SAS Tutorial: Unconventional Merges

Peter L. Rikard

SAS Tools

This started off being a set of unusual "merge" applications. Then it became "tricks" and now it is a set of tools. (Some source code is at the end of the paper for demonstration) There is no intention that these are "THE" ways to do things in SAS, rather methods that can be useful. All of the methods shown are using SAS79.5, newer releases may dispense with some of these tools.

Merging One Observation with All

If you have data that you wish to put with every observation in a dataset, using obvious methods in SAS you run into several problems.

EX: PROC MEANS DATA=MAIN NOPRINT; VAR X1-X10; OUTPUT OUT=NEW MEAN=MEANI-MEANIO; DATA TOTAL; something MAIN NEW; (so that MEANI-MEANIO are now with every observation)

To MERGE, there is no matching variable in every observation.

DATA TOTAL;
MERGE MAIN NEW,
without a BY MEANS are only on first observation.
with a BY we must create a dummy CONSTANT BY variable.

To SET, variables mentioned in THE SET statement cannot be retained.

DATA TOTAL;
SET NEW MAIN;
RETAIN MEANI-MEAN10;
(doesn't work)

TOOl: Conditionally executed SET statement.

DATA TOTAL;
SET MAIN;
IF _N_ =1 THEN SET NEW;

Variables mentioned in SET statements are set to missing values and bring in new values when they are executed. By only executing the SET statement on the first observation, all variables are "retained" until the SET statement is executed again. In this example that never occurs and all variables from NEW will be in every observation in TOTAL;

Selective Merging of a SAS Dataset

In a merge application where two datasets have sometimes slightly differing BY values a simple merge doesn't work. If with logical statements you can determine that you have the "matching" observation you can perform your own merge.

EX:

Dataset ONE has variables V1-V4
Dataset TWO has variables V1-V4
A normal MERGE would be
BY V1 V2 V3 V4; But in TWO, V4 is sometimes missing, in which case ONE is to be merged with TWO by V1-V3.

TOOl: Selected observation.

DATA FINAL;
SET ONE;
BY V1-V4;
IF FIRST.V4 THEN LINK R1;
(more statements) RETURN;
R1: SET TWO (RENAME=(V1=VT1 V2=VT2 V3=VT3 V4=VT4));
IF VT1=V1 AND VT2=V2 AND VT3=V3 AND VT4=V4
THEN RETURN;
(complete merge)
IF VT1=V1 & VT2=V2 AND VT3=V3 AND VT4=.
THEN RETURN;
(incomplete merge)
GO TO R1;
(no match, try again)

For each pass through the loop SAS picks up a new observation. When you find the matching observation that observation will remain as a MERGE until the block is executed again. Users opposed to the GO TO may substitute a group of LINK.. SET.. RETURN.. statements. The BY variables are renamed so that they will not replace BY variables from DATA1.

Merging with pointers.

When selecting observations from a large STATIC dataset by merging a dataset of ID's with the large dataset and selecting only matches, the POINTER= provides an improved efficiency.

EX: A large SAS dataset contains many observations for each SSN for each individual in a company. A normal requirement is to select off all observations belonging to one person and print a report.
METHOD 1:
DATA SELECT;
   INPUT SSN ;
   CARDS;
   123456789
DATA REPORT;
   MERGE SELECT(IN=IN1)
   BIG.DATASET;
   BY SSN;
   IF IN1;
   (report)

Every observation in the large dataset must be read by SAS. While there are ways to improve this run, instead consider pointers. (SAS Technical Report P-115, page 1.3)

METHOD 2
DATA SELECT (as before)
DATA SELECT2;
   MERGE SELECT(IN=IN1)
   BIG.KEYS;
   BY SSN;
   IF IN1;

KEYS is a previously created dataset. It must be created every time BIG.DATASETS is created. KEYS has one observation per person and three variables: SSN and FIRSTOBS and LASTOBS which contain the beginning and ending observation numbers for each person.

(Creation of KEYS dataset)
DATA BIG.KEYS;
   SET BIG.DATASET;
   BY SSN;
   IF FIRST.SSN THEN FIRSTOBS=_N_;
   IF LAST.SSN;
   LASTOBS=_N_;
   RETAIN FIRSTOBS;
   KEEF SSN FIRSTOBS LASTOBS;

To select records from BIG.DATASET, first the record is selected from KEYS for the individual, second appropriate records can then be selected from BIG.DATASET using the POINTER= option.

DATA REPORT;
   SET SELECT2;
   DO POINTER=FIRSTOBS TO LASTOBS;
   SET BIG.DATASET POINT=POINTER;
   OUTPUT;
   END;

Now only the observations belonging to the individuals that you want will be read from BIG.DATASET. If there were 1000 people and 200,000 observations in the large dataset; Method 1 implies 200,000 reads for a selection. Method 2, implies 1,000 reads of KEYS and 200 reads on BIG.DATASET.

Match Merge without a MERGE

The merge operation is useful in building a new dataset that has only matching observations from two or more datasets.

EX: DATA TOTAL;
   MERGE ONE(IN=IN1) TWO(IN=IN2);
   BY ID;
   IF IN1 AND IN2;

The new dataset only contains matched observations. IDs which appeared only in one dataset will not be output. When the MERGE takes place variables that exist in more than one dataset(other than BY variables) will be lost from all but the last named dataset containing the variable. (see MERGE examples in SET MERGE UPDATE Tutorial). This can be prevented by appropriate renaming, drop, keep, actions but may be unnecessary.

EX: Two datasets, one contains one observation per person with "header" information in variables A,B, and C. The other contains "detail" information in those same variables A,B, and C. We want to use PROC PRINT, to print the data for matched observations.

TOOL: FIRST., LAST., and IN variables

DATA TOTAL;
   SET HEADER (IN=INH)
   DETAIL (IN=IND);
   BY NAME;
   IF LAST.NAME AND INH
   THEN DELETE;
   (only a header record for this person)
   IF FIRST.NAME THEN
   IF INB THEN DELETE='YES';
   (only detail records)
   ELSE DELETE='NO';
   (both types available)
   RETAIN DELETE;
   IF DELETE='YES' THEN DELETE;

No renaming is needed, no data is lost, and the new dataset can be printed using PROC PRINT, rather than writing your own report procedure. Note, that this works only with the first dataset containing a maximum of one observation per person.

Maxim

Given any problem, if you find you have to work very hard to solve it, there is probably an easier way.
POSTERS

General

Session Leader:

James Hosking, University of North Carolina