Use ODS Generating Patient Profiles

Haibin Shu, Duramed Research, Bala Cynwyd, PA
Young Zhuge, Duramed Research, Bala Cynwyd, PA

ABSTRACT
There are many approaches to using SAS ODS engines to generate patient profiles for submission or ad hoc report purposes. Many of them involve a manual, lengthy programming effort. Also most of the programming modules cannot be ported into different protocols or have to be modified greatly before they can be reused. At Duramed Research, we found a novel approach to take full advantage of SAS 9.1 features and present patient profiles in a clear, readable format with highly condensed data information. These patient profiles can serve as a reliable and effective data source for references in writing a study report, analyzing both safety data and efficacy endpoints and for communicating between the biostatistician, data management and clinical departments. Most importantly we standardized reporting structures and implemented highly efficient programs. The whole process can be easily ported into different protocols with minimum customization yet with consistent output formats.

INTRODUCTION
We developed a methodology to generate patient profiles using SAS ODS. The methodology establishes a standard data format and takes a systematic approach to convert any source data into this standard data format. We build in rich structures in this standard format by adding a number of ODS in-line formats. All output processing is done through PROC REPORT to ensure a consistent output presentation. On average, a subject’s patient profile spans from about 10 pages (discontinued patients) to 15 pages (study completers) with a header section about important patient information, a body section presenting modularized source data, a footnote section with any necessary annotations, and a pagination section controlling the page break on each page.

HEADER SECTION
Information presented in this section usually is a patient’s important milestone dates (such as first dose date and last dose date), baseline characteristics (such as age and race), visit schedules and end of study status (completed or discontinued). This section always stays at the top of a patient profile page and carries from one page to another. A good header section facilitates better understanding of the CRF data presented in the content or body of the patient profile. Including the visit schedules can be extremely useful when the body section contains AE, Concomitant Medication, Drug Accountability, and Laboratory Result data because people like to look at those data via linking with the corresponding study dates.

We came up with a standard header structure across studies. It consists of baseline demographic data, first and last study drug dates and visit schedule information. Bold font is used (through ODS font code) to highlight variable labels. Below is a simplified version of our header structure:

<table>
<thead>
<tr>
<th>INV/PAT: 01/123(XYZ)</th>
<th>Trx: Placebo</th>
<th>Age (DOB): ## (00-00-0000)</th>
<th>Race: xxxxxxxxx</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consent Date: 00-00-0000</td>
<td>First Dose: 00-00-0000</td>
<td>Last Dose: 00-00-0000</td>
<td>Completed Study</td>
</tr>
<tr>
<td>V0(scr) : 00-00-0000</td>
<td>V1(week #): 00-00-0000</td>
<td>V2(week ##): 00-00-0000</td>
<td>V3(week ##): 00-00-0000</td>
</tr>
<tr>
<td>V4(week ##): 00-00-0000</td>
<td>V5(cos): 00-00-0000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
In this simplified header section, special care should be taken to line up the variables in order to make the overall presentation neat and clear (refer to code example in Appendix I for illustration). Header structure should remain short and straightforward. An ideal structure would be:

- Horizontal – columns are evenly spaced
- Vertical – variable labels are lined up

Dynamically arranging positions of variables is necessary in order to keep the header section in a tight format. We tried to not print out an empty spot if there is no value. Using the above structure as an example, we’ll use only one line to describe the visit schedule instead of two if the subject withdrew after Visit 2:

V0(scr) : 00-00-0000  V1(week #) : 00-00-0000  V2(week ##) : 00-00-0000  V5(et) : 00-00-0000

**BODY SECTION**

The body section contains all the data collected for a patient. An obvious way to organize the data is through data modules (such as Inclusion/Exclusion, AE, Lab Tests) and in the same order as they appear in CRFs. Both CRF data and non-CRF data (such as central Labs) are included. To make it easy to read, do not present more than three modules on one page.

We developed a standard body structure with 6 reportable columns and 2 line holders (refer to code example in Appendix II for illustration). Each data module is transformed to these 6 columns. For landscape page size, 6 seems to be the maximum number of columns that can be nicely reported on one page – vertically lined up and horizontally evenly spaced out among columns. Module names will be printed out at the beginning of a module in a large font size on a separate line (through line holder variable). Column labels occupy another line by the other line holder. Lining up the column headers is an effort-taking task to distinguish “good” programmers from “bad” programmers since those labels are held in one big long variable (SAS 8 or lower versions do not work).

Here is an example:

<table>
<thead>
<tr>
<th>Inclusion Criteria:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>1 xxxxxxxxxxxxxx?</td>
</tr>
<tr>
<td>(Yes)</td>
</tr>
</tbody>
</table>

Lab data can be presented with alert flag. Here is an example of Urinalysis:

<table>
<thead>
<tr>
<th>Laboratory Data (Urinalysis): Blood(+), Specific Gravity, Urine Glucose, Urine Ketones, Urine Protein(3+), pH, Microscopic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visit/Coll Date</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>SCR/26JUN2002</td>
</tr>
<tr>
<td>ET/03MAR2003</td>
</tr>
</tbody>
</table>

The Specific Gravity at 03MAR2003’s ET Visit equals 1.007, which is lower than LLN. “1.007 L” was used to combine both the value and the alert flag.

**FOOTNOTE SECTION**

At the bottom of the body section is a footnote section for a footnote and annotation. It can be used for explanation for special symbols, or text for codes, or notice for protocol amendments.

