PhUSE 2012

Paper TS03

A Macro to replace PROC REPORT!?

Katja Glass, Bayer Pharma AG, Berlin, Germany

ABSTRACT

Some companies have macros for everything. But is that really required? Our company even has a macro to replace PROC REPORT! This SAS procedure is so powerful and flexible, so for what would a standard macro be required? We have this macro since the 90s and yes, it fits our company requirements much better than a PROC REPORT. The following paper will present reasons why a macro replacing a standard procedure like PROC REPORT can enhance our programming environment. It will also provide tips and links for PROC REPORT listing output.

INTRODUCTION

Tables and Listings are typically created with a PROC REPORT. This procedure provides the most important options to create any kind of table layout. Nevertheless, the same commands have to be used over and over again for each and every table. Especially the definitions of the column widths can become very time-consuming as this is required for nearly every variable.

This time consuming and repeating code-creation can be done much faster by a macro where the user only has to use simple parameters to define the layout. The report creation for experienced users may be programmed quite fast. But even if a few minutes can be saved, a macro like this is worth the effort as each programmer creates and updates many tables each day. Furthermore whenever the data changes, the output has to be revisited to ensure no new unwanted splits occur, e.g. when a variable which had been empty before is now filled or longer words appear. Aggregating this time consuming tasks, the overall time saving for an output creation with a macro can be drastically noticeable. The study level validation effort decreases as well when having a validated standard which usage would require also less and structured source.

This paper will present the options of a table creation macro called %datalist which replaces the need for PROC REPORT and will demonstrate possible enhancements which such a macro could bring. The main objective with the %datalist macro is higher programming efficiency, lower validation effort and better standardization. As part of a larger macro system, the %datalist macro provides all required functionalities to print and style outputs in a user friendly way.

EXAMPLE TABLE OUTPUT

The following example demonstrates a simple table creation with the %datalist macro in contrast to the traditional PROC REPORT output. For the %datalist macro obviously less source code is required and the layout is also more appropriate as deliverable.

```
%datalist(data = sashelp.class
                 (WHERE=(name < 'I')),
          by   = sex name,
          var  = age height weight);
```

```
PROC REPORT DATA=sashelp.class
       (WHERE=(name < 'I'))
       nowd headline spacing=2;
   COLUMN sex name age height weight;
   DEFINE sex / ORDER;
   DEFINE name / ORDER;
RUN;
```

SEX	NAME	AGE	HEIGHT	WEIGHT
F	Alice	13	56.5	84
	Barbara	13	65.3	98
	Carol	14	62.8	102.5
M	Alfred	14	69	112.5
	Henry	14	63.5	102.5

Sex	Name	Age	Height	Weight
F	Alice	13	56.5	84
	Barbara	13	65.3	98
	Carol	14	62.8	102.5
M	Alfred	14	69	112.5
	Henry	14	63.5	102.5

FUNCTIONALITY

A major advantage of %datalist in contrast to pure PROC REPORT is that many options are already build-in. The user does not have to create additional commands to create appropriate lines or handle table splits. Furthermore the handling can be controlled by additional layout parameters. This chapter characterizes the most important out-of-the-box features and parameter functionalities.

Lines. Some forced layout options are implemented in the %datalist macro. One of this is the line-drawing above and below the column headers and above the footnote.

To achieve this in the PROC REPORT is not that easy. The column statement can be modified to use a line as spanning header for the column titles [1]. To get the additional two lines an additional title and footnote line statement can be used [2]. Alternatively COMPUTE statements can be used.

Different programmers can come up with various solutions and finally will have verifying layouts. With %datalist, the layout is always the same and the users do not need to maintain additional sources.

```
Table: Line Example
_____________________________________
SEX   NAME    AGE  HEIGHT  WEIGHT
_____________________________________
 F   Alice    13    56.5      84
     Barbara  13    65.3      98
     Carol    14    62.8    102.5
 M   Alfred   14      69    112.5
     Henry    14    63.5    102.5
_____________________________________
```

Variable Selection. The most important layout option is the definition of the printout variables. With respect to the requested layout a variable can be used to split pages and use it as a sub-header (PAGE), to sort and group the layout (BY) or just for display (VAR). Of course pure sorting variables without display are possible as well. (ORDER).

