5
Data Preparation

Before we can utilize the analytic power of Tableau, the very first step is connecting with the data. In a perfect world, we would have perfect, clean data that we could easily analyze in Tableau. But alas, in reality, the data that we need to use will most likely need to be cleaned, transformed, and managed before we can effectively use it in Tableau.

Tableau's philosophy with data preparation is to enable anyone at anytime to make fundamental changes to their data connection. This means the capabilities need four key attributes to empower you:

- **Smart**: They should apply automatically and have a deep sense of the data
- **Fast**: They need to operate at near real time even on big data
- **Repeatable**: They need to allow for changes to the underlying data, such as new values, rows, and columns
- **Flexible**: They need to allow you to make significant changes at any time while preserving your work

There are tools that exclusively help clean and reshape data. Many refer to these as **ETL** (**Extract**, **Transform**, and **Load**) tools. While Tableau is not an ETL tool, it has the ability to help clean or prepare data if it is not possible to clean or prepare it at the data source.

In this chapter, we will cover the following topics:

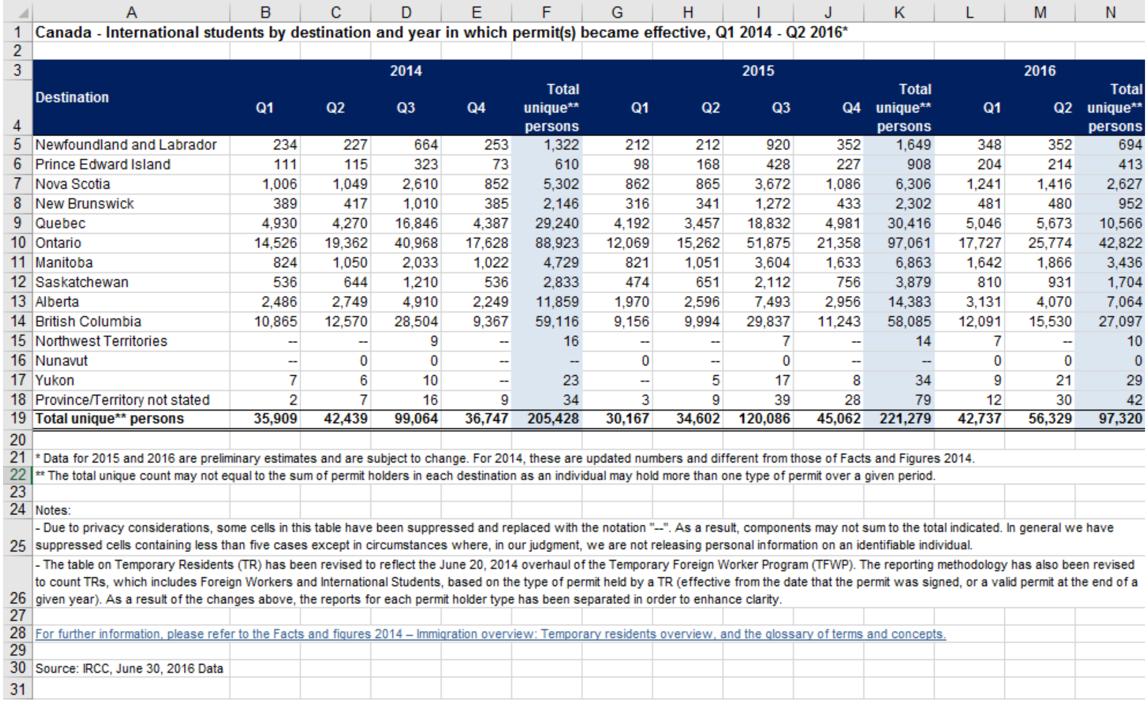
- Using the Data Interpreter and pivots
- Using the legacy Jet driver
- Using schema.ini to resolve data type issues
- Pivoting columns

- Using unions
- Using joins
- Using blends

Using the Data Interpreter and pivots

Tableau works best with clean, tall, and narrow data instead of short and wide data. The same measures should ideally be provided in a single column instead of spread out.

Let's clean up the following spreadsheet on Canada International Student Permits and ready it for Tableau:



Canada - International students by destination and year in which permit(s) became effective, Q1 2014 - Q2 2016*

Destination	2014					2015					2016		
	Q1	Q2	Q3	Q4	Total unique** persons	Q1	Q2	Q3	Q4	Total unique** persons	Q1	Q2	Total unique** persons
Newfoundland and Labrador	234	227	664	253	1,322	212	212	920	352	1,649	348	352	694
Prince Edward Island	111	115	323	73	610	98	168	428	227	908	204	214	413
Nova Scotia	1,006	1,049	2,610	852	5,302	862	865	3,672	1,086	6,306	1,241	1,416	2,627
New Brunswick	389	417	1,010	385	2,146	316	341	1,272	433	2,302	481	480	952
Quebec	4,930	4,270	16,846	4,387	29,240	4,192	3,457	18,832	4,981	30,416	5,046	5,673	10,566
Ontario	14,526	19,362	40,968	17,628	88,923	12,069	15,262	51,875	21,358	97,061	17,727	25,774	42,822
Manitoba	824	1,050	2,033	1,022	4,729	821	1,051	3,604	1,633	6,863	1,642	1,866	3,436
Saskatchewan	536	644	1,210	536	2,833	474	651	2,112	756	3,879	810	931	1,704
Alberta	2,486	2,749	4,910	2,249	11,859	1,970	2,596	7,493	2,956	14,383	3,131	4,070	7,064
British Columbia	10,865	12,570	28,504	9,367	59,116	9,156	9,994	29,837	11,243	58,085	12,091	15,530	27,097
Northwest Territories	--	--	9	--	16	--	--	7	--	14	7	--	10
Nunavut	--	0	0	--	--	0	--	0	--	--	0	0	0
Yukon	7	6	10	--	23	--	5	17	8	34	9	21	29
Province/Territory not stated	2	7	16	9	34	3	9	39	28	79	12	30	42
Total unique persons**	**35,909**	**42,439**	**99,064**	**36,747**	**205,428**	**30,167**	**34,602**	**120,086**	**45,062**	**221,279**	**42,737**	**56,329**	**97,320**

* Data for 2015 and 2016 are preliminary estimates and are subject to change. For 2014, these are updated numbers and different from those of Facts and Figures 2014.
** The total unique count may not equal to the sum of permit holders in each destination as an individual may hold more than one type of permit over a given period.

Notes:
- Due to privacy considerations, some cells in this table have been suppressed and replaced with the notation "--". As a result, components may not sum to the total indicated. In general we have suppressed cells containing less than five cases except in circumstances where, in our judgment, we are not releasing personal information on an identifiable individual.
- The table on Temporary Residents (TR) has been revised to reflect the June 20, 2014 overhaul of the Temporary Foreign Worker Program (TFWP). The reporting methodology has also been revised to count TRs, which includes Foreign Workers and International Students, based on the type of permit held by a TR (effective from the date that the permit was signed, or a valid permit at the end of a given year). As a result of the changes above, the reports for each permit holder type has been separated in order to enhance clarity.

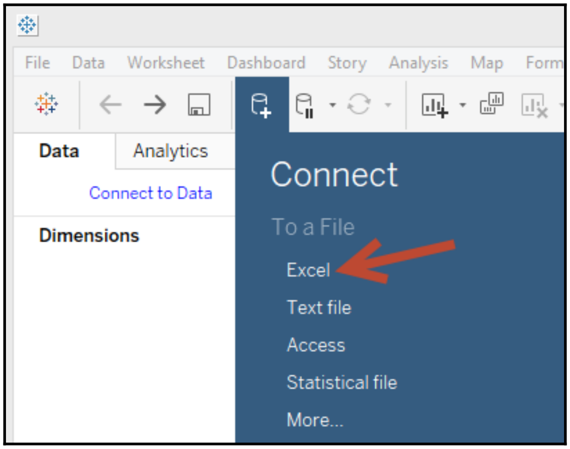
For further information, please refer to the Facts and figures 2014 – Immigration overview: Temporary residents overview, and the glossary of terms and concepts.

Source: IRCC, June 30, 2016 Data

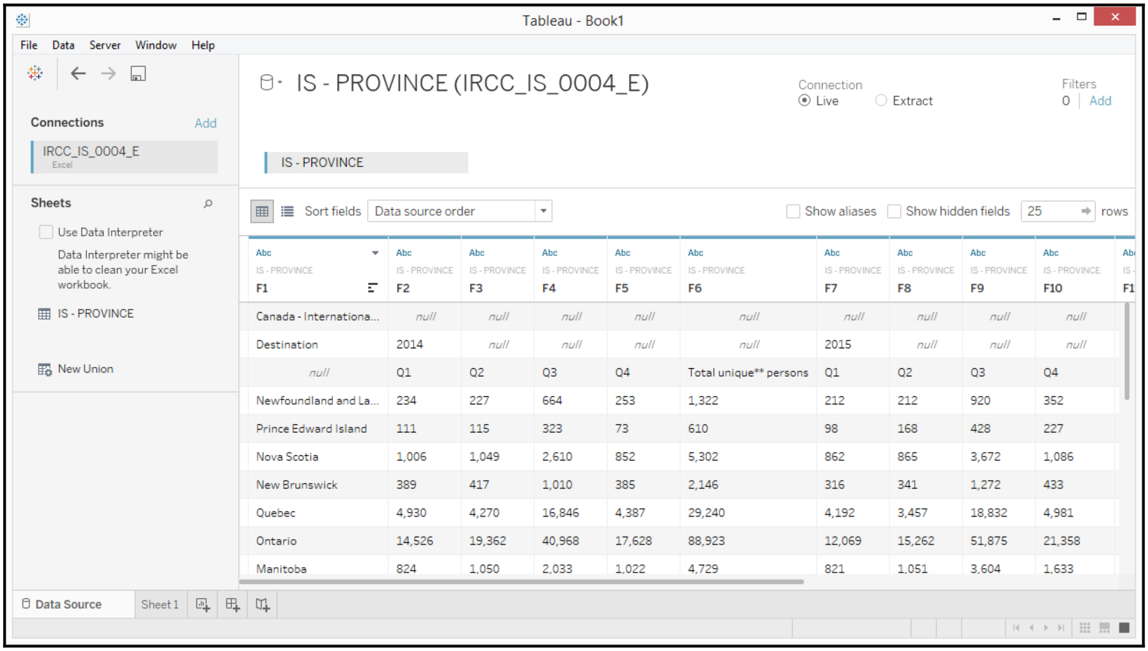
1. Download the file from the Citizenship and Immigration Canada website using the following URL:

```
http://www.cic.gc.ca/opendata-donneesouvertes/data/IRCC_IS_0004_E.
xls
```

2. Connect to the Excel file in this example. Make sure you choose **Excel** from the **To a File** section:



3. When you first connect to this Excel file, this is what you will see:



IS - PROVINCE (IRCC_IS_0004_E)

Connection ● Live ○ Extract

Filters 0 | Add

IS - PROVINCE

□ Use Data Interpreter

Data Interpreter might be able to clean your Excel workbook.

Sort fields Data source order

□ Show aliases □ Show hidden fields 25 rows

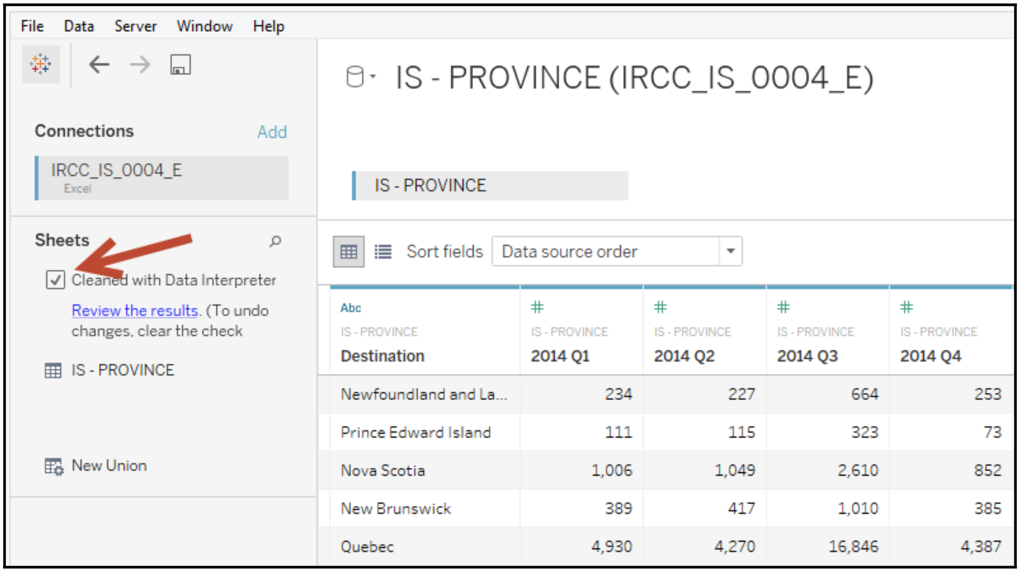
Abc	Abc	Abc	Abc	Abc	Abc	Abc	Abc	Abc	Abc
IS - PROVINCE	IS - PROVINCE	IS - PROVINCE	IS - PROVINCE	IS - PROVINCE	IS - PROVINCE	IS - PROVINCE	IS - PROVINCE	IS - PROVINCE	IS - PROVINCE
F1	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**	**F9**	**F10**
Canada - Internationa...	*null*	*null*	*null*	*null*	*null*	*null*	*null*	*null*	*null*
Destination	2014	*null*	*null*	*null*	*null*	2015	*null*	*null*	*null*
null	Q1	Q2	Q3	Q4	Total unique** persons	Q1	Q2	Q3	Q4
Newfoundland and La...	234	227	664	253	1,322	212	212	920	352
Prince Edward Island	111	115	323	73	610	98	168	428	227
Nova Scotia	1,006	1,049	2,610	852	5,302	862	865	3,672	1,086
New Brunswick	389	417	1,010	385	2,146	316	341	1,272	433
Quebec	4,930	4,270	16,846	4,387	29,240	4,192	3,457	18,832	4,981
Ontario	14,526	19,362	40,968	17,628	88,923	12,069	15,262	51,875	21,358
Manitoba	824	1,050	2,033	1,022	4,729	821	1,051	3,604	1,633

4. The original Excel file is a common type of file that many data professionals have to work with. The Excel file has a header and a footer, and the measures are spread across the columns. The number of international students--a measure--is spread out across 13 columns.

This file needs to be cleaned up:

- The header and footer needs to be removed
- Year values need to be a dimension, since these are descriptors for the measure
- The measure, which is the number of international students, needs to be placed in a single column

1. Check the checkbox beside **Use Data Interpreter**. Note that when this checkbox is checked, the label changes to **Cleaned with Data Interpreter**:



IS - PROVINCE (IRCC_IS_0004_E)

IRCC_IS_0004_E
Excel

IS - PROVINCE

☑ Cleaned with Data Interpreter

Review the results. (To undo changes, clear the check

IS - PROVINCE

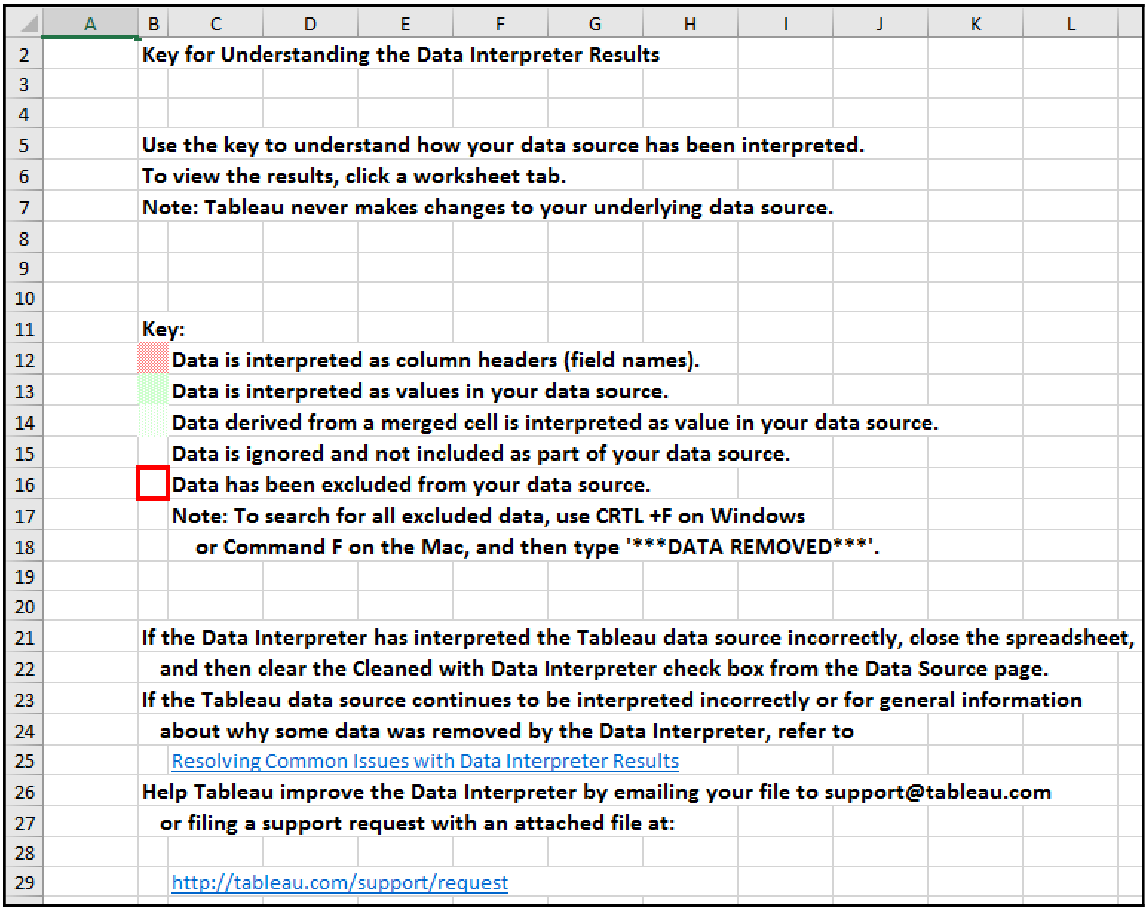
New Union

Sort fields: Data source order

Abc	#	#	#	#
IS - PROVINCE	IS - PROVINCE	IS - PROVINCE	IS - PROVINCE	IS - PROVINCE
Destination	**2014 Q1**	**2014 Q2**	**2014 Q3**	**2014 Q4**
Newfoundland and La...	234	227	664	253
Prince Edward Island	111	115	323	73
Nova Scotia	1,006	1,049	2,610	852
New Brunswick	389	417	1,010	385
Quebec	4,930	4,270	16,846	4,387

2. Select all fields except for **Destination**.

   When we use the Tableau Data Interpreter, it will clean up the headers and footers, but will not clean up the year dimension and the measure for the number of international students. When we run the Data Interpreter, we can also choose to review the results by clicking on the provided link. The first tab, presented here, provides the key to what the Data Interpreter does:



Key for Understanding the Data Interpreter Results

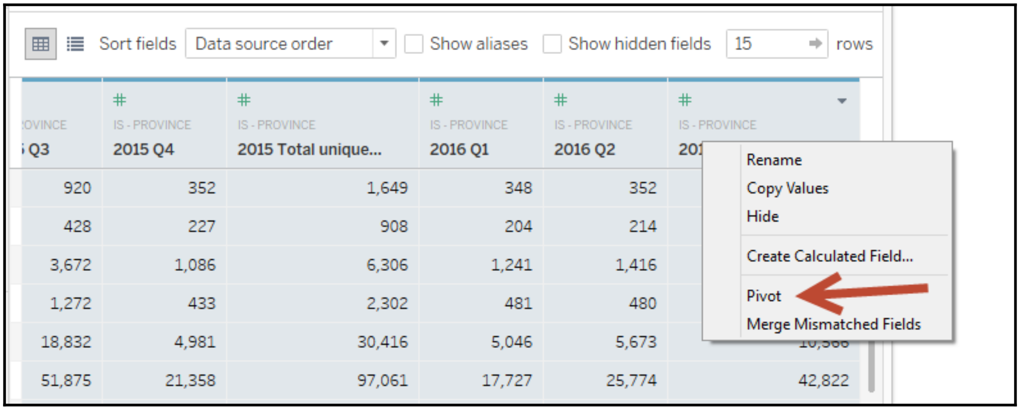
Use the key to understand how your data source has been interpreted.
To view the results, click a worksheet tab.
Note: Tableau never makes changes to your underlying data source.

