Reordering Legend Items in Power BI Charts: A Tutorial

Authored by

Mohammed looti

November 12, 2025

The Critical Need for Custom Legend Ordering in Power BI

In advanced data visualization, the arrangement of elements within a chart is just as important as the data itself. When building interactive reports in Power BI, the ordering of items displayed in the chart Legend field must often transcend simple alphabetical or numerical defaults. While Power BI automatically sorts these categorical items based on the underlying text or value, this default behavior rarely aligns with the logical flow of real-world processes, business chronology, or designated hierarchies.

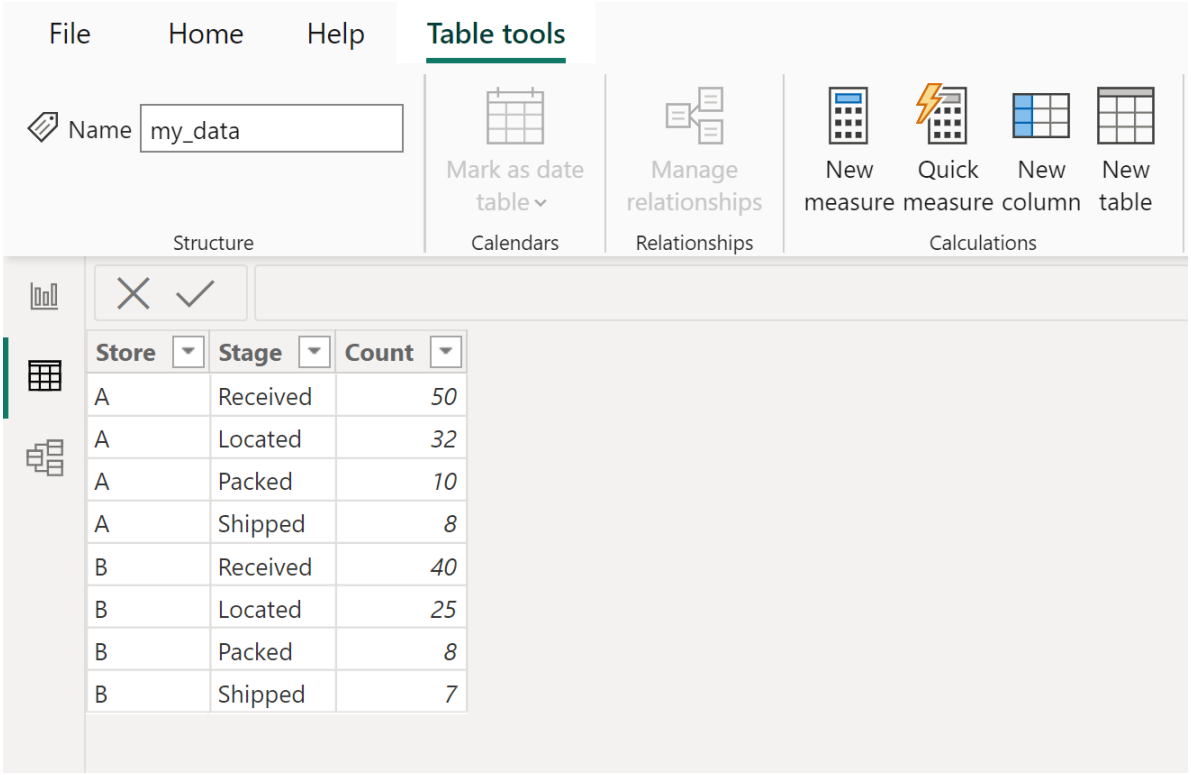
For example, if you are tracking stages in a manufacturing or fulfillment pipeline, the physical sequence of events--such as 'Initiated', 'In Progress', and 'Completed'--should be reflected accurately in the visualization. If the stages are sorted alphabetically, they might appear as 'Completed', 'In Progress', 'Initiated', which actively misleads the viewer and confuses stakeholders trying to interpret the operational flow. This disconnect compromises the integrity of the data narrative and diminishes the utility of the visual asset.

