Proc Format Advanced Techniques: Multi-label and Nested Formats

Tatiana Nevmyrych, PPD, Morrisville, NC
Jennifer Clark, PPD, Morrisville, NC

ABSTRACT
Most of us are aware of basic PROC FORMAT options, like `value/invalue` to create user-defined formats. But there are lesser known format options that enhance the flexibility of programs that are often overlooked. The use of a multi-label format in conjunction with the preloadfmt option in PROC MEANS, TABULATE and SUMMARY allows us to include all possible combinations of formatted class variable values to create group/total counts, as well as to get counts for every value in a format regardless of if it exists in the data. This is useful for various demographic, or other types of summaries so frequently utilized in reports in our industry. These format options help to avoid unnecessary data doubling, and creation of dummy datasets, which saves quite some processing time.

The paper also addresses nested formats (i.e. referring to another existing format when creating new format `invalue`). We will illustrate the power of these techniques with examples.

INTRODUCTION
There are always numerous ways to achieve the same result in SAS - some require numerous data/proc steps while others make your life so much easier if you know the shortcut. Manipulation with formats is one of those shortcuts. This paper is going to demonstrate how to get counts of categorical data, even if some of the categories do not exist in the original data (like missing race, gender, subtotals, totals). We will also briefly overview the notion of nested formats.

MULTI-LABEL FORMATS
Suppose you have some simple demographics data. There are 3 treatment groups in this study, but there are no patients in the treatment 3 group for this particular subset. Let’s say we need to count patients by gender, race, all 3 treatment groups, and subtotal (treatment 1 and 3) and total (treatments 1, 2 and 3). The frequently used method of counting subtotals is to double the data as illustrated in Method 1 – which can be inefficient and time-consuming:

METHOD 1 : DATA DOUBLING

Patient Demographic Data (MYDATA SAS™ Dataset)

<table>
<thead>
<tr>
<th>SAFE</th>
<th>PT</th>
<th>TRT</th>
<th>SEX</th>
<th>RACE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>001</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>002</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>003</td>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>004</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>005</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>006</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>007</td>
<td>1</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>008</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>009</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>010</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

This code creates a new dataset that is 3 times as large as the original data.

```sas
data addtrt;
  set mydata;
  if trt in(1,2,3) then newtrt=trt;
  output;
  if trt in(1,3) then newtrt=4;
  output;
  if trt (1,2,3) then newtrt=5;
  output;
run;
```

Obtain counts

```sas
proc means data=addtrt
  noprint completetypes nway;
  class newtrt;
  var safe;
  output out=bign
    (drop=_type_ _freq_) n=N;
run;
```
Note that the zero count for treatment 3 is missing. In order to add it to the output one would have to create a dummy data set with zero counts and merge it back with this BIGN output dataset as illustrated below:

<table>
<thead>
<tr>
<th>NEWTRT</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>

These are two additional steps to complete a simple task.

METHOD 2: MULTI-LABEL FORMAT WITH PROC MEANS PRELOAD OPTION

Patient Demographic Data (MYDATA SAS™ Dataset)

<table>
<thead>
<tr>
<th>SAFE</th>
<th>PT</th>
<th>TRT</th>
<th>SEX</th>
<th>RACE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>001</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>002</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>003</td>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>004</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>005</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>006</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>007</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>008</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>009</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>010</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

New BIGN Output Dataset

Obtain counts adding preloadfmt & mlf options as highlighted

```
proc format;
  value trt (multilabel)
    1 = '1Treatment 1'
    2 = '2Treatment 2'
    3 = '3Treatment 3'
    1,3 = '4Subtotal'
    1,2,3 = '5Total';
  value sex
    1 = '1Male'
    2 = '2Female';
run;
```

```
proc means data=dummy
  noprint
  completetypes nway;
  format sex sex. trt trt.;
  class trt sex /preloadfmt mlf;
  var sex;
  output out=bign1
    (drop=_type_ _freq_) n=N;
```
The preloadfmt option along with mlf generates counts of all values of sex by treatment group defined originally in proc format even if values of some categories didn’t exist in the original data. Thus, even though there were no patients in treatment group 3, we still have zero counts for it in the output dataset because treatment 3 was accounted for in proc format. Likewise we did not have any females in treatment group 1 in our original data – however our final dataset contains a zero count for this category. Hence, there is no need to create unnecessary dummy datasets with zero counts.

NESTED FORMATS
Starting from SAS V8 nested formats can be used to avoid having multiple copies of codes or look-up values.

In this instance a tumor scan dataset has one record for each patient. The variable RESPONSE identifies the type of response that was recorded after the scan was done. Meaning of responses can be seen in the response. format. Values of RESPONSE 1 - 5 are recorded when the scan was actually done, but only 1and 2 qualify for OBJECTIVE response. We want to create an indicator/count variable for every patient to show whether they had an objective response and if they had a scan done.

```sas
data counts;
  set responses;
  objective = input(response, object.); /* variable to indicate objective response*/
  scan = input(response, scan.); /* variable to indicate if the scan was done*/
```

The tumor scan dataset has one record for each patient. The variable RESPONSE identifies the type of response that was recorded after the scan was done. Values of RESPONSE 1 - 5 are recorded when the scan was actually done, but only 1 and 2 qualify for OBJECTIVE response. We want to create an indicator/count variable for every patient to show whether they had an objective response and if they had a scan done.
The advantage of nested formats in this case, is that it enables code values to be defined once rather than be repeated in multiple formats. Also note that nesting more than 2 or 3 deep can impact performance - even if you can keep track of the logic.

CONCLUSION
This paper illustrated just a couple of ways that formats can be used to increase the efficiency, simplicity, and flexibility of programming. When obtaining counts for categorical data, consider using multi-label formats along with the preload option in proc means, tabulate or summary to get counts for all categories listed in a defined format – including total and subtotal counts. In addition – in order to avoid having to repeat values in multiple formats – consider using nested formats to save time.

CONTACT INFORMATION
Your comments and questions are valued and encouraged. Contact the authors at:

Tatiana Nevmyrych
PPD
3900N Paramount Pkwy
Morrisville, NC
Email: Tatiana.Nevmyrych@rtp.ppdi.com

Jennifer Clark
PPD
3900N Paramount Pkwy
Morrisville, NC
Email: Jennifer.Clark@rtp.ppdi.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.