9320

Importing Data Directly from PDF into SAS® Data Sets

William Wu, Puma Biotechnology, Inc., South San Francisco, CA
Steven Li, Medtronic Inc., Minneapolis, MN
Yun Guan, Puma Biotechnology, Inc., South San Francisco, CA

ABSTRACT

Creating electronic documents in PDF file format is becoming increasingly popular nowadays. Combining ODS PDF statement and the REPORT procedure in SAS can create various PDF output files with different styles. During the process of validating those data in PDF file, there is demand to import PDF summary tables or listings into SAS datasets. A utility is developed which reads in a SAS generated uncompressed PDF file, extracts and converts the data from the PDF file into a SAS datasets. The overview of this utility is presented in this paper.

INTRODUCTION

PDF stands for "portable document format". It was introduced to ease the sharing of printable documents between computers across operating system platforms when one needs to save files that cannot be modified. For the detail, see reference section <PDF Reference, version 1.7 - Adobe>. In the biotech and many other industries, RTF and

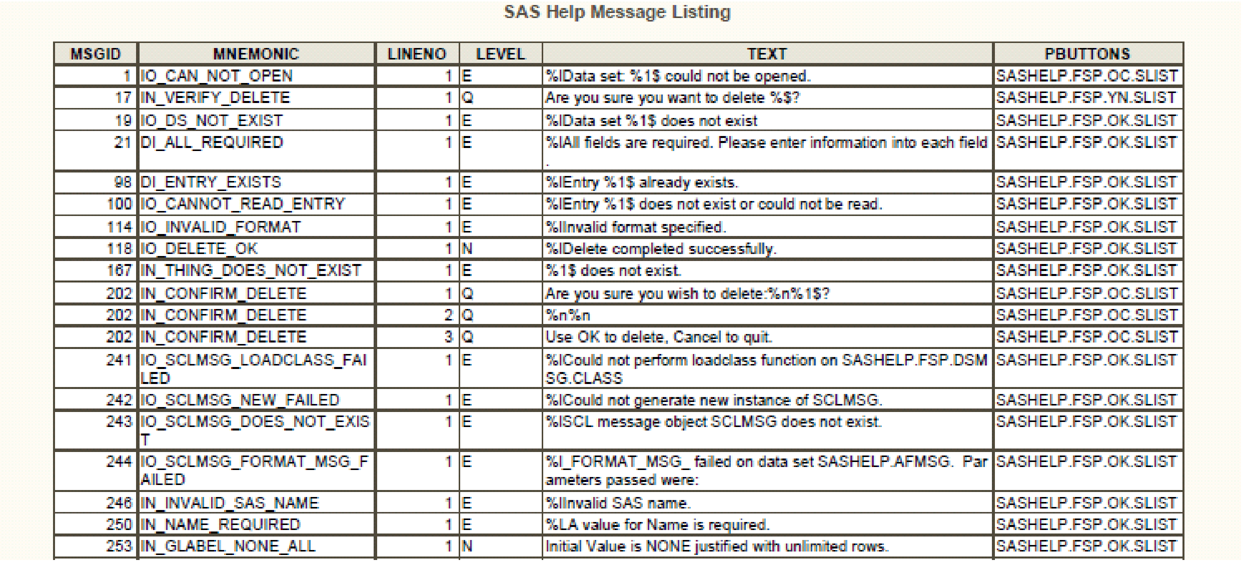
PDF output files have been used extensively to present reports and analysis. Many involve importing RTF data into SAS datasets but not much has been done for PDF data due to raised level of complexity and difficulty in parsing PDF formats.

By default, statement ODS PDF usually generates a compressed PDF file with default setting ‘compress=6’. This paper talks about importing SAS generated uncompressed PDF (‘compress=0’) output into SAS data set by the utility macro %PDF2SAS which is introduced in the following sections. This macro reads in the whole uncompressed PDF file as strings using INFILE and INPUT statements in a DATA step. Perl Regular Expression function separates and extracts targeted information, including text strings and attributes for each cell. The final step is to identify each cell in terms of the row and column locations and concatenate multiple wrapped text strings together to form a complete cell when needed. %PDF2SAS is a light weight macro which imports PDF data to a SAS data set quickly and efficiently.

OVERVIEW OF THE PROGRAM

SAMPLE TABLE

An uncompressed PDF file can be easily generated in statement ODS PDF with option COMPRESS=0 which is shown on display 1. In statement ODS PDF, a style option can be set as: STYLE=<style name>. In Appendix1, the SAS code demonstrates how to create an uncompressed PDF file. Following is the output PDF file “afmsg_Analysis.pdf”:



SAS Help Message Listing

MSGID	MNEMONIC	LINENO	LEVEL	TEXT	PBUTTONS
1	IO_CAN_NOT_OPEN	1	E	%IData set: %1$ could not be opened.	SASHELP.FSP.OC.SLIST
17	IN_VERIFY_DELETE	1	Q	Are you sure you want to delete %$?	SASHELP.FSP.YN.SLIST
19	IO_DS_NOT_EXIST	1	E	%IData set %1$ does not exist	SASHELP.FSP.OK.SLIST
21	DI_ALL_REQUIRED	1	E	%IAll fields are required. Please enter information into each field.	SASHELP.FSP.OK.SLIST
98	DI_ENTRY_EXISTS	1	E	%IEntry %1$ already exists.	SASHELP.FSP.OK.SLIST
100	IO_CANNOT_READ_ENTRY	1	E	%IEntry %1$ does not exist or could not be read.	SASHELP.FSP.OK.SLIST
114	IO_INVALID_FORMAT	1	E	%IInvalid format specified.	SASHELP.FSP.OK.SLIST
118	IO_DELETE_OK	1	N	%IDelete completed successfully.	SASHELP.FSP.OK.SLIST
167	IN_THING_DOES_NOT_EXIST	1	E	%1$ does not exist.	SASHELP.FSP.OK.SLIST
202	IN_CONFIRM_DELETE	1	Q	Are you sure you wish to delete:%n%1$?	SASHELP.FSP.OC.SLIST
202	IN_CONFIRM_DELETE	2	Q	%n%n	SASHELP.FSP.OC.SLIST
202	IN_CONFIRM_DELETE	3	Q	Use OK to delete, Cancel to quit.	SASHELP.FSP.OC.SLIST
241	IO_SCLMSG_LOADCLASS_FAILED	1	E	%ICould not perform loadclass function on SASHELP.FSP.DSMSG.CLASS	SASHELP.FSP.OK.SLIST
242	IO_SCLMSG_NEW_FAILED	1	E	%ICould not generate new instance of SCLMSG.	SASHELP.FSP.OK.SLIST
243	IO_SCLMSG_DOES_NOT_EXIST	1	E	%ISCL message object SCLMSG does not exist.	SASHELP.FSP.OK.SLIST
244	IO_SCLMSG_FORMAT_MSG_FAILED	1	E	%I_FORMAT_MSG_ failed on data set SASHELP.AFMSG. Parameters passed were:	SASHELP.FSP.OK.SLIST
246	IN_INVALID_SAS_NAME	1	E	%IInvalid SAS name.	SASHELP.FSP.OK.SLIST
250	IN_NAME_REQUIRED	1	E	%LA value for Name is required.	SASHELP.FSP.OK.SLIST
253	IN_GLABEL_NONE_ALL	1	N	Initial Value is NONE justified with unlimited rows.	SASHELP.FSP.OK.SLIST

Display 1: Uncompressed output PDF file which is created by ODS PDF and PROC REPORT.

To see how different the PDF file’s format is, let’s open it in NOTEPAD which is a basic text editor in Windows OS. The following shows the format of cell with “IO_CAN_NOT_OPEN” in the table.

In PDF file afmsg_Analysis.pdf:

“BT /TT2 9.5 Tf 86.88 527.52 Td (IO_CAN_NOT_OPEN)Tj ET”

We can see that the ‘IO_CAN_NOT_OPEN’ is embedded in some numbers and characters. Those numbers and characters represent the attributes of the cell ‘IO_CAN_NOT_OPEN ‘.

The text ‘IO_CAN_NOT_OPEN is placed at 527.52 points (7.33 inches) from the bottom of the page and 86.88 points (1.21 inches) from the left edge, using 9.5 point Arial font. Point is a unit and usually used in typography, computers font sizes and printing as the smallest unit. The abbreviation is "pt". 1 Inch = 72 Points.

A typical format for text in PDF is as follows:

1. BT: Begin a text object.

2. /TT2 9.5 Tf: Set the font and font size to use, installing them as parameters in the text state. The font size is 9.5 point. The font resource identified by the name TT2 specifies the font externally known as Arial, according to the following string inside of the file: obj<</Type/Font/Subtype/TrueType/Name/TT2/BaseFont/CCUAAB+Arial,Regular/…

3. 86.88 527.52 Td: Specify a starting position on the page, setting parameters in the text object.

