SAS1: Introduction to SAS

University of Guelph

Revised May 2011

Table of Contents

Data for SAS sessions

Dataset: Canadian Tobacco Use Monitoring Survey 2010 – Person File

This survey tracks changes in smoking status, especially for populations most at risk such as the 15- to 24-year-olds. It allows Health Canada to estimate smoking prevalence for the 15- to 24-year-old and the 25-and-older groups by province and by gender on a semi-annual basis.

The sample data used for this series of SAS workshops only includes respondents from the province of Quebec and only 14 of a possible 202 variables are being used.

To view the data, open the Excel spreadsheet entitled CTUMS_2010.xls

Variable Name	Label for Variable
PUMFID	Individual identification number
PROV	Province of the respondent
DVURBAN	Characteristic of the community
HHSIZE	Number of people in the household
HS_Q20	Number of people that smoke inside the house
DVAGE	Age of respondent
SEX	Respondent’s sex
DVMARST	Grouped marital status of respondent
PS_Q30	Age smoked first cigarette
PS_Q40	Age begin smoking cigarettes daily
WP_Q10A	Number of cigarettes smoked – Monday
WP_Q10B	Number of cigarettes smoked – Tuesday
WP_Q10C	Number of cigarettes smoked – Wednesday
WP_Q10D	Number of cigarettes smoked – Thursday
WP_Q10E	Number of cigarettes smoked – Friday
WP_Q10F	Number of cigarettes smoked – Saturday
WP_Q10G	Number of cigarettes smoked – Sunday
SC_Q100	What was the main reason you began to smoke again?
WTPP	Person weight (survey weight variable)

Variable PROV : Province of the respondent

Values	Categories
10	N.L.
11	P.E.I.
12	Nova Scotia
13	N.B.
24	Quebec
35	Ontario
46	Manitoba
47	Saskatchewan
48	Alberta
59	B.C.

Variable DVURBAN : Characteristic of community

Values	Categories
1	Urban
2	Rural
9	Not stated

Variable HHSIZE : # of people in the household

Values	Categories
1	
2	
3	
4	
5	5 or more

Variable DVMARST : Grouped marital status of respondent

Values	Categories
1	Common-law/Married
2	Widow/Divorced/Separated
3	Single
9	Not stated

Variable SC_Q100 : What was the main reason you began to smoke again?

Values	Categories
1	To control body weight
2	Stress, need to relax or to calm down
3	Boredom
4	Addiction / habit
5	Lack of support or information
6	Going out more (bars, parties)
7	Increased availability
8	No reason / felt like it
9	Family or friends smoke
10	Other
96	Valid skip
97	Don't know
98	Refusal

Variable PS_Q30 : Age smoked first cigarette
Variable PS_Q40 : Age begin smoking cigarettes daily
Variable HS_Q20 : # of people that smoke inside the home
Variable WP_Q10A : # of cigarettes smoked-Monday
Variable WP_Q10B : # of cigarettes smoked-Tuesday
Variable WP_Q10C : # of cigarettes smoked-Wednesday
Variable WP_Q10D : # of cigarettes smoked-Thursday
Variable WP_Q10E : # of cigarettes smoked-Friday
Variable WP_Q10F : # of cigarettes smoked-Saturday
Variable WP_Q10G : # of cigarettes smoked-Sunday

Values	Categories
96	Valid skip
97	Don't know
98	Refusal
99	Not stated

99	Not stated

SAS Availability

Faculty, staff and students at the University of Guelph may access SAS three different ways:

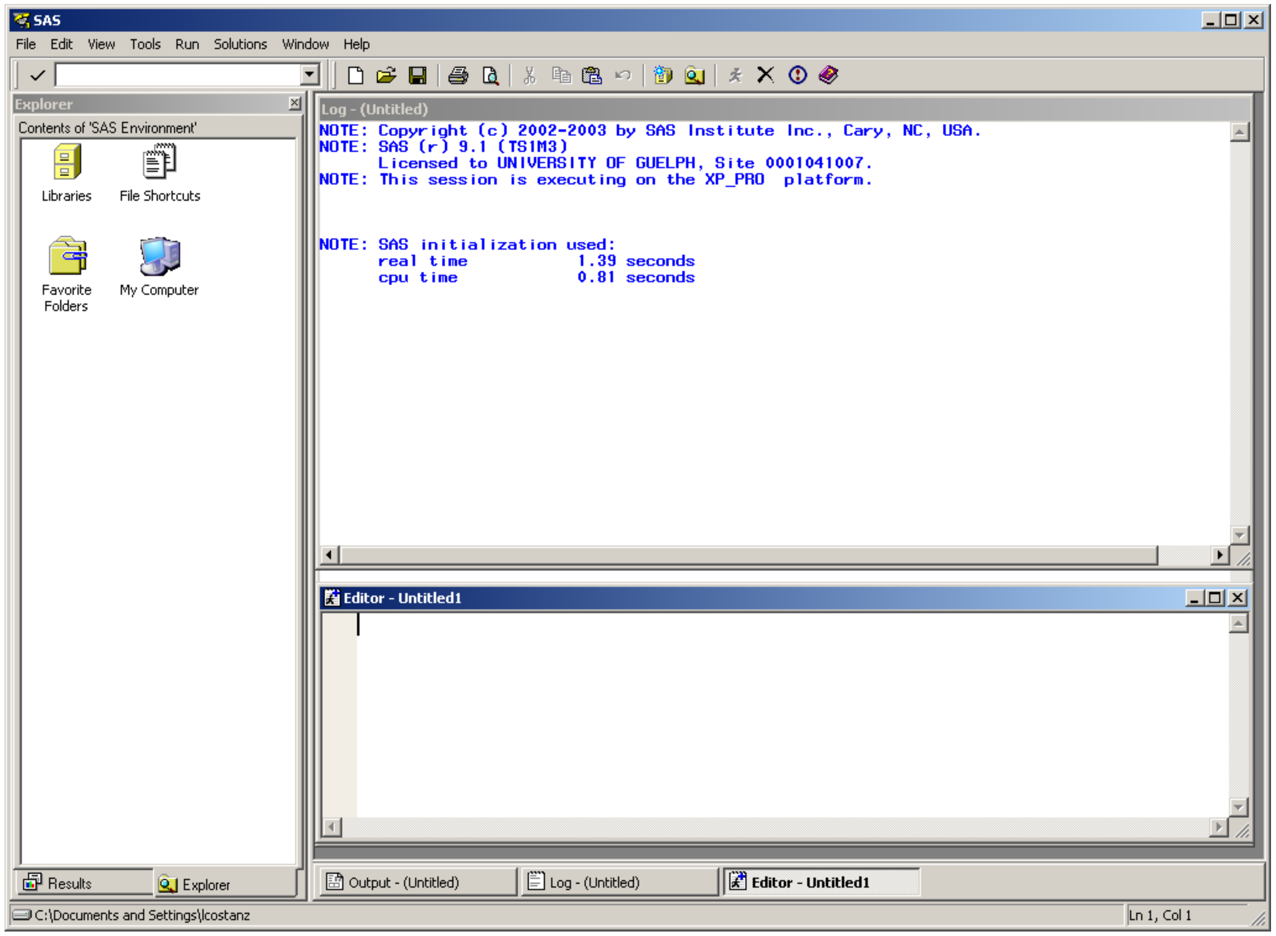
1. **Library computers**
   On the library computers, SAS is installed on all machines.

