

%SYSFUNC: How and where.

Foster Kerrison

BASUG, December 4th, 2003

ABSTRACT

%SYSFUNC is a very powerful and flexible component of the SAS ® system. This BASUG presentation is designed to teach you how to understand the basic elements of %SYSFUNC, and how to apply it to your daily tasks. %SYSFUNC is a macro function, but you don't need to be a macro guru to use it. %SYSFUNC allows you to undertake a wide range of tasks related to your data by using consistent syntax. Understanding how to use %SYSFUNC is the key to success, and this presentation will teach you how.

INTRODUCTION

After reading this paper, and the programs attached, you should be able to incorporate %SYSFUNC into your daily tasks. Over time, and with experience, you will learn to add new and more complex features of %SYSFUNC, until you become adept at all the options in %SYSFUNC.

%SYSFUNC

%SYSFUNC was added in one of the version 6 enhancements, and is now available in all current versions. %SYSFUNC runs on all systems, so you can use it anywhere.

%SYSFUNC is a SAS ® utility which allows a programmer access to almost all SAS functions, and 61 SCL functions.

MACROS

The use of %SYSFUNC does not require that you know how to write macros, although a few macro basics will help you get the best out of %SYSFUNC. The primary macro statement to understand is the %LET statement. The syntax for this is %LET *name* = *value* ;

This is a macro language statement that assigns a value to a macro variable name. If you use the %LET statement in open code it will automatically create a *global macro variable* (meaning you can use it anywhere in that SAS ® session). If you use it inside a macro, then it creates a *local macro variable* (meaning you can use it within the macro). To illustrate:

```
%let country = Ireland ;
```

To use the assigned value we call it by putting an ampersand (&) in front of it, like this: &country.

I could then add it to a title statement:

```
TITLE "It is green in &country ";
```

And the title would print like this:

It is green in Ireland.

So the value of the macrovar is assigned by the %LET statement, and called by &country.

Being able to pass values around like this is very convenient because we will see that many of the %SYSFUNC operations can be assigned in open code and we can pass the values around.

USING %SYSFUNC

%SYSFUNC can be used with: SAS DATASETS, EXTERNAL FILES, SAS CATALOGS, and FILE DIRECTORIES.

%SYSFUNC can be used in: open code (outside a dataset or macro), in a dataset, or inside a macro. It is, therefore, very adaptable and of course very powerful.

%SYSFUNC with Functions

First, let's have a look at how %SYSFUNC works with regular SAS® functions.

%SYSFUNC can access all SAS® functions, other than, DIF, INPUT, PUT, DIM, LAG, RESOLVE, HBOUND, LBOUND and SYMGET.

%SYSFUNC should not be confused with the SAS® %functions (e.g. %upcase(var)).

BASIC %SYSFUNC SYNTAX:

```
%SYSFUNC(FUNCTION(ARGUMENT<,>,<FORMAT>);
```

To illustrate its use with functions, I used a dataset to assign a value:

```
DATA NOM ;
  NAMED = ' PADDIE ' ;
  NEWNAME = TRANSLATE(NAMED,"X"," ") ;
  CALL SYMPUT('NOMAMA',NEWNAME) ;
RUN ;
%PUT &NOMAMA ;
XXXPADDIEXXX
```

Then I used %SYSFUNC to drop the X:

```
%let MAMA =
%SYSFUNC(COMPRESS(&NOMAMA,X));
%PUT &MAMA ;
PADDIE
```

And then I added the X back again:

```
%let MAMA = %sysfunc(TRANSLATE(&NOMAMA,X, ));
%PUT &MAMA ;
XXXPADDIEXXX
```

So I used a SAS function with %SYSFUNC to flip over and back between:

```
PADDIE
XXXPADDIEXXX
```

Of course, we don't really need the dataset, because we can assign the value using a %let statement:

```
%let BENIGN = ' PADDIE ' ;
%put BENIGN = &BENIGN ;
BENIGN = ' PADDIE '
```

Then use %sysfunc to change it:

```
%let NASTY = %SYSFUNC(translate(&BENIGN,X, ));
%put NASTY = &NASTY;
NASTY = 'XXXPADDIEXXX'
```

Now change it back again:

```
%let BENIGN2 = %SYSFUNC(translate(&NASTY, ,X)) ;
%put BENIGN2 = &BENIGN2;
BENIGN2 = ' PADDIE '
```

and finally, clean up the value (remove the quotation marks):

```
%let BENIGN2 = %SYSFUNC(compress(&benign2,'')) ;
%put BENIGN2 = &BENIGN2;
BENIGN2 = PADDIE
```

You are probably wondering if they can be nested. The answer is: "of course!" The key is to nest each %SYSFUNC. Like this:

```
%let ONELINE=
%SYSFUNC(compress(%SYSFUNC(translate((%SYSFUNC(
C(translate(&BENIGN,"X"," "))), " ","X")), "()" ) ) ;
```

I just used two functions here for illustration: translate, and compress, but you have a vast array of functions available to you.

