What are some resources for learning about Informats and Formats in SAS?

Authored by

stats writer

June 24, 2024

There are a variety of resources available for individuals looking to gain a deeper understanding of Informats and Formats in SAS. These resources include online tutorials, official SAS documentation, and user forums. Additionally, attending SAS training courses or workshops can provide hands-on learning experiences. For those who prefer self-study, there are also books and e-books available that cover the topic comprehensively. It is important to carefully evaluate and choose the most suitable resource based on individual learning preferences and level of proficiency in SAS. With the help of these resources, individuals can enhance their knowledge and skills in Informats and Formats, which are essential for data manipulation and analysis in SAS.

What are Informats and Formats?

In SAS, formats and informats are pre-defined patterns for interpreting or displaying data values. In this tutorial, we'll focus on SAS's built-in formats, which mostly cover numeric, date, and character variables.

Motivation

To understand the need for informats and formats, let's start with a simple example.

Suppose I tell you that a person's birthday is on 12-01-99. How do you know if I mean "December 1, 1999" or "January 12, 1999"?

If you're in the United States, you probably write your dates using MM-DD-YY order, and would therefore interpret the date as "December 1, 1999". But if you're in Europe or Canada, you probably write your dates using DD-MM-YY order, and therefore interpret the date as "12 January 1999". That's quite a difference!

When reading data, SAS also must make the same judgement to interpret the true meaning of the values. Informats are the way we give SAS an explicit rule to follow so that it makes the right judgements.

Formats, on the other hand, allow SAS to change the display value "after the fact" -- i.e., once SAS knows that 12-01-99 should be interpreted as MM-DD-YY, it knows that date could also be displayed as "12/01/1999" or "December 1, 1999" or "1 December 1999".

So **informats and formats are a shared set of common patterns for reading and writing data values** -- the only difference is whether we apply them at the "interpretation" stage (informats) or at the "display" stage (formats).

Built-In Formats & Informats

Formats and informats in SAS use a common set of pattern codes.

There are three main types of built-in informats in SAS: character, numeric, and date. Generically, the informat/format codes follow these patterns:

Type	Format/Informat Name	What it Does
Character	`$w.`	Reads in character data of length w.
Numeric	`w.d`	Reads in numeric data of length w with d decimal points
Date	`MMDDYYw.`	Reads in a date of length *w.* assuming the order MDY

In these codes, *w* denotes the width of the variable, and *d* denotes the number of decimal places.

Notice that the format/informat names contain a period. This helps SAS recognize that it is an informat name rather than a variable name. **SAS will not recognize the informat name without the dot.**

In addition to the above generic informats, there are also *many* specific display formats. Here's a small selection of built-in SAS formats that can change the display of numeric variables:

Scientific notation (`Ew.` format)Comma formatting for large numbers (`COMMAw.d` format)Formatting for dollar amounts (`DOLLARw.d` format)Formatting for percentages (`PERCENTw.d` format)Writing numbers as roman numerals (`ROMANw.` format)Writing numbers as words (`WORDSw.` format)

This may seem like a small matter, but it's incredibly powerful: it allows you to have variables in your dataset that function as numbers (i.e. you can add, subtract, multiply, and divide them), but arbitrarily change the formatted display of those numbers without sacrificing the "numeric-ness" of the variable.

For a full list of built-in formats, see the SAS documentation:

SAS(R) 9.3 Formats and Informats: Formats by CategorySAS(R) 9.3 Language Reference: About SAS Date, Time, and Datetime Values

User-Defined Formats

In addition to the built-in formats, it's possible to define your own formats in SAS. This is particularly useful when you have numerically coded categorical variables; for example, a variable representing a multiple-choice question.

For more on defining your own formats, check out the User-Defined Formats tutorial.

A note about formats for date variables

Regardless of the informat, date values in SAS are stored as the number of days since January 1, 1960. This means that stored date values can be negative (if the date is before January 1, 1960) or positive (if the date is after January 1, 1960). For example, the date June 30, 1999 will be stored in SAS as the number 14425 because June 30, 1999 was 14,425 days after January 1, 1960.

