Introduction

This paper presents the techniques used to create a computer system which generates and executes SAS programs from one-word TSO/CMS commands. The combination of one-word invocation and SAS program generation and execution facilitates the extraction of information from SAS databases for both novice and expert computer users. These techniques are described and illustrated below, using a sample application.

The SAS dataset used in this example consists of one year of monthly sales history for various items sold in a store. The variables are: item number - NUMBER, item description - DESCRIP, and 12 monthly sales amounts. There is one observation for each item. The variable for the first month, MONTH1, represents sales for January, 1981. The variable for the twelfth month, MONTH12, is for December, 1981.

Development of Techniques

The first step in building the system is to determine what outputs are wanted. For this example, the system should be able to answer three basic types of questions: 1) Which items make up the top 10% of total sales for 1981? 2) What were the top 3 selling items in March of 1981? 3) Which items sold over $1,000.00 in the first quarter of 1981? As an enhancement, it could also be possible to extract items from the bottom strata of the item list, as well as letting the cutoff numbers take on variable values. The enhancement would allow questions such as: 1) Which items make up the bottom 25% of total sales for 1981? 2) What was the worst selling item last month? 3) Which items sold less than $100.00 in December, 1981?

To answer the above questions, in an interactive TSO/CMS environment where the information is stored in a SAS database, the choice of implementing the system in Command Language and SAS is reasonable. By having the Command language program interact with the user to determine his question and then generate and start the execution of a SAS program to answer the question, both programming languages are used effectively. The CLIST (Command Language program) communicates with SAS by writing SAS macro bodies into a macro library. The macro library is concatenated to the front of the SAS program, which consists largely of macro names, to produce the answer to the user's question. The macro bodies tailor the SAS program to answer any of the possible questions that have been allowed for by the programmer.

To produce the single SAS program, separate programs should be written to answer each of the three basic question types (see Listing 1). They are written to be as close to each other in general form as possible. Then the single program is written by comparing the separates and writing macro names where their code is different, and writing the code itself where the separate programs are alike (see Listing 2). This combining process yields a single SAS program capable of doing everything the separate programs could do, if it is provided with the correct macro bodies.

Next comes the preparation of the CLIST. Its role in the system is to interact with the user, generate SAS macro bodies derived from the interaction, and cause the execution of the SAS job. To keep the example simple, the SAS program is executed in batch mode (background processing).

The placement of the CLIST is what allows it to be invoked by a single-word TSO command. In most systems, there is a partitioned dataset which has been allocated with the file name of SYSPROC. By creating the CLIST as a member of that dataset, it may be invoked by giving its member-name as a TSO command.

When interacting with the user, the CLIST should first determine which type of question is to be answered and then gather the specific details of the user's query. This information should then be written as SAS macros into the macro library (see Listing 3).

All that remains is for the CLIST to have the SAS program executed. By submitting a SAS job where the DD name of SYSIN has been assigned to the concatenation of the macro library and the SAS program (see Listing 4), the user's
questions will be answered via a SAS-generated output report. The report will show the items which were selected by the specified criteria.

Enhancement of System

The basic system presented here can be improved in many ways. By letting the user specify more extensive logic with control over more variables, a much wider range of questions can be answered. By executing the SAS program in interactive mode, the results can be produced faster. Interactive execution (foreground processing) also raises the question of what to do with the output report. By having the print file go to a specific dataset, the results can be saved, viewed at the terminal, or printed out for hard copy.

If SPF (System Productivity Facility) is installed, the system can be tremendously improved. Panels can be used to interact with the user, and file tailoring can produce the macro library. The output report can then be "browsed" and printed.

Conclusion

By using the described techniques, a flexible database query tool has been developed. Such a system is useful when people who do not program in SAS would like to get information from a SAS database on a regular basis. Instead of having a programmer write or modify a specific program to answer each query, the user can get the information for himself. These techniques have been applied to a real database with much success.

Figure 1

SAMPLE INTERACTION (User Response is Underlined)

READY
QUERY
QUERY SYSTEM FOR SAS DATABASE
THERE ARE 12 MONTHS OF DATA
ENTER RANGE OF MONTHS FOR THIS QUERY BY NUMBER
EXAMPLE: 1,3 FOR 1ST QUARTER
ENTER MONTH RANGE: 1,12
1. SELECT ITEMS AS PERCENTAGE OF ALL ITEMS, FROM TOP OR BOTTOM.
2. SELECT SPECIFIC NUMBER OF ITEMS FROM TOP OR BOTTOM.
3. SELECT ITEMS THAT ARE ABOVE OR BELOW CUT-OFF VALUE.
ENTER 1, 2, OR 3 FOR TYPE OF QUESTION DESIRED: 1
1. TOP OR 2. BOTTOM
ENTER 1 OR 2 FOR STRATA DESIRED (TOP OR BOTTOM): 1
FOR QUESTION NUMBER: 1 ENTER THE PERCENTAGE OF SALES: 10
ENTER TITLE (MAXIMUM OF 26 WORDS)
ITEMS MAKING UP TOP 10 PERCENT OF 1981 SALES
HERE IS THE JOB TO ANSWER YOUR QUESTION
TOC1444A JOB(200) SUBMITTED
READY

