THE SESUG INFORMANT

SouthEast SAS Users Group

Volume 4, Issue 1

Spring 2002

EDITOR'S MESSAGE

Randy C. Finch

As you probably remember, this front page is normally shared by SESUG's president and this newsletter's editor, which is me. Well, unfortunately (or perhaps fortunately for some of you) our president, Joe Kelley, failed to meet the deadline for his column. Therefore, I am claiming all his space as mine. Yes, it is MINE! ALLLLL MIIIIIINE!!!!!! Uh, well, excuse me. I got a bit carried away.

I don't know about the rest of you, but I really enjoyed SESUG's joint conference (SSU 2001) with the South Central SAS Users Group (SCSUG) in New Orleans this past August, and I am not that big a fan of New Orleans. Dave Riba and Debbie Buck, along with their entire conference team did a great job putting this conference together. And considering that this was the first ever joint conference of two regional SAS user groups, I would have to say they did a *TREMENDOUS* job.

Now it is time to begin looking toward SESUG '02, which will take place in Savannah, Georgia, on September 22-24. This conference is in the capable hands of David Maddox and Heidi Markovitz. They have their conference team together and are putting together what appears to be another successful SESUG conference. Of course, its ultimate success lies in your hands since without any attendees, it is extremely difficult to have a successful conference. So, all of you reading these words, be sure to make plans to attend SESUG '02. Convince your friends and co-workers to attend also. We want to see you all.

SESUG '02 is a milestone since it will be the tenth SESUG conference. It was in February 1993 that SESUG kicked off its existence with SESUG '93 at the Tradewinds in St. Petersburg Beach, Florida. (I have to admit that my favorite part of that conference was the guy that did a Groucho Marx imitation at the opening session. My apologies to all the paper presenters at that conference.) I would personally like to thank Dave Riba for being one of the co-founders of SESUG and for being actively involved in the organization ever since.

This issue of *The SESUG Informant* marks the beginning of the fourth year of its existence. I would like to personally thank Randy Finch for being its editor from the beginning. (Hold the complaints please. Since Joe wasn't here to thank me, I had no choice but to thank myself.) I hope you enjoy this issue. We have some of our regular columnists back with us. Ian Whitlock has another interesting article about the inner workings of SAS. Kirk Lafler treats us to two articles, one of which was co-written with Charles Shipp. As a special treat, SESUG's co-founder Dave Riba pens a review for a new book by long-time SESUG supporter Ron Cody. Also, be sure to look at our advertisements. Without the support of our sponsors and advertisers, this newsletter could not be printed and distributed with no cost to its recipients.

Have you written any short articles that you would like to see in this newsletter. I'll be glad to review them and consider them for a future issue. My contact information is to the right. If you are part of a company that would like some advertising exposure to the highly targeted audience receiving this newsletter, please contact me.

Until next time, I bid you adieu.

Inside this issue:

The SESUG Informant; Spring 2002; Volume 4, Issue 1; ©2002 SouthEast SAS Users Group, Inc.

The SESUG Informant is a publication of the SouthEast SAS Users Group, Inc. It is published twice a year; in the spring before the SAS Users Group International (SUGI) conference and again in the fall before the SESUG conference.

If you would like to help out by advertising or writing articles for this newsletter, please contact the editor, Randy Finch, by calling 256-386-2197 or E-mailing rcfinch@tva.gov.

Ian Whitlock

RETAINING VALUES IN A SAS DATA STEP

Let's take look at when and how values are retained in the SAS DATA step. Beginners are often confused when their counter becomes missing. All mature SAS programmers know that the RETAIN statement (or use of the SUM statement) provides the answer, but relatively few know all the rules of when values are set to missing, so let's explore them. Most programmers believe that a RETAIN step will prevent a variable from becoming missing without explicit assignment or reading of data, but that belief is not true.

Oh, you think you already know the rules. Well then, stop to take this test before proceeding.

