

# Getting Started with the SAS System

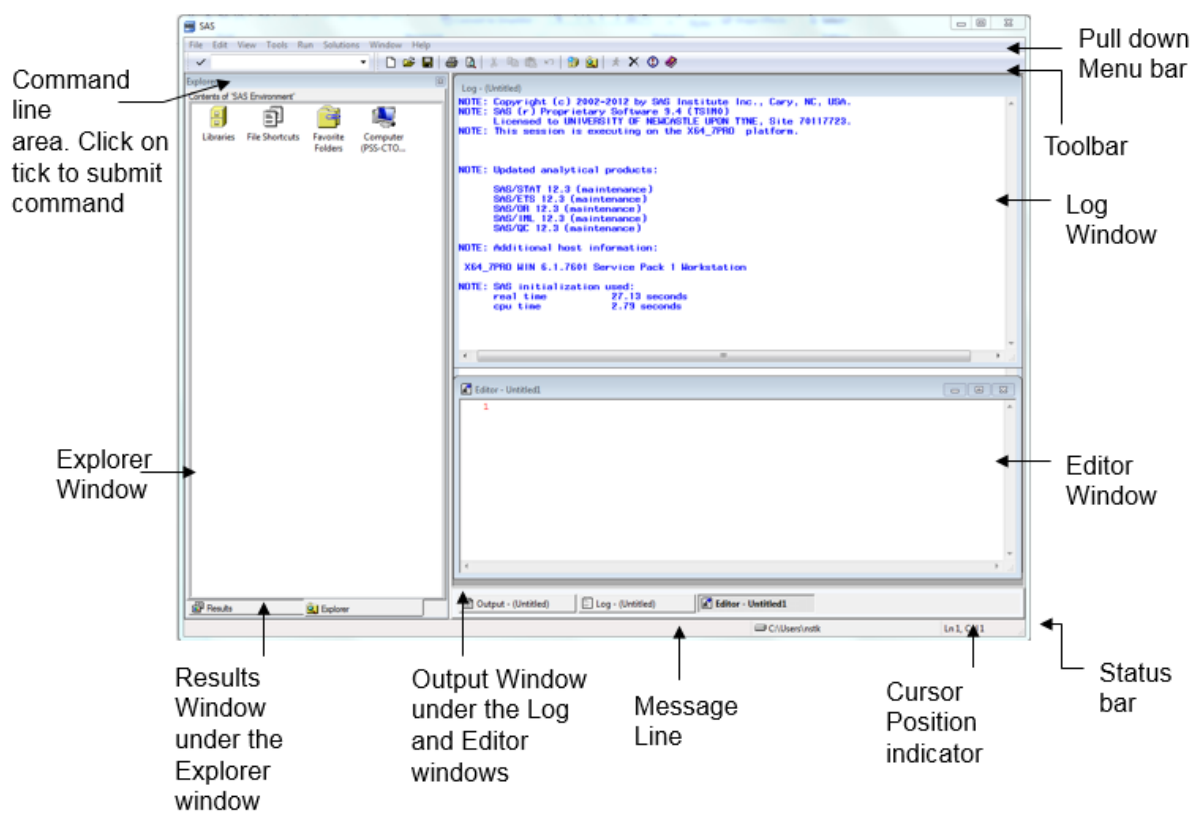
## Point and Click Approach

You are advised to use this document alongside the program.

### Starting SAS

To start SAS click on **Start -> Programs -> SAS -> SAS 9.4 (English)**.

