



NEW SOFTWARE CLASSES & SERVICES FOR 2000

Systems Seminar Consultants is excited about changes coming in 2000. Several new training courses have been developed, existing ones are being updated, and new consulting services are debuting.

NEW COURSES

SAS® Efficiencies

Designed for users with some SAS software experience, this course will enrich SAS skills and help students improve the efficiency of SAS programs. Working with large data files, students will learn how SAS code can be "tweaked" to reduce both file storage size and processing time.

JCL/ISPF®

Designed for users with little or no experience on an MVS mainframe, this course will emphasize

how to log on to the mainframe, edit programs, submit jobs, and review the results. Understanding and coding basic JCL (Job Control Language) is also covered.

Upgrading to SAS® Version 7 & 8

Designed for SAS software users who need a quick overview of the key changes in these new versions, this course focuses on enhancements including the Output Delivery System (ODS), additions to SAS Input/Output capabilities, and changes in the DATA step and procedure steps.

SyncSort®

This course provides an introduction to the SyncSort product. The emphasis is on sorting, selecting, and copying data. Learning to use this MVS software tool offers extremely fast performance with large data files.

SAS® for Business Applications

This course teaches users to harness the power of SAS to solve business problems. The course covers common business analytical tools in SAS and emphasizes situations that teach how to use each tool. It also covers interpreting results.

NEW SERVICES

SAS® Web Tools

SSC consultants offer expert assistance with SAS Web tools including HTML-ready reports, webAF, SAS/IntrNet, and the new Output Delivery System. We are also developing a course on this topic which will be offered later this year.

Star Client Program™

SSC is proud to offer a new program to our Star Clients. The program offers volume discounts, priority consulting, on-site placement, customized training, public class vouchers, and more.

Call Us

Contact Russ Lutz, (608) 278-9964 ext. 305 or rlutz@sys-seminar.com, for more information on any of these classes or programs.

IN THIS ISSUE

New Software Classes & Services for 2000: David Beam introduces SSC's new products and services for the upcoming year.....Page 1

Puzzler #3: Your chance to win a \$100 Training Certificate.....Page 1

Connecting with SAS/CONNECT®: Steve First continues his article on SAS/CONNECT with Data Transfer Services and Remote Library Services.....Page 2

Controlling Print Setup Options: Guest Author Delayne Stokke discusses issues for printing SAS output in Windows.....Page 4

SAS® Help Desk I/O: Kim Kolbe-Ritzow answers a help desk question on informats.....Page 5

Technical Credit:.....Page 7

Public Class Schedule:.....Page 8

Puzzler #3

MVS JCL's SET statement sets symbolics to values that can be used in the JCL stream. The SYSPARM option passes parameters from JCL to SAS.

In the example that follows, a non-SAS program, PGM1, needs to be given a parameter and it also needs to be passed to SAS. How can you alter the EXEC SAS statement to pass the SASPARM value to SAS and set the SYSPARM macro variable to 01JAN1999 from the SET statement?

```
//XXXX JOB (XXXX)
//SETUP SET SASPARM='01JAN1999'
//S01 EXEC PGM=PGM1,PARM=&SASPARM
//S02 EXEC SAS ---line to alter
PUT ***** SYSPARM=&SYSPARM;
```

Desired log

```
NOTE: SAS SYSTEM OPTIONS
SPECIFIED ARE:
      SYSPARM="01JAN1999"
```

```
%PUT ***** SYSPARM=&SYSPARM;
***** SYSPARM=01JAN1999
```

The first three readers to fax correct answers to us will each win \$100 training certificates. See April's issue for the results.



SYSTEMS SEMINAR CONSULTANTS, INC.
2997 Yarmouth Greenway Drive
Madison, WI 53711
Website: www.sys-seminar.com
Email: train@sys-seminar.com
Phone: (608) 278-9964
Fax: (608) 278-0065

NEW IDEAS

GUEST AUTHOR COLUMN



As you'll notice on the front page of this issue, Systems Seminar Consultants, Inc. is serious about keeping abreast with new SAS-related technologies. We have developed several new SAS courses and even a few non-SAS mainframe technology courses. We can now provide expert SAS internet development through our SAS IntrNet and webAF consulting. As we grow, we are also growing our areas of expertise. We are

expanding our consulting services into new technologies such as Cold Fusion and MS Access. Look for a future article on these exciting new consulting services.

