Table.AddIndexColumn` function, and adds it to the list of applied steps for the query. This ensures the transformation is repeatable and traceable.

It is vital to review the newly added column to confirm that the sequence starts correctly at 1 and continues through every row without interruption. This confirmation step verifies the integrity of the transformation before the data is committed to the main data modeling layer of Power BI Desktop. If you had required a different starting point or step interval, you would have selected the Custom option to define parameters such as starting at 100 with a step of 5, for instance.

As shown in the image below, selecting **From 1** results in the immediate insertion of the "Index" column, fulfilling our requirement for a one-based sequential identifier:

	Team	Points	Assists	Index
1	Mavs	22	4	1
2	Rockets	14	5	2
3	Spurs	19	5	3
4	Spurs	15	4	4
5	Mavs	20	8	5
6	Rockets	34	7	6
7	Spurs	31	9	7
8	Hornets	21	6	8
9	Mavs	15	5	9
10	Mavs	18	4	10
11	Hornets	10	4	11

Committing Changes and Finalizing the Data Load

Once all necessary transformations, including the addition of the Index Column, have been successfully applied and reviewed within the Power Query Editor, the changes must be committed and loaded back into the Power BI data model. This process involves closing the Power Query Editor window and instructing Power BI Desktop to execute the recorded transformation steps on the actual dataset. To do this, navigate to the **Home** tab within the Power Query Editor and click the **Close & Apply** button. This button signals the editor to cease operations and push the modified data structure back to the main analytical environment.

When you attempt to exit the editor, especially if significant data changes have occurred, a confirmation dialog box may appear. This box typically asks if you would like to apply your pending changes to the data model. You must click **Yes** to ensure the new index column is finalized and available for use in relationship management, DAX calculations, and visual reporting components. Failure to click **Yes** or selecting "Close" instead of "Close & Apply" will result in the loss of the newly added index column transformation.

Upon successful application, the Power BI Desktop environment will refresh the data model, incorporating the table with the new index column. The final table, now residing in the Data View or Report View, will clearly show the index, ready to be utilized for various analytical purposes. In our example, the resulting table now possesses a robust, unique identifier column covering every record:

The screenshot displays the Microsoft Power BI Desktop interface. The 'Table tools' ribbon is active, showing options for 'Clipboard' (Paste, Cut, Copy), 'Data' (Get data, Excel workbook, OneLake data hub, SQL Server, Enter data, Dataverse, Recent sources), and 'Query' (Transform data). Below the ribbon, a data table is visible with the following columns: Team, Points, Assists, and Index. The table contains 11 rows of data, with the 'Index' column numbered sequentially from 1 to 11.

Team	Points	Assists	Index
Mavs	22	4	1
Rockets	14	5	2
Spurs	19	5	3
Spurs	15	4	4
Mavs	20	8	5
Rockets	34	7	6
Spurs	31	9	7
Hornets	21	6	8
Mavs	15	5	9
Mavs	18	4	10
Hornets	10	4	11

As observed in the final dataset visualization, the newly added Index column successfully ranges from 1 to 11, perfectly matching the total row count of the sample table. This confirms the index column has been successfully added and integrated into the primary data model.

Advanced Considerations and Usage Scenarios

While the primary function of the index column is to provide a unique row identifier, its strategic importance extends deep into advanced data modeling techniques. One key advanced scenario involves creating robust relationships between tables where foreign keys might be ambiguous or non-existent in the source data. By adding index columns to both the fact and dimension tables, you can use these sequential identifiers as a reliable basis for temporary or surrogate keys, facilitating accurate join operations in the Power Query phase or relationship definition in the Data View.

Furthermore, index columns are invaluable when performing complex data merging operations. When using the Merge Queries function in Power Query, if the goal is to align rows based purely on their original sequential order rather than matching specific values, the index column becomes

the definitive join key. This is particularly useful in scenarios where data integrity requires maintaining the original row order from the source file. It is important to remember that if the index column is used for merging, the source table must be consistently sorted before the index step is applied to guarantee reliable results.

Finally, the index column can be leveraged within custom calculations using the [DAX](#) language. For example, if you need to calculate running totals, moving averages, or perform calculations that depend on the relative position of a row within a sorted context, the index column provides the necessary integer sequence for iterative processing functions. Although DAX has specialized tools for row context manipulation, a pre-existing sequential index column simplifies many common sequencing requirements, enhancing performance and clarity in complex measures.

Further Exploration of Power BI Tutorials

Mastering the Power BI environment involves proficiency in a wide array of data preparation and analytical tasks. The index column is merely one tool in a vast suite of transformation capabilities. To further enhance your expertise, consider exploring the foundational elements of [DAX](#), the nuances of the M Language for custom transformations, and best practices for creating optimized data schemas. Continuing education in these areas will empower you to handle increasingly complex data challenges.

The following tutorials provide additional guidance on how to perform other common and essential tasks within the [Power BI](#) ecosystem, building upon the basic transformation skills demonstrated here:

How to implement effective filtering strategies in Power Query.

Advanced techniques for merging and appending queries.

Writing custom functions using the [M Language](#).

Creating calculated columns and measures using [DAX](#).