Paper 267-2013

# Getting an Overview of SAS® Data in Three Steps

Yu Fu, Oklahoma State University; Shirmeen Virji, Oklahoma State University; Miriam McGaugh, Oklahoma State University; Goutam Chakraborty, Oklahoma State University

## ABSTRACT

For SAS programmers, one of the most important steps before manipulating the dataset for further analysis is to get an overview of it. In order to get an idea of the dataset, normally three areas are looked into: variable names, statistical description, and relationship of one dataset with other datasets within a library. The macro that we have generated writes out the names of all the variables present in a file of a particular library, gives descriptive statistics of all the variables that are classified as numeric, and draws a diagram to show the relationships among the datasets. All three steps are performed by running just one code.

## INTRODUCTION

We have all been in a situation where we are given a very big library full of files and don't know where to begin our analysis. Or worse, if a newly hired employee has been given disks full of datasets, but he has no idea what variables are common in the files. What about the datasets that have to be merged with the existing dataset, but the types of the variables are different and it will take a long time to go into every file and check for particular details? While working SAS on daily basis, many users face these problems and the only option available to them to answer these questions is to spend more and more of their precious time to process this mundane information. We have created this macro to resolve all these problems in a three-step macro program. The three steps that our macro program consists of are variables, statistics, and relationships.

## STEP-BY-STEP

Firstly, macro program produces a table of column properties that contains the number of variables, name of the dataset of which they belong to, variable name, type, length of the variable, starting point, format structure (if any), and label.

Secondly, macro program produces analysis of the variables in which it prints out statistics of all the numeric variables present in the dataset. The resulting columns are number of observations, mean, standard deviation, and minimum value, and maximum value of the variable.

Lastly, macro, by using the power of GraphicViz, produces a relationship diagram where the relationships among datasets are shown graphically. If any dataset has a variable that has the same name and type that another dataset in the library also has, macro will draw that out on the report.

## USES

Uses of this macro are not limited to any particular industry. Any person who works with SAS datasets will need to know the variables and relationships at some point. This macro becomes very useful when the user is either not familiar with the dataset or it has been a while since the dataset has been used and, thus, the contents have been forgotten.

## WHAT'S IN IT?

The CONTENTS Statement in BASE SAS is used to generate the information related to variables of the data set. Statistical description is created by the MEAN Statement. We use an open source software called "Graphviz" to generate the relationship diagram. CALL SYSTEM Routine is used in the macro to interact with the external Graphviz application. Our final report produces the combined result from all three steps: contents, mean, and graphics and can be produced onto different types of outputs (pdf, rtf, html) by using ODS DOCUMENT. In this paper, we explain how to use these methods to generate the overview of the dataset step by step.

## METHOD

### EXTRACTING VARIABLE INFORMATION:

Before we begin to generate any information, we have to gather all the data files into one SAS library because in this paper we focus on generating the overview of all data sets in the same directory.

```
%local path fileref rc did dnum dmem memname didc ext;
%let dslist=;
%let path = %sysfunc(pathname(&lib));
%let rc=%sysfunc(filename(fileref,&path));
%let did=%sysfunc(dopen(&fileref));
%let dnum=%sysfunc(dnum(&did));

%do dmem=1 %to &dnum;
    %let memname=%sysfunc(dread(&did,&dmem));
    %if %upcase(%scan(&memname,-1,.)) = &ext %then
    %do;
          %let dsn = %scan(&memname,1,.);
          %let dslist = &dslist &dsn;
    %end;
%end;
%let didc=%sysfunc(dclose(&did));
%let rc=%sysfunc(filename(fileref));
```

With this macro code, all data set names that meet the defined file extension are put into the macro variable dslist. Next step is to use the CONTENTS Statement to extract the variable information from each data set. Normally, the CONTENTS Statement is used to write out the contents of one or more SAS data sets, but to include variable information in our output, we use the following code:

```
ods output variables= var;
ods listing close;
proc contents data=dsn;
run;
```

Here, we use the ODS OUTPUT statement to define the contents output that we need. After running the above code, a temporary dataset, var, is generated that contains the variables information of dataset, dsn.

