Procedures and Guidelines

Clerissa Copeland, Jason Pither, and Mathew Vis-Dunbar

2021-07-15

Contents

Welcome

Write any prefatory content here.

Copyright

Please use the following for citing this document

Copeland, C., Pither, J., Vis-Dunbar, M. (2021). *Procedures and Guidelines*. url

All source files are available url.

Conventions

File and Data Management

Chapter 1

Introduction

Well organized data is critical to transparency, reproducibility and generally maintaining one's sanity when conducting research. When we talk about file and data management we may be referring to one of many aspects of making our data understandable to others, to a computer, or to our future selves that have succumb to memory lapses. Making data comprehensible is really about well structured and communicated metadata that is, whenever possible, implemented according to conventions or standards.

So, when we talk about file and data management, broadly speaking, we're talking about

- File naming and file naming conventions
- Directory structures
- Organizing and formatting data at the variable level

Directory structures, being more complicated, bring with them the need to add additional documentation, such as a description of the directory structure and what we might expect to find where. It will also often include more detailed documentation about how to interpret what is inside specific files; one example of this is the data dictionary that describes each of the variables collected for the study.

Organizing and formatting data is a discussion about the best way to sort and parse our data into columns and rows so that we can effectively produce summaries, statistical calculations, and visualizations from these data; a core concept you'll be introduced to in this guidelines document is that of "tidy data".

Chapter 2

File Naming Conventions

File naming isn't exactly fun, but it is crucially important to being able to organize, describe, and manage any kind of work, especially research. So let's talk about file naming conventions, and those that you'll be expected to use while in Biology at UBCO.

We'll start with the rules, we'll then break these down and explain the process.

2.1 Quick Reference

- File names should only contain letters in the English alphabet, numbers from 1-9, dashes "-", and underscores "_".
- Do not use spaces or special characters, including # % & < > : " / | ? * { } $! ' @ + ` =
- File names should be broken down into components that are separated by underscores "_".
- If more than one word is needed in each component, these are separated by dashes "-".
- All file should start with your last name and all other components should be meaningful (read on for what it means for a file name to be meaningful!)

There are four variations on how these gudielines are implemented depending on what your file contatins.

2.1.1 Lab reports and manuscripts

Format LastName_Project_File-contents_Version.file-type

Example Pither_BIOL116RProject_Manuscript_V0.docx

2.1.2 Figures and plots

Format LastName_Project_Figure-title_Version.file-type

Example Pither_BIOL116RProject_Figure-freq-plot_V1.png

2.1.3 Analysis

Format LastName_Project_Analysis_Version.file-type

Example Pither_BIOL116RProject_Analysis_V0.xlsx

2.1.4 Data

Format LastName_Date_Project_Data-file-description.file-type

Example Pither_20210921_BIOL116RProject_Data.csv

2.2 What's in a name

File names need to achieve two primary goals, they need to make sense to a human reading them and they need to be constructed in a way that allows a computer to parse or process them. That is, file names should be **human interpretable** and **machine readable**. How do we achieve this?

Human interpretable

To be human interpretable, a file name needs to be meaningful. To do this, it needs to convey some basic information to a person reading it. We do this by integrating metadata into the file name. The metadata elements we include are:

- Who created the file
- The date on which it was created
- The project to which it is connected
- The nature of the contents of the file
- If it's been modified
- The type or format of the file

That is, we should be able to look at a file and tell, *who* created it, *when* it was created, *what* it is related to, *what* is inside of it, if it has been *updated*, and what *application* I should expect to be able to open it with. As we'll see shortly, we don't always include a date, and we don't always include information about modifying a file.

All said though, that's a fair bit of information to hold in a file name!

Machine readable

What does it mean for a file name to be machine readable or machine interpretable? This means building our file names in such a way that we can easily organize our files so that they can be sorted by an application and in a way that makes sense to us. It also means building our names according to set patterns, which can then be parsed along known delimiters. Lastly, it means building our names in such a way that if we move them from one computer to another, from one application to another, or from one operating system to another, the files remain interpretable in exactly the same way.

How do we this? We avoid special characters and we follow conventions.