2. **Acquire a copy for your own computer**
   If you are faculty, staff or a student at the University of Guelph, you may obtain the site-licensed standalone copy of SAS at a cost. However, it may only be used while you are employed or a registered student at the University of Guelph. To obtain a copy, go to the CCS Software Distribution Site (http://www.uoguelph.ca/ccs/software).

3. **Central statistical computing server**
   SAS is available in batch mode on the UNIX servers (stats.uoguelph.ca) or through X-Windows.

SAS Windowing Environment

Within SAS, the windowing environment allows you to enter and run programs, view resulting output, access online help, and many other functions can be executed within these windows. To be precise, five main windows exist within SAS including the Explorer, Results, Program Editor, Log, and Output windows.



```
NOTE: Copyright (c) 2002-2003 by SAS Institute Inc., Cary, NC, USA.
NOTE: SAS (r) 9.1 (TS1M3)
      Licensed to UNIVERSITY OF GUELPH, Site 0001041007.
NOTE: This session is executing on the XP_PRO  platform.



NOTE: SAS initialization used:
      real time           1.39 seconds
      cpu time            0.81 seconds
```

Explorer Window

The Explorer window allows you to manage your files in the windowing environment. For example, the SAS Explorer allows you to view lists of your SAS files, create new SAS files, open any SAS file and view its contents. As well, it allows you to move, copy and delete files or libraries.

Program Editor Window

The Program Editor window enables you to enter, edit, submit and SAS programs.

Log Window

The Log window enables you to view messages about your SAS session and your SAS programs. If the program you submit has unexpected results, then the Log helps you identify the error. A PUT statement can be used to write program output to the Log.

Output Window

The Output window enables you to view listing output from your SAS programs. By default, the Output window is positioned behind the windows. When you created output, the Output window automatically moves to the front of your display.

Results Window

The Results window enables you to view output from a SAS program. Within the Results

SAS Language

A SAS program consists of a series of SAS statements which are used to define, read in, manipulate and analyze data. A typical SAS program is organized into three parts:

1. **Data Definition and Options**
   Defines the location of the data and set environment options

2. **DATA step**
   Reads, transforms, subsets and writes the data for the analysis

3. **Procedures**
   Performs an action on the data including sorting, computing means, running a regression and many more procedures are available

Data Definitions, Options and Titles

Data Definitions

Data definitions and options are specified in the first lines of a SAS program. Data definitions allow users to specify the location of the data. The filename statements point to specific files and libname statement points to directories. To read in the data stored as 'c:\sasfiles\data1.csv' on your local computer, the resulting data definition is:

```
libname folder 'c:\sasfiles';
filename 'data1.csv';
```

Options

An options statement is used to define an environment for the program. It changes the standard settings. Some common options include:

- **ls or linesize** – specifies the number of columns in the output window
- **ps or pagesize** – determines the number of lines on a page in the output window
- **date or nodate** – allows the date to either be included or not in the header of each page
- **obs** – limits the number of observations processed to allow for program testing on a small subset rather than reading in the entire data set
- **nocenter or center** – writes all the output in the log and listing files flush left or center

For example, to set a page in the output with 76 columns wide and 56 lines long, flush to the left with no date then a statement with all these options defined would appear as follows:

```
options ls=76 ps=56 nocenter nodate;
```

Titles

Titles allow for you to give descriptive headers at the top of each page in the Output window. The TITLE statement can be place anywhere in the SAS program. To define a title, the keyword "TITLE" begins the statement followed by a string of characters enclosed within single or double quotes. For example, if you would like to title a section as "Statistical Analysis for First Treatment" then the syntax would be:

```
title "Statistical Analysis for First Treatment";
```

Comments

Comments allow you to place strings of text to document the program and are ignored by SAS when the program is executed. There are two types of comments:

1. Comment line which use an asterisk (*) at the beginning of the line and a semi-colon (;) at the end.

```
*Transform annual salary to weekly salary;
```

2. Comment paragraph that is surrounded by a slash asterisk (/*) at the beginning and an asterisk slash (*/) at the end of the paragraph.

```
/* skip next statement
salary = salary * 12; */
```

In order to conduct any analysis in SAS, data must be converted into either a temporary or permanent SAS data set using a DATA step. If a temporary SAS data set is created, it will disappear once the SAS program is terminated. With a permanent SAS data, it is saved to disk and can be used each time the SAS program is started up. As well, the DATA step allows for the definition of variables, creation of new variables, merging of data sets, transformation of values, formatting and labelling of variables and assignment of missing values.

DATA Step

Guidelines Used to Specify SAS Statements

The following guidelines will be helpful to make the code more readable and maintainable not only for original programmer and for any other programmers. As well, it will facilitate troubleshooting done by others than other programmers.

In order to successfully run a SAS program, SAS statements must:

- begin with keyword which specifies the purpose of the SAS statement
- end with a semicolon (;)
- contain spaces between each separate item entered

In terms of formatting, SAS statements can:

- commence anywhere on the line
- begin on one line and continue onto the preceding lines, but you cannot split a word between two lines
- appear on the same line with other SAS statements
- spaces are not treated as a character

Tip!

