

Building and Using Corporate-Wide Libraries for Macros and Formats

Dennis McBride, Mutual of Omaha

What Are They?

Let's begin by defining a "corporate-wide library." In the literal sense, these would be libraries which are actually available to the entire "corporation". In reality, the techniques we are talking about can be used for any SAS® formats or macros which will be used by two or more people. In other words, any library which is not private is essentially corporate.

Carrying that scenario one step further, let's make these libraries AUTOMATICALLY accessible to everyone so that these macros and/or formats can be used by anyone without any effort on their part to make them available. Our libraries will appear as though they are SAS system libraries. This accessibility is, in the end, what differentiates them from other MVS data sets or CMS mini disks which can be read by multiple users.

Another desirable trait of a "corporate" library is that it is created and maintained by a single individual or team who accepts the responsibility of testing and insuring the currency of items in that library. While macros or formats could certainly be submitted by others, they would become the property and responsibility of this team after testing and application of any coding standards.

Why Bother?

Reasons for maintaining corporate libraries differ somewhat depending on whether we are looking at format libraries or macro libraries.

For macros, the justification is convenience and productivity. Often, a macro which is of benefit to the individual who creates it could be of equal benefit to many others in the corporation. Placing these macros in a corporate library makes them available to all and, at the same time, restricts any conversion efforts to the single team who is responsible for maintaining the library.

Format libraries reap those same benefits while adding even more important justifications of standardization and accuracy. Our company is divided into "budget centers" which are designated by a numeric code. These codes get increasingly complex as we work our way down through operations, divisions, areas, teams, and projects. Furthermore, we have what seems to be a continuous reorganization program in progress, which means that budget center codes are constantly changing. It only makes sense that

the department responsible for establishing the codes be the ones to see that a "corporate" version of that table is available to all and that anyone who uses budget center codes should use the corporate table rather than a makeshift table of their own creation which MAY have been accurate the week it was created.

Any organization of any size has a similar situation. It may be product codes, class codes, sales office identifiers - anything. A single library gives us accuracy and uniformity and reduces the maintenance effort in direct relation to the number of users and the frequency of changes.

So Why Haven't We Been Doing This All Along?

No doubt, many of us have, especially on MVS systems. On MVS, format libraries could be made available by concatenating a corporate loadlib containing Version 5 formats under a STEPLIB DD statement and I would assume that we aren't the only company to have done this. Some complications would arise due to the fact that not everyone used the same MVS PROC (or CLIST) to initiate SAS, and the same format libraries were not included in all versions of the MVS PROCs. The same generalities applied to macro libraries on MVS.

Under CMS, the same ability to concatenate libraries does not exist, however. Until SAS Version 6, libraries were defined using CMS FILEDEF commands and they do not allow concatenation. You could store PROC FORMAT code on a public disk and individuals could use %INCLUDE to include them in the job stream, but this wasn't very efficient and produced a number of hassles.

So What Changed?

The SASAUTOS= system option may or may not have been available on your system in Version 5. This option can be used to specify one or multiple autocall macro libraries and, in conjunction with the MAUTOSOURCE system option, make them available to all users on your system.

New to Release 6.07 is the FMTSEARCH= system option. Like the SASAUTOS= system option, FMTSEARCH= allows you to specify one or more format catalogs to be included in the default search path. You can even alter the default priorities if you wish.

Conclusion

After years of doing without the corporate library concept, I 'discovered' the possibilities out of a business need. Now that the concept is a reality, I can't understand how we managed without them. We have found them to be invaluable. Try 'em - you'll like 'em.

The business need which prompted me follows. Maybe it's something you can use.

In the insurance business, we have a constant need to know the date for the "previous business day." Given any single date, that might not present such a problem. But given an unknown date in multiple years full of weekends and holidays with an occasional emergency closing thrown in, it can become quite a challenge to calculate.

