

A Macro to Manage LIBNAME Assignments on Dual Systems

Ian Whitlock, Westat, Inc.

Abstract

One is often faced with the need to set up a SAS® program to run on two or more systems, for example under VMS and Windows or in test mode and production. A tool box macro (i.e. one of a library of macros which have general use across a wide spectrum of people and are automatically available through the autocall facility) solution is suggested.

Assuming the system will be determined by &SYSSCP, the macro invocation for the first case might be

```
%wlib ( name=main ,
        vmsdsn=d60:[proj] ,
        windsn=c:\proj )
```

In the second case a parameter KEY with a non-null value is used to indicate the active "dsn" parameter.

```
%wlib ( key=test ,
        name=main ,
        testdsn=c:\proj\test ,
        proddsn=c:\proj\prod )
```

The PARMBUFF option is used to allow the user to determine the parameter interface within constraints. Both calls illustrated above refer to one macro. In addition to the code providing a useful macro, it gives a nice illustration of using the PARMBUFF option.

Introduction

The macro should create one libref, pointing to the appropriate physical data set whenever it is run under Windows on the PC or VMS on the VAX. In general, the macro should meet the following requirements:

- Physical data set names must stay prominently visible near the top of the code.
- No code change should be required to change operating systems (e.g. VMS and Windows).
- No more than one parameter change should be required to switch between systems that cannot be detected by the SAS supervisor (e.g. Production and Test).
- The user should be able to add a new system (e.g. UNIX).

We can use the automatic variable, SYSSCP, to determine the operating system. Now we could have a parameter WINDSN and a parameter VMDSN to name the physical data sets and the LIBNAME could be

```
libname xxx &&&sysscp.dsn ;
```

To add the MVS operating system all we need is a new parameter, MVSDSN. Since &SYSSCP isn't always suitable as part of a parameter name, we make our own values with the macro SYSSCP.

```
%macro sysscp ( dummy ) ;
    %if &sysscp = PC DOS %then PC ;
    %else &sysscp ;
%mend sysscp ;
```

Now the invocation, %sysscp(), will resolve to VMS on the VAX, WIN under Windows, and PC under PC DOS. The parameter DUMMY is added to allow '()' as an ender of the macro invocation, since empty parameter lists are not allowed in version 6. If we make macro variables, KEY, with the value %sysscp(), and NAME with the value of the libref, then under Windows &key.dsn resolves to WINDSN, hence our LIBNAME statement can be

```
libname &name "&&&key.dsn" ;
```

This version of %SYSSCP will not solve all problems with the value of the variable SYSSCP, but you can modify the macro to cover any new problem.

Now how can a user add new parameters to a macro without modifying the macro code? The answer is the PARMBUFF option on the MACRO statement. With this option the user can write anything between the parentheses. All we have to do is provide the structure for interpreting it. The local automatic variable SYSPBUFF holds the two parentheses and everything in between. Let's make NAME (giving the libref) and KEY permanent parameters by naming them in the MACRO statement. We can then expect parameters &KEY.DSN (giving the physical data set), &KEY.ENG (optionally giving an engine), and &KEY.OPT (optionally giving any options the user wants to add after the physical data set specification). We can ignore whatever else is given in the macro invocation.

The macro parses the parameter list by using the comma to determine a parameter/assignment combination, and breaking it into parts using the equal sign. Then it checks to see that the required parameters, NAME and &KEY.DSN do not have null values. All parameters are declared local and given assignments in order to not assume that KEY is listed first. Triple ampersands are required to resolve variables like &KEY.DSN (e.g. &&&KEY.DSN).

Your tool box should also contain a macro to handle parallel FILENAMES, but that is an easy modification of %LIB, and therefore left as exercise for the reader.

I can be contacted at

Westat Inc.
1650 Research Boulevard
Rockville, MD. 20850-3129

or by E-mail at whitloil@westat.com.

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries.
® indicates USA registration.

The Macro

```

%macro lib ( key = ,          /* null defaults to %sysscp */
              /* VMS, WIN, MVS, etc. */
              name = ,        /* Libref for external file. */
              /*** remaining options appear in comment below.
              ___dsn = ,      Data set name, quotes optional
              ___eng = ,      BASE, XPORT, etc.
              ___opt =        Options after the dsn.
              Where ___ stands for value given by KEY, e.g. ___ could be
              replaced by VMS, WIN, MVS, etc. as many as required. ***/
) / parmbuff ;

%local args i parm w name key ;

/* remove surrounding parentheses from &syspbuff */
%let args = %qsubstr ( &syspbuff , 2 , %length ( &syspbuff ) - 2 ) ;
%let i = 1 ;          /* prepare to loop through the parameter list */
%let w = %qscan ( &args , &i , %str(,) ) ;
%do %while ( %length(&w) > 0 ) ;          /* parse parameter list */
    %let parm = %scan ( &w , 1 , = ) ;    /* get parameter name */
    %local &parm ;                          /* make it local */
    %let &parm = %quote(%scan ( &w , 2 , = )) ; /* assign value */
    %let i = %eval ( &i + 1 ) ;          /* prepare for next iteration */
    %let w = %qscan ( &args , &i , %str(,) ) ;
%end ;

%if %length(&key) = 0 %then %let key = %sysscp() ;
%else %let key = %unquote(&key) ;          /* allow macro assignment */
%local &key.dsn &key.eng &key.opt ;        /* ensure existence of */
                                          /* expected parameters */
%if %length(&name) = 0 or %length ( &key.dsn ) = 0 %then
%do ;          /* cannot make libname statement */
    %put ERROR: LIB - Parameter NAME or %upcase(&key)DSN is missing ;
    %goto mexit ;
%end ;

libname &name &&&key.eng
    %if %index(%str('%"') , %qsubstr(&&&key.dsn,1,1)) = 0 %then
        "&&&key.dsn" ;
    %else
        &&&key.dsn ;
    &&&key.opt
;

%mexit:
%mend lib ;

```