If you supply an informat for a date variable but not a format, SAS will default to displaying the number of days before/since January 1, 1960. This display method makes perfect sense for doing date arithmetic, but is inconvenient for human readers. In order to view date variables "normally", you must apply a date format to the variable.

Applying Informats

Because informats define how variables should be "read" or "interpreted", their use is generally limited to inside the data step. Specifically, they are relevant if you will be reading data from a file using an `INFILE` statement, or manually creating cases using the `DATALINES` command.

In both of these cases, we can include our informats as part of the INPUT statement, which spells out the name and order of the variables in the dataset being created. Its general syntax is:

```
DATA dataset-name;
INPUT variable-name-1 VARIABLE-1-INFORMAT variable-name-2 VARIABLE-2-INFORMAT;
```

In the first line, we declare a new dataset with the name *dataset-name*. On the second line, the INPUT statement tells SAS the names and order of the variables in the dataset. The variable names should be listed in the order they appear, and the variable's format should be given immediately after its name. If you do not include an informat code after a variable name, SAS will assume the "default" informat: a numeric variable of "size" 8.2. (If you omit informats and your data includes a string variable, you will see an error message.)

Alternatively, it's possible to declare your informats separately from the variable names in the INPUT statement:

```
DATA dataset-name;
INFORMAT variable-name-1 VARIABLE-1-INFORMAT;
INFORMAT variable-name-2 VARIABLE-2-INFORMAT;
INPUT variable-name-1 variable-name-2;
```

Here the first word (`INFORMAT`) is the SAS keyword that tells it to assign an informat to a variable. The second word is the name of the variable you want to assign to a format. Finally, type the name

of the format followed by a period and a semicolon. Note that any character variables in the `INPUT` statement should have a `$` after them, even if you've declared them as a character variable in an `INFORMAT` statement.

For string variables

Additionally, for lengthy string variables, you may also need to supply the LENGTH statement in addition to the INFORMAT statement:

```
LENGTH variable-name $w.;
```

This is because string variables in SAS are, by default, only 8 characters long. So if you do not want a string variable to be truncated, you'll need to supply the LENGTH statement in addition to the INFORMAT statement. Also note that any LENGTH statements should be placed after your INFORMAT statements in the data step.

Every variable in any SAS dataset will have an informat, and it's always a good idea to check your SAS data to see what the informat is for each variable. This will help you ensure that the imported data were read in properly. It is also just good practice to look at the variable informats so that you understand the dataset better.

Applying Formats

The general syntax of a format statement is:

```
FORMAT variable-nameFORMAT-NAME.;
```

Here the first word (`FORMAT`) is the SAS keyword that tells it to assign a format to a variable. The second word is the name of the variable you want to assign to a format. Finally, type the name of the format followed by a period and a semicolon.

Unlike informats, the `FORMAT` command can be used in either a data step or a proc step:

Assigning a format to a variable in a data step will permanently store the variable's format assignment as part of the dataset. Any subsequent procedures run on that dataset will have the format applied to that variable. The format will also be applied when viewing the data in a Viewtable window.Assigning a format to a variable in a proc step will temporarily store the variable's format assignment for the proc step it is used in. This is ideal if the format is only useful in a particular proc step's context (e.g., want to use PROC PRINT to see actual data values, but want to create a frequency table using PROC FREQ that uses "pretty" defined formats instead of unlabeled numeric codes).

Not all variables will need to have a format applied; the default numeric variable formatting may be perfectly sufficient. You only need to apply formats to variables where the interpretation and/or readability would be improved. Dates and date-times are the most important case of this.

A note about formats for date variables

Regardless of the informat, date values in SAS are stored as the number of days since January 1, 1960. This means that stored date values can be negative (if the date is before January 1, 1960) or positive (if the date is after January 1, 1960). For example, the date June 30, 1999 will be stored in SAS as the number 14425 because June 30, 1999 was 14,425 days after January 1, 1960. This display method makes perfect sense for doing date arithmetic, but is inconvenient for human readers.