Here is a DATA step with some missing code. The comments spell out the rules restricting what kind of code you can write.

```
data _null_ ;
 retain x 0 ;
 put x= ;
 /* Some DATA step code.
  If X is assigned a value, then
   that value cannot be missing.
  If X appears in any function (or
   routine) call, then X should
   not be set to missing as a
   consequence.
  If X is read by an INPUT
   statement, then that value
   should never be missing.
  If X is a variable in any SAS
   data set mentioned, then it
   should never have a missing
   value in the set.
 */
run ;
```

Find code to place in front of the step and code to replace the comment to produce the following log.

```
x=0
x=.
x=5
x=.
x=7
x=.
```

Ok, so you got it. Now, can you get a second solution that does not use any of the SAS words in your first solution?

While we have the experts tied up let's get on with the business at hand. First of all RETAIN doesn't mean *retain*. Values can change. Usually one reads that with RETAIN, values are not set to missing at the top of the implied loop of the DATA step. While true, the statement is misleading.

There are three kinds of named variables in DATA step:

1. Variables that come from SAS data sets.
2. Variables to which SAS assigns values other than missing (These variables are often referred to as automatic variables. Examples include _N_, END=, LAST. VAR, etc.).
3. All other named variables.

The RETAIN statement is significant only for the third category of variable, although there will be no complaints when applied in the other cases. For the third category of variable, it is true that SAS will not set these variables to missing when explicitly mentioned in a RETAIN statement. Usually the variable of interest does fall into the third category, and that is precisely why the misleading statement about the RETAIN statement does not cause more trouble than it does.

As far as I know, there are no variables in the second category that are ever set to missing unless they also fall in the first cate-

(Continued on page 10)

"First of all RETAIN doesn't mean retain. Values can change. Usually one reads that with RETAIN, values are not set to missing at the top of the implied loop of the DATA step. While true the statement is misleading."

Kirk Paul Lafler

Kirk Paul Lafler is a SAS Alliance Partner® and SAS Certified Professional® with over 25 years of SAS software experience. He provides hands-on SAS training around the world. Kirk has written over one hundred articles for professional journals and SAS User Group proceedings. His popular SAS Tips column appears regularly in the SANDS and SESUG Newsletters. His expertise includes application design and development, training, and programming using base-SAS, SQL, ODS, SAS/FSP, SAS/AF, SCL, FRAME, and SAS/EIS software.

"Because date and time values are stored and represented differently than we may normally be familiar with, the SAS System provides a variety of date and time formats and informats."

DATE AND TIME TIPS

Tip: ***Storing a date as a numeric value.***

In the SAS System, a numeric variable containing a date value is stored as the number of days from the fixed date value of 01/01/1960 (January 1, 1960). The SAS date value for January 1, 1960 is represented as 0 (zero). A date earlier than this is represented as a negative number and a date later than this is represented as a positive number.

Tip: ***Storing a time as a numeric value.***

In the SAS System, a numeric variable containing a time value is stored as the number of seconds since midnight. The SAS time value for midnight is represented as 0 (zero). The SAS time value for 23:59:59.9 (or 11:59 p.m.) is 86399.9.

Tip: ***Representing dates and time values with formats and informats.***

Because date and time values are stored and represented differently than we may normally be familiar with, the SAS System provides a variety of date and time formats and informats. Without date and time formats and informats, handling date and time values would be somewhat difficult and awkward. Let's see how a date/time formats is used.

A date format tells the SAS System how to display or write a date. Displaying a date value with a date format is accomplished with a FORMAT statement. Suppose you wanted to display the date value 01/01/1960 using the month name, day, and year as follows: January 1, 1960. The following construct could be specified using a FORMAT statement:

```
FORMAT DATE WORDDATE18.;
```

Tip: ***Applying a date format with a FORMAT statement.***

A FORMAT statement can be used in either a DATA or PROC step. When specified in a DATA step, a format is applied to the variable and stored as part of the variable's attributes in the data set. When specified in a PROC step, the format is applied to the variable only for the duration of executed step.

