What are the steps involved in the SAS Data Step?

Authored by

stats writer

June 24, 2024

The SAS Data Step is a structured process used to manipulate and analyze data in the SAS programming language. It involves the following steps:
1. Importing data: The first step is to import the data into the SAS environment from various sources such as flat files, databases, or other SAS datasets.
2. Creating a data set: Once the data is imported, a new SAS dataset is created to store and manipulate the data.
3. Reading and processing: The data is then read and processed line by line in a sequential manner.
4. Defining variables: The variables in the dataset are defined, which includes specifying their names, types, and formats.
5. Manipulating data: Various data manipulation techniques can be applied, such as sorting, merging, and subsetting, to modify the dataset as needed.
6. Applying functions: SAS provides a wide range of built-in functions that can be applied to the data to perform calculations, conversions, and other operations.
7. Writing output: The final step involves writing the processed data into a new dataset or exporting it to an external file for further analysis or reporting.

Overall, the SAS Data Step is a systematic approach to transform raw data into a structured and organized format for analysis and decision making.

The Data Step

Recall that SAS programs consist of two main blocks of code: the data step and the procedure (proc) step. The data step is where data is created, imported, modified, merged, or calculated.

The data step follows the following format:

```
DATA Dataset-Name (OPTIONS);
.
.
.
RUN;
```

In the SAS program file above, `DATA` is the keyword that starts the data step, meaning that it tells SAS to create a dataset. *Dataset-Name* is the name of the dataset that you want to create or manipulate. If you want to add any of the dataset options (see below), they would go in the parenthetical after you name the dataset. In between the first and last lines are the statements that create and manipulate the dataset. Note the data step ends with a `RUN` statement and a semicolon.

The SET Statement

When you need to **copy or modify an existing dataset**, use the `SET` statement in the data step. In general the code will follow this form:

```
DATA New-Dataset-Name (OPTIONS);
SET Existing-Dataset-Name (OPTIONS);
.
.
.
RUN;
```

The statements above tell SAS to create a new dataset (*New-Dataset-Name*) that is an exact copy of an existing SAS dataset (*Existing-Dataset-Name*). This allows you to create new variables or recode existing variables without permanently changing the original data. (It is strongly recommended that you do not alter your original data files.)

Copying or cloning an existing dataset

A data step containing only the `SET` statement will create an exact copy of the dataset. For example, the program

```
DATA new_sample;
SET sample;
RUN;
```

creates a new temporary dataset called *new_sample* that is a clone of the already existing dataset called *sample*. You might use code like this when you want to copy a dataset from the temporary library to a permanent library or vice versa.

Modifying an existing dataset

If you do not want to make a copy of a dataset, and instead wish to modify an existing dataset, then you can simply use the same dataset name in the `DATA` statement and in in the `SET` statement.

```
DATA sample;
SET sample;
<other commands here>
RUN;
```

However, you should be aware that this will permanently overwrite the existing dataset. That is, **if you use the same names, then SAS will overwrite the existing dataset with the new dataset you are creating.**

Data Step Options

Data step options generally perform variable-level actions, like renaming or dropping variables from a dataset. Options usually appear in parentheses right after the name(s) of a dataset that is referenced in the DATA statement or in the SET statement.

Drop Variables or Keep Variables

In the data step, DROP and KEEP are used to "throw out" certain variables from your dataset:

`KEEP` tells SAS to keep only the listed variables; all other variables are removed from the dataset.`DROP` tells SAS to remove only the listed variables from the dataset; all other variables are kept.

These two options can accomplish the same thing, but in a given situation one will likely be easier than another. If you only want to remove a couple of variables from a dataset, then using a `DROP` option would be easier than specifying all the variables to stay in a `KEEP` option. Conversely, if you only want to keep a couple of variables in the dataset then using a `KEEP` option would be easier than specifying all the variables to remove in a `DROP` option.

