From Report and Analysis Plan to SAS Report Programs
— A Mass Production Approach for Report Writing

Sam X. Ye, Corning Besselaar Inc., Princeton, NJ

ABSTRACT

A Report and Analysis Plan (RAP) for a clinical study is the central document for SAS report writing. The major part of a RAP consists of a number of customized tables called mock-ups. Final report tables should be the same as mock-ups with real data displayed.

Because of the diversity and complexity of custom tables and the need for last minute modifications, coding to reproduce a mock-up table in a SAS program is time consuming. And coding needs to be done for each distinct table, one by one.

The Mass Production Approach (MP approach) presented in this paper is a data-driven approach. Using this technique, the time consuming process described above for an entire study can be finished instantly, provided that the text file of mock-ups exists. The difficult coding effort becomes a simple task of assigning program names for tables in a program-writer macro.

OVERVIEW

A Report and Analysis Plan (RAP) for a clinical study is a central document for producing custom report tables. The major part of RAP consists of a series of customized and real-sized tables and/or listings called mock-ups. Mock-ups contain all information defining the tables: titles, column headings, footnotes, and maybe some data examples, value formats or programming notes. It’s quite common to have more than one hundred mock-up tables in a RAP. Final report tables should look similar or the same as mock-ups with the real data displayed in them. It is also an important fact that the form and the contents of a report table could be changed frequently, before or after the programs are put in production. Good report programs must be ready for any last minute modification.

Because of the diversity, complexity, and uncertainty of the custom report tables, it is a time consuming process to code mock-ups into productive SAS report programs.

Adopting the data-driven concept, this paper developed the Mass Production Approach which can produce dozens of report programs for hundreds of tables instantly. What involved in this approach can be summarized as follows:

1. Transfer mock-up file (usually a word processor file) into a text file. Minor editing may be needed. This text file is the ‘data’ file to the MP approach.

2. Invoke the program-writer macro FMWRITE. The macro FMWRITE reads and analyzes the ‘data’ file and produces a number of SAS report programs assigned by the programmer. These report programs are called machine-made skeleton programs.

3. The macro FMWRITE also creates two global macro variable definition programs: one for all table identification numbers and one for all footnotes. An execute file (script in UNIX) is created for convenience. If it is executed right away, all skeleton programs will be run and all tables will be produced without any syntax error. These tables would have no data displayed, but the titles, headings, and footnotes would be just like mock-up tables.

4. What makes this MP approach possible and meaningful is the use of the table-generator macro MFORM. For each table, the macro FMWRITE writes an invocation of macro MFORM, i.e. %MFORM(…) in a skeleton program. %MFORM is responsible for generating the table. This general macro has two major features:

   1. It uses a simple parameter structure. The simplicity of MFORM makes it possible for the program-writer macro FMWRITE to transfer the information carried by mock-ups into a working SAS report program. This working program primarily consists of %MFORM(s).

   2. It uses a flexible specification method for the parameters. It is compatible to the PUT specification. The flexibility of macro MFORM makes the modification of a table very easy. Thus, the skeleton program can be developed to a completed one without difficulty.

(4) The %MFORM(…) in a skeleton program contains strings of titles and headings, symbolized positions for these strings, column position numbers, and global macro variables which represent table identification number and footnotes. The machine-made skeleton program is then modified by the programmer for completion. This can be done by adding in the derived data and feeding the %MFORM(…) with the variable names. At this time, the execute file can be run to produce all real report tables.
AN EXAMPLE

Assume a mock-up file MOC.TAB for a study contains DATA LISTING 1 to DATA LISTING 20, and TABLE 1 to TABLE 60. A small part of this file is shown at the end of this paper.

