# Tableau Introduction

Tableau is a data visualization software that is menu-driven and more user-friendly, compared to scripting languages (R, Python). Under the hood, Tableau uses VizQL language to convert our dragged-and-dropped activities into queries that result in the graphics.

We are using Tableau Desktop for this class. We will work with a few datasets to make various visualizations. For this chapter, the dataset we are using is called “patient insurance claims”. It is a simulated dataset which is in Excel format (.xlsx) and has the following columns: “patient\_id”, “state”, “insurance\_payment”, “deductible”, “copay”, “payment\_year”, and “region”. The dataset is at the patient level (each row is one unique patient). The cost columns represent the respective patient’s costs in a particular year. The dataset has 30,000 rows.

We will open Tableau Desktop that we have downloaded and installed. We should see the following options after opening the application.

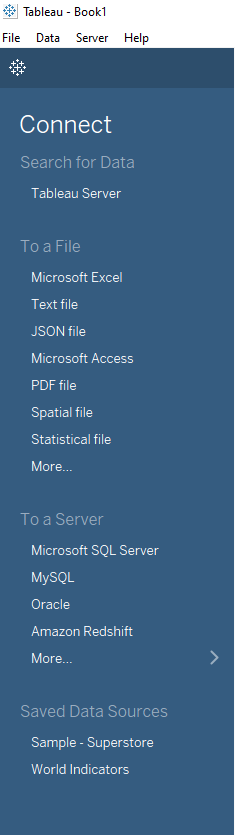


Figure : Screenshot of the connection choices we get when we open Tableau Desktop

We can see the options for different data types that Tableau can connect to. As we have an Excel dataset, we will use the Microsoft Excel option to connect to our data.

Once we connect to the data, we see:

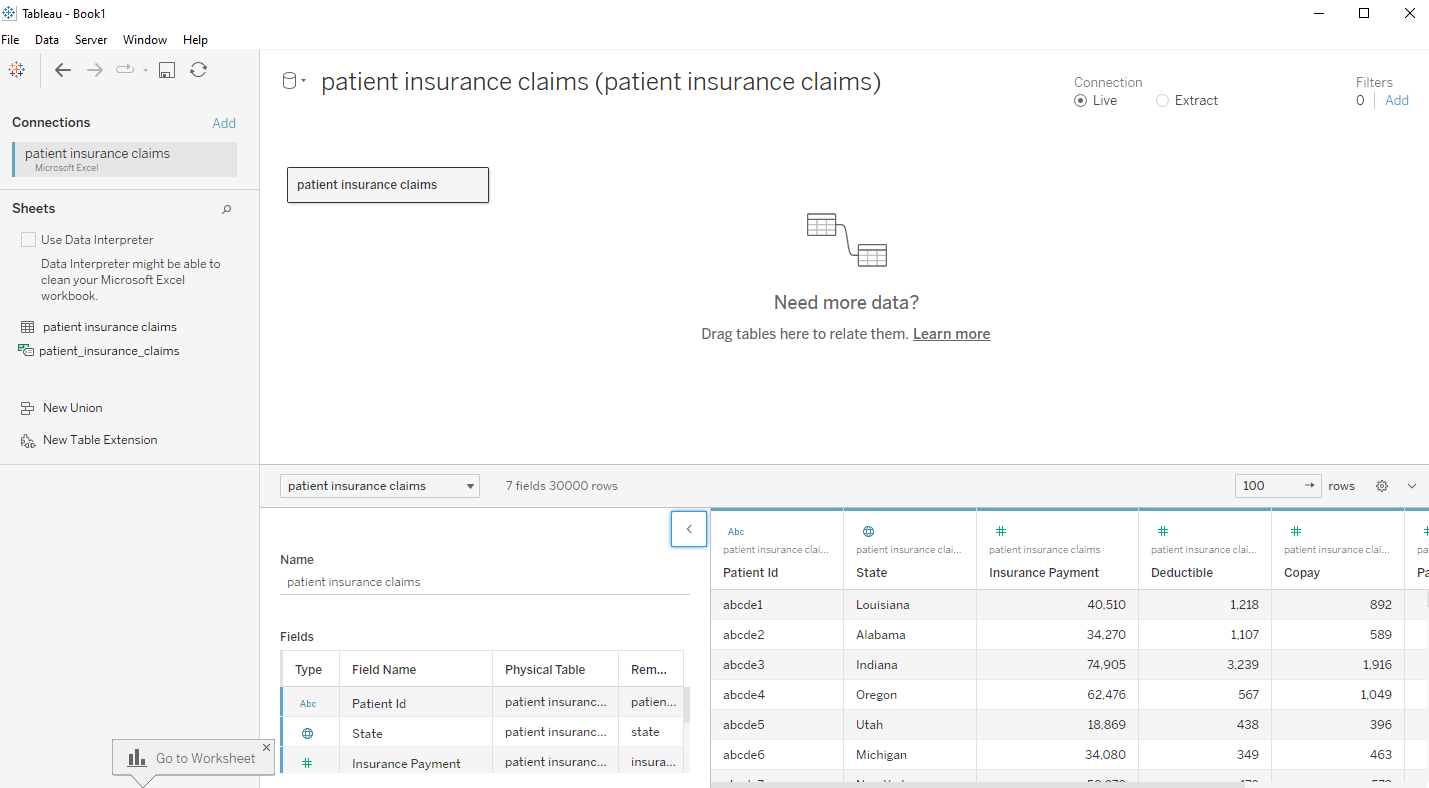
**

Figure : Screenshot of the Data source page after connecting to patient insurance claims data

The section under the red rectangular box shows the metadata, which includes all the column names, their types and the name of the table (dataset) that the columns come from. We can hide this section by clicking on the “<” symbol. In Tableau, columns are known as fields and datasets are known as tables. The section under the blue rectangular box shows the number of fields and rows in the connected table. The section under the yellow rectangular box shows the actual data (fields and rows). In this notes, we will represent Tableau fields using square brackets, [ ].