Key:
🟥 **Data is interpreted as column headers (field names).**
🟩 **Data is interpreted as values in your data source.**
▨ **Data derived from a merged cell is interpreted as value in your data source.**
Data is ignored and not included as part of your data source.
□ **Data has been excluded from your data source.**
Note: To search for all excluded data, use CRTL +F on Windows
or Command F on the Mac, and then type '*DATA REMOVED***'.**

If the Data Interpreter has interpreted the Tableau data source incorrectly, close the spreadsheet,
and then clear the Cleaned with Data Interpreter check box from the Data Source page.
If the Tableau data source continues to be interpreted incorrectly or for general information
about why some data was removed by the Data Interpreter, refer to
Resolving Common Issues with Data Interpreter Results
Help Tableau improve the Data Interpreter by emailing your file to support@tableau.com
or filing a support request with an attached file at:

http://tableau.com/support/request

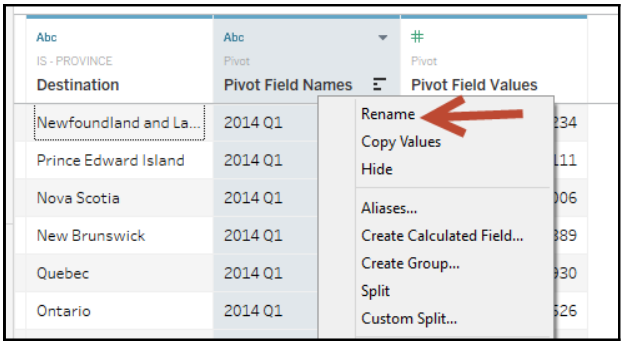
3. To further clean our data source, we need to pivot the remaining so year the values and number of international students are stored in single columns. While the fields are selected, right-click and choose **Pivot**:



Sort fields Data source order □ Show aliases □ Show hidden fields 15 rows

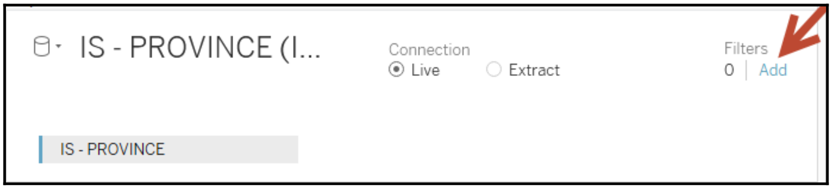
OVINCE Q3	IS - PROVINCE 2015 Q4	IS - PROVINCE 2015 Total unique...	IS - PROVINCE 2016 Q1	IS - PROVINCE 2016 Q2	IS - PROVINCE 201
920	352	1,649	348	352	
428	227	908	204	214	
3,672	1,086	6,306	1,241	1,416	
1,272	433	2,302	481	480	
18,832	4,981	30,416	5,046	5,673	10,966
51,875	21,358	97,061	17,727	25,774	42,822

4. Right-click the new fields to rename them:
   a. Change **Pivot Field Names** to **Period**
   b. Change **Pivot Field Values** to **International Students**:

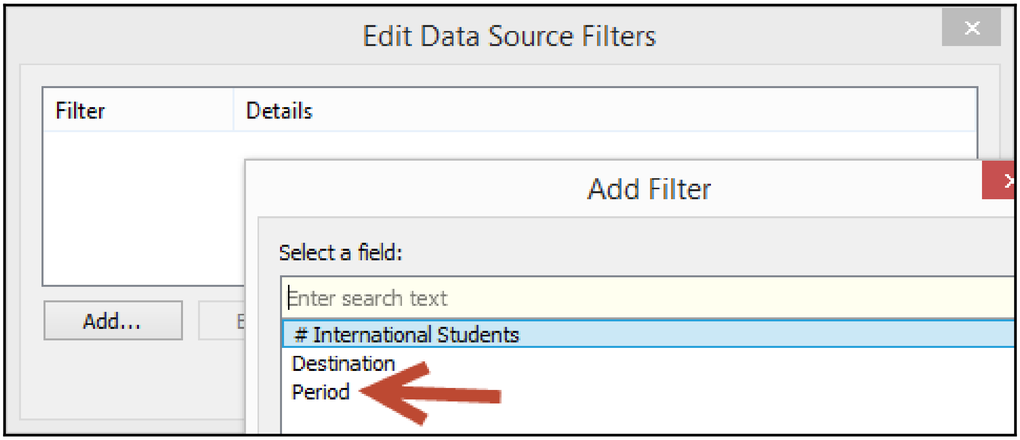


Abc IS - PROVINCE Destination	Abc Pivot Pivot Field Names	# Pivot Pivot Field Values
Newfoundland and La...	2014 Q1	34
Prince Edward Island	2014 Q1	11
Nova Scotia	2014 Q1	06
New Brunswick	2014 Q1	89
Quebec	2014 Q1	30
Ontario	2014 Q1	26

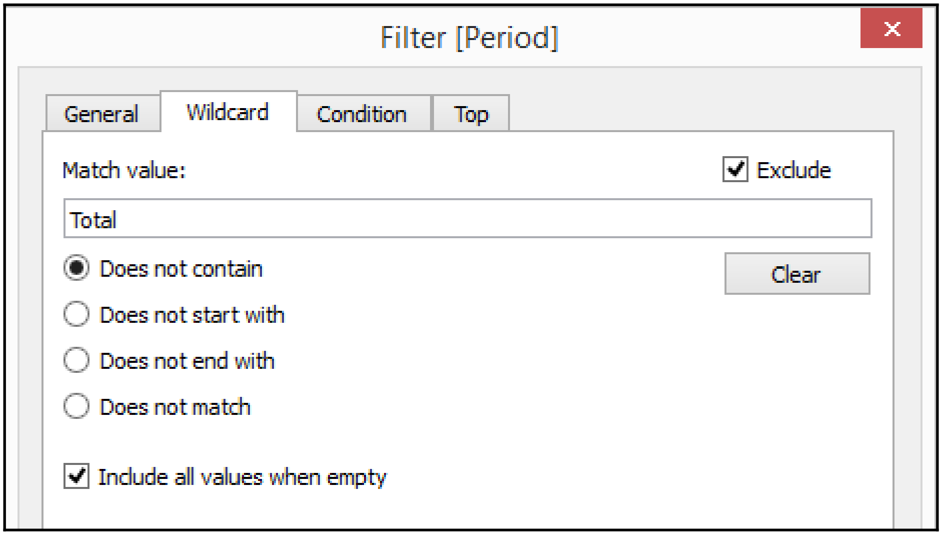
   c. Click on**Add**underneath**Filters**:



5. In the **Select a field:** option, choose **Period**:



6. In the filter window for **Period**, under the **Wildcard** tab, type `Total` and check the **Exclude** checkbox:



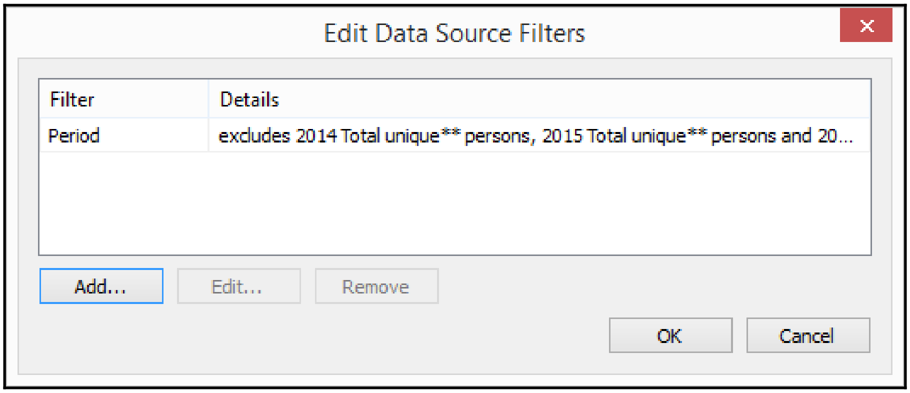
Match value: ☑ Exclude

Total

● Does not contain
○ Does not start with
○ Does not end with
○ Does not match

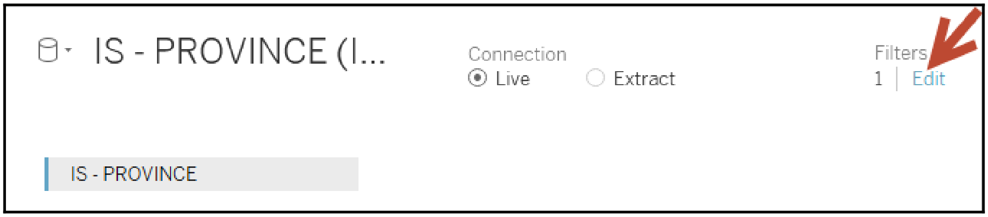
☑ Include all values when empty

7. Once you click **OK**, you should see the following in the **Edit Data Source Filters** box:



Filter	Details
Period	excludes 2014 Total unique** persons, 2015 Total unique** persons and 20...

8. Click on **OK** when done.The original Excel file has some total fields, which we excluded, so that we can keep the granularity of the measure consistent; for example, we would not want to sum all the measures and the field for total unique persons.
9. Under **Filters**, click on **Edit** to add one more filter:

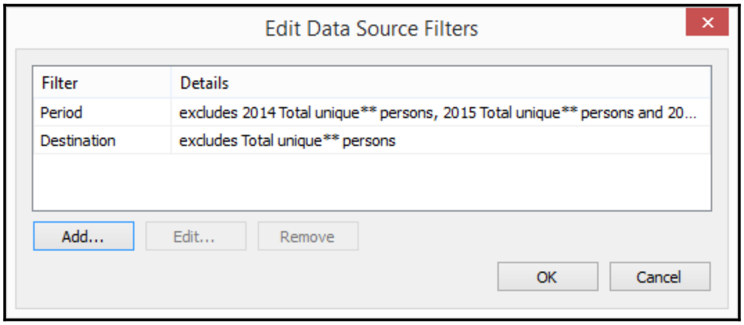


Connection
● Live ○ Extract

Filters
1 | Edit

10. This time, choose the **Destination** field.
11. In the filter window for **Destination**, under the **Wildcard** tab, type `Total` and check the **Exclude** checkbox.

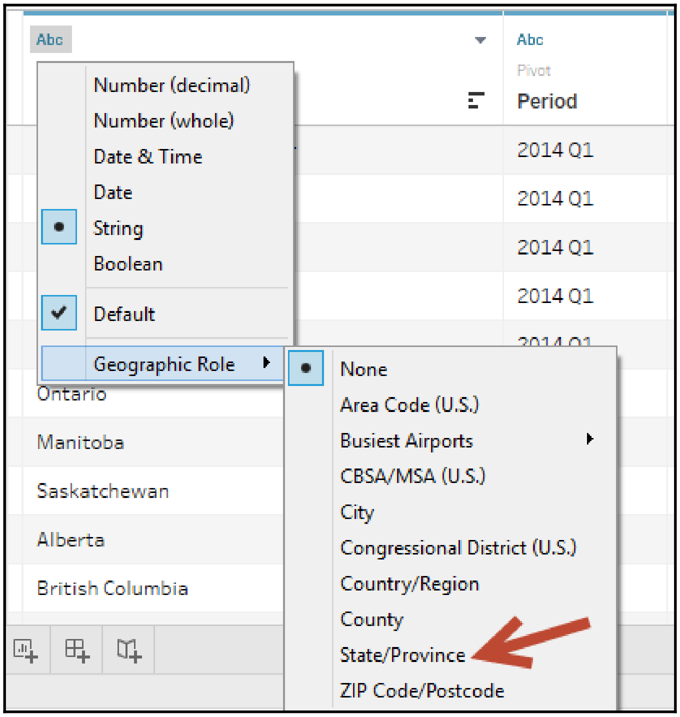
12. Once you click **OK**, you should see the following in the **Edit Data Source Filters** box:



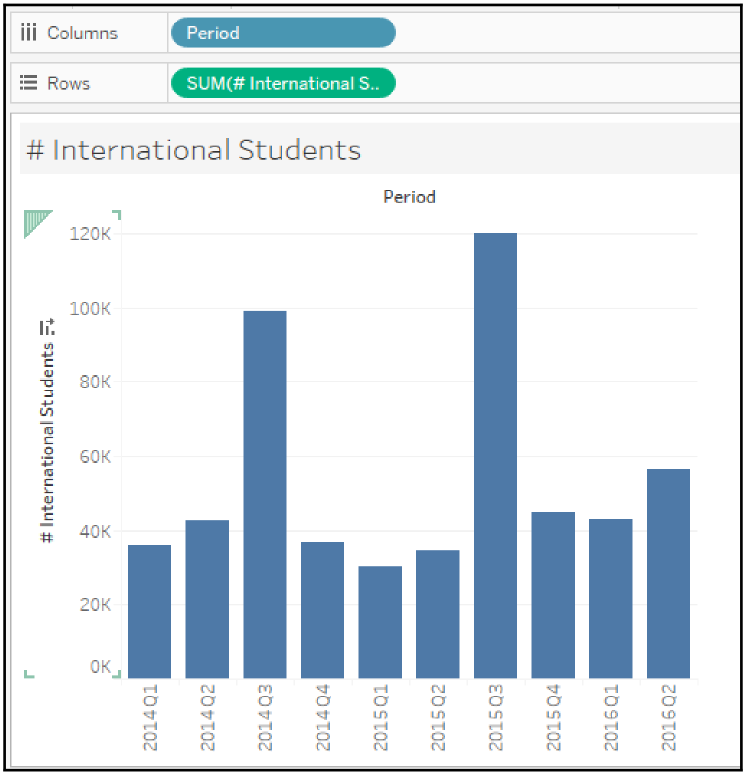
Edit Data Source Filters

Filter	Details
Period	excludes 2014 Total unique** persons, 2015 Total unique** persons and 20...
Destination	excludes Total unique** persons

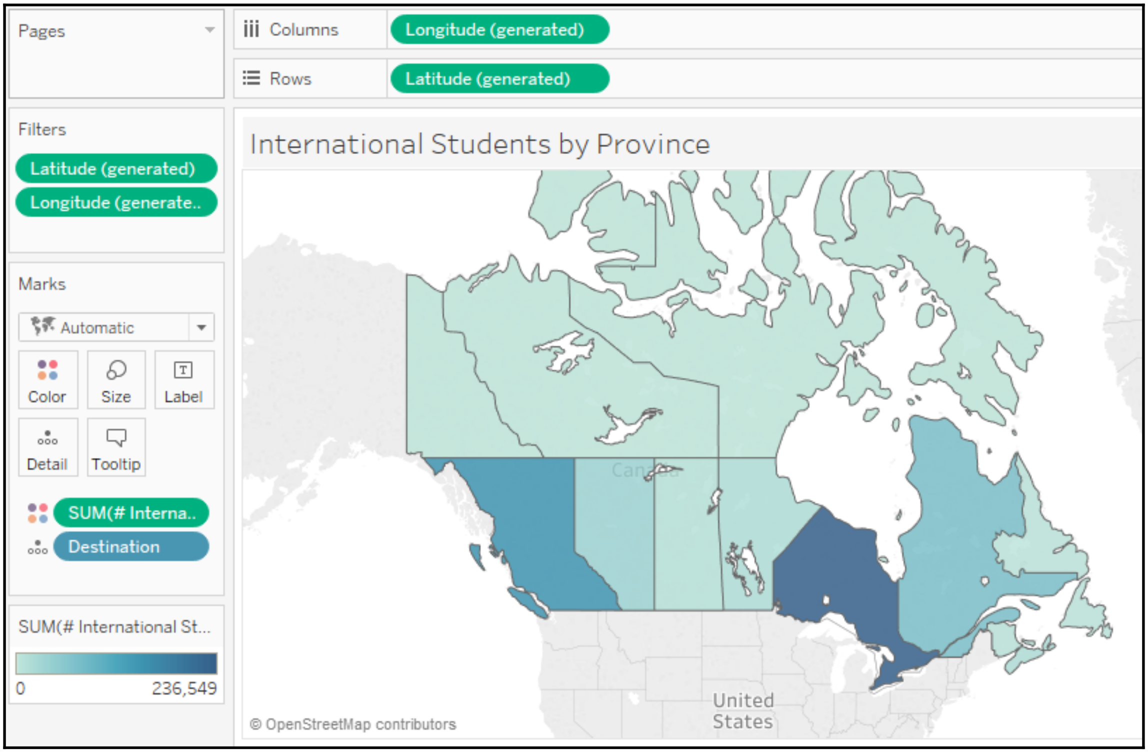
13. In the preview pane, click on the **Abc** symbol above **Destination** and change **Geographic Role** to **State/Province**:



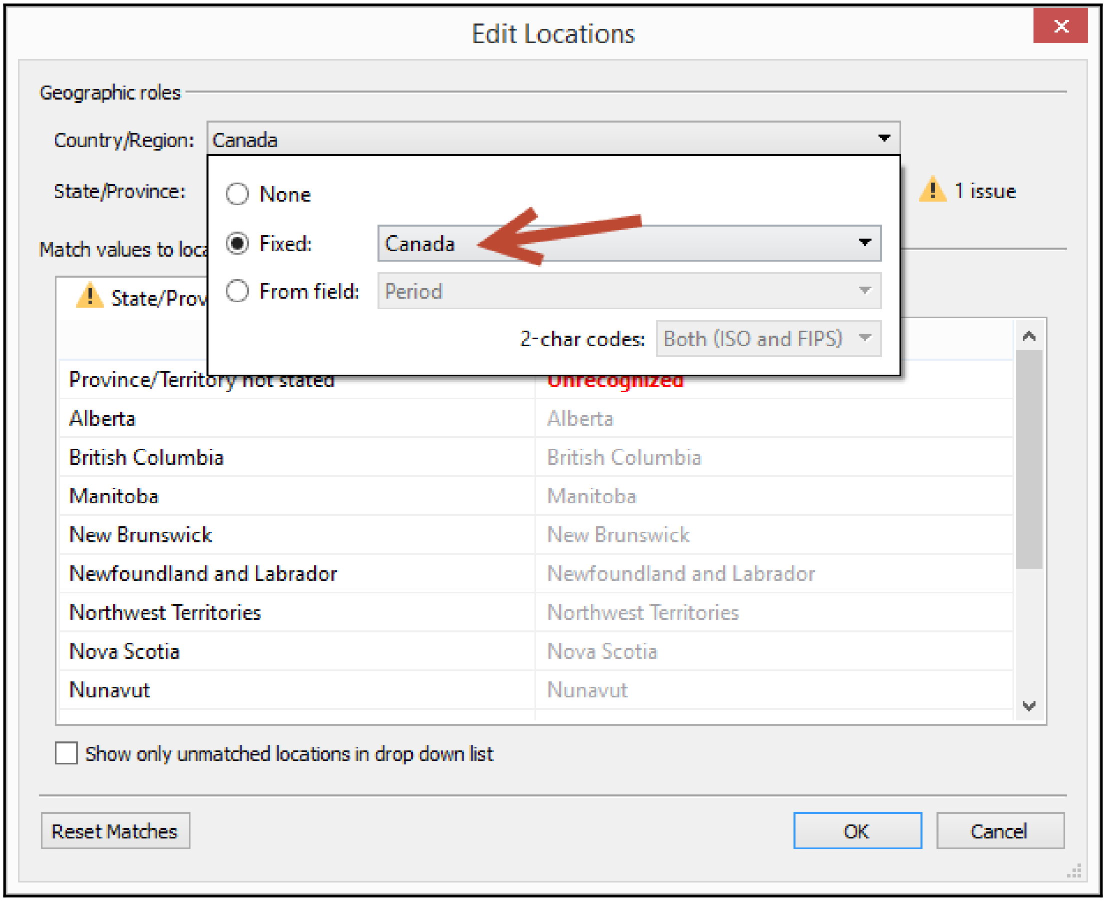
14. From here, we can create a new sheet and create visualizations that are easier to work with in Tableau. The following screenshot depicts the number of students per period:



15. Since we have geocoded the **Destination** and assigned it the **State/Province** geographic role, we can also create a filled map to see where students are going:



16. Although the **Destination** field is geocoded to **State/Province**, we will still need the **Country** information before we can successfully create a map. For this data set, we can simply set the country manually by going to the Mapmenu item, and selecting **Edit Locations**. We can set this to **Canada**:



Geographic roles

Country/Region: Canada

State/Province:

○ None

● Fixed: Canada

○ From field: Period

2-char codes: Both (ISO and FIPS)

⚠ 1 issue

Match values to loca

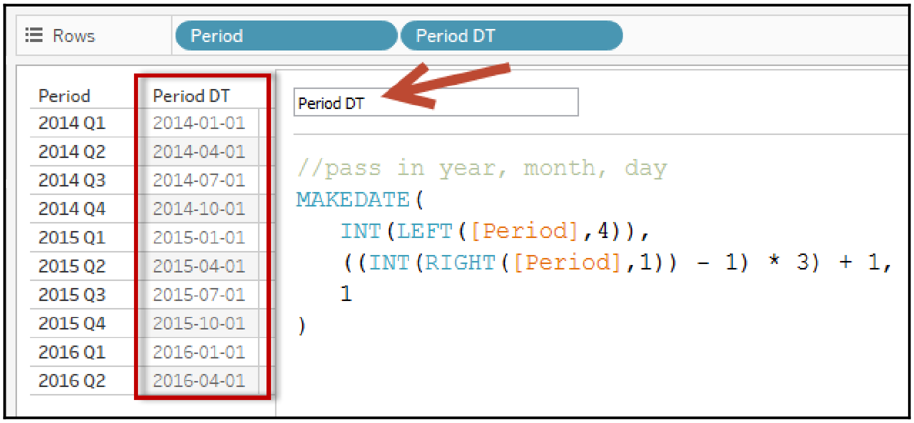
⚠ State/Prov

Province/Territory not stated	Unrecognized
Alberta	Alberta
British Columbia	British Columbia
Manitoba	Manitoba
New Brunswick	New Brunswick
Newfoundland and Labrador	Newfoundland and Labrador
Northwest Territories	Northwest Territories
Nova Scotia	Nova Scotia
Nunavut	Nunavut

□ Show only unmatched locations in drop down list

17. Alternatively, we can create a field for **Country** and use that in the geocoding.

18. You can probably see that there is additional cleanup and transformation that can be done. **Period**, for example, can be split further into year and quarter. We can even go as far as creating a date for the start of the period. This can be done using a calculated field:



Period	Period DT
2014 Q1	2014-01-01
2014 Q2	2014-04-01
2014 Q3	2014-07-01
2014 Q4	2014-10-01
2015 Q1	2015-01-01
2015 Q2	2015-04-01
2015 Q3	2015-07-01
2015 Q4	2015-10-01
2016 Q1	2016-01-01
2016 Q2	2016-04-01

Period DT

```
//pass in year, month, day
MAKEDATE(
    INT(LEFT([Period],4)),
    ((INT(RIGHT([Period],1)) - 1) * 3) + 1,
    1
)
```

Using the legacy Jet driver

Let's use the New York Restaurant Inspections Excel file and use the legacy Jet driver to shape the file so that we can have both the inspection date and grade date in the same column.

The challenge here is that we often have a universal notion of date, that is, a date is a day that isn't specific to any events. We may want to summarize or aggregate measures based on this universal notion of dates. However, in reality, dates may exist in different fields with different contexts, and this can limit our ability to work on them as a single unit.

In the Excel file for this recipe, we want to count how many restaurants were inspected and how many were graded for a specific date. The Excel file does not have a generic date field that allows us to count how many were inspected or graded. Thus, we need to re-shape our data so that **Inspection Date** and **Grade Date** exist in one column instead of two.

If we are using an Excel file as our data source, we can potentially use the legacy Jet connection, which allows custom SQL statements against the Excel file.

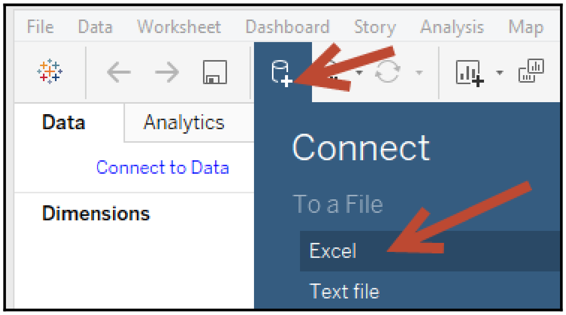
The legacy connection option was introduced in Tableau 8.2. You can learn more about this in the Tableau KB article *Differences between Legacy and Default Excel and Text File Connections*, which can be found at `http://onlinehelp.tableau.com/current/pro/desktop/en-us/help.htm#upgrading_connection.html`.

