CHAPTER 59

Using Level of Detail Expressions to Create Benchmarks

One of the most powerful features that shipped with Tableau 9.0 was "Level of Detail" (LOD) functionality, which provides a syntax for explicitly assigning a different level of detail, or granularity, to a measure. This unlocks many valuable analysis opportunities, with just a few outlined in Tableau's post, Top 15 LOD Expressions. I admit that I've probably leaned on this functionality *too* much since its release because it is such a handy way to get the exact results I am looking for.

This chapter shares one of my favorite uses for LOD expressions: the ability to create a constant measure that can be used as a benchmark. For example, sometimes you want to compare the performance of a KPI now versus a KPI last year. Another example is in split testing, where it is common to need to compare the performance of a variant recipe to the performance of a control recipe. In the past, table calculations were the easiest way to create these comparisons, but the simple LOD calculation I'm about to share makes the analysis slightly more foolproof because you do not need to worry about the scope or direction of a table calculation.

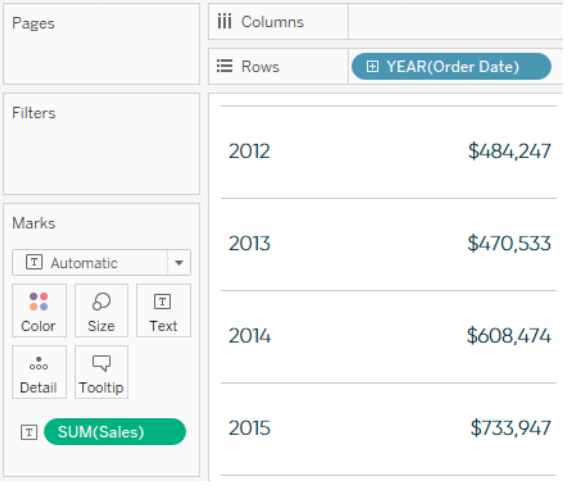
This chapter shares a specific use case for using LOD expressions, but is not a comprehensive training on what LOD expressions are and how they work. If you need more information, I encourage you to read the Tableau whitepaper, "Understanding Level of Detail (LOD) Expressions".

How to Use Tableau Level of Detail (LOD) Expressions to Create Benchmarks

There are several analyses where a fixed comparison measure would be valuable. To name just a few: the performance during a specific year, a goal, a competitor, or the

performance of a control recipe. For the purposes of this illustration, I'm going to use the Sample - Superstore data to compare the sales each year to the sales in the year 2012.

Here's a simple table showing sum of sales by year:

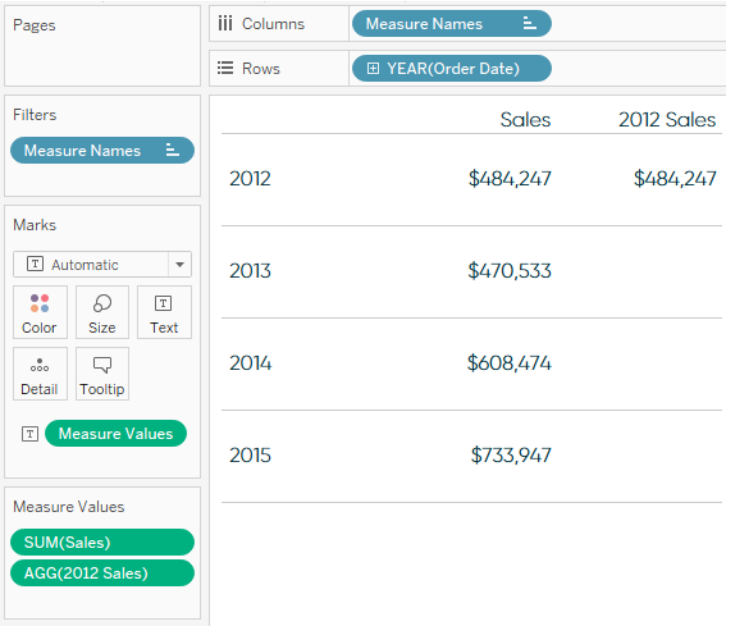


As you can see, we have four years of sales data. Let's pretend that because 2012 is the oldest year in our company's dataset, management wants to use 2012 as a benchmark for the lowest acceptable sales number. So we will create a comparison metric that isolates the sales in 2012. I realize this isn't the best example of a true business scenario, but work with me momentarily—I'm showing how you can isolate a comparison metric without using table calculations.

In order to isolate 2012 sales, your first instinct may be to create a calculated field that looks like this:

```
SUM(IF YEAR([Order Date]) = 2012 THEN SALES END)
```

I'm going to go ahead and make a calculated field with this formula and add it as a second column in the view:



Our calculated field worked, but you can see that only the row for 2012 is populated with the 2012 Sales amount. That's because Tableau is looking at each row and running our IF/THEN logic. It looks at the first row and sees that the year of 2012 matches the year in the calculated field, so it shows a result. It then looks at the second row, which is for 2013, sees that YEAR([Order Date]) does not equal 2012, so it does not display the sales amount. The same is true for 2014 and 2015, so there are nulls for 2012 Sales in the cells for every year other than 2012. This doesn't provide any value for our analysis because we can't, for example, divide 2013's sales amount by 2012's sales amount. In order to create that type of growth calculation, 2013 Sales and 2012 Sales would need to be on the same row.

This is where a LOD expression is handy in creating a fixed comparison measure for 2012 sales.

There are three types of LOD expressions: EXCLUDE, INCLUDE, and FIXED. To create my benchmark for 2012 Sales, I am going to leverage the EXCLUDE expression, which basically means Tableau is going to ignore whichever dimension I put in the expression.

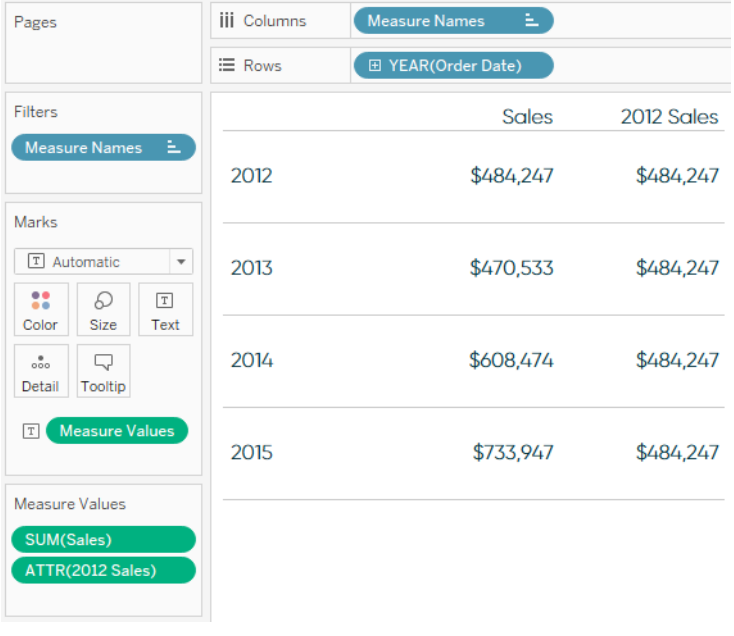
My calculated field using LOD expressions for 2012 Sales is:

```
{EXCLUDE [Order Date]: SUM(IF YEAR([Order Date]) = 2012 THEN Sales END)}
```

This calculation says, "Regardless of the Order Date dimension, which is currently being used to display one year per row, always show the sum of sales from 2012."

Note that because I am wanting to ignore the date dimension in this specific example, Order Date is the dimension following the EXCLUDE function. If you are creating a benchmark for something else such as a product category, manager, or test recipe, substitute Order Date with the appropriate dimension for your use case.

Now my table looks like this:

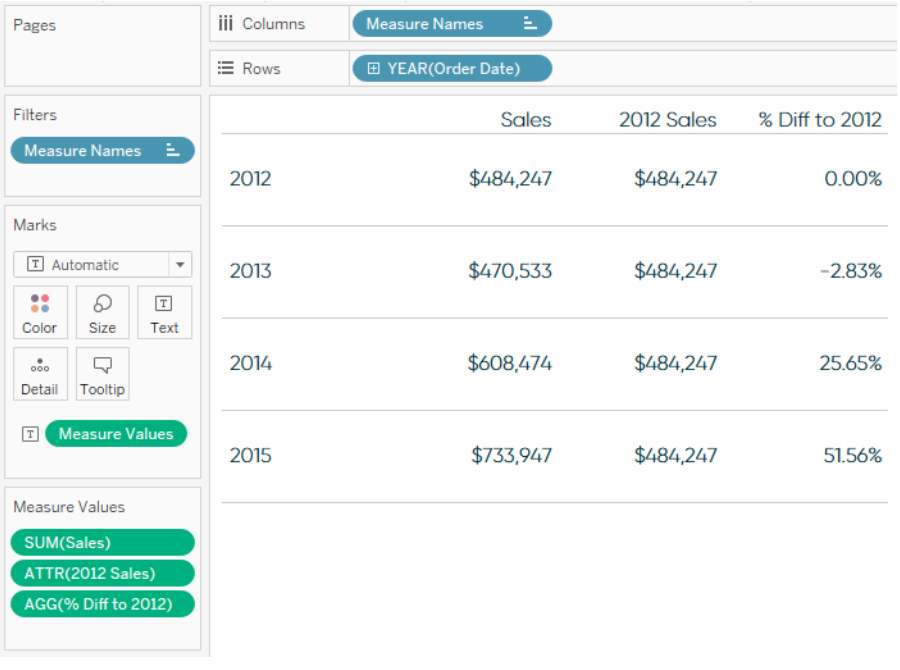


Now every year displays the 2012 Sales amount in the 2012 Sales column. Now that the sales for each respective year and the sales for 2012 are on the same row, they can be used for calculations, such as percent growth.

As just one example, I'll create a calculated field to show percent difference from 2012 Sales and place it on the view as a third column. The formula is:

```
SUM(Sales) / SUM(2012 Sales)-1
```

Here's the final view:



The percent difference from 2012 Sales is just one example, but now that this benchmark measure is available, it can be used in a variety of calculated fields. I have personally used it to calculate t-statistics in test results and to create more reliable 100-point index scores in Tableau (discussed in Chapter 61). You can even create a parameter to dynamically change the benchmark measure. For example, in my formula for 2012 Sales, I could replace "2012" with an integer parameter that allows you to toggle between any of the four years on the view. This would allow the end user to change the benchmark metric on the fly. I have found this approach to be a simple alternative to table calculations that provides a more consistent, user-friendly, and foolproof result.

CHAPTER 60
Designing Device-Specific Dashboards

One of the most common questions I receive while conducting Tableau training is "What dashboard dimensions should I use?"

As is the case with many aspects of analytics, the answer is largely dependent on the audience. Or more specifically, the answer in this case comes down to how the audience will be consuming the dashboard.

For example, will the dashboard be consumed by people at work on large desktop monitors? Or even larger conference room monitors?

Do you expect the end users to consume the dashboard on their laptops outside of working hours? What about their mobile devices outside of working hours?

As you know, there are many different devices that your dashboard can be consumed on, all with their own unique dimensions. So which one should you design for? Sure, you can use automatic sizing, but the results are unpredictable and not always ideal.

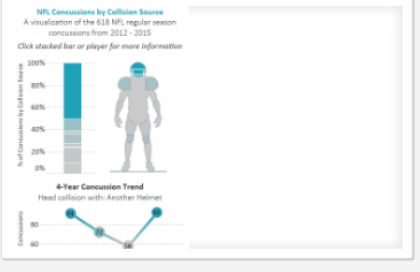
If I wasn't positive on this answer in the past, my default dashboard size of choice was 850 pixels wide by 1100 pixels tall. These dimensions are the same as an 8½- by 11-inch piece of paper, so at the very least, I knew the dashboard would look great if it was printed out or saved as a PDF and attached to an email.

Fortunately, I no longer have to pick just one dashboard size, thanks to a new feature in Tableau 10: *Device-Specific Dashboards*, or DSD. The DSD feature allows the Tableau dashboard author to lay the same dashboard out with different dimensions, then Tableau will automatically detect the screen size the dashboard is being consumed on and display the appropriate version.

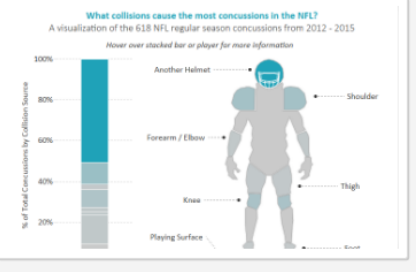
Before Tableau 10, it was possible to display a dashboard with different dimensions based on the end user's screen size, but it required a separate file to be built for each dashboard as well as technical understanding of CSS code. While not truly "respon-

sive," DSD makes it possible to deliver a much better user experience that is tailored to the end user's environment.

To illustrate how to design device-specific dashboards in Tableau 10, we will look at my dashboard, NFL Concussions by Collision Source. This dashboard was featured in the online version of *U.S. News & World Report*. Since *U.S. News & World Report* receives a large amount of traffic across several different devices, they asked me to create two different versions of the dashboard. Their web development team would then add some code within the article to display the most-appropriate version of the dashboard based on the viewer's screen size. For this reason, you can see two separate workbooks in my Tableau Public profile for the same analysis:



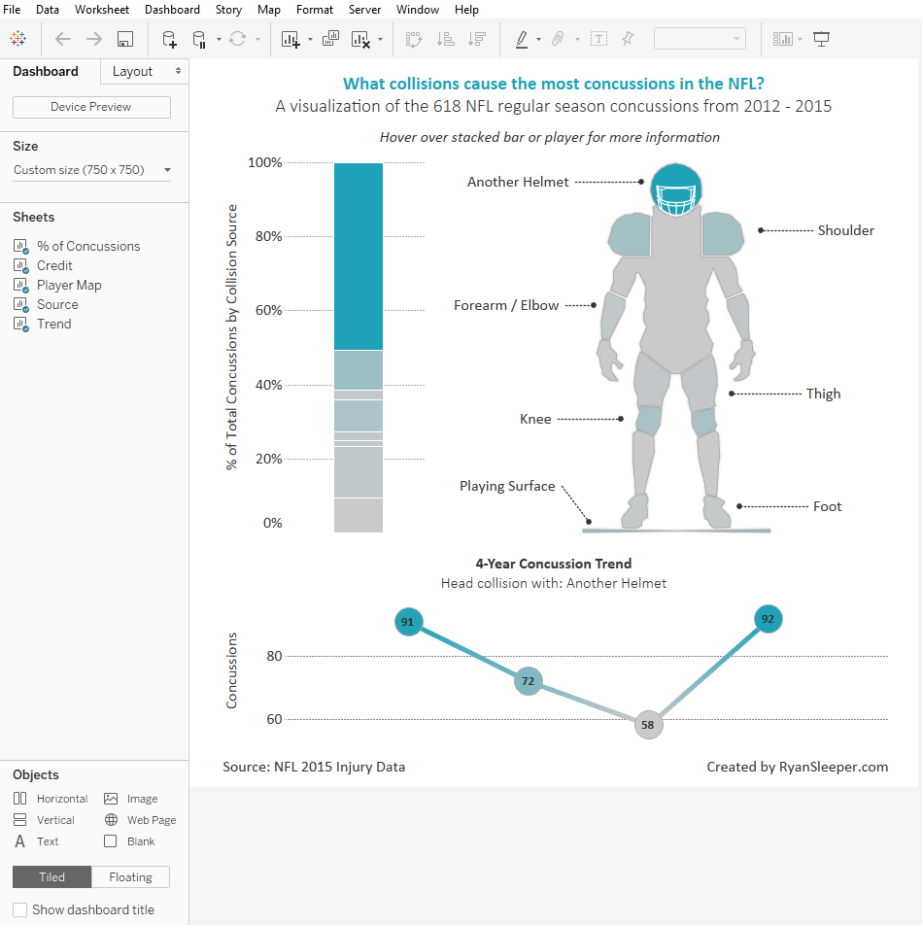
NFL Concussions by Collision Source - Mobile
2702 views
Originally published on: ryansleeper.com



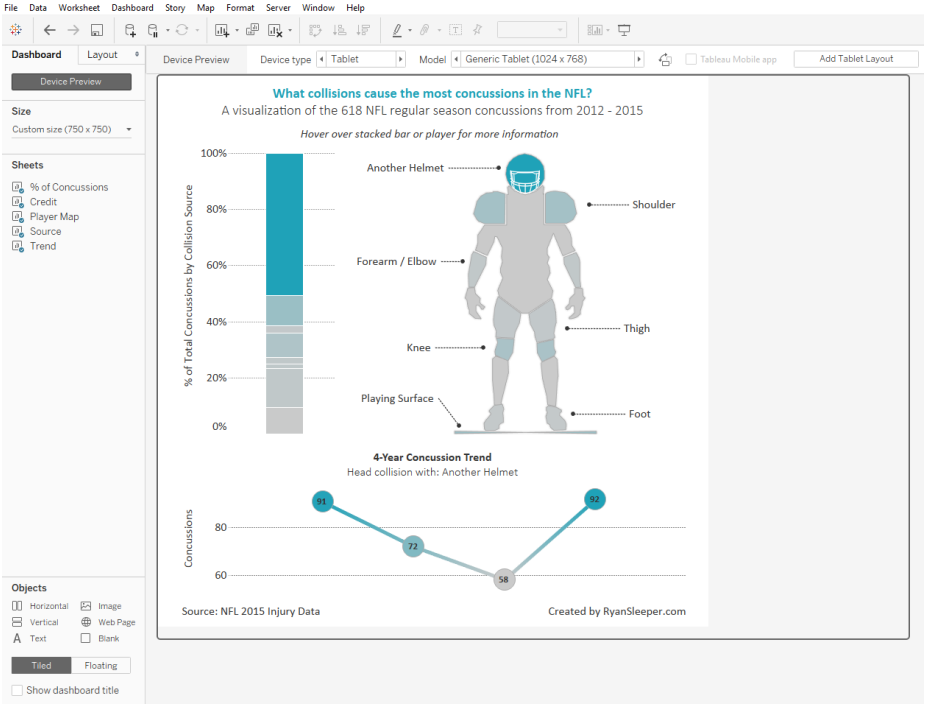
NFL Concussions by Collision Source
12736 views
Originally published on: ryansleeper.com

With the DSD feature in Tableau 10, I'm able to combine the different versions into one file and have the best one displayed automatically. I also don't need any technical web design knowledge to make it happen.

I will start this tutorial by downloading the original version of the NFL Concussions by Collision Source visualization from my Tableau Public profile page and opening it in Tableau 10:

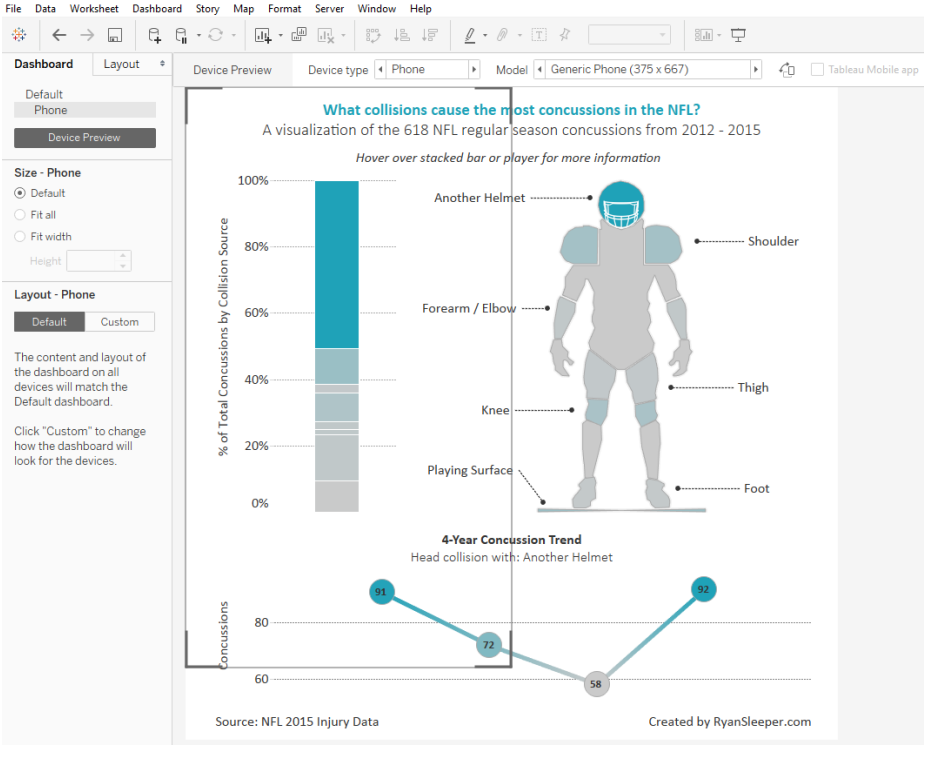


As you can see in the left dashboard design panel under Size, the dashboard is 750 pixels wide by 750 pixels high. What's new is that I now have the ability to click Device Preview see how this 750 by 750 dashboard will look on different devices. If I were to click the Device Preview button, Tableau draws a border around the dashboard to show you the dimensions of common devices. By default, the first preview is for a tablet in landscape (or horizontal) layout:

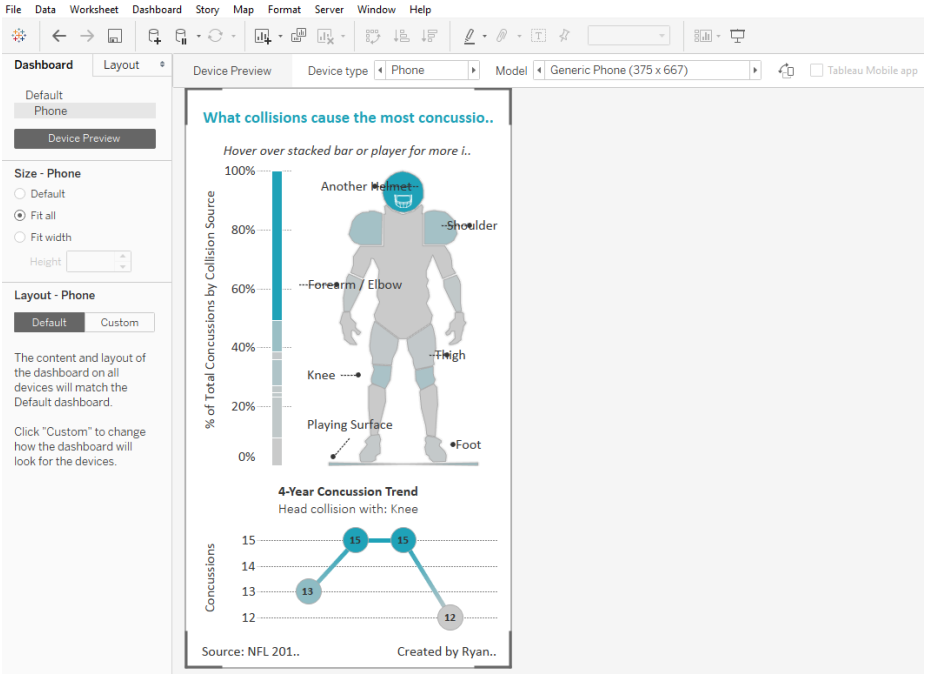


The Generic Tablet dimensions are 1024 wide by 768 high. I designed the visualization to be 750 by 750 so there is some blank space on the right and 18 pixels of blank space along the bottom.