In the first example, a numeric variable of 15614 (or 10/01/2002) is stored in data set INFO as variable DATE. The value of DATE is displayed or written in the form of MM/DD/YYYY by specifying a FORMAT statement in a PROC step as follows:

```
PROC PRINT DATA=INFO;
 FORMAT DATE MMDDYY10.;
 VAR DATE;
RUN;

Result
10/01/2002
```

In the next example, a numeric variable containing a value of 15614 (or 10/01/2002) is displayed or written in the form of MM/DD/YYYY by specifying a FORMAT statement in a DATA step as follows:

```
DATA INFO;
 DATE=MDY(10,01,2002);
 FORMAT DATE MMDDYY10.;
RUN;

Result
10/01/2002
```

Tip: ***Applying a date format with an ATTRIB statement.***

An ATTRIB statement can be used in either a DATA or PROC step. When specified in a DATA step, a format is applied to the variable and stored as part of the variable's attributes in the data set. When specified in a PROC step, the format is applied to the variable only for the duration of executed step.

In the first example, a numeric variable containing a value of 15614 (or 10/01/2002) is stored in data set INFO as the variable DATE. The formatted date value is displayed or written in the form MM/DD/YYYY by specifying an ATTRIB statement in a PROC step as follows:

```
PROC PRINT DATA=INFO;
 ATTRIB DATE MMDDYY10.;
```

(Continued on page 11)

Kirk Paul Lafler

and

Charles Edwin Shipp

NO PHOTO AVAILABLE

BUILDING DRILL-DOWN SAS APPLICATIONS

"As a general rule, the best type of user interface design for transaction-based applications is a drill-down user interface opposed to a character-based one. It's referred to as drill-down because a user drills down through the data, layer by layer, until the desired information is found."

Introduction

In a world where business decisions must be made quickly, an application must be able to deliver information in a blink of an eye. Trends, facts, and figures must be translated into meaningful information so its meaning can get inside the heads of people. Decision makers are not satisfied with looking at rows and rows of numbers in an attempt to unmask the hidden meaning of data. Data must be provided visually as in a chart or graph so the picture itself tells the story without a word of explanation.

Data visualization techniques and technologies bring new levels of understanding to your data. The SAS Output Delivery System *(ODS)* is a marvelous tool used for formatting output generated by SAS procedures and DATA steps. ODS enables effective application design strategies to be implemented while providing flexibility and ease of use. Combined with products such as SAS/GRAPH software, new levels of interactivity bring your data to life with drill-down approaches. End users start out with summary information and can dive into the details at the click of their mouse button using a graphical user interface. This article describes the process of building a simple drill-down application with the SAS System. All you need is base-SAS and SAS/GRAPH software.

Drill-down Applications

As a general rule, the best type of user interface design for transaction-based applications is a drill-down user interface opposed to a character-based one. It's referred to as drill-down because a user drills down through the data, layer by layer, until the desired information is found.

Two significant design objectives exist for drill-down user interfaces. The first objective is to permit users to get the information they need quickly in order to make good business decisions. The second objective is to display only that data that is necessary to the user at a time.

The key to building successful drill-down applications requires systems analysts and system designers to understand what users are trying to achieve with the data. These individuals must recognize the tasks users engage in while trying to access the desired information. These tasks are then translated into a series of selection criteria that users should be able to select from.

Drill-down with Layered Charts

As the amount of data available on the Internet grows exponentially, systems analysts and system designers are faced with huge challenges. First, how should meaningful information be extracted from data? Second, what is the best way to stay current as information changes?

A popular method of visually presenting data while bringing out significant trends and outliers is accomplished with charts. Charts frequently display data as pie charts, vertical and horizontal bar charts, geographical maps, as well as a variety of other visual displays. Besides offering visual trends of data, charts provide convenience and offer a standard way of performing data analysis through drill-down navigation.

Drill-down approaches are extremely flexible, allowing any chart to act as a graphical front-end interface to additional graphical or textual information. Data is frequently displayed by layering one chart on another providing an effective point-and-click user interface. At the top layer data is presented in a highly summarized fashion. As the user clicks on a designated area of the chart, data is displayed in significantly more detail. The user navigates down the various layers of data with one or more clicks of the mouse or a combination of cursor and keyboard commands to see more and more details. It's simple and, above all else, elegant.

Building a Drill-down Application

To simplify the process of building a graphical drill-down application in the SAS System, five easy steps are presented.