We will now switch to the Worksheet. We get:

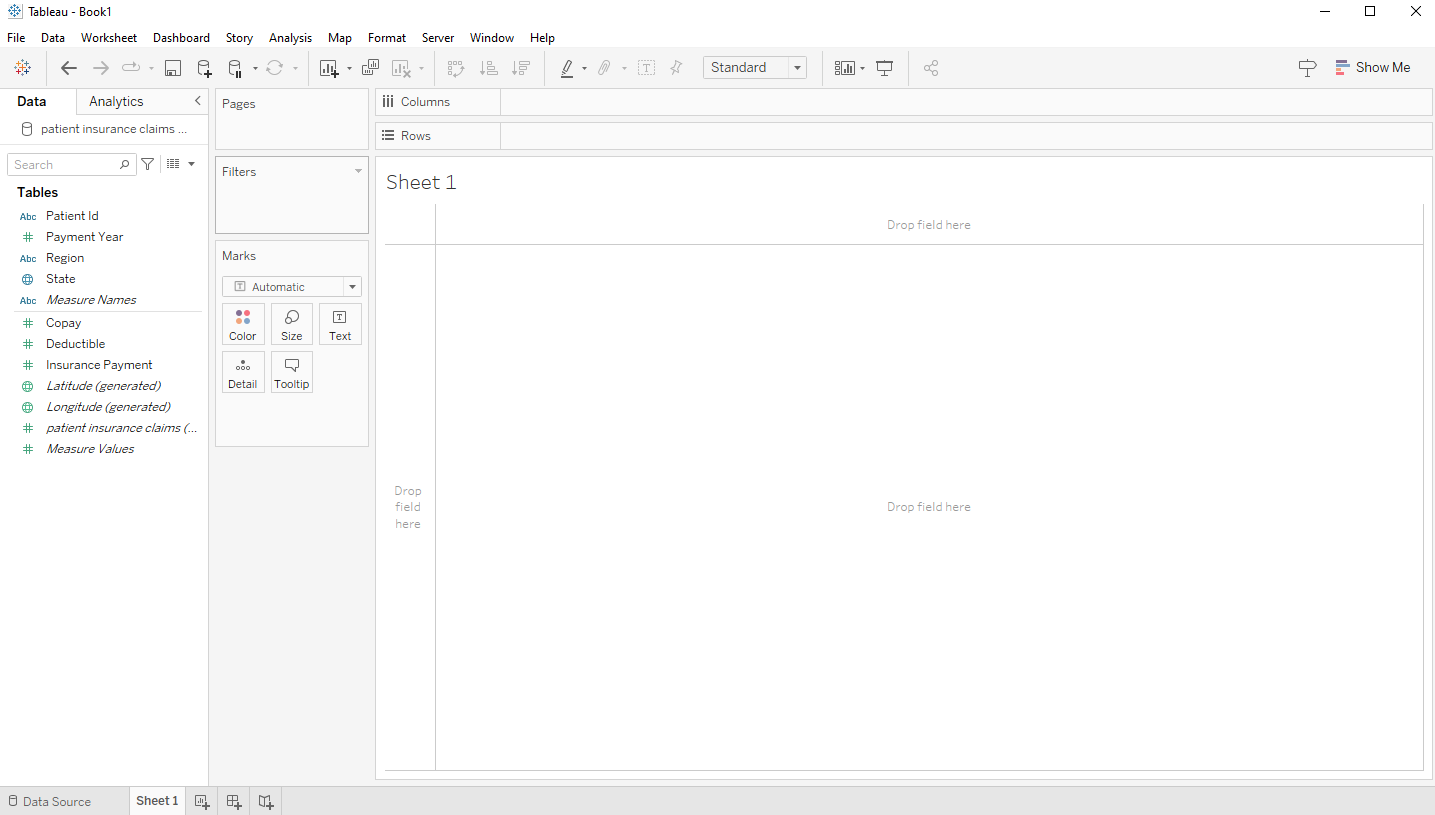


Figure : Screenshot of an empty worksheet

On the leftmost side under Data, we see the Tables and all the field names for the table. That section is known as Data Pane. The Pages, Filters, Columns and Rows are known as shelves, and the space where graphics are generated is known as viz or worksheet canvas. The Data Pane shows the names of columns in the connected dataset/table. The Pages shelf is used to create multiple visualizations with one visualization per page. Filters shelf is used to filter the data based on dimensions or measures. Columns and Rows give the columns and rows, respectively, for tables or other graphics. The values of a field are displayed as columns or rows depending on which shelf is used. For example, if we drag the [Latitude (generated)] to Rows and [Longitude (generated)] to Columns, we get the world map.

