Paper 043-30

Applying Microsoft Word Styles to ODS RTF Output

Lauren Haworth, Genentech, Inc., South San Francisco, CA

➢ ABSTRACT

With the Output Delivery System, PROC TEMPLATE can be used to create Rich Text Format (RTF) output with elaborate formatting to meet specific business needs. When the results are opened in Microsoft Word, the output has exactly the fonts, sizes, weights, and borders that were specified.

However, the output must be inserted into another Word document, this can all fall apart. An existing Word document uses Word styles to identify titles, headings, tables, and body text. When a SAS-generated table is pasted in, it is assigned the Word style “Normal”. As a result, most of the custom ODS formatting is lost, and the output does not display properly.

The secret to solving this problem is to insert RTF tags for Word styles into the ODS output. Then when the results are pasted into another document, the styles are pre-applied, so the Normal style is not needed. The output maintains its original formatting.

This paper shows how to embed the proper RTF tags to create this customized output by modifying the original SAS program. This paper is intended for users with some prior experience with the Output Delivery System, and is based on SAS version 8.2 or higher.

➢ **INTRODUCTION**

Here's some sample ODS RTF output, with custom formatting applied. This is what it looks like after opening the RTF document directly in Word. The output shows all of the desired fonts, sizes, and weights.

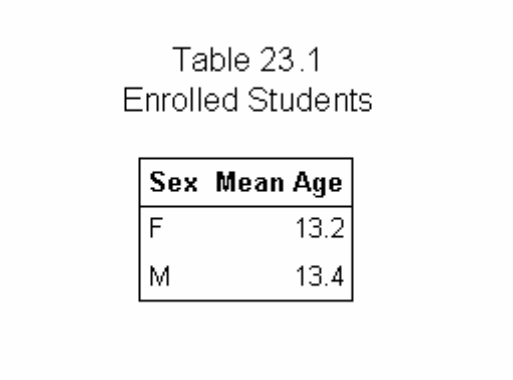
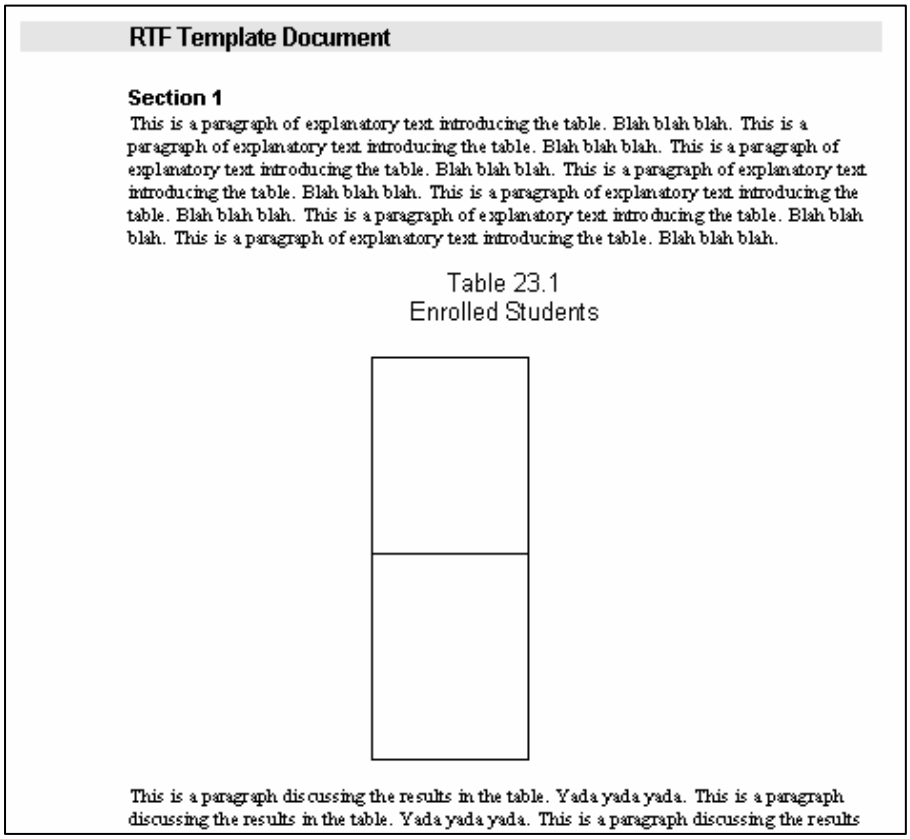


