PharmaSUG2010 - Paper TT13

%rtf2data : A utility macro to convert RTF Table to SAS® dataset

Neeral Beladia, MaxisIT Inc., Metuchen, NJ

ABSTRACT

Often in the pharmaceutical industry, we face the challenge of reverse engineering, where we have archived summary Tables or Listings in the form of RTF Table, however the programs that led to their generation are untraceable. In scenarios like these or when there is a need to perform extended statistical analyses on the RTF Table data or when it is critical to ensure a match between multiple versions of RTF Table, it would be nice if we would be able to convert the RTF Table into a SAS® dataset. This paper focuses on presenting a utility macro(%rtf2data) that converts most commonly used RTF Table templates to SAS® dataset. The macro executes in a 2-phase process. In the first phase, the macro reads in a user-defined meta-data that defines the structure of the RTF file and also defines the attributes of variables in the output SAS® dataset. In the second phase, the macro reads in the RTF Table file, parsing through the underlying RTF scripts that build the RTF Table, identifying the free text data and generates the SAS® dataset using the attributes defined by the meta-data read in the first phase. The macro can also be used to convert RTF Table that has logical boundaries as in the case of Baseline Characteristics Tables wherein each category defines start of a new logical section though having the same underlying structure; or physical boundaries where unequal structured RTF sections are stacked up to form a RTF Table into separate SAS® datasets.

INTRODUCTION

The Rich Text Format (RTF) Specification is a method of encoding formatted text and graphics for easy transfer between applications. An RTF file consists of unformatted text, control words, control symbols, and groups. With the RTF Specification, documents created under different operating systems and with different software applications can be transferred between those operating systems and applications.

A control word is a specially formatted command that RTF uses to mark printer control codes and information that applications use to manage documents. A control word cannot be longer than 32 characters. A control word takes the following form:

\LetterSequence<Delimiter>

The LetterSequence is made up of lowercase alphabetic characters between "a" and "z" inclusive. RTF is case sensitive, and all RTF control words must be lowercase.

The delimiter marks the end of an RTF control word, and can be one of the following:

- A space. In this case, the space is part of the control word.
- A digit or a hyphen (-), which indicates that a numeric parameter follows. The subsequent digital sequence is then delimited by a space or any character other than a letter or a digit.
- Any character other than a letter or a digit. In this case, the delimiting character terminates the control word but is not actually part of the control word.

If a space delimits the control word, the space does not appear in the document. Any characters following the delimiter, including spaces, will appear in the document.

A control symbol consists of a backslash followed by a single, non-alphabetic character. For example, \~ represents a non-breaking space. Control symbols take no delimiters.

A group consists of text and control words or control symbols enclosed in braces ({ }). The opening brace ({) indicates the start of the group and the closing brace (}) indicates the end of the group. Each group specifies the text affected by the group and the different attributes of that text. The RTF file can also include groups for fonts, styles, screen color, pictures, footnotes, comments (annotations), headers and footers, summary information, fields, and bookmarks, as well as document-, section-, paragraph-, and character-formatting properties. A cell may have more than one paragraph in it; the cell is terminated by a cell mark (the \cell control word)

It is important to note that PROC REPORT automatically makes a bookmark named IDX# for every output table produced. Thus, if you call PROC REPORT three times, your output RTF file will contain bookmarks IDX1, IDX2, and IDX3, marking the beginning of each output RTF table. The following syntax allows you to create an RTF bookmark:
{*\bkmkstart *bookmark_name*}*display_text*{*\bkmkend *bookmark_name*}

The following syntax allows you to create an external hyperlink with any destination and any display text you choose. The hyperlink will open the file specified by filename_and_path and the actual link will read display_text.
{\field {*\fldinst HYPERLINK "filename_and_path"}{\fldrslt display_text}}

The aim of the macro %rtf2data is to identify free text that is contained in each cell of the RTF Table, identify sections and generate SAS® dataset(s).

Below are some of the situations where the macro %rtf2data can be used

- Validation of RTF Tables programmatically
- Comparison of multiple versions of the same RTF Table file
- Providing an additional file format for external raw data
- Extended Statistical analysis on data from RTF Table
- Extended analysis based on population defined by data from RTF Table style Listings

Often, in the pharmaceutical industry, we come across RTF Tables with N (%) in the top portion of the Table with Treatment groups as columns and Between Treatment group comparison Confidence Interval and P-value in the bottom portion of the Table with CI and P-value as columns and Treatment Comparison groups as rows. The macro provides a distinct feature of converting stacked up unequal number of column RTF Tables into separate SAS® datasets.

The paper explains in detail the layout and structure of the meta-data; algorithmic steps to walk through the process/technique used by the %rtf2data macro for converting RTF Table to SAS® dataset(s); test cases and the corresponding output of executing the macro on the same.

CREATION OF META-DATA

The meta-data is an Excel sheet that consists of:

- Number of Sections in the RTF file : integer
- Section Dataset Name : text
- Section Dataset Label : text
- Section Identifier Text : text delimited by '|'
- Section Dataset Append Flag : boolean
- Section Flag to indicate whether column headers appear after Section Identifier : boolean
- Section Variable Name : text
- Section Variable Header/Label : text
- Section Variable Length : text
- Section Variable Header deletion Flag : boolean