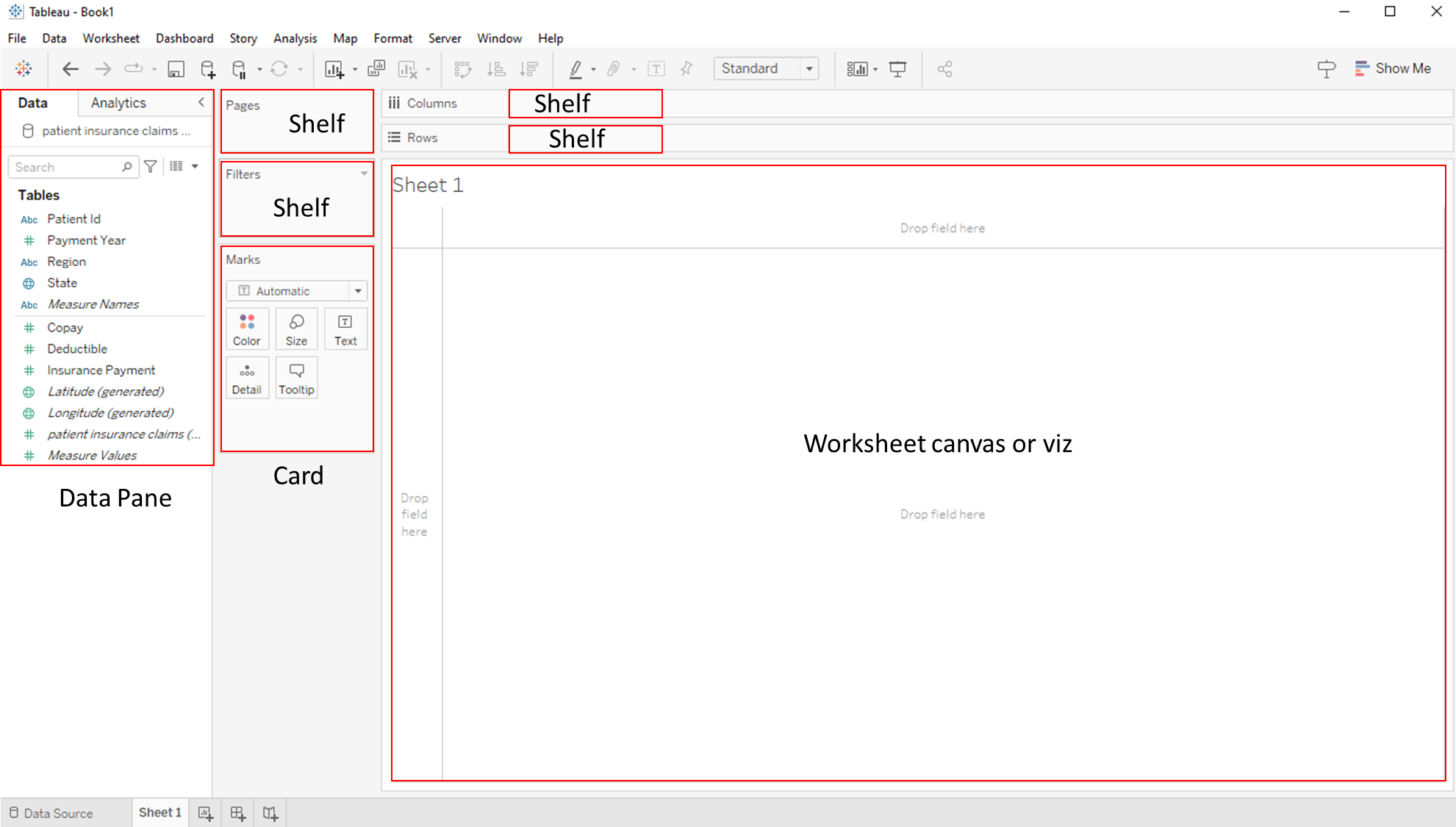


Figure : Worksheet with different section labelled

The fields/columns shown on the Data Pane are divided into two groups, separated by the thin horizontal line. The columns above the line are recognized by Tableau as Dimensions and the ones below are considered Measures. We can move columns from Dimensions to Measures and the other way round but depending on the context the visualizations may not be appropriate. Dimensions are the column type that are used for slicing and dicing the data; they are what the data are aggregated upon. Measures are the columns that are aggregated and that are used for the visualizations. We will look into the column types in detail now.

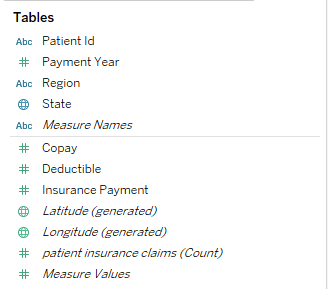


Figure : Columns in the connected dataset

First, we see that there are different icons before the column names. These icons represent the type of the column: Abc means the column is of text/string type; # means the column is a number, the globe icon means the column is of geographic type. There are also other types such as Date, Date and Time, and Boolean.

There were 7 fields/columns in the dataset but here we see that Tableau is showing 12 fields. Tableau internally generates new fields that are useful for creating geographic visualizations, visualizations using multiple measures, or for data validation. These Tableau-generated fields are italicized by default. The [patient insurance claims (Count)] is a Tableau-generated field that is a counter for each row. In the above dataset, we can imagine this count field/column as:

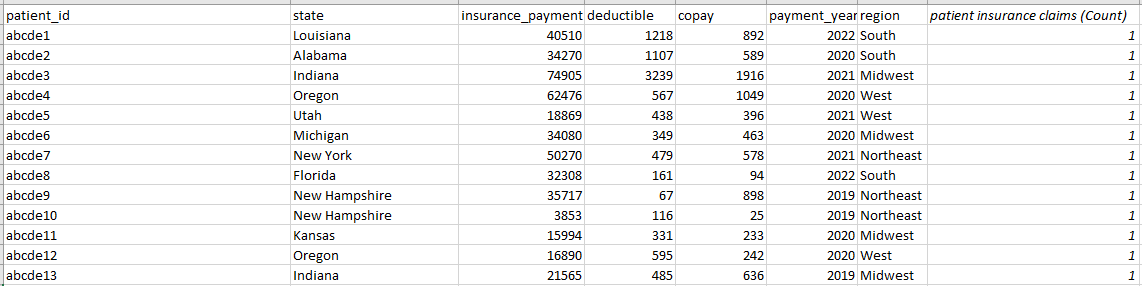


Figure : Screenshot of the dataset with Tableau-generated Count field shown

This count field/column can be used to check the number of rows overall, or the number of rows by some dimensions. Also, it can be used to create visualizations where the number of rows is the measure of interest.

[Latitude (generated)] and [Longitude (generated)] are the columns that hold geographic information. When the dataset has a geographic column, these are generated. In our example above, the State column is recognized by Tableau as a geographic column, so those two columns are generated to aid in visualization using maps.

[Measure Names] and [Measure Values] hold all the measures in the connected dataset. For example, in the dataset above, the measure names are [Copay], [Deductible], [Insurance Payment], and [patient insurance claims (Count)]. We can think of this as Tableau creating an internal dataset/table as:

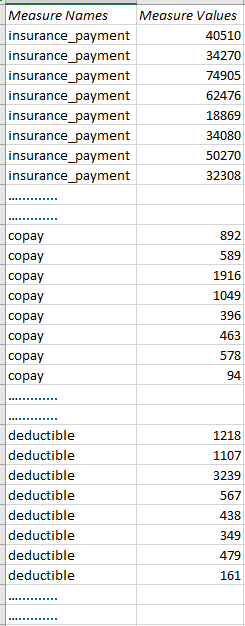


Figure : Screenshot of a dataset that can be thought of as how Tableau stores Measure Names and Measure Values

[Measure Names] and [Measure Values] are important when we need multiple measures in our graphic. What they internally do is create a copy of all measures in the dataset in a long format (instead of their likely wide format in the dataset).

# Measures and Dimensions Vs Discrete and Continuous

In addition to Measures and Dimensions, Tableau also divides fields into discrete and continuous. Discrete fields are those which have discrete values i.e., the values observed for a discrete field are distinct. Continuous fields, by contrast, have continuous values i.e., all values observed in the data are in a continuum and values falling in that continuum but not observed are also considered legitimate values. For example, if a field has the values of 1, 3 and 5 and is discrete, the field is considered to have three values and values such as 2 and 4 are not considered as plausible values. If, however, the field is continuous, then values 2 and 4 are considered possible values the field could take but was the case that they were not observed in the particular data we had. Discrete fields are identified by blue color in the field type icon, while continuous fields have green color. In figure 5, patient id and region are discrete text fields, while state is a discrete geographic field. Likewise, copay, deductible and insurance payment are continuous numeric fields. Measure Names is a discrete text field, while Measure Values is a continuous numeric field.

It does look like dimensions are discrete and measures are continuous. However, not all dimensions are discrete: for example, a continuous date (day-level field) is a dimension, as we aggregate measures based on some details of date (months, years). Similarly, age in years could be both discrete and continuous measure depending on how we want to use it in our visualization.

In the above example, we will convert payment year field to a discrete date type.

# Making Basic Graphics

We make graphics on Tableau using Rows/Columns and Marks card. When we drag a discrete dimension to Columns and Rows, we get headers (or the values of the dimension).

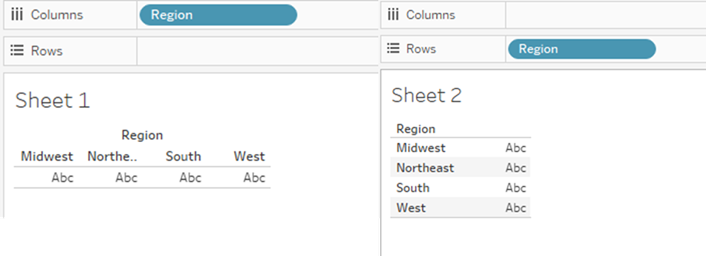


Figure : A discrete dimension on Columns and Rows

Since we only added dimension and no measure, the tables do not have any values (hence the “Abc”). The [Region] on Columns/Rows is known as pill (for discrete fields: blue pills; for continuous fields: green pills). When we drag and drop a continuous field (either dimension in the form of date, or measure), an axis is created.