To make it easier to troubleshoot your code, it is generally a good idea to use indentation and to place SAS statements on separate lines. Also, when SAS runs a program, it is case insensitive with respect to statements but, it distinguishes between upper and lowercase characters when it reads in data.

Reading Data Using CARDS/DATALINES

When either CARDS or DATALINES appear in a SAS program, the data for the analysis is part of the SAS program. In the following example, the DATA step creates a temporary SAS data set named exp1_data with 8 observations and six variables called id, sex, age, measure1, measure2, measure3, and measure4.

```
DATA exp1_data;
   INPUT id sex $ age measure1 measure2 measure3;
   DATALINES;
11  F  25 5 4 1
12  F  67 2 3 2
73  F  98 6 2 3
65  M  12 7 0 8
94  F  54 6 4 5
90  M  65 5 5 4
21  F  34 5 2 6
34  M  39 7 5 1
;
```

Tip!

In the input statement, $ is used to indicate alphanumeric variables. Keep in mind only numeric variables may be used in any analysis such as regressions. So, even if all of the values are numbers, if a variable is defined as character, you cannot use it for analysis.

Reading Data Using Column Input

In a data file, values can be entered in specific columns and the INPUT statement specifies the columns from which data value is to be read. When column input is used, you do not to code a dot (.) for numeric missing values, blanks will be interpreted as missing values. For example, using the data from above:

```
DATA exp1_data;
   INPUT id 1-3 sex $ 5 age 8-10 measure1 11-12 measure2 13-14 measure3 15-16;
   DATALINES;
11  F  25 5 4 1
12  F  67 2 3 2
73  F  98 6 2 3
65  M  12 7 0 8
94  F  54 6 4 5
90  M  65 5 5 4
21  F  34 5 2 6
34  M  39 7 5 1
;
```

Reading Raw Data

In order to read data saved to a text file, the INFILE statement must be used before the INPUT statement. The INFILE statement indicates the location of the text file on the computer. Here is an example:

```
DATA exp2_data;
   INFILE "C:\TestData.txt";
   INPUT id sex $ age measure1 measure2 measure3;
;
```

Here are some special cases.

Tab-Delimited File
If the text file contains delimiters or special characters indicating where the fields/variables are separated, then the delimiter option must be specified. For example, if the file TestData.txt contained delimiters which are tabs then the resulting code is:

```
DATA exp3_data;
   INFILE "C:\Documents and Settings\Desktop\IntroSAS\btemphrt.dat" delimiter='09'x;
   INPUT SUB_ID BODYTEMP BTEMPC TEMPCAT GENDER HRTRATE;
;
```

Comma Separated File
If the file TestData.txt contained delimiters which were commas (,) then the resulting code is:

```
DATA exp4_data;
   INFILE "C:\TestData.txt" delimiter=',’;
   INPUT id sex $ age measure1 measure2 measure3;
;
```

Variables or Comments at Top of Data File
If variable names or comments appear in the top lines then the firstobs option must indicate which line to skip to. For example, if variables names appear in the first line of the data file then the resulting code is:

```
DATA exp5_data;
   INFILE "C:\Documents and Settings\Desktop\IntroSAS\btemphrt.dat" delimiter='09'x firstobs=2;
   INPUT SUB_ID BODYTEMP BTEMPC TEMPCAT GENDER HRTRATE;
;
```

Reading a SAS Data Set

There may be situations where you may need to read from a permanent SAS data set to conduct your data analysis. This will require the use of the SET and LIBNAME statement. The SET statement refers to the filename of the permanent SAS data set and LIBNAME refers to the location of the SAS data set. In this example, the DATA step creates data set EXP2 by reading data from data set PERM.EXP.

```
libname PERM 'C:\Documents and Settings\Desktop\IntroSAS’;
DATA EXP2;
      set PERM.btemphrt;
RUN;
```

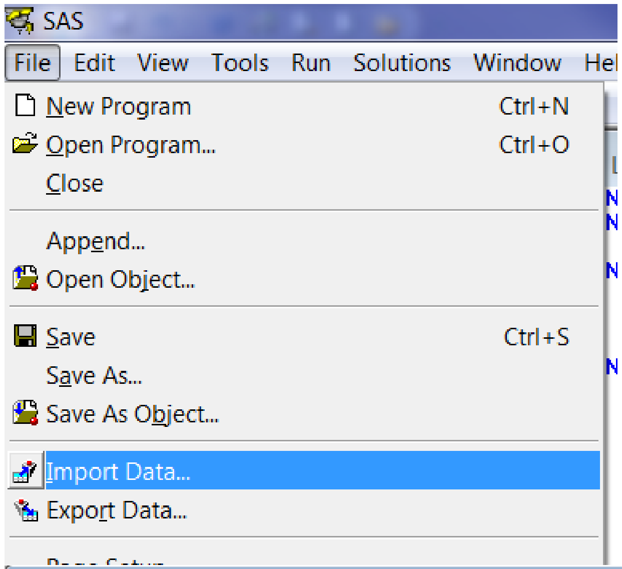
Importing an Excel file

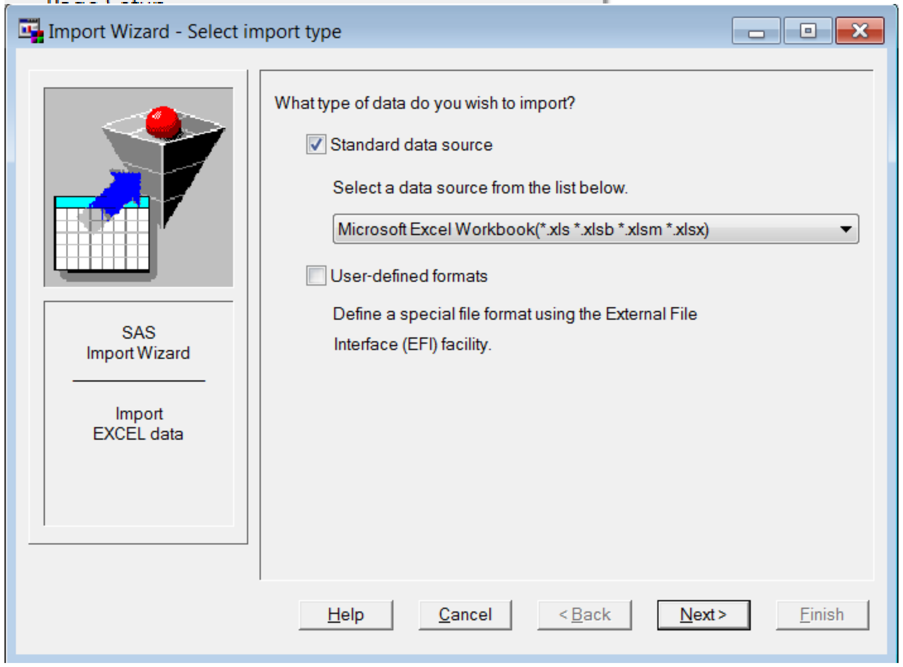
Most of us use Excel to enter our data and would like to be able to import our Excel data into SAS

Tip!