One other common use of %SYSFUNC is in reformatting dates. First assign the date:

```
%let txtdate = 120403 ; * MMDDYY6 ;
%let trydate = 12/04/03 ; * MMDDYY8 ;
```

Using a macro approach:

```
%macro change_date (redate,sasfmt) ;
  %let txtdate = %sysfunc(inputn(&&redate,&sasfmt)) ;
%mend change_date ;
```

```
%change_date (&txtdate,mmddyy6) ;
```

or %SYSFUNC in open code:

```
%let trydate = %sysfunc(inputn(&trydate,mmddyy8)) ;
```

gives us the SAS® date:

```
MACRO CONVERTED THE TXTDATE TO: 16043 ;
OPEN CODE CONVERTED THE DATE TO: 16043 ;
```

You can also use your own user created formats or SAS® formats with %SYSFUNC.

%SYSFUNC with datasets

Another powerful, and useful, operation is to use %SYSFUNC with a dataset. %SYSFUNC offers an extensive range of dataset operations.

For illustration, so that they are easier to follow, I broke them into two elements:

1. Dataset information
2. Dataset manipulation

This may be a little simplistic, because in many cases you will want to mix different operations, so the differentiation does not always hold.

The two major functions in 1. to look at dataset information are ATTRN and ATTRC. The difference is that one returns a numeric code (ATTRN) and the other returns a character (ATTRC). I have identified codes for the major functions in appendix 1. Check more of them yourself, by trying the code.

To illustrate how this works, I want to check to see if a datasets exists. Then I want to know various bits of information about it such as how many observations and variables it has, how long the record is, when it was modified, if it is

password protected, whether it is sorted and so on. Could I get this from PROC CONTENTS?

To answer the first question: does the dataset exist?

```
%LET OK = %SYSFUNC(EXIST(<dataset name>));
```

If the dataset exists then the value for &OK is 1, but if it does not exist then &OK will be 0.

It is always a good idea to use IF statements to check if certain steps took place. Because you use %IF statements you need to enclose the code in a macro.

Assuming that the dataset exists, I want to have a look at it.

The first key, and critical, step is to open the dataset. An equally critical step is to close it when you are finished, otherwise it will not be available. So, putting these all together:

```
%IF %SYSFUNC(EXIST(&DS)) = 1 %THEN %DO ;

    /* ASSIGN THE DATASET NAME */
    %LET DSNAME = &DS ;
    /* OPEN THE DATASET */
    %LET DSID = %SYSFUNC(OPEN(&DSNAME)) ;
    /* IF THE DATASET OPENED ... */
    %IF &DSID %THEN %DO ;
    /* DOES IT HAVE OBS? */
    %LET ANY = %SYSFUNC(ATTRN(&DSID,ANY)) ;
    /* HOW MANY OBS DOES IT HAVE? */
    %LET OBS = %SYSFUNC(ATTRN(&DSID,NOBS)) ;
    /* HOW MANY VARS DOES IT HAVE? */
    %LET VAR = %SYSFUNC(ATTRN(&DSID,NVARS)) ;
    /* WHAT IS THE LRECL? */
    %LET LREC = %SYSFUNC(ATTRN(&DSID,LRECL)) ;
    /* WHEN WAS IT CREATED? */
    %LET DTEC = %SYSFUNC(ATTRN(&DSID,CRDTE)) ;
    /* WHEN WAS IT MODIFIED? */
    %LET DTE = %SYSFUNC(ATTRN(&DSID,MODTE)) ;
    /* IS IT PASSWORD PROTECTED? */
    %LET PWD =
    %SYSFUNC(ATTRN(&DSID,ALTERPW)) ;
    /* WHAT ENGINE WAS USED TO CREATE IT? NOTE:
    ATTRC, NOT ATTRN */
    %LET ENG = %SYSFUNC(ATTRC(&DSID,ENGINE)) ;
    /* HAS IT BEEN SORTED? NOTE: ATTRC, NOT
    ATTRN */
    %LET SRT = %SYSFUNC(ATTRC(&DSID,SORTEDBY)) ;
    /* WHAT TYPE OF DATASET IS IT? NOTE: ATTRC,
    NOT ATTRN */
    %LET MDE = %SYSFUNC(ATTRC(&DSID,MODE)) ;
    /* CLOSE THE DATASET - A CRITICAL STEP! */
    %LET RC = %SYSFUNC(CLOSE(&DSID)) ;
    /* END THE DO STATEMENT */
%END ;
```

I have a copy of a program to check on datasets. This will run in a windows environment.

Feel free to modify this to meet your own needs.

%SYSFUNC with files and directories

%SYSFUNC is also a very powerful tool with files and directories, as we will see.

The basic syntax structure remains the same as we saw in datasets, but the commands, as you would expect, are different.

The process is also similar to what we saw in looking at datasets, open the file, investigate the contents, and close the file.

SAS® uses the *file data buffer (FDB)* to read the file contents.

The first thing to do is to check if the file exists. So, create a fileref:

```
FILENAME TEST2 "C:\TESTS\TESTER.TXT" ; RUN ;
```

Then check for the file:

```
%LET RC = %SYSFUNC(FEXIST(TESTER)) ;
%IF &RC = 1 %THEN %PUT THE FILE EXISTS ;
```