	A	B	C	D	E	F	G	H	I	J
1	Sections	4								
2										
3	section	dsetnm	dsetlbl	identifier	append	incol	varnm	varhdr	varlen	delhdr
4	1	outlib.cat1	Category 1	Category 1	0	1	trt	Treatment	$30	1
5	1	outlib.cat1	Category 1	Category 1	0	1	sub1	Subcat1 n(%)	$20	1
6	1	outlib.cat1	Category 1	Category 1	0	1	sub2	Subcat2 n(%)	$20	1
7	1	outlib.cat1	Category 1	Category 1	0	1	tot	Total N	$20	1
8	2	outlib.cat2	Category 2	Category 2	0	1	trt	Treatment	$30	1
9	2	outlib.cat2	Category 2	Category 2	0	1	sub1	Subcat1 n(%)	$20	1
10	2	outlib.cat2	Category 2	Category 2	0	1	sub2	Subcat2 n(%)	$20	1
11	2	outlib.cat2	Category 2	Category 2	0	1	tot	Total N	$20	1
12	3	outlib.cat3	Category 3	Category 3	0	1	trt	Treatment	$30	1
13	3	outlib.cat3	Category 3	Category 3	0	1	sub1	Subcat1 n(%)	$20	1
14	3	outlib.cat3	Category 3	Category 3	0	1	sub2	Subcat2 n(%)	$20	1
15	3	outlib.cat3	Category 3	Category 3	0	1	sub3	Subcat3 n(%)	$20	1
16	3	outlib.cat3	Category 3	Category 3	0	1	sub4	Subcat4 n(%)	$20	1
17	3	outlib.cat3	Category 3	Category 3	0	1	tot	Total N	$20	1
18	4	outlib.footn	Footnotes	Foot1	0	0	footer	Foot1	$200	0

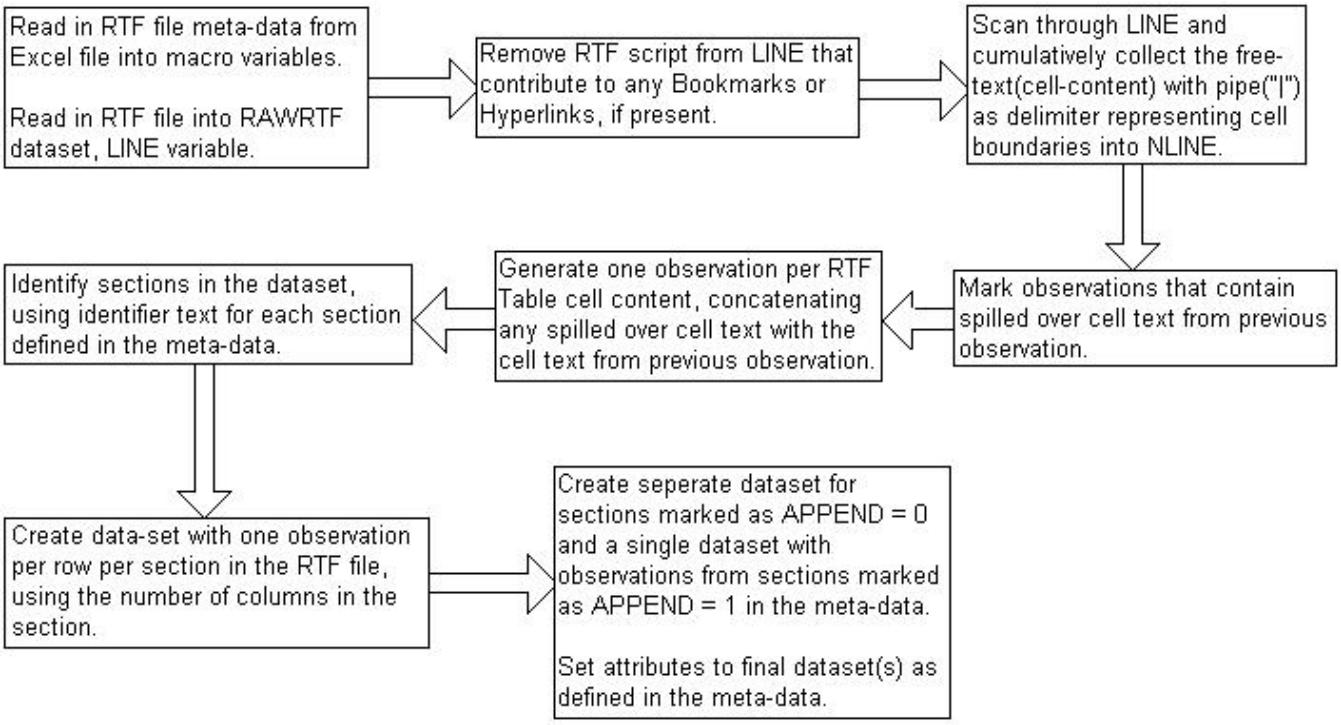
The macro %read_meta reads in the meta-data into macro variables which would be used in %rtf2data macro to identify the lay-out of the RTF file and attributes of the target dataset(s) to be generated.

The first step in generating the meta-data would be to identify physical or logical boundaries in a RTF Table for the identification of start of section and the number of sections. The header/label of the first column of each section is usually a good candidate for IDENTIFIER text. However, there may be cases where homogenous sections start without column labels since all column labels appeared at the start of the RTF Table. In such cases, each section may precede with some text/label to identify the section. In cases where footnotes exists, the first few words that could uniquely identify the footnote section becomes the IDENTIFIER text for the section.

APPEND flag is set to 1 for identical sections(sections with homogenous attributed columns) that appear in the RTF Table. INCOL is set to 1 for sections that have columns headers repeated within the RTF Table. Hence, if INCOL = 1, usually APPEND is set to 0. DELHDR is another flag that is set to 1(usually the case) for section where user does not intent to have column identifiers to be part of the data. One case, where DELHDR is usually 0 is for footnote section.

ALGORITHM OVERVIEW

This section below will walk step-by-step through the technique used and the process to convert RTF Table(s) to SAS® dataset(s).



For better understanding, a RTF Table of 2 columns and 2 data rows has been chosen as an example for the Algorithm purpose. The left side of the below image shows the RTF Table and the right side shows the meta-data set up for the same.

