SAS® Dates: Facts, Formats, and Functions

Debbie Buck, PharmaNet/i3, Houston, Texas, USA

ABSTRACT

Among the first features of SAS that users learn is that SAS dates (and times) have unique characteristics. A SAS date isn't a "standard" numeric or character variable – when displayed it can look like character data but is stored as a number. The date of January 1, 1960, has a value of zero (0) – prior dates are negative numbers, those after have positive values.

Despite SAS dates being part of the initial learning curve, there are a number of factors concerning dates that are frequently less than obvious to users, even experienced users.

In this presentation, we focus on displaying/outputting SAS dates (formats), reading dates (informats), importing/exporting dates, calculating intervals and differences (functions), and extracting or combining portions of dates (more functions).

INTRODUCTION

Regardless of the industry in which you work, you almost certainly use dates in some capacity. This might include simply displaying dates, sorting in ascending or descending order of date (most recent to least recent or vice versa), summarizing by date, calculating differences between dates, manipulating portions of dates, and/or determining intervals or durations. There are a number of techniques in SAS to help you complete these tasks.

We start with exploring ways to read dates into SAS (this includes informats), as well as how to display them in a form that is more readable to those of us who don't readily recognize number of days since January 1, 1960 as a calendar date (formats). Then we will examine how to execute some of the other tasks listed above, including using SAS date functions.

Although SAS time and datetime values also have associated informats, formats, and functions, we will focus only on SAS dates in this paper.

READING IN / CREATING SAS DATES

There are a number of ways to read in or create SAS dates. These include reading in a flat file or instream datalines and using an INPUT statement with an INFORMAT to tell SAS that the field is a SAS date, creating a SAS date constant, and importing dates from another application, such as Excel.

READING IN SAS DATES USING INFORMATS

When creating a SAS data set from raw data (sometimes referred to as a flat file) or from instream datalines we use an INPUT statement to tell SAS the variable names, types, and lengths. We can also use SAS date INFORMATs to read in dates correctly and ensure that dates are stored correctly as SAS dates.

A date INFORMAT tells SAS how to read a value as a SAS date value and store it as an integer which is the number of days since January 1, 1960. An INFORMAT ends with a period, generally specifies the field width, and indicates the form of the value being read.

The following shows an example of SAS code for a DATA step using an INPUT statement with INFORMATs:

```
data one;
 input date1 mmddyy10. date2 date9. date3 monyy. date4 ddmmyy10. date5 mmddyy8.;
 put date1= date2= date3= date4= date5= ;
datalines;
4-7-2011 7APR2011 APR11 07042011 04/07/11
;
run;
```

The SAS log shows the following:

```
NOTE: Invalid data for date2 in line 5 11-19.
NOTE: Invalid data for date3 in line 5 20-24.
NOTE: Invalid data for date4 in line 5 25-34.
date1=18724 date2=. date3=. date4=. date5=-17801
RULE:      ----+----1----+----2----+----3----+----4----+----5----+----6----+----7----
5          4-7-2011 7APR2011 APR11 07042011 04/07/11
date1=18724 date2=. date3=. date4=. date5=-17801 _ERROR_=1 _N_=1
NOTE: The data set WORK.ONE has 1 observations and 5 variables.
```

Looking at the stored date values (and the SAS log NOTE messages), it is obvious something did not work as expected. What went wrong? When examining the actual columns shown by the RULE in the log that the dates reside in, you see a mismatch in the columns in which SAS is expecting to find the dates. By specifying an informat for DATE1 that indicates a width of 10 columns, SAS is looking in column 11 for DATE2. DATE2 actually begins in column 10 because only 1 digit is included for both month and day. How can you fix this? There are two possible solutions, both of which have problems associated with them.

