Name: Caroline Kaney

Date: 19 Feb, 2021

**Goal: learn how to create good spreadsheets!**

**Purpose:**

1. **What is a spreadsheet and how do they work**
2. **What separates a good dataset from a bad one**
3. **How can we apply best practices to create a spreadsheet that is useful to other**

What is clean data? Clean data are in a format that is ready to analyze. Characteristics of clean data include data that are:

Characteristics of clean data is complete data with no empty cells, being consistent with the data and names, putting just one thing in one cell, and don’t use font colors or highlighter in data.

Common symptoms of messy data include data that contain:

* Special characters (e.g. commas in numeric values)
* Numeric values stored as text/character data types
* Duplicate rows
* White space
* Missing data
* Zeros instead of null values
* “highlights” and colors used to denote information
* More than one type of info stored in a cell

# Part A: For each spreadsheet identify at least one thing that makes it problematic. Then describe how you might go about fixing it (i.e., what could you do to make it ‘clean.’

**For help see:** [**https://datacarpentry.org/spreadsheets-socialsci/02-common-mistakes/index.html**](https://datacarpentry.org/spreadsheets-socialsci/02-common-mistakes/index.html) **or watch the online video I filmed and put on AsULearn for Week 4**

1. **Sheet 1**

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| **Sample** | **Date excavated** | **Date analyzed** | **Element** |
| **1** | **5/9/2019** | **1/5/20** | **Radius** |
| **2** | **5/1/2019** | **1/5/20** | **Radius** |
| **3** | **5/3/2019** | **1/6/20** | **Ulna** |
| **4** | **W** | **1/6** | **femur** |

**Problem:**

There is an empty row on the top of the data set, the dates are not consistent, and there is an unknown or key for the “W”.

**Better version:**

I would erase the top, empty row, make all the dates consistent with (YYYY-MM-DD), and create a key or clear up what “W” represents.

1. **Sheet 2**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Pit A |  | Pit B |  |  |
|  |  |  |  |  |  |
| Munsell | weight | Date | Munsell | weight | Date |
| 10 yr 4/3 | 1.5 | 2020-02-03 | 10 yr 2/3 | 7.5 | 2020-02-05 |
| 10 yr 4/3 | 1.7 |  | 10 yr 2/3 | 3 |  |
| 10 yr 4/3 | 2.0 |  | 10 yr 2/3 |  |  |
| 10 yr 4/3 | 2.1 | 2020-02-04 | 10 yr 2/3 | 3.3 |  |
| 10 yr 4/4 | 2 |  | 10 yr 2/3 | 9 |  |

**Problem:**

There are empty cells in both the “Date” and “Weight” columns and the second row is empty. There are also more than one thing in a single cell in the “Munsell” columns along with missing measurements for the “Weight” data.

**Better version:**

I would delete the second, empty row. Create an extra column for the “Munsell” data so that there aren’t more than one thing in a single cell. I would also create an extra column for the measurement of the weight data so that it is clearer. Also put NA in the empty cells or fill those in with data.

1. **Sheet 3**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  | Site | elements | Biological sex | Analysist |  |
|  | Skeleton UW-503 |  |  |  |  |  |
| Transect A |  | Oblion | Pelvis | Male | GP |  |
|  |  |  | femur | male | GP |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  | Skeleton UW-504 |  |  |  |  |  |
| Transect B |  | PineTree | Cranium | Female | SL |  |
|  |  |  | Pelvis | Female | SL |  |
|  |  |  | Tibia | NA | SL |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

**Problem:**

There are MANY empty cells and just chaos all around.

**Better version:**

I would delete all the empty rows and columns that are not needed in the data set and make it more concise. Then fill in any empty cells with NA or missing data. I would also consider moving Transect A and B on the top of the data table so that it makes it easier to understand.

1. **Sheet 4**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Week 1 |  | Week 2 |  | Week 3 |  |  |
| ID | SEX | date | weight | date | weight | Date | weight |  |
| 002 | M |  |  |  |  |  |  |  |
| 005 | F |  |  |  |  |  |  |  |
| 006 | F |  |  |  |  |  |  |  |
| 010 | M |  |  |  |  |  |  |  |
| 012 | M |  |  |  |  |  |  |  |
| 015 | M |  |  |  |  |  |  |  |

**Problem:**

There is missing data and many empty cells.