Several DEFINEs, BREAKs and a COMPUTE AFTER are required in PROC REPORT to get the same layout. The macro call is much easier to read and interpret.

Continued. The example demonstrates the implemented "(Continued)" functionality of the macro. All pages but the last contains the flag to allow the reviewer to acknowledge the continuing table.

To create this flag in PROC REPORT is a complex task, which can be done with conditional footnotes [3] [4].

```
page = sex
by   = name
var  = age height weight

SEX: F
_____________________________________
 NAME    AGE  HEIGHT  WEIGHT
_____________________________________
Alice    13    56.5      84
Barbara  13    65.3      98
Carol    14    62.8    102.5
_____________________________________

(Continued)

SEX: M
_____________________________________
 NAME    AGE  HEIGHT  WEIGHT
_____________________________________
Alfred   14      69    112.5
Henry    14    63.5    102.5
_____________________________________
```

Basic Options. There are many basic options for a PROC REPORT which can be used through parameters with the %datalist macro. The macro provides an easy way to specify free lines, the spacing between columns, the alignment of column headers, printout of labels when no label is specified, split characters for the column headers and furthermore a printout repetition of BY variables. SAS features like WHERE conditions for the input data together with the "_ALL_" and column syntax are supported as well by the macro. Spanning headers can be used for one level, currently no further column header subgrouping is implemented.

Split in Width. Sometimes tables needs to be created which do not fit the line size. Instead of requiring a separate table output, splits are handled within the %datalist macro. The output shows on the bottom line whether the displayed page continues and which part this is (/#.>>). It would be very tough to program this on demand for every table with PROC REPORT whenever required.

REGION	SUBSIDIARY	PRODUCT	Number of Stores	Total Sales
Africa	Cairo	Boot	20	$4,846
		Men's Casual	25	$360,209
		Men's Dress	5	$4,051
		Sandal	9	$10,532
		Slipper	9	$13,732
		Sport Shoe	3	$2,259
		Women's Casual	14	$328,474
		Women's Dress	3	$14,095

/1.>>

REGION	SUBSIDIARY	PRODUCT	Total Inventory	Total Returns
Africa	Cairo	Boot	$18,965	$229
		Men's Casual	$1,063,251	$9,424
		Men's Dress	$45,962	$97
		Sandal	$50,430	$598
		Slipper	$54,117	$1,216
		Sport Shoe	$20,815	$44
		Women's Casual	$940,851	$10,124
		Women's Dress	$51,145	$745

/2.

Column Widths. Manual processing to investigate the required width for different columns is very often required for PROC REPORT. Especially when the character default length of $200 is used, the width within a PROC REPORT needs to be specified. Another problem comes up with new data. When tables are designed with draft data the final data could require larger columns. After the initial output creation many tables have to be adapted to correct column widths.

This task is done by the %datalist macro. The macro investigates the maximum length of the cell content and uses this for the width calculation. Within our company there are three further parameters to influence the width. Instead of requesting the length of the single column, the user sets MAXLEN (integer), BYLEN (integer) and OPTIMAL (no/yes).

With MAXLEN and BYLEN the maximum width for variables are defined. The column header or the cell content is split in several lines whenever larger than the given length. When the column could be smaller as the maximum width of the cell content requires less space, then only this width is used. The additional OPTIMAL option uses additional routines to calculate internally column widths according predefined rules.

Logical Splits. The macro provides an option to calculate optimal splits in height with respect to the content. It is important for the reviewer that groups that belong together are presented on the same page whenever possible. The %datalist macro has implemented a TOGETHER parameter where a variable is specified which repeating content has to be kept on the same page.

So when four observations would fit on a page, but we want to keep the three female and two male observations together, then the table split is after three females and the male group is shifted to the next page.

TOGETHER =

SEX	NAME	AGE	HEIGHT	WEIGHT
F	Alice	13	56.5	84
	Barbara	13	65.3	98
	Carol	14	62.8	102.5
M	Alfred	14	69	112.5

(Continued)

SEX	NAME	AGE	HEIGHT	WEIGHT
M	Henry	14	63.5	102.5

TOGETHER = sex

SEX	NAME	AGE	HEIGHT	WEIGHT
F	Alice	13	56.5	84
	Barbara	13	65.3	98
	Carol	14	62.8	102.5

(Continued)

SEX	NAME	AGE	HEIGHT	WEIGHT
M	Alfred	14	69	112.5
	Henry	14	63.5	102.5

To produce such a logical order automatically for PROC REPORT is quite complex [5]. Even if this logical grouping should be done for some tables only to support the reviewer, the manual programming and validation effort without macro support would be very high.