4. (IO_CAN_NOT_OPEN)Tj: Paint the glyphs for a string of characters at that position.

5. ET: End the text object.

Therefore, when reading in the whole PDF file the first and most important step is to extract the useful text and attributes from the long codes into SAS data.

PROGRAM FLOW AND DETAILS

Now, we start to introduce the code of the macro. The interface of the macro is very simple. There are 2 input parameters – ‘datafile’ and ‘out’.

```
%macro pdf2sas(
  datafile = /* Specifies the complete path and filename for the input PDF file     */
, out      = /* Identifies the output SAS data set with a one or two-level SAS name */
);
```

The program flow of the pdf2sas macro can roughly be outlined as follows:

1. Validate the parameter’s values and check if the input parameters are valid.

2. Read the PDF file into SAS and then extract each relevant text and attribute as a string.

3. Separate text and attributes from the string.

4. Obtain row and column number based on the text and their attributes.

5. Re-construct text from data set when the text is wrapped into a cell as multiple line

1. VALIDATING THE PARAMETER’S VALUES

In the beginning of the macro, the code checks the validity of input parameters. Namely the existence of the pdf file name and the lib reference of output SAS dataset.

```
%local _msg_ _lin_;
%let datafile = %scan(&datafile,1,.);

%if %sysfunc(fileexist(&datafile..pdf))=0 %then
      %let _msg_ = In parameter DATAFILE, PDF file "&datafile..pdf" does not
exist.;
%else %if %sysfunc(libref(%scan(%scan(&out,-2,.) work,1)))^=0 %then
%let _msg_ = In parameter OUT, libname "%scan(&out,1,.)" is not assigned.;
%if %length(&_msg_)>0 %then %do;
```

```
%ER:%let _lin_ = %sysfunc(repeat(=,%length(%bquote(&_msg_))+24));
      %put %str(ERR)OR: &_lin_;
      %put %str(ERR)OR: &_msg_ Macro stoped processing.;
      %put %str(ERR)OR: &_lin_;
      %return;
%end;
```

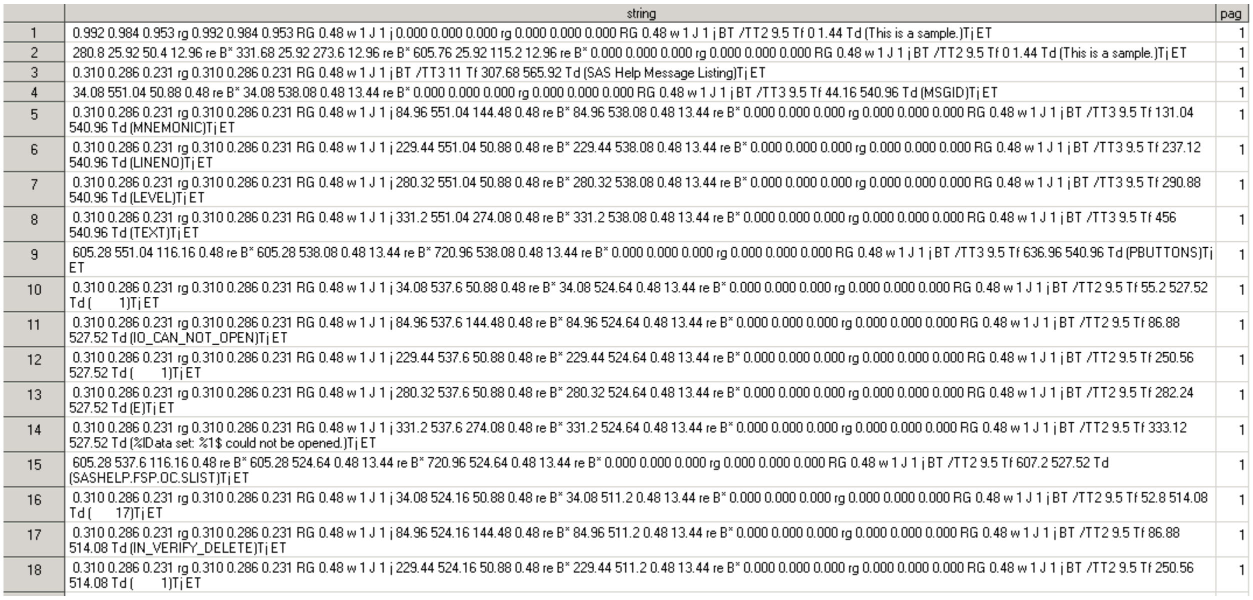
2. READ PDF FILE INTO SAS

At the first significant step, the macro reads the whole PDF file including the texts and the description of the PDF file format into a SAS dataset. Then it extracts each text field and useful features as a string. The code is as below.

```
%local _t_1 _t_2 _t_3 _rc_;
%let _t_1 = \d+\.?\d*;
%let _t_2 = ( (&_t_1) &_t_1 &_t_1 &_t_1 re B\*( \[\]0 d){0,2})?( &_t_1 &_t_1
&_t_1 rg &_t_1 &_t_1 &_t_1 RG( &_t_1 w( \[\]0 d)? \d J \d j( \[\]0 d)?)?)?;
%let _t_3 = &_t_2&_t_2&_t_2 BT\s?\/(TT|F)\d &_t_1 Tf (&_t_1) (&_t_1) Td
\((.*)\)Tj ET;
%let _rc_ = 4;
options mprint noquotelenmax varlenchk=nowarn ls=140;

%* Get values for variable String and Page *;
data _tmp_1(keep=string page _p:);
      length _S_0 $32767 string $1000;
      array _S_[&_rc_] $32000;
      retain _O_ 0 re page;
      if _N_=1 then re = prxparse("/&_t_3/i");
    infile "&datafile..pdf" truncover lrecl=%eval(32000*&_rc_) end=eof;
      input (_S_1-_S_&_rc_) ($char32000.);
      _i_  = 1;
      _p0_ = 768;
      substr(_S_0,_p0_) = _S_[_i_];
      _p1_ = 1;
      do until(_i_=&_rc_ & _p1_=0);
            if _i_<&_rc_ & (_p0_>32000 | _p1_=0) then do;
                  _i_ + 1;
                  _p0_ = ifn(_p0_>32000,_p0_-32000,1);
                  substr(_S_0,_p0_) = substr(_S_0,_p0_+32000)||_S_[_i_];
            end;
            _p1_ = find(_S_0,')Tj ET','i',_p0_);
            if _p1_=0 then continue;
            call prxnext(re,_p0_,_p1_+5,_S_0,_p1_,_L1_);
            if _p1_=0 then continue;
            if _N_>_O_ then page + 1;
            string = substr(_S_0,_p1_,_L1_);
            output;
            _O_ = _N_;
      end;
      if eof then call symputx('_O_',_O_,'L');
run;
%if &_O_=0 %then %do;
%let _msg_ = Input PDF file (Empty/Compressed/Cryptographic?)
"&datafile..pdf" is not recognized by macro.;
%goto ER;
%end;
```

In the code above, the INFILE and INPUT statements read the PDF file into a SAS data set. The record length is very long because one record is obtained for a whole page of the texts and the related description of the PDF file format so we use 4 strings(_S_[1] to _S_[4]) with length 32000 (total length 128000), then concatenate them and get useful texts and features. Here, the Perl Regular Expression function extracts the useful information into variable ‘string’ in output data set ‘_tmp_1’. Perl regular expression is a special text string for describing a search pattern**.** prxparse() and prxnext() are Perl Regular Expression function in SAS. Another variable ‘page’ is for the page number.