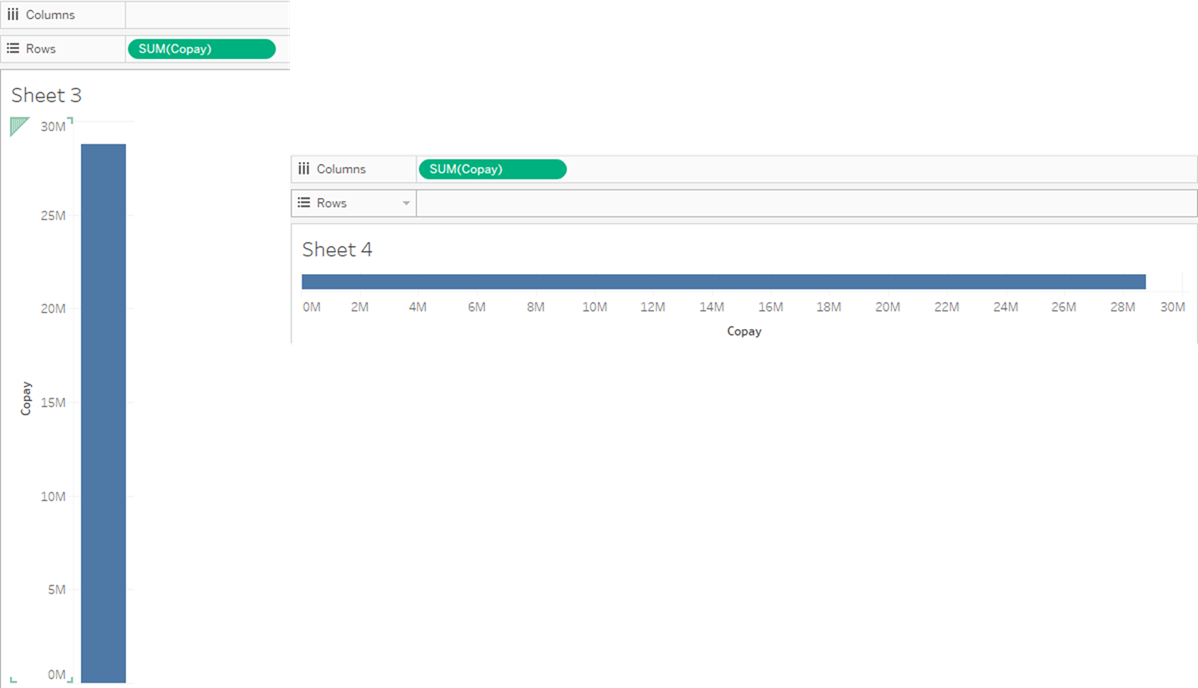


Figure : A continuous measure on Rows and Columns

In the figure 9, the [Copay] is aggregated as total (SUM) for the entire dataset and shown as a bar as a row or column. The default is SUM in this case; we can change the default aggregation or change it after we drop the measure in one of the shelves or Marks card.

Now, I will drag-and-drop [Region] to Columns and [Copay] (defaults to SUM(Copay)) to Rows. I get separate bars for SUM(Copay) for each Region type. This is essentially a bar chart. When a dimension ([Region], in this case) is dropped on Columns or Rows, the data is divided by the Region and the measures are aggregated by the dimension type.

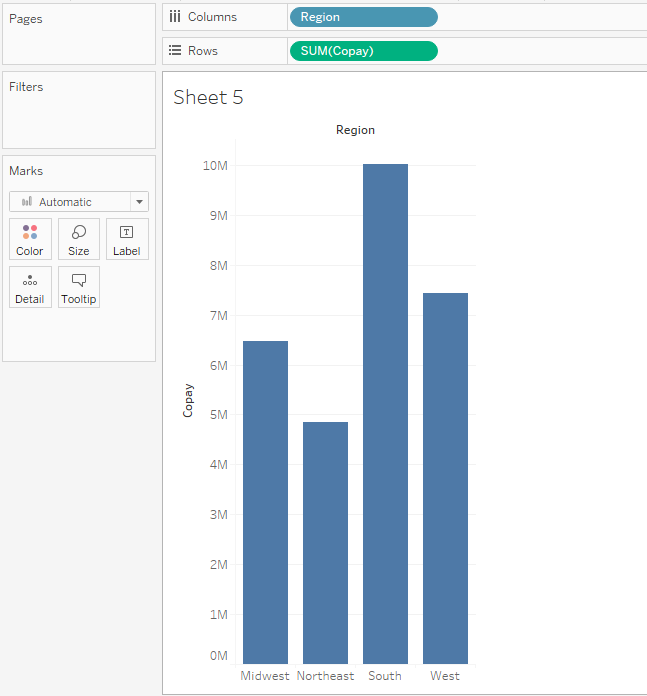


Figure : Bar chart showing total copay by Region

Instead of dragging and dropping, we could double-click on the dimension or measure, and Tableau would give you a response based on what it assumes you wanted. When I double-click [Copay] in the above example, it is added to Rows shelf. When I double-click [Region] with [Copay] already on Rows, it is added to Columns, resulting in the identical bar chart as that shown in figure 10.

Now, we will convert copay from continuous to discrete. We can do that by clicking on the field and selecting “Convert to Discrete”.

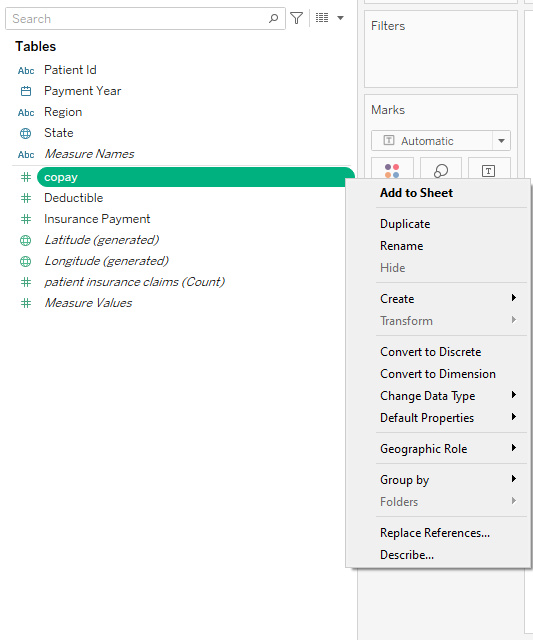


Figure : Converting a measure from continuous to discrete

Now when we drop [Region] and [Copay] to Rows and/or Columns, we get:

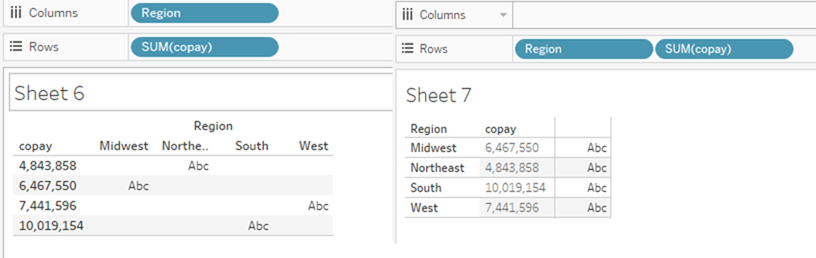


Figure : Table with two discrete fields (one dimension and one measure)

As we can see, headers or values are listed for discrete fields, whether they are dimensions or measures. For the dimension, Tableau expects values to follow, so the “Abc” place-holder is seen after [Region]. We will convert [copay] back to continuous for other visualizations below.