Special characters

Special characters are any characters that are **not** part of the English alphabet, a number from 0 - 9, or one of either a dash "-" or underscore "_". This means that a space " " is a special character, which means that your file names should not have spaces.

When operating in a multi-lingual or non-English environment, this can prove problematic, but it is an unfortunate legacy of the development of computer standards that has yet to be fully resolved.

Conventions

Convention has file naming proceed in the following order, with each element separated by an underscore "_", and words within an element joined with a dash "-". The file type is generally added with a period "." and usually automatically generated when an application creates a file.

Element-1	_Element-2	_Element-3	_Element-4	_Element-5	.Element-5
Last-Name	_Date	_Project	_File-Contents	_Version	.File-type

Dates

Dates should be written in the format **yyyymmdd**. No spaces, no dashes, no words, just 8 numbers representing the year, month, and day, with months and days that are from 1 - 9 being led by a 0. So, January 23, 2020, would be written **20200123**. And September 5, 2021 would be written **20210905**. When written this way, they will always be sorted by your computer from earliest date to latest date.

Dates are very important for things like data, because the date of collection has direct relevance. Dates are less important for things like figures and manuscripts, as these are derived from the already dated data.

Versions

Version tracking is achieved in file naming by adding _Vn where n is the version number. With each major change, we increase n by 1. So version 1 would read _V1 and when updated, it would read _V2.

Versions are very important for things like manuscripts and interpretations of data, such as figures and other visualizations, where we will continue to change and modify these throughout a project. Our data, however, while it has a collection date, should not be modified, and should not then be versioned.

2.3 An example

So what does this look like?

Say you're in BIOL 116 and you're working on your research project. Your research project involves:

- Preparing the beginning of a manuscript that states your research question, hypothesis, and proposed methods.
- Conducting your experiment and recording your data. This process might span more than one day.
- Updating your manuscript, describing any changes that were made to your methods
- Organizing the results of your experiment and interpreting and visualizing your data
- Updating your manuscript to include your results and your interpretation of these results, including a visual interpretation
- Completing your manuscript by discussing the importance of and / or limitations of the experiment, and finally producing a conclusion.

In this scenario, we have **1 project**, **1 manuscript**, **1 dataset**, and at least **1 figure**. In addition, our dataset is constructed from data collected over several days, and our manuscript is revised 3 times before final submission.

So, first we will come up with a project name, and then we will **date our data** and **version our figures and manuscript**. And we'll see how this evolves over the course of several days.

Day 1

We create the following file:

Pither_BIOL116RProject_Lab-report_V0.docx

This is our manuscript, so it will get a version, but no date. Looking at it, we quickly see that this is a lab report (Lab-report), authored by someone with the last name Pither (Pither) associated with a BIOL 116 Research Project (BIOL116RProject), that it has only just been created (V0), and that I should expect it to open in Microsoft Word (docx).

Let's imagine that I will put my research question, hypothesis, and methods in this document and submit it.

Day 2

Today, I conducted the first part of our experiment and collected some data. Now we have the following files:

```
Pither_20210921_BIOL116RProject_ph-data.csv
Pither_BIOL116RProject_Lab-report_V0.docx
```

We have not changed our manuscript, so there's no change to the name. We have collected some data though. We can easily see who collected this data (Pither), when it was collected (September 21, 2021), that it's connected to the BIOL 116 Research Project (BIOL116RProject), and that it's data related to PH exposure. Lastly, it is formatted as comma separated values (csv), which can be opened by any spreadsheet program or text editor.

Day 3-5

I continue to collect data over the next several days, and here is what my files now look like:

```
Pither_20210921_BIOL116RProject_ph-data.csv
Pither_20210922_BIOL116RProject_ph-data.csv
Pither_20210923_BIOL116RProject_ph-data.csv
Pither_20210924_BIOL116RProject_ph-data.csv
Pither_BIOL116RProject_Lab-report_V0.docx
```

Again, we have not changed our manuscript, so there's no change to the name. We have collected some more data though, all related to PH, and we have one file for each day, organized from earliest day of collection to most recent.