With the help of Perl Regular Expression as &_t_3 (in the code above), we extract the information into variable ‘string’. For input PDF file ‘afmsg_Analysis.pdf’, the SAS dataset ‘_tmp_1’ is as follows:



	string	pag
1	0.992 0.984 0.953 rg 0.992 0.984 0.953 RG 0.48 w 1 J 1 j 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT2 9.5 Tf 0 1.44 Td (This is a sample.)Tj ET	1
2	280.8 25.92 50.4 12.96 re B* 331.68 25.92 273.6 12.96 re B* 605.76 25.92 115.2 12.96 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT2 9.5 Tf 0 1.44 Td (This is a sample.)Tj ET	1
3	0.310 0.286 0.231 rg 0.310 0.286 0.231 RG 0.48 w 1 J 1 j BT /TT3 11 Tf 307.68 565.92 Td (SAS Help Message Listing)Tj ET	1
4	34.08 551.04 50.88 0.48 re B* 34.08 538.08 0.48 13.44 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT3 9.5 Tf 44.16 540.96 Td (MSGID)Tj ET	1
5	0.310 0.286 0.231 rg 0.310 0.286 0.231 RG 0.48 w 1 J 1 j 84.96 551.04 144.48 0.48 re B* 84.96 538.08 0.48 13.44 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT3 9.5 Tf 131.04 540.96 Td (MNEMONIC)Tj ET	1
6	0.310 0.286 0.231 rg 0.310 0.286 0.231 RG 0.48 w 1 J 1 j 229.44 551.04 50.88 0.48 re B* 229.44 538.08 0.48 13.44 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT3 9.5 Tf 237.12 540.96 Td (LINENO)Tj ET	1
7	0.310 0.286 0.231 rg 0.310 0.286 0.231 RG 0.48 w 1 J 1 j 280.32 551.04 50.88 0.48 re B* 280.32 538.08 0.48 13.44 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT3 9.5 Tf 290.88 540.96 Td (LEVEL)Tj ET	1
8	0.310 0.286 0.231 rg 0.310 0.286 0.231 RG 0.48 w 1 J 1 j 331.2 551.04 274.08 0.48 re B* 331.2 538.08 0.48 13.44 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT3 9.5 Tf 456 540.96 Td (TEXT)Tj ET	1
9	605.28 551.04 116.16 0.48 re B* 605.28 538.08 0.48 13.44 re B* 720.96 538.08 0.48 13.44 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT3 9.5 Tf 636.96 540.96 Td (PBUTTONS)Tj ET	1
10	0.310 0.286 0.231 rg 0.310 0.286 0.231 RG 0.48 w 1 J 1 j 34.08 537.6 50.88 0.48 re B* 34.08 524.64 0.48 13.44 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT2 9.5 Tf 55.2 527.52 Td (1)Tj ET	1
11	0.310 0.286 0.231 rg 0.310 0.286 0.231 RG 0.48 w 1 J 1 j 84.96 537.6 144.48 0.48 re B* 84.96 524.64 0.48 13.44 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT2 9.5 Tf 86.88 527.52 Td (IO_CAN_NOT_OPEN)Tj ET	1
12	0.310 0.286 0.231 rg 0.310 0.286 0.231 RG 0.48 w 1 J 1 j 229.44 537.6 50.88 0.48 re B* 229.44 524.64 0.48 13.44 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT2 9.5 Tf 250.56 527.52 Td (1)Tj ET	1
13	0.310 0.286 0.231 rg 0.310 0.286 0.231 RG 0.48 w 1 J 1 j 280.32 537.6 50.88 0.48 re B* 280.32 524.64 0.48 13.44 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT2 9.5 Tf 282.24 527.52 Td (E)Tj ET	1
14	0.310 0.286 0.231 rg 0.310 0.286 0.231 RG 0.48 w 1 J 1 j 331.2 537.6 274.08 0.48 re B* 331.2 524.64 0.48 13.44 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT2 9.5 Tf 333.12 527.52 Td (%lData set: %1$ could not be opened.)Tj ET	1
15	605.28 537.6 116.16 0.48 re B* 605.28 524.64 0.48 13.44 re B* 720.96 524.64 0.48 13.44 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT2 9.5 Tf 607.2 527.52 Td (SASHELP.FSP.OC.SLIST)Tj ET	1
16	0.310 0.286 0.231 rg 0.310 0.286 0.231 RG 0.48 w 1 J 1 j 34.08 524.16 50.88 0.48 re B* 34.08 511.2 0.48 13.44 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT2 9.5 Tf 52.8 514.08 Td (17)Tj ET	1
17	0.310 0.286 0.231 rg 0.310 0.286 0.231 RG 0.48 w 1 J 1 j 84.96 524.16 144.48 0.48 re B* 84.96 511.2 0.48 13.44 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT2 9.5 Tf 86.88 514.08 Td (IN_VERIFY_DELETE)Tj ET	1
18	0.310 0.286 0.231 rg 0.310 0.286 0.231 RG 0.48 w 1 J 1 j 229.44 524.16 50.88 0.48 re B* 229.44 511.2 0.48 13.44 re B* 0.000 0.000 0.000 rg 0.000 0.000 0.000 RG 0.48 w 1 J 1 j BT /TT2 9.5 Tf 250.56 514.08 Td (1)Tj ET	1

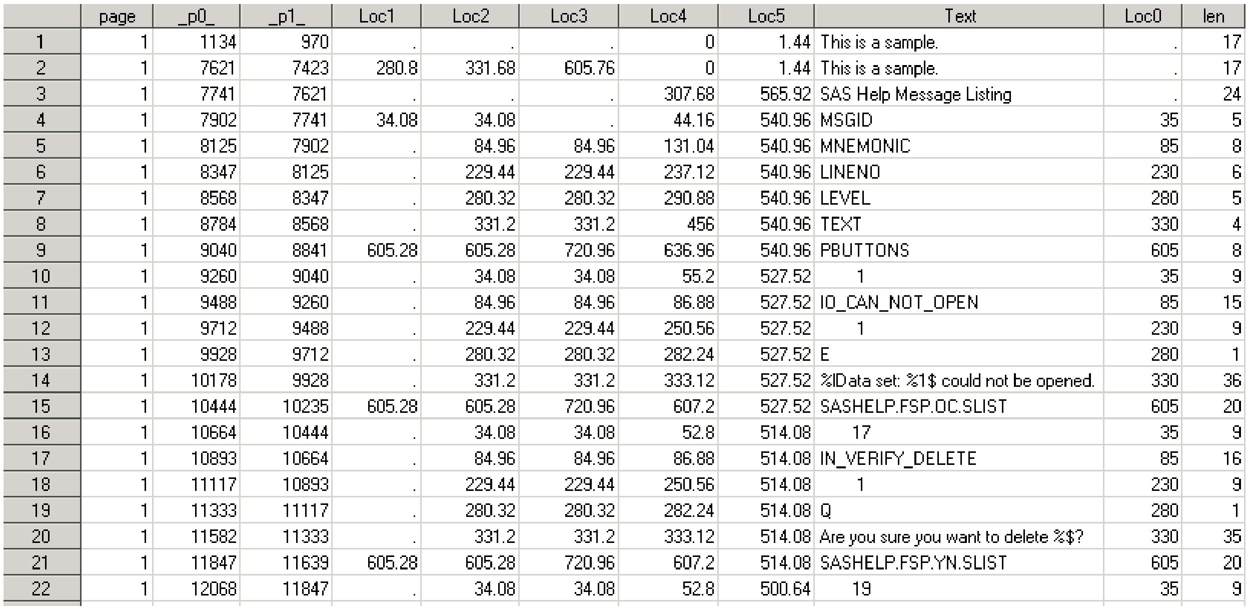
Display 2: In SAS data set ‘_tmp_1’, each record includes a text and formating information for each cell.

3. EXTRACT TEXT AND ATTRIBUTES

For the next step, we need to separate the text and each feature from the string. The following code can fulfill this goal.

```
%* Get Loc1-Loc5 for location/format info of the page and Text *;
data _tmp_2(drop=string i _Ps_);
      array _R_[6] _temporary_;
      if _N_=1 then do i=1 to 6;
            _R_[i] = 
prxparse(cat("s/&_t_3/$",choosen(i,2,9,16,23,24,25),"/"));
      end;
      set _tmp_1;
      by page;
      array Loc[5];
      length Text $400;
      do i=1 to 5; Loc[i] = input(prxchange(_R_[i],-1,string),best.); end;
      if n(Loc1,Loc2,Loc3)>0 then if min(Loc1,Loc2,Loc3)<Loc4 then Loc0 = 
round(min(Loc1,Loc2,Loc3),2.5);
      Text = prxchange(_R_[6],-1,string);
      call prxposn(_R_[6],25,_Ps_,len);

run;
```

Similar to extract useful info into variable ‘string’, the Perl Regular Expression function is still used for separating the text and each attributes.



	page	_p0_	_p1_	Loc1	Loc2	Loc3	Loc4	Loc5	Text	Loc0	len
1	1	1134	970	.	.	.	0	1.44	This is a sample.	.	17
2	1	7621	7423	280.8	331.68	605.76	0	1.44	This is a sample.	.	17
3	1	7741	7621	.	.	.	307.68	565.92	SAS Help Message Listing	.	24
4	1	7902	7741	34.08	34.08	.	44.16	540.96	MSGID	35	5
5	1	8125	7902	.	84.96	84.96	131.04	540.96	MNEMONIC	85	8
6	1	8347	8125	.	229.44	229.44	237.12	540.96	LINENO	230	6
7	1	8568	8347	.	280.32	280.32	290.88	540.96	LEVEL	280	5
8	1	8784	8568	.	331.2	331.2	456	540.96	TEXT	330	4
9	1	9040	8841	605.28	605.28	720.96	636.96	540.96	PBUTTONS	605	8
10	1	9260	9040	.	34.08	34.08	55.2	527.52	1	35	9
11	1	9488	9260	.	84.96	84.96	86.88	527.52	IO_CAN_NOT_OPEN	85	15
12	1	9712	9488	.	229.44	229.44	250.56	527.52	1	230	9
13	1	9928	9712	.	280.32	280.32	282.24	527.52	E	280	1
14	1	10178	9928	.	331.2	331.2	333.12	527.52	%IData set: %1$ could not be opened.	330	36
15	1	10444	10235	605.28	605.28	720.96	607.2	527.52	SASHELP.FSP.OC.SLIST	605	20
16	1	10664	10444	.	34.08	34.08	52.8	514.08	17	35	9
17	1	10893	10664	.	84.96	84.96	86.88	514.08	IN_VERIFY_DELETE	85	16
18	1	11117	10893	.	229.44	229.44	250.56	514.08	1	230	9
19	1	11333	11117	.	280.32	280.32	282.24	514.08	Q	280	1
20	1	11582	11333	.	331.2	331.2	333.12	514.08	Are you sure you want to delete %$?	330	35
21	1	11847	11639	605.28	605.28	720.96	607.2	514.08	SASHELP.FSP.YN.SLIST	605	20
22	1	12068	11847	.	34.08	34.08	52.8	500.64	19	35	9

Display 3: SAS data set '_tmp_2' included the separated text and format information for each cell.