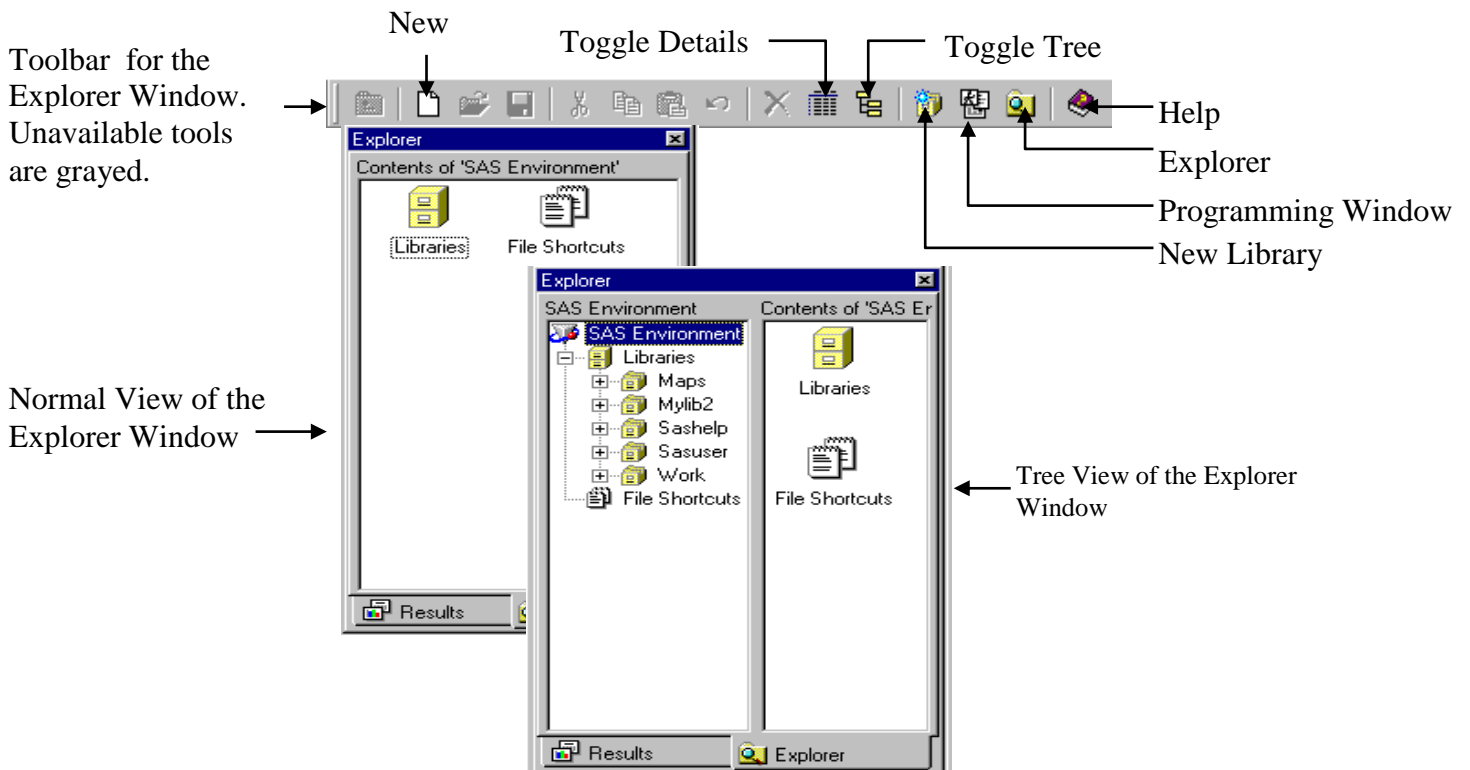
The SAS software has five main windows: *Explorer*, *Results*, *Editor (Program)*, *Log* and *Output* Windows. Windows may overlay one another. See Fig.1 below.



**Fig. 1 The SAS Windows**

### Explorer window

Use this window for file management tasks such as moving, copying and deleting files. In this window, you can also create new libraries and SAS files. You can display the *Explorer* window with or without a tree view of its contents. That is, **View -> Show Tree**. You can also use the Toggle Tree on the toolbar to do this. See Fig. 2. When copying files from one library to another it is best to use the tree view of the Explorer window.

