PharmaSUG2010 - Paper AD07

Let your title macro report study progress
Yang Chen, Consultant, Jersey City, NJ

ABSTRACT
It is tedious for the lead programmer to check the table/figure/listing (TFL) production status, especially when the clinical study involves a large number of TFLs. This paper introduces a tool that uses title dataset to report the TFL progress automatically.

Whenever the programmer calls the title macro to add titles/footnotes, the title macro also updates the production status of the TFL in a tracking sheet, with detailed information such as table number, corresponding creator name, creation date/time, validator name and validation date/time.

This tool is useful to study management in that it provides accurate and timely progress of the study. The lead can benefit by easily finding out which TFL is missing, not validated or obsolete.

INTRODUCTION
A clinical trial study can have hundreds of tables/figures/listings (TFL). The lead programmer of the study often has to check with the team members regarding the production/validation status. Apparently this method is inefficient and time consuming.

Meanwhile, to have centralized control and to avoid typos, many programmers put the titles and footnotes for the TFLs in a central location, which can be a SAS dataset, an Excel workbook or simply a text file. Regardless of its format, the programmers call a title macro to add titles/footnotes to the individual TFL in their programs. However, the value of the central title location is underestimated if it is only used for titles/footnotes production.

This paper explains how the title data can be used to report the TFL progress of the entire study. This is very valuable to the manager and the study lead programmer. They can have a prompt understanding of the TFL production/validation progress by checking the tracking report, which is accurate and timely.

ASSUMPTION
The logic of using the title dataset to monitor the TFL status is that the title dataset is often created from the table shell. If the table shell in a company has consistent layout, programmers usually are able to use some tool to extract the titles/footnotes. The title dataset thus already have the complete list of the TFLs in a study.

I use Word VBA macro code to extract the titles/footnotes from the table shell and save it as an ASCII file (SHELL.TXT). The VBA macro checks the extracted titles against the Table of Contents (TOC) of the table shell to ensure the completeness of the title dataset. The content of the SHELL.TXT file typically looks like:

```plaintext
... Table 1.1
title text line 1 for Table 1.1
title text line 2 for Table 1.1
... Notes: footnote text line 1 for Table 1.1
Footnote text line 2 for Table 1.1
Footnote text line 3 for Table 1.1
...

Table 1.2
title text line 1 for Table 1.2
```

The benefit of using TXT is that it is very easy to be edited. However, whether you choose to use text, SAS dataset or excel sheet does not matter here.
HOW TO USE

The macro tool named %title is very easy to use, simply insert the macro named %title with the table number as the parameter before PROC REPORT in the individual SAS program. Like:

```
%title(Table 1.1);
PROC REPORT DATA=report data ...;
...
```

%title will then search for the titles and footnotes of Table 1.1 in shell.TXT and convert them into TITLE statements and FOOTNOTE statement (or COMPUTE blocks if there are more than 10 footnotes). This is typically how a title macro adds titles/footnotes to the output. Therefore the code is not presented here.

At the same time, %title checks the presence of the tracking report (named REPORT.TXT) in the output area by using FILEEXIST function.