Day 6

Today, I did two things. I have no more data to collect, so I updated my manuscript to include any modifications made to my original methods section, I then submitted this. I also started to analyze my data; to do this, I merged all of the data that I have into a single file for analysis. Now my files look like this:

```
Pither_20210921_BIOL116RProject_ph-data.csv
Pither_20210922_BIOL116RProject_ph-data.csv
Pither_20210923_BIOL116RProject_ph-data.csv
Pither_20210924_BIOL116RProject_ph-data.csv
Pither_BIOL116RProject_Analysis_V0.xlsx
Pither_BIOL116RProject_Lab-report_V0.docx
Pither_BIOL116RProject_Lab-report_V1.docx
```

At this stage, I have my data collated into a document where I can work with it without impacting the original data. We can see that I have done this in Excel (xlsx), and that I should expect to be able to open this file in Excel. I also now have a V1 of my manuscript, as I have now added a new section to it; when submitting it, my TA knows that the file with V1 should have this updated section.

Day 7

Today, I built two visualizations using the data in my Analysis document, one linear regression and one bar plot of frequency counts; I save these as images to insert into my manuscript. I then updated my manuscript to include my results and these two figures and submitted V2 of my manuscript. Now my files look like this:

```
Pither_20210921_BIOL116RProject_ph-data.csv
Pither_20210922_BIOL116RProject_ph-data.csv
Pither_20210923_BIOL116RProject_ph-data.csv
Pither_20210924_BIOL116RProject_ph-data.csv
Pither_BIOL116RProject_Analysis_V0.xlsx
Pither_BIOL116RProject_Figure-freq-plot_V0.png
Pither_BIOL116RProject_Figure-linear-reg_V0.png
Pither_BIOL116RProject_Lab-report_V0.docx
Pither_BIOL116RProject_Lab-report_V1.docx
Pither_BIOL116RProject_Lab-report_V2.docx
```

We can start to see the advantage here of naming conventions. I can easily see which files are which, what they contain, and what their timeline of development

is. Also, my computer easily sorts these into meaningful categories - my data is grouped together, sorted by date. My analyses, figures, and manuscripts are all respectively grouped together and sorted by version.

Day 8

I got feedback that my linear regression model had an error in it. So I fixed this today, added the new figure into my manuscript, and wrote the discussion and conclusion sections. I'm now ready to submit. Here is what my files look like now (I will be submitting V3 of my manuscript):

```
Pither_20210921_BIOL116RProject_ph-data.csv
Pither_20210922_BIOL116RProject_ph-data.csv
Pither_20210923_BIOL116RProject_ph-data.csv
Pither_20210924_BIOL116RProject_ph-data.csv
Pither_BIOL116RProject_Analysis_V0.xlsx
Pither_BIOL116RProject_Figure-freq-plot_V0.png
Pither_BIOL116RProject_Figure-linear-reg_V0.png
Pither_BIOL116RProject_Figure-linear-reg_V1.png
Pither_BIOL116RProject_Lab-report_V0.docx
Pither_BIOL116RProject_Lab-report_V1.docx
Pither_BIOL116RProject_Lab-report_V2.docx
Pither_BIOL116RProject_Lab-report_V3.docx
```

Chapter 3

Directory Structure Conventions

Now that we've covered naming conventions, let's pretend that you store everything on your computer in a single folder. Imagine how long it would take you to find data you collected on a specific day a few years ago. Instead of keeping every document in a single place, we often organize our files using directory (aka folder) structures. This helps us save precious time and improve our productivity.

One major aspect of Open Science is ensuring transparency in the research process. This includes sharing files from all the steps of the research lifecycle (ie. a priori hypothesis, study design, data, analysis etc.) with others so that our research can be understood and replicated more easily.

The way we currently organize folders and files on our computer may make sense to us, but a problem arises when we need to share those folders and files with other people. A folder name that is meaningful for us may make no sense to another person. So let's cover some conventions to help you organize the files on your computer in a way that is meaningful to both you and others.

3.1 Directory Hierarchies

First, let's talk about how to properly structure a folder hierarchy.