Table 23.1
Enrolled Students

Sex	Mean Age
F	13.2
M	13.4

However, if this same table is then copied from the original document and pasted into another Word document, most of the formatting is lost. Here's an example of the same output after it has been pasted into an existing Word document:



RTF Template Document

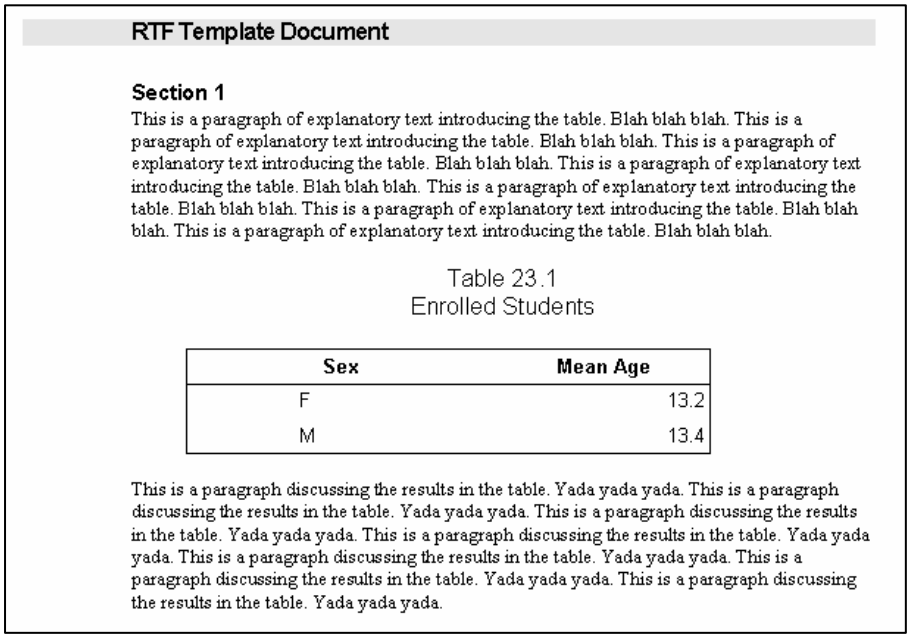
Section 1

This is a paragraph of explanatory text introducing the table. Blah blah blah. This is a paragraph of explanatory text introducing the table. Blah blah blah. This is a paragraph of explanatory text introducing the table. Blah blah blah. This is a paragraph of explanatory text introducing the table. Blah blah blah. This is a paragraph of explanatory text introducing the table. Blah blah blah. This is a paragraph of explanatory text introducing the table. Blah blah blah.

Table 23.1
Enrolled Students

This is a paragraph discussing the results in the table. Yada yada yada. This is a paragraph discussing the results in the table. Yada yada yada. This is a paragraph discussing the results

The text has disappeared! Actually, it's still there, but the paragraph spacing and indenting are so large that the text no longer fits into the table cells so it can't be seen. If the table columns are widened, the text reappears:



RTF Template Document

Section 1

This is a paragraph of explanatory text introducing the table. Blah blah blah. This is a paragraph of explanatory text introducing the table. Blah blah blah. This is a paragraph of explanatory text introducing the table. Blah blah blah. This is a paragraph of explanatory text introducing the table. Blah blah blah. This is a paragraph of explanatory text introducing the table. Blah blah blah. This is a paragraph of explanatory text introducing the table. Blah blah blah.

