Handling Dates in the Macro Facility

Toby Dunn, AMEDDC&S, Fort Sam Houston

Abstract

Ah, shimmering SAS® date and time values. They brighten our code everywhere, from input, through processing, to output. For the most part these values pose no ambiguity, they are simply formatted integers vividly manipulated with arithmetic, formats, and interval based functions. Alas, when we try to use date and time values in SAS Macro, things start looking dark. The clarity we know and love becomes murky and unclear. Should they be this dim and shadowy? I say no! With the light of knowledge and a few dazzling macro functions this darkness can be illuminated like the shining sun rising after a moonless night. This papers intended audience is beginner and intermediate SAS programmers.

Introduction

Programmers who are new to the macro language find handling Date, DateTime, and Time values can be quit perplexing. The reason for this is that the Macro Language is essentially a text handling language, while the most efficient way to handle Date and Time values is to convert them to SAS Date and Time values (a numeric scaler). To compensate for lack of macro know-how, programmers, tend to use the more familiar Data _Null_ step to handle all the date and time manipulation.

Example 1

```
%Let Date = 01JAN2006 ;

Data _null_ ;
 Call Symput( 'Date' , put( input( "&Date" , Date9. ) , WordDate20. ) ) ;
Run ;

%Put  &Date ;
January 1, 2006
```

However, handling Date and Time values in a Data _Null_ step unnecessarily complicates and clutters the macro code while preventing the programmer from creating a function style macro. This paper demonstrates how to use %Sysevalf and %Sysfunc along with Inputn, Putn, Intck, and Intnx to manipulate Date and Time values in the macro language without the help of a Data _Null_ step.

%Sysevalf

Date and Time literals are the easiest way to get a Date and/or Time value into a SAS Date and/or Time value. So we shall start with these.

Example 2

Data Step Code

```
Data _null_ ;
  Date     = '01JAN2006'd ;
  DateTime = '01JAN2006:12:30:00'dt ;
```

```
 Time     = '12:30:00't ;

Put Date=
   Datetime=
   Time=  ;
Run ;

Date=16802 DateTime=1451737800 Time=45000
```

Macro Equivalent

```
%Put       Date = %Sysevalf( '01JAN2006'd )
       DateTime = %Sysevalf( '01JAN2006:12:30:00'dt )
             Time = %Sysevalf( '12:30:00't ) ;

Date = 16802  DateTime = 1451737800  Time = 45000
```

The actual Date and Time literal code is the same whether it is in the data step or macro. The only difference is that to accomplish the same task in the macro language one has to use the literal values inside of the %Sysevalf function. While the %Eval function is used for most of the numeric operations in the macro language it also performs character comparisons. Thus the Date, DateTime, and Time that follow the quoted Date and Time values for literals are considered as true characters and not part of the Date and Time values.

%Sysfunc, InputN and PutN

While Date and Time literals are nice they don't allow for little flexibility in reading the Date and Time values of different forms into SAS and they don't allow the programmer to format these SAS Date and Time values. To do this in the macro language one needs to move beyond the %Sysevalf function. In the data step these functions are performed by the Input and Put function. It just so happens that the macro language can utilize most of the data step functions with the %Sysfunc macro function. However, the Input and Put functions are not among them, what %Sysfunc can use instead is the InputN and PutN data step functions.

Example 3

Data Step Code

```
Data _null_ ;

 Date              = Input( '01JAN2006' , Date9. ) ;
 DateTime          = Input( '01JAN2006:12:30:00' , DateTime18. ) ;
 Time              = Input( '12:30:00' , Time8. ) ;
 Date2             = Put( Date , Date9. ) ;
 DateTime2         = Put( DateTime , DateTime18. ) ;
 Time2             = Put( Time , Time8. ) ;

Put Date=          Date2=
      Datetime=    DateTime2=
      Time=        Time2= ;
Run ;

Date=16802                 Date2=01JAN2006
DateTime=1451737800        DateTime2=01JAN06:12:30:00
Time=45000                 Time2=12:30:00
```