**Better version:**

Not sure what else to do beside fill in the data and the empty cells.

1. **Sheet 5**

|  |  |  |
| --- | --- | --- |
| Sample | date | Glucose level |
| 101 | 2019-01-14 | 150.5 |
| 102 | 2019-01-14 | 93.6 |
| 103 | 2019-01-14 | 99.5 |
| 104 | 2019-01-15 | 108.0 |
| 105 | 2019-01-15 | 7.9 |
| 106 | 2019-01-15 | 102.7 |

**Problem:**

There is a highlighted cell in the third row.

**Better version:**

Unhighlight the cell

# Part B: fixing/data cleaning examples

Sadly, most folks don’t follow the above suggestions. Here is an example, modified from a colleague’s assignment, that lets you see how to work with messy data

You can download the data for this part on [this google drive folder](https://drive.google.com/drive/u/1/folders/1zQtdbwQuVrxtHrFG7lXdW02ZV0r5cKnu): (https://drive.google.com/drive/u/1/folders/1zQtdbwQuVrxtHrFG7lXdW02ZV0r5cKnu)

The file are from New York City and give info on real estate sales for different boroughs ( administrative divisions that make up New York City) …Let’s say you wanted to help a friend who was trying to learn more about this and they gave you these files and asked you what you can learn. Put the data into your current project folder so that you can access it easily (if you don’t have Excel on your machine don’t worry about that. You should be able to open the file just to view it on the Drive)

## Setup

To do this part of the lab make sure you run the following code as we will need these R libraries.

library(tidyverse)

install.packages("janitor") #this helps to clean our data

library(janitor)

library(readxl) # this package comes when you install the tidyverse

#package but is only loaded individually

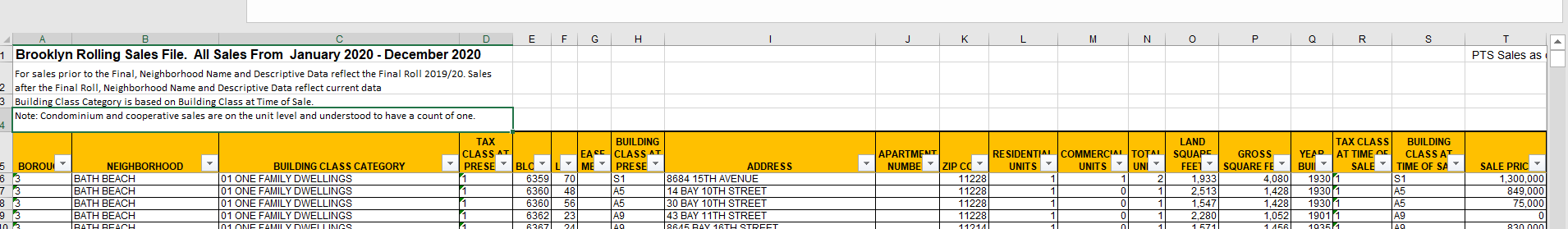
## Step 1: take a look at the raw data

* + Take a look at the Excel sheet either in the Google folder or after downloading and make some notes about what problems you see that you may want to edit/fix as you go

## Step 2: get the data into R

In the folder you will see multiple versions of essentially the same file (one is an excel sheet and the other is in a CSV (with some of the text edited out…not ideal but c’est la vie).

* + - If you have Excel on your computer open it and take a look…you will see something like I copied below. Note that the first few lines are info that we don’t need in our data table,,,



* + - We are going to read the data in with read\_xls(), a function in the readxl package
    - The function read\_xls takes a number of arguments (the stuff that goes in between the parenthesis)
* path = where on your machine the file is stored
* sheet = the name of the sheet (some excel files have multiple sheets)
* na = what to count as missing value (by default it counts blank cells as missing
* skip = Minimum number of rows to skip before reading anything, be it column names or data. Leading empty rows are automatically skipped, so this is a lower bound. Ignored if range is give
  + - looking at these data it seems that all we need here is to tell it to skip the first 4 line
    - on my computer, I have the data in my working directory in a folder called “datasets.” You will have to change this to wherever you put the data. If you are having trouble PLEASE reach out. Getting a handle on this is one of the biggest hurdles and I am here to help!

