Learning to Remove Blank Rows in Power BI: A Step-by-Step Guide

Authored by

Mohammed looti

November 12, 2025

The Critical Role of Data Integrity and Handling Blank Rows in Power BI

Achieving effective data cleansing is arguably the most fundamental step when constructing accurate and reliable reports and dashboards within the Power BI ecosystem. Data analysts frequently encounter a pervasive challenge: dealing with extraneous or unwanted rows, most commonly appearing as entirely blank entries. These blank records typically infiltrate datasets during the initial data ingestion phase, often stemming from poorly structured source systems, messy file exports (such as spreadsheets containing thousands of trailing empty rows), or unforeseen glitches within complex ETL (Extract, Transform, Load) pipelines. While these entries might seem harmless, they possess the capacity to severely distort analytical outcomes, particularly when performing aggregations or when the data model relies on specific cardinality relationships, inevitably leading to misleading visualizations. Consequently, the proactive removal of these extraneous records is paramount to upholding the integrity and accuracy of the final dataset used for critical business intelligence operations.

In the specialized vocabulary of data modeling, the term "blank rows" generally signifies records where every single column entry registers as either a technical Null value or an empty string, essentially representing a record devoid of any meaningful information. Although Power BI's robust visualization engine can tolerate some imperfections, retaining these empty rows introduces unnecessary overhead by consuming memory and processor time, thereby significantly slowing down report refresh cycles and reducing the overall efficiency of the underlying data model. The dedicated environment within Power BI specifically engineered to manage these essential pre-modeling transformation tasks is the **Power Query Editor**. This integrated tool provides an intuitive, user-friendly interface that allows analysts to execute sophisticated transformation logic without needing extensive coding expertise, making it the definitive starting point for all data preparation activities, including the efficient elimination of unwanted blank records.

Fortunately, the most streamlined and direct approach for addressing fully blank rows involves utilizing a specialized, built-in functionality located directly within the **Power Query Editor** interface. This powerful feature, aptly labeled **Remove Blank Rows**, is designed to rapidly scan the entirety of the selected table and instantly eliminate any record where all fields are uniformly detected as Null or empty. Employing this targeted function saves significant time compared to implementing manual filtering rules or constructing complex conditional statements, thereby ensuring that the foundational dataset is optimally clean, lean, and immediately prepared for advanced analytical tasks. The subsequent sections of this guide will detail the precise, step-by-step instructions necessary to access and execute this crucial transformation, utilizing a realistic data example to thoroughly illustrate the effectiveness of the process.

