Date Created: 2021 June 21

Date Updated: 2021 June 21

To the Reader:

This documents is meant to serve as a generalized checklist for evaluating
a submitted dataset as part of the "ZIG Data Harmonization Team". Please note that is only to serve as a template, and that you may need to expand beyond this template in the event the data value checking and cleaning steps are more intensive or demanding.

Directory Structures:

- Be sure that you have modified the derived products output directory as well as the QC figures directory. This should be in the first section of the script.
    - The **derived_products** directory should be named something like `<DataProviderLastName>_<LakeName>_disaggregated`, meaning that the data are kept at the original spatial and temporal resolutions, but they have been checking for correct values. These are the data that will be used for the data product as well as the analytical datasets for successive projects.
    - The **XXXX_qc** directory should be located within the figures directory and contain all figures produced from the associated harmonization script.
- Be sure to build all scripts using a relative file path, where the "home" directory is the **scripts** folder. All data are read from the **data/inputs** directory and written to the **data/derived_products** directory. If you are new(er) to relative file paths, it means that there should never be a hardcoded directory within your script (this helps promote future reproducibility without need for machine/user specific file nomenclature). To check if your direcotories are set up correctly, you should use `getwd()` within your R console. If it is *NOT* set to ~/GLEON_ZIG/scripts, then you can use the RStudio GUI by clicking `Set Working Directory > Source to File Location` and your directory *should* be properly set.

Assessing Quality Control

- **Very Important:** Please remember that we are operating under the "stay in your lane doctrine". This means that each member of the team will have their own piece that they are working on, and no one else should touch someone else's script. This will help prevent merge conflicts in the long run. To do this efficiently, everyone will need to make their own R script (1 R script per dataset). It should follow the nomenclature: `00aa_<DataProviderLastName>_<LakeName>_<DataCleanerLastName>.R` -- example `00a_currie_ontario_meyer.R`. The `00` refers to this script as being labeled a "cleaning script". The letters following `00` refer to the unique identifier this script has. These identifiers will be very important when we submit the full workflow, and they help our data harmonization routine retain a consistent structure.
- The companion R script (which you will adapt/create based on MFM and SEF's scripts) is meant to largely capture high-level issues and signal where low-level issues may occur. A high level issue would be something like one year/month of data having abnormally high values (potentially indicative of lessened data quality). While there is nothing we can do about the data's quality assurance, this scripted routine assures that the original data integrity and quality are maintained throughout are aggregation procedure.
- In an ideal case, you would manually check plots and in the event of unexpected values pattern - you can manually investigate those data and document abnormalities. This process will also help expedite analysis down the road, as those performing the analysis may have questions about data integrity arise as they begin to work with the dataset.
- There may be instances where you are able to work more inefficiently by manually inspecting certain dates and timepoints within R. That is 100% okay, but that checking should be documented in your companion README document. MFM created a template for what this document could/should look like within the `derived_products/obertegger_tovel_disaggregated` directory. While you are welcome to change the format, each README document should contain the same information, at the very least.
- While each script will be unique to each dataset, there should be some high-level checks to think about:

- Are values formatted correctly in the script?
- When assessing distributions of values, do they generally fall in a similar and acceptable range?
- Are taxa spelled consistently throughout the CSVs?
- Does each measurement have an associated method? (With the exception of DO…)
- Are there any issues that you think we should bring up with the data provider?
  -Are you outputing each cleaned tab to a unique derived_products directory?

Formatting your scripts

The script can be divided into the following sections:

1. Lake information
2. Station information
3. Water Parameters
4. Taxa List
5. Zooplankton Abundance
6. Zooplankton Length
7. Lake Timeline
8. Equipment
9. Additional Data
10. Joins

Because data were submitted as a .xlsx format, you will likely need to use the `read_excel()` function from the `readxl` package in R. All outputs should be in a CSV format though.

A quick trick to see if something is strange in your data -- when you are reading in data, `read_excel()` assign any mixed column type (e.g., numbers, special characters, letters) as a "character" type. If the expected column should be a numeric type, this can help you flag columns that will need attention. The most likely culprit may be NAs entered as strings into the submitted data or the < sign being inserted as a value being below a minimal detection limit.