TIPS AND TRICKS: IMPROVE EFFICIENCY TO YOUR SAS PROGRAMMING

Guillaume Colley,
Lead Data Analyst,
BCCFE

BRITISH COLUMBIA
CENTRE *for* EXCELLENCE
in HIV/AIDS

Providence
HEALTH CARE
How you want to be treated.

a place of mind
UBC

Contents

- Customized SAS Session
  - Run system options as SAS starts
  - Labels management
  - Shortcut keys
- Coding Tips
  - Import data from Excel in 1 line
  - Export table/output to Excel
  - Sub-group analysis in 1 step
  - User functions
  - Output statement to duplicate rows

Customized SAS session

- You can make SAS run system options by default every time SAS starts up.
- Open C:\Program Files\SASHome\SASFoundation\9.3\nls\en\sasv9.cfg

sasv9.cfg - Notepad

File Edit Format View Help

```
/* set default locations                                          */
-TRAINLOC ""

/* set the default fileref for the PARMCARDS= option              */
-SET FT15F001 'FT15F001.DAT'

/*-----------------------------------------------------------------\
| SAS System FORMCHARS, used by pressing ALT then the decimal      |
| number for the Extended ASCII character.                         |
\-----------------------------------------------------------------*/

/* This is the ANSI character set (for SAS Monospace font and ANSI Sasfont) */
-FORMCHAR '82838485868788898A8B8C2B3D7C2D2F5C3C3E2A'x


/*-----------------------------------------------------------------\
| WARNING:  INSTALL Application edits below this line.  User       |
|           options should be added above this box comment.        |
|           INSTALL Application maintains and modifies the         |
|           following options; -SASAUTOS, -SASHELP, -SASMSG,       |
|           -PATH, and -MAPS.  It also maintains and modifies      |
|           the following CONFIG variables with the -SET option;   |
|           INSTALL, USAGE, LIBRARY, SAMPSIO, SAMPSRC, SASCBT,     |
|           and SASEXT01-SASEXT50.  It preserves all lines above   |
|           the line containing 'DO NOT EDIT BELOW THIS LINE'.     |
\-----------------------------------------------------------------*/


/* DO NOT EDIT BELOW THIS LINE - INSTALL Application edits below this line */
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
-SET SASROOT "C:\Program Files\SASHome\SASFoundation\9.3"
```

Customized SAS session

- Can be any system option;
  - from APPEND to YEARCUTOFF
  - PAGESIZE, DATESTYLE, FMTSEARCH, CMPLIB…
- A recommended option to set-up as default is MERGENOBY
  - By default, SAS will not say anything if you perform a MERGE without a BY statement
  - This option forces SAS to tell you that something is wrong (you can choose warning or error)

```
/* This is the ANSI character set (for SAS Monospa
-FORMCHAR '82838485868788898A8B8C2B3D7C2D2F5C3C3E2
-mergenoby=ERROR
/*-----------------------------------------------
| WARNING:  INSTALL Application edits below this
|           options should be added above this box
```

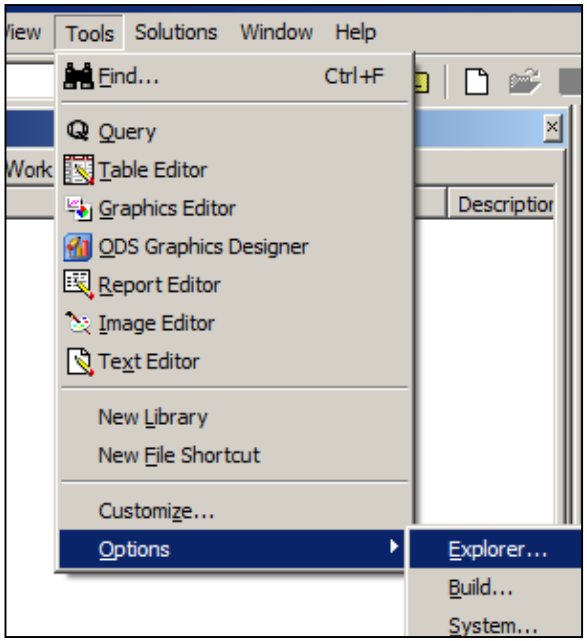
```
ERROR: No BY statement was specified for a MERGE statement.
NOTE: The SAS System stopped processing this step because of errors.
WARNING: The data set WORK.TEST may be incomplete.  When this step wa
         observations and 11 variables.
WARNING: Data set WORK.TEST was not replaced because this step was st
NOTE: DATA statement used (Total process time):
      real time           0.01 seconds
      cpu time            0.01 seconds
```

