

Structuring Data for Analysis

By Lari McEdward

There are certain concepts that are fundamental to understanding data prep and how to structure data. Data can be generated, captured, and stored in a dizzying variety of formats, but when it comes to analysis, not all data formats are created equal.

Data preparation is the process of getting well formatted data into a single table so it can be analyzed in Tableau. This includes both the structure, ie rows and columns, as well as aspects of the data cleanliness (such correct data types and correct data values).

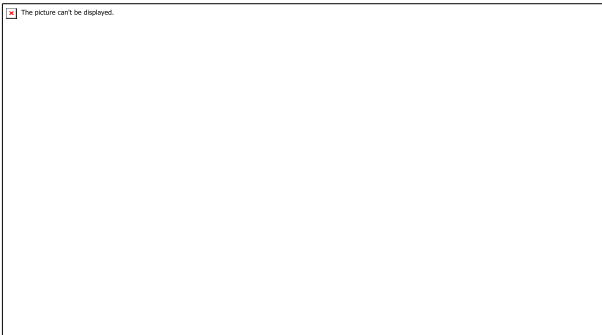
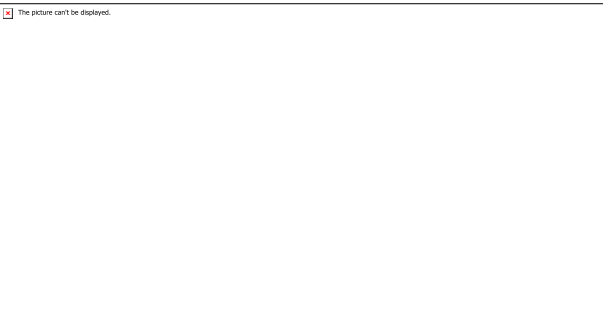
Data Structure

Tableau works best with data that is formatted like a spreadsheet. That is, data that is stored in rows and columns, with column headers in the first row. So what should be a row or column?

What is a row?

A row, or record, can be anything from information around a transaction at a retail store, to weather measurements at a specific location, or stats about a social media post.

It's important to know what a record (row) in the data represents. This is the **granularity** of the data.

Here, each record is a day	Here, each record is a month
 The image shows a large empty rectangular box with a thin black border. In the top-left corner, there is a small red square icon followed by the text "The picture can't be displayed." This is a placeholder for a table or diagram illustrating data where each record represents a day.	 The image shows a large empty rectangular box with a thin black border. In the top-left corner, there is a small red square icon followed by the text "The picture can't be displayed." This is a placeholder for a table or diagram illustrating data where each record represents a month.

A best practice is to have a unique identifier, a value that identifies each row as a unique piece of data. Think of it like the social security number or URL of each record. In Superstore, that would be Row ID. Note that not all data sets have a UID but it can't hurt to have one.

Try to make sure you can answer the question "What does a row in the data set represent?". In Tableau-speak, this would be "What does the Number of Records field represent?". If you can't articulate that, the data might be structured poorly for analysis.

▼ Aggregation and Granularity

A concept related to what makes up a row is the idea of aggregation and granularity (which are opposite ends of a spectrum).

- Aggregation
 - refers to how data is combined together, such as summing all the searches for Pumpkin Spice or taking the average of all the temperature readings around Seattle on a given day.
 - By default, measures in Tableau are always aggregated. The default aggregation is SUM. You can change the aggregation to things like Average, Median, Count Distinct, Minimum, etc.
- Granularity
 - refers to how detailed the data is. What does a row (aka record) in the data set represent? A person with malaria? A provinces' total cases of malaria for the month? That's the granularity.
 - Knowing the granularity of the data is crucial to working with LOD (level of detail) expressions.

Understanding aggregation/granularity is a really critical concept for many reasons; it impacts things like finding useful data sets, building the visualization you want, blending/joining data correctly, and using LOD expressions.

Check out this training video on [Aggregation and Granularity](#) to see how this concept impacts basic vizzes and calcs in Tableau. There's also a Help topic on [aggregation](#).

What is a field/column?

A **column** of data in a table comes into Tableau as a **field** in the data pane, but they are essentially interchangeable terms. (We save "column" in Tableau for use in the columns/rows shelf and in vizzes.) A column of data should contain items that can be grouped into a larger relationship. The items themselves are called values or **members** (only discrete dimensions contain members).