Then open it, and work with it:

```
%LET FILRF = TEST2 ;
%LET RC = %SYSFUNC(FILENAME(FILRF,
C:\TESTS\TESTER.TXT)) ;
%IF &RC = 0 %THEN %PUT FILE REF SUCCESSFUL ;
%LET FID = %SYSFUNC(FOPEN(&FILRF,I)) ;
%IF &FID = 1 %THEN %PUT FILE OPEN WAS A
SUCCESS ;
%IF &FID = 0 %THEN %PUT %SYSFUNC(SYMSG()) ;
%IF &FID > 0 %THEN %DO ;
    %LET RC = %SYSFUNC(FREAD(&FID)) ;
    %LET RC = %SYSFUNC(FGET(&FID,MYSTRING,10)) ;
    %PUT ;
    %PUT THE DATA STRING IS: &MYSTRING ;
    %PUT ;
    %LET INFOS =
    %SYSFUNC(FOPTNUM(&FID)) ;
    %PUT THE NUMBER OF FILE OPTIONS ARE:
    &INFOS AS FOLLOWS ;
    %DO I = 1 %TO &INFOS ;
    %LET NAME =
    %SYSFUNC(FOPTNAME(&FID,&I)) ;
    %LET VALUE =
    %SYSFUNC(FINFO(&FID,&NAME)) ;
    %PUT %UPCASE(&NAME) HAS A VALUE
    OF &VALUE ;
    %END ;
    %PUT ;
    %LET RC = %SYSFUNC(FCLOSE(&FID)) ;
    %IF &RC = 0 %THEN
    %PUT THE FILE WAS CLOSED SUCCESSFULLY ;
    %END ;
```

In this case, we opened the file (FOPEN), then looked at the data in the first record (FREAD)

which I forced to read the first 10 characters - the default is to read to the first token - and then identify the number of options (FOPTNUM), then I created a loop to run through the names of each of the options (FOPTNAME), outputting those values to the log, and then closed (FCLOSE) the file.

There are many file manipulation options available, including appending data, identifying various records within the file, moving the cursor around in the file to read data, and so on.

We can also work with the files themselves. Let's say that we want to delete a file, we call the fileref and in one line of code delete the file:

```
%let rd = %sysfunc(fdelete(test));
```

In a directory, we can do similar operations. First, let's see how many files are in a directory:

```
%LET DIR = C:\TEST ;  
  
%LET DREF = TSTDIR ;  
%LET RC = %SYSFUNC(FILENAME(DREF,&DIR)) ;  
%LET DID = %SYSFUNC(DOPEN(&DREF)) ;  
%LET MEMBERS = %SYSFUNC(DNUM(&DID)) ;  
%LET RC = %SYSFUNC(DCLOSE(&DID)) ;  
%PUT FILES = &MEMBERS ;
```

Here, we opened the directory (DOPEN), and checked to see how many members there were (DNUM), closed the directory (DCLOSE), and then printed the number of members in the log.

That's a useful thing to know, but you might also want to look at the names of the members. If you do, then try this macro:

```
%MACRO MEMBER ;  
%LET DIRECTORY = C:\TEST ;  
%LET DREF = TSTDIR ;  
%LET RC=%SYSFUNC(FILENAME(DREF,&DIRECT));  
%LET DID = %SYSFUNC(DOPEN(&DREF)) ;  
%LET MEMBERS = %SYSFUNC(DNUM(&DID)) ;  
%DO I = 1 %TO &MEMBERS ;  
%LET NAMED&I = %SYSFUNC(DREAD(&DID,&I)) ;  
%END ;  
%LET RC = %SYSFUNC(DCLOSE(&DID)) ;  
%PUT ;  
%PUT THE NUMBER OF MEMBERS IN &DIRECT IS:  
= &MEMBERS ;  
%DO T = 1 %TO &MEMBERS ;  
%PUT FILENAME&T = &&NAMED&T ;  
%END ;  
%MEND MEMBER ;  
  
%MEMBER ;
```

This will produce a listing in your log of the number of files in the directory, and their names.

If you have a library on MVS, you can use this code to read the number of members, and their names; just change the directory name to suit your library name. If you have hundreds of files in your directory you might not want to print them in your log, but that's up to you. Remember, the basic steps are, open the directory, check the contents, and close the directory.

So can we mix these operations to pick individual files, read them and manipulate them? Yes you can, and how you do it is a matter of how adept and creative you can become using %SYSFUNC.

RETURN CODES

I mentioned earlier that I was including return codes for the major functions, and I recommend that you become familiar with these return codes. Knowing what code is returned - they are not always 1 for success and 0 for unsuccessful - will make your life much easier.

I created a table of codes with the associated %SYSFUNC code, and you might find it useful as a start point to become familiar with return codes.

DO IT YOURSELF

%SYSFUNC has so many options to work with that you really have to build your own approach and technique. Build your own, "homemade," solutions to undertake complex tasks to reconfigure data, or files.

For more information about %SYSFUNC within the SAS® system look in online docs, and drill down to find %SYSFUNC, or check the references below for examples.

Hopefully this paper will give you the basics that allow you to use %SYFUNC. You are the best person to determine your own needs, and you are the best person to build on the basics in this paper, so I encourage you to test and use %SYSFUNC approaches to make your life easier, and add to your SAS® skillsets.

CONTACT INFORMATION

Foster Kerrison

Phone: 603-520-7520

Email: kerrison@metrocast.net

Web: www.fosterkerrison.com

REFERENCES:

SAS Institute Inc., SAS ® Macro Language : Reference, first edition, SAS Institute, Cary NC, USA, 1997. 304pp.

Burlew, Michele M. SAS® MACRO Programming Made Easy, Cary, NC SAS Institute Inc., 2001. 208pp.

Ysidra, Chris, “%SYSFUNC – The brave new macro world”, SUGI23, 1998.

