PROC FSVIEW: A REAL PROGRAMMER'S TOOL
(or "a real programmer doesn't use PROC PRINT")

Marge Scerbo and Alan Wilson
CHPDM-University of Maryland at Baltimore County
UMDNJ-Robert Wood Johnson Medical School

Abstract

Although SAS/FSP® is often regarded as the domain of application developers using SAS/AF®, SAS/EIS®, SCL and FRAME, new enhancements have made PROC FSVIEW an efficient and effective tool for programmers and researchers to browse and edit data, debug SAS code, perform multi-level queries on-line and create reports and new data sets interactively.

An on-line demonstration will show subsetting with WHERE and WHERE ALSO clauses, interactive SORTing, and rearranging the displayed columns as scrolling or ID variables. This tutorial will also include examples of modifying existing variables and creating new variables interactively using FSVIEW formulas. In addition, and perhaps most importantly, the when and why to use PROC FSVIEW will be discussed.

Introduction

One could say that PROC PRINT is a simple snapshot while PROC FSVIEW is an interactive video. The PRINT procedure allows a programmer to create a static view of the data; in an FSVIEW session the programmer can interactively change the display of the data and turn it up, down and around to allow alternate views of the data. In addition, FSVIEW provides the opportunity to edit the data set interactively.

In a perfect world, data is entered cleanly with validation of each field as well as each record. The observations in the file create a whole and work together. But this is not a perfect data world. Validating each field can slow a system to a crawl. There is rarely validation across records or across files except for table lookup checks. A researcher cannot progress to reporting and data analysis without really "seeing" what is in the file. Grasping a basic understanding of where or how outliers occur is needed. And so it comes down to the tedious task of "looking" at the data. There is simply no way around an intimate understanding of the data if the research is to be valid. In order to understand the data, it must be viewed from several perspectives.

PROC FSVIEW can play an important role in the everyday life of a researcher. It can be used to view the initial data as it is stored in the data set. It can then be used to track the manipulation of the data values as the DATA and/or PROC steps progress. It can also be used to view the 'final' data set to be analyzed; this 'final' copy may have been subset, sorted, and contain new calculations, all which could have been constructed prior to the view and/or interactively during the session. This 'new' data set can be filed under a new name for future analysis.

This paper will step through the process of 'looking at the data' using FSVIEW. The steps themselves are easy and could fit into a beginning tutorial. But the processes discussed will best suit the needs of more advanced programmers who may be faced with the problems mentioned. The goal of this tutorial is to introduce the new facets of this procedure and their potential use in the everyday life of a programmer or researcher.

The DATA

One data set will be discussed and manipulated throughout the paper. This data set contains 17 variables and over 2500 observations. Here is the PROC CONTENTS output of the data set:

NAME	TYPE	LENGTH	LABEL
AGE	1	4	Age as of March 31
AID1	1	4	Eligibility Category
BEGDOS	1	5	Beginning date
COUNTY	1	4	County
DOSDIS	1	5	Date of discharge
DX2	2	5	Secondary diagnoses
MCARE	2	1	Medicare Indicator
PAY	1	8	Payment
PDX	2	5	Primary diagnoses
PLACE	1	4	Place of service
PROCODE	2	5	Procedure code
RACE	1	4	Race
RECID	2	11	Recipient ID
SERVCD	1	4	MA service code
SEX	1	4	Gender
SPEC	1	4	Provider specialty

This data is a sample from Maryland Medicaid claims data. The file includes data on each procedure performed by a

physician in a variety of sites (PLACE): hospitals, clinics, office, etc. The records will also include diagnosis codes (ICD9-CM codes), dates, and payments for the treatment and patient information such as gender, age, race, and a flag indicating Medicare coverage. The data were only partially validated at the time of data entry.

PROC PRINT

PROC PRINT has long been a basic tool of the SAS programmer and there is no reason for this not to remain so. It provides a simple method to view the contents of a data set. This procedure allows selection and order of variables (with use of the VAR statement) and selection of observations (with use of the WHERE statement). But once a PROC PRINT output has been produced, the information is fixed (unless of course someone chooses to manually edit it).

PROC PRINT has been around since the beginning of base SAS. SAS/FSP for Version 5 was developed over ten years ago by the Display Products Division, under the leadership of a certain Dr. James H. Goodnight. The predecessor to PROC FSVIEW was PROC FSPRINT, which is still accepted as an alias.

