# RTF_READ: Decoding RTF Files

Sandeep Juneja, Ockham, Cary, NC
Andrew Illidge, Genzyme, Cambridge, MA
Daniel Boisvert, Genzyme, Cambridge, MA

## ABSTRACT:

RTF has quickly become the standard medium for the production of tables and listings. RTF, while very useful to medical writers and other downstream processes, has hindered SAS® programmers in that automating QC of tables and listings seemed to become unattainable. Previously, when outputs were produced in ASCII format (.lst) the outputs could simply be read in to create a SAS dataset which could then be compared to the output dataset of the QC program. This paper presents a detailed process of decoding RTF files using SAS and generating SAS dataset(s), thus enabling us to automate QC. In this paper we explain how to identify important tokens in RTF files to extract data, convert special symbols to SAS tokens, handle complex column header spanning and distinguish between the Title and Footnote data from the normal document data. This paper employs only functions that are available in BASE/SAS though due to the nature of parsing text files this paper is aimed at an audience very familiar with complex data step processing.

## INTRODUCTION:

### Rich Text Format Specification (RTF) version 1.6:

The Rich Text Format (RTF) Specification provides a format for text and graphics interchange that can be used with different output devices, operating environments, and operating systems. With the RTF Specification, documents created under different operating systems and with different software applications can be transferred between those operating systems and applications.

### RTF Syntax

RTF syntax breaks down into four basic categories: commands, escapes, groups, and plaintext.

#### RTF Command

An RTF *command* is like \pard or \fs120: a backslash, some lowercase letters, maybe an integer (which might have a negative sign before it), and then maybe a single meaningless space character. In terms of regular expressions, a command matches /\\[a-z]+(-?[0-9]+)? ?/ (including the optional space at the end). An RTF parser knows that a command has ended when it sees a character that no longer matches that pattern. For example, an RTF parser knows that \i\b is two commands because the second "\" couldn't possibly be a continuation of the \i command.

#### RTF Escapes

RTF *escapes* are like *commands* in that they start with a backslash, but that is where the similarity ends. The character after the backslash is **not** a letter. If the escape is \' then the next two characters will be interpreted as hex digits and the escape is understood to mean the character whose ASCII code is the given hex number. For example, the escape \'ea means the ê character because character 0xEA in ASCII is ê. If the character after the \ isn't an apostrophe, the escape consists of just that one character.

There are only three escapes that are of general interest: \~ is the escape that indicates a nonbreaking space; \- is an optional hyphen (a.k.a. a *hyphenation point*); and _ is a nonbreaking hyphen (that is, a hyphen that that's not safe for breaking the line after). The escape * is also part of a construct discussed later. Be sure to note that there is no optional meaningless space after escapes; while \foo\bar is the same as \foo \bar, \'ea\'ea means something different than \'ea \'ea. The first one means "êê" (no space) and the second one means "ê ê" (with a space).

#### RTF Group

An RTF *group* is whatever is between a { and the matching }. For example, {\i Hi there!} is a group that contains the *command* \i and the literal text Hi there!. Some groups are only necessary for certain constructs (like the {\fonttbl...} construct we saw earlier). But most groups have a more concrete purpose: to act as a barrier to the effects of character formatting commands. If you want to italicize the middle word in "a sake cup", use the code a {\i sake} cup. In terms of how this is parsed, the { means "save all the character formatting attributes now," and the } means "restore the character formatting attributes to their most recently saved values."

#### Plaintext

The final bit of RTF syntax is *plaintext*: the text that is sent right through to the document, character for character. For example, when we had Hello, World! in our document, it turned into the text that said simply "Hello, World!".

**Control Words:**
Although there might be lot of control words used in RTF, only a few are required to understand the basic structure of the RTF file and extract data out of it.

| Control Word | Comment | Control Word | Comment |
|---|---|---|---|
| 1. \header | Identify Title in Header Section | 2. \headery | Identify Tiles in Document section |
| 3. \footer | Identify Footnotes in Footer Section | 4. \footery | Identify Footnotes in Document Section |
| 5. \trhdr | Identify Column header | 6. \trowd | Identify Table Row |
| 7. \cell | Identify the Table Cell | 8. \cellx | Identify the cell size in twips |
| 9. \line | Identify line break | 10. \li200 | Works line tab, use to enter extra space. Note numeric value represents twips. |

## SAS & RTF

The process for converting SAS datasets to RTF files is well documented. However, there is little information available about reading RTF files. Decoding RTF files can be a complex process depending upon the level of automation and the requirements of the output dataset desired. The process below explains various steps required in converting an RTF file into a SAS dataset.