# Tables

We saw how a basic bar chart is made in Tableau. We can also make tables. To reiterate an important point, when a continuous field is dropped on Rows/Columns, an axis is generated, which can be used for different graphics such as bar charts, line graphs and maps, among others. To create a table, however, we don’t want an axis. Here, we can avail of options available in the Marks card, one of which is Text/Label. When we drop a field to Text, the values of the field is shown in the viz.

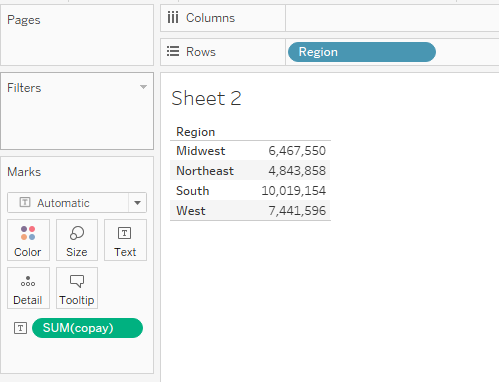


Figure : Table with total copay by Region

Similar to Rows/Columns, the fields with the aggregation dropped on Marks card are called pills (blue: discrete; green: continuous).

We can also have multiple dimensions in the table.

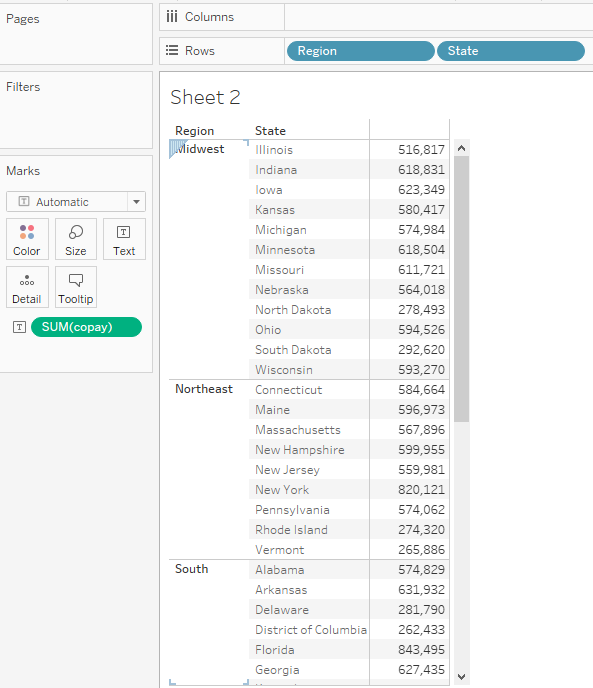


Figure : Table with total copay by Region and State

We can also have multiple measures in a single table. We can do this by dropping multiple measures on the Text/Label Marks card.

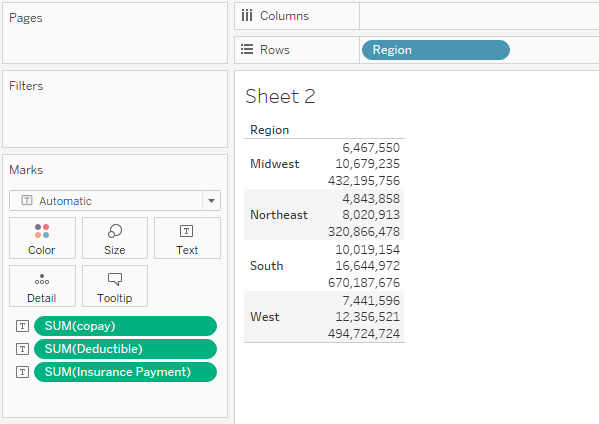


Figure : Table with total copay, total deductible and total insurance payment, by Region

One issue with this table is that it’s hard to tell which values are for which measures and aggregation. This is where we can use [Measure Names] and [Measures Values]. When we drag and drop [Measure Names] to Rows and [Measure Values] to Text/Label, we get:

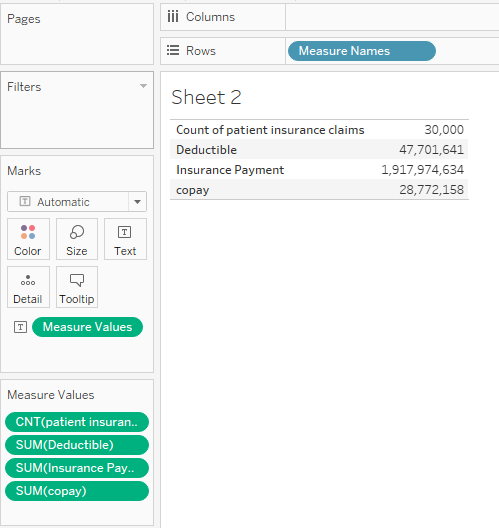


Figure : Table with Measure Names and Measure Values

This table shows the names of Measures and the default aggregations. We can use the underlying approach here to improve our table in figure 15. To do that, first we will create a table with [Region] and SUM([copay]). Then, instead of dropping on Text we will drop in the viz, like:

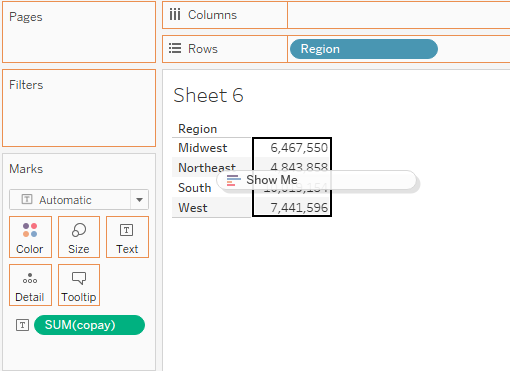


Figure : Dropping another measure to an existing table

This gives us a table of the following format:

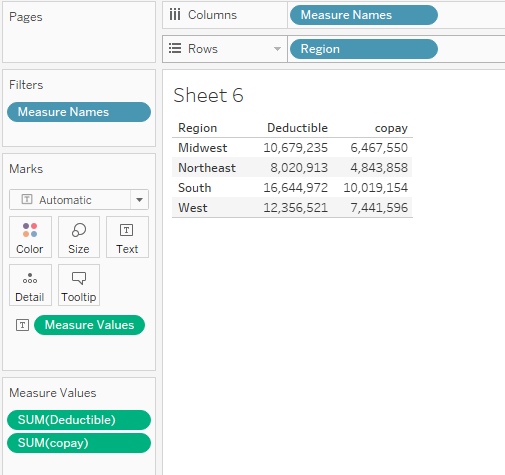


Figure : Table with total copay and total deductible, by Region

The way this table is formed warrants a bit deeper dive. First, [Region] is on Rows, similar to the previous table. Second, there is [Measure Names] on Columns, which results in the names of Measures being displayed in the table. Then, there is [Measure Names] on Filters. Since we had [copay] in our table and we added [Deductible] the [Measure Names] on Filters filter out other Measures in the dataset ([Count of patient insurance claims] and [Insurance Payment]). A new card for [Measure Values] shows up, which shows the aggregation and fields included in the table.

We can drop another measure in the table.

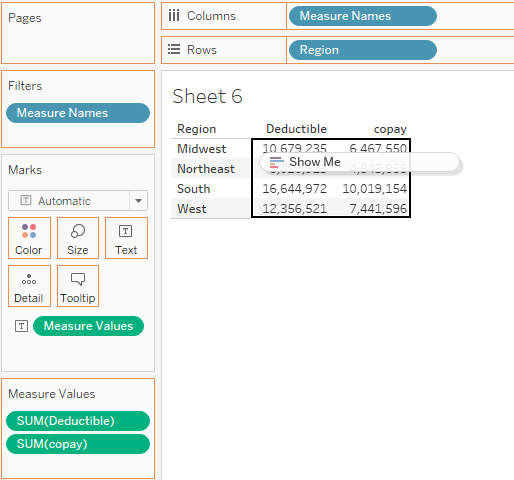


Figure : Adding Insurance Payment to a table with copay and deductible by Region

This results in the following table.

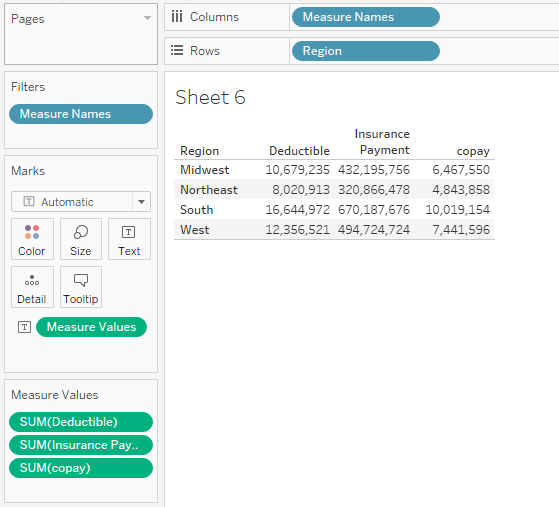


Figure : Table with total copay, total deductible and total insurance payment, by Region

# Marks card in detail

We will go back to the bar chart of total copay by [Region] (figure 10). The Marks card shows Color, Size, Text/Label and Tooltip. When we click on the drop-down under Marks, we can see:

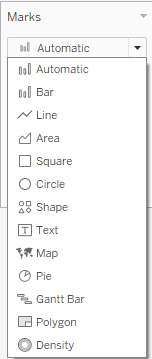


Figure : Drop-down choices under Marks card

The Automatic shows what Tableau displays by default, which is based on the types of fields dropped on Rows/Columns and Marks card.

The Color option is used to change color of the graphic. For the bar chart in figure 10, we can change the color of the bars.

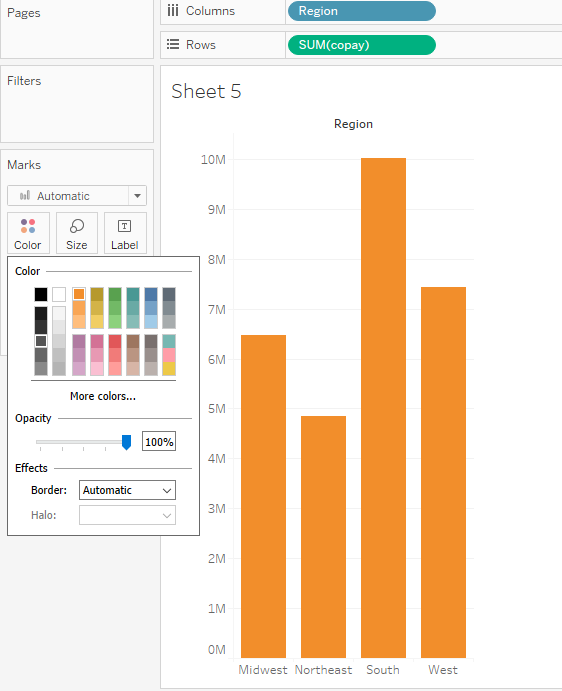


Figure : Bar chart for total copay by Region (orange color for bars)

Using the Size Marks card option, we can change the size of the bars.

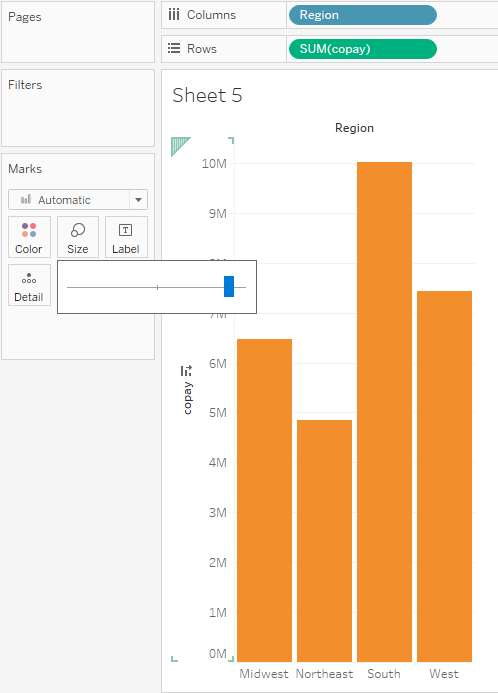


Figure : Bar chart for total copay by Region (wider bars)

With the Text/Label Marks card, we can add the values to the bars.