PROC FSVIEW

PROC FSVIEW is an interactive procedure allowing full screen browsing and editing of SAS data sets and SAS data views created with the SQL procedure or SAS/ACCESS. FSVIEW will display the data set or view as a table with observations as rows and variables as columns. It can be run from a batch program or from the display manager menus or the command line. FSVIEW can be used to manipulate the display of the data; to calculate new variables (either permanent or temporary); and create a new data set, be it a subset, resort or recalculation of the original data set. The use of menu objects and the mouse, available in a windowing environment, makes it easier for most people to use, ***Select Globals-> Manage- > Open Table***. For those oldtimers who avoid the use of the mouse, commands can be entered from the command line.

Proc FSVIEW is but one of the procedures included in the FSP software product line. Other procedures include FSEDIT and FSLIST, but FSVIEW is probably the most useful procedure to a researcher or anyone who deals with dirty, inconcise, or poorly documented data.

Browse vs Edit Mode

By default, an FSVIEW session is opened in browse mode which does not allow editing. The letter (B) is displayed at the top of the window. In almost all cases, the processes described in this paper can be run in browse mode with the exception of sorting the data set.

In using PROC FSVIEW in place of PROC PRINT as is suggested by the title of the paper, browse mode indeed offers almost all of the tools needed to interpret and understand the data. Understanding the data is a key element in research.

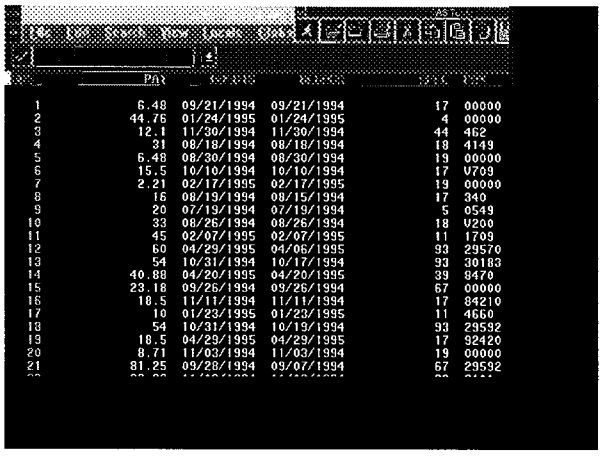
Note that FSVIEW allows the editing of the data set in place -- not a copy but the actual data set. The emphasis of this paper is the manipulation of the display of the data, calculation of new variables (permanent or temporary) and the creation of a new data set, either a subset, resort or recalculation of the original data set.

This procedure is a tool that can grow on a programmer. The more it is used, the more useful it becomes. This is particularly the case in displaying data that is not clearly documented and/or validated. It provides the programmer or researcher a starting point on which to continue the study. With more use, it also becomes an invaluable utility for viewing the progress of data set manipulation or creation.

To submit an executable statement to SAS, the program can be as simple as:

```
PROC FSVIEW DATA = DEMO.CLAIMS; RUN;
```

assuming of course that a library has been defined as DEMO. There are options available for the FSVIEW statement, including VAR, WHERE, etc. The default screen for the above code would be displayed as:



	PAY			
1	6.48	09/21/1994	09/21/1994	17 00000
2	44.76	01/24/1995	01/24/1995	4 00000
3	12.1	11/30/1994	11/30/1994	44 462
4	31	08/18/1994	08/18/1994	18 4149
5	6.48	08/30/1994	08/30/1994	19 00000
6	15.5	10/10/1994	10/10/1994	17 V709
7	2.21	02/17/1995	02/17/1995	19 00000
8	16	08/19/1994	08/15/1994	17 340
9	20	07/19/1994	07/19/1994	5 0549
10	33	08/26/1994	08/26/1994	18 V200
11	45	02/07/1995	02/07/1995	11 1709
12	60	04/29/1995	04/06/1995	93 29570
13	54	10/31/1994	10/17/1994	93 30183
14	40.88	04/20/1995	04/20/1995	39 8470
15	23.18	09/26/1994	09/26/1994	67 00000
16	18.5	11/11/1994	11/11/1994	17 84210
17	10	01/23/1995	01/23/1995	11 4660
18	54	10/31/1994	10/19/1994	93 29582
19	18.5	04/29/1995	04/29/1995	17 92420
20	8.71	11/03/1994	11/03/1994	19 00000
21	81.25	09/28/1994	09/07/1994	67 29532