What values are allowed in a given field are determined by the **domain*** of the field. For example, a column for "grocery store departments" might contain the members "deli" "bakery", "produce", etc., but it wouldn't include "bread" or "salami" because those are items, not

departments. Phrased another way, the domain of the department field is limited to just the possible grocery store departments.

Additionally, a well-structured data set would have a column for “Sales” and a column for “Profit”, not a single column for “Money”, because profit is a separate concept from sales.

- The domain of the Sales field would be values ≥ 0 , since sales cannot be negative.
- The domain of the Profit field, however, would be all values, since profit can be negative.

***domain** can also mean the values present in the data. If the column "grocery store department" erroneously contained "salami", that would be in the domain of the column. The definitions are slightly contradictory. One is the values that *could/should* be there, the other is values that *actually are* there.

Defining a column can be tricky as it depends on the rows. A column contains a value which represents an attribute, description, or numeric quantity and corresponds to the item in each row

Categorizing fields

Each column in the data table comes into Tableau as a field, which shows up in the data pane. Fields in Tableau must be either a dimension or measure (separated spatially in the data pane) and either discrete or continuous (color coded: blue fields are discrete and green fields are continuous).

- **Dimensions** are qualitative, meaning they can't be measured but are instead described. Dimensions are often things like city or country, eye color, category, team name, etc. Dimensions are usually discrete.
- **Measures** are quantitative, meaning they can be measured and recorded with numbers. Measures can be things like sales, height, clicks, etc. In Tableau, measures are automatically aggregated; the default aggregation is SUM. Measures are usually continuous.
- **Discrete** means individually separate or distinct. Toyota is distinct from Mazda. In Tableau, discrete values come into the view as a label and they create panes.
- **Continuous** means forming an unbroken, continuous whole. 7 is followed by 8 and then it's the same distance to 9, and 7.5 would fall midway between 7 and 8. In Tableau, continuous values come into the view as an axis.
- Dimensions are usually discrete, and measures are usually continuous. However, this is not always the case. Dates can be either discrete OR continuous.
 - Dates are dimensions and automatically come into the view as discrete (aka date parts, such as “August”, which considers the month of August without considering other information like the year). A trend line applied to a timeline with discrete dates will be broken into multiple trend lines, one per pane.

- We can choose to use continuous dates if preferred (aka date truncations, such as “August 2017”, which is different than “August 2018”).
- (There is a training video on [Understanding Pill Types](#) that goes over the concepts of dimension/measure and discrete/continuous function in Tableau. There are also Help topics on [dimensions & measures](#) and [continuous & discrete](#).)

Tableau Prep Builder does not distinguish between measures and dimensions. However, like, Tableau Desktop, it does assign data types to domains. These types include string, number (decimal), number (whole), date, and date and time. Understanding the concepts behind discrete or continuous are important, however, for things like understanding the detail vs summary presentation of data in the profile pane.

- Detail = the detail view shows every domain element (as a discrete label)
 - And has a visual scrollbar to provide a visual overview of all the data
- Summary = the summary view shows the values as binned on a continuous axis, a histogram

Binning & Histograms

A field like age or salary is considered continuous. There is a relationship between the age 34 and 35, and 34 is as far from 35 as 35 is from 36. However, once we're past age 10 or so, we usually stop saying things like we're “9 and a half” or “7 and $\frac{3}{4}$ ”. We're already “binning” our age to neat year-sized increments. Someone who is 12,850 days old is older than someone who is 12,790 days old, but we draw a line and say “you're both 35”. Similarly, age groupings are often used in place of actual ages. Child prices for movie tickets are for kids “12 and under”, or a survey may ask you to select your age group, such as 20-24, 25-30, etc.

Histograms are used to visualize the distribution of numerical data using binning. A histogram is similar to a bar chart, but rather than being discrete categories per bar, the rectangles making up the histogram span a “bin” of a continuous axis, such as range of the number of days (0-4, 5-9, 10-14, etc.). The height of the rectangles is determined by frequency or count. Here, the y-axis is the count of countries that fall into each bin. Seven countries offer 0-4 days of paid leave, two countries offer 5-9 days, and 43 countries offer 20-24 days of paid leave.



In Tableau Prep Builder, the summary view is a histogram of binned values. The detail view has a visual scrollbar off to the side that shows the overall distribution of the data.

