ABSTRACT
One of the ongoing debates in the pharmaceutical research universe concerns which tool is best for reporting study results – PROC REPORT or DATA _NULL_. Proponents of PROC REPORT contend the procedure is more user-friendly and the code easier to understand and consequently debug. Those who prefer the DATA _NULL_ approach argue that it allows the user to fully customize reports, thus avoiding the need to tell a customer (internal or external), “Sorry, we can’t put that there.” Regardless of one’s preference, both of these reporting vehicles have their place in the study reporting world. The purpose of this paper is not necessarily to compare the two tools, but rather to (1) present general examples demonstrating which tool is the best fit for a given scenario and (2) offer some tips on both how to enhance the output for each and how to streamline the use of your reporting tool of choice. This presentation is based on SAS version 8.2 or above, is not limited to any particular operating system, and is intended for beginner to intermediate SAS users who have at least some familiarity with either PROC REPORT or DATA _NULL_.

KEYWORDS: SAS, PROC REPORT, DATA _NULL_

INTRODUCTION
Some statistical programming groups in the pharmaceutical research industry may choose to use either the REPORT procedure or the DATA _NULL_ step exclusively to produce their table and listing reports for clinical trial analyses. Other groups may choose to utilize both tools depending upon which method they deem most appropriate for the report to be generated. Within its capabilities, PROC REPORT is a procedure that can be very user-friendly in terms of code-writing, and the amount of code required to produce even reports with many columns can be relatively small. By contrast, using DATA _NULL_ to produce reports may require more SAS code as well as knowledge of other, more esoteric SAS statements and options, but, at the same time, can offer the programmer greater flexibility in presentation of data.

PROC REPORT: EXAMPLES OF APPROPRIATE USAGE
The REPORT procedure lends itself very well to creating listing reports as well as tables that present simple frequency counts.

The following is an example of a patient listing similar to what might be included in a clinical trial report:
And a simple frequency table similar to what would appear in a clinical study report might look like this:

Table 1
Summary of Adverse Events
Safety Population

<table>
<thead>
<tr>
<th>System Organ Class</th>
<th>Study Treatment (N=100)</th>
<th>Placebo (N=100)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preferred Term</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number (%) of Subjects Reporting AEs</td>
<td>32 (32.0%)</td>
<td>41 (41.0%)</td>
</tr>
<tr>
<td>Cardiac disorders</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Palpitations</td>
<td>5 (5.0%)</td>
<td>6 (6.0%)</td>
</tr>
<tr>
<td>Tachycardia</td>
<td>3 (3.0%)</td>
<td>4 (4.0%)</td>
</tr>
<tr>
<td>Ear and labyrinth disorders</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vertigo</td>
<td>5 (5.0%)</td>
<td>7 (7.0%)</td>
</tr>
<tr>
<td>Hyperacusis</td>
<td>2 (2.0%)</td>
<td>1 (1.0%)</td>
</tr>
</tbody>
</table>

Line breaks or page breaks within these types of reports (e.g. page breaks between each different treatment group) can usually be generated via use of the BREAK statement. Also, many reports require that text be inserted prior to a related group of observations in order to identify how they are related (e.g. inserting investigative site information prior to the observations of patients from that particular site). In PROC REPORT, this can be accomplished through the use of COMPUTE blocks. Detailed information about the BREAK statement and COMPUTE blocks is presented later in the section entitled ‘Tips for Making Your PROC REPORT Easier’.

DATA _NULL_: EXAMPLES OF APPROPRIATE USAGE
Tables that include blocks of statistics such as the mean, standard deviation, LS Means, and p-values often are easily produced using the DATA _NULL_ method. The following is an example of such a report:
Table 2
Summary of Efficacy Scores by Visit