If you do not supply a format for a date variable, SAS will default to displaying the number of days before/since January 1, 1960. (It will not automatically apply the date informat you used!) In order to view date variables "normally", you must apply a date format to the variable.

Every variable will have a format, regardless of whether you assign one or you let SAS assign one automatically. It is to your advantage to assign formats that make sense to you and that can be easily interpreted when you see the values displayed in the dataset or in your output.

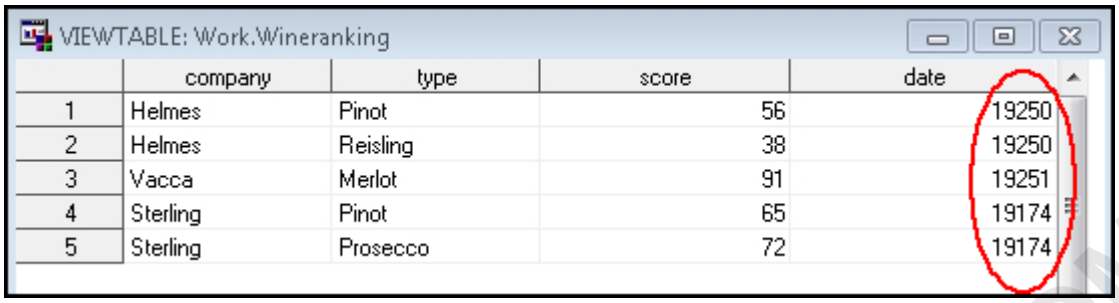
Example: Informats vs. Formats

The following syntax reads in a small dataset using the `INPUT` and `DATALINES` statements. Notice that the `INPUT` statement is where we tell SAS what informats to use. In particular, we specify that variables company and type are character variables (with no specific length requirement); score is a numeric variable of length 3; and date is a date variable in the form MM/DD/YYYY.

```
DATA WineRanking;
INPUT company $ type $ score 3. date MMDDYY10.;
DATALINES;
Helmes Pinot 56 09/14/2012
Helmes Riesling 38 09/14/2012
Vacca Merlot 91 09/15/2012
Sterling Pinot 65 06/30/2012
Sterling Prosecco 72 06/30/2012
;
RUN;
```

Now if you look at the data using the viewtable, you can see that the values for the variable date look like 19250, 19251, and 19174, rather than 9/14/2012, 9/14/2012, etc. This is because we only

told SAS the informat to use. Because we did not explicitly tell SAS what format to use, it used its default format for dates (which is not convenient for human readers).

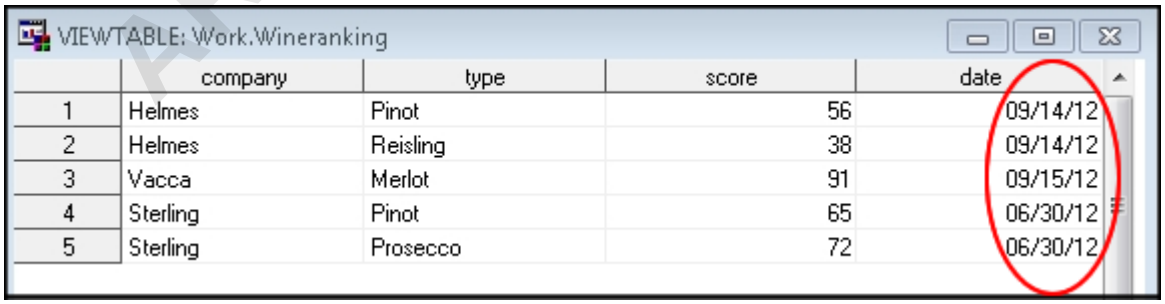

VIEWTABLE: Work.Wineranking

	company	type	score	date
1	Helmes	Pinot	56	19250
2	Helmes	Reisling	38	19250
3	Vacca	Merlot	91	19251
4	Sterling	Pinot	65	19174
5	Sterling	Prosecco	72	19174

We can revise the above block of code so that it reads the dates in one format , but prints the dates in a different format:

```
DATA WineRanking;
INPUT company $ type $ score 3. date MMDDYY10.;
FORMAT date MMDDYY8.;
DATALINES;
Helmes Pinot 56 09/14/2012
Helmes Reisling 38 09/14/2012
Vacca Merlot 91 09/15/2012
Sterling Pinot 65 06/30/2012
Sterling Prosecco 72 06/30/2012
;
RUN;
```

Executing this program in SAS will assign the variable date to the format `MMDDYY8.`, which will display it as MM/DD/YY.


