

Using Multiple Output Statements in the Summary Procedure

Britney D. Gilbert, PPD, Inc., Wilmington, North Carolina

ABSTRACT

Multiple Output Statements in the Summary procedure is not commonly documented syntax; however, it is useful in programming. Using multiple output statements with a data step can accomplish a vertical dataset structure, eliminating the need for further procedures (e.g. Proc Transpose), by defining specific conditions and statistics for each output statement. Conditions can be defined for each output statement using Where Statements involving the Class variables and or the default variable `_TYPE_` that the Summary Procedure produces coupled with a list of statistical options. This is helpful when combining descriptive statistics, using multiple denominators in dataset algorithms, or needing quality data checks within a program.

INTRODUCTION

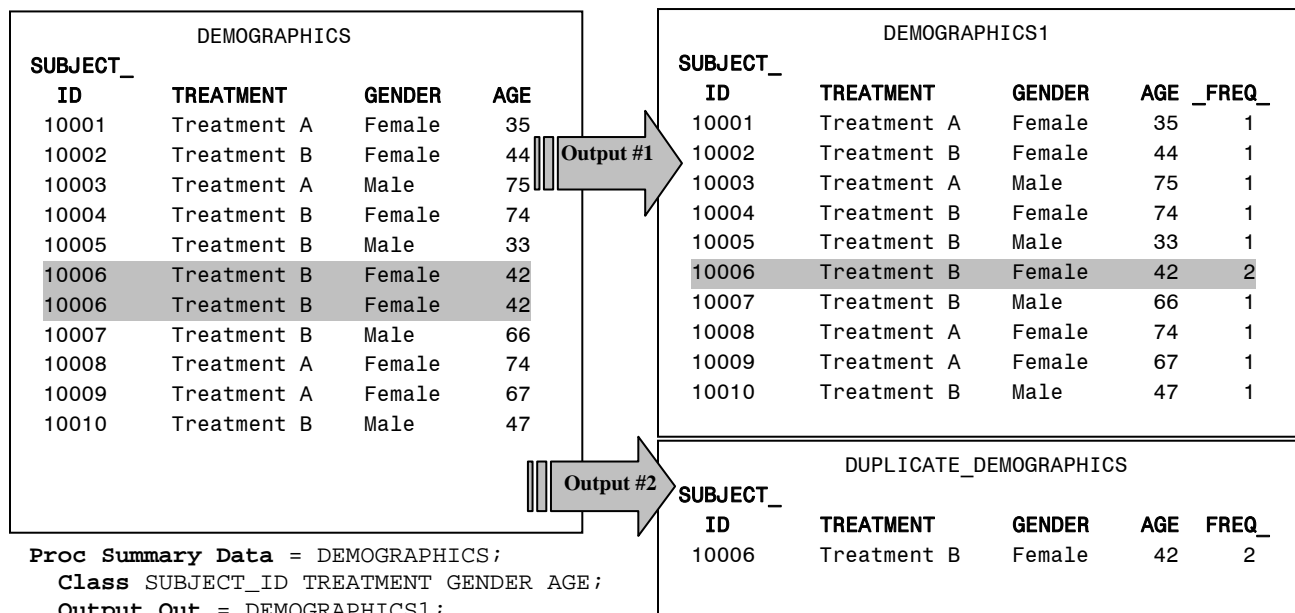
When programming a table with descriptive statistics, there are often times when the same data needs to be rearranged or subsetting more than once. Programming multiple output statements in the summary procedure can save time and eliminate steps to accomplish this. This paper will present several examples of usage for multiple output statements – presenting descriptive statistics, organizing multiple counts, and programming quality checks.

PROGRAMMING QUALITY CHECKS

The usage of multiple output statements in the summary procedure can aid in programming quality checks without disrupting the code's flow.

In Example 1, a duplicate record exists within the demographics data set. This may be an issue where a quality check is needed and the duplicate record omitted. The first output statement eliminates the duplicate record. The second output statement captures the duplicate by using the default variable `_FREQ_` and stores it in a separate data set for review.

