Merging Historical Data Automatically while Dynamically Sorting the Variables

Tim Qingfeng Wang

Texas Education Agency
1701 N. Congress Avenue, Austin TX

ABSTRACT

When one merges yearly based tables together to form one historically aggregated table, variables are often arranged naturally by order of time (year), such as VarA_07, VarB_07, VarC_07, VarA_08, VarB_08, VarC_08, VarA_09, VarB_09, VarC_09. However, in order to perceive a trend of one particular variable over the time, it's often desired to re-arrange same variables but from different years together, such as VarA_07, VarA_08. VarA_09. There is no one direct SAS procedure to fulfill this kind of task. But SAS system table DICTIONARY.COLUMNS stores variables in alphabetical order, and SAS CONTENTS procedure can also generate an output table with variables ordered. Together with SAS Macro language, this work presents two approaches to merge historical data automatically and at the same time to have the variables sorted dynamically. Using similar approaches, one can also apply formats to certain group of variables without typing the variable names.

INTRODUCTION

While we are working with Special Education data, we need to merge yearly based historical data together to examine how a district behaves over the time in terms of compliance with the federal/state rules. For this purpose, it's ideal to have same named variables (with different 2 digit year suffix) grouped together in ascending order as shown in Fig 1. But in the process of data merging, variables are naturally ordered by time as shown in Fig 2.

Region	District	Ind11_07	Ind11_08	Ind12_07	Ind12_08	Ind13_07	Ind13_08	Status_07	Status_08
01	091175	NC	C	NC	C	NC	C	NA	MR
01	092072	NC	NC	C	NC	NC	NC	NA	NS

Fig. 1 Variables in alphabetical order (Both district number and respective data are faked)

Region	District	Ind11_07	Ind12_07	Ind13_07	Status_07	Ind11_08	Ind12_08	Ind13_08	Status_08
01	091175	NC	NC	NC	NA	C	C	C	MR
01	092072	NC	C	NC	NA	NC	NC	NC	NS

Fig. 2 Variables in order of time (Both district number and respective data are faked)

VAR statement (in PROC PRINT) or RETAIN statement (in a DATA step) is often used to manually re-order variables sequence by typing them one by one, as follows.

```
VAR Region District Ind11_07 Ind11_08 Ind12_07 Ind12_08 Ind13_07 Ind13_08 Status_07 Status_08;
```

ATTRIB, LENGTH, and a few other statements could achieve similar purpose. It's also true that when you want to apply format to just certain variables, you usually type them one by one in the FORMAT statement. However, the work load of manually arranging variables can be huge and error chance increases when the number of variables become hundreds or more. There is no single built-in SAS

procedure to perform this kind of task. Two approaches are reported here to re-order variables dynamically as the data set merging takes place.

APPROACH 1 – Using DICTIONARY.COLUMNS

SAS Dictionary tables store SAS session metadata, which is a description or definition of data or information. The Dictionary.Columns table contains information (such as name, type, length, and format) about all columns in all tables that are known to the current SAS session. When you write a query against the Dinctionary.Columns, the variables of the specific dataset, which is defined in WHERE clause in the SQL procedure, will be output in alphabetical order. The alphabetically ordered variables can then be written into a macro variable, which will be used to re-order the dataset variables permanently, or just to print the report with variables re-ordered but leaving the variable order in the dataset untouched. Prior to the programs of re-ordering the variables, current year data is first merged to the historically aggregated data up to last year. The multi-steps/procedures are included in a macro in order to automate the merging and re-ordering processes.

```
%MACRO APRCH1(YR=08); /*Macro parameter: YR -- current year*/
   /*Define a macro variable LAST_YR*/
   %LET LAST_YR=%EVAL(&YR-1);
   %IF &LAST_YR LT 10 %THEN %LET LAST_YR=0&LAST_YR; %ELSE;
   /*Merge current year data with the historically aggregated data up to last year*/
   DATA DESK.UPTO_&YR;
      MERGE DESK.UPTO_&LAST_YR DESK.DT_&YR;
      BY REGION DISTRICT;
   RUN;

   /*Query against SAS Dictionary.Columns to define a macro variable VAR to hold the
   dataset variables in alphabetical order*/
   PROC SQL NOPRINT;
      SELECT DISTINCT NAME INTO :VAR SEPARATED BY ' '
      FROM DICTIONARY.COLUMNS
      WHERE LIBNAME = 'DESK' AND MEMNAME = "UPTO_&YR";
   QUIT;

   /*Re-order the dataset variables by using the newly defined macro variable VAR*/
   DATA FINAL_&YR;
      RETAIN &VAR;
      SET DESK.UPTO_&YR;
   RUN;
   /*Have REGION, DISTRICT as the 1st two variables, rest still in alphabetic order*/
   DATA DESK.FINAL_&YR;
      RETAIN REGION DISTRICT;
      SET FINAL_&YR;
   RUN;
%MEND;
```

