

20 SAS Macros Tips in 30 Minutes

March, 2009

Stephen Hanks

Campaign Reporting Manager

Customer Analytics, Brand and Marketing



Tip 1 - Create a Macro Variable

```
%let a=this is a macro variable;
```

```
%let report_date=01Jan2009;
```

Tip 2 - %Put

```
%let a=this is a macro variable;
```

```
%put &a;
```

this is a macro variable

```
%put a=&a;
```

a=this is a macro variable

Tip 2 - %Put

```
%put _all_;
```

```
GLOBAL A this is a macro variable  
AUTOMATIC AFDSID 0  
AUTOMATIC AFDSNAME  
AUTOMATIC AFLIB  
AUTOMATIC AFSTR1  
AUTOMATIC AFSTR2  
AUTOMATIC FSPBDV  
AUTOMATIC SYSBUFFR  
AUTOMATIC SYSCC 0  
AUTOMATIC SYSCHARWIDTH 1  
AUTOMATIC SYSCMD  
AUTOMATIC SYSDATE 06MAR09  
AUTOMATIC SYSDATE9 06MAR2009
```

Tip 2 - %Put

```
%put _user_;
```

GLOBAL A this is a macro variable

Other options include:

- _user_
- _automatic_
- _global_
- _local_
- _all_

Tip 3 - Create a Macro Variable in a Data Step

```
data _null_;  
    date=intnx('month',today(),-2,'B');  
    call symput('run_date',put(date,date9.));  
run;
```

```
%put run_date=&run_date;
```

```
12 data _null_;  
13   date=intnx('month',today(),-2,'B');  
14   call symput('run_date',put(date,date9.));  
15 run;
```

NOTE: DATA statement used (Total process time):

real time	0.03 seconds
cpu time	0.03 seconds

```
16  
17 %put run_date=&run_date;  
run_date=01JAN2009
```

Tip 4 - Create a Macro Variable Using SQL

```
proc sql noprint;  
    select max(age) into :maxage  
    from sashelp.class;  
quit;
```

```
%put maxage=&maxage;
```

```
19      %put maxage=&maxage;
```

```
maxage=    16
```

Tip 5 - Create a Macro Variable Using SQL (separated by)

```
proc sql noprint;
    select name into :names separated by ','
    from sashelp.class;
quit;

%put names=&names;
```

```
20    %put names=&names;
```

```
names=Alfred,Alice,Barbara,Carol,Henry,James,Jane,Janet,Jeffrey,John,Joyce,Judy,Louise,Mary,Philip,Robert,Ronald,Thomas,William
```

Tip 6 - Create a SAS Macro

```
%macro newmac;  
    *sas staements used in here;  
%mend;  
  
*to call the macro;  
%newmac;
```

Tip 7 - Macro for Reusing Code

```
%macro month_data(month=);  
proc sort data=SourceD.acct_&month. (keep=acct_key open_date) out=acct;  
  by acct_key;  
  where status_code='OP';  
run;  
  
data acct_&month.;  
  merge acct(in=a) sourced.acct_fin_&month. (keep=acct_key bal_amt);  
  by acct_key;  
  if a;  
run;  
  
%mend;  
  
%month_data(month=200901);  
%month_data(month=200812);  
%month_data(month=200811);
```

Tip 8 – Use Macros to Build up SAS Code

```
%macro runit;  
data all2;  
  set all;  
  %do i=1 %to 10;  
    if status_code_&i='OP' then open_&i=1;  
    else open_&i=0;  
    %if &i ne 10 %then %do;  
      if open_&i=0 and then open_this_month_&i=1;  
      else open_this_month_&i=0;  
    %end;  
  %end;  
run;  
%mend runit;  
%runit;
```

Tip 9 – Sending SAS Code to a File

```
options mprint mfile;
filename mprint 'G:\working\Steve\macro_code.sas';
%macro runit;
    data final(keep=month type open pos_bal new closed);
        set summary;
            %do i=200801 %to 200812;
                month="&i";
                open=open_&i.;
                output;
            %end;
    run;
%mend;
%runit;
```

Tip 9 – Sending SAS Code to a File

```
data final(keep=month type open pos_bal new closed);  
set summary;  
month="200801";  
open=open_200801;  
output;  
month="200802";  
open=open_200802;  
output;  
month="200803";  
open=open_200803;  
output;  
month="200804";
```

Tip 10 – SYMEXIST to check if a Macro Variable Exists

```
%let a=1;
data _null_;
if symexist('a') then put '**** a exists';
Else put '**** a doesnt exist';
if symexist('b') then put '**** b exists';
Else put '**** b doesnt exist';
run;
```

**** a exists

**** b doesnt exist

NOTE: DATA statement used (Total process time):

real time	0.00 seconds
cpu time	0.00 seconds

Tip 11 – SYMGLOBAL and SYMLOCAL

```
%let c=yes;
%macro test;
%let d=yes;
data _null_;
if symlocal ('c') then put '**** c is local';
if symglobal('c') then put '**** c is global';
if symlocal ('d') then put '**** d is local';
if symglobal('d') then put '**** d is global';
run;
%mend test;
%test;
```

**** c is global

**** d is local

NOTE: DATA statement used

(Total process time):

real time 0.00 seconds

cpu time 0.00 seconds

Tip 12 – Deleting a Macro Variable

```
%let a=yes;
data _null_;
if symexist('a') then put '**** a exists';
call symdel('a');
if symexist('a') then put '**** a still exists';
else put '**** a doesnt exist';
run;
```

**** a exists

**** a doesnt exist

NOTE: DATA statement used (Total
process time):

real time	0.01 seconds
cpu time	0.01 seconds

Tip 13 - MPRINTNEST

```
options mprintnest mprint;

%macro print(table=);
    proc print data=&table;
    run;
%mend print;

%macro runit;
    data test;
        set sashelp.vtable;
        if libname='SOURCED';
        count=_n_;
    run;

    %print(table=test);
%mend runit;

%runit;
```

Tip 13 - MPRINTNEST

```
MPRINT(RUNIT): data test;  
MPRINT(RUNIT): set sashelp.vtable;  
MPRINT(RUNIT): if libname='SOURCED';  
MPRINT(RUNIT): count=_n_;  
MPRINT(RUNIT): run;
```

NOTE: There were 2246 observations read from the data set SASHELP.VTABLE.