SAS Institute Inc., TS-DOC:TS-544-Base Language Changes for release 6.09E Maintenance, SAS Institute, Cary NC, USA, 2000.

```

/*****
/*  TITLE: USING FUNCTIONS WITH %SYSFUNC
/*  AUTHOR: FOSTER KERRISON
/*  PHONE: 603-520-7520
/*  EMAIL: KERRISON@METROCAST.NET
/*  PURPOSE: BASUG PRESENTATION, DECEMBER 4TH 2003.
/*  PROGRAM: THIS PROGRAM ILUSTRATES HOW %SYSFUNC CAN BE USED WITH
/*            DIFFERENT FUNCTIONS, HOW %SYSFUNC CAN BE NESTED IN
/*            ONE LINE OF CODE, AND HOW A DATE VALUE CAN BE CHANGED.
/*  WARRANTY: NO WARRANTY IS IMPLIED OR EXPRESSED
*****/

/* USING A DATASTEP */
/* A DATASTEP ASSIGNS THE NAME, TRANSLATES THE SPACES, AND SYMPUTS THE RESULT */
DATA NOM ;
    NAMED = '    PADDIE    ' ;
    NEWNAME = TRANSLATE(NAMED,"X"," ") ;
    CALL SYMPUT('NOMAMA',NEWNAME) ;
RUN ;

/* THE RESULTING MACRO VAR NAME IS NOMAMA */
%PUT &NOMAMA ;

/* TAKE ALL THE "Xs" OUT USING THE COMPRESS FUNCTION */
%let MAMA = %sysfunc(COMPRESS(&NOMAMA,X)) ;
%PUT *&MAMA* ;

/* TRANSLATE THE SPACES BACK INTO Xs */
%let MAMA = %sysfunc(TRANSLATE(&NOMAMA,X, )) ;
%PUT &MAMA ;

/* THAT IS THE END OF A MIXED DATASTEP AND OPEN CODE EXAMPLE */

*****;

/* NOW LETS LOOK AT AN OPEN CODE EXAMPLE TO ACHIEVE THE SAME RESULT*/
/* STEP 1: ASSIGN THE VALUE */
%let BENIGN = '    PADDIE    ' ;
%put BENIGN = &BENIGN ;

/* NOW USE TRANSLATE TO CHANGE THE SPACE TO X */
%let NASTY = %sysfunc(translate(&BENIGN,"X"," ") ) ;
%put NASTY = &NASTY;

/* NOW TRANSLATE THE X TO SPACES */
%let BENIGN2 = %sysfunc(translate(&NASTY," ","X")) ;
%put BENIGN2 = &BENIGN2;

/* AND FINALLY GET RID OF THE QUOTES */
%let BENIGN2 = %sysfunc(compress(&benign2,'"')) ;
%put BENIGN2 = &BENIGN2;

/* COMBINE ALL OF THE ABOVE INTO ONE LINE OF CODE*/
/* THE KEY IS THAT EACH FUNCTION, MUST HAVE A %SYSFUNC NESTED INSIDE THE OTHER */
/* YOU CANNOT HAVE ONE %SYSFUNC STATEMENT TO COVER ALL THE FUNCTIONS */
/* THE SYNTAX IS CRITICAL */

```

```

%let ONELINE= %sysfunc(compress(%sysfunc(translate((%sysfunc(translate(&BENIGN,"X","
")))," ","X")), "'() ") ) ;
%put ONELINE = &ONELINE ;

*****
;

/* USE %SYSFUNC TO CONVERT DATES INTO SAS DATES */
/* ASSIGN THE DATES */

%let txtdate = 120403 ; * MMDDYY6 ;
%let trydate = 12/04/03 ; * MMDDYY8 ;

/* A MACRO APPROACH ... */

%macro change_date (redate,sasfmt) ;
    %let txtdate = %sysfunc(inputn(&redate,&sasfmt)) ;
%mend change_date ;

%change_date (&txtdate,mmddyy6) ;

/* USING SYSFUNC IN OPEN CODE ... */

%let trydate = %sysfunc(inputn(&trydate,mmddyy8)) ;

%put THE MACRO CONVERTED THE TXTDATE TO: &txtdate ;
%put THE OPEN CODE CONVERTED THE DATE TO: &trydate ;

```

```

/*****
/*  TITLE:  TESTING DATASETS WITH %SYSFUNC                                */
/*  AUTHOR: FOSTER KERRISON                                              */
/*  PHONE:  603-520-7520                                                */
/*  EMAIL:  KERRISON@METROCAST.NET                                       */
/*  WEB:    FOSTERKERRISON.COM                                           */
/*  PURPOSE: BASUG PRESENTATION, DECEMBER 4TH 2003.                     */
/*  PROGRAM: THIS PROGRAM ILUSTRATES HOW %SYSFUNC CAN BE USED TO EXPLORE */
/*           SAS DATASETS. IT ALSO SHOWS HOW TO USE THE WINDOW IN SAS    */
/*  WARRANTY: NO WARRANTY IS IMPLIED OR EXPRESSED                       */
*****/

```

```

/* MANUALLY ASSIGN THE DATASET NAME */

```

```

%GLOBAL DS OK ;

```

```

/* OUTPUT A WINDOW AND SEEK INPUT */

```

```

%window AAGGH IROW = 0 ROWS = 150 ICOLUMN = 0 COLUMNS = 125 COLOR = GREY

```

```

#4 @30 'THIS IS YOUR DATASET TEST' attr = underline COLOR = GREEN

```

```

#12 @12 'ENTER THE DATASET NAME'

