Multi-Source Filtering using Parameters in Tableau

by Mohit Kumar – Feb 22, 2025

In 2022, I encountered a challenge where I needed to filter my dashboard based on fields from different data sources. While joining or blending could work for filtering data from multiple sources, the real issue was providing users with an intuitive interface that allowed them to view detailed information about their selection. Let me illustrate the problem using Sample Superstore data.

Suppose you've created a dashboard and need to filter it based on specific sub-categories. These sub-categories are stored in a separate data source with the following structure:

Report Owner	View	Report Object	Report Object Name	Creation Date	Update Date
Mia Caldwell	ALL	1	Accessories	Aug 18 ,2024	Oct 20, 2024
Mia Caldwell	ALL	2	Appliances	Aug 18 ,2024	Oct 20, 2024
Mia Caldwell	ALL	3	Art	Aug 18 ,2024	Oct 20, 2024
Mia Caldwell	ALL	4	Binders	Aug 18 ,2024	Oct 20, 2024
Mia Caldwell	ALL	5	Bookcases	Aug 18 ,2024	Oct 20, 2024
Mia Caldwell	ALL	6	Chairs	Aug 18 ,2024	Oct 20, 2024
Mia Caldwell	ALL	7	Copiers	Aug 18 ,2024	Oct 20, 2024
Mia Caldwell	ALL	8	Envelopes	Aug 18 ,2024	Oct 26, 2024
Mia Caldwell	ALL	9	Fasteners	Aug 18 ,2024	Oct 20, 2024
Mia Caldwell	ALL	10	Furnishings	Aug 18 ,2024	Oct 20, 2024
Mia Caldwell	ALL	11	Labels	Aug 18 ,2024	Oct 20, 2024
Mia Caldwell	ALL	12	Machines	Aug 18 ,2024	Oct 20, 2024
Mia Caldwell	ALL	13	Paper	Aug 18 ,2024	Oct 20, 2024
Mia Caldwell	ALL	14	Phones	Aug 18 ,2024	Oct 20, 2024
Mia Caldwell	ALL	15	Storage	Aug 18 ,2024	Oct 20, 2024
Mia Caldwell	ALL	16	Supplies	Aug 18 ,2024	Oct 20, 2024
Mia Caldwell	ALL	17	Tables	Aug 18 ,2024	Oct 30, 2024
Mia Caldwell	ALL F&T	1	Accessories	Nov 12, 2024	Nov 12, 2024
Mia Caldwell	ALL F&T	7	Copiers	Nov 12, 2024	Nov 12, 2024
Mia Caldwell	ALL F&T	12	Machines	Nov 12, 2024	Nov 12, 2024
Mia Caldwell	ALL F&T	14	Phones	Nov 12, 2024	Nov 12, 2024
Mia Caldwell	ALL F&T	5	Bookcases	Nov 12, 2024	Nov 12, 2024
Mia Caldwell	ALL F&T	6	Chairs	Nov 12, 2024	Nov 12, 2024
Mia Caldwell	ALL F&T	10	Furnishings	Nov 12, 2024	Nov 12, 2024
Mia Caldwell	ALL F&T	17	Tables	Nov 12, 2024	Nov 12, 2024
Ethan Brooks	ALL	1	Accessories	Nov 15, 2024	Nov 15, 2024
Ethan Brooks	ALL	7	Copiers	Nov 15, 2024	Nov 15, 2024
Ethan Brooks	ALL	12	Machines	Nov 15, 2024	Nov 15, 2024
Ethan Brooks	ALL	14	Phones	Nov 15, 2024	Nov 15, 2024
Ethan Brooks	Office	11	Labels	Oct 12, 2024	Oct 12, 2024
Ethan Brooks	Office	13	Paper	Oct 12, 2024	Oct 12, 2024
Ethan Brooks	Office	15	Storage	Oct 12, 2024	Oct 12, 2024
Ethan Brooks	Office	16	Supplies	Oct 12, 2024	Oct 12, 2024

Each Report Owner can have multiple views, which they can update. Each view consists of several sub-categories, with the corresponding information stored in the Report Object and Report Object Name. The goal is to enable users to filter the main dashboard based on their selected view.

Now that we have a clear understanding of what is needed, let's briefly explore why we cannot use other methods and what approach we can take instead.