Download the file from the New York City Open Data website using the following URL:

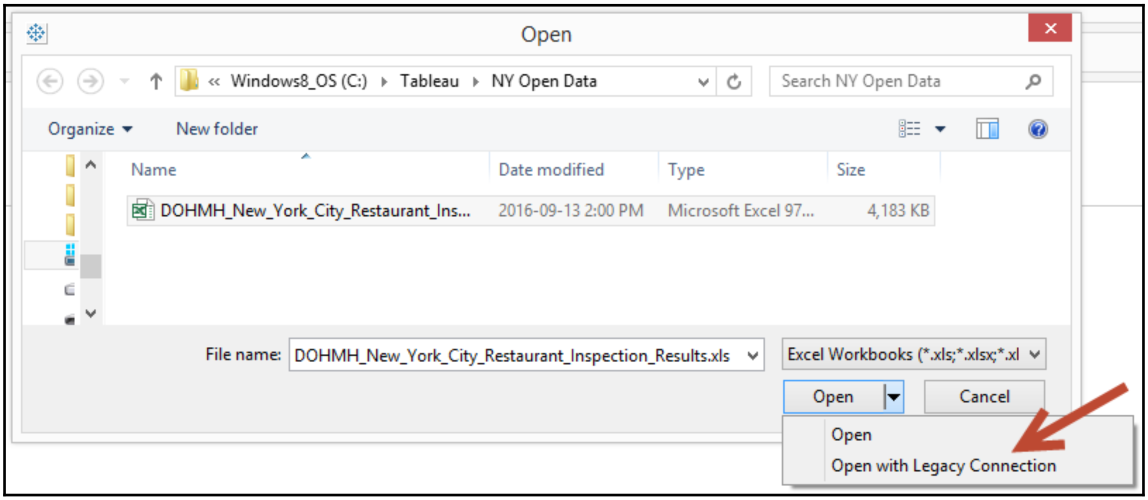
`https://nycopendata.socrata.com/Health/DOHMH-New-York-City-Restaurant-Inspection-Results/xx67-kt59/data`

Once you have downloaded the data, save the file as `DOHMH_New_York_City_Restaurant_Inspection_Results.xls` (Microsoft Excel 97-2003 worksheet). Note that the records may have been updated between the time of writing and the time of your download:

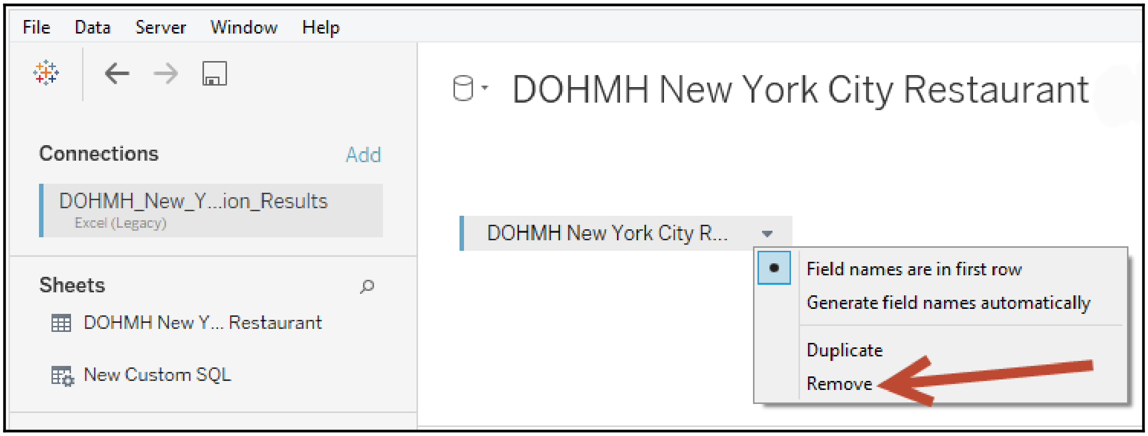
1. Click on **New Data Source** icon, and choose **Excel**:



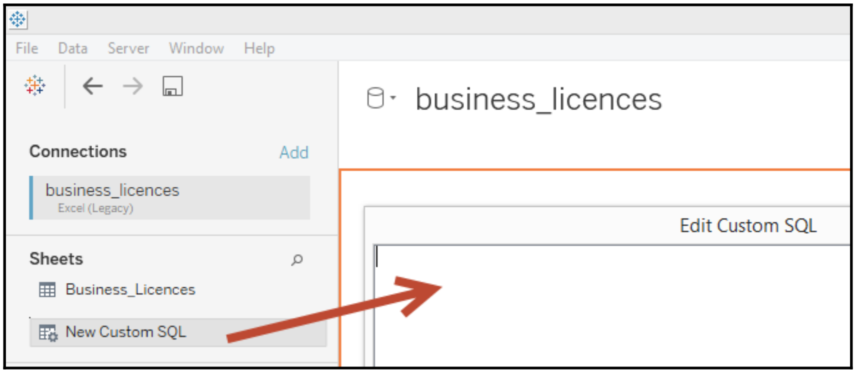
2. Choose `DOHMH_New_York_City_Restaurant_Inspection_Results.xls`, and select **Open with Legacy Connection**:


Open
« Windows8_OS (C:) ▸ Tableau ▸ NY Open Data
Search NY Open Data
Organize ▾
New folder
Name
Date modified
Type
Size
DOHMH_New_York_City_Restaurant_Ins...
2016-09-13 2:00 PM
Microsoft Excel 97...
4,183 KB
File name: DOHMH_New_York_City_Restaurant_Inspection_Results.xls
Excel Workbooks (*.xls;*.xlsx;*.xl
Open
Cancel
Open
Open with Legacy Connection

3. In the **Connections** window, remove the existing connection to the one sheet in the Excel file:


File Data Server Window Help
DOHMH New York City Restaurant
Connections
Add
DOHMH_New_Y...ion_Results
Excel (Legacy)
Sheets
DOHMH New Y... Restaurant
New Custom SQL
DOHMH New York City R...
Field names are in first row
Generate field names automatically
Duplicate
Remove

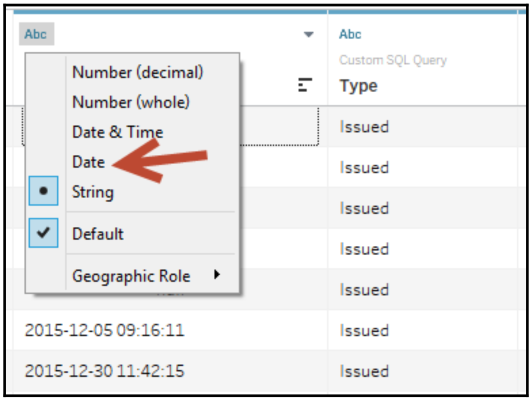
4. Drag **New Custom SQL** to the main connection pane:



5. Add the following code to the **Edit Custom SQL** window:

```
SELECT
[DBA],
[CAMIS],
[CUISINE DESCRIPTION],
[INSPECTION DATE],
[GRADE DATE],
[INSPCTION DATE] AS [Date],
'Inspected' AS [Type]
FROM [DOHMH New York City Restaurant$]
UNION ALL
SELECT
[DBA],
[CAMIS],
[CUISINE DESCRIPTION],
[INSPECTION DATE],
[GRADE DATE],
[GRADE DATE] AS [Date],
'Graded' AS [Type]
FROM [DOHMH New York City Restaurant$]
```

6. In the preview window, click on the **Abc** symbol above the **Date** field and select **Date** to change the data type to Date:



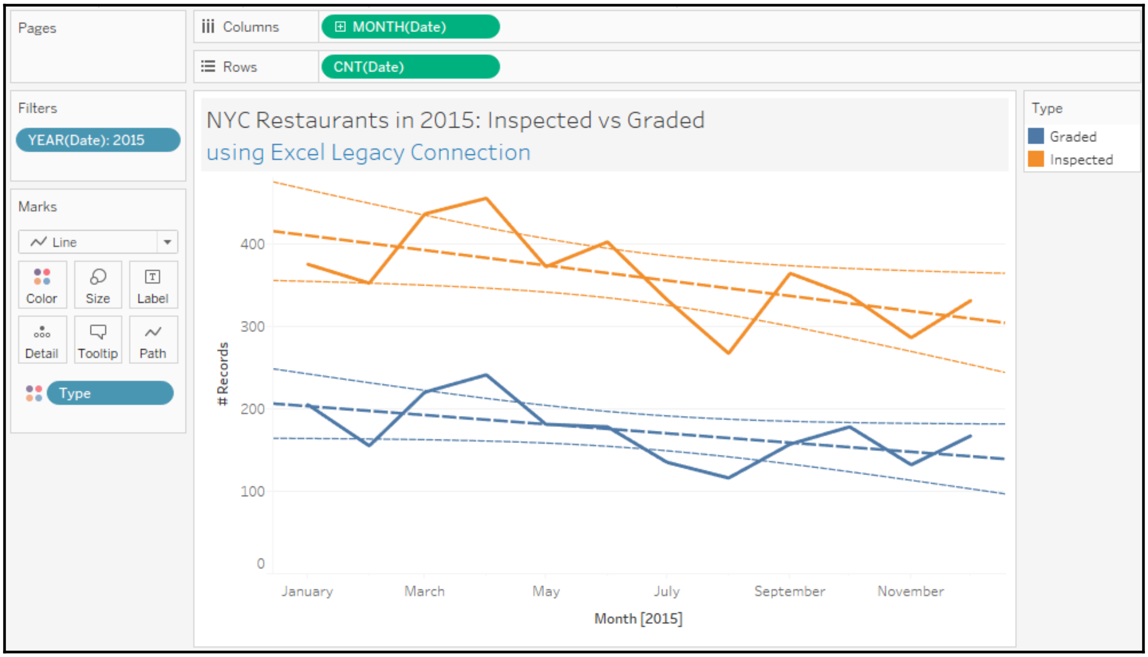
When we query our Excel spreadsheet, each tab will be treated as a table and referenced as the worksheet name with a `$` symbol at the end and enclosed in square brackets, like so: `[DOHMH New York City Restaurant$]`.

QuerySurge has a good short tutorial on using SQL against Excel spreadsheets here: `http://bit.ly/QuerySurge-SQL-against-Excel`.

What we will do in this query is stack two copies of the original data set on top of each other using the `UNION ALL` set operator, and introduce two new fields - **Date** and **Type**. This forces one field to contain the two dates we are interested in.

The first set uses **INSPECTION DATE** as the value for:**Date**, and **Inspected** as the value for **Type**. The second set uses **GRADE DATE** as the value for **Date**, and **Graded** as the value for **Type**. If you need to add additional fields for your analysis, you can simply add the field names to both `SELECT` statements.

Once we have the fields in place, we can analyze and visualize our data. For example, we can create a time series graph with trend lines. Since we have a single date field to consider, we can simply drag that **Date** field and create a continuous axis. Since we also have a single field to differentiate what event that date was related to, we can use that in **Color** in the **Marks** card to create two separate lines for the **Graded** and **Inspected** events:



The measure in this example is **CNT(Date)** because **Date** will have a value if it is related to the event, and null (and will not be counted) if it is not.

Be careful when doing other kinds of analysis. Since we stacked two copies of our data set, we essentially doubled our record count.

We are only using the legacy connection because our data source is an Excel file. If your data source is different, for example, if you are using a relational data source, you can re-shape the data using those data source's query mechanisms. In a relational data source, you may be able to do a union or a self-join at the data source level before the data is consumed by Tableau.

Tableau 10 introduces a new feature called cross database join, which we can also consider. Cross database join allows you to connect to multiple data sources and join them from within the Tableau connection interface. In the following example, we have essentially connected to the same Excel worksheet three times:

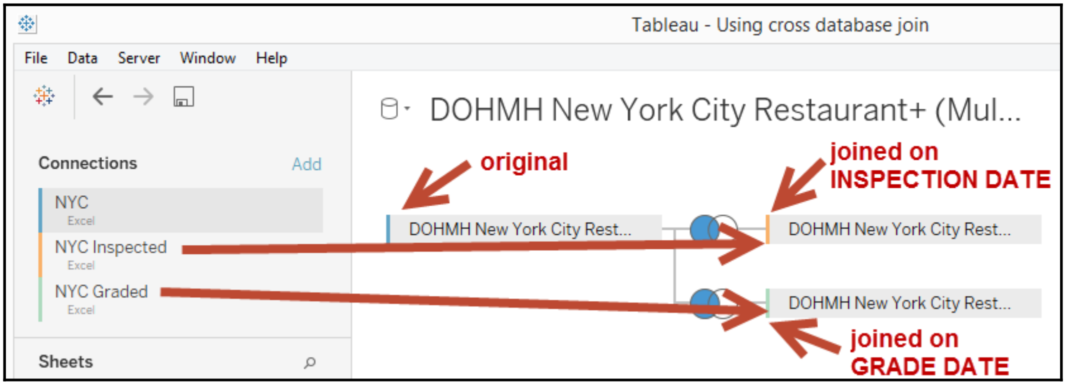
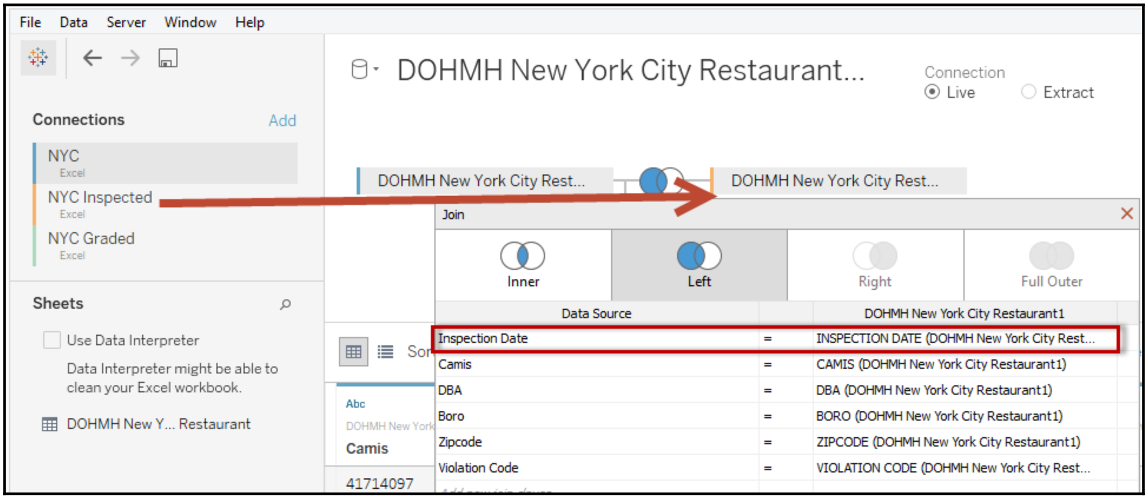


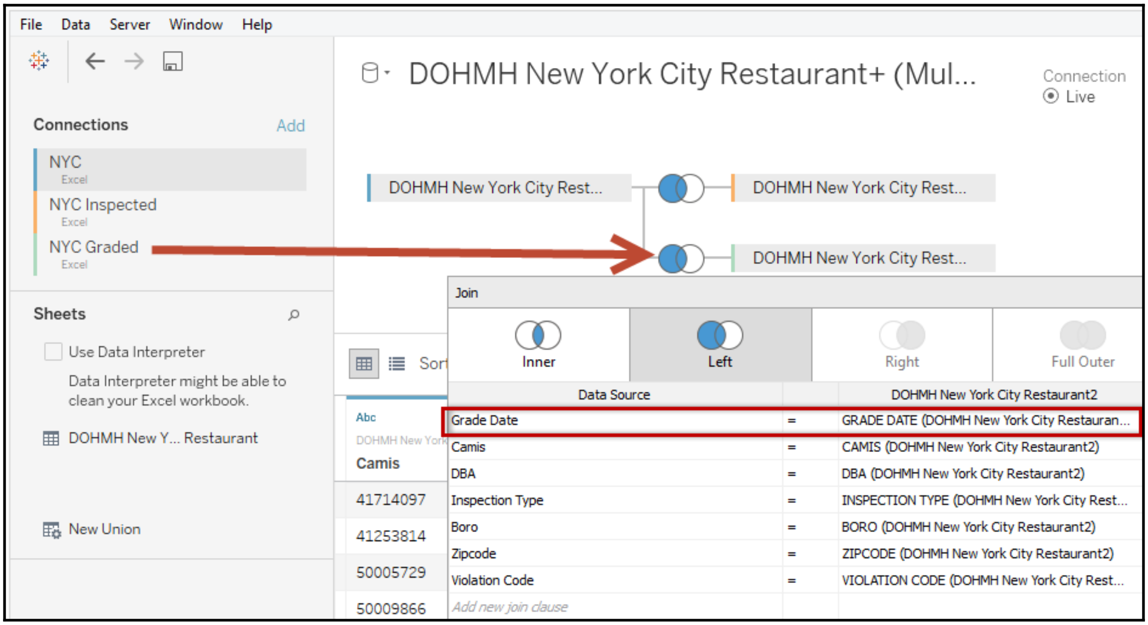
Tableau - Using cross database join
File Data Server Window Help
Connections
Add
NYC
Excel
NYC Inspected
Excel
NYC Graded
Excel
Sheets
DOHMH New York City Restaurant+ (Mul...
original
joined on INSPECTION DATE
DOHMH New York City Rest...
DOHMH New York City Rest...
DOHMH New York City Rest...
joined on GRADE DATE

Each connection is a left join. The first one connects mainly based on the **INSPECTION DATE**. There are other fields being considered in the join to ensure we are only matching the correct records. Otherwise, we will end up with something called a cross join and may match one record to all other records of restaurants that were inspected on the same date:


File Data Server Window Help
Connections
Add
NYC
Excel
NYC Inspected
Excel
NYC Graded
Excel
Sheets
Use Data Interpreter
Data Interpreter might be able to clean your Excel workbook.
DOHMH New Y... Restaurant
DOHMH New York City Restaurant...
Connection
Live
Extract
DOHMH New York City Rest...
DOHMH New York City Rest...
Join
Inner
Left
Right
Full Outer
Sor
Abc
DOHMH New York
Camis
41714097

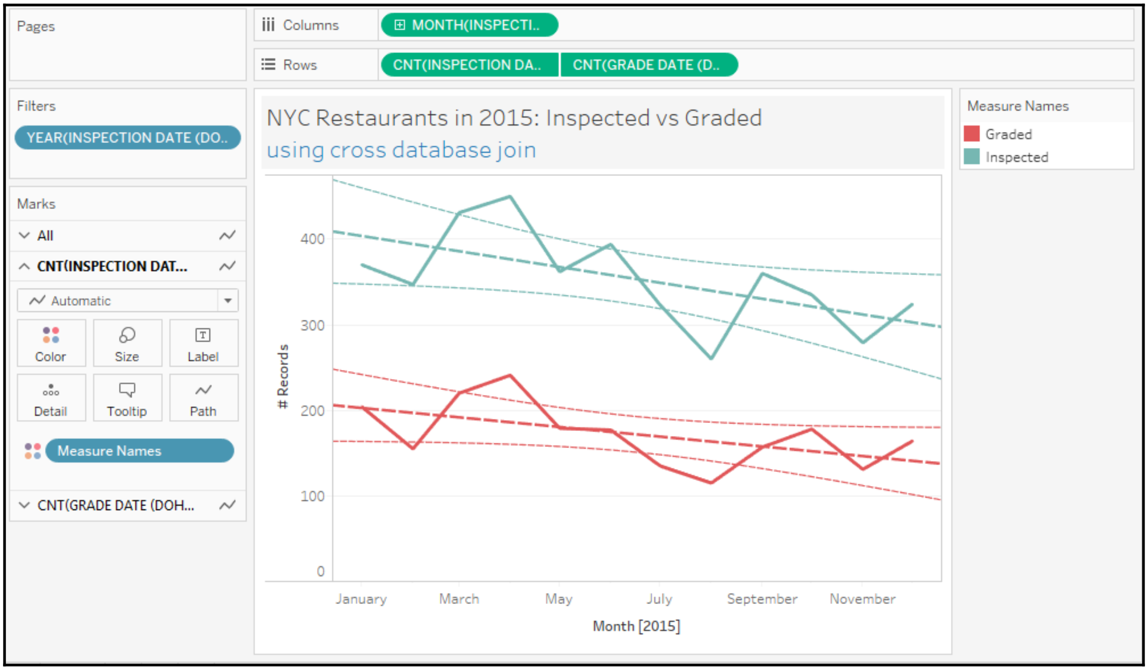
Data Source		DOHMH New York City Restaurant1
Inspection Date	=	INSPECTION DATE (DOHMH New York City Rest...
Camis	=	CAMIS (DOHMH New York City Restaurant1)
DBA	=	DBA (DOHMH New York City Restaurant1)
Boro	=	BORO (DOHMH New York City Restaurant1)
Zipcode	=	ZIPCODE (DOHMH New York City Restaurant1)
Violation Code	=	VIOLATION CODE (DOHMH New York City Rest...

The second one connects mainly based on the **GRADE DATE**. As with the previous join, we also still need to consider other fields in the join to avoid mismatching records:

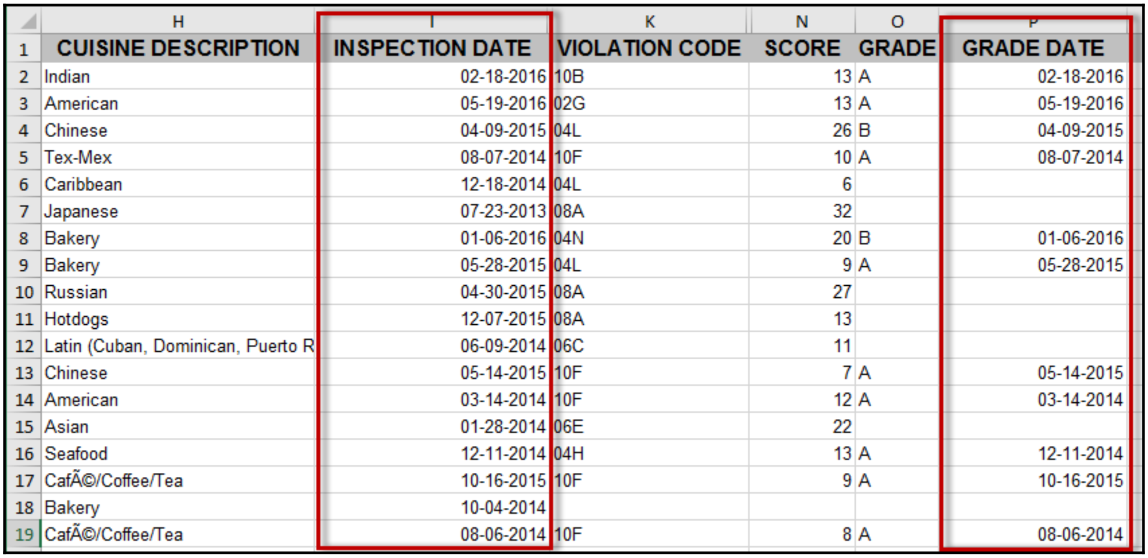

File Data Server Window Help
DOHMH New York City Restaurant+ (Mul...
Connection
Live
Connections
Add
NYC
Excel
NYC Inspected
Excel
NYC Graded
Excel
Sheets
Use Data Interpreter
Data Interpreter might be able to clean your Excel workbook.
DOHMH New Y... Restaurant
New Union
DOHMH New York City Rest...
DOHMH New York City Rest...
DOHMH New York City Rest...
Join
Inner
Left
Right
Full Outer
Sort
Abc
DOHMH New York
Camis
41714097
41253814
50005729
50009866

Data Source		DOHMH New York City Restaurant2
Grade Date	=	GRADE DATE (DOHMH New York City Restauran...
Camis	=	CAMIS (DOHMH New York City Restaurant2)
DBA	=	DBA (DOHMH New York City Restaurant2)
Inspection Type	=	INSPECTION TYPE (DOHMH New York City Rest...
Boro	=	BORO (DOHMH New York City Restaurant2)
Zipcode	=	ZIPCODE (DOHMH New York City Restaurant2)
Violation Code	=	VIOLATION CODE (DOHMH New York City Rest...
Add new join clause		

Once the connections are set up, we can create a similar visualization to the one we created using the legacy connection. The following visualization uses a slightly different approach. Since our measures come from different data sources, we are using a dual axis graph for the **COUNT of INSPECTION DATE**s from one data source, and **COUNT of GRADE DATE**s from another data source:



This will allow us to visualize how many restaurants were inspected and graded for a specific date:



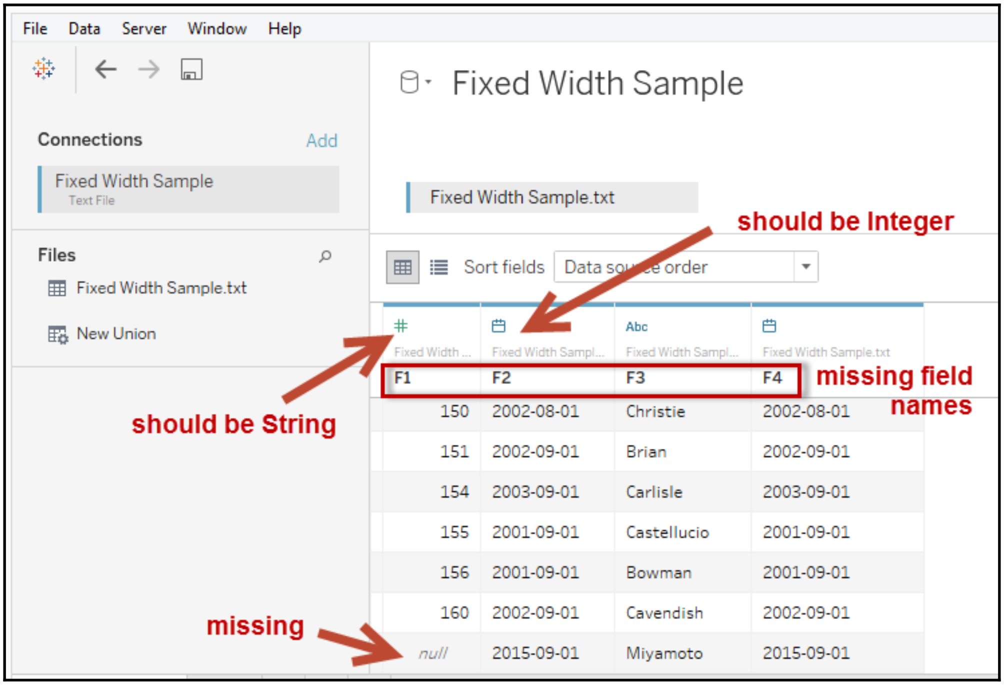
	H	I	K	N	O	P
1	CUISINE DESCRIPTION	INSPECTION DATE	VIOLATION CODE	SCORE	GRADE	GRADE DATE
2	Indian	02-18-2016	10B	13	A	02-18-2016
3	American	05-19-2016	02G	13	A	05-19-2016
4	Chinese	04-09-2015	04L	26	B	04-09-2015
5	Tex-Mex	08-07-2014	10F	10	A	08-07-2014
6	Caribbean	12-18-2014	04L	6		
7	Japanese	07-23-2013	08A	32		
8	Bakery	01-06-2016	04N	20	B	01-06-2016
9	Bakery	05-28-2015	04L	9	A	05-28-2015
10	Russian	04-30-2015	08A	27		
11	Hotdogs	12-07-2015	08A	13		
12	Latin (Cuban, Dominican, Puerto R	06-09-2014	06C	11		
13	Chinese	05-14-2015	10F	7	A	05-14-2015
14	American	03-14-2014	10F	12	A	03-14-2014
15	Asian	01-28-2014	06E	22		
16	Seafood	12-11-2014	04H	13	A	12-11-2014
17	CafÃ©/Coffee/Tea	10-16-2015	10F	9	A	10-16-2015
18	Bakery	10-04-2014				
19	CafÃ©/Coffee/Tea	08-06-2014	10F	8	A	08-06-2014

Using schema.ini to resolve data type issues

Connecting to text files can sometimes be more challenging than connecting to a database or server-based data source. Relational databases will typically have the data types and constraints built in. Tableau can read this metadata and interpret the correct types and settings for the data set.