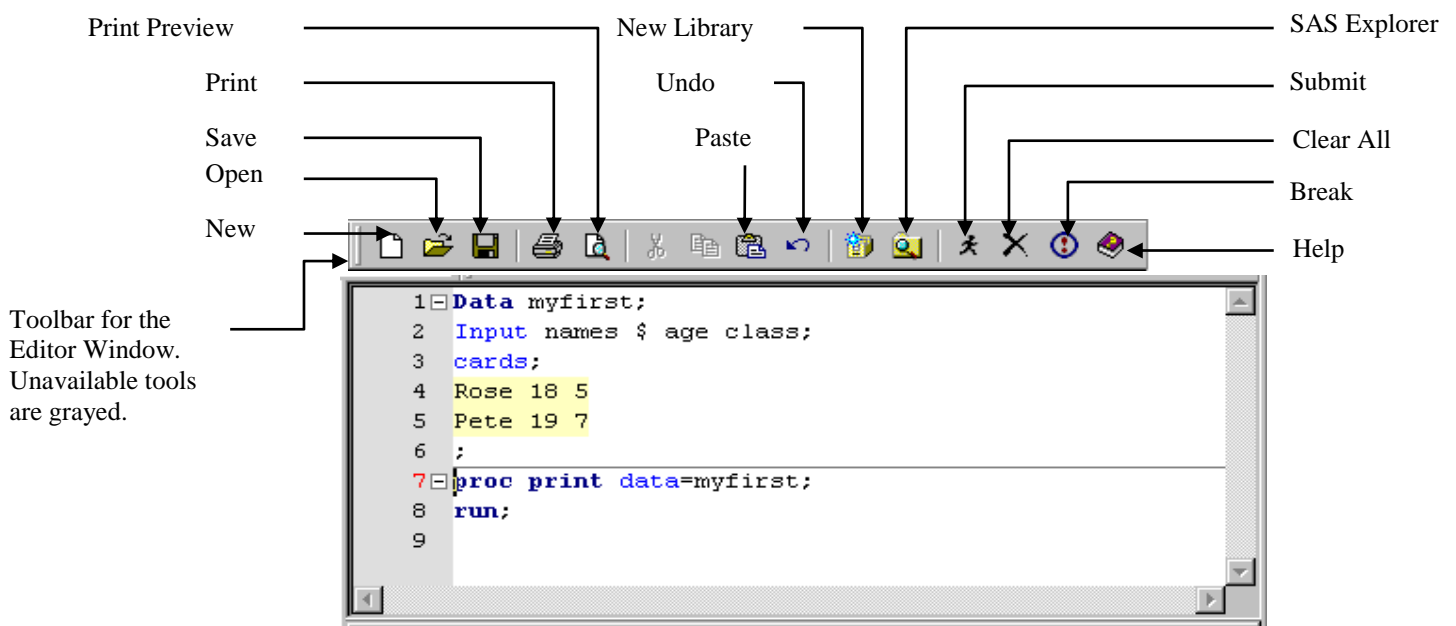


**Fig. 2 The Explorer Window**

**Program Editor Window**

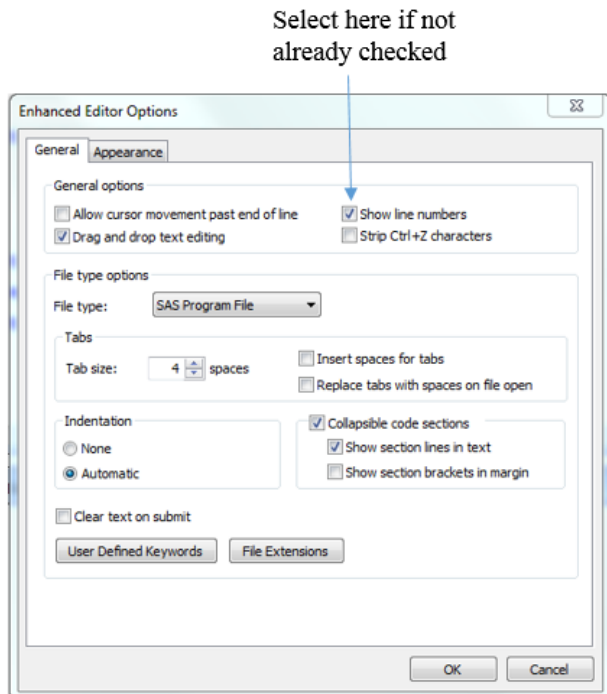
In this window, you can enter, edit, and submit SAS programs. You can use more than one *Editor* window. The main features of this window include:

- colour coding and syntax checking of SAS language
- expandable and collapsible sections, click on the (+) to expand and (-) to collapse
- multi-level *undo* and *redo*. See Fig. 3 for details.