1. Create a data set containing the location of the HTML link variable.

(Continued on page 8)

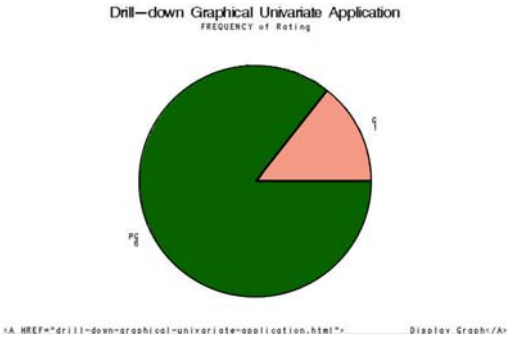
Kirk Paul Lafler
and
Charles Edwin Shipp

(Continued from page 6)

2. Create HTML path with BODY and optional files.
3. Create graph using HTML= option and link variable.
4. Create detail list drill-down pages.
5. Use Web browser to navigate through resulting application.

Example Drill-down Application

In the following drill-down application, a pie chart is used to display summary information on G-rated and PG-rated movies. A horizontal or vertical bar chart could also be used as the graphical user interface. To display detailed information on a movie category, a user would only need to click on the desired piece of the pie chart that they had interest in, as shown in the illustration below. Control would then be passed, via hyperlinks, to the underlying detail output created in step #4 above.



Example Code

The SAS coding steps used to in build a drill-down application under the Windows® platform is presented below.

```
*STEP 1 - Define data set and link
          variable;
data movie_ratings_g_pg;
  set odslib.movies;
  length urllink $30.;
  if upcase(rating) = "G" then
    urllink="href=g-ratings-body.
             html";
  else if upcase(rating) = "PG" then
    urllink="href=pg-ratings-body.
             html";
run;

*STEP 2 - Create HTML path and BODY
          file;
ods html path='c:\sas app'
         body='drill-down-graphical-
               application.html';

*STEP 3 - Create graph using
          HTML= option;
goptions device=gif hsize=7in
         vsize=4in;
proc gchart data=movie_ratings_g_pg;
  title1 "Drill-down Graphical
          Application";
  pie rating / html=urllink;
  where upcase(rating) in ('G',
                           'PG');
run;
quit;

*STEP 4 - Create G-rated drill-down
          list;
ods HTML path='c:\sas app'
         body='g-ratings-body.html';
proc univariate data =
                movie_ratings_g_pg ;
 title1 'Creating a Drill-down
         Application';
 footnote1 '<A HREF="drill-down-
         graphical-application.html">
         Display Graph</A>';
  where upcase(rating)="G";
run ;
ods HTML close;

*STEP 5 - Create PG-rated drill-down
          list;
ods HTML path='c:\sas app'
         body='pg-ratings-body.html';
proc univariate data =
                movie_ratings_g_pg ;
 title1 'Creating a Drill-down
         Application';
 footnote1 '<A HREF="drill-down-
         graphical-application.html">
         Display Graph</A>';
  where upcase(rating)="PG";
run ;
ods HTML close;
ods Listing;
```

Conclusion

With the growing popularity of the Internet, the Output Delivery System (ODS) is extremely useful in deploying selected pieces of output on the Web, Intranet, or Extranet. ODS makes the development of drill-down SAS applications a snap. Decision makers will appreciate the ability to see a visual representation of the data in a custom graphical user interface.

Acknowledgments

I would like to thank Randy Finch for accepting this paper, and the SESUG Leadership for their support of SAS users in the Southeast region of the United States.

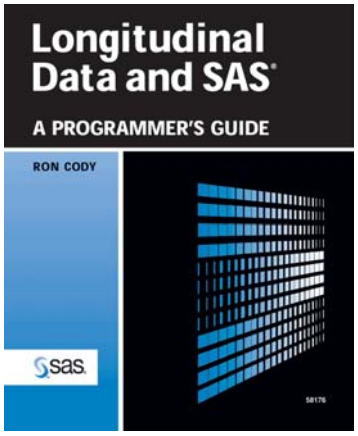
Kirk Paul Lafler is a SAS Alliance Partner® and SAS Certified Professional® with over 25 years of SAS software experience. He provides hands-on SAS training around the world. Kirk has written over one hundred articles for professional journals and SAS User Group proceedings. His popular SAS Tips column appears regularly in the SANDS and SESUG Newsletters. His expertise includes application design and development, training, and programming using base-SAS, SQL, ODS, SAS/FSP, SAS/AF, SCL, FRAME, and SAS/EIS software.

