PharmaSUG China 2017 - Paper 64

How to read RTF files into SAS® datasets?

Chunpeng Zhao, Boehringer-Ingelheim, Shanghai, China

ABSTRACT

There are two ways to develop summary tables and listings for SAS® programmers. One is through SAS® ODS. The other is through SAS® LISTING (i.e. generate SAS® LISTING outputs first. Then convert SAS® LISTING outputs into RTF or PDF files).

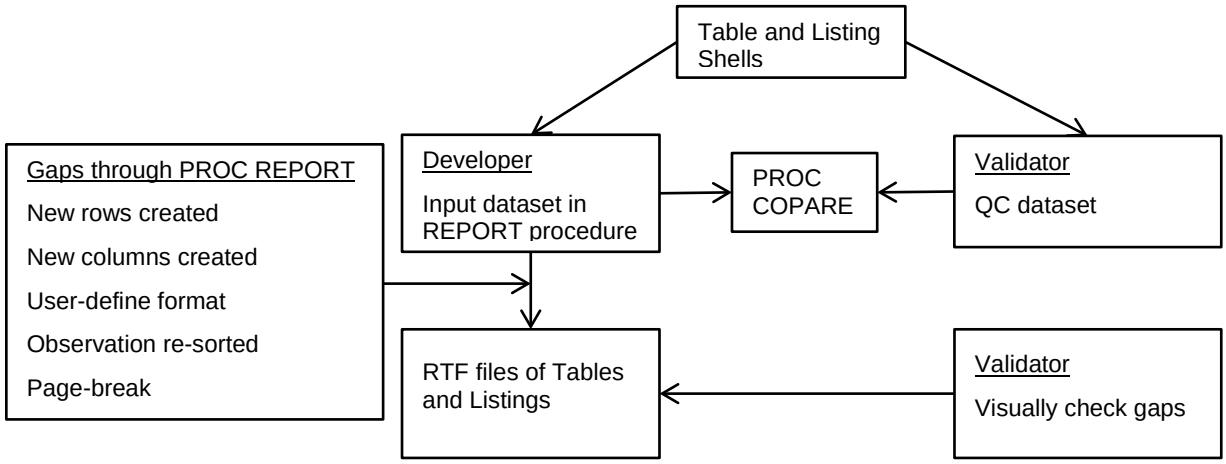
For summary tables and listings generated through SAS® ODS, Michiel Hagendoorn etc. gave solutions to read RTF files into SAS® datasets in the paper “Save Those Eyes: A Quality-Control Utility for Checking RTF Output Immediately And Accurately” already.

For summary tables and listings generated through SAS® LISTING, how read the data from an indicated RTF file into a SAS® data set? This pager will provide a solution.

INTRODUCTION

There are two ways to develop summary tables and listings for SAS® programmers. One is through SAS® ODS. The other is through SAS® LISTING (i.e. generate SAS® LISTING outputs first. Then convert SAS® LISTING outputs into RTF or PDF files). In RFT files created through SAS® ODS, data can be selected by column, but not in RFT files created through SAS® LINSTING.

During validation, commonly input datasets in REPORT procedure are compared with independently generated Quality Control datasets via the COMPARE procedure. Then visually check cosmetic format of summary tables’ and listings’ layouts.