Example 1. Quality Data Checks



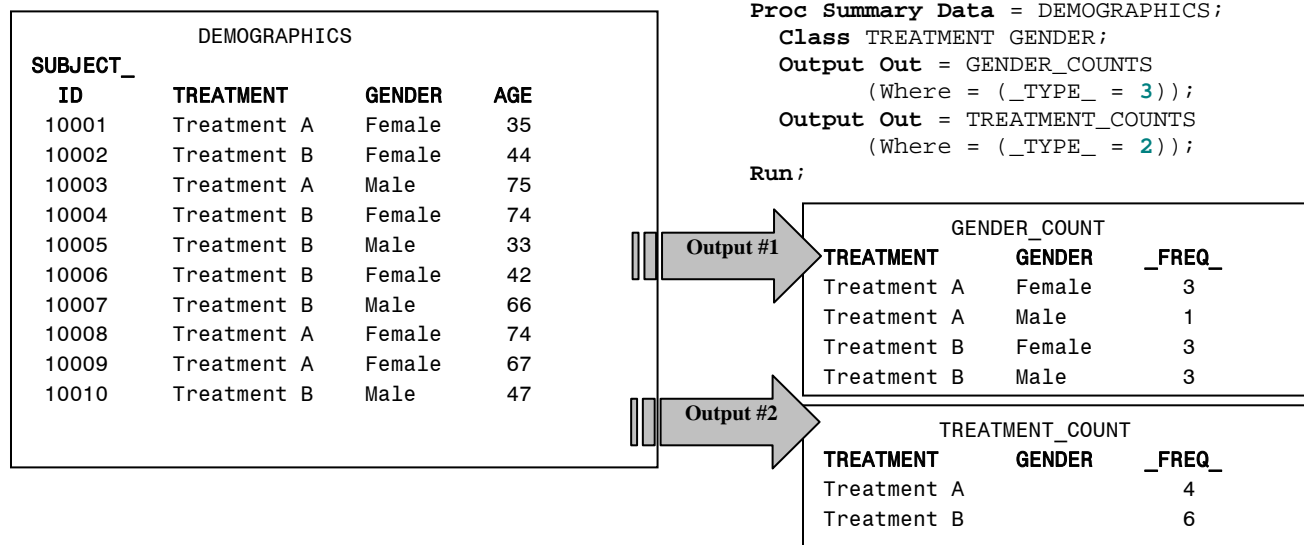
ORGANIZING MULTIPLE COUNTS

Using multiple output statements in the summary procedure is useful when organizing multiple counts. For each output statement, the programmer can take advantage of the default variable `_TYPE_` and data step options, e.g. KEEP, DROP, RENAME, WHERE, etc.

Paper CC17

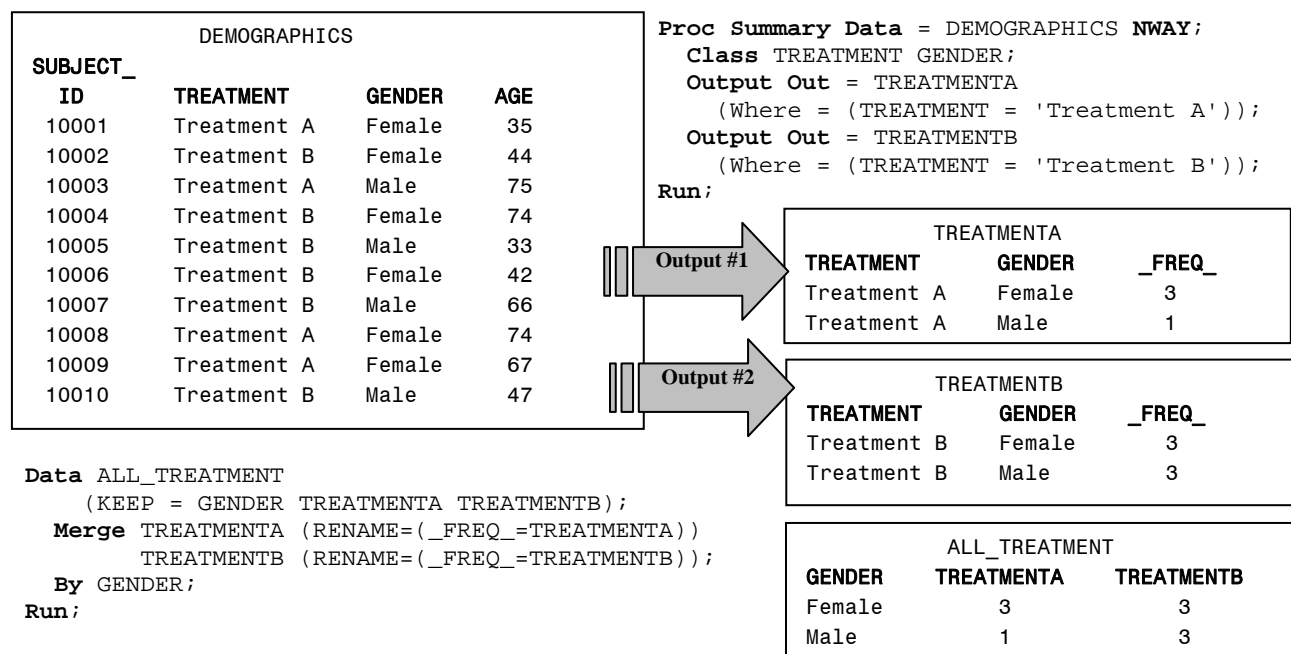
In Example 2, two data sets are created from one summary procedure. From the first output statement, gender counts for each treatment are created. The second output statement creates the overall counts for each treatment. Now, calculating percentages, etc. can be better managed in a program.

Example 2. Overall and Subtotal Counts



In Example 3, again, two data sets are created. From the first output statement, only counts for Treatment A are stored. The second output statement creates counts for Treatment B. Now, these two data steps can be merged in a data step, eliminating the need for a transpose procedure.