<table>
<thead>
<tr>
<th></th>
<th>Study Treatment (N=100)</th>
<th>Study Placebo (N=100)</th>
<th>Overall P-value vs. Placebo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>Mean (SD)</td>
<td>4.6 (0.61)</td>
<td>4.7 (0.69)</td>
<td></td>
</tr>
<tr>
<td>Median</td>
<td>5.0</td>
<td>5.0</td>
<td></td>
</tr>
<tr>
<td>Min-Max</td>
<td>4 – 6</td>
<td>4 – 6</td>
<td></td>
</tr>
<tr>
<td>Week 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>97</td>
<td>98</td>
<td></td>
</tr>
<tr>
<td>Mean (SD)</td>
<td>2.8 (1.27)</td>
<td>2.6 (1.09)</td>
<td></td>
</tr>
<tr>
<td>Median</td>
<td>3.0</td>
<td>2.0</td>
<td></td>
</tr>
<tr>
<td>Min-Max</td>
<td>1 – 6</td>
<td>1 – 6</td>
<td></td>
</tr>
<tr>
<td>Change from Baseline to Week 1</td>
<td>0.0368</td>
<td></td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>97</td>
<td>98</td>
<td></td>
</tr>
<tr>
<td>Mean (SD)</td>
<td>-1.8 (1.31)</td>
<td>-2.1 (1.16)</td>
<td></td>
</tr>
<tr>
<td>LSMean</td>
<td>-1.9</td>
<td>-2.0</td>
<td></td>
</tr>
<tr>
<td>Median</td>
<td>-2.0</td>
<td>-2.0</td>
<td></td>
</tr>
<tr>
<td>Min-Max</td>
<td>-5 – 1</td>
<td>-5 – 1</td>
<td></td>
</tr>
<tr>
<td>Difference in LSMean Change</td>
<td></td>
<td>-0.2</td>
<td></td>
</tr>
<tr>
<td>95% C.I.</td>
<td>-0.5, 0.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P-Value</td>
<td>0.4121</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The DATA _NULL_ step is effective for producing tables such as these since the description of a particular statistic (e.g. 'Difference in LSMean Change') may sometimes span more than one column in the table. Additionally, it is often desirable in such reports to present a complete block of statistics for a particular parameter (e.g. an efficacy measurement by study visit) together on one page rather than breaking the block up across two separate pages. The DATA _NULL_ methodology typically can accomplish this in a much more dynamic fashion than PROC REPORT.

THERE ARE EXCEPTIONS
The examples provided previously for using PROC REPORT vs. DATA _NULL_ can be followed for the production of most tables and listings. But exceptions to these do, of course, exist. For instance, in the creation of listings, many sponsors may prefer to see the investigator name and/or number as a separate line prior to the listing of the patients from that particular site. This in itself is easy to accommodate in PROC REPORT by using a COMPUTE block. If however the listing of patients for a particular clinical site spans more than one page, the sponsor may also want to see the investigator name and/or number at the top of each subsequent page followed by something like '(cont’d)'. An example of this is provided below:
Concomitant Medications
Study Treatment Group

<table>
<thead>
<tr>
<th>Subject</th>
<th>Medication</th>
<th>Total Dose</th>
<th>Start Date</th>
<th>Stop Date</th>
<th>Indication</th>
</tr>
</thead>
<tbody>
<tr>
<td>John Smith, MD</td>
<td>IBUPROFEN</td>
<td>200 MG</td>
<td>12JAN2006</td>
<td>Continuing</td>
<td>HEADACHE</td>
</tr>
<tr>
<td>(02) (cont’d)</td>
<td>PARACETAMOL</td>
<td>2 TABS PRN</td>
<td>12JAN2006</td>
<td>17JAN2006</td>
<td>SINUS PAIN</td>
</tr>
<tr>
<td></td>
<td>RANITIDINE</td>
<td>300 MG QD</td>
<td>30OCT2005</td>
<td>16DEC2005</td>
<td>GERD</td>
</tr>
<tr>
<td>020005</td>
<td>VICODIN</td>
<td>500 MG PRN</td>
<td>07JUL2005</td>
<td>10JUL2005</td>
<td>PAIN FROM TOOTH EXTRACTION</td>
</tr>
</tbody>
</table>

Producing a report that meets this criterion can be easily accomplished with the DATA _NULL_ method. An example of DATA _NULL_ code that can produce this result will be provided and discussed later.

Conversely, the type of table containing blocks of statistics that is provided previously as an excellent case for using DATA _NULL_ can also be produced using the REPORT procedure. The disadvantage here to using PROC REPORT is that the code tends to be more static than that which would be created in a DATA _NULL_ step. For example, one might need to know exactly how many lines of output could be printed to a particular output page or exactly what the widths of table columns are prior to writing the PROC REPORT code (i.e. any changes that occur to the table, such as added footnotes or additional titles, might require more tweaking in PROC REPORT vs. DATA _NULL_).

**TIPS FOR MAKING YOUR PROC REPORT EASIER**

There are a number of ways in which rudimentary PROC REPORT output can be enhanced to achieve the expected outcome. Two of the most useful tools that can be used to do this include the COMPUTE block and the BREAK statement.