So, with a little prodding from some customers, I came up with a macro to do the calculation. Weekends can be determined using built in SAS functions. But I needed a table (format?) of holidays. And, to be useful and maintainable, I needed both the macro and the format to exist in single libraries accessible to all. The discovery of the FMTSEARCH= system option in the Changes and Enhancements Technical Report for Release 6.07 allowed everything to fall into place. Macro %PBDAY has become one of the most used "utilities" in our company.

I have included the macro code and an example of our holiday table at the end of this paper. Please feel to use it if it solves a problem for you.

Comments or Questions may be addressed to:

Dennis McBride
Mutual of Omaha
Mutual of Omaha Plaza
Omaha, NE 68175

Phone: (402) 342-7600
Fax: (402) 351-8025

SAS is a registered trademark of SAS Institute, Inc. in the USA and other countries.

* indicates USA registration.

Instructions for Creating Macro Libraries in CMS

1. Code and thoroughly test the Macro. Do not let code go past column 71.
2. Place Macro into a CMS file with a filetype of COPY.

NOTE 1: A filetype of MACRO is theoretically acceptable. CMS wants to place sequence numbers in columns 72 through 80 on a MACRO file, though. While there are ways to deal with that, I have found it simpler to work with the COPY filetype and just avoid the issue altogether.

NOTE 2: This will be "public" code. As such, it should be thoroughly documented. When was the last time you needed something fixed and could actually locate the person who originally wrote it?

3. Using the CMS MACLIB command, place the macro in a CMS MACLIB on a publicly accessible disk.

For a new library.....

```
MACLIB GEN libname filename1 (filename2.....)
```

where *libname* is the name of the MACLIB you wish to create and *filename* is the name of the COPY file containing the macro code.

For an existing library.....

```
MACLIB ADD libname filename1 (filename2.....)
```

NOTE: For a complete discussion of MACLIBs and the MACLIB command, see the CMS Command Reference for your version of CMS.

4. Make the macro (MACLIB) library available to the public by setting the SAS system option for SASAUTOS.

Example:

```
SASAUTOS=('CORPORAT MACLIB *','SASAUTOS MACLIB *')
```

For additional information and hints on writing SAS macros for general use, see "Writing Macros For General Use" in SAS Guide to Macro Processing, Version 6 Second Edition.

For additional information on the SASAUTOS= system option, see "SAS System Options" in SAS Language Reference, Version 6 First Edition.

Instructions for Creating Format Libraries in CMS

1. Code and thoroughly test the format.
2. Using PROC FORMAT with the LIBRARY= option, create the permanent format and place it in the desired SAS format catalog on a publicly accessible mini disk.

Example:

```
LIBNAME CORPORAT 'L';  
  
PROC FORMAT LIBRARY=CORPORAT;  
VALUE .....
```

3. Use the FMTSEARCH= SAS system option (new in Release 6.07) to place this format catalog in the default search path. This will allow users to access the format just as they would a SAS provided format.

```
FMTSEARCH= (CORPORAT);
```

For additional information on creating permanent SAS format catalogs, see "The FORMAT Procedure" in SAS Procedures Guide, Version 6 Third Edition.

For additional information on the FMTSEARCH= system option, see "SAS System Options" in SAS Technical Report P-222, Changes and Enhancements to Base SAS Software, Release 6.07.

Instructions for Creating Macro Libraries in MVS

1. Create a partitioned data set to function as an autocall library.
2. Code and thoroughly test the macro before placing the code in a member in the partitioned data set. Note that the name of the PDS member is the name of the macro.
3. Use the SASAUTOS= system option to make the PDS available to all users.

For additional information and hints on writing SAS macros for general use, see "Writing Macros For General Use" in SAS Guide to Macro Processing, Version 6 Second Edition.

For additional information on the SASAUTOS= system option, see "SAS System Options" in SAS Language Reference, Version 6 First Edition.

For additional information pertaining specifically to MVS, see "Macro Facility" in SAS Companion for the MVS Environment, Version 6 First Edition.

Instructions for Creating Format Libraries in MVS

1. Using PROC FORMAT with the LIBRARY= option, create the permanent format and place it in the desired SAS format catalog in a publicly accessible MVS data set.