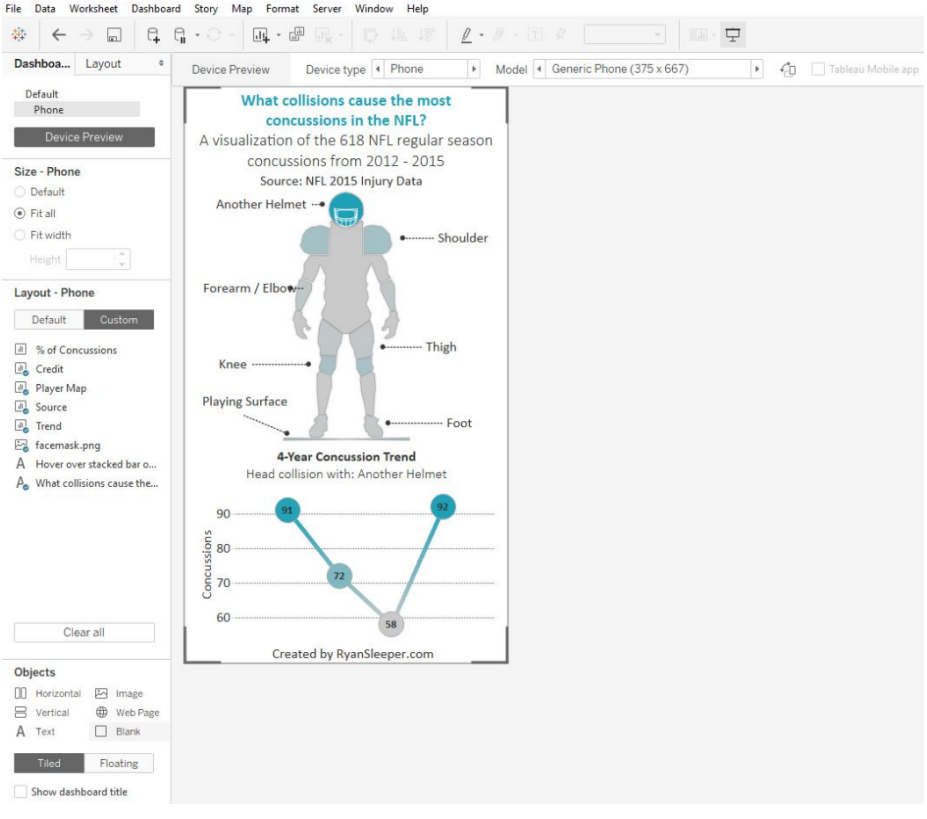
I can also flip through the device previews by choosing different device types and even models at the top. For this real-life example, I wanted to create one 750 by 750 dashboard that would be the original full-size version, but also one additional version suitable for a phone. To create a separate layout for phone, navigate to the Phone device type from within the device preview, choose a model, and click Add Phone Layout. I have chosen the Generic Phone dimensions:



As with the tablet preview before, you can now see how my full-size dashboard would fit when being consumed on a phone. Since I clicked Add Phone Layout, you also see this version show up in the upper-left corner. To get a head start on resizing the dashboard within these generic phone dimensions, choose "Fit all" from the options on the left. After making this selection, my dashboard looks like this:



You can see many different problems with this view. What's scary is this is close to how my dashboard would look on a phone if I chose automatic sizing! For this version, I will resize each dashboard component to make it look better when viewed on a phone. To show the full functionality of DSD, I will even delete an entire dashboard component so you can see how much you can customize device-specific dashboards. Here's how my phone-specific layout looks:



Notice that on the phone-sized version, I opted to delete the stacked bar chart in the upper-left corner of the dashboard. Even though I deleted it from the phone-sized version, this chart still exists on the original version. I can compare the two versions by toggling back and forth between Default and Phone in the upper-left corner of the dashboard authoring interface.

Now when I publish this, Tableau will identify the screen size for me and automatically display the most appropriate version!

CHAPTER 61

How to Make a Stoplight 100-Point Index

As an analyst, the stoplight 100-point index is a reporting mechanism that I cannot live without. If you are not familiar, the 100-point index is a simple way to know whether a measure is outperforming or underperforming a comparison measure, with the comparison measure typically being a goal or the performance during a prior period. An index score of 100 means that the measure in question matches the comparison measure; a score greater than 100 indicates the measure is outperforming the comparison; a score less than 100 indicates the measure is underperforming the comparison.

This chapter shares three ways for getting the most out of 100-point index scores in Tableau: how to set up a 100-point index, how to make a custom color palette to be used in a stoplight 100-point index highlight table, and a trick for how to handle reverse 100-point index scores (when outperforming the comparison means a negative impact on the business).

What Is a Stoplight Index?

A stoplight index enhances the traditional 100-point index score by encoding the scores by color for faster processing. This is essentially a highlight table with 100-point index scores for different measures. Outperform the comparison (i.e., a score >= 100) and the indicator is green; come close but fall just short (i.e., the score is between 90 and 99) and the indicator is yellow; underperform goal or prior period (i.e., a score < 90) and the indicator is red.

Here is an example showing a year-over-year index. For the purposes of illustration, this is the Sample – Superstore data with filters for year (keeping only 2014 and 2013) and Sub-Category (keeping only Chairs):

	2013	2014	Index
Discount	15.94%	17.88%	112
Profit	$6,228	$5,763	93
Profit Ratio	8.68%	6.87%	79
Quantity	528	614	116
Sales	$71,735	$83,919	117

Note that Discount, Quantity, and Sales were all up year over year, so they are colored green. Profit was down year over year, but still within 10%, so it is colored yellow. Profit ratio is down more than 10% year over year, so it is colored red.

Why Do I Have to Use the Fancy Approach You're About to Share?

If you're used to working in Tableau, your first instinct to get the desired color encoding shown and described here would be to make a calculated field, similar to this:

```
IF [Measure 100-Point Index] >= 100 THEN "Green"
ELSEIF [Measure 100-Point Index] < 100 AND [Measure 100-Point Index] >= 90
THEN "Yellow"
ELSE "Red"
END
```

You would then place this calculated field on the Color Marks Card to encode the values.

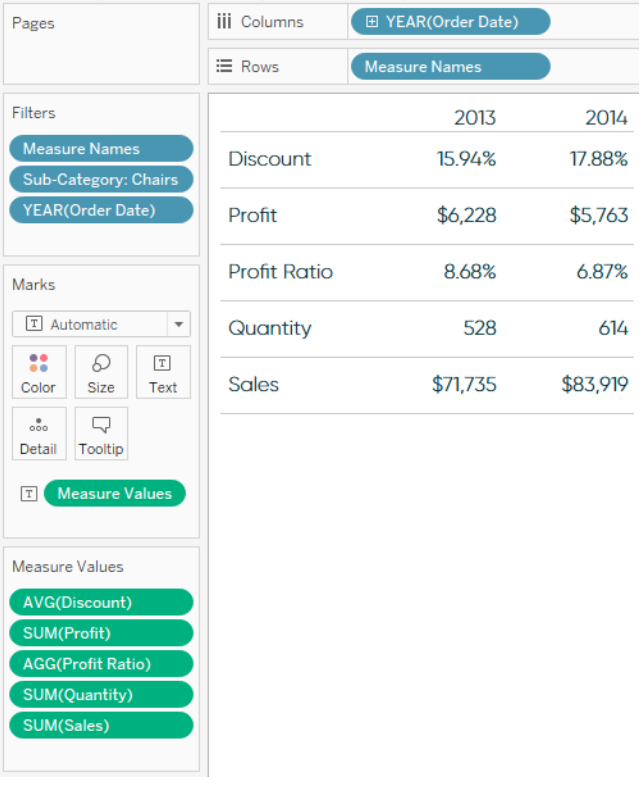
This approach would work for each individual measure, but as discussed in Chapter 57, not if you are wanting to show more than one KPI at a time. It's also a relatively manual approach because you would have to re-create these color instructions for every single measure. This approach would be much easier if you were able to use the Measure Values field in calculated fields, but that functionality is not currently available.

How to Set Up a 100-Point Index

There are always multiple ways to accomplish the same objective in Tableau, and each approach tends to have its own pros and cons. I am going to demonstrate how to set up 100-point index scores using table calculations, which have a major pro of being one of the fastest ways to set up calculated fields, but can be tricky to work with if you

lose track of the direction that the table calculation is being calculated. Note that this is an alternative to the approach covered in Chapter 59.

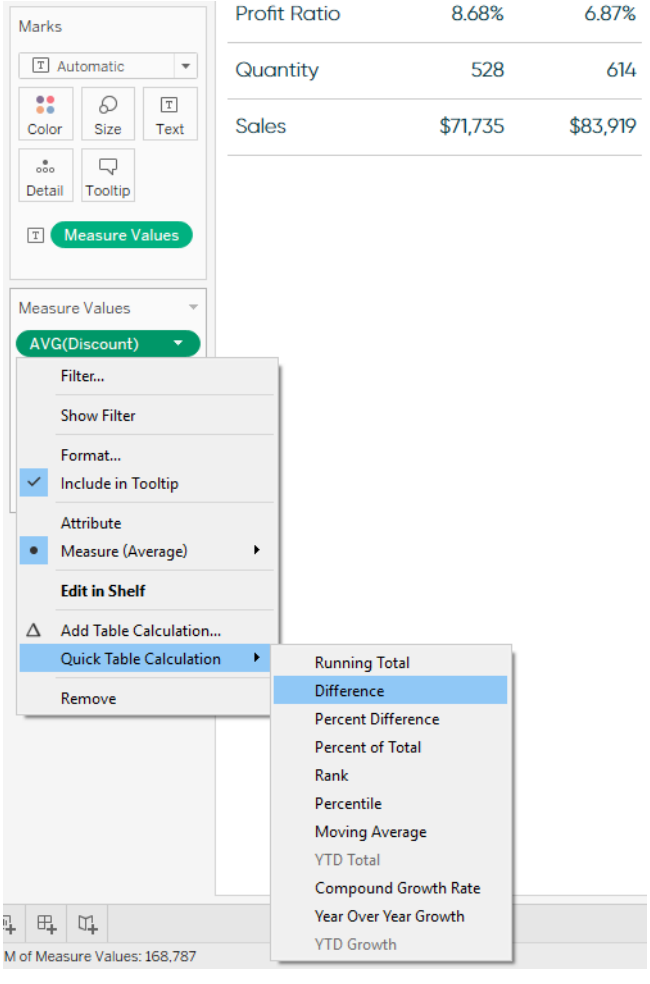
Here is how my table of year-over-year KPIs is set up in Tableau:



The formula for a 100-point year-over-year index is: (This Year's Performance / Last Year's Performance) * 100

Note you could use a different comparison point than last year's performance by just substituting "Last Year's Performance" with something else, such as a goal.

To create this calculated field in Tableau, you could manually create a calculated field and type out the logic, but I am going to lean on table calculations for a headstart. First, right-click a measure on the Measure Values Shelf and choose Quick Table Calculation. The Difference table calculation is the closest to my index formula so I am going to choose that:



After adding the table calculation for difference, the formula looks like this:

```
ZN(AVG([Discount])) - LOOKUP(ZN(AVG([Discount])), -1)
```

In English, this is just saying if there is a value for Discount in the right column, take that value and subtract the value from one column prior.

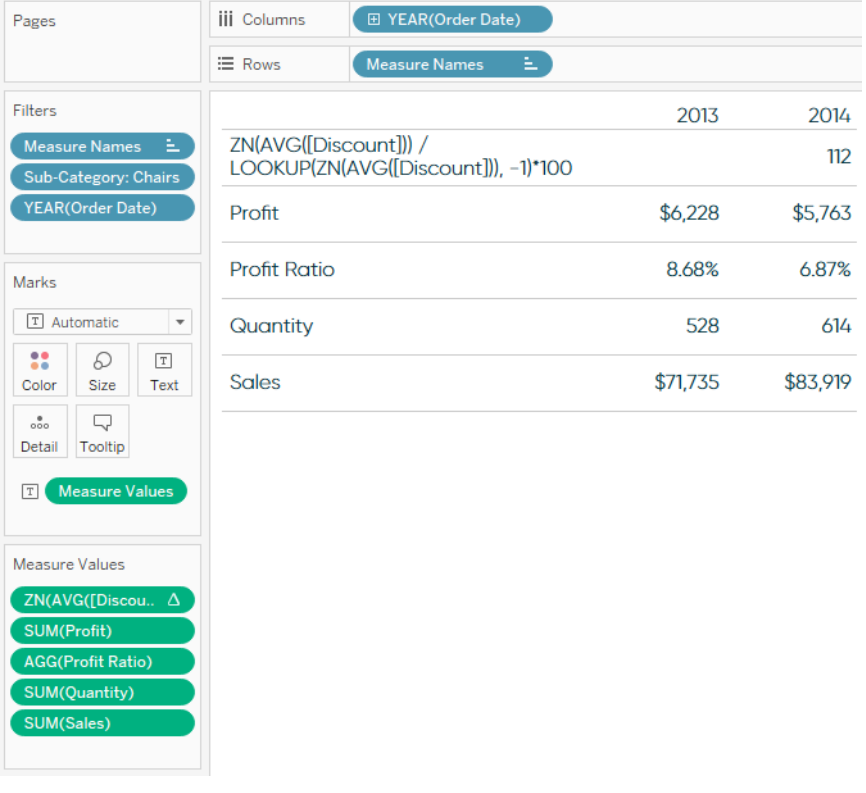
There are just two small changes that need to be made to this calculation to convert it into a 100-point index score. These changes can be made by simply double-clicking the measure that you just added the table calculation to on the Measure Values Shelf.

First, change the minus sign to a division sign ("/"). Then add "*100" to the end. The formula for a 100-point index score looking at the Discount measure is:

```
ZN(AVG([Discount])) / LOOKUP(ZN(AVG([Discount])), -1)*100
```

If your measure had any predefined number formatting, you will want to change the formatting to number custom with no decimal places. This can be accomplished by right-clicking the measure, choosing Format, then changing the number format. For example, my discount measure is normally displayed as a percentage, so I need to change the number format for my index score so that it is not displayed as a percentage or multiplied by 100 again.

At this point, my view looks like this:



If you want to give the calculation a better name and have it available for future use, drag the newly created measure from the Measure Values Shelf to the Measures area of the Data pane where the rest of your measures reside. You will then be able to give it a name and Tableau will save the calculation as a calculated field in your workbook.

Repeat this process for the rest of your measures on the view and you have a nice table of 100-point index scores.

Adding Color to a 100-Point Index Table

The view we have created at this point is providing a lot of value because the user can very quickly determine at what scale the KPIs are underperforming or outperforming the comparison point. However, the view can be processed even faster by converting the crosstab of index scores into a highlight table. For more on highlight tables, see Chapter 19.

I mentioned earlier that Measure Values cannot currently be used in calculated fields, which prevents us from creating one calculated field to provide logic on how Tableau should color each index score. Fortunately, Measure Values *can* be used on the Color Marks Card. Now that all of our scores are normalized on a 100-point scale, they can share the same color scale. If you drag the Measure Values field onto the Color Marks Card, the higher index scores will be colored darker green, and the lower index scores will be colored lighter green.

Now here's the cool part.

You can use a custom ordered-sequential color palette, combined with specific step-sizes and ranges, to get the index scores colored as you would like.

Creating custom color palettes is beyond the scope of this chapter, but if you would like to learn more about color, see Chapter 55.

In the meantime, follow these steps and you can use the custom color palette shown next:

1. On your computer, navigate to Documents → My Tableau Repository.
2. Right-click the file called Preferences and open it with Notepad or TextEdit (or your favorite text editor).
3. Paste this code over everything in the file (assuming you don't have other custom colors), then save the file:

```
<?xml version='1.0'?>
<workbook>
<preferences>
<color-palette name="Stoplight 100-Point Index (1 Scale)"
   type="ordered-sequential" >
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#edc64c</color>
```

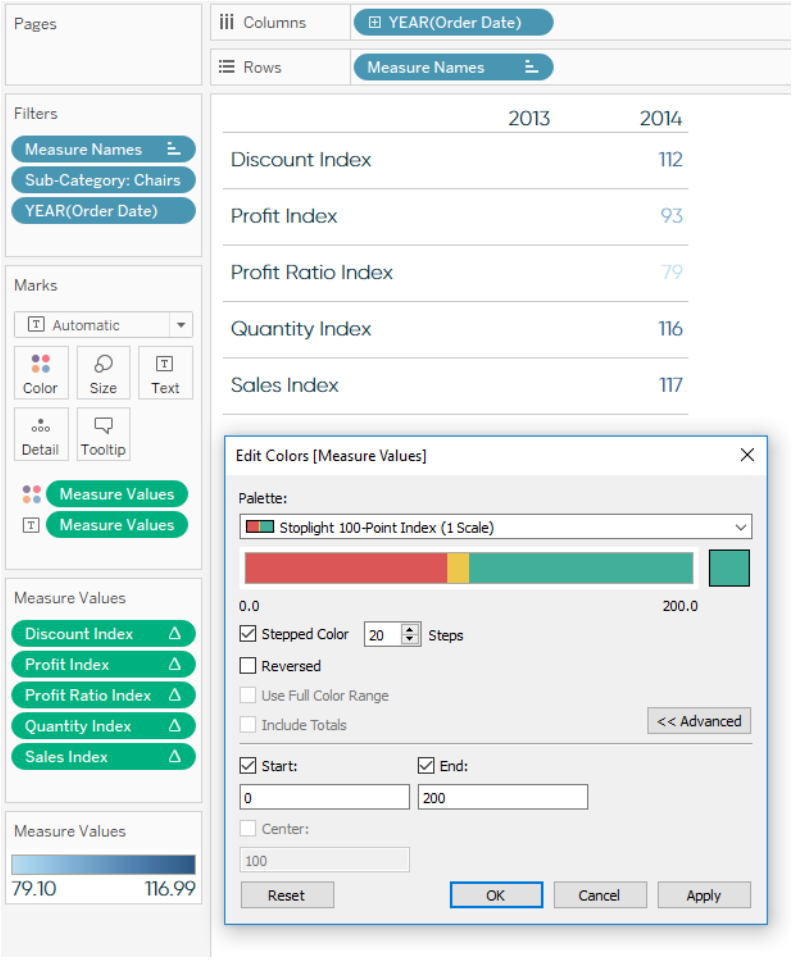
```
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
</color-palette>
</preferences>
</workbook>
```

In order for new custom colors to be available, you have to close and re-open Tableau.

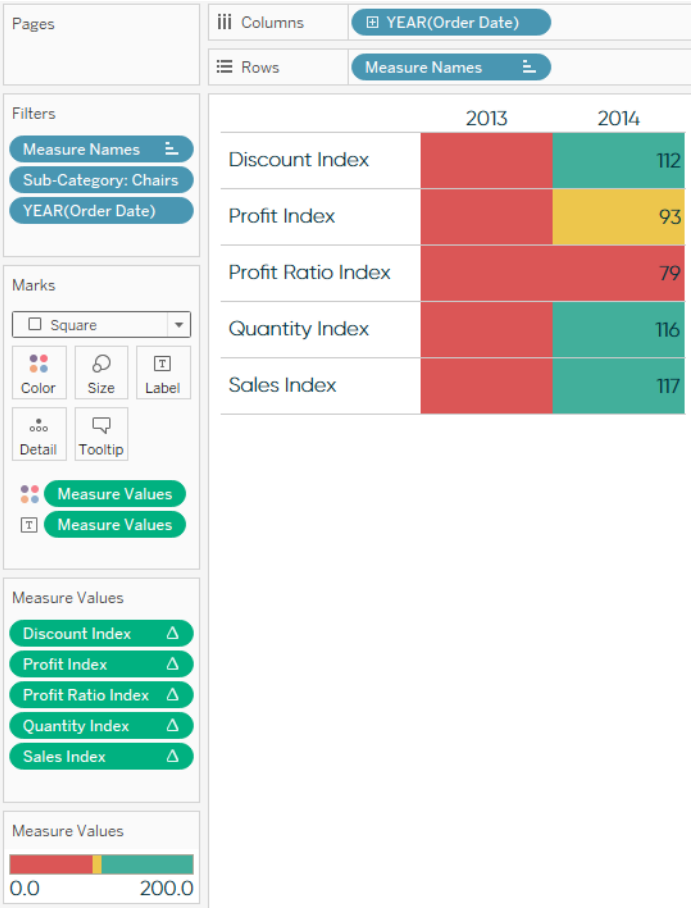
Now that you have this color palette available, you can choose this palette and apply some additional settings so that every score above 100 is colored green, every score between 90 and 99 is colored yellow, and every color below 90 is colored red. To change the color palette and edit the settings, click the color legend for Measure Values.

My color palette has 20 colors in it that go in sequential order. The first nine colors are red, which covers scores 0–90; the tenth slot is yellow, which covers scores between 90–100; the rest of the slots are green and cover scores 100 and up. With this in mind, the color settings are the result of some basic mathematical problem solving. I'll make the scale go from 0–200 and set the step size to 20 so that I know that each color on my sequential palette will represent 10 points or 5% of the spectrum.

Here's how my view looks with my color settings:



After saving the color settings, convert the view to a highlight table by changing the mark type from Automatic to Square. My final index score view looks like this:



From here, I could hide the 2013 column or even float this view next to the original table of numbers on a dashboard as pictured at the beginning of this chapter.

What If Outperforming the Comparison Is Bad?

Sometimes an index score greater than 100 is bad. For example, we probably don't want to increase the discounts we are handing out to customers each year, but the index table we just made shows discount as green because it increased 12 points year over year.

To alleviate this, I've come up with a little trick for creating stoplight 100-point index tables, even when an index greater than 100 can mean something positive or negative. First, I've extended my custom sequential color palette to include 40 steps. The first 20 steps cover scores that would be bad if the index is greater than 100; the next 20

steps cover our existing index scores, when a score above 100 is positive. Here is the palette if you want to add it to your custom colors as outlined before:

```
<color-palette name="Stoplight 100-Point Index (2 Scales)"
   type="ordered-sequential" >
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#edc64c</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#db5656</color>
<color>#edc64c</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
<color>#42af9b</color>
</color-palette>
```

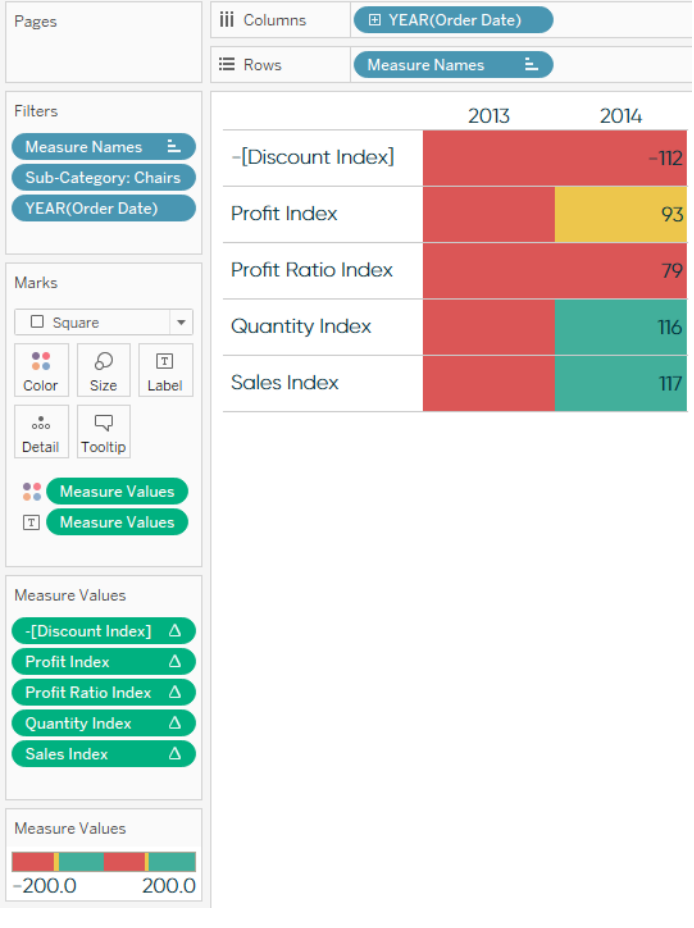
Once you have saved the new colors and closed/reopened Tableau, there are just two steps to get the desired effect:

1. Using the new custom color palette, change the steps to 40 and the range from −200 to 200.

2. For any measure that you don't want to outperform the comparison, multiply it by –1.

In the Sample - Superstore example that we are working with, I will multiply Discount by –1 by double-clicking the Discount Index field on the Measure Values Shelf and adding "-" to the beginning. Adding a negative sign to the beginning is the same thing as multiplying the calculation by –1.

My view now looks like this:



Now the index score for discount will be green only if it's less than 100. If it's within 10% higher, it will be colored yellow, and if it's more than 10% higher it will be colored red.

You'll see that a negative sign was added to the score. You can edit the format of negative numbers, but cannot get rid of the extra distinction. I personally like to have a visual indicator that some metrics are on a different scale than the others. Here is how my final view looks with the KPI table and index highlight table on two scales:

	2013	2014	Index
Discount	15.94%	17.88%	(112)
Profit	$6,228	$5,763	93
Profit Ratio	8.68%	6.87%	79
Quantity	528	614	116
Sales	$71,735	$83,919	117

Finally, I would be remiss if I didn't mention that the stoplight color palette is not the best choice for the colorblind. Not to mention that (in my humble opinion) red and green is one of the ugliest color pairings. I provided the stoplight example because those are the most common colors requested, and the stoplight concept, as well as the meaning of the stoplight's traditional colors, are (for better or worse) ingrained in the human psyche.