In order to cleanly import Excel data in SAS follow these guidelines:
Clear variable names in the first row of the Excel spreadsheet – follow standard SAS naming conventions. Data starts in the second row. Do NOT leave a blank row between the variable names (column headings) and your data. SAS interprets this as a character or string and all your data will be imported as strings and NOT numbers. Remove all formatting from your Excel file. If you’ve created a worksheet with formatting and formulae – copy your data onto a new worksheet and use this for your SAS.

To import the Excel File – in SAS select File from the top menu then select Import Data





What type of data do you wish to import?

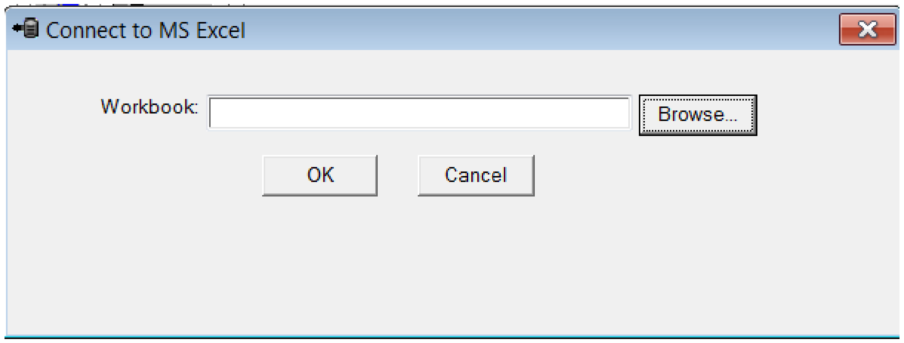
[✓] Standard data source

Select a data source from the list below.

Microsoft Excel Workbook(*.xls *.xlsb *.xlsm *.xlsx)

[ ] User-defined formats

Define a special file format using the External File Interface (EFI) facility.

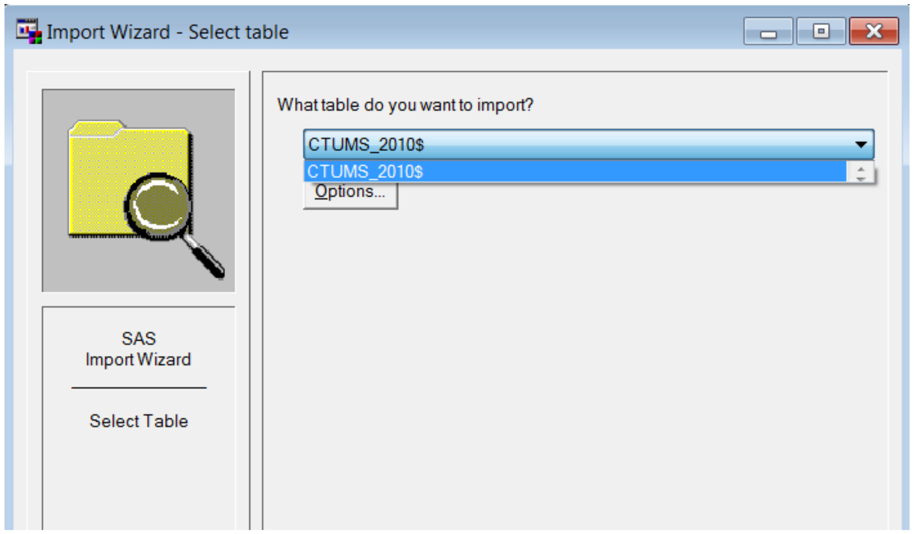


Tip!

Excel 2007 and 2010 file endings are .xlsx – in SAS once you browse to find your file – please be sure to select File Types – all or the *.xlsx ending.

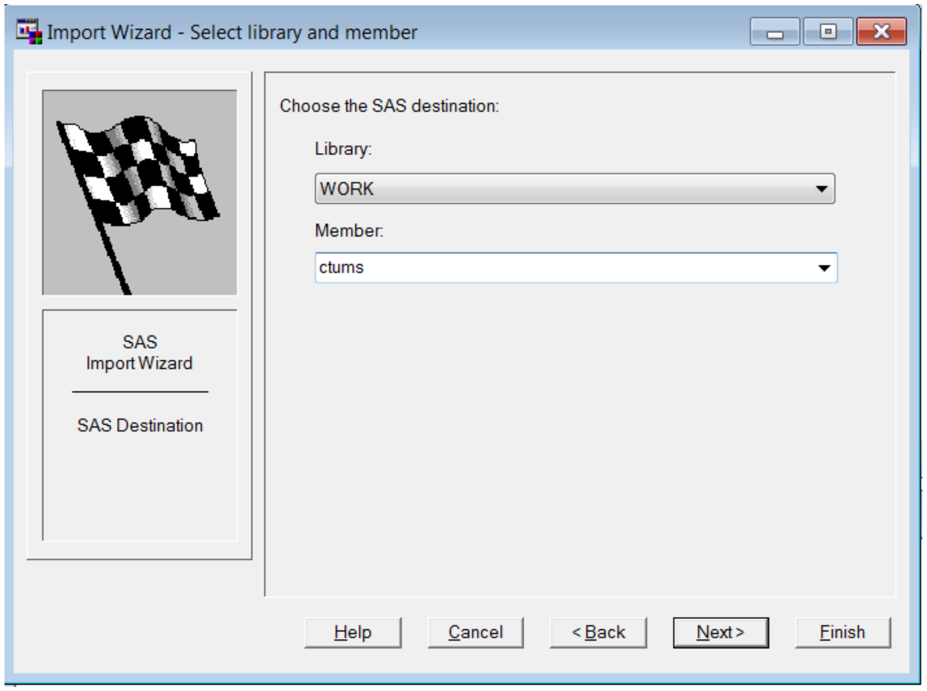
Tip!