Transpose. Sometimes a table output needs to be transposed. A simple transpose option is already implemented, so the user would not have to manually perform a transposition and handle somehow the spanning column header label. This is all done by the macro which saves finally a lot of work and validation effort. The following example displays a simple transposition.

		Johannesburg			Nairobi		
REGION	PRODUCT	Total Sales	Total Inventory	Total Returns	Total Sales	Total Inventory	Total Returns
Africa	Boot	$8,365	$33,011	$483	$16,282	$66,017	$844
	Men's Dress	.	.	.	$8,587	$20,877	$363
	Sandal	$17,337	$63,003	$809	$16,289	$47,406	$1,175
	Slipper	$39,452	$130,025	$1,565	$34,955	$87,438	$1,320
	Sport Shoe	$5,172	$29,368	$139	$2,202	$11,328	$174
	Women's Dress	$42,682	$120,127	$966	$28,515	$62,740	$678

Several Tables. It happens quite often that the same table layout should be used several times and just the content changes. To create laboratory tables per parameter for example, users could create a study macro or repeat the same source several times and differ only the selection and title. The %datalist macro provides an option to create separate tables per variable characteristic. The title is updated according to the content and a pre-selection is done per table. There is even an option available, that columns are only displayed if they are not empty. This is quite useful as some laboratory tests have units and some not, so with that option the unit column is not displayed when always empty for one of the tests.

HISTORY

The %datalist macro was implemented in the 90s and consists mainly of data steps. The final output is done with PUT statements to have the best control about everything. Beginning in 2005 additional user requirements have been collected and led to many updates. One major update in 2005 was the shift from the Listing output destination to RTF. For the RTF printout a PROC REPORT is used which supports most of the Listing functionalities.

The shift from the Listing to the RTF destination was quite easy for the users, as they did not need to learn the new ODS features which are used automatically through the macro. This is the most comfortable solution as the layout requirements for the RTF output coming from Medical Writing is quite high and do need many additional PROC REPORT source lines and even some post processing. [6]

Many user change requests resulted in updates which made the macro more and more powerful and user friendly. As examples, the initial macro call structure was not as intuitive as it could be and has been enhanced a lot. Furthermore the title and footnote handling had been done within the macros whereas by now the SAS title and footnote handling is used instead of providing additional parameters.

Even though we try to keep the number of parameters low, the enhanced functionality requires parameter driven options. So we started with 23 parameters and have now 29. Additionally to the parameters different syntaxes like "DESCENDING" for sort variables are supported. For a typical table the user would only need to set about five parameters.

Initially the macro has only been used at one site by a few people, but soon it became famous and was used by all statistical programmers and statisticians in several locations. In 2006 the last remaining location switched to use the MoSTO macro package containing the %datalist macro. This made the interchange much easier, as our deliverable layouts and programming techniques are harmonized. After the company grows the MoSTO macro system was rolled out in additional sites as well, as it has been shown that the macro set containing the %datalist macro was more flexible and powerful than the available alternative. Due to harmonization one solution should be maintained. More than 95% of our table and listing outputs are created with this printout macro by now. The only exceptions are previously programmed tables that are still in use.

VALIDATION AND DOCUMENTATION

When the macro is used on study level, the validation effort for the users is quite low as the macro is already validated. To validate a global standard macro is quite complex and time consuming as it will be used in various situations. The validation should test next to the typical cases especially non-standard scenarios and error cases which could occur. Also the documentation takes a lot of effort to provide enough user support to answer most questions which could appear efficiently.

Of course bugs have been found from time to time and are corrected as soon as possible. It is not possible to check all possible scenarios even with an excessive macro validation. Initially a single quote as label indicating the minute unit had not been supported for example.

For the %datalist macro itself we do have by now 131 test cases including 23 error and 108 functional checks. For each update all existing tests are repeated and if needed are updated or newly created. For this the robustness of the macro increases as more and more data situations are checked.