If REPORT.TXT does not exist, that means this is the first time that a TFL is produced in the study. %title will then create REPORT.TXT, which contains all the TFL names sorted by section number (TABNO). Notice that at this time, information about all the TFLs other than their names is not available, and will thus remain blank. The SAS code is as below:

```
%IF not %sysfunc(fileexist(&outputpath/REPORT.TXT)) %then %do;

***Dataset report contains all the TFL names of the study;
DATA alltfl;
   SET shell;
   WHERE titleflg=1; *** only use the 1st title line, i.e. Table x.x ***;
   LENGTH wholeln $200;
   wholeln=tabno;  *** no other information available except table name ***;
RUN;

***Sort the title by its number ***;
PROC SORT DATA=alltfl;
   BY tabno;
RUN;

***Output all the TFL names to a list ***;
FILENAME report "&outputpath/REPORT.TXT"
DATA _NULL_;  FILE report;
   SET alltfl end = eof;
   IF _n_=1 THEN DO;
      put /* primary section on the left half */
         @1 "Package"
         @10 "T/F/L"
         @30 "Name"
         @50 "Warnings"
         @70 'Date & Time'
      /* validation section on the right half */
         @88 'Name'
         @99 'Match? (Diff# if no)'
         @120 'Date & Time';
   put @1
   "_________________________________________________________________________________";
   END;
   PUT @1 package @10 wholeln;
RUN;

***Under UNIX, make the REPORT rewritable by all team members ***;
DATA _null_;  CALL system("setfacl -m group:stat:rwx &outputpath/REPORT.TXT");
RUN;

%END;
```
If REPORT.TXT already exists, then %title will automatically sign in on this tracking sheet with the following information: programmer name, date & time and other miscellaneous stuff.

This is done by first read in the REPORT line by line into a SAS dataset so that the latest information is retained, then the line pertaining to the TFL currently being produced/validated (&thistabno) will be updated. Then REPORT.TXT will be recreated with the updated SAS dataset.

Notice that in my working environment whether the programmer is the primary or the validator depends on the working path name. If the last part of the working path is “validation”, then he/she is the validator, otherwise primary. You should use your own way to define your role.

The code is as below:

```plaintext
***OUTPUT to Summary TXT ***;
%LET username=%sysget(USER); ***Get the programmer name ***;

***read in REPORT as of now ***;
DATA alltfl;
  infile "&outpath/REPORT.TXT" DSD lrecl=32767 firstobs=5/*skip the titles */;
  length tabno $19 wholeln $200;
  input @10 tabno @1 wholeln;
    tabno=strip(tabno);
RUN;

***Only update the current TFL (thistabno) information ***;
DATA alltfl;
  SET alltfl;
  IF upcase(tabno) eq upcase("&thistabno") THEN DO;
    %IF %scan(&outpatht, -1, '/') ne validation %THEN %DO;
      /* if you are the primary programmer */
      wholeln=PUT("&thispackage", $9.)||PUT("&thistabno", $19.)||PUT("&username", $10.)||PUT("&footnote_warn", $30.)||"&today";
    %END;
    %ELSE %IF %scan(&outpatht, -1, '/') eq validation %THEN %DO;
      /* if you are the validation programmer */
      wholeln=substr(wholeln, 1, 88)||PUT("&username", $10.)||'Manual Check'||"&today";
    %END;
  END;
RUN;

PROC SORT DATA=alltfl;
  BY tabno;
RUN;

***Recreate the REPORT.TXT with the updated information ***;
FILENAME report "&outpath/REPORT.TXT";
DATA _NULL_; file report;
  SET alltfl end = eof;
  IF _n_=1 THEN DO;
    put /* primary section on the left half */
      @1 "Package"
      @10 "T/F/L"
      @30 "Name"
      @50 "Warnings"
      @70 'Date & Time'
      /* validation section on the right half */
      @88 'Name'
      @99 'Match?(Diff# if no)'
      @120'Date & Time';
  END;
  put @1 "___________________________________________________________________";
  put @1 wholeln;
RUN;
```
INTERPRET THE REPORT

The layout of REPORT.TXT is mainly composed of two parts: Primary and Validator.

The left half is for the Primary Programmer, which has 4 sub columns: TFL name, Primary Programmer name, Warning and production Date & Time.

The right half is for the Validation Programmer, which has 3 sub columns: Validator Programmer name, whether their count match or not, validation Date & Time.

From the example below, we can quickly find out who and when the TFLs are produced/validated. For example, Table 1.1 is produced by Tom at 14:03 on 2009-11-20, and validated by Jack at 15:27 on 2009-11-20.

Table 1.3 is produced by Tom at 09:03 on 2009-11-28, but is not validated yet or it was validated but the table was rerun and the validation record was wiped out.