Achieving a custom, process-driven sequence is therefore an essential skill for any professional report creator. Fortunately, Power BI offers a robust mechanism to enforce a specific order. This technique leverages the power of DAX (Data Analysis Expressions) to create a dedicated numerical sorting column, which then dictates the precise order in which the original category names appear in the legend. Mastering this practice ensures that your visualizations are both technically accurate and chronologically meaningful.

Analyzing Power BI's Default Sorting Mechanism

To illustrate the necessity of custom ordering, let us examine a typical dataset. We are working with a table named **my_data**, which tracks the status of retail items through their various fulfillment stages. These key stages include: Received, Located, Packed, and Shipped. These stages represent a clear, chronological sequence in the inventory management process.

The structure below shows a snippet of the raw data, detailing the movement and current status of items across different retail operations. Notice the textual nature of the **Stage** column, which will be the source of Power BI's automatic sorting conflict.

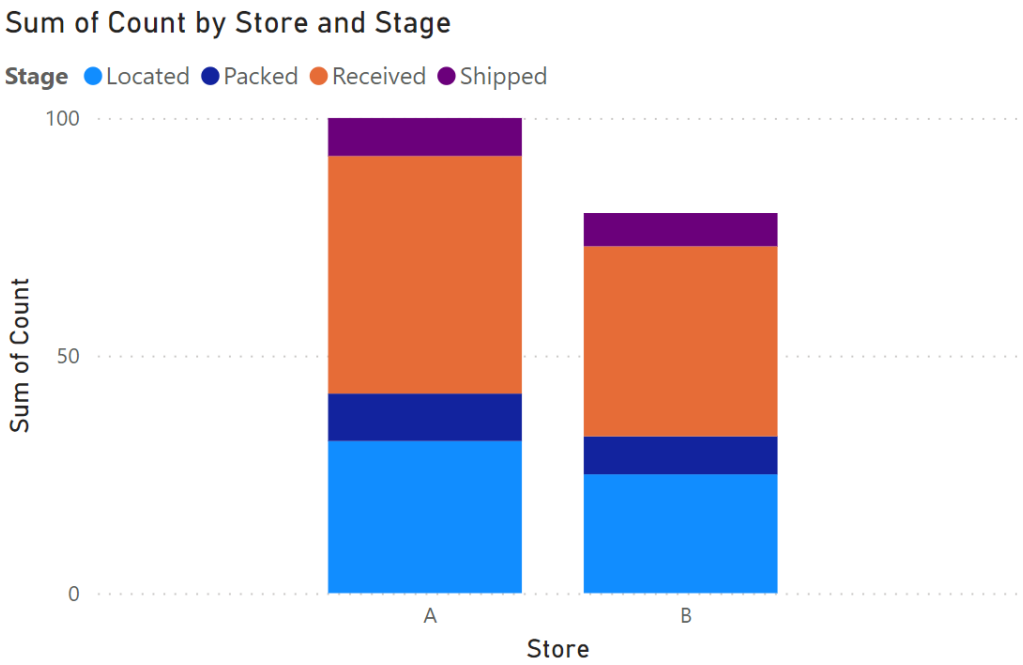


Store	Stage	Count
A	Received	50
A	Located	32
A	Packed	10
A	Shipped	8
B	Received	40
B	Located	25
B	Packed	8
B	Shipped	7

Our primary objective is to create a Stacked Column Chart where the segments within the bars, and subsequently the legend entries, strictly follow the established business sequence. This desired, chronological order must be:

Received
Located
Packed
Shipped

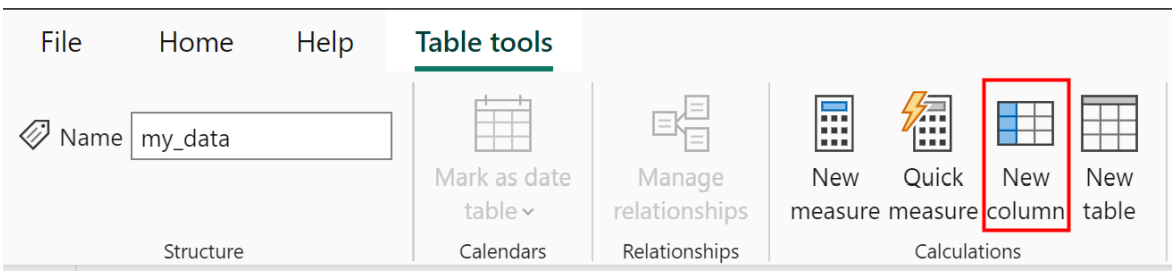
When we initially configure the visualization using the raw 'Stage' column, Power BI automatically imposes an alphabetical sort. The resulting visual flow is illogical and disruptive to the narrative, displaying the stages in the following, incorrect order: Located, Packed, Received, Shipped. This outcome clearly demonstrates why relying on default settings is insufficient when presenting sequential data, emphasizing the necessity of implementing the custom sort strategy detailed in the following steps.



Step 1: Defining the Custom Sequence with a DAX Calculated Column

The foundation of custom legend ordering is the assignment of a numerical rank to each category. This numerical rank will serve as the non-alphabetical sorting key. We begin this process by creating a new calculated column within our data model using DAX. This column will map the chronological stage name to a unique, sequential integer.

The specific ranking we need to establish, mirroring the desired workflow, is: Received (1), Located (2), Packed (3), and Shipped (4). To create this column, navigate to the **Table tools** ribbon in Power BI Desktop and select the **New column** option. This action opens the DAX formula bar, ready for input.

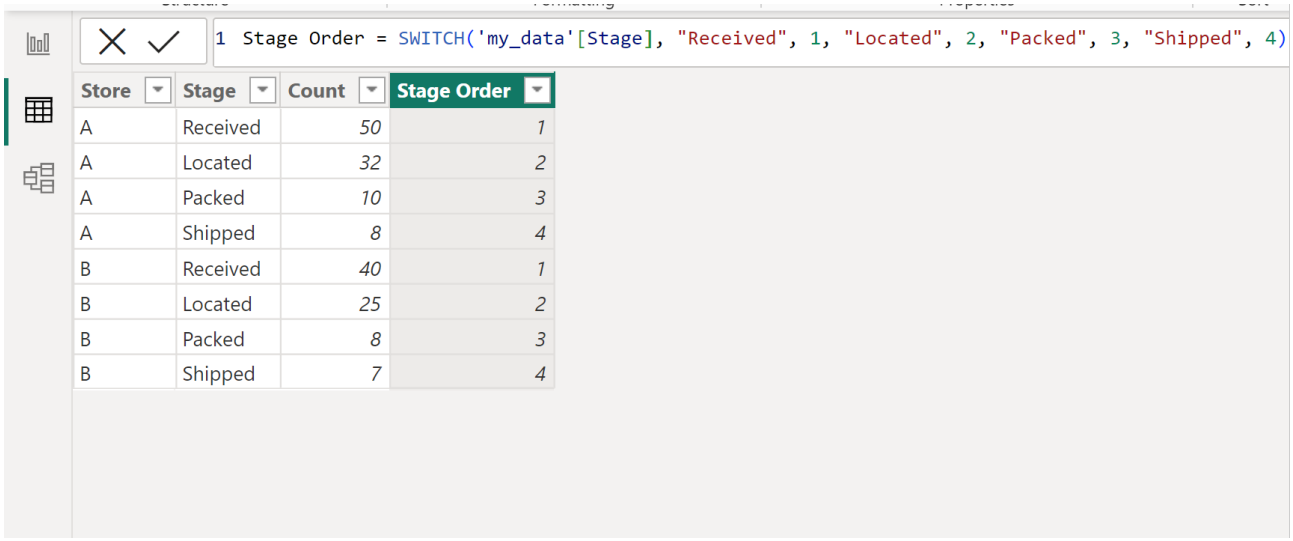


We will utilize the powerful SWITCH function in DAX, which is the most efficient tool for assigning different numerical outcomes based on discrete text inputs. Input the following DAX expression

precisely into the formula bar to define the custom stage hierarchy:

Stage Order = SWITCH('my_data', "Received", 1, "Located", 2, "Packed", 3, "Shipped", 4)

Upon execution, this formula generates the new numerical column, **Stage Order**, within the **my_data** table. This column now holds the definitive sorting key that will override Power BI's inherent alphabetical defaults. Reviewing the data view confirms that each stage name is correctly paired with its intended numerical rank.