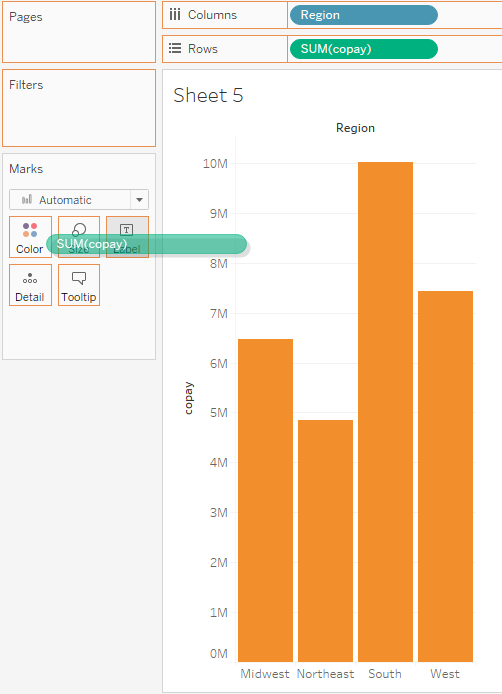


Figure : Bar chart with total copay being dropped to Text/Label

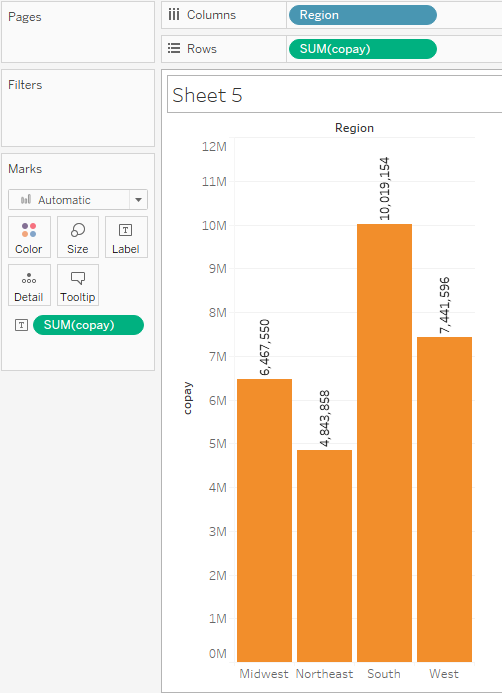


Figure : Bar chart of total copay by Region (with values added to bars)

Tooltip Marks card is used to customize information that is displayed when we hover on the bars. For example, in the figure below, we hovered on South in the bar chart.

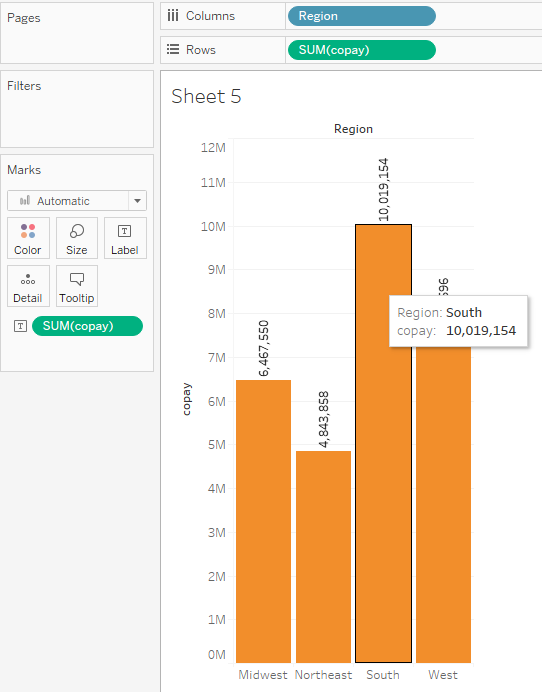


Figure : Bar chart of total copay by Region (with tooltip shown)

We can add total of [Insurance Payment] to Tooltip. The bar chart will still display total [copay] by [Region] but will also show total [Insurance Payment] for the respective [Region] when we hover over one of the regions.

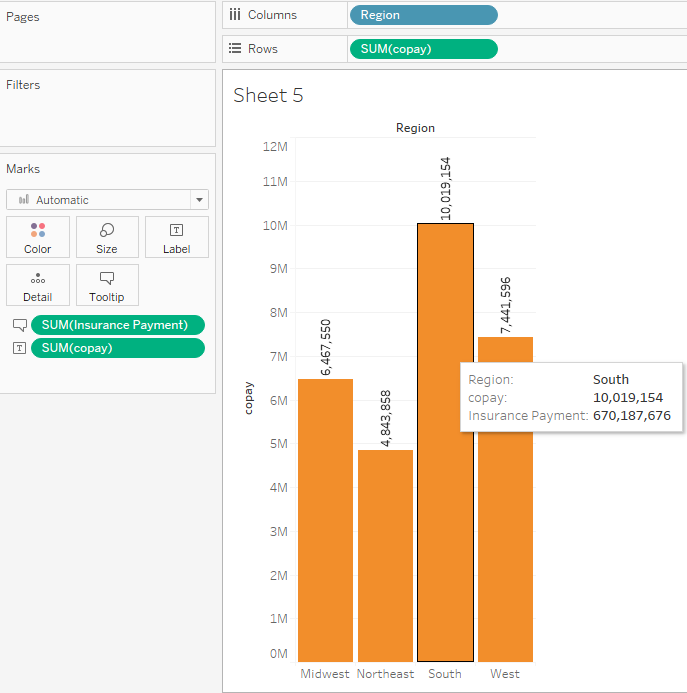


Figure : Bar chart of total copay by Region (with total insurance payment by Region on Tooltip)

The Detail Marks card adds the level of detail or granularity at which the graphic will be displayed. When we move Region from Columns to Detail, we get:

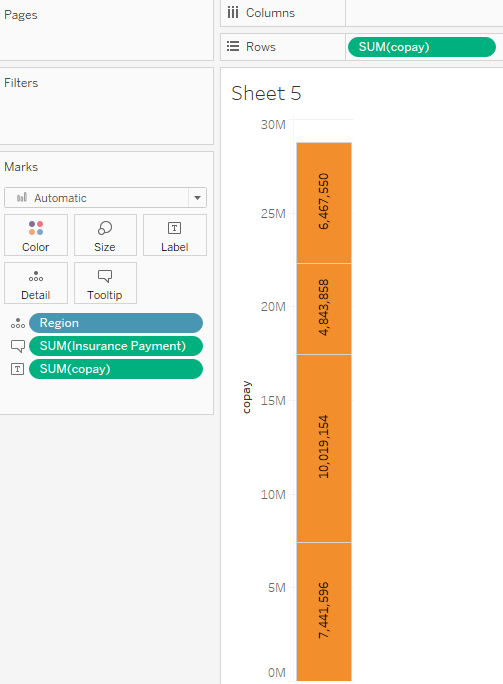


Figure : Bar chart for total copay with Region on Detail

It’s difficult to tell from the bar which value/bar pertains to which Region. To make that clear, we can drag [Region] to Label/Text as well. When we drag [Region] from Detail and drop it on Label/Text, it gets removed from the Detail. That’s not what we want; we want [Region] on both Detail and Label/Text. To do this, we can drag and drop [Region] from the data pane, or we can press *cntrl* on keyboard and drag from the Detail (I use Windows, so for Mac users, I believe it would be *cmd* + drag and drop).