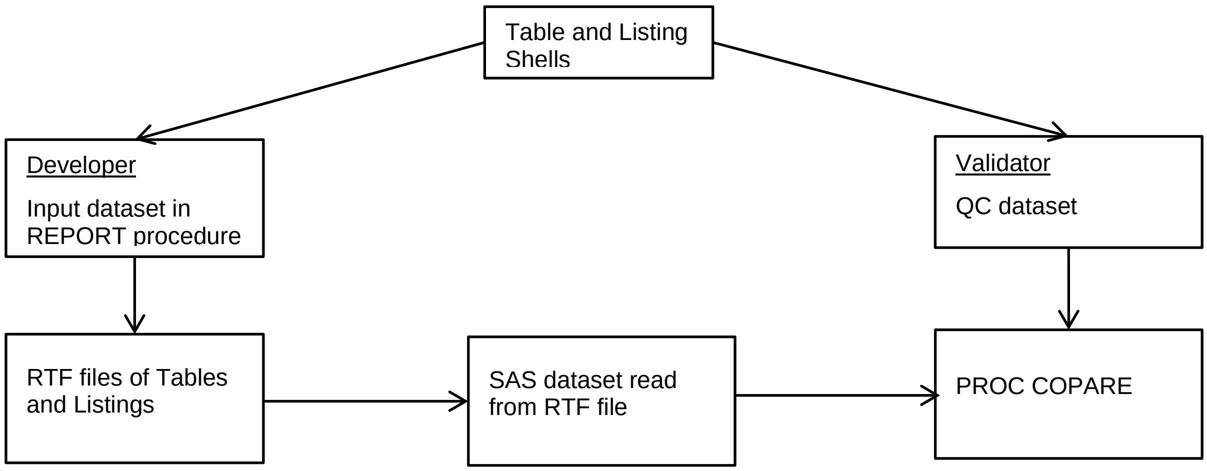


Actually there are gaps from input datasets in REPORT procedure to summary tables and listings. New statistics columns or rows can be created through REPORT procedure. User-defined formats can be assigned to variables in REPORT procedure. Observations could be re-sorted by sorted variables through REPORT procedure. And Observations will be split into different pages by page-break-variables in REPORT procedure. So input data used in REPORT procedure could be different from that in summary tables and listings.

Usually an agreement about input dataset used in REPORT procedure will be made between developers and validators. I.e. the names of variables corresponding to columns in summary tables and listings will be specified. The values of variables and the order of observations should be exactly same as that in summary tables and listings. It requires that some functions are not allowed to be used in REPORT procedure, e.g. user-defined formats in DEFINE statement, Summary and Statistics for variables, etc. And communication to get the agreement also consumes extra effort of both developers and validators. Even if the agreement is followed, it is still difficult to find some discrepancy issues via COMPARE procedure, which are generated in REPORT procedure, e.g. observation sorted issue, and page split issue (i.e. incorrect page-break variable used) etc. Manual check work on these discrepancies will be unavoidable.

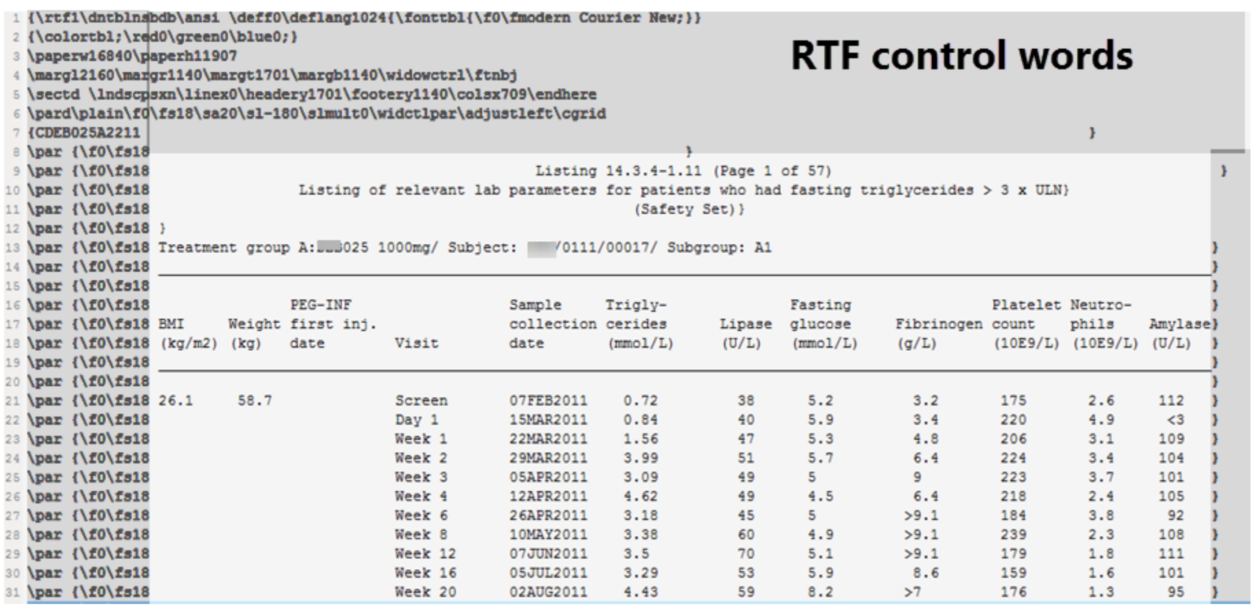
For validation, it is a better way to focus on final summary tables and listings instead of intermediate input dataset in REPORT procedure. I.e. read RTF files into SAS® datasets. Then compare these SAS® datasets with independently

generated Quality Control datasets via the COMPARE procedure. In this way, communication time between developers and validators on the data structure of input dataset in REPORT procedure, and validators' manual check work on gaps from input dataset in REPORT procedure to RTF file will be saved.