That being said, I encourage you to substitute the hex color values for green, yellow, and red with your own alternatives. Try blue for good, white for OK, and orange for bad. Or maybe there's a way to put the stoplight index into your own brand's colors. I assure you it will be fine if you try something other than red and green!

CHAPTER 62

The Case for One-Dimensional Unit Charts

This year I have found myself gravitating toward stacked bar charts as a way to show comparisons. The chart type is featured prominently in both my NBA Records by Player and 50 Years of AFC vs. NFC Matchups visualizations. I even used a stacked bar chart as the primary visualization to share how much progress I was making while writing *Practical Tableau*.

Something is different about these stacked bars though. I'm generally not a fan of stacked bars because if the stacks are different sizes, and if the stacks do not sit on the baseline, it can be challenging to compare and contrast the different pieces of the bars.

In all three cases I just mentioned, the stacks have equal units. This removes the challenges I have with traditional stacked bars, so I do not think the name gives these charts justice. So that got me wondering if these should be called unit charts.

If you're not familiar with unit charts, here is a well-done example from one of my data viz heroes, Andy Cotgreave. My issue with the multiple columns and multiple rows approach to most unit charts—especially when they are showing a part to whole relationship—is that these are pie charts in disguise. Very sneaky. Andy always does an amazing job of eliciting dialogue around data visualization in the community so I assume he had a reason behind his chart type selection. Even if he didn't, I'm not mad at him; we all have guilty pleasures. I once made a donut chart… and I enjoyed it.

Pie charts are different though. Pie charts are terrible. I could not have one of my new favorite chart types associated with a pie chart. So I kept searching.

"What are these charts called?!" I screamed (or Googled, actually).

The best I could come up with was a paper authored by another one of my Data Viz Heroes, Stephen Few.“” In classic Few fashion, the paper is unapologetically called, “Unit Charts Are For Kids”.

Stephen Few has a reputation of being critical, demeaning, and perhaps unkind. I had the privilege of having dinner with him in 2016, and found him to be exceptionally kind and humble. He introduces himself as Stephen, asks questions about you, and intently listens. Even in the cases where the first sentence in this paragraph is true, I have tremendous respect for his passion about the data visualization discipline and his tireless efforts to educate on the topic.

He even opens up the comments on his own popular site to take on debates from anyone. And I do mean anyone. The person that made this chart could go onto Few’s site and tell him why he doesn’t know anything about data visualization. I find that admirable.

In the paper, Few walks through why you shouldn’t use unit charts. But what really stood out to me is the image of the chart type I was searching for… and a name!

One-Dimensional Unit Charts

Even better, Few *almost* gives this visualization type praise:

> *The simplest form of a unit chart displays a single row or column of units, rather than a matrix of both as we saw in the previous example. As you can see, a one-dimensional unit chart is simpler to read than a two-dimensional version.*

However, he goes on to say:

> *Given improved ease of use, are one-dimensional unit charts worthwhile? We can read them much as we read bar graphs, with one minor difference—the segmentation of values into units inclines us to slow down and count, as opposed to the simpler, faster task of comparing their overall heights and then decoding their values in relation to a quantitative scale, which is missing. Not a big problem, some might argue, but significant enough to discourage their use when better means are available.*

As promised in the title of this chapter, here’s my case for one-dimensional unit charts.

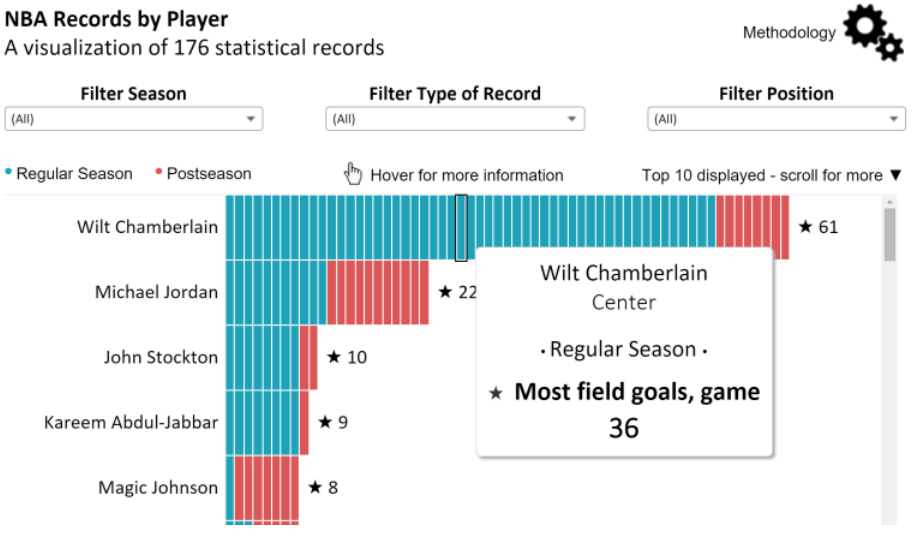
By the way, if I were to make a unit chart about the portion of this chapter so far showing how many seconds you’ve spent reading while not learning the case for one-dimensional unit charts versus the seconds you’ve spent learning the case for one-dimensional unit charts, it would look something like this:

So let's get to it.

I agree with Few that at the highest level, bar charts communicate data faster than one-dimensional unit charts. He makes a great point that this chart type inclines people to slow down and count the individual units.

But what if the units each mean something unique? The values are equal, but each unit has its own story to tell. That is the case with all of the examples that I've mentioned from my portfolio.

Here is the NBA Records by Player viz:



In the interactive version, if you hover over each unit, you are presented with additional information such as which record was achieved, what type of record was achieved, and whether or not the record is shared with any other player. In tools like Tableau, you can take this a step further by linking to additional context. This is the case with my *Practical Tableau* progress to completion unit chart:

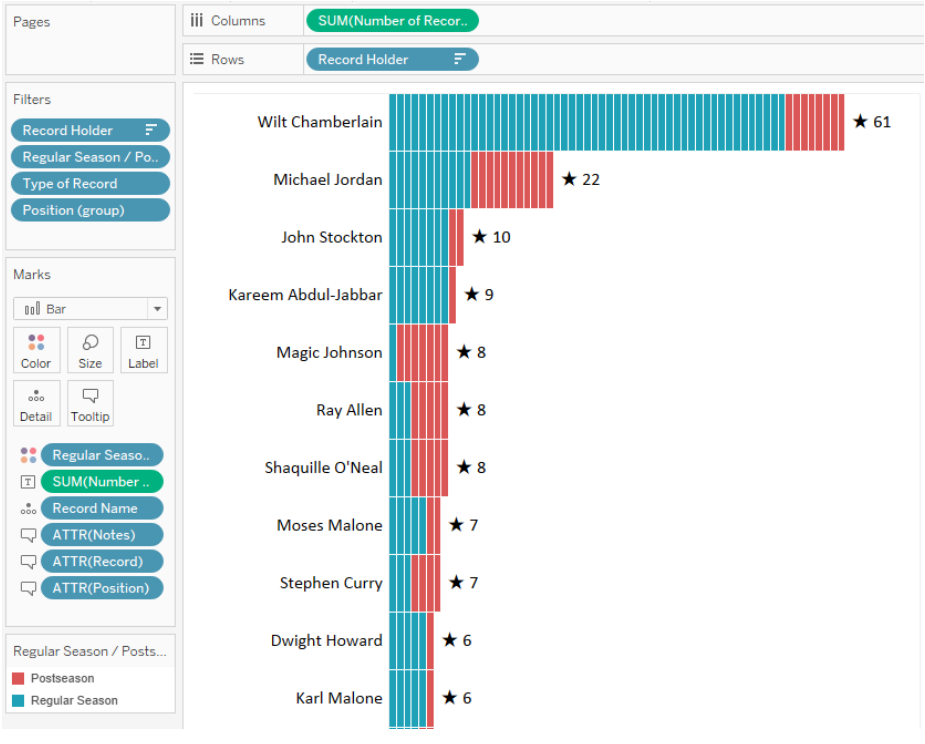
This moves the bar chart from a high-level *descriptive* visualization, to a *prescriptive* visualization, which helps add context to why something is the way that it is. Descriptive bar charts certainly have their place with certain audiences, but generally a prescriptive view is always going to be more valuable.

Not only that, in my personal opinion, the units help add context to the scale, even when the final view will be a flat, descriptive chart.

How to Make One-Dimensional Unit Charts in Tableau

One-dimensional unit charts are easy to make in Tableau as long as you remember one thing, and that is that each unit needs some type of unique identifier in the data. In the case of the NBA Records by Player viz, the unique ID was NBA record.

From here, you build a bar chart just like you always do in Tableau, then drag the unique ID to the Detail Marks Card. Here's a look at the NBA Records by Player viz under the hood:



If you don't have a unique ID and are purely going for the unit chart look (sorry Stephen Few!), you can simply add reference lines that will draw a line across the equal units you want to display. This is a manual approach, but has the advantage of allowing you to format the reference lines to your liking.

Finally, people always ask how to add totals to stacked bars (like you see here with the star and total number of NBA records per player). I considered doing a chapter to cover this topic by itself because it is a valuable capability, but it has been covered regularly and is just plain easy. All you have to do is add a reference line by right-clicking the axis of the measure in question, set the scope to cell, change the label to value, and set the formatting of the line to none.

CHAPTER 63

How to Highlight a Dimension

In my US Income by Age and Marital Status viz (which we'll look at momentarily), I wanted my end users to be able to choose their personal combination of age, marital status, and income—then have the chart not only filter down to their demographic information, but highlight their income in context of their other selections. This is another example of driving engagement by making the end user part of the story.

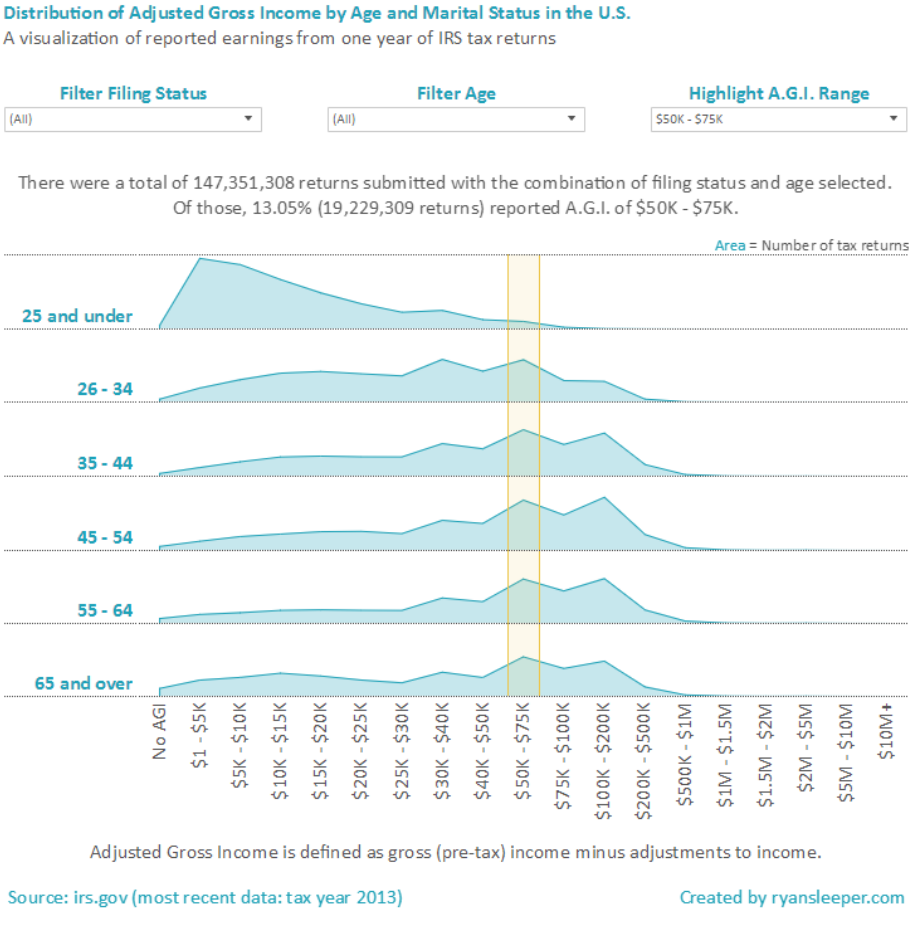
I thought this would be as easy as right-clicking the axis for income, a field that is a bin dimension in my viz (i.e., used to make a histogram), and choosing "add reference line." I would then make a reference band to get the nice yellow highlight effect you see pictured.

This is when I found out you can only add reference lines to measures or date dimensions.

This chapter shares a workaround for adding a "reference line" to a dimension.

How to Add a Reference Line to a Dimension

Let's start by taking a look at the visualization. In the interactive version, plugging in your own combination of age, marital status, and income filters and highlights the viz:



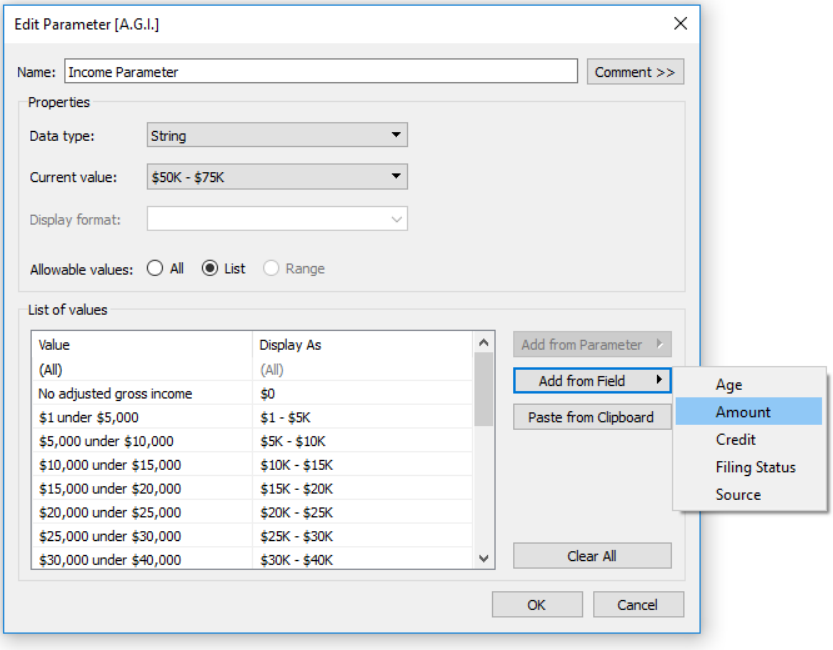
The first two filters, filing status and age, are simply filters that filter the entire viz to the end user's selection. The viz also works when the end user does not change these filters, in the case that he wants to see how his income compares across the entire population of tax returns.

The trick comes in with the income filter, which is actually a parameter. This parameter, combined with a calculated field on a dual-axis combination chart, is what produces the highlight effect. Here's how it's done.

1. Create a parameter for the dimension you want to highlight.

   Create a parameter with a data type of String and allowable values of List. Parameters are very sensitive in that the values have to match the underlying data exactly, including casing, to work how you want. As a shortcut, you can add the

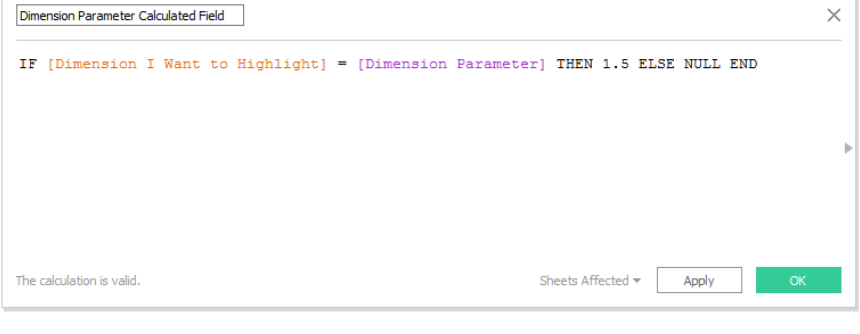
members of your dimension to the allowable values list by choosing "Add from Field" when creating your parameter. Here is how my income parameter looks:



2. Create a calculated field with your dimension parameter.

   The second step is to create a calculated field that will eventually be used as the measure on a dual-axis combination chart. The formula is:

```
IF [Dimension I Want to Highlight] = [Dimension Parameter] THEN 1.5 ELSE
NULL END
```

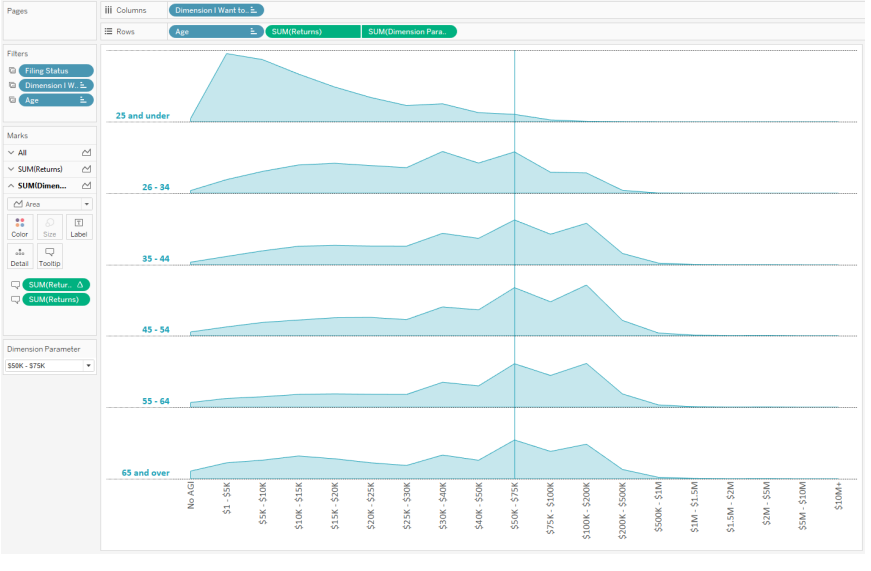


This calculated field is simply saying, "If the dimension member equals the current value selected in the parameter, then show the number 1.5; otherwise don't show anything." The 1.5 is kind of arbitrary because this approach will work as long as the axis for this new measure is fixed to be a smaller number than the number in your calculated field.

3. Create a dual-axis combination chart.

   Assuming you've created a chart that you want to add this highlight to, put the newly created calculated field on the opposite axis to create a dual-axis chart.

   In my case, the original chart was a histogram with a mark type of Area. When I put the new calculated field on the opposite axis, it inherits the same mark type as the original chart by default. Here is how my view looks at this point:



4. From here, there is just a little cleanup to make a nice highlight/reference band effect:

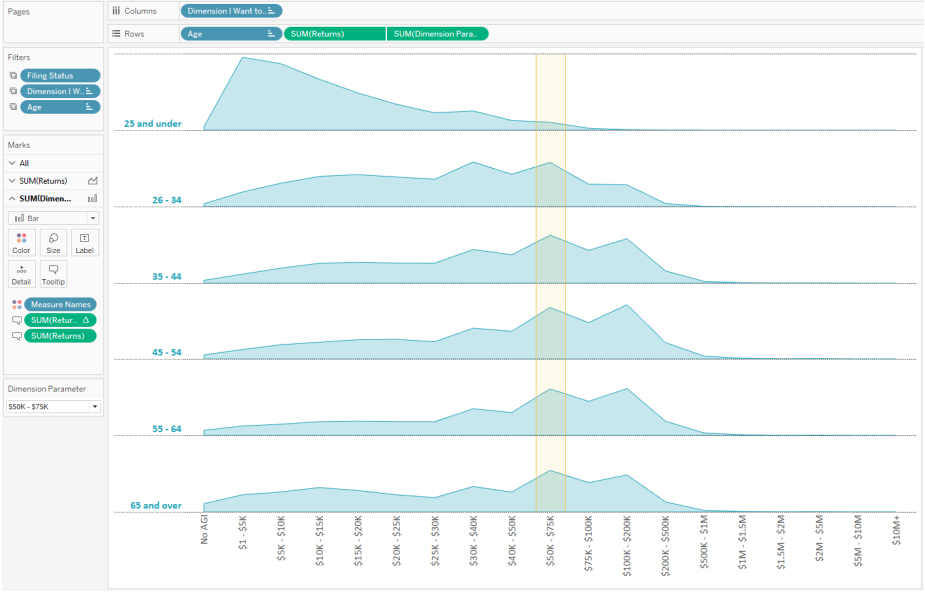
   a. Fix the axis of the right axis by right-clicking the axis and choosing Edit Axis. For the best results, fix the axis at a number smaller than the number in your newly created calculated field (mine is fixed at 1, which is less than 1.5).

   b. Change the mark type of the second axis to Bar, which will create the band look.

   c. Change the format of the bars to the size and color that you want. I recommend adding borders and transparency.

d. Hide the headers for your second axis by right-clicking the axis and unchecking Show Header.

My final product looks like this:



Now if the name of the dimension member matches the parameter selection, you will see a nice highlight of that choice on the x-axis—a reference band on a dimension in just three steps!

CHAPTER 64

Allow Users to Choose Measures and Dimensions

This book has often discussed how powerful parameters are in Tableau because outside of filters and dashboard actions, they're one of the only methods for putting control into the hands of your end users. In other chapters, I show you how to create and compare segments (Chapter 53), how to change the date aggregation on a line graph (Chapter 66), and how to create a what-if analysis in Tableau—all using the power of parameters (Chapter 48).

This chapter will walk you through how to leverage this same functionality to allow you and your dashboards' end users to decide which dimensions or measures are displayed on your views. This is a great approach for keeping analyses focused as well as saving real estate on your dashboard by displaying only one dimension and measure at a time.

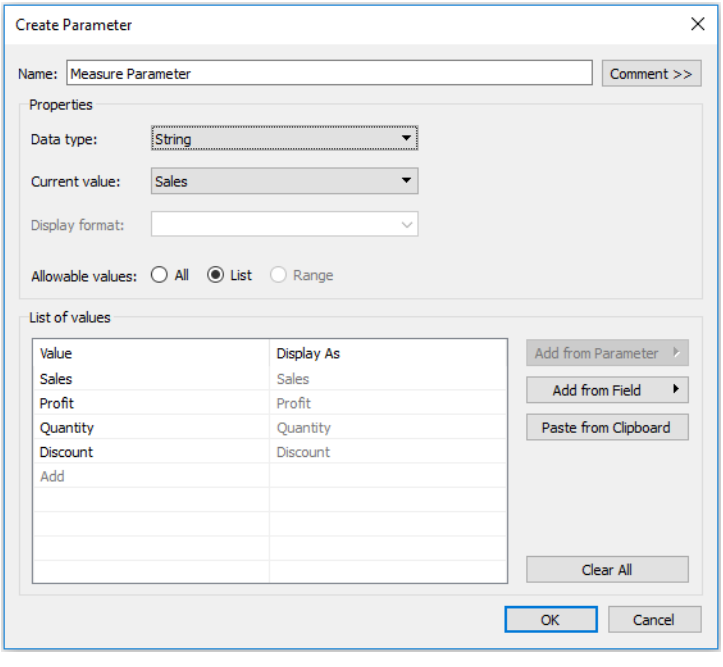
How to Use Parameters to Select a Measure in Tableau

In this example, let's assume we have a continuous line trend showing the measure that we care about by month. We want to set up the view so that we (and our end users) have the ability to change which trend is being displayed between Sales, Profit, Quantity, and Discount.

1. Create a parameter for your four measure choices.

   The most intuitive way to create a parameter is to right-click somewhere in the Parameters area of the Data pane in the lower-left corner of an individual sheet, and clicking Create Parameter. You are presented six data type options for your parameter. To allow users to select which measure is displayed on a view, we are going to be using a data type of *String*. You will also want to change the allowable

values to *List* so that you can specifically define which options the users will have. Once you choose List you will see a new menu that allows you to input the values for each of the choices. Type in the names of each measure, give the parameter a name, and your parameter should end up looking like this:



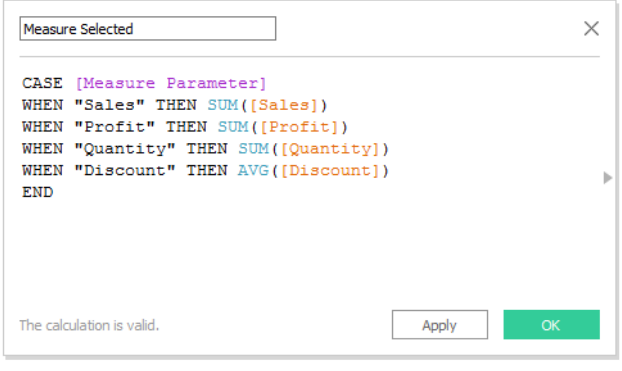
2. Create calculated measure.