The highest ranking folder is generally called the **root directory**, or sometimes the top-level folder. We'll call it the root directory here. This folder will contain all of the subfolders and files related to a particular project, including its data, analysis, lab reports etc. It will also contain what is called a readme file.

The structure should look something like the following:

```
Project-Folder/Experiment-Data/File-1
Project-Folder/Experiment-Data/File-2
Project-Folder/Experiment-Analysis/File-1
Project-Folder/Experiment-Report/File-1
Project-Folder/Experiment-Report/File-2
```

Here, our root folder is called Project-Folder and it contains three subdirectories, one for data, Experiment-Data, one for analyses, Experiment-Analysis, and one for a report Experiment-Report. Each subdirectory then contains one or many files.

3.2 Directory Naming

Key file naming conventions, such as avoiding special characters, are equally as important for directories as they are for naming files. Remember, we consider special characters to be anything other than letters in the English alphabet, numbers from 1-9, dashes "-", and underscores "_".

In case there is any doubt, here are some examples of what are considered special characters - characters you want to avoid!

% & < > : " / | ? * { } $! ' @ + ` =

Remember spaces, " ", qualify as special characters when it comes to file naming.

Remember when we name the root folder we want to clearly communicate what the project is. And similarly, within this root folder, we want clearly labeled subfolders for each relevant aspect of the project. Common discrete subdirectories include ones for figures, data, manuscript etc.

3.3 readme files and data dictionaries

When naming files we embed metadata into our file naming conventions to encode relevant information for the reader. But we can only store so much information in a file name. So we also include three additional files

- one called _README that resides in our root folder and elaborates on the contents of our folder structure;
- a second, also called _README, but that resides in our data directory and discusses some of the particulars of the how, where, and who of the actual data collection; and
- one called _DATA-DICTIONARY that also resides in our data directory and elaborates on how our data is stored and organized.

These files - containing a brief description of the major folder contents, naming conventions that were followed, and data structure - are critical for transparency and reproducibility, because they allow others to easily understand the contents of your directory and data without needing to ask you. This is especially helpful when working with a group or sharing directories with others.

General rules

A readme file and data dictionary should

- exist in at least two locations, the root directory and the data directory.
- be prepended with an underscore "_". This will push these files to the top of the directory for easy access.
- _README and _DATA-DICTIONARY files should be in all caps, so they really stand out; this should be the first thing you look at when looking at any directory or folder, as this is your guide to its contents.

File formats

readme files and data dictionaries should be written in plain text, this will ensure that the file describing your project and all of its files can be opened on any computer. You will often see readme files called _README.txt or _DATA-DICTIONARY.txt.

Our example readme and data dictionaries use a plain text format called markdown.

markdown

We recommend that you create your readme files as markdown, a way of formatting plain text files, allowing us to provide additional meaning to our content. For example, in plain text, if we want to emphasize content, we don't really have a way of doing this. In markdown, we can use italics and bolding if needed. We can also create lists and tables.

Learn the basic syntax of markdown here.

3.4 Root folder readme

To create a root folder readme file, use any markdown or text editor (ie. Typora, notepad etc), open a new file and save it to the root folder for your project ensuring the file type is .md.

Name your readme file _README.md.

Next we want to add some content to our _README.md. The purpose of this document is to describe the directory structure of our project. To adequately describe our directory structure we should include:

- A brief description of the project or purpose of the root folder
- Date when the root folder was created and who created it
- Date when the readme file was last updated and who updated it
- A brief description of the contents of each major folder within the root folder
- A brief description of file naming conventions used within the directory

To see an example root directory readme file click here.

3.5 Data directory readme

Next, we want to create another readme file but this file will be placed within the subfolder that contains our project's data. To do this, open any markdown or text editor (ie. Typora, notepad etc), open a new file and save it to the data subfolder for your project ensuring the file type is .md. Name your readme file _README.md.

The purpose of this readme file is to provide a description of data collection methods. We will include:

- Date when the data directory was created and who created it
- Date when the readme file was updated and who updated it
- A brief description of each data that was collected, the methods used for collection, and the date range for when each dataset was collected
- A description of who was involved in data collection
- A brief description of where the data was collected

To see an example data directory readme file click here.