Text files can be tricky. We usually need to identify delimiters (that is, how is one field separated from another). If we want headers, we will need to either manually assign them from within Tableau, or override them in a configuration file.

If we connect to the file from Tableau without a configuration (or `schema.ini`) file, this is what we will get:



F1	F2	F3	F4
150	2002-08-01	Christie	2002-08-01
151	2002-09-01	Brian	2002-09-01
154	2003-09-01	Carlisle	2003-09-01
155	2001-09-01	Castellucio	2001-09-01
156	2001-09-01	Bowman	2001-09-01
160	2002-09-01	Cavendish	2002-09-01
null	2015-09-01	Miyamoto	2015-09-01

should be Integer

missing field names

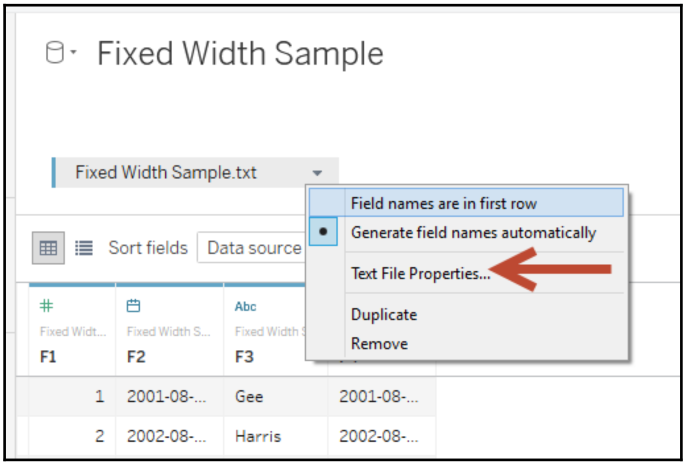
should be String

missing

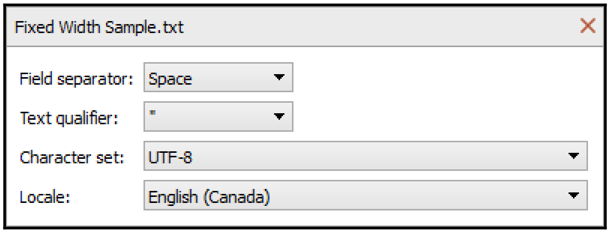
There are a few things that are incorrect or missing:

- The field names are missing.
- The first field contains a **null** value for the very last record, because Tableau assumes this field is numeric based on the first few rows. The last record has an alphanumeric value of C160, which is invalid for a numeric field. Show how to change with just default text driver properties clicking on the dropdown.
- The second field is interpreted as a date because the values, while numeric, can assume the format of *yyyymmdd*.

Tableau does allow us some flexibility when working with text files. When you click on the drop-down for the text file, there is an option for **Text File Properties**:



This provides another window that allows us to specify the field separators, text qualifiers (that is, character that encloses text values), character set, and locale:



Fixed Width Sample.txt

Field separator: Space

Text qualifier: "

Character set: UTF-8

Locale: English (Canada)

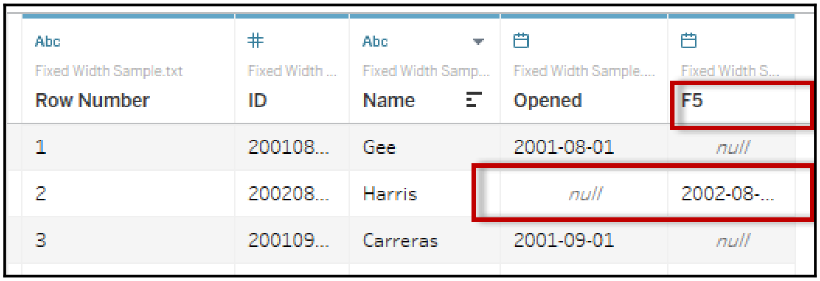
This still makes working with fixed width files without column headers a challenge. Microsoft recommends using `schema.ini` for all fixed length files. `schema.ini` provides a way to specify the data types and other configurations for the text file that Tableau can read. It does not solve all cases, but it can help with some.

The format, supported fields, and options for `schema.ini` are documented in the MSDN page called `Schema.ini` File (Text File Driver), which can be found at `http://bit.ly/msdn-schema-ini`.

Tableau also has a KB article called *Resolving Incorrect Data Type Issues Related to Jet*, which can be found at the following URL: http://bit.ly/tableau-jet-engine.

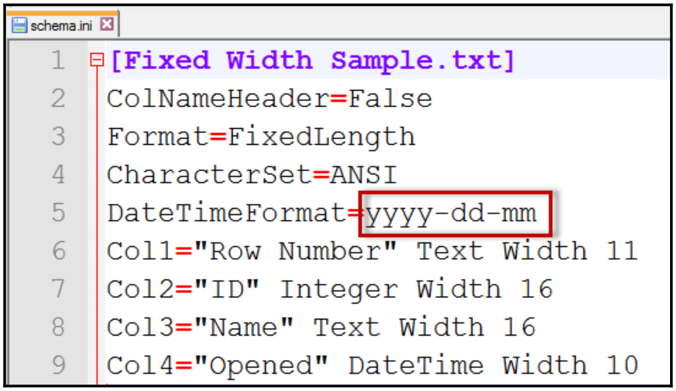
What we used in this recipe is one of the simpler text files that can be cleaned up using a `schema.ini` file. In reality, there are many limitations.

If we had spaces in the third column, for example, if record #2's name is *Harris Jr*, this is what we will get in Tableau even if we specified the width of the string in the `schema.ini` file:



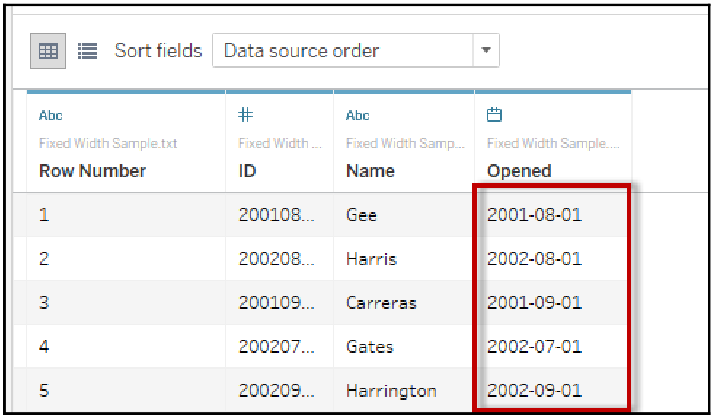
Abc	#	Abc	📅	📅
Fixed Width Sample.txt	Fixed Width ...	Fixed Width Samp...	Fixed Width Sample....	Fixed Width S...
Row Number	ID	Name	Opened	F5
1	200108...	Gee	2001-08-01	*null*
2	200208...	Harris	*null*	2002-08-...
3	200109...	Carreras	2001-09-01	*null*

What if the date format was `yyyy-dd-mm` and we specified it in the `schema.ini` like this?



```
[Fixed Width Sample.txt]
ColNameHeader=False
Format=FixedLength
CharacterSet=ANSI
DateTimeFormat=yyyy-dd-mm
Col1="Row Number" Text Width 11
Col2="ID" Integer Width 16
Col3="Name" Text Width 16
Col4="Opened" DateTime Width 10
```

Tableau still uses the date format `yyyy-mm-dd` and ignores the specification in the `schema.ini` file:



Sort fields Data source order

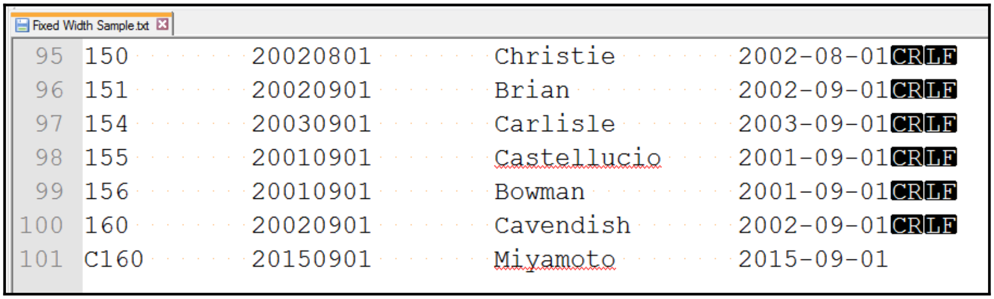
Abc Fixed Width Sample.txt Row Number	# Fixed Width ... ID	Abc Fixed Width Samp... Name	📅 Fixed Width Sample.... Opened
1	200108...	Gee	2001-08-01
2	200208...	Harris	2002-08-01
3	200109...	Carreras	2001-09-01
4	200207...	Gates	2002-07-01
5	200209...	Harrington	2002-09-01

There are other variations that demonstrate the limitations of `schema.ini`. Sometimes, the best way to approach data wrangling problems is to either export to another format that Tableau can more easily read, or to resort to other tools, or even scripting. For example, Python, R, or even PowerShell are great, powerful scripting tools that can give you much more flexibility with how to shape your data.

Let's use a `schema.ini` file to resolve the data types when we connect to a fixed width text file data source with four columns.

Download this chapter's files from the Packt website and use the file called `Fixed Width Sample.txt`.

This is what the file looks like when opened in a text editor showing special characters:



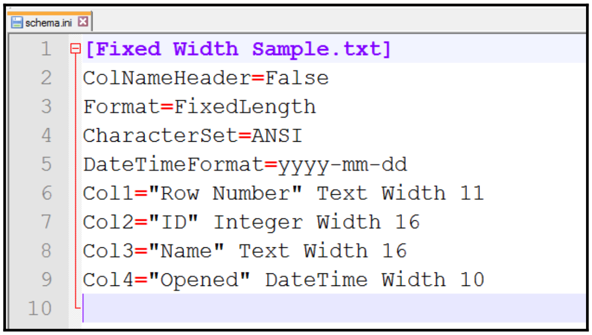
Fixed Width Sample.txt

```
150        20020801        Christie        2002-08-01CRLF
151        20020901        Brian           2002-09-01CRLF
154        20030901        Carlisle        2003-09-01CRLF
155        20010901        Castellucio     2001-09-01CRLF
156        20010901        Bowman          2001-09-01CRLF
160        20020901        Cavendish       2002-09-01CRLF
C160       20150901        Miyamoto        2015-09-01
```

Note that this file does not have any column headers. In addition, note the following:

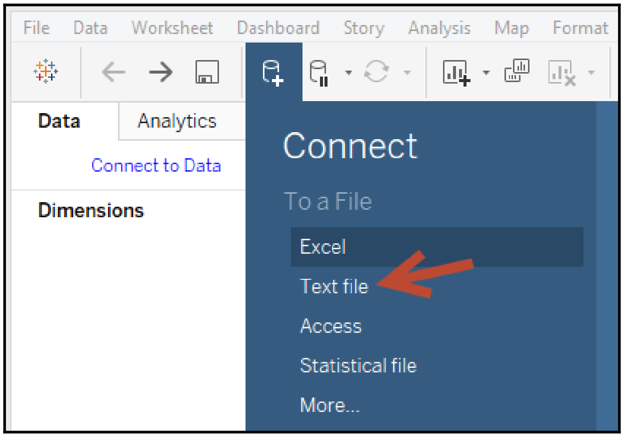
- The first column should be text
- The second column should be integers
- The third column should be text
- The fourth column should be dates

1. Create a text file with the following contents:

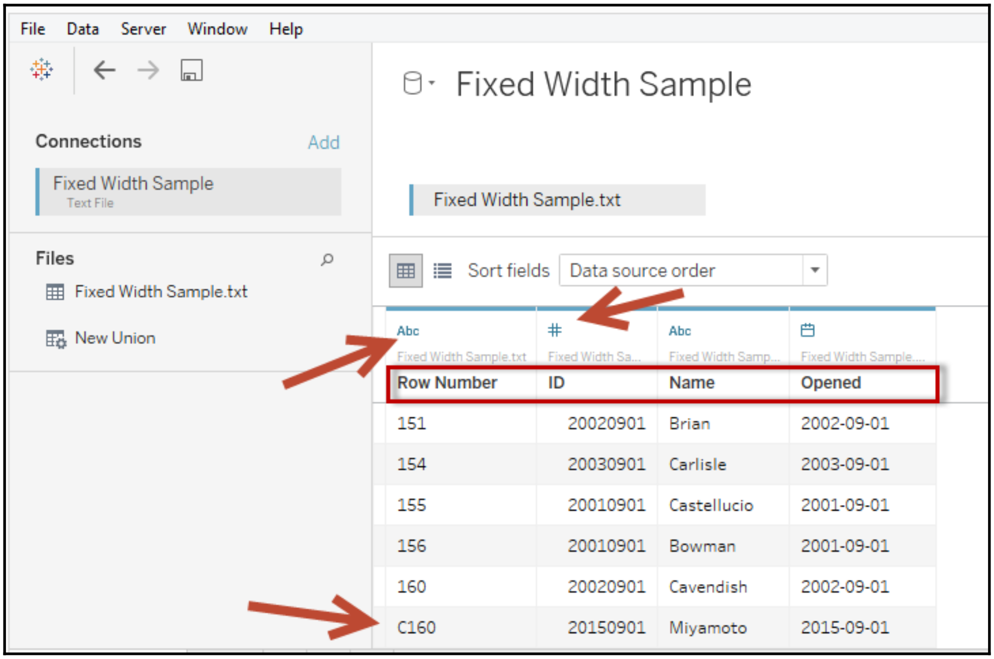


```
[Fixed Width Sample.txt]
ColNameHeader=False
Format=FixedLength
CharacterSet=ANSI
DateTimeFormat=yyyy-mm-dd
Col1="Row Number" Text Width 11
Col2="ID" Integer Width 16
Col3="Name" Text Width 16
Col4="Opened" DateTime Width 10
```

2. Save the file as `schema.ini` and save it in the same directory as the `Fixed Width Sample.txt` file.
3. Connect to the text file in Tableau:



4. Confirm that there are four fields in the Tableau preview window, with the same configuration as specified in the `schema.ini` file:
    - First field is text
    - Second field is number
    - Third field is text
    - Fourth field is date:

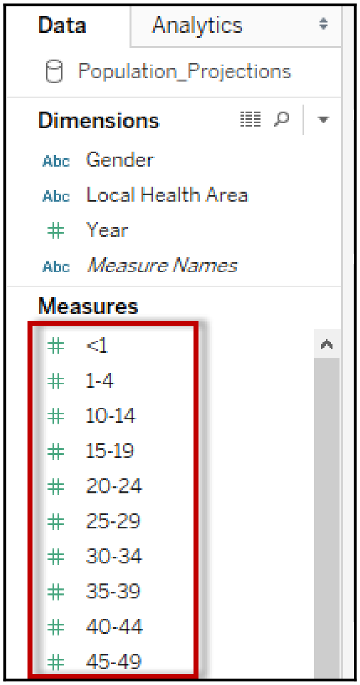


Row Number	ID	Name	Opened
151	20020901	Brian	2002-09-01
154	20030901	Carlisle	2003-09-01
155	20010901	Castellucio	2001-09-01
156	20010901	Bowman	2001-09-01
160	20020901	Cavendish	2002-09-01
C160	20150901	Miyamoto	2015-09-01

5. Add a new sheet and create your visualization using this data set.

Pivoting columns

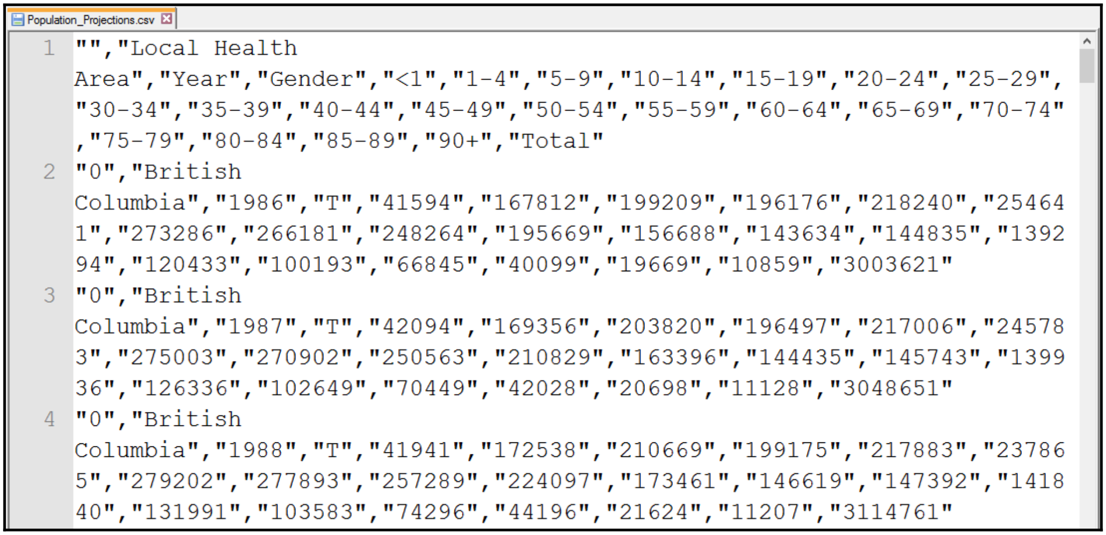
In the file that we are using, the measure field--population--is split by age group. Each population value for an age group is provided as a column, so we end up with multiple measures:



This format is hard to work with because all these measures are supposed to be a single measure. If we had a single measure for population values, and another dimension for age group, the analysis will be more flexible. We can slice and dice population by age group if we need to.

Tableau provides a way for us to shape this file by pivoting the values, using the original measure names as a dimension, and collecting all the population values into a single column. Although you may also be able to pivot at the data source level, it is great to have this capability within Tableau.

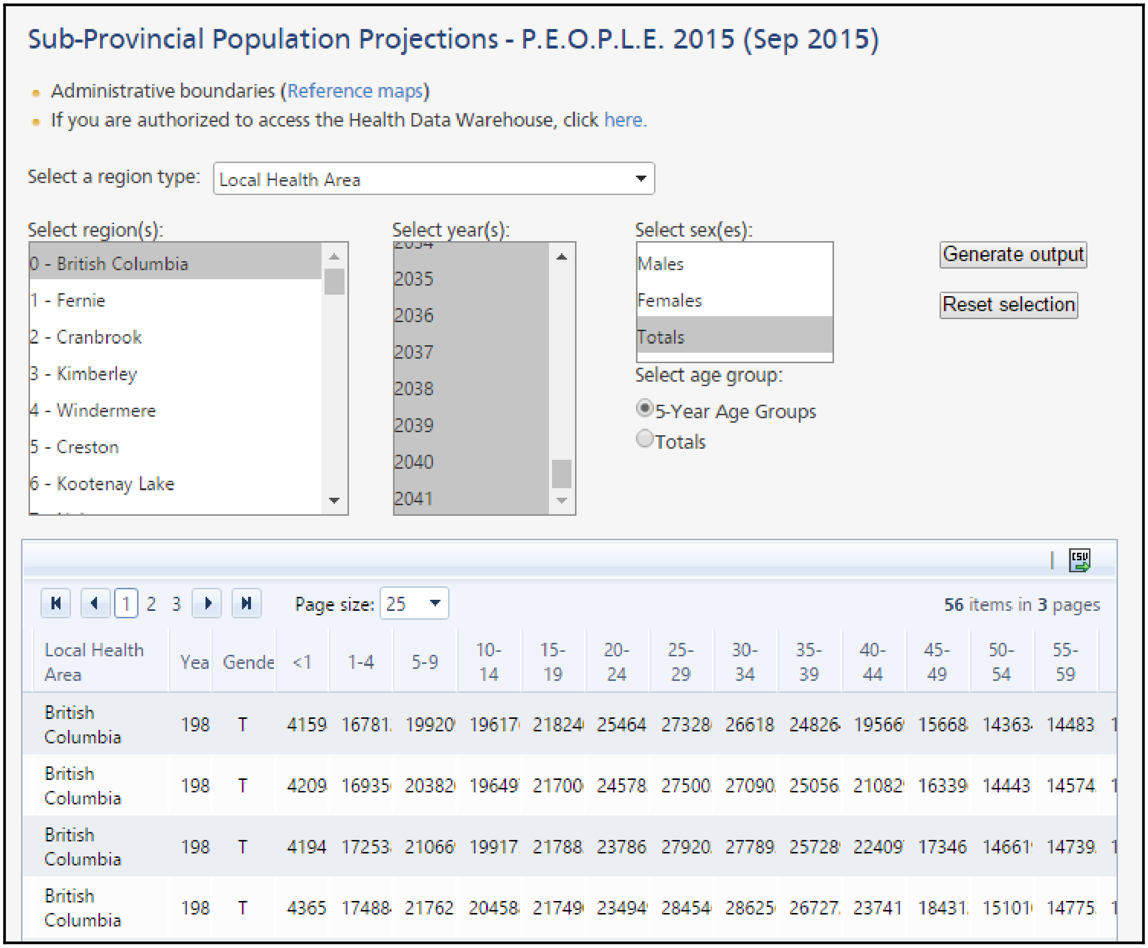
Let's prepare the data set and prepare the `.csv` file:



```
"","Local Health
Area","Year","Gender","<1","1-4","5-9","10-14","15-19","20-24","25-29",
"30-34","35-39","40-44","45-49","50-54","55-59","60-64","65-69","70-74"
,"75-79","80-84","85-89","90+","Total"
"0","British
Columbia","1986","T","41594","167812","199209","196176","218240","25464
1","273286","266181","248264","195669","156688","143634","144835","1392
94","120433","100193","66845","40099","19669","10859","3003621"
"0","British
Columbia","1987","T","42094","169356","203820","196497","217006","24578
3","275003","270902","250563","210829","163396","144435","145743","1399
36","126336","102649","70449","42028","20698","11128","3048651"
"0","British
Columbia","1988","T","41941","172538","210669","199175","217883","23786
5","279202","277893","257289","224097","173461","146619","147392","1418
40","131991","103583","74296","44196","21624","11207","3114761"
```

1. Download the file from BCStats using the following URL:
   `http://www.bcstats.gov.bc.ca/StatisticsBySubject/Demography/PopulationProjections.aspx`

2. When you download, make the following selections and click on **Generate Output**:
   - Select **British Columbia** for **Region**
   - Select all the years
   - Select **Totals**
   - Select **5-Year Age Groups**:



Sub-Provincial Population Projections - P.E.O.P.L.E. 2015 (Sep 2015)

- Administrative boundaries (Reference maps)
- If you are authorized to access the Health Data Warehouse, click here.

