- You are allowed at most one `DATA` steps and six `PROC` steps to complete this assignment.

Specifications:

- Create the InputDS *libref* and RawData *fileref* that are both associated with the data folder on your shared drive. As always, do this using relative paths. Also using relative paths that point directly to your HW2 folder, create your *libref* and *fileref* that are associated with the single location where you want to save your results from HW2. (You won't be using this *fileref* in this assignment, but I want you to practice setting it up.)
- As with all data-based reports, pay attention to titles and footnotes - they aren't just there to copy and paste! They often give crucial information about the report you've been asked to present! This is true for almost all of our assignments and is definitely true for this assignment. It is often up to you to figure out how to apply the restrictions described in the report based on the variables you have available and the coding tools we have already learned.
- Read in the Baseball.dat file data set from your shared library on the L drive and use it to produce a report that matches my reports (plural!) found in HW2 Duggins Baseball Report.pdf and HW2 Duggins Baseball Report.rtf. There are a few elements you'll need that cannot be inferred from the provided report, for those I've provided these pointers! (Note: just because this data set is about baseball, there is no baseball-related knowledge that is required to complete it! Focus on working with the data logically. We didn't need to know anything about shoes to finish HW1!)
  - The PDF uses the Journal style and the RTF uses the Sapphire style.
  - Name your files the way we did before, including your name as part of the file. For example, Tony Stark would create a file called HW2 Stark Baseball Report.
  - Subtitles use 10pt font and footnotes use 8pt font.
  - Variable names and labels for this data set tend to make use of some abbreviations: BB for Walks, RBI for Runs Batted In, and HR for Home Runs. The Position(s) variable uses lots of abbreviations, but we don't have any reason to worry about those for this assignment.
  - The two reports are not identical. The RTF includes some results that are excluded from the PDF. You already know how to identify and select/exclude output objects - but here you need to selectively place them in a particular destination. To do this, you'll need the following statements - it is up to you to figure out where to put them!
    * `ODS PDF EXCLUDE NONE;`
    * `ODS PDF EXCLUDE ALL;`
  - At several points in the report, I have applied a custom format to the Salary variable. That format is named Salary and is located in the InputDS library. As discussed in class and in your reading, it is possible to use a format from a permanent library. To use it in this assignment, you'll need to do the following:
    1. Include an `OPTIONS` statement that includes `FMTSEARCH = (InputDS)` after you've established this library. This will give you access to all formats in this library. (While you're at it - throw in a `NODATE` option in the same statement to get rid of the dates and times in your report headers.)
    2. Print out the format (only in the RTF) to see its definition. If necessary, you can choose which formats are included in your printout by using the SELECT statement and naming the format(s) you want to see.
    3. Do not redefine the format yourself! The goal here is to use one that has been provided to you!
  - The labels are rather long, and make the `PROC MEANS` results look pretty messy. Just like `PROC PRINT` has a `LABEL` option, `PROC MEANS` has a `NOLABELS` option - use it!
  - There are many players for whom the Salary is missing. However, we don't want to exclude these players. In both the `MEANS` and `FREQ` steps, these will be automatically filtered out because of the `CLASS` and `TABLES` statements, respectively. To fix this, use the `MISSING` option in both of these statements.

    In these cases, as in many statements in SAS, options are applied after a forward slash in the statement you want to modify. For example, saying `CLASS Avar Bvar / MISSING;` adds the `MISSING` option to this `CLASS` statement.
- As always, I strongly urge you to make a plan and write out your code by hand first. Going to SAS and putting in the code should be one of the last steps of your process. You have all the resources (notes, lectures, slides, assignments, PPC, etc.) necessary to complete these assignments - it's just about practicing putting the pieces together in a way that meets the required goals!