**COMPUTE Blocks.** The COMPUTE block is an excellent tool that PROC REPORT provides to insert characters or text into your report. Footer lines, left-justified subtitles, and identifying text (e.g. investigative site information that precedes a block of subjects sharing that particular clinical site) are all examples of items that can be easily produced through the use of a COMPUTE block.

**BREAK Statement**

The most common uses of the BREAK statement are to produce line breaks between observations (or sets of observations) and page breaks between different categories of observations.
Applying the COMPUTE Block and BREAK Statement
Consider the following demographics listing:

Listing 3
Demographic Characteristics

Treatment Group: Study Treatment

<table>
<thead>
<tr>
<th>Subject</th>
<th>Age (yrs)</th>
<th>Gender</th>
<th>Race</th>
</tr>
</thead>
<tbody>
<tr>
<td>010001</td>
<td>82</td>
<td>Male</td>
<td>White or Caucasian</td>
</tr>
<tr>
<td>010002</td>
<td>66</td>
<td>Female</td>
<td>White or Caucasian</td>
</tr>
<tr>
<td>010003</td>
<td>69</td>
<td>Female</td>
<td>White or Caucasian</td>
</tr>
<tr>
<td>010004</td>
<td>70</td>
<td>Male</td>
<td>Hispanic or Latino</td>
</tr>
<tr>
<td>010005</td>
<td>64</td>
<td>Male</td>
<td>Black or African American</td>
</tr>
<tr>
<td>010006</td>
<td>68</td>
<td>Female</td>
<td>White or Caucasian</td>
</tr>
</tbody>
</table>

Site Number 02

<table>
<thead>
<tr>
<th>Subject</th>
<th>Age (yrs)</th>
<th>Gender</th>
<th>Race</th>
</tr>
</thead>
<tbody>
<tr>
<td>020001</td>
<td>70</td>
<td>Male</td>
<td>Black or African American</td>
</tr>
<tr>
<td>020002</td>
<td>47</td>
<td>Female</td>
<td>Hispanic or Latino</td>
</tr>
<tr>
<td>020003</td>
<td>63</td>
<td>Female</td>
<td>White or Caucasian</td>
</tr>
<tr>
<td>020004</td>
<td>41</td>
<td>Male</td>
<td>Hispanic or Latino</td>
</tr>
<tr>
<td>020005</td>
<td>75</td>
<td>Male</td>
<td>White or Caucasian</td>
</tr>
<tr>
<td>020006</td>
<td>61</td>
<td>Female</td>
<td>White or Caucasian</td>
</tr>
</tbody>
</table>

......

NOTE: Patients who failed screening are not included.

The code to produce this listing in PROC REPORT might look like the following:

```sas
%let lsize=120;
options ls=&lsize;
data _null_;  
  line=repeat('_',&lsize);  
  call symput('ftline',line);  
run;
proc sort data=analysis.demo out=demo;  
  by trtgrpcd siteid subjid;  
run;
proc report data=demo center missing headline headskip nowd spacing=19;  
  column trtgrpcd trtgrp siteid subjid age sex race;  
  define trtgrpcd / group NOPRINT;  
  define trtgrp / group NOPRINT;  
  define siteid / group NOPRINT;  
  define subjid / width=9 "Subject";  
  define age / center width=6 "Age/(yrs)";  
  define sex / width=6 "Gender";  
  define race / width=42 "Race";
```
NOTE: For this example, the input dataset DEMO must first be sorted by TRTGRPCD, SITEID, and SUBJID. The variables TRTGRPCD and SITEID must then be designated as group variables in order for the COMPUTE blocks and BREAK statements to execute correctly.

**Discussion of PROC REPORT Code**
The first COMPUTE block above instructs PROC REPORT to produce the treatment group (in this case ‘Study Treatment’) left-justified between the table titles and the body of the report and to skip a line thereafter. (NOTE: This is a task that can alternatively be accomplished using #BYVAL methodology.) The next COMPUTE block serves to print site information (Site Number XX) prior to the display of the subjects enrolled from that site. The last COMPUTE block adds the footer line and footer at the end of each page.

**TIP:** The ‘before _page_’ syntax means prior to the body of the report (i.e. after any explicit titles) while ‘after _page_’ executes at the end of the body of the report (i.e. prior to any footnotes produced with a FOOTNOTE statement).

