

CC009

What to wear on a date?

A short discussion of SAS date functions, formats and informats.

Louise Hadden, Abt Associates Inc., Cambridge, MA

ABSTRACT

Representing, reading in, putting out, and converting dates has long been a thorny issue for programmers using any application. This paper will present both new and old methods of representing dates using SAS[®] formats, SAS[®] macro language, and user-written formats, and will briefly touch on the use of SAS[®] date functions, macro language, and informats. Some of the date formats and informats described are specific to Version 8; thus this paper is targeted at SAS[®] V8 and V9 users.



This example results in a numeric value that represents the current date.

New Version 8 Informats (YYMMNw is likely to become a favorite!):

JDATEYMD
reads Japanese kanji date values in the format yymmdd or yyyymmdd.

JNENGO
reads Japanese Kanji date values in the form yymmdd.

PDJULGw.
reads packed Julian date values that are produced on IBM mainframe systems in the hex representation form of yyydddF. and converts the values to SAS date values.

PDJULw.
reads packed Julian date values that are produced on IBM mainframe systems in the form ccyyddF. and converts the values to SAS date values.

SHRSTAMP.w
reads date and time values of SHR records.

YYMMNw.
reads date values that are in the form yymm or yyyymm and that contain no delimiters, then adds a date value of 01 so that these values become valid SAS date values.

Detailed information on SAS[®] date and time informats can be found in the SAS Version 8 Language Reference Volumes or the online documentation supplied with your SAS installation.

Outs:

Once a SAS[®] date has been created, it can be used as any number could be used in various calculations, or used in data reporting. SAS[®] functions and some macro language elements can be used to manipulate SAS[®] data. The many provided SAS[®] formats or specialized user written formats can be used for data reporting purposes.

Examples of the use of SAS[®] functions in the data step:

```
data mydates;
  set lib.myfile;

  /* find the number of days between two SAS dates */
  ndays=datdif(mysasdate1,mysasdate2,actual);

  /* output a 7 digit Julian date */
  myjuldate=juldate7(mysasdate);

run;
```

INTRODUCTION

SAS[®] provides us with a wide selection of "canned" date formats, informats, and functions to use to represent SAS[®] dates. In addition, elements of SAS[®] macro language such as %SYSFUNC may be used to manipulate dates. Users may also write their own date informats and formats to customize the reporting of SAS[®] dates, as well as using the data step to create SAS[®] dates. The combination of all these tools provides SAS[®] users with the ability to select the perfect outfit to wear on their SAS[®] dates.

INS AND OUTS

Ins:

A SAS[®] date is simply a numeric representation of the distance in days from a particular point in time, namely January 1, 1960. Thus, December 29, 1959 is represented by -3, January 1, 1960 is represented by 0, January 4, 1960 is represented by 3, and so on.

SAS[®] date values are created in one of four ways: using a SAS[®] informat to read in raw data; using SAS[®] functions to convert character or numeric variables, using a SAS[®] date constant, or using SAS[®] macro language elements such as %SYSDATE9 to retrieve the current date.

Informat example:

Input: text field 19600101

Input mysasdate yymmdd8.;

SAS[®] function example:

Input: Chrmonth (2 column character month variable) with a value of '01', chrdy (2 column character day variable) with a value of '01', and chryear (4 column character year variable) with a value of '1950'

```
mysasdate=mdy(input(chrmonth,2.),input(chrdy,2.),input(chryear,4.));
```

SAS[®] date constant example:

```
mysasdate='01jan1960'd
```

All these examples above result in a numeric value of 0, which represents January 1, 1960.

Current date retrieval example:

```
mysasdate=date()
```

New Version 8 Date Functions:**DATDIF**

returns the number of days between two dates.

JULDATE7

returns a seven digit Julian date from a SAS date value.

YRDIF

returns the difference in years between two dates.

