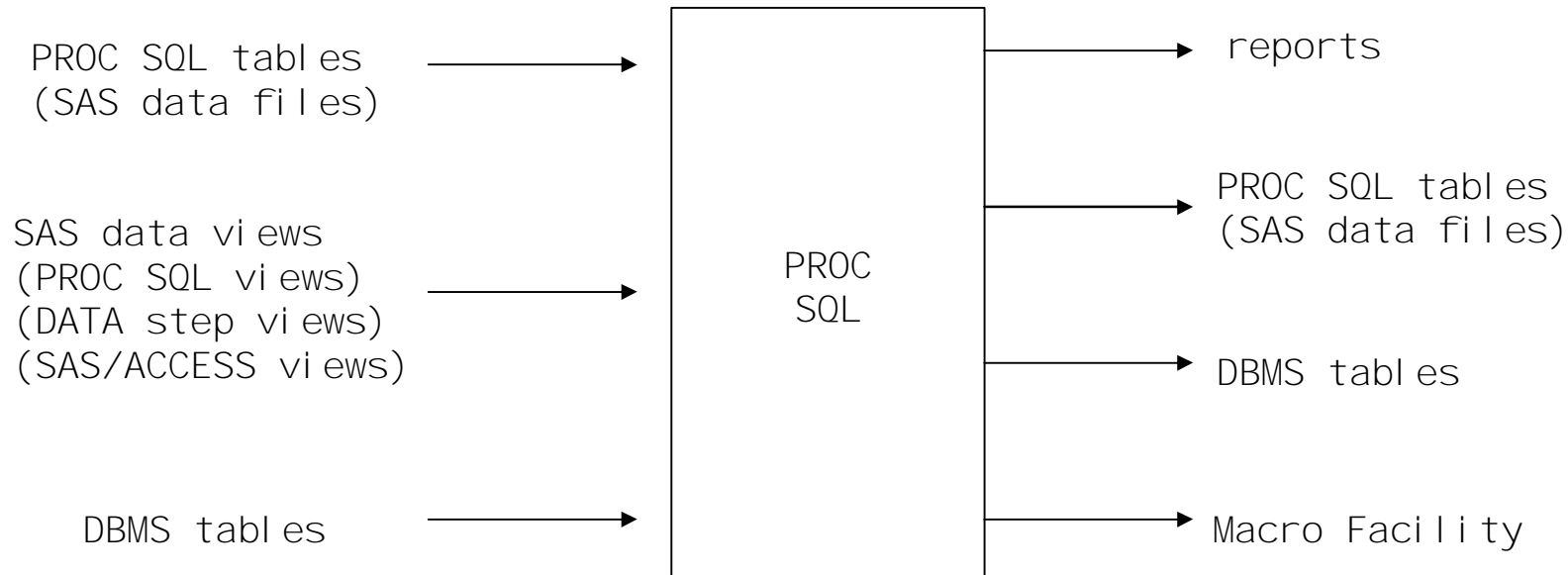


Creating Macro Variables with Proc SQL



SYSTEMS SEMINAR CONSULTANTS, INC.

Gerald Frey

2997 Yarmouth Greenway Drive Madison, WI 53711

(608) 278-9964

www.sys-seminar.com

Introduction



Quite often we need to:

- Read data into a data step
- Use Call SYMPUT to create macro variables
- Use those macro variables in a macro loop.

Proc SQL can be an alternative to SYMPUT!

A Typical Set of Data



Emergency room visits with doctor names and charged amounts.

```
data ervi si ts;  
input doctor $ charged;  
datal ines;  
Whi te 358  
Smi th 935  
Whi te 421  
Jones 144  
Smi th 105  
Jones 1234  
;  
run;
```

Get First Value From Dataset into Macro Variable



Program

```
data _null_;  
  set ervisits;  
  if _n_ = 1 then  
    call symput('frstdoc', doctor);  
run;  
%put First ER Visit on Record was treated by &frstdoc;
```

Log

The First ER Visit on Record was treated by White

PROC SQL Syntax



A very powerful procedure in Base SAS®.