```

```

#14 @15 "THE LIBRARY NAME IS? "

```

```

      attr=UNDERLINE c = BLUE

```

```

      LIBR 20 attr=UNDERLINE c = BLACK REQUIRED = YES

```

```

#19 @15 "THE DATASET NAME IS? "

```

```

      attr=UNDERLINE c = BLUE

```

```

      DSNM 20 attr=UNDERLINE c = BLACK REQUIRED = YES

```

```

#25 @25 "PRESS ENTER TO CONTINUE"

```

```

;

```

```

%display AAGGH ;

```

```

/* CREATE THE MACROVAR AND POPULATE THE VALUE */

```

```

%LET NOM1 = &LIBR ;

```

```

%LET NOM2 = &DSNM ;

```

```

%LET ds = &NOM1..&NOM2 ;

```

```

%LET OK = %SYSFUNC(EXIST(&DS)) ;

```

```

%MACRO TESTER ;

```

```

%DO R = 1 %TO 3 ;

```

```

%IF %SYSFUNC(EXIST(&DS)) NE 1 %THEN %DO ;

```

```

/* OUTPUT A WINDOW WITH THE BAD NEWS AND SEEK NEW INPUT */

```

```

%window AAGGH IROW = 0 ROWS = 150 ICOLUMN = 0 COLUMNS = 125 COLOR = GREY

```

```

#4 @30 'THIS IS YOUR DATASET REPORT' attr = underline COLOR = GREEN

```

```

#10 @12 "OOPS! "

```

```

#12 @12 "THE DATASET: &DS DOES NOT EXIST ON YOUR SYSTEM"

```

```

#14 @12 'PLEASE CHECK THAT YOU TYPED THE CORRECT DATASET NAME'

```

```

#16 @12 'RE-ENTER THE DATASET NAME TO TRY AGAIN '

```

```

#18 @12 'YOU GET THREE TRIES BEFORE THE PROGRAM SHUTS DOWN '

```

```

#20 @15 "THE LIBRARY NAME IS? "

```

```

      attr=UNDERLINE c = BLUE

```

```

      LIBR 20 attr=UNDERLINE c = white REQUIRED = YES

```

```

#25 @15 "THE DATASET NAME IS? "

```

```

        attr=UNDERLINE c = BLUE
        DSNM 20 attr=UNDERLINE c = white REQUIRED = YES
    ;
%display AAGGH ;

    /* RE-POPULATE THE VALUE */
%LET NOM1 = &LIBR ;
%LET NOM2 = &DSNM ;
%LET ds = &NOM1..&NOM2 ;

%END ;

%END ;

%IF %SYSFUNC(EXIST(&DS)) NE 1 %THEN %DO ;

/* OUTPUT A WINDOW WITH THE BAD NEWS THAT IT IS CLOSING DOWN */
%window AAGGH IROW = 0 ROWS = 150 ICOLUMN = 0 COLUMNS = 125 COLOR = GREY

    #4 @30 'THIS IS YOUR DATASET REPORT' attr = underline COLOR = GREEN

    #14 @12 "                                UNFORTUNATELY YOUR DATASET DOES NOT EXIST                                "
    #16 @12 "THIS PROGRAM IS NOW GOING TO CLOSE TO ALLOW YOU TO CHECK THE SYSTEM"
    ;
%display AAGGH ;

%END ;

/* IF THE DATASET EXISTS THIS WILL LOOK AT IT AND PRODUCE A REPORT */
%IF %SYSFUNC(EXIST(&DS)) = 1 %THEN %DO ;

    /* ASSIGN THE DATASET NAME */
    %LET DSNAME = &DS ;
/* OPEN THE DATASET */
    %LET DSID = %SYSFUNC(OPEN(&DSNAME)) ;
/* IF THE DATASET OPENED ...*/
    %IF &DSID %THEN %DO ;
/* DOES IT HAVE OBS? */
    %LET ANY = %SYSFUNC(ATTRN(&DSID,ANY)) ;
/* HOW MANY OBS DOES IT HAVE? */
    %LET OBS = %SYSFUNC(ATTRN(&DSID,NOBS)) ;
/* HOW MANY VARS DOES IT HAVE? */
    %LET VAR = %SYSFUNC(ATTRN(&DSID,NVARS)) ;
/* WHAT IS THE LRECL? */
    %LET LREC= %SYSFUNC(ATTRN(&DSID,LRECL)) ;
/* WHEN WAS IT CREATED? */
    %LET DTEC = %SYSFUNC(ATTRN(&DSID,CRDTE)) ;
/* WHAT WAS THE DATE AND TIME THAT IT WAS MODIFIED? */
    %LET DTE = %SYSFUNC(ATTRN(&DSID,MODTE)) ;
/* IS IT PASSWORD PROTECTED? */
    %LET PWD = %SYSFUNC(ATTRN(&DSID,ALTERPW)) ;
/* WHAT ENGINE WAS USED TO CREATE IT? NOTE: ATTRC, NOT ATTRN*/
    %LET ENG = %SYSFUNC(ATTRC(&DSID,ENGINE)) ;
/* HAS IT BEEN SORTED? NOTE: ATTRC, NOT ATTRN*/
    %LET SRT = %SYSFUNC(ATTRC(&DSID,SORTEDBY)) ;
/* WHAT TYPE OF DATASET IS IT? NOTE: ATTRC, NOT ATTRN*/
    %LET MDE = %SYSFUNC(ATTRC(&DSID,MODE)) ;