Example: Dropping variables by name

Suppose we want to create a new dataset with a variable BMI computed from the existing variables height and weight. Suppose that we also don't want the height and weight variables to be carried over into the new dataset. The following example creates two new variables (bmi and height2) based on the existing variables height and weight, but removes height and weight from the new dataset *sample_new_vars*.

```
DATA sample_new_vars (DROP = height weight);
SET sample;
bmi = (weight / (height*height) ) * 703;
height2 = height * 0.0254;
RUN;
```

Example: Keeping all variables of a specific type

Let's say that we want to make a copy of our dataset, but only keep the character variables (and

the ID variable, ids). SAS has special syntax for "name lists", which allow you to use a special nickname to refer to "all character variables in this dataset" (`_CHAR_`) or "all numeric variables in this dataset" (`_NUMERIC_`). These name lists can be used in a DROP or KEEP statement, in place of (or in addition to) typical variable names.

```
DATA sample_stringonly;
SET sample(KEEP=ids _CHAR_);
RUN;
```

DATA sample_numericonly;
SET sample(KEEP=ids _NUMERIC_);
RUN;

Rename Variables

The `RENAME` option tells SAS to **change the name of one or more variables**. Its general form is:

```
RENAME = (oldvariable1=newvariable1 oldvariable2=newvariable2 ...)
```

You can rename more than one variable within the parentheses as long as each pair of old and new variable is separated by a space.

Example

To change the names of the variables Gender and DOB to Sex and Date_of_Birth, respectively, we could use the following syntax:

```
DATA sample2 (RENAME=(Gender=Sex DOB=Date_of_Birth));
SET sample;
RUN;
```

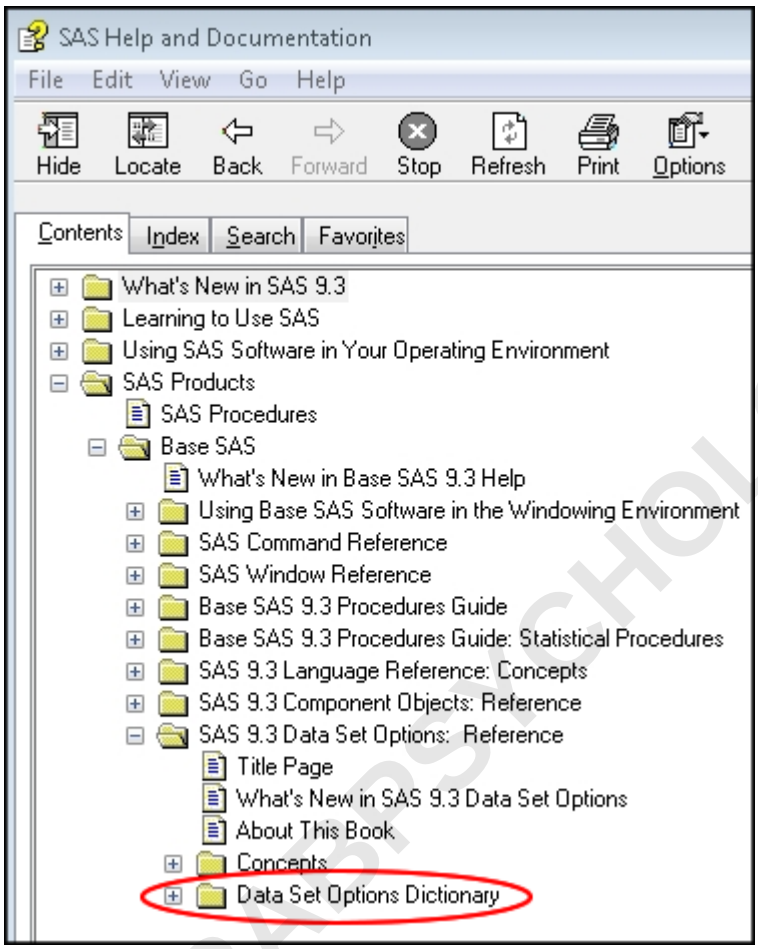
What is the difference between putting the DROP, KEEP, or RENAME options in the DATA statement instead of the SET statement?

You may have noticed in the above examples that some included the DROP or KEEP options in the SET statement, while others put it in the DATA statement (the first line of the data step).

Options in the SET statement take effect immediately. They affect the data that we're using as a "base" or "starting point" within that data step.
If you rename a variable in the SET statement, you'll use the new name in any computations or formulas within that data step.Options in the DATA statement take place last. Think of them as

"finishing touches" that take place after all of your computations within the data step are processed.
If you rename a variable in the DATA statement, it won't change the variable names right away - so it won't affect syntax you put within that data step.

More Data Step Options



Data step options provide SAS with additional instructions on how to read or write the dataset you name. They are generally attached to an output dataset (one that SAS is going to create), but they can also be attached to an input dataset (one that SAS is going to read, like when a SET statement is used).

We have covered some of the most common data step options here. You can discover more options in the SAS Help and Documentation window.

References

SAS 9.2 Language Reference: SAS Variable Lists