Charles Edwin Shipp is a Senior Programmer and Consultant with over 25 years experience working with the SAS System. He has authored numerous articles and co-authored the popular Books by Users (BBU) book, Quick Results with SAS/GRAPH Software. His expertise includes application design and development, training, and programming using base-SAS, JMP, SQL, ODS, SAS/FSP, SAS/AF, SCL, FRAME, and SAS/GRAPH software. He has also been involved with web design, content creation, and development.

REVIEW OF RON CODY'S "LONGITUDINAL DATA AND SAS: A PROGRAMMER'S GUIDE"

S. David Riba

If you use SAS software, you probably manage data -- read it, write it, analyze it. Data has many forms, and SAS provides many different ways of dealing with data. Beginning SAS classes teach the rudiments of working with data. However, most SAS users only scratch the surface of SAS capabilities to work with different types of data.



Longitudinal Data and SAS, by Ron Cody, is a comprehensive look at the techniques to deal with longitudinal data -- data that spans multiple observations. Ron's book looks at the problems encountered when working with longitudinal data, or in restructuring data into longitudinal data, and then examines techniques to solve each problem in detail. Some are very simple techniques, some are very novel techniques. Each is a useful addition to the basic knowledge that every SAS user should possess.

Longitudinal Data and SAS is a very detailed book, with many examples to illustrate each idea and technique. It is an ideal book for the Beginning SAS user. The discussions on fundamental data handling topics, and the insights into how SAS processes instructions like the LAG function, FIRST. and LAST., and the use of SAS procedures to deal with this type of data, are all highly useful and informative. More advanced SAS users will find much of the book's content is a refresher to knowledge they should already possess. However, every so often the user will encounter a little gem that will make this book a worthwhile addition to their SAS library, too. In general, I found Longitudinal Data and SAS to be an informative and thorough examination of the subject matter.

S. David Riba is CEO of JADE Tech, Inc., a SAS Quality Partner who specializes entirely in applications development, consulting, and training in the SAS System. Dave has presented papers and assisted in various capacities at SUGI, PharmaSUG, and regional conferences.

Dave is a SAS Certified Professional and an unrepentant SAS bigot. His major areas of interest are efficient programming techniques and applications development using the SAS System.

Dave is the founder of the Florida Gulf Coast SAS Users Group. He chartered and served as Co-Chair of the first SouthEast SAS Users Group conference, SESUG '93. Recently, he served as Co-Chair for SSU 2001, the first-ever joint conference between two regional SAS user groups, SESUG and SCSUG.

Ian Whitlock

Dr. Ian Whitlock is a senior systems analyst at Westat working on survey research problems. He has been programming largely in SAS since 1981 using an interactive environment to develop programs for execution in a batch environment. He is active on SAS-L, and has presented many different papers at national, regional, and local SAS User Group meetings.

"In a merge (with BY statement), SAS may set a variable Q to missing at the beginning of each BY-group when the MERGE statement is executed."

(Continued from page 2)

gory. It is generally a bad idea to ever let an automatic variable also come from a SAS data set, because the results are either misleading or disastrous. The only exception is in solving SAS puzzles.

Now the first category of variable gets quite tricky. These variables will be set to missing once before the step begins unless they are explicitly retained with a value in a RETAIN statement. But these variables can also be set to missing on the MERGE or SET statement. I leave it to you to decide the issue for other forms of SAS input.

In a merge (with BY statement), SAS may set a variable Q to missing at the beginning of each BY-group when the MERGE statement is executed. This happens, if Q is in one of the data sets in the MERGE statement and no record is read for the BY-group from the data set that contains Q. Or to put it another way, variables participating in a merge will be set to missing at the beginning of a BY-group when they are not actually read for that BY-group. Let's look at an example.

```
data w1 ;
  do key = 1 to 3 ;
    output ;
  end ;
run ;
data w2 ;
  q = 7 ;
  do key = 1, 3 ;
    output ;
  end ;
run ;
data w3 ;
  merge w1 w2 ;
  by key ;
  put key= q= ;
run ;
```

Q will be missing when KEY = 2. This is probably what you expected unless you remembered that values from SAS data sets are retained. The preceding paragraph spells out an exception. From these facts it is easy to get an answer to the test question given at the beginning by constructing appropriate data sets to merge. Stop here and try it to test your understanding of how to apply this information about a merge.