Column 1	Column 2
CELL 1	CELL 2
CELL 3	CELL 4

	A	B	C	D	E	F	G	H	I	J
1	Sections	1								
2										
3	section	dsetnm	dsetlbl	identifier	append	incol	varnm	varhdr	varlen	delhdr
4	1	outlib.algo	Example	Column 1	0	0	col1	Column 1	$20	1
5	1	outlib.algo	Example	Column 1	0	0	col2	Column 2	$20	1
6										

STEP 1

Read in the meta-data Excel spreadsheet meta_<filename>.xls to save the following information in macro variables
All references to i represents Section #i; and j represents Variable #j within Section #i.

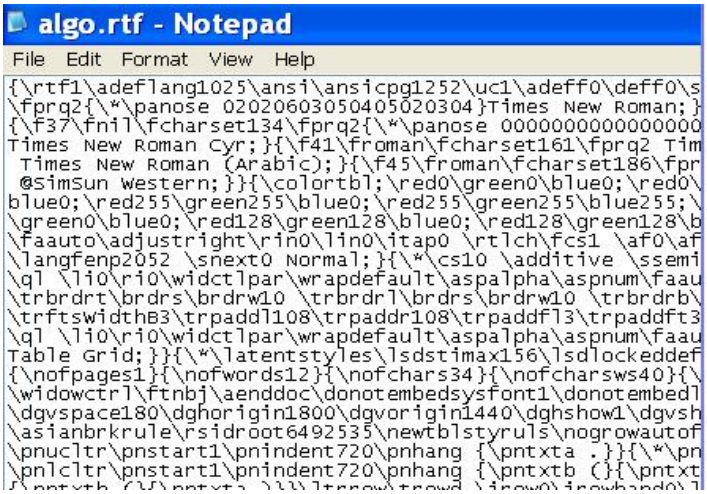
- SECTIONS : &sections
- DSETNM : &&dnm_&i
- DSETLBL : &&dbl_&i
- IDENTIFIER : &&&idval_&i_&j
- IDENTIFIER COUNT : &&idcnt_&i
- APPEND DATASET FLAG : &&app_&i
- IN TABLE COLUMN FLAG : &&incol_&i
- VARIABLE NAME : &&&varnm_&i_&j
- VARIABLE HEADER : &&&varhdr_&i_&j
- VARIABLE LENGTH : &&&varlen_&i_&j
- SECTION VARIABLE HEADERS(s) DELETION FLAG(s) : &&delhdr_&i

STEP 2

INITIALIZE flags
 LASTSEC = 0
 SEC_CNT1,…,SEC_CNT&sections = 0
 IDXFLG = 0
 LINKFLG = 0
Replace all occurrences of '\\' by '@@'; '\{' by '^O'; '\}' by '^C';
'\line' by '#';
'{' or '}' by a <space>; and
'\cell' by '$ * \cell'.

STEP 3

Read the rtf file one line at a time to a SAS® dataset(rawrtf.sas7bdat) in variable LINE



	LINE
1	{\rtf1\adeflang1025\ansi\ansicpg1252\uc1\adeff0\deff0\stshfdbch13\stshflo 020206030504050203 04}Times New Roman;}{\f13\fnil\fcharset134\fprq2{*\panose 020106000301010101 01}SimSun{*\falt ??\'a8\'ac?};}
2	{\f37\fnil\fcharset134\fprq2{*\panose 000000000000000000 00}@SimSun;}{\f38\froman\fcharset238\fprq2 Times New Roman CE;}{\f39\froman\fcharset204\fprq2 Times New Roman Cyr;}{\f41\froman\fcharset161\fprq2 Times New Roman Greek;}
3	{\f42\froman\fcharset162\fprq2 Times New Roman Tur;}{\f43\fbidi \froman\fcharset177\fprq2 Times New Roman (Hebrew);}{\f44\fbidi \froman\fcharset178\fprq2 Times New Roman (Arabic);}{\f45\froman\fcharset186\fprq2 Times New Roman Baltic;}
4	{\f46\froman\fcharset163\fprq2 Times New Roman (Vietnamese);}{\f170\fnil\fcharset0\fprq2 SimSun Western{*\falt ??\'a8\'ac?};}{\f410\fnil\fcharset0\fprq2 @SimSun Western;}}{\colortbl;\red0\green0\blue0;\red0\green0\blue255;\red0\green2
5	\red0\green255\blue0;\red255\green0\blue255;\red255\green0\blue0;\red2
6	\red128\green128\blue128;\red192\green192\blue192;}{\stylesheet{\ql \li0\ri0\widctlpar\wrapdefault\aspalpha\aspnum\faauto\adjustright\rin0\lin0\it \rtlch\fcs1 \af0\afs24\alang1025 \ltrch\fcs0
7	\fs24\lang1033\langfe2052\loch\f0\hich\af0\dbch\af13\cgrid\langnp1033\l \snext0 Normal;}{*\cs10 \additive \ssemihidden Default Paragraph Font;}{*

STEP 4

IF LINE contains "bkmkstart IDX"
 IDXFLG = 1
IF LINE contains "\fldinst"
 LINKFLG = 1

STEP 5

IF IDXFLG == 1 && LINE contains "bkmkstart IDX" THEN DO
 IF LINE contains occurrence of Label of the first variable (&&varhdr_&i_1) for section #i;
 where I = LASTSEC + 1 THEN DO
 remove all text between start of text "bkmkstart IDX" and start of "&&varhdr_&i_1"
 IDXFLG = 0
 END
END
ELSE IF IDXFLG == 1 THEN DO
 IF LINE contains occurrence of Label of the first variable (&&varhdr_&i_1) for section #i;
 where I = LASTSEC + 1 THEN DO
 remove all text between position 1 in LINE and start of "&&varhdr_&i_1"
 IDXFLG = 0
 END
 ELSE LINE = <null string>
END