In this issue, we are also trying something new. We have published a section of an upcoming SUGI paper written by Delayne Stokke, of Ames, IA. Delayne shared his paper with us at a recent SAS users group conference. We thought the topic was quite interesting and wanted to share his findings with our readers. As a result, you'll find Delayne's article starting on page 4 of this issue.

If you have any tips or articles you would like to share with our audience, feel free to email us. We are always open to new ideas. On the other hand, if you would like to see an article on a particular topic, also feel free to email us.

Sincerely,

Steve First
President



CONNECTING WITH SAS/CONNECT®

SECOND IN A SERIES

In our last issue, I discussed SAS/CONNECT connections and Compute Services which allows users to submit SAS jobs remotely. This article will examine the movement of data (upload and download) through two SAS/CONNECT services. In a future article we will look at moving SAS data without SAS/CONNECT via transport files.

SAS/CONNECT Services

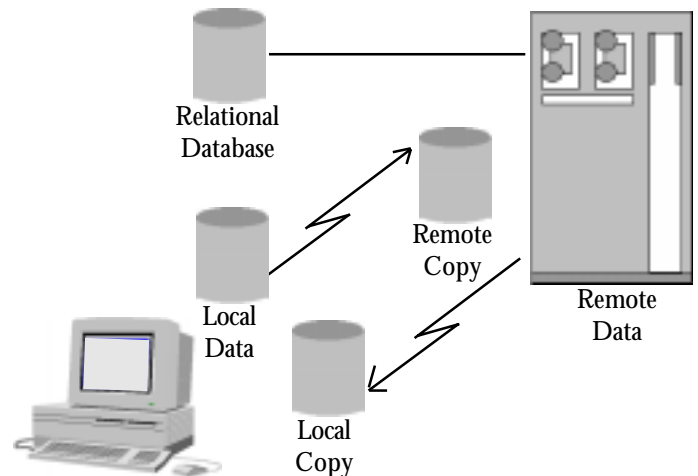
To review, three major services are available:

1. *Compute Services (remote submit)*: allows users to submit code on a local machine, run some or all of that code on a remote machine, and then transfer the results back to the local machine.
2. *Data Transfer Services*: allows users to upload and download SAS data files, text files, and binary files.
3. *Remote Library Services*: allows data that resides on a remote machine to appear as if it resides on the local machine.

Data Transfer Services

Two SAS procedures are available in SAS/CONNECT to move data to and from a remote computer. PROC DOWNLOAD moves data from the remote machine to the local machine, and PROC UPLOAD moves data from the local machine to the remote machine.

A Diagram of Data Transfer Services



Many users may want to move data between machines, and it is sometimes more difficult than it should be. There are many products that move data from one machine to another, and most times they work quite well.

Upload/Download Issues

After a connection is made, a basic architecture problem occurs as IBM compatible computers store character data using EBCDIC codes, while most other machines, including PCs, use ASCII codes. The EBCDIC code dates back to early computers and teletype equipment. Also, most mainframes store text data in fixed length records, and the system stores the record length in a control block. ASCII machines typically use a variable length record terminated by two special binary characters: the Carriage Return (CR) and the Line Feed (LF) characters. These again



SYSTEMS SEMINAR CONSULTANTS, INC.

Copyright © 2000 Systems Seminar Consultants, Inc. Madison, WI

All rights reserved. Printed in USA. The Missing Semicolon is a trademark of Systems Seminar Consultants, Inc., SAS, Base SAS, SAS/ACCESS, SAS/AF, SAS/CONNECT, SAS/FSP, SAS/GRAPH, and SAS/IntrNet are registered trademarks or trademarks of SAS Institute Inc., PowerPoint and Windows are registered trademarks of Microsoft Corporation, SyncSort is a registered trademark of SyncSort, Inc., and ISPF is a licensed program of International Business Machines Corporation, in the USA and other countries.

date back to the teletype machines that actually moved the carriage to the left and advanced the paper ahead one line.

When uploading/downloading character data, something needs to tell the download program to translate the characters to the different code being used. We also need to indicate whether the Carriage Return and Line Feed characters need to be added.