Blending: Blending does have limitations when it comes to providing clear and organized selection information, as it often lacks the ability to provide seamless interaction between data sources in a user-friendly way.

Joins: Joins can significantly increase data size, especially when combining large tables, which could lead to performance issues. However, in some cases, joins can be necessary when combining data that naturally belongs together. The issue here is that it might result in more data than required, leading to inefficiencies.

Relationships: Tableau's relationships allow for more flexible combinations of data sources, but they can limit your ability to perform complex calculations across those sources, especially if the fields are at different granularities or involve different levels of detail. Additionally, if you're using published data sources, there are a few additional limitations to consider.

So, what should we do? The answer lies in using parameters and actions.

Allow me to guide you through my solution to this problem.

I've imported the data into Tableau, and this is how it appears:

View	Report Owner	Report Object	Report Object Name	Creation Date	Update Date
ALL	Ethan Brooks	1	Accessories	15-11-2024	15-11-2024
		7	Copiers	15-11-2024	15-11-2024
		12	Machines	15-11-2024	15-11-2024
		14	Phones	15-11-2024	15-11-2024
	Mia Caldwell	1	Accessories	18-08-2024	20-10-2024
		2	Appliances	18-08-2024	20-10-2024
		3	Art	18-08-2024	20-10-2024
		4	Binders	18-08-2024	20-10-2024
		5	Bookcases	18-08-2024	20-10-2024
		6	Chairs	18-08-2024	20-10-2024
		7	Copiers	18-08-2024	20-10-2024
		8	Envelopes	18-08-2024	26-10-2024
		9	Fasteners	18-08-2024	20-10-2024
		10	Furnishings	18-08-2024	20-10-2024
		11	Labels	18-08-2024	20-10-2024
		12	Machines	18-08-2024	20-10-2024
		13	Paper	18-08-2024	20-10-2024
		14	Phones	18-08-2024	20-10-2024
		15	Storage	18-08-2024	20-10-2024
		16	Supplies	18-08-2024	20-10-2024
		17	Tables	18-08-2024	30-10-2024
ALL F&T	Mia Caldwell	1	Accessories	12-11-2024	12-11-2024
		5	Bookcases	12-11-2024	12-11-2024
		6	Chairs	12-11-2024	12-11-2024
		7	Copiers	12-11-2024	12-11-2024
		10	Furnishings	12-11-2024	12-11-2024
		12	Machines	12-11-2024	12-11-2024
		14	Phones	12-11-2024	12-11-2024
		17	Tables	12-11-2024	12-11-2024
Office	Ethan Brooks	11	Labels	12-10-2024	12-10-2024
		13	Paper	12-10-2024	12-10-2024
		15	Storage	12-10-2024	12-10-2024
		16	Supplies	12-10-2024	12-10-2024

We will provide the user with a button that they can click to filter the corresponding sub-categories.

Let's create a couple of calculations:

Update Date

// Update the Update Date to the maximum update date for a view.

```
{FIXED [Report Owner], [View]: MAX([Update Date])}
```

@Previous Value

// Concatenate the result of the current row with the previous row, which will be used for searching the text.

```
""" + STR(MIN([Report Object])) + "'," + PREVIOUS_VALUE('')
```

@Last

// This checks if the current row is the last row for a view.

```
LAST() = 0
```

@PV | RO

// Concatenate all the Report Object Names in a proper format.

```
PREVIOUS_VALUE('')
+
IF FIRST() = 0 THEN ""
ELSEIF LAST() = 0 THEN " & "
ELSE ", "
END
+
MIN([Report Object Name])
```

@Applied?