Macro Equivalent

```
%Let Date         = %Sysfunc( InputN( 01JAN2006 , Date9 ) ) ;
%Let Date2        = %Sysfunc(   PutN(  &Date  , Date9 ) ) ;

%Let DateTime     = %Sysfunc( InputN( 01JAN2006:12:30:00 , DateTime18 ) ) ;
%Let DateTime2  = %Sysfunc(   PutN(  &DateTime , DateTime18 ) ) ;

%Let Time         = %Sysfunc( InputN( 12:30:00 , Time8 ) ) ;
%Let Time2        = %Sysfunc(   PutN(   &Time , Time8 ) ) ;

%Put Date         = &Date           Date2           = &Date2
     DateTime     = &DateTime       DateTime2       = &DateTime2
     Time         = &Time           Time2           = &Time2 ;

Date              = 16802                  Date2              = 01JAN2006
DateTime          = 1451737800             DateTime2          = 01JAN06:12:30:00
Time              = 45000                  Time2              = 12:30:00
```

Please note that in the macro code the first argument to the InputN and PutN functions need not be quoted and doing so could introduce errors. Remember that the macro facility considers everything not preceded by a % or & to be text, which is different from the data step where words have meaning and we quote the words we want to be considered as text. The other thing of significance is that the informat and format used in the InputN and PutN can be any valid SAS informat or format and there is no trailing decimal. Here the macro facility uses decimals to determine the end of a macro variables name. When a format in this context uses a decimal SAS has to make a decision as to whether the decimal means the end of a macro variable name or a format. Since it is used in the macro facility it merely gets consumed by the macro processor.

Before moving on I would also like to note that the %Sysfunc has a second argument which allows the programmer to add a format to be applied to the result of the function referenced by %Sysfunc.

Example 4

```
%Put %Sysfunc( InputN( 01JAN2006 , Date9 ) , MonYY7 ) ;
JAN2006
```

Intck and Intnx

One of the more common problems that programmers encounter is to find how many units (days, months, years) one date is from another or how to increase or decrease a Date and Time value by a certain amount of units. Intck and Intnx are used in the data step. We will also use %Sysfunc as we did in the previous section and then access these functions. This provides as transparent an equivalent as possible.

Example 5

Data Step Code

```
Data _null_ ;
 StartDate        = '01JAN2004'd ;
 EndDate          = '01JAN2006'd ;
 NumOfMonths      = Intck( 'Month' , StartDate , EndDate ) ;
 NewDate          = Intnx( 'Month' , StartDate , NumOfMonths ) ;

 Put StartDate=   EndDate=    NumOfMonths=
      NewDate= Date9. ;
run ;

StartDate= 16071          EndDate= 16802          NumOfMonths= 24          NewDate= 01JAN2006
```

Macro Equivalent

```
%let StartDate            = %Sysevalf( '01JAN2004'd ) ;
%let EndDate              = %Sysevalf( '01JAN2006'd ) ;
%let NumOfMonths          = %Sysfunc( Intck( Month , &StartDate , &EndDate ) ) ;
%let NewDate              = %Sysfunc( Intnx( Month , &StartDate , &NumOfMonths ) ) ;

%put StartDate =          &StartDate
     EndDate  =           &EndDate
     NumOfMonths =        &NumOfMonths
     NewDate =            &NewDate ;

StartDate = 16071     EndDate   = 16802     NumOfMonths = 24     NewDate = 16802
```

Again as in the previous set of examples the constant text values of the Intck and Intnx functions do not need to be in quotes. Otherwise they work exactly the same as they do in the data step. Finally, notice the value of the macro variable NewDate, it is a SAS Date value. While this is good for manipulating the value it is hard for humans to read. The simplest way to change the printed values is to use the second argument to the %Sysfunc as in example 6.

Example 6

```
%let NewDate = %Sysfunc( Intnx( Month , &StartDate , &NumOfMonths ) , WordDate20 ) ;
%put NewDate = &NewDate ;

NewDate = January 1, 2006
```

Conclusion

This paper does cover all of the date and Time functions that can be utilized, but rest assured that one can use these in the macro language by imbedding them in a %Sysfunc. I hope that I have provided, through the examples shown, the knowledge and ability to forego using Data _Null_ steps to simply manipulate Date and Time values in the macro facility.

Thanks to: Paul St. Louis, Paul Choate, Ron Fehd, and Nat Wooding for their kind words, technical review and editing abilities.

Contact Information

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

Toby Dunn
AMEDDC&S Fort Sam Houston, Tx
tobydunn@hotmail.com