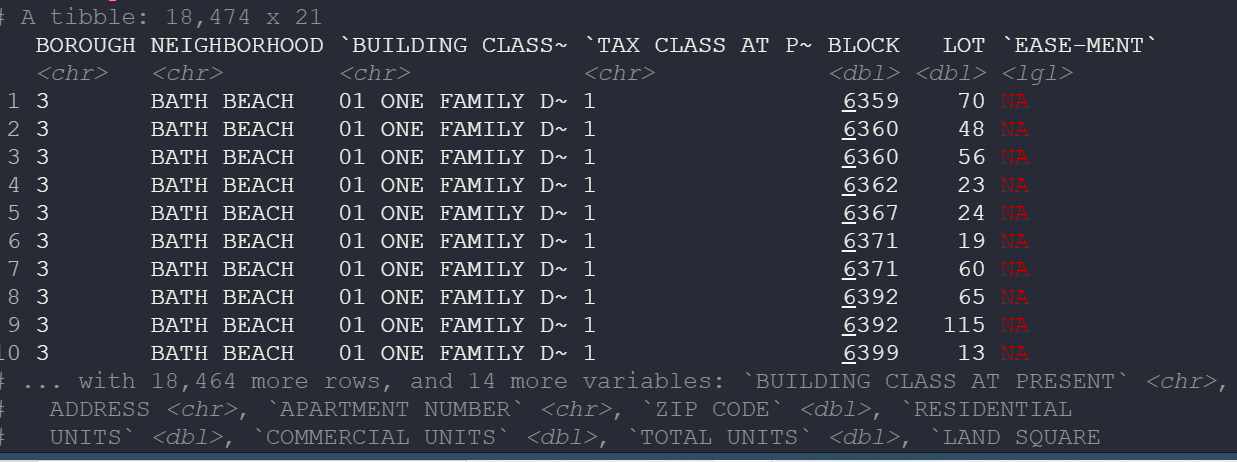
brooklyn <- read\_xls(path ="datasets/rollingsales\_brooklyn.xls", skip = 4)

Note: the ‘path’ is where the data sheet is located on the computer. If you write path = “” and hit the tab key between the quotation marks you should see the files in your current working directory pop up. That will let you choose the file you want

* + We now have the dataset stored as an object called “brooklyn.” To examine it we can just type the function’s name into R

brooklyn

you will see something like this:



* + Note that it tells us there are 11,874 rows and 21 columns of data…that is a lot of data!
  + It also shows the first few rows and data.
  + Now, enter the following into R to create two new objects that have the same data source but use different *functions* for reading in the data (read.csv and read\_csv). Note that this time we are using the .csv file rather than the .xls file

brooklyn\_csv\_base <- read.csv("datasets/rollingsales\_brooklyn\_csv.csv", skip = 4)

brooklyn\_csv\_readr <- read\_csv("datasets/rollingsales\_brooklyn\_csv.csv", skip = 4)

Now compare the three different datasets….what differences do you see in how the data are stored? One way to do this is with the Tidyverse’s glimpse() function, which gives a nice summary of the object. You will notice that with glimpse you can see the name of each column and then the class of that column. Pay attention to each object and when the class of a column is different.

glimpse(brooklyn\_csv\_base)

glimpse(brooklyn\_csv\_readr)

glimpse(brooklyn)

jot down some notes about how these objects differ:

glimpse(brooklyn\_csv\_base) : ­­­uses <int> and <chr> beside the data and the categories on the left are mainly letters and numbers.

glimpse(brooklyn\_csv\_readr) : uses <abl>, <chr>, and <lgl> beside the data and is similar to the last by having numbers and letters in the left column

glimpse(brooklyn) : uses <abl>, <chr>, <lgl>, and <dttm> beside the data and has the full column name written out on the left side.

