Lab 2. Data Manipulation

In this lab, we'll cover the following topics:

- Joining data sources
- Adding a secondary data source
- Data blending
- Data union
- Using Tableau Pivot
- Preparing data

Introduction

Before doing any work in Tableau, we must first connect to the data we'll be working on. In the first recipe in this book, [*Connecting to data*], we learned how to connect to a data source. In this lab, we'll build on that knowledge to become more skillful at manipulating, joining, unioning, and transforming data sources. In the last recipe of this lab, [*Preparing data*], we'll also address the topic of data preparation and getting our data ready for visualizing and further analysis.

In this lab, we'll be using multiple datasets. In the first recipe, [*Joining data sources*], we'll be using two datasets, `Public_Schools_1.csv` and `Public_Schools_2.csv`, which contain data on Boston public schools for the school year 2018/2019. The dataset originally comes from Kaggle's official site. In the[*Adding a secondary data source*] and [*Data blending*] recipes, we'll use the `Internet_satisfaction.csv` and `Internet_usage.csv` datasets, which describe the results of a consumer survey on internet usage in Serbia, as well as data on internet penetration per region of the country. In the [*Data union*] recipe, we'll be using `Bread_basket_by_year.xlsx` (originally found on Kaggle.com), which contains data about transactions in a bakery divided into two tables holding data for different years. In the[*Using Tableau Pivot *]recipe, we'll use `Internet_satisfaction_by_region.csv`, which holds some results of a customer satisfaction survey found in the `Internet_satisfaction.csv` file, but organized in a slightly different format. Finally, in the[*Preparing data*] recipe, we'll use the `Winery.csv` dataset (originally found on Kaggle.com), containing data on wines, their origin, pricing, and rating.

Joining data sources

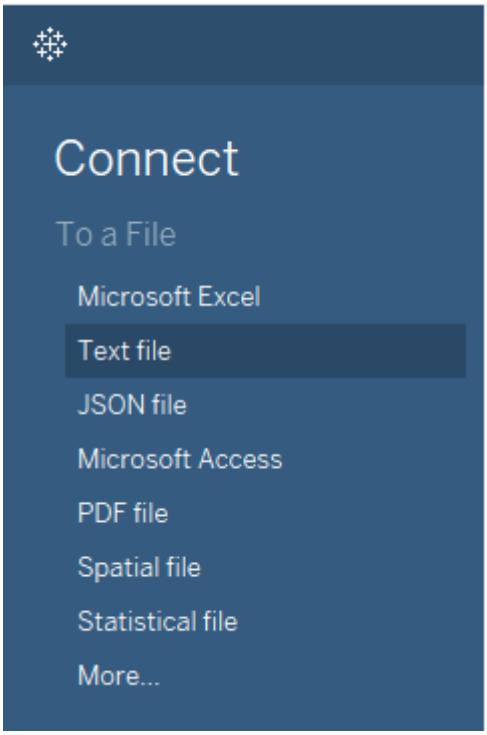
The data we are working in is often contained in multiple tables. We can create a single, virtual table out of multiple original tables by joining them using common fields or keys. The result is a wider table that contains columns originating from different tables. The rows are matched by the values of the column, key, and fields.

Getting ready

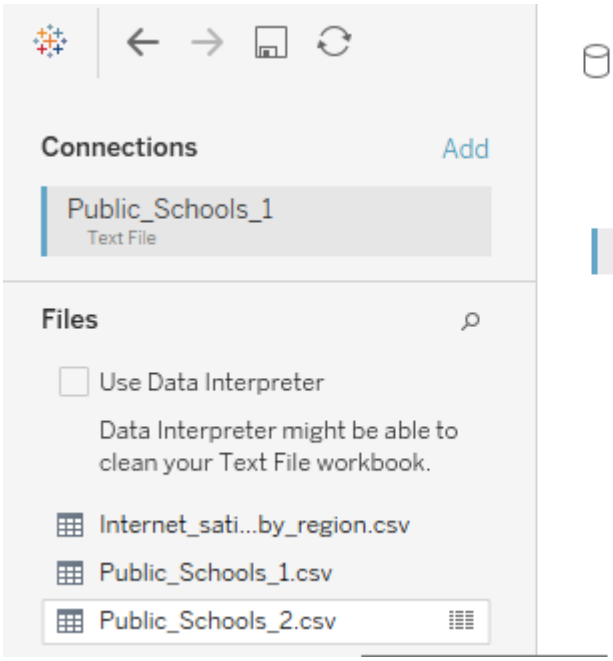
In this lab, we'll be using two datasets, `Public_Schools_1.csv` and `Public_Schools_2.csv`. Make sure you have both datasets saved to your device.

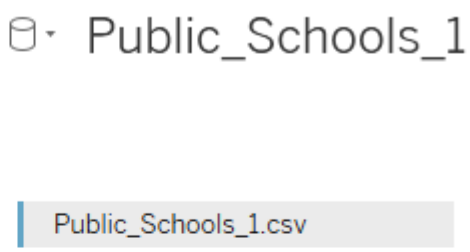
How to do it...

1. Upon opening Tableau, from the **`Connect`** pane on the left-hand side, choose **`Text file`** as shown in the following screenshot:

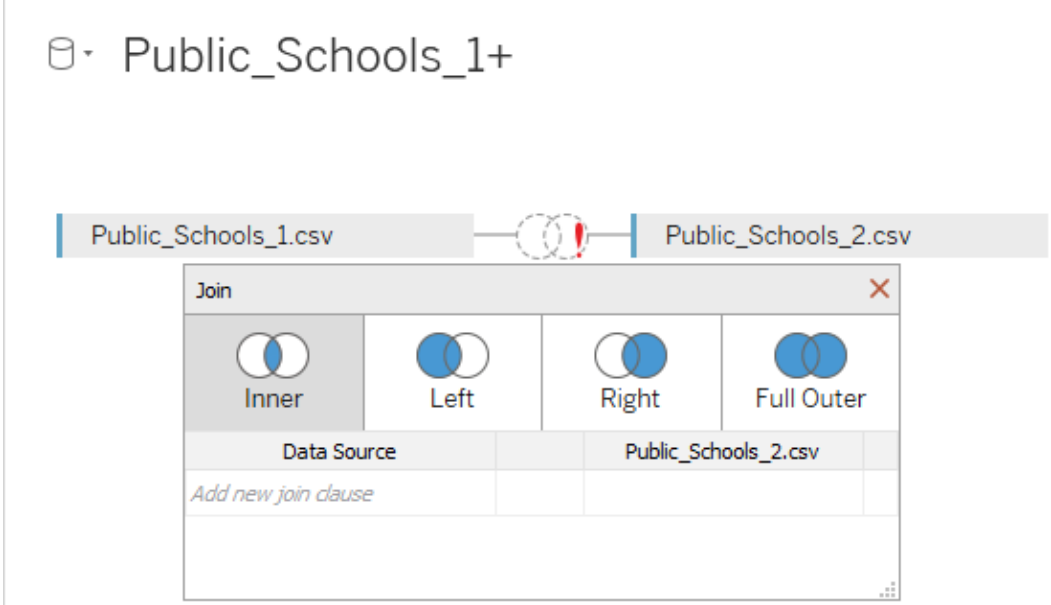


2. When the **`Open`** window opens, navigate to your local copy of `Public_Schools_1.csv`, select it, and click on **`Open`**.
3. Tableau will take you to the **`Data Source`** page, showing that you've successfully connected to the chosen data source. On the left-hand side of the page, under **`Files`**, all of the text files in the same folder are listed, and among them is `Public_Schools_2.csv`:

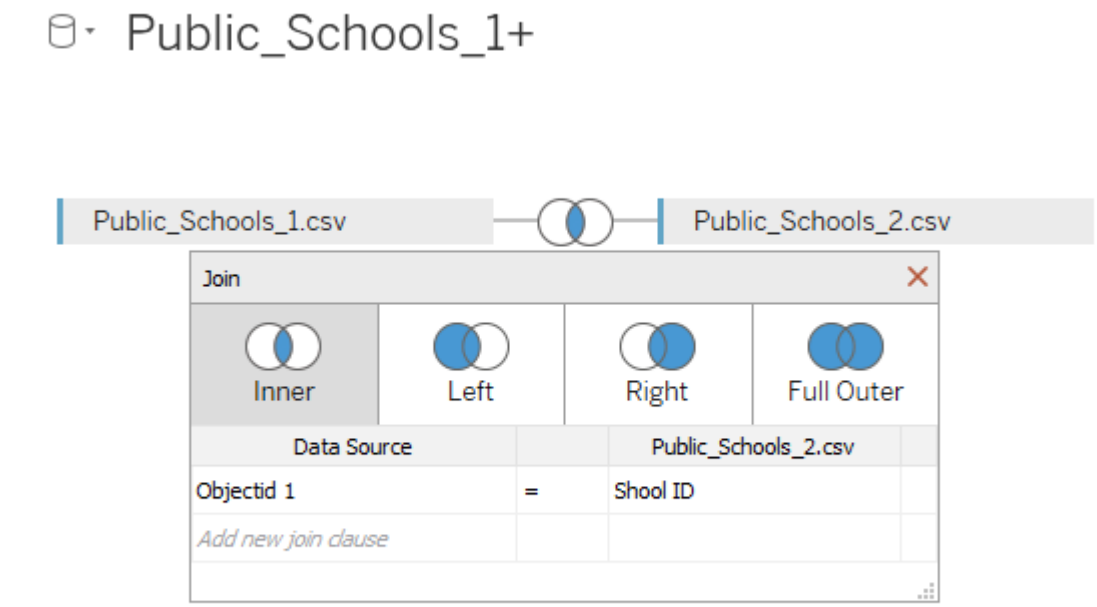




4. Drag and drop `Public_Schools_2.csv` from **`Files`** to the whitespace next to `Public_Schools_1.csv`:

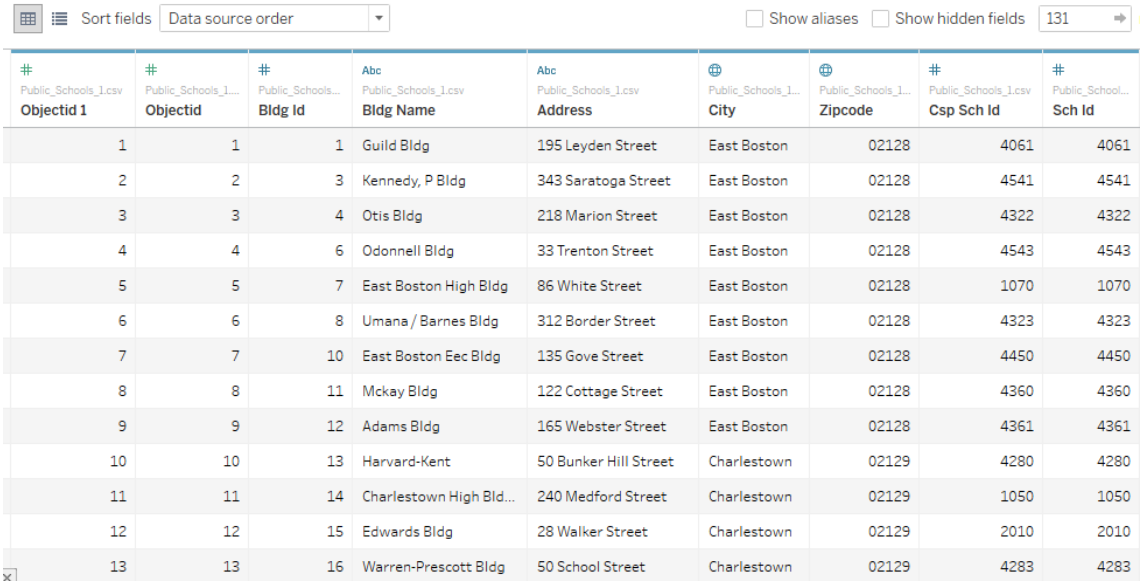


5. In the `Join` window that opens, click on the `Add new join clause` drop-down menu under `Data Source` and select `Objectid 1`.
6. Click on the white field on the right-hand side of the `=` sign, under `Public_Schools_2.csv`.
7. From the drop-down menu, select `School ID`:



8. Click on `X` to close the `Join` window. You have successfully joined the data, and you can inspect the new joined

table in the preview, which is shown in the following screenshot:



Sort fields | Data source order | Show aliases | Show hidden fields | 131

Objectid 1	Objectid	Bldg Id	Bldg Name	Address	City	Zipcode	Csp Sch Id	Sch Id
1	1	1	Guild Bldg	195 Leyden Street	East Boston	02128	4061	4061
2	2	3	Kennedy, P Bldg	343 Saratoga Street	East Boston	02128	4541	4541
3	3	4	Otis Bldg	218 Marion Street	East Boston	02128	4322	4322
4	4	6	Odonnell Bldg	33 Trenton Street	East Boston	02128	4543	4543
5	5	7	East Boston High Bldg	86 White Street	East Boston	02128	1070	1070
6	6	8	Umana / Barnes Bldg	312 Border Street	East Boston	02128	4323	4323
7	7	10	East Boston Eec Bldg	135 Gove Street	East Boston	02128	4450	4450
8	8	11	Mckay Bldg	122 Cottage Street	East Boston	02128	4360	4360
9	9	12	Adams Bldg	165 Webster Street	East Boston	02128	4361	4361
10	10	13	Harvard-Kent	50 Bunker Hill Street	Charlestown	02129	4280	4280
11	11	14	Charlestown High Bld...	240 Medford Street	Charlestown	02129	1050	1050
12	12	15	Edwards Bldg	28 Walker Street	Charlestown	02129	2010	2010
13	13	16	Warren-Prescott Bldg	50 School Street	Charlestown	02129	4283	4283

How it works...

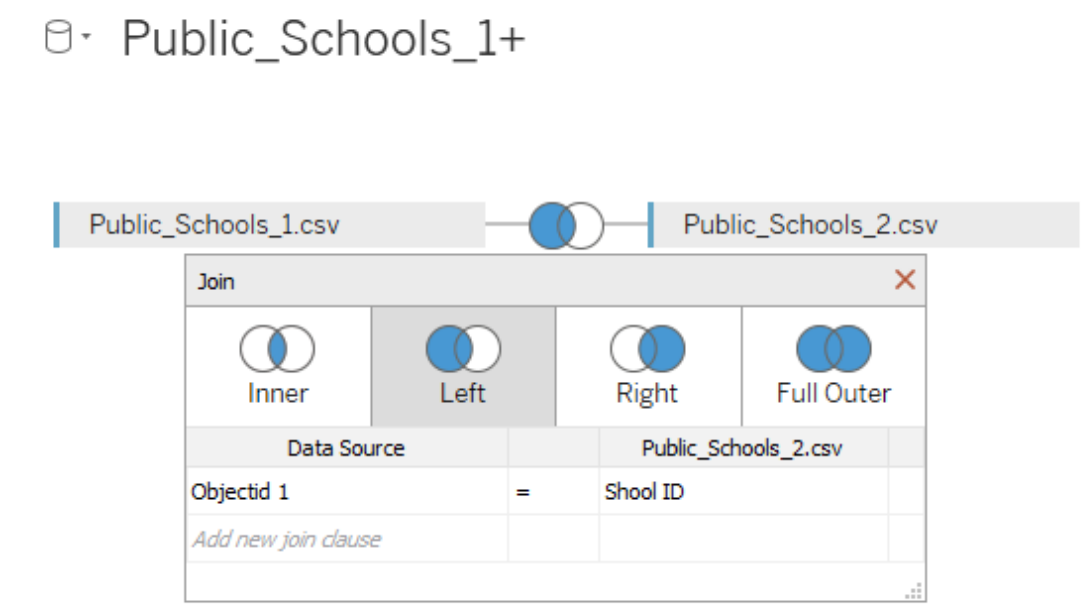
In the preceding steps, we joined two tables, `Public_Schools_1` and `Public_Schools_2`. We matched the rows in the two tables using a unique key. In the **`Join`** window, we specified that the key fields in the two datasets are **`Objectid 1`** and **`School ID`**, respectively. These two fields contain exact same values, which allows Tableau to create one-on-one mapping between the tables. However, they have different names, which is why we need to specify them manually. If their names were the same, Tableau would automatically make the connection.

There's more...

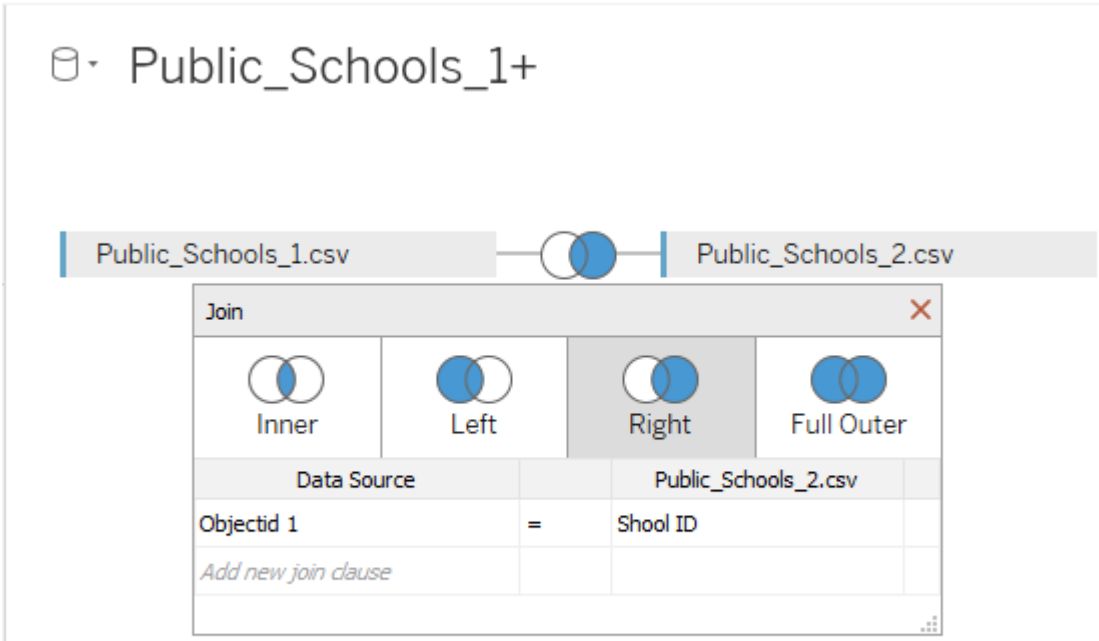
What we performed is an **`Inner`** join of the two tables. There are also other joins, such as **`Left`**, **`Right`**, and **`Full Outer`**, offered in Tableau. They're generally available to switch between, in the **`Join`** window.

Using an inner join was appropriate in our case, because all of the rows in both tables could be matched---thanks to a unique key field. The inner joins produce tables with cases that have been matched in both tables.

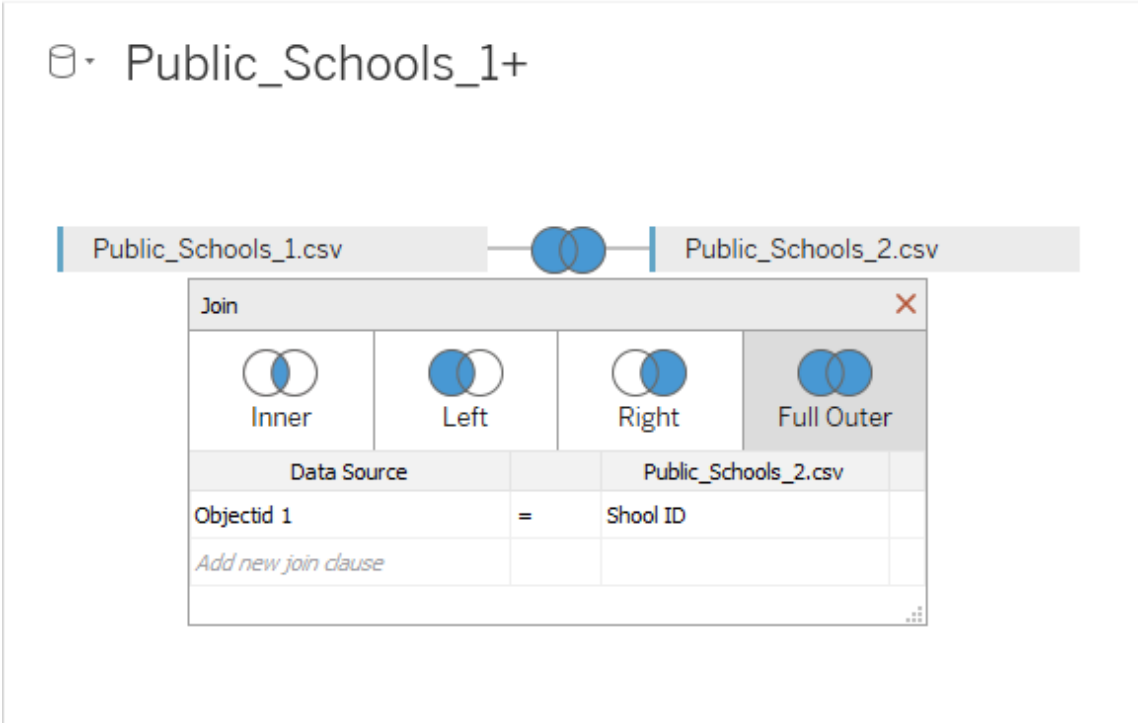
The **`Left`** join keeps all of the values from the left tables, while using only the cases from the right table that have a match in the left table, as shown in the following screenshot:



The `Right` join works on the same principle, but keeping all of the values from the right table, as shown in the following screenshot:



Finally, a `Full Outer` join keeps all of the cases from both tables, regardless of whether they have a match in the other table or not. Null values will be placed in cases where a match isn't found:



As of the 2018.2 release, Tableau also offers another kind of joins--- `Spatial` joins. They allow you to join points and polygons from spatial tables on the basis of their location. This is achieved through a new joining predicate known as [**intersect**], which matches the location of points from one table to polygons in another table, joining data when a point lies within a polygon.