Select a region type: Local Health Area

Select region(s): 0 - British Columbia

Select year(s): 2035, 2036, 2037, 2038, 2039, 2040, 2041

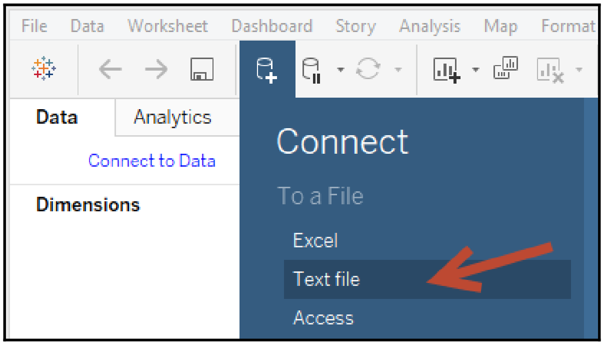
Select sex(es): Totals

Select age group: ● 5-Year Age Groups ○ Totals

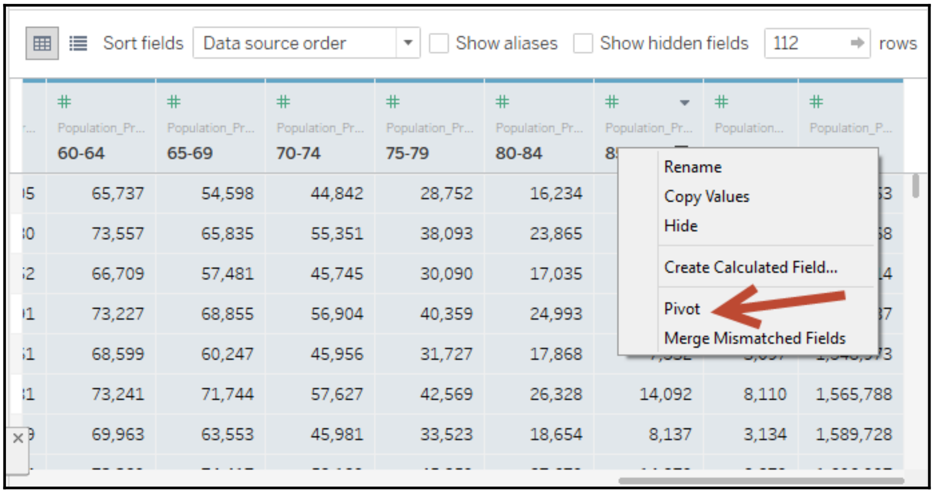
Local Health Area	Yea	Gende	<1	1-4	5-9	10-14	15-19	20-24	25-29	30-34	35-39	40-44	45-49	50-54	55-59	
British Columbia	198	T	4159	16781	19920	19617	21824	25464	27328	26618	24826	19566	15668	14363	14483	1
British Columbia	198	T	4209	16935	20382	19649	21700	24578	27500	27090	25056	21082	16339	14443	14574	1
British Columbia	198	T	4194	17253	21066	19917	21788	23786	27920	27789	25728	22409	17346	14661	14739	1
British Columbia	198	T	4365	17488	21762	20458	21749	23494	28454	28625	26727	23741	18431	15101	14775	1

3. Beside the results pane, click on the CSV icon at the top-right corner of the results pane to download the `.csv` file. Save the file as `Population_Projections.csv`.

4. Click on the **New Data Source** icon and connect to the text file in this recipe:



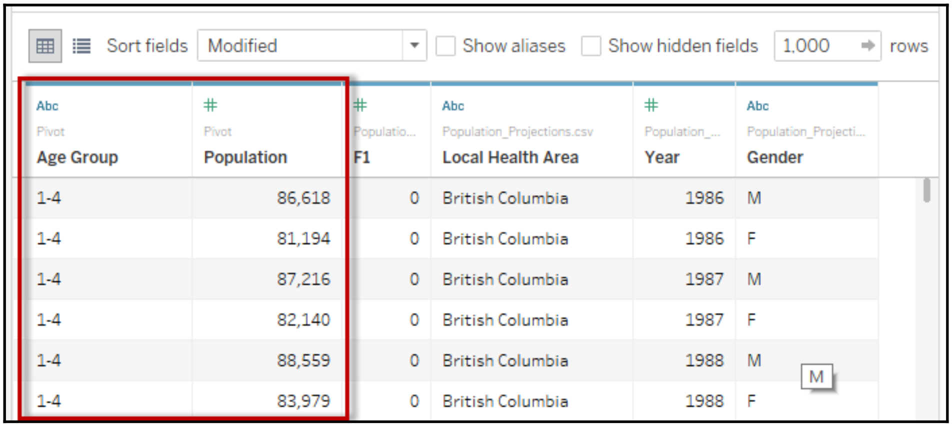
5. Select all the age groups that are presented as individual columns.
6. While all the age group columns are selected, right-click on one of the selected fields and choose **Pivot**:



Population_Pr... 60-64	Population_Pr... 65-69	Population_Pr... 70-74	Population_Pr... 75-79	Population_Pr... 80-84	Population_Pr... 85	Population...	Population_P...
65,737	54,598	44,842	28,752	16,234			53
73,557	65,835	55,351	38,093	23,865			58
66,709	57,481	45,745	30,090	17,035			4
73,227	68,855	56,904	40,359	24,993			37
68,599	60,247	45,956	31,727	17,868			73
73,241	71,744	57,627	42,569	26,328	14,092	8,110	1,565,788
69,963	63,553	45,981	33,523	18,654	8,137	3,134	1,589,728

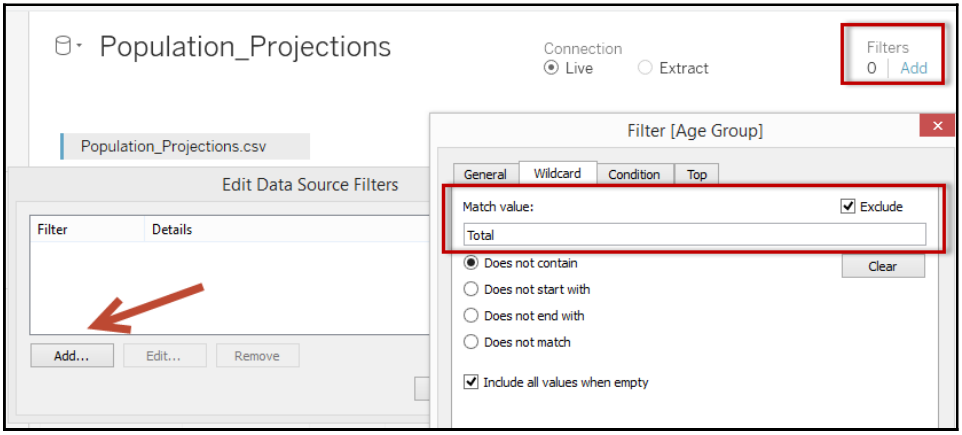
7. Right-click on the newly created **Pivot Field Names** field and choose **Rename**. Rename this field **Age Group**.

8. Right-click on the newly created **Pivot Field Values** field and choose **Rename**. Rename this field **Population**:



Abc Pivot Age Group	# Pivot Population	# Populatio... F1	Abc Population_Projections.csv Local Health Area	# Population_... Year	Abc Population_Projecti... Gender
1-4	86,618	0	British Columbia	1986	M
1-4	81,194	0	British Columbia	1986	F
1-4	87,216	0	British Columbia	1987	M
1-4	82,140	0	British Columbia	1987	F
1-4	88,559	0	British Columbia	1988	M
1-4	83,979	0	British Columbia	1988	F

9. Under **Filters**, click on **Add**.
10. In the **Edit Data Source Filters** window, click on **Add**.
11. In the **Age Group filter** window, select the **Wildcard** tab.
12. Type `Total` under **Match value** and check the **Exclude** checkbox:



Match value: Total

☑ Exclude

● Does not contain
○ Does not start with
○ Does not end with
○ Does not match

☑ Include all values when empty

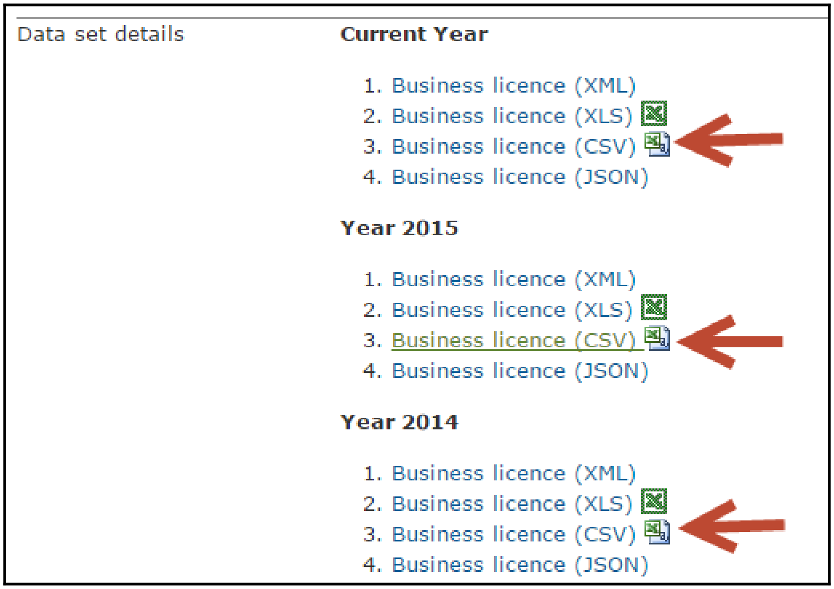
13. Click **OK** when done.
14. Add a new sheet and create your visualization using this data set.

Using unions

A union operation allows multiple sets of data to be appended to each other, that is, new records will be added to the end of the existing set of records.

Let's combine a number of **comma-separated value** (**CSV**) files into a single data set in Tableau:

1. Download the business license files from the City of Vancouver's website from `http://data.vancouver.ca/datacatalogue/businessLicence.htm`:



Data set details

Current Year

1. Business licence (XML)
2. Business licence (XLS)
3. Business licence (CSV)
4. Business licence (JSON)

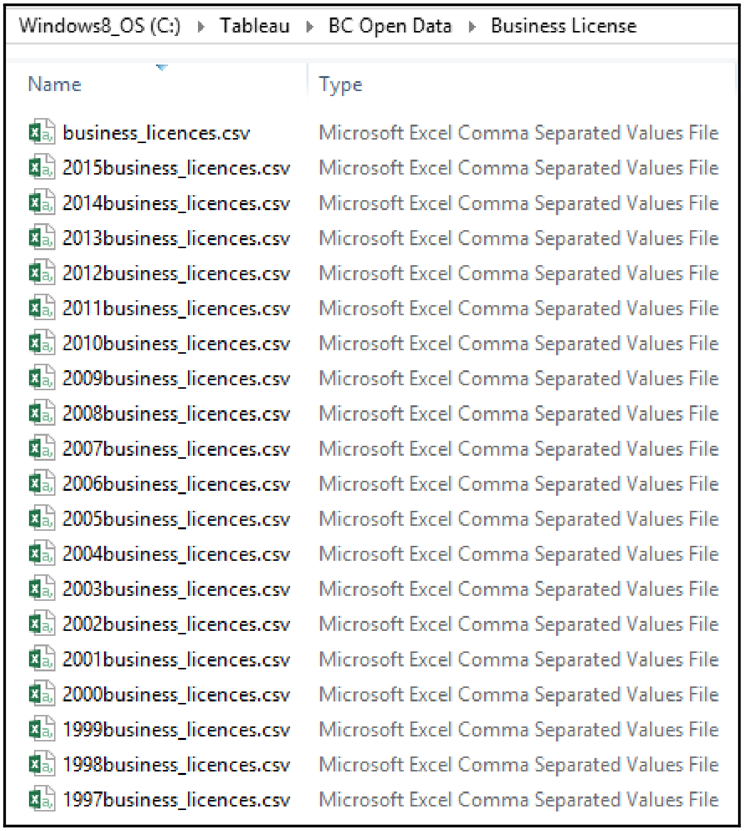
Year 2015

1. Business licence (XML)
2. Business licence (XLS)
3. Business licence (CSV)
4. Business licence (JSON)

Year 2014

1. Business licence (XML)
2. Business licence (XLS)
3. Business licence (CSV)
4. Business licence (JSON)

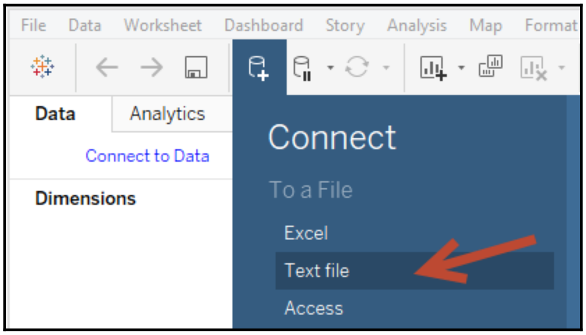
2. Download the CSV version, and save all the files in a local directory in your computer:



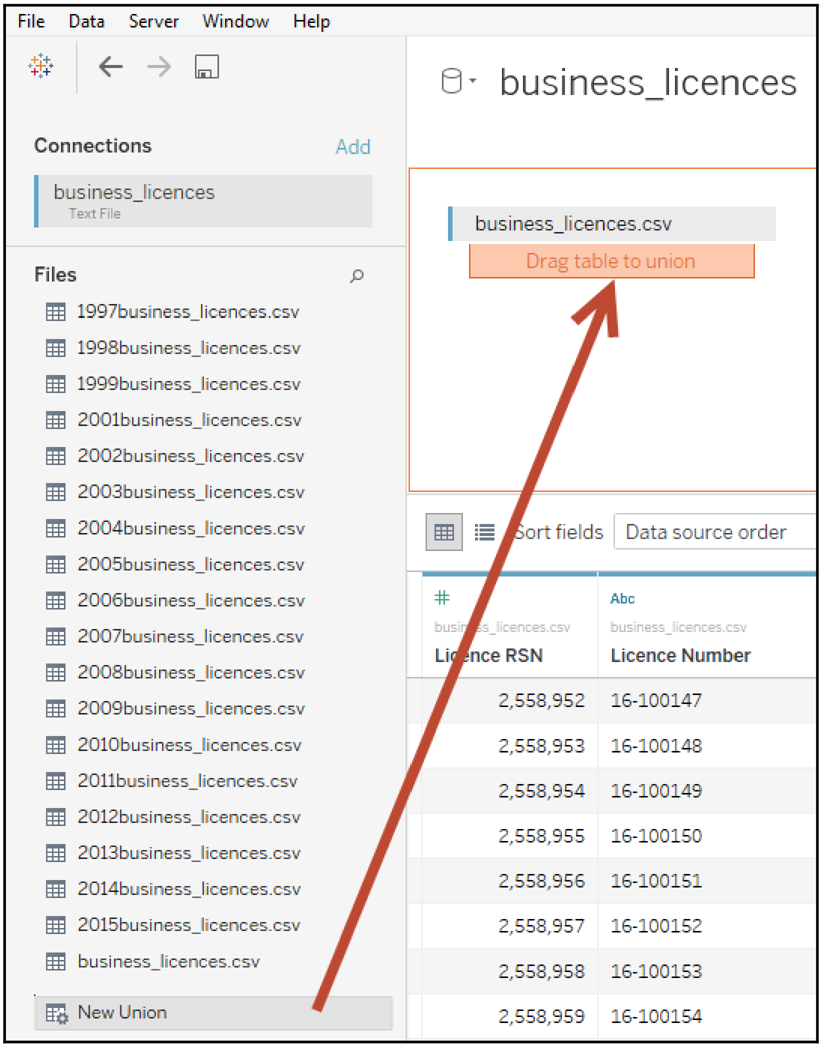
Windows8_OS (C:) ▸ Tableau ▸ BC Open Data ▸ Business License

Name	Type
business_licences.csv	Microsoft Excel Comma Separated Values File
2015business_licences.csv	Microsoft Excel Comma Separated Values File
2014business_licences.csv	Microsoft Excel Comma Separated Values File
2013business_licences.csv	Microsoft Excel Comma Separated Values File
2012business_licences.csv	Microsoft Excel Comma Separated Values File
2011business_licences.csv	Microsoft Excel Comma Separated Values File
2010business_licences.csv	Microsoft Excel Comma Separated Values File
2009business_licences.csv	Microsoft Excel Comma Separated Values File
2008business_licences.csv	Microsoft Excel Comma Separated Values File
2007business_licences.csv	Microsoft Excel Comma Separated Values File
2006business_licences.csv	Microsoft Excel Comma Separated Values File
2005business_licences.csv	Microsoft Excel Comma Separated Values File
2004business_licences.csv	Microsoft Excel Comma Separated Values File
2003business_licences.csv	Microsoft Excel Comma Separated Values File
2002business_licences.csv	Microsoft Excel Comma Separated Values File
2001business_licences.csv	Microsoft Excel Comma Separated Values File
2000business_licences.csv	Microsoft Excel Comma Separated Values File
1999business_licences.csv	Microsoft Excel Comma Separated Values File
1998business_licences.csv	Microsoft Excel Comma Separated Values File
1997business_licences.csv	Microsoft Excel Comma Separated Values File

3. Click on the **New Data Source** icon and connect to `business_licenses.csv`, which contains the most recent year's records:

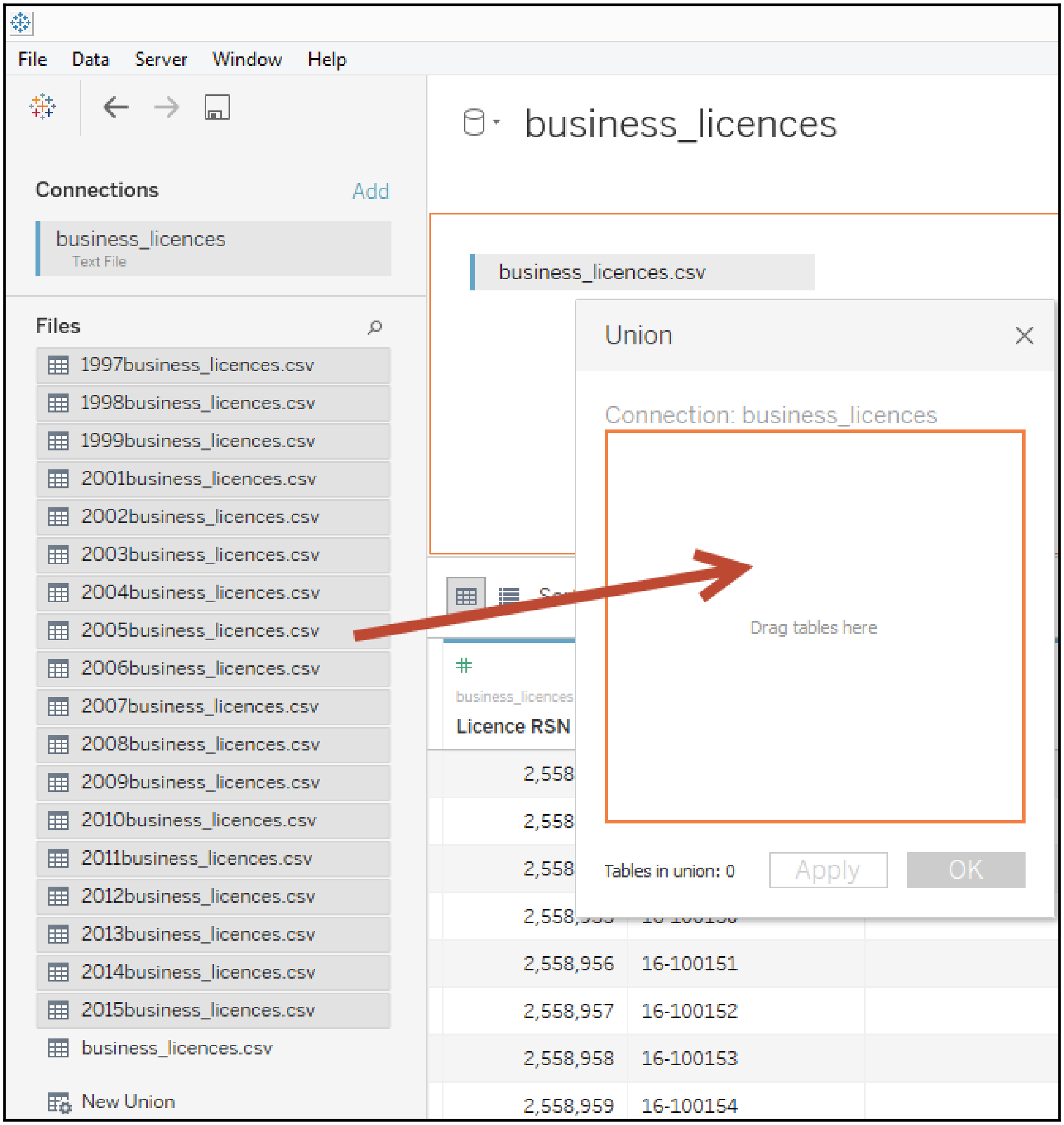


4. Drag **New Union** to just underneath `business_licenses.csv` until you see the **Drag** table to union message:

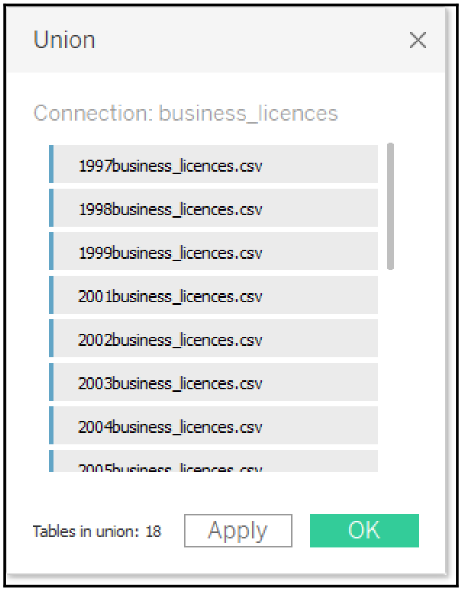

File
Data
Server
Window
Help
business_licences
Connections
Add
business_licences
Text File
Files
1997business_licences.csv
1998business_licences.csv
1999business_licences.csv
2001business_licences.csv
2002business_licences.csv
2003business_licences.csv
2004business_licences.csv
2005business_licences.csv
2006business_licences.csv
2007business_licences.csv
2008business_licences.csv
2009business_licences.csv
2010business_licences.csv
2011business_licences.csv
2012business_licences.csv
2013business_licences.csv
2014business_licences.csv
2015business_licences.csv
business_licences.csv
New Union
business_licences.csv
Drag table to union
Sort fields
Data source order

# business_licences.csv Licence RSN	Abc business_licences.csv Licence Number
2,558,952	16-100147
2,558,953	16-100148
2,558,954	16-100149
2,558,955	16-100150
2,558,956	16-100151
2,558,957	16-100152
2,558,958	16-100153
2,558,959	16-100154

5. Select all other CSV files from the **Files** pane and drag them to the **Union** window:


File
Data
Server
Window
Help
business_licences
Connections
Add
business_licences
Text File
Files
1997business_licences.csv
1998business_licences.csv
1999business_licences.csv
2001business_licences.csv
2002business_licences.csv
2003business_licences.csv
2004business_licences.csv
2005business_licences.csv
2006business_licences.csv
2007business_licences.csv
2008business_licences.csv
2009business_licences.csv
2010business_licences.csv
2011business_licences.csv
2012business_licences.csv
2013business_licences.csv
2014business_licences.csv
2015business_licences.csv
business_licences.csv
New Union
business_licences.csv
Union
Connection: business_licences
Drag tables here
Tables in union: 0
Apply
OK
business_licences
Licence RSN
2,558
2,558
2,558
2,558,956
16-100151
2,558,957
16-100152
2,558,958
16-100153
2,558,959
16-100154

6. Click on **OK** after you confirm that all the files have been added to the **Union** window:



Union

Connection: business_licences

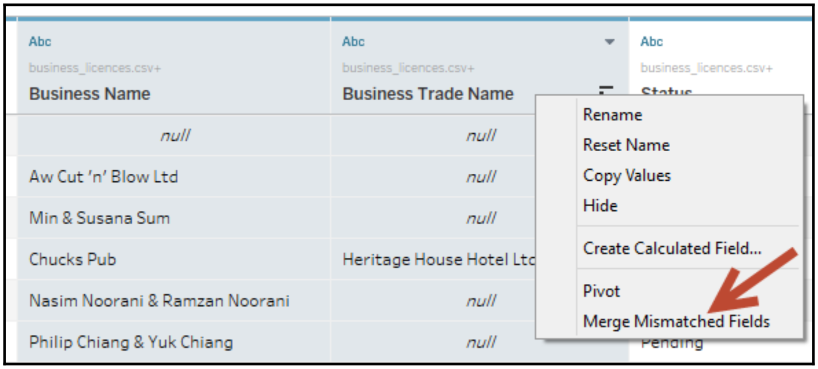
1997business_licences.csv
1998business_licences.csv
1999business_licences.csv
2001business_licences.csv
2002business_licences.csv
2003business_licences.csv
2004business_licences.csv
2005business_licences.csv

Tables in union: 18

A union in relational databases requires what is called union compatibility. This means the two sets of records need to have the same number of columns and similar data types.