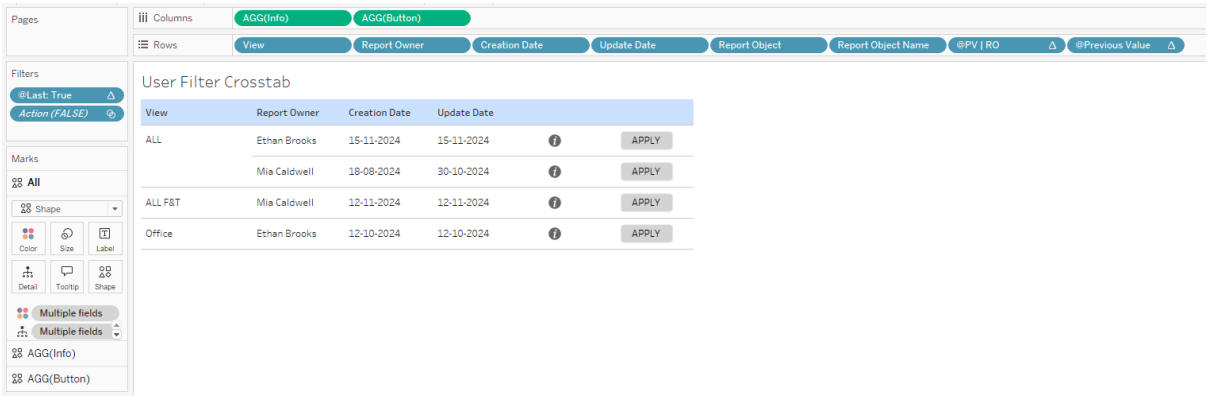
// To modify the colour and text of the selection.

```
IF [View] = [P | View] AND [Report Owner] = [P | Report Owner]
   THEN "APPLIED"
ELSE "APPLY"
END
```

Now that we have our calculations ready, let's build the view.

1. Add the following fields to rows: [View], [Report Owner], [Report Object], [Report Object Name], [Create Date], [Update Date], [@Last], [@Previous Value], and [@PV | RO]. Ensure that all the fields are discrete.
2. Ensure that all your table calculations ([@Last], [@Previous Value], and [@PV | RO]) are computed along [Report Object] and [Report Object Name].
3. Create a dummy axis named **Info** using MIN(0) on columns. Change the chart type to "Shape" and set the shape to an information icon.
4. Add another dummy axis named **Button** using MIN(0) to columns. Change the chart type to "Shape" again and set the shape to a rectangular icon. Add the [@Applied?] calculation to both the text and colour marks. Align the text to Centre.
5. Move the [@Last] field to the filter shelf and select **TRUE**.
6. Clear the tooltip from the **All** Marks card and add <AGG(@PV | RO)> to the **Info** Marks card:

With a bit of formatting, your crosstab should look like the following:



Now, let's create three string parameters: **[P | Report Objects]**, **[P | Report Owner]**, and **[P | View]**, with no value set as the current value.

Add three parameter actions to update the parameters above. Set them to run on select, keeping the current value with a clear selection. Use the following mappings:

Target Parameter: **[P | Report Objects]**

Source Field: **[@Previous Value]**

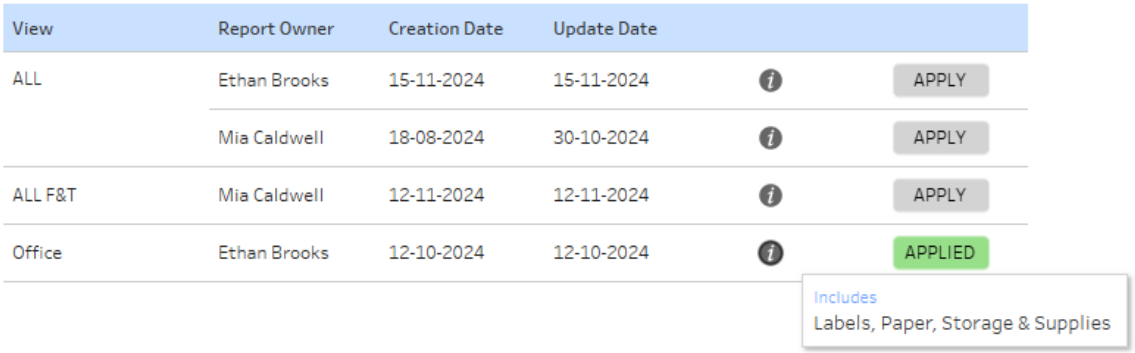
Target Parameter: **[P | Report Owner]**