Store	Stage	Count	Stage Order
A	Received	50	1
A	Located	32	2
A	Packed	10	3
A	Shipped	8	4
B	Received	40	1
B	Located	25	2
B	Packed	8	3
B	Shipped	7	4

Step 2: Ensuring Data Integrity by Creating a Category Copy

Although we have the numerical sorting column, a crucial best practice in professional Power BI data modeling dictates that we should avoid linking the custom sort directly to the original dimension column. This separation of concerns prevents unintended side effects, especially in complex models involving multiple relationships or calculated measures. Therefore, before applying the sort key, we must create an exact duplicate of the original 'Stage' text column.

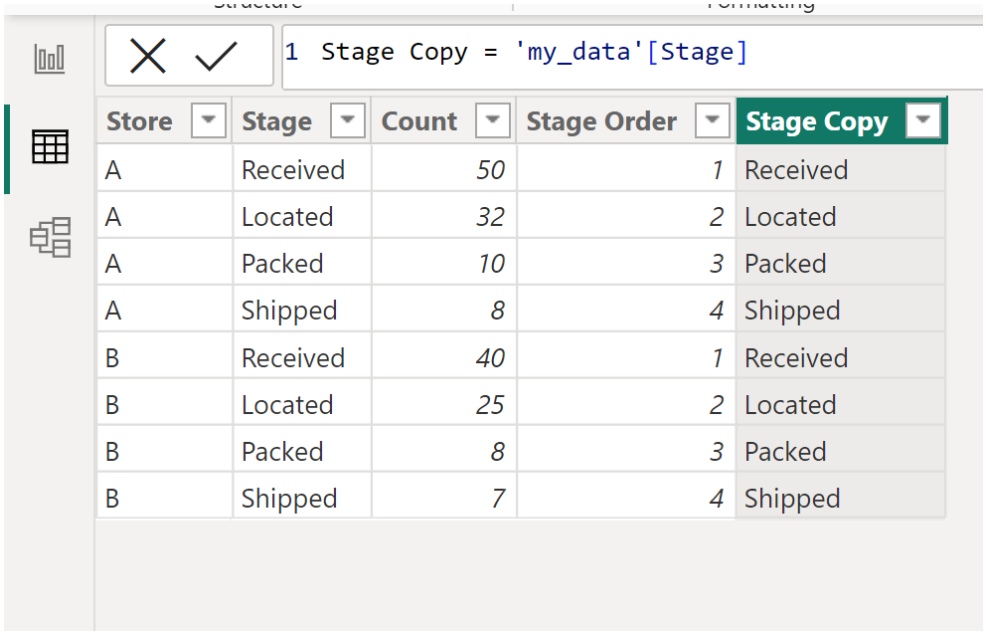
This duplicate column will be the one used in the visualization's Legend well, inheriting the custom sort without modifying the source data field. To create the replica, select the **New column** option again. The DAX required for this step is straightforward, simply referencing the existing column:

Stage Copy = 'my_data'

The resulting column, named **Stage Copy**, is an identical textual representation of the stage names. While seemingly redundant, this intermediate column is essential. It acts as the visible element in the report, allowing us to link it exclusively to the numerical sorting column (**Stage**

Order) without impacting the original 'Stage' column's behavior or structure.

This deliberate process of duplicating the category field ensures robust and stable reporting. By keeping the visible category and the numerical sort key distinct, we establish a clean and maintainable data model that can easily adapt to future changes or complexity.