```

```

        /* CLOSE THE DATASET - A CRITICAL STEP!*/
%LET RC = %SYSFUNC(CLOSE(&DSID)) ;
%END ;

/* CONVERT THE SASDATE TO SOMETHING WE CAN READ */
/* THIS IS FOR THE CREATION DATE */
%LET DTXC = %SYSFUNC(PUTN(&DTEC,DATETIME25.));
/* THIS IS FOR THE MODIFIED DATE */
%LET DTX = %SYSFUNC(PUTN(&DTE,DATETIME25.));

/* LOOK AT THE RC FOR ANY */
%IF &ANY = 1 %THEN %DO ;
    %LET ANX = THE DATASET HAS OBSERVATIONS ;
%END ;

%ELSE %DO ;
    %LET ANX = THE DATASET HAS NO OBSERVATIONS ;
%END ;

/* TEST THE PASSWORD PROTECTION */
%IF &PWD = 1 %THEN %DO ;
    %LET PWX = A PASSWORD IS NEEDED TO ALTER THIS DATASET ;
%END ;

%ELSE %DO ;
    %LET PWX = NO PASSWORD IS REQUIRED TO ALTER THIS DATASET ;
%END ;

/* TEST TO SEE IF THE DATASET HAS BEEN MODIFIED */
/* IF IT WAS NOT MODIFIED */
%IF &DTEC = &DTE %THEN %DO ;
    %LET MOD = THE DATASET HAS NOT BEEN MODIFIED SINCE IT WAS CREATED ;
%END ;

/* IF IT WAS MODIFIED */
%ELSE %IF &DTEC NE &DTE %THEN %DO ;
    %LET MOD = THE DATASET WAS LAST MODIFIED ON: &DTX ;
%END ;

/* TEST TO SEE IF IT WAS SORTED */
%IF %LENGTH(&SRT) %THEN %DO;
    %LET SXT = THE DATASET IS SORTED BY &SRT ;
%END ;

%ELSE %DO ;
    %LET SXT = THE DATASET HAS NOT BEEN SORTED ;
%END ;

/*THIS IS THE REPORT ON THE GOOD DATASET */
%window BLOW IROW = 0 ROWS = 150 ICOLUMN = 0 COLUMNS = 125 COLOR = GREY

#4 @20 'THIS IS YOUR DATASET REPORT' attr = underline COLOR = GREEN
#6 @12 "GOOD NEWS! "
#8 @12 "THE DATASET: &DSNAME EXISTS ON YOUR SYSTEM"
#10 @12 'FOR YOUR INFORMATION:'

```

```

#12 @12 "THE SAS ENGINE USED TO ACCESS THE DATASET WAS: &ENG "
#14 @12 "&ANX"
#16 @16 "THERE ARE: &OBS OBSERVATIONS "
#18 @12 "THERE ARE: &VAR VARIABLES      "
#20 @12 "THE LRECL IS: &LREC           "
#22 @12 "THE DATASET WAS CREATED ON: &DTXC "
#24 @12 "&PWX "
#26 @12 "&MOD "
#28 @12 "&SXT "
#30 @12 "THE DATASET MODE WHEN OPENED WAS &MDE"
;
%display BLOW ;

/* THIS IS A SECOND REPORT ON THE GOOD DATASET */
%window BLOW IROW = 0 ROWS = 150 ICOLUMN = 0 COLUMNS = 125 COLOR = GREY

#4 @20 'THIS IS YOUR DATASET REPORT' attr = underline COLOR = GREEN

#6 @12 "WHAT DID SAS REALLY SEE?"
#8 @12 "OBSERVATIONS    = &OBS "
#10 @12 "VARIABLES      = &VAR "
#12 @12 "RECORD LENGTH  = &LREC"
#14 @12 "CREATION DATE   = &DTEC"
#16 @12 "MODIFIED DATE   = &DTE "
#18 @12 "PASSWORD       = &PWD "
#20 @12 "SORTED         = &SRT "
#22 @12 "DATASET CLOSED = &RC  "
#24 @12 "MODE          = &MDE "
#26 @12 "ANY OBS?      = &ANY "
#28 @12 "ONE KEY TO SUCCESS IS KNOWING WHAT THE CODES ARE "
#30 @12 "MORE ABOUT RETURN CODES AND THEIR MEANINGS LATER. "
#32 @12 "A COPY OF CODES AND SYNTAX COMES WITH YOUR HANDOUT "
;

%display BLOW ;

%END ;

%MEND TESTER ;

%TESTER ;