You can only import one worksheet at a time. Be sure to select the correct one.



What table do you want to import?

CTUMS_2010$



Choose the SAS destination:

Library: WORK

Member: ctums

Tip!

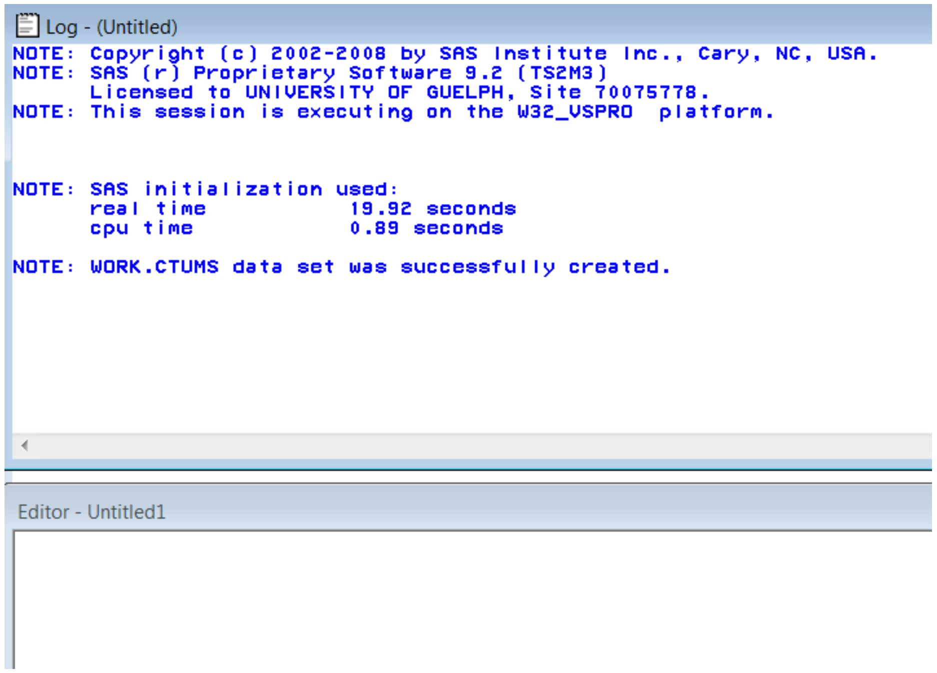
In the Member: box – please type the name of the dataset you are creating. Remember SAS naming conventions are still in effect!

Also note that you have a Finish Flag – click on Finish to import the data.

If you will be importing several files – you can move to the Next step and save the program SAS creates to import a file.

Tip!

This is the correct result! You will see NO data only a note that states that the named dataset has been created.



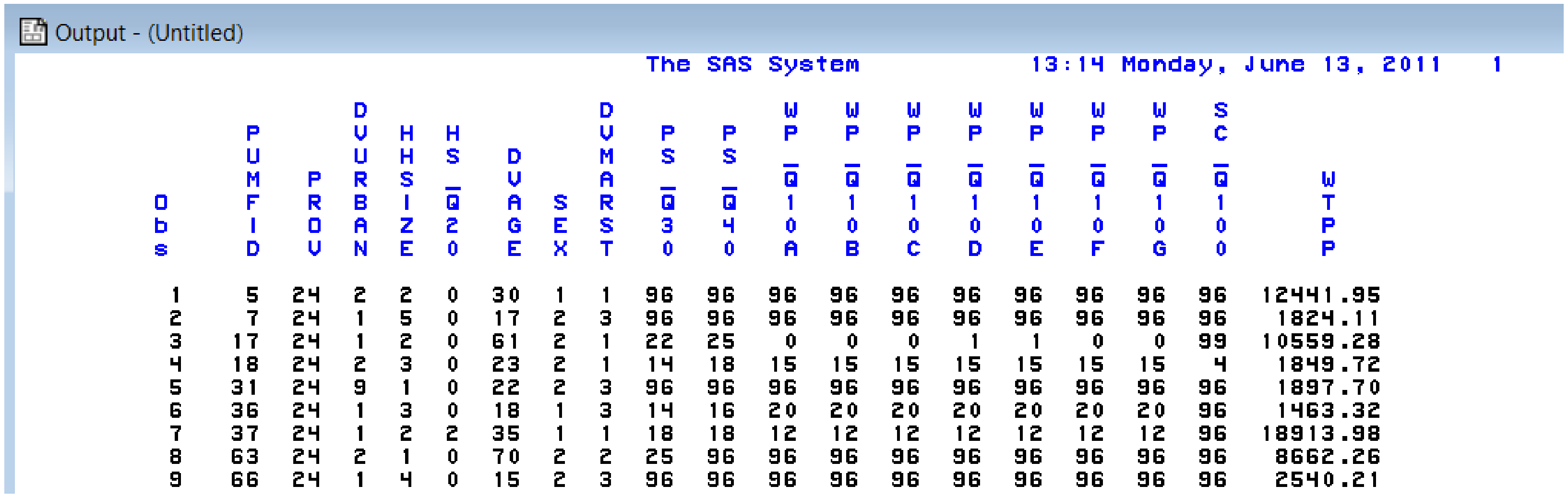
```
NOTE: Copyright (c) 2002-2008 by SAS Institute Inc., Cary, NC, USA.
NOTE: SAS (r) Proprietary Software 9.2 (TS2M3)
      Licensed to UNIVERSITY OF GUELPH, Site 70075778.
NOTE: This session is executing on the W32_VSPRO  platform.



NOTE: SAS initialization used:
      real time           19.92 seconds
      cpu time            0.89 seconds

NOTE: WORK.CTUMS data set was successfully created.
```