STEP 6

IF LINKFLG == 1 && LINE contains "\fldinst" THEN DO
 IF LINE contains "\fldrslt" THEN DO
 remove all text between start of text "\fldinst" and start of "\fldrslt"

```
                LINKFLG = 0
        END
END
ELSE IF LINKFLG == 1 THEN DO
        IF LINE contains “\fldrslt” THEN DO
                remove all text between position 1 in LINE and start of “\fldrslt”
                LINKFLG = 0
        END
        ELSE LINE = <null string>
END
```

STEP 7

	LINE	NLINE
1	\b\insrsid6492535\charrsid6492535 \hich\af0\dbch\af13\loch\f0 Column 1 $ * \cell \hich\af0\dbch\af13\loch\f0 Column 2 $ * \cell \pard\plain \ltrpar\ql \li0\ri0\widctlpar\intbl\wrapdefault\aspalpha\aspnum\faauto\adjustright\rin0\lin0 \rtlch\fcs1	Column 1 $ | Column 2 $
2	\af0 \ltrch\fcs0 \insrsid6492535 \hich\af0\dbch\af13\loch\f0 CELL \li0\ri0\widctlpar\wrapdefault\aspalpha\aspnum\faauto\adjustright\rin0\lin0\itap0\li0\ri0\widctlpar\wrapdefault\aspal	CELL
3	1 $ * \cell \hich\af0\dbch\af13\loch\f0 CELL \li0\ri0\widctlpar\wrapdefault\aspalpha\aspnum\faauto\adjustright\rin0\lin0\itap0\li0\ri0\widctlpar\wrapdefault\aspal \li0\ri0\widctlpar\wrapdefault\aspalpha\aspnum\faauto\adjustright\rin0\lin0\itap0 \li0\ri0\widctlpar\wrapdefault\aspalpha\aspnum\faauto\adjustright\rin0\lin0\itap0 2 $ * \cell \pard\plain \ltrpar\ql \li0\ri0\widctlpar\intbl\wrapdefault\aspalpha\aspnum\faauto\adjustright\rin0\lin0 \rtlch\fcs1 \af0\afs24\alang1025	1 $ | CELL 2 $
4	\af0 \ltrch\fcs0 \insrsid6492535 \hich\af0\dbch\af13\loch\f0 CELL 3 $ * \cell \hich\af0\dbch\af13\loch\f0 CELL 4 $ * \cell \pard\plain \ltrpar\ql \li0\ri0\widctlpar\intbl\wrapdefault\aspalpha\aspnum\faauto\adjustright\rin0\lin0 \rtlch\fcs1 \af0\afs24\alang1025	CELL 3 $ | CELL 4 $

Scan through LINE.

```
NLINE = <null string>
IF LINE contains “* \cell” THEN DO
        TMPLINE = LINE
        DO WHILE(TMPLINE contains “* \cell”)
                read text from the position of last “* \cell” + 8(initially from position 1) to the start of position of the next occurrence of “* \cell” into TMPLINE
                scan through TMPLINE collecting free-text(text devoid of “\”), inserting “|”(pipe) wherever “* \cell” occurs.
                NLINE = NLINE + “ | ” + <collected-free-text>
        END
        // for the remaining text, i.e. from the last occurrence of “* \cell” to the end of string in LINE //
        IF LENGTH(LINE) > last position of “* cell” + 8 THEN DO
                TMPLINE = remaining text from last position of “* cell” + 8 to the end of string in LINE
                scan through TMPLINE collecting free-text(text devoid of “\”), inserting “|”(pipe) wherever “* \cell” occurs.
                NLINE = NLINE + “ | ” + <collected-free-text>
        END
END
ELSE DO
        scan through LINE collecting free-text(text devoid of “\”)
        NLINE = NLINE + <collected-free-text>
END
```

STEP 8

Delete all observations that have a null string NLINE and LINE that has no occurrences of ‘* \cell’.

STEP 9

Derive LASTADD(boolean flag) such that LASTADD = 1 for observations that have partial cell free-text spilled over from the previous observation.

	LINE	NLINE	LASTADD
1	\b\insrsid6492535\charrsid6492535 \hich\af0\dbch\af13\loch\f0 Column 1 $ * \cell \hich\af0\dbch\af13\loch\f0 Column 2 $ * \cell \pard\plain \ltrpar\ql \li0\ri0\widctlpar\intbl\wrapdefault\aspalpha\aspnum\faauto\adjustright\rin0\lin0 \rtlch\fcs1	Column 1 $ | Column 2 $	0
2	\af0 \ltrch\fcs0 \insrsid6492535 \hich\af0\dbch\af13\loch\f0 CELL \li0\ri0\widctlpar\wrapdefault\aspalpha\aspnum\faauto\adjustright\rin0\lin0\itap0\li0\ri0\widctlpar\wrapdefault\a	CELL	0
3	1 $ * \cell \hich\af0\dbch\af13\loch\f0 CELL \li0\ri0\widctlpar\wrapdefault\aspalpha\aspnum\faauto\adjustright\rin0\lin0\itap0\li0\ri0\widctlpar\wrapdefault\a \li0\ri0\widctlpar\wrapdefault\aspalpha\aspnum\faauto\adjustright\rin0\lin0\itap0 \li0\ri0\widctlpar\wrapdefault\aspalpha\aspnum\faauto\adjustright\rin0\lin0\itap0 2 $ * \cell \pard\plain \ltrpar\ql \li0\ri0\widctlpar\intbl\wrapdefault\aspalpha\aspnum\faauto\adjustright\rin0\lin0 \rtlch\fcs1 \af0\afs24\alang1025	1 $ | CELL 2 $	1
4	\af0 \ltrch\fcs0 \insrsid6492535 \hich\af0\dbch\af13\loch\f0 CELL 3 $ * \cell \hich\af0\dbch\af13\loch\f0 CELL 4 $ * \cell \pard\plain \ltrpar\ql \li0\ri0\widctlpar\intbl\wrapdefault\aspalpha\aspnum\faauto\adjustright\rin0\lin0 \rtlch\fcs1 \af0\afs24\alang1025	CELL 3 $ | CELL 4 $	0

For e.g. if a cell text contains “ABC DEF XYZ” and the cell text is spilled over 3 observations, i.e. NLINE = “ABC” for 1st observation, “DEF” for 2nd observation and “XYZ” for the 3rd observation. In this case, LASTADD would have value of 0 for the 1st observation and a value of 1 for the 2nd and 3rd observation.