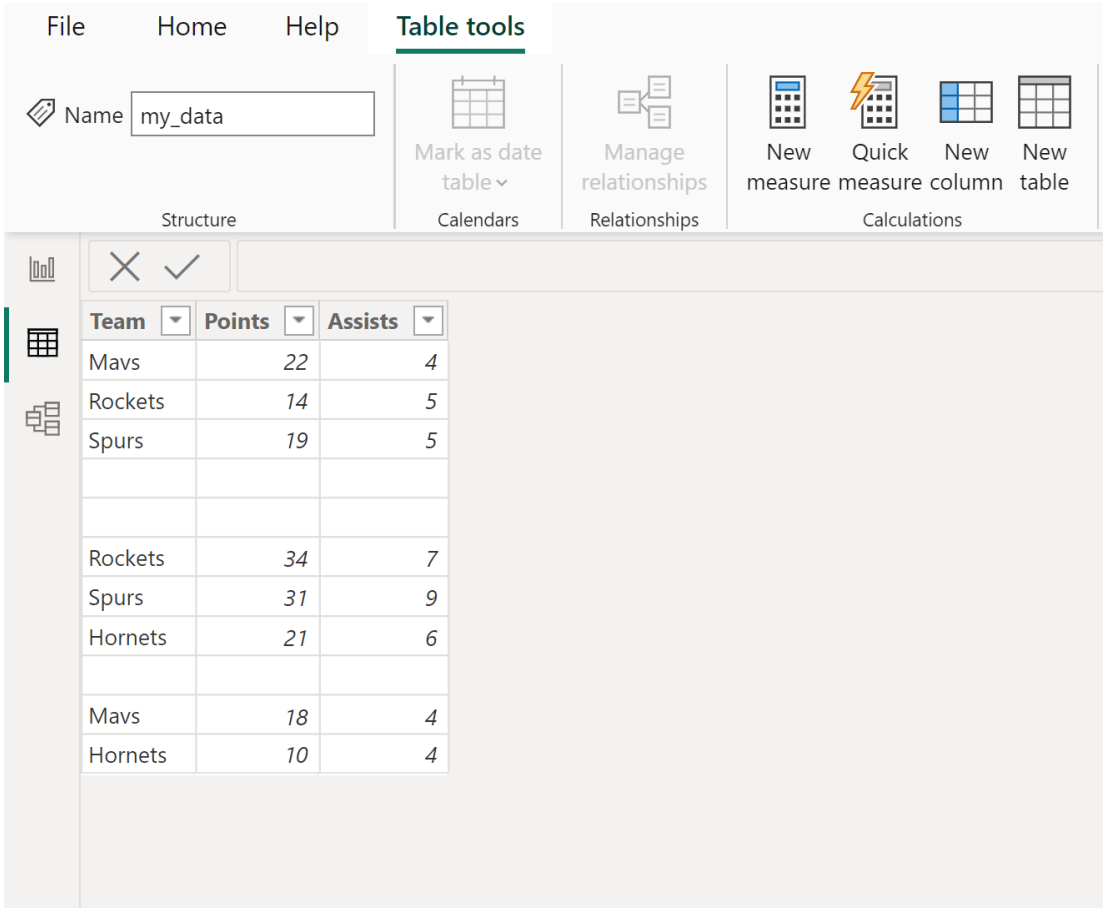
The Definitive Method: Mastering the Power Query Editor Interface

The **Power Query Editor** functions as the critical staging area where raw data is systematically cleaned, shaped, and refined before being formally loaded into the Power BI Data Model. Accessing this editor is the compulsory first step for nearly every data transformation activity. The graphical user interface (GUI) within Power Query has been thoughtfully designed to offer swift, point-and-click access to common data preparation tasks, and the removal of blank rows stands out as one of the most frequently utilized functions. This specific approach is strongly recommended for all users who require a rapid, dependable solution without the need to engage with the technical complexities of the M language (the core scripting language driving Power Query), delivering a transformative experience that is automatically translated into executable, repeatable steps.

The specific function we are targeting is housed within the **Remove Rows** group, which aggregates various options for pruning datasets based on diverse criteria, such as eliminating duplicates, errors, or, in this context, blank entries. It is essential to emphasize that this feature strictly targets rows that are entirely blank across every defined column. If a row contains legitimate data in even one column but Nulls in all others, the blanket **Remove Blank Rows** command will not eliminate it; addressing partially blank records demands the use of more sophisticated filtering techniques, which we will explore later in this guide. However, for datasets where accidental empty lines have been inadvertently imported from source systems, this dedicated button offers the definitive, efficient solution, capable of executing the transformation across massive datasets containing millions of rows in mere seconds.

To effectively demonstrate this powerful capability, we will examine a typical scenario involving imported data. Consider a dataset detailing basketball players that was poorly exported from its source, resulting in several scattered rows that contain no useful statistical information interspersed throughout the table structure. This sample data provides an ideal illustration of the exact problem the **Remove Blank Rows** feature is engineered to solve. Our primary objective is to ensure that the dataset retains only those records containing valid player statistics, thereby establishing a pristine foundation essential for accurate subsequent analysis, such as calculating detailed average scores or tracking specific team performance metrics.