PROC PRINT

This procedure either prints to screen all the observations or a subset of a specified SAS data set. The general syntax is as follows:

```
PROC PRINT data=dataset;
RUN;
```

```
Proc print data= ctums;
Run;
```



```
                                          The SAS System           13:14 Monday, June 13, 2011   1

                    D                        D              W    W    W    W    W    W    W    S
          P         V    H    H              V    P    P    P    P    P    P    P    P    P    C
          U         U    H    S    D         M    S    S    _    _    _    _    _    _    _    _
          M    P    R    S    _    V         A    _    _    Q    Q    Q    Q    Q    Q    Q    Q          W
    O     F    R    B    I    Q    A    S    R    Q    Q    1    1    1    1    1    1    1    1          T
    b     I    O    A    Z    2    G    E    S    3    4    0    0    0    0    0    0    0    0          P
    s     D    V    N    E    0    E    X    T    0    0    A    B    C    D    E    F    G    0          P

    1     5   24    2    2    0   30    1    1   96   96   96   96   96   96   96   96   96   96   12441.95
    2     7   24    1    5    0   17    2    3   96   96   96   96   96   96   96   96   96   96    1824.11
    3    17   24    1    2    0   61    2    1   22   25    0    0    0    1    1    0    0   99   10559.28
    4    18   24    2    3    0   23    2    1   14   18   15   15   15   15   15   15   15    4    1849.72
    5    31   24    9    1    0   22    2    3   96   96   96   96   96   96   96   96   96   96    1897.70
    6    36   24    1    3    0   18    1    3   14   16   20   20   20   20   20   20   20   96    1463.32
    7    37   24    1    2    2   35    1    1   18   18   12   12   12   12   12   12   12   96   18913.98
    8    63   24    2    1    0   70    2    2   25   96   96   96   96   96   96   96   96   96    8662.26
    9    66   24    1    4    0   15    2    3   96   96   96   96   96   96   96   96   96   96    2540.21
```

Limiting observations when printing:
The following example specifies within PROC PRINT to only display the first 50 observations in the data set. The obs keyword specifies the last observation to display.

```
PROC PRINT data=ctums(obs=50);
RUN;
```

To print a subset of data to screen, specify the first observation by using firstobs keyword and the last observation with the obs keyword. That is if you wish to output to screen observations 10 to 43, the code would be as follows:

```
PROC PRINT data=ctums(firstobs = 20 obs = 50);
RUN;
```

Adding Formats

PROC FORMAT

This procedure allows us to add labels to our values. We've already seen that in order to understand what the numbers in our dataset represent we need a "translation" – something, a piece of paper or a codebook, to inform us what the 1, 2, or 3s in a givin variable represent. To do this in SAS is a 2 step process.

Step1:

Create a format (think of it as a box that holds information) that contains the label and its value – like the table below

Values	Categories
10	N.L.
11	P.E.I.
12	Nova Scotia
13	N.B.
24	Quebec
35	Ontario
46	Manitoba
47	Saskatchewan
48	Alberta
59	B.C.

```
Proc format;
  value prov
      10    ="N.L."
      11    ="P.E.I."
      12    ="Nova Scotia"
      13    ="N.B."
      24    ="Quebec"
      35    ="Ontario"
      46    ="Manitoba"
      47    ="Saskatchewan"
      48    ="Alberta"
      59    ="B.C.";
Run;
```

Step 2:

Now we need to link the formats with the correct variable. This is done in a Data Step. We will rename our dataset ctums2

```
Data ctums2;
  set ctums;

  format
      PROV prov.
      DVURBAN urban.
      SEX sex.
      HHSIZE hhsize.
      HS_Q20 extracode.
      DVMARST dvmarst.
      PS_Q30 extracode.
      PS_Q40 extracode.
      WP_Q10A extracode.
      WP_Q10B extracode.
      WP_Q10C extracode.
      WP_Q10D extracode.
      WP_Q10E extracode.
      WP_Q10F extracode.
      WP_Q10G extracode.
      SC_Q100 extracode.;
Run;
```

Can you see what Proc Format does?

Remember to view you data you need to use PROC PRINT.

```
Proc print data=ctums2;
Run;
```

```
                                    The SAS System          13:14 Monday, June 13, 2011   23

  Obs  PUMFID   PROV   DVURBAN      HHSIZE     HS_Q20  DVAGE   SEX     DVMARST

    1       5   Quebec  Rural        2 people      0      30   Male    Common-law/Married
    2       7   Quebec  Urban        5 or more     0      17   Female  Single
    3      17   Quebec  Urban        2 people      0      61   Female  Common-law/Married
    4      18   Quebec  Rural        3 people      0      23   Female  Common-law/Married
    5      31   Quebec  Not stated   1 person      0      22   Female  Single
    6      36   Quebec  Urban        3 people      0      18   Male    Single
    7      37   Quebec  Urban        2 people      2      35   Male    Common-law/Married
    8      63   Quebec  Rural        1 person      0      70   Female  Widow/Divorced/Separated
    9      66   Quebec  Urban        4 people      0      15   Female  Single
   10      74   Quebec  Urban        2 people      0      72   Male    Common-law/Married
```

Transforming Data (Selecting and Modifying the Data)

Assignment Statements

Assignment statements enable the programmer to create new variable or change the values of existing variables. This is done within a DATA step where the variable name is to the left of the equals sign and the value/expression to the right.

variable = expression

Create a new variable

In this dataset we have a variable that tells us how many cigarettes the respondents smoked each day of a particular week. We will now create a new variable that calculates the total number of cigarettes the respondents smoked that week.

In the Data step we will add the following code:

```
  totcig = WP_Q10A + WP_Q10B + WP_Q10C + WP_Q10D + WP_Q10E + WP_Q10F + WP_Q10G;
```

Try a Proc Print – What's wrong????

IF statements

Generally, the IF statement is used to generate a subset of a larger data set by selecting cases based on certain conditions. Alternatively, the IF statement can be used to delete a subset of data. This is the syntax for the IF statement:

IF <*expression*> THEN <*statement*> [ELSE <*statement*>]

We need to remove all the Valid Skip data before we calculate the total. To do this we can use an If... Then statement

What is the code for Valid Skip?

We need to add this statement to our Data step. Do we need any additional statements?

```
  if WP_Q10A = 96 then delete;
```

Try a Proc PRINT to evaluate your results.

EXERCISE

You've been asked to recode the age variable into the following groups:

Group	Ages
1	15-24
2	25-34
3	35-44
4	45-54
5	55-64
6	65-74
7	75-84
8	85 and over

1. Create a new variable called agegroup
2. Create value labels for the new variable
3. Print the age and agegroup variable to see whether you were successful.

