Selective Parameter Passing
A. Cecilia Mauldin

ABSTRACT
Many of the tables, listings and figures (TLFs) that are needed in a study are similar and not totally unique. There are several ways to try to limit the code that will be needed to generate multiple TLFs with one main program.

One way is to %include a generic program into each specialized program and setting parameters to be used by the generic unique file. A second way is to create one program with a macro(s) and call the macro passing parameters.

The first approach has the problem that there are many files needed, one for each TLF plus the shared program that does the work, and all the programs need to be archived and maintained. The second approach has the problem that all the TLFs are generated when only one TLF is needed and all the new TLFs need to be validated.

I am proposing a new variation on the second approach (SPP – Selective Parameter Passing) where the program can be run in batch and we only pass the parameters for the tables that are needed. SPP could be flexible with an additional parameter which would run all the possible parameters and produce all the possible tables.

INTRODUCTION
One program can produce several tables, listings or figures (TLFs) that share a similar base and each different use a different population, analysis variable or condition.

This paper will show a macro called SPP that allow a program to run all or only one TLF that is produced by the generic program.

THE DATA AND FORMATS

DATA AE;
  LENGTH PT $10
  AE $20
; INPUT PT $1-10 TRT AE $13-29-grade OUTCOME A_RED A_DEL A_INT A_DISC RELAT SAE;
CARDS;
Nina 1 Hunger 4 1 1 1 1 1 1 1 1 1
Jadzia 1 Lack of sleep 2 2 0 1 1 1 1 1 1 1
Piojo 1 Hunger 4 3 1 0 1 1 1 1 1 1
Bruja 1 Bil Jack Adiction 3 1 0 3 0 1 1 1 1 1
Azrael 1 Hunger 2 2 1 1 1 1 1
Jadzia 1 Lack of sleep 1 3 0 0 1 1 1 1 1 1
Nina 1 Hunger 4 1 1 1 1 1 1 1 1 1
Nina 1 Hunger 3 2 0 1 0 1 1 1 1 1
Camille 1 Rash 2 1 2 3 4 1 1 1 1 1
Garuña 1 Lack of Hunger 1 1 1 1 1 1 1
RUN;

Formats that are associated with the parameters, if there are rows or titles that depend on the parameter, these values can be assigned using formats.
**COUNT OF TOTAL NUMBER OF PATIENTS IN EACH TREATMENT;**

PROC SQL;
CREATE TABLE TOTALS AS SELECT
  TRT,
  COUNT(DISTINCT PT) AS TOTAL
FROM AE
GROUP BY TRT
;
QUIT;

**CREATE MACRO VARIABLES THAT GO IN THE HEADER OF EACH COLUMN;**

DATA _NULL_;
  SET TOTALS;
  CALL SYMPUT('N'||PUT(TRT, 1.), COMPRESS(PUT(TOTAL, 3.)));
  CALL SYMPUT('_'||PUT(TRT, 1.), put(TRT, TRT.));
RUN;

The macro, REPORT, is has the code that the generic code would have. The macro will be compiled only one for all the possible TLFs cases.
%MACRO REPORT(param=, where=);

Assign the title for each TLF depending on the value of the parameter;

DATA _NULL_;
  TITLE="&PARAM";
  CALL SYMPUT("TITLE", INPUT(TITLE,$TITLE.));
RUN;

TITLE "&TITLE";

Get the count of unique patients by Adverse Event and Treatment; Adverse events are filtered with the macro variable &where;

PROC SQL;
  CREATE TABLE KEEP AS SELECT
      TRT, AE LENGTH = 100, COUNT(DISTINCT PT) AS HMANY
  FROM AE
  &WHERE
  GROUP BY TRT, AE
  ORDER BY TRT
;

Get the count of unique patients with at least one adverse event filtering with the same macro variable &where.

  CREATE TABLE FLINE AS SELECT
      TRT, COUNT(DISTINCT PT) AS HMANY
  FROM AE
  &WHERE
  GROUP BY TRT
  ORDER BY TRT
;
QUIT;

Both data sets together, the variable SORTO is to print the report in the appropriate order.

DATA ALL(where=(trt gt .));
  SET KEEP(IN=A)
    FLINE(IN=B)
    ;
  IF A THEN SORTO=2;
  ELSE IF B THEN DO;
    SORTO=1;
    AE='Patients with at Least 1 '||INPUT("&PARAM",$FLINE.);
  END;
RUN;

PROC SORT DATA=ALL;
  BY TRT;
RUN;
** CALCULATING AND FORMATING PERCENTAGES;**

DATA PRETTY DROP=HMANY PCT TOTAL;
LENGTH CHAR $25;
MERGE ALL
   TOTALS;
BY TRT;
   IF HMANY GT 0 THEN DO;
      PCT = 100 * HMANY/TOTAL;
      IF PCT GE 1 THEN
         CHAR = TRIM(LEFT(PUT(HMANY,BEST.) || ' (' || PUT(ROUND(PCT,1.0),3.) || '%')
      ELSE IF PCT LT 1 THEN
         CHAR = trim(left(PUT(HMANY,BEST.) || ' ( < 1%)');
   END;
RUN;
PROC SORT DATA=PRETTY;
   BY SORTO AE TRT;
RUN;
PROC PRINT DATA=PRETTY (WHERE=(SORTO GT .)) SPLIT='*';
   BY SORTO;
   ID SORTO;
   LABEL TRT = "&_1 *(N= &N1.)"
      AE = '*'
      CHAR='*'
   ;
   VAR TRT AE CHAR;
RUN;
%MEND REPORT;

The macro SPP checks if the program is being run in batch and a parameter is being passed using SYSPARM() or if the program is being run interactively when you can comment the cases that are not being used in this case. In both cases, the macro variable PARAM, that is used for titles and line formats and WHERE that filters the required data are being produced.