STEP 10

	NLINE
1	Column 1 $
2	Column 2 $
3	CELL1 $
4	CELL 2 $
5	CELL 3 $
6	CELL 4 $

Scan through the dataset and generate one observation per cell text, concatenating NLINE values with the previous observation NLINE value, if LASTADD = 1. Note, that since LASTADD was 1, text “CELL” and “1” was concatenated without adding a space.

STEP 11

	NLINE	SECTION	LASTSEC
1	Column 1 $	1	1
2	Column 2 $	1	1
3	CELL1 $	1	1
4	CELL 2 $	1	1
5	CELL 3 $	1	1
6	CELL 4 $	1	1

Identify the sections in rawrtf dataset, using &&idval_&i_&j to identify the start of each section.
Increment the cumulative count for each section sec_cnt&i by 1 for each occurrence of “&&idval_&i_&j” in NLINE.
Perform check for match for identifier only for sections yet to be identified (i.e. &i > LASTSEC)

```
IF sec_cnt&i == &&idnt&i THEN DO
      SECTION = &i
      LASTSEC = &i
END
```

Delete all observations before the start of section #1.

STEP 12

	NLINE1
1	Column 1 $|Column 2 $
2	CELL1 $|CELL 2 $
3	CELL 3 $|CELL 4 $

Scan through observations in the dataset and generate one observation at a time from one or more observations, such that all variable values that constitute to one observation in a section are present in NLINE1 with ‘|’ delimiting the variable values.

STEP 13

```
Convert back '@@' to '\', '$ *' to <space>, '^O' by '{' and '^C' by '}'
%DO i = 1 %TO &SECTIONS
        %LET cntvar&i = number of variables in section #i
        IF SECTION = &i THEN DO
                create dataset &&dnm_&i
                %DO j = 1 %TO &&cntvar&i
                        DO WHILE(scan(NLINE1, &j, "|") ^= "")
                                &&&varnm_&i_ = scan(NLINE1, &j, "|")
                                K = K + 1
                        END
                %END
                OUTPUT TO &&dnm_&i
        END
%END
```

STEP 14

	Column 1	Column 2
1	CELL1	CELL2
2	CELL3	CELL4

Create separate datasets for sections that have &&app_&i = 0.
Create a single dataset 'final' consisting of observations from sections in order for which &&app_&i = 1.
Assign variable and dataset attributes as dictated by the meta-data read-in into macro variables in STEP 1.

OVERVIEW OF TEST CASES

The following are the test cases to demonstrate the results of executing the macro %rtf2data. Each test case has been detailed with a RTF Template Table and a snapshot of the output SAS® dataset generated.

CASE 1

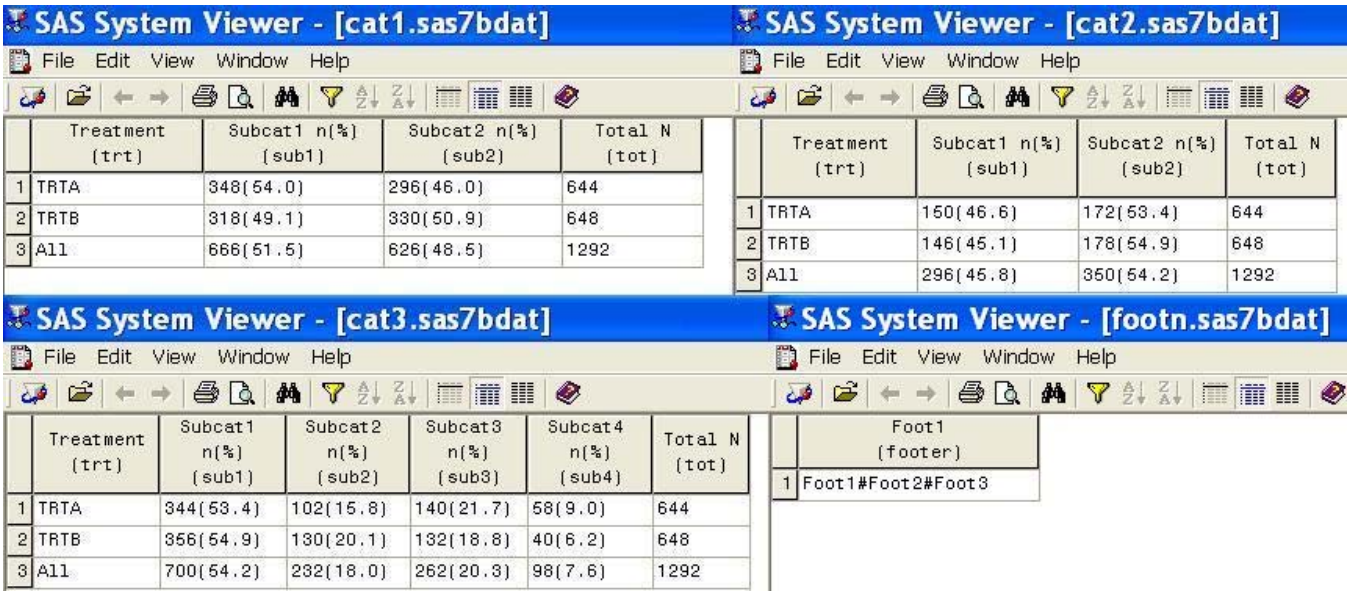
Test Case #1 depicts a RTF Table template(case1.rtf) often used for summarizing Baseline Characteristics. Please refer to "CREATION OF META-DATA" section for the meta-data example for case 1. The initial step is to identify logical or physical boundaries that define start of each section within RTF Table. Here, each section, except the footnote section, has a header "Category #". Hence, "Category #" becomes a strong candidate for the identifier value in the meta-data for the first 3 sections in the RTF Table.