In the SAS dataset _tmp_2, each cell's text and features are represented in one observation. Each variable has the meaning as follows:

(1) Loc1 to Loc3: One of them will be the x-coordinate for the cell's frame on the left side of the border (if having).
(2) Loc4: The x-coordinate for the cell's text on the bottom- left of the border
(3) Loc5: The y-coordinate for the cell's text
(4) Variable Text: the text in each cell
(5) Variable Len: the length (include the leading blank) of text in each cell
(6) Variable Page: the page number where each cell is located in

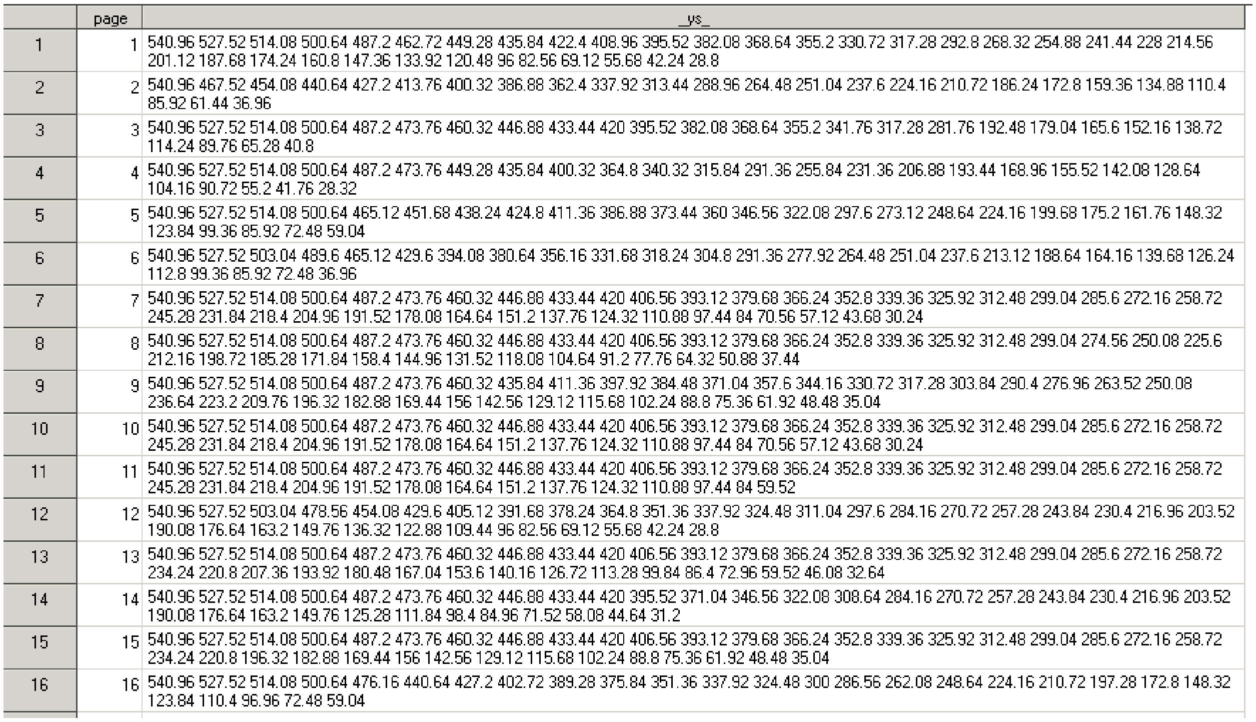
The unit of the length is point (1 inch equal to 72 point).

4. OBTAIN ROW AND COLUMN NUMBER

In SAS dataset '_temp_2', the page number is the value of variable 'Page'. But, we also need to get the column and row number for each cell in the page. By integrating the cell frame information, the row and column number can be obtained. The following is the code.

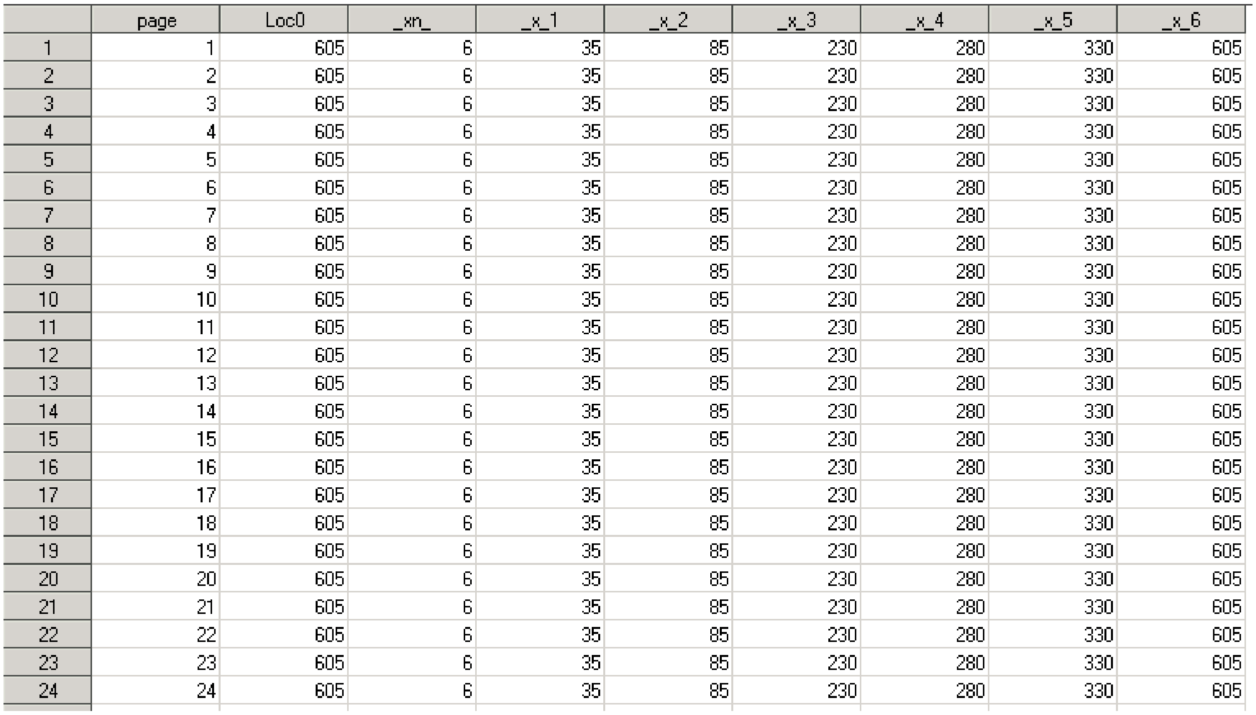
```
%* Using proc sql for the row/column information on coordinate: Loc5, Loc0 *;
proc sql noprint;
      create table _rl_0 as select * from
   (select page, Loc5, Loc0, n(1) as _xn_ from _tmp_2 group by page, Loc5)
       group by page having _xn_=max(_xn_) order by page, Loc5 desc;

      select _xn_ into :_xn_ trimmed from _rl_0(obs=1);
quit;

%* Get the x,y coordinator for the frame *;
%local j;
%do j=1 %to 2;
proc sql;
      create table _rl_&j as select distinct page, Loc%scan(5 0,&j), _xn_
        from _rl_0 %if &j=1 %then order by page, Loc5 desc; %else where
Loc0>.;;
```

```
quit;
data _rl_%eval(&j+2);
      set _rl_&j;
      by page;
  %if &j=1 %then %do;
      length _ys_ $400;
      retain _ys_;
      if first.page then _ys_ = '';
      _ys_ = catx(' ',_ys_,Loc5);
  %end;
  %else %do;
      array _x_[&_xn_];
      retain _x_1-_x_&_xn_;
      if first.page then i = 0;
      i + 1;
      _x_[i] = Loc0;
  %end;
      if last.page;
run;
%end;
```