The first solution is to use formatted input with column pointer control, that is, to explicitly specify which column each variable begins. The following program includes the starting columns for each date variable.

```
   data two;
    input @1 date1 mmddyy10. @10 date2 date9. @19 date3 monyy.
          @25 date4 ddmmyy10. @34 date5 mmddyy8.;
    put date1= date2= date3= date4= date5= ;
   datalines;
   4-7-2011 7APR2011 APR11 07042011 04/07/11
   ;
   run;
```

The SAS log shows the following:

```
NOTE: Invalid data for date4 in line 13 25-34.
date1=18724 date2=18724 date3=-17807 date4=. date5=-17801
RULE:      ----+----1----+----2----+----3----+----4----+----5----+----6----+----7----
13         4-7-2011 7APR2011 APR11 07042011 04/07/11
date1=18724 date2=18724 date3=-17807 date4=. date5=-17801 _ERROR_=1 _N_=1
NOTE: The data set WORK.TWO has 1 observations and 5 variables.
```

Well, now you have only one missing date, but the values for date3 and date5 don't look right either. What went wrong with the date4 value? Note that ddmmyy is the correct form, but the raw date value is only eight digits wide rather than ten. (A width of ten allows for separators.) If you change the informat to ddmmyy8. the date4 value is now correct.

Before we discuss the problems with date3 and date5 there are a couple of important issues you should consider:

- This example is based on one observation. What happens if you indicate the starting column for Date2 based on the Date1 value in this example and the next observation has a date of 4-17-2011 for Date1? Now your columns are incorrect again.

- The best way to handle this issue is to use a consistent input form for the date value (e.g. 04/07/2011 – two digits each for month and day) so that the starting column is consistent.

- And… MOST importantly – KNOW YOUR DATA!

When you look at the date3 and date5 values shown in the SAS log above you see that these are negative values. This means that SAS is reading them as dates prior to January 1, 1960. Why would SAS assume this? Notice that the date3 and date5 values only have 2 digits for the year value. The likely culprit is the YEARCUTOFF= system option.

The YEARCUTOFF= option specifies the first year of a 100 year span. SAS uses this to determine the century of a two-digit year. For example, if YEARCUTOFF=1920, then the years 20-99 are read as 1920-1999 and 00-19 are read as 2000-2019.

If the program above is rerun resetting the YEARCUTOFF to 1920 produces the following:

```
options yearcutoff=1920;
data one;
 input date1 mmddyy10. date2 date9. date3 monyy. date4 ddmmyy10. date5 mmddyy8.;
 put date1= date2= date3= date4= date5= ;
datalines;
4-7-2011 7APR2011 APR11 07042011 04/07/11
;
run;
```

The SAS log shows the following:

```
date1=18724 date2=18724 date3=18718 date4=18724 date5=18724
```

Note that SAS uses the first day of the month for the MONYY. format since no day value is included.

Table 1 shows some frequently used date informats. (Results below assume YEARCUTOFF=1920.)

SAS Date Informat	**Calendar Date**	**Result**
DATE.	07APR2011	18724
DATE9.	07APR2011	18724
DDMMYY.	070411	18724
DDMMYY8.	07042011	18724
MMDDYY.	04072011	18724
MMDDYY10.	04/07/2011	18724
MONYY.	APR11	18718
YYMMDD.	110407	18724
YYMMDD8.	20110411	18724
YYQ.	1104	18718
YYQ6.	201104	18718

Table 1. Commonly Used SAS Date Informats

Starting with SAS V9 there is also an informat that can be used for any of the above calendar dates – ANYDTDTEw. (where w is the width of the input field). The default width is 9, so if the value is wider than 9 be sure to specify the width in order to read the date correctly.

CREATING SAS DATE CONSTANTS

Sometimes you may have the need to use a date constant in a SAS data set or in an expression. SAS allows creation of a SAS date using the form ‘ddmmmyy’d or ‘ddmmmyyyy’d. (The second form includes all four year digits.) The quotes before and after the date are necessary, as is the letter d after the closing quote. If the letter d is omitted, SAS reads the date as a character value.

The following shows an example of SAS code for a DATA step including SAS date constants:

```
data two;
 date6=’07APR11’D;
 date7=’07apr2011’d;
 date8=’07Apr2011’d;
 put date6= date7= date8= ;
run;
```

The SAS log shows the following:

```
date6=18724 date7=18724 date8=18724
```

Note that it does not matter whether the month abbreviation is all upper case, all lower case, or mixed. It also does not matter whether the letter d following the closing quote is in upper or lower case.

