

PROC FORMAT - AN eMERGING ALTERNATIVE

Melissa Garreans

John Alden Life Insurance Company

INTRODUCTION

Combining information from multiple files can be done in several ways using SAS®. In version 5 the MERGE statement was often used, but the files had to be in the same order to merge. Version 6 introduced two new options PROC SQL and PROC FORMAT that eliminate the need to have the files in the same order. In this paper I will illustrate how to use PROC FORMAT to create a format that is used to select records from very large files and "merge" data from one file to another. I have found that in many applications the format approach requires fewer resources than SQL or MERGE.

THE DATA

This example uses as a U.S. Post Office file containing zip code, state, county name and city name (sorted by zip code) and a claim file with more than 30 million observations (sorted by claim number).

MY ASSIGNMENT

I need to know the number of providers receiving payment in North and South Carolina by state, county and city. The claims file contains the provider's zip code, but the state, city and county must be determined from the Post Office file based on zip code. Here is how I approached the problem.

BUILD THE ZIP CODE FORMAT

In order to use PROC FORMAT with the CNTLIN option you must have three variables in your dataset: START, LABEL, and FMTNAME. Instead of creating a new variable called START, I do a RENAME on the KEEP statement and I set the value of FMTNAME in the RETAIN statement.

I like to use HLO (high/low/other) variable. Setting the HLO variable to 'O' creates the OTHER

category in the format. The OTHER category will be used to validate the claims data.

The LENGTH statement is also important since in the IF _N_ = 1 logic I set the value of LABEL to '?????'. Without the LENGTH statement the length of the variable LABEL would be 5.

The first OUTPUT statement is needed to write the record with HLO = 'O', then the HLO variable is reset to spaces. Many times I forget to put the second OUTPUT statement. The second OUTPUT statement is required or you end up with a format with only an OTHER entry.

```

/* BUILD THE ZIP CODE FORMAT                                */
DATA ZIPCODE
  (KEEP=ZIPCODE LABEL FMTNAME HLO
   RENAME = (ZIPCODE = START));
  SET libref.ZIPFILE;
  RETAIN FMTNAME '$ZIPFMT';
  LENGTH LABEL $55;
  /* SET UP THE "OTHER" CATEGORY                            */
  IF _N_ = 1 THEN
    DO;
      HLO = 'O';
      LABEL = '?????';
      OUTPUT;
      HLO = ' ';
    END;
  /* STATE IS CHAR 2, CNTYNAME IS CHAR 25,                  */
  /* CITYNAME CHAR 28                                       */
  LABEL = STATE||CNTYNAME || CITYNAME;
  OUTPUT;
RUN;

```

WATCH OUT FOR DUPLICATES

If you have duplicate values for START, you will not be able to create the format. A PROC SORT with the NODUPKEY option or a data step with FIRST./LAST. processing will eliminate the duplication.

```

PROC SORT NODUPKEY;
  BY START;
RUN;

```

CREATE FORMAT LIBRARY

Now using the CNTLIN option create a temporary format library containing the zip code information.

```
PROC FORMAT CNTLIN = ZIPCODE;
RUN;
```

READ AND "MERGE"

Now it is time to read the claims file. Using the PUT function I access the format ZIPFMT to return into the variable VALUE the concatenated state, county name and the city name. If '?????' is returned I have invalid information on the claim file. I print the information needed to correct these zip codes. I subset the claims file based on the state from the ZIPFMT.

```
DATA MEDLCLM;
  SET libref.MEDLCLM;
  VALUE = PUT(ZIPCODE,$ZIPFMT.);
  /* VALIDATE THE PROVIDER'S ZIPCODE */
  IF VALUE = '?????' THEN
    PUT 'INVALID PROVIDER ZIPCODE: '
      PROVID= ZIPCODE=;
  STATE = SUBSTR(VALUE,1,2);
  /* SELECT ONLY NORTH & SOUTH CAROLINA */
  IF STATE IN ('NC' 'SC');
  CNTYNAME = SUBSTR(VALUE,3,25);
  CITYNAME = SUBSTR(VALUE,28,28);
RUN;
```

Now all that is left is to print the report!

SYSTEM LIMITATIONS

First, there are memory considerations. Since most of my work is done in MVS on an IBM 3090, I can create some very large formats with the systems defaults (by large I mean more than 250,000 observation in my format). When I need even larger formats, I would have one job create a permanent format library (by using PROC FORMAT CNTLIN =dataset name LIBRARY=libref and then use this library in the job that reads the claims file. To override the system defaults, use MEMSIZE=0M on the OPTIONS statement and hardcode the region parameter on the JOB card.

ACKNOWLEDGMENT

SAS® is a registered trademark of SAS Institute, Inc., Cary, North Carolina.