For summary tables and listings generated through SAS® ODS, Michiel Hagendoorn etc. gave solutions to read RTF files into SAS® datasets in the paper "Save Those Eyes: A Quality-Control Utility for Checking RTF Output Immediately And Accurately". For summary tables and listings generated through SAS® LISTING, how read the data from an indicated RTF file into a SAS® data set? This pager will provide a solution.

Summary tables and listings are generated through SAS® LISTING. Then SAS® LISTING outputs are converted into RTF files by adding RTF control words. The control words are visible as below figure when the RTF file is viewed in a text editor like Notepad on Windows. Unlike RTF files directly generated through SAS® ODS, for SAS® LISTING converted RTF files, there are not RTF control words to identify the page, the title, the table header, the body of the table, the rows of the table, the cells of the table, and the footnotes. It is a real challenge to read SAS® LISTING converted RTF files into SAS® datasets.




```
{\rtf1\dntblnsbdb\ansi \deff0\deflang1024{\fonttbl{\f0\fmodern Courier New;}}
{\colortbl;\red0\green0\blue0;}
\paperw16840\paperh11907
\margl2160\margr1140\margt1701\margb1140\widowctrl\ftnbj
\sectd \lndscpsxn\linex0\headery1701\footery1140\colsx709\endhere
\pard\plain\f0\fs18\sa20\sl-180\slmult0\widctlpar\adjustleft\cgrid
{CDEB025A2211                                                                  }
\par {\f0\fs18                                             }
\par {\f0\fs18                         Listing 14.3.4-1.11 (Page 1 of 57)                                }
\par {\f0\fs18         Listing of relevant lab parameters for patients who had fasting triglycerides > 3 x ULN}
\par {\f0\fs18                                   (Safety Set)}
\par {\f0\fs18 }
\par {\f0\fs18 Treatment group A:[redacted]025 1000mg/ Subject: [redacted]/0111/00017/ Subgroup: A1                       }
\par {\f0\fs18 ___________________________________________________________________________________________}
\par {\f0\fs18                                                                                            }
\par {\f0\fs18                PEG-INF                 Sample     Trigly-             Fasting                 Platelet Neutro-           }
\par {\f0\fs18 BMI     Weight first inj.              collection cerides      Lipase  glucose    Fibrinogen count    phils    Amylase}
\par {\f0\fs18 (kg/m2) (kg)   date       Visit        date       (mmol/L)     (U/L)   (mmol/L)   (g/L)      (10E9/L) (10E9/L) (U/L)  }
\par {\f0\fs18 ___________________________________________________________________________________________}
\par {\f0\fs18                                                                                            }
\par {\f0\fs18 26.1     58.7             Screen       07FEB2011    0.72         38     5.2          3.2      175       2.6      112  }
\par {\f0\fs18                           Day 1        15MAR2011    0.84         40     5.9          3.4      220       4.9       <3  }
\par {\f0\fs18                           Week 1       22MAR2011    1.56         47     5.3          4.8      206       3.1      109  }
\par {\f0\fs18                           Week 2       29MAR2011    3.99         51     5.7          6.4      224       3.4      104  }
\par {\f0\fs18                           Week 3       05APR2011    3.09         49     5            9        223       3.7      101  }
\par {\f0\fs18                           Week 4       12APR2011    4.62         49     4.5          6.4      218       2.4      105  }
\par {\f0\fs18                           Week 6       26APR2011    3.18         45     5           >9.1      184       3.8       92  }
\par {\f0\fs18                           Week 8       10MAY2011    3.38         60     4.9         >9.1      239       2.3      108  }
\par {\f0\fs18                           Week 12      07JUN2011    3.5          70     5.1         >9.1      179       1.8      111  }
\par {\f0\fs18                           Week 16      05JUL2011    3.29         53     5.9          8.6      159       1.6      101  }
\par {\f0\fs18                           Week 20      02AUG2011    4.43         59     8.2         >7        176       1.3       95  }
```

What information is needed to be read from RTF files into SAS® datasets for validation? Let's focus on the body of the table and subgroups, which are located between the title and the table header commonly. RTF control words are invisible in Microsoft Word®. The title, the table header and the footnotes are exactly same across pages, which can be visually compared to the shells of summary tables and listings.