NOTE: The data set WORK.TEST has 1640 observations and 40 variables.

NOTE: DATA statement used (Total process time):

real time	59.65 seconds
cpu time	3.17 seconds

```
MPRINT(RUNIT.PRINT): proc print data=test;  
MPRINT(RUNIT.PRINT): run;
```

NOTE: There were 1640 observations read from the data set WORK.TEST.

NOTE: PROCEDURE PRINT used (Total process time):

real time	0.29 seconds
cpu time	0.15 seconds

```
MPRINT(RUNIT): ;
```

Tip 14 - %EVAL and %SYSEVALF

```
%put a=%eval(7/3);  
%put b=%sysevalf(7/3);
```

```
77 %put a=%eval(7/3);  
a=2  
78 %put b=%sysevalf(7/3);  
b=2.3333333333333333
```

Tip 15 – Using %SYSFUNC For Report Date

```
proc print data=sashelp.class;  
title "We can put a todays (%sysfunc(today()),date9.) date  
in a report title";  
run;
```

We can put a todays (09MAR2009) date in a report title

1

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0
4	Carol	F	14	62.8	102.5

Tip 16 – Using %SYSFUNC Functions

```
%let dsid=%sysfunc(open(datamart.customer)) ;  
%let nobs=%sysfunc(attrn(&dsid,nobs)) ;  
%let dsid=%sysfunc(close(&dsid)) ;  
%put The number of obs in datamart.customer is &nobs ;
```

```
81 %let dsid=%sysfunc(open(datamart.customer)) ;  
82 %let nobs=%sysfunc(attrn(&dsid,nobs)) ;  
83 %let dsid=%sysfunc(close(&dsid)) ;  
84 %put The number of obs in datamart.customer is &nobs ;  
The number of obs in datamart.customer is 6299497
```

Tip 17 - Macro Variables to Save Time

Not Using Macros

```
options fullstimer;  
  
data test;  
  set Sourced.acct  
    (where=(intnx('month',today(),-2,'B')<open_date));  
run;
```

NOTE: There were 133764 observations read from the data set SOURCED.ACCT.

WHERE open_date>INTNX('month', TODAY(), -2, 'B');

NOTE: The data set WORK.TEST has 133764 observations and 35 variables.

NOTE: DATA statement used (Total process time):

real time	1:12.04
user cpu time	10.26 seconds
system cpu time	1.64 seconds
Memory	221k

Tip 17 - Macro Variables to Save Time

Using Macros

```
data _null_;  
    date=intnx('month',today(),-2,'B');  
    call symput('run_date',date);  
run;
```

```
data test;  
    set Sourced.acct(where=( &run_date.<open_date) );  
run;
```

NOTE: There were 133764 observations read from the data set SOURCED.ACCT.

WHERE open_date>17898;

NOTE: The data set WORK.TEST has 133764 observations and 35 variables.

NOTE: DATA statement used (Total process time):

real time	1.96 seconds
user cpu time	0.64 seconds
system cpu time	1.04 seconds
Memory	217k

Tip 18 – Using Macros to Comment Out Code

If there is a large section of code you don't want to run?

Just put a “`%macro name`” at the top and a “`%mend`” at the end and never call the macro.

This effectively comments out the code without running it.

No need to worry about other comment styles in the code.

(Not) Tip 19 – Statement and Command Style Macros

```
options implmac;  
  
%macro printit(dset) / stmt;  
    proc print data=&dset;  
        run;  
  
%mend p;  
  
printit sashelp.class;
```

```
options cmdmac;  
  
%macro ls(lib) /cmd;  
    dir &lib;  
    tile;  
  
%mend ls;
```

Tip 20 – Resolve Function

```
%let first=Steve;  
%let last=Hanks;
```

```
data _null_;  
    name1='Mr '!!"&first"!!' '!!"&last";  
    name2=resolve('Mr &first &last');  
    put name1;  
    put name2;  
run;
```

```
105 data _null_;  
106   name1='Mr '!!"&first"!!' '!!"&last";  
107   name2=resolve('Mr &first &last');  
108   put name1;  
109   put name2;  
110 run;
```

```
Mr Steve Hanks  
Mr Steve Hanks
```

References

Thanks to:

- SAS online doc and notes
<http://support.sas.com/>
- Roland Rashleigh-Berry - Tips on Writing Macros
<http://www.datasavantconsulting.com/roland/macrotips.html>
- Phil Mason – SAS 9 Tips
http://www.cmg.org/measureit/issues/mit45/m_45_10.html
- Amadeus Software – Tips Page
<http://www.amadeus.co.uk/TipsList.aspx?ST=1>

Questions and Contact Details

Stephen Hanks
Campaign Reporting Manager
Brand and Marketing
St George
Email : hankss@stgeorge.com.au