Save the above macro to your folder, e.g. DESKTOP here. When data of next year (e.g. 2011) arrives, simply run following statements and a macro call.

```
LIBNAME DESK "DESKTOP";
%INCLUDE 'DESKTOP\APRCH1.SAS' /SOURCE2;
%APRCH1(YR=10)
```

APPROACH 2 – Using PROC CONTENTS

PROC CONTENTS is often used to show the descriptor information for an individual data set, including a list of variables in alphabetic order, which can be output as a SAS data set. The alphabetically ordered

variables will be read as values of a list of macro variables, which is further used to change the order of variables in the data set using RETAIN statement in a DATA step.

```
%MACRO APRCH2(YR=08); /*Macro parameter: YR -- current year*/
   /*Define a macro variable LAST_YR*/
   %LET LAST_YR=%EVAL(&YR-1);
   %IF &LAST_YR LT 10 %THEN %LET LAST_YR=0&LAST_YR; %ELSE;
   /*Merge current year data with the historically aggregated data upto last year*/
   DATA DESK.UPTO_&YR;
      MERGE DESK.UPTO_&LAST_YR DESK.DT_&YR;
      BY REGION DISTRICT;
   RUN;

   /*PROC CONTENTS list variables alphabetically unless otherwise specified,like
   ORDER=VARNUM*/
   PROC CONTENTS DATA=DESK.UPTO_&YR OUT=TMPT_&YR (KEEP=NAME) NOPRINT; RUN;

   /*Create a list of macro variables to hold these variable names in alphabetic
   order*/
   DATA _NULL_;
      SET TMPT_&YR;
      CALL SYMPUT(COMPRESS('VAR'||_N_), TRIM(NAME)); /*SYMPUT routine*/
      CALL SYMPUT('TOTAL', TRIM(LEFT(PUT(_N_,8.))));
   RUN;
   %PUT &VAR1, &VAR2, &VAR3, &VAR4, &VAR5, &TOTAL;

   /*Re-order the variable order in the dataset*/
   DATA FINAL_&YR;
      RETAIN %DO J=1 %TO &TOTAL; &&VAR&J %END;;
      SET DESK.UPTO_&YR;
   RUN;
   /*Have REGION, DISTRICT as the 1st two variables, rest still in alphabetic order*/
   DATA DESK.FINAL_&YR;
      RETAIN REGION DISTRICT;
      SET FINAL_&YR;
   RUN;
%MEND;
```

Similarly, when new data arrives, call the macro.

```
LIBNAME DESK "DESKTOP";
%INCLUDE 'DESKTOP\APRCH2.SAS' /SOURCE2;
%APRCH2(YR=10)
```

APPLY FORMAT TO SELECTED GROUP OF VARIABLES

In stead of listing variables one by one in the FORMAT statement, you can also create a macro variable to host your interested variables and use it in the FORMAT statement.

```
PROC SQL NOPRINT;
   SELECT DISTINCT NAME INTO :VAR SEPARATED BY ' '
   FROM DICTIONARY.COLUMNS
   WHERE LIBNAME='DESK' AND NAME =* 'Status%';
QUIT;
PROC PRINT DATA = DESK.FINAL_10 (OBS=45) NOOBS WIDTH=MIN;
FORMAT &VAR $FMT.;
RUN;
```


ACKNOWLEDGEMENTS

I would like to thank Keith Cranford for his review and encouragement of this short paper.