SAS® dates are usually formatted. The number 0, which represents January 1, 1960 is useful for the purposes of calculations, etc. but is not terribly informative. Dates can be formatted temporarily (as in a procedure) or permanently (i.e. the format is permanently associated with the variable). As a general rule, whenever I create a SAS® data set containing dates, I format all SAS® dates as MMDDYY10. If a SAS® date has an associated format, any format specified in a procedure overrides the permanent format for the duration of the procedure. SAS® supplies a number of “canned” formats for the presentation of SAS® dates.

Examples of the use of SAS® provided date formats:

/ Produce a frequency table, formatting the dates for the duration of the procedure */*

```
proc freq data=lib.sasdates;
  tables mysasdate1 mysasdate2 mysasdate3 mysasdate4
  / missing;
format mysasdate1 mmddyy10. mysasdate2 yymmdd8.
      mysasdate3 worddate. mysasdate4 yymon7.;
run;
```

/ Permanently associate a SAS® date format with a SAS® date */*
/ Note this could also be done with an ATTRIB statement */*

```
data lib.sasdatef;
  set lib.sasdates;
  format mysasdate1 mmddyy10.;
run;
```

New Version 8 SAS® provided date formats:**DDMMYYxw.**

writes date values in the form ddmmyy or ddmmyyyy and specifies separators or no separators between date values.

MMDDYYxw.

writes date values in the form mmddyy or mmddyyyy and specifies separators or no separators between date values.

PDJULGw.

writes packed Julian date values in the hexadecimal format yyyydddF for IBM.

PDJULIw.

writes packed Julian date values in the hexadecimal format ccydddF for IBM.

YYMMDDxw.

writes date values in the form yymmdd or yyyyymmdd and specifies separators or no separators between date values.

On occasion, a SAS® provided format just doesn't represent the data in exactly the way that is desired. Users can write their own temporary or permanent date formats to accomplish their reporting goals.

Examples of user-written date formats:

/ produce a user written format which groups dates in different ways depending on how far away a date is from a point in time */*

```
proc format;
  value newdatef low-'31dec2001'd = [year4.]
    '01jan2002'd-'31dec2002'd = [yyq5.]
    '01jan2003'd-'31dec2003'd= [monyy7.]
    other= 'Out of range';
run;
```

```
proc freq data=lib.mysasdates;
  tables mysasdate1 / missing;
  format mysasdate1 newdatef.;
run;
```

/ produce a user written format which groups by color (“traffic lighting”) for use in HTML output */*

```
proc format;
  value coldatef low-'31dec2001'd = 'red'
    '01jan2002'd-'31dec2002'd = 'green'
    '01jan2003'd-'31dec2003'd= 'blue'
    other= 'gray';
run;
```

```
ods html body='traffic.htm' path=odsout style=styles.sasweb;
```

```
proc tabulate data=lib.mysasdates;
  class classvar;
  var mysasdate1;
  tables classvar,mysasdate1*{s={background=coldatef.}};
run;
```

```
ods html close;
```

**CONCLUSION**

SAS Version 8 provides a wide variety of options for creating and displaying your SAS dates. Informats, functions, macro language and formats all play a part in helping the SAS programmer decide what to wear on a date.

REFERENCES

SAS Online Documentation (PC SAS V612, PC SAS V8, AIX UNIX SAS V612, AIX UNIX SAS V8, RED HAT LINUX 6.2 SAS V8)

ACKNOWLEDGMENTS

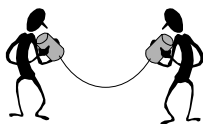
The author wishes to acknowledge Bruce Gilsen of the Federal Reserve Board for his illuminating paper “Date Handling In the SAS® System”.

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are registered trademarks or trademarks of their respective companies.

CONTACT INFORMATION

Your comments and questions are valued and encouraged.
Contact the author at:

Louise Hadden
Abt Associates Inc.
55 Wheeler St.
Cambridge, MA 02138
Work Phone: 617-349-2385
Fax: 617-349-2675



Email: louise_hadden@abtassoc.com

KEYWORDS

BASE SAS[®]; %SYSDATE9; FORMAT; INFORMAT; DATE;
FUNCTION