Customized SAS session

- **Getting rid of labels** (because these are really annoying, aren't they ?)
- Proc datasets:

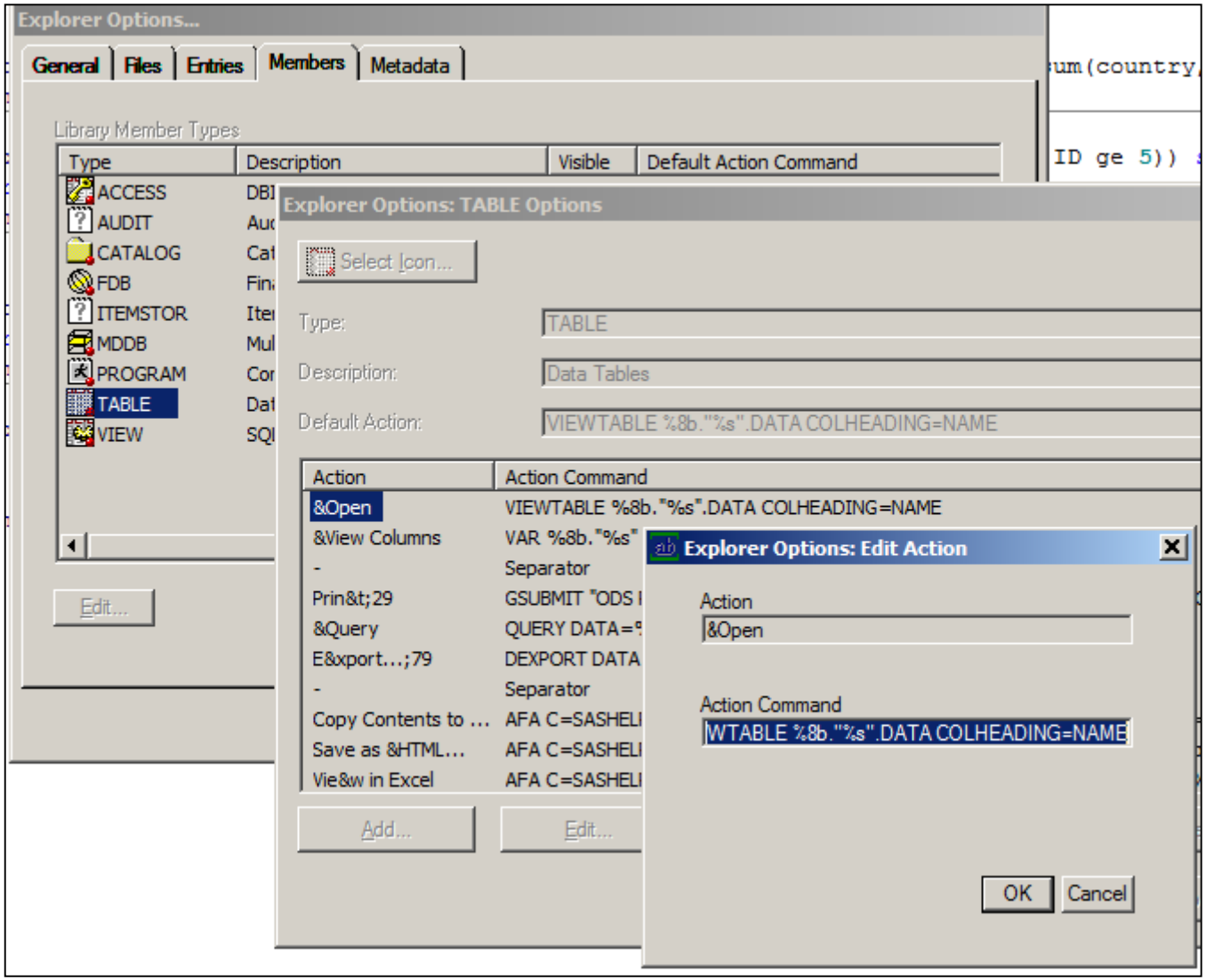
```
proc datasets lib=work memtype=data noprint;
   modify test;
     attrib _all_ label=' ';
run;
```