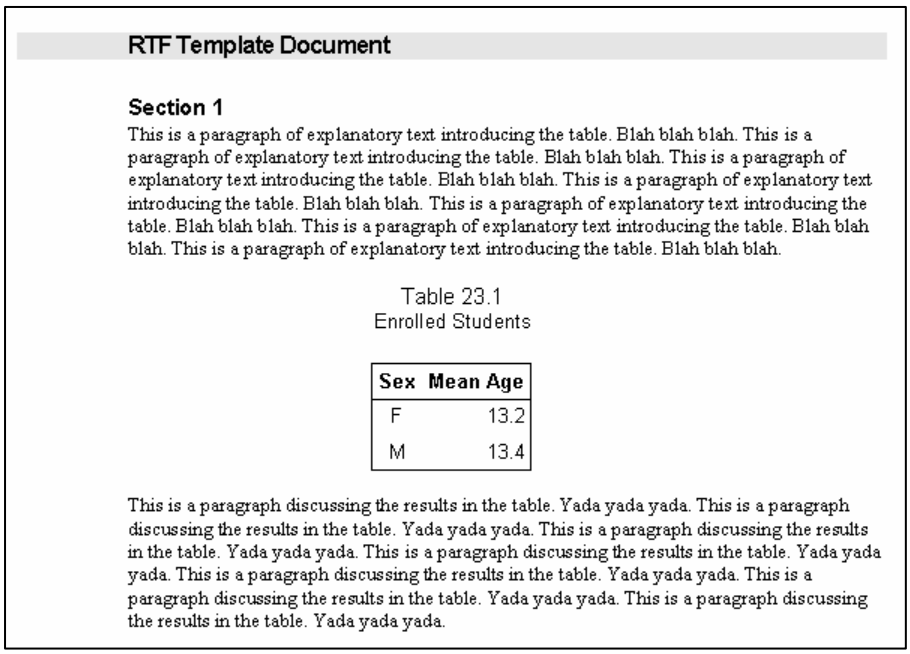
Table 23.1
Enrolled Students

Sex	Mean Age
F	13.2
M	13.4

This is a paragraph discussing the results in the table. Yada yada yada. This is a paragraph discussing the results in the table. Yada yada yada. This is a paragraph discussing the results in the table. Yada yada yada. This is a paragraph discussing the results in the table. Yada yada yada. This is a paragraph discussing the results in the table. Yada yada yada. This is a paragraph discussing the results in the table. Yada yada yada. This is a paragraph discussing the results in the table. Yada yada yada.

The reason the formatting of the table is lost is that the SAS output doesn't use Word styles. Everything gets pasted in as the "Normal" style, which means the formatting is replaced by Word's default formatting for the "Normal" style. On rare occasion, the output is close enough to the desired appearance that the output's appearance is acceptable. But most of the time, the output has to be manually reformatted in Word to restore the proper appearance.

To fix this, the solution is to embed Word style information into the ODS RTF output, so that it stays with the output when it is cut and pasted into another Word document. Here's what the inserted output looks like when there are embedded Word styles in the ODS RTF output:



RTF Template Document

Section 1

This is a paragraph of explanatory text introducing the table. Blah blah blah. This is a paragraph of explanatory text introducing the table. Blah blah blah. This is a paragraph of explanatory text introducing the table. Blah blah blah. This is a paragraph of explanatory text introducing the table. Blah blah blah. This is a paragraph of explanatory text introducing the table. Blah blah blah. This is a paragraph of explanatory text introducing the table. Blah blah blah.

Table 23.1
Enrolled Students

Sex	Mean Age
F	13.2
M	13.4

This is a paragraph discussing the results in the table. Yada yada yada. This is a paragraph discussing the results in the table. Yada yada yada. This is a paragraph discussing the results in the table. Yada yada yada. This is a paragraph discussing the results in the table. Yada yada yada. This is a paragraph discussing the results in the table. Yada yada yada. This is a paragraph discussing the results in the table. Yada yada yada. This is a paragraph discussing the results in the table. Yada yada yada.