In this paper, the cases for _AE and OTHERWISE are using do statements to show how more than one macro variable can be created if needed.

%MACRO SPP;
   DATA _NULL_;
   %* SYSPARM() IS THE WAY PARAMETERS CAN BE PASSED INTO SAS IN BATCH;
   PARAM=UPCASE(SYSPARM());
   %* CREATE MACRO VARIABLE &PARAM;
   CALL SYMPUT('PARAM',COMPRESS(PUT(PARAM,$25.)));
%* CREATE MACRO VARIABLE &WHERE THAT FILTERS DATA WITH SPECIFICATIONS;

SELECT (PARAM);
  WHEN ('_AE')        DO;
    CALL SYMPUT('WHERE','');
  END;
  WHEN ('_RELAE')     CALL SYMPUT('WHERE','WHERE RELAT GT 1');
  WHEN ('_SERAE')     CALL SYMPUT('WHERE','WHERE SAE EQ 1');
  WHEN ('_RELSERAE')  CALL SYMPUT('WHERE','WHERE RELAT GT 1 AND SAE EQ 1');
  WHEN ('_AEDISC')    CALL SYMPUT('WHERE','WHERE A_DISC EQ 1');
  WHEN ('_RELAEDISC') CALL SYMPUT('WHERE','WHERE A_DISC EQ 1 AND RELAT GT 1');
  WHEN ('_RELAEREDUCE') CALL SYMPUT('WHERE','WHERE RELAT GT 1 AND A_RED EQ 1');
  WHEN ('_RELAEDELAY') CALL SYMPUT('WHERE','WHERE RELAT GT 1 AND A_DEL EQ 1');
  WHEN ('_AEDEATH')   CALL SYMPUT('WHERE','WHERE OUTCOME EQ 4');
  WHEN ('_RELAEDEATH') CALL SYMPUT('WHERE','WHERE OUTCOME EQ 4 AND RELAT GT 1');
  WHEN ('_RELAEINTERR') CALL SYMPUT('WHERE','WHERE A_INT EQ 1 AND RELAT GT 1');
  OTHERWISE DO;
    %PUT This program is being run interactively;
  END;
END;
RUN;
%* IF THERE IS A PARAMETER, ONLY IN BATCH RUN THE REQUIRED TABLE;

%IF &PARAM GT %THEN %DO;
  %REPORT(WHERE=&WHERE, PARAM=&PARAM);
%END;

%* IF THERE IS NO PARAMETER, ASSIGN PARAMETERS PARAM AND WHERE AND CALL THE MACRO AS MANY TIMES AS NEEDED
%* AND RUN ALL THE TABLES;
%ELSE %IF &PARAM EQ %THEN %DO;
  %REPORT(PARAM=_AE);
  %REPORT(PARAM=_RELAE, WHERE = WHERE RELAT GT 1);
  %REPORT(PARAM=_SERAE, WHERE = WHERE SAE EQ 1);
  %REPORT(PARAM=_RELSERAE, WHERE = WHERE RELAT GT 1 AND SAE EQ 1);
  %REPORT(PARAM=_AEDISC, WHERE = WHERE A_DISC EQ 1);
  %REPORT(PARAM=_RELAEDISC, WHERE = WHERE A_DISC EQ 1 AND RELAT GT 1);
  %REPORT(PARAM=_RELAEREDUCE, WHERE = WHERE RELAT GT 1 AND A_RED EQ 1);
  %REPORT(PARAM=_RELAEDELAY, WHERE = WHERE RELAT GT 1 AND A_DEL EQ 1);
  %REPORT(PARAM=_AEDEATH, WHERE = WHERE OUTCOME EQ 4);
  %REPORT(PARAM=_RELAEDEATH, WHERE = WHERE OUTCOME EQ 4 AND RELAT GT 1);
  %REPORT(PARAM=_RELAEINTERR, WHERE = WHERE A_INT EQ 1 AND RELAT GT 1);
%END;
%MEND CALL;
%SPP;

CONCLUSION
You don't need several programs for each TLF when the generic work can be done in one program. With the use of a Selective Parameter Passing parameter variable that can be done in batch or interactively, one program can produce all the output. If it is not desired that all the TLFs are produced, the output can be limited with the parameter that is passed and

Many of the tables, listings and figures (TLFs) that are needed in a study are similar and not totally unique. There are several ways to try to limit the code that will be needed to generate multiple TLFs with one main program.

CONTACT INFORMATION
A. Cecilia Mauldin
mamadenina@hotmail.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. (r) indicates USA registration.

Other brand and product names are trademarks of their respective companies.