	page	_ys_
1	1	540.96 527.52 514.08 500.64 487.2 462.72 449.28 435.84 422.4 408.96 395.52 382.08 368.64 355.2 330.72 317.28 292.8 268.32 254.88 241.44 228 214.56 201.12 187.68 174.24 160.8 147.36 133.92 120.48 96 82.56 69.12 55.68 42.24 28.8
2	2	540.96 467.52 454.08 440.64 427.2 413.76 400.32 386.88 362.4 337.92 313.44 288.96 264.48 251.04 237.6 224.16 210.72 186.24 172.8 159.36 134.88 110.4 85.92 61.44 36.96
3	3	540.96 527.52 514.08 500.64 487.2 473.76 460.32 446.88 433.44 420 395.52 382.08 368.64 355.2 341.76 317.28 281.76 192.48 179.04 165.6 152.16 138.72 114.24 89.76 65.28 40.8
4	4	540.96 527.52 514.08 500.64 487.2 473.76 449.28 435.84 400.32 364.8 340.32 315.84 291.36 255.84 231.36 206.88 193.44 168.96 155.52 142.08 128.64 104.16 90.72 55.2 41.76 28.32
5	5	540.96 527.52 514.08 500.64 465.12 451.68 438.24 424.8 411.36 386.88 373.44 360 346.56 322.08 297.6 273.12 248.64 224.16 199.68 175.2 161.76 148.32 123.84 99.36 85.92 72.48 59.04
6	6	540.96 527.52 503.04 489.6 465.12 429.6 394.08 380.64 356.16 331.68 318.24 304.8 291.36 277.92 264.48 251.04 237.6 213.12 188.64 164.16 139.68 126.24 112.8 99.36 85.92 72.48 36.96
7	7	540.96 527.52 514.08 500.64 487.2 473.76 460.32 446.88 433.44 420 406.56 393.12 379.68 366.24 352.8 339.36 325.92 312.48 299.04 285.6 272.16 258.72 245.28 231.84 218.4 204.96 191.52 178.08 164.64 151.2 137.76 124.32 110.88 97.44 84 70.56 57.12 43.68 30.24
8	8	540.96 527.52 514.08 500.64 487.2 473.76 460.32 446.88 433.44 420 406.56 393.12 379.68 366.24 352.8 339.36 325.92 312.48 299.04 274.56 250.08 225.6 212.16 198.72 185.28 171.84 158.4 144.96 131.52 118.08 104.64 91.2 77.76 64.32 50.88 37.44
9	9	540.96 527.52 514.08 500.64 487.2 473.76 460.32 435.84 411.36 397.92 384.48 371.04 357.6 344.16 330.72 317.28 303.84 290.4 276.96 263.52 250.08 236.64 223.2 209.76 196.32 182.88 169.44 156 142.56 129.12 115.68 102.24 88.8 75.36 61.92 48.48 35.04
10	10	540.96 527.52 514.08 500.64 487.2 473.76 460.32 446.88 433.44 420 406.56 393.12 379.68 366.24 352.8 339.36 325.92 312.48 299.04 285.6 272.16 258.72 245.28 231.84 218.4 204.96 191.52 178.08 164.64 151.2 137.76 124.32 110.88 97.44 84 70.56 57.12 43.68 30.24
11	11	540.96 527.52 514.08 500.64 487.2 473.76 460.32 446.88 433.44 420 406.56 393.12 379.68 366.24 352.8 339.36 325.92 312.48 299.04 285.6 272.16 258.72 245.28 231.84 218.4 204.96 191.52 178.08 164.64 151.2 137.76 124.32 110.88 97.44 84 59.52
12	12	540.96 527.52 503.04 478.56 454.08 429.6 405.12 391.68 378.24 364.8 351.36 337.92 324.48 311.04 297.6 284.16 270.72 257.28 243.84 230.4 216.96 203.52 190.08 176.64 163.2 149.76 136.32 122.88 109.44 96 82.56 69.12 55.68 42.24 28.8
13	13	540.96 527.52 514.08 500.64 487.2 473.76 460.32 446.88 433.44 420 406.56 393.12 379.68 366.24 352.8 339.36 325.92 312.48 299.04 285.6 272.16 258.72 234.24 220.8 207.36 193.92 180.48 167.04 153.6 140.16 126.72 113.28 99.84 86.4 72.96 59.52 46.08 32.64
14	14	540.96 527.52 514.08 500.64 487.2 473.76 460.32 446.88 433.44 420 395.52 371.04 346.56 322.08 308.64 284.16 270.72 257.28 243.84 230.4 216.96 203.52 190.08 176.64 163.2 149.76 125.28 111.84 98.4 84.96 71.52 58.08 44.64 31.2
15	15	540.96 527.52 514.08 500.64 487.2 473.76 460.32 446.88 433.44 420 406.56 393.12 379.68 366.24 352.8 339.36 325.92 312.48 299.04 285.6 272.16 258.72 234.24 220.8 196.32 182.88 169.44 156 142.56 129.12 115.68 102.24 88.8 75.36 61.92 48.48 35.04
16	16	540.96 527.52 514.08 500.64 476.16 440.64 427.2 402.72 389.28 375.84 351.36 337.92 324.48 300 286.56 262.08 248.64 224.16 210.72 197.28 172.8 148.32 123.84 110.4 96.96 72.48 59.04

Display 4: SAS data set ‘_rl_3’. Variable ‘_ys_’ is the list of row starting location for the y-coordinate



	page	Loc0	_xn_	_x_1	_x_2	_x_3	_x_4	_x_5	_x_6
1	1	605	6	35	85	230	280	330	605
2	2	605	6	35	85	230	280	330	605
3	3	605	6	35	85	230	280	330	605
4	4	605	6	35	85	230	280	330	605
5	5	605	6	35	85	230	280	330	605
6	6	605	6	35	85	230	280	330	605
7	7	605	6	35	85	230	280	330	605
8	8	605	6	35	85	230	280	330	605
9	9	605	6	35	85	230	280	330	605
10	10	605	6	35	85	230	280	330	605
11	11	605	6	35	85	230	280	330	605
12	12	605	6	35	85	230	280	330	605
13	13	605	6	35	85	230	280	330	605
14	14	605	6	35	85	230	280	330	605
15	15	605	6	35	85	230	280	330	605
16	16	605	6	35	85	230	280	330	605
17	17	605	6	35	85	230	280	330	605
18	18	605	6	35	85	230	280	330	605
19	19	605	6	35	85	230	280	330	605
20	20	605	6	35	85	230	280	330	605
21	21	605	6	35	85	230	280	330	605
22	22	605	6	35	85	230	280	330	605
23	23	605	6	35	85	230	280	330	605
24	24	605	6	35	85	230	280	330	605

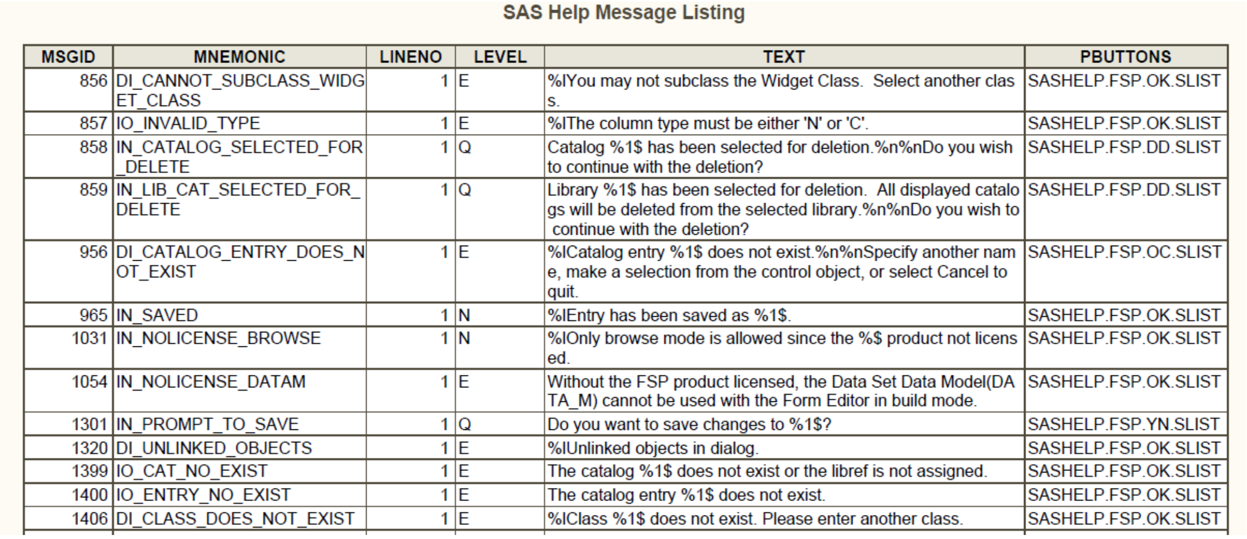
Display 5: SAS data set '_rl_4'. Variable _x_1 - _x_5 listed the 5 column's start location of x-coordinate.