Adding a secondary data source

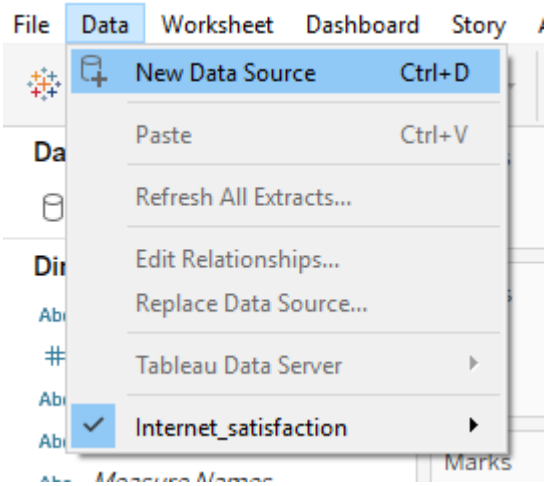
When creating a workbook, or a dashboard, we often have multiple data sources that're relevant for the topic and want to include the data from all of them. However, we might not want to join them---we just want to include all of the visualizations that come from unrelated data sources in the same workbook or dashboard. We can easily do that by simply adding data sources to our workbook. Let's see how.

Getting ready

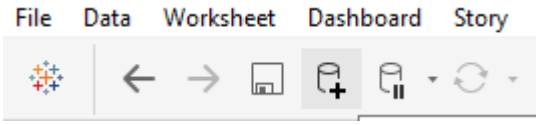
In this lab, we'll be using two datasets, `Internet_satisfaction.csv` and `Internet_usage.csv`, as data sources. Make sure you have them both saved to your device.

How to do it...

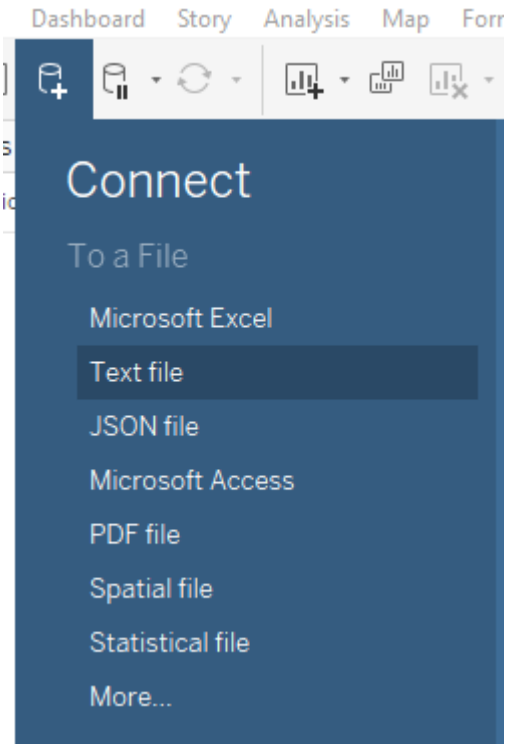
1. Upon opening Tableau, from the `Connect` pane on the left-hand side, choose `Text file`.
2. When the `Open` window opens, navigate to your local copy of `Internet_satisfaction.csv`, select it, and click `Open`.
3. Tableau will take you to the `Data Source` page, where you can preview your file. Click on the `Sheet 1` tab to open a new blank worksheet.
4. In the new blank worksheet, at the top of the `Data` pane, notice that `Internet_satisfaction` is your data source. Now, let's add a secondary data source. In the main menu toolbar, click on `Data`.
5. From the drop-down menu, select `New Data Source`:



6. Alternatively, click on the new data source icon:

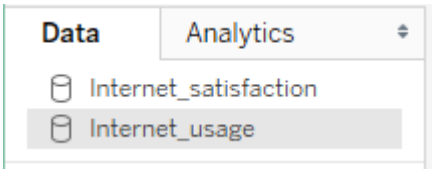


7. A **`Connect`** drop-down menu will open. From it, select **`Text file`**:



8. In the **`Open`** window that opens, navigate to your copy of `Internet_usage.csv`, select it, and click on **`Open`**.

9. Once again, the **`Data Source`** pane will open, this time previewing the `Internet_usage.csv` data source. Click on the ** `S``hee` **t 1 tab again.
10. Notice that **`Internet_usage`** has now appeared in the top of the **`Data`** pane alongside **`Internet_satisfaction`**:

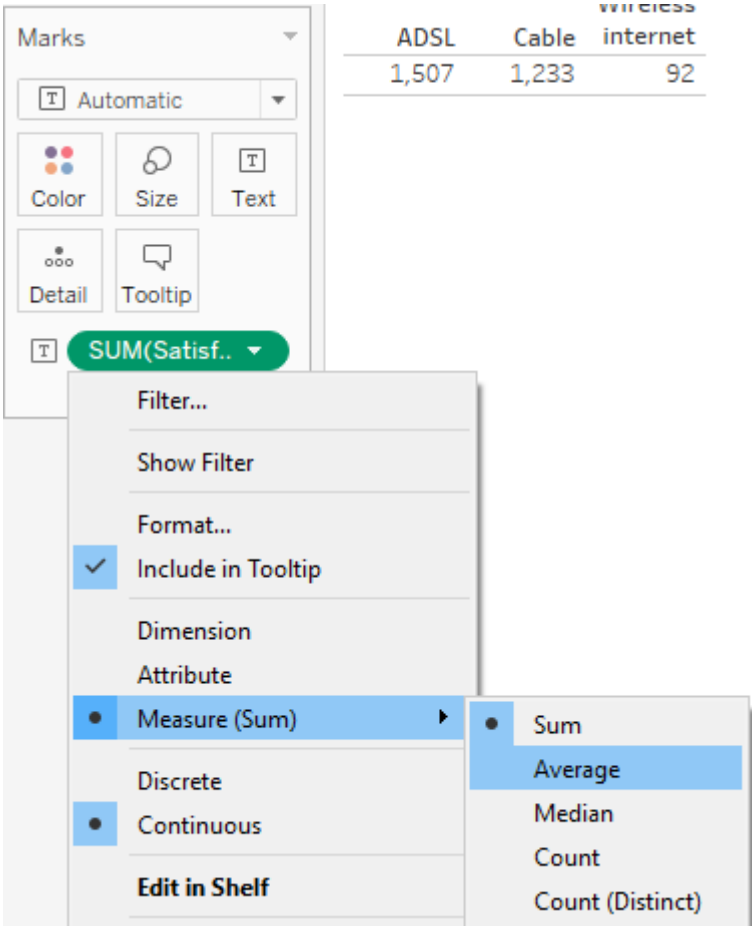


11. Try clicking on the data sources to switch between them. Notice how measures and dimensions change when you select a different data source.

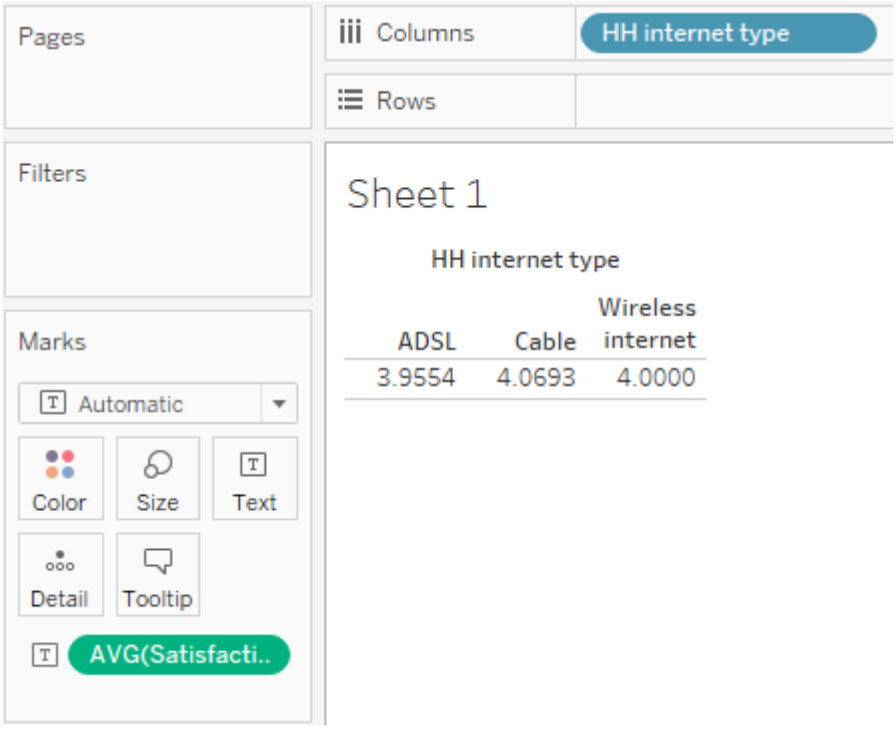
How it works...

Once we've connected to two data sources, we can use them both to create views. Let's try it:

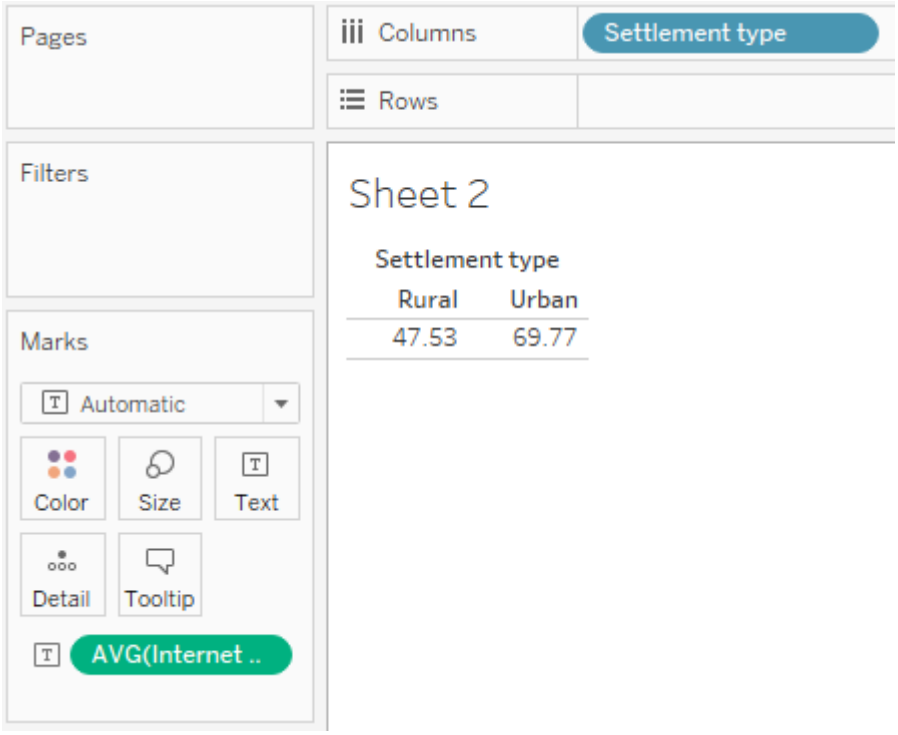
1. In the top of the **`Data`** pane, select **`Internet_satisfaction`** as the active data source by clicking on it.
2. Drag and drop **`HH internet type`** from **`Dimensions`** into the **`Columns`** shelf.
3. Drag and drop **`Satisfaction overall`** from **`Measures`** onto **`Text`** in the **`Marks`** card.
4. Right-click on the **`SUM(Satisfaction overall)`** pill in the **`Marks`** card, navigate to **`Measure (Sum)`**,** **and from the drop-down menu, select **`Average`**:



5. We've created a view using the `Internet_satisfaction` data source:



6. Now, open a new blank worksheet by clicking on the New Worksheet tab at the bottom of the workspace.
7. In the top of the **`Data`** pane, select **`Internet_usage`** as the active data source by clicking on it.
8. Drag and drop **`Settlement type`** from **`Dimensions`** into the **`Columns`** shelf.
9. Drag and drop **`Internet``penetration`** from **`Measures`** onto **`Text`** in the **`Marks`** card.
10. Right-click on the **`SUM(Internet penetration)`** pill in the **`Marks`** card, navigate to **`Measure (Sum)`** and from the drop-down menu select **`Average`**:



Here, we've created two tables that contain data and fields from two different data sources, as shown in the preceding screenshot.