Now consider a SET statement with multiple SAS data sets in it. Here, whether or not a BY statement is present, is irrelevant to our discussion. Now SAS will set a variable Q to missing at the SET statement, whenever the iteration starts reading from a data set that does not contain Q, i.e. whenever SAS switches data sets in one SET statement and Q is not in the new data set. Using the same data as above this can be illustrated with the following code.

```
data w3 ;
  set w2 w1 ; /* note reversal of
                 order */
  put _n_= q= ;
run ;
```

Here Q will be missing when _N_ >= 3. What if we assign Q = 99 before the SET statement? Q will still be missing for _N_ = 3, but now the last two messages show Q is 99. Hence Q was set to missing once when SAS switched from W2 to W1.

Now you are in a position to give a second answer to the test question which makes use of neither the MERGE nor the BY statement by simply using a sequence of one-record data sets in the SET statement where alternate sets are missing the variable X. If you think carefully about these rules, and remember that in a merge, records are read only once, then your understanding of how a SAS merge works should be strengthened. I have seen some otherwise very professional SAS programmers caught by failure to fully understand a SAS merge.

(Continued from page 4)

```
 VAR DATE;
RUN;

Result
10/01/2002
```

In the next example, a numeric variable containing a value of 15614 (or 10/01/2002) is displayed or written in the form of MM/DD/YYYY by specifying an ATTRIB statement in a DATA step as follows:

```
DATA INFO;
 DATE=MDY(10,01,2002);
 ATTRIB DATE MMDDYY10.;
RUN;

Result
10/01/2002
```

Kirk Paul Lafler

SouthEast SAS
Users Group

Index to Advertisers

We're on the Web!
www.sesug.org

SouthEast SAS Users Group

SESUG EXECUTIVE COUNCIL MEMBERS	TELEPHONE	E-MAIL
Gregory Barnes Nelson (1999 co-chair)	919-465-0322 x351	greg.barnesnelson@statprobe.com
Dan Bruns, Vice President (1997 co-chair)	423-751-6430	debruns@tva.gov
Deborah Buck (2001 co-chair)	281-256-1619	dbbuck@compuserve.com
Philip J. d'Almada, Secretary (2000 co-chair)	404-639-6120	pdalmada@cdc.gov
Frank C. Dilorio (1994 & 1996 co-chair)	919-942-2028	fcd1@mindspring.com
Randy C. Finch, Treasurer, Editor (1998 co-chair)	256-386-2197	rcfinch@tva.gov
Maribeth H. Johnson (2000 co-chair)	706-721-3785	maribeth@stat.mcg.edu
F. Joseph Kelley, President (1998 co-chair)	706-542-5359	jkelley@arches.uga.edu
Andrew T. Kuligowski (1997 co-chair)	727-738-3000 x5203	kuligoat@tvratings.com
David Maddox (2002 co-chair)	205-977-0415	david.maddox@bellsouth.com
Heidi Markovitz (2002 co-chair)	305-365-0439	HeidiM@Simply-Systems.com
George Matthews (1999 co-chair)	706-542-5359	gmatthew@arches.uga.edu
Steve Noga (2003 co-chair)	919-408-8000 x235	snoga@rhoworld.com
S. David Riba (1993 & 2001 co-chair)	727-726-6099	dave@jadetek.com
Gary E. Schlegelmilch (2003 co-chair)	703-670-0451	geschlegel@juno.com

A Rap Ode to SAS

Randy C. Finch

SAS is a tool that's no man's fool.
It's good in school and as a business tool.
From the executive cool to the secretary's pool
it'll work like a mule without feeding it gruel.
So don't be cruel to the ones you rule.
Add a little fuel to their holiday Yule.
Get them the tool that's devoid of ghouls.
SAS is THE tool, SAS is a JEWEL!

The SESUG Informant
POB 8345
Clearwater, FL 33758