QUICK 'N DIRTY
SMALL, USEFUL, UTILITY MACROS

GHSUG – 2015-10-30

Harry Droogendyk

Stratia Consulting Inc.

Introduction

- don't like macros ?
  - perhaps intimidated by monster macros
- never saw them useful ?
- live in the adhoc world ?
- not lazy enough ? ;-)
- this is not about big macros
- nor even so much *these* macros
- prompt you to think about similar macros

Four macros

- SAS is a tool box
  - robust, parameterized macros might be overkill
  - how about ones that free you from the mundane?

- %dups – code snippet, finds dup obs
- %fiscal – offset, formatted SAS dates
- %cleanup – work data / macro variables
- %single – '&resolve_THIS_hotdog'

What is Macro ?

- not as mysterious as it seems
- really nothing more than *text substitution*
  - don't repeatedly key same code or values
  - define it once
    - macro variable
    - macro to generate code or values
  - use it many times
- macro processor runs before SAS compiler

Duplicate Observations

- lousy data is our life
  - duplicate data is one of the problems
- PROC SORT has DUPOUT= option
  - unique obs are sent to the OUT= dataset
  - dups are output to DUPOUT= dataset
- helpful to have all observations with dup keys in one dataset

Duplicate Observations

```
data a;
 j = 'a'; k = 1; m = 1; output;
 j = 'a'; k = 1; m = 2; output;
 j = 'a'; k = 1; m = 3; output;
 j = 'a'; k = 2; m = 4; output;
 j = 'b'; k = 2; m = 5; output;
 j = 'b'; k = 2; m = 6; output;
run;
data dups;
 set a;
  by j k;
 if not(first.k and last.k);
run;
```

Duplicate Observations

```
%macro dups(v);
 if not ( first.&v and last.&v )
%mend;

data dups;
 set a;
  by j k;
 %dups(k);
run;
```

Dates by Fiscal Year

- work for a bank ?
  - in Canada their fiscal year ends Oct 31
  - summaries must report stuff in the right fiscal year / quarter / month
- %fiscal
  - accepts a date parameter
  - advances it two months
  - applies a format

Dates by Fiscal Year

```
%macro fiscal(help,date=,format=year.);

 %if &help = ? or %upcase(&help) = HELP %then %do;

 %put ;
 %put %nrstr(%fiscal(date=,format=););
 %put %nrstr(Returns date relative to bank fiscal year for the supplied SAS internal
 date.);
 %put %nrstr(Parms: &date   - SAS variable containing internal date value);
 %put %nrstr(     &format - output format, eg. month., year., yymmn6., yyq6.);
 %put %nrstr(Use:   fiscal = %fiscal(date=sas_date_variable,format=format););

 %end; %else %do;
```

Dates by Fiscal Year

`%fiscal(?);`

```
%fiscal(date=,format=);
Returns date relative to bank fiscal year for the supplied SAS
  internal date.

Parms: &date   - SAS variable containing internal date value
       &format - output format, eg. month., year., yymmn6., yyq6.
Use:   fiscal = %fiscal(date=sas_date_variable,format=format);
```

Dates by Fiscal Year

```
%if &sysprocname = %then %do;  %*  executing from macro ;

 %sysfunc(intnx(month,&date,2),&format)

%end; %else %do;    %*  executing from data or SQL step ;


 put(intnx('month',&date,2),&format)

%end;

%end;
%mend fiscal;
```

Dates by Fiscal Year

```
data a;
 date_fld = '22dec2014'd;
 fiscal_year = %fiscal(date=date_fld);
 put 'Date is: ' date_fld date9. ',
      Fiscal Year is: 'fiscal_year;
run;
```

Log output:

```
Date is: 22DEC2014,
Fiscal Year is: 2015
```

Cleaning Up

- batch processes start clean
  - config files and autoexec programs
- interactive SAS
  - initially pristine
  - run Program_A.sas
    - creates WORK datasets and macro variables
  - run Program_B.sas
    - first program's leftovers are hanging around

Cleaning Up

```
%macro cleanup(help,data=Y,macro=Y);

 %if %upcase(&data) = Y %then %do;

  %put NOTE: %nrstr(%cleanup is deleting WORK data);
  proc datasets lib=work nolist nowarn nodetails
       kill;
  quit;

 %end;
```

Cleaning Up

```
%if %upcase(&macro) = Y %then %do;
 %put NOTE: %nrstr(%cleanup is deleting GLOBAL macro
           variables);
 data _null_;
      length cmd $200;
      set sashelp.vmacro;
        where scope = 'GLOBAL' and offset = 0 and name
                      ne: 'SYSDB' and name ne: 'SYS';
      cmd = '%nrstr(%symdel ' || trim(name) || ' /
             nowarn );';
      call execute(cmd);
 run;
 %end;
%mend cleanup;
```

Cleaning Up

```
774  %cleanup;

NOTE: %cleanup is deleting WORK datasets
NOTE: Deleting WORK.NPV_CAMPAIGN (memtype=DATA).
NOTE: Deleting WORK.SASMACR (memtype=CATALOG).
NOTE: File WORK.SASMACR (memtype=CATALOG) cannot be deleted because
  it is in use.
<snip>
NOTE: %cleanup is deleting GLOBAL macro variables
NOTE: There were 1 observations read from the data set
  SASHELP.VMACRO.
      WHERE (scope='GLOBAL') and (offset=0) and (name not =:
  'SYSDB');
<snip>
NOTE: CALL EXECUTE generated line.
1   +  %symdel CAMPAIGN_CODE / nowarn ;
```

Resolving in Single Quotes

- do macro variables resolve within single quotes?
  - eg. '&test_var'

```
24    %let a = Hello;
25    %put '&a';
'&a'
26    %put "&a";
"Hello“
```

- is there ever need to use single quotes ?

Resolving in Single Quotes

- solutions
  - %sysfunc(compress("'&macro_var'",%str(%")))
  - %unquote(%str(%')&macro_var%str(%'))
- prefer the second
  - but I'm lazy, too much typing
  - probably mess up % or brackets
- another small utility macro, %single

Resolving in Single Quotes

```
%macro single(v);

 %unquote(%str(%')&v%str(%'))

%mend single;
```

Resolving in Single Quotes

```
%let date = %sysfunc(today(),yymmddd10.);
proc sql;
 connect to db2 ( database = ABC );
 select * from connection to db2 (
     select count(*) as cnt
       from schema.table
      where transaction_date =
          %single(&date) );
quit;
```

Resolving in Single Quotes

```
43   proc sql;
44         connect to db2 ( database = ABC );
45
46         select * from connection to db2 (
47               select count(*) as cnt
48                           from schema.table
49                where transaction_date
SYMBOLGEN:  Macro variable DATE resolves to 2015-10-
 30
49 !                                    = %single(&date)
 );
SYMBOLGEN:  Macro variable V resolves to 2015-10-30
MPRINT(SINGLE):   '2015-10-30'
```

Wrap

- four macros covered here NOT life changing
- however
  - demonstrates that macro isn't complex
  - illustrates simple implementations
  - may prompt your own simple macros
    - save typing
    - reduce errors
    - share code

Author

Harry Droogendyk

conf@stratia.ca

www.stratia.ca