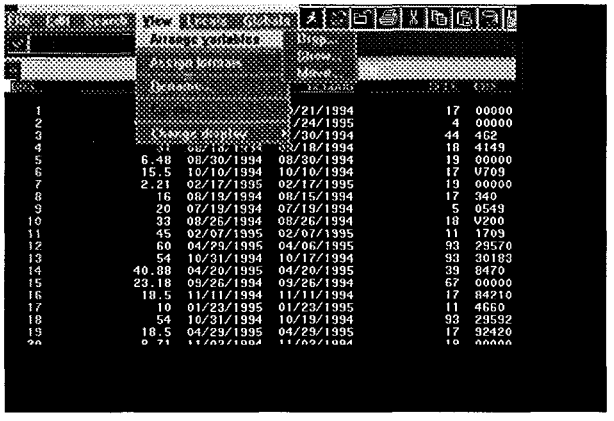
Note that the variables appear as columns and are listed in the order in which they were stored in the data set. The records are displayed in rows, each identified by an observation number. PROC FSVIEW displays as many observations and variables at a time as can fit on the

screen. If the data set is longer or wider than can be displayed on a screen and a mouse is identified, scroll bars will appear on the bottom and right side of the screen; these can be used to move from one variable to another or from one observation or another. If a mouse is not available, the commands right and left, up and down, etc, can be used on the command line or assigned to function keys.

Manipulating the Display

Once a PROC PRINT has been run, 'What you see is what you get'. This is not the case with PROC FSVIEW.

Often there are more variables displayed than are actually needed for a current indepth study of the data. To 'drop' a variable or variables, select ***View->Arrange Variables***. The three options under this submenu are ***Drop...***, ***Show...*** and ***Move....*** With these options, the user can control which variables are to be displayed and in what order. The screen displayed is:



To drop one variable, simply enter the variable name in the box and press ENTER. Often there are multiple variables to be removed from the display. To accomplish this without individually naming each variable, click on the OK and a list of variables will be displayed. Highlight (press the Enter key or mouse button) the variables to be dropped. If there are more 'pages' of records, use the page up and page down key to move backward and forward through the data set.

The list of variables is useful in that it contains the variable name, variable type, and label. In a large data set this makes the process of distinguishing one variable from another much easier.

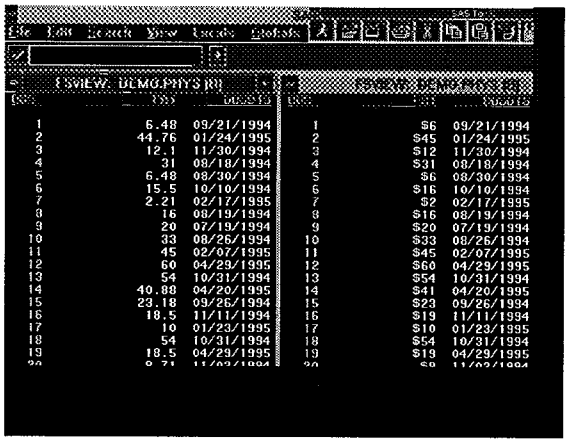
The list displayed on the screen would be:

If a dropped variable needs to be redisplayed on the screen, again select the ***View->Arrange Variables*** submenu, but choose ***Show....*** Use the default VAR option and click on OK. Select the variables to be 'added' to the display and press OK.

Another option found in the ***Arrange Variables->Show...*** menu is the ability to set an ID variable (non-scrolling variable). (The default color for ID variables differs from scrolling variables on color display monitors.) Choose the menu option ***Show....*** Rather than displaying the observation number in the leftmost column, this column will be set to any variable or variables chosen from the data set. The maximum combined length of all ID variables is 60 characters. When multiple observations have the same key, the ID variable will clarify the groupings of the observations. The observation number may be determined by entering the OBS command while the cursor is on the record of interest.

Use the ***View->Rename*** selection to change the variable name and the ***View- >Assign Formats->Format...*** and ***Informat...*** options to change the variable attributes. For example, to change the format of the variable, PAY, to a more readable value, choose ***View->Assign Formats->Format...*** from the menu. Enter the variable name, PAY, and the format, DOLLAR10.. Use formats or informats which are SAS system or user-defined formats. When a user-defined format is to be used, it must have been identified by a temporary PROC FORMAT statement preceding the PROC FSVIEW or a LIBNAME LIBRARY statement pointing to the location of a Format catalog.