Distributions and outliers

Seeing the distribution of a data set can help with outlier detection.

- Distribution = the shape of the data in a histogram (this depends on the size of the bins). Being able to see all your data in a histogram view can help you identify if the data seems correct and complete. The shape of the distribution will only be of use if you know the data and can interpret whether or not the distribution makes sense.
 - For example, if we were to look at a data set of the number of homes with broadband internet from 1940-2017, we'd expect to see a very skewed distribution. However, if we were to look at the number of homes with broadband internet from January 2017 to December 2017 we'd expect a fairly uniform distribution.
 - If we were to look at a data set of Google searches for "Pumpkin Spice Latte", we'd expect to see a fairly sharp peak in the fall, whereas searches for "convert Celsius to Fahrenheit" would likely be fairly stable.
- Outlier = a value that is extreme compared to other values. Outliers may be correct values or they may be indicative of an error.

- Some outliers are correct and indicate actual anomalies, these should NOT be removed or modified.
- Some outliers indicate issues with data cleanliness (such as a salary of \$50 instead of \$50,000 because a period was typed instead of the comma).

▼ Visual outlier detection with distributions

If you were to see a list like this:

00-09: ■ ■
10-19: ■ ■ ■ ■
20-29: ■ ■ ■ ■ ■
30-39: ■ ■ ■
90-99: ■

at first glance it doesn't look odd. But if, instead of a list of labels, this was plotted on a continuous binned axis, it would look like this:

00-09: ■ ■
10-19: ■ ■ ■ ■
20-29: ■ ■ ■ ■ ■
30-39: ■ ■ ■
40-49:
50-59:
60-69:
70-79:
80-89:
90-99: ■







And it's much more obvious that the last observation is farther away from the first and may be an outlier due to error.

Data Types

Databases, unlike spreadsheets, usually enforce strict rules on data types. Data types classify the type of data in a given field and provide information about how the data should be formatted, interpreted, and what operations can be done to that data. For example, numerical fields can “have math done to them” and geographic fields can be mapped.

Tableau assigns whether a field is a dimension or measure, but fields have other characteristics that depend on their data type. These are indicated by the [icon](#) each field has (though some types share an icon). As previously stated, Tableau Prep Builder uses the same types (except geographic, which is assigned as a data role). If data type is enforced on a column and an existing value doesn't match its assigned data type, it may be displayed as null (because “purple” doesn't mean anything as a number).

Some functions require specific data types. For example, you cannot use CONTAINS with a numerical field. Type functions are used to change the data type of a field, such as DATEPARSE can take a text date in a specific format and make it a date, enabling things like automatic drill down in the view.

Icon	Data Type	Notes
	Text (string)	String fields can handle pretty much everything, formatting-wise.
	Date	Tableau automatically supports a wide variety of date formats . DATEPARSE can help with dates in other formats . Note that the locale setting in Tableau impacts how dates are interpreted (i.e. 30/4/16 vs 4/30/16).
	Date & Time	
	Numerical	AKA a number. Numerical fields can be either decimal (aka floating point) or whole (aka integer)
	Boolean	Boolean data can have one of two values, true (T) or false (F). A field can be boolean, as can the result of certain logical expressions (such as [Age] > 7, which will be true or false for various values of Age.)
	Geographic	If a value is assigned a geographic data type but doesn't appear in Tableau's internal geographic database, it cannot be mapped by default.

NB: There is a Tableau Help article on [Data Types](#).

Null

When a value is unknown, the data value is “null”. Null does not mean the same thing as zero. Imagine someone asked you to list what you had for each meal for the last week. If you can’t remember, you could leave that entry blank. This doesn’t mean you didn’t eat, just that what you ate isn’t known. In terms of data, null has to be handled in special ways. For example, you can’t do math with null ($3 + \text{null} = \text{null}$, because adding three to an unknown is still unknown). NB: There is a Help article on [Null values](#).

Pivoting and Unpivoting Data

People-friendly data is often captured and recorded in a wide format, with many columns. Machine-readable data, like Tableau prefers, is better in a tall format, with fewer columns and more rows.


A note on the industry standard definitions of pivot and unpivot

Traditionally, pivoting data means going from tall to wide (columns to rows), and unpivoting means going from wide to tall (rows to columns). However, Tableau uses the word “pivot” to mean going from wide (people-friendly) to tall (machine-readable) by turning rows into columns. In this document, “pivot” will be refer to the Tableau sense of the word.