WHAT SAS PRODUCES IN RESPONSE TO THE INTERACTION

ITEM MAKING UP TOP 10 PERCENT OF 1981 SALES

<table>
<thead>
<tr>
<th>OBS</th>
<th>NUMBER</th>
<th>DESCRIPI</th>
<th>SALES</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RZ43</td>
<td>WIDGET</td>
<td>$43,476.22</td>
</tr>
<tr>
<td>2</td>
<td>BS20</td>
<td>FLANGE</td>
<td>$31,982.73</td>
</tr>
<tr>
<td>3</td>
<td>AT66</td>
<td>GEAR</td>
<td>$26,776.20</td>
</tr>
<tr>
<td>4</td>
<td>BR25</td>
<td>TRANSISTOR</td>
<td>$19,114.86</td>
</tr>
<tr>
<td>5</td>
<td>AJ18</td>
<td>DISK DRIVE</td>
<td>$15,006.56</td>
</tr>
</tbody>
</table>
LISTING 1  ORIGINAL SEPARATE PROGRAMS TO ANSWER THREE QUESTION TYPES

/* QUESTION 1 */
/* WHICH ITEMS MAKE UP THE TOP 10% OF TOTAL SALES IN 1981 */
DATA ONE (KEEP = NUMBER DESCRIPT SALES);
SET INFO. ITEMS END=EOF; /* SOURCE DATASET */
SALES = SUM OF (OF MONTHI-MONTHI2); TOTAL + SALES;
OUTPUT ONE; /* 1 OBSERVATION PER ITEM WITH 1 SALES FIGURE */
IF EOF THEN OUTPUT SUM; /* 1 OBSERVATION, 1 VARIABLE - TOTAL SALES */
PROC SORT DATA = ONE; BY DESCENDING SALES;
DATA TWO; /* HOLDS ITEMS WHICH PASS SELECTION CRITERIA */
SET ONE; IF N = 1 THEN SET SUM; /*GET TOTAL SALES*/
TEST + SALES; IF TEST > (TOTAL/100+10) THEN DELETE;
RETAIN TOTAL; DROP TEST TOTAL;
PROC PRINT DATA = TWO;
TITLE ITEMS MAKING UP TOP 10 PERCENT OF 1981 SALES;

/* QUESTION 2 */
/* WHAT WERE THE TOP 3 SELLING ITEMS IN MARCH 1981 */
DATA ONE(KEEP=NUMBER DESCRIPT SALES);
SALES = MONTH3;
PROC SORT DATA=ONE;
DATA TWO; /* REMOVE DESCENDING TO GET BOTTOM 3 */
SET ONE; IF N > 3 THEN DELETE;
PROC PRINT DATA=TWO;
TITLE TOP 3 SELLING ITEMS OF MARCH 1981;

/* QUESTION 3 */
/* WHAT ITEMS SOLD OVER $1000 IN 1ST QTR 1981 */
DATA ONE(KEEP=NUMBER DESCRIPT SALES);
SALES = SUM(OF MONTH1 - MONTH3);
PROC SORT DATA = ONE; BY DESCENDING SALES;
DATA TWO;
SET ONE;
IF SALES < 1000 THEN DELETE;
PROC PRINT DATA = TWO;
TITLE ITEMS WHICH SOLD OVER $1,000 IN 1ST QTR 1981;
LISTING 2  THIS IS THE COMBINED SAS PROGRAM

Line #

1 /* THIS PROGRAM IS IN DATASET 'SASLIB(QUERY)'
2   */
3 /* MACRO NAMES ARE PRECEDED BY AN UNDERSCORE */
4 DATA ONE (KEEP = NUMBER DESCRIP SALES)
5 _SUM_SET;
6 SET INFO.ITEMS;
7 SALES = TIME;
8 _SUMARIZ;
9 PROC SORT DATA=ONE;
10 BY UP DOWN SALES;
11 DATA-TWO; SET ONE;
12 SELECT;
13 PROC PRINT DATA = TWO;
14 TITLE _TITLE;

Line #  Explanation
1 Create dataset ONE. It will have one observation per item with the variables
   NUMBER (item number), DESCRIP (item description), and SALES. SALES will hold
   the sum of the specified monthly values. This is common code that appeared in
   all three separate programs.