- You can make SAS explorer show variable ***names*** by default:
  - Select SAS explorer, then go
  - Tools,
  - Options,
  - Explorer



Customized SAS session

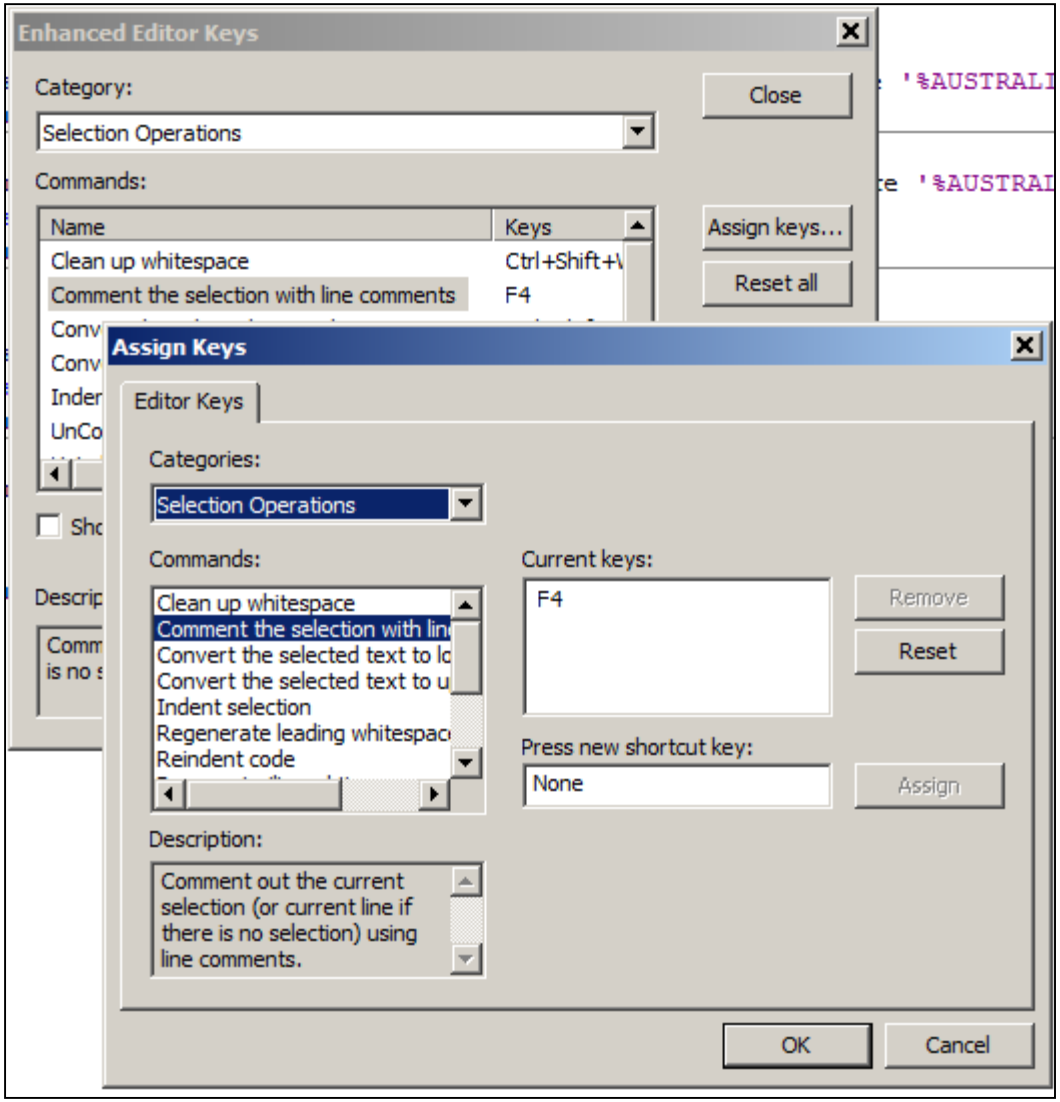
- Select the « Members » tab,
- Table
- &Open
- Add the following to the end of the action command:
- COLHEADING= NAME



Customized SAS session

- **Setting up shortcut keys**
  - Upper and lower case selection
  - Extend selection
  - Comment and Uncomment selection

- With the Editor being active:
- Tools, Options, Enhanced Editor Keys

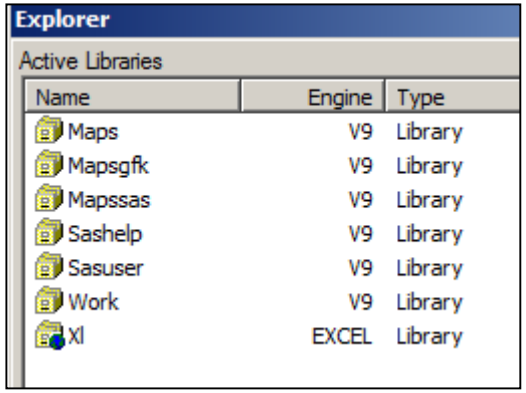


Coding Tips

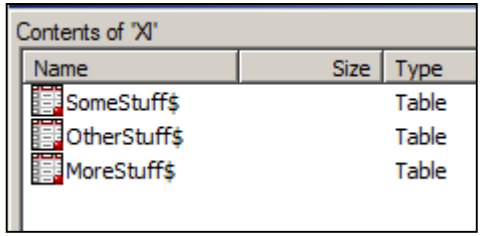
- **Import multiple Excel sheets in 1 line**

```
libname XL excel "C:\Temp\MyExcelFile.xlsx";
```

Your library name | Library type | Excel file path



- Will appear as a library and each worksheet will appear as a table



- Need to use name literals to be able to work with the above tables:
  - Use: `xl.'morestuff$'n`
  - `xl.morestuff$` will not work

Coding Tips

- **Export into Excel**
- Why you don't need to bother with labels, formatting, and try to make a SAS output look nice:
  - It is faster and easier to format your output in an Excel spreadsheet, and it makes it more workable/re-usable
  - Sas output is not going to look nice anyway

- One-time only exports:
  - Tables: right click on the table in the explorer, View in Excel
  - Proc output: use the Output Delivery System (ODS):

```
ODS HTML FILE="C:\Users\Guillaume\Desktop\TEMP.xls";
/* Any proc which exports something to the output window would work */
proc contents data=pcs_all;
run;
ods html close;
```

  - Many different options if you want to customize your ODS output

Coding Tips

- **Export into Excel**
- Recurring exports:
  - Spend time getting a nicely formatted Excel document
  - Use Dynamic Data Exchange (DDE) to make SAS write your results exactly where you want in your opened Excel spreadsheet:

```
filename name dde 'excel|Sheet1!r6c2:r9c4';
```

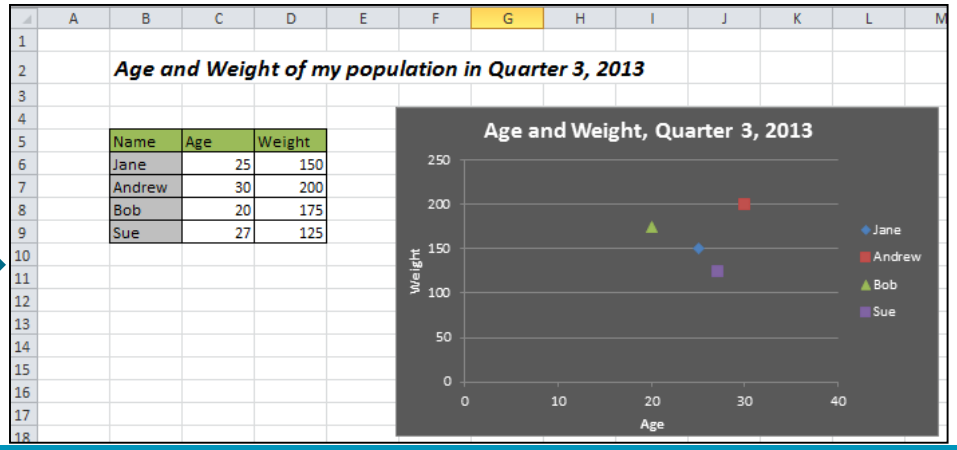
Defines where you are going to write your SAS data in Excel: sheet, top left cell and bottom right cell

```
data _null_;
set test;          name of SAS table to export
file name;         previously assigned filename
put var1 var2 var3 etc.;
run;
```

Writes your SAS data in the specified location

VIEWTABLE: Work.Test

	name	age	weight
1	Jane	25	150
2	Andrew	30	200
3	Bob	20	175
4	Sue	27	125



Coding Tips

- **Sub-group analysis in 1 step**
- You can use complex WHERE statements in data step headers or proc headers
  - To avoid the creation of sub-population tables

VIEWTABLE: Work.Canada

	PROVINCE	CDCODE	NAME	PROVNAME	men	women	happy
129	35	07	GRENVILLE	ONTARIO	11169	45340	1
130	35	08	LEEDS	ONTARIO	19523	44971	1
131	35	09	LANARK	ONTARIO	12132	37955	1
132	35	10	FRONTENAC	ONTARIO	32149	40286	0

```
proc means data=canada (where=(sum(women,men) le 20000 )) nway noprint;
class provname;
output out=small (drop=_type_) median=;
run;
```

```
proc means data=canada (where=(upcase(name) like '%DIVISION%' and women le men)) nway noprint;
class provname happy;
output out=happy (drop=_type_) sum=;
run;
```

Coding Tips

- **User functions**
- You can create, test, and store SAS subroutines, then use them in other SAS procedures or DATA steps

```
libname fct "X:\SAS Functions";

proc fcmp outlib=fct.userfuncs.example;

          function myfunct(arg1, arg2 $, arg3, arg4 $) $ ;
          /*** Your function's body ***/
          return(expression);
          endsub;
```

$ if the function returns a character variable (numeric otherwise)

Function compiler (FCMP) doesn't need a "run"

```
libname fct "X:\SAS Functions";
options cmplib=fct.userfuncs;

data test;
set blabla;
if myfunct(cd4, Name, pvl, city)='ELIGIBLE' then ;
run;
```