**Fig. 3 Program Editor Window**

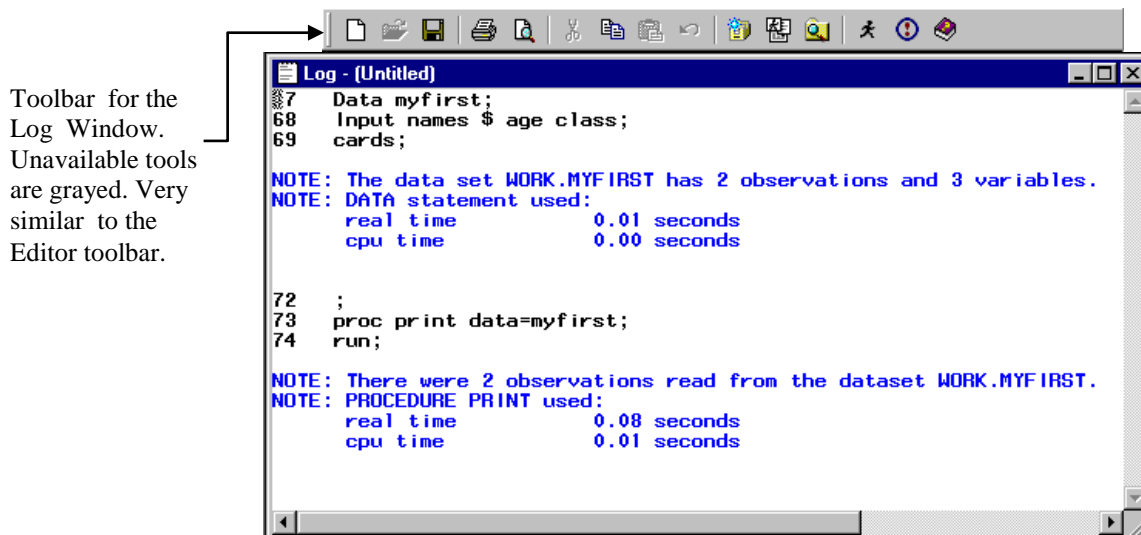
If the row numbers are not displayed on the Editor window then follow these instruction to display them. Make sure the Editor window is the active window by clicking anywhere on the Editor window. From the menu bar select **Tools > Options > Enhanced Editor** and the dialogue box below will be display. If **Show line numbers** is not selected then select it.



**Fig 3.1 Enhanced Editor Options dialogue box**

### Log window

This window display important information about your SAS session and programs you submit. Always examine the log window before you look at your output for any error or warning messages.