3.6 Data dictionary

Lastly, we need to create a data dictionary which elaborates on how our data is stored and organized. To do this, open any markdown or text editor (ie. Typora, notepad etc), open a new file and save it to the data subfolder for your project. This time we will save the file as _DATA-DICTIONARY.md.

A data dictionary helps others understand the meaning of each element in your datasets within the broader context of the project. Typically you will have an individual readme file for each dataset. This file should include:

- Date when the data dictionary was created and who created it
- Date when the data dictionary file was updated and who updated it
- A description of the raw data file
- A description of each variable for all datasets including data type, units, number of levels if categorical, and a description of variable levels where relevant
- When describing variables you need to provide the full names and definitions of each variable because often variables are abbreviated in datasets

To see an example data dictionary click here.

3.7 Example BIOL 116

In BIOL 116, we used a flat folder structure to hold all of our files. In the example used, no readme files were created, as we made the assumption that the project was "simple" enough in its structure to not warrant a readme file. On reflection, this was an oversight and we probably should have created a readme file describing what was in each file. Neither did we create a data dictionary. We'll do better on future assignments now that we know about the value of both forms of documentation. Anyway, in that example, we ended up with one folder of files that looked like the following before submitting our final assignment:

```
Pither_20210921_BIOL116RProject_ph-data.csv
Pither_20210922_BIOL116RProject_ph-data.csv
Pither_20210923_BIOL116RProject_ph-data.csv
Pither_20210924_BIOL116RProject_ph-data.csv
Pither_BIOL116RProject_Analysis_V0.xlsx
Pither_BIOL116RProject_Figure-freq-plot_V0.png
Pither_BIOL116RProject_Figure-linear-reg_V0.png
Pither_BIOL116RProject_Figure-linear-reg_V1.png
Pither_BIOL116RProject_Lab-report_V0.docx
Pither_BIOL116RProject_Lab-report_V1.docx
Pither_BIOL116RProject_Lab-report_V2.docx
Pither_BIOL116RProject_Lab-report_V3.docx
```

We can see that this might start to get unwieldy if we have a few more files joining the party. So let's break this apart into folders…

Top Level folder

```
BIOL116RProject/
```

Inside of BIOL116RProject we have one file and four subdirectories:

```
 _README.md
Data/
Analysis/
Figures/
Report/
```

Note that we created a _README.md file to describe our directory structure. We'll now distribute our files across these folders…

Data Folder

Creating a _README.md and a _DATA-DICTIONARY.md to describe our data files and their contents…

```
_DATA-DICTIONARY.md
_README.md
Pither_20210921_BIOL116RProject_ph-data.csv
Pither_20210922_BIOL116RProject_ph-data.csv
Pither_20210923_BIOL116RProject_ph-data.csv
Pither_20210924_BIOL116RProject_ph-data.csv
```

Analysis Folder

```
Pither_BIOL116RProject_Analysis_V0.xlsx
```

Figures Folder

```
Pither_BIOL116RProject_Figure-freq-plot_V0.png
Pither_BIOL116RProject_Figure-linear-reg_V0.png
Pither_BIOL116RProject_Figure-linear-reg_V1.png
```

Report Folder

```
Pither_BIOL116RProject_Lab-report_V0.docx
Pither_BIOL116RProject_Lab-report_V1.docx
Pither_BIOL116RProject_Lab-report_V2.docx
Pither_BIOL116RProject_Lab-report_V3.docx
```

Screenshot

And finally a screenshot from our desktop file manager…

3.8 Example BIOL 125

Let's work through another example where we start our project off using both appropriate directory structure and file naming conventions. Say you're a student in BIOL 125 working on a research project testing mealworm food preferences…

Day 1

On day one of our research project, we are asked to prepare the beginning of a lab report that states our research question, hypothesis, and proposed methods. First, we need to create the root folder for our project:

```
BIOL125MealwormProject/
```

Within our root folder we create a _README.md file to describe our directory structure. Let's add our project name, date the folder was created and who created it, a short description of the project, group member names, file structure (major folders and their proposed content), and naming conventions to this readme file. Your file should look something like this:

_README.md

Since we have just started our project, there won't be files in most subfolders we create. However, it's good to have the skeleton of what we want our directory to look like so everyone in the group places new files in the correct location. Later, if needed, we can modify our directory structure and update our readme file to reflect those changes.