There's more...

It's possible to connect to more than two data sources. However, be careful to use them in separate worksheets to prevent data blending. If you try using two data sources in the same worksheet, Tableau will automatically try to blend them. We will talk more about data blending in the next recipe, [*Data blending*].

Data blending

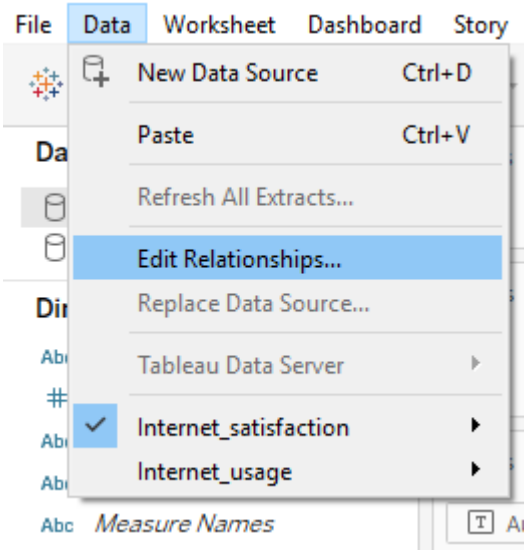
Sometimes, our data sources might not be suitable for joining for multiple reasons. For example, they might be on different levels of aggregation, resulting in duplicates upon joining, or you might wish to use data source types that don't support cross-database joins, such as Google Analytics. But, we also want to make visualizations that contain fields from different data sources. That's when data blending comes in handy---it allows us to make a connection between data sources when joining isn't appropriate or possible.

Getting ready

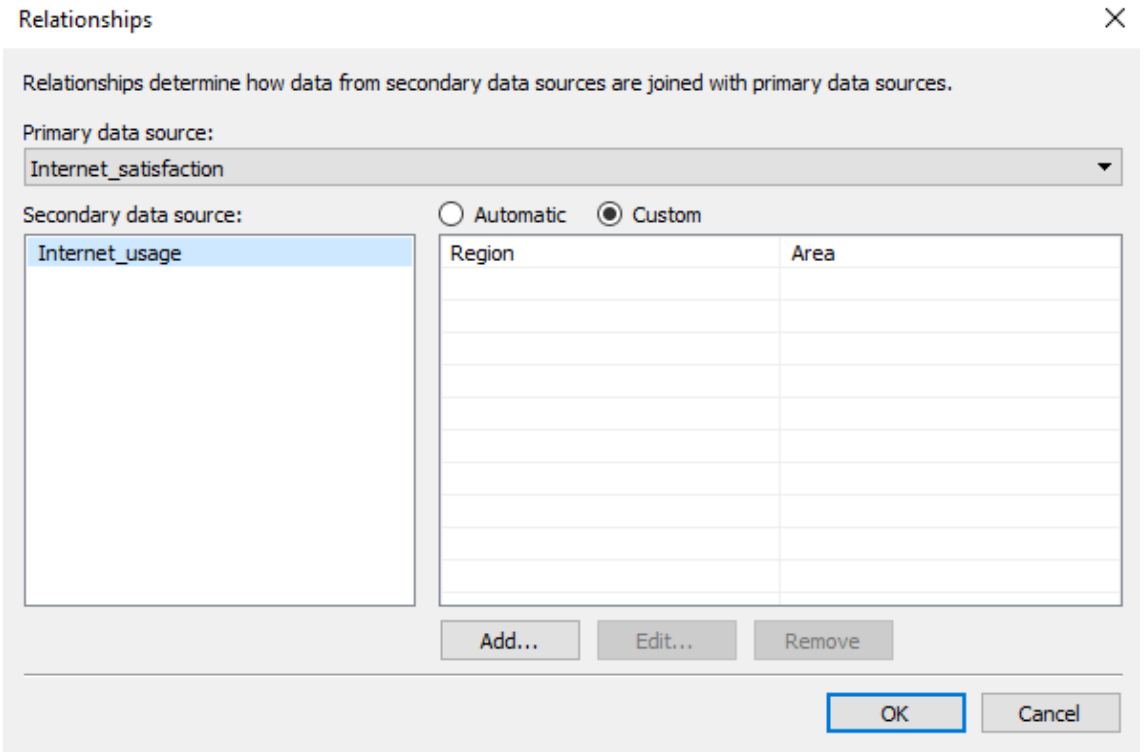
Follow the steps from the previous recipe, [*Adding a secondary data source*], to connect to the `Internet_satisfaction.csv` and `Internet_usage.csv` data sources.

How to do it...

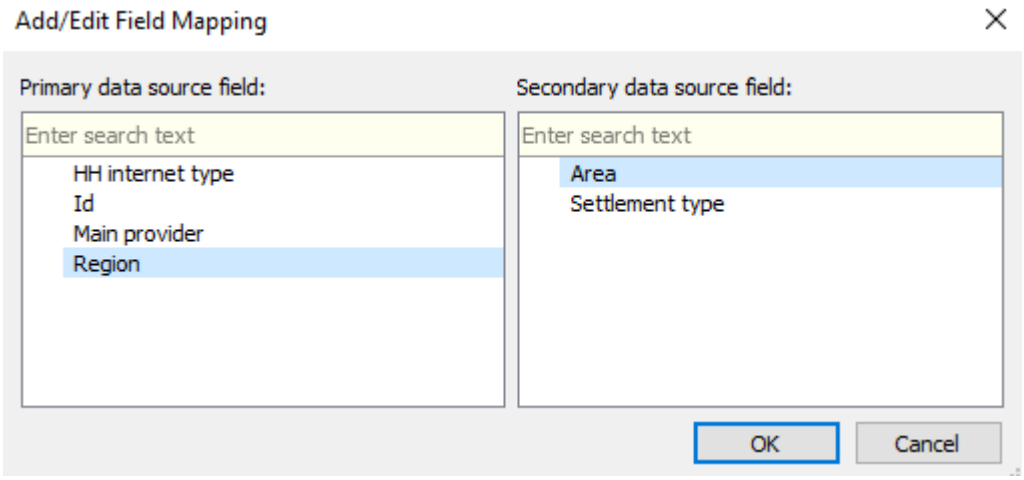
1. We start off connected to two data sources: `Internet_satisfaction.csv` and `Internet_usage.csv` and a new open blank worksheet opened.
2. In the main menu toolbar, navigate to **`Data`**.
3. From the drop-down menu, select **`Edit Relationships`**...:



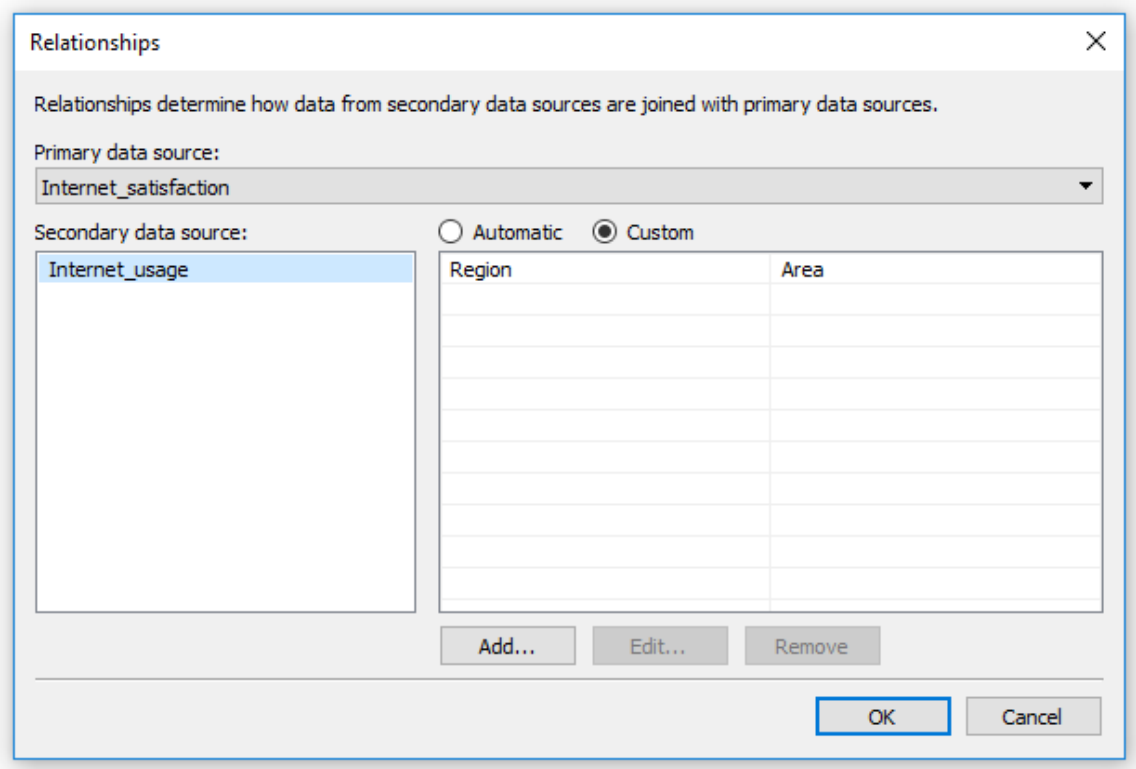
4. In the **`Relationships`** window, select **`Custom`**.
5. Click on **`Add`**... :



6. In the **`Add/Edit Field Mapping`** window, click on **`Area`** in one pane and **`Region`** in the other pane, so that they're highlighted in blue:



7. Click on **`OK`**.
8. Mapping of **`Region`** and **`Area`** will appear in the **`Relationships`** window now. Click on **`OK`** to exit it as well:

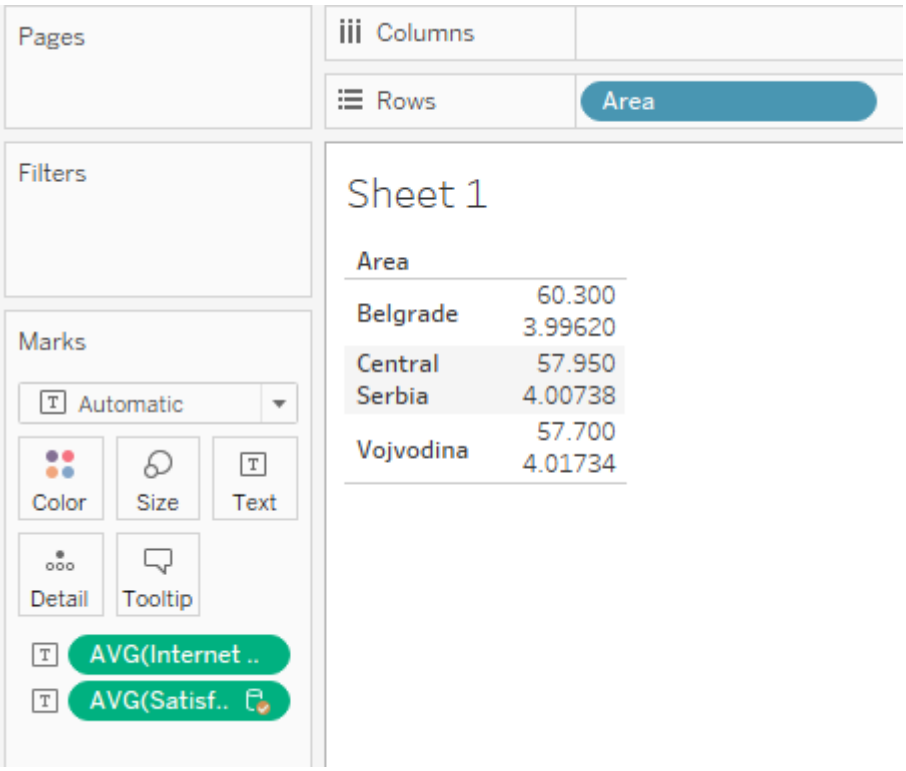


We've successfully blended our data sources!