2 _SUM_SET is a macro name. When this program is used to answer Question 1, its
   macro body consists of: SUM(KEEP~TOTAL). This allows the creation of a
   second dataset with 1 observation and 1 variable which holds the total sales
   value for all items in the specified time period. For Questions 2 and 3, the
   macro body consists only of blanks because the SUM dataset is not needed.

3 INFO.ITEMS is the source dataset with 12 separate monthly sales values.
   Again, this is common code, so it is written as a normal SAS statement.

4 The macro _TIME allows the SALES variable to equal the months specified by the
   user.

5 The macro _SUMARIZ is like the macro _SUM_SET in Line 2, for Questions 2 and 3
   its body will be blanks. Only for Question 1, which is creating 2 datasets in
   this data step, does SAS need to be told when to output to each of the data-
   sets.

6 The items are to be sorted by SALES value for all three programs, so the com-
   mon code is written as normal SAS.

7 Macro_UP_DOWN controls which strata of the item list items are selected from.
   If the macro body consists of the keyword DESCENDING, items will be selected
   from the top strata. If the body consists of blanks, items will be selected
   from the bottom strata.

8 Dataset TWO will hold the items which pass the selection criteria set by the
   user's question. Once more, this is a common code.

9 The body of macro_SELECT determine which items will answer the user's
   question. This section of code is different in all three of the separate
   programs.

10 PROC PRINT will list our results. Again, common code.

11 The title needs to be different for each question, so the text of the title is
   contained in the body of macro _TITLE.
LISTING 3  THIS IS THE CLIST THAT DRIVES THE SYSTEM

/* THIS IS IN DATASET 'SYS1.PROCLIB(QUERY)' WITH FILENAME 'SYSPROC' */
WRITE QUERY SYSTEM FOR SAS DATABASE
WRITE THERE ARE 12 MONTHS OF DATA
WRITE EXAMPLE: 1,3 FOR 1ST QUARTER
WRITENR ENTER MONTH RANGE:
READ M1,M2
SET TIME=&STR(SUM(MONTH&M1 - MONTH&M2))
WRITE WHICH TYPE OF QUESTION?
WRITE 1. SELECT ITEMS AS PERCENTAGE OF ALL ITEMS, FROM TOP OR BOTTOM. This section of code asks the user for the range of months to sum together and then builds the body of the _TIME macro.
WRITE 2. SELECT SPECIFIC NUMBER OF ITEMS FROM TOP OR BOTTOM. Find out which type of question is being asked and put the number in variable QUEST.
WRITE 3. SELECT ITEMS THAT ARE ABOVE OR BELOW CUT OFF VALUE. This section determines the numeric value for the selected question.
WRITENR ENTER 1,2, OR 3 FOR TYPE OF QUESTION DESIRED:
READ QUEST
WRITE 1. TOP OR 2. BOTTOM
WRITENR ENTER 1 OR 2 FOR STRATA DESIRED (TOP OR BOTTOM):
READ STRATA
~ Determines whether to take items from the top of the list or the bottom.
IF &QUEST=1 THEN SET NOUN=PERCENTAGE OF SALES
IF &QUEST=2 THEN SET NOUN=NUMBER OF ITEMS
IF &QUEST=3 THEN SET NOUN=CUT OFF VALUE
WRITE FOR QUESTION NUMBER:&QUEST ENTER THE &NOUN:
WRITENR ENTER THE VALUE
READ NUMBER
WRITE ENTER TITLE (MAXIMUM OF 26 WORDS)
/* ONE VARIABLE FOR EACH WORD OF TITLE */
READ A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Asks the user for the title. Each word is put into a separate variable.
IF &QUEST=1 THEN DO
SET SUMSET = &STR(SUM(KEEP-TOTAL))
SET SUMRIZ = &STR(TOTAL+SALES; OUTPUT ONE; IF EOF THEN OUTPUT SUM;)
END
ELSE 00
SET SUMSET=
SET SUMRIZ­
END
For question types 2 and 3, the macro bodies should be empty.
ALLOCFILE(MACL1B),DA('MACLIB(QUERY)'),OLD
OPENFILE MACLIB OUTPUT
This gives the dataset 'MACLIB(QUERY)' the filename of MACLIB and opens the file for output. The macro bodies will be written into this file.
SET MACLIB=&STR(MACLIB _SUM_SET &SUMSET %)
PUTFILE MACLIB
SET MACLIB=&STR(MACLIB _SUMARIZ &SUMRIZ %)
PUTFILE MACLIB
SET MACLIB=&STR(MACLIB _TIME &TIME %)
PUTFILE MACLIB
PUTFILE MACLIB
This writes the macro for the title.
IF &QUEST=1 THEN DO
SET MACLIB=&STR(MACLIB _SELECT IF N =1 THEN SET SUM; TEST+SALES;)
PUTFILE MACLIB
SET MACLIB=&STR(T=TOTAL/100* &NUMBER ; IF TEST > T THEN DELETE;)
PUTFILE MACLIB
SET MACLIB=&STR(RETAIN TOTAL; DROP TEST T TOTAL; %)
END
This writes the _SELECT macro for question type 1. It involves 3 lines of code but works just the same.