### Step 1: Read the RTF document in SAS and confirm it is generated by SAS

Read any input RTF File using below lines of code. Because the RTF generated by Microsoft or other applications is very different to the RTF generated by SAS we must first confirm that this RTF is in the structure we are expecting. By default, the keyword "generator" won't be present if the file is generated by SAS but if it is modified outside SAS it will contain "Microsoft generator" token. To make sure it is modified outside SAS we can also check for file version. By default any fresh run of file will produce version 1 when generated in SAS and if it is greater than 1 that double confirms that its modified outside SAS and we can gracefully exit the macro, otherwise we can continue to process the input file. Once we confirm our RTF was produced by SAS we can continue parsing.

```
data READFILE;
  length ver $1000;
  infile "&RTFFile." missover length = l
         end = lastobs lrecl = 2000;
  input string $varying1500. l;
  rownum = _n_;
  string=_infile_;

  rc1=prxparse("/\\*\\generator/");
  rc2=prxparse("/\\version\d+/");
  if prxmatch(rc1,string) then MWFLG=1;
  ……
```

### Step 2: Identify if the Titles and Footnotes are present in the Header and Footer section or document section of the file

```
%* Header Section;
if index(string,'\header')>0 and index(string,'\footer')=0
then hdrflg=1;
if hdrflg=1 then do; %* HEADER SECTION;
%* PATTERN 1 HEADER INFORMATION *;
 if index(string,'\trowd')>0 then hdr= hdr + 1;
 else if index(string,'\pard{\par}')>0 then do;
    hdr= 0;
    hdrflg=0;
 end;
end;
else do; %* DOCUMENT SECTION;
%* PATTERN 2 HEADER INFORMATION **;
  if index(string,'\headery')>0 and index(string,'\footery')>0
  then hdrflg1=1;
  if index(string,'\pard{\par}')>0 then hdrflg1=0;
  if hdr=1 and index(string,'\trowd')>0 then tblid = tblid + 1;
```

Identifying the presence of \header or \headery and \footer or \footery let us know whether the Titles and Footnotes are located in the header & footer section or document section of the file. Once the location is identified using \trowd token and \trhdr token let us identify presence of different variables or columns in output file.

### Step 3: Extract the Data into SAS dataset:

```
*Extract data from file and store it in Col1 variable.;
if index(string,'{')>0 and index(string,'\cell')>0
   and index(string,'}')>0 then do;
   _col1='';
   col1=trim(left(substr(string,(index(string,'{')+1))));
   col1=tranwrd(col1, '\cell}' , '');
end;
```

Identify the location of the \cell token and it lets you extract the table cells data into col1 variable. Since the structure for the content of the cell is {<data>\cell}, where <data> would be any text contain in the

```
else if index(string,'{')>0 and index(string,'\cell')=0
and index(string,'}')=0 then do;
    _col1=trim(left(substr(string,(index(string,'{')+1))));
end;
else if index(string,'{')=0 and index(string,'\cell')=0
and index(string,'}')=0 and index(string,'\')=0 then do;
    col1=trim(left(_col1)) || " " || trim(left(col1));
    _col1='';
end;
* if \ is missing means no tokens only data,
   read full string;
    _col1=trim(left(_col1))|| " " || trim(left(string));
end;
else if index(string,'{')=0 and index(string,'\cell')>0
and index(string,'}')>0 then do;
    col1=trim(left(substr(string,1,(index(string,'\cell')-1))));
```

current cell, this code snippet checks for the existence of {, \cell and }. Once it finds all three tokens, it extracts the data from the control word and captures it in col1.

There are multiple possibilities for \cell control word. This code snippet will capture correct data.