Step 2: connecting datasets together

Note how each borough is setup the same in the sheets. This is a happy thing for us. Lets say we want to look at all the data at one go!

brooklyn <- read\_excel("rollingsales\_brooklyn.xls", skip = 4)

bronx <- read\_excel("rollingsales\_bronx.xls", skip = 4)

manhattan <- read\_excel("rollingsales\_manhattan.xls", skip = 4)

staten\_island <- read\_excel("rollingsales\_statenisland.xls", skip = 4)

queens <- read\_excel("rollingsales\_queens.xls", skip = 4)

# Bind all dataframes into one, save as "NYC\_property\_sales"

NYC\_property\_sales <- bind\_rows(brooklyn, bronx, manhattan, staten\_island, queens)

glimpse(NYC\_property\_sales)

Now we have all the data in one dataset. Note the column “BOROUGH” is a numerical code for the borough so that info is still present. We could rename them if we wanted to…

Step 3: Clean up the data: Let’s circle back to Brooklyn dataset from above

* Take a look at the col names :

colnames(brooklyn)

take a moment and jot down some thoughts about what we might want to fix about the column names

there are some spaces in the names and in all caps.

To me, a few things stick out about the names of the columns that may make data analysis tricky

* 1. All capitalized
  2. Spaces in the col names

We can fix this in a number of ways:

* **To get the capitalization fixed:**

Take a look at the function below. A few things are happening here. The “%>%” is a function from the tidyverse that makes coding a bit easier. It is called a pipe and basically it says “then”. So you can read the code as “take the column names from brooklyn then use the str\_to\_title function. This function, also part of the Tidyverse, take a string (an ordered sequences of characters) and changes the case of the letters. Play around and see what other str\_to\_ functions exist and what they do

colnames(brooklyn) %>% str\_to\_title()

* But this code doesn’t change anything. to save this we just assign it to colname(brooklyn)

colnames(brooklyn) <- colnames(brooklyn) %>% str\_to\_title()

**Now, we want to remove the spaces**

Thankfully Tidyverse gives us that with the useful str\_replace\_all() function, Here, the str\_replace\_all() takes two arguments. The first one is what we want to replace and the second it what we want to put in its place.

colnames(brooklyn) %>% str\_replace\_all(" ", "\_")

In this instance, we are replacing a space with a underscore. How would you replace the space with it with a . ?

To replace a space with “.” The code would be str\_replace\_all(“ “,”.”)

Remember to save it:

colnames(brooklyn) <- colnames(brooklyn) %>% str\_replace\_all(" ", "\_")

fun fact: you could do this all in a single line like this: colnames(Brooklyn) <- colnames(brooklyn) %>% str\_to\_title() %>% str\_replace\_all(" ", "\_")

**Protip:**

Now that you did that I’m going to teach you a trick using the janitor package we installed at the beginning. In that package is a function called clean\_names() which will change the case and remove spaces in one go!

brooklyn %>% clean\_names()

# Part 3: Try it on your dataset

Take a look at the data you are examining for this course. Thinking about what we talked about this week and in this lab is there anything in your dataset that needs to be ‘cleaned to that it can work well. If so, what steps might you think about here

# Part 4: assessment

1. How can you apply ‘clean’ spreadsheet techniques to other work you do?

I can apply ‘clean’ spreadsheet techniques to other work I do by being consistent in all my work and data, making it easier to read and understand by have no missing parts and using proper capitalization.

1. After doing this lab, what sorts of reasons might you give for the importance of good data organization?

Good data organization is important for an easy and understandable data set. It will be easier to work with and analyze.

1. “Reading in” data to R, especially *messy* data, is one of the most frustrating parts of data analysis. Do you think you are getting the hang of it? What could make this easier?

I am definitely getting the hang of it! There was a small blip with getting the data into R but once that was figured out it was smooth sailing. I personally need step by step instructions to understand new ways of doing things and the way you lay things out makes it easier.

1. Next week we are starting to think about data visualization. For your dataset what sorts of figures might you want to make?

For my dataset, the figures I might want to make include some sort of bar graph to show the reptations in the script of “The Office” and how that relates to the characters and maybe the rating of the episode or the season… not too sure yet.