PROC SQL *options*;

SELECT *column(s)*

FROM *table-name | view-name*

WHERE *expression*

GROUP BY *column(s)*

HAVING *expression*

ORDER BY *column(s)*

The Select Clause INTO



Give SQL an interface to the SAS[®] macro language.

SELECT <DISTINCT> *object-item* <,*object-item*>...

INTO *:macro-variable-specification*
< , *:macro-variable-specification*>...

Where *:macro-variable-specification* is one of the following:

:macro-variable <SEPARATED BY '*character*' <NOTRIM>>;

:macro-variable-1 - :macro-variable-n <NOTRIM>;

Get First Value From Dataset into Macro Variable



SQL will stop after selecting one value.

```
proc sql noprint;  
  SELECT doctor  
  INTO :frstdoc  
  from ervi si ts;  
quit;
```

```
%put First ER Vi si t on Record was treated by &frstdoc;
```

LOG

```
The First ER Vi si t on Record was treated by White
```

Select ALL Values Into a Single Macro Variable



The Separated clause inserts all values and separators.

```
proc sql noprint;
  SELECT doctor
  INTO :docs separated by ', '
  from ervi si ts;
quit;
```

```
proc tabulate data=ervi si ts;
  title "The following Providers are ";
  title2 "Included in the Summary Info: ";
  title3 "&docs";
  var charged;
  tables charged*f=dol l ar10. 2*mean;
run;
```

Select ALL Values Into a Single Macro Variable



Results

The following Providers are
Included in the Summary Info:
White, Smith, White, Jones, Smith, Jones
„ffffffffffff†
, charged ,
‡ffffffffffff%
, Mean ,
‡ffffffffffff%
, \$532.83,
Šffffffffffff€

Select ALL Unique Values Into a Macro Variable



DISTINCT eliminates duplicates.

```
proc sql noprint;
  SELECT distinct doctor
  INTO :docs separated by ', '
  from ervi si ts;
quit;
```

```
proc tabulate data=ervi si ts;
  title "The fol lowi ng Provi ders are ";
  title2 "Incl uded i n the Summary Info: ";
  title3 "&docs";
  var charged;
  tables charged*f=dol l ar10. 2*mean;
run;
```

Select ALL Unique Values Into a Macro Variable



Results

The following Providers are
Included in the Summary Info:
Jones, Smith, White

```
„ ffffffff†  
, charged ,  
‡fffffffff%  
, Mean ,  
‡fffffffff%  
, $532.83,  
Šfffffffff€
```

Select ALL Values into Macro Variable with QUOTES



Quote produces double quotes, compbl deletes consecutive blanks.

```
proc sql noprint;
  SELECT distinct quote(compbl (doctor))
  INTO :docs separated by ', '
  from ervi si ts;
quit;
```

```
%put &docs;
```

LOG

```
"Jones ", "Smi th ", "Whi te "
```

To Change Double Quotes to Single Quotes



Then TRANWRD data step function can translate characters.

```
%put &docs before;  
%let docs=  
%sysfunc(tranwrd(%bquote(&docs), %bquote("), %bquote(' )));  
%put &docs after;
```

LOG

```
" Jones ", " Smith ", " White " before
```

```
' Jones ', ' Smith ', ' White ' after
```

Quoted String Application



Quoted strings work nicely with the IN clause.

```
proc sql ;
  create table allclms as
  select *
  from claims
  where doctor in (&docs);
quit;
```

LOG

```
378 proc sql ;
379   create table allclms as
380   select *
381   from claims
382   where doctor in (&docs);
```

SYMBOLGEN: Macro variable DOCS resolves to ' Jones ', ' Smith
' , ' White '