**Fig. 4 Log Window**

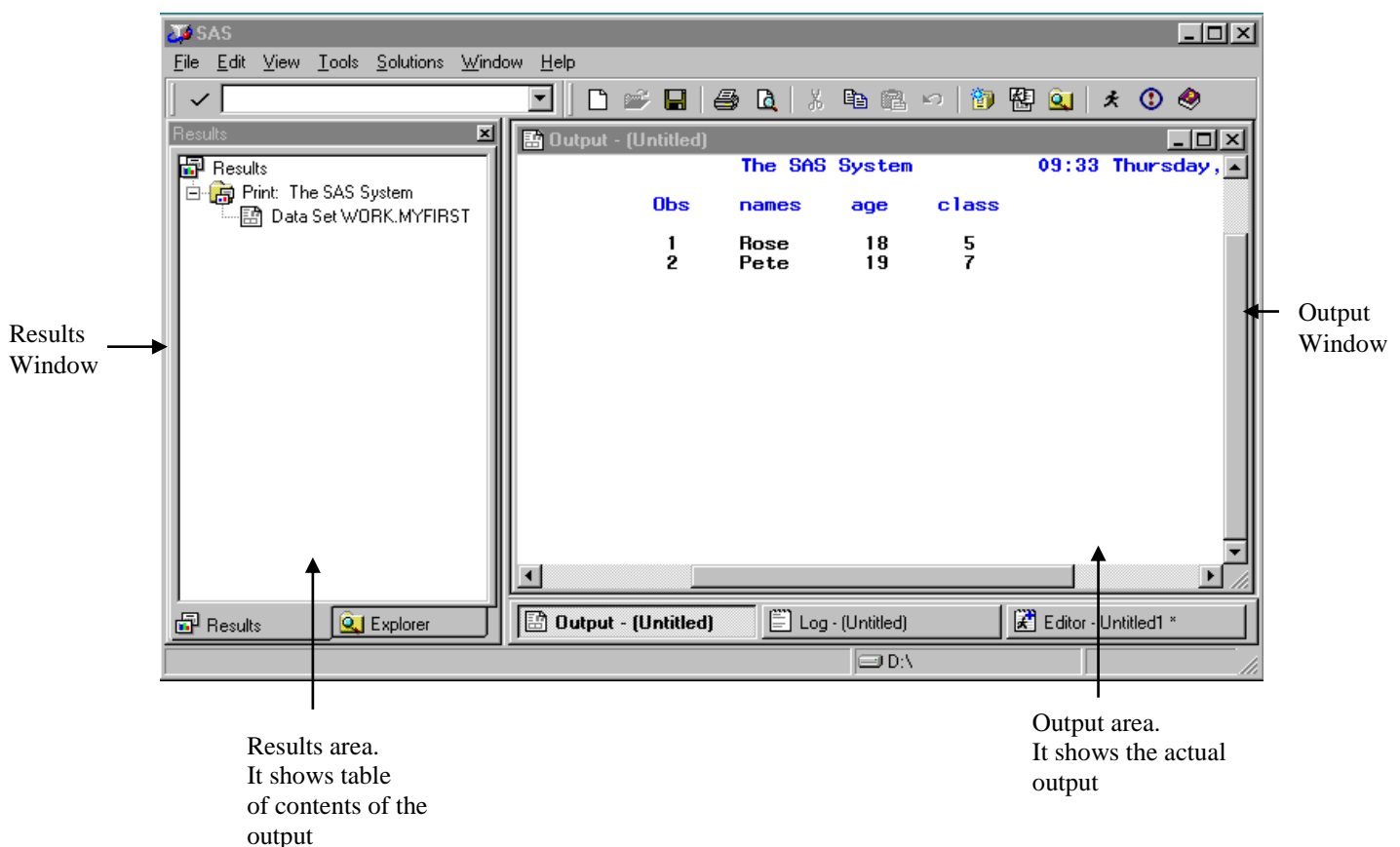
## Output Window

You can browse output from SAS programs that you submit in this window. By default, the window is positioned behind the *Editor* and *Log* windows. The window automatically moves to the front for your first output in your SAS session. For subsequent output in the same session, you will need to select the output button to see the output.

## Results Window

This window helps you to navigate and manage output from SAS programs that you submit. You can view, save and print individual items of output. By default, the *Results* window is positioned behind the *Explorer* window and it is empty until you submit a program that creates output. Then it moves to the front of your display.

The output and results window are shown together in Fig. 5 below.



**Fig. 5 Output and Results Windows**

## Using SAS window features

The active window determines what items are available on the main menu bar. In other words, each window has its own menu selections that reflect the actions you can perform in the window. For example, click the *Explorer* window and then select **View**. Examine the options available. Click the *Editor* window and select **View** again. Examine the options available and notice that it offers different selections.

Note:

- you can minimize the *Log*, *Editor* and *Output* windows
- right-click in any window to see the pop-up menus
- toolbar displays icons for many of the actions you perform most often in a particular window. Select the various Windows and examine how the toolbar changes.