In Tableau, the union operation does not necessarily require union compatibility. If some of the incoming fields do not match the existing fields, the mismatched fields will simply have null values.

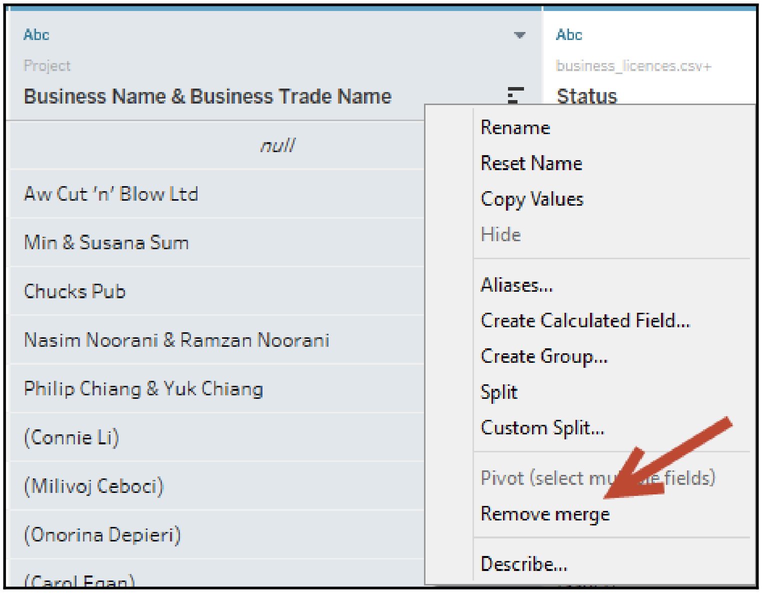
For example, if in some of our files, the **Business Name** field was called **Business Trade Name** instead, we can use Tableau's **Merge Mismatched Field** operation:



Abc business_licences.csv+ Business Name	Abc business_licences.csv+ Business Trade Name	Abc business_licences.csv+ Status
null	*null*	
Aw Cut 'n' Blow Ltd	*null*	
Min & Susana Sum	*null*	
Chucks Pub	Heritage House Hotel Ltd	
Nasim Noorani & Ramzan Noorani	*null*	
Philip Chiang & Yuk Chiang	*null*	Pending

What this operation does is combine the fields into a single field in the resulting data set. It will take the first non-null value for this new combined field. Thus, we have to take care to ensure that the fields are indeed supposed to be the same but just named differently; otherwise, we risk losing information.

Should you need to undo the merge, Tableau also provides a way to remove the merge:



Abc Project Business Name & Business Trade Name	Abc business_licences.csv+ Status
null	
Aw Cut 'n' Blow Ltd	
Min & Susana Sum	
Chucks Pub	
Nasim Noorani & Ramzan Noorani	
Philip Chiang & Yuk Chiang	
(Connie Li)	
(Milivoj Ceboci)	
(Onorina Depieri)	
(Carol Egan)	

When we union files or worksheets, Tableau adds metadata fields in the resulting data set. Tableau has the **Table Name** dimension for text files, which uses the original file name as the value:

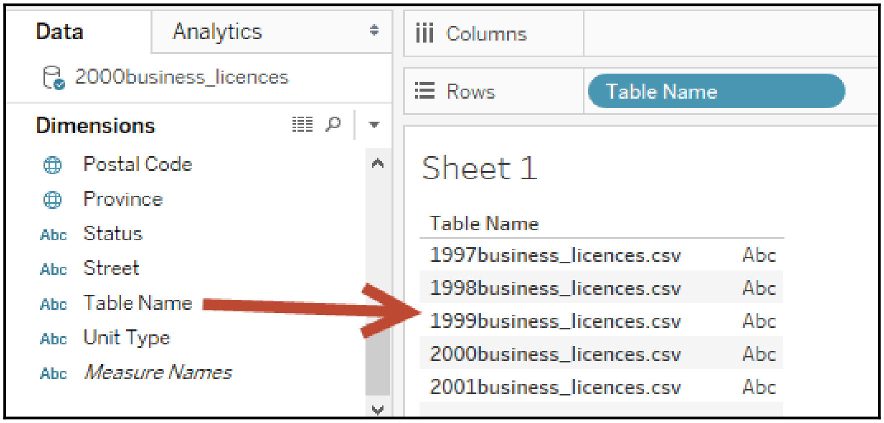
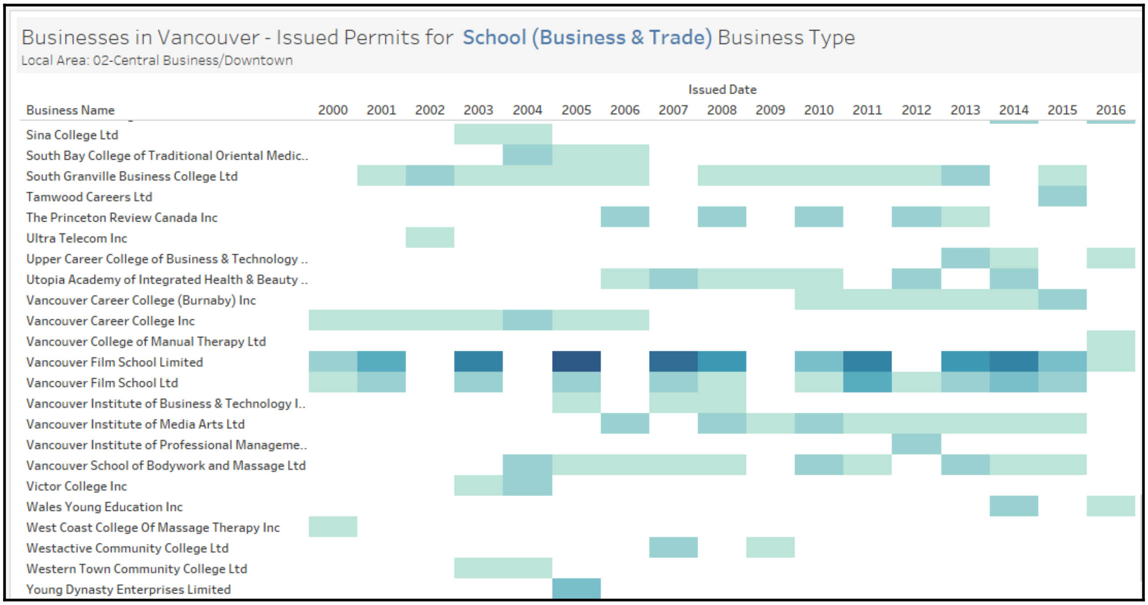


Table Name	
1997business_licences.csv	Abc
1998business_licences.csv	Abc
1999business_licences.csv	Abc
2000business_licences.csv	Abc
2001business_licences.csv	Abc

After we union our files, we can do our analysis. One possibility is a heat map. In the view below, we have a heat map of issued business licenses in downtown Vancouver. This type of visualization can indicate how long businesses have been operating:

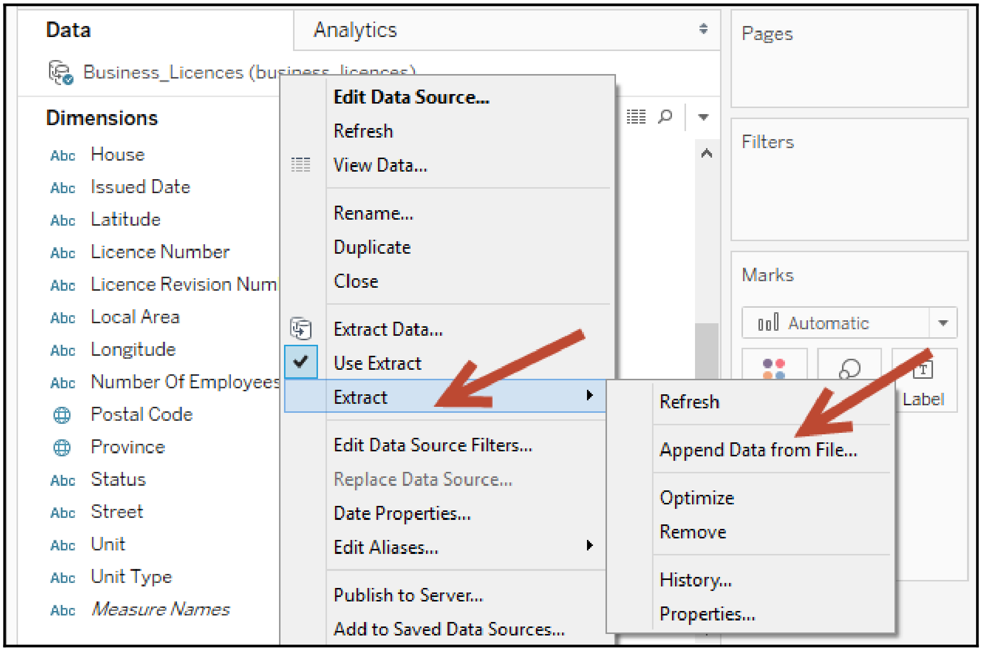


In the past, a union with multiple worksheets in the same Excel workbook could be done using a custom SQL in Excel, if using the legacy Jet connection.

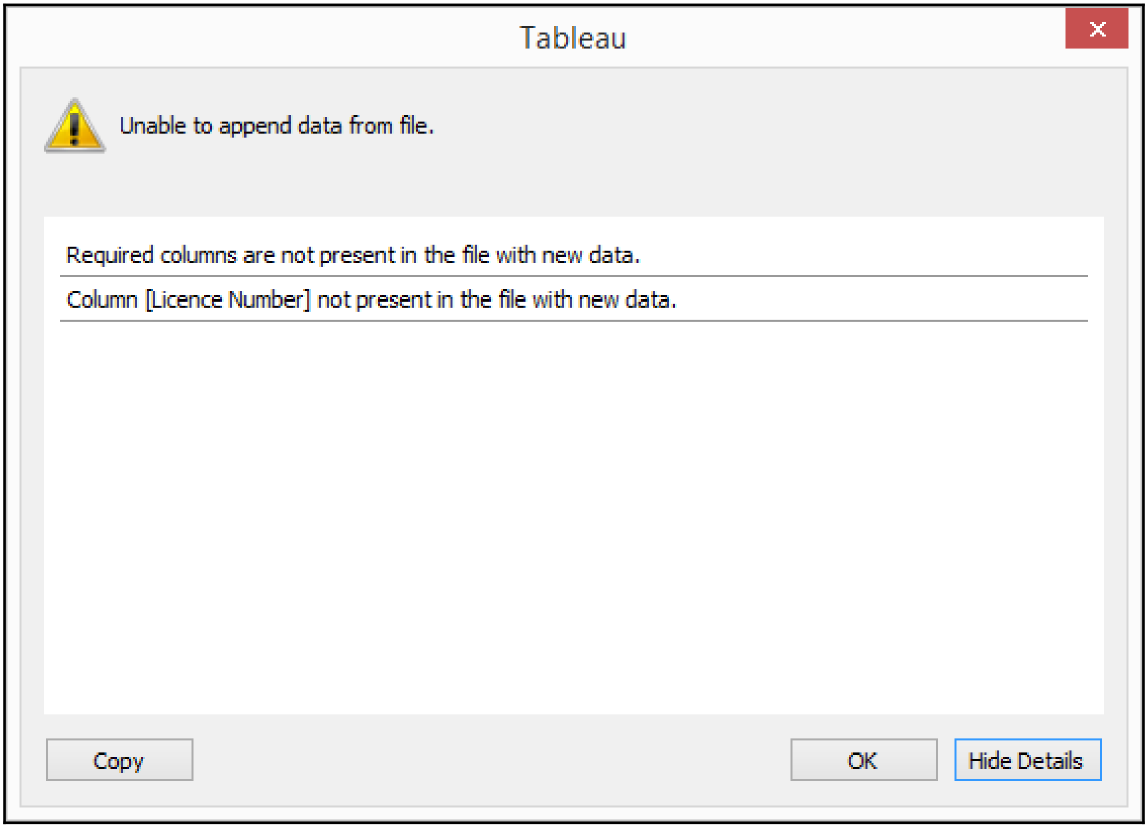
In Tableau 10, the union operator is baked into the product. In this version, union works with text files (including the `.csv` and `.txt` file extensions) and multiple worksheets in Excel if saved in a single workbook. What if you need to combine multiple Excel files?

One improvement that is being promised in the future, and was showcased in the 2015 Tableau conference, is a wildcard union. This allows the union to operate on multiple files based on specific patterns on the filename. While not available in the initial release of Tableau 10, this will for sure be a much-awaited feature improvement for this operator.

A possible alternative to adding multiple Excel files is using data extracts. When you create an extract, you can append additional records from another file:



This is more restrictive than the union operator because you need to ensure the worksheet names are the same. You also need to ensure union compatibility; otherwise, you may encounter errors during the extract process. The following error is produced by the field name mismatch between the original file in the extraction and the incoming field names in the file being appended:



Unable to append data from file.

Required columns are not present in the file with new data.

Column [Licence Number] not present in the file with new data.

This field mismatch issue can be resolved in the new Tableau 10 Merge Mismatched Fields feature.

Learn more about the union operator from the Tableau online documentation: `https://onlinehelp.tableau.com/current/pro/desktop/en-us/union.html`

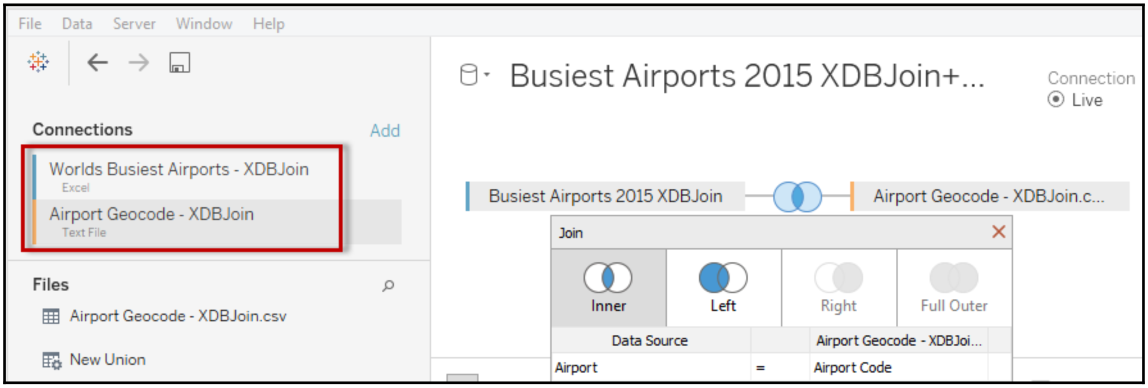
Using join

A join is primarily a relational database concept that allows you to combine records from different tables using common fields. When data sets are joined, all fields are combined based on the join conditions provided.

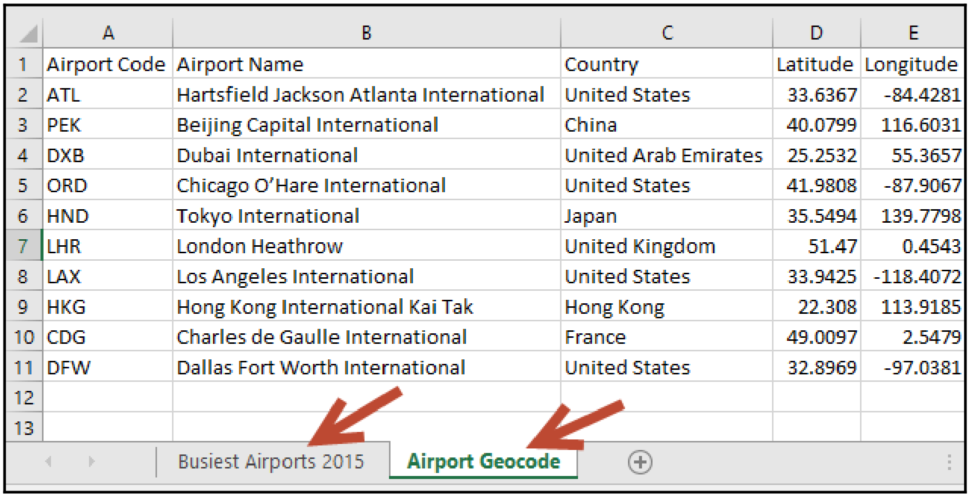
Joins are fundamentally different from unions. In unions, the record sets are stacked on top of each other, thus producing a taller result set. A join works by combining records and fields horizontally based on common fields, thus creating wider data sets that have all the combined fields together. A join also does not require union compatibility.

Before Tableau 10, joins were limited to combining tables from the same data source, that is, the tables needed to be using a single data connection. Tableau v10 adds flexibility to the join operation by allowing cross-database joins. Tables are no longer restricted to coming from the same data source. Joins can be done on file-based data sources as well. In Excel files, each tab or worksheet acts like a table with records. If your data source is text files, each file in a folder is considered a table.

In the following example, we can see that there are two color-coded connections on the left-hand pane. One is an Excel connection, and the other is a text file connection. In the middle connection window, we can see that the join operation was allowed between the two data sources:


File
Data
Server
Window
Help
Connections
Add
Worlds Busiest Airports - XDBJoin
Excel
Airport Geocode - XDBJoin
Text File
Files
Airport Geocode - XDBJoin.csv
New Union
Busiest Airports 2015 XDBJoin+...
Connection
Live
Busiest Airports 2015 XDBJoin
Airport Geocode - XDBJoin.c...
Join
Inner
Left
Right
Full Outer
Data Source
Airport Geocode - XDBJoi...
Airport
=
Airport Code

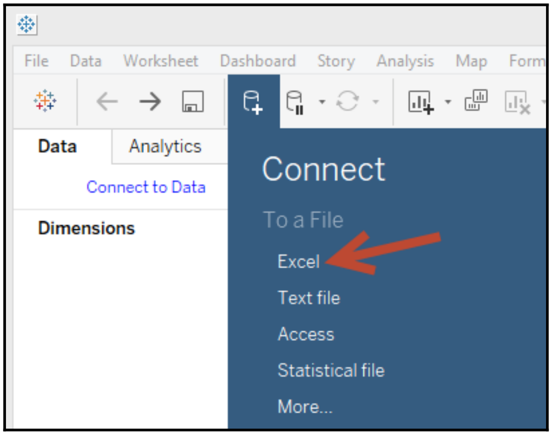
Let's combine the fields in two different Excel worksheets into one:



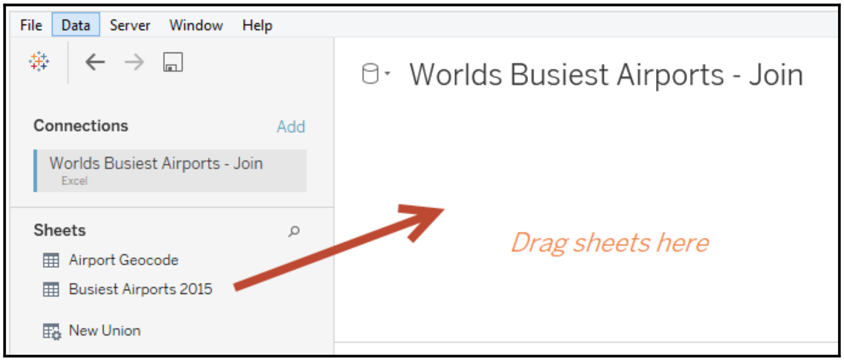
	A	B	C	D	E
1	Airport Code	Airport Name	Country	Latitude	Longitude
2	ATL	Hartsfield Jackson Atlanta International	United States	33.6367	-84.4281
3	PEK	Beijing Capital International	China	40.0799	116.6031
4	DXB	Dubai International	United Arab Emirates	25.2532	55.3657
5	ORD	Chicago O'Hare International	United States	41.9808	-87.9067
6	HND	Tokyo International	Japan	35.5494	139.7798
7	LHR	London Heathrow	United Kingdom	51.47	0.4543
8	LAX	Los Angeles International	United States	33.9425	-118.4072
9	HKG	Hong Kong International Kai Tak	Hong Kong	22.308	113.9185
10	CDG	Charles de Gaulle International	France	49.0097	2.5479
11	DFW	Dallas Fort Worth International	United States	32.8969	-97.0381
12					
13					

Download this chapter's files from the Packt website and use the file called `Worlds Busiest Airports-Join.xls`.