Sample Download Command (not SAS/CONNECT's)

```
RECEIVE C:\TEMP\PCDAT.DAT 'MY.DATA' ASCII CRLF
```

Binary files do not use the ASCII or EBCDIC codes and can also be moved, but in this case we need to NOT tell the system to translate or add CR and LF characters. Again, this is a fairly easy command and usually works well.

```
RECEIVE C:\TEMP\PCDAT.BIN 'MY.BIN'
```

A Basic Problem

While it appears the above commands will handle our upload/download problems, these commands cannot move data correctly when the file contains a mixture of character and binary data. This is because the ASCII and CRLF operands apply to the entire record.

One Solution: Use a SAS Dataset

One of the best features of SAS datasets is that the user does not need to know or care how data is stored in SAS files. So, by storing our data in a SAS dataset instead of a text or binary file, PROC UPLOAD/DOWNLOAD will handle all translation and transfer for the user automatically. This means that our SAS data can be a mixture of character and numeric data, and when transferred, we will get an equivalent SAS dataset on the other machine.

Not only can UPLOAD/DOWNLOAD move SAS datasets, it can also move entire SAS libraries, SAS catalogs, and text files. In addition, UPLOAD/DOWNLOAD can do WHERE processing and selecting on some of the members in a library.

Benefits of Data Transfer Services

- off-load from a remote system
- reduce connect time, decreasing cpu charges
- add robustness to local systems
- reduce network traffic with WHERE
- use transfer as a backup resource
- off-load to remotes for testing & development until productionalization

SEVERAL EXAMPLES OF PROC UPLOAD/DOWNLOAD

Transfer Select Records in a SAS Dataset to a Local Machine

```
PROC DOWNLOAD DATA=MFILE /* REMOTE SAS DS IN */
              OUT=PCFILE; /* LOCAL SAS DS OUT */
WHERE RATE > 5; /* SELECT SOME ROWS */
RUN;
```

Upload an Entire Library to a Remote Machine

```
PROC UPLOAD INLIB=PCFILE /* LOCAL SAS LIB IN */
           OUTLIB =MYFILE; /* REMOTE SAS LIB OUT */
RUN;
```

QUICK TIP

— To document your datasets, use labels; they don't take up much space and are easy to use.

run;

Download a Text File

```
PROC DOWNLOAD INFILE='MY.DATA'
              /* TEXT FILE IN */
              OUTFILE='C:\TEMP\PCDAT.DAT';
              /* TEXT FILE OUT */
RUN;          /* ASSUME ASCII, CRLF */
```

Download a Binary File

```
PROC DOWNLOAD INFILE='MY.DATA'
              /* BIN FILE IN */
              OUTFILE='C:\TEMP\PCDAT.DAT';
              /* BIN FILE OUT */
              BINARY; /* LEAVE BINARY */
RUN;
```

Generate and Run a Program to Download all Members of an MVS PDS to a PC Directory

```
RSUBMIT:
FILENAME PDS 'MY.SAS.PROGRAMS' SHR;
/* ALLOC PDS IN */
FILENAME PGM 'GENPROG.SAS' LRECL=80
/* PGM GEN */
BLKSIZE=6160 DISP=(,CATLG)
UNIT=SYSDA SPACE=(TRK,50);
PROC SOURCE NODATA NOPRINT INDD=PDS
/* READ SOURCE MEMBERS*/
OUTDD=PGM; /* GEN PROGRAM HERE */
BEFORE 'PROC DOWNLOAD INFILE=PDS(XXXXXXXX)'
28 NOBLANK;
BEFORE " OUTFILE='XXXXXXXX.SAS';" 28 NOBLANK;
BEFORE 'RUN;';
RUN;
%INC PGM; /* RUN GENERATED PGM */
ENDRSUBMIT;
```