   Parameters are dependent in that they don't do much on their own. In order to make them work, you also have to provide Tableau with instructions on what each of the parameter inputs should do. This can be accomplished through a calculated field. Since we are starting with a way to allow users to choose between one of four *measures*, we will create a calculated measure that will tell Tableau which measure to display based on the parameter value selected. In case you are not familiar with the syntax, the definitions will look like this in the calculated field:

```
CASE [Measure Parameter]
WHEN "Sales" THEN SUM([Sales])
WHEN "Profit" THEN SUM([Profit])
WHEN "Quantity" THEN SUM([Quantity])
WHEN "Discount" THEN AVG([Discount])
END
```

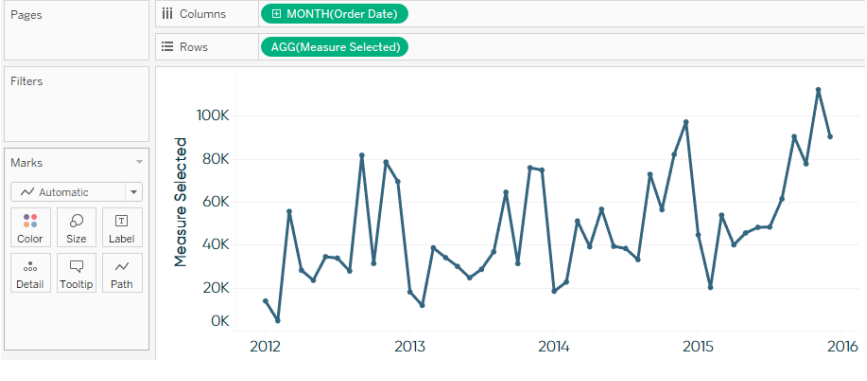
   Note that I gave every measure an aggregation. If I didn't assign the aggregations here, all four measures would share the same aggregation. It doesn't make sense

for us to sum up the discounts, so I assigned that measure an aggregation of average using AVG. My final calculated field looks like this:



3. Create the view.

   At this point, we are ready to create our continuous line graph with our newly created calculated measure. I know this is meant to be a continuous monthly trend over time, so I start by putting my date dimension on the Columns Shelf with an aggregation of continuous month. To create the line graph, I put my newly created "Measure Selected" measure onto the Rows Shelf. At this point, we see a monthly trend for whatever measure is the current value in the parameter that we created. By default, the current value will be the first value (Sales) in the parameter. The view currently looks like this:



4. Add parameter control to view.

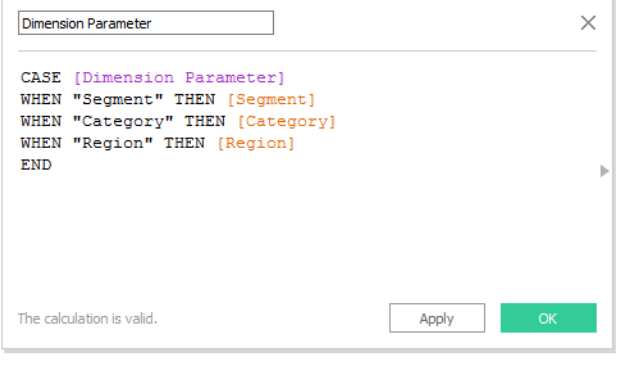
   There is just one last step to allow your end users to choose between the four options you have created. Right-click the parameter that was created on the Parameters area of the Data pane and select Show Parameter Control. You will see a new menu appear in the upper-right corner that will allow you and your

end users to choose between the four different measures. Changing the selection will change the measure being displayed on the line graph.

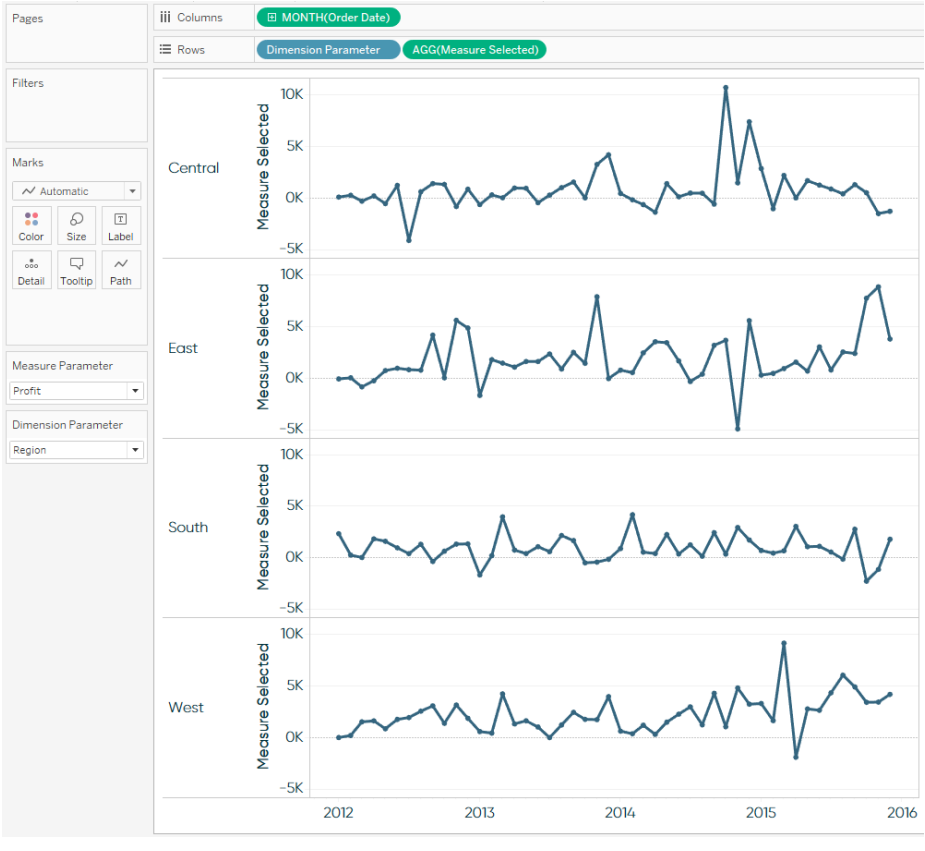
How to Use Parameters to Select a Dimension in Tableau

If you want to allow users to select a dimension instead of or in addition to a measure, follow the same four steps just outlined with your dimension options instead of your measure options. The only difference is that the dimensions in your calculated field will not need an aggregation. For example, let's say that in addition to allowing our users to choose between the measures of Sales, Profit, Quantity, and Discount, we also want them to be able to slice and dice those measures by the dimensions of Segment, Category, and Region.

We would follow the preceding steps and create a new string parameter with an entry for each of our dimensions. We would then create a calculated field that looks like this:



This newly created calculated field can now be placed on the Rows Shelf of our line graph to allow users to view the trends of not only four different measures, but also by three different dimensions. The final product looks like this:



With three different dimension and four different measure choices, we have provided users with *twelve* different view options—all of which they are able to control without any Tableau development experience needed!

CHAPTER 65

How to Dynamically Format Numbers

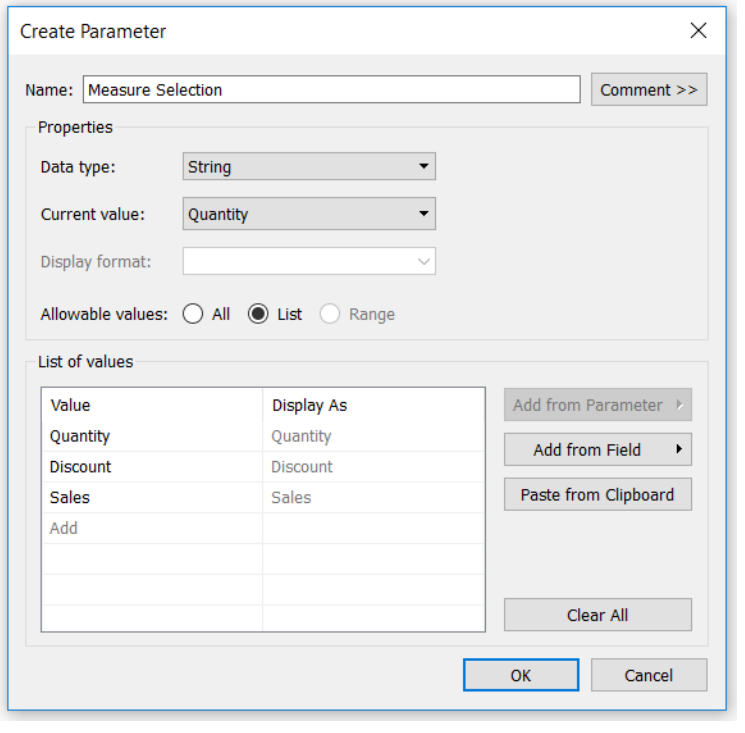
One of my favorite tricks in Tableau is to provide the ability for myself and my end users to choose which dimension or measure is displayed on a view (see Chapter 64). This user experience is provided by setting up a parameter with the options and then creating a calculated field that tells Tableau what to display when each option is selected. The parameter approach to dimension and measure display has two huge benefits: (a) it puts the power of the analysis into the hands of end users (which allows discovery), and (b) it saves valuable real estate on a dashboard by displaying only what is selected.

However, there is one big drawback to dynamically displaying measures: number formats. To produce the user experience described, a calculated measure must be created, and calculated measures can only have one default number format. This is problematic if you are allowing your end users to choose between measures that have varying number formats such as integer, percent, and/or currency. Having one shared number format across all of the measure options is usually not a deal breaker, but wouldn't it be great if you could pick a different number format for each number?

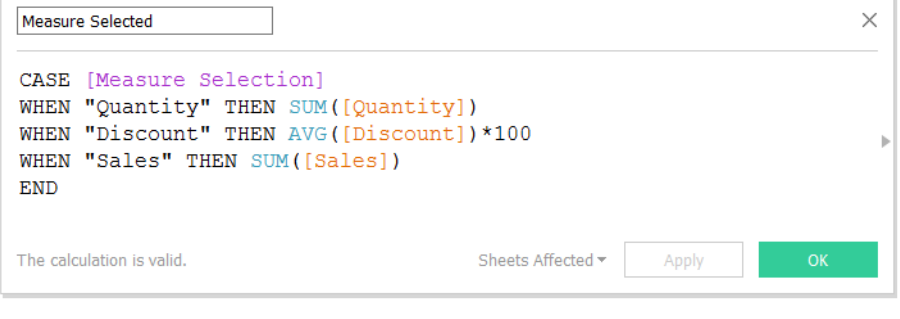
This chapter shares how to dynamically format measures in Tableau. We will be using the Sample – Superstore dataset to dynamically format the Quantity, Discount, and Sales measures into integer, percent, and currency formats, respectively.

How to Dynamically Format Numbers in Tableau

This is the most flexible approach possible to dynamically formatting numbers in Tableau. It allows you to customize both the prefix and suffix for any measure and works whether the values are negative or positive. To begin, set up a string parameter with the three measure options:



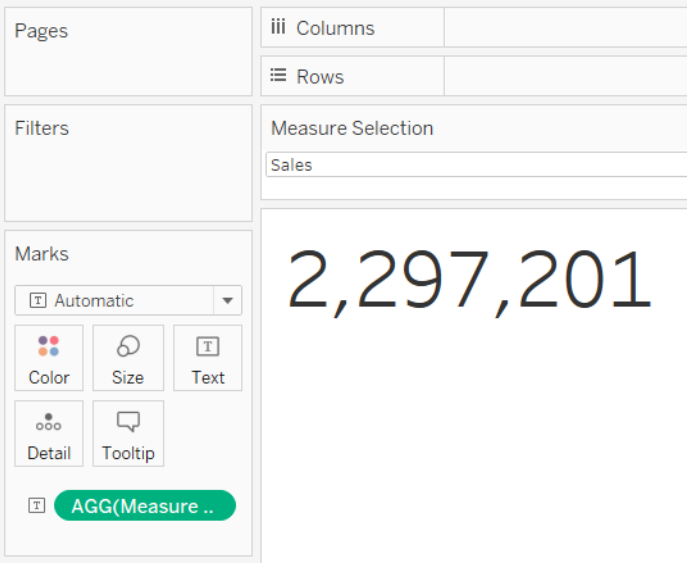
Next, you have to create a calculated field that tells Tableau how to handle each parameter selection:



Note that in this case, I handled the Discount measure differently from Quantity and Sales. I've forced the aggregation to be average and multiplied it by 100 so the decimal moves over two spots to the right (making it easier to read the percentages).

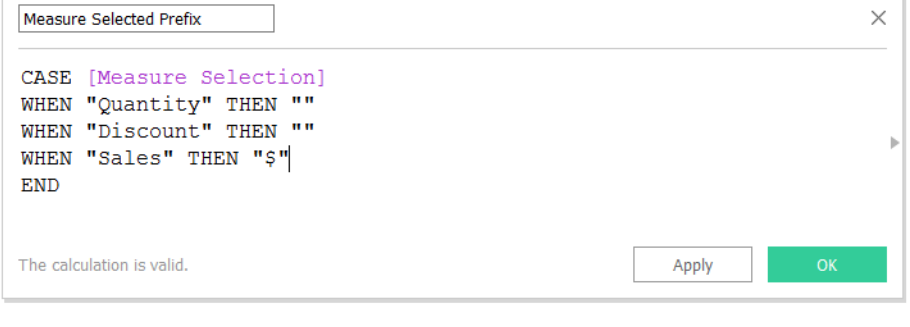
Now whenever this Measure Selected KPI is used in combination with the Measure Selection parameter control—which appears by just right-clicking it and choosing

Show Parameter Control—the end user will have the ability to choose between the three measures. This is a great user experience, but look what happens when the Sales measure is selected, for example:



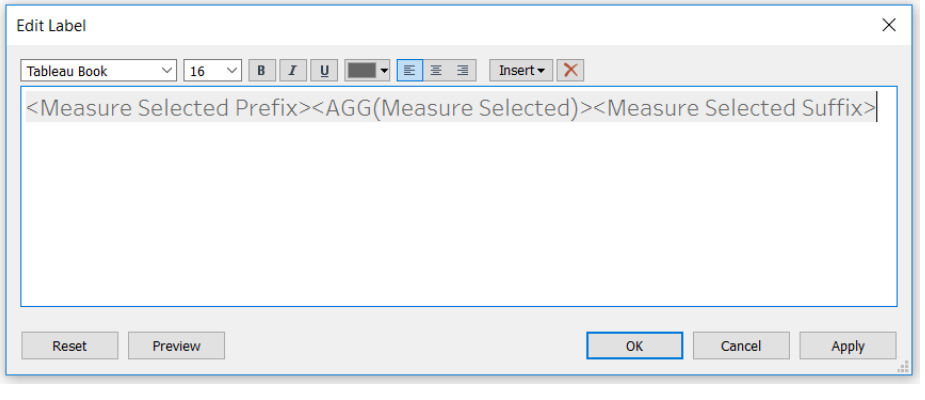
Sales should be in currency format, but there is no dollar sign. That's because all three measures in our Measure Selected calculated field have to share the same format and, by default, the format is set to Automatic. We could change the format to currency, but then a dollar sign would be shown for the Quantity and Discount measures. This same problem happens when the Discount measure is selected because it is the only measure of these three that should display a percent sign.

My solution involves setting up two additional calculated fields that are based on the parameter; one for the Measure Selected Prefix, and one for the Measure Selected Suffix. For these three measures, my prefix calculated field would look like this:

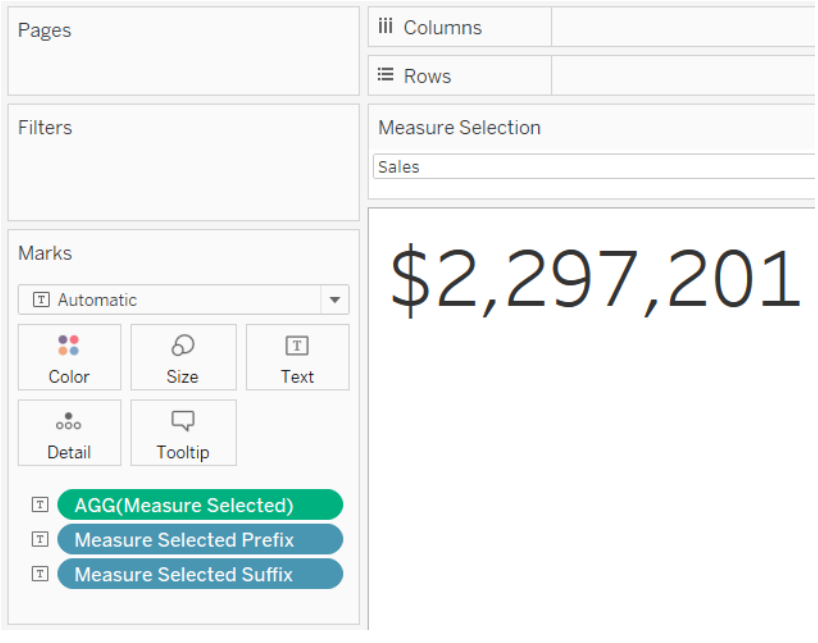


This formula is telling Tableau that if the Sales measure is selected, display a dollar sign; otherwise don't display anything. Since the Discount measure needs a percent sign added to the end of the number when it is chosen, I will also set up a calculated field for the number suffix:

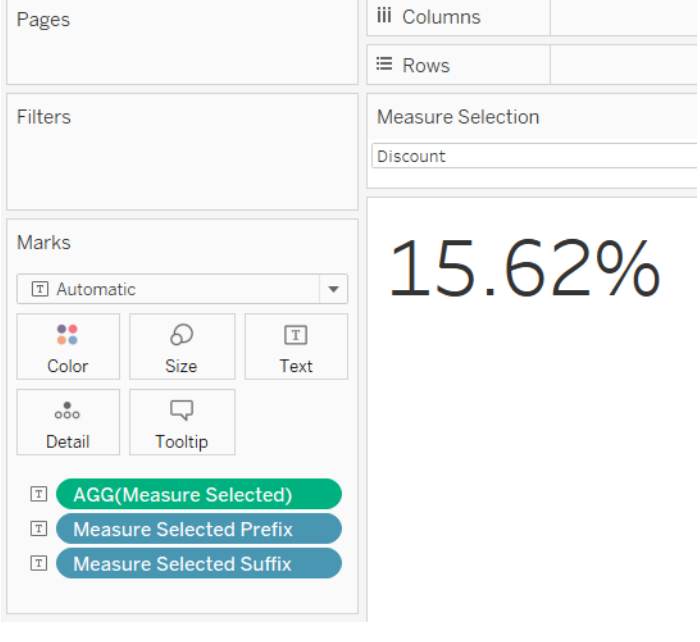
Now I will add both the prefix and suffix calculated fields to the Text Marks Card. After all of the fields needed are on the view, click into the Text Marks Card to edit the order that the fields are displayed. To get the measure selected to display properly with its respective format prefix and suffix, place the prefix calculated field in front of the measure selected field, and the suffix calculated field behind the measure selected field:



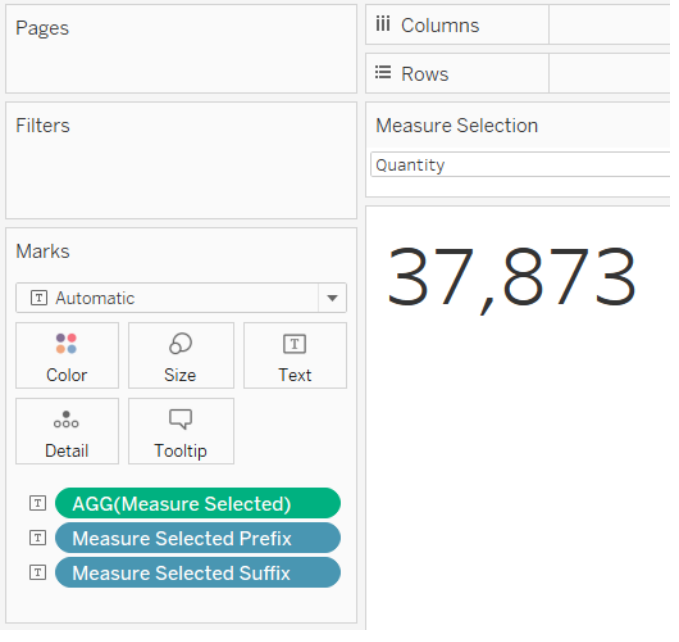
Now when the Sales measure is selected, a dollar sign is shown as the prefix:



When the Discount measure is selected, the dollar sign goes away and a percent sign is shown as the suffix:



When the Quantity measure is selected, no prefix or suffix is displayed:



We were using a basic view for the purpose of illustration, but this approach works in larger crosstabs, with mark labels, and tooltips.

CHAPTER 66

How to Change Date Aggregation Using Parameters

Problem: Tableau makes selecting and changing the aggregation of a date dimension very easy while you are building a view. However, unless an end user is viewing an individual sheet in Tableau Desktop, she can't easily pivot the date part between day, week, month, quarter, and/or year on her own.

Solution: Create a parameter that includes each date granularity option you want your end users to have access to (i.e., Day, Week, etc.), and create a calculated field that will act as your aggregation-changeable date.

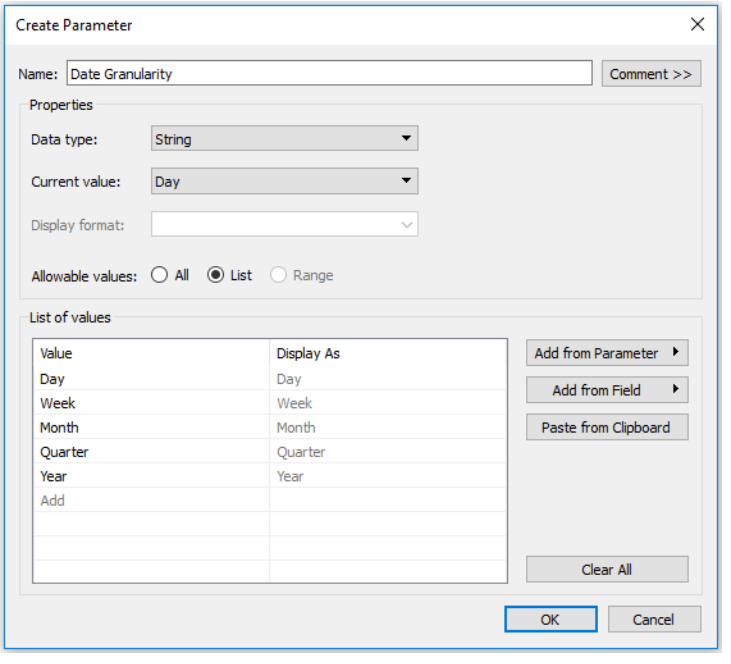
In many cases, it makes sense to change the granularity of a line graph over time. The Sample – Superstore dataset, for example, includes four years of daily data. If you are trying to view your sales over time and you set the date aggregation to year, you are provided a 10,000-foot view of your sales trend, but no seasonal insight. On the other hand, set the date granularity to continuous day, and while outliers stand out, it is nearly impossible to differentiate between individual days because you are looking at more than a thousand marks at the same time.

As you can see, viewing your sales over time at different levels of date granularity will tell very different stories. Why permanently choose the date aggregation of your view when you can allow your end users to choose for themselves?