IF &QUEST=2 THEN +
SET MACLIB=&STR(MACLIB _SELECT IF N > &NUMBER THEN DELETE; %)
IF &QUEST=3 THEN DO
IF &STRATA=1 THEN SET COMPARE=&STR(<)
ELSE SET COMPARE=&STR(>)
SET MACLIB=&STR(MACLIB _SELECT IF SALES &COMPARE &NUMBER THEN DELETE; %)
END
This writes the SELECT macro is Questions 2 or 3 have been chosen
LISTING 3 (Continued)

PUTFILE MACLIB  | To reduce the number of program statements in the CLIST, the final (or only) PUTFILE for the _SELECT macro is done here.

IF &STRATA=1 THEN SET MACLIB=&STR(MACRO _UP_DOWN DESCENDING %)
ELSE SET MACLIB=&STR(MACRO _UP_DOWN %)

This writes the UP_DOWN macro. If the top strata has been chosen, the SAS keyword DESCENDING is used as the body, for the bottom strata, it is left empty.

PUTFILE MACLIB

CLOSFILE MACLIB FREE,FILE(MACLIB)

This closes the MACLIB output file and frees the file for use by the SAS job later.

WRITE HERE IS THE JOB TO ANSWER YOUR QUESTION SUBMIT 'JCLLIB(QUERY)'

This starts the SAS job.

LISTING 4 JCL TO EXECUTE SAS JOB

// JOB CARD

/* THIS IS IN DATASET 'JCLLIB(QUERY)' /*
//STEP1 EXEC PROC=SAS /*
//INFO DD DSN=SAS.DATA,DISP=SHR /*
//SYSIN DD DSN=MACLIB(QUERY),DISP=SHR /*
This concatenates the macros written by the CLIST to the front of the single, combined SAS program.

Phillip's output for the MACLIB macro. User has specified question type 1.

MACRO _SUM_SET SUM(KEEP=TOTAL)%
MACRO _SUMARIZ TOTAL + SALES; OUTPUT ONE; IF EOF THEN OUTPUT SUM;%
MACRO _TIME SUM(OF MONTH1-MONTH12) %
MACRO _TITLE ITEMS MAKING UP TOP 10 PERCENT OF 1981 SALES%
MACRO _SELECT IF N =1 THEN SET SUM; TEST SALES;
T=TOTAL/100* &NUMBER ; IF TEST > T THEN DELETE;
RETAIN TOTAL; DROP TEST T TOTAL;%
MACRO _UP_DOWN DESCENDING %

This concatenates the macros written by the CLIST to the front of the single, combined SAS program.

Contents of 'MACLIB(QUERY)'. User has specified question type 1.

MACRO _SUM_SET SUM(KEEP=TOTAL)%
MACRO _SUMARIZ TOTAL + SALES; OUTPUT ONE; IF EOF THEN OUTPUT SUM;%
MACRO _TIME SUM(OF MONTH1-MONTH12) %
MACRO _TITLE ITEMS MAKING UP TOP 10 PERCENT OF 1981 SALES%
MACRO _SELECT IF N =1 THEN SET SUM; TEST SALES;
T=TOTAL/100* &NUMBER ; IF TEST > T THEN DELETE;
RETAIN TOTAL; DROP TEST T TOTAL;%
MACRO _UP_DOWN DESCENDING %

Contents of 'SASLIB(QUERY)'.

/* THIS PROGRAM IS IN DATASET 'SASLIB(QUERY)'
/* MACRO NAMES ARE PRECEDED BY AN UNDERSCORE */
DATA ONE (KEEP = NUMBER DESCRIP SALES)
SUM SET;
SET INFO.ITEMS;
SALES = TIME;
_SUMARIZ;
PROC SORT DATA=ONE;
BY UP DOWN SALES;
DATA=TWO; SET ONE;
_SELECT;
PROC PRINT DATA = TWO;
TITLE _TITLE;