Since we will be using the naming conventions outlined in Chapter 1, we can list those naming conventions here. It may seem strange to outline the naming conventions for documents that haven't been created yet, but having a strategy for naming files from the beginning of the project is very important. It ensures everyone is following the same set of rules when they add or edit files in the project, which helps everyone stay on the same page when working in a shared directory.

Now that our root folder and root readme files are set up, we need to create the subfolders within BIOL125MealwormProject/. Since we outlined the major subfolders in our readme file as Report, Data, Analysis, and Figures, we'll use these same names when we create the subfolders. Finally, we can open up a new Word document for our lab report and save it to the Report folder using the appropriate naming conventions.

```
Pither_BIOL125MealwormProject_report_V0.docx
```

Here is what our project directory looks like so far:

Day 2

Today, we completed a pilot experiment and collected some data. We saved this data file into our project's corresponding Data folder using appropriate naming conventions.

New files:

```
Pither_20210915_BIOL125MealwormProject_food-preference-data.csv
```

Since we have added our first dataset into our project folder, we need to create a corresponding data directory _README.md and _DATA-DICTIONMARY.md.

Let's start by creating the data directory readme and provide a description of our data set, collection methods, who collected the data, and where it was collected. It should look something like this:

_README.md

Next, let's create a data dictionary for our new dataset. It should look something like this:

_DATA-DICTIONARY.md

Now our project directory looks like this:

Day 3

Now that our group has completed its pilot project, we decided to expand our data collection and start recording how far mealworms are willing to travel to get each food. In addition to this new distance data we continued to collect data on food preferences.

New files:

```
Pither_20210916_BIOL125MealwormProject_food-preference-data.csv
Pither_20210916_BIOL125MealwormProject_distance-data.csv
```

Since we have a new dataset we'll have to update our data directory readme file with a description of the new dataset. Remember to note down the date it was updated and who it was updated by. Our updated data directory _README.md file should look something like this:

_README.md

Now our updated data directory readme file includes descriptions for both datasets.

In the interest of organization, let's keep our food preferences and distance data in their own subfolders. So, we'll create two new subfolders within the Data folder. We'll call one Food-preferences/ and the other Distance/. This way we can organize csv files into the folder for the corresponding dataset. After doing this, we need to update the _README.md in our root folder, since we've modified our directory structure. This readme should now look something like this:

_README.md

Next, let's also create a data dictionary for our new dataset within our distance subfolder. Remember to note down the date it was updated and who it was updated by. It should look something like this:

_DATA-DICTIONARY.md

Since we made some updates to our project design and methods, I'll also go ahead and updated our lab report to reflect those changes alongside justification for the changes. Then, I'll be sure to save my updated lab report using the appropriate naming conventions.

```
Pither_BIOL125MealwormProject_report_V1.docx
```

Now our project directory looks like:

Day 4-5

Over these days, we collected our last rounds of data, created some figures, and analyzed the data. So we have a bunch of new files that we need to make sure are placed correctly within our project directory.

New files:

```
Pither_20210917_BIOL125MealwormProject_food-preference-data.csv
Pither_20210917_BIOL125MealwormProject_distance-data.csv
Pither_20210918_BIOL125MealwormProject_distance-data.csv
Pither_20210918_BIOL125MealwormProject_Analysis_V0.xlsx
Pither_20210918_BIOL125MealwormProject_Figure-mosaic_V0.png
Pither_20210918_BIOL125MealwormProject_Figure-scatterplot_V0.png
```

We'll save all 3 new data files into the Data/ subfolder of our directory. Since we've already described these datasets in the data directory _README.md file and have a corresponding _DATA-DICTIONARY.md for both, there are no more updates needed.

Next, we'll save our analysis into the Analysis/ subfolder and the figures into the Figure/ subfolder.

Now our project directory looks like:

Day 6

Today is the last day of our project and we completed the final copy of our report. We will make sure this is saved into the Report folder using the appropriate naming conventions.