### Step 4: Distinguish between different rows and columns

Keep only records where \cell control word is present because \cell is the control word which contains data.

```
* Intialize counters for identifying Rows and Cols;
data DATARECS;
  set PARSEFILE1;
  by tblid;
  where indexw(string,'\cellx')=0; * Keep only data records;
  retain rowid colid 0;
  if first.tblid then rowid=0;
  if index(string, '\trowd') or index(string,'\pard{\par}')>0
  then do;
     rowid=rowid + 1;
     colid=0;
  end;
  else do;
     if (index(string,'\cell')>0 then colid=colid+1;
     if (index(string,'\bkmkstart')>0 or
index(string,'\bkmkend')>0)
     then colid= colid + 1;
     if index(string,'{\row}')>0 or index(string,'\footer')>0
     then colid=0;
  end;
run;
*Transpose Vertical data to Horizontal table like structure.;
proc transpose data=DATARECS1 out=TDATARECS(drop=_NAME_)
prefix=c;
 by tblid rowid hdr;
 id colid;
 var col1;
run;
```

Initialize row counter to keep track of different records and for each occurrence of \cell token in each row represents different variable.

Once the data is captured in dataset, transpose the

dataset to replicate it to the input file structure.

**Note:** This step will produce dataset which will to much extent imitate the input file. Depending upon the automation and requirements for the output dataset, below steps can be followed.

### Step 5: Handling the spanned Column headers

Spanning headers are difficult to identify. A spanning header is a text that sits on top of 2 or more columns. In order to identify a header as spanning we must identify the twip values for all columns and compare them.

1. Once we know the twip value and content of the header cell, flag the blank columns
2. MXCNT: Identify maximum number of columns/table to identify if there is any spanning in a table or not.

```
%*Extract twip values and generate twip
data;
data &prefix._twipdata;
  set &prefix._hdr1;
  by tblid hdr;
  where index(string,'\cellx')>0;
```

```
if index(string,'\cellx')>0 then
twpval=scan(string,-1,'\cellx');

  %* Blank Column Flag, to stop spanning from
over-writing blank columns;
    if cval='' then bcol=1;
    else bcol=0;
run;
```

3. SHDR: Identify if header row is spanning or not by comparing number of cols for each header row against max column count maxc.
4. HDRLIST: Get list of unique headers for each table.
5. DUMMY: Create dummy dataset with each header row having multiple columns;
6. Merge Dummy dataset containing one row per table/header with actual data.
7. Merge with Spanning column flag dataset
8. Using the Spanhdr flag(sflag) and Blank flag(Bflag) re-assign the data values for spanning columns.
9. BLANK value is assign for blank columns and later removed.

### Step 6: Distinguish Document data from Titles and Footnotes

```
%*Identify whether table data is TFs or DATA table.;
……
if (hdrflg=1 or hdrflg1=1) then do;
  if hdr=1 then pageid = pageid + 1;
  if hdr>0 then type='TITLE';
  else if hdr=0 then type='FOOTNOTE';
end;
else do;
  if hdr=0 then type='DATA';
  else if hdr>0 then type='HEADER';
end;
…….
```

Once the spanning headers are identified, the title and footnote data needs to be separated from the document data so that they can be processed differently. Based upon the derivation of hdrflg and hdrflg1 variables (explained in Step 2), we can differentiate between Title and Footnote data and Document data.

### Step 7: Concatenate Headers

This step could be optional. In this step, different column headings for same column are concatenated together into one row where different data rows are separated by a delimiter="|". The purpose behind it is it makes easy to read the data and it is easy to replicate for the validator to replicate one header row rather than multiple header rows.

| pageid | tblid | H1 | | H6 | H7 |
|---|---|---|---|---|---|
| 1 | 2 | System Organ Class | Preferred Term | Total (N=12)\|Events, n | Total (N=12)\|Patients, n (%) |
| 2 | 4 | System Organ Class | Preferred Term | Total (N=12)\|Events, n | Total (N=12)\|Patients, n (%) |
| 3 | 6 | System Organ Class | Preferred Term | Total (N=12)\|Events, n | Total (N=12)\|Patients, n (%) |

**Figure 1. Multiple column headers for same row concatenated together**

### Step 8: Extract Unique Titles and Footnotes into a SAS dataset and create a TF_id.

Create Title and Footnote (T&F) dataset with unique Titles and Footnotes and initialize the tf_id counter whose value increases with any change in either title or footnote.

| tf_id | type | count | value |
|---|---|---|---|
| 1 | TITLE | 1 | AGAL |
| 1 | TITLE | 2 | Table 2 Patients with Treatment-Emergent Adverse Events by MedDRA SOC and Preferred Term Safety Set |
| 1 | FOOTNOTE | 1 | Note: If a patient had more than one event for a particular SOC, he/she is counted only once for that SOC. Note: If a patient had more than one event for a particular PT, he/she is counted only once for that PT. Note: Patient percentages are based on the total number of treated patients in the particular treatment group. |
| 1 | FOOTNOTE | 2 | Names of input datasets: RAWDATA.AEX, RAWDATA.DMX, RAWDATA.EXX and RAWDATA.IEX |

**Figure 2. Unique Titles and Footnotes**

### Step 9: Generate Output Datasets

Finally create the horizontal and vertical data structure by re-arranging the variables.