Category 1					
Treatment	Subcat1 n (%)		Subcat2 n (%)		Total N
TRT A	348 (54.0)		296 (46.0)		644
TRT B	318 (49.1)		330 (50.9)		648
All	666 (51.5)		626 (48.5)		1292
Category 2					
Treatment	Subcat1 n (%)		Subcat2 n (%)		Total N
TRT A	150 (46.6)		172 (53.4)		644
TRT B	146 (45.1)		178 (54.9)		648
All	296 (45.8)		350 (54.2)		1292
Category 3					
Treatment	Subcat1 n (%)	Subcat2 n (%)	Subcat3 n (%)	Subcat4 n (%)	Total N
TRT A	344 (53.4)	102 (15.8)	140 (21.7)	58 (9.0)	644
TRT B	356 (54.9)	130 (20.1)	132 (18.8)	40 (6.2)	648
All	700 (54.2)	232 (18.0)	262 (20.3)	98 (7.6)	1292
Foot1 Foot2 Foot3					

For the footnote section, since there are no column headers, the only way to identify the start of footnote section would be to have the text “Foot1” as identifier value for the footnote section.

An important thing to notice as far as the meta-data file is concerned, is how the values for INCOL and DELHDR are set for different sections. Since, the column headers exists at the start of each section, INCOL has been set to 1 for each section other than the footnote section. Also, DELHDR is set to 1 for all sections other than the footnote section, which informs the macro to delete the column headers for each section except the footnote section.

The below image shows the output of executing the macro %rtf2data on case2.rtf. Notice how the macro generates 4 separate datasets, one for each section. The dataset and variable attributes were set as defined in the meta-data excel sheet. Notice how the ‘footn’ dataset contains only one observation, with all 3 footnotes separated by “#”(for line feed). This is because each of the footnotes were part of the same cell, separated by line feed(“\line” in RTF).



SAS System Viewer - [cat1.sas7bdat]

File Edit View Window Help

	Treatment (trt)	Subcat1 n(%) (sub1)	Subcat2 n(%) (sub2)	Total N (tot)
1	TRTA	348(54.0)	296(46.0)	644
2	TRTB	318(49.1)	330(50.9)	648
3	All	666(51.5)	626(48.5)	1292

SAS System Viewer - [cat2.sas7bdat]

File Edit View Window Help

	Treatment (trt)	Subcat1 n(%) (sub1)	Subcat2 n(%) (sub2)	Total N (tot)
1	TRTA	150(46.6)	172(53.4)	644
2	TRTB	146(45.1)	178(54.9)	648
3	All	296(45.8)	350(54.2)	1292

SAS System Viewer - [cat3.sas7bdat]

File Edit View Window Help

	Treatment (trt)	Subcat1 n(%) (sub1)	Subcat2 n(%) (sub2)	Subcat3 n(%) (sub3)	Subcat4 n(%) (sub4)	Total N (tot)
1	TRTA	344(53.4)	102(15.8)	140(21.7)	58(9.0)	644
2	TRTB	356(54.9)	130(20.1)	132(18.8)	40(6.2)	648
3	All	700(54.2)	232(18.0)	262(20.3)	98(7.6)	1292

SAS System Viewer - [footn.sas7bdat]

File Edit View Window Help

	Foot1 (footer)
1	Foot1#Foot2#Foot3

CASE 2

Test Case #2 depicts a RTF Table template(case2.rtf) of a conventional define.rtf. Since the layout of both “Adverse Events” and “Vital Signs” sections are identical in terms of the column headers and number columns, one could choose to append the sections setting the append flag to 1 in the meta-data for both the sections.

Adverse Events

Variable	Label	Variable Number	Type	Codes	Comments
AEBEGDP	DAYS POST TX TO DATE BEGAN	37	NUM		AEBEGDT - TXENDDT
AEBEGTM	TIME BEGAN	18	TIME5		CRF page 34
AEBODSYS	EVENT MEDDRA SOC TEXT	16	CHAR		Decoded body system for MedDRA dictionary assigned
EMERGFLG	TREATMENT EMERGENT FLAG (1=EMERGENT)	12	NUM	0=NO , 1=YES	Treatment emergent algorithm. See DAP for details.

Vital Signs

Variable	Label	Variable Number	Type	Codes	Comments
AGE	AGE AT ENTRY	5	NUM		Integer of (Date of Screening - Birth Date)/365.25
BMASMTDT	DATE OF ASSESSMENT.	12	DATE9		CRF page 15
BMASMTDY	DAYS IN PERIOD TO ASSESSMENT	17	NUM		BMASMTDT - TXBEGDT+1

Also, since the column headers are repeated within each section, INCOL flag needs to be set to 1 in the meta-data. Note that, even though the column headers are repeated at the start of each section and at the start of each page, the RTF file header i.e. “Adverse Events” on the first page was used to identify the sections.

The below image shows the output of executing the macro %rtf2data on case2.rtf. Notice how the macro generates a single dataset, with observations stacked up in the order of the existence of their sections in the RTF Table since APPEND was set to 1 in the meta-data. Note the addition of variable “idval” whose values were based on the identifier text entered in the meta-data for the corresponding section. The dataset and variable attributes were set as defined in the meta-data excel sheet.