WHERE Statements

The WHERE statement allows researchers to selects observations which, meet certain conditions. The specified condition is an arithmetic or logical expression that generally consists of a sequence of operands and operators. This example will create a dataset with only females aged 15-44 yrs of age.

```
DATA ctumsFemales;
   SET ctums2;
   WHERE sex = 1 and agegroup <4;
RUN;
```

Here is a summary of symbol abbreviations for use with IF and WHERE statements:

Operator Symbol	Abbreviation	Purpose
<, <=	LT, LE	Less than and less than and equal to
>, >=	GT, GE	Greater than and greater than and equal to
=, ^=	EQ, NE	Equal to, and not equal to

Tip!

The WHERE statement is used before the data enters the input buffer and the IF statement is applied after the data enters the program.

Procedures

A group of SAS procedure statements is called a PROC step. SAS procedures analyze data in SAS data sets to produce statistics, tables, reports, charts, and plots, to create SQL queries, and to perform other analyses and operations on your data. SAS procedures also give you ways to manage and print SAS files.

PROC FREQ

PROC FREQ will produce a frequency table for each listed variable. In general, the format for this procedure is:

```
PROC FREQ data=dataset;
        TABLES variable-list;
RUN;
```

If the **tables** keyword is not specified, an large amount of output will be produced. Generally, **PROC FREQ** is best suited for categorical (nominal or discrete) variables. For either interval or ratio variables, it is best to use **PROC UNIVARIATE**.

```
proc freq data=ctums2;
  tables sex;
Run;
```

```
                              The SAS System              13:14 Monday, June 13, 2011  158

                            The FREQ Procedure

                                   SEX

                                            Cumulative     Cumulative
   SEX       Frequency       Percent        Frequency        Percent
 ---------------------------------------------------------------------
 Male               90         50.85               90          50.85
 Female             87         49.15              177         100.00
```

Crosstabulations
Proc freq will produce a crosstable for each pairing of variables listed after the TABLES keyword. The format of this procedure is:

```
PROC FREQ data=dataset;
        TABLES row-variable*column-variable;
RUN;
```

Example:

```
Proc freq data=ctums2;
  tables sex*SC_Q100f;
Run;
```

Output:

```
                          The SAS System          13:14 Monday, June 13, 2011 160

                        The FREQ Procedure

                      Table of SEX by SC_Q100

SEX(SEX)      SC_Q100(SC_Q100)

Frequency|
Percent  |
Row Pct  |
Col Pct  |       1|       2|       4|       5|       6|       8|  Total
---------+--------+--------+--------+--------+--------+--------+
Male     |      0 |      9 |      8 |      0 |      4 |      5 |     90
         |   0.00 |   5.08 |   4.52 |   0.00 |   2.26 |   2.82 |  50.85
         |   0.00 |  10.00 |   8.89 |   0.00 |   4.44 |   5.56 |
         |   0.00 |  34.62 |  44.44 |   0.00 | 100.00 | 100.00 |
---------+--------+--------+--------+--------+--------+--------+
Female   |      1 |     17 |     10 |      1 |      0 |      0 |     87
         |   0.56 |   9.60 |   5.65 |   0.56 |   0.00 |   0.00 |  49.15
         |   1.15 |  19.54 |  11.49 |   1.15 |   0.00 |   0.00 |
         | 100.00 |  65.38 |  55.56 | 100.00 |   0.00 |   0.00 |
---------+--------+--------+--------+--------+--------+--------+
Total           1       26       18        1        4        5      177
             0.56    14.69    10.17     0.56     2.26     2.82   100.00
(Continued)

                      Table of SEX by SC_Q100
```

PROC MEANS

PROC MEANS procedure provides data summarization tools to compute descriptive statistics for variables across all observation and within groups of observations. For example, PROC MEANS:

- calculates descriptive statistics based on moments
- estimates quantiles, which includes the median
- calculates confidence limits for the mean
- identifies extreme values
- performs a t test

The format of this procedure is:

```
PROC MEANS data=dataset;
        VAR variables;
RUN;
```

Example:

```
Proc means data=ctums2;
  var totcig;
Run;
```

Output:

```
                         The SAS System           13:14 Monday, June 13, 2011 163

                       The MEANS Procedure

                  Analysis Variable : totcig

  N            Mean         Std Dev         Minimum         Maximum
 ------------------------------------------------------------------
177     107.6949153     137.0765282               0     691.0000000
 ------------------------------------------------------------------
```

If you want Standard Error of the mean you need to add this to your Proc means statement. For example in this dataset I would use:

```
Proc means data=ctums2 mean stderr min max;
  var totcig;
Run;
```

```
                         The SAS System          13:14 Monday, June 13, 2011 164

                       The MEANS Procedure

                   Analysis Variable : totcig

          Mean        Std Error         Minimum          Maximum
    ----------------------------------------------------------------
   107.6949153       10.3033028               0      691.0000000
    ----------------------------------------------------------------
```

ODS – Output Delivery System

ods – Output Delivery System – a way to select particular pieces of the output window and display them in Html, rdf, pdf, etc…

ods select – to select sections of the SAS output to display in the output window.

To determine what the names of the output tables are use the following code:

```
ods trace on;
Proc univariate data=t_ctums normal plot;
  var log_totcig;
Run;
ods trace off;
```

Copied from Log Window:

```
569  ods trace on;
570  Proc univariate data=t_ctums normal plot;
571    var log_totcig;
572  Run;


Output Added:
-------------
Name:       Moments
Label:      Moments
Template:   base.univariate.Moments
Path:       Univariate.log_totcig.Moments
-------------

Output Added:
-------------
Name:       BasicMeasures
Label:      Basic Measures of Location and Variability
Template:   base.univariate.Measures
Path:       Univariate.log_totcig.BasicMeasures
--------------------------

Output Added:
-------------
Name:       TestsForLocation
Label:      Tests For Location
Template:   base.univariate.Location
```

```
Path:       Univariate.log_totcig.TestsForLocation
-------------

Output Added:
-------------
Name:       TestsForNormality
Label:      Tests For Normality
Template:   base.univariate.Normal
Path:       Univariate.log_totcig.TestsForNormality
-------------
```

If I only want to see the Test for Normality section and none of the Descriptive statistics then I would use the following code:

```
ods select TestsForNormality;
Proc univariate data=t_ctums normal plot;
  var log_totcig;
Run;
```

SAS Output:

```
                                The UNIVARIATE Procedure
                                  Variable:  log_totcig

                                  Tests for Normality

                  Test                  --Statistic---    -----p Value------

                  Shapiro-Wilk          W     0.899483    Pr < W     <0.0001
                  Kolmogorov-Smirnov    D     0.158385    Pr > D     <0.0100
                  Cramer-von Mises      W-Sq  1.180171    Pr > W-Sq  <0.0050
                  Anderson-Darling      A-Sq  6.596734    Pr > A-Sq  <0.0050
```

ODS can also create HTML, RTF, and PDF files of your output window.

