%RTFTable – A Powerful SAS® Tool to Produce Rich Text Format Tables
Eric Qi, Merck Research Labs, Merck & Co., Inc., West Point, PA
Liping Zhang, Merck Research Labs, Merck & Co., Inc., West Point, PA

ABSTRACT
This paper describes an approach to generating RTF tables through a SAS macro, %RTFTable. This is a powerful SAS tool to convert SAS data sets into highly customized tables in rich text (RTF) or Microsoft Word (DOC) format. The macro gives users full control over the table appearance: e.g., defining table titles, footnotes, and headers; merging cells in the table header; customizing the data format for each column; adjusting column width and alignment; controlling text appearance (bold, italic, underline or plain text) in the table body; and more. This paper provides an overview of the unique macro design, describes the macro syntax and parameters, highlights some of the nice features and gives an example of a macro call. %RTFTable can be used on any platform to produce RTF tables. The macro is easy to use and rich in functionality.

INTRODUCTION
In the pharmaceutical industry, it is quite desirable for rich text format (RTF) or Word tables to be generated automatically by SAS programs and later inserted directly into a Word document, such as a final clinical study report. For this reason, many tools/macros or systems have been developed over the past several years.

The following is a brief review of some approaches that have been taken. Three different approaches have been taken:

Converting SAS Output (Text/ASCII Files) to Word Tables
This approach uses ASCII or text files as input. %SASWORD1 (Cunningham, 1998) is a macro which can convert existing text tables to Word tables by using Dynamic Data Exchange (DDE), Dynamic Link Libraries (DLLs), and Word’s Convert-Text-Table feature in Microsoft Window Environment. %SAS2WORD (Zhou, 2001) is a macro which can convert an ASCII file to Word by utilizing the RTF tags as a bridge to link Word via DDE.

A limitation of %SAS2WORD is that it can only handle monospace fonts. In contrast, both monospace and proportional fonts can be used in %SASWORD1. However, %SASWORD1 has quite rigorous requirements for the input text file and sometimes requires a post-macro-execution editing of the Word table.

Converting SAS Data Sets to RTF/Word Tables
This approach starts with a SAS dataset. %PRINT (Wehr, 1996) and %RTF (Peszek et al., 1998 and 1999) macros can create RTF/Word tables directly from SAS data sets. Both are very powerful, flexible and able to create complex RTF/Word tables. However, both of them need to be executed multiple times within a DATA _NULL_ step, which may involve a lot of programming effort and could be difficult for inexperienced SAS users.

Unlike the tools/macros mentioned above, the Output Generating System (OGS) (Zhou, 2002) system is a collection of 10 macros to produce RTF/Word tables. It links the 10 macros together to perform separate but inter-related tasks based on the layout and attributes of a table page. It can provide quite a few nice features and can also be used on many platforms and operating systems. However, if a table has a relative large number of columns with many layers of spanning headers, the column definition (%Zdefine, %Zcs and %Zsend) can be cumbersome and difficult to visualize.

SAS Output Delivery System (ODS) (SAS Institute, since version 7) can provide an option for users to send their output to RTF table format. Users can use the default table template to quickly produce a simple format RTF table. However, if users want to generate more complex tables, which is normally the case, they have to use PROC TEMPLATE to define their own customized table templates, which can be tedious and difficult.

This paper introduces an extension of the second approach to convert a SAS data set into a highly customized rich text format (RTF) or Word table through %RTFTable. It is a front-end macro with %RTF macro embedded in it. %RTFTable is easy to use and rich in functionality and can be used in any platform to produce RTF tables. We start with an overview of the unique macro design, then introduce the macro syntax and parameters, then highlight some of the nice features of the macro. Finally, we give an example of a macro call to illustrate that %RTFTable macro is not only relatively easy to use but also powerful and flexible enough to generate highly customized RTF/Word tables.

THE MACRO DESIGN
In 1995, a %RTF macro was developed at Merck & Company to generate RTF tables. The macro was published by Peszek et al. in 1998. It has been used since then with minimum upgrades. %RTF is very powerful and flexible. However, one drawback is that it has to be used in a DATA _NULL_ step. %RTF can generate the RTF tags for users, but the users have to write their own PUT statements to generate the RTF table. Therefore, it may involve a lot of programming effort and may not be suitable for inexperienced programmers.

In order to make the macro more user friendly, %RTFTable was designed to be a front-end macro, which writes DATA _NULL_ statements for the users and uses %RTF macro to build a link between SAS and Microsoft Word. The only thing that users need to do is to call %RTFTable and provide the parameter values.

The %RTFTable macro tries to use one single macro call to convert a SAS data set into a Word table in a format fully controllable by user. The user can use the macro easily to define table titles, footnotes, and headers; merge cells in the table header; customize the data format for each column; adjust column width and alignment; control text appearance (bold, italic, underline or plain text) in the table body and more. In terms of table overall appearance, the macro can control font type, font size, table orientation, table width, space between text and cell border, and page length. In addition, the macro allows the user to break pages through data set information.

The macro was designed to control table appearance in the following four sections: table title, header, body, and footnotes.

TABLE TITLES AND FOOTNOTES
One of the biggest challenges to design a table macro is to try to guess how many titles and footers the user will need. The
TABLE HEADERS
Another challenge for a table macro is how to control table headers. The method to control table headers should be easy to use and powerful to control table appearance. We chose a graphic way to describe the header information. We used the HeadRows parameter to let the macro know how many headers the user is going to define, and the %let HEADERi = statement outside the macro call to define each row of header content itself, where i is the order of the row in the header.

In each header, a vertical line symbol '|' is used to separate columns. If the user does not want to display a vertical line between the columns, '!' should be used instead of '|'. If the user wants to merge cells, '^' should be used to replace '|'. If all the columns in every header definition were lined up, the user will have very good idea about what the table header will look like. In the following example, there are two headers, defined as:

```
%let header1= |Laboratory AE | Total No | AE Event ^ |;
%let header2= | Term | Patients | n ! % |;
```

The generated headers will look like:

```
Laboratory AE    Total No AE Event
Term             Patients n %
```

Furthermore, the macro can easily handle multiple level, complex headers. The headers showed in the following table contain 5 levels and 3 places with cell merging. However, the %RTFTable macro code used to generate the header looks straight forward (attached below).

```%let header1= |Group A**'s ^^ | |
%let header2= | Vaccine A^^ | |
%let header3= | Component | Postvaccination ^ | |
%let header4= | Measurement 1 | Measurement 2 | |
%let header5= | (Assay) | N(95% CI) (95% CI) |
```

```
```

Header definition code:

```
%LET HEADER1=%nstr(|Group A's ^^ | |);
%LET HEADER2=%nstr( | Vaccine A^^ | |);
%LET HEADER3=%nstr( | Component | Postvaccination ^ | |);
%LET HEADER4=%nstr( | Measurement 1 | Measurement 2 | |);
%LET HEADER5=%nstr( | (Assay) | N(95% CI) (95% CI) | |);
```

Parameter description:

The parameters can be classified into 3 groups.

**GROUP 1 - INPUT AND OUTPUT**
- **ds** | Input SAS data set name
- **DocLibr** | Output file path, e.g., C:\RTFTABLE
- **DocFile** | Output file name, e.g., table1.rtf or table1.doc.

**GROUP 2 - NUMBER OF TITLES, HEADERS AND FOOTNOTES**
(Note: Users have the control of defining titles, headers and footnotes by using %LET statement outside the macro. Refer to example 1 for details.)
- **TitleRow** | Number of title rows
GROUP 3 - TABLE RELATED PARAMETERS

Group 3 can be divided further into three sub-groups:

a. Column related
   - Var: Column variable list separated by space
   - Pmt: Format for each column variable separated by space
   - Width: Relative column width separated by space
   - Align: Alignment for each column separated by space
   - Decimal: Decimal alignment position for each column
   - Borders: Specify vertical borders (shown or not shown) for each column, user can merge cells by choosing not shown the vertical column border(s)

b. Row related
   - NoRepeat: List of at least one or more variables separated by spaces. The values of these variables will only be printed once when they 1st appear
   - SkipBy: Skip one row /line before the value of this variable changes
   - StyleVar: This parameter applies specific style formats to certain rows in your table. You can provide up to 3 types of styles with this parameter separated by a vertical line "|": (1) font formatting (2) horizontal line formatting (3) vertical line formatting.

c. Table (as a whole) related
   - Font: Font used in the table, default is 1 for Times New Roman font.
   - FontSize: Font size for the table and title
   - PageBy: PageBy variable list separated by a space
   - PageByF: Formats for PageBy variable(s) separated by a space
   - PortLand: Page orientation, default is portrait.
   - Pagesize: Table length (page size), measure by the number of lines printed
   - Twdth: Table width measured by inches*100.
   - Append: Append several tables into one file, default -1 means no appending
   - Colspc: Space between column edge and the 1st character printed in the cell (in inch), default is 0.15

HIGHLIGHT SOME OF THE USEFUL FEATURES

%RTFTable is very rich in its functionality. The following is to just highlight a few of them:

PAGE CONTROL

The macro has three different approaches to control how and when to break to a new page. Even better, the control information is carried in the input SAS data set, so the user doesn’t need to hard code this information in the macro call. Rather, it is necessary only to assign flags in the input SAS data set.

Among these three approaches, two of them can present the page breakdown information in the tables. One is to print the information on the left top corner outside the table but below table titles (See Table 1 as an example). The second way is to print the information in the table body, as one of the table columns (See Table 2 as an example).

<table>
<thead>
<tr>
<th>Table 1. Print PageBy variable in the header by PageByF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Week</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 2. Print PageBy variable in the table body (only show 1st page, 2nd page not show)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Week</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

The third approach can easily break tables into several pages without presenting this information (See Table 3 as an example).

<table>
<thead>
<tr>
<th>Table 3. Do not Print PageBy variable (only show 1st page, 2nd page not show)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Week</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

SE = Standard error.
specifies both PageBy and PageByF parameters, the macro will break into new pages when the PageBy variable changes values and print PageBy variable values on the top left corner of the page (like table 1). If the user specifies only the PageBy parameter and also lists the PageBy variable in the VAR parameter, the macro will break into new pages when the PageBy variable changing values and print the PageBy variable values in the table body (like table 2). If the user specifies only the PageBy parameter, the macro will only break into new pages when the PageBy variable changes values and will not present the PageBy information at all in the table (like table 3).

DECMIAL ALIGNMENT

Another useful feature of this macro is that it can make decimals line up within each column in the table. Furthermore, it can control the alignment position in each column. The difference between this alignment and the traditional left, right, and center control the alignment position in the column.

Suppose we divide a cell (column) into 64 shares. By default, the decimal alignment is 2/64th (1/32th) of the cell width on the right of the cell center (this is about the center of a cell). You can only use an integer number between 0 and 64 for this parameter. If you specify a number less than 32, say 6, then the alignment places is 6/64th of the cell width on the right of the cell center. If you specify a number greater than 32, say 36, then the alignment is 4/64th (1/16th) of the cell width on the left of the cell center (note 4=36-32).

The %RTFTable macro designed the Decimal parameter to allow the user to specify decimal alignment information for each column in the table. This parameter needs to be used together with the parameters Var and Align. The parameter Var is used to specified all variables needs to go into the table, while the parameter Align is used to specify the alignment for each variable (column). The Align parameter has 4 options, L for left, R for right, C for center, and D for decimal. If for any of the columns Align = D is specified, the parameter Decimal must also be specified for all the columns.

The following Table 4 is an example of using Decimal to control decimal alignment. Note that column 2 was moved to the left by specifying a larger number for the column (54), and column 4 was moved to the right by specifying a smaller number for the column (14). Part of the macro calling code is attached below.

Table 4. Using DECIMAL parameter to control decimal alignment.

<table>
<thead>
<tr>
<th>Statistics</th>
<th>Drug 1</th>
<th>Drug 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10 mg</td>
<td>30 mg</td>
</tr>
<tr>
<td>Group 1</td>
<td>N</td>
<td>99</td>
</tr>
<tr>
<td></td>
<td>Mean</td>
<td>26.5</td>
</tr>
<tr>
<td></td>
<td>SD</td>
<td>2.9</td>
</tr>
<tr>
<td></td>
<td>Median</td>
<td>26.0</td>
</tr>
<tr>
<td></td>
<td>Range</td>
<td>22 to 35</td>
</tr>
</tbody>
</table>

Part of the macro call:

```sas
%RTFTable(Ds=test,  
   Var-statname T2 T3 T6 ,  
   Align=1  d  d  d ,  
   Decimal=2  54  x  14 ,  
   ...  
   );
```

TABLE ROW STYLE CONTROL

To let certain rows in a table stand out from the rest of the rows in the table, you can print the rows within a table with different style than the rest of the table by using StyleVar parameter. The StyleVar parameter contains the following information:

1) A variable name that is in the input SAS data set. The values of the variable can be bold, italic, underline, plain text, or blank. If the values of the variable are other than blank, the style of the rows in the table corresponding to these records will be printed in the style specified by this variable and the following two items.

2) Specification of a horizontal line style for these rows. You can select to print line(s) at the top, bottom, both top, and bottom or no line around these rows.

3) Vertical line style for these rows. You can select to print or not to print the rows with vertical line(s) between columns.

Table 5 below shows that the section header lines ‘Gender’, ‘Age (years)’ etc. have a specified style, i.e. bold text, with both top and bottom lines and no vertical lines printed. The ‘section headers’ in the following table really stand out to highlight the table into several sections.

Table 5. Using StyleVar to control section header.

<table>
<thead>
<tr>
<th>Gender</th>
<th>Drug A (N = 20)</th>
<th>Placebo (N = 20)</th>
<th>Total (N = 40)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female</td>
<td>20 (100%)</td>
<td>20 (100%)</td>
<td>40 (100%)</td>
</tr>
<tr>
<td>Age (years)</td>
<td>21.4</td>
<td>21.1</td>
<td>20.2</td>
</tr>
<tr>
<td>Mean</td>
<td>19-22</td>
<td>18-22</td>
<td>18-22</td>
</tr>
<tr>
<td>Range</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Furthermore, by extending our knowledge to RTF syntax, we can highlight any single cell in the table. For the above highlight cell “Asian”, you need to embedded the RTF code into you data set and the value of the variable will change from `Asian` to `\i Asian`.

SPECIAL SYMBOLS

It is an essential request in our environment to use special symbols, such as ‡, †,  ′, etc., in our formal reports. Instead of defining these special symbols each time when we use them, we developed a macro, named %RTFSYMBL (Zhang, et al., 2002). %RTFSYMBL is a companion macro with %RTFTTable developed at Merck. What the macro does is to define each special symbol as a global macro variable. If a user wants to add special symbols in the tables, the user needs to invoke %RTFSYMBL macro prior to %RTFTTable call, then choose macro variables defined in the %RTFSYMBL macro corresponding to the symbols you want to use.

Special symbols can be inserted anywhere in the tables, i.e. they can appear in the titles, the footers, the headers and even in the table body. What the user needs to do is to find out the macro variable name for the symbol interested and refer to it whenever the user wants the symbol appeared in the table.

The table below illustrates that special symbols are added in the titles, the 3rd column header and table body and footnote.

Table 6. Illustration of special symbol used in the %RTFTable
macro.

APPENDING TABLES

%RTFTable has an appending function which allows users to append two or more tables into one single Word file from several macro calls. This is especially useful when you don't want to create too many files for your output. This function is carried out by the Append parameter. The Append parameter has four options: -1 means for no appending, 0 for appending and the first macro call, 1 for appending and any middle calls, and 100 for appending and last call. This parameter must be used together with parameters DocLibr and DocFile. If you choose to append files, all the calls must assign same value for DocLibr and DocFile. Typically, users can use this feature in the following two different situations: (1) Appending tables during a SAS program developing process; or (2) Appending tables after a program is finished.