This paper will demonstrate an approach to include this additional formatting information in ODS RTF output.

➢ THE RTF TAGS

In order to understand this technique, some background information is required on RTF and style tags. If an RTF file is viewed in a text editor like Notepad, it can be seen that the file is made up of the main text of the document, surrounded by a lot of cryptic codes in curly brackets. These codes are RTF tags, which control all aspects of the structure and appearance of

the RTF document. When an RTF document is opened in Word, these tags are used to determine how to display and format the document. The following text is an example of what an RTF document looks like when viewing the raw text in Notepad rather than opening it up in Word.

```
{\rtf1\ansi\ansicpg1252\uc1\deff0\stshfdbch0\stshfloch0\stshfhich0\stshfbi0\deflang1033\deflangfe1041{\fonttbl{\f0\froman\fcharset0\fprq2{\*\panose
02020603050405020304}Times New Roman;}{\f1\fswiss\fcharset0\fprq2{\*\panose 020b0604020202020204}Arial;}
{\f2\fmodern\fcharset0\fprq1{\*\panose 02070309020205020404}Courier New;}{\f3\froman\fcharset2\fprq2{\*\panose
05050102010706020507}Symbol;}{\f44\fswiss\fcharset0\fprq2{\*\panose 020b0604030504040204}Tahoma;}
{\f141\froman\fcharset0\fprq2{\*\panose 02020603060405020304}Times;}{\f155\froman\fcharset238\fprq2 Times New Roman CE;}{\f156\froman\fcharset204\fprq2
Times New Roman Cyr;}{\f158\froman\fcharset161\fprq2 Times New Roman Greek;}
{\f159\froman\fcharset162\fprq2 Times New Roman Tur;}{\f160\froman\fcharset177\fprq2 Times New Roman (Hebrew);}{\f161\froman\fcharset178\fprq2 Times New
Roman (Arabic);}{\f162\froman\fcharset186\fprq2 Times New Roman Baltic;}
{\f163\froman\fcharset163\fprq2 Times New Roman (Vietnamese);}{\f165\fswiss\fcharset238\fprq2 Arial CE;}{\f166\fswiss\fcharset204\fprq2 Arial
Cyr;}{\f168\fswiss\fcharset161\fprq2 Arial Greek;}{\f169\fswiss\fcharset162\fprq2 Arial Tur;}
{\f170\fswiss\fcharset177\fprq2 Arial (Hebrew);}{\f171\fswiss\fcharset178\fprq2 Arial (Arabic);}{\f172\fswiss\fcharset186\fprq2 Arial
Baltic;}{\f173\fswiss\fcharset163\fprq2 Arial (Vietnamese);}{\f175\fmodern\fcharset238\fprq1 Courier New CE;}
{\f176\fmodern\fcharset204\fprq1 Courier New Cyr;}{\f178\fmodern\fcharset161\fprq1 Courier New Greek;}{\f179\fmodern\fcharset162\fprq1 Courier New
Tur;}{\f180\fmodern\fcharset177\fprq1 Courier New (Hebrew);}
{\f181\fmodern\fcharset178\fprq1 Courier New (Arabic);}{\f182\fmodern\fcharset186\fprq1 Courier New Baltic;}{\f183\fmodern\fcharset163\fprq1 Courier New
```

To fully explain RTF tags would be well beyond the scope of this paper. However, to use this programming technique, it is only necessary to know about the style tags. These tags exist in two places. First, in the main body of the document, tags like \s2, \s3, \s4, etc. appear in front of the various places where text appears in the document. These tags indicate which styles to apply to what parts of the text. The \s2 tag might be calling for an 8 point Arial font, while the \s3 tag might call for a 10 point Times italic font.