### ODS OUTPUT STATEMENT:

The variables information of data set is defined to output to the temporary data set, var. The output produced by the CONTENTS Statement combines several parts. The code to specify data sets to which CONTENTS data is directed is shown below:

```
ods output attributes=atr
    variables=var
     enginehost=eng
    indexes=ind
    integrityconstraints = ic
    sortedby= sb;
```

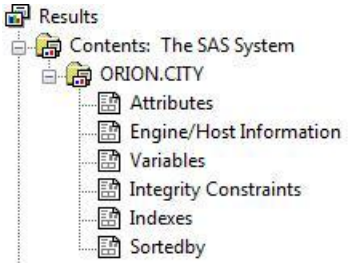



**Figure 1. CONTENTS Output**

### ODS LISTING <ACTION>:

The option of action here is used to close the listing destination that suppresses the output to the list temporarily.

### CREATING STATISTICAL DESCRIPTION:

The following proc step generates the statistical description of the SAS data set and gives a statistical overview of numeric variables in the data set. By running this code, the user will get mean, median, standard deviation, and minimum and maximum value of the numeric variables.

```
proc means data=dsn maxdec=2;
run;
```

## GENERATING A RELATIONSHIP DIAGRAM:

This part is the most exciting section of the paper. In this section, the macro program interacts with an external graph visualization application to create the relationship image and then produces the image onto our output.

We use open source visualization software called “Graphviz” to generate graphical relationships. Graphviz represents structural information in the form of diagrams of abstract graphs especially in networking, bioinformatics, software engineering, database, and web designing. The code below shows how to create a relationship diagram by using Graphviz.

```
digraph G{
    P1->P2[label=Friend];P1->P3[label=Son];
    P1->P4[label=Father];P1->P5[label=Husband];
}
```

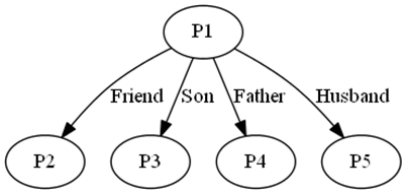



**Figure 2. Relationship Diagram**

In order to draw a relationship diagram on Graphviz, a program has to be written in Base SAS to create dot file (dot file is one of the executable files for Graphviz).

```
%let header= digraph G {;
%let tail = };
%let command=;
%let id=%sysfunc(open(rmap));
%let NObs=%sysfunc(attrn(&id,NOBS));
%let rc = %sysfunc(close(&id));
data _null_;
    set Rmap;
    call symputx("fds"||left(_n_),membera,'L');
    call symputx("sds"||left(_n_),memberb,'L');
    call symputx("variable"||left(_n_),variable,'L');
run;
%do m=1 %to &num;
%let command=;
%let dsn=%scan(&dslist,&m);
%do n=1 %to &NObs;
    %let fmem = %sysfunc(tranwrd(&&fds&n,%upcase(&lib..),));
    %let smem = %sysfunc(tranwrd(&&sds&n,%upcase(&lib..),));
          %if %upcase(&dsn)= &fmem %then
                %do;
                      %let str = &fmem.->&smem.[label=&&variable&n]%str(;);
                      %let command = &command&str;
                      %put ERROR: &command;
                %end;
          %else %if %upcase(&dsn)= &smem %then
                %do;
                      %let str = &smem.->&fmem.[label=&&variable&n]%str(;);
                      %let command = &command&str;
                      %put ERROR: &command;
                %end;
          %else %put ERROR: &dsn &&fds&n &&sds&n &&variable&n;
%end;

data _null;
    file "c:\temp\relation.dot";
    put "&header&command&tail";
run;

data _null_;
X 'cd G:\Graphviz\bin';
call system("G:\Graphviz\bin\dot -Tjpg c:\temp\relation.dot -o c:\temp\&dsn..jpg");
run;
```

The above piece of code shows how to generate the relationship diagram in three steps.

The first step is to create the Graphviz description text language for each dataset in the library. Previously generated relationships among the datasets are stored in Rmap dataset. Here, each dataset name is compared with each relationship observation in Rmap, and then only those relationships are picked up that have the same dataset name. The selected relationship will be translated into the Graphicviz code and later all the codes will be concatenated into a macro variable by the following of code.

```
%let str = &fmem.->&smem.[label=&&variable&n]%str(;);
%let command = &command&str;
```

The next step is to save the completed Graphviz program into a dot file. To perform this, we use data step and output the result into a dot file without creating a new data set.

```
data _null;
    file "c:\temp\relation.dot";
    put "&header&command&tail";
run;
```

## FILE STATEMENT:

The dot file is specified here for the current out file for the PUT Statement. This dot file will be interpreted by Graphviz to create the image file in a later step.