```

```

/*****
/*  TITLE:  TESTING DIRECTORIES AND FILES WITH %SYSFUNC          */
/*  AUTHOR: FOSTER KERRISON                                     */
/*  PHONE:  603-520-7520                                         */
/*  EMAIL:  KERRISON@METROCAST.NET                               */
/*  WEB:    FOSTERKERRISON.COM                                   */
/*  PURPOSE: BASUG PRESENTATION, DECEMBER 4TH 2003.             */
/*  PROGRAM: THIS PROGRAM ILUSTRATES HOW %SYSFUNC CAN BE USED TO EXPLORE */
/*            FILES AND DIRECTORIES.                             */
/*  WARRANTY: NO WARRANTY IS IMPLIED OR EXPRESSED              */
*****/

/* ASSIGN VALUES FOR THE DIRECTORY AND FILE */
%LET DIREC = MY DOCUMENTS ;
%LET FNAME = MANNER ;

/* CREATE THE FILEREF */
FILENAME TESTER "C:\&DIREC\FNAME..TXT" ;
RUN ;

/* SET UP SOME SAMPLE DATA IN THE FILE*/
DATA _NULL_ ;
  INFILE CARDS ;
  INPUT @ 01 NAMES $4. ;
  FILE TESTER ;
  PUT @01 NAMES ;

CARDS ;
JACK
JILL
MARY
JOE
;

/* TEST TO SEE IF IT EXISTS */
%LET RC = %SYSFUNC(FEXIST(TESTER)) ;
%PUT RC = &RC ;

/* NOW CHECK THE DIRECTORY TO SEE HOW MANY FILES ARE THERE */
/* NOTE: THE DIRECTORY IS C:\TESTS */
%LET DREF = TSTDIR ;
%LET RC = %SYSFUNC(FILENAME(DREF,C:\&DIREC)) ;
%LET DID = %SYSFUNC(DOPEN(&DREF)) ;
%LET MEMCOUNT = %SYSFUNC(DNUM(&DID)) ;
%LET RC = %SYSFUNC(DCLOSE(&DID)) ;
%PUT FILES = &MEMCOUNT ;

/* NOW DELETE THE FILE WE JUST CREATED */
%let rd = %sysfunc(fdelete(tester)) ;
%put RD = &RD ;

/* TEST TO SEE IF IT'S STILL THERE - IT SHOULD BE GONE */
%LET RC = %SYSFUNC(FEXIST(TESTER)) ;
%PUT RC = &RC ;

/* NOW CHECK THE DIRECTORY AGAIN TO SEE HOW MANY FILES WE HAVE */
%LET DREF = MYDIR ;

```

```

%LET RC = %SYSFUNC(FILENAME(DREF,C:\&DIREC)) ;
%LET DID = %SYSFUNC(DOPEN(&DREF)) ;
%LET MEMCOUNT = %SYSFUNC(DNUM(&DID)) ;
%LET RC = %SYSFUNC(DCLOSE(&DID)) ;
%PUT FILES = &MEMCOUNT ;

*****;

/* THIS PROGRAM SHOWS YOU THE RETURN CODES FOR FILES TESTED IN OPEN CODE */

/* STEP1: ASSIGN THE FILENAME */
/* THIS FILE EXISTS */
filename testfile "C:\TESTS\SYSFUNC.TXT" ;
run ;
/* THIS FILE DOES NOT EXIST */
filename testx "C:\tests\mess.txt" ;
run ;

/* NOW, IN OPEN CODE TEST FOR THE FILES */
%let rc = %sysfunc(fexist(testfile));
%let rx = %sysfunc(fexist(testx));

/* PUT THE RESULT TO YOUR LOG */
%put THE RETURN CODE FOR THE GOOD FILE WAS &rc ;
%put THE RETURN CODE FOR THE MISSING FILE WAS &rx ;

*****;

/* THIS SECTION SHOWS HOW TO MANIPULATE FILE CONTENTS */

/* CREATE A FILE IN THE TEST DIRECTORY */

/* ASSIGN VALUES FOR THE DIRECTORY AND FILE */
%LET DIREC = TESTS ;
%LET FNAME = TESTER.TXT ;

/* CREATE THE FILEREF */
FILENAME TEST2 "C:\&DIREC\&FNAME" LRECL = 10 ;
RUN ;

/* SET UP SOME SAMPLE DATA IN THE FILE*/
DATA _NULL_ ;
  INFILE CARDS ;
  INPUT @ 01 NAMES $4. ;
  FILE TEST2 ;
  PUT @01 NAMES
      @06 "BLACK" ;

CARDS ;
JACK
JILL
MARY
JOE
;

```

%MACRO HOLDALL ;

/* TEST TO SEE IF THE FILE EXISTS */

%LET RC = %SYSFUNC(FEXIST(TEST2)) ;
%IF &RC = 1 %THEN %PUT THE FILE EXISTS ;

%LET FILRF = TEST2 ;
%LET RC = %SYSFUNC(FILENAME(FILRF,C:\&DIREC\FNAME)) ;
%IF &RC = 0 %THEN %PUT FILE REF SUCCESSFUL ;
%LET FID = %SYSFUNC(FOPEN(&FILRF,I)) ;
%IF &FID = 1 %THEN %PUT FILE OPEN WAS A SUCCESS ;
%IF &FID = 0 %THEN %PUT %SYSFUNC(SYMSG()) ;
%IF &FID > 0 %THEN %DO ;
%LET RC = %SYSFUNC(FREAD(&FID)) ;
%LET RC = %SYSFUNC(FGET(&FID,MYSTRING,10)) ;
%PUT ;
%PUT THE DATA STRING IS: &MYSTRING ;
%PUT ;
%LET INFOS = %SYSFUNC(FOPTNUM(&FID)) ;
%PUT THE NUMBER OF FILE OPTIONS ARE: &INFOS AS FOLLOWS: ;
%DO I = 1 %TO &INFOS ;
%LET NAME = %SYSFUNC(FOPTNAME(&FID,&I)) ;
%LET VALUE = %SYSFUNC(FINFO(&FID,&NAME)) ;
%PUT %UPCASE(&NAME) HAS A VALUE OF &VALUE ;
%END ;
%PUT ;
%LET RC = %SYSFUNC(FCLOSE(&FID)) ;
%IF &RC = 0 %THEN %PUT THE FILE WAS CLOSED SUCCESSFULLY ;
%END ;