Store	Stage	Count	Stage Order	Stage Copy
A	Received	50	1	Received
A	Located	32	2	Located
A	Packed	10	3	Packed
A	Shipped	8	4	Shipped
B	Received	40	1	Received
B	Located	25	2	Located
B	Packed	8	3	Packed
B	Shipped	7	4	Shipped

Step 3: Linking Category and Sort Columns in Power BI Desktop

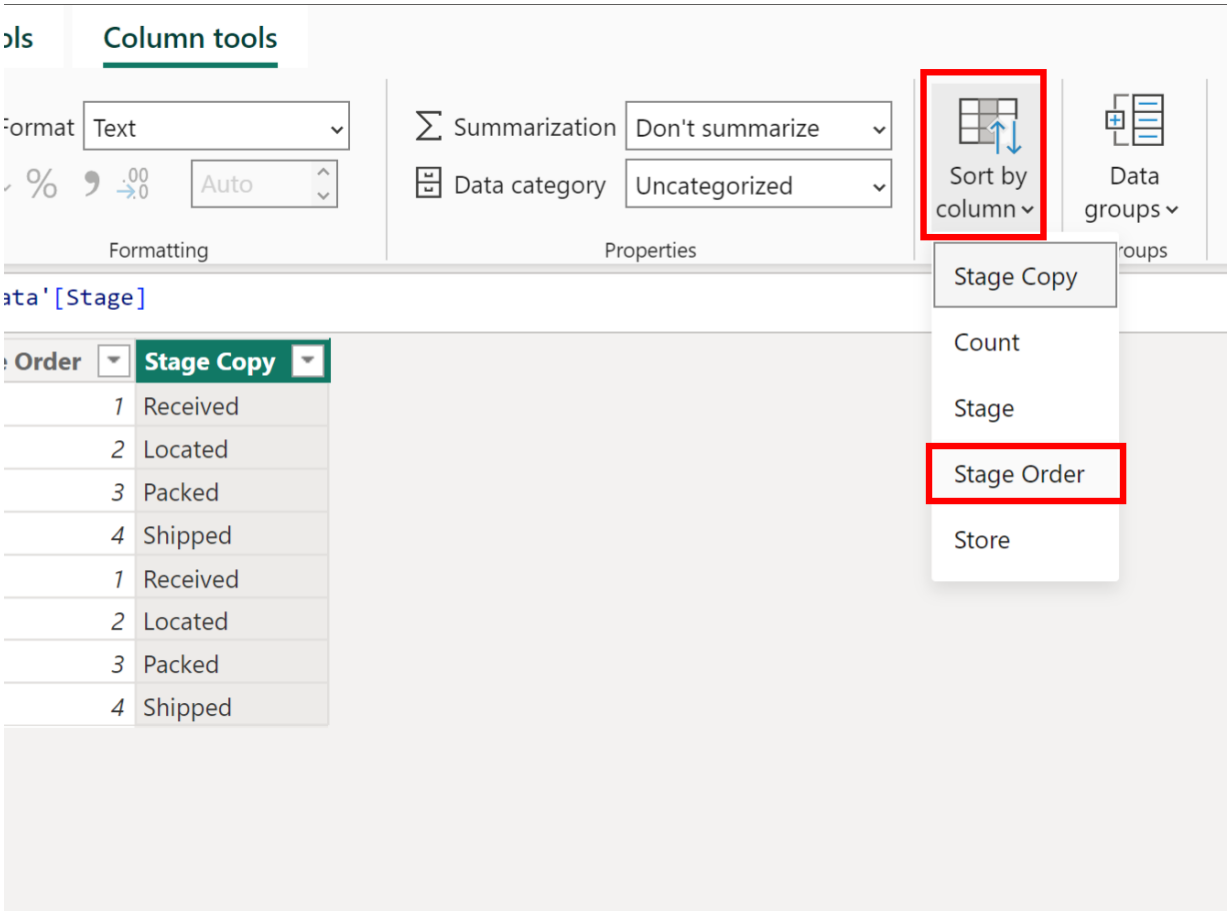
This step is the linchpin of the entire process, where the newly created columns are formally linked within the Power BI data environment. With both the categorical replica (**Stage Copy**) and the numerical sort key (**Stage Order**) ready, we must now instruct Power BI to use the numerical column to govern the display order of the text column.