This screen shows the before and after effects of a format imposed on the variable, PAY:



For advanced programmers the mechanism for use of the ***Arrange Variables->*** menu item is fairly intuitive and easy. The strength of this menu is best defined as a method for visualizing unknown or undefined data. In data collected by other sources, including government agencies, the file layout is only the tip of the iceberg in the comprehension of the data. Fields which are defined in one fashion may actually include wrong data, undefined data, and other anomalies. These can affect the outcome of the study. Therefore, getting down and dirty with the data can help validate the study.

Where Subsetting

There are two methods for imposing a WHERE subset on a data set to be viewed by FSVIEW, a permanent subset and a temporary one. If the FSVIEW statement is submitted to SAS with a WHERE statement as in:

```
PROC FSVIEW DATA = DEMO.CLAIMS;
      WHERE PAY GT 10000;
RUN;
```

Only observations which meet the specification(s) will be displayed with the word (**Subset**) in the window title. The data set can then be viewed and further subsetted, but those initially 'removed' observations will not be included. This is a permanent WHERE clause and is equivalent to a WHERE= data set option. The above code affects only the display of the data set; the number of observations in the data set upon exiting FSVIEW will remain the same as the original data set before the WHERE subset was imposed, even if edit mode was selected.

Submitting an FSVIEW statement without any initial subset makes the entire data set is available for viewing. By selecting the ***Search-> Where...*** menu item, various forms of the temporary WHERE clauses are available, and the word (**Where...**) appears in the window title when imposed. The primary clause is the straightforward WHERE. A box will be displayed which allows the entry of a WHERE clause (the WHERE keyword is not needed).

Note that as long as a WHERE clause is active, the observation numbers cannot be used to scroll a particular observation and the SORT command is not allowed.

A WHERE clause includes a valid arithmetic or logical condition statement. This statement should be specific to either character or numeric types, dependent on the variable to be used for the subset. Character values must be enclosed in quotes and appear in the same case as the variable values. Functions can be used in the statement as well. For example, WHERE

```
ROUND(PAY) GE 12000
      or
DOSDIS = TODAY()
```

A WHERE clause can be augmented by use of the ***Search-> Where also...*** option. This subset is imposed on the initial or subsequent subsets. A WHERE clause imposed on an already WHERE-subsetted data set will undo the last subset. Use **Where Also** to impose a second subset. To remove the last (most recent) subset, use the ***Where -> Undo last where*** selection. To cancel all WHERE subsets, simply select ***> Where...*** and submit it with no criteria, or enter WHERE CLEAR on the command line.

These WHERE statements may become even more useful with additional operators. For instance, the CONTAINS operator is very helpful in character string searches. Searching through a table for a hospital name that includes 'HOPK' to find all hospital which might be associated with JOHNS HOPKINS, such as WHERE

```
HOSPNAME CONTAINS 'HOPK'
```

The IN operator is useful when looking for observations where data may be found in a list of numbers or character strings. For example searching a primary diagnosis (PDX) within a set of codes where only the first three characters are used, WHERE

```
SUBSTR(PDX,1,3)IN('295','300','301')
```

Another useful technique is to use the BETWEEN-AND operators. This allows a search between two different numbers or character strings. It is a simpler form of a phrase which includes less than and greater than. For example WHERE

```
PROCODE BETWEEN 'R2100' AND 'R3400'
```

To view all observations where a variable with only nonmissing values is displayed:

```
PROCODE IS NOT MISSING
```

By using various combinations of WHERE clauses, the mix of data stored in a particular data set can be clarified. It can help lead to a more efficient method of searching the data set for a particular set of observations in a larger program, and it may reveal which fields contain outliers or errors in entry.

Formulas

PROC FSVIEW provides the ability to use formulas to create new variables and redefine existing variables. These formulas can be calculated from the values of any variables in the data set (even if these variables are not displayed).

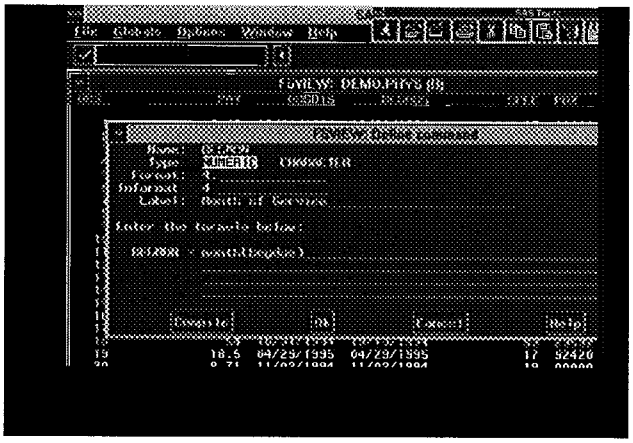
Formulas can be created, reviewed, cleared, saved and reused with the following menu options: ***Locals->Define formula, Locals-> Review formula,, Locals-> Clear formula..., File-> Save as -> Save Formula as..., File-> Open-> Read a formula...*** .