VIEWTABLE: Work.Wineranking

	company	type	score	date
1	Helmes	Pinot	56	09/14/12
2	Helmes	Reisling	38	09/14/12
3	Vacca	Merlot	91	09/15/12
4	Sterling	Pinot	65	06/30/12
5	Sterling	Prosecco	72	06/30/12

Example: Changing a format in the DATA step

Using the sample dataset, let's change the format of the date of birth variable (bday). If you used the Import Wizard to import the data, the SAS default was to assign the variable a `DATE9.` format, which looks like DDMMMYYYY (i.e., a two-digit day of the month, followed by a three-letter abbreviation for the month, followed by a four-digit year):

	ids	bday	Gender	Athlete
1	43783	22MAR1995	0	0
2	20278	01JAN1995	0	0
3	20389	31DEC1994	0	0
4	22820	01DEC1994	1	0
5	24559	10NOV1994	1	1
6	28980	17SEP1994	0	1
7	33312	27JUL1994	0	1
8	40274	03MAY1994	0	1
9	40390	30APR1994	1	1
10	28942	17SEP1993	1	1

?

Let's permanently change the format to `MMDDYY10.` which will make the date values appear as MM/DD/YYYY:

```
DATA students_formatted;
SET sample;
FORMAT bday MMDDYY10.;
RUN;
```

Now when you view the values of variable bday, you can see that they use a two-digit value for the month, followed by a two-digit day, followed by a four-digit year:

	ids	bday	Gender	Athlete
1	43783	03/22/1995	0	0
2	20278	01/01/1995	0	0
3	20389	12/31/1994	0	0
4	22820	12/01/1994	1	0
5	24559	11/10/1994	1	1
6	28980	09/17/1994	0	1
7	33312	07/27/1994	0	1
8	40274	05/03/1994	0	1
9	40390	04/30/1994	1	1
10	28942	09/17/1993	1	1

Example: Changing a format during a PROC step

Using our sample dataset, let's change the format of the date of birth variable (bday) so that it appears a different way when the dataset is printed. To print the data, we will use a proc step called `PROC PRINT`. We will cover this and other proc steps later on, but for now just note that you can put a format statement in a proc step so that the variable has a different format for the output you produce in the proc step. This will not change the format of the variable in the dataset.

```
PROC PRINT DATA = students_formatted;
VAR ids bday;
FORMAT bday WORDDATE20.;
RUN;
```

The SAS System

Obs	ids	bday
1	43783	March 22, 1995
2	20278	January 1, 1995
3	20389	December 31, 1994
4	22820	December 1, 1994
5	24559	November 10, 1994
6	28980	September 17, 1994
7	33312	July 27, 1994
8	40274	May 3, 1994
9	40390	April 30, 1994
10	28942	September 17, 1993

	ids	bday	Gender	Athlete
1	43783	03/22/1995	0	0
2	20278	01/01/1995	0	0
3	20389	12/31/1994	0	0
4	22820	12/01/1994	1	0
5	24559	11/10/1994	1	1
6	28980	09/17/1994	0	1
7	33312	07/27/1994	0	1
8	40274	05/03/1994	0	1
9	40390	04/30/1994	1	1
10	28942	09/17/1993	1	1

Note that although bday prints in the output (upper panel) with the format `WORDDATE20.`, the format of the bday variable itself is unchanged in the dataset (lower panel).

Help for Built-In Informats and Formats

You can find an extensive list of built-in formats and informats, listed alphabetically and by category, in the SAS Help Manual.