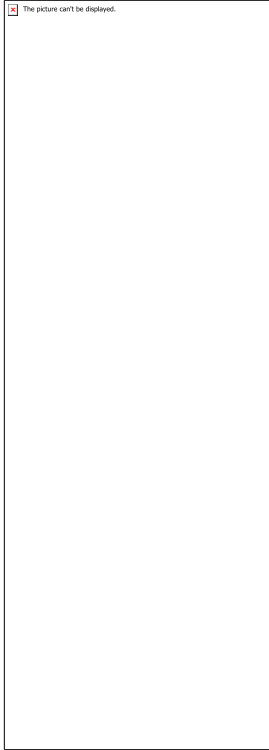
NB: There are Help articles on [pivoting](#) and tips for [working with data](#).

Wide data

In the WHO malaria data set, there is a column for country, then a column per year. Each cell represents the number of cases of malaria for that country and year. There are 108 rows (for 107 countries plus the header row) and 16 columns (15 years, plus the country column).

 The picture can't be displayed.

It’s easy for a person to read and understand this format. However, if we were to bring this data into Tableau, we get a field per column. We have a field for 2014, a field for 2013, a field for 2012, etc.




To think of it another way, there are 15 fields that all represent the same basic thing--number of reported cases of malaria--and no single field for time. This makes it very hard to do analysis across time as the data is stored in separate fields.

▼ Example

Q: how would we create a map that shows the total number of malaria cases per country from 2000 to 2014?

A: create a calculated field to sum all the years.

 The picture can't be displayed.

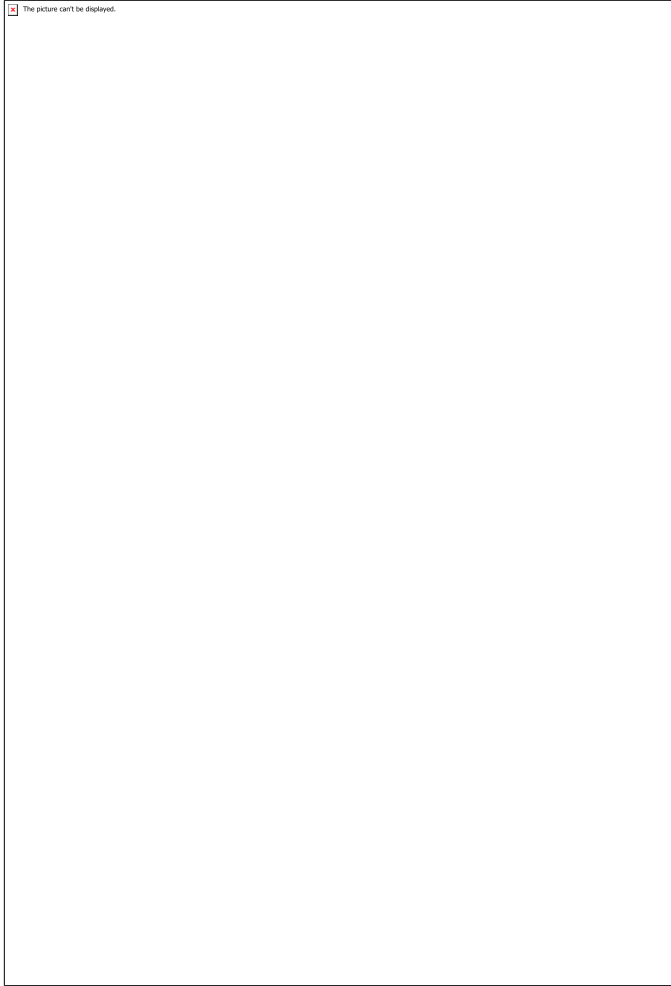
Another indication that this format isn't ideal for analysis can be seen in the fact that nowhere do we have information about what the actual values mean. For Algeria in 2012, we have the value 55. Fifty five what? It's not clear from the structure of the data.