SAS System Viewer - [case2.sas7bdat]

File Edit View Window Help

	idval (idval)	Variable (variable)	Label (label)	Variable Number (varnum)	Type (type)	Codes (codes)	Comments (comments)
1	Adverse Events	AEBEGDP	DAYS POST T	37	NUM		AEBEGDT- TXENDDT
2	Adverse Events	AEBEGTM	TIME BEGAN	18	TIME5		CRF page 34
3	Adverse Events	AEBODSYS	EVENT MEDDRA	16	CHAR		Decoded body syste
4	Adverse Events	EMERGFLG	TREATMENT EM	12	NUM	0=NO ,1=YES	Treatment emergent
5	Vital Signs	AGE	AGE AT ENTR	5	NUM		Integer of (Date
6	Vital Signs	BMASMTDT	DATE OF ASS	12	DATE9		CRF page 15
7	Vital Signs	BMASMTDY	DAYS IN PER	17	NUM		BMASMTDT - TXBEGDT

CASE 3

Test Case #3 depicts a RTF Table template(case3.rtf) often used for summarizing Baseline Characteristics and is very similar to Test Case #1 except the fact that the column headers are consistent across the 2 sections and hence are not repeated at the start of each section. Most of the format of meta-data that was used for Case #1 could be used to Case #3 with a few modifications., i.e. set INCOL to 0 and column headers. APPEND could be set to 1, as is done in the below run.

Treatment	N	Mean	SD	Median	Range
Category 1					
Treatment 1	2322	124.0	34.4	128.9	38.7 to 202.2
Placebo	1324	121.8	34.0	123.4	36.0 to 199.2
All	3646	122.9	34.2	126.9	36.0 to 202.2
Category 2					
Treatment 1	2322	48.1	10.8	46.5	25.0 to 84.5
Placebo	1324	48.5	11.8	47.0	24.5 to 98.5
All	3646	48.3	11.3	47.0	24.5 to 98.5

The below image shows the output dataset generated as a result of executing the macro %rtf2data on case3.rtf. Notice how rtf2data generates a single dataset with data from each section stacked up in the other of their appearance in the RTF file.

SAS System Viewer - [case3.sas7bdat]

File Edit View Window Help

	idval (idval)	Treatment (trt)	N (count)	Mean (mean)	SD (sd)	Median (median)	Range (range)
1	Category 1	Treatment1	2322	124.0	34.4	128.9	38.7to202.2
2	Category 1	Placebo	1324	121.8	34.0	123.4	36.0to199.2
3	Category 1	All	3646	122.9	34.2	126.9	36.0to202.2
4	Category 2	Treatment1	2322	48.1	10.8	46.5	25.0to84.5
5	Category 2	Placebo	1324	48.5	11.8	47.0	24.5to98.5
6	Category 2	All	3646	48.3	11.3	47.0	24.5to98.5

CASE 4

Test Case #4 depicts a RTF Table template(case4.rtf) often used for summarizing Lab/Vital Signs results at various study timepoints. The titles “Lab Test: HEMOGLOBIN” and “Lab Test: PLATELET” for the 2 sections were used as identifier text in the meta-data. Both APPEND and INCOL were set to 1 in the meta-data.

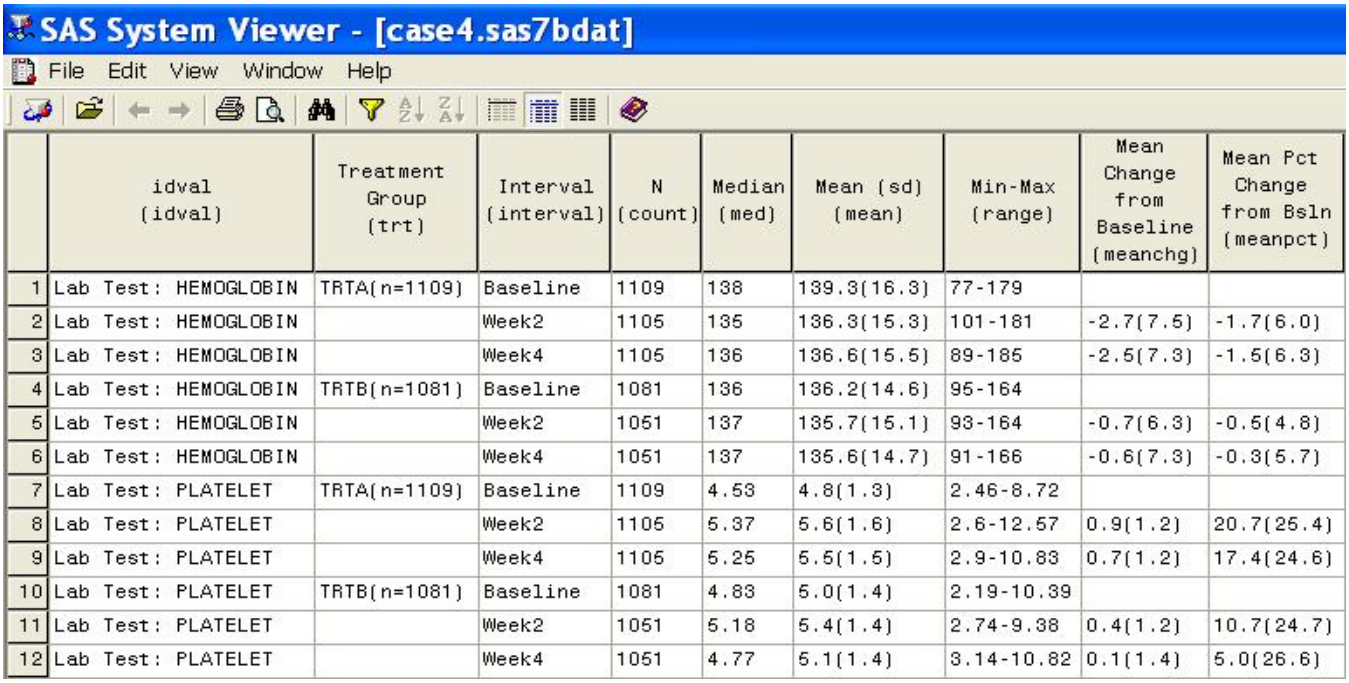
Lab Test: HEMOGLOBIN (g/L)

