Quick SAS Tips

Heide Jackson
heidej@umd.edu

University of Maryland Population Research Center

August 2019

High Level Things to Know

- SAS is a statistical software package well suited to handling large data sets
- Federal data sets are commonly stored as SAS data files (these end in the extension sas7bdat)
- SAS can handle multiple data sets simultaneously and keep them in its remote memory.
- SAS is not case sensitive and has extensions for using other languages particularly SQL and Python.
- Sas supports comments within /* */ symbols.

SAS Versions

- SAS 9,4 is available for download for University of Maryland affiliates.
- SAS can be accessed online via SAS Studio

The Basics of Doing Anything in SAS

Loading in Data

- Most data is loaded, modified and saved within a data statement. This corresponds to:

```
data new; /*Saves new data*/
set existing; /*Loads existing data set*/
run; /*Runs or compiles the statement*/
```

The Basics of Doing Anything in SAS

Analyzing Data

- Once the data is loaded, analysis can be run using the following general structure.

```
proc somecommand data=existing;
intermediate statements;
run;
```

Modifying a Data Set

Subsetting Data

- Variables can be created, modified and deleted within data statements.
- If a variable is dropped or kept in the set statement, this controls what is loaded in.
- If a variable is dropped or kept in the data statement, this controls what is saved in the new data set
- data can be subset with a simple if statement
- An example:

```
/*Keeps these variables in the data set being made*/
data (keep=id var1 var2);
/*Drops variable from the data being loaded in*/
set (drop=var3);
if var1=1; /*Keeps observations only when var1=1*/
/*IMPORTANT changes aren't implemented until run*/
run;
```

Modifying a Data Set

Creating or Modifying Variables

- Variables can be created using an equal statement.
- Variables can be modified with if then statements.
- Some examples of the language used are shown on the next slide.

Modifying a Data Set

Creating or Modifying Variables – Examples

```
data something;
set somethingelse;
/*Creates a variable var1, always equal to 1*/
var1=1;
/Creates a variable var1, always equal to var 2*/
var1=var2;
/*If var1=1 then var2 is modified to be equal to 2*/
if var1=1 then var2=2;
/*Modifies var2 if both conditions are met*/
if var1=1 and var3=1 then var2=2;
; /Modified var2 if either condition is met*/
if var1 IN (1:5) or var3=1 then var2=2
/*IMPORTANT changes aren't implemented until run*/
run;
```

Finding and Working with Data

- By default SAS looks for data and stores data within a temporary working directory. This can be found in SAS's explorer window.
- Data sets also exist in a directory internal to sas called sashelp.
- Users can tell SAS to look at external directories with a libname statement.

```
libname somename "/specified/full/location";
```

Working with Outside Data Locations

- Once a libname is specified, data can be loaded in and saved in other locations.
- Data can be saved by adding such a location to the data step.
- Here's a hypothetical example:

```
libname save "/specified/full/location";
libname load "/specified/full/location2";

data save.data;
set load.read;
run;
```

Exporting Data

- Data can be easily exported to other programs and format.
- Two formats frequently used are xlsx (excel) and dta (stata).

```
proc export data=somesasdata filename="\full
location\
dataname.extension" dbms=extension;
/*Example of common extensions
are .xlsx or .dta*/
run;
```

General Suggestions for Getting Started

- The commands shown here can be entered via SAS's program editor.
- The program editor will be the script that allows users to create and reproduce data and results.
- Always use the log file to confirm that code was executed correctly.
- Unless data needs to be shared, there isn't a need to save it to a physical location. The working directory works.
- It is good practice to use a different name for data in the data step versus the set step–this prevents data from being overwritten.