Sign on to a Remote Machine, Use PROC SQL to Create a Table from a Sybase System, Download the Resulting Table

```
DM 'SIGNON';
RSUBMIT;
PROC SQL OUTOBS=10; /* TO SYBASE */
CONNECT TO SYBASE (USER=XXXXXX
                  PASSWORD=YYYYY
                  DATABASE=MAST CLAIMS
                  SERVER=UNIX01);
CREATE TABLE WORK.BIO AS /* SAS TABLE OUT */
SELECT MEMBER, /* OUTPUT COLUMNS */
GROUP,
MEMSEQ,
DATEPART(BIRTH) AS BIRTH
FORMAT=MMDYY10. ,
SEX
FROM CONNECTION TO SYBASE /* FROM SYBASE IN */
(SELECT * /* ALL COLUMNS */
FROM MAST BIO /* INPUT SYBASE TABLE */
WHERE BIRTH BETWEEN
"1996/01/01" AND "1996/01/31");
QUIT;
PROC DOWNLOAD DATA=WORK.BIO
              /* DOWNLOAD WORK FILE */
              OUT=PC.BIO;
ENDRSUBMIT;
```

Continued on page 8.....

CONTROLLING PRINT SETUP OPTIONS

FROM WITHIN A WINDOWS® SAS® PROGRAM

I've written many programs in SAS to produce reports. One of the problems I've had is keeping track of the appropriate Print Setup for each report. In the past, I've documented the appropriate Print Setup in program comments, but sometimes I've failed to change the Print Setup before running my report. When I didn't get the results that I wanted, I had to change the Print Setup and rerun the report. Recently I decided that there must be a better way. I've researched ways to control the Print Setup from inside a SAS program and have condensed my findings into a SAS macro called %SETPRINT. Using this macro, I can control the page orientation, the font size, and the font used for my printed output. This macro helps to ensure consistent results every time I run a report.

CONTROLLING PAGESIZE & LINESIZE

The Options Statement

Most SAS programmers have probably used the LINESIZE and PAGESIZE options at one time or another. PAGESIZE specifies the number of lines that can be printed per page of SAS output (page length) while LINESIZE controls the line width. Specifically, these options control the way print image outputs are formatted. By "print image output", I mean any output that is directed to the LOG or OUTPUT windows. LINESIZE affects output directed to both the LOG and OUTPUT windows, while PAGESIZE only affects output directed to the OUTPUT window. The most important thing to note here is that these controls only affect the way the output is formatted and displayed in the LOG or OUTPUT windows – they do NOT affect the print setup options at all.

The Print & Page Setup Dialogs

The Print & Page Setup dialogs allow you to interactively control the printer settings. Within these dialogs, you can:

- select a font, from a list of available fonts
- set the font size
- set the page orientation
- select a printer, set page margins, and change other settings

When you interactively change any settings through the SETUP dialogs, SAS automatically recalculates the LINESIZE and PAGESIZE values.

When SAS prepares output for the OUTPUT window, it uses the LINESIZE and PAGESIZE options that are currently in effect to format the pages. The current settings of these options could have been set explicitly through an OPTIONS statement, or they could have been set implicitly through the SETUP dialogs – whichever instruction was most recently received.

Warning

If the LINESIZE and PAGESIZE options that were in effect at the time the report was created have values greater than the LINESIZE and PAGESIZE calculated by the SETUP dialogs, there may be truncation or overflow of the printed report.

On the Windows platform, I recommend the following process for creating reports:

- change the printer setup BEFORE the program creates the report
- DO NOT include an OPTIONS statement with explicit LINESIZE and PAGESIZE settings in your program.

A PROGRAM STATEMENT TO SET THE FONT & FONT SIZE

As discussed above, you can interactively select a font and set the font size using the Print Setup dialog. You can also do this by using the SYSPRINTFONT= option. This option can be coded on an OPTIONS statement, for releases 6.12 and 8.00, as follows:

```
SYSPRINTFONT='fontname' <pointsize>
```

- Where...*fontname* is the name of the font to use for printing. This must be a valid, case-sensitive font face name (for example, 'SAS Monospace' or 'Courier'). This is a required argument.
- And...*pointsize* is the base point size to use for printing. This must be an integer from 1 to 7200, inclusive. If you omit this argument, SAS uses 10 points by default. Throughout this paper, I refer to this option with the term "font size" (or "fontsize").

PROGRAM STATEMENTS TO CHANGE PAGE ORIENTATION

Release 8.00 – It's Easy!

In SAS 8.00, you can specify the desired page orientation on an OPTIONS statement. The syntax for this option is:

```
ORIENTATION='orient'
```

- Where...*orient* is the desired orientation, either PORTRAIT or LANDSCAPE

Release 6.12 – Almost as Easy!