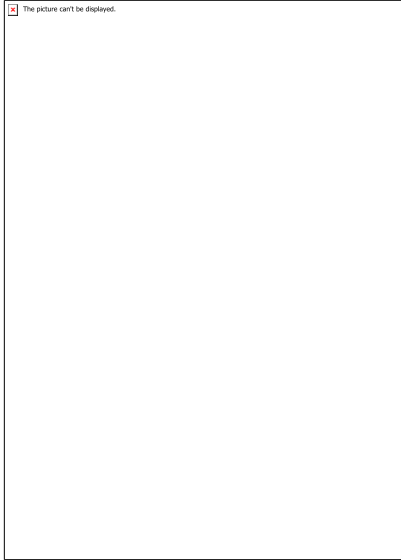
If the name of the column isn't describing what the values are but rather conveys additional information, this is a sign the data needs to be pivoted.

Tall data

However, if we pivot the data, we reshape the data from wide to tall. Now, rather than having a column for each year, we have a single column, "Year", and a new column, "Reported Cases". In this format, we have 1,606 rows and 3 columns. This data format is taller rather than wider.




Now in Tableau, we have a field for Year and a field for Reported Cases as well as the original Country field. It's much easier to do analysis because each field represents a unique quality about the data set--location, time, and value.



▼ Example

Q: how would we create a map that shows the total number of malaria cases per country from 2000 to 2014?

A: use the Reported Cases field.

 The picture can't be displayed.

Now it's easy to see that for Algeria in 2012, the 55 refers to the number of reported cases (because we could label this new column).



A note about tall vs wide data:

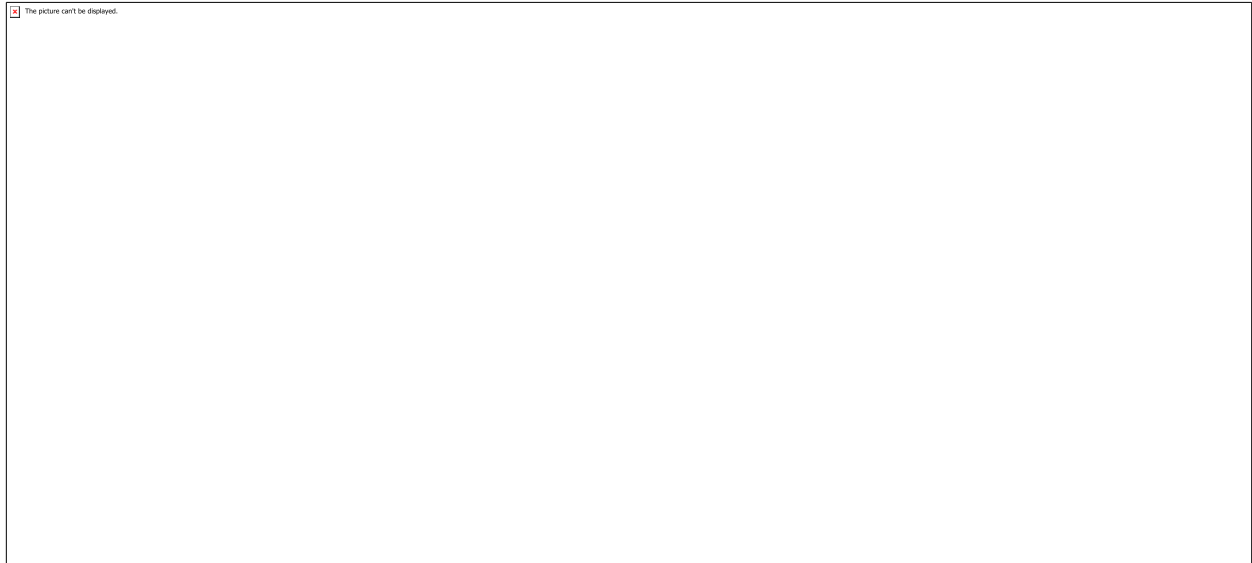
In this example, when working with wide data there's a single record per country. With the tall data format, there are now 15 rows for each country (one for each of the 15 years in the data). Be very careful during analysis that you're keeping this in mind. If there was a column for "land area", that value would be repeated for each of the 15 rows for each country. If you created a bar chart by bringing out Country to Rows and Land Area to columns, by default you'd be summing the land area for all 15 rows per country. For some columns it may be necessary to, say, filter to a single year, or to aggregate with an average or minimum rather than sum to avoid double counting values.

Normalization

Relational databases are made up of multiple tables that can be related or linked together in some way. Each table contains a unique identifier, or key, per record. By joining on the keys, records can be linked to provide more information than is contained in a single table. What information goes into each table is dependent on the data model used, but the general principle is around reducing duplication.