Begin by navigating to the Data View in Power BI Desktop. Ensure that the text column, **Stage Copy**, is actively selected in the Fields pane. Next, locate the **Column tools** ribbon at the top of the interface. Within this ribbon, find and click the crucial **Sort by column** function.

Instead of allowing **Stage Copy** to sort itself alphabetically (the default setting), select **Stage Order** from the dropdown menu. This action establishes a binding instruction: "Whenever the **Stage Copy** field is placed into a visual, its text values must be sorted according to the corresponding numerical rank found in the **Stage Order** column."

Immediately after establishing this sort relationship, the custom order (1=Received, 2=Located, 3=Packed, 4=Shipped) is enforced globally for the **Stage Copy** dimension. Any visual utilizing this column will now automatically inherit the desired process-driven sequence, permanently resolving

the initial alphabetical sorting problem.

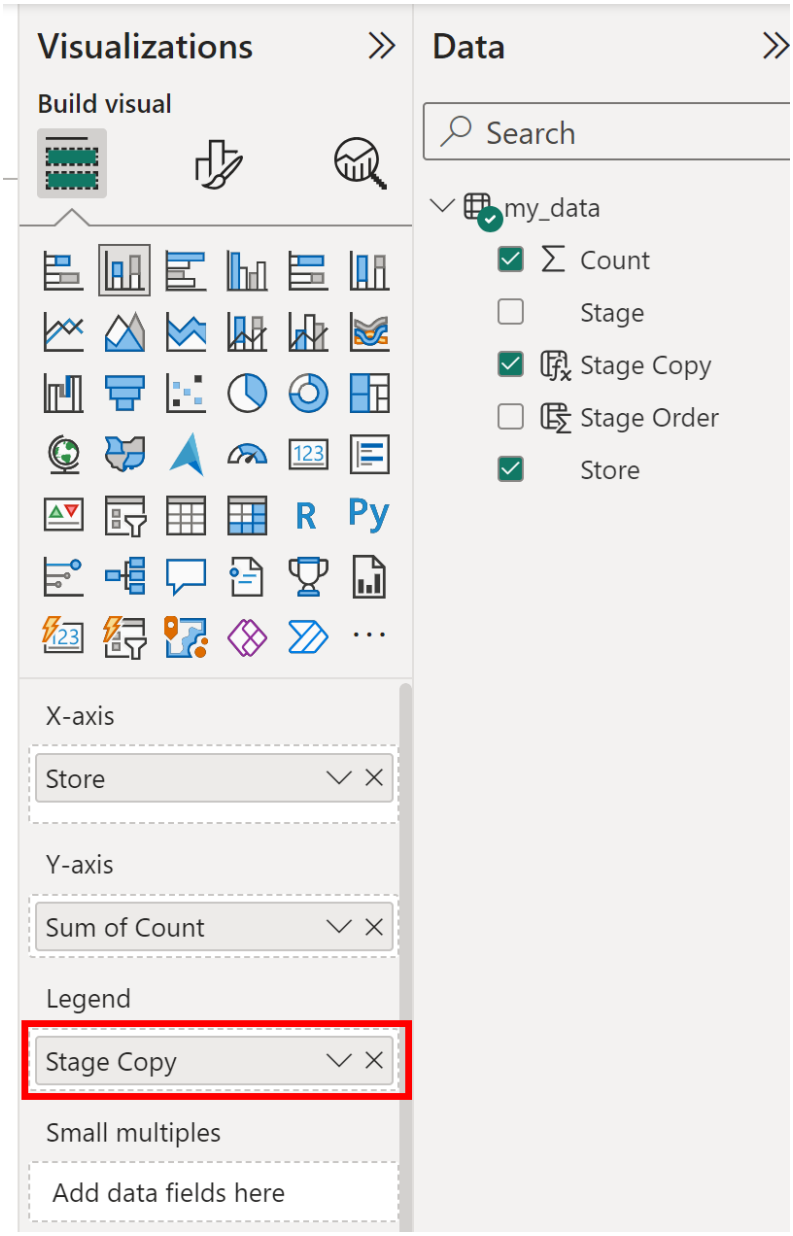


Step 4: Finalizing the Visual and Enhancing Readability

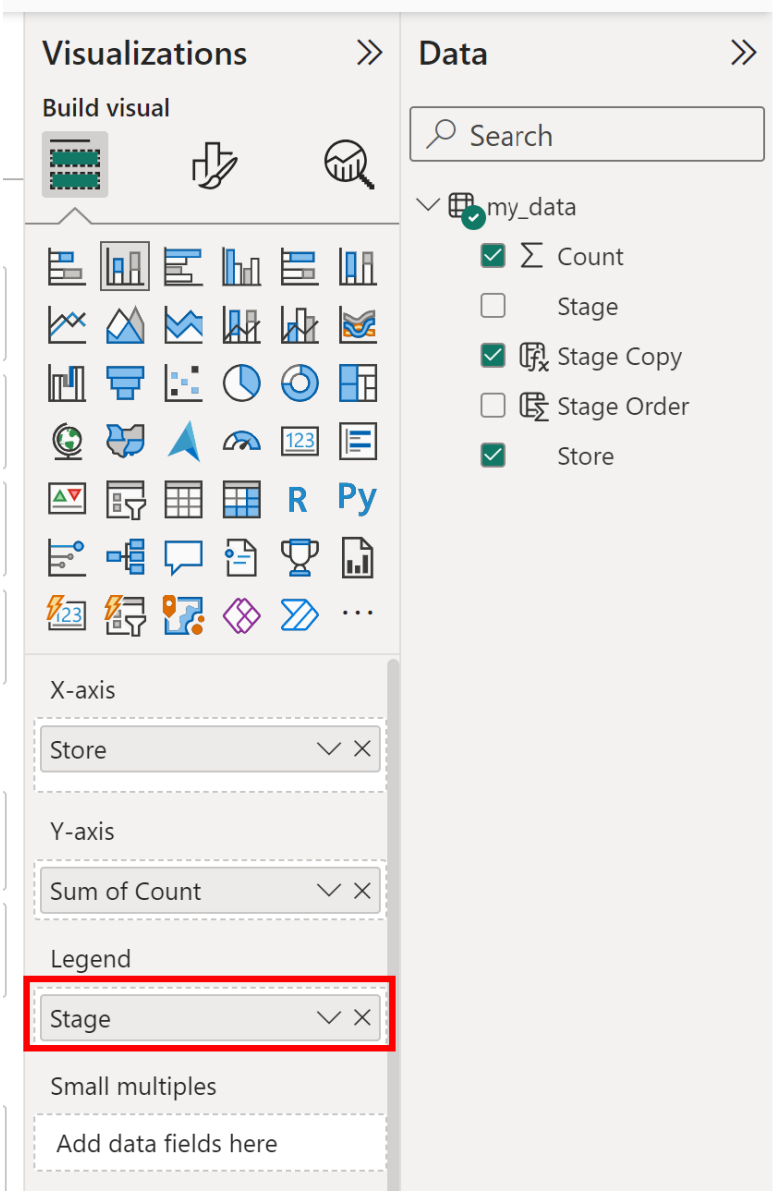
The final phase involves updating the report canvas to utilize the newly configured, sorted field. Return to the Report View and select the target visualization--in this case, the stacked column chart--to activate its properties in the Fields pane.