The SAS Display Manager for Windows provides the DLGPRTSETUP command to open the Print Setup dialog. The syntax for this command is:

```
DM 'DLGPRTSETUP ORIENT=orient NODISPLAY';
```

- Where...*orient* is the desired orientation, either PORTRAIT or LANDSCAPE. 'NODISPLAY' coded on this command prevents the Print Setup Dialog window from being displayed.

Continued on page 6.....

QUICK TIP

To clear up space in the middle of your programs, delete any unneeded data sets as soon as they are no longer needed.

```
Example: PROC DATASETS;  
         DELETE TEST;  
         QUIT;
```

```
or PROC SQL;  
     DROP TABLE TEST;  
     QUIT;
```



SAS® HELP DESK I/O

SOLUTIONS FROM OUR HELP DESK SERVICE



"I've heard that you can create formats with PROC FORMAT. Is it possible to create informats?"



For years SAS users have been using *SAS-defined* informats to transform their data as it is being read in. The most commonly used SAS-defined informats are date informats which help read in dates in various formats, transform them via the SAS date informat (i.e., DATE7., MMDDYY8., JULIAN7., etc.), and then store them internally as a SAS date value. There are also a number of other SAS-defined informats which help read in special types of data (packed dates using PDx., zoned decimal values using ZDx., and so on). An example of using SAS-defined informats:

```
DATA TEST;
  INFILE 'location-of-data';
  INPUT @1 DATE1 DATE7.
        @10 SALES PD4.2;  RUN;
```

What most people don't realize is that starting in Version 6 of SAS Software, you can also create your own user-defined informats to help with special types of data transformations:

```
PROC FORMAT;
  INVALUE $REGFM
    'OH','IL','WI'='MIDWEST'
    'GA','SC','FL'='SOUTH'
    'OR','ID','WA'='NORTHWEST'
    'NY','CT','PA'='NORTHEAST';  RUN;

DATA CHANGEIT;
  LENGTH STATE $9;
  INFILE 'location-of-data';
  INPUT @1 STATE $REGFM.;  RUN;
```

Specifying user-defined informats is one of the most efficient ways to perform a simple reassignment or transformation of a value because the transformation takes place *as the values are being read in*. Upon using a user-defined informat, the value is *now physically stored as its new value* (i.e., 'MIDWEST'). In order to have access to both values 'OH' and 'MIDWEST', the variable will have to be read in two different ways:

```
DATA CHANGEIT;
  LENGTH STATE $9;
  INFILE 'location-of-data';
  INPUT @1 STATE $REGFM.
        @1 STATE2 $2.;  RUN;
```

Prior to the introduction of user-defined informats in Version 6 of SAS Software, users would use IF-THEN logic, SELECT statements, or user-defined FORMATS to perform the simple transformations. The problem with these methods is that they required additional passes of the data, and in the case of the IF-THEN and SELECT statements, required additional variables to be created to perform the transformation, thus, diminishing their efficiency. User-defined informats, on the other hand, are more efficient because they allow the transformation to take place *at the point in which the data are being read in*.

Transformations using user-defined informats are only useful when performing simple reassignments. Complicated reassignment requiring AND, OR, IN, and NOT logic will still require the use of IF-THEN or SELECT statement logic to perform the transformations.

QUICK TIP

— To put the number of observations in a title, use the SET statement and the NOOBS option to query the compiler. SYMPUT can then create a macro variable usable in a title.

```
Example: DATA _NULL_;
          CALL SYMPUT('TOTMEMBS',TRIM
                    (LEFT(PUT(NMEMB,8))));
          SET INDATA NOOBS=NMEMB;
          STOP; RUN;
```



Using IF-THEN Logic

```
DATA CHANGEIT;
  LENGTH REGION $9;
  INFILE 'location-of-data';
  INPUT@1 STATE $2.
        @5 SALES 5.;
  IF STATE IN ('OH','IL','WI')
    AND SALES > 500
    THEN REGION='MIDWEST';
  ELSE IF STATE IN ('GA','SC','FL')
    AND SALES >= 300
    THEN REGION='SOUTH';
  ELSE IF STATE IN ('OR','ID','WA')
    AND SALES < 350
    THEN REGION='NORTHWEST';
  ELSE IF STATE IN ('NY','CT','PA')
    AND SALES > 1500
    THEN REGION='NORTHEAST';
  ELSE DELETE;  RUN;
```

