Paper 319-2013

Reordering Columns after PROC TRANSPOSE (or anytime you want, really)

Tony Yiu, Kaiser Permanente Southern California, Pasadena, CA

ABSTRACT

There are times when we want to rearrange the columns of a SAS data set in a particular order. This occurs most often after a PROC TRANSPOSE procedure, when the newly transposed columns do not appear in the order that we want. This paper shows how users can either sort the columns by their names, or put the columns in any custom order.

INTRODUCTION

Why would we want to reorder the columns in a SAS data set? The same reason we sort rows by using the PROC SORT procedure. We use PROC SORT to sort the rows into a meaningful order so that it can be analyzed more effectively. This is also true for columns. Sometimes columns need to be sorted into meaningful order as well. We can sort the columns by putting their column names in a sorted (alphabetical) order.

THE PROBLEM

To be honest, sorting columns is not something that we do every day. Based on personal experience, this is most commonly done after a PROC TRANSPOSE. For those who don't know, PROC TRANSPOSE is a powerful tool for converting variables into observations or observations into variables. To better explain this, consider the following example:

Salesman	Day	Units_Sold
Alice	Monday	10
Alice	Thursday	6
Alice	Friday	5
Bob	Monday	7
Bob	Wednesday	6
Bob	Friday	8
Chris	Tuesday	15
Chris	Wednesday	7
Chris	Thursday	8

Table 1. Input Table

There are three employees who each work in a retail electronic store three times a week. Table 1 shows how many computers each one have sold on any particular day. The following SAS code transposes this data.

```
proc transpose data=input out=output;
  by salesman;
  var units_sold;
  id day;
run;
```

The transposed data set looks like this:

Salesman	Monday	Thursday	Friday	Wednesday	Tuesday
Alice	10	6	5		
Bob	7		8	6	
Chris		8		7	15

Table 2. Output Table

However, notice the columns are not in chronological order. Ideally, we want Monday to be followed by Tuesday, then Wednesday, Thursday and Friday. Why doesn't SAS do that automatically?

When SAS reads the first observation of the input data set **(Alice, Monday, 10)**, the day is Monday. Therefore SAS puts Monday to be the first column of the output data set. The next new day it sees is Thursday from the second observation. Therefore Thursday is the second column of the output data set. Following this logic, the next new day is Friday from the third observation, then Wednesday from the fifth observation, and finally Tuesday from the seventh observation. In other words, the column order of the output data set is dependent on the order they appear in the input data set.

THE FIX

When the number of columns is small, the simplest solution is to use the RETAIN statement in the DATA step. All we need to do is list the columns in the desired order in the RETAIN statement. The RETAIN statement must be placed before the SET statement.

```
data fix1;
  retain salesman monday tuesday wednesday thursday friday;
  set output;
run;
```

Alternatively, we can use PROC SQL to do the same thing. Similar to using the RETAIN statement, we list the columns in the desired order in the SELECT clause of the SQL statement.

```
proc sql;
  create table fix2 as
  select salesman, monday, tuesday, wednesday, thursday, friday from output;
quit;
```

Both methods produce the exact same result. It is up to the programmer's preference to choose either one.

These methods work well in this particular example, because the number of columns is small. However, imagine if the same input data set is now used for each day of the month. There are now thirty columns in the transposed data set. Listing each column name becomes less desirable. What if the input data set is used for each day of the year? Is it still a good idea to type the names of over three hundred columns?

THE OTHER FIX

At some point, listing each column name becomes infeasible and impractical. Fortunately, there is another way to reorder a large number of columns. Its underlying logic can be broken down into four steps.

1. Save the column names into a separate SAS data set
2. Sort this data set alphabetically
3. Create a macro variable that stores the content of the sorted data set
4. Put the macro variable in the RETAIN statement in a DATA step

Using the same example, suppose the input data set now stores the same information for each day of the year. The transposed data set will have 365 columns that need to be reordered. The columns are named **day001**, **day002**, **day003** and so on, all the way to **day365**. Here is the SAS code to reorder these columns:

```
proc contents data=output out=col_names(keep=name) noprint;
run;
```

The OUT= option in the PROC CONTENTS procedure outputs the attributes of each column of a data set. The name of the column is saved in a variable called NAME. This is the only variable that we are interested in.

```
proc sort data=col_names out=col_names_sorted;
  by name;
run;
```

This step sorts the column names in alphabetical order.

```
data _null_;
  set col_names_sorted;
  by name;
  retain sorted_cols;
  length sorted_cols $2500.;

  if _n_ = 1 then  sorted_cols = name;
  else  sorted_cols = catx(' ', sorted_cols, name);
  call symput('sorted_cols', sorted_cols);
run;
```

This DATA step creates a MACRO variable named SORTED_COLS. It stores a character string which contains every column name in sorted order, each separated by a space. Be sure to set the appropriate length for this variable, such that it is sufficient to store every column name.

```
data output_sorted;
  retain &sorted_cols;
  set output;
run;
```

Finally, use the RETAIN statement as shown before. However, instead of listing each column, we simply call the MACRO variable.

Alternatively, we can also use PROC SQL to accomplish the same task, with fewer lines of code.

```
proc sql;
 select name into :sorted_cols separated by ' '
 from dictionary.columns
 where libname = 'WORK' and memname = 'OUTPUT'
 order by name;
quit;

data output_sorted;
 retain &sorted_cols;
 set output;
run;
```

The SQL statement covers the first three steps by itself. The DATA step with the RETAIN statement is the same as before. Please note that the SQL statement requires the name of the data set (**memname**), as well as the name of the library that it resides (**libname**).

ANOTHER WAY TO SORT

If the column names share the same prefix, followed by consecutive numbers, an easier way to sort them is to use a SAS variable list. In a previous example, the columns are named **day001**, **day002**, **day003**, …, **day365**. They can be represented by the syntax **day001-day365**. Using a SAS variable list in the RETAIN statement will effectively put the columns in sorted order.

```
data fix3;
 retain salesman day001-day365;
 set output;
run;
```

CONCLUSION

The first method allows users to put the columns in any custom order. However, this requires listing each column name individually. Therefore, this should only be done when the SAS data set has a small number of columns.

If the number of columns is large, the only feasible solution is to sort the columns by their names (i.e. the second method). Before doing so, it is recommended that users name their columns in such a way that they are easy to understand when they are sorted alphabetically.

Having the columns in a SAS data set in a sorted order makes data interpretation and analysis much easier. This paper provides different ways to reorder the columns, in both small and large data sets.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name: Tony Yiu
Enterprise: Department of Research and Evaluation, Kaiser Permanente Southern California
Address: 100 S. Los Robles Ave, 2nd Floor
Pasadena, CA 91101
Work Phone: (626) 564-3290
Fax: (626) 564-3430
E-mail: Tony.S.Yiu@kp.org