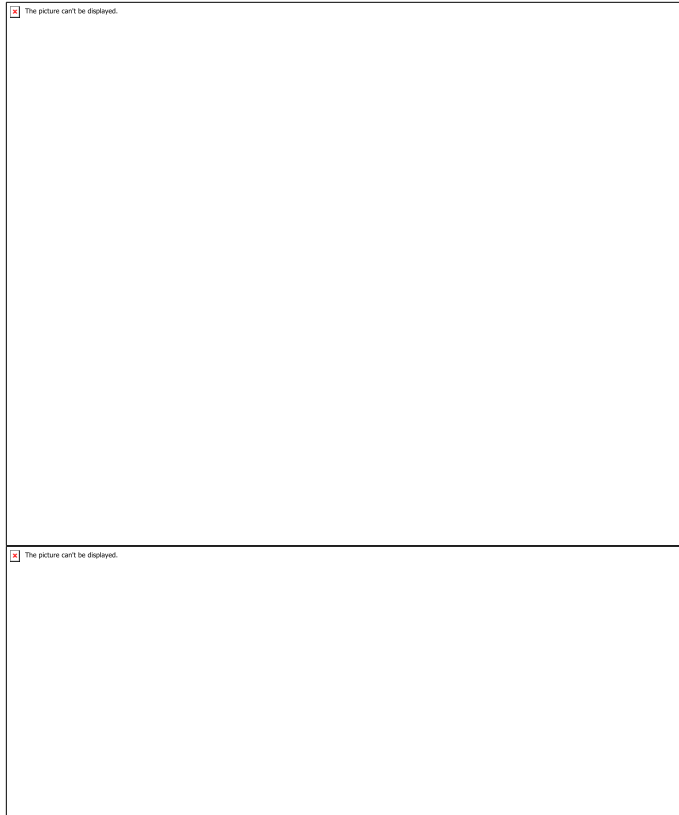
For example, I definitely used Tableau to help plan my wedding.

I could have created one massive table to keep all my guest information:



When I got my first “undeliverable” returned save the date card, I saw I had to fix that address multiple times. Clearly this wasn’t an optimal structure.

I saw some information needed to be kept at the level of the guest (such as dietary restrictions, what table they’d be sitting at, and if they were coming or not...) but some was at the level of the party (one mailing address, if the RSVP had been sent out, if they’d given a gift, if they were with the bride or groom...). So I broke it up into two tables:



It was much easier to have a table for guest-level information (one record = one guest) and a table for party-level information (one record = one party). Then, I could track dietary information per person yet only have one address for the party. It was also much easier to figure out how many chairs I'd need (Number of Records from the guest table) and how many stamps I'd need for thank-yous (Number of Records from the party table). (WHOOOO PARTY TABLE!)

The process of breaking my data up into multiple tables and figuring out which columns went into each one is **normalization**. (Or, as [Wikipedia](#) puts it: the process of organizing the [columns](#) (attributes) and [tables](#) (relations) of a [relational database](#) to reduce [data redundancy](#) and improve [data integrity](#). Normalization is also the process of simplifying the design of a database so that it achieves the optimal structure composed of atomic elements. It was first proposed by [Edgar F. Codd](#), as an integral part of a [relational model](#).)

However, when I wanted to bring the data back into Tableau and figure out if my seating assignments (guest table) mixed things up between the bride and groom (party table), I needed to **join** the tables together. Fortunately, I'd known that and built my tables correctly. There was a linking field, Party, in both tables. I recorded which party each guest belonged to in the guest table, then joined on the party column from the party table and essentially get back to my original combined (aka **denormalized**) table.

So why didn't I just keep the combined table? Well, it was harder to maintain (I'd have to update multiple rows every time I sent an RSVP card) and it was also storing redundant information. My

wedding was only 120 people so that didn't matter, but for a database with massive quantities of data, storing the same information over and over isn't efficient.

Normalized tables require a few key properties:

- Each row needs a unique identifier
- Each table needs a column (or columns) that can be used to connect it back to other tables (key).

To access information from multiple tables at the same time, the tables need to be joined. A join combines columns from different tables based on values they both contain. In the wedding example, the join clause would be something like: `Guest.Party = Party.Party` (aka "join when the value for Party from the Guest table matches the value for Party in the Party table). This would result in the combined table above.