IMPORTING DATES FROM EXCEL INTO SAS

Often data is imported into SAS from Excel using either the IMPORT procedure or the IMPORT Wizard in the SAS Display Manager. Care should be taken that the imported dates have been read into SAS correctly. Old dates can be a particular problem since Excel has a different zero point than SAS. The zero point for Excel is January 1, 1900, while January 1, 1960 is the zero point for SAS. While SAS can include negative number dates (before the zero point), Excel cannot. Therefore, a date prior to January 1, 1900 will be a missing date value unless this field is stored as a character variable.

Handling a character field imported from Excel that you need to use as a date value in your SAS data set is not normally a difficult problem if the character value 'looks' like a date (i.e., there is an informat in the same form). There is an INPUT function that will help you convert the character to an actual date value.

Let's say you have character field named date in Excel in the form mmddyy10. (04/07/2011). When imported into SAS there is a $10. informat associated with it. Unfortunately, this cannot be used as a SAS date unless a conversion takes place. The assignment statement below within a SAS DATA step would accomplish this.

```
    .
    .
    .
   date9=input(date,mmddyy10.);
   put date9= ;
    .
    .
    .
```

The SAS log shows the following:

```
   date9=18724
```

A few things to remember when using the INPUT function:

- The form of the SAS statement is:
  `new_variable_name=INPUT(old_variable_name, informat.);`

- The informat (as the second expression in the INPUT function) MUST match the form of the old_variable_name value.

DISPLAYING SAS DATES WITH FORMATS

In the previous topic we discussed how to read dates into SAS using INFORMATs. Note, however, that all of the examples displayed the resulting dates as the number of days since January 1, 1960. In general, this is not a very useful form for readability.

How can you see the stored date value as a calendar date? Just as SAS has date INFORMATs for reading in/creating SAS date values, it also has date FORMATs for writing out/displaying SAS date values. Like an INFORMAT, a FORMAT ends with a period, frequently specifies the field width and indicates the form of the value to display.

The following table shows some of the date formats available. All of these are based on a date value of 18724 which is April 7, 2011.

Table 1 shows some frequently used date formats.

SAS Date Format	Displayed Date
DATE.	07APR11
DATE9.	07APR2011
DAY.	7
DDMMYY.	07/04/11
DDMMYY10.	07/04/2011
DDMMYYB10.	07 04 2011
DDMMYYC10.	07:04:2011
DOWNAME	Thursday
MMDDYY.	04/07/11
MMDDYY10.	04/07/2011
MMDDYYD10.	04-07-2011
MMDDYYP10.	04.07.2011
MMYYD.	04-2011
MONNAME.	April
MONTH.	4
MONYY.	APR11
WEEKDATE.	Thursday, April 7, 2011
WEEKDAY.	5
WORDDATE.	April 7, 2011
WORDDATX.	07 April 2011

Table 2. Commonly Used SAS Date Formats

The B – blank, C – colon, D - dash, and P – period additions to the formats (as well as some others) allow you to specify the symbols you would like to include in your output.

The list above provides examples of available date formats. There are also year, quarter, and year/quarter formats.

You can assign a temporary format within a SAS procedure – where it remains in effect only for that procedure, or a permanent format within a SAS DATA step – where it becomes permanently associated with the variable. Regardless of whether it is a temporary or permanent format the form is the same:

SAS statement:

```
format variable_name format.;
```

Examples of this would be to use date6 and date7 from an above DATA step.

```
data two;
 date6=’07APR11’D;
 date7=’07apr2011’d;
 format date6 worddate. date7 ddmmyyb10.
 put date6 = date7 = ;
run;
```

The SAS log shows the following:

```
date6=April 7, 2011 date7=07 04 2011
```

Even if a variable is stored with a permanent format you can still display it in a different way in a PROC step with a temporary format.

SAS DATE FUNCTIONS

SAS has a number of date functions. Some refer to the current day, some extract the day, month or year, some combine a day, month and year, and some calculate intervals or differences.

DATE FUNCTIONS THAT USE A DATE VARIABLE BASED ON TODAY'S DATE

There are several functions that can be used to create a variable containing the current day or to include the current date in a title or footnote.