Table 3.3 is not produced by any programmer since the primary programmer name is blank. The lead programmer should be alarmed.

In addition, there are two optional columns:
- **Warning**: %title also provides warning message if the titles and footnotes of the TFL are absent or too long.
- **Match? (Diff#)**: by default, this column will be filled with “Manual Check”, i.e. the validator checks the contents of that table produced by the primary programmer manually. However, if the validator chooses to use my automatic validation tool, it will report whether there is a 100% match or number of different lines in the table (refer to my Three Step Table Validation, PharmaSUG, Denver, Colorado, 2007)

<table>
<thead>
<tr>
<th>T/F/L</th>
<th>Name</th>
<th>Warning</th>
<th>Date &amp; Time</th>
<th>Name</th>
<th>Match? (Diff#)</th>
<th>Date &amp; Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1.1</td>
<td>Tom</td>
<td></td>
<td>2009-11-20 14:03</td>
<td>Jack</td>
<td>Yes</td>
<td>2009-11-20 15:27</td>
</tr>
<tr>
<td>Table 1.2</td>
<td>Tom</td>
<td>No footnotes</td>
<td>2009-11-21 08:53</td>
<td>Jack</td>
<td>No (10)</td>
<td>2009-11-21 15:18</td>
</tr>
<tr>
<td>Table 1.3</td>
<td>Tom</td>
<td></td>
<td>2009-11-28 09:03</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Table 2.1</td>
<td>Dick</td>
<td></td>
<td>2009-11-21 16:27</td>
<td>Mary</td>
<td>Yes</td>
<td>2009-11-22 14:03</td>
</tr>
<tr>
<td>Table 2.2</td>
<td>Dick</td>
<td>Footnotes Length &gt; 132</td>
<td>2009-11-22 14:03</td>
<td>Mary</td>
<td>Yes</td>
<td>2009-11-22 18:32</td>
</tr>
<tr>
<td>Table 2.3</td>
<td>Dick</td>
<td></td>
<td>2009-11-20 10:46</td>
<td>Mary</td>
<td>Manual Check</td>
<td>2009-11-20 17:41</td>
</tr>
<tr>
<td>Table 2.4</td>
<td>Dick</td>
<td></td>
<td>2009-11-19 12:03</td>
<td>Mary</td>
<td>Yes</td>
<td>2009-11-20 18:39</td>
</tr>
<tr>
<td>Table 3.1</td>
<td>Harry</td>
<td></td>
<td>2009-11-18 09:21</td>
<td>Jane</td>
<td>No (22)</td>
<td>2009-11-19 14:03</td>
</tr>
<tr>
<td>Table 3.2</td>
<td>Harry</td>
<td></td>
<td>2009-11-22 17:08</td>
<td>Jane</td>
<td>Yes</td>
<td>2009-11-22 18:09</td>
</tr>
<tr>
<td>Table 3.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CONCLUSION

This %title macro not only outputs titles and footnotes for TFL production, but also reports the TFL production/validation progress automatically. If the same table is rerun, it will be updated. The lead programmer or study manager no longer needs to check with his/her individual team members for production status. A glance at the report will reveal whether any TFL in the study is missing, unvalidated or outdated. Of course, the tracking report can be TXT, CSV or even HTML based on your preference.

As long as your title dataset contains all the TFL names, you can always insert the sample code into your own title macro. If you work under UNIX or any other operating system, where the writing access is limited, you have to make sure that the tracking report REPORT.TXT is readable and writable by all the team members.

Now whenever you run your TFL program, a complete report of the study will be produced for your convenience!
CONTACT INFORMATION

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

- Name: Yang Chen
- Enterprise: Independent Consultant
- Address: 340 Mercer Loop
- City, State ZIP: Jersey City, NJ 07302
- Work Phone: (201)253-6625
- E-mail: snail_chen@hotmail.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.