In the following code, data set _tmp_2 is merged with _rl_3 and _r1_4 to get the row and column number.

```
%* According to the size frame, get row and column number *;
data _tmp_3(keep=page Text len part row col);
      merge _tmp_2 _rl_3(keep=page _ys_) _rl_4(keep=page _x_1-_x_&_xn_);
      by page;
      array _x_[&_xn_];
      retain part row col_ n_r L_y;
      if first.page then do;
            call missing(col_,row);
            n_r = 1;
            L_y = input(scan(_ys_,n_r,' '),best.);
      end;
      do i=1 to &_xn_;
            if i<&_xn_ & Loc4>_x_[i+1] then continue;
            col = i;
            leave;
      end;
      if row=. then do;
            if Loc5^=L_y then part = 1+(Loc5>250);
            else do;    row = 1;  part = 3; end;
      end;
      else if Loc5<L_y then do;
            n_r + 1;
            L_y = input(scan(_ys_,n_r,' '),best.);
            if Loc5=L_y | Loc5>L_y & col=1<col_ then row + 1;
      end;
      else if Loc5>=L_y & col=1<col_ then row + 1;
      col_ = col;
run;
```

5. RE-CONSTRUCT TEXT FROM DATA SET

After the number of rows and columns are obtained, a special situation needs to be considered. That is: the text is wrapped into a cell as multiple lines. Let's see PDF file in Display 6. When the column TEXT is 'Without the FSP product licensed, the Data Set Data Model(DATA_M) cannot be used with the Form Editor in build mode.', the column TEXT is wrapped and the 'TA_M) cannot be used with the Form Editor in build mode.' is located in the second line of the cell which is shown on Display 6. The Re-construction is necessary for the dataset _tmp_3.

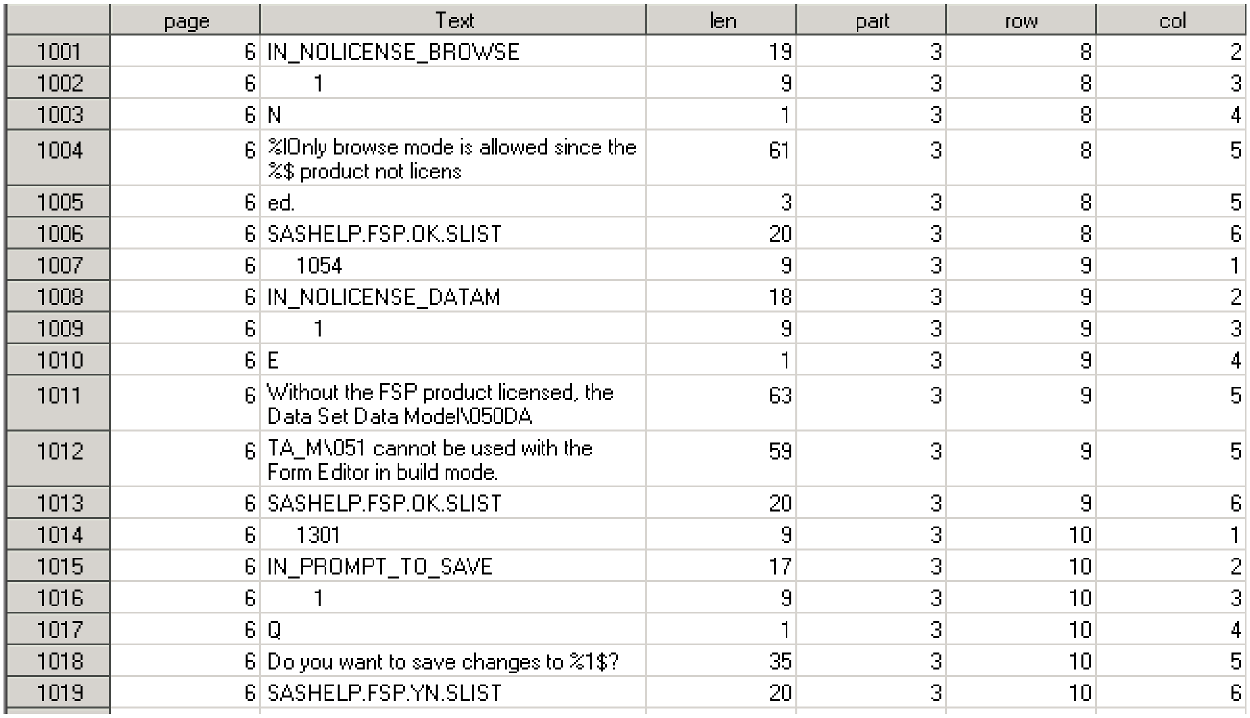


SAS Help Message Listing

MSGID	MNEMONIC	LINENO	LEVEL	TEXT	PBUTTONS
856	DI_CANNOT_SUBCLASS_WIDG ET_CLASS	1	E	%IYou may not subclass the Widget Class. Select another clas s.	SASHELP.FSP.OK.SLIST
857	IO_INVALID_TYPE	1	E	%IThe column type must be either 'N' or 'C'.	SASHELP.FSP.OK.SLIST
858	IN_CATALOG_SELECTED_FOR _DELETE	1	Q	Catalog %1$ has been selected for deletion.%n%nDo you wish to continue with the deletion?	SASHELP.FSP.DD.SLIST
859	IN_LIB_CAT_SELECTED_FOR_ DELETE	1	Q	Library %1$ has been selected for deletion. All displayed catalo gs will be deleted from the selected library.%n%nDo you wish to continue with the deletion?	SASHELP.FSP.DD.SLIST
956	DI_CATALOG_ENTRY_DOES_N OT_EXIST	1	E	%ICatalog entry %1$ does not exist.%n%nSpecify another nam e, make a selection from the control object, or select Cancel to quit.	SASHELP.FSP.OC.SLIST
965	IN_SAVED	1	N	%IEntry has been saved as %1$.	SASHELP.FSP.OK.SLIST
1031	IN_NOLICENSE_BROWSE	1	N	%IOnly browse mode is allowed since the %$ product not licens ed.	SASHELP.FSP.OK.SLIST
1054	IN_NOLICENSE_DATAM	1	E	Without the FSP product licensed, the Data Set Data Model(DA TA_M) cannot be used with the Form Editor in build mode.	SASHELP.FSP.OK.SLIST
1301	IN_PROMPT_TO_SAVE	1	Q	Do you want to save changes to %1$?	SASHELP.FSP.YN.SLIST
1320	DI_UNLINKED_OBJECTS	1	E	%IUnlinked objects in dialog.	SASHELP.FSP.OK.SLIST
1399	IO_CAT_NO_EXIST	1	E	The catalog %1$ does not exist or the libref is not assigned.	SASHELP.FSP.OK.SLIST
1400	IO_ENTRY_NO_EXIST	1	E	The catalog entry %1$ does not exist.	SASHELP.FSP.OK.SLIST
1406	DI_CLASS_DOES_NOT_EXIST	1	E	%IClass %1$ does not exist. Please enter another class.	SASHELP.FSP.OK.SLIST

Display 6: Sample PDF output with the wrapped text in the cell

The following is what happens in the dataset _tmp_3 in Display 7. Observe that both obs=1011 and 1012 belong to row 9 and column 5.



	page	Text	len	part	row	col
1001	6	IN_NOLICENSE_BROWSE	19	3	8	2
1002	6	1	9	3	8	3
1003	6	N	1	3	8	4
1004	6	%IOnly browse mode is allowed since the %$ product not licens	61	3	8	5
1005	6	ed.	3	3	8	5
1006	6	SASHELP.FSP.OK.SLIST	20	3	8	6
1007	6	1054	9	3	9	1
1008	6	IN_NOLICENSE_DATAM	18	3	9	2
1009	6	1	9	3	9	3
1010	6	E	1	3	9	4
1011	6	Without the FSP product licensed, the Data Set Data Model\050DA	63	3	9	5
1012	6	TA_M\051 cannot be used with the Form Editor in build mode.	59	3	9	5
1013	6	SASHELP.FSP.OK.SLIST	20	3	9	6
1014	6	1301	9	3	10	1
1015	6	IN_PROMPT_TO_SAVE	17	3	10	2
1016	6	1	9	3	10	3
1017	6	Q	1	3	10	4
1018	6	Do you want to save changes to %1$?	35	3	10	5
1019	6	SASHELP.FSP.YN.SLIST	20	3	10	6

Display 7: SAS data set '_tmp_3'. Column number and row number are obtained for text in each cell.