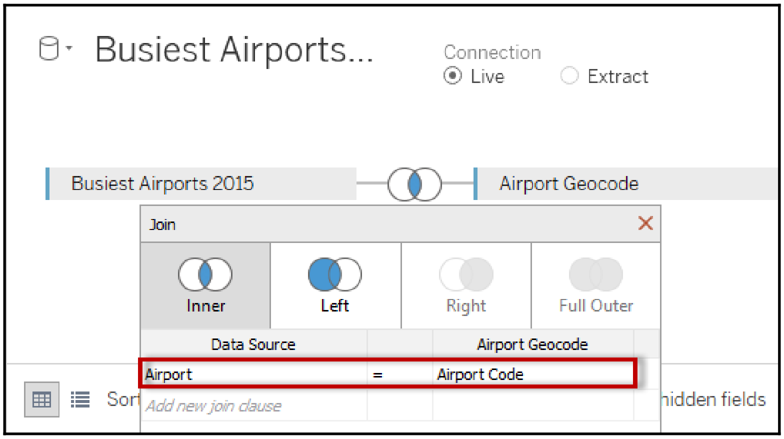
1. Connect to the **Excel** file in this recipe. Make sure you choose Excel from the **To a File** section:



2. Drag **Busiest Airports** 2015 from the **sheets** section to the data connection window:



3. Drag **Airport Geocode** to the right of **Busiest Airports** 2015 in the data connection window.
4. In the **Join** window that comes up, choose **Airport** from **Busiest Airports 2015** to match up to the **Airport Code** field from the **Airport Geocode** sheet:

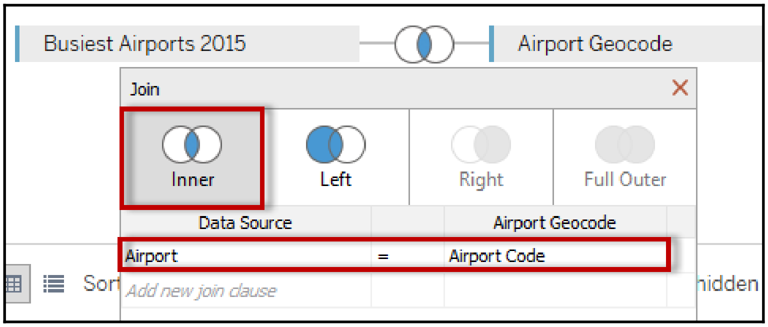


Connection: ● Live ○ Extract

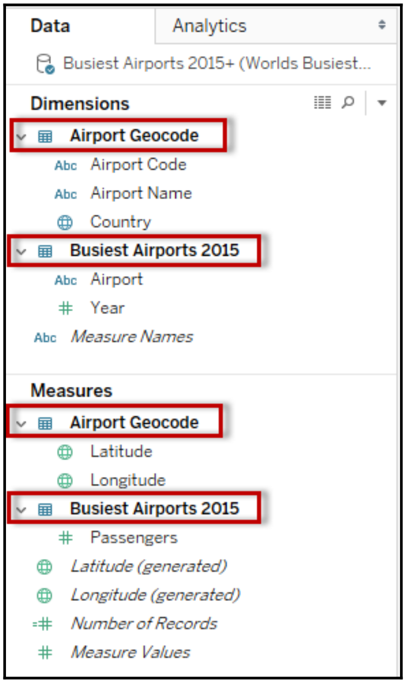
Data Source		Airport Geocode
Airport	=	Airport Code
Add new join clause		

5. Add a new sheet and create your visualization using this data set.

We combined two worksheets from the same Excel workbook. Records in both worksheets will be combined only if the **Airport** field from **Busiest Airports 2015** has the same value as the **Airport Code** field in the **Airport Geocode** worksheet. This join based on the equality of values is also called an equi-join:



Once the fields are joined, you will find the fields from both worksheets represented in the sidebar. Fields are grouped based on their source:



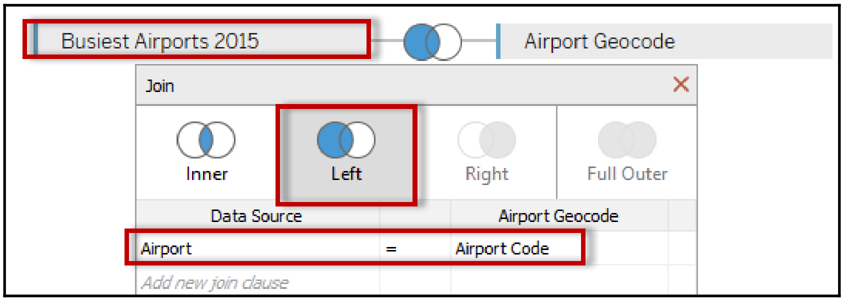
In general, we have two types of join: `inner` and `outer joins`.

`Inner joins` find matching values from both tables based on the join condition. The join condition is not always based on equality. There are cases where you may use other operators, such as greater than (>), greater than or equal to (>=), less than (<), less than or equal to (<=), or even not equal to (<>). Depending on the data source, some of these operators may not be supported.

`Outer joins`, also called preserving joins, preserve one or both sides of the tables as well as matching records. Outer joins can be further classified as `left outer`, `right outer`, and `full outer`. Some data sources do not support certain types of outer joins. Outer joins are positional; the placement of the tables relative to the JOIN operator affects the results.

A `left outer join` preserves the table to the left of the join operator and finds the matching values from the table on the right side of the operator. If a record on the left table being preserved does not have a matching value in the right table, that record is preserved but the fields from the other table will show NULL. A NULL value means the absence of value.

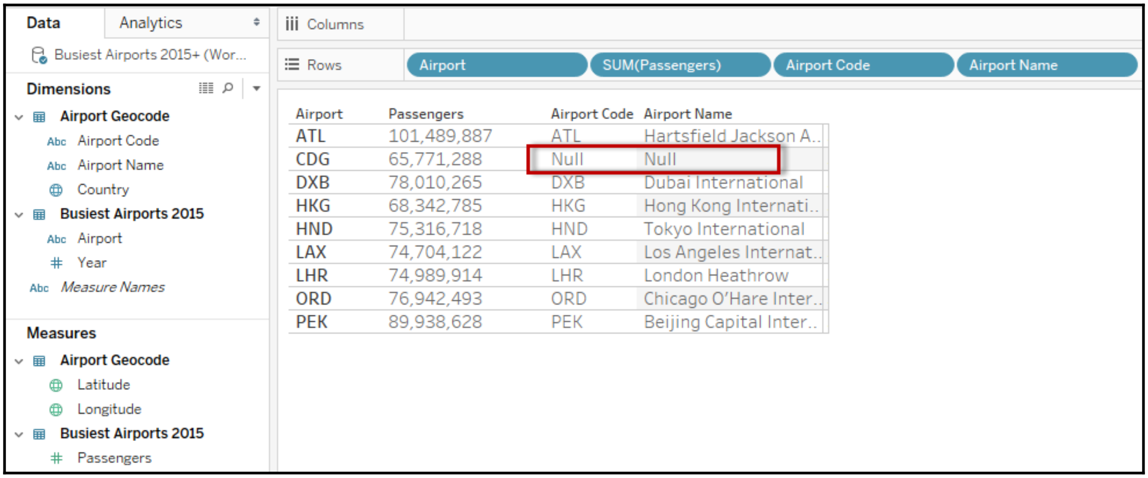
Here is an example of a `LEFT OUTER JOIN` using our worksheet in this recipe. The records in the table to the left, **Busiest Airports 2015**, are matched up to the records to the right, **Airport Geocode**, based on **Airport** and **Airport Code** fields respectively:



Busiest Airports 2015 — Airport Geocode

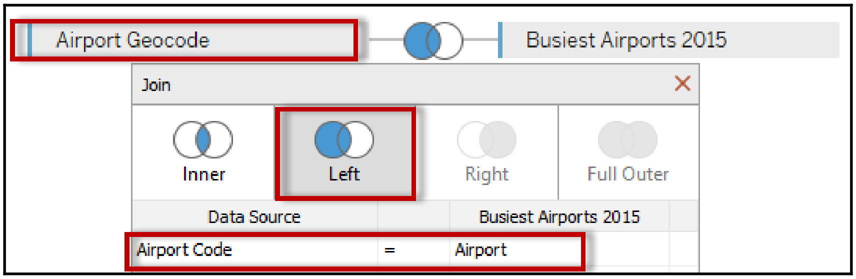
Data Source		Airport Geocode
Airport	=	Airport Code
Add new join clause		

Busiest Airports 2015 has a record for **Airport** value **CDG**, but this **Airport Code** does not exist in the **Airport Geocode** worksheet. Hence, as can be seen in the following screenshot, the corresponding **Airport Geocode** fields are reporting **Null** for the **CDG** airport:



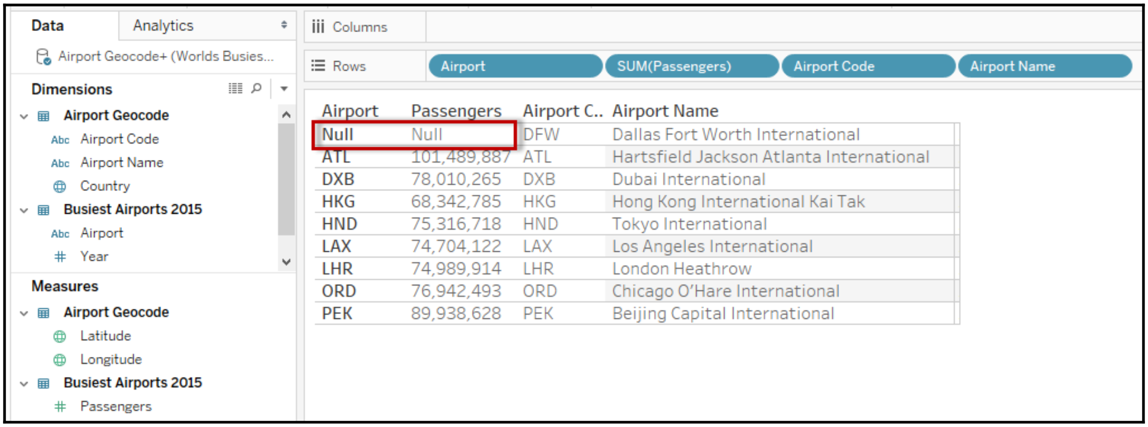
Airport	Passengers	Airport Code	Airport Name
ATL	101,489,887	ATL	Hartsfield Jackson A..
CDG	65,771,288	Null	Null
DXB	78,010,265	DXB	Dubai International
HKG	68,342,785	HKG	Hong Kong Internati..
HND	75,316,718	HND	Tokyo International
LAX	74,704,122	LAX	Los Angeles Internat..
LHR	74,989,914	LHR	London Heathrow
ORD	76,942,493	ORD	Chicago O'Hare Inter..
PEK	89,938,628	PEK	Beijing Capital Inter..

A `right outer join` is the reverse; it preserves the records from the right table and finds matching values from the left table. `Right outer joins` are not natively supported in Excel data sources. However, we could simply switch the data sources--putting **Airport Geocode** to the left and **Busiest Airports 2015** to the right --to achieve the same desired result:



Data Source		Busiest Airports 2015
Airport Code	=	Airport

Airport Geocode has a record for **DFW**, but the **Busiest Airports 2015** worksheet does not have this. The resulting records will report **Null** for the **Busiest Airports 2015** columns for the **DFW** record:

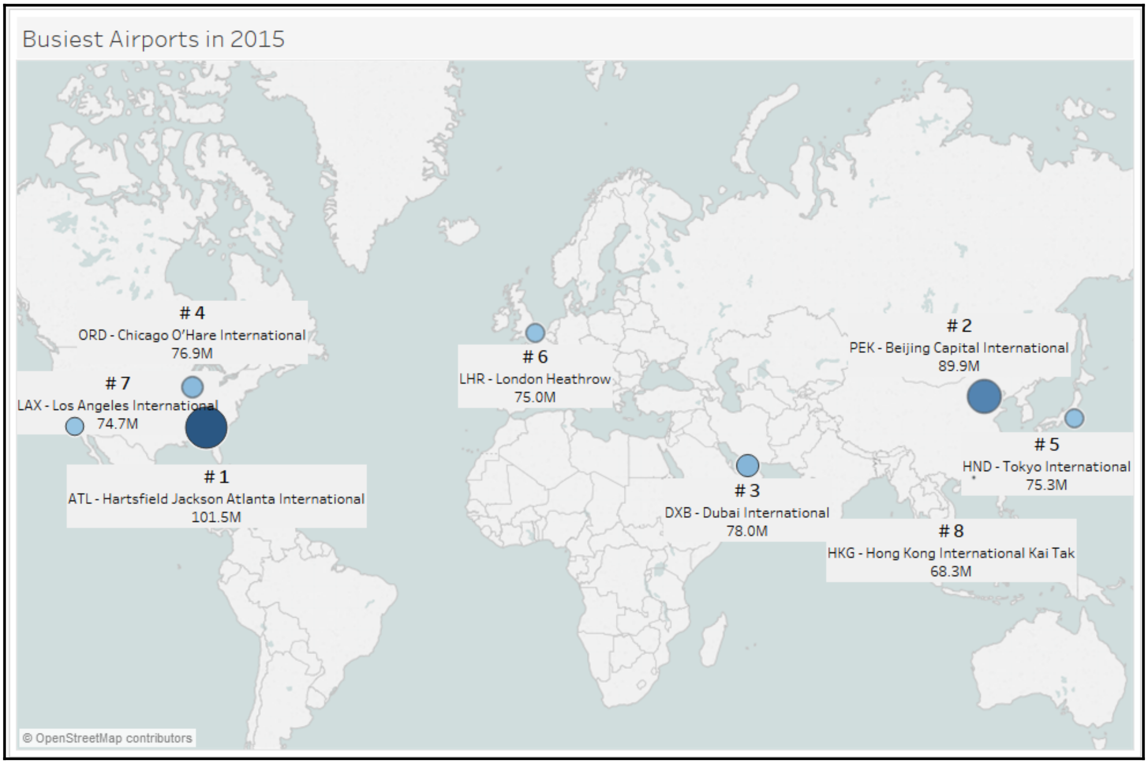


Airport	Passengers	Airport C..	Airport Name
Null	Null	DFW	Dallas Fort Worth International
ATL	101,489,887	ATL	Hartsfield Jackson Atlanta International
DXB	78,010,265	DXB	Dubai International
HKG	68,342,785	HKG	Hong Kong International Kai Tak
HND	75,316,718	HND	Tokyo International
LAX	74,704,122	LAX	Los Angeles International
LHR	74,989,914	LHR	London Heathrow
ORD	76,942,493	ORD	Chicago O'Hare International
PEK	89,938,628	PEK	Beijing Capital International

A `full outer join` preserves both tables being operated on. If the data source driver does not support this, a `full outer join` result can be derived by getting the result of the `left outer join` and appending it to the result of the `right outer join`.

There are a few other types of join--a `self-join` and a `cross join`. A `self-join` simply means that the same table is joined to itself. The actual join type can be inner or outer or even cross join. A cross join gets the cartesian product of the records in the tables being cross joined. When we get a cartesian product, we match up the records from one table to all records in the other table. If we have mof records in one table and nof records in another table, after a cartesian product, we will end up with *m x n* records.

Once we have combined the fields, we can start visualizing our records. Here is a possibility - creating a map that depicts the busiest airports and ranks them based on the average number of passengers:


Busiest Airports in 2015
# 4
ORD - Chicago O'Hare International
76.9M
# 7
LAX - Los Angeles International
74.7M
# 1
ATL - Hartsfield Jackson Atlanta International
101.5M
# 6
LHR - London Heathrow
75.0M
# 2
PEK - Beijing Capital International
89.9M
# 5
HND - Tokyo International
75.3M
# 3
DXB - Dubai International
78.0M
# 8
HKG - Hong Kong International Kai Tak
68.3M
© OpenStreetMap contributors

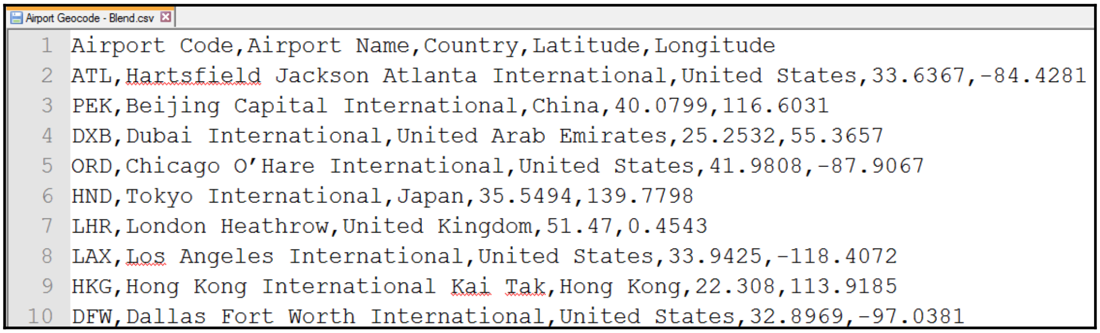
Using blends

Blends are great for data mashups. Blending in Tableau allows multiple data sources to be linked together. The data sources can be of different types - for example, one could be an Excel file while another could be a text file.

In previous versions of Tableau, blend was the only way from within Tableau to link multiple data sources together. Starting in Tableau 10, cross-database joins are supported.

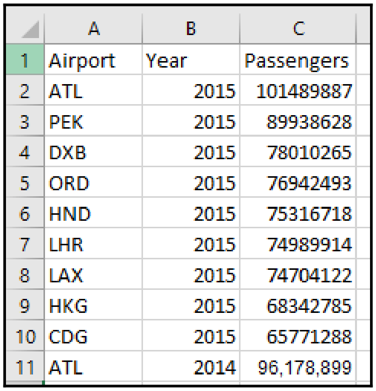
Let's combine the records from a text file and an Excel file using a blend:

1. Download this chapter's files from the Packt website and use the following files:
    - The `Airport Geocode-Blend.csv` file
    - The Worlds `Busiest Airports-Blend.xlsx` file
2. This is the content of the `Airport Geocode-Blend.csv` file:



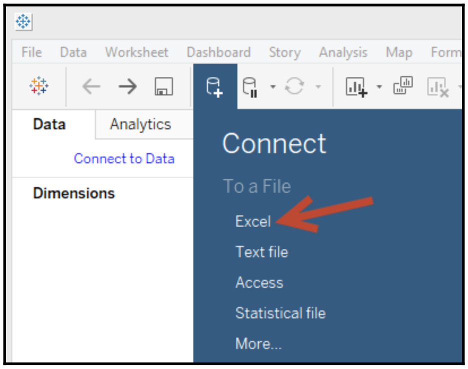
```
Airport Code,Airport Name,Country,Latitude,Longitude
ATL,Hartsfield Jackson Atlanta International,United States,33.6367,-84.4281
PEK,Beijing Capital International,China,40.0799,116.6031
DXB,Dubai International,United Arab Emirates,25.2532,55.3657
ORD,Chicago O'Hare International,United States,41.9808,-87.9067
HND,Tokyo International,Japan,35.5494,139.7798
LHR,London Heathrow,United Kingdom,51.47,0.4543
LAX,Los Angeles International,United States,33.9425,-118.4072
HKG,Hong Kong International Kai Tak,Hong Kong,22.308,113.9185
DFW,Dallas Fort Worth International,United States,32.8969,-97.0381
```

3. These are the records in the `Worlds Busiest Airports - Blend.xlsx` file:

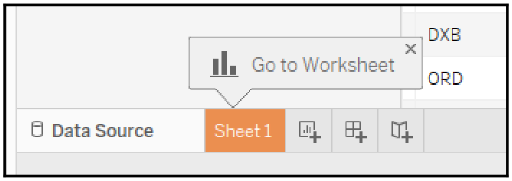


	A	B	C
1	Airport	Year	Passengers
2	ATL	2015	101489887
3	PEK	2015	89938628
4	DXB	2015	78010265
5	ORD	2015	76942493
6	HND	2015	75316718
7	LHR	2015	74989914
8	LAX	2015	74704122
9	HKG	2015	68342785
10	CDG	2015	65771288
11	ATL	2014	96,178,899

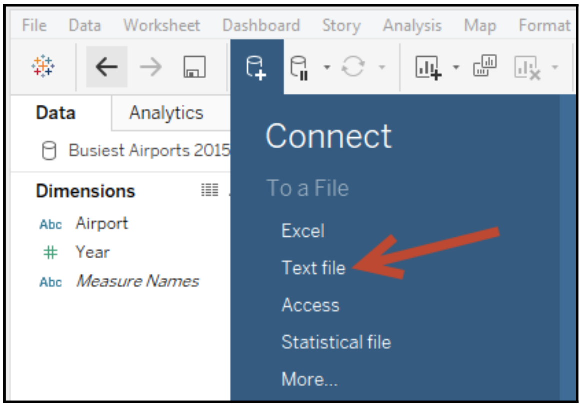
4. Connect to the Excel file in this recipe. Make sure you choose **Excel** from the **To a File** section:



5. Go to new worksheet:

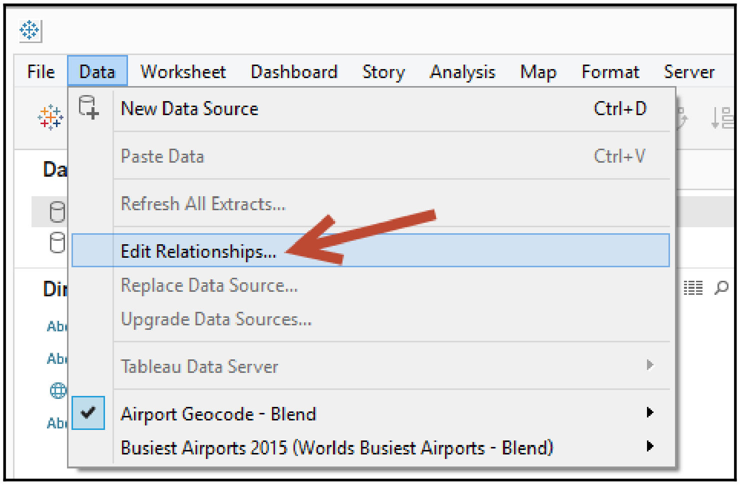


6. Click on the**New Data Source** icon, and this time connect to a **Text file**. Connect to the text file in this recipe:

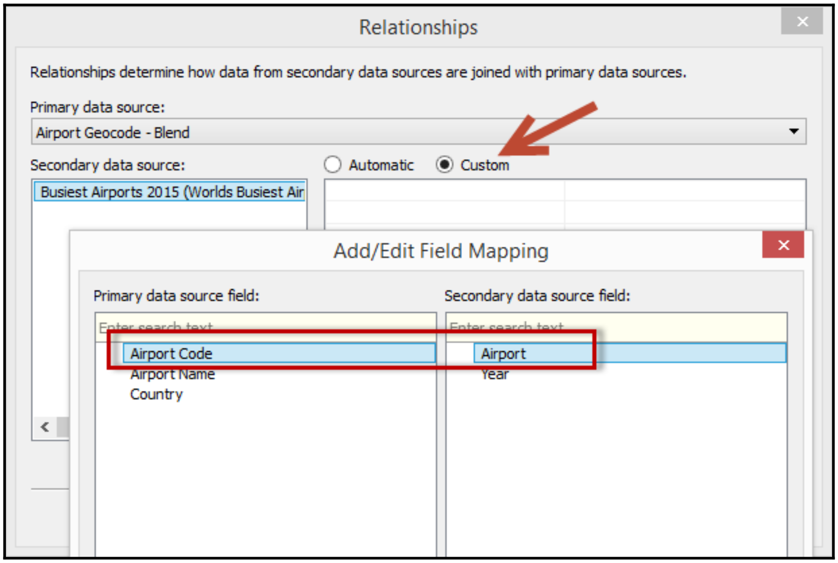


7. If you are directed back to the initial connection screen, go back to `Sheet 1`.

8. Under the **Data** menu, click on **Edit Relationships**:



9. While **Airport Geocode-Blend** is selected as the **Primary data source**, click on **Custom** and match up **Airport Code** field to **Airport**:


Relationships

Relationships determine how data from secondary data sources are joined with primary data sources.

Primary data source: Airport Geocode - Blend

Secondary data source: Busiest Airports 2015 (Worlds Busiest Air

○ Automatic ● Custom

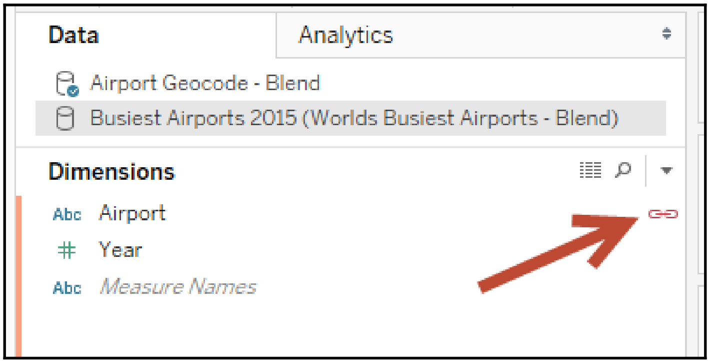
Add/Edit Field Mapping

Primary data source field: Airport Code

Secondary data source field: Airport

10. Click on **OK** when done.
11. While **Airport Geocode-Blend** is selected as the data source, drag Airport Code to the **Rows** shelf.