Source Field: **[Report Owner]**

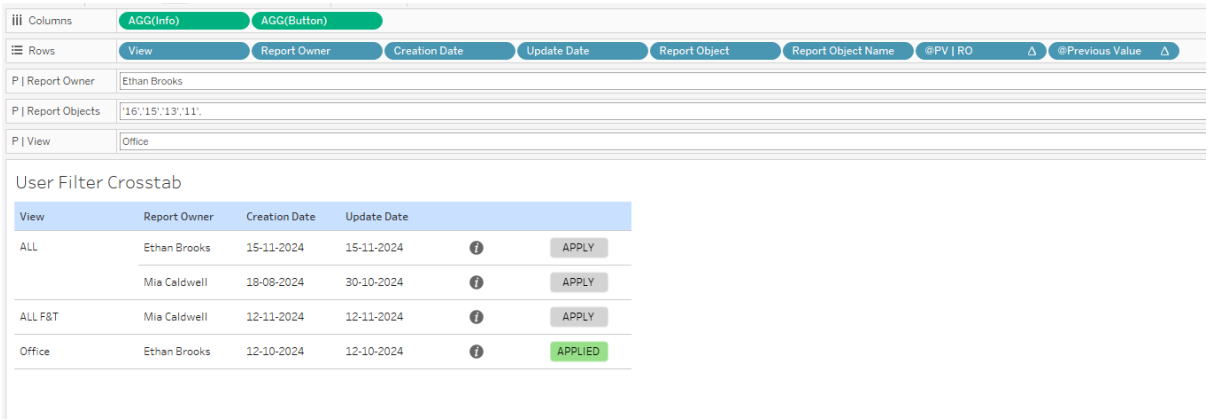
Target Parameter: **[P | View]**

Source Field: **[View]**

Hover over the information icon ⓘ to see which sub-categories are part of the view.



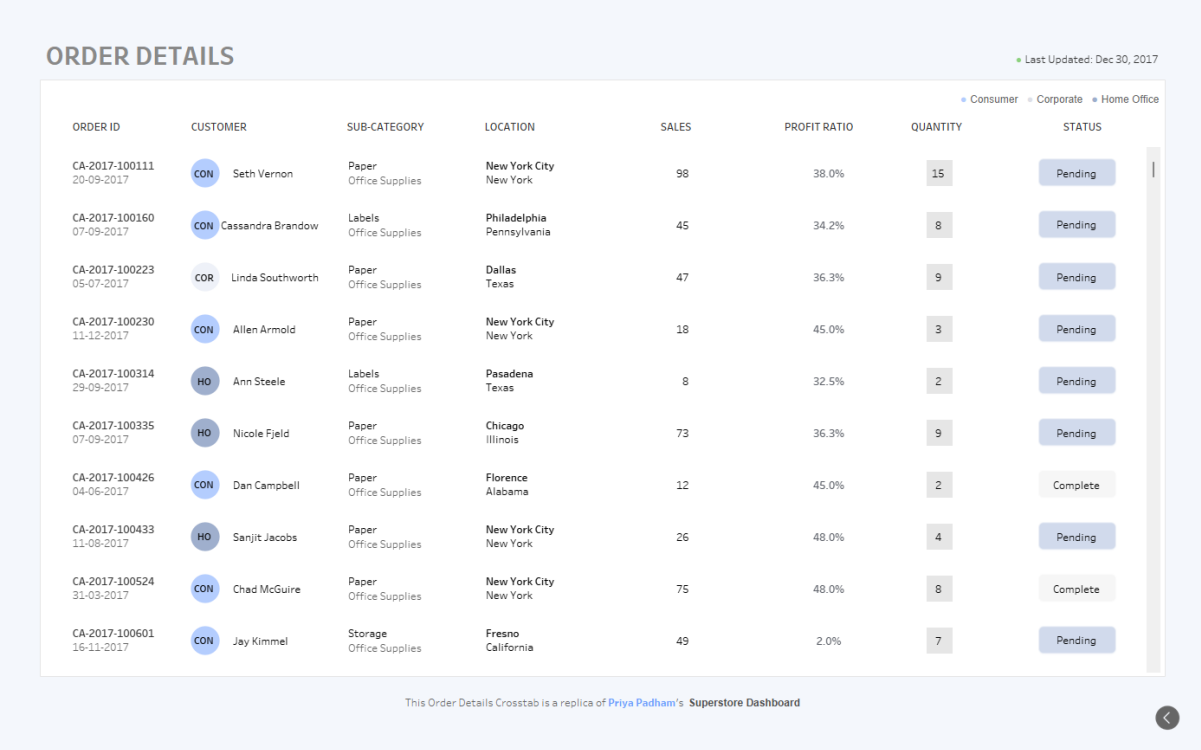
Select a combination of **View** and **Report Owner** by clicking the **Apply** button. This will populate the parameters accordingly.