To determine which styles to apply for each \s2 or \s3 tag, there's a section in the header of the RTF document that contains additional tags related to styles. Here there is a list of style numbers which links them to the name of the style that will be used by Word. For example, the text might look something like this:

```
Normal;}{\s2 TableNumber;}{\s3 TableTitle;}{\s4 TableHeader;}{\s5 TableData;}
```

This section of the header indicates that the document uses 5 styles. The Normal style applies to anything that doesn't have a tag indicating one of the other styles. Any part of the document that is prefaced with the \s2 tag will have the TableNumber style applied. Anything prefaced by \s3 will use the TableTitle style. And so on.

To embed the appropriate Word styles into an RTF document, first insert the appropriate \s2 or \s3 tags in front of the text in the various parts of the output. Then, add the appropriate tags to the header, to link the \s2 and \s3 tags to the appropriate style names. When the RTF document is opened in Word, the appropriate Word styles will be applied to the various parts of the text. So to get the appropriate Word styles, it is necessary to insert extra text (the style tags) into the RTF document.

IN-LINE FORMATTING

In-line formatting is a good way to embed extra text into SAS output. This technique uses a special character to indicate that the subsequent text is an in-line formatting command. The first step in using this technique is to pick the special character. This is done with the ODS ESCAPECHAR statement. It is important to pick a character that is not used in any other way in the SAS output. In this example, the caret "^" will be used. The syntax is as follows:

```
ods escapechar='^';
```

The second step in using this technique is to use an escape sequence to identify the RTF tags that need to be inserted into the ODS RTF output. An escape sequence consists of the escape character "^" followed by an in-line formatting command. In-line formatting commands do things like insert superscripts or subscripts, or apply formatting attributes such as color or font size. For this technique, an escape sequence is used that contains an in-line formatting command that passes through special text directly to the output destination, without any modification by the SAS system. This sequence is ^R, and it's followed by the text to be inserted. The "^" is the escape character, and the "R" identifies the subsequent text as raw text to be inserted into the final output. The full escape sequence to add the RTF tag \s2 to the ODS output would be:

```
^R'\s2 '
```

Notice that the text to be passed through is contained in single quotes, and it follows the escape sequence ^R. Also notice the space after the \s2 tag. These style tags must be followed by a space when they are inserted into the RTF output. This is an RTF syntax requirement.

FIGURING OUT WHICH WORD STYLES TO USE

Now that the appropriate techniques have been identified – RTF tags for the styles, and in-line formatting to insert the tags – it's time to write some SAS code.

The first step to creating ODS RTF output with embedded Word styles is to identify which styles need to embedded. These styles will be part of the main Word document; the one where the ODS output is to be inserted. For the example program in this paper, the assumption is that the Word document contains 4 styles that need to be used. A style called TableNumber is used to format the "Table 23.1"; a style called TableTitle will be used for the "Enrolled Students" title; a style called TableHeader will be used for the column headings in the table, and a style called TableData will be used for the cells inside the table. These style names are entirely arbitrary, and will vary depending on the setup of the main Word document where the output will be inserted.

➢ Applying RTF Tags for Word Styles to Titles

In the example program in this paper, special formatting is needed for the titles, column headings, and data cells in the output table. The goal is to apply particular Word styles to the various parts of the output. Starting at the top, the first task is to apply the style TableNumber to the first title, and TableTitle to the second title. Here is the technique for applying styles to the titles.

With Word styles, the key to inserting the RTF tags is to get them into the right part of the output. The \s2 tag needs to come right before the text that needs to be assigned the style associated with \s2. For the styles to be applied to the table titles, the way to do this is to add the escape sequence to the text in the title statement. These are the original title statements used for this output.

```
title "Table 23.1";
title2 "Enrolled Students";
```

These are the new title statements, after the escape sequences have been used to add the RTF tags:

```
title "^R'\s2 'Table 23.1";
title2 "^R'\s3 'Enrolled Students";
```

Notice that the escape sequence is added inside the quotes that enclose the title text, and that double quotes are used around the entire title, with single quotes being used around the text to be inserted. Also notice that two different style tags are used here. This allows for two different Word styles to be indicated: one for the first title, and one for the second title.