How it works...

By blending the two data sources, we've instructed Tableau to treat **`Area`** from the `Internet_usage` data source and **`Region`** from the `Internet_satisfaction` data source as the same field. Now, we can create views that include fields from both data sources, despite them being on very different levels of detail and hence unsuitable for joining. To see how this works, let's create a new visualization that includes fields from both data sources:

1. On **Sheet 1**, in the **Data** pane, select **Internet_usage** as the active data source by clicking on it.
2. Drag and drop **Area** from **Dimensions** into the **Rows** shelf.
3. Drag and drop **Internet penetration** from **Measures** onto **Text** in the **Marks** card.
4. Right-click on the **Internet penetration** pill in the **Marks** card, navigate to **Measure (Sum)**, and from the drop-down menu, select **Average**.
5. In the **Data** pane, select **Internet_satisfaction** as the active data source by clicking on it.
6. Drag and drop **Satisfaction overall** from **Measures** onto **Text** in the **Marks** card.
7. Right-click on the **Satisfaction overall** pill in the **Marks** card, navigate to **Measure (Sum)**, and from the drop-down menu, select **Average**:



We've now created a single table showing internet penetration and overall satisfaction per region, with our measures coming from different data sources, as shown in the preceding screenshot.

There's more...

Note the orange link symbol

on the right-hand side of the **Region** field under **Dimensions**. It denotes that this is the linking field. You can easily stop using it as the linking field by clicking on the link symbol. It will change to a gray symbol

to signal that the link is broken.

If two fields in your data sources have the same name and the same values, Tableau will automatically blend your data sources using them. For example, if we renamed **Area** to **Region**, we wouldn't need to manually match these two fields---Tableau would automatically do it for us.

Note

Even if Tableau performed the data blending automatically, it's always a good idea to check whether the fields have been matched correctly by opening the **`Relationship`** window and inspecting existing relationships.

Data union

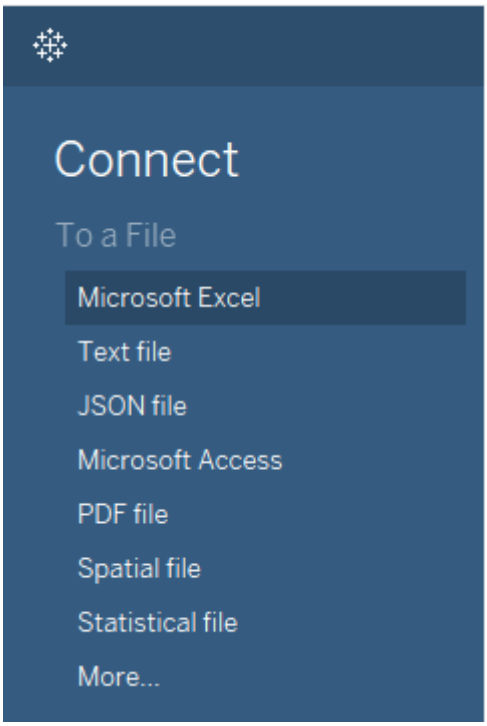
Tableau data union functionality allows us to merge multiple tables by appending the rows of one table to another.

Getting ready

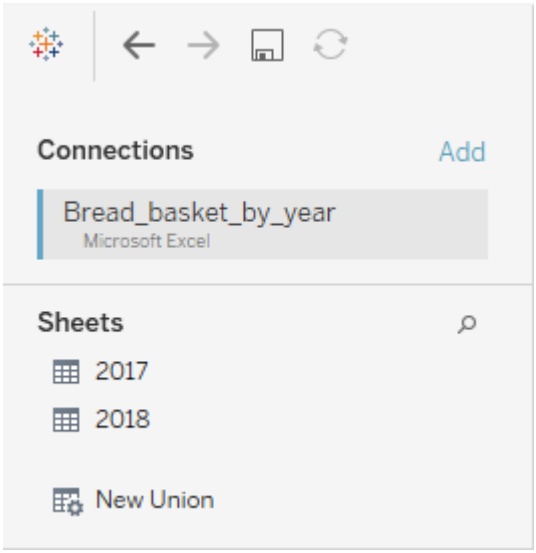
In this lab, we'll be using the `Bread_basket_by_year.xlsx` file, so make sure you have saved it to you device.

How to do it...

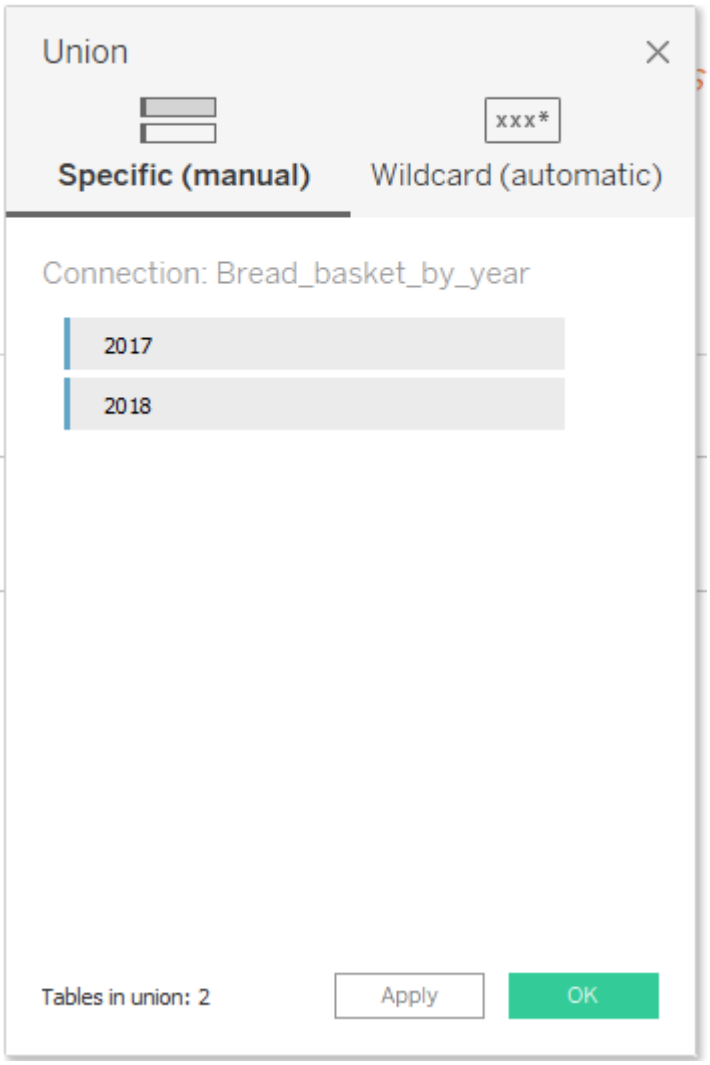
1. Upon opening Tableau, from the **`Connect`** pane on the left-hand side, choose **`Microsoft Excel`**:



2. When the **`Open`** window opens, navigate to your local copy of `Bread_basket_by_year.xlsx`, select it, and click **`Open`**.
3. On the **`Data Source`** page, two tables contained within our data source, **`2017`** and **`2018`**, appear. Beneath them, we see the **`New Union`** option:



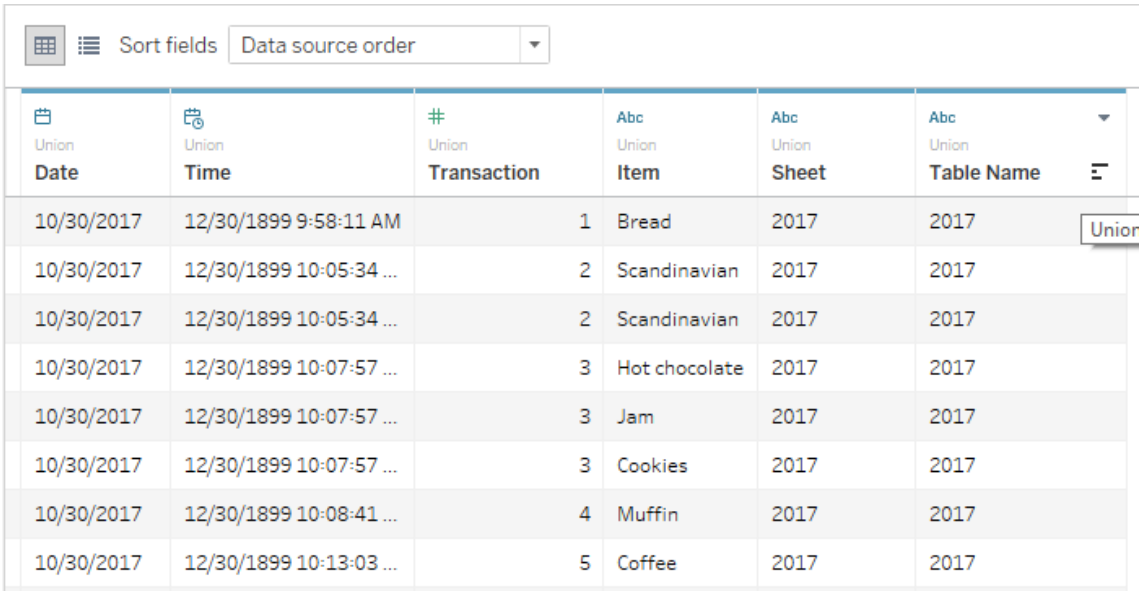
4. Drag and drop **`New Union`** onto the canvas.
5. A new **`Union`** window will open. Drag and drop **`2017`** from **`Sheets`** into the **`Union`** window as well, as shown in the following screenshot:



6. Click on **Apply** and then on **OK** **to exit the** **Union** window.

How it works...

We have successfully unioned the two tables. In the **Data Source** page, they're now previewed as a single table, with two new columns that don't exist in either of the datasets--- **Sheet** and **Table Name**. These two columns provide the metadata about the union, by denoting the source of the rows as shown in the following screenshot:



What actually happened is that the rows from both datasets were merged into one table. We can see that if we open a new blank worksheet and drag and drop **`Date`** from **`Dimensions`** into the **`Rows`** shelf, the new unioned data source contains years coming from both datasets. Our union was successful because the tables had the same number of columns and the same column names. This allowed Tableau to match columns correctly.