I created the **Order Details** dashboard, which is almost a replica of the **Sample Superstore – Order Details** dashboard by **Priya Padham**.

You can view her dashboard by clicking here.

Now, let's apply a filter to it.



To filter this dashboard with our selection, we just need to add a filter. So, let's create one last calculation.

@Filter Report Objects

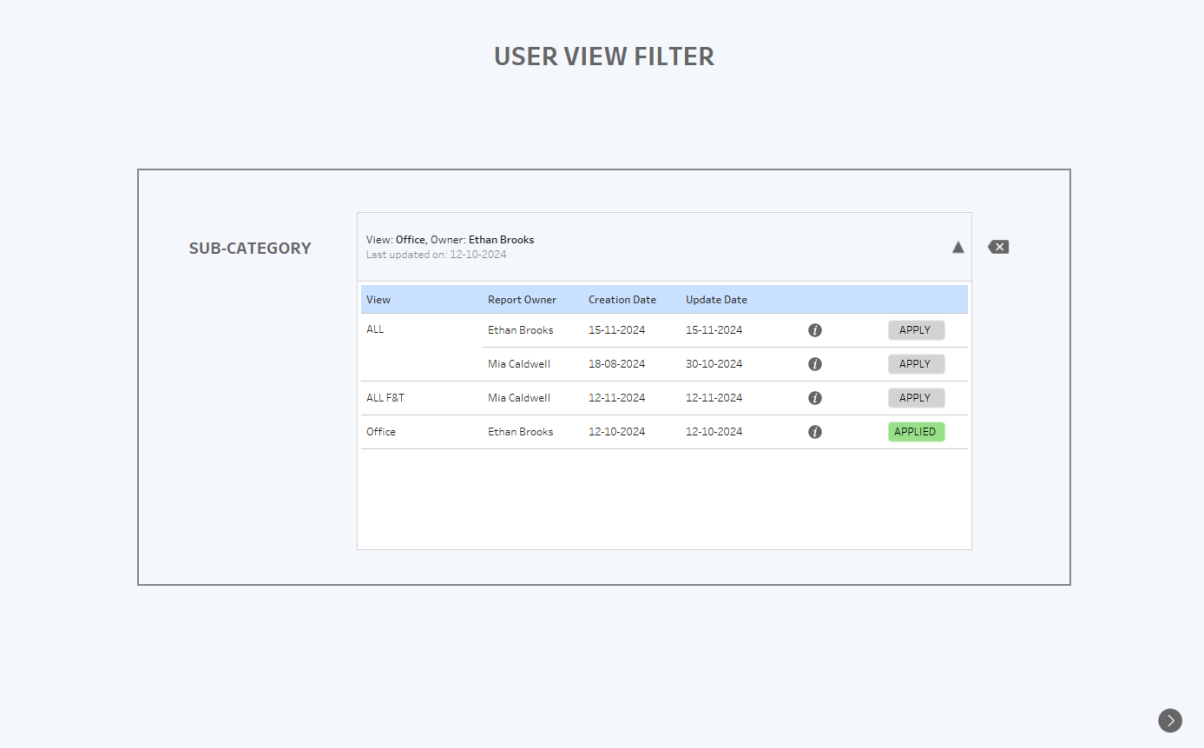
```
// This will search for the Report Object in the parameter and return true if a match is found.

IF [P | Report Objects] = "" THEN TRUE
ELSE
   IF FIND([P | Report Objects], "'" + STR([Sub-Cat ID]) + "',") = 0
   THEN FALSE
   ELSE TRUE
   END
END
```

Add **[@Filter Report Objects]** to your main crosstab and select **TRUE**.

That's it! Your dashboard should now be able to filter using a different data source.

I created a separate dashboard for making this selection. If you have multiple such views, you can add sections using a similar approach.



The solution works, but there is a downside. Since the filtering occurs after searching for values in the parameters, it could slow down performance if there are too many report objects.

Additionally, adding multiple views requires a lot of work.

I'd love to hear your thoughts.

Thanks!

Mohit