SESUG Paper 244-2018

Divide and Conquer: A Macro to Split Data Based on Duplicate Values

Meredith Tayshetye and David Eppler, Kobie Marketing, Inc.

ABSTRACT

Have you ever had a dataset that needed to be split into smaller datasets so that one variable did not have duplicate values? An email program the team was using did not allow more than one email to an address within the same send. So, if one email address was listed multiple times for multiple records, the program would only send an email to one record, excluding the others. As a workaround, team members would manually copy duplicates into different files and treat each duplicate file as a separate send. While we could not solve for the limitations of the email program, we could make it easier to split the files.

Using SAS Enterprise Guide 7.13, we have created a macro that solves this problem. The macro will determine the maximum number of times any particular value appears and will create that many files. Though an email address can be linked to multiple records, you may not initially know the maximum number of times an address appears. For example, if a file has one email address linked to ten different records, there will be ten output files. The macro will automatically determine the maximum number of files needed. Then, it will write out the necessary number of files to a specified directory where they can be used for other purposes such as an input to the email program.

This macro is customizable to a variety of data and industries. It is easy to use and can save companies valuable time and manual work.

INTRODUCTION

The email program we needed to use did not allow duplicate email addresses in a list. But, what if an email address was supposed to receive more than one email? An email could be linked to more than one account. Creating multiple send lists solved this issue but the manual process to create multiple lists was time consuming. A few lines of SAS code made a tedious process much easier. This code can be modified to break apart data sets based on other values as well.

THE PROBLEM

Our goal was to create multiple lists from one master list. In the output files, each email address should be present only once per file. We started with a file like the example below. Email 1 is present twice. Email 2 is present only once. Email 3 is present four times. So, we need to split this file into four separate files.

This is an example of the file that needs to be split:

User_Name	Email
User A	email1@email.com
User B	email1@email.com
User C	email2@email.com
User D	email3@email.com
User E	email3@email.com
User F	email3@email.com
User G	email3@email.com

This is what we needed the output files to look like:

File 1

User_Name	Email
User A	email1@email.com
User C	email2@email.com
User D	email3@email.com

File 2

User_Name	Email
User B	email1@email.com
User E	email3@email.com

File 3

User_Name	Email
User F	email3@email.com

File 4

User_Name	Email
User G	email3@email.com

The sum of the record counts from the output files is equal to that of the original data set. Each email appears only once per output data set and can now be used as input to the email program.

THE CODE

This is the macro that does the work:

```
/*Sort by email address.  MYDATASET is the master file with all
records. Email is the variable we want to split on.*/
   PROC SORT data= mydataset; by email; run;

   *Create a counting variable that will determine the number of times a value
   appears.  This data set will be used as input in the next step;

   DATA mydataset;
   SET mydataset;
   by email;
   if first.email then emailnum=0;
   emailnum +1;
   RUN;
/*The macro variable LASTVAR will determine the maximum number of
times the value will appear. For example, if an email address appears
a maximum of five times, this value will be set to five.  This will be
the number of output files you will need and tells the macro how many
loops to run.*/
   PROC SQL;
    select max(emailnum) into :lastvar
      from mydataset;
      QUIT;
```

```
%PUT &lastvar; *Output value to the log for checking purposes;

*SPLIT is the name of the macro doing the work;
%MACRO split;

*Create multiple data sets, each with unique email addresses;
 DATA %do i=1 %to %eval(&lastvar.);
  mydataset_out&i (drop=emailnum)
     %end;;
  SET mydataset;
     %do i=1 %to %eval(&lastvar.);
     if emailnum= %eval(&i) then output mydataset_out&i;
     %end;
 RUN;

*User specifies where to output the data sets and in what format;
     %do i=1 %to %eval(&lastvar.);
       PROC EXPORT data= mydataset_out&i
                outfile="C:\filepath\myfilename_&i..csv"
                 DBMS=CSV REPLACE;
       RUN;
     %end;
%MEND; *End of the macro;
%split; *Execution of the macro;
```

CONCLUSION

Using a macro to automate a tedious task saves the user valuable time and makes the process less prone to error. We created this code to create input for an external email program but it is easily customizable to a variety of uses.

CONTACT INFORMATION

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

Meredith Tayshetye
Kobie Marketing, Inc.
meredith.tayshetye@kobie.com

David Eppler
Kobie Marketing, Inc.
david.eppler@kobie.com