So, the concatenation is necessary. The following code is for the concatenation.

```
%* Combine the texts which is in same cell *;
data _tmp_4;
      set _tmp_3(where=(row>.));
      by page row col;
      length Text_ $400;
      retain Text_;
      if first.col=1 & last.col=1 then;
      else do;
            if first.col=1 then do;
                  Text_ = Text;
                  _len_ = len;
            end;
            else do;
                  Text_ = substrn(Text_,1,_len_)||Text;
                  _len_ + len;
            end;
            if last.col=0 then delete;
            else Text = Text_;
      end;
      Text = tranwrd(tranwrd(Text,'\050','('),'\(','(');
      Text = tranwrd(tranwrd(Text,'\051',')'),'\)',')');
      Text = prxchange('s/\\177|\\000/□/',-1,Text);
      Text = prxchange('s/\\134|\\\\/\\/',-1,Text);
run;
```

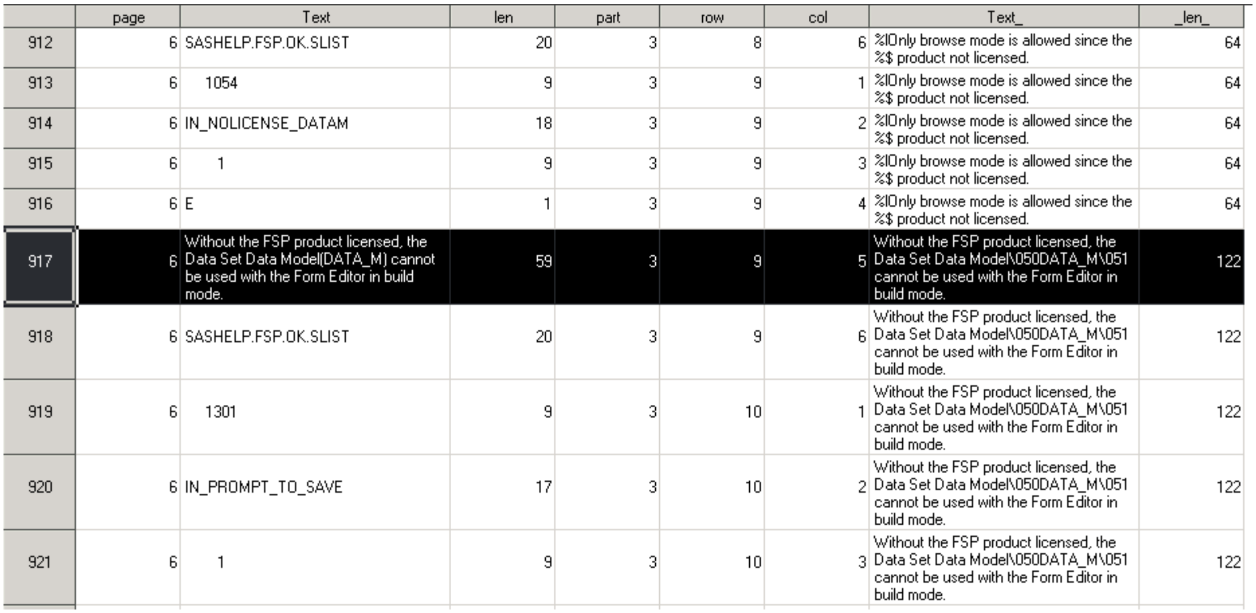
While we concatenate the wrapped text from dataset _tmp_3, it's almost verbatim text for each cell except for following 4 characters:

(1) Character '(' --- open parenthesis, ASCII code 40, as 50 in octonary number system
(2) Character ')' --- closing parenthesis, ASCII code 41, as 51 in octonary number system
(3) Character '□' --- a character in ASCII code 127, as 177 in octonary number system
(4) Character '\' --- backslash, ASCII code 92, as 134 in octonary number system

In PDF file, above 4 characters is presented as following way:

(1) Character '(' is presented as: (a) '\050' by some style, or (b) '\(' by some other style
(2) Character ')' is presented as: (a) '\051' by some style, or (b) '\)' by some other style
(3) Character '□' is presented as: (a) '\177' by some style, or (b) '\000' by some other style
(4) Character '\' is presented as: (a) '\134' by some style, or (b) '\\' by some other style

In the code for generating dataset _tmp_4, the last four lines show how to convert the 4 kind of characters back to the original symbol.

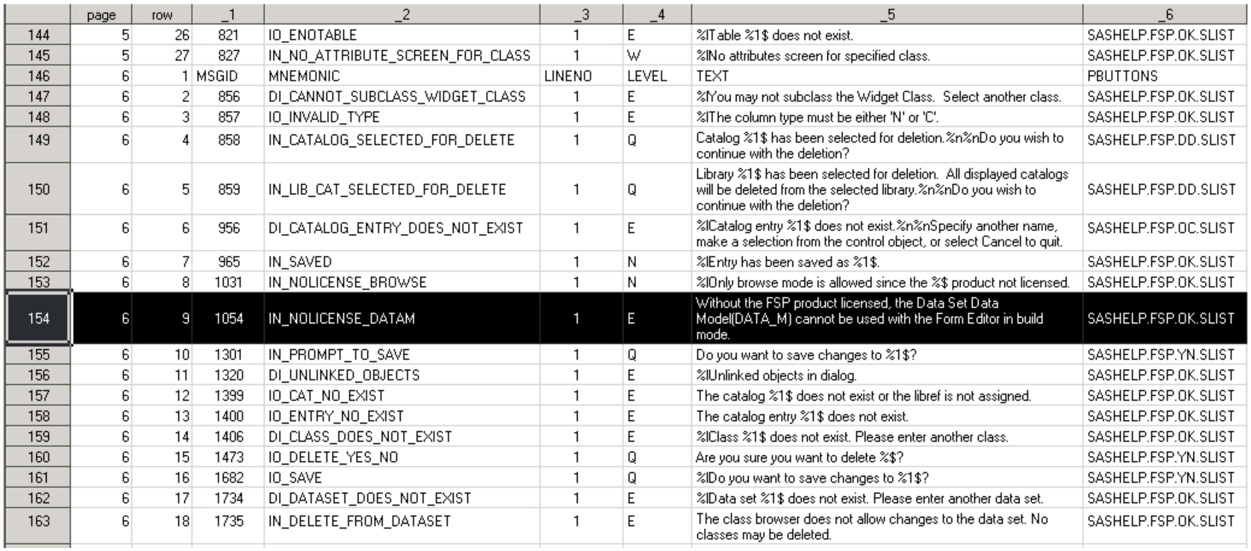


	page	Text	len	part	row	col	Text_	_len_
912	6	SASHELP.FSP.OK.SLIST	20	3	8	6	%IOnly browse mode is allowed since the %$ product not licensed.	64
913	6	1054	9	3	9	1	%IOnly browse mode is allowed since the %$ product not licensed.	64
914	6	IN_NOLICENSE_DATAM	18	3	9	2	%IOnly browse mode is allowed since the %$ product not licensed.	64
915	6	1	9	3	9	3	%IOnly browse mode is allowed since the %$ product not licensed.	64
916	6	E	1	3	9	4	%IOnly browse mode is allowed since the %$ product not licensed.	64
917	6	Without the FSP product licensed, the Data Set Data Model(DATA_M) cannot be used with the Form Editor in build mode.	59	3	9	5	Without the FSP product licensed, the Data Set Data Model\050DATA_M\051 cannot be used with the Form Editor in build mode.	122
918	6	SASHELP.FSP.OK.SLIST	20	3	9	6	Without the FSP product licensed, the Data Set Data Model\050DATA_M\051 cannot be used with the Form Editor in build mode.	122
919	6	1301	9	3	10	1	Without the FSP product licensed, the Data Set Data Model\050DATA_M\051 cannot be used with the Form Editor in build mode.	122
920	6	IN_PROMPT_TO_SAVE	17	3	10	2	Without the FSP product licensed, the Data Set Data Model\050DATA_M\051 cannot be used with the Form Editor in build mode.	122
921	6	1	9	3	10	3	Without the FSP product licensed, the Data Set Data Model\050DATA_M\051 cannot be used with the Form Editor in build mode.	122

Display 8: SAS data set '_tmp_4'. Wrapped text are combined and the special characters '(', ')' are converted

```
%* Keep the texts in the same row as the different variables *;
proc transpose data=_tmp_4 out=_tmp_5 prefix=_;
      by page row;
      id col;
      var Text;
run;
```

Look at the Display 7,8 in dataset _tmp_3 and _tmp_4, multiple consecutive observations have same row number but different column number, a transpose is necessary for the re-construction (to generate a dataset _tmp_5).