Followed is an example using the footnote section to indicate when a question is not collected at a visit:
### Laboratory Assessment:

<table>
<thead>
<tr>
<th>Visit</th>
<th>Fast 12 hours?</th>
<th>Clinically Significant?</th>
<th>Lab Repeat?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>*</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

*Not collected.

### PAGINATION SECTION
This section does not have a visible place on the pages, but is very important in controlling where the page will break to ensure that the header section will carry to next new page.

### SUMMARY
The BASIC approach in generating patient profiles comes from standardization of the patient profile pages. A typical page representation in a patient's profile consists of a header section, a body section and a footnote section. The pagination section does not show up on the page but stores controls to break pages and ensures proper carry over. The header section is at the top with demographic information, first and last dose dates, and visit schedule information. The body section contains all collected patient information (CRF and non-CRF) and is presented in 6 columns and sequenced by modules. The footnote section is always at the bottom with annotations or explaining remarks.

Here is an illustration of the relative positions of the four sections:

<table>
<thead>
<tr>
<th>Header</th>
<th>Body</th>
<th>Footnote</th>
<th>Page (invisible)</th>
</tr>
</thead>
</table>

### CONCLUSION
A systematic approach for generating patient profiles through SAS ODS was developed, which paved the way for maximum program sharing. The four sections can be programmed independently therefore generating patient profiles and can be assigned to different programmers. Because data modules are represented by a standard structure, they can also be implemented by an independent programming effort. The method makes the study portable due to its standardized representation and systematic approach.

### REFERENCES
FDA Guidance for Industry (1997), *Archiving Submissions in Electronic Format - NDAs*

Light S, Gilbert P, Genereux G, *A Novel Approach to Developing a Patient Profile Reporting Application, SUGI 26, p042-26*

### TRADEMARKS
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. Other brand and product names are registered trademarks or trademarks of their respective companies.
CONTACT
Your comments are valuable and welcome.

Haibin Shu
Duramed Research
One Belmont Ave, 11th Floor
Bala Cynwyd, PA 19004
HShu@barrlabs.com

Young Zhuge
Duramed Research
One Belmont Ave, 11th Floor
Bala Cynwyd, PA 19004
YZhuge@barrlabs.com
APPENDIX I Example: Line-up Variable Positions

array _vis(7) vis1 vis2 vis3 vis4 vis5 vis5_5 vis6;
/*max visits*/
do i=7 to 1 by -1;
   if _vis(i)>. then do;
      j=i;
      leave;
   end;
end;

if j>=1 then do;
   /*invpat3*/
do i=1 to min(j,4);
      if i=1 then invpat3='\S={font_weight=bold}V1(week -4) :\S={} '||
         %mmddyy(vis1);
      else invpat3=trim(invpatis3)||' \S={font_weight=bold}V'||
         ||%cstr(put(i,1.))||' \S={} '||%cstr(put(i,vis.))||
      ') :\S={} '||%mmddyy(_vis(i));
end;
if sd_tm_dt>. & j<=3 then
   invpat3=trim(invpatis3)||' \S={font_weight=bold}V7('||
   %cstr(put(7,vis.))'||') :\S={} '||%mmddyy(vis7);
/*invpat4*/
do i=(min(j,4)+1) to j;
   if i=6 then invpat4=trim(invpatis4)||
      '\S={font_weight=bold}V5a(week  8):\S={} '||
      %mmddyy(_vis(i));
   else if i=7 then invpat4=trim(invpatis4)||
      '\S={font_weight=bold}V6(week 12) :\S={} '||
      %mmddyy(_vis(i));
   else invpat4=trim(invpatis4)||
      '\S={font_weight=bold}V'||
      %cstr(put(i,1.))||' \S={} '||%cstr(put(i,vis.))||
      ') :\S={} '||%mmddyy(_vis(i));
end;
if sd_tm_dt>. & j>=4 then invpat4=trim(invpatis4)||
   '\S={font_weight=bold}V7('||
   %cstr(put(7,vis.))'||') :\S={} '||%mmddyy(vis7);
invpat4=left(invpat4);
end;
APPENDIX II Example: Generating Standard Data Structure for Patient Profile

/*prepare Inclusion data set for patient profile listing*/
proc sort data=dw.inclus out=ic;
  by inv_no patid;
quit;

data ic2;
  set ic;
  format line1 line2 f1 f2 f3 f4 f5 f6 $1000.;
  array inc(10) inc_1-inc_10;
  array f(6) f1-f6;
  /*section title*/
  line1='Inclusion Criteria:';
  line2='';
  /*panel order*/
  p_ord=2;
  /*lines per record*/
  lpr=6;
  do i=1 to 6;
    f(i)='\S={font_weight=bold}'||%cstr(put(i,8.))||' ' ||
      %cstr(put(i,inc.))||' \S={font_weight=bold}(' ||
      %cstr(put(inc(i),yn.)) ||') \S={}'
  end;
  order+1;
  output;
  do i=7 to 10;
    f(i-6)='\S={font_weight=bold}'||%cstr(put(i,8.))||' ' ||
      %cstr(put(i,inc.))||' \S={font_weight=bold}(' ||
      %cstr(put(inc(i),yn.)) ||') \S={}'
  end;
  f5='';
  f6='';
  order+1;
  output;
  keep order p_ord inv_no patid line1 line2 f1-f6 lpr;
run;

proc sort data=ic2 out=pp.sect2;
  by inv_no patid line1 line2 order;
quit;