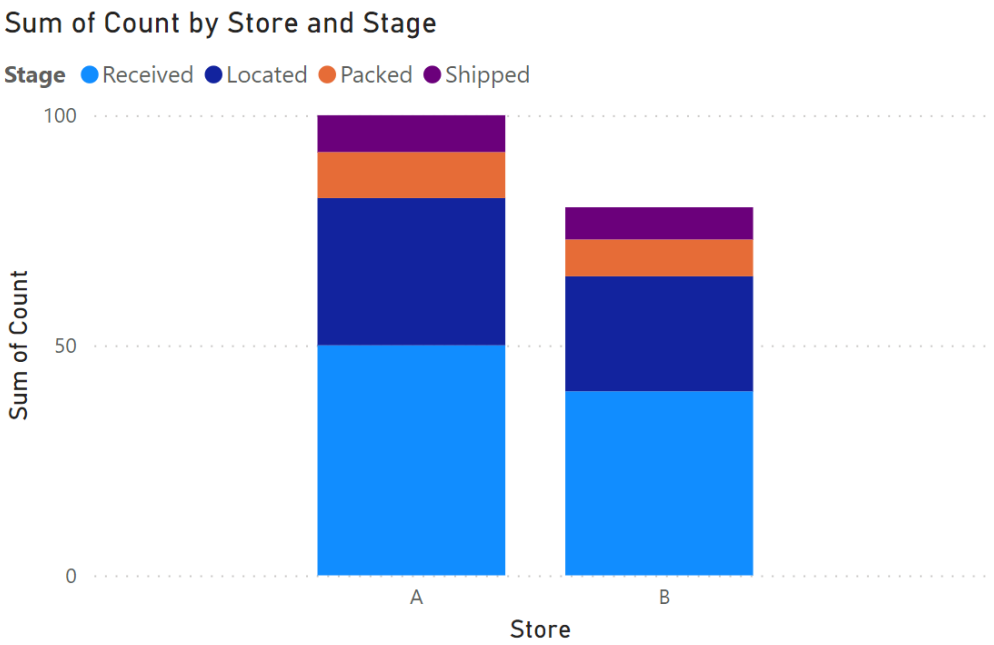
Locate the original 'Stage' field currently placed in the **Legend** well. Remove this field, and then drag the custom-sorted **Stage Copy** field into the same **Legend** well. Because **Stage Copy** is now linked to **Stage Order**, the chart segments and the legend entries will instantly reorganize themselves to reflect the precise chronological sequence.

Although the functionality is correct, the field name 'Stage Copy' may appear unprofessional to the end-user. To ensure optimal report readability, we can rename the field specifically within the context of this visualization. Right-click on the **Stage Copy** field located in the **Legend** well, select **Rename for this visual**, and then change the display name back to the user-friendly label: **Stage**.



This final aesthetic polish ensures that the visualization is not only technically accurate in its sorting but also professionally labeled for consumption. The resulting stacked column chart now provides clear and accurate insights into the retail fulfillment process, displaying stages in the exact, business-logic order (Received, Located, Packed, Shipped) defined by the custom numerical column.


Visualizations
Build visual
Data
Search
my_data
Count
Stage
Stage Copy
Stage Order
Store
X-axis
Store
Y-axis
Sum of Count
Legend
Stage
Small multiples
Add data fields here



Conclusion: Achieving Granular Control Over Data Visualization

The ability to implement a custom sort order for legend items in Power BI is a hallmark of sophisticated data modeling. By moving beyond the limitations of default alphabetical sorting, report creators can align the visual data hierarchy precisely with real-world chronology or business significance. The technique involves defining the desired sequence using a numerical sort column, created efficiently with the DAX SWITCH function, and then linking this key to a copy of the category field.

This methodology ensures granular control, resulting in high-quality, professional data visualization. Furthermore, because the custom sort behavior is defined at the data model level, the **Stage Copy** column can be reused across different visuals and reports, guaranteeing consistency wherever the dimension is utilized. Mastering this core technique ensures that your legends always communicate the correct narrative, regardless of how the underlying text values are structured.

Additional Resources for Power BI Mastery

To further enhance your skills in managing complex datasets and report presentation within Power BI, explore these related resources that address common data transformation and visualization challenges: