Keep it Clean

A. Cecilia Mauldin, Comprehensive Drug Development (CDD), Morrisville, NC

ABSTRACT
An advantage of using Interactive PC SAS® consists on using data sets, formats, macro variables, macros that have been created previously in the same session. This advantage, can be dangerous if don’t pay attention and use one program’s data, formats, macros that were intended for another program.

Keeping unnecessary data can expensive because of the wasted storage space and can be dangerous if we use data/formats/macros that were created for one program in another.

The purpose of this paper is to show different ways to ‘clean’ temporary data, formats, macro variables, macros.

INTRODUCTION
When I work in interactive PC SAS, I open one session, and create, data sets, formats, macros and macro variables as I need, and if I open a new editor, there is the risk that I can use the data from the first program in the second editor. I need to check that when I am calling the dataset with counts for demographics, I am not getting the data set with the sum of Adverse Events, and the same holds for macro variables and macros.

Please note that I use the word TABLE and DATA SET indistinctively. PROC SQL, sees SAS data sets as tables.

Cleaning temporary data sets
I know two ways of removing temporary data sets or data sets that are stored in the WORK library, one with PROC SQL and another one with PROC DATASET.

In SQL we need to know the name of the datasets.

```sql
PROC SQL noprint;
    select strip(memname) into :DROPT separated by ','
    from dictionary.tables
    where libname='WORK'
;
    drop table &DROPT;
quit;
```

The first step creates a macro variable that contains the name of the tables in the WORK library. The second step drops the tables.

This works, but if there are no tables to be drop, the second step will produce a WARNING in the log indicating that there you need to write a table name after table; To avoid this you can use a macro that checks for the number of tables in the WORK directory and will only drop the tables if there is at least one.
%macro cleantables;
PROC SQL noprint;
    select strip(memname) into :DROPT separated by ','
    from dictionary.tables
    where libname='WORK'
    ;
    %let droptab=&sqlobs;
    %if &droptab>0 %then %do;
        drop table &DROPT;
    %end;
QUIT;
%mend cleantables;

From SAS Help and Documentation in the PROC SQL section:

- If no output table, macro variable list, or macro variable range is created, then SQLOBS contains the value 1.
- If an output table is created, then SQLOBS contains the number of rows in the output table.
- If a single macro variable is created, then SQLOBS contains the value 1.
- If a macro variable list or macro variable range is created, then SQLOBS contains the number of rows that are processed to create the macro variable list or range.

A second alternative is to use proc datasets, but you still need to check if there are tables in the WORK directory to avoid getting a WARNING message in the log.

proc datasets library=WORK kill memtype=data;
quit;

Cleaning temporary formats
I only know of one way to drop formats and it is using proc dataset.

proc datasets library=WORK kill memtype=catalog;
quit;

You will get a warning about macros in the catalog that cannot be deleted because they are in use, you can limit the times that you run the statement to only the times that there are formats in the directory.

proc sql;
    select count(*)
    from dictionaryFormats
    where libname='WORK'
    and memname='FORMATS'
    ;
quit;

Removing GLOBAL macro variables
There are GLOBAL macro variables that are created by SAS and should not be removed, also, there may be other GLOBAL macro variables that you may need to keep. The first step is to find the name of them, but I know that I do not want to delete (or create) any variable that starts with 'SYS', 'SQL', 'AF', 'FS'. DICTIONARY.MACROS has the name of all the current macro variables; we will limit our selection to the macro variable(s) where scope is GLOBAL.
```sql
proc sql;
select distinct name into :macvar separated by ' ' from dictionary.macros
where scope='GLOBAL'
and name and like 'SYS%' and name not like 'SQL%' and name not like 'AF%' and name not like 'FSBP%';
quit;

And to remove this variables:

%SYMDEL &MACVAR;
```

### Removing Macros, not macro variables?

SAS recommends that Macros are should removed during a session, if you go to WORK through explorer, you will be able to delete data sets and formats, but you will not be allowed to remove a macro, but, it is possible to recreate the macro to avoid confusion.

The first step, is to locate the name of the macros that exist,

```sql
PROC SQL;
CREATE TABLE MACNAMES AS
    SELECT OBJNAME FROM DICTIONARY.CATALOGS
    WHERE LIBNAME='WORK'
    AND OBJTYPE='MACRO';
QUIT;
```

Then, if there are any macros in the Work library, you can 'rewrite' the macro by;

```sas
DATA _NULL;
SET MACNAMES;
CALL EXECUTE( ' %MACRO '||STRIP(OBJNAME) ||'; %PUT MACRO '|| STRIP(OBJNAME) ||' NO LONGER EXISTS; %MEND ; ' );
RUN;
```

If the macro CLEANTABLES that was created above is called again the macro the log will show:

```sas
%cleantables;
MACRO CLEANTABLES NO LONGER EXISTS.
```

A problem will arise if this process is put in a macro, because the macro that is being executed cannot be overwritten.

### Cleaning macros, macro variables, formats and data sets.

Because in none of the methods described in this paper, there is a need to mention anything specific about the data or the catalogs that will be cleaned. I would store the individual macros in a directory that is not WORK.

When removing a macro this method involved creating a dataset, and to drop temporary tables, this method needs to create macro variables. further more if the macro that removes temporary macros is temporary, it would try to rewrite itself and the following message will be found in the Log: "ERROR: Macro library damaged", a cycle of dirty rags left behind.
CONCLUSION
It is possible to clean datasets, formats, macros and macro variables. It is desirable to have a fresh start and avoid confusion, but if the tools showed above are used, it is better to have them in the macros directory so they won’t interfere with each other.

But if you need to have them in your program as macros, be careful of the order that they are called and do not put the code that rewrites macros in a macro.

REFERENCES
Carpenter, Art, Carpenter’s Complete Guide to the SAS® Macro Language

ACKNOWLEDGEMENTS
Thank you to George Clark for your help.

CONTACT INFORMATION
A. Cecilia Mauldin
Comprehensive Drug Development
One Coplay Parkway, Suite 534
Morrisville, NC 27560
cmauldin@cnsmail.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. © indicates USA registration.