Joins

For a basic overview of joins and join types, check out [the join type training video](#)

The picture can't be displayed.

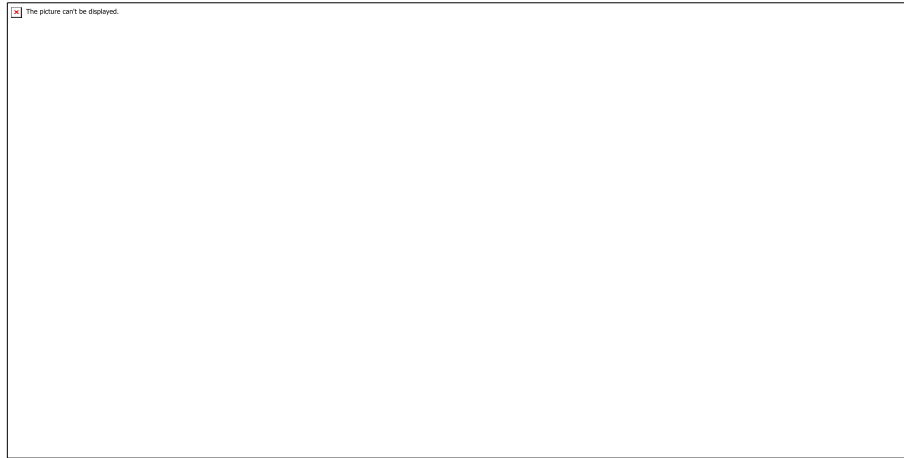
“Tidy” Data

(This section is paraphrased/directly taken from a 2014 article “Tidy Data” from the Journal of Statistical Software (August 2014, Volume 59, Issue 10) by Hadley Wickham. A version of the article can be found [here](#))

Abstract: “A huge amount of effort is spent cleaning data to get it ready for analysis, but there has been little research on how to make data cleaning as easy and effective as possible. This paper tackles a small, but important, component of data cleaning: data tidying. Tidy datasets are easy to manipulate, model and visualize, and have a specific structure: each variable is a column, each observation is a row, and each type of observational unit is a table.”

The framework of tidy data is based on relational databases but instead of being phrased in database terminology, it borrows from statistics and experimental design (which therefore lends itself well to analysis).

Data sets are “rectangular tables made up of rows and columns”. But what makes a column or row can vary, so we need better terminology.



Values: the numbers or strings making up the data set.

Variable: “contains all values that measure the same underlying attribute (like height, temperature, duration) across units”. For the data above, the **variables** are

- Person (John Smith, Jane Doe, Mary Johnson)
- Treatment (a & b)
- Result (2, 16, 3, etc.)

Observation: “contains all values measured on the same unit (like a person, or a day, or a race) across attributes.” For the data above, the observations are the combination of person and treatment.

In a tidy data set, each variable is a column and each observation is a row:



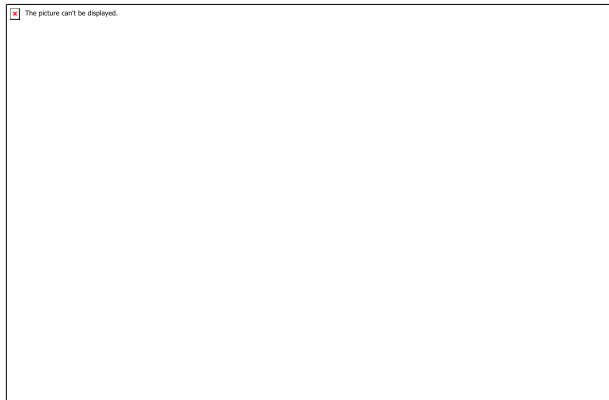
(The article goes on to say “and each type of observational unit is a table”. This would be relevant in a relational database but is less important for Tableau, which works on single tables. For more information, see the section on Normalization.)

“For a given dataset, it is usually easy to figure out what are observations and what are variables, but it is surprisingly difficult to precisely define variables and observations in general. For example, if the columns in the Table 1 were height and weight we would have been happy to call them variables. If the columns were height and width, it would be less clear cut, as we might think of height and width as values of a dimension variable. If the columns were home phone and work phone, we could treat these as two variables, but in a fraud detection environment we might want variables phone number and number type because the use of one phone number for multiple people might suggest fraud.”