When you are in the programming developing process, you know in advance which tables will go to the same output file. You can assign each call with associated APPEND values. Another situation is that you have already finished all programs for, say a protocol, but you want to re-run all the programs and save the tables in a single file. The %RTFTable macro allows you to use 3 global macro variables outside the macro calls to re-define the values of parameters: Append, DocLibr and DocFile and re-direct the output tables into another location. The nice feature about this function is that you don’t need to modify your programs to accomplish this re-run. A limitation about this re-run is that all the calls must use same page orientation.

EXAMPLE OF A MACRO CALL AND ITS OUTPUT

The following example illustrates a sample of macro call and its output.

SAMPLE DATA SET:

<table>
<thead>
<tr>
<th>ROWLABEL</th>
<th>DRUG_A</th>
<th>DRUG_B</th>
<th>TOTAL</th>
<th>STYLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gender</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Female</td>
<td>27 (%)</td>
<td>13 (%)</td>
<td>40 (%)</td>
<td>b</td>
</tr>
<tr>
<td>Age (years)</td>
<td>20.4</td>
<td>21.1</td>
<td>20.7</td>
<td>b</td>
</tr>
<tr>
<td>Mean</td>
<td>1.7</td>
<td>1.8</td>
<td>1.8</td>
<td>b</td>
</tr>
<tr>
<td>Median</td>
<td>20.0</td>
<td>22.0</td>
<td>20.5</td>
<td>b</td>
</tr>
<tr>
<td>Range</td>
<td>17-23</td>
<td>18-23</td>
<td>17-23</td>
<td>b</td>
</tr>
<tr>
<td>Race/Ethnicity</td>
<td>0 (0%)</td>
<td>2 (15.4%)</td>
<td>2 (5.0%)</td>
<td>b</td>
</tr>
<tr>
<td>Asian</td>
<td>0 (0%)</td>
<td>2 (15.4%)</td>
<td>2 (5.0%)</td>
<td>b</td>
</tr>
<tr>
<td>Black</td>
<td>1 (3.7%)</td>
<td>0 (0%)</td>
<td>1 (2.5%)</td>
<td>b</td>
</tr>
<tr>
<td>Caucasian</td>
<td>22 (81.5%)</td>
<td>10 (76.9%)</td>
<td>32 (80.0%)</td>
<td>b</td>
</tr>
<tr>
<td>Hispanic</td>
<td>2 (7.4%)</td>
<td>0 (0%)</td>
<td>2 (5.0%)</td>
<td>b</td>
</tr>
<tr>
<td>Native American</td>
<td>0 (0%)</td>
<td>1 (7.7%)</td>
<td>1 (2.5%)</td>
<td>b</td>
</tr>
<tr>
<td>Other</td>
<td>2 (7.4%)</td>
<td>0 (0%)</td>
<td>2 (5.0%)</td>
<td>b</td>
</tr>
</tbody>
</table>

This example demonstrates that the macro is relatively easy to use and flexible enough to create a highly customized table.

CONCLUSION

%RTFTable represents an enhanced approach to convert a SAS data set into a highly customized rich text format (RTF) or a Word table. This front-end utility macro is easy to use and rich in functionality and can be used on any platform and operating system to produce RTF tables.
REFERENCES


TRADEMARKS
SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries

ACKNOWLEDGMENTS
We would like to thank Ms. Cindy Song for the discussions and supports over RTF issues. We would also like to thank Haoli Chai, since the macro parameter setting was inspired from his %Wtable macro. Finally, we thank Dr. Gregory Golm for reviewing the paper.

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

Eric Qi
BL3-2
Merck Research Labs
Merck Co., & Inc.
West Point, PA 19486
(484) 344-7796
(484) 344-2931(Fax)
eric_qi@merck.com

Liping Zhang
UN-A102
Merck Research Labs
Merck Co., & Inc.
West Point, PA 19486