Creating a Macro Variable with Results of a Function



SAS[®] functions can be part of the select clause, %let trims leading blanks.

```
proc sql noprint;
  SELECT max(charged) format=dollar8. ,
         min(charged) format=dollar8.
  INTO :maxchrg, :minchrg
  from ervisi ts;
quit;
%let maxchrg=&maxchrg;
%let minchrg=&minchrg;
%put maxchrg=&maxchrg;
%put minchrg=&minchrg;
```

Log:

```
maxchrg=    $1,234
minchrg=     $105
```

Creating a Macro Variable with Results of a Function



The macro values can be used in any code such as a title;

```
proc sort data=ervisi ts;  
  by doctor;  
run;
```

```
title "ER Visits Range in Cost from &minchrg to &maxchrg";  
proc print data=ervisi ts;  
  by doctor;  
  id doctor;  
  sum charged;  
  format charged dollar8. ;  
run;
```

Creating a Macro Variable with Results of a Function



RESULTS

```
ER Visits Range in Cost from $105 to $1,234
```

doctor	charged
Jones	\$144
	\$1,234
-----	-----
Jones	\$1,378
Smi th	\$935
	\$105
-----	-----
Smi th	\$1,040
Whi te	\$358
	\$421
-----	-----
Whi te	\$779
	=====
	\$3,197

Creating a Macro Variable with Number of Obs



COUNT function counts unique values of doctor.

```
proc sql noprint;
  select count(distinct doctor)
  into :numrows
  from ervi sites;
quit;
```

```
%let numrows=&numrows;
%put numrows=&numrows;
```

Log:

```
numrows=3
```

A Macro Application Using Symput



Generate a PROC PRINT for each doctor.

```
%macro loop;
proc sort data=ervisi ts out=temperv nodupkey;
  by doctor;
run;
data _null_;
  set temperv end=eof;
  call symput('doc'!!left(put(_n_, 2.)), doctor);
  if eof then
    call symput('numrows', left(put(_n_, 2.)));
run;
%put _user_;
```

A Macro Application Using Symput (continued)



```
%do i = 1 %to &numrows;
  proc print data=ervi si ts;
    Title "Er Vi si ts for Doctor &&doc&i ";
    where doctor=" &&doc&i ";
  run;
%end;

%mend l oop;
%l oop
```

The Generated Code Displayed in The (Log)



```
GLOBAL DOC1 Jones
GLOBAL DOC2 Smith
GLOBAL DOC3 White
GLOBAL NUMROWS 3
MPRINT(LOOP): proc print data=ervisits;
MPRINT(LOOP): Title "Er Visits for Doctor Jones ";
MPRINT(LOOP): where doctor="Jones ";
MPRINT(LOOP): run;
MPRINT(LOOP): proc print data=ervisits;
MPRINT(LOOP): Title "Er Visits for Doctor Smith ";
MPRINT(LOOP): where doctor="Smith ";
MPRINT(LOOP): run;
MPRINT(LOOP): proc print data=ervisits;
MPRINT(LOOP): Title "Er Visits for Doctor White ";
MPRINT(LOOP): where doctor="White ";
MPRINT(LOOP): run;
```

Creating a Macro Variable with Number of Obs



Replace the sort and data step with PROC SQL

```
%macro loop;  
proc sql noprint;  
  select count(distinct doctor)  
    into :numrows  
    from ervisits;
```

```
%let numrows=&numrows;
```

```
proc sql noprint;  
  select distinct doctor  
    into :doc1-:doc&numrows  
    from ervisits;
```

```
quit;
```

```
%put _user_;
```

Creating a Macro Variable with Number of Obs



The rest of the macro is unchanged.

```
%do i =1 %to &numrows;
proc print data=ervi si ts;
  Title "Er Vi si ts for Doctor &&doc&i ";
  where doctor=" &&doc&i ";
run;
%end;
%mend l oop;
%l oop
```

Questions and comments?



Gerald Frey

608 278-9964 x321

gfrey@sys-seminar.com