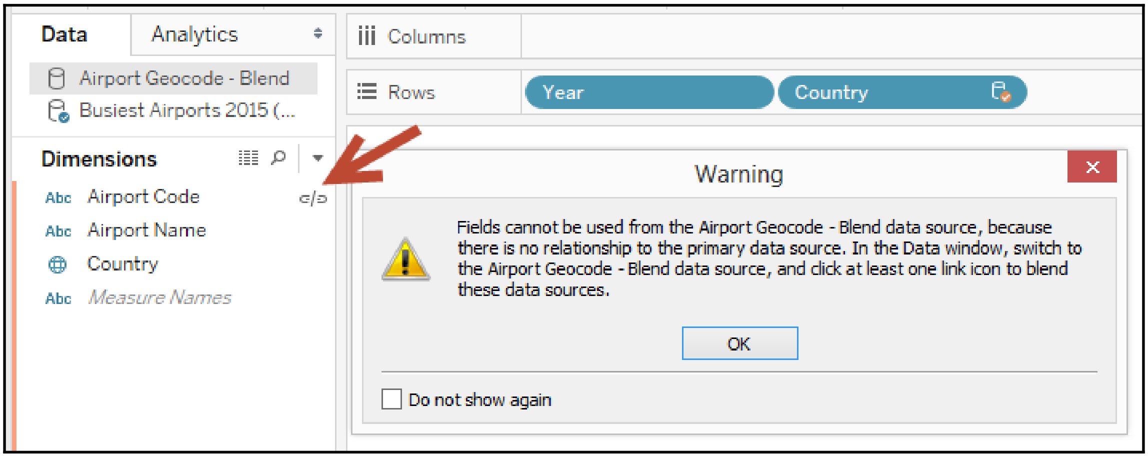
12. Switch data source to the Excel file. Notice that **Airport** now has an orange link icon beside it:



13. Continue to create your visualization using this dataset.

Note that the data sources must have some common fields before they can be blended in Tableau. By default, Tableau looks for the same field names in the data sources and links the sources together based on these fields.

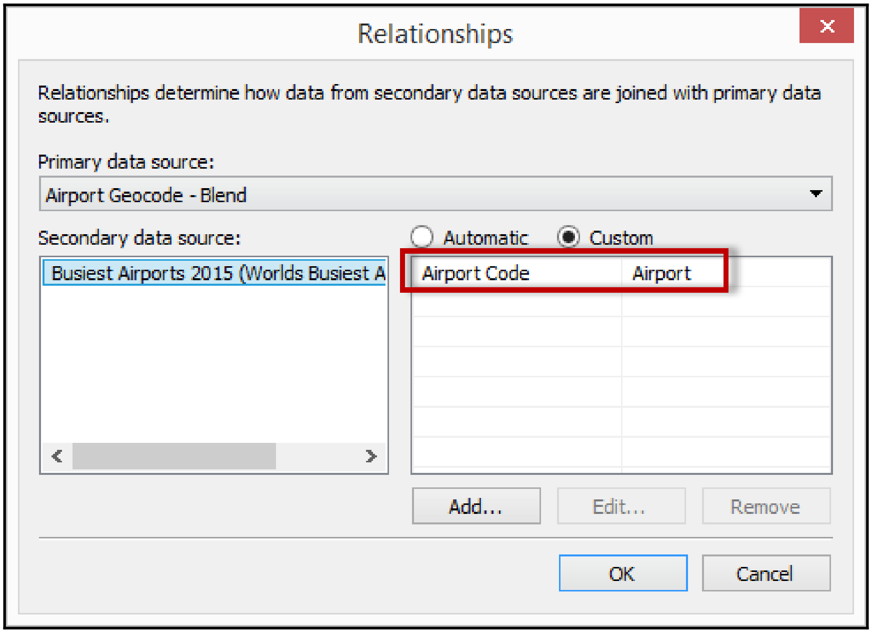
However, if the fields have different names, Tableau will give a warning message indicating that there is no relationship between the data sources. You will also find that when you start using fields from one or both data sources, there will be a broken link icon:



Fields cannot be used from the Airport Geocode - Blend data source, because there is no relationship to the primary data source. In the Data window, switch to the Airport Geocode - Blend data source, and click at least one link icon to blend these data sources.

□ Do not show again

If the field names are different, the relationship needs to be defined. To do this, we can go to the worksheet menu and select **Edit Relationships**. From there, instead of **Automatic**, **Custom** can be chosen as well as identify which fields from both sources should match up:



Relationships

Relationships determine how data from secondary data sources are joined with primary data sources.

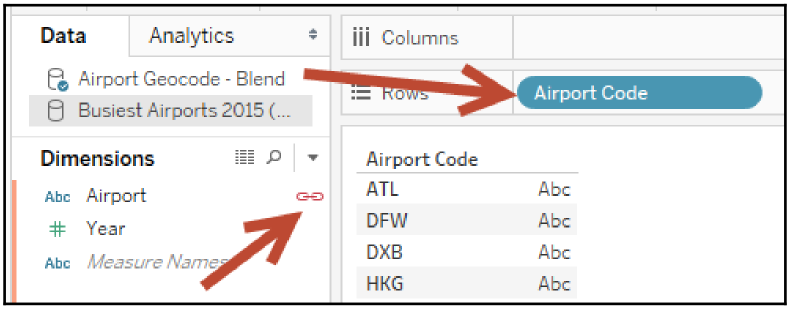
Primary data source: Airport Geocode - Blend

Secondary data source: Busiest Airports 2015 (Worlds Busiest A

○ Automatic ● Custom

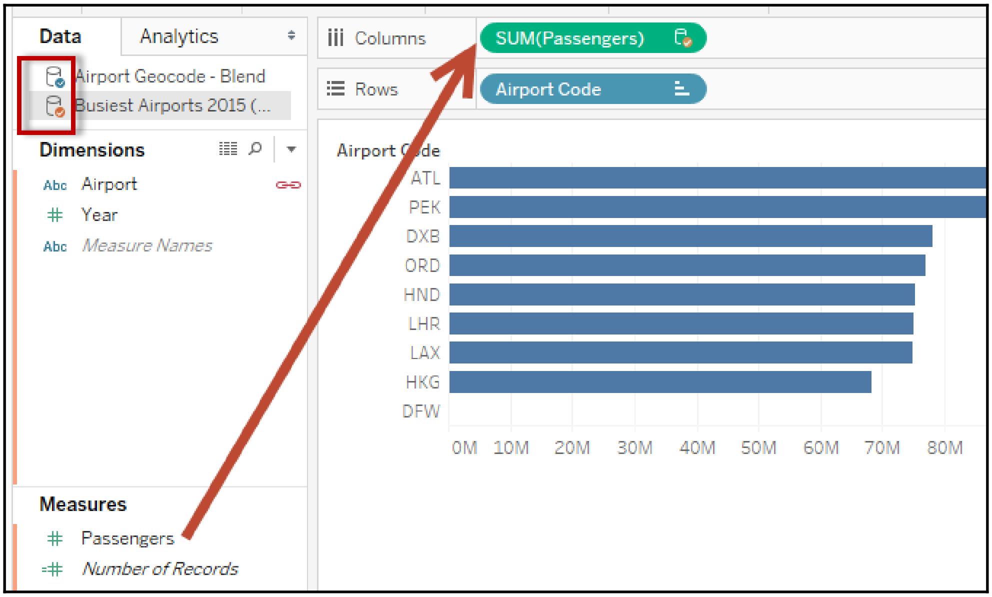
Airport Code	Airport

After the relationship is set, you will find that the link will be enabled. This link will only appear after you have dragged one of the blending fields in the view. If none of the blending fields are in the view, the icon will still appear as broken:

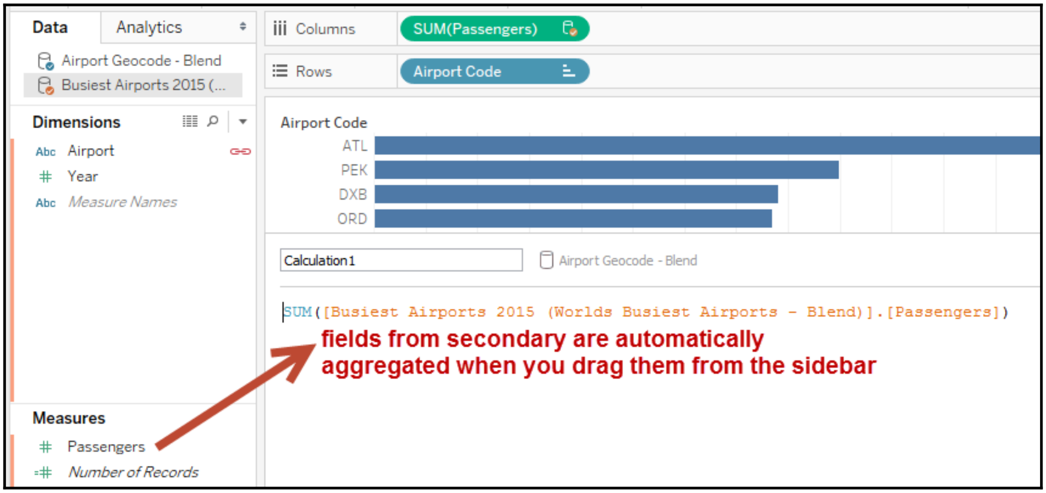


Airport Code	
ATL	Abc
DFW	Abc
DXB	Abc
HKG	Abc

With a blend, there must be only one primary and at least one or multiple secondary data sources. The primary data source is identified by a blue check arrow icon beside it, and the secondary data sources have an orange check arrow icon:



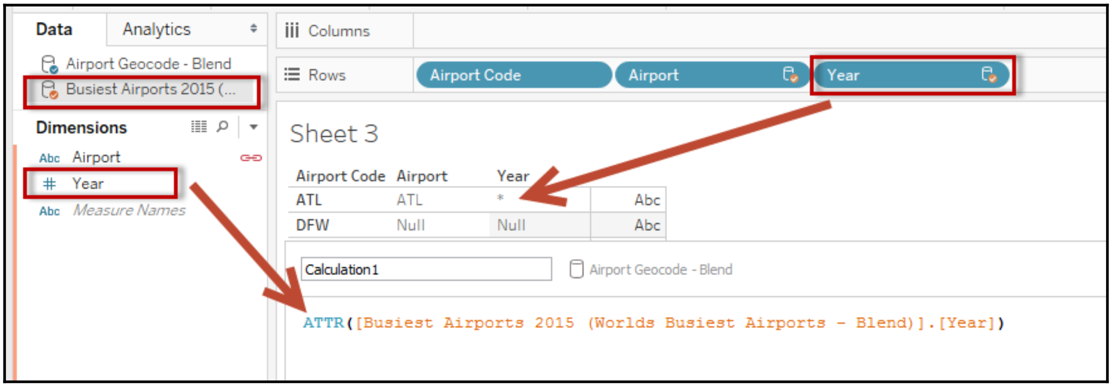
Fields from the secondary data sources will automatically be aggregated when dragged into the view or used in a calculated field. The level of aggregation follows that of the primary:



```
SUM([Busiest Airports 2015 (Worlds Busiest Airports - Blend)].[Passengers])
```

fields from secondary are automatically aggregated when you drag them from the sidebar

Dimension fields will also be aggregated using the **ATTR** function. If there are many related records in the secondary data source and if there are multiple values for that field, the **ATTR** function will return an asterisk (*):



Sheet 3

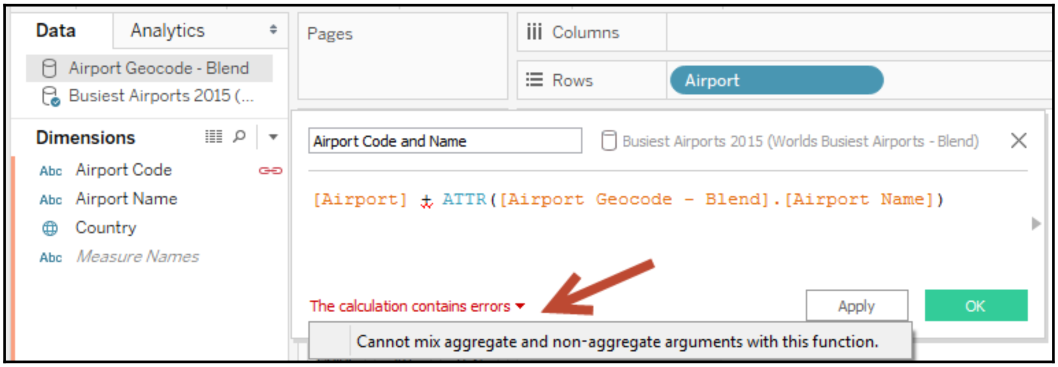
Airport Code	Airport	Year	
ATL	ATL	*	Abc
DFW	Null	Null	Abc

Calculation1

Airport Geocode - Blend

```
ATTR([Busiest Airports 2015 (Worlds Busiest Airports - Blend)].[Year])
```

This leads to a common issue faced in blends when creating calculated fields. We need to make sure that we have the primary and secondary data source fields in an aggregated format when we use them in our expressions. Otherwise, we will get the error **Cannot mix aggregate and non-aggregate arguments with this function**:



Airport Code and Name

Busiest Airports 2015 (Worlds Busiest Airports - Blend)

```
[Airport] + ATTR([Airport Geocode - Blend].[Airport Name])
```

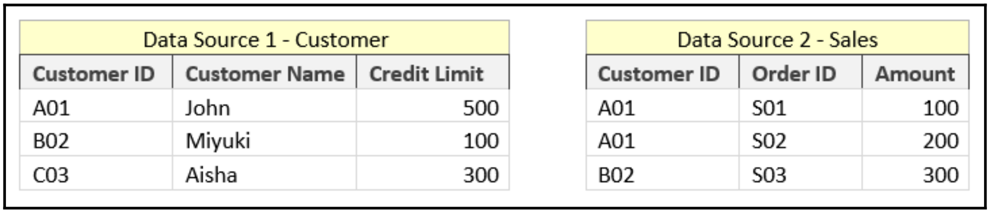
The calculation contains errors

Cannot mix aggregate and non-aggregate arguments with this function.

Blend settings are per worksheet. If you create a new worksheet, the data source you drag from the first will be the primary.

Now that Tableau 10 supports cross-database joins, why would we still want to consider blending data? There are still some compelling reasons to go with blends. The first is, currently, the cross-database join functionality is not supported in all possible connections. Second, we may want to achieve a level of aggregation first before combining data sources .

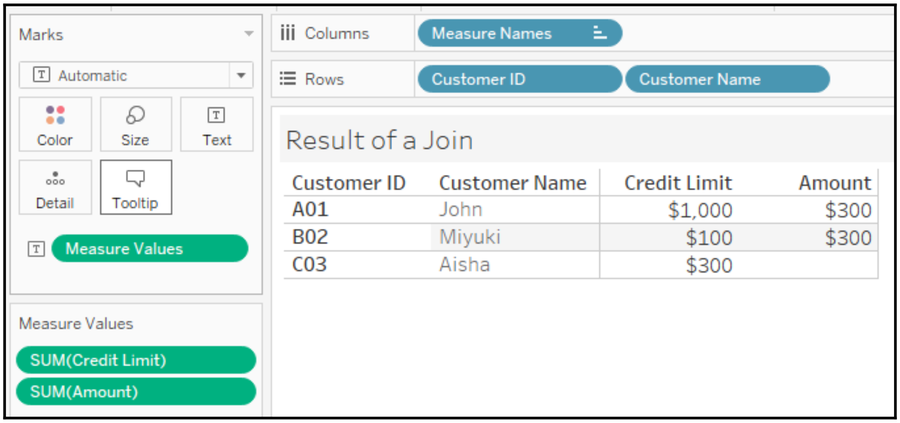
To better illustrate this, let's consider the following two data sources:



Data Source 1 - Customer		
Customer ID	Customer Name	Credit Limit
A01	John	500
B02	Miyuki	100
C03	Aisha	300

Data Source 2 - Sales		
Customer ID	Order ID	Amount
A01	S01	100
A01	S02	200
B02	S03	300

If we were to use a join operation (specifically a `left outer join`, with the customer on the left side of the join operator so it is preserved), we would get the following result. The **Credit Limit** for **Customer IDA01** is incorrect because the credit limit was doubled--**$1,000** is being reported when it really is only **$500**:



Columns: Measure Names

Rows: Customer ID, Customer Name

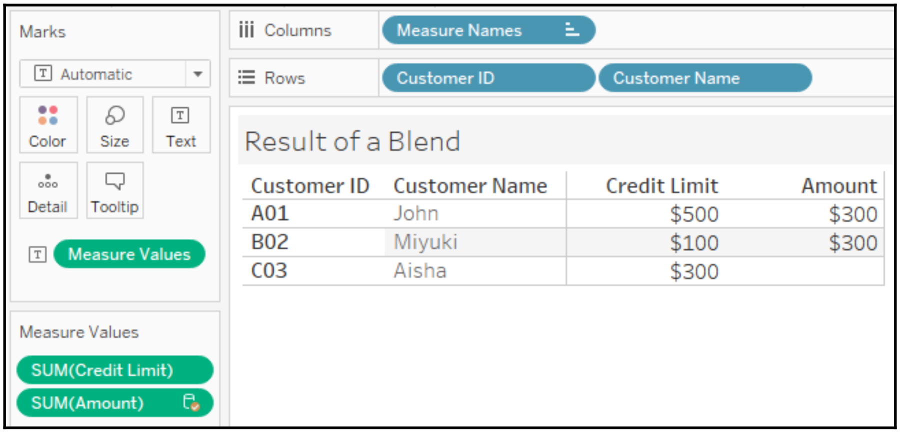
Measure Values: SUM(Credit Limit), SUM(Amount)

Result of a Join

Customer ID	Customer Name	Credit Limit	Amount
A01	John	$1,000	$300
B02	Miyuki	$100	$300
C03	Aisha	$300	

This is the nature of joins, however. The join is working perfectly - it finds the matching values from the other table. Since **Customer IDA01** bought twice, **Customer ID** from the **Customer** table matched twice to the **Sales** table and, inherently, reported the credit limit twice.

If we were to blend, however, we would get the following result set, reporting some different values:



Columns: Measure Names

Rows: Customer ID, Customer Name

Measure Values: SUM(Credit Limit), SUM(Amount)

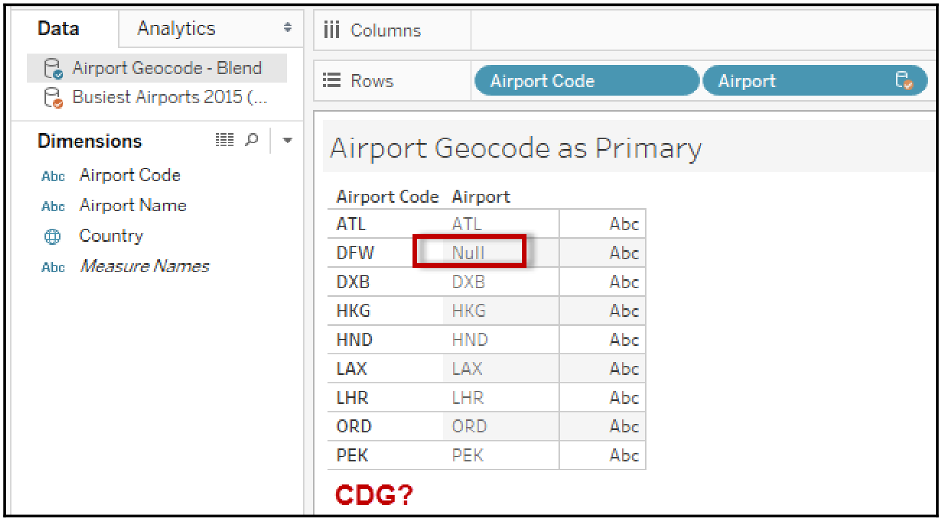
Result of a Blend

Customer ID	Customer Name	Credit Limit	Amount
A01	John	$500	$300
B02	Miyuki	$100	$300
C03	Aisha	$300	

In a blend, the aggregation happens at the data source level first before the records from the two data sources are combined. Notice in the **Measure Values** card, the pill still says **SUM(Credit Limit)**--the same expression you see in the previous join operation. This time, though, the **SUM(Credit Limit)** happens at the customer data source only not at the resulting joined records. The **SUM(Credit Limit)** for **Customer IDA01** in the customer data source is still **$500** because there is only one record for that **Customer ID** in that data source.

One more important thing to know about blends is that after the records in both data sources are aggregated to the same level, the records are combined using an operation akin to a left outer join. This means that if some values in the blending field are absent in the primary, they will not be reported at all.

For example, if our primary is the **Airport Geocode**, and it does not have the airport code **CDG** which our secondary has, **CDG** will not be pulled into any view:

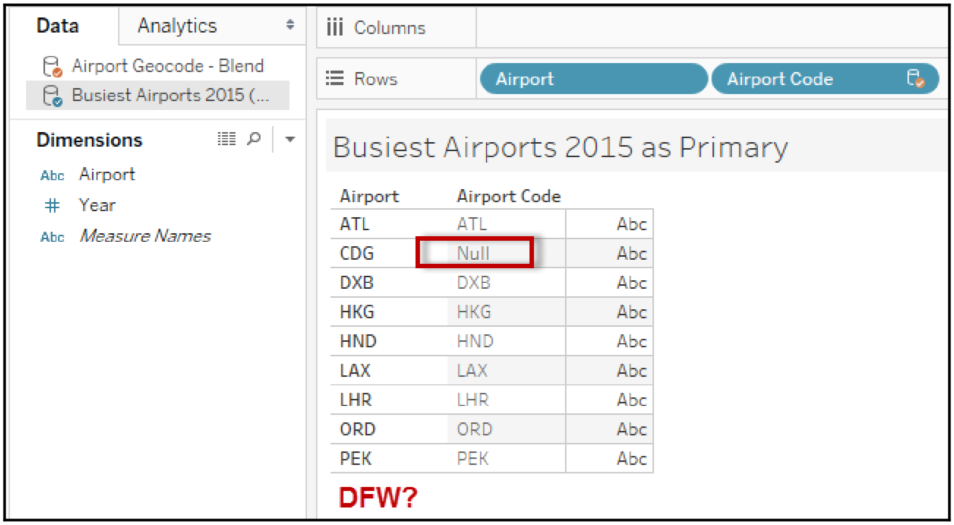


Airport Geocode as Primary

Airport Code	Airport	
ATL	ATL	Abc
DFW	Null	Abc
DXB	DXB	Abc
HKG	HKG	Abc
HND	HND	Abc
LAX	LAX	Abc
LHR	LHR	Abc
ORD	ORD	Abc
PEK	PEK	Abc

CDG?

The same issue will occur even if we reverse the primary and secondary data sources, and if the new primary is missing some values that are present in the secondary. The following shows what you would see if we made **Busiest Airports 2015** the new primary data source, but it is missing the code for **DFW**:



Busiest Airports 2015 as Primary

Airport	Airport Code	
ATL	ATL	Abc
CDG	Null	Abc
DXB	DXB	Abc
HKG	HKG	Abc
HND	HND	Abc
LAX	LAX	Abc
LHR	LHR	Abc
ORD	ORD	Abc
PEK	PEK	Abc

DFW?

There is no magic bullet solution for this issue, however. What we need is to have another data source that has the complete set of values and make that our primary. Or, if this is a data quality issue, this is great way to illustrate why data quality is of utmost importance with data analysis. Remember--good data in, good data (analysis/visualization) out; not-so-good data in, not-so-good data (analysis/visualization) out.

Summary

In this chapter, we covered how to prepare our data for effective use in Tableau. We covered Data Interpreter and pivots to clean our data source. We then used the legacy Jet driver to shape the file and schema.ini to resolve data type issues. Next, we covered pivoting the values into a single column. We also used unions to combine different data sets, and joins to combine records from different tables using common fields. Lastly, we used blends for data mashups.

In the next chapter, we will see how calculations can be used in many ways.