We begin with the following imported table in Power BI, which contains information about various basketball players:



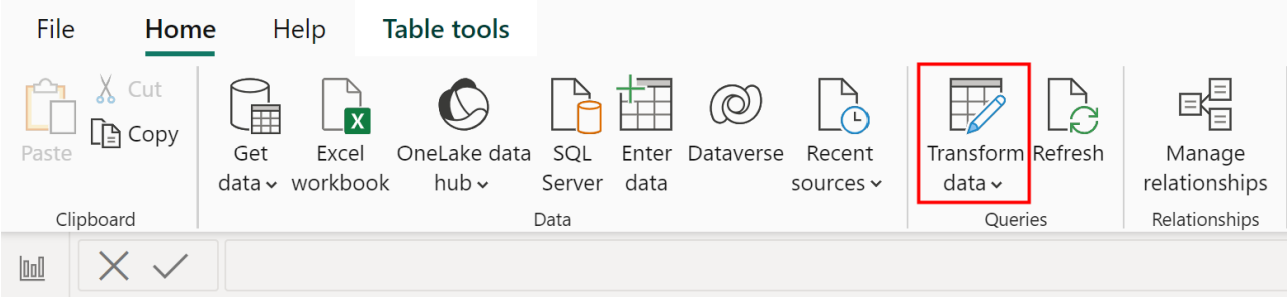
Team	Points	Assists
Mavs	22	4
Rockets	14	5
Spurs	19	5
Rockets	34	7
Spurs	31	9
Hornets	21	6
Mavs	18	4
Hornets	10	4

Observe clearly that multiple rows within this sample contain blank values across every single column. These are precisely the records we intend to eliminate swiftly and systematically using the specialized functionality embedded within the **Power Query Editor**.

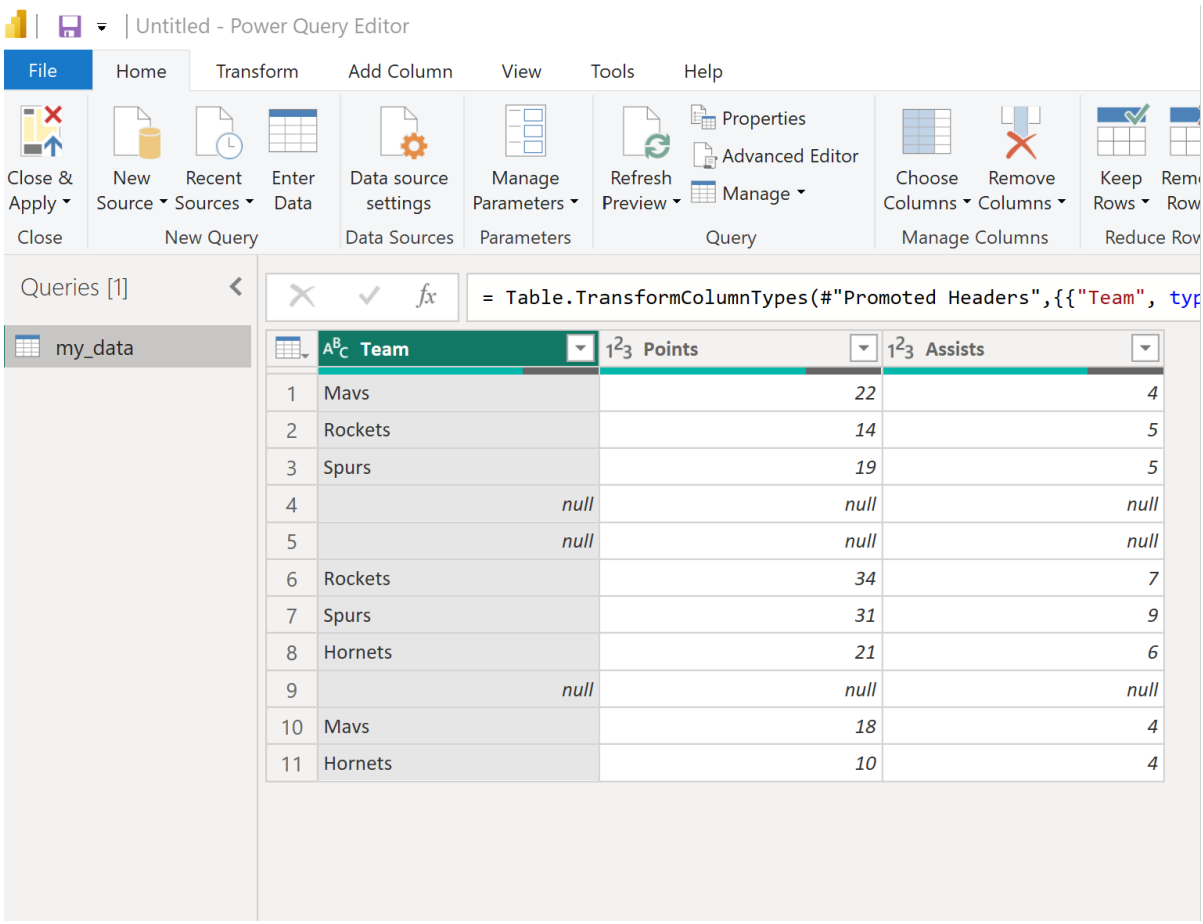
Step-by-Step Walkthrough: Executing the Blank Row Removal