As an example, we want SAS report programs for DATA LISTING 4, DATA LISTING 19 to 20, TABLE 2.2, and TABLE 60. Below is the macro invocation to produce these programs.

```sas
%FMWRITE (JOBNAME=SY,
    MOCKIN=moc.tab,
    TABSTR=LISTING\TABLE,
    TABGRP= r1acct: L4
        \ rlae: L19 L20
        \ rtdisc: T2.2
        \ rtptdiol: T60,
    ENDSTR=],
    PROGMR=Sam Ye
);
```

This macro is very easy to use. The parameter JOBNAME is used to facilitate the naming of the execute file and the footnotes involved in this job. Parameters TABSTR and ENDSTR are designed for recognizing the boundary of a table in the mock-up file.

The parameter TABGRP groups tables to the programs. TABGRP can be specified as follows.

```sas
program name : list of table names
   \ program name : list of table names > ...
```

where a table name is defined as the first character of TABSTR followed by the table number. The program names also serves as the roots of global macro variable name for table identification in the report programs.

This macro call writes one table identification file, one footnote file, one execute file, and four report programs: r1acct, rlae, rtdisc, and rtptdiol.

The table identification file, MTABNO.SAS, defines the global macro variables for tables to be referenced in the report programs. The program names and the second line of the title are also written as comments. Any change of table number can be done in this file without touching report programs.

```sas
*******;
** mtabno.sas: TABLE IDENTIFICATION AND REFERENCE;

%let lacct= PATIENT DATA LISTING 4;
    *<ract > PATIENT ACCOUNTABILITY;

%let lael = PATIENT DATA LISTING 19;
    *<lael > ADVERSE EXPERIENCES;
```

Similarly, a footnote file MFOOT.SAS is written by the macro FMWRITE. Following is a part of this file.

```sas
******;
** mfoot.sas: FOOTNOTES AND THEIR REFERENCES;
** THE FORM OF FOOTNOTE REFERENCES IS:
    (JOB NAME)\#\#;
    THE FIRST \# INDICATES HOW MANY TABLES
    IN THIS JOB USE THAT FOOTNOTE
    THE SECOND \# is the sequence number
    WITH THE SAME FIRST number;

%let syln1 = %STR("** = Estimated date
    was used for adverse experience with
    unknown onset day.");
%let syln2 = %STR("+ Relative to the
day of randomization.");
%let syln3 = %STR("+ Baseline = last
    measurement obtained prior to
    randomization.");
%let syln4 = %STR("++ Relative to day
    of first dose of open-label
    medication.");
%let syln5 = %STR("1 add scheduled
    pre-baseline visit (i.e., screen).");
%let syln6 = %STR("DB = double-blind
    period.");
%let syln7 = %STR("OL = open-label
    period.");
```

Following is an example of one of the skeleton programs created by the macro FMWRITE -- Rlacct.SAS.

```sas
******;
** rlacct.sas
** FUNCTION: GENERATE TABLE(S) -- lacct
** PROGRAMMER: Sam Ye, DATE: 24JUL95

option mprint nosymbolgen nomlogic;
%inc mfront;
%let programe=rlacct;

data dummy;
    dum=1;
%mfont(font=0, footrows=15);
%MFORM(DATA=_last_,

```
sortby=,
pagenby=,
pagesf=,
TITLE1= // "&lacct ",
TITLE2='PATIENT ACCOUNTABILITY',
COL = &margin 9 9 9 9 15 11 ,
COLHD1 = '//Treatment Group: [Placebo or drug],
COLHD2 = 'Pat.' 'Visit' 'Days on'
COLHD3 = 'No.' 'Period' '(Week)'
'Study+' 'Open-Label++'
'Disposition',
BODY1= ,
FOOT1 = !&syln2 ,
FOOT2 = !&syln4 ,
FOOT3 = !&syln6 ,
FOOT4 = !&syln7 
);

The tables produced by the skeleton program and the completed program are shown at the end of this paper.

THE MACRO MFORM

From the above example, it is clear that the macro MFORM plays a central role in the MPM Approach. This section describes the macro MFORM in detail.

