ABSTRACT

For SAS® Enterprise Guide® users, sometimes macro variables and their values need to be brought over to the local workspace from the server, especially when multiple data sets or outputs need to be written to separate files in a local drive. Manually retyping the macro variables and their values in the local workspace after they have been created on the server workspace would be time-consuming and error-prone, especially when we have quite a number of macro variables and values to bring over. Instead, this task can be achieved in an efficient manner by using dictionary tables and the CALL SYMPUT routine, as illustrated in more detail below. The same approach can also be used to bring macro variables and their values from the local to the server workspace.

SYNTAX AND EXPLANATION

Suppose that we have the following code lines in the server workspace that would produce a series of user-defined macro variables and assign specific values to them:

```sas
%let first_summer = 200830;
%let first_fall = 200910;
%let last_summer = 201430;
%let last_fall = 201510;
%let majors = BIO,MATH,PHYS,COMM,CHEM,HIST,GEO,ENG,FIN,INS;

data colleges;
  infile datalines;
  input college $;
  datalines;
  AH
  AR
  BE
  BU
  DN
  ED
  EG
  GR
  HS
  IN
  LF
  MD
  NR
  PH
  SW
  VR
  WS;
  run;
```
data depts;
infile datalines;
input dept $;
datalines;
BUSS
WRLD
NURS
EDUC
BIOC
MASC
CHSC
ANAT
SLWK
MUSC
GVPA
; run;

proc sql noprint;
* calculate the total number of rows in the "colleges" data set;
select distinct count(*) into: n_coll from colleges;
* read each of the values in the column "college" into a macro variable;
select distinct college into :coll_1 - :coll_%trim(&n_coll.) from colleges;
quit;

proc sql noprint;
* calculate the total number of rows in the "depts" data set;
select distinct count(*) into: n_dept from depts;
* read each of the values in the column "dept" into a macro variable;
select distinct dept into :depts separated by " " from depts;
quit;

%put _user_;

After being executed, the above code lines would produce the following macro variables and values:

GLOBAL _SASSERVERNAME  'SASApp' *(This is the server workspace.)*
GLOBAL COLL_1   AH
GLOBAL COLL_10   IN
GLOBAL COLL_11   LF
GLOBAL COLL_12   MD
GLOBAL COLL_13   NR
GLOBAL COLL_14   PH
GLOBAL COLL_15   SW
GLOBAL COLL_16   VR
GLOBAL COLL_17   WS
GLOBAL COLL_2   AR
GLOBAL COLL_3   BE
GLOBAL COLL_4   BU
GLOBAL COLL_5   DN
What we would like to do now is bring these macro variables and their values to the local workspace. In order to achieve this, we need to obtain a SAS macro dictionary table where all macro variables and their values are stored.

```
proc sql;
create table macro_list_all as select distinct * from dictionary.macros
; quit;
```

Below is what our SAS macro dictionary table looks like. It contains all macro variables and values that have been created by the system or the current user during a session. The ones highlighted in yellow are those we would want to bring to the local workspace.

<table>
<thead>
<tr>
<th>scope</th>
<th>name</th>
<th>offset</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTOMATIC</td>
<td>SYSDATE</td>
<td>0</td>
<td>15JUN15</td>
</tr>
<tr>
<td>AUTOMATIC</td>
<td>SYSDATE9</td>
<td>0</td>
<td>15JUN2015</td>
</tr>
<tr>
<td>AUTOMATIC</td>
<td>SYSDAY</td>
<td>0</td>
<td>Monday</td>
</tr>
<tr>
<td>AUTOMATIC</td>
<td>SYSSCP</td>
<td>0</td>
<td>LIN X64</td>
</tr>
<tr>
<td>AUTOMATIC</td>
<td>SYSSCPL</td>
<td>0</td>
<td>Linux</td>
</tr>
<tr>
<td>AUTOMATIC</td>
<td>SYSSITE</td>
<td>0</td>
<td>0070001453</td>
</tr>
<tr>
<td>AUTOMATIC</td>
<td>SYSVLONG</td>
<td>0</td>
<td>9.02.02M3P041310</td>
</tr>
<tr>
<td>AUTOMATIC</td>
<td>SYSVLONG4</td>
<td>0</td>
<td>9.02.02M3P04132010</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_1</td>
<td>0</td>
<td>AH</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_10</td>
<td>0</td>
<td>IN</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_11</td>
<td>0</td>
<td>LF</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_12</td>
<td>0</td>
<td>MD</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_13</td>
<td>0</td>
<td>NR</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_14</td>
<td>0</td>
<td>PH</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_15</td>
<td>0</td>
<td>SW</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_16</td>
<td>0</td>
<td>VR</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_17</td>
<td>0</td>
<td>WS</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_2</td>
<td>0</td>
<td>AR</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_3</td>
<td>0</td>
<td>BE</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_4</td>
<td>0</td>
<td>BU</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_5</td>
<td>0</td>
<td>DN</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_6</td>
<td>0</td>
<td>ED</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_7</td>
<td>0</td>
<td>EG</td>
</tr>
</tbody>
</table>
We trim the macro dictionary table a bit and just keep macro variables and values that we have created and would like to bring to the local workspace. These user-defined macro variables have global scope and their names do not start with ‘SYS’, ‘SQL’, ‘SAS’, or ‘_’.

```sas
data macro_list; set macro_list_all;
where scope = 'GLOBAL'
and (substr(name, 1, 3) not in ('SYS', 'SQL', 'SAS')
and substr(name, 1, 1) not in ('_'));
run;
```