Example 3. Combining Counts



Paper CC17

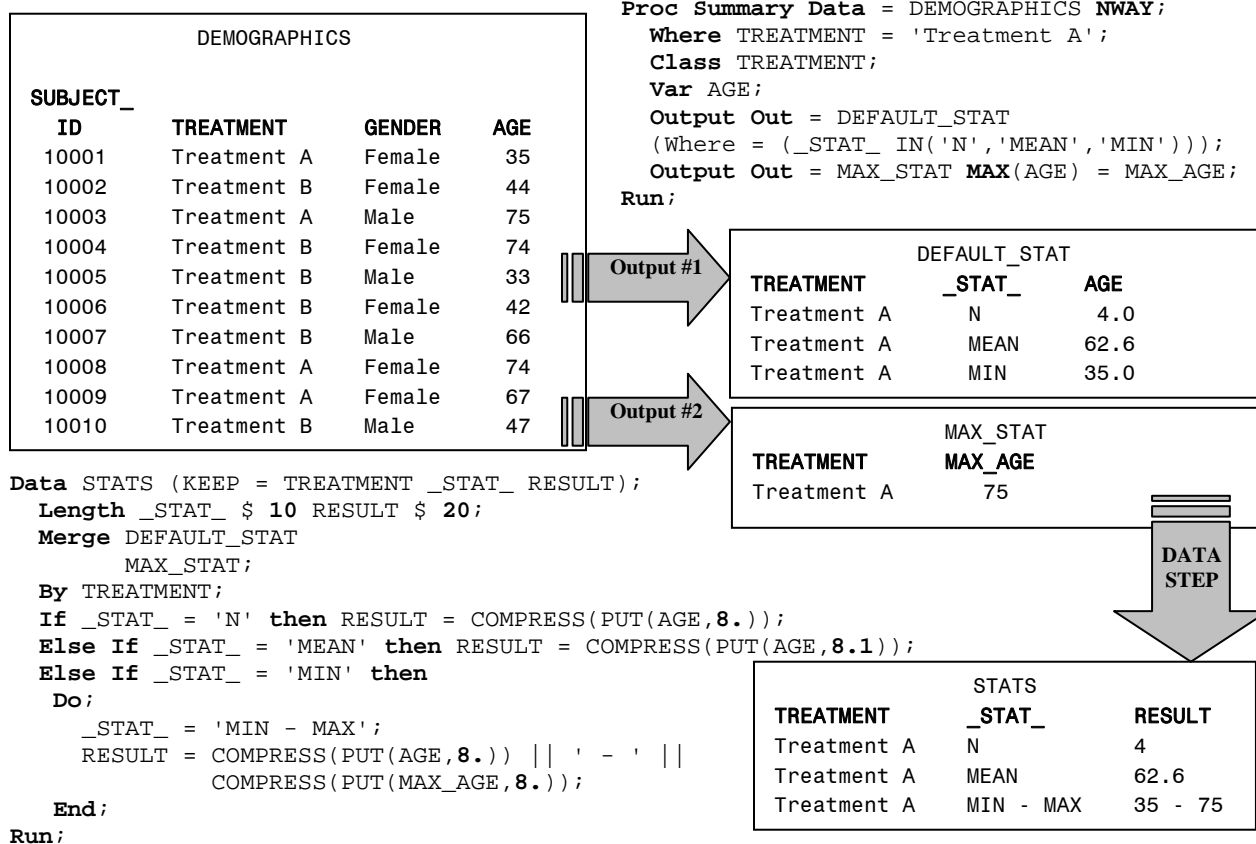
When “counting” more than one variable for each treatment, this style can eliminate multiple transpose procedures and potentially further data steps.

PRESENTING DESCRIPTIVE STATISTICS

Using multiple output statements in the summary procedure is useful when presenting descriptive statistics. Each default statistic in the summary procedure will appear in a separate row creating a vertical table structure. Using additional output statements defining specific statistical options, the programmer can combine the output datasets and manipulate the display of values. This style of programming will eliminate the use of the transpose procedure and potentially addition data steps.

In Example 4, two data sets are created for the Summary procedure for Treatment A. From the first output statement, selected default statistics (N, MEAN, MIN) are created. From the second output statement, the maximum statistical option has been specified, and a separate data set has been created. To present the range, these two data sets have been merged and the minimum and maximum concatenated.

Example 4. Presenting Descriptive Statistics



CONCLUSION

This paper has demonstrated the just a few applications of multiple output statements in the Summary procedure. There are many more applications. The few presented here can be expanded to make a programmer's code more efficient and concise.

CONTACT INFORMATION

Britney D. Gilbert
 Statistical Programmer/Analyst
 PPD, Inc.
 929 North Front Street
 Wilmington, NC 28401-3331
 (910) 558-7091
 (910) 654-1057
britney.gilbert@wilm.ppd.com