The initial BREAK statement in the above PROC REPORT instructs SAS to skip to a new page upon encountering a new treatment group (from prior examples, this would be the placebo group). The second BREAK statement produces a blank line after each group of site observations.

**TIPS FOR A MORE EFFORTLESS DATA _NULL_**
As previously stated, there are times when PROC REPORT may not easily produce the desired result for your table or listing and, in most of these cases, the DATA _NULL_ method is an excellent alternative. With the help of some basic SAS code and/or custom macros, the DATA _NULL_ step can be used to accomplish a number of relevant report tasks, including (1) dynamically determining whether a block of report text (e.g. statistics) will fit on the remaining lines of a page, (2) producing columns with appropriate spacing and widths, and (3) creating upper and lower table boundary lines.
Producing a Typical Report with DATA _NULL_
The following is an example of a listing that might be produced with the DATA _NULL_ method:

Listing 4
Brief Physical Examination
Study Treatment Group

<table>
<thead>
<tr>
<th>Subject</th>
<th>Visit</th>
<th>Date of Visit</th>
<th>Brief PE Performed?</th>
<th>Abnorm- alities?</th>
<th>Abnormality Findings</th>
</tr>
</thead>
<tbody>
<tr>
<td>James Thomas, MD (05) (cont’d)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>050003</td>
<td>Baseline</td>
<td>21NOV2005</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Week 1</td>
<td>29NOV2005</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Week 2</td>
<td>05DEC2005</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Week 3</td>
<td>12DEC2005</td>
<td>Yes</td>
<td>Yes</td>
<td>HIGH BLOOD PRESSURE</td>
</tr>
<tr>
<td></td>
<td>Week 4</td>
<td>19DEC2005</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>050004</td>
<td>Baseline</td>
<td>17AUG2005</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Week 1</td>
<td>24AUG2005</td>
<td>Yes</td>
<td>Yes</td>
<td>FOOT AND ANKLE EDEMA</td>
</tr>
<tr>
<td></td>
<td>Week 2</td>
<td>31AUG2005</td>
<td>No</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

And the SAS code to create this report in DATA _NULL_ would look something like this:

```sas
%let linesze=90;

*** SET UP TABLE COLUMNS ***;

%let numcols=6;
%let colspace=2;
%let colwd1=8;
%let colwd2=8;
%let colwd3=9;
%let colwd4=10;
%let colwd5=8;
%let colwd6=37;
%let colhd1=Subject;
%let colhd2=Visit;
%let colhd3=Date of Visit;
%let colhd4=Brief PE Performed?
%let colhd5=Abnorm-alities?
%let colhd6=Abnormality Findings
%setcols;

*** SORT BPE DATA FOR STUDY TREATMENT GROUP & DETERMINE ***
*** NUMBER OF LINES IN EACH SUBJECT GROUPING ***;

proc sort data=analysis.bpe out=bpe;
  by siteid subjid;
  where trtgrpcd=1;
run;
```
data numlines (keep=siteid subjid numlinz);
   set bpe;
   by siteid subjid;
   if first.subjid then numlinz=1;
   else numlinz+1;
   if last.subjid;
run;

data final;
   merge bpe
      numlines;
   by siteid subjid;
run;

*** CREATE THE LISTING ***;

title 'Listing 4';
title2 'Brief Physical Examination';
title3 'Study Treatment Group';
%macro null;

filename temp "bpe.out";

data _null_; 
   set final end=lastobs;
   by siteid subjid;
   file temp n=1
      linesleft=remlines
      print;
   retain pageno newpage 0;
   if _n_=1 then link header;
   *** DETERMINE IF ENTIRE PATIENT BLOCK WILL FIT ON REMAINING PAGE ***;
   if first.subjid and numlinz>remlines-10 then do;
      link footer;
      link header;
   end;
   *** INCLUDE SITE INFORMATION PRIOR TO ITS GROUP OF PATIENTS ***;
   if first.siteid then put / @&c1 siteinfo;
   if newpage=1 and ^first.siteid then put / @&c1 siteinfo "(cont'd)";
   newpage=0;
   if first.subjid and ^first.siteid then put / @;
   if ^first.subjid then coll='';
   %do cidx=1 %to &cnum;
      put @&c1&cidx col&cidx $char&cw&cidx... @;
   %end;
   put;
   if lastobs then link footer;
   else if remlines<10 then do;
      link footer;
      link header;
   end;
return;
Discussion of DATA _NULL_ Code