Example:

```
LIBNAME CORPORAT 'your.format.dataset';
```

```
PROC FORMAT LIBRARY=CORPORAT;  
VALUE .....
```

3. Use the FMTSEARCH= SAS system option (new in Release 6.07) to place this format catalog in the default search path. This will allow users to access the format just as they would a SAS provided format.

```
FMTSEARCH= (CORPORAT);
```

For additional information on creating permanent SAS format catalogs, see "The FORMAT Procedure" in SAS Procedures Guide, Version 6 Third Edition.

For additional information on the FMTSEARCH= system option, see "SAS System Options" in SAS Technical Report P-222, Changes and Enhancements to Base SAS Software, Release 6.07.

Previous Business Day Macro

```
%MACRO PBDAY(DTE,VARNAME);
/*****
/** NAME:          PBDAY
/** AUTHOR:       DENNIS MCBRIDE
/** DATE:        NOVEMBER, 1992
/**
/** USAGE:       %PBDAY (DATE,VARIABLENAME)
/**
/** DESCRIPTION:
/** RETURNS THE DATE OF THE LAST BUSINESS DAY PRIOR TO
/** THE DATE PASSED AS A PARAMETER.
/**
/** RETURNED DATE IS CONTAINED IN A VARIABLE SPECIFIED
/** BY THE VARIABLENAME PARAMETER PASSED AT MACRO
/** INVOCATION AND IS IN THE FORM OF A SAS SYSTEM DATE.
/**
/** NOTES:
/** PASSED DATE SHOULD BE CONTAINED IN A VARIABLE IN
/** THE FORM OF A SAS DATE.
/**
*****/

%IF "&VARNAME" = "" %THEN %LET VARNAME = PDAY;
PDAY = &DTE - 1;
DONE=0;
DO UNTIL(DONE);
  IF (WEEKDAY(PDAY)=1 | WEEKDAY(PDAY)=7)
    OR PUT(PDAY,HOLI.)-='NO HOLIDAY'
    THEN PDAY=PDAY-1;
  ELSE DONE=1;
END;
&VARNAME=PDAY;
%IF "&VARNAME" = "PDAY" %THEN DROP PDAY;
DROP DONE;
%MEND PBDAY;
```

Holiday Dates File

```
12/25/91 CHRISTMAS
01/01/92 NEW YEAR'S DAY
05/25/92 MEMORIAL DAY
.
.
.
.
ETC.
```

Code To Create Holiday Formats

```

/*****
**  THIS PROGRAM CREATES THE FORMATS FOR
**  COMPANY HOLIDAYS.  THESE FORMATS ARE
**  CREATED IN A CORPORATE FORMAT LIBRARY
**  NAMED 'OFORMATS CORPORAT' WHICH RESIDES
**  ON THE 'L' DISK.
**
**  INPUT FILE:
**  HOLIDAY DATES L - THIS FILE MUST BE
**                     MAINTAINED ON AN
**                     ANNUAL BASIS BY TSS
**
**  OUTPUT:
**  'HOLI' FORMAT - MUST BE USED WITH
**                 A SAS DATE FIELD.
**
**                 THE FORMAT RETURNS
**                 A VALUE OF 'HOLIDAY'
**                 IF THE DATE IS A
**                 COMPANY HOLIDAY.
**
*****/
LIBNAME CORPORAT 'L';
```

```

/*****
**  READ IN DATA SET TO BE USED FOR FORMATS
*****/
DATA HOLIDAYS;
  INFILE 'HOLIDAY DATES L' END=LASTREC;
  FORMAT START $5.;
  INPUT @1 HDATE    MMDDYY8.
        @10 LABEL   $20.
        ;
  START=HDATE;
  FMTNAME='HOLI';
  OUTPUT;
  IF LASTREC THEN DO;
    START='OTHER';
    LABEL='NO HOLIDAY';
    FMTNAME='HOLI';
    OUTPUT;
  END;
RUN;
/*****
**  CREATE FORMATS
*****/
PROC FORMAT LIB=TESTFMT CNTLIN=HOLIDAYS;
RUN;
```