How to Change Date Aggregation Using Parameters

In order to provide you and your end users with the ability to change the date part on the fly, follow these steps:

1. Create a string-based parameter with each level of date aggregation, as follows:

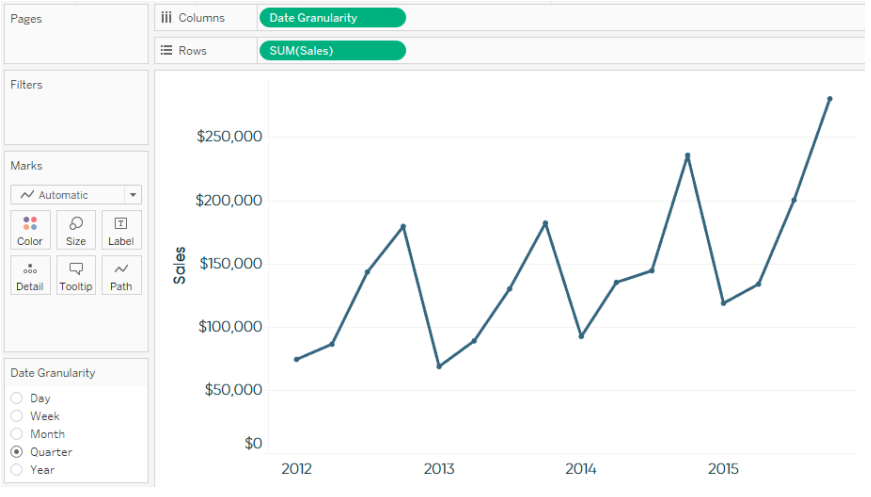


Note that the values have to be lowercase for this to work properly.

2. Create a calculated field, leveraging the DATETRUNC function to change the date aggregation to the appropriate level based on which parameter option is selected. Here is the logic:

```
DATETRUNC([Date Granularity parameter from step 1],[Order Date])
```

3. Instead of Order Date, use your newly created Date Granularity field. For best results, add the date field to the Columns Shelf by right-clicking and dragging; then choose the first option (Continuous).
4. Ensure you right-click the Date Granularity parameter and choose Show Parameter Control so your end users can choose their level of date aggregation. Your final product will look like this:


Pages
Columns
Date Granularity
Rows
SUM(Sales)
Filters
Marks
Automatic
Color
Size
Label
Detail
Tooltip
Path
Date Granularity
Day
Week
Month
Quarter
Year
Sales
$250,000
$200,000
$150,000
$100,000
$50,000
$0
2012
2013
2014
2015

CHAPTER 67

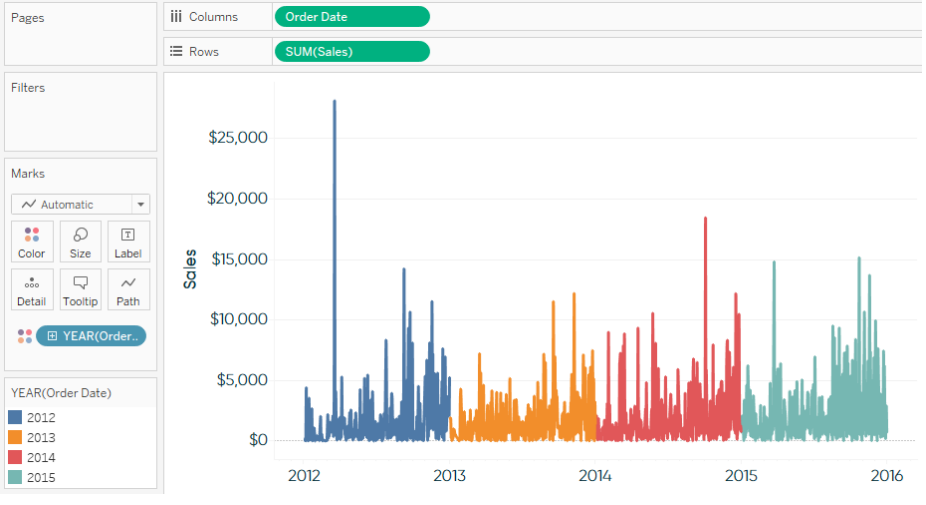
How to Equalize Year-Over-Year Dates

Comparing an exact date in the current year to the same date in prior years—at least on the *same* date axis—is tricky in Tableau. The challenge comes from the lack of a "Month/Day" date part option when using dates in Tableau. If I were to right-click and drag the Order Date dimension from the Sample - Superstore dataset in Tableau to the Columns Shelf, I am presented with these options:

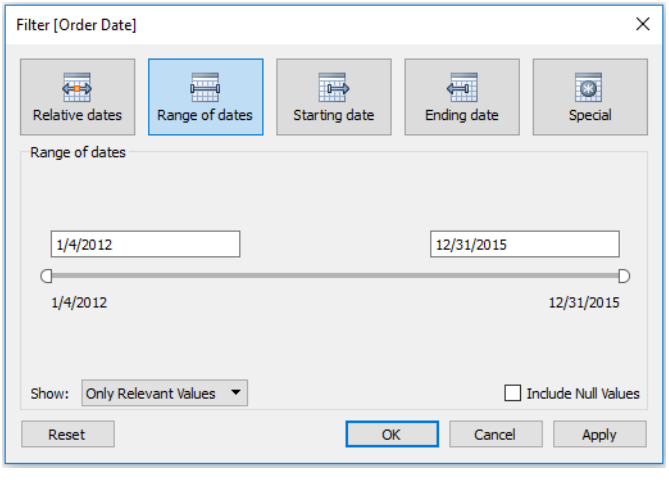
```
Order Date (Continuous)
Order Date (Discrete)
YEAR(Order Date)
QUARTER(Order Date)
MONTH(Order Date)
DAY(Order Date)
WEEK(Order Date)
WEEKDAY(Order Date)
MY(Order Date)
MDY(Order Date)
CNT(Order Date)
CNTD(Order Date)
MIN(Order Date)
MAX(Order Date)
YEAR(Order Date)
QUARTER(Order Date)
MONTH(Order Date)
WEEK(Order Date)
DAY(Order Date)
ATTR(Order Date)
```

The only choices that include the most-granular, day-level aggregation that I want to use, also include year. This prevents me from having the lines for each year in my analysis overlap for easy year-over-year comparison. It also makes it impossible to choose a date range such as 2/1/2018–3/30/2018, and still be able to see a line for all of the years in my data (this range would filter the data to just the year 2018). To illus-

trate, if I looked at the Sales measure in the Sample – Superstore data by the continuous `DAY(Order Date)`, then colored the lines by `YEAR(Order Date)`, the view would look like this:



And if I wanted to choose a smaller range such as 2/1–3/30 for each year:



No luck because I have to choose the month, day, *and* year.

How to Equalize Year-Over-Year Dates in Tableau

One way to equalize year-over-year dates in Tableau is create a calculated field that adds the appropriate number of years to prior years, so that all of the dates end up on the current year's axis. For example, if the year is 2018, you would equalize the year-over-year dates so that everything would be on an axis for the year 2018. If you were wanting to equalize three years of data, the formula would be:

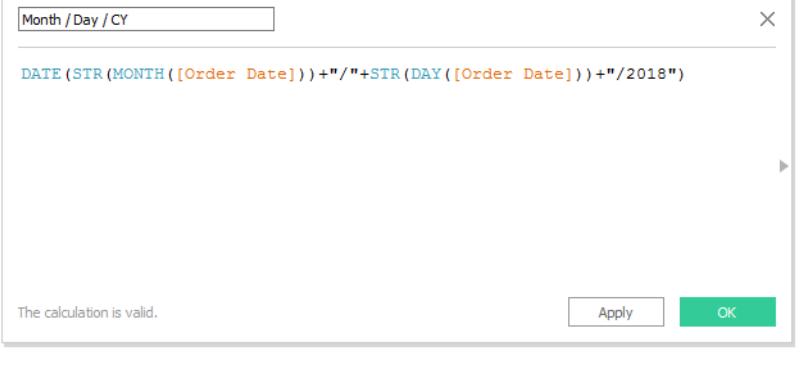
```
IF YEAR([Order Date]) = 2018 THEN [Order Date]
ELSEIF YEAR([Order Date]) = 2017 THEN [Order Date]+365
ELSEIF YEAR([Order Date]) = 2016 THEN [Order Date]+365+365
ELSE NULL
END
```

This formula is saying that if the Order Date is in the current year, then show the Order Date; if the Order Date is sometime last year, show the Order Date from that year, but add 365 days to it to get the dates on the current year's axis; if the Order Date is from two years ago, then add two years' worth of dates; if the Order Date isn't in the last three years, don't show anything.

This solution works, but is a bit manual to set up and dealing with leap years is a confusing topic.

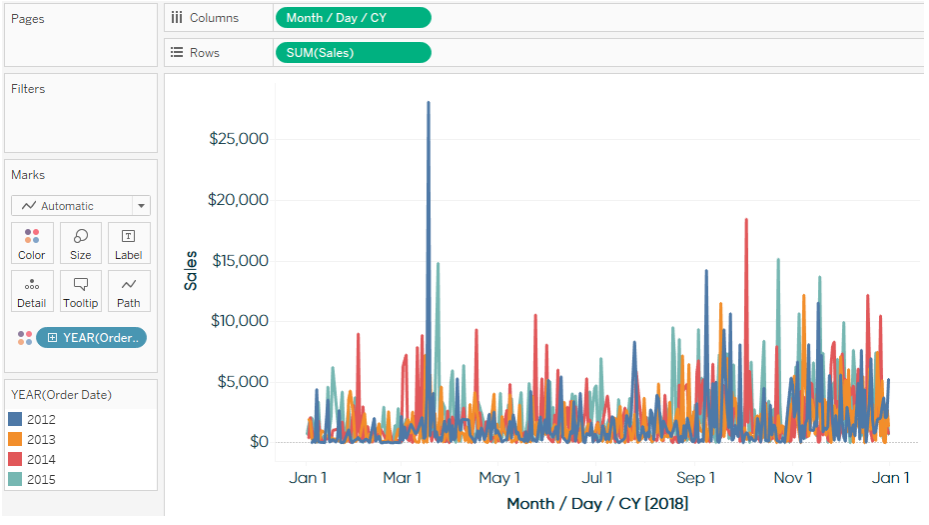
A second, and more effective approach, is to create a calculated field that provides the "Month/Day" date aggregation that was mentioned in the chapter introduction. The formula for this calculation is:

```
DATE(STR(MONTH([Order Date]))+"/"+STR(DAY([Order Date]))+"/2018")
```



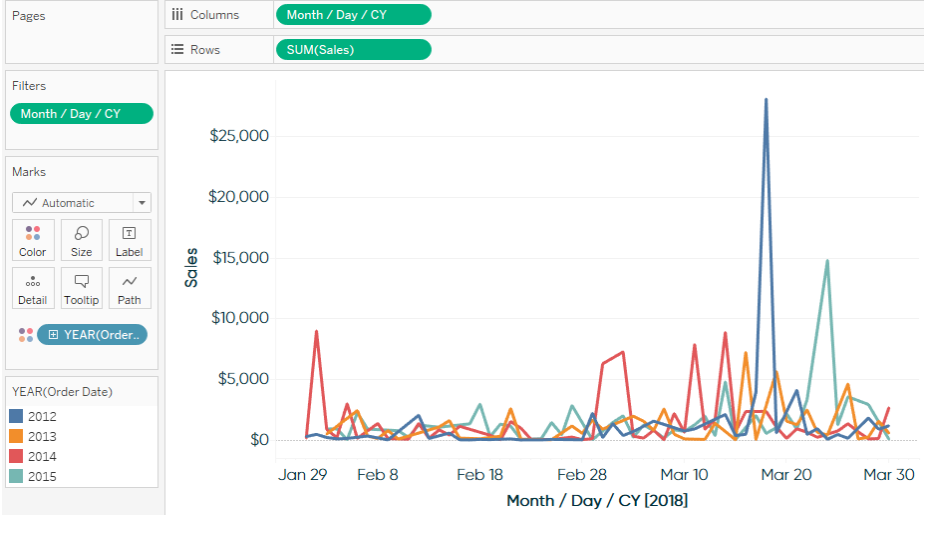
This calculation is telling Tableau to take the individual month and day from each Order Date, but make the year for all of them the current year.

Now if I look at Sales by this Month/Day/Current Year calculated date, I can look at all of the years in my data on the same date axis:



Note that if the year you are using for the date axis does not contain Leap Day (February 29th), but one or more of the lines in the graph do, that line that contains Leap Day will add the result from February 29th to March 1st together into one date (March 1st). Conversely, if the year you are using for the date axis does contain Leap Day, February 29th will be skipped for years that do not have Leap Day.

I can also use the newly created Month/Day/Current Year calculated field to filter the equalized date range. Changing the date range to 2/1/2018–3/30/2018 makes it much easier to compare individual, year-over-year dates, on the same date axis!



As a final note, now that you have the dates equalized on the current year axis at the most granular level, you can also change the aggregation to something less granular such as month or year and the dates will still line up.

CHAPTER 68

How to Filter Out Partial Time Periods

One of the most panic-inducing visuals in data visualization is a line graph that has a sudden, and steep, decline. When there is truly an issue with the business that needs to be addressed, it is important to share the illustrated insight with stakeholders that can take action.

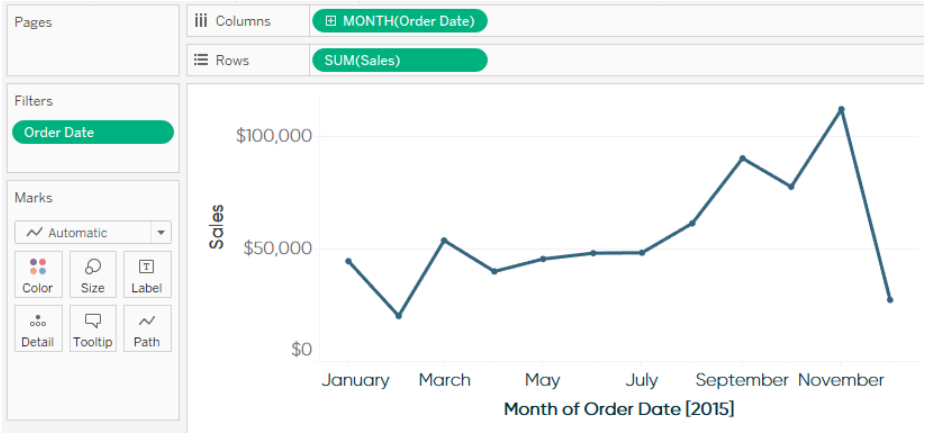
However, an apparent steep drop on a line graph can also be caused when time periods are not the same number of days, as is the case when a week, month, or year starts over (i.e., one day in the current week versus seven days in the prior week).

When the latter scenario happens, it can lead to misinterpreted findings and unnecessary panic; both of which you want to avoid in data visualization. Misinterpretations can lead to the wrong actions and unnecessary panic is a distraction from finding the real stories in the data.

This chapter shares a technique for filtering out partial time periods in an analysis.

How to Filter Out Partial Time Periods in Tableau

To illustrate when it helps to filter out partial time periods in Tableau, let's first take a look at a view that often shows up in real life. Using the Sample – Superstore dataset, we'll say we're looking at an annual report for the year 2015 with data through December 6th:



By the looks of this graph, after a brief downturn in February, the business recovered and has been steadily trending upward. That is, until December, when we see sales plummet again to near the low-point for the year. There is nothing on this graph that indicates that December's data is not yet complete. If we are looking at this graph on December 7th, we likely will understand why December is not complete, but there are other reasons that may cause the December view to not be ready to share. The data may take time to collect, process, and update, for example.

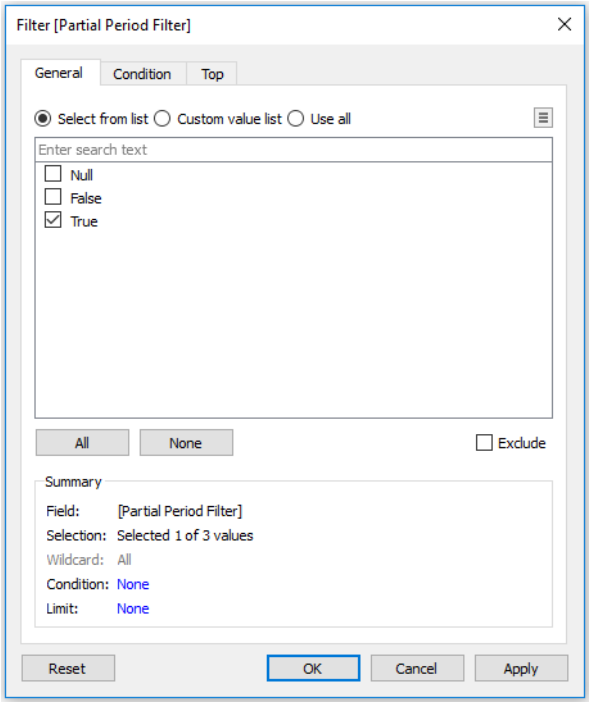
If you are in a similar situation and want to filter incomplete time periods out of the view, build a calculated field with this formula (use your own date dimension in place of Order Date):

```
DATETRUNC('month',[Order Date]) <> DATETRUNC('month',TODAY())
```

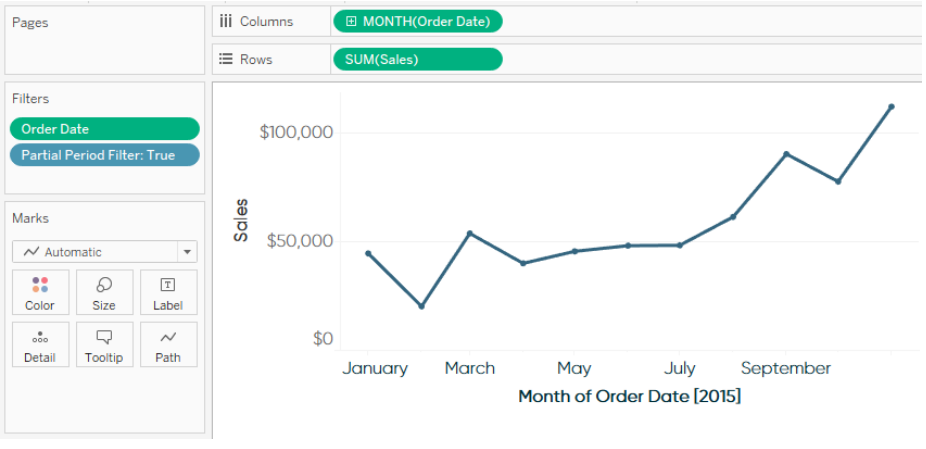
This calculation is telling Tableau whether or not the first day of the Order Date's time period matches the first day of today's time period. Today will always be part of an incomplete time period, so if we want to filter out partial time periods, we would keep only the Order Dates that don't match today's time period. This is a Boolean formula meaning it's either true or false; in this case, we want to keep only the "True" results.

To show how this works using the Sample – Superstore dataset, I have replaced `TODAY()` with the hardcoded date `2015-12-06`:

Once I have this calculated field, I will drag it to the Filters Shelf and choose True:



And after applying the filter—the partial time period disappears from the view, and holiday-happiness and visions of year-end bonuses are restored!



This technique can be combined with the trick for changing the date granularity of a line graph so that the partial period filter works whether you are looking at weeks, months, quarters, or years. If you are using a parameter to change the date granularity from day, to week, to quarter, etc., just replace the "month" granularity in the preceding formula with the parameter. The formula would look something like this:

```
DATETRUNC([Date Granularity Parameter],[Order Date]) <>
DATETRUNC([Date Granularity Parameter],TODAY())
```

As long as the values in the parameter are lowercase, the partial period filter will dynamically filter out the incomplete period as you choose different aggregations, whether it be the most-recent incomplete week, quarter, month, or year.

CHAPTER 69

How to Compare Two Date Ranges on One Axis

In Chapter 51, I shared that before I started using Tableau, I began my career in digital analytics using Google Analytics. One of the features I utilize most in my analysis work in Google Analytics is the ability to compare the performance during any date range to the performance during an equal date range immediately preceding it. For example, if I choose a date range of 10 days, I would like to see the performance of those 10 days as well as the 10 days that preceded my selected range. In Google Analytics, this is the first option if you choose a comparison date range, but it is tricky in Tableau.

I have seen a solution to this that uses an axis of number of days, but this solution leverages a date equalizer calculation to compare any date range to the equivalent prior period on the same *date* axis—a much friendlier user experience! There's an example of this experience in my Tableau Public visualization, Super Sample Superstore.

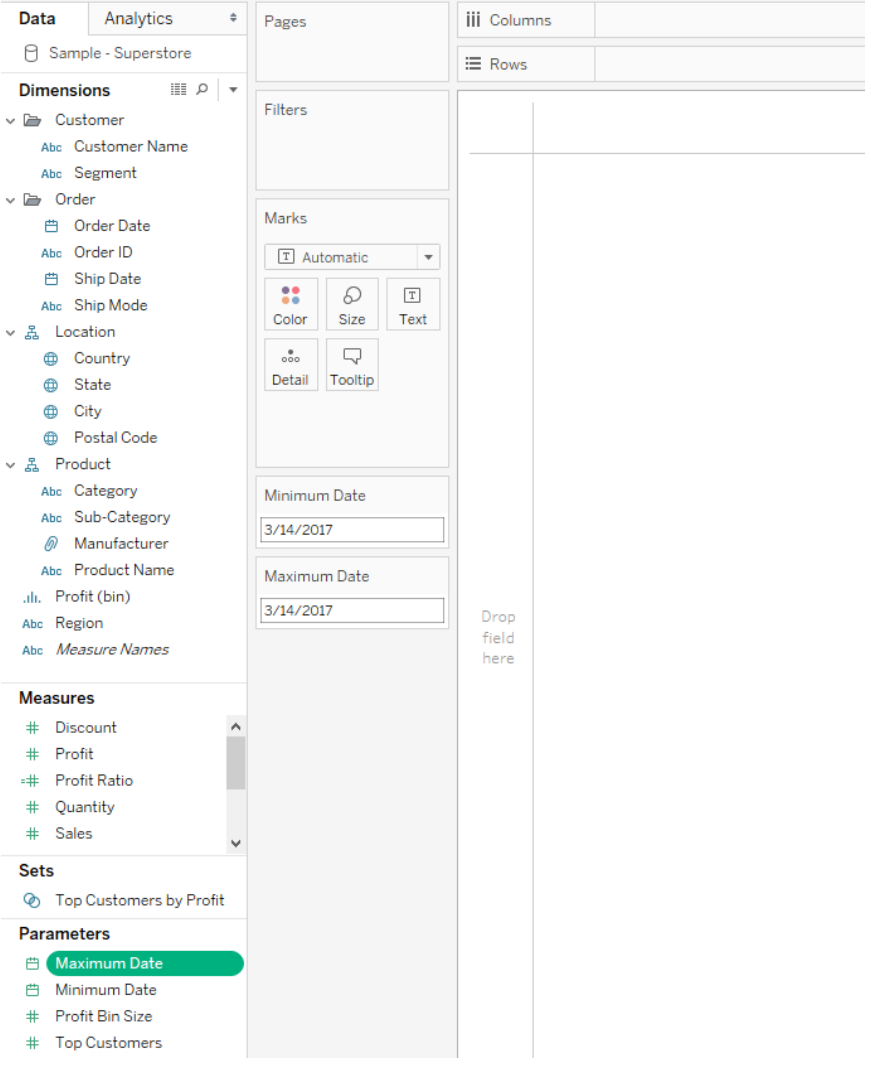
How to Compare Any Date Range to the Previous Date Range on the Same Axis in Tableau

If you ever need to equalize dates to compare two ranges on the same axis, follow these steps:

1. Create parameters for date range.

   Build two separate parameters with a data type of "Date"; one will be the minimum end of your range, and the other will be for the maximum end of your range. Once these are created, right-click each one and choose Show Parameter

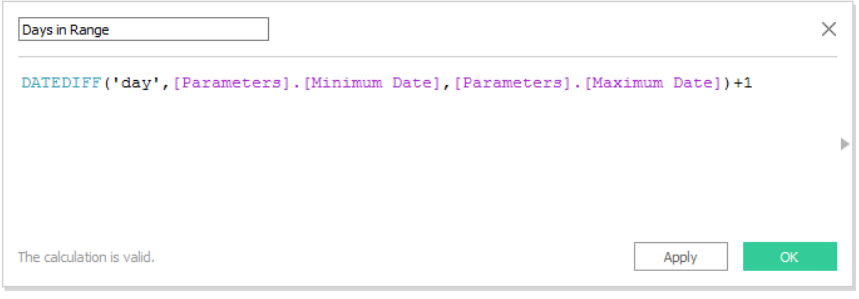
Control. If you would like to follow along using Sample – Superstore, the view looks like this at this point:



2. Create a calculated field for days in range.

   Create a calculated field that uses the DATEDIFF function to calculate the number of days in the selected range. In this tutorial, we are using a date part of day, so we will want to include a "+1" to ensure we capture the current day. The calculated field looks like this:

```
DATEDIFF('day',[Parameters].[Minimum Date],[Parameters].[Maximum Date])+1
```



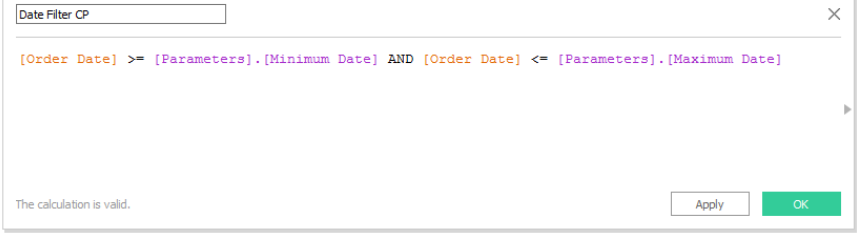
Note that if you want to spot check that your formula is working correctly and calculating the correct answer, add this calculated field to your view with an aggregation of average. By default, the aggregation will be sum so the calculated field will take the number of days multiplied by the number of records (which is not what we want).