1. THE STRUCTURE OF PARAMETERS

The form of typical report tables can be viewed in following ways.

From top to bottom, the tables fall into three sections: heading, body, and footnotes. The heading includes the header (eg. company name, project name, date, and page number), the titles (ie. centered table name and table descriptions), and other headings after the titles (eg. treatment group, and the descriptions of the columnar data).

From left to right, data in tables are usually arranged by columns. Note that columns in tables are an ambiguous concept. It is a visual feeling of a cluster of data or character strings. People usually want the data or strings in columns aligned or balanced in various styles. Thus, even data or strings in the same column, may be in different positions depending on the data and may differ from line to line. The macro should be equipped with an easy way for adjusting the position according a variable's format. The design of the macro structure and the parameters of MFORM are based on the above considerations.

Four major groups of parameters reflect the basic structure of MFORM. They are:

TITLE1-20 = strings (and variables) for titles
HEAD1-20 = strings and variables for headings
BODY1-20 = variables and strings, for putting anything between heading and footnotes.
FOOT1-20 = strings (and variables) for footnotes.

The number after a parameter root indicates the order of the parameter executed in the same group.

Other important parameters are:

DATA = data set name for the table.
COLUMN = integer numbers for column spaces.
MTOP = macro name for the top of the table.
SORTBY = variable list for sorting.
PAGEBY = a variable name in the list of SORTBY, for page breaking.
PAGEIF = a condition for page breaking.
BTWBDPT = an integer number for setting the minimum number of lines between the body of the data and the footnotes.
BIATITLE = an integer for moving the center of the titles.

Global macro variables needed for MFORM include:
PTITLE1-n for printing project or study titles on the top of the table. LINESIZE, PAGESIZE, and LMARGIN (left margin) for page setting. These global macro variables are usually set by other macros for efficiency and consistency.

Some additional parameters are not essential. But they may add convenience and flexibility for more complex requirements. These parameters correspond to the major four groups of parameters:

MITLE=, MHEAD=, MBODY=, and MFOOT= which can receive macro names designed for sharing common code among tables. The content of these macros can be any SAS or macro statements allowed within a data step.

TITLEIN=, HEADIN=, BODYIN=, and FOOTIN= can receive any SAS or macro statements allowed within a data step. These parameters are designed for handling extremely complicated tables.

2. PARAMETER SPECIFICATION

In MFORM, the four groups of parameters control what, where, and how to print a table. The key concept of specifying these parameters is the POSITION-SYMBOlIZED specification. This technique simplifies the programming in two ways:

1. A single and uniform symbol "!" replaces all column position pointers @number (or variable). Position calculations or changing variable names for all lines and all columns are eliminated.

2. A number of keywords greatly reduce the number of SAS statements needed (e.g., WRAPVARS, SPEFORMAT) or greatly simplify the specification of the PUT statement (e.g., UNDERLINE).

HEAD and BODY Parameters

The form for specifying parameters for HEAD and BODY are exactly the same, although HEAD mainly specifies strings for printing headings and BODY mainly specifies variables for printing data in the body of the table. There are two forms for specifying these parameters: the basic form and the extended form. Below is a formal description.

(1) The basic form

```
<IF ... THENPUT>  POSITION-SYMBOlIZED specification
<ELSEPUT position-symboled specification>
```

where the position-symboled specification is

```
< front keywords >  :  <put-specification>
!column-unit  < ...  !column-unit>
< underline keywords >  <put-specification>
```

where the put-specification is anything allowed in the PUT statements after the SAS keyword PUT.

The front keywords could be:
SPESTRING (for writing multi-line header strings)
SPEFORMAT (for writing multi-line variable formats)
WRAPVARS (for wrapping and writing variables)
JUMPTOCOL# (for moving start column to column #)
ALLINDENT# (for moving position from defined positions for all columns).