SAS® LISTING converted RTF files follow the same hierarchy structure as RTF files directly created by SAS® ODS, i.e. the page, subgroups of sections, the body of the table, the rows of the table and the column of the table, although

there is not corresponding RTF control words to identify them in SAS® LISTING converted RTF files. If there is a way to identify the hierarchy structure in SAS® LISTING converted RTF files, RTF files can be read into SAS® datasets.

IDENTIFYING THE TITLE, THE TABLE HEADER AND THE FOOTNOTES

The title, the table header and the footnotes are exactly same and located at the same rows across pages in a RTF file. If a row has the same contents across pages in a RTF file, this row will be assumed as a part of the title, the table header and the footnotes. Only variant parts across pages will be read into SAS® datasets. Invariant part across pages can be visually checked.

IDENTIFYING COLUMN TEXT

Read a RTF file as a plain text file into a SAS® dataset. Remove RTF control words. Remove page numbers. Remove subgroups. Remove the title, the table header and the footnotes. Only keep the body of the table. Then try to split each row into different columns. Columns are segregated by blank characters. If vertical bars consisting of blank characters, runs through from the first page until the last page, then each row will be split into different columns.

PROGRAM FLOW AND DETAILS

Logic steps are roughly outlined as follows.

1. Read a SAS® LISTING converted RTF file as a plain text file into a SAS® dataset.
2. Remove RTF control words.
3. Extract page numbers into a variable.
4. Remove the title, the table header and the footnotes, which are exactly same and located at the same rows across pages.
5. Extract subgroups into category variables.
6. Split the table into columns.

The sections below provide a step-by-step review of the macro code. Presenting the entire code is beyond the scope of this paper. The code segments discussed here were selected to provide a sense of how to read RTF files into a SAS® data set that is ready for a PROC COMPARE.

1. Read a RTF file into a SAS® dataset.

Read a RTF file as a plain text file into a SAS® dataset.

```
filename rtf "&filename";
infile rtf missover length = l lrecl = 2000;
input STRING $varying2000. l;
```

2. Remove RTF control words.

```
  if index(STRING,'\par {\f0\fs18')>0;
  TEXT=substr(STRING, 16);
  if substr(TEXT, lengthn(TEXT))="}" then
    substr(TEXT, lengthn(TEXT))=" ";
```

3. Extract page numbers into a variable.

```
   PATID = prxparse("/\(Page\s*\d*\s*of\s*\d*\)/");
   PAGE=input(scan(substr(TEXT, prxmatch(PATID,TEXT)+1), 2), best.);
```

4. Remove the title, the table header and the footnotes.

/*Read the Row Number and Content of each row*/

```
proc sql noprint;
  create table PAGE1 as
    select LINE_N, TEXT
      from rtf_1
      where PAGE=1;
```

/*If a row appear in all pages, the sum of the page number with the row should be equal a constant, i.e. the sum of all page numbers in the RTF file */

```
  create table PAGE1_1 as
    select a.LINE_N, a.TEXT, sum(distinct b.PAGE) as SUM_P
      from page1 as a
        left join rtf_1 as b
        on a.LINE_N=b.LINE_N and
          a.TEXT=b.TEXT
      group by a.LINE_N, a.TEXT;

  create table rtf_2 as
    select a.*, SUM_P
      from rtf_1 as a
        left join PAGE1_1 as b
        on a.LINE_N=b.LINE_N and
          a.TEXT=b.TEXT
      where SUM_P not in (select sum(distinct PAGE) from rtf_1) and
        a.TEXT ne " "
      ;
quit;
```

5. Extract subgroups into category variables.

/*Different subgroups are put into different variables (RETAIN_STR&i) in dataset retain_str. And merge it with RTF dataset. Then retain subgroup values. &SPLIT_NUM is number of subgroups */

```
  %do i=1 %to &SPLIT_NUM;
    %let LENGTH_STR=&LENGTH_STR RETAIN_STR&i;
  %end;
proc sql;
  create table rtf_3 as
    select a.*, %sysfunc(tranwrd(%sysfunc(compbl(&LENGTH_STR)), %str( ), %str(,)))
      from rtf_2 as a
        %do i=1 %to &SPLIT_NUM;
          left join (select distinct RETAIN_STR&i
                       from retain_str
                       where not missing(RETAIN_STR&i)) as b
          on left(a.TEXT)=left(b.RETAIN_STR&i)
        %end;
      order by N;
quit;

data rtf_4;

length %sysfunc(tranwrd(%sysfunc(compbl(&LENGTH_STR)), %str(RETAIN_STR), %str(CAT)))
$200;

retain %sysfunc(tranwrd(%sysfunc(compbl(&LENGTH_STR)), %str(RETAIN_STR), %str(CAT))
);
  set rtf_3;
  %do i=1 %to &SPLIT_NUM;
    if not missing(RETAIN_STR&i) then
      do;
        CAT&i=RETAIN_STR&i;
        delete;
      end;
  %end;
run;
```