Also note that the Minimum Date parameter is first in our calculation. If you do this backwards, you will get the wrong result.

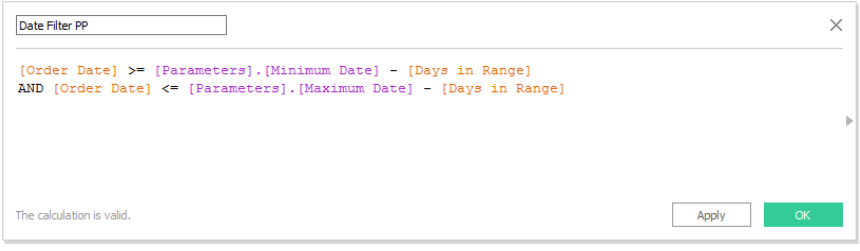
3. Create boolean calculated fields for current period and prior period.

   The final step in the setup process is to create two calculated fields—one which will limit the dates to the current period, and one which will limit the date to the period immediately preceding the range selected in the parameters from step 1:

```
[Order Date] >= [Parameters].[Minimum Date] AND [Order Date] <=
[Parameters].[Maximum Date]
```




```
[Order Date] >= [Parameters].[Minimum Date] - [Days in Range]
AND [Order Date] <= [Parameters].[Maximum Date] - [Days in Range]
```

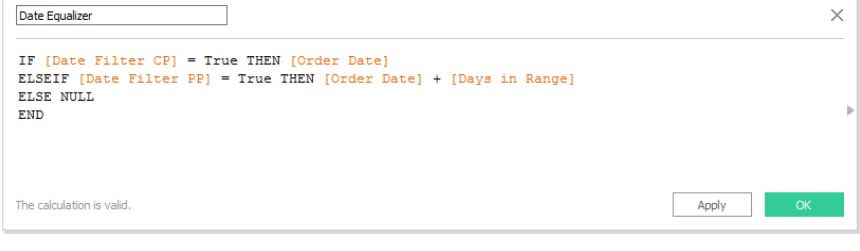


The date logic used to create these boolean calculations could have instead been nested within the calculations in the following steps but I prefer to leave them as their own fields in my data.

4. Create a date equalizer.

   The date equalizer is a calculated field that will put both the date range selected and the date range immediately preceding the selection on a single axis! The formula is:

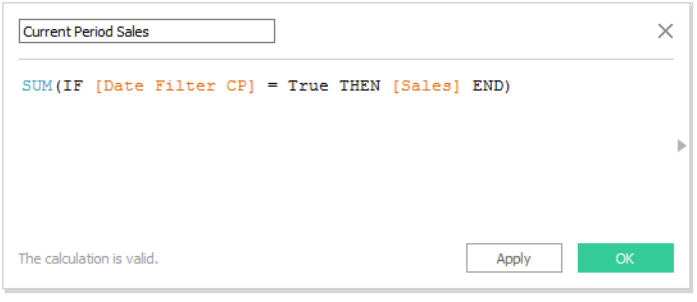
```
IF [Date Filter CP] = True THEN [Order Date]
ELSEIF [Date Filter PP] = True THEN [Order Date] + [Days in Range]
ELSE NULL
END
```



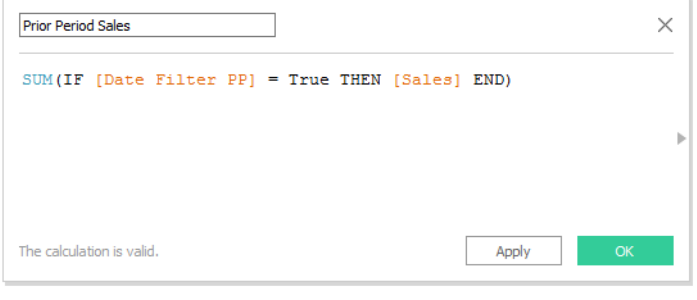
5. Create calculated fields for your measure.

   We are now ready to use the special date fields we set up before to create the measures that will be used on our view. For this to work, you will need to set up a calculated measure that shows the performance for the current period, and a second calculated measure that shows the performance for the prior period. I am going to use Sales as my measure, but this same approach works for any measure without having to replicate the first four steps. Here are my two calculated measures for Sales:

```
SUM(IF [Date Filter CP] = True THEN [Sales] END)
```




```
SUM(IF [Date Filter PP] = True THEN [Sales] END)
```



6. Create the view.

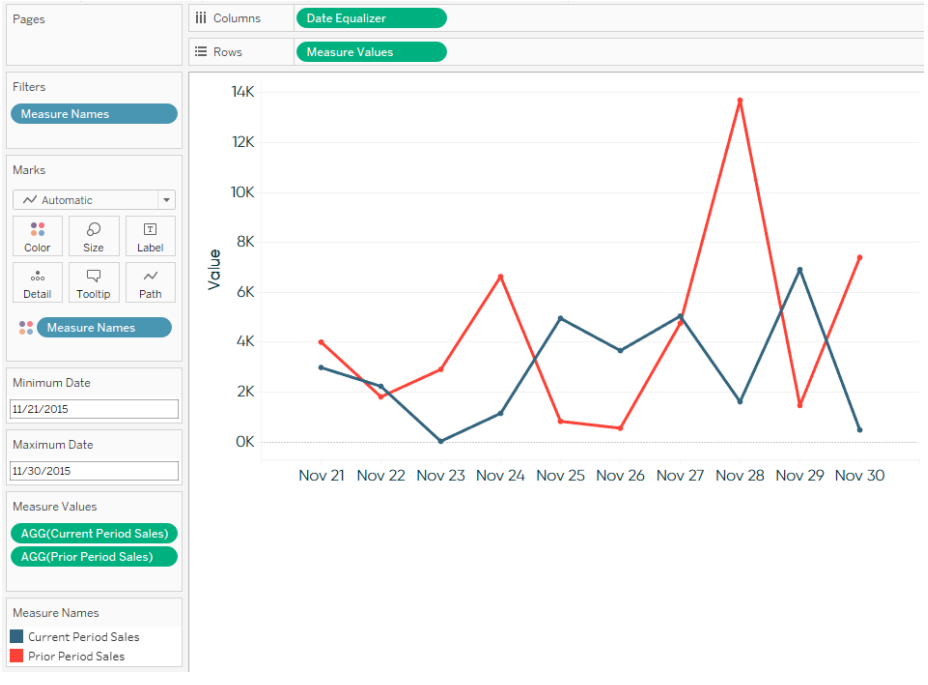
   If you are wanting to create a continuous line graph comparing the performance of a measure to the selected date range and the date range immediately preceding it on the same axis:

   a. Place the newly created Date Equalizer dimension onto the Columns Shelf with an aggregation of continuous day.

   b. Place the newly created Current Period Sales measure on the Rows Shelf.

   c. Drag the newly created Prior Period Sales measure onto the same axis as Current Period Sales—you could also create a dual-axis with one measure on each, but this would require you to synchronize the axes; a step that is unnecessary since these two measures should be an apples-to-apples comparison on the same y-axis.

      In my example, I have chosen the final 10 days of November 2015. The blue line represents the performance during my selected range, and the red line represents the performance during the 10 days prior to my selected range (11/11/2015–11/20/2015):



You can now select any date range you want and have Tableau show the performance during the current period to the prior period immediately preceding it—on the same date axis!

Once you've got the foundation down, try using parameters to change which metric is being graphed (Chapter 64) or change the date part (Chapter 66) of the date equalizer.

CHAPTER 70

How to Compare Unequal Date Ranges on One Axis

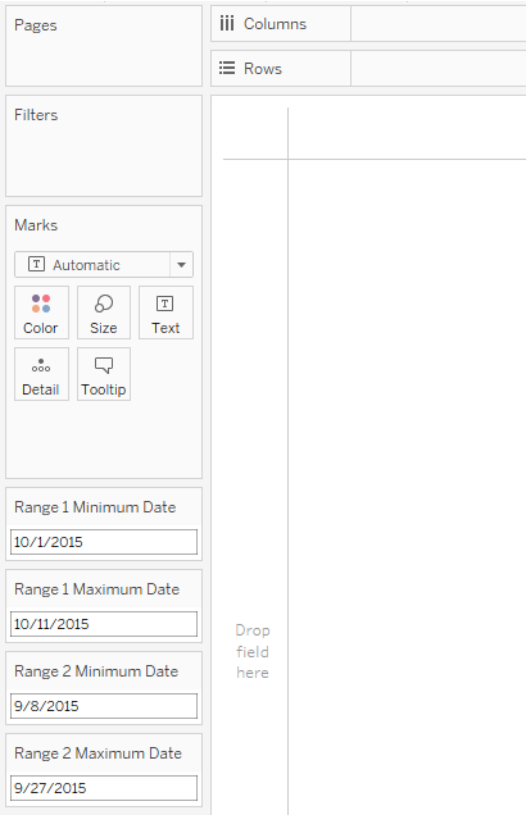
In the previous chapter, I shared a technique for comparing any date range in Tableau to the equal date range that immediately preceded the selection. But what do you do if you want to pick both ranges, even if they aren't right next to each other on the calendar? What if one range has a different number of days than the other?

This chapter shows how to compare the performance during any two date ranges on one axis, even if the selected date ranges have unequal durations.

First, create four separate parameters with a data type of Date:

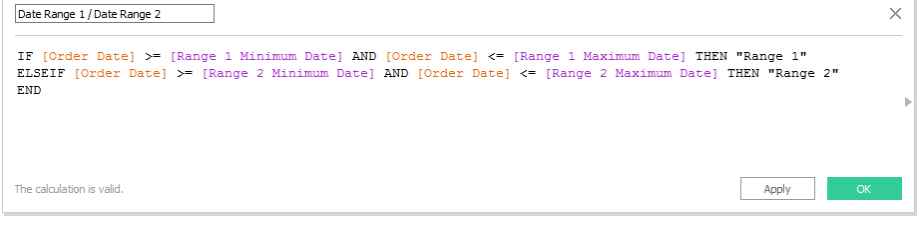
- Range 1 Minimum Date
- Range 1 Maximum Date
- Range 2 Minimum Date
- Range 2 Maximum Date

To start the view, show all four parameter controls by right-clicking on them and choosing "Show parameter control." This allows the end user to choose the two date ranges. For the purposes of this illustration, I will compare 10/1/2015–10/11/2015 to 9/8/2015–9/27/2015. Notice that not only are these date ranges disconnected (there are three days skipped between them), but the first range is 11 days while the comparison range is 20 days:



Next, set up a calculated field that classifies whether the date falls into Range 1 or Range 2. If you are following along using the Sample – Superstore dataset, the formula looks like this when using Order Date as the date dimension:

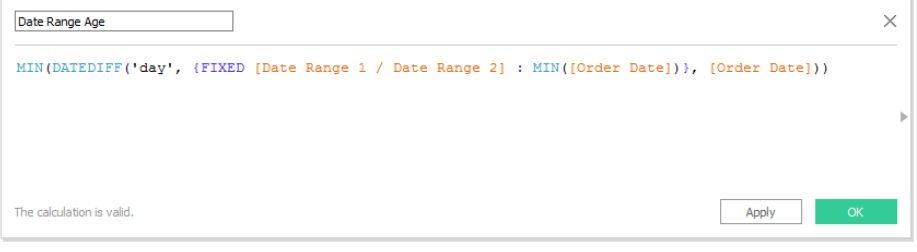
```
IF [Order Date] >= [Range 1 Minimum Date] AND [Order Date] <=
   [Range 1 Maximum Date] THEN "Range 1"
ELSEIF [Order Date] >= [Range 2 Minimum Date] AND [Order Date] <=
   [Range 2 Maximum Date] THEN "Range 2"
END
```



The last component needed to compare any two date ranges on one axis is a calculated field with a level of detail expression that computes the "age" of each date range and normalizes the two ranges by a relative date. This is where the magic happens. Hat tip to Tableau Zen Master, Joshua Milligan, who provided the foundation for this calculation in his blog post, "Tweaking Data Stories in Tableau".

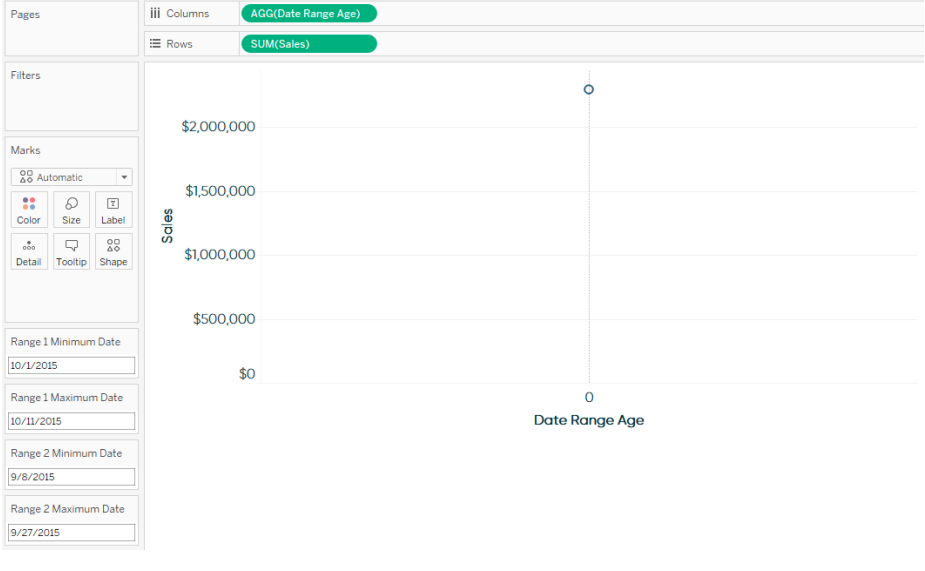
The formula is:

```
MIN(DATEDIFF('day', {FIXED [Date Range 1 / Date Range 2] :
   MIN([Order Date])}, [Order Date]))
```



Once you have the four date range parameters, a calculated field that classifies dates as Range 1 or Range 2, and a calculated field that calculates the date range age, you are ready to build the view.

Start a line graph by adding the measure you want to visualize across the two date ranges onto the Rows Shelf and the Date Range Age measure onto the Columns Shelf. Here's how the view looks so far using Sales as the KPI:



This view is showing a single mark because we need to include the Order Date dimension for Tableau to calculate the age of the date range. This can be accomplished by dragging the Order Date field (discrete at the M/D/Y level) to the Detail Marks Card:

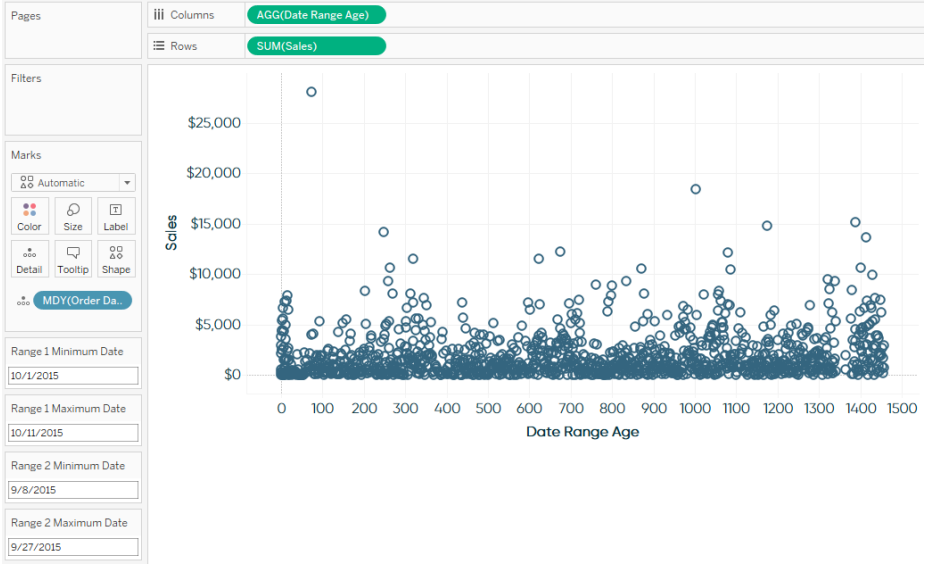
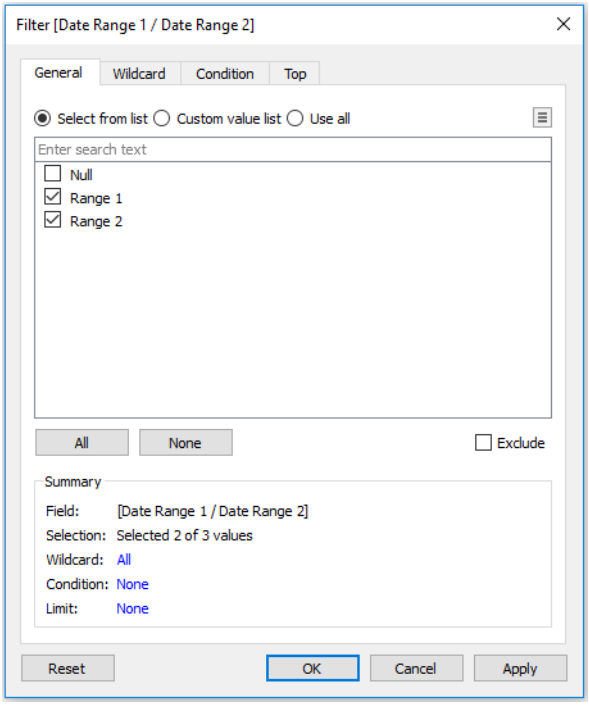
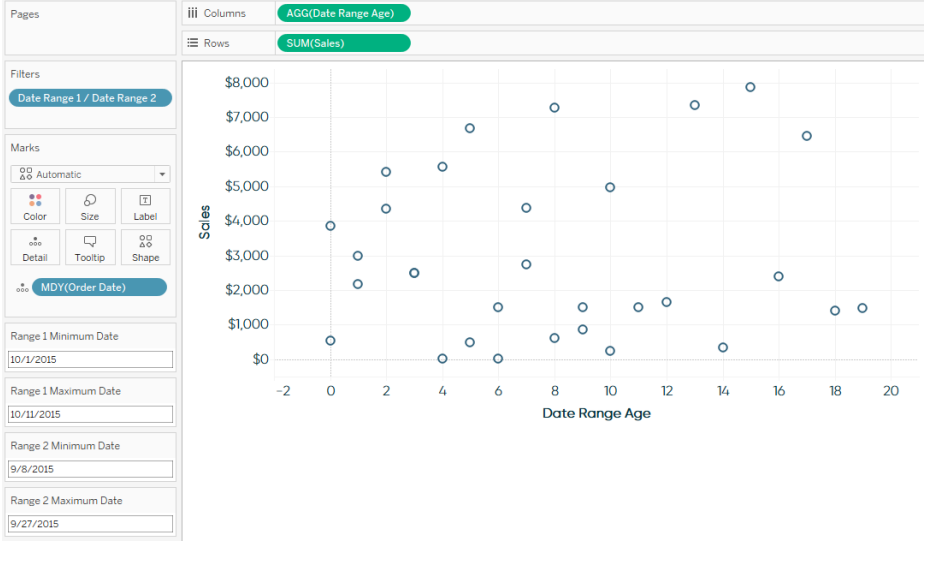


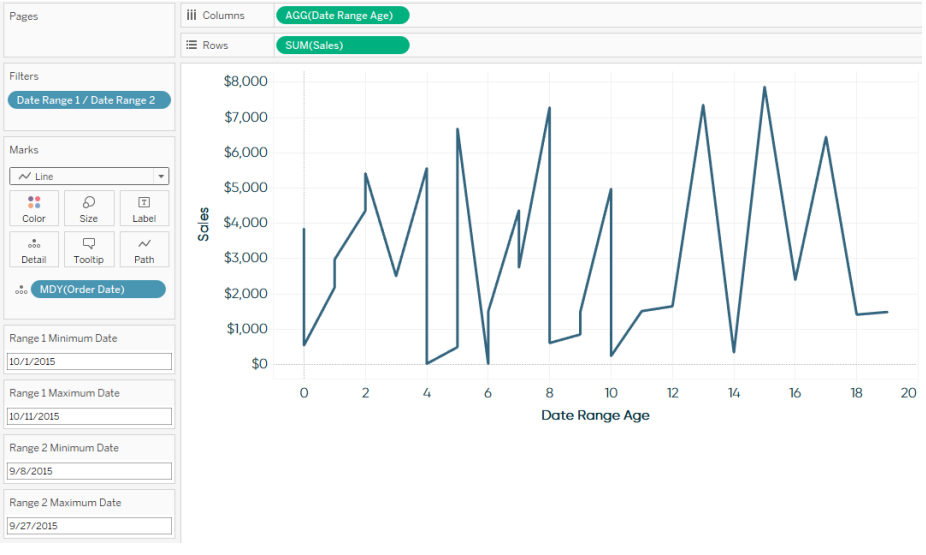
Tableau is now able to do the calculation, but we've now got 1,237 marks—one for every date in the Sample – Superstore dataset. To filter the view to use only the relevant dates for this analysis, drag the Date Range 1/Date Range 2 calculated field to the Filters Shelf and keep only the dates classified as Date Range 1 or Date Range 2:



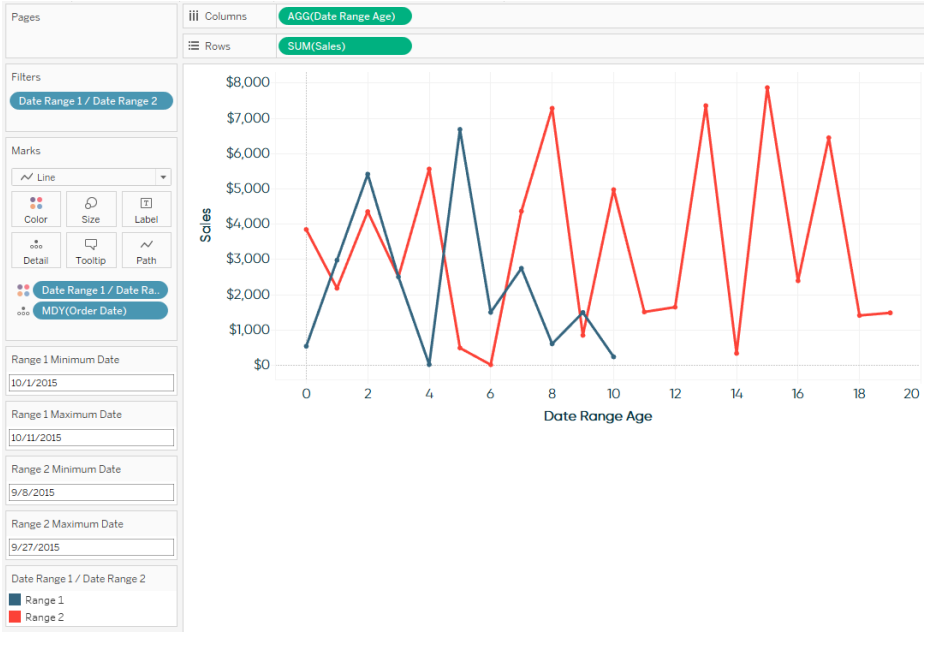
After applying the filter, we are left with only the 31 marks for the date ranges we have selected using the parameter controls:



To change this to a line graph, change the mark type from Automatic, which is currently Shape, to Line. This creates one continuous line graph:



The final step to compare the performance between the two date ranges is to drag the Date Range 1/Date Range 2 dimension to the Color Marks Card. Here is how my final view looks after coloring the lines by date range and polishing the formatting:



We are now able to choose *any* two date ranges and compare them to each other on the same relative date axis!