The underline keywords could be:
UNDERLINE (for underscoring all strings in the specification)
LINEACROS# (for underlining columns).

The column-unit is

```
< + (integer) >  <variable < format > >  <string >.
```

(2) The extended form

```
<IF ... THENPUT:>  <Position-symbolized specification>  %str(); any SAS statements allowed within a data step, use %str() to quote any semicolon.
<ELSEPUT the same as above>
```

It is obvious that the basic form is simply the first part of extended form. The reason to separate them is only for convenience in description and later reference.

The optional IF ... THENPUT and ELSEPUT are keywords in MFORM. If these keywords exist, the following code will be executed at the location where the parameter represents:

```
if ... then
  %mput( the contents after THENPUT );
else
  %mput( the contents after ELSEPUT );
```
The macro MPUT is a key macro within MFORM, responsible for generating appropriate SAS put statements according to the parameter specification, mainly the position-symbolized specification.

Because of the important role of these parameters in generating tables, examples will be given in the next section.

**FOOT Parameters**

Specifying FOOT parameters is similar to specifying HEAD and BODY, although actual specifications for FOOT are usually much simpler:

\[ \text{IF ... THENPUT: > POSITION-SYMBOLIZED specification} \]

But, there is a difference in the meaning of the keywords, IF condition THENPUT. The keywords here are used for dynamic footnotes. As long as the stated condition is met somewhere on the output page, the footnote described in this parameter will appear on that page. If the condition is not true for all lines of a page, the footnote would not be printed. There is no keyword ELSEPUT in the specification because there is no need.

The following is an example of a dynamic footnote specification:

- FOOT1 = // 10**",
- FOOT2 = if nmiss(sex, age) > 0 then:
  - '!- Missing data.';
- FOOT3 = '!* Relative to the first day of study.';

If sex or age are present on a page, the footnotes for that page would be:

\* Relative to the first day of study.

If sex or age are missing on a page, the footnotes for that page would be

\* - Missing data.

\* Relative to the first day of study.

**TITLE Parameters**

The form for specifying TITLE parameters is simpler. You simply need to ensure that the expression for the entire title has character attributes:

\[ \text{</... > string < | | variable > < string > } \text{</... >} \]

where "/" means the same as the PUT statement, i.e., move to the next line. Any number of "/"s could be specified. The string should be quoted. If the variable is numeric, PUT(variable, format) should be used here to convert the variable.

If another parameter, BIATITLE, has been assigned to an integer, it would shift the center of all titles in MFORM from the center of a page to the left (negative integer) or to the right (positive integer).

The following is an example of a dynamic title specification:

\[ \text{TITLE1 = } // \text{ 'TABLE 2' /,} \]
\[ \text{TITLE2 = 'SUMMARY FOR' } || \text{ symptom /,} \]

or, if symptom is a numeric variable:

\[ \text{TITLE2 = 'SUMMARY FOR' || PUT(symptom, sym.) /,} \]

\[ \text{TITLE and FOOT parameters are quite easy and intuitive to use.} \]

**CONCLUSION**

A program-writer macro FMWRITE is able to read and analyze mock-up file and produce hundreds of report skeleton programs instantly using the table-generator MFORM. Because the macro MFORM is general and flexible, these machine-made skeleton programs can be modified in the framework of MFORM. The great time saving of this mass production therefore continues into the modifying process.

**ACKNOWLEDGEMENTS**

The author would like to express his appreciation to Susan Fehrer for her special review on this paper.

SAS is a registered trademark of SAS Institute Inc. Cary, NC, USA. Other brand and product names are registered trademarks or trademarks of their respective companies.

**AUTHOR CONTACT**

Sam X. Ye  
Statistical Programming  
Corning Besselvar, Inc.  
210 Carnegie Center  
Princeton, NJ 08540
Following is the table produced directly from the machine-made skeleton program rlacct.sas (see AN EXAMPLE section).