Furthermore the manual with 57 pages including 30 pages of examples and one quick overview page support users a lot. By now we only get new user requests and rarely bug fix reports and no "how to use" questions. Even though the parameters are very intuitive there would be many user questions arising without a detailed documentation which would need to be handled otherwise.

LIMITATION

Of course not all PROC REPORT options are available through the macro itself. A macro can reduce the complexity on behalf of functionality. The current %datalist implementation does not provide any SUMMARY functionality as the PROC REPORT does. Furthermore special COMPUTE processing is not intended, as an example a separating line with text is not possible. A detailed column width specification for the single columns is also not implemented.

The intention for the %datalist macro is to support the output creation of already summarized data or listings with less complexity and for this flexibility. The possible layouts are fixed and harmonized within our function. The source code is much easier to maintain and validate on study level then having single PROC REPORTS and supports also exchanges of responsible analysts.

When new layout requirements according shifting requirements arise, they are implemented as needed. For this the limitations can be reduced with the cost of increased complexity, meaning new parameters or special parameter syntaxes. As an example a "DESCENDING" syntax for sorting purposes had been implemented and spanning headers are supported, which has not been intended at the beginning. So whatever limits exist, these can be reduced as required, but be aware that the macro complexity will increase.

SUMMARY

The PROC REPORT creation can become very tricky when special layouts are requested, the tables are too long or width. It could make sense to provide a macro to replace PROC REPORT, so it is possible to handle special layouts and data situations automatically instead of manual adaption in a standardized way. Even though per table output one or two minutes can only be saved when having a macro, the overall benefit is available for the creation and the validation. Statistical Programmers create quite a lot of outputs per day and the data behind also changes often during output development.

The effort for development, validation, documentation and training needs to be invested at the beginning. Further maintenance to collect and update new user requirements will support the user acceptance and up-to-date standards and layout requirements. Even an output destination change is easily possible without massive user training. To answer the initial question: yes, it makes sense to create a macro to replace PROC REPORT! Within our company the macro has proven its value and it will still save a lot of programming resources and gives the users time to concentrate on the content as they can rely on the provided layout.

REFERENCES

[1] Chris Moriak [2001], „Two Quick Tips for PROC REPORT: Making a Line above the Headers and the PAGE Option“, Paper NESUG 01
http://www.nesug.org/proceedings/nesug01/cc/cc4012.pdf

[2] Daphne Ewing, Ray Pass [1998], „So Now You're Using PROC REPORT. Is it Pretty and Automated?“, Paper NESUG 98
http://www.nesug.org/proceedings/nesug98/btut/p106.pdf

[3] Yumi Sembongi [2010], „PROC REPORT COMPUTE Block and Conditional Footnote“, Paper PharmaSUG 2010
http://www.lexjansen.com/pharmasug/2010/po/po06.pdf

[4] Angelina Cecilia Casas [2002], „Data Dependent Footnotes using PROC REPORT“, Paper PharmaSUG 2002
http://www.lexjansen.com/pharmasug/2002/proceed/techtech/tt20.pdf

[5] Michael Knoessl [2007], "A SAS Macro for Breaking Pages", Paper PhUSE 2007
http://www.lexjansen.com/phuse/2007/cs/cs07.pdf

[6] Katja Glass [2007], "Advanced RTF layout with SAS", Paper PhUSE 2007
http://www.lexjansen.com/phuse/2007/cs/cs08.pdf

RECOMMENDED READING

[1] Keith Cranford [2008], " Learning PROC REPORT by Comparison", Paper SAS Global Forum 2008
http://www2.sas.com/proceedings/forum2008/170-2008.pdf

[2] Malachy J. Foley [2001], "PROC REPORT: How To Get Started", Paper SESUG 2001
http://analytics.ncsu.edu/sesug/2001/P-819.pdf

[3] David Trenery, Hoechst Marion Roussel, Denham, „Jazzing up Your Reports - Some Tricks with PROC REPORT", Paper SUGI 24
http://www2.sas.com/proceedings/sugi24/Coders/p080-24.pdf

CONTACT INFORMATION

Author: Katja Glaß
Institution: Bayer Pharma AG
Address: Sellerstr. 32, 13353 Berlin, Germany
E-mail: Katja.Glass@Bayer.com