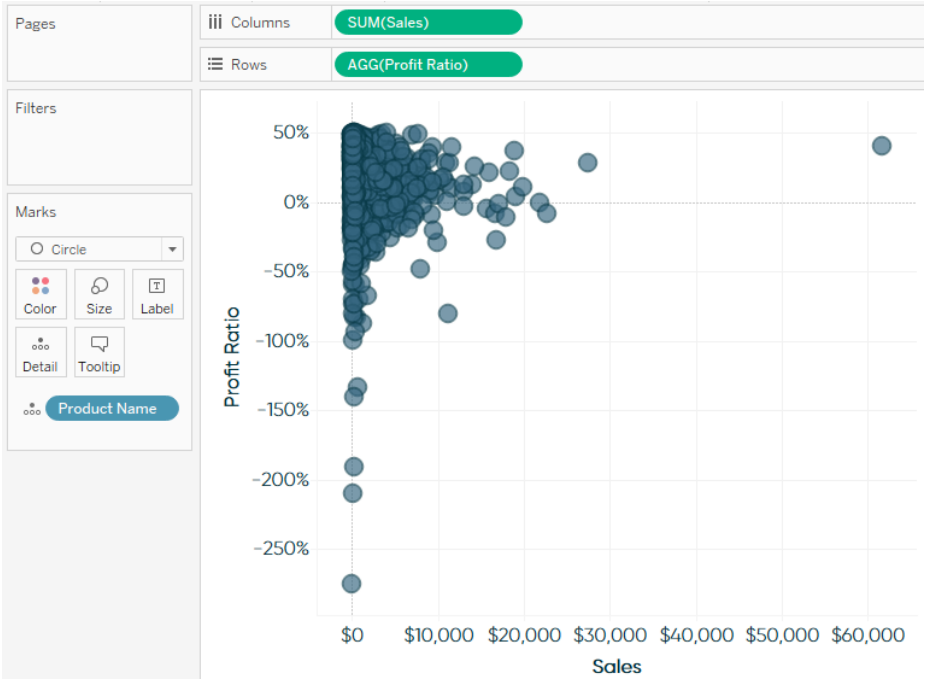
CHAPTER 71

How to Make a Cluster Analysis

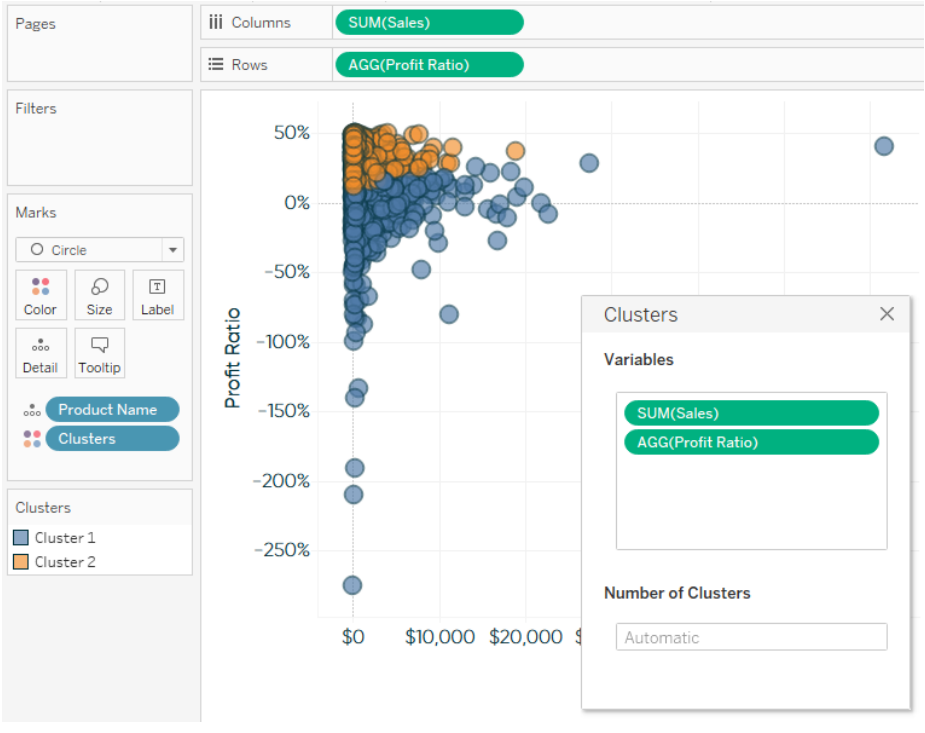
Clustering is a powerful feature released with Tableau 10 that allows you to easily group similar dimension members. This type of clustering helps you create statistically based segments that provide insight into how different groups are similar as well as how they are performing compared to each other. I've always leaned on segmentation as a tactic for making my analyses with Tableau more powerful, and Chapter 22 in Tableau, shares how to make a simple quadrant-based segmentation. Clustering takes this a step further by statistically grouping the objects on a view using the variables on the view.

This chapter shows you how to use the cluster feature in Tableau and how to make the generated clusters more permanent for segmentation analyses.

To demonstrate, I will first re-create the scatter plot just mentioned, which looks at sales and profit ratio by the Product Name dimension in the Sample – Superstore dataset:



To create quadrant-based segments, at this point I would add a reference line for average on each axis. Now with Tableau 10, we can make this segmentation much more scientific by using the Cluster feature. Cluster lives on the Analytics pane in Tableau, so to create a cluster analysis, simply navigate to the Analytics pane (toward the upper-left corner of the authoring interface) and drag "Cluster" onto the view:



A few things to point out here:

- By default, Tableau created the clusters from the variables on the view (Sales and Profit Ratio). You can add or take away variables to customize the clusters.
- "Clusters" was added to the Color Marks Card, which colored each circle by its respective cluster segment. Remember that you can only color marks by one thing at a time in Tableau.
- Tableau automatically identified two similar groups of marks. This number may go up or down based on the variables in the cluster analysis. You can also manually set the number of clusters by entering a number in the box that says Automatic.

After you're done creating the clusters, click the X in the upper-right corner of the Clusters dialog box to remove it from the view. You can always edit the clusters again by right-clicking the Clusters dimension on the Color Marks Card and choosing Edit clusters.

Creating a cluster analysis is that easy, but here are a few more ways this feature can be used:

- If your resident data scientist is scoffing at your work because they don't know how the segments were generated, you can right-click the Clusters dimension on the Color Marks Card and choose "Describe clusters." This will provide all of the summary statistics and modeling you can handle!
- You can filter the view to look at only certain clusters at a given time. Dragging Cluster from the Analytics pane onto the view does not create a permanent dimension on the Dimensions area of the Data pane, but you can still filter on the field by right-clicking it from the Color Marks Card and choosing Show Filter.
- If you would like to create a permanent segment from a cluster for future analyses, filter the view to the cluster of interest, use Ctrl-A to select all of the marks (or left-click and draw a box around all of the marks), right-click one of the highlighted marks and choose Create Set.

For more on filters, review Chapter 11.

For more on sets, review Chapter 15.

CHAPTER 72

Five Tips for Making Your Tableau Public Viz Go Viral

To this day, I am grateful when even a single person checks out one of my data visualizations. I often think back fondly about my very first Tableau Public visualization, a simple view showing NFL interconference records over time. The viz is not my best work—if anything, it's my worst work—but I was so excited to discover that I can share interactive data stories for free through Tableau Public (so awesome!).

The viz now has over 25,000 views.

What I like most about this viz is that it's a constant reminder of how much you can improve if you stick with something, and it's also a testament that any viz has a chance to be a popular success. Among a few other vizzes with at least 25,000 views, I now have five others with at least 50,000 views, one that recently cracked the six-figure milestone, and even one with over a quarter-million views (shown later in this chapter)! My work has been featured by *The Guardian*, *U.S. News & World Report*, and *Grantland*—all with zero marketing budget and no established journalism platform on which to share my work. This chapter shares the five tactics that made it possible.

While I certainly do not believe that a visualization's value should be judged exclusively by how many views it has, I do believe that part of a data visualization's success should be judged on whether or not it made an impact. And sometimes you need a larger audience to see a viz in order for it to make an impact. I have seen incredible work on Tableau Public with fewer than 100 views. In fact, that's why I wanted to write this chapter—to help as many Tableau Public authors as possible have their deserving work discovered by a larger audience.

Tip #1: Create "Remarkable" Content

For your viz to have any chance of going viral, it is critical that it is remarkable in some way. When I say remarkable, I am talking about Seth Godin's literal definition of remarkable, meaning your work is interesting enough in some way that it makes people remark about it. In today's social media age, remarking about your work often means sharing it on social media with new audiences—one of the biggest contributors to viral success.

Tableau Public visualizations can be remarkable in different ways. Perhaps you have found a compelling story to share or a unique approach to sharing that story in Tableau. Or maybe you have done a phenomenal job using the next tip to your advantage: balance data and design.

Tip #2: Balance Data and Design

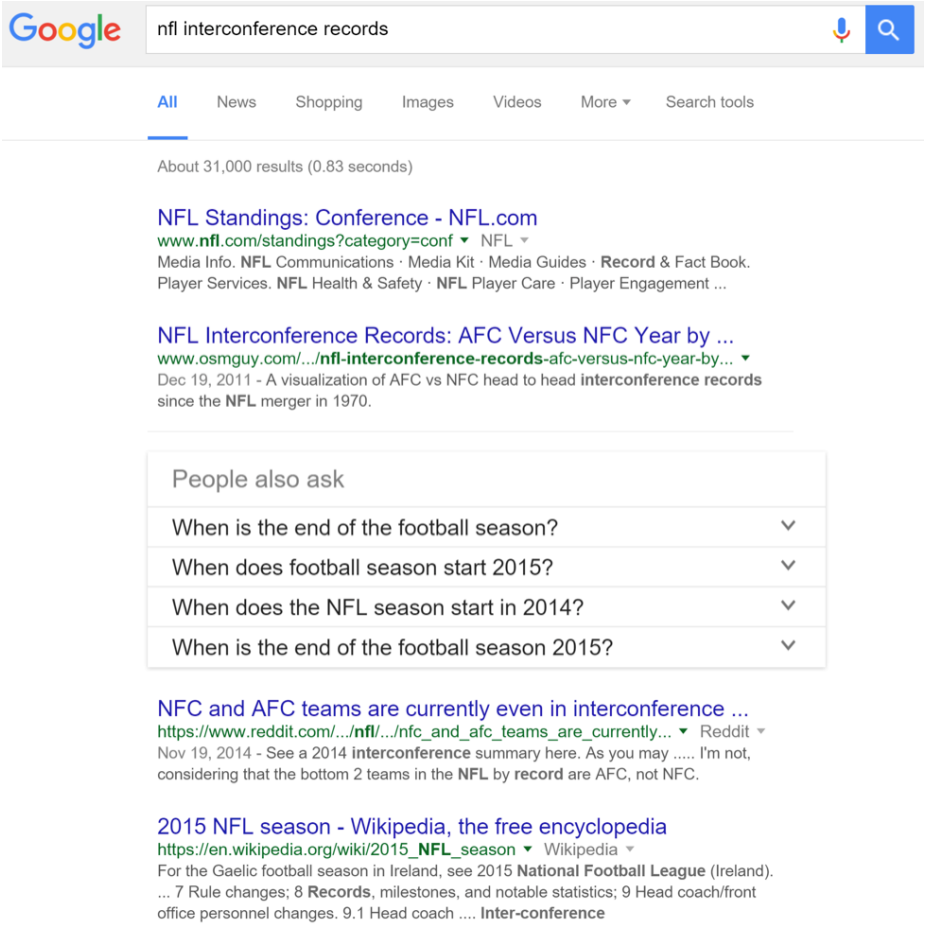
It seems there are two camps in data visualization: the data purists who feel aesthetics are unnecessary or even counterproductive, and those that believe design is a worthy component to data visualization. I am firmly in the second camp. My stance stems from how I define the purpose of data visualization: to find and share actionable stories that are based in quantitative evidence.

To make your insights actionable, this means sharing them with the most relevant, and many times, largest audience possible. If you don't provide your data stories in a well-packaged design, you drastically minimize the chances of your work spreading, and thus, it is consumed and acted on by fewer people.

The good news is that anybody can improve the design of their Tableau Public visualizations. See Chapter 54 for some inspiration.

Tip #3: Leverage Search Engine Optimization (SEO)

I mentioned in the introduction that even my first and most basic Tableau Public visualization has hit the 25,000 views milestone. Here's a Google search result for the term "NFL Interconference Records":



Among the 31,000 results, there's a link to my (possibly worst) Tableau Public viz in the second position, right behind the NFL itself and ahead of popular websites Reddit and Wikipedia. I attribute this search result directly to search engine optimization, or SEO.

SEO can feel like a mysterious marketing strategy, and in some ways, it is. After all, nobody knows Google's search algorithm and we have no control over it. However, there are some basic SEO tactics that will dramatically improve the chances of your viz going viral by being discovered by the right person in search.

These tips are assuming that you have the ability to embed your visualizations on a platform that you control. If you don't have your own blog, I encourage you to work through the "A-Z Miniguide on Setting Up a Data Blog".

Most importantly, try to get in the heads of your audience and think about the terms they might use to talk about your viz. People go to Google to ask questions, and if you can integrate the words people are using to ask those questions into your work, there will be a better chance that Google will believe your viz is the best answer to their question.

Create a descriptive page title using the terms people are searching for. It is tempting to try to be cute with your page titles, but descriptive titles do better in search. For example, I could have named my Odds of Going Pro in Sports viz and chapter "You won't believe how hard it is to do this in sports." Like a bad clickbait ad with a strange picture on it, this title may do a good job of getting clicks when it's shared on social media, but it's not what people are searching for in Google. So after the initial social push, it would be difficult for this to be found again. Instead, people are searching for "What are the odds of going pro in sports?" and this viz continues to be found regularly today, years after it was originally published.

Provide some thoughts around your viz in your post. Google's algorithms are designed explicitly to understand textual content that's written for people—and they're getting better and better at it every day. Unfortunately, there is very little metadata for Google to find in a Tableau Public visualization itself. For this reason, it's important to write at least 250–500 words about your viz to describe what it's showing and what key insights it provides. This will give Google more information about what your viz is about so that it knows if your viz can help answer a searcher's question.

Tip #4: Network

Everyone knows that if you want a visualization to be publicly consumed, one of the first places to share it is on social media networks such as Facebook, Twitter, LinkedIn, Pinterest, and so on. To take this a step further, I note whenever I have a breakthrough. That's when a popular account shares my work or a journalist picks it up for his or her own content. I will then reach out to them directly the next time I have new material that is relevant to them. This tactic takes time to cultivate, but the snowball effect can be extremely impactful.

There is a fine line between building relationships and spamming, so beware. I am not suggesting you send an email to everybody in your audience who has ever looked at your work, but I may pick two to three people to share each new viz with directly, and only occasionally when I genuinely think the work is relevant to them. This tactic has led directly to second shares on *U.S. News & World Report* as well as the popular social sharing site Digg. Be sure to pay back these members of your network by supporting their work as well. And also be sensitive to not spamming them if you are getting hints that your work may not be as relevant as you may think it is for them anymore.

Tip #5: Use Reddit

The reddit.com/r/dataisbeautiful subreddit has an enormous and relevant audience for Tableau Public authors. Using Reddit provided the majority of views on my most popular viz to date, The Cost of Attending the Baseball Championship Series, and provided a spring board for the most viral viz I've ever seen, Tableau Zen Master Adam McCann's Analysis of the Beatles. With such a huge potential audience, it is exciting to jump right in and think you'll instantly gain fame and fortune. However, it took me several years to have a breakthrough on Reddit, and here's what I learned along the way.

You have to genuinely immerse yourself in the community there. You cannot get away with only posting your own content. Support others by upvoting and commenting on their work or by sharing content from others.

When you share your own content, make sure a thumbnail preview image is picked up with your post. The easiest way to guarantee this is to embed your Tableau Public visualizations on your own site, with a prominent feature image somewhere in the post. Studies have shown that email campaigns with images earn a 42 percent higher clickthrough rate, and the same principle applies to the dataisbeautiful subreddit.

Reddit says that original content (OC), or content submitted directly by the content's creator, performs better on its platform compared to non-OC content. This may be true after you have been using Reddit for a long time and built a strong reputation in the community, but this has not been my personal experience. My work has always performed better when somebody else submitted it. I don't have the science to back this up, but I think people are less likely to view and share work if they think there is even a chance that the person posting the content is self-promoting or sharing the content for the wrong reasons. I encourage you to experiment with both OC and non-OC content. If you don't have a colleague handy to submit your content, I'd be happy to take a look and share relevant visualizations myself (just let me know @ryanvizzes on Twitter).

Ultimately, we don't have full control over how popular a Tableau Public visualization will be, but don't let that discourage you. Not every single one of my Tableau Public visualizations have been a hit, but there is always something positive that comes out of every one. I've either discovered a new approach to visualizing certain types of data or created a new example that I can use in training. At a minimum, I get a chance to practice and learn something new for next time.

I also know that if the five tips mentioned here are taken to heart, the viz can be discovered and go viral at any time. If you adopt these tactics as part of your process and stick with it, I'm confident you will see a noticeable lift in your Tableau Public visualization views.

CHAPTER 73

Three Ways to Make Beautiful Bar Charts in Tableau

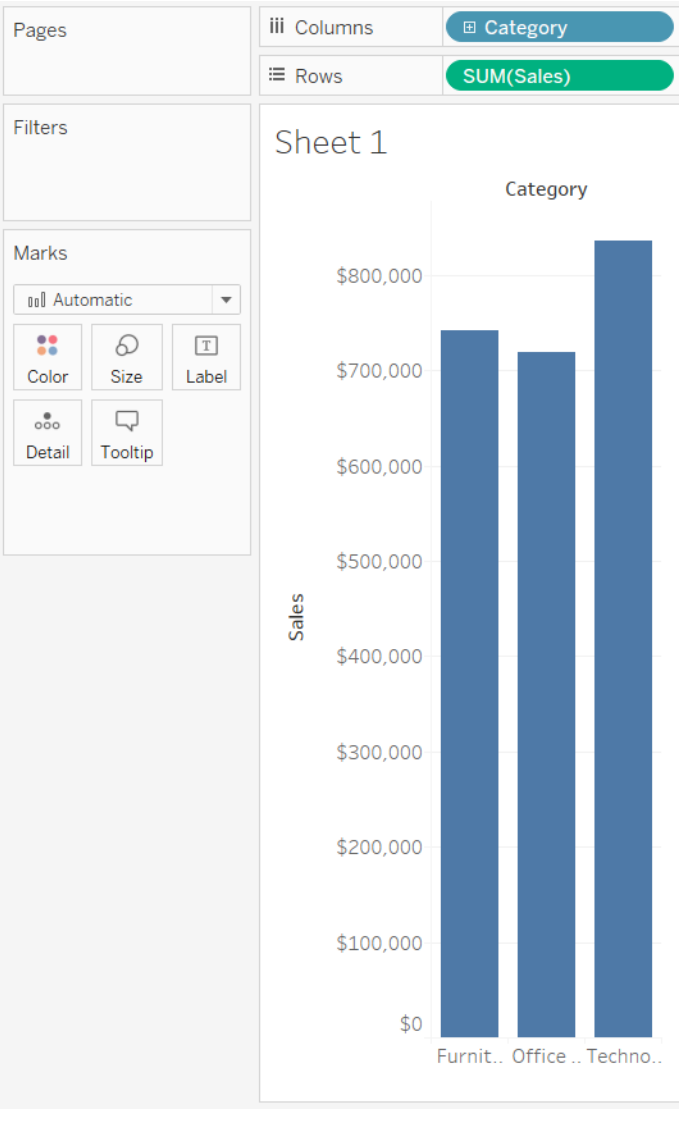
When it comes to data visualization, bar charts are still king. With all due respect to my other favorite fundamental chart types such as line graphs and scatter plots, nothing has the flexibility, ease of use, and ease of understanding, as the classic bar chart. Used to compare values of categorical data, bar charts work well because they take advantage of a basic preattentive attribute: length. Our ability to process the length of bars with extreme efficiency and accuracy makes the bar chart arguably the most powerful data visualization choice available to us.

The invention of the bar chart is credited to William Playfair, with his *Exports and Imports of Scotland to and from different parts for one Year from Christmas 1780 to Christmas 1781* being the first appearance. Extraordinarily long and descriptive titles aside, bar charts have been making an impact for a long time. In fact, I hypothesize that the fact bar charts have been around for so long is one of the reasons some attempt to find a "more engaging" chart type to tell their data story.

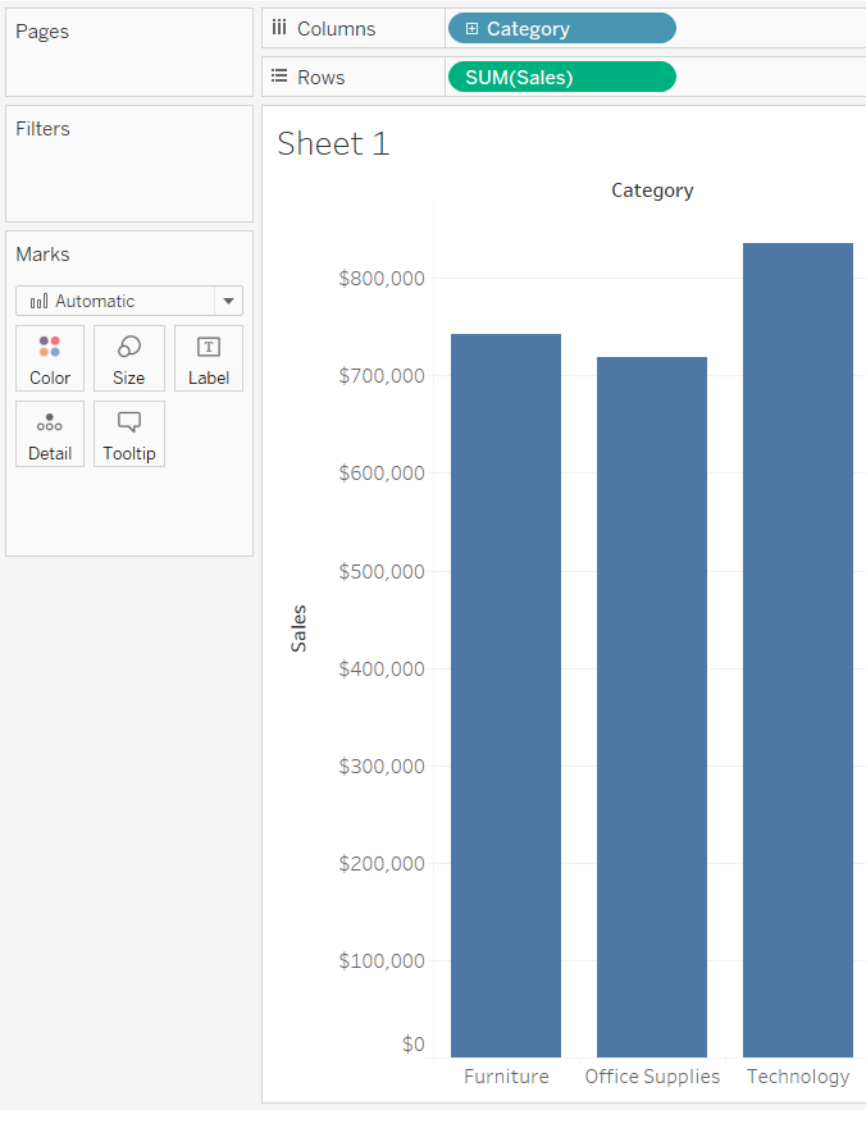
This chapter attempts to add some love for bar charts by sharing three ways to make them more engaging in Tableau.