The initiation of the data transformation process must always start from the main Power BI Desktop environment. The user's first action is to navigate to the dedicated area for data preparation by launching the Power Query interface. This action effectively pauses the data modeling view and shifts the analytical focus to the transformation engine, enabling the application of critical cleansing and shaping steps before the data is formally loaded or refreshed within the report view.

Accessing the Transformation Tool: To commence the process, click the **Home** tab located along the top ribbon in Power BI Desktop. Within this tab, locate and click the distinctive **Transform data** icon. This command instantly launches the separate, dedicated window that houses the powerful Power Query Editor.

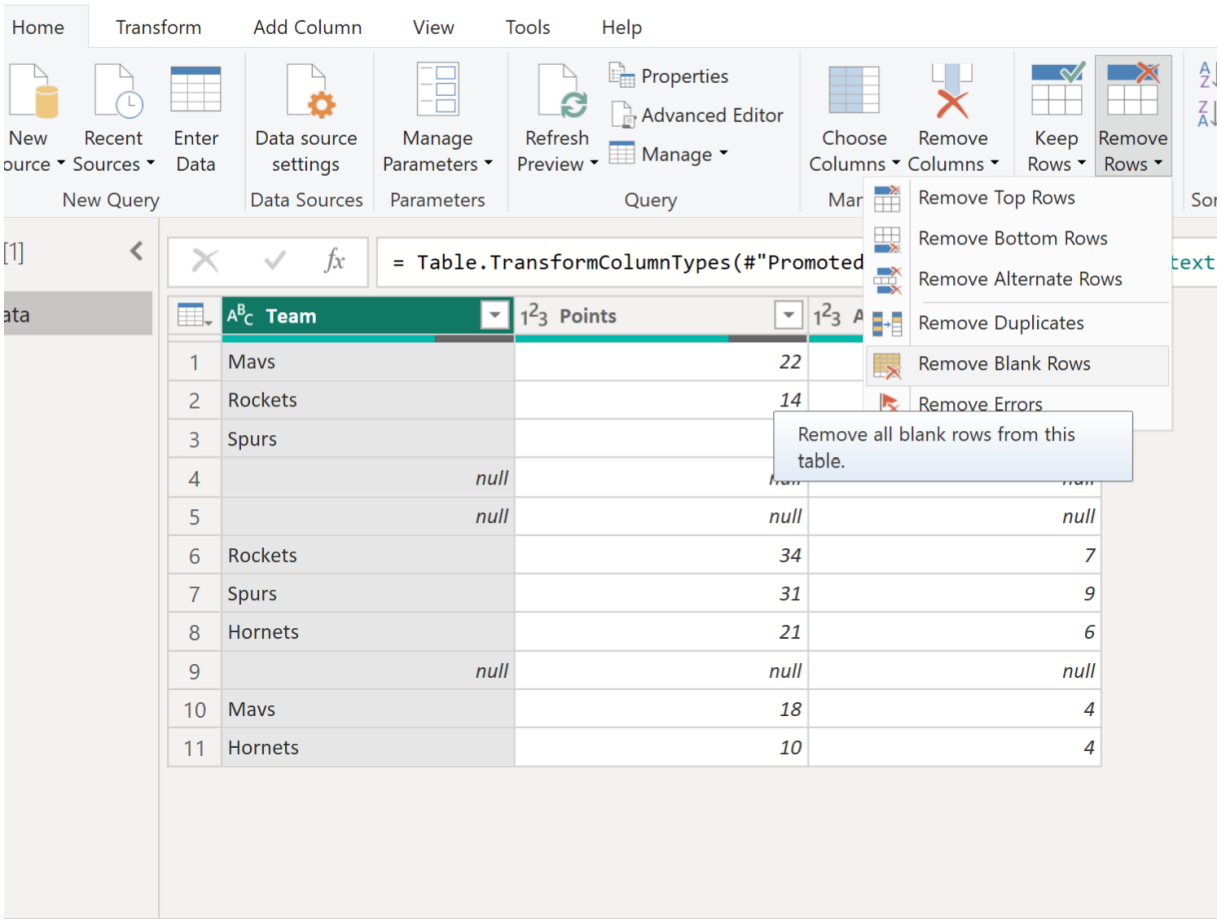


Executing this step successfully opens the **Power Query Editor** interface, displaying the currently selected table ready for transformation.



Executing the Removal Command: Once inside the **Power Query Editor**, verify that the target table is correctly selected in the Queries pane located on the left side. Navigate back to the **Home** tab within the Power Query ribbon. Locate the **Reduce Rows** group, and click the **Remove Rows** icon. This action triggers a dropdown menu that presents various options for row reduction. From this menu, select the specific command: **Remove Blank Rows**.

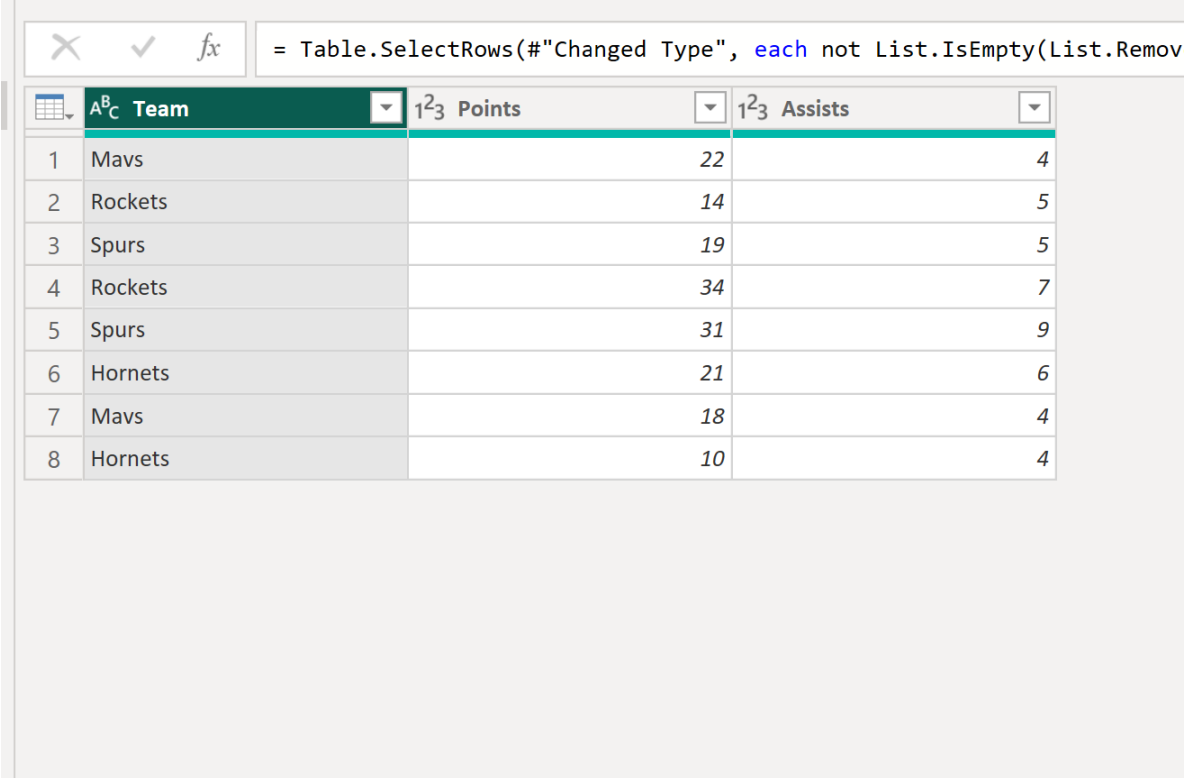
To successfully remove the blank rows, click the **Remove Rows** icon, and then proceed to click **Remove Blank Rows**:



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

Verifying the Transformation: Immediately upon executing the command, the Power Query Editor automatically processes the entire table. The result is instant and visible: all rows that contained only null or empty values across all columns are systematically eliminated. The resulting table is now condensed and contains only valid, data-bearing records. Crucially, this transformation step is meticulously logged in the **Applied Steps** pane on the right side of the editor. This logging ensures that the step is automatically reapplied every time the underlying data source is refreshed, guaranteeing persistent and reliable data quality moving forward.

This single action automatically removes all of the blank rows in the table, yielding a clean, optimized dataset:



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

Applying Changes to the Data Model: The final and most critical step is committing these finalized transformations back to the main Power BI Data Model. This is accomplished by clicking the **Close & Apply** button, strategically positioned under the **Home** tab of the Power Query Editor ribbon. As you exit the **Power Query Editor**, the system will prompt you to confirm the application of the changes made to the original table. Click **Yes** or **Close & Apply** to load the newly cleansed data into Power BI Desktop, making the implemented transformation permanent for this specific data source.

Advanced Cleansing: Differentiating Between Nulls and Empty Strings

While the dedicated **Remove Blank Rows** feature is highly efficient for blanket elimination, professional data cleansing frequently necessitates a more nuanced and sophisticated approach, particularly when addressing the subtle technical differences between various forms of missing data. Within robust data management systems, two primary indicators signal missing information: **Null values** and empty strings (represented as ""). A **Null value** fundamentally signifies the complete absence of a value--the data point simply does not exist or its status is unknown. Conversely, an empty string constitutes an actual, defined value--a text field containing zero characters--which the Power Query Editor often interprets differently from a technical Null. The general "Remove Blank Rows" function successfully handles both when they characterize an entire row, but when dealing with partially incomplete data, this critical distinction becomes paramount.

Consider complex scenarios where a row contains valid data in three columns but holds empty strings or nulls in a fourth, critical column (such as a unique key identifier). In this instance, the blanket "Remove Blank Rows" function will fail to eliminate the record because the row is not universally blank. For such surgical cleaning operations, analysts must employ specific column filtering or dedicated transformation steps. For example, if the "Player Name" column contains a mix of **Null values** and empty strings, the analyst must specifically filter that column to exclude those undesirable entries. This process is executed by clicking the filter icon on the column header, deliberately deselecting the "(blank)" option (which typically captures technical Nulls), and often also deselecting the explicit empty string entry if the column's data type permits its presence.

For highly robust and programmatic solutions, especially those spanning numerous columns, the powerful M language offers advanced transformation functions. A common and highly effective technique is utilizing the `Table.SelectRows` function within the Advanced Editor to define explicit, logical conditions for row retention. This allows the user to precisely specify criteria such as: "Only retain rows where Column A is not null AND Column B does not equal an empty string." This level of analytical precision is absolutely essential in complex data environments where the operational definition of "blank" must be strictly governed by specific business rules. Mastery of these advanced techniques is crucial for achieving and maintaining high standards of data cleansing and preparation within the Power Query environment.

Alternative Method: Strategic Filtering on Key Columns

While the dedicated **Remove Blank Rows** button offers undeniable convenience, its efficacy is predicated on the entire row being empty. When handling structured data imports, it is often a more strategic and professional approach to identify and remove rows based exclusively on the emptiness of one or two primary key or non-nullable columns. This alternative methodology ensures that the removal logic is intrinsically tied to the integrity of the most crucial data points, rather than depending on the collective state of every auxiliary column. This selective approach proves superior in situations where data sources might legitimately contain missing data in secondary columns (e.g., an empty "Notes" field) but must maintain values in core identifiers (e.g., "Transaction ID").