New files:

```
Pither_BIOL125MealwormProject_report_V2.docx
```

Our final directory looks like this:

We can start to see that if you were to share your entire project directory with another person it would be relatively easy for them to locate files and understand the meaning behind each document in our project. They would also know when changes were made and who made these changes, so if they had any questions, they'd know exactly who to ask!

Chapter 4

Tidy data

We've talked about file naming, directory structures, and documentation to ensure accessible, interpretable, and transparent data. Now it's time to talk about organizing individual variables within a given file. When well organized, data values can be effectively analyzed, summarized, and visualized. When not, they can be onerous to work with and risk misinterpretation.

In general, your data files should adhere to the principles of "tidy data". Tidy data is governed by the following 3 rules[1]:

- Each variable must have its own column.
- Each observation must have its own row.
- Each value must have its own cell.

It's easy to veer from these rules, as it's often easier to collect data using data collection tools that violate these rules. When this is the case, we need to know how to re-organize our data to make it "tidy".

4.1 Wide Data

Non-tidy data - sometimes called "wide" data as it tends to use more columns, but fewer rows - tends to lump observations together in cells. It is often easier to collect data in this way.

So, say we collected data about the number of trout caught at local lakes across several days. We might end up with the following data tables if we used a separate table, sheet of paper etc. to record our findings.

[1]See: Wickham, H. & Grolemund, G. (2017). Tidy Data. *In R for Data Science*.

Site	Trout_Caught_Day_1
Mabel-lake	1
Postill-lake	3
Ellison-lake	0

Site	Trout_Caught_Day_2
Mabel-lake	1
Postill-lake	3
Ellison-lake	0

Site	Trout_Caught_Day_3
Mabel-lake	1
Postill-lake	3
Ellison-lake	0

It is also very likely though that we set up an Excel sheet where we recorded the site as the first column and our days and fish caught combined in subsequent columns, one column for each day. Even if we hadn't collected our data this way, we might be tempted to group our above data together for analysis or assignment submission in this way.

Doing this, we'd end up with a table something like the following:

Site	Trout_Caught_Day_1	Trout_Caught_Day_2	Trout_Caught_Day_3
Mabel-lake	1	3	3
Postill-lake	3	4	5
Ellison-Lake	0	5	1

But for analysis - for "tidy" data - we want one column per variable. In this case, we have three variables:

- site
- day
- quantity caught

So let's get this cleaned up…

4.2 Tidy Data

In the previous example, our data were organized where day and quantity caught shared common columns. That is, not every variable had a dedicated column and consequently, not every variable had a value in every given cell - day did not have any cell values.

Tidy data breaks this down and reserves on column per variable and one row per observation. Remember, we have three variables: site, day, and quantity caught. So let's transform this…

First, working with a collection tool where we have one table per day:

Site	Day	Trout_Caught
Mabel-lake	1	1
Postill-lake	1	3
Ellison-lake	1	0

Site	Day	Trout_Caught
Mabel-lake	2	3
Postill-lake	2	4
Ellison-lake	2	5

Site	Day	Trout_Caught
Mabel-lake	3	3
Postill-lake	3	5
Ellison-lake	3	1

And second, gathering this data into a single dataset, sorted by site:

Site	Day	Trout_Caught
Mabel-lake	1	1
Mabel-lake	2	3
Mabel-lake	3	3
Postill-lake	1	3
Postill-lake	2	4
Postill-lake	3	5
Ellison-lake	1	0
Ellison-lake	2	5
Ellison-lake	3	1

Now that's tidy data!

4.3 Side by Side Comparison

Wide Data

Site	Trout_Caught_Day_1	Trout_Caught_Day_2	Trout_Caught_Day_3
Mabel-lake	1	3	3
Postill-lake	3	4	5
Ellison-Lake	0	5	1

Tidy Data

Site	Day	Trout_Caught
Mabel-lake	1	1
Mabel-lake	2	3
Mabel-lake	3	3
Postill-lake	1	3
Postill-lake	2	4
Postill-lake	3	5
Ellison-lake	1	0
Ellison-lake	2	5
Ellison-lake	3	1