Move your mouse to a tool and hold it there for a moment to see the name of the tool.

## Getting Help in the SAS System

Like most software, help is available for all products in the SAS System. To appreciate the extensive help provided by the SAS System, try the following:

- Select **Help**. Take your time and examine the following:
  - Using This Window
  - SAS Help and Documentation
  - Getting Started with the SAS Software
  - Learning SAS Programming
  - SAS on the Web
- Make the *Explorer* window active, then select **Help -> Using This Window**. Take a few moments and examine the task-oriented help available.

### SAS Products


Base SAS, Enterprise Miner, SAS Enterprise Guide, SAS/ACCESS, SAS/AF, SAS/ASSIST, SAS/CALC, SAS/CONNECT, SAS/EIS, SAS/ETS, SAS/FSP, SAS/GIS, SAS/GRAPH, SAS/IML, SAS/INSIGHT, SAS/IntrNet, SAS/LAB, SAS/MDDDB, SAS/OR, SAS/QC, SAS/SHARE, SAS/SPECTRAVIEW/ SAS/STAT, SAS/TOOLKIT, SAS/TUTOR, SAS/Warehouse Administrator.

### Programming Language

The SAS Software is also a very powerful programming language. However, SAS Enterprise Guide is a point and click software where you can use the full power of SAS without knowing the programming language.


### Exploring Files in SAS

In the SAS System, you can explore and manage both SAS files and other files. In the *Explorer* window, you can view and manage SAS files, which are stored in libraries. A library is simply a link to the physical location of a file such as a directory or folder. If you delete the library the file still exist in the directory or folder.

Make the *Explorer* window active and double-click on **Libraries**. All the active libraries (directories) are listed. Double-click the **Sashelp** library. All members (files and directories) are listed. Move back up to the top level, i.e. **View > Up One Level**, or just click on its icon (  ).

SAS allows you to explore files in your operating environment via **View > My Favorite Folders**.

### Working with SAS files: Details and Sort

To see detail information such as *name, size, type, description* and *date modified* for any file double-click on its library and then click the detail tool () or **View -> Details**. In the *Explorer* window, you can sort files by any column. Just click the heading of the column to sort the files. For example, click on the **Type** column to sort the files by file type. Click it again to reverse the order. Go back to the original order by **View -> Refresh**.

### File properties

The SAS System allows you to view detail properties on a file. Just right-click on the file in the *Explorer* window and select **Properties** from the pop-up menu. The **General Properties** window will be displayed. To see other properties, click the tab at the top of this window. Try this using any file in the **sashelp** library.

To open a file just double-click its icon in the *Explorer* window. The file will be opened using the **Viewtable** window. Try this again using any file in the **sashelp** library.