6. Split the table into columns.

/*Suppose the linesize=120. Read the body of table into 120 columns. Each column contains 1 character. Retain 120 variables. If some variables' values are still missing, then these columns will be blank vertical bars run through all rows. Each row will be split by these blank vertical bars into different columns*/

```
data rtf_5;
  length COL_NULL $480;
  retain COL1-COL120;
  array COL{120} $1 COL1-COL120;
  set rtf_4 end=lastobs;

  do i=1 to 120;
    if COL(i)<substr(TEXT, i, 1) then
      COL(i)=substr(TEXT, i, 1);

    if lastobs then
      do;
        if missing(COL(i)) then
          COL_NULL=catx(",", COL_NULL, i);
      end;
  end;

  if lastobs then
    call symputx("COL_NULL", COL_NULL);
run;

data rtf_6;
  length TEXT_NEW $120;
  array COL{120} $1 COL1-COL120;
  retain COL_NUM 0;
  set rtf_5 end=lastobs;
  do i=1 to 120;
    COL(i)=substr(TEXT, i, 1);
  end;

  do i=&COL_NULL;
    COL(i)="|";
  end;

  TEXT_NEW=cat(of COL1-COL120);

  TEXT_LEN=length(TEXT_NEW);

  T_NUM=count(compbl(tranwrd(tranwrd(TEXT_NEW, " ", "00"x), "|", " ")), " ")
    -(substr(TEXT_NEW, length(TEXT_NEW), 1)="|" and substr(TEXT_NEW, 1, 1)="|")
    +(substr(TEXT_NEW, length(TEXT_NEW), 1) ne "|" and substr(TEXT_NEW, 1, 1) ne
"|");
  if COL_NUM<T_NUM then
    COL_NUM=T_NUM;

  if lastobs then
    call symputx("COL_NUM", COL_NUM);
run;

data &outds;
  array COLS{&COL_NUM} $120;
  set rtf_6;
  COL_NUM= &COL_NUM;
  CAT_NUM= &SPLIT_NUM;
  do i=1 to &COL_NUM;
    COLS(i)=scan(TEXT_NEW, i, "|");
  end;
  keep PAGE LINE_N CAT: COLS: TEXT: COL_NUM
  ;
run;
```

THE REPORT CONTAINS MORE COLUMNS THAN FITS ON ONE PAGE

When the report contains more columns than fits on one page, the 1st pages of all rows can be read firstly, then the 2nd ones, 3rd ones…. Different pages containing different columns are read into different SAS® datasets. Then merge them into one SAS® dataset by ID columns whose values are able to identify unique rows.

SUBSEQUENT WOK

After a RTF file is read into a SAS® dataset, some subsequent work is needed as follows.

Populate Order variables' values

Value of an order variable should be populated repeatly from one row to the next until the value changes.

Wrap multiple rows into one

When value of a variable is too long, this value will flow into multiple rows. These multilple rows are needed to be wrapped back into one row in SAS® dataset.

CONCLUSION

As shown in the paper, SAS® LINSTING converted RTF files can be read back into SAS® datasets standalone, without referring developer's SAS® code. Then validators are able to focus on final RTF files. Communication time between developers and validators on the data structure of input dataset in REPORT procedure, and validators' manual check work on gaps from input dataset in REPORT procedure to RTF file will be saved.

REFERENCES

Michiel Hagendoorn, Jonathan Squire and Johnny Tai "Save Those Eyes: A Quality-Control Utility for Checking RTF Output Immediately And Accurately" http://www2.SAS®.com/proceedings/sugi31/066-31.pdf

CONTACT INFORMATION

Your comments and questions are valued and encouraged. The SAS® code can be sent upon request. Contact the author at:

Name: Chunpeng Zhao
Enterprise: Boehringer Ingelheim
Address: 29/F, Park Place 1601 Nanjing Road (West)
City, State ZIP: Shanghai, China
Work Phone: +86 (21) 5288-0209
E-mail: chunpeng.zhao@boehringer-ingelheim.com