#### Horizontal Dataset

It is very similar to the input RTF file. It contains the following columns.

1. TF_ID: This variable value will increase incase there is a change in any title or footnote value.
2. HD_ID: This variable value will increase incase there is a change in any Column header value.
3. H1-Hn: These variable(s) will contain the column header values, where multiple column headers value/column will be separated by specified/DEFAULT DELIMITER.

| tf_id | hd_id | H1 | H2 | H5 | C1 | c2 | c5 |
|---|---|---|---|---|---|---|---|
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Patients with Events | 106 ( 99) | 322 ( 100) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Nervous system disorders | 81 ( 76) | 265 ( 82) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Headache | 27 ( 25) | 179 ( 55) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Paraesthesia | 18 ( 17) | 71 ( 22) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Hypoaesthesia | 21 ( 20) | 69 ( 21) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Multiple sclerosis relapse | 30 ( 28) | 63 ( 20) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Dysgeusia | 22 ( 21) | 59 ( 18) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Dizziness | 11 ( 10) | 48 ( 15) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Sensory disturbance | 8 ( 7) | 21 ( 7) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Multiple sclerosis | 7 ( 7) | 19 ( 6) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Migraine | 3 ( 3) | 17 ( 5) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Ataxia | 12 ( 11) | 15 ( 5) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Balance disorder | 5 ( 5) | 15 ( 5) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Burning sensation | 5 ( 5) | 13 ( 4) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Tremor | 4 ( 4) | 13 ( 4) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Muscle spasticity | 7 ( 7) | 11 ( 3) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Cognitive disorder | 3 ( 3) | 10 ( 3) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Paraparesis | 5 ( 5) | 10 ( 3) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Somnolence | 1 ( 1) | 10 ( 3) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Hemiparesis | 4 ( 4) | 9 ( 3) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Dysarthria | 4 ( 4) | 8 ( 2) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Nystagmus | 4 ( 4) | 8 ( 2) |
| 1 | 1 | [Body System or Organ Class Dictionary-Derived Term | TRT 1\|N=107\|Patients n (%) | Overall\|N=323\|Patients n (%) | Carpal tunnel syndrome | 2 ( 2) | 7 ( 2) |

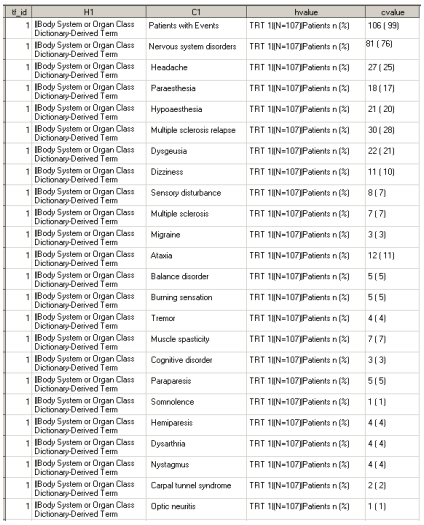
**Figure 3. Horizontal Data Structure**

4. C1-Cn: These variable(s) will contain the column data as contained in the output file. The value of the columns specified by the Group_Column Range will be retained, in case there is any grouped value.

The hvalue column is sorted by variables in order based on their representation in the output file.

**Vertical Dataset**

The horizontal dataset is fine when all columns in the table fit across the page. The problem with this structure is when a table has too many columns to fit on a page. When this happens, second column on the page has one header for the first X pages, then another header for the rest. For the validator, there is no way to know how many columns fit on the page nor is it correct to assume that the number of columns on the page is static. To allow the use of RTF_READ for all QC the Vertical Dataset structure was developed. This structure can be thought of as a normalized version of the horizontal dataset. Think of transposing the column headers down by the first (c1) column on the page.