Coding Tips

- **User functions example**

If you want to use an array as argument you need to specify the VARARGS option

```
* ICD10;
proc fcmp outlib=fct.userfuncs.ICD;
            function HIVICD10(diagx1 $,diagx2 $, diagx3 $, diagx4 $, diagx5 $, diagx6 $, diagx7 $, diagx8 $, diagx9 $, diagx10 $,
diagx11 $,  diagx12 $, diagx13 $, diagx14 $, diagx15 $,diagx16 $, diagx17 $, diagx18 $, diagx19 $,
diagx20 $,  diagx21 $, diagx22 $, diagx23 $, diagx24 $,diagx25 $);

            IF (substr(diagx1,1,3)  in ( 'B20', 'B21',  'B22', 'B23', 'B24', 'R75', 'Z21')  )
   or  (substr(diagx2,1,3)  in ( 'B20', 'B21',  'B22', 'B23', 'B24', 'R75', 'Z21')  )
   or  (substr(diagx3,1,3)  in ( 'B20', 'B21',  'B22', 'B23', 'B24', 'R75', 'Z21')  )
   or  (substr(diagx4,1,3)  in ( 'B20', 'B21',  'B22', 'B23', 'B24', 'R75', 'Z21')  )
   or  (substr(diagx5,1,3)  in ( 'B20', 'B21',  'B22', 'B23', 'B24', 'R75', 'Z21')  )
   or  (substr(diagx6,1,3)  in ( 'B20', 'B21',  'B22', 'B23', 'B24', 'R75', 'Z21')  )
   or  (substr(diagx7,1,3)  in ( 'B20', 'B21',  'B22', 'B23', 'B24', 'R75', 'Z21')  )
   ...
   or  (substr(diagx7,1,5) = '79571'  )
   or  (substr(diagx8,1,5) = '79571'  )
   or  (substr(diagx9,1,5) = '79571'  )
   or  (substr(diagx10,1,5) = '79571' )
   or  (substr(diagx11,1,5) = '79571' )
   or  (substr(diagx12,1,5) = '79571' )

           then return(1);
           else return(0);
endsub;
```

```
data test;
set hospitalizations;
if HIVICD10(of diagx1-diagx25)=1 then HIV_pos=1; else HIV_pos=0;
run;
```

Coding Tips

- **Output in a data step**
- Implicit output

```
data TEST;
set VIRLOAD;
LN_VL=log(VLOAD);
run;
```

VIRLOAD data set

PATIENTID	DATE	VIRLOAD

TEST data set

PATIENTID	DATE	VIRLOAD	LN_VL

- Explicit output
  - The OUTPUT statement tells SAS to write the current observation to a SAS data set immediately, not at the end of the DATA step

```
data LOW_VL HIGH_VL;
set VIRLOAD;
LN_VL=log(VLOAD);
if LN_VL < 10 then output LOW_VL;
            else output HIGH_VL;
run;
```

VIRLOAD data set

PATIENTID	DATE	VIRLOAD

HIGH_VL dataset

PATIENTID	DATE	VIRLOAD	LN_VL

LOW_VL dataset

PATIENTID	DATE	VIRLOAD	LN_VL

Coding Tips

- **Using explicit output to duplicate rows**
- Weekly dataset to convert in a daily dataset

```
data DAILY (drop=i);
set WEEKLY;
    do i=0 to 6;
      output;
      dt=dt-1;
      end;
run;
```

WEEKLY data set

id	episode	dt
100041	1	19-Jul-06
100041	1	26-Jul-06
100041	1	2-Aug-06
100041	1	9-Aug-06
100041	1	20-Jan-10

DAILY data set

id	episode	dt
100041	1	19-Jul-06
100041	1	18-Jul-06
100041	1	17-Jul-06
100041	1	16-Jul-06
100041	1	15-Jul-06
100041	1	14-Jul-06
100041	1	13-Jul-06
100041	1	26-Jul-06
100041	1	25-Jul-06
100041	1	24-Jul-06
100041	1	23-Jul-06
100041	1	22-Jul-06
100041	1	21-Jul-06
100041	1	20-Jul-06
100041	1	2-Aug-06
100041	1	1-Aug-06
100041	1	31-Jul-06

Thank you

BRITISH COLUMBIA
CENTRE *for* EXCELLENCE
in HIV/AIDS

Providence
HEALTH CARE
How you want to be treated.

a place of mind
UBC