Applying RTF Tags for Word Styles to Column Headings

For the tags that need to go into the column headings of the output, the RTF tags can be added to the labels for each column, using the same approach as with the titles. In this case, the output comes from PROC REPORT. The labels for each column are included in the define statement for that column. Here is the original PROC REPORT code:

```
proc report data=sashelp.class nowd;
   column sex age;
   define sex / group "Sex";
   define age / analysis mean "Mean Age" format=5.1;
run;
```

Here is the same code again, with escape sequences used to identify a Word style that will be applied to the column headings.

```
proc report data=sashelp.class nowd;
   column sex age;
   define sex / group "^R'\s4 'Sex";
   define age / analysis mean "^R'\s4 'Mean Age" format=5.1;
run;
```

➢ Applying RTF Tags for Word Styles to Data Cells Inside a Table

Now tags to embed styles into the titles and the column headers have been added to the output. These two areas were easy, because the text in question is included directly in the SAS program. But for the text in the table cells, there isn't a place in the program that shows it. The table cell values are computed by PROC REPORT. There's no text in quotes where the required style tags can be added.

Instead, the text can be inserted using PROC TEMPLATE. By defining a custom ODS style, it is possible to specify that a particular escape sequence be applied prior to the text in every data cell. The full PROC TEMPLATE code is shown in Appendix A, but the important part of the code is as follows:

```
style data from data /
     pretext="^R'\s5 '";
```

This code calls for the text ^R'\s5 ' to be inserted in front of the text in every table cell, so that the style referenced by \s5 will be applied to each cell.

➢ Adding the Required Style References to the RTF File Header

Now the \s2, \s3, \s4, and \s5 tags have been added to the correct places in the output, and Word will know where to apply these styles. The last step is to name the styles that will be used in each place. To do this requires adding information to the document header.

ODS does not supply a way to add information to the RTF header, so the solution is to edit the RTF document after it is created by ODS. The first step is to create the string of text to be inserted into the header. This text can be stored in a macro variable as follows:

```
%let styles=%nrbquote({\s2 TableNumber;}{\s3 TableTitle;}{\s4 TableHeader;}{\s5
TableData;});
```

The second step is to insert the text into the right part of the document header. A data _null_ step is used to read the file in, edit the header, and write it back out again.

```
data _null_;
  infile 'c:\temp\sample2.rtf' lrecl=500;
  file 'c:\temp\sample3.rtf' lrecl=500;
  length temp $ 600;
  input ;
  if index(_infile_,"Normal;}")>0 then do;
     temp=tranwrd(_infile_,"Normal;}","Normal;}&styles");
     put temp;
  end;
  else put _infile_;
run;
```

The key part of this code is the if statement. This statement uses the index function to find the line of the output that contains the style information. In the original RTF document, the only style listed is Normal, so that is the text to search for. Once that line is found, the tranwrd function can be used to replace the original RTF code "Normal;}" with the revised code "Normal;}&styles". When the macro is resolved, the text "Normal;} ({\s2 TableNumber;}{\s3 TableTitle;}{\s4 TableHeader;}{\s5 TableData;} " is what will be inserted. Only this one line of the output file will be changed. All other lines are just read in and written back out as is.

➢ Inserting the Results into an Existing Word Document

The full program is shown in Appendix A. Once these few modifications are made, the program can be run just like any other. The difference now is that when the RTF document is opened in Word, the text will be linked to the appropriate Word styles. When the output is copied and pasted into another Word document, the styles are carried along.

However, the styles will only take effect if the output is pasted into a Word document that uses the same Word styles. If the styles TableNumber, TableTitle, TableHeader, and TableData aren't defined in the Word document, then the inserted text will be assigned those styles, but Word won't know how to format them.

➢ Using Other Styles

This example used custom style names, which would need to be set up in a special Word document. But it's also possible to use style names that are built in to Word. For example, instead of "TableTitle", the style "Header 2" could have been used. This style is predefined in Word.