| tf_id | H1 | C1 | hvalue | cvalue |
|---|---|---|---|---|
| 1 | [Body System or Organ Class Dictionary-Derived Term | Patients with Events | TRT 1\|N=107\|Patients n (%) | 106 ( 99) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Nervous system disorders | TRT 1\|N=107\|Patients n (%) | 81 ( 76) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Headache | TRT 1\|N=107\|Patients n (%) | 27 ( 25) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Paraesthesia | TRT 1\|N=107\|Patients n (%) | 18 ( 17) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Hypoaesthesia | TRT 1\|N=107\|Patients n (%) | 21 ( 20) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Multiple sclerosis relapse | TRT 1\|N=107\|Patients n (%) | 30 ( 28) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Dysgeusia | TRT 1\|N=107\|Patients n (%) | 22 ( 21) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Dizziness | TRT 1\|N=107\|Patients n (%) | 11 ( 10) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Sensory disturbance | TRT 1\|N=107\|Patients n (%) | 8 ( 7) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Multiple sclerosis | TRT 1\|N=107\|Patients n (%) | 7 ( 7) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Migraine | TRT 1\|N=107\|Patients n (%) | 3 ( 3) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Ataxia | TRT 1\|N=107\|Patients n (%) | 12 ( 11) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Balance disorder | TRT 1\|N=107\|Patients n (%) | 5 ( 5) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Burning sensation | TRT 1\|N=107\|Patients n (%) | 5 ( 5) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Tremor | TRT 1\|N=107\|Patients n (%) | 4 ( 4) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Muscle spasticity | TRT 1\|N=107\|Patients n (%) | 7 ( 7) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Cognitive disorder | TRT 1\|N=107\|Patients n (%) | 3 ( 3) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Paraparesis | TRT 1\|N=107\|Patients n (%) | 5 ( 5) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Somnolence | TRT 1\|N=107\|Patients n (%) | 1 ( 1) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Hemiparesis | TRT 1\|N=107\|Patients n (%) | 4 ( 4) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Dysarthria | TRT 1\|N=107\|Patients n (%) | 4 ( 4) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Nystagmus | TRT 1\|N=107\|Patients n (%) | 4 ( 4) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Carpal tunnel syndrome | TRT 1\|N=107\|Patients n (%) | 2 ( 2) |
| 1 | [Body System or Organ Class Dictionary-Derived Term | Optic neuritis | TRT 1\|N=107\|Patients n (%) | 1 ( 1) |

**Figure 4. Vertical Data Structure**

The structure of this dataset is one record per c1 value, per hvalue
There are multiple advantages to this structure

- If columns do not fit horizontally across the page, they could be transposed to a vertical structure and can be easily validated.
- It has less columns
- Easy to validate

## ADVANTAGES:

1. It will work for any RTF file generated by SAS
2. Its comprehensive approach lets user generate a SAS dataset which can be easily be created by the validator their by helping to speed up the validation process.
3. It is designed to handle spanning headers
4. it co-relates the data on different pages with column header id (hd_id) and title and footnote id (tf_id) which can identify any change in the title and footnotes or column headers.
5. It generates both horizontal data which is similar to that of the output file and vertical data which is much easier to validate.
6. It separates Title and footnotes data from document data
7. This process of decoding RTF Files eases the validation process and saves lot of times for multiple rounds of validation if required.

## CONCLUSION

This paper simplifies the process of understanding RTF file and converting them into SAS dataset which otherwise is a difficult process. Hopefully you will be able to leverage the information presented in this paper to help automate QC within your organization.

## REFERENCE


- **RTF Pocket Guide** By Sean Burke
  Publisher: O'Reilly Media  Released: July 2003

- **SAS RTF Content**
  **Website: http://support.sas.com/rnd/base/ods/templateFAQ/Template_rtf.html**


## ACKNOWLEDGEMENT


We would like to thank Vijaya (Vijaya.Raut@cognizant.com), Geeta (Geeta.Nagwekar@cognizant.com) and Ruchita (Ruchita.Srivastav@cognizant.com) from Cognizant Validation team, Mumbai, India who played a vital role in validation and deployment of this utility


## CONTACT INFORMATION


**Sandeep Juneja**
8000 Regency Pkwy, Suite 360
Cary, NC 27518
Phone:919-653-3982
Email: sjuneja@Ockham.com

**Daniel Boisvert / Andrew Illidge**
500 Kendall Street
Cambridge, MA 02142
Phone: 617-768-6061 / 4452
Email: Daniel.Boisvert@genzyme.com
Email: Andrew.Illidge@genzyme.com