Using the SELECT Statement

```
DATA CHANGEIT;
  LENGTH REGION $9;
  INFILE 'LOCATION-OF-DATA';
  INPUT@1 STATE $2.
        @5 SALES 5.;
  SELECT(STATE);
    WHEN ('OH','IL','WI')
    DO;
      IF SALES > 500 THEN
        REGION='MIDWEST';
    END;
  WHEN ('GA','SC','FL')
  DO;
      IF SALES >= 300 THEN
        REGION='SOUTH';
    END;
  WHEN ('OR','ID','WA')
  DO;
      IF SALES < 350 THEN
        REGION='NORTHWEST';
    END;
  WHEN ('NY','CT','PA')
  DO;
      IF SALES > 1500 THEN
        REGION='NORTHEAST';
    END;
  OTHERWISE DELETE;
END;  RUN;
```

In Summary

User-defined INFORMATS perform effective transformations for simple data values as they are being read in because the number of times the data are passed is reduced and the creation of additional variables in the data set is not required. Additionally, the PROC FORMAT code could be "lifted out" and used in more than one place via %INCLUDE statements or by saving the code to a permanent library.



CONTROLLING PRINT SETUP OPTIONS

CONTINUED FROM PAGE 4

CHECKING FOR THE CURRENT PAGE ORIENTATION

In some cases, I don't want to change the page orientation; I just want to know what it is.

Release 8.00 – It's Easy!

In SAS 8.00, you can determine the current page orientation using the GETOPTION function. For example, the statement:

```
ORIENT = GETOPTION('ORIENTATION');
```

- ...returns the current page orientation to the variable named ORIENT.

Release 6.12 – NOT so Easy!

SAS 6.12 doesn't provide a command that will return the value of the current page orientation. But I discovered a "feature" that would allow me to figure it out. Through much experimentation I discovered that in 6.12, the LINESIZE and PAGESIZE are NOT recalculated unless the page orientation changes. This is true even if you change the font and/or font size. Here's how I take advantage of this to determine the current page orientation:

- Use an OPTIONS statement to set the LINESIZE and PAGESIZE to their maximum possible values.
- Use a DM command to set the Page Orientation to PORTRAIT.
- Query the system to determine the current LINESIZE and PAGESIZE.
- If the current LINESIZE and PAGESIZE are still equal to their maximum values, the beginning Page Orientation is PORTRAIT; otherwise, the beginning Page Orientation is LANDSCAPE.

Sample code to illustrate the method follows:

```
%MACRO MORIENT;
options linesize=MAX pagesize=MAX;

      /* set orientation to portrait */
dm 'dlgprtsetup orient=PORTRAIT
nodisplay';

      /* get cur linesize & pagesize */
%let linesize =
  %sysfunc(getoption(linesize));
%let pagesize =
  %sysfunc(getoption(pagesize));

      /* are they still at the max? */
%if &linesize = MAX AND
  &pagesize = MAX %then %do;

      /* if so, orientation=portrait*/
%let orient = PORTRAIT;
```

```
%end;

      /*if not, landscape */
%else %do;
  %let orient = LANDSCAPE;
%end;
%MEND MORIENT;
%MORIENT
```

Controlling the Margins

SAS 8.00 provides the options LEFTMARGIN, RIGHTMARGIN, TOPMARGIN, and BOTTOMMARGIN to control the indicated margins. Just specify the desired margins in decimal inches. For example:

```
options leftmargin=0.5 rightmargin=0.5;
```

You can't change margins with program statements in SAS 6.12. You must activate the Print Setup dialog and then open the Page Setup dialog. I've not found this to be a big problem, since most of the time I leave all the margins set to their defaults. However, there are times when it would be nice to be able to control the margins from within a program.