According to the author, the five most common problems with messy data sets are as follows. We’ll go through and discuss each in the context of analysis in Tableau.

#1 - Column headers are values, not variable names.

- AKA the data is in crosstab format. Solve this by pivoting columns to rows



- - Let’s say the raw data is 3 bags of M&Ms, and the columns are something like red, blue, and green, and the numbers are the counts of each color. We instead want a column for “color” and a column for “count”, as seen on the right. This can be achieved via a pivot.

#2 - Multiple variables are stored in one column.

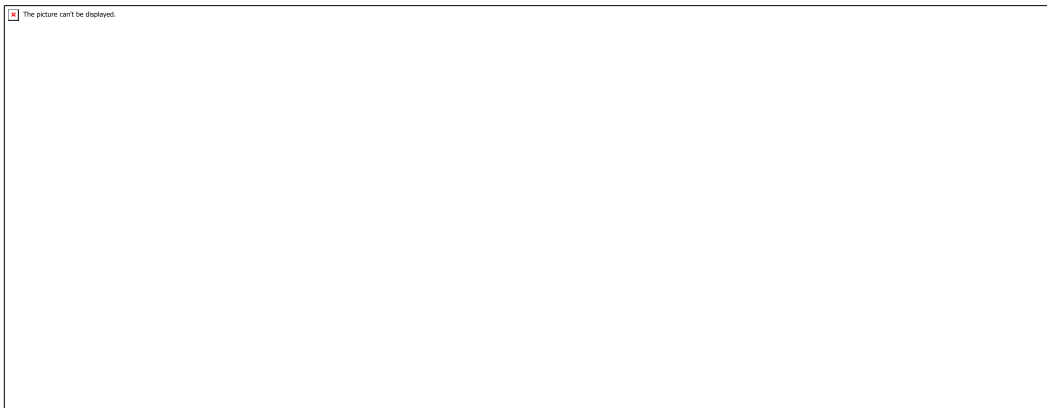
- Split the contents of the column into separate values.



- - Here, the “column” on the left is both sex and age information, and the desired format is a column for sex and a column for age. This can be achieved via a Split or other calculation.

#3 - Variables are stored in both rows and columns.

1.
 - A variant of #1, but which results in a sort of over-formatted data set so that after pivoting columns to rows, something like Min and Max are stored in the same column. Solve by pivoting rows to columns

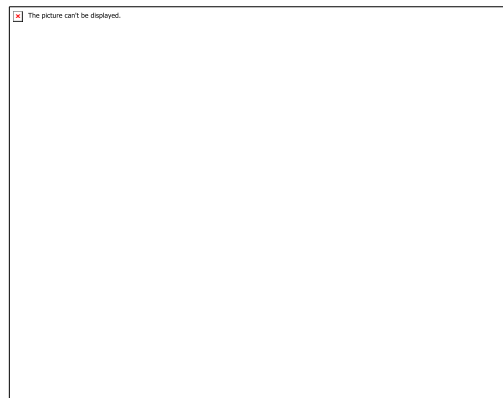


•

- The original table (above) needs to be pivoted like in #1 to bring the days out of columns and into rows (in this example, the day is combined with the year and month to make a date column) as seen below:



- - However, there is the additional issue of the “Element” column which should really be two separate columns, because we want to use each one independently. We need to pivot rows to columns to create a column for max and one for min.



NB: This is where some subjectivity comes in. Why is it better to bring Day out of columns and into rows but to have Min and Max as columns rather than rows? For one, It's hard to do something like compute the temperature range when the values are stored as rows, but much more simple to do that type of calculation across columns. To a certain extent, it depends on the type of analysis desired. Tableau can handle both formats, this becomes more about stylistic choices than anything else.

#4 - Multiple types of observational units are stored in the same table.

- 1.

- This would be addressed by normalizing the data. However, Tableau (like most analysis software) requires all the data in a single table, aka denormalized data (achieved by joining the various tables).

#5 - A single observational unit is stored in multiple tables.

- For example, [daily air quality data from the EPA](#) is stored in a table (or file) per year. These would need to be unioned together to be easily analyzed in Tableau.

What next?

- [Data Formatting Issue: varying levels of detail](#)
- [Relational Algebra 101](#)