Let's create a PDF document of the **UNIVARIATE** output.

SAS Code:

```
ods listing close;
ods pdf file ='C:\Documents and Settings\edwardsm.CFS\My Documents\Teaching\GSLI\SAS_workshops\univ.pdf';

Proc univariate data=cchs2005 normal plot;
  var weight;
Run;

ods pdf close;
ods listing;
```

ods listing close – stops the output from going to the Output window
ods pdf file – tells SAS that you are redirecting the output to a pdf file.
='C:\Documents and Settings\edwardsm.CFS\My Documents\Teaching\GSLI\SAS_workshops\univ.pdf ' – tells SAS the name of the file you want to create and where to save it.

Once you've run your Procedure you need to close the ODS system.

ods pdf close – finished creating PDF files
ods listing – redirecting the SAS output back to the Output window. If you forget to do this – you can run the Procedure but there will be no output created.

SAS/GRAPH

SAS has several ways of creating Graphes – however the SAS/GRAPH procedures are now the preferred method. **Proc PLOT** will provide you with a basic plot. Here is an example of **Proc GPLOT** to show you the options.

Available **Proc**edures in SAS/GRAPH:

Proc GCHART – used to create bar charts (vertical or horizontal, pie charts, 3-D pie charts and donut charts
Proc GPLOT – two-dimensional scatter plots, simple line plots, regression plots, high-low plots, bubble plots
Proc G3D – surface plots, scatter plots
Proc GCONTOUR – contour plots
Proc GMAP – 2 or 3-dimensional maps, block maps, choropleth maps, prism maps, surface maps

Example: Mean total cigarettes smoked for Age Groups by Gender

We need to create a dataset with the mean weight for each age_group and gender combination. To accomplish this we will use **Proc MEANS.**

SAS Code:

```
Proc means data=newctums nway;
  class agegroup sex;
  var totcig;
  output out=ctumsplot mean=mtotcig;
Run;
```

Proc means … nway – keep only the age_group gender combinations. Without the **nway** option the resulting datafile will hold overall means for males, females, then each age group, and finally the age_group*gender combinations.

Class – tells SAS by what variables we would like the means calculated.

Var – which variable to calculate the means

Output – tells SAS we want to save the results in a new dataset.
  out= gives the name of the new dataset – ‘cchsplot’ in this example
  mean= we only want to save the means in the dataset and we want to call this variable ‘mweight’

```
Proc print data=cchsplot;
```

```
Run;
```

SAS Output:

```
                    Obs     agegroup       SEX       _TYPE_     _FREQ_     mtotcig

                      1    15-24 years    Male          3          47       88.128
                      2    15-24 years    Female        3          48      112.979
                      3    25-34 years    Male          3           9      101.000
                      4    25-34 years    Female        3           6       92.667
                      5    35-44 years    Male          3           7      143.000
                      6    35-44 years    Female        3          13      115.538
                      7    45-54 years    Male          3          10      120.000
                      8    45-54 years    Female        3           8      102.000
                      9    55-64 years    Male          3          13      139.846
                     10    55-64 years    Female        3          11       96.545
                     11    65-74 years    Male          3           3      141.000
                     12    65-74 years    Female        3           1      175.000
```

Now we're ready to proceed to creating our graph.

```
goptions reset=global gunit=pct border cback=white
         colors=(black blue green red)
         ftitle=swissb ftext=swiss htitle=6 htext=3;
```

goptions statement – sets the environment for your output device

- reset – resetting any prior settings
- gunit - Specifies the default unit of measure to use with height specifications. – PCT= percentage of the graphics output area
- border – create a border around the output window
- cback – background colour
- colours – colours that will be used in the output
- ftitle and ftext – font to be used in the title and the text
- htitle and htext – title and text height

```
title1 'Average number of cigarettes smoked by Age Group';
footnote1 j=l ' Source: Canadian Tobacco Use Monitoring Survey - 2010 Person File';

symbol1 interpol=join
        value=dot
        height=3;
```

Setting up the environment for our output – be it a graph or a slide.

Title and **footnote –** self explanatory

Symbol – This is where we can set out how we want the dots to look on our graph. In this case we've chosen a dot for each graph value, we've set the height to 3 units, and we want SAS to join the dots.

```
proc gplot data=ctumsplot;
   plot mtotcig*agegroup=sex / hminor=0;
run;
```

Proc gplot – using the gplot procedure and plotting mean_weight by age_group for each gender.

Hminor – specifies the number of minor ticks we would like to see on the H-axis

```
symbol1 color=green interpol=spline
        width=2 value=triangle
        height=3;
symbol2 color=blue interpol=spline
        width=2 value=circle
        height=3;
```

Changing the symbols for each gender. We can change the colour, the size (height and width), and the method of joining the dots.

```
axis1 label=('Age Group')
       offset=(0)
      width=3;
axis2 label=('Mean number of Cigarettes smoked in a week' justify=right 'Number')
      order=(80 to 200 by 10)
      width=3;
```

Now we are customizing the axes. We create axis1 and we later tell SAS whether it should be used as the horizontal axis (Y-axis) or as the vertical axis (X-axis). We are adding labels to each value – making it easier to read.

```
proc gplot data=ctumsplot;
   plot mtotcig*agegroup=sex / haxis=axis1 hminor=0
                               vaxis=axis2 vminor=1
                               caxis=red;
run;
quit;
```

Options of the plot statement allow us to specify which axis is which along with minor ticks, the colour of the axes (caxis=red) and whether a legend is included or not.

These are the basics to using SAS/GRAPH – there are many more options and procedures available. With many of the new Stats procedures, SAS is developing accompanying graphs.