This process involves utilizing the standard filtering mechanism available on the header of every column within the **Power Query Editor**. When the filter icon is clicked, the editor presents options to exclude specific values from view. Crucially, if the column contains missing data, the filter list will prominently display an item labeled **(blank)**. Deselecting this item effectively targets both technical Null values and, depending on the column's assigned data type, often automatically includes empty strings as well. By unchecking the **(blank)** option and confirming the selection, the analyst explicitly instructs Power Query to remove all rows where the data in that specific column is definitively missing.

This highly targeted filtering approach translates into a precise step in the M code that resembles: `Table.SelectRows(#"Previous Step", each <> null)`. This explicit definition ensures that the transformation is narrowly focused and transparent, thereby significantly minimizing the risk of accidentally discarding potentially valuable rows that might be partially incomplete yet still contain necessary identifying information. For professional-grade data preparation and auditing, relying on targeted column filtering often provides superior control and greater transparency over the transformation logic compared to the broad, single-click removal tool, particularly when managing datasets with complex schemas or varying standards of data quality.

Best Practices for Transformation Efficiency and Quality Control

Maintaining consistently high data quality standards must be viewed as an ongoing, continuous commitment, not merely a one-time fix applied during initial import. When implementing transformations, such as the systematic removal of blank rows, several key best practices should be rigorously observed to guarantee the long-term integrity, efficiency, and maintainability of the Power BI data model. The first essential best practice involves meticulously renaming transformation steps within the **Applied Steps** pane. When the **Remove Blank Rows** button is used, Power Query assigns the step a generic name (e.g., "Removed Blank Rows"). Renaming this to something highly descriptive, such as "Removed Fully Blank Rows from Source Import," drastically improves documentation and makes the transformation logic immediately understandable to future maintainers or auditors.

Secondly, always ensure that correct data types are assigned immediately following the initial data ingestion. Incorrect data typing can severely compromise how missing values are interpreted by the engine. For instance, if a numeric column is mistakenly typed as "Text," a truly missing value might be interpreted as an empty string instead of a technical Null value, potentially confusing subsequent filtering steps. Type assignments should typically be executed early in the Power Query Editor workflow, ideally right after the source connection is established and column headers have been correctly promoted.

Finally, leverage Power Query's powerful profiling tools for rigorous quality control. Features such as **Column Quality** and **Column Distribution** (accessible under the View tab) provide immediate visual feedback on the percentage of error, empty, and valid values within each column. Both before and after applying the "Remove Blank Rows" transformation, analysts should diligently check these column quality metrics. A successful transformation should demonstrate a significant decrease in the percentage of empty values, unequivocally confirming that the transformation performed precisely as expected and substantially bolstering confidence in the cleanliness of the final dataset loaded into the report.

Summary of Techniques and Further Resources

Effective data preparation forms the essential bedrock of reliable business intelligence. The pervasive presence of blank or empty rows is a common data quality issue that must be addressed proactively to prevent skewed analytical calculations and performance degradation within Power BI reports. The most efficient and straightforward methodology for handling globally blank rows is the built-in **Remove Blank Rows** feature located within the **Power Query Editor**, providing a fast, GUI-driven solution that requires no coding. For scenarios demanding greater analytical granularity, leveraging targeted column filters or writing explicit transformation logic using the powerful M language allows analysts to precisely control which specific types of missing data are systematically removed.

By meticulously following the detailed steps--accessing the transformation editor, utilizing the dedicated removal button, and applying the committed changes--users can rapidly transform raw, potentially messy data into a clean, optimized table ready for advanced data modeling and visualization. The consistent application of these data cleansing techniques, combined with the diligent use of Power Query's profiling and auditing tools, ensures ongoing data integrity and maximizes the efficiency of the Power BI environment for all stakeholders.

The following tutorials offer explanations on performing other essential data management tasks in Power BI:

Understanding Null Values: A detailed guide on how the Power Query Editor internally handles Null values versus explicit empty strings.
Custom M Code: Instructions for writing custom functions tailored for advanced row filtering using the M language.
Data Type Management: Essential best practices for assigning and modifying column data types early in the Power Query workflow.