Translating Microsoft Character Date-Time Strings to SAS DATETIME Variables

Reading Microsoft Date-Time Stamped Logs in SAS

By Paul D. McDonald, MBA
Overland Park, Kansas

Abstract

This is a quick tip on how to read in Microsoft's standard date-time strings into a SAS data set. The given solution is a simple data step method.

Disclaimer

This system works with the SAS Display Manager system and has been tested only in Microsoft Windows (specifically, Microsoft Windows(XP) in SAS v8.2) but it should work well in any SAS system.

The Basics

Microsoft and other companies tend to use a specific style of date-time stamps. For example, September 9, 2006 at 9:00 AM may be given as:

```
9/9/2006 9:00:00.00 AM
```

There are three problems with this format, at least as far as SAS is concerned.

1. There may not be leading zeroes on month, day, and hour.
2. The AM/PM indicator needs to be handled properly.
3. The date-time value exists as a character string and not as a SAS DATETIME value.

Good news! There is a current coding solution to handle these issues. The sample code below works around the above three issues.

Setup

There are some simple setup steps to ensure that this translation works smoothly:

1. Make sure that your character date variable is at least the length of $25 (Character – twenty-five).
2. If your string uses any other delimiters (such as a hyphen "-") you will need to modify the code to use the delimiters "/" and ":"
3. Make sure all letters are in UPPERCASE.

Leading Zeroes

The first three statements (the "IF SUBSTR" statements) ensure that leading zeroes are in place for the month, day, and hour locations of the textual date-time string. The logic is based on the location of the MDY delimiter (the "/" character) and the HMS delimiter (the ":" indicator).

AM/PM Indicator and Read In SAS DATETIME value

The next two statements (the "IF INDEX" and "ELSE" statements) reads in the date-time variable using the "DATETIME." informat. This gets the value into a SAS DATETIME variable, but unfortunately ignores the AM/PM indicator—it reads all data in as though the time were AM. To work around this, the INTNX function is used to advance the time by precisely 12 hours if the should the PM flag exist in the string and is simply read in as a regular time if not.

Done!

That's it! You have successfully translated a Microsoft traditional Date-Time text stamp into a SAS DATETIME value!

About the Author

Paul D. McDonald, MBA, is a SAS Certified Professional and independent SAS Programming Consultant living in Overland Park, Kansas. He can be reached by phone at (913) 708-3037 or by e-mail at paulmcdonald@sbcglobal.net.

Visit http://www.spikeware.com/ for more fun tips, tricks, and free SAS training!

Translating Microsoft Character Date-Time Strings to SAS DATETIME Variables

Reading Microsoft Date-Time Stamped Logs in SAS

By Paul D. McDonald, MBA
Overland Park, Kansas

The Code

```
data _null_ ;
   length _chardt_ $25 ;
   format datetime1 datetime2 dateampm. ;

   _chardt_ = '9/9/2006 9:00:00.00 AM' ;

   if substr(left(_chardt_), 2, 1) = '/' then _chardt_ = '0'||trim(left(_chardt_)) ;
   if substr(left(_chardt_), 5, 1) = '/' then _chardt_ = trim(left(substr(_chardt_, 1, 3)))||'0'||trim(left(substr(_chardt_, 4))) ;
   if substr(left(_chardt_), 13, 1) = ':' then _chardt_ = trim(left(substr(_chardt_, 1, 11)))||' 0'||trim(left(substr(_chardt_, 12))) ;
   if index(_chardt_, 'P') = 0
     then datetime = input(put(input(substr(left(_chardt_), 1, 10), mmddyy10.), date9.)||' '||trim(left(substr(left(_chardt_), 12))), datetime.) ;
   else datetime = intnx('hour',
                         input(put(input(substr(left(_chardt_), 1, 10), mmddyy10.), date9.)||' '||trim(left(substr(left(_chardt_), 12))), datetime.),
                         12) ;

   datetime1 = datetime ;

   _chardt_ = '9/9/2006 9:00:00.00 PM' ;

   if substr(left(_chardt_), 2, 1) = '/' then _chardt_ = '0'||trim(left(_chardt_)) ;
   if substr(left(_chardt_), 5, 1) = '/' then _chardt_ = trim(left(substr(_chardt_, 1, 3)))||'0'||trim(left(substr(_chardt_, 4))) ;
   if substr(left(_chardt_), 13, 1) = ':' then _chardt_ = trim(left(substr(_chardt_, 1, 11)))||' 0'||trim(left(substr(_chardt_, 12))) ;
   if index(_chardt_, 'P') = 0
      then datetime = input(put(input(substr(left(_chardt_), 1, 10), mmddyy10.), date9.)||' '||trim(left(substr(left(_chardt_), 12))), datetime.) ;
   else datetime = intnx('hour',
                         input(put(input(substr(left(_chardt_), 1, 10), mmddyy10.), date9.)||' '||trim(left(substr(left(_chardt_), 12))), datetime.),
                         12) ;

   datetime2 = datetime ;

   if datetime1 > datetime2 then put 'dt1 > dt2' ;
   else if datetime1 < datetime2 then put 'dt1 < dt2' ;
   else if datetime1 = datetime2 then put 'dt1 = dt2' ;
   put datetime1= datetime2= ;
run ;

dt1 < dt2
datetime1=09SEP06:09:00:00 AM datetime2=09SEP06:09:00:00 PM
NOTE: DATA statement used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds
```