<table>
<thead>
<tr>
<th>scope</th>
<th>name</th>
<th>offset</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLOBAL</td>
<td>COLL_1</td>
<td>0</td>
<td>AH</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_10</td>
<td>0</td>
<td>IN</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_11</td>
<td>0</td>
<td>LF</td>
</tr>
<tr>
<td>GLOBAL</td>
<td>COLL_12</td>
<td>0</td>
<td>MD</td>
</tr>
</tbody>
</table>
The next step is to bring this "macro_list" data set to the local workspace and run the following data step to recreate the macro variables and their values:

```plaintext
data _null_; set macro_list;
call symput(name, value);
run;

%put _user_;
```

The log shows the following results:

```
GLOBAL _SASSERVERNAME 'Local' (This is the local workspace.)
GLOBAL COLL_1   AH
GLOBAL COLL_10  IN
GLOBAL COLL_11  LF
GLOBAL COLL_12  MD
GLOBAL COLL_13  NR
GLOBAL COLL_14  PH
GLOBAL COLL_15  SW
GLOBAL COLL_16  VR
GLOBAL COLL_17  WS
GLOBAL COLL_2   AR
GLOBAL COLL_3   BE
GLOBAL COLL_4   BU
GLOBAL COLL_5   DN
GLOBAL COLL_6   ED
GLOBAL COLL_7   EG
GLOBAL DEPTS 0 ANAT BIOC BUSS CHSC EDUC GVPA MASC MUSC NURS SLWK WRLD
GLOBAL FIRST_FALL 0 200910
GLOBAL FIRST_SUMMER 0 200830
GLOBAL LAST_FALL 0 201510
GLOBAL LAST_SUMMER 0 201430
GLOBAL MAJORS 0 BIO,MATH,PHYS,COMM,CHEM,HIST,GEO,ENG,FIN,INS
GLOBAL N_COLL 0 17
GLOBAL N_DEPT 0 11
```

The next step is to bring this "macro_list" data set to the local workspace and run the following data step to recreate the macro variables and their values:

```plaintext
data _null_; set macro_list;
call symput(name, value);
run;

%put _user_;
```

The log shows the following results:

```
GLOBAL _SASSERVERNAME 'Local' (This is the local workspace.)
GLOBAL COLL_1   AH
GLOBAL COLL_10  IN
GLOBAL COLL_11  LF
GLOBAL COLL_12  MD
GLOBAL COLL_13  NR
GLOBAL COLL_14  PH
GLOBAL COLL_15  SW
GLOBAL COLL_16  VR
GLOBAL COLL_17  WS
GLOBAL COLL_2   AR
GLOBAL COLL_3   BE
GLOBAL COLL_4   BU
GLOBAL COLL_5   DN
GLOBAL COLL_6   ED
GLOBAL COLL_7   EG
```
GLOBAL COLL _8  GR
GLOBAL COLL _9  HS
GLOBAL DEPTS  ANAT BIOC BUSS CHSC EDUC GVPA MASC MUSC NURS SLWK WRLD
GLOBAL FIRST_FALL  200910
GLOBAL FIRST_SUMMER  200830
GLOBAL LAST_FALL  201510
GLOBAL LAST_SUMMER  201430
GLOBAL MAJORS  BIO,MATH,PHYS,COMM,CHEM,HIST,GEO,ENG,FIN,INS
GLOBAL N_COLL  17
GLOBAL N_Dept  11
CONCLUSION

For various purposes, we sometimes need to bring to the local workspace the macro variables and values that we have created in the server workspace. Retyping those macros and their values would be time-consuming and error-prone, especially when we have quite a number of macro variables and values to bring over.

This task can be achieved by using dictionary tables and call symput routine in the following steps:

- Obtaining a macro dictionary table that stores all macro variables and values already created in the server workspace during a session
- Bring this macro dictionary table over to the local workspace
- Use call symput routine to recreate these macro variables and values in the local workspace

When create a user-defined macro variable, it is advisable that the name of the macro variable does not start with 'SYS','SQL', 'SAS', or '_' as these characters are used by SAS to create systematic macro variables.

The approach illustrated in this presentation can also be used to bring macro variables and their values from the local to the server workspace.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Khoi Dinh To, PhD
Office of Planning and Decision Support, Virginia Commonwealth University
todinhkhoi@yahoo.com, kdto@vcu.edu

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. © indicates USA registration.

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