Creation of a formula can be done interactively in the ***Locals->Define Formula...*** window, using as many SAS code statements as will fit on the four lines available in the window as shown below.

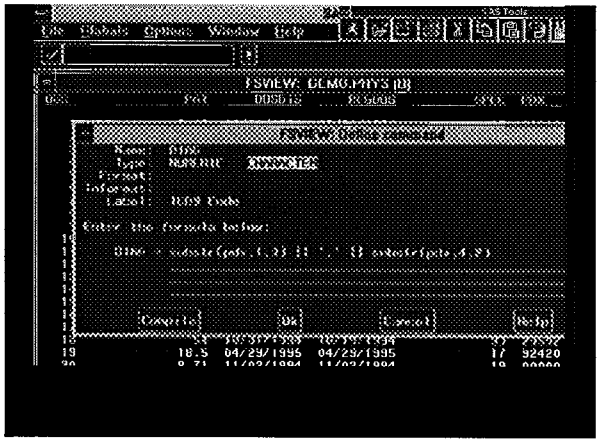
The Name field defines the variable to be modified or created. Enter the name of an existing variable to modify its values and press ENTER. To create a new variable, specify a new variable name whose values are the result of the calculations. (This new variable is added only to the display, not to the data set.) The Type field specifies the variable type. To select a type, select either NUMERIC or CHARACTER (NUMERIC is the default.) The Label field assigns a descriptive label up to 40 characters long, although a label associated with the variable is not displayed in the FSVIEW window.

The formulas defined in a session can be Reviewed or Reset by the corresponding commands. The selection ***Locals->Review formula.*** opens the Review window. The selection ***Locals->Clear formula...*** will open the Clear formula window. Enter the selected variable name or ALL to reset (delete) all computed variables.

To create a formula, specify one or more statements (as many as will fit on the four lines), separating multiple statements with semicolons. Here is a short formula that creates a new variable which contains the month the procedure was performed:



Any function or operator is available for use in a formula. The concatenation operator, | |, is used with the substring function to create a formatted diagnosis to match typical ICD-9 codes.



The length of stay can be computed by subtracting the beginning date of service from the date of discharge:

```
LOS = DOSDIS - BEGDOS
```

The COMPILE button performs a syntax of the formula. To locate the message concerning the syntax error in the formula, use the LOCALS menu to move to the MESSAGE WINDOW. To return from the Define Formula window, select OK at the bottom of the window. To cancel the formula definition and close the window, select the CANCEL command.

The formula is immediately applied to the data set and displayed. The new variable appears at the far right of the display (the last variable in the data set). Although this field is added to the display, it is not added to the data set unless a new data set is created.

If browse mode, any newly calculated values will be temporary. However if the edit mode is in effect (the letter (E) is displayed), then any edited variable values will be replaced and newly calculated variables will be stored immediately.

Exiting the FSVIEW session without saving the formula will cause a message warning to appear which states that the formula will not be stored.

```
WARNING: No formula catalog.
Use SAVE FORMULA command to store formulas
```

Unless a formula created has been stored, it will be deleted at the close of the FSVIEW session. Remember that the changes made affect the display only and are not stored in the data set at the end of the FSVIEW session. To modify the data set permanently with a formula, edit mode must imposed.

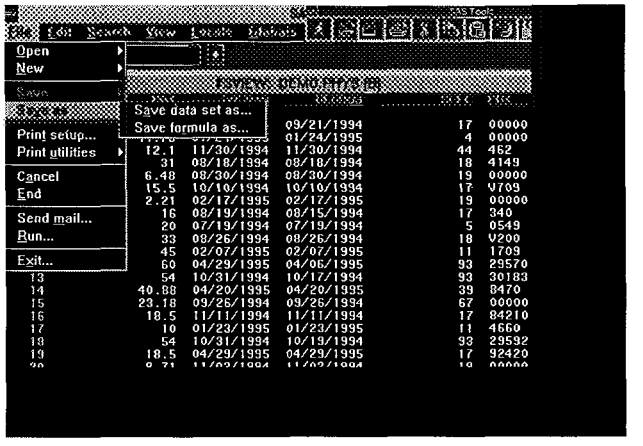
To store the formula permanently, choose ***File-> Save as -> Save Formula as...*** . Enter the location and name of the formula. If this method is not used, by default, saved formulas are stored in the SASUSER.PROFILE catalog, a user profile catalog that the SAS System creates. Therefore, if OK is selected from the FSVIEW Save Formula as dialog box, an entry with the same name as the data set is stored in the SASUSER.PROFILE. The formula stores the names and order of the variables displayed and their formats and informats, plus the formula definitions and screen colors. The formula does not record which observations are currently displayed (determined by WHERE clauses).