	page	row	_1	_2	_3	_4	_5	_6
144	5	26	821	IO_ENOTABLE	1	E	%ITable %1$ does not exist.	SASHELP.FSP.OK.SLIST
145	5	27	827	IN_NO_ATTRIBUTE_SCREEN_FOR_CLASS	1	W	%INo attributes screen for specified class.	SASHELP.FSP.OK.SLIST
146	6	1	MSGID	MNEMONIC	LINENO	LEVEL	TEXT	PBUTTONS
147	6	2	856	DI_CANNOT_SUBCLASS_WIDGET_CLASS	1	E	%IYou may not subclass the Widget Class. Select another class.	SASHELP.FSP.OK.SLIST
148	6	3	857	IO_INVALID_TYPE	1	E	%IThe column type must be either 'N' or 'C'.	SASHELP.FSP.OK.SLIST
149	6	4	858	IN_CATALOG_SELECTED_FOR_DELETE	1	Q	Catalog %1$ has been selected for deletion.%n%nDo you wish to continue with the deletion?	SASHELP.FSP.DD.SLIST
150	6	5	859	IN_LIB_CAT_SELECTED_FOR_DELETE	1	Q	Library %1$ has been selected for deletion. All displayed catalogs will be deleted from the selected library.%n%nDo you wish to continue with the deletion?	SASHELP.FSP.DD.SLIST
151	6	6	956	DI_CATALOG_ENTRY_DOES_NOT_EXIST	1	E	%ICatalog entry %1$ does not exist.%n%nSpecify another name, make a selection from the control object, or select Cancel to quit.	SASHELP.FSP.OC.SLIST
152	6	7	965	IN_SAVED	1	N	%IEntry has been saved as %1$.	SASHELP.FSP.OK.SLIST
153	6	8	1031	IN_NOLICENSE_BROWSE	1	N	%IOnly browse mode is allowed since the %$ product not licensed.	SASHELP.FSP.OK.SLIST
154	6	9	1054	IN_NOLICENSE_DATAM	1	E	Without the FSP product licensed, the Data Set Data Model(DATA_M) cannot be used with the Form Editor in build mode.	SASHELP.FSP.OK.SLIST
155	6	10	1301	IN_PROMPT_TO_SAVE	1	Q	Do you want to save changes to %1$?	SASHELP.FSP.YN.SLIST
156	6	11	1320	DI_UNLINKED_OBJECTS	1	E	%IUnlinked objects in dialog.	SASHELP.FSP.OK.SLIST
157	6	12	1399	IO_CAT_NO_EXIST	1	E	The catalog %1$ does not exist or the libref is not assigned.	SASHELP.FSP.OK.SLIST
158	6	13	1400	IO_ENTRY_NO_EXIST	1	E	The catalog entry %1$ does not exist.	SASHELP.FSP.OK.SLIST
159	6	14	1406	DI_CLASS_DOES_NOT_EXIST	1	E	%IClass %1$ does not exist. Please enter another class.	SASHELP.FSP.OK.SLIST
160	6	15	1473	IO_DELETE_YES_NO	1	Q	Are you sure you want to delete %$?	SASHELP.FSP.YN.SLIST
161	6	16	1682	IO_SAVE	1	Q	%IDo you want to save changes to %1$?	SASHELP.FSP.YN.SLIST
162	6	17	1734	DI_DATASET_DOES_NOT_EXIST	1	E	%IData set %1$ does not exist. Please enter another data set.	SASHELP.FSP.OK.SLIST
163	6	18	1735	IN_DELETE_FROM_DATASET	1	E	The class browser does not allow changes to the data set. No classes may be deleted.	SASHELP.FSP.OK.SLIST

Display 9: SAS data set '_tmp_5'. The data set is transposed by column numbers.

```
%* Convert some the character variable as numeric *;
data _tmp_6;
      retain re;
      if _N_=1 then re = prxparse('/^\s*-?\d{0,20}\.?\d{0,20}\s*$/');
      set _tmp_5 end=eof;
      by page row;
      array _A_[*] $400 _1-_&_xn_;
      array _V_[*] _V_1-_V_&_xn_;
      array _x_[&_xn_] _temporary_(&_xn_*0);
      array _L_[&_xn_] _temporary_(&_xn_*0);
      array _H_[&_xn_] $32 _temporary_;
      length _Var_ $32 _Keep_ _Rename_ _Length_ $1000;
      if first.page then do;
            if page=1 then do i= 1 to &_xn_;
                  _H_[i] = translate(cats(_A_[i]),'______','(-/ %)');
            end;
            delete;
      end;
      else do i=1 to &_xn_;
            _L_[i] = max(_L_[i],length(_A_[i]));
            if _x_[i]>0 then continue;
            if prxmatch(re,_A_[i])=0 then _x_[i] + 1;
            else _V_[i] = input(_A_[i],best12.);
      end;
      if eof then do;
            do i=1 to &_xn_;
                  _Fc_  = 50**(_L_[i]>32);
                  _Var_ = catt(ifc(_x_[i]=0,'_V_','_'),i);
                  _Keep_      = catx(' ',_Keep_,_Var_);
                  _Rename_= catx(' ',_Rename_,_Var_,'=',_H_[i]);
                  _length_= catx('
',_length_,_H_[i],ifc(_x_[i]=0,'8',cat('$',_Fc_*2**ceil(log2(_L_[i]/_Fc_)))))
;
            end;
            call symputx('_Keep_',  _Keep_,  'L');
            call symputx('_Rename_',_Rename_,'L');
            call symputx('_Length_',_Length_,'L');
      end;
run;
```

In the code above for data set _tmp_6, the first thing is to separate the first row in each pages as the header for the name of each column/variable. Then, please notice that the PDF present a numeric value as the character in numeric form. In _tmp_5, all text-related variables are in character. In code for dataset _tmp_6, we are looking for the variable with all value as numeric form, and convert them into numerical variables.

```
%* Set up the data set label, the variable name and suitable length *;
data %str(&out)(label="%scan(&datafile,-1,\/)");
      length &_Length_;
      set _tmp_6(keep=&_Keep_ rename=(&_Rename_));
run;
```

And also, the variable Text has initial length 400, from output SAS data set, each individual character variables get his suitable length by setting a length statement.

The combined code sections above is the entire program of macro %PDF2SAS.

CONCLUSION

The SAS Output Delivery System (ODS) is a powerful system to help manipulate and customize outputs. But, this technique is only one-way (or called single trip). By means of the Perl Regular Expression, a whole PDF file can be successfully separated into many pieces where there are important information to re-construct them back into a SAS data set. The macro %PDF2SAS can make the PDF file easily and quickly to be converted to SAS datasets and to be validated.

REFERENCE

<PDF Reference, version 1.7 - Adobe>: Available at
http://www.adobe.com/content/dam/Adobe/en/devnet/acrobat/pdfs/pdf_reference_1-7.pdf

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

William Wu
Puma Biotechnology, Inc.
701 Gateway Blvd.
South San Francisco, CA 94080
willywu2001@hotmail.com

Steven Li
Medtronic Inc.
8200 Coral Sea St NE,
Minneapolis, MN 55112

Yun Guan
Puma Biotechnology, Inc.
701 Gateway Blvd.
South San Francisco, CA 94080

Appendix 1:

```
%global _Name_ _o_;
%let _Name_ = sashelp.afmsg;

ods _all_ close;
ods output Template.Stats=Stats(where=(Type='Style'));
proc template;
      list styles;
run;
ods output close;
ods listing;

proc sql noprint;
      select distinct cat(scan("&_Name_",-1,'.'),'_',scan(Path,-1,'.')) into
:_f_1-:_f_99
        from Stats where index(Path,'NoFontDefault')=0;
      %let _o_ = &sqlobs;
quit;
```

```
%macro getpdf;
title1 j=
c 'SAS Help Message Listing';
footnote1 j=l 'This is a sample.';
ods escapechar=' ';

ods listing close;
%do i=1 %to &_o_;
ods pdf file="&path\PDF\&&_f_&i...pdf" style=%scan(&&_f_&i,-1,_) compress=0;
proc report nowd data=&_Name_ style(column hdr)={asis=on} split=' ';
      column MSGID MNEMONIC LINENO LEVEL TEXT PBUTTONS;
      define MSGID     /display style=[cellwidth=0.7in];
      define MNEMONIC  /display style=[cellwidth=2.0in];
      define LINENO    /display style=[cellwidth=0.7in];
      define LEVEL     /display style=[cellwidth=0.7in];
      define TEXT      /display style=[cellwidth=3.8in];
      define PBUTTONS /display style=[cellwidth=1.6in];
run;
ods pdf close;
%end;
ods listing;
%mend getpdf;

%getpdf;
title;
footnote;
```