All code relevant to the tasks discussed is highlighted above in bold. The first block of bolded code assigns a series of macro variable values to set column attributes, including the number of columns (NUMCOLS), the spacing between each column (COLSPACE), the width of each column (COLWDx), and the headers for each of the six table columns (COLHDx). These in turn are used by the called macro %SETCOLS to create the columns (in this case the macro creates the variables COL1 – COL6 and assigns each column’s attributes based on the above macro variable values as well as specified font size, margins, etc.) in the DATA _NULL_. The product of the macro is then used by the blocks of code beginning ‘%do cidx=1 %to &cnum;’ and ‘%do hdidx=1 %to &maxhdrs;’ to respectively populate the columns and apply the column headers. Although a detailed discussion of such a macro is beyond the scope of this paper, this is an excellent example of a custom tool that can be created by the user to make the DATA _NULL_ step easier and more transportable across projects. This can also help to keep the DATA _NULL_ code more static (i.e. less tweaking across programs and/or projects).

Oftentimes in tables or listings such as the above example, the sponsor may want to keep all patient records together in one block for display in a report (as opposed to splitting the records across pages). The two DATA STEPs highlighted prior to the DATA _NULL_ are the first movement toward accomplishing this goal. In short, these DATA STEPs determine the total number of records in each patient block (by creating the variable NUMLINZ). The DATA _NULL_ step then uses this information in conjunction with the LINESLEFT= option on the FILE statement to determine if the entire block of patient information will fit on the remaining available lines of the page (in this instance, REMLINES-10). If the entire block will not fit then it is displayed on a fresh page.
As discussed previously, sponsors sometimes prefer to see site or investigator information displayed prior to that particular site’s group of patient records. This is easily done with the code ‘if first.siteid then put / @&c1 siteinfo;’ which simply instructs SAS to include the site/investigator information first upon encountering the respective site’s initial patient record. The sponsor may also however want to display the site information at the top of each subsequent page of that site’s patient records with a trailing ‘(cont’d)’ or ‘(cont.)’, and this is accomplished via use of the code ‘if newpage=1 and ^first.siteid then put / @&c1 siteinfo "(cont’d)";’. The variable NEWPAGE is created as a flag variable and set to a value of 1 in the HEADER section to let the DATA _NULL_ know that output is being written to a fresh page (note that this value is reset to 0 after execution of the above code). If SAS detects a NEWPAGE value of 1 and determines that the next patient record to be printed is not the first observation for its respective site, then the site/investigator information is displayed at the top of the new page followed by text to indicate continued (in this case, ‘(cont’d)’).

Lastly, the above DATA _NULL_ example creates the boundary lines that appear in the listing with the code ‘put @&c1 &linesze*_’;’ that occurs within the HEADER and FOOTER sections.

**CONCLUSION**

In summary, both PROC REPORT and DATA _NULL_ are excellent means for creating tables and listings in support of clinical trial reports. The REPORT procedure typically works very well for producing listings and frequency tables while the DATA _NULL_ method can be more accommodating for more complex tables (and sometimes listings). Customized macros can be created to work with either PROC REPORT or DATA _NULL_ to perform such tasks as creating and applying titles and footnotes (as well as report boundary lines), setting up table parameters (e.g. page and line sizes, font sizes, margins, etc), creating columns and column attributes, and producing page numbers in a desired format and location. Statistical programming groups that wish to implement just one of these reporting tools as their standard for producing tables and listings (as well as those who like to use both) can achieve desired results through the use of such macros as well as a number of other complimentary SAS tools.

**ACKNOWLEDGEMENTS**

The authors would like to extend a special thanks to Jackie Lane, Alla Kapralova, and Robert Stemplinger for their input to and review of this paper, as well as their ongoing support for progress, creativity, and innovation.
CONTACT INFORMATION
The authors can be contacted at:

David W. Carr
ICON Clinical Research
555 Twin Dolphin Drive, Suite 400
Redwood City, CA 94065
carrd@iconus.com

Lanie Quintana
ICON Clinical Research
555 Twin Dolphin Drive, Suite 400
Redwood City, CA 94065
quintanal@iconus.com

Akari Yamagami
ICON Clinical Research
555 Twin Dolphin Drive, Suite 400
Redwood City, CA 94065
yamagamia@iconus.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.