The "SETPRINT" Macro

I've compiled what I know about using SAS program statements to control the print setup into an easy-to-use macro. The macro is named "SETPRINT.SAS" and accepts up to 3 positional parameters:

- "orient" is the first parameter, and it is used to specify the page orientation. Possible values are PORTRAIT or LANDSCAPE. If not specified, the macro will determine the current page orientation. For Version 8.00 this parameter defines the setting used by the ORIENTATION option. For Version 6.12 this parameter is the same as the ORIENT= parameter used by the DLGPRSETUP command. This parameter is optional.
- "fontsize" is the second parameter, and represents the base point size to use for printing. This must be an integer from 1 to 7200, inclusive. This parameter is the same as the "pointsize" parameter used by the SYSPRINTFONT option. However, the default behavior for the SETPRINT Macro is different. As used by SETPRINT, this parameter has no default value, and can be specified without specifying a "font". In that case SETPRINT uses "SAS Monospace" as the default font.

This parameter is optional, however, it becomes **required** if a font is specified using the "font" parameter (described below).

- "font" is the third parameter, and is the name of the font to use for printing. This must be a valid, case-sensitive font face name (for example, 'SAS Monospace' or 'Courier'). This parameter is the same as the "fontname" parameter used by the SYSPRINTFONT option. Again, the default behavior for SETPRINT is different than the SAS option. As used by SETPRINT, this parameter has no default value, unless a "fontsize" has been specified. In that case, "font" will take on a default value of "SAS Monospace". This parameter is optional.

QUICK TIP

— Instead of merging a dataset with another to add a variable from the second, why not use a format?



Additionally, SETPRINT will accept up to 4 optional keyword parameters. Use the following parameters to set the respective margins in decimal inches when running under SAS 8.00.

- “left=” sets the LEFTMARGIN
- “right=” sets the RIGHTMARGIN
- “top=” sets the TOPMARGIN
- “bottom=” sets the BOTTOMMARGIN

Setprint Design Considerations

In designing the macro, I changed the default behavior of the “fontsize” and “font” parameters to suit my purposes. I found that for most reports, I wanted to control the page orientation and the font size. I also reasoned that “SAS Monospace” should be available on any Windows system running SAS and is almost always an acceptable choice for a report. For those reasons, I chose to make it the default.

When invoked, SETPRINT will make the specified Print Setup changes, and then write a message box to the SAS Log to display the current print setup specifications. See Figure 1.

(This article and its appendix, which includes figures 1, 2, and 3, can be found at <http://www.sys-seminar.com/papers.htm>.)

In addition to being able to change print setup specifications, I also wanted the macro to be able to report on the current print setup – without changing any values. For that reason, none of the parameters have default values. If no parameters are specified, the macro will simply write the Print Setup message to the SAS Log. Since there is no way to determine the currently selected font or font size, those items are listed as “Unknown”. See Figure 2, also at www.sys-seminar.com/papers.htm.

SETPRINT Usage

On my system, I’ve put the SETPRINT.SAS source code into the :SAS\CORE\SASMACRO folder. This is a system folder that is defined as the default macro autocall library. By placing SETPRINT there, it is always available within my SAS session.

The SETPRINT macro is easy to use. During the process of developing a report, I determine the print setup that is appropriate. If the report is best formatted using a Portrait orientation with a 10 Point SAS Monospace font, I just include the following line in my program, somewhere before the report is created:

```
%setprint (Portrait,10)
```

Specifying an orientation and a font size is how I usually use the macro. But if I wanted to change the font and the font size without changing the orientation, I could include the following macro call:

```
%setprint (font=Courier,fontsize=8)
```

Figure 3, also at SSC’s website, contains the entire source code for the SETPRINT macro. I’ve tried to include sufficient comments to clearly describe the operation of the macro.



QUICK TIP

— Sometimes it is necessary to include additional variables in the output datasets created in PROC summary. To take along the earliest value, use the idmin option. In the example below, we want to save the earliest sale date for each customer, along with total sales.

```
Example: PROC SUMMARY DATA=SALES NWAY IDMIN;
VAR SALEAMT;
ID SALEDATE;
CLASS CUSTNUMB;
OUTPUT OUT=SALESUM
N(SALEAMT)=NUMSALES
SUM(SALEAMT)=TOTSALES; RUN;
```



TECHNICAL CREDIT AND RECOGNITION



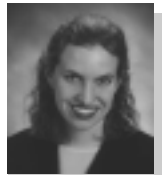
David Beam
Vice President
Author of this issue's:
New Software Classes & Services for 2000, Page 1