Treatment Group	Interval	N	Median	Mean (sd)	Min-Max	Mean Change from Baseline (sd)	Mean % Change from Baseline (sd)
TRT A (n = 1109)	Baseline	1109	138	139.3(16.3)	77-179		
	Week 2	1105	135	136.3(15.3)	101-181	-2.7(7.5)	-1.7(6.0)
	Week 4	1105	136	136.6(15.5)	89-185	-2.5(7.3)	-1.5(6.3)
TRT B (n = 1081)	Baseline	1081	136	136.2(14.6)	95-164		
	Week 2	1051	137	135.7(15.1)	93-164	-0.7(6.3)	-0.5(4.8)
	Week 4	1051	137	135.6(14.7)	91-166	-0.6(7.3)	-0.3(5.7)

Lab Test: PLATELET COUNT (10^9/L)

Treatment Group	Interval	N	Median	Mean (sd)	Min-Max	Mean Change from Baseline (sd)	Mean % Change from Baseline (sd)
TRT A (n = 1109)	Baseline	1109	4.53	4.8(1.3)	2.46-8.72		
	Week 2	1105	5.37	5.6(1.6)	2.6-12.57	0.9(1.2)	20.7(25.4)
	Week 4	1105	5.25	5.5(1.5)	2.9-10.83	0.7(1.2)	17.4(24.6)
TRT B (n = 1081)	Baseline	1081	4.83	5.0(1.4)	2.19-10.39		
	Week 2	1051	5.18	5.4(1.4)	2.74-9.38	0.4(1.2)	10.7(24.7)
	Week 4	1051	4.77	5.1(1.4)	3.14-10.82	0.1(1.4)	5.0(26.6)

Note, how the first column(Treatment Group) has a display similar to one generated when you have a GROUP option in PROC REPORT. The macro correctly identifies those as null values as could be seen in the resulting dataset case4.sas7bdat below, in the “Treatment Group”, “Mean Change from Baseline” and “Mean Pct Change from Bsln” columns.



	idval (idval)	Treatment Group (trt)	Interval (interval)	N (count)	Median (med)	Mean (sd) (mean)	Min-Max (range)	Mean Change from Baseline (meanchg)	Mean Pct Change from Bsln (meanpct)
1	Lab Test: HEMOGLOBIN	TRTA(n=1109)	Baseline	1109	138	139.3(16.3)	77-179		
2	Lab Test: HEMOGLOBIN		Week2	1105	135	136.3(15.3)	101-181	-2.7(7.5)	-1.7(6.0)
3	Lab Test: HEMOGLOBIN		Week4	1105	136	136.6(15.5)	89-185	-2.5(7.3)	-1.5(6.3)
4	Lab Test: HEMOGLOBIN	TRTB(n=1081)	Baseline	1081	136	136.2(14.6)	95-164		
5	Lab Test: HEMOGLOBIN		Week2	1051	137	135.7(15.1)	93-164	-0.7(6.3)	-0.5(4.8)
6	Lab Test: HEMOGLOBIN		Week4	1051	137	135.6(14.7)	91-166	-0.6(7.3)	-0.3(5.7)
7	Lab Test: PLATELET	TRTA(n=1109)	Baseline	1109	4.53	4.8(1.3)	2.46-8.72		
8	Lab Test: PLATELET		Week2	1105	5.37	5.6(1.6)	2.6-12.57	0.9(1.2)	20.7(25.4)
9	Lab Test: PLATELET		Week4	1105	5.25	5.5(1.5)	2.9-10.83	0.7(1.2)	17.4(24.6)
10	Lab Test: PLATELET	TRTB(n=1081)	Baseline	1081	4.83	5.0(1.4)	2.19-10.39		
11	Lab Test: PLATELET		Week2	1051	5.18	5.4(1.4)	2.74-9.38	0.4(1.2)	10.7(24.7)
12	Lab Test: PLATELET		Week4	1051	4.77	5.1(1.4)	3.14-10.82	0.1(1.4)	5.0(26.6)

LIMITATIONS OR ASSUMPTIONS

The macro has a few limitations/assumptions that would help identify whether or not a given RTF Table would be a suitable candidate for converting to SAS® dataset using the %rtf2data macro. The macro requires that the first column header at the start of each section must have a non-missing identifier text that doesn't appear as part of the data. The macro also requires that the column header separators(cell boundaries) for the header row holding the identifier text must match with the column separators of the data within the section.

CONCLUSION

The utility macro %rtf2data macro presented in this paper provides an option to entirely avoid visual checking of contents of RTF Table, an exercise performed as part of validation of Summary Tables. The macro also helps in checking updates to a RTF Table between current and previous run. The macro avoids manual intervention of copy-paste of RTF Table contents to a commonly used external file format in order to generate SAS® dataset and not only does it offer the help for performing extended analysis on the contents of the RTF Table, but also provides an additional file format (RTF Table) to be used as external raw data file, often provided as supplementary data to programmers.

ACKNOWLEDGEMENTS

I would like to thank the management of MaxisIT Inc. for their kind support and co-operation.

REFERENCES

- Scott Osowski. Thomas Fritchey (2006): Hyperlinks and Bookmarks with ODS RTF
- Microsoft® Rich Text Format(RTF) Specification, RTF v1.7

CONTACT INFORMATION

Author Name: Neeral Beladia
Company : MaxisIT Inc.
Address : 203 Main St
City : Metuchen State : NJ ZIP : 08840
Email : neeral_beladia@yahoo.com

Please contact the author via email to request for the SAS® code of the %rtf2data macro.