%LET RC = %SYSFUNC(FILENAME(FILRF)) ;
%IF &RC = 0 %THEN %PUT THE FILEREF WAS DEALLOCATED ;

%MEND HOLDALL ;

%HOLDALL ;

*****;

/* FINALLY, SOME CODE TO CHECK ON DIRECTORY MEMBERS */

%MACRO MEMBER ;

%LET DIRECT = C:\TESTS ;

%LET DREF = TSTDIR ;
%LET RC = %SYSFUNC(FILENAME(DREF,&DIRECT)) ;
%LET DID = %SYSFUNC(DOPEN(&DREF)) ;
%LET MEMBERS = %SYSFUNC(DNUM(&DID)) ;
%DO I = 1 %TO &MEMBERS ;
%LET NAMED&I = %SYSFUNC(DREAD(&DID,&I)) ;
%END ;
%LET RC = %SYSFUNC(DCLOSE(&DID)) ;
%PUT ;
%PUT THERE ARE A TOTAL OF &MEMBERS FILES IN &DIRECT.. ;
%PUT ;

```
%PUT THIS IS A LIST OF THE FILES: ;
%PUT ;
%DO T = 1 %TO &MEMBERS ;
    %PUT %SYSFUNC(UPCASE(&&NAMED&T)) ;
%END ;
```

```
%MEND MEMBER ;
```

```
%MEMBER ;
```

%SYSFUNC	EFFECT	sysfunc RETURN CODE	MEANING
ATTRN(<dataset_id>,ALTERPW)	ALTER PASSWORD	0	DS IS NOT ALTER PROTECTED
ATTRN(<dataset_id>,ANOBS)	KNOWS # OF OBS	1	SAS KNOWS THE # OF OBS
ATTRN(<dataset_id>,ANY)	# OF OBS KNOWN	1	THE DS HAS OBS
ATTRN(<dataset_id>,ARAND)	RANDOM ACCESS	1	SUPPORTS RANDOM ACCESS
ATTRN(<dataset_id>,ARWU)	MANIPULATE FILES	1	YOU CAN MANIPULATE FILES
ATTRN(<dataset_id>,CRDTE)	CREATION DATE	1385658274.9	DS CREATION DATE
ATTRN(<dataset_id>,INDEX)	INDEX	1	THE DATASET DOES NOT SUPPORT INDEX
ATTRN(<dataset_id>,ISINDEX)	IS INDEXED	0	THE DATASET IS NOT INDEXED
ATTRN(<dataset_id>,ISSUBSET)	IS A SUBSET	0	THE DS IS NOT A SUBSET?
ATTRN(<dataset_id>,LRECL)	LOGICAL RECORD	5	THE LOGICAL RECORD LENGTH
ATTRN(<dataset_id>,MODTE)	DATE OF CHANGE	1385658274.9	MODIFIED DATE
ATTRN(<dataset_id>,NDEL)	DELETED OBS	0	THE NUMBER OF DELETED OBS
ATTRN(<dataset_id>,NLOBS)	OBS NOT AVAILABLE	1	# OF LOGICAL OBS (NOT DELETED)
ATTRN(<dataset_id>,NOBS)	OBS	1	NUMBER OF PHYSICAL OBS IN DS
ATTRN(<dataset_id>,NVAR)	VAR	1	# OF VAR ON THE DS
ATTRN(<dataset_id>,PW)	PASSWORD	0	THE DS IS NOT PROTECTED
ATTRN(<dataset_id>,READPW)	READ PASSWORD	0	THE DS NOT READ PROTECTED
ATTRN(<dataset_id>,TAPE)	ON A TAPE	0	THE DS IS NOT ON TAPE
ATTRN(<dataset_id>,WHSTMT)	WHERE	0	NO WHERE STATEMENT WAS USED TO MAKE THE DS
ATTRC(<dataset_id>,LIB)	LIBRARY	WORK	NAME OF DATASET LIBRARY
ATTRC(<dataset_id>,ENGINE)	ENGINE	V8	SAS ENGINE USED TO MAKE DATASET
ATTRC(<dataset_id>,MTYPE)	MEMBER DETAIL	DATA	MEMBER TYPE
ATTRC(<dataset_id>,MEM)	DATA SET NAME	ONE	DATASET NAME
ATTRC(<dataset_id>,MODE)	MODE	I	THIS IS THE MODE IN WHICH THE DATASET WAS
OPENED: I MEANS INPUT MODE			
CLOSE(DATASETID)	CLOSE THE DATASET	0	DATASET SUCCESSFULLY CLOSED
CUROBS(DATASETID)	THE CURRENT OBS	1	THE OBS #
DCLOSE(DIRECTORYID)	CLOSE THE DIRECTORY	0	THE DIRECTORY WAS CLOSED
DINFO(DIRECTORYID,INFO_ITEM)	DIRECTORY INFORMATION	DEPENDS	VARIES BY OPTION
DNUM(DIRECTORYID)	DIRECTORY IDENTIFIER	DEPENDS	VARIES BY OPTION
DOPEN(FILEREF)	ASSIGNED FILEREF	>0	DIRECTORY WAS OPENED