In later FSVIEW sessions, to reuse a stored formula associated with the data set and perform the calculations specified in an existing formula, access the formula by selecting ***File-> Open-> Read a formula...*** To read in a formula simply specify its name and exact location.

Formulas can be of great use in the analysis process. A researcher can view the data from different angles. This option provides the ability to create new variables from existing variables and then sort or subset the data set and view the results. The process becomes a 'living data set'. The data is static in its original form but is 'alive' under the microscope of FSVIEW.

Saving the File

The ***File*** option of the menu bar allows for the saving of the original data set or a new data set. To create a new data set, choose the ***File- > Save As...*** :



All the variables, or optionally only a subset, can be selected for the new data set.

Sorting the Data Set

The capability to sort SAS data sets interactively has been available since FSPRINT (Version 6.03 for PC's) and continues in FSVIEW. To sort the data set the default browse mode denoted by (B) must be changed to the edit (E) mode. The SORT command sorts the physical data set, not just the current display. To preserve a copy of the original data, be sure to save a copy of the data set **before** the SORT command has been issued.

The SORT command (with the Ascending|Descending modifiers) can be applied to one or more variables.

```
SORT AGE SEX DESCENDING PAY
```

Because SORT cannot use computed variables, create a new data set containing the new variable and then SORT. In addition, SORT cannot be performed if a WHERE (permanent or temporary) clause is in effect.

Printing the Display

Choose the ***File-> Print utilities->*** menu option and then

the ***Screen print*** option to print a 'snapshot' of the screen. This is useful after various techniques have been executed and the data displayed shows a particular trend. The printout will contain only the variables and the observations displayed and not the entire data set.

The Why, When and When not

Why use it?

Use it because the better a researcher or analyst knows the data, the cleaner and surer the results will be.

When else can FSVIEW be used?

Use it generously to debug the development of a data analysis and to check the results of logical procedures to make sure what resulted is what was intended. Place PROC FSVIEW calls liberally among the steps of a program. After test analysis is complete, comment out the FSVIEW portions and rerun, to avoid having to interact with the completed data analysis code. This step by step process can be of use to both new and old programmers alike. For experienced users, it can be used as a method of 'tutoring' new researchers in how to track the data and its changes through the DATA and PROC steps.

When not to use?

Since it is possible to interactively change the data, a common mistake is to edit with no audit trail. If the data set is derived from an external source which contains the original errors, if and when the data set is recreated, the original errors are restored. Result: Wasted work and time.

Always use caution when working within an edit session of FSVIEW; all edits and sorts change the data set immediately--interactively. The ***File-> Cancel...*** selection does not 'undo' any of these changes.

For researchers, understanding the data and how to analyze it is far more important than the ability to change the values. Exclusion of certain observations is a more accepted method for the analysis of data than editing of the data values.

What does the future hold?

The Data Table and Data Form objects (experimental in Release 6.11) can be used for browsing and editing SAS data sets. Invoked from within a SAS/AF FRAME application, the Data Form object is similar in some ways to the SAS/FSP FSEDIT procedure, while the Data Table is like PROC FSVIEW.

Among the advantages of the Data Table are that column widths are adjustable and column labels can be displayed.

References:

Computer Based Training (CBT) SAS tutor and FSP Online Help Screens

PP56001 SAS/FSP Software: Usage and Reference, Version 6, First Edition

PP55147 SAS Screen Control Language: Reference, Version 6, Second Edition

PP59133 SAS Technical Report P216, SAS/AF, SAS/FSP and SAS SCL: Changes and Enhancements, release 6.07

Observations 1Q93:58

Observations 1Q92:57

Observations 1Q96:

Scott Williams, Rapid Applications Development using the Data Form and Data Table Objects in SAS/AF for Release 6.11, SAS Users Group International Proceedings. SUGI 21, March 1996

For more information contact:

Marge Scerbo
410-455-6807
scerbo@umbc.edu

Alan Wilson
908-235-7857
awilson@umdnj.edu