Table 1 shows some frequently used date functions that refer to the current date.

SAS Date Function	**Result**
TODAY()	SAS date for today
DATE()	SAS date for today (equivalent to Today () function)
SYSDATE	Date at time of SAS initialization in DDMMMYY form
SYSDATE9	Date at time of SAS initialization in DDMMMYYYY form

Table 3. Commonly Used SAS Date Functions for the Current Date

The TODAY() function is particularly useful when calculating differences between the current date and another date. This can be a difference from a prior date (such as in an age calculation) or a future date (for example in calculating an upcoming deadline date).

The statements below use current date functions.

```
data four;
 birth_date='04DEC1994'd;
 date_today=today();               *date_today is current date;
 age=((date()–birth_date)/365.25); *current age, but still needs rounding;
 deadline_date=today()+ 30;        *deadline is 30 days away;

 put date_today= mmddyy10. birth_date= mmddyy10. age= 5.1 deadline_date= mmddyy10.;
run;
```

The SAS log shows the following:

```
date_today=03/26/2012 birth_date=12/04/1994 age=17.3 deadline_date=04/25/2012
```

Also, a TITLE statement in a PROC PRINT step can include the current date by using the following.

```
 proc print data=four;
  title "This program was run on &sysdate9";
run;
```

FUNCTIONS THAT CREATE NEW VARIABLES FROM PORTIONS OF SAS DATES

SAS has a group of functions that can extract portions of dates from SAS dates. These include month, day of the month, day of the week, quarter and year.

Table 4 shows some of these functions. (These are all based on a variable named EXP_DATE of April 7, 2011 with a SAS date value of 18724.)

SAS Date Function	**Example**	**Result**
MONTH(SAS date variable name)	MONTH(EXP_DATE)	4 (4th month of date = April)
DAY(SAS date variable name)	DAY(EXP_DATE)	7 (7th day of month)
WEEKDAY(SAS date variable name)	WEEKDAY(EXP_DATE)	5 (5th day of week = Thursday)
YEAR(SAS date variable name)	YEAR(EXP_DATE)	2011 (year of date)
QTR(SAS data variable name)	QTR(EXP_DATE)	2 (2nd quarter of year)

Table 4. Functions that Extract Portions of a SAS Date

FUNCTIONS THAT COMBINE PORTIONS OF DATES INTO SAS DATES

SAS also has a function that can combine portions of date values into SAS dates. This is the MDY function.

Table 5 shows the primary date function. (There are also a number of datetime and time functions.)

SAS Date Function	Example	Result
MDY(numeric month, numeric day, numeric year)	MDY(4,7,2011)	18724

Table 5. Functions that Combine Portions into a SAS Date

The statement below uses this date function.

```
data four;
 date=mdy(04,07,2011);

 put date= date9.;
run;
```

The SAS log shows the following:

```
date=07APR2011
```

FUNCTIONS THAT RETURN A TIME INTERVAL OR ADVANCE A DATE

Two more SAS date functions to consider are the INTCK and INTNX functions.

The INTCK function returns the number of specified time intervals that lie between two date values, while INTNX advances a date value by a given interval and returns a date value.

Because these are so well covered/explained in previous papers, you should refer to those in the SUGGESTED READINGS below for detailed explanations.

CONCLUSION

We have examined a number of ways of reading/displaying/using/handling SAS dates.

Although this paper has presented some of these, there are many more. There are also many ways of working with SAS datetimes and SAS times. If you are working with SAS dates, datetimes or times you should explore these multiple options.

REFERENCES

SAS Institute Inc. 2009. *Base SAS® 9.2 Help and Documentation*. Cary, NC, USA.

SUGGESTED READING

Bilenas, Jonas V., 2007, “Using SAS Dates and Times - A Tutorial”, Proceedings of the SAS Global 2007 Conference.

Morgan, Derek., 2008, “The Essentials of SAS Dates and Times”, Proceedings of the SAS Global 2008 Conference.

CONTACT INFORMATION

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

Name: Debbie Buck
Enterprise: PharmaNet/i3
City, State ZIP: Houston, TX 77095
Work Phone: 281-256-7437
E-mail: Deborah.Buck@i3global.com