Approach #1: Use Formatting Available in Tableau

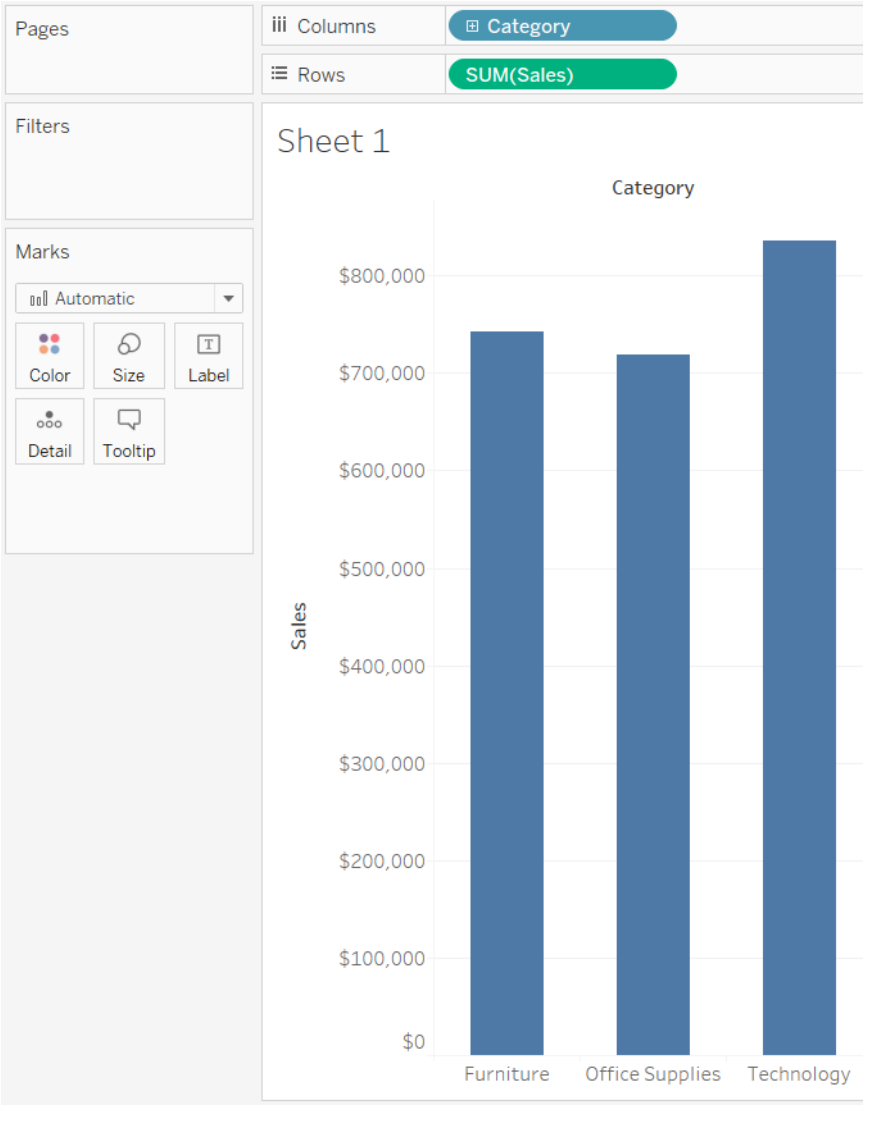
My first tip for making beautiful bar charts is to use the formatting options you already have available in Tableau. Consider the following Sales by Category bar chart that shows all of the default Tableau settings:



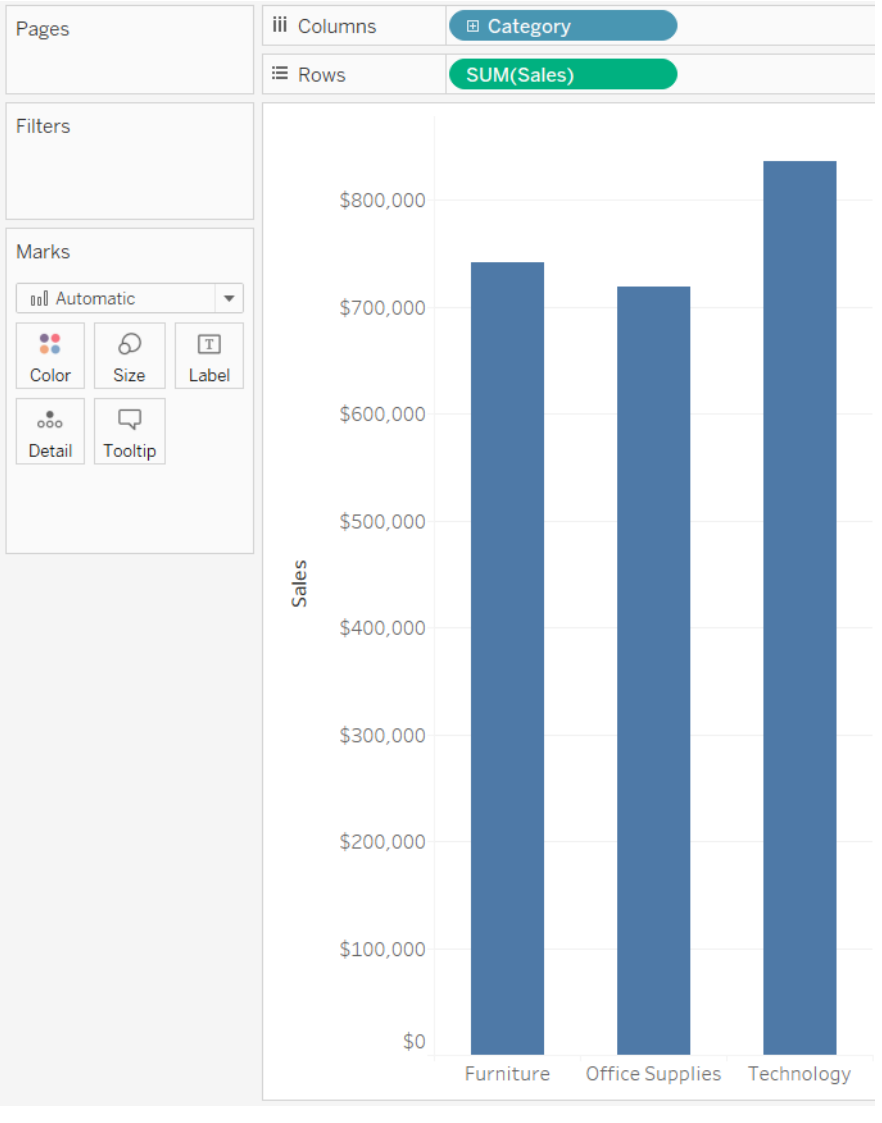
This bar chart gets the job done, as you can immediately decipher that Technology leads the way with over $800,000 in sales, Furniture contributes the second most, and Office Supplies contribute the least. However, there are several opportunities to make this bar chart more engaging and effective. The most obvious of which is to widen the columns so the categories can be read:



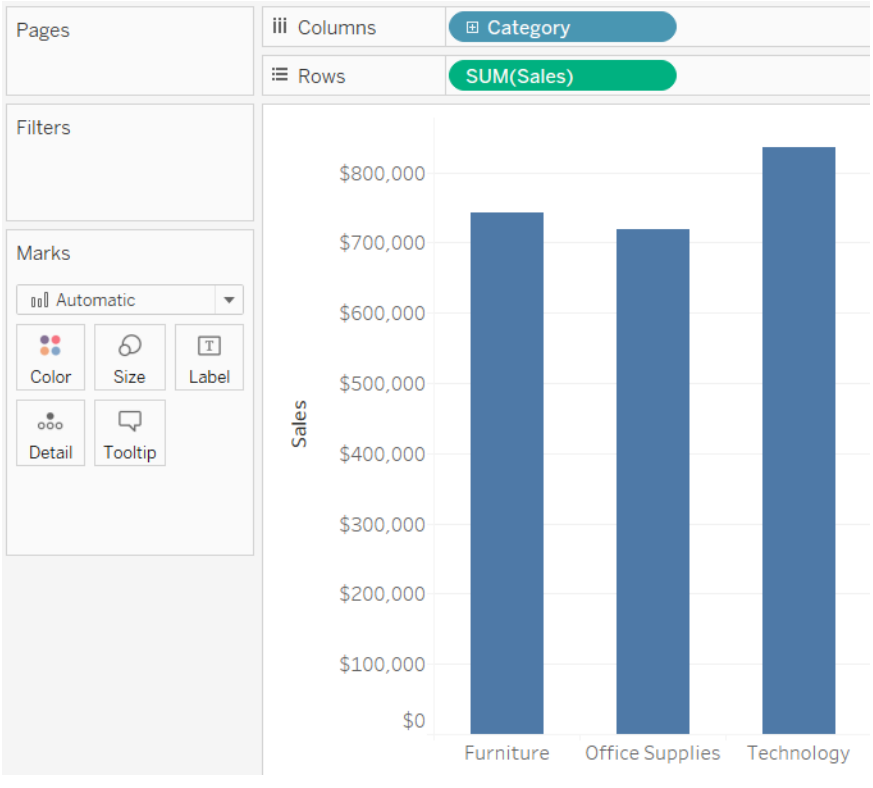
Making the columns wider makes the bars themselves wider. In my opinion, these bars are now too heavy relative to the rest of the visual. The next step I'll take is to reduce the size of the bars by clicking the Size Marks Card and dragging the slider to the notch in the middle:



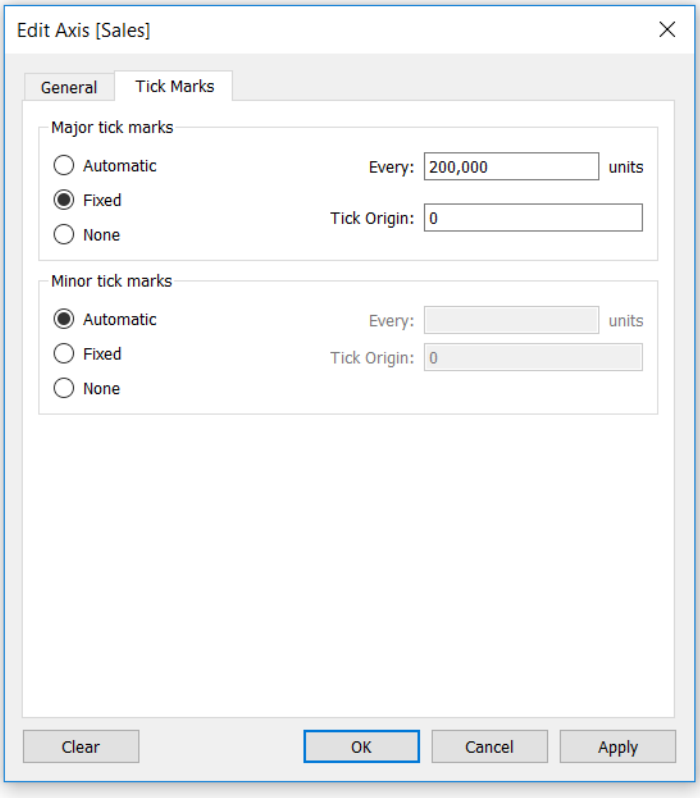
The next tip is arguable, but I'm not as descriptive as William Playfair was with his 110-character chart name. In my experience, the context of the chart is provided in surrounding text and/or dashboard titles, so I am going to hide the sheet name by right-clicking the title and choosing Hide Title. I am also going to right-click the bar chart header, Category, and click "Hide Field Labels for Columns." If this is a stand-alone visualization, I recommend keeping the title:



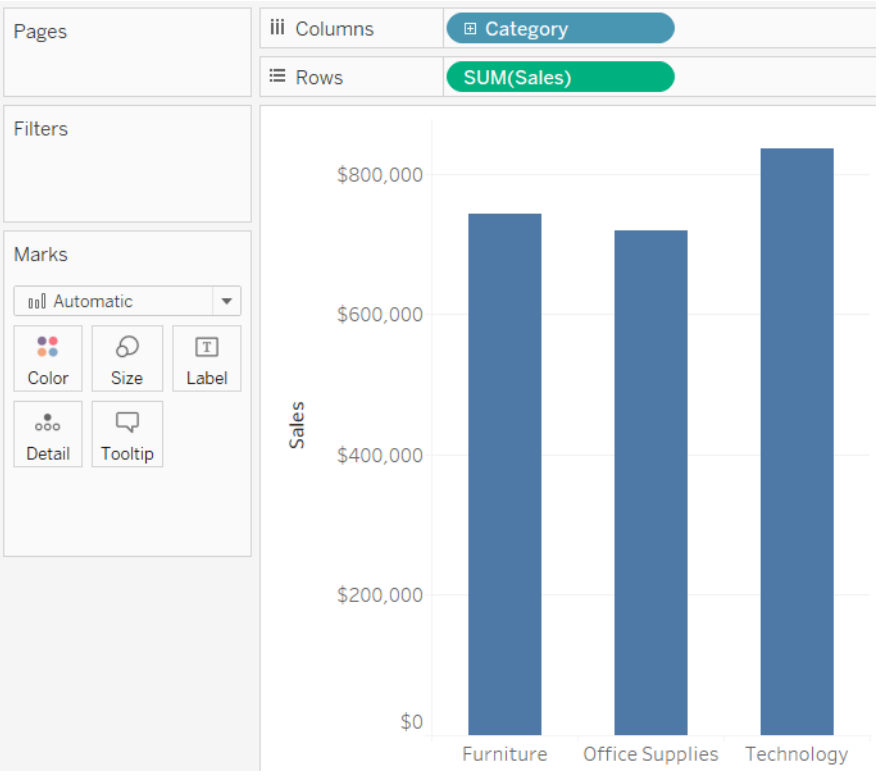
The bars in this chart are unnecessarily tall because there is not much variance between the categories in this analysis. Here's how the bar chart looks after I reduce the height by about 40%:



Take this next step on a case-by-case basis, but another side effect of having limited variance between the three bars is that there are too many gridlines and axis marks. This is negatively impacting the data-ink ratio, and can be cleaned up. To reduce the number of axis ticks, right-click the axis, click Edit Axis, and navigate to the Tick Marks tab:



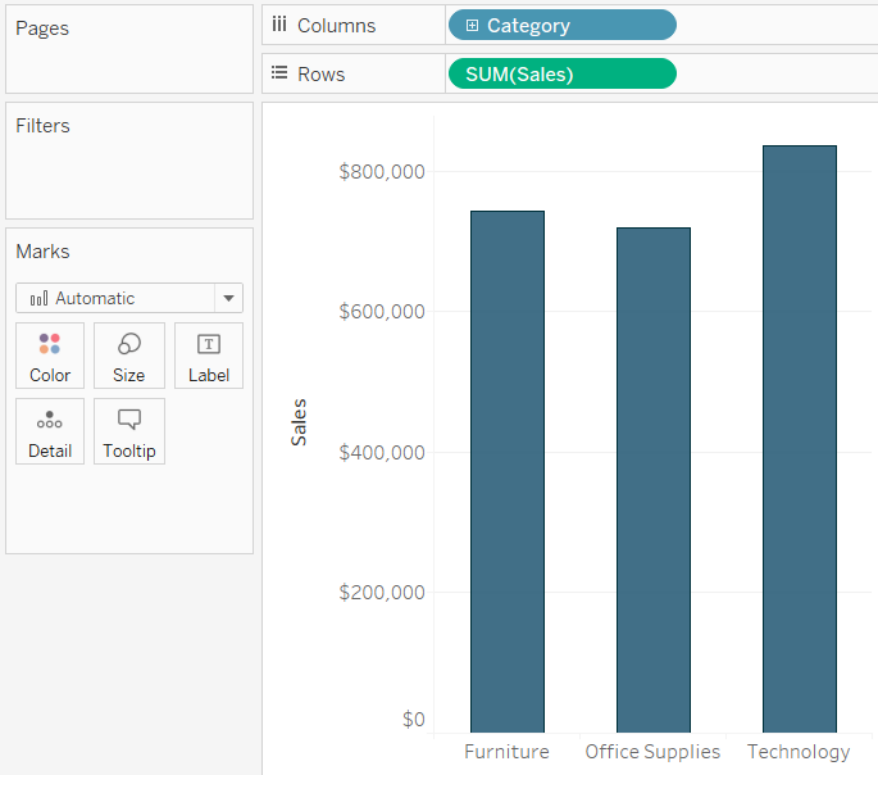
Here's how the bar chart looks after fixing the tick marks at 200,000:



Last, but not least, color. I have written quite a bit about color, including Chapter 55, and in Chapter 75 and storytelling tip #3 in Chapter 88. There is so much to be said about color, but for the purposes of this chapter, I will offer just three thoughts on coloring your bar charts:

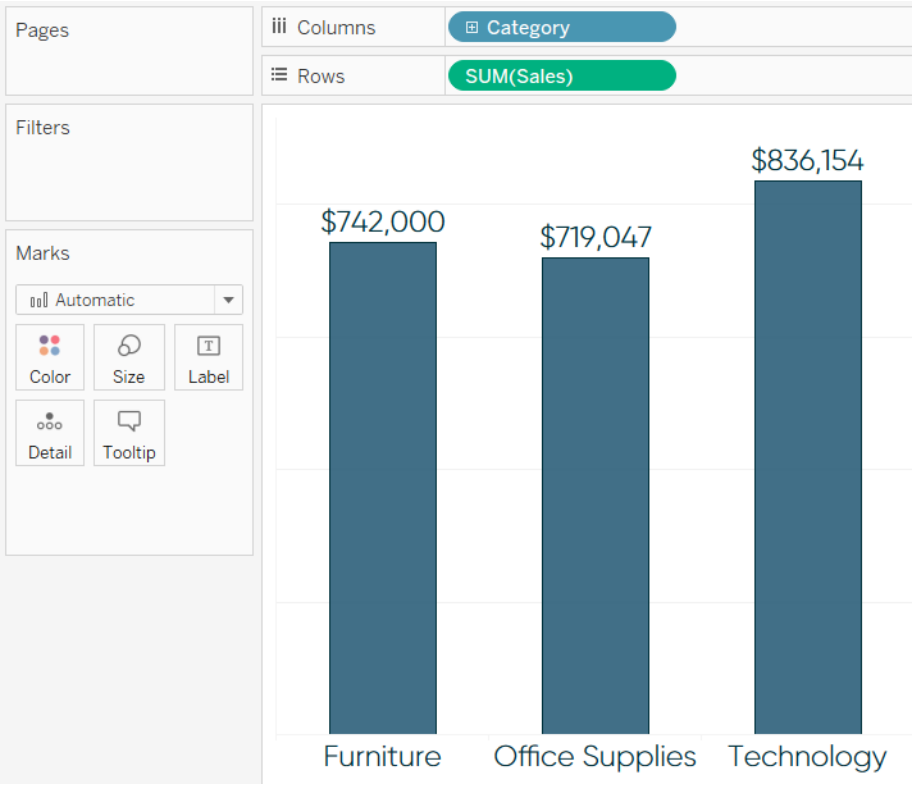
- Avoid double encoding. The bars in our chart are already separated by category. Adding category again to the Color Marks Card to color each bar with a unique color is unnecessary and potentially confusing. The one exception to this is if the colors are being used to provide a link between multiple visuals on a dashboard.
- Reduce the opacity from 100% to 80%–90% by clicking the Color Marks Card and moving the opacity slider to the left. This is a very subtle technique for reducing the saturation of the color and making the visual a little easier on the eyes.
- Use this as an easy opportunity to brand your data visualization. If you can't decide on a color, consider using a primary or secondary color from you or your end users' brand.

Here's how my final bar chart looks after choosing a secondary color from my personal brand and reducing the opacity of the bars to 90%. Note that I also added a very subtle border to the bars, which can be found in the options on the Color Marks Card:

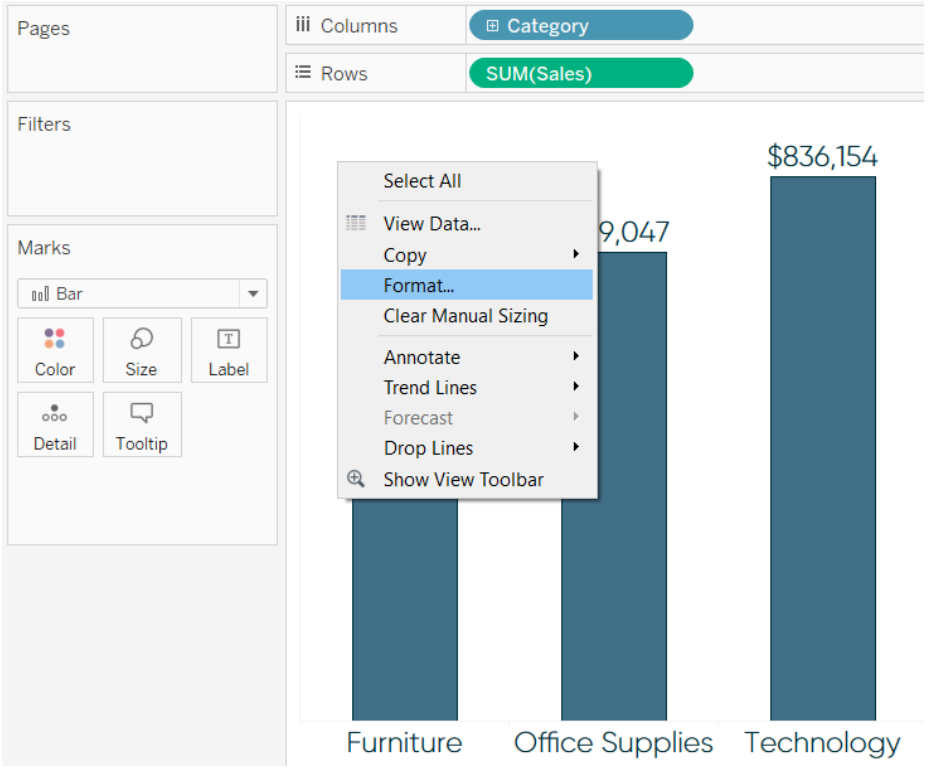


Approach #2: Use Axis Rulers to Add a Baseline

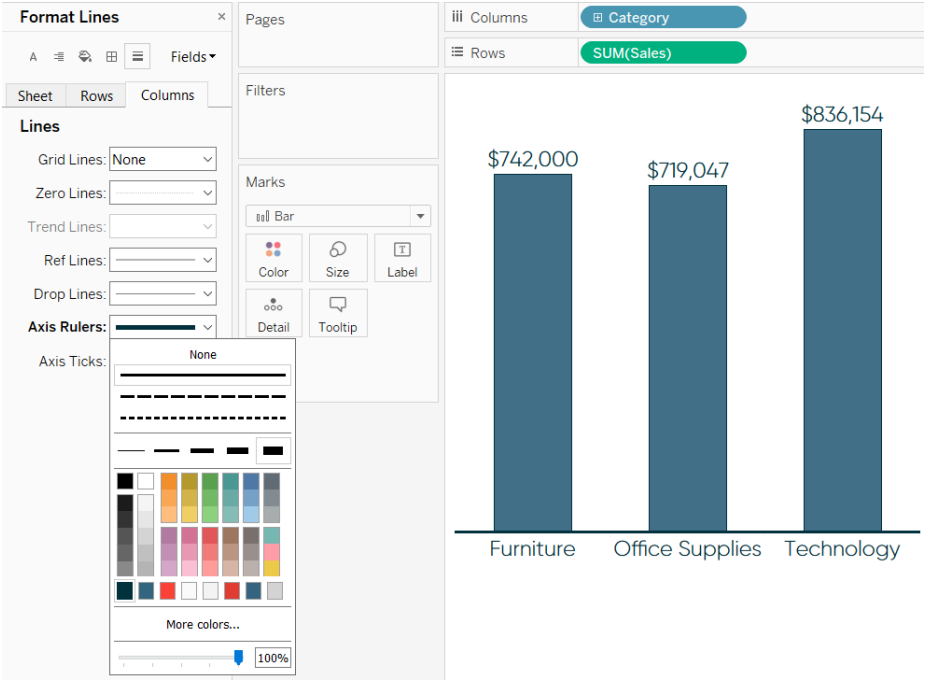
In the last tip, I mentioned the data-ink ratio. This a concept introduced by Edward Tufte that essentially says you should dedicate as much "ink" on a data visualization to the data as possible. One way I sometimes achieve this is by hiding the axis altogether and adding labels to the bars:



To add to the first tip, formatting, I also made the font larger and in brand. I like this look, but don't like how the bars appear to be floating. What I would like to do is add a solid foundation for the bars to sit on; this provides a practical purpose and also enhances the design. The easiest way to add a baseline is to modify the formatting of the view's axis rulers. By default, axis rulers are set to be a very light, thin gray line. To make the line heavier and match the color of the bar chart you are creating, right-click the view and click Format:



This will open the Format pane on the left. Navigate to the Format Lines tab and modify the formatting for the Axis Rulers for Rows and Columns. To clean up the view, I'm going to set the Axis Rulers for Rows at None, which removes the thin gray vertical line on the left side of the bar chart. I'm then going to format the Axis Rulers for Columns to be a solid, thick line that matches the color of the bars' borders:



Here's how my final bar chart looks after removing all of the other lines except for the baseline. Compare this to the default Tableau bar chart in the first image!