## PUT STATEMENT:

The PUT Statement is used to concatenate three macro variables into the completed Graphviz program and to write that to the external file defined in the FILE Statement above.

The third step is to call the Graphviz software to run the program that was generated in the step 2.

```
data _null_;
X 'cd G:\Graphviz\bin';
call system("G:\Graphviz\bin\dot -Tjpg c:\temp\relation.dot -o c:\temp\&dsn..jpg");
run;
```

## X STATEMENT:

A Windows command is called by X Statement to ensure that the path navigator is positioned at Graphviz's running directory.

## CALL SYSTEM STATEMENT:

A Graphviz command is submitted by the CALL SYSTEM Statement to generate the image in the defined folder with the specified type.

## TJPG OPTION:

Tjpg option is used to define the output type of the image. You can also use other options to generate the different images, for example using Tpng option to create png image.

## O OPTION:

The generated image will be stored in a temp folder in the C drive.

## PRINTING THE OVERVIEW INTO FILE:

In the above steps, we introduced how to create the overview for a data set. Now, we will print them into pdf, rtf, or html file. If this code is printed directly, the structure of the output will be unreadable. Therefore, we use the ODS DOCUMENT Statement to solve this issue.

```
ods document name=dictemp.&dsn;
    /*print variable information*/
    proc print data = dictemp.&dsn;
    run;
```

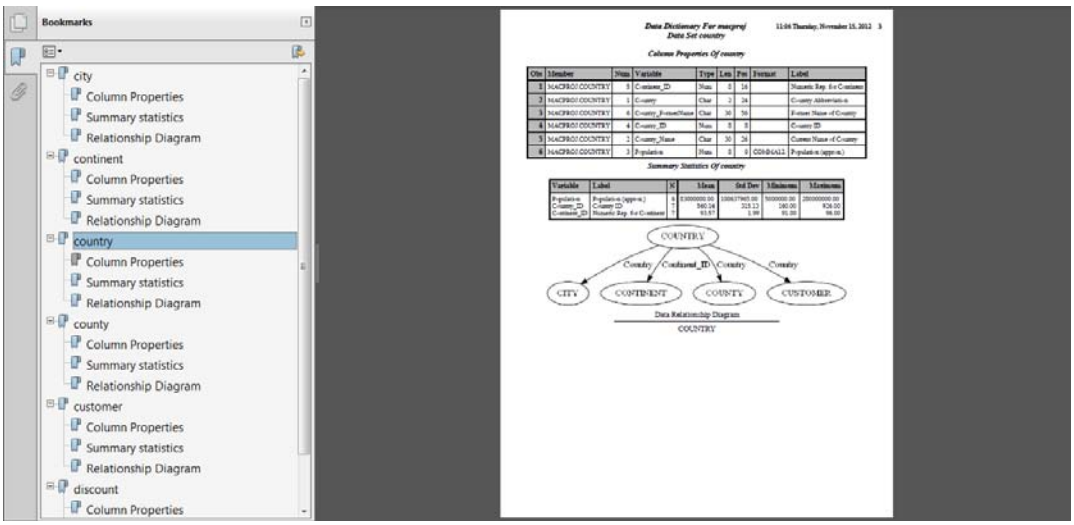
```
    /*print statistical description*/
    proc means data=&sourcelib..&dsn maxdec=2;
    run;

/*print relationship diagram*/
proc print data=image noobs label style=[preimage="C:\temp\&dsn..jpg" frame=void];
    var image /
    style(data) =[font_size=12pt
    cellwidth=9 cm just=center
    cellheight=0.9 cm vjust=middle]
    style(header)=[background=white
    font_size=10pt
    font_weight=medium
    just=center
    cellheight=0.9 cm vjust=middle];
    label image = "Data Relationship Diagram";
    run;

ods document close;

/*print ods document to file:pdf,rtf,html*/
    ods &filetype file="&filename..&filetype" startpage=&startpage;
    ods escapechar='^';
    title "Data Dictionary For &sourcelib";
    proc document name=dictemp.&dsn;
       replay;
    run;
ods &filetype close;
```