There's more...

It's also possible to union tables in Tableau using wildcard search. This means setting up search criteria to search for a string in tables' names, and letting Tableau automatically union the tables the names of which satisfy the specified criteria.

Using Tableau Pivot

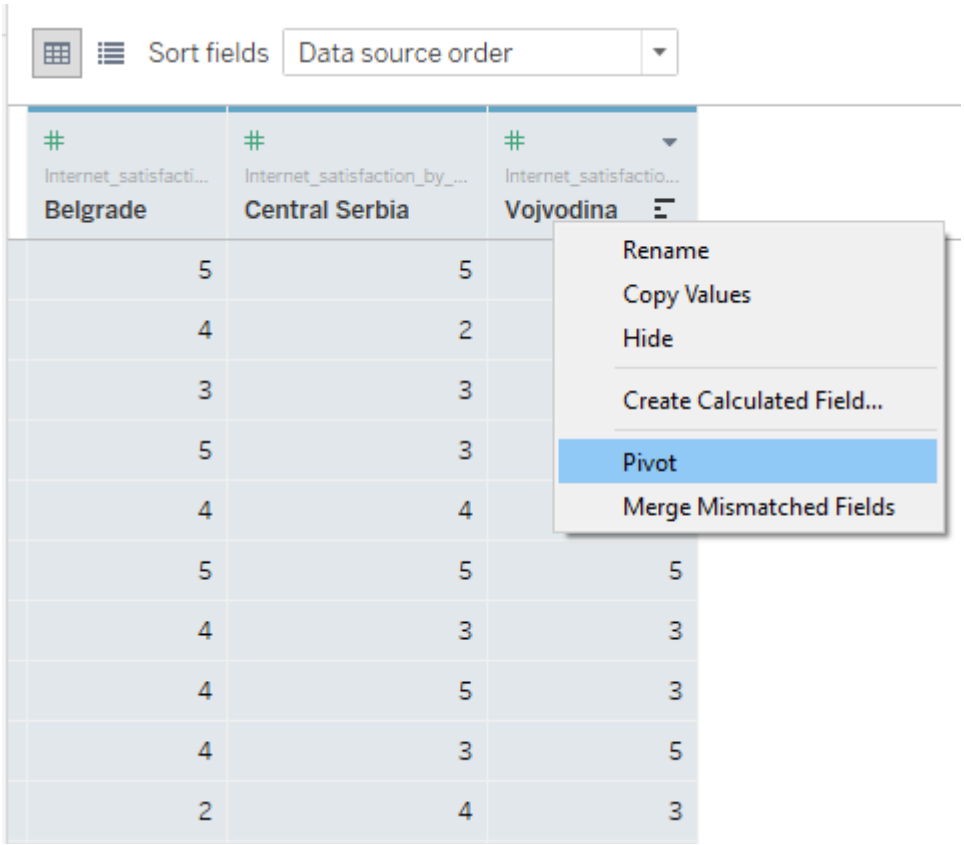
Sometimes, our data isn't organized in a format that's suitable for creating views we would like to produce. Tableau Pivot functionality offers an easy way to restructure our data into a format that might be more suited to our needs.

Getting ready

To perform the steps In this lab, we'll be using the `Internet_satisfaction_by_region.csv` file as the data source. Make sure you have it saved to your device and connect to it.

How to do it...

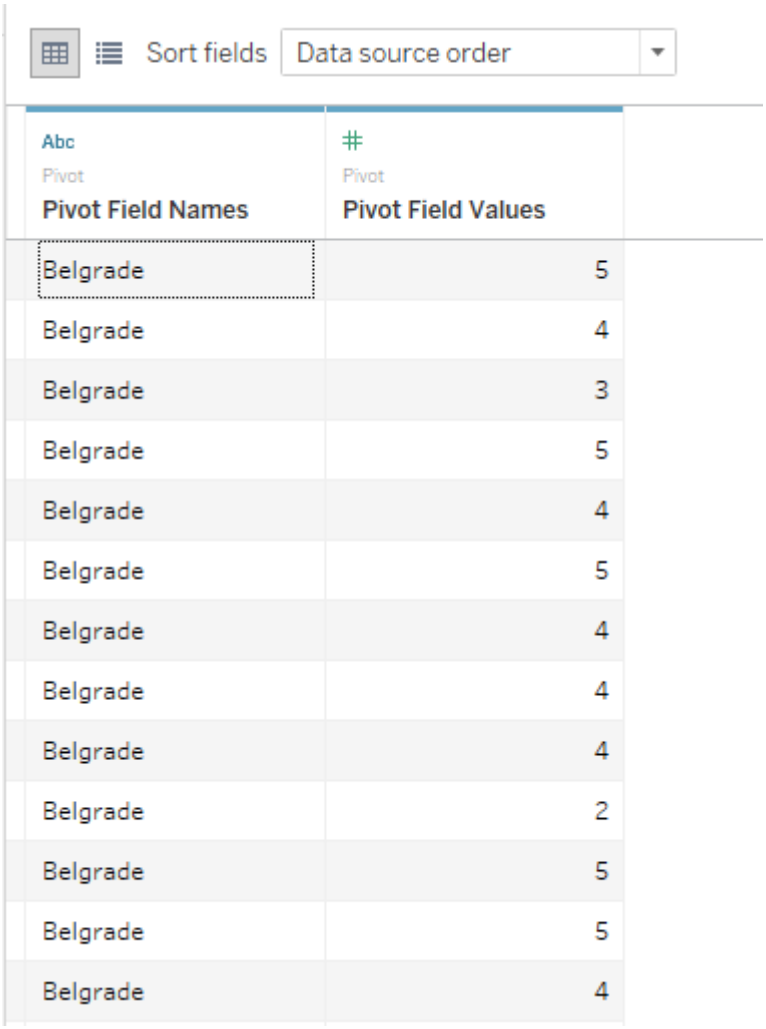
1. In the **`Data Source`** page, hold the [*Ctrl*] key and click on the header of all three fields in the data source preview to select them all.
2. Release the [*Ctrl*] key and right-click on any of the headers.
3. In the drop-drown menu, select **`Pivot,`** as shown in the following screenshot:



How it works...

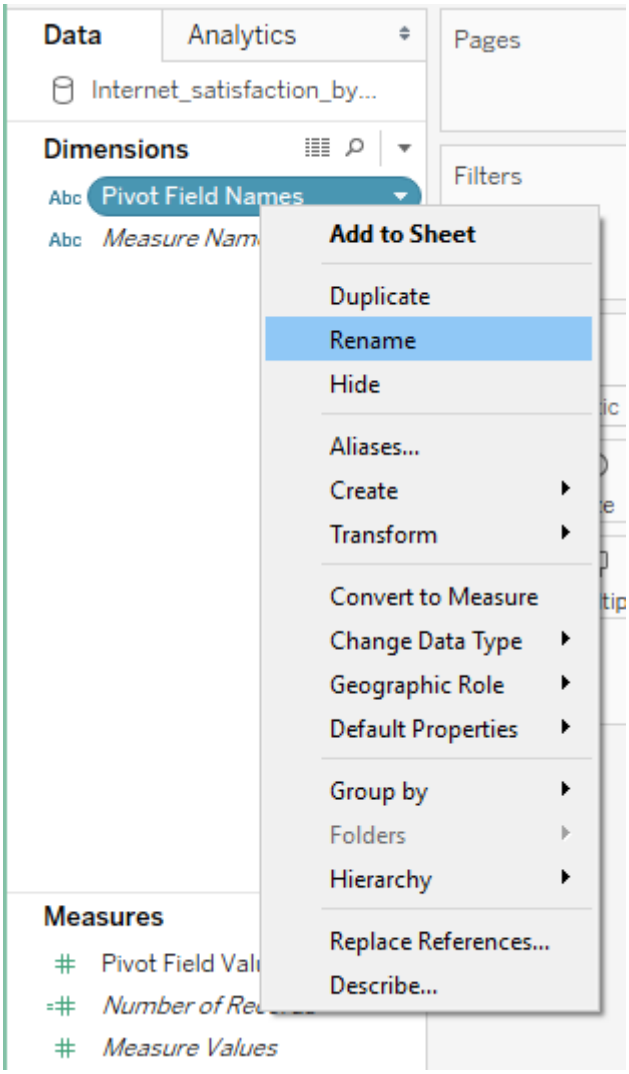
In the preview on the `Data Source` page, we see that the format of the data has changed. The original column names--- `Belgrade`, `Central Serbia`, and `Vojvodina`— **have now become labels in a new column--- `Pivot Field Names`**.

On the other hand, the values from the three original columns have all been merged into one new column--- `Pivot Field Values`:



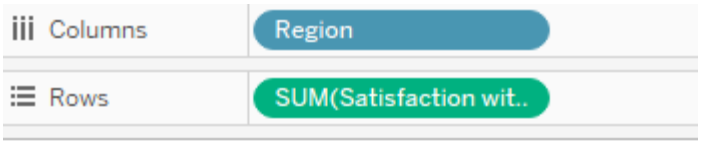
What happened is that we've simply transposed the table, so that we have values from all three columns placed into one column (one beneath the other), while the labels in the `Pivot Field Names` column denote the original column each value (row) originates from.

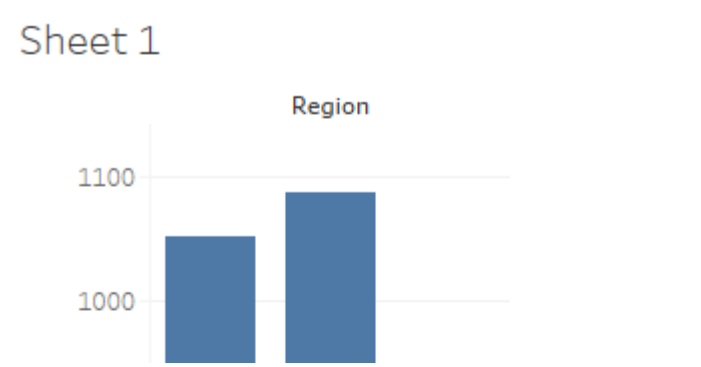
If we navigate to `Sheet 1,` we'll see that we now have `Pivot Field Names` as a dimension, and `Pivot Field Values` as a measure. For ease of use, we can rename them into something more intuitive. We can do this by right-clicking on the field and selecting `Rename` from the drop-down menu. We can then type in the desired name. For example, we can rename `Pivot Field Names` to `Region` and `Pivot Field Values` to `Satisfaction``with internet`:

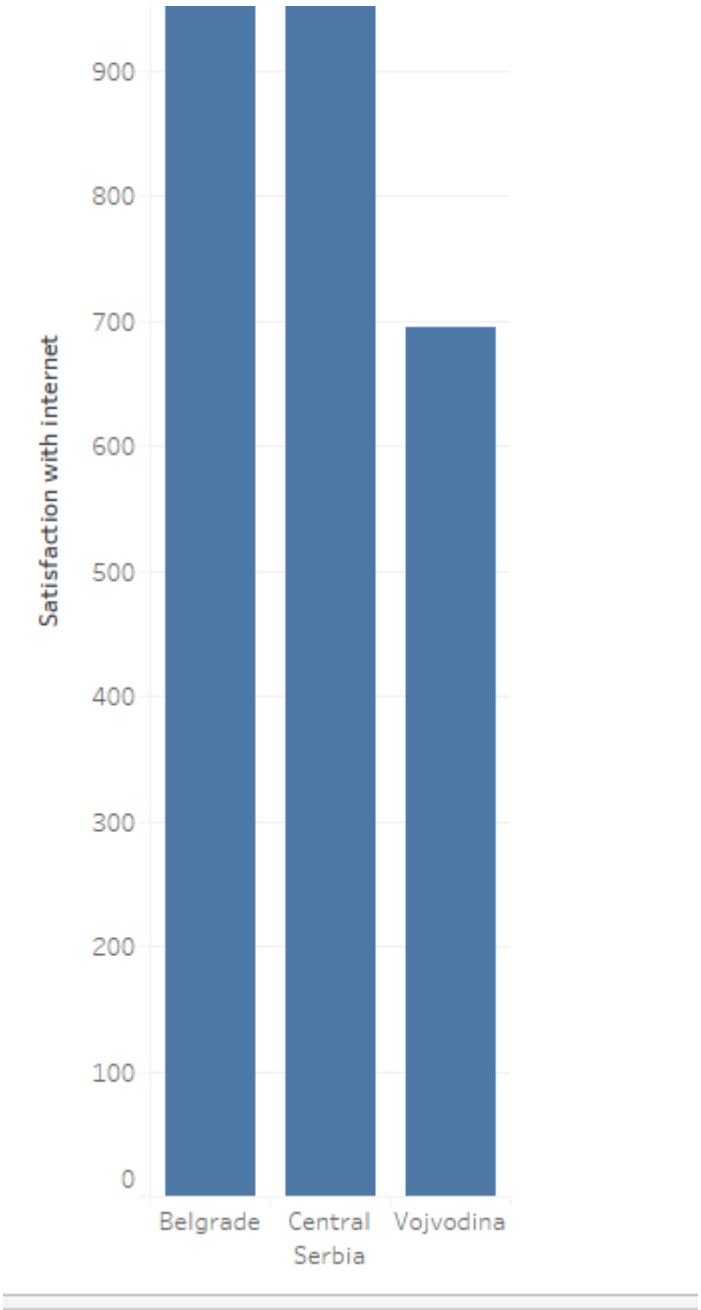


This new structure of the data source is much handier than the original one, because it allows us to make visualizations containing all three regions. Let's see how:

1. Drag and drop `Region` from `Dimensions` into the `Columns` shelf.
2. Drag and drop `Satisfaction with` internet from `Measures` into the `Rows` shelf:







We've created a chart with a single measure and single dimensions that we'd previously created by pivoting our data source.

Note

Keep in mind that other sheets using a data source might be affected by pivoting. If any of the original columns that're being pivoting have already been used in a view, they won't be available anymore after pivoting.

There's more...

You can also pivot your data by using a custom SQL query, by simply adding the `UNION ALL` operator to it.

Preparing data

Once we have connected to our data, we want to start making visualizations. However, our data might need some more touch-ups before it's ready to produce some stunning charts. This recipe will cover some common steps we need to take upon importing a data source in order to prepare it for further analysis.

Getting ready

Throughout this recipe, we'll be using the `Winery.csv` dataset. Make sure you have it saved to your device and are connected to it.

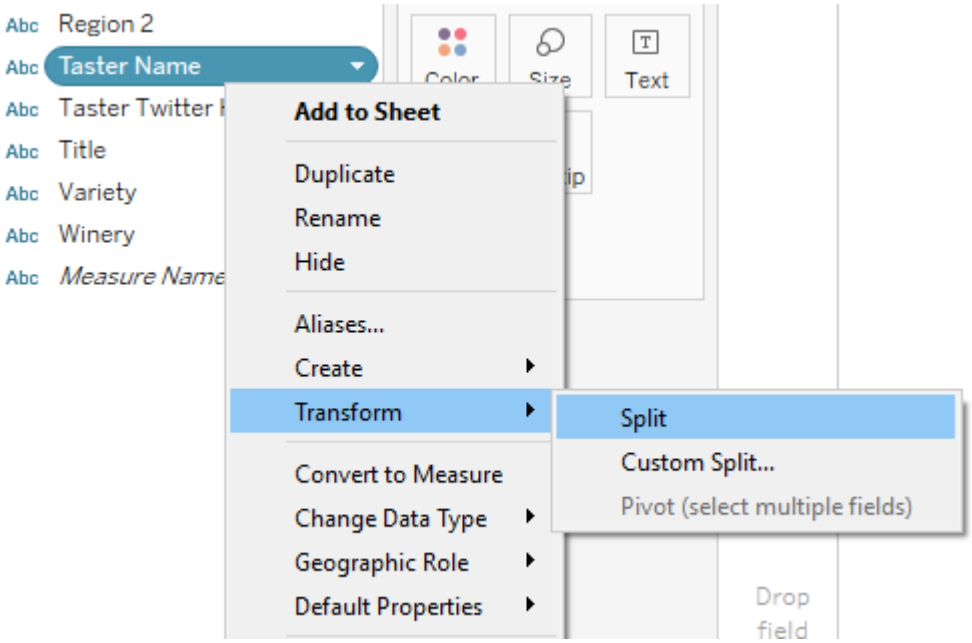
How to do it...

The following short recipes will guide you through several steps you'll often need to take when preparing your data for visualizing.

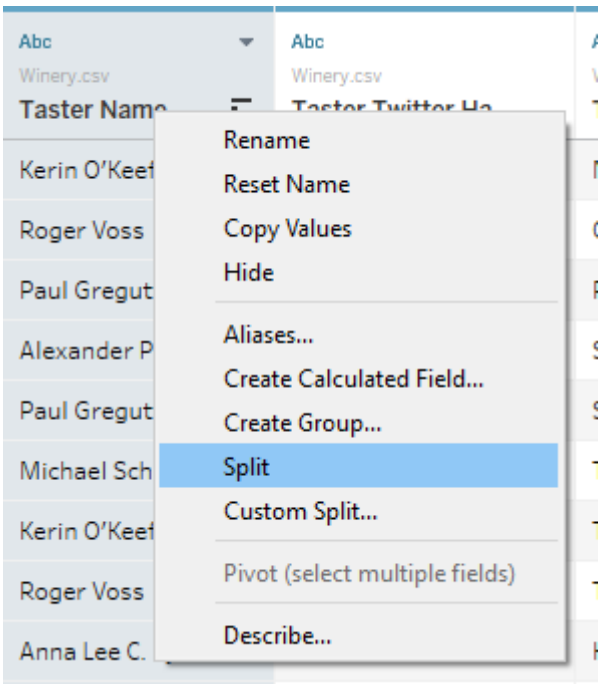
Splitting fields

In this lab, we'll learn how to split a field into multiple fields:

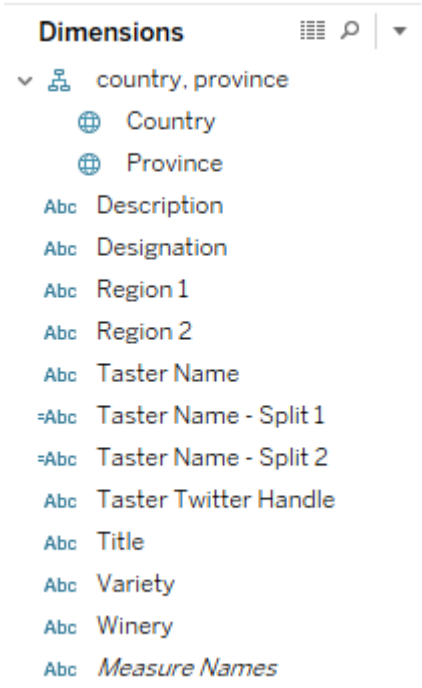
1. In the **`Data Source`** page, right-click on the header of the **`Taster Name`** field.
2. Alternatively, navigate to **`Sheet 1`**, and right-click on the **`Taster Name`** field under **`Dimensions`**. In the drop-down menu, navigate to **`Transform`**:



3. From the drop-down menu, select **`Split:`**



We've now created two new fields--- `Taster Name - Split 1` and `Taster Name - Split 2`:

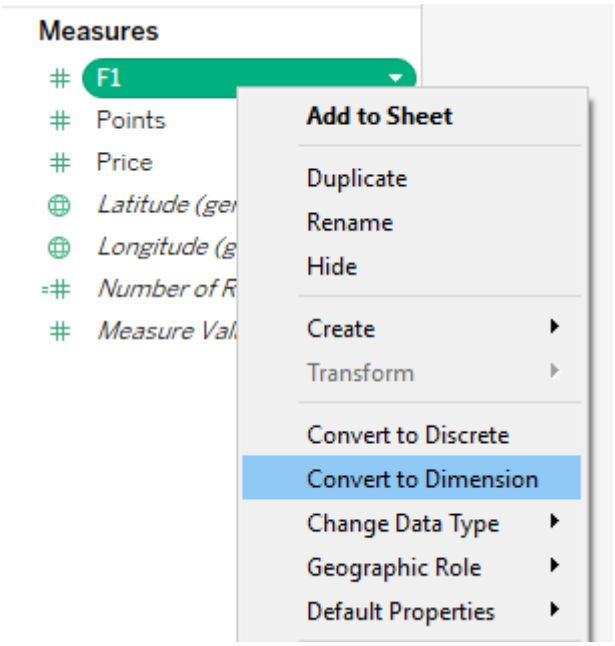


Converting measures into dimensions

Now, let's see how to convert measures into dimensions:

1. Right-click on the `F1` field under `Measures`.

2. From the drop-down menu, select `Convert to Dimension`:

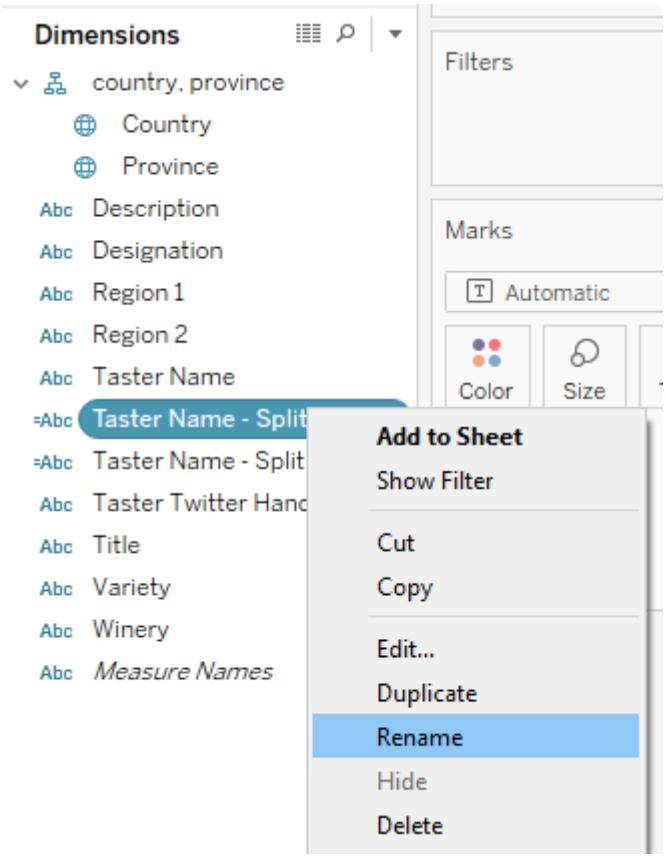


Renaming fields

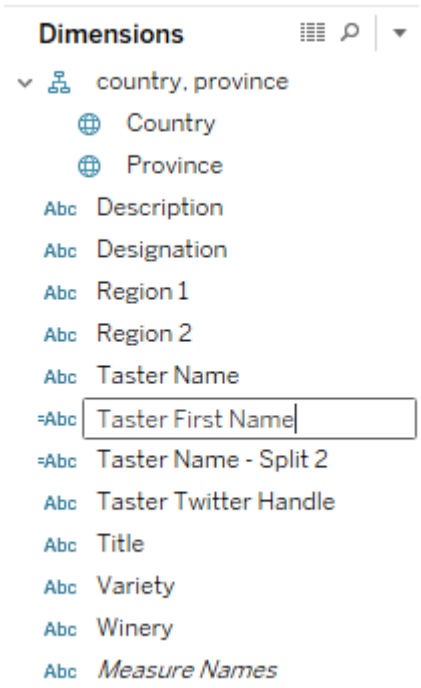
Let's rename some of our fields to make them easier to use:

1. In the `Data Source` page, double-click on the `Taster Name - Split 1` text in the column header:

2. Alternatively, navigate to `Sheet 1`, right-click on the `Taster Name - Split 1` field under `Dimensions` in the `Data` pane, and select `Rename`:

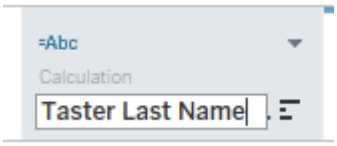


3. Type in the new name of the field as `Taster First Name`:



4. Double-click on **`Taster Name - Split 2`** in the **`Data Source`** page:

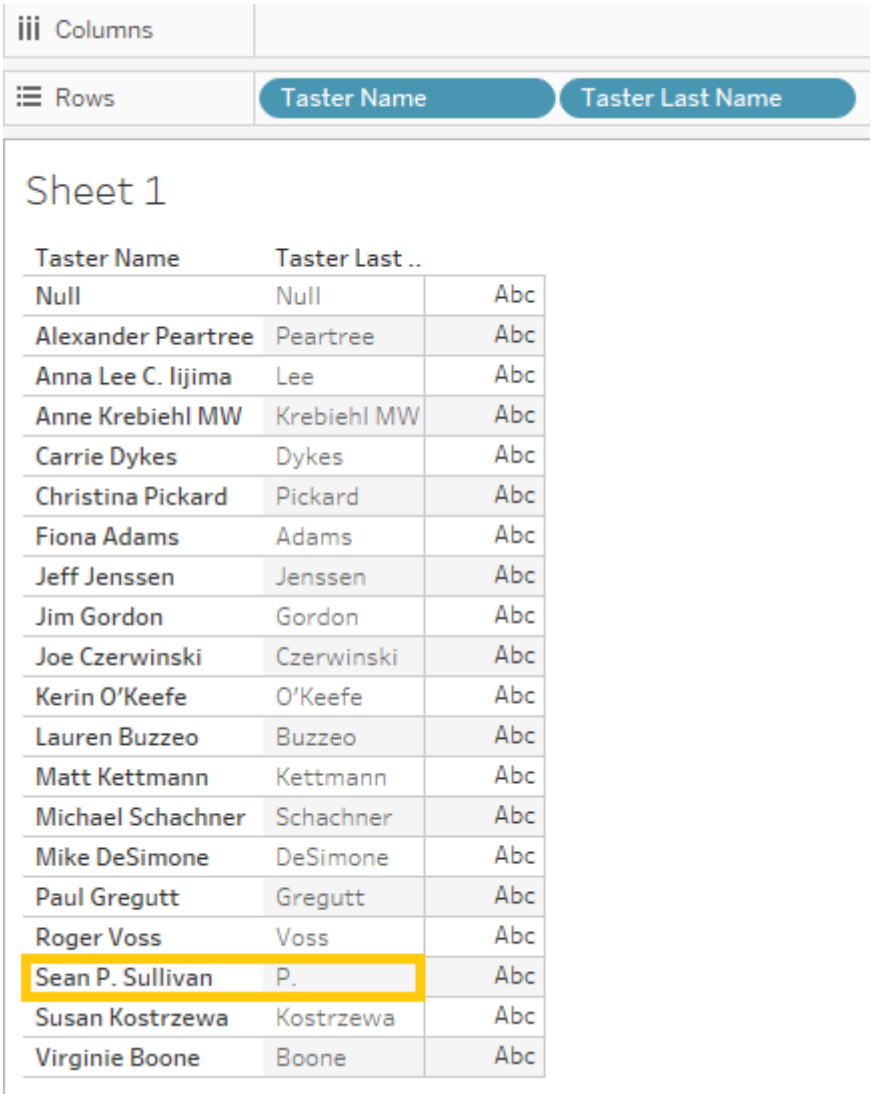
5. Rename the field to `Taster Last Name`:



Adding aliases

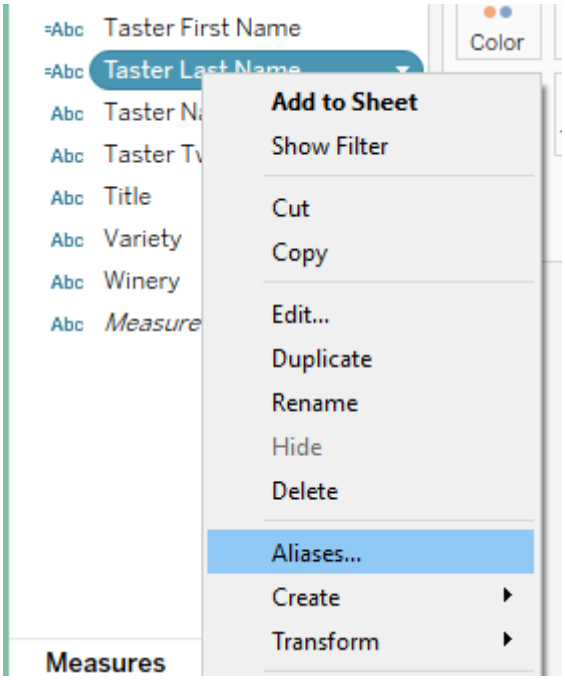
Finally, let's add an alias to one of our fields:

1. Navigate to **`Sheet 1`**.
2. Drag and drop **`Taster Name`** from **`Dimensions`** into the **`Rows`** shelf.
3. Drag and drop **`Taster Last Name`** from **`Dimensions`** into the **`Rows`** shelf. We see that the taster named **`Sean P. Sullivan`** had his name incorrectly parsed due to having a middle initial---his actual last name was cut off and his middle initial is saved as his last name, as shown in the following screenshot:

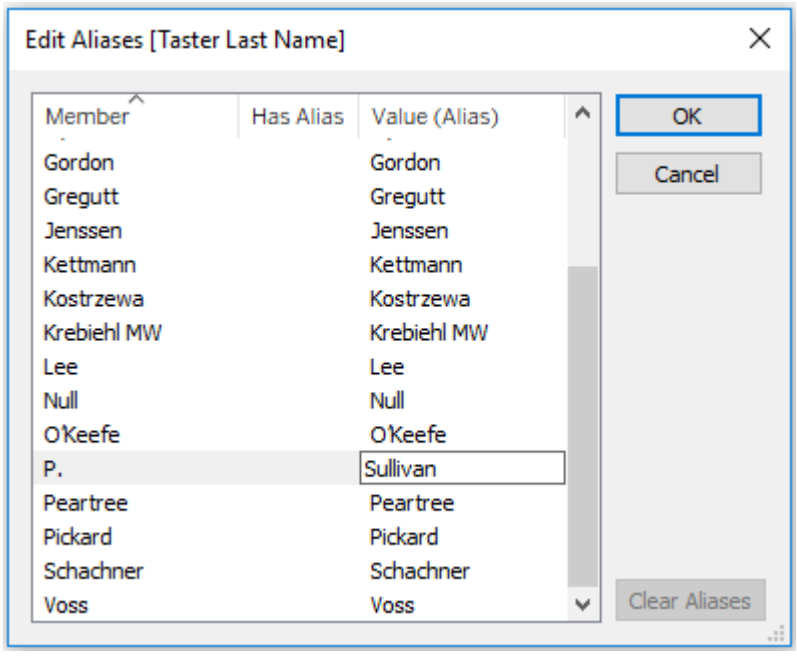


Let's correct that by adding an alias.

4. Right-click on the **`Taster Last Name`** field under **`Dimensions`**.
5. From the drop-down menu, select **`Aliases`**:



6. Alternatively, right-click on the **`Taster Last Name`** pill in the **`Rows`** shelf, and select **`Edit Aliases...`**.
7. In the **`Edit Aliases [Taster Last Name]`** window, scroll down to find the value **`P`**.
8. Click on the text **`P`**. in the right column, under **`Value (Alias)`**.
9. Instead of **`P`**., type in the actual last name of the wine taster--- `Sullivan`:



10. Click on **`OK`** to exit the window.

How it works...

In this lab, we've performed some basic data preparation.

In the first step, we split the field `Taster Name` into two fields---one containing the first and one the last name of the wine taster. Of course, this step wasn't necessary, but it's a step that often needs to be performed when dealing with string fields. It allows for more detail and flexibility in data analysis---instead of having the first and the last name concatenated, we now have them in separate columns, and we have used them separately to analyze data only by first name, last name, or both.

The next step we performed was converting a field, `F1`, from a measure into a dimension. We performed this step because `F1` is actually an index field. It isn't really a continuous measure, although Tableau automatically designated it as one due to its numerical content. But we know its values are actually discrete, because they don't represent a quantity---they serve merely as a unique identifier of the cases (rows). When to convert a measure into a dimension, and vice versa, depends exclusively on our judgement and familiarity with the dataset. We should have a good understanding of what the fields in our dataset actually represent and adjust them to a measure/dimension according to our knowledge.

In the third step, we renamed the fields we produced when splitting the `Taster Name` field. The names automatically assigned by Tableau weren't very informative. When creating views, we always want to make sure both we ourselves and the end users of our workbook or dashboard know exactly what's being presented. We should always strive to make fieldnames clear, unambiguous, and informative.

Finally, we assigned an alias to the `Taster Last Name` field member. Aliases are alternative names we can assign to members of discrete dimensions. In this case, we used an alias to correct a case of inappropriately parsed text resulting from a split. But you can use aliases whenever you would like to change a dimension member's name. As we saw, it isn't necessary to assign aliases to all of the members of a dimension---you can assign an alias to only one member, to all of them, or to any number in between.

There's more...

When splitting string fields, it's important to carefully inspect your data before and after splitting. As we saw in this example, it can easily happen that some instances don't follow the same format as the majority, and they won't be properly split.

Sometimes, the splits we need to make are quite complex---the fields might not contain the same number of separators (as was the case our example) or they might contain different separators. More frequently than not, the dataset you're using will also be too large to manually correct the fields that aren't properly split. For those and similar cases, Tableau offers custom splits, as well as splitting the fields through regular expressions.