Steve First
President
Author of this issue's:
Connecting with SAS/CONNECT®, Page 2



Kim Kolbe-Ritzow
Trainer/Consultant
Author of this issue's:
SAS® Help Desk I/O, Page 5



Katie Minten
Trainer/Consultant
Author of this issue's:
Quick Tips, Pages 3-7

Controlling Print Setup Options.....Delayne Stokke
PublisherJodie Schmidt
EditorsDavid Beam, Russ Lutz, & Cindy Kersten



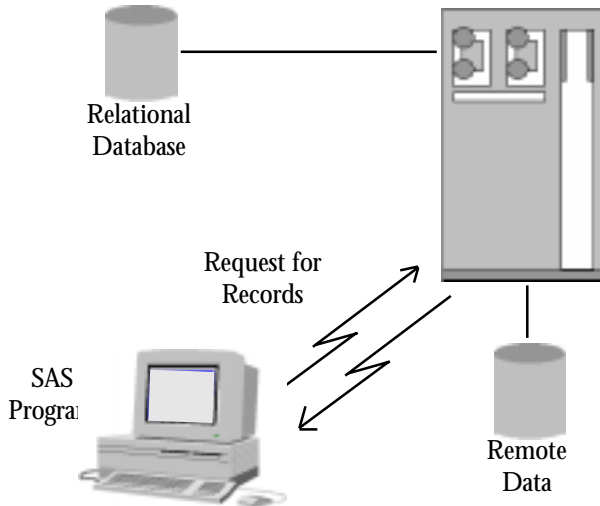
CONNECTING WITH SAS/CONNECT®

CONTINUED FROM PAGE 3

Remote Library Services (RLS)

When using RLS, a SAS LIBNAME statement is submitted locally to assign a libref to a library on a remote server. From that point on the library acts like a local library with SAS getting data as it is requested. While this is the most transparent method in SAS/CONNECT for moving data, it is only efficient when moving small amounts of data.

A Diagram of Remote Library Services



Assign a Libref to a Remote Library from a PC

```
LIBNAME MVSLIB 'MY.MVSSAS.DATA' /* ALLOCATE ON */
SERVER=MVS1; /* ANOTHER HOST */
PROC PRINT DATA=MVSLIB.DATA1(OBS=10);
/* LOCAL NOW */
RUN;
```

I have shown a few examples of how SAS/CONNECT can move data. For more information about SAS/CONNECT feel free to call me, or consult the SAS/CONNECT documentation.



PUBLIC CLASS SCHEDULE

Introduction to SAS®

January 25-27	\$725	St. Paul, MN
February 7-9	\$725	Madison, WI
March 13-15	\$725	St. Paul, MN
April 3-5	\$725	Madison, WI
April 10-12	\$725	St. Paul, MN
May 15-17	\$725	St. Paul, MN
June 12-14	\$725	St. Paul, MN
June 19-21	\$725	Madison, WI

The SAS® SQL Procedure

May 18	\$350	St. Paul, MN
--------	-------	--------------

Introduction to PROC Report

January 28	\$350	St. Paul, MN
------------	-------	--------------

SAS® Report Writing

April 6-7	\$525	Madison, WI
April 13-14	\$525	St. Paul, MN

SAS® Efficiencies - New Class!

February 17	\$350	Madison, WI
-------------	-------	-------------

Advanced SAS®

February 23-25	\$725	St. Paul, MN
June 7-9	\$725	St. Paul, MN
June 22-23	\$525	Madison, WI

SAS® Macros

February 10-11	\$525	Madison, WI
March 16-17	\$525	St. Paul, MN

We also provide a variety of private on-site SAS classes. Call (608) 278-9964, ext. 311 for details. Course descriptions for the SAS classes we offer are available on our website.

To register call (608) 278-9964 or visit www.sys-seminar.com.



SYSTEMS SEMINAR CONSULTANTS, INC.

2997 Yarmouth Greenway Drive
Madison, WI 53711

PRSR STD
U.S. POSTAGE
PAID
MADISON, WI
PERMIT #2783

Call or visit our website for your
free subscription to
The Missing Semicolon™.