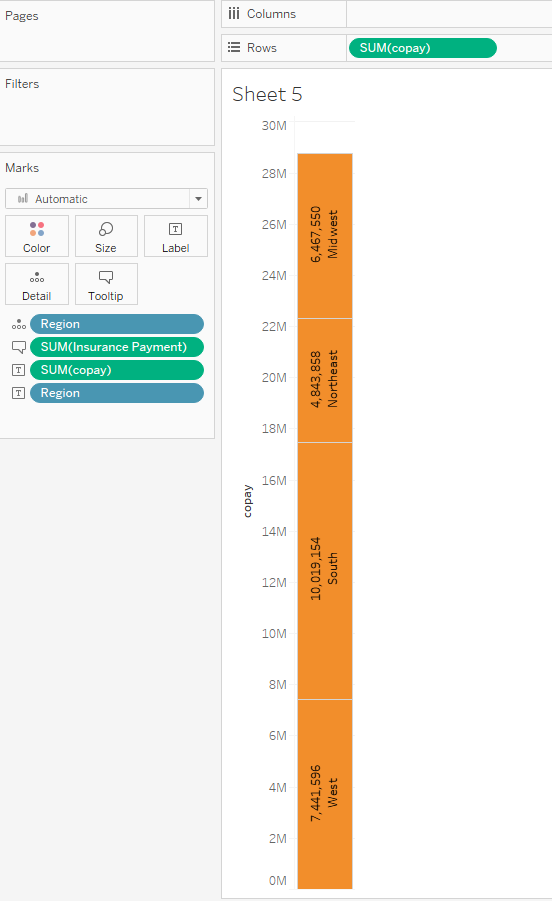


Figure : Bar chart with Region on both Detail and Label/Text

Instead of moving [Region] to Detail, we will leave [Region] on Columns and drag [Payment Year] to Detail. We get:

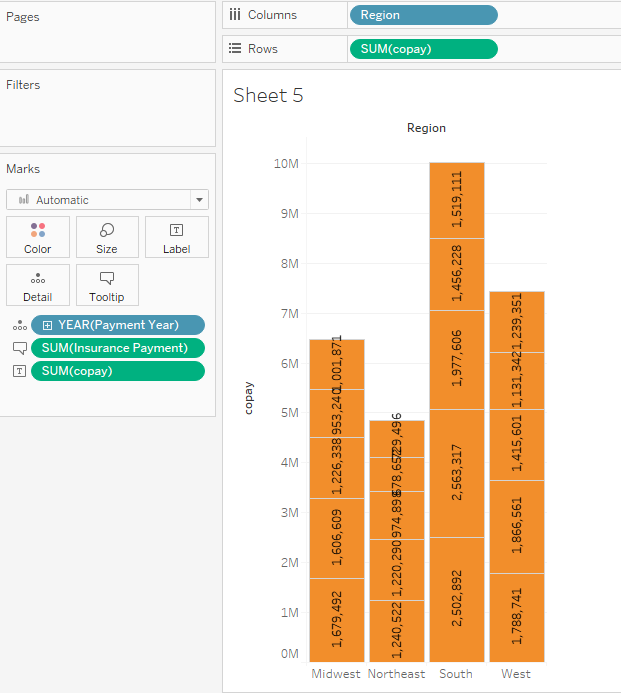


Figure : Bar chart of total copay by Payment year and Region

As you can see, the visualization is divided by year of [Payment Year] in addition to [Region]. Here again we can add YEAR[Payment Year] on Label/Text to clearly identify which portion of bar reflects which year.

We will switch to the regular bar chart of total [copay] by [Region]. Using the Marks card options, we can switch from a bar chart to a dot plot. To do this, we can click on the drop-down menu in the Marks card and select circle. We get:

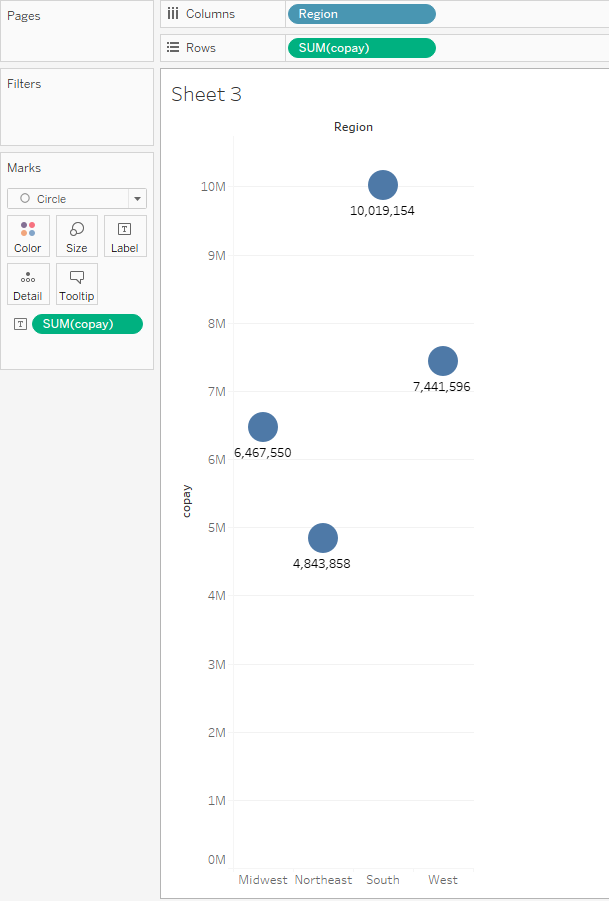


Figure : Dot plot of total copay by Region

Instead of using circles for dot plots, we could select Shapes from the drop-down menu and pick different shapes for the total [copay] value.

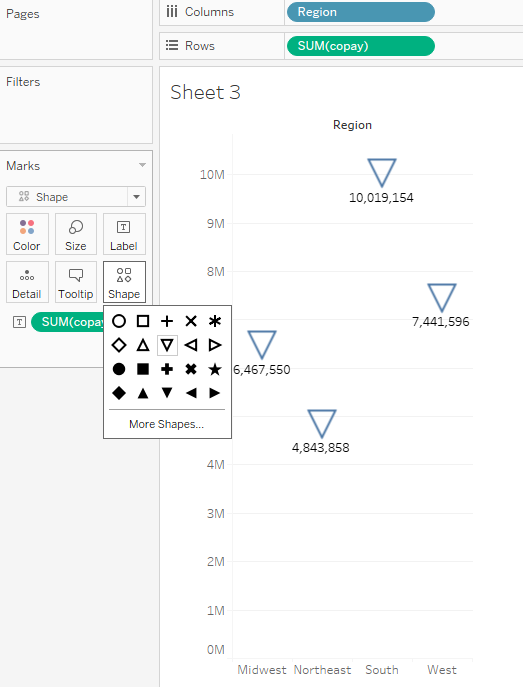


Figure : Variation of dot plot using different shapes