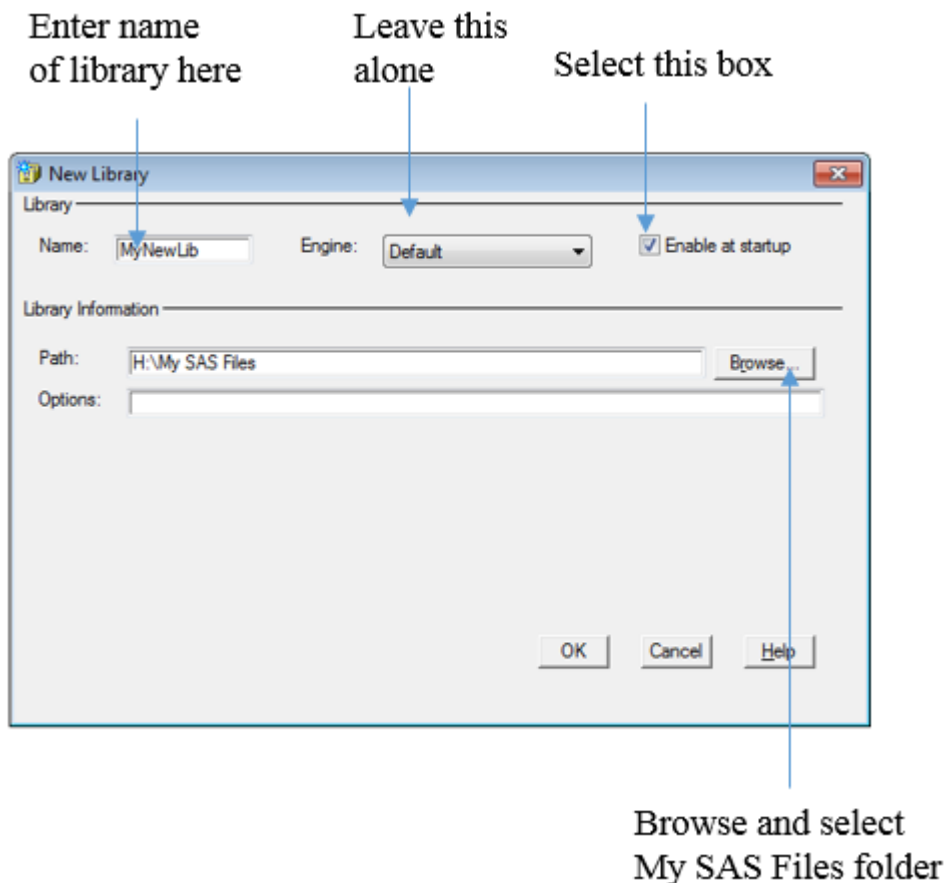
### Working with SAS Libraries

#### Creating a new library

There is already a folder in your H Drive (Documents) called *My SAS Files*. Check to confirm that this folder is there. We need to create a new library in SAS and link it to this folder (*My SAS Files*). All the SAS files that we create in this workshop will be stored in the folder (*My SAS Files*).

To create a new *library* follow these instructions:

- Make the *Explorer* window active and double-click **Libraries**.
- Select **File > New**.
- In the *New Library* window, type a name for the new library, e.g. *MyNewLib*. Leave the default engine selected.
- Select **Enabled at Startup**.
- Click **Browse** and go H Drive (Documents) and select *My SAS Files*.
- Click **OK** to create the library. *MyNewLib* appears in the active **Libraries** list. The completed dialogue box is shown below.



### Copying a table (file or data set)

- Make sure that the *Explorer* window is active, then select **View -> Show Tree**. The window will now have two panes, left and right.
- Double-click the *Sashelp* library located in the left pane. Its content will be displayed in the right pane. Scroll down and look for **Prdsale** table.
- Click on **Prdsale**, drag and drop it on *MyNewLib* on the left pane.
- Open *MyNewLib* and confirm that the table was copied successfully.

Note: You can change the name of a table you have just copied quite easily. Right-click the name of the table and select **Rename** from the pop-up menu. Re-name **Prdsale** as **Myproductsales**. Click **OK**.

# Programming using SAS Base

## Introduction

As mentioned already the SAS software has a very powerful programming language. There are many things that you can do through programming that you can't do through the point and click approach. In fact, the power of SAS still lies in the programming language. This section introduces some of the basic ideas in SAS programming.

**SAS programs:** A SAS program is simply a sequence of statements executed in order. This can be simple or complex in nature.

**SAS statements:** Every SAS statement ends with a **semicolon (;)**. Some SAS statements have optional parts or sections.

## Layout of SAS programs:

- SAS statements can be in upper or lowercase
- Statement can continue on the next line as long as you don't split words in two
- More than one statement can be on the same line
- Statements can start in any column

**Comments:** Including comments in your program makes it easy for others to read and understand. There are two ways to include comments in your SAS programs:

- One starts with an asterisk (\*) and ends with a semicolon (;).
- The other starts with slash asterisk (/\*) and ends with asterisk slash (\*/).

That is, your comment goes between these characters. SAS ignores comments during the execution of your program.

## SAS Data Sets

You write SAS programs to create, to manipulate or to analyse data sets. It is therefore important to understand what a SAS data set is. The following applies to SAS data sets:

**Variables and observations:** In SAS terminology, data set consists of variables and observations. In terms of relational databases, SAS