<table>
<thead>
<tr>
<th>Treatment Group: [Placebo or drug]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PATIENT DATA LISTING 4</strong></td>
</tr>
<tr>
<td><strong>PATIENT ACCOUNTABILITY</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pat. No.</th>
<th>Visit Period</th>
<th>Days on (Week)</th>
<th>Days on Study+</th>
<th>Days on Open-Label++</th>
<th>Disposition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

+ Relative to the day of randomization.
++ Relative to day of first dose of open-label medication.
DB = double-blind period.
OL = open-label period.

END

Following is a fictitious output after above machine-made program being modified by the programmer (see AN EXAMPLE section).

<table>
<thead>
<tr>
<th>Treatment Group: Placebo</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PATIENT DATA LISTING 4</strong></td>
</tr>
<tr>
<td><strong>PATIENT ACCOUNTABILITY</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pat. No.</th>
<th>Visit Period</th>
<th>Days on Study+</th>
<th>Days on Open-Label++</th>
<th>Disposition</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>DB 2 (1)</td>
<td>6</td>
<td>.</td>
<td>Adverse Experience</td>
</tr>
<tr>
<td>015</td>
<td>DB 4 (4)</td>
<td>24</td>
<td>.</td>
<td>Other</td>
</tr>
<tr>
<td>037</td>
<td>OL 7 (13)</td>
<td>89</td>
<td>23</td>
<td>Protocol Violation</td>
</tr>
</tbody>
</table>

+ Relative to the day of randomization.
++ Relative to day of first dose of open-label medication.
DB = double-blind period.
OL = open-label period.

END
Below is a small part of transferred (from WordPerfect) mock-ups, the "data" file for the Mass Production Approach. The text may look messy, but it is good enough for the macro FMWRITE to read and analyze. The redlined areas show the text which need to be added by the programmer. // indicated the lines which will be used for counting column positions. Two or more spaces signal a new column. ]] indicated the end of a table.

PATIENT DATA LISTING 4
PATIENT ACCOUNTABILITY

Treatment Group: [Placebo or Drug]

Pat. Visit Days on Days on Open-Label++ Disposition
No. Period (Week) Study++

Investigator:

Completed Double-Blind
Adverse Experience
Protocol Violation
Other

+ Relative to the day of randomization.
++ Relative to day of first dose of open-label medication.
DB = double-blind period.
OL = open-label period.

TABLE 27
NUMBER OF PATIENTS WITH ADVERSE EXPERIENCES
BY BODY SYSTEM, ADVERSE EXPERIENCE AND SEVERITY
REGARDLESS OF RELATIONSHIP TO STUDY DRUG
DOUBBLE-BLIND PERIOD

Placebo                Drug
(N=    )              (N=    )
Adverse Experience Mild Moderate Severe Mild Moderate Severe
n (%)    n (%)    n (%)    n (%)    n (%)    n (%)

Total Number of Patients
With Adverse Experience(s)

Body as a Whole
Death

Includes adverse experiences that began during the double-blind period after randomization.
Adverse experiences that began within 4 weeks of discontinuation of double-blind treatment are
included if the patient did not enter the open-label period.
If a patient experience more than one episode of a given experience, than the most severe
experience was included in the summary.

TABLE 56
PATIENTS WITH SERIOUS ADVERSE EXPERIENCES
DOUBBLE-BLIND PERIOD

Treatment Group: [Placebo or Drug]

Pat. Visit Onset Duration Number of Change Relationship Corrective
No. Costart Term (Week) Day+ (D:H:M) Episodes Intensity Dosage Test Drug Test Drug Therapy

Investigator:

[Sort by Treatment, Investigator, Patient, Dictionary, Visit, and Onset Day]

+ Relative to day of randomization
Includes adverse experiences that began during the double-blind period after randomization.
Adverse experiences that began within 4 weeks of discontinuation of double-blind treatment are
included if the patient did not enter the open-label period.