## ODS DOCUMENT STATEMENT:

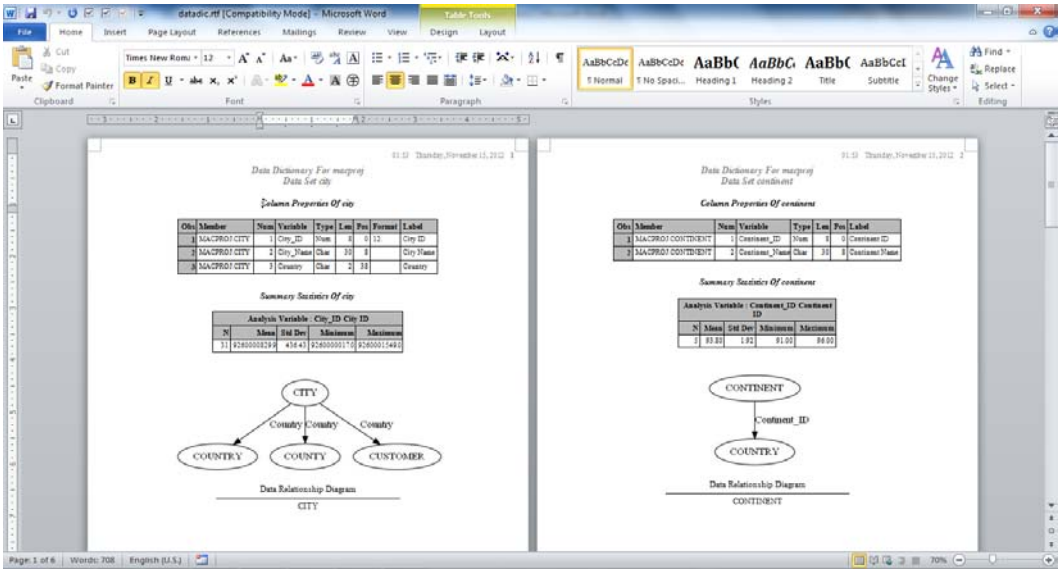
The generated document can reprint ODS output without rerunning the procedures. In the following DOCUMENT procedure, the results are replayed and outputs are generated into a designated file. Note: We have performed several modifications on the structure of the output in this document, but that code has not been included in this paper as we assume that it is outside of the scope of this paper.

## ODS PDF/RTF/HTML STATEMENT:

We used a macro variable filetype to represent the output file type. The value of the variable filetype can be pdf, rtf, or html. The program prints the output to different files according to the value specified in the macro filetype.




**Display 1. Sample Output in pdf**




**Display 2. Sample Output in rtf**

## LIMITATIONS

In this SAS macro, an overview can only be created for sas7bdat file. Other types of dataset files will be ignored. Most of the work to support the other type of data files will be on the dataset reading. We are working to add support for other types of data files especially for the raw data file.

## CONCLUSION

In this paper, we introduced a macro program to generate the information for a SAS data set including variables information, statistical description, and relationship diagram. This program can help programmers to build the data dictionary of data set quickly and to be familiar with the data set in a short time.

## REFERENCES

Bessler,LeRoy. 2005. “Getting Started with, and Getting the Most out of, SAS® ODS PDF: No Mastery of PROC TEMPLATE Required. ” SAS Conference Proceedings: Technical Solutions, Phuse, Heidelberg, Germany

“DOT Language Document [Internet].” 2012[cited 2012 Nov 16]. Available at http://www.graphviz.org/content/dot-language

## CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Name: Yu Fu
Enterprise: Oklahoma State University
Address: Department of MSIS, Oklahoma State University
City, State ZIP: Stillwater, OK – 74075
Work Phone: 405-309-9356
E-mail: yu.fu@okstate.edu

Name: Shirmeen Virji
Enterprise: Oklahoma State University
Address: Department of MSIS, Oklahoma State University
City, State ZIP: Stillwater, OK – 74074
Work Phone: 901-857-1400
Fax: 901-471-4185
E-mail: shirmeen.virji@okstate.edu

Name: Dr. Goutam Chakraborty
Enterprise: Oklahoma State University
Address: Department of Marketing, Oklahoma State University
City, State ZIP: Stillwater, OK - 74074
Work Phone: (405)744-7644
E-mail: goutam.chakraborty@okstate.edu