CONCLUSIONS

This technique can be a little confusing at first, and it takes a bit of work to set up appropriate tags. If there is one table that needs to be inserted, then it would be faster to manually reformat it in Word. However, if there are 200 tables, or if the one table must be rerun every day of the year, then it may be worth the investment to add the style tags to the ODS RTF output.

➢ RESOURCES

PROC TEMPLATE documentation is in the References chapter of:

Guide to the Output Delivery System in SAS Online Doc, version 8, ©1999, SAS Institute Inc., Cary, NC, USA.

Preliminary documentation of new ODS features and sample programs can be found at:

http://www.sas.com/rnd/base/index-ods-resources.html.

Documentation of the RTF Specifications (useful for looking up RTF tags):

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnrtfspec/html/rtfspec.asp

➢ ACKNOWLEDGEMENTS

➢ CONTACTING THE AUTHOR

Please direct any questions or feedback to the author at: info@laurenhaworth.com

APPENDIX A: SAMPLE PROGRAM

KEY MODIFICATIONS ARE HIGHLIGHTED

```
ods listing close;

*****************************************************************************
* Create a custom style
*****************************************************************************;
proc template;
   define style PrettyTable;
      parent=styles.rtf;
      replace fonts /
         'TitleFont2' = ("Arial",12pt,Normal)
         'TitleFont' = ("Arial",12pt,Normal)
         'StrongFont' = ("Arial",10.1pt)
         'EmphasisFont' = ("Arial",10.1pt)
         'headingEmphasisFont' = ("Arial",10.1pt)
         'headingFont' = ("Arial",10.1pt)
         'docFont' = ("Arial",10.1pt)
         'EmphasisFont' = ("Arial",10.1pt)
         'FixedEmphasisFont' = ("Courier",9pt)
         'FixedStrongFont' = ("Courier",9pt)
         'FixedHeadingFont' = ("Courier",9pt)
         'BatchFixedFont' = ("Courier",6.7pt)
         'FixedFont' = ("Courier",9pt);
      style table from table /
         rules=groups
         frame=box
         cellpadding = 3pt
         cellspacing = 0pt
         borderwidth = 0.5pt;
      replace Body from Document /
         bottommargin = 1in
         topmargin = 1in
         rightmargin = 1in
         leftmargin = 1.25in;
      replace HeadersAndFooters from Cell /
         font = fonts('docFont')
         font_weight = bold
         foreground = colors('datafg')
         background = colors('databg')
         protectspecialchars=off;
      replace SystemFooter from TitlesAndFooters /
         font = fonts('docFont')
         just = L;
      style data from data /
            pretext="^R'\s5 '";
   end;
run;

*****************************************************************************
* Set up macro variable with RTF tags for Word styles
*****************************************************************************;
%let styles=%nrbquote({\s2 TableNumber;}{\s3 TableTitle;}{\s4 TableHeader;}{\s5
TableData;});

*****************************************************************************
* Create the titles and table
*****************************************************************************;
ods escapechar='^';
```

```
ods rtf file='c:\temp\sample2.rtf' style=PrettyTable bodytitle;
title "^R'\s2 'Table 23.1";
title2 "^R'\s3 'Enrolled Students";
proc report data=sashelp.class nowd;
   column sex age;
   define sex / group "^R'\s4 'Sex";
   define age / analysis mean "^R'\s4 'Mean Age" format=5.1;
run;
ods rtf close;

*******************************************************************************
* Edit the document header
*******************************************************************************;
data _null_;
  infile 'c:\temp\sample2.rtf' lrecl=500;
  file 'c:\temp\sample3.rtf' lrecl=500;
  length temp $ 600;
  input ;
  if index(_infile_,"Normal;}")>0 then do;
     temp=tranwrd(_infile_,"Normal;}","Normal;}&styles");
       put temp;
  end;
  else put _infile_;
run;
```