SAS* MACROS FOR DESIGNING HIERARCHICAL DATA FILES

Robert Landon, Southern California Edison Company

ABSTRACT

SAS® has the capability of efficiently handling a wide variety of complex system data files. When creating files for input or output in SAS, the programmer can employ macro coding techniques to design hierarchical file structures. These are especially useful when data is to be stored for subsequent use in a SAS library. When this method is used to input data from tape files stored in COBOL library formats (e.g., COBOL copylibs), it greatly enhances the ease of determining data positions, counting table occurrences, handling related data in macro "blocks", and preserving data integrity between files.

To effectively create SAS* macros for data operations, the programmer groups lower level data for input, selects a meaningful macro name for the group, and includes documentation of data elements within each macro group by using comments. A macro or a combination of macros is used for selecting data for program operations. Data is then grouped in hierarchical SAS output files and stored in the SAS library.

Using macros to enhance data input and manipulation provides a convenient solution to tasks involving voluminous or complex data requirements. Macros increase computer run time and space requirements. However, this method may improve the quality of data and the ease of data manipulation both within the SAS program and the SAS library.

INTRODUCTION

Southern California Edison utilizes a number of SAS® procedures for a wide variety of its data processing needs. Source input data to SAS programs is often in the form of tape files stored in COBOL libraries. These libraries are well documented and provide an excellent opportunity to employ SAS macro coding techniques for conversion to SAS dataset libraries (Figure 1).

DATA GROUPING

The program extracts in Figures 2 and 3 below show input data which is grouped exactly as it was grouped in the COBOL library. Among the advantages of doing this are:

1. Data may be documented by the macro, as is shown in the comment blocks accompanying each data element in the macro groups. The SAS® dataset may later be referenced for documentation questions regarding specific data elements.

2. Data detail is preserved between COBOL files and SAS® files. The SAS file includes the same level of detail appearing in the COBOL file. In some cases, the programmer would otherwise miss an important data element and have to remount the tape to retrieve the data, but this procedure allows the entire block to be easily input into the SAS library.

3. Data appearing in tables, such as the data group identified by the macro "MTR_HIST", may be entered in blocks. Since there are times the programmer does not know in advance how many times data values are repeated in a table, the macro also uses control variables, such as the variable SCI_M_H, to determine table data content. In this case, the file record format is fixed block and SCI_M_H controls the number of times a non-zero table entry occurs in the file.

4. Data within the SAS® program is often much easier to select and manipulate when it is grouped using SAS macro statements. For example, data defined in the macro "MTR_HIST" is all related to the customer's meter data while data defined in the macro "DEM_BIL" is all related to the customer's demand data. When data selection or manipulation occurs, the programmer simply addresses the data conveniently by the appropriate macro name MTR_HIST, or DEM_BIL.

DATA INPUT

Once data groups are defined and documented, data input is accomplished by specifying an input statement followed by the macro name. For example, the macro blocks are input (e.g., mtr_hst, dem_bil, etc.), until the desired data is entered. The control blocks are used to input the data organized in tables. If a master file of the data is desired, macro "OUT" contains all other macros defined in the program. If logically related subsets of data are desired, macro "OUT" includes only macros containing the data needed.

DATA SELECTION AND MANIPULATION

As suggested earlier, certain data may be eliminated simply by locating the appropriate macro group, selecting the data, and eliminating other data. An example of this in the program extract below is seen in macro "OUT" where only one rate class was selected, rate "TOU-ALMP-1" among several rate classes available on the tape file (Figure 4). An
example of data manipulation and hierarchical file creation appears in the data statement YY.ONE (representing month one), where the program exits the data step after one pass of the macros involving data tables (Figure 5). In data statement YY.TWO (month two), the program exits the data step after the second pass of all macros containing tables, replacing those values on the first pass with those on the second pass. This method avoids the problem of dealing with arrays in creating SAS* data libraries. The program then concatenates the monthly data into a file called "YY.YEAR", and saves the previous twelve months data to the SAS library. Then the program sets "YY.YEAR" in work data set "ONE" and performs calculations. Using this technique, it is possible to track monthly customer information such as current name and address, as well as monthly electrical usage information, such as that appearing in data tables read by looping macros. This method illustrates the creation of multilevel SAS files, which may be saved in a SAS library for use in other programs. A master file can be saved in the SAS Library of all data documented in the input file simply by including all macro names in macro "OUT". If data is carefully grouped within macros when input, entire blocks of data may be easily handled. Later, a program using data from that SAS Library may not require all variables previously defined within the macro "block" and extraneous variables may simply be dropped from the dataset.

CONCLUSION

SAS* provides an excellent mechanism to input, manipulate, and output hierarchical multilevel COBOL files to a SAS Library. The use of macro statements to accomplish this task has several advantages and greatly reduces the complexity of working with multilevel COBOL files and data tables. For further information, the author can be contacted by writing to the following address:

Robert Landon
Southern California Edison Company
Revenue Requirements Department, Room 321
2244 Walnut Grove Avenue
Rosemead, California 91770
(818) 302-6866

* SAS is a registered trademark of SAS Institute Inc., Cary, NC, USA.
471 MACRO OUT
472 INPUT D1077Z RATE &12.91
473 IF RATE = 'TOL-ALMP-1';
474 INPUT ACNT_SEG;
475 INPUT PRF_SEG;
476 INPUT MTR_SEG;
477 INPUT CUST_SEG;
478 INPUT CONTROL;
479 INPUT BILL_INF;
480 IF I = Z THEN GOTO EXIT1;
481 END;
482 EXIT1:
483 INPUT DEMAND;
484 IF I = Z THEN GOTO EXIT2;
485 END;
486 EXIT2:
487 INPUT TOU_HIST;
488 IF I = Z THEN GOTO EXIT3;
489 END;
490 EXIT3:
491 INPUT PHR_BILL;
492 IF I = Z THEN GOTO EXIT4;
493 END;
494 EXIT4:
495 INPUT SUP_BILL;
496 INPUT CHARGES;
497 INPUT REAC_HTR;
498 INPUT BILLINF;
499 IF I = Z THEN GOTO EXIT5;
500 END;
501 EXIT5:
502 DATA YY.ONE;
503 INPUT XX MISSOVER;
504 Z = 11;
505 OUT;

NOTE: INFILE XX IS:
DNAME=PROD.TAPE.CIS.CIRM40M1.00010V900,
UNIT=TAPE,VOL=SER=020158,DISP=OLD,
DCB=(BLKSIZE=24096,RECL=12048,RECFM=FB)

NOTE: 5815 LINES READ FROM INFILE XX.
NOTE: DATA SET YY.ONE HAS 251 OBSERVATIONS AND 410 VARIABLES.
NOTE: THE DATA STATEMENT USED 5.92 SECONDS AND 444K.

1240 DATA YY.TWO;
1241 Z = 2;
1242 OUT;

NOTE: INFILE XX IS:
DNAME=PROD.TAPE.CIS.CIRM40M1.00010V900,
UNIT=TAPE,VOL=SER=020158,DISP=OLD,
DCB=(BLKSIZE=24096,RECL=12048,RECFM=FB)

NOTE: 5815 LINES READ FROM INFILE XX.
NOTE: DATA SET YY.TWO HAS 251 OBSERVATIONS AND 410 VARIABLES.
NOTE: THE DATA STATEMENT USED 5.92 SECONDS AND 444K.

1989 DATA YY.THREE;
1990 INFILE XX MISSOVER;
1991 Z = 3;
1992 OUT;

NOTE: INFILE XX IS:
DNAME=PROD.TAPE.CIS.CIRM40M1.00010V900,
UNIT=TAPE,VOL=SER=020158,DISP=OLD,
DCB=(BLKSIZE=24096,RECL=12048,RECFM=FB)

NOTE: 5815 LINES READ FROM INFILE XX.
NOTE: DATA SET YY.THREE HAS 251 OBSERVATIONS AND 410 VARIABLES.
NOTE: THE DATA STATEMENT USED 5.92 SECONDS AND 444K.

2730 DATA YY.FOUR;
2731 INFILE XX MISSOVER;
2732 Z = 4;
2733 OUT;

NOTE: INFILE XX IS:
DNAME=PROD.TAPE.CIS.CIRM40M1.00010V900,
UNIT=TAPE,VOL=SER=020158,DISP=OLD,
DCB=(BLKSIZE=24096,RECL=12048,RECFM=FB)

NOTE: 5815 LINES READ FROM INFILE XX.
NOTE: DATA SET YY.FOUR HAS 251 OBSERVATIONS AND 410 VARIABLES.
NOTE: THE DATA STATEMENT USED 5.92 SECONDS AND 444K.

3471 DATA YY.FIVE;
3472 INFILE XX MISSOVER;
3473 Z = 5;
3474 OUT;

DATA YY.YEAR;
SET YY.ONE YY.TWO YY.TREE YY.FOUR YY.FIVE YY.SIX YY.SEVEN
YY.EIGHT YY.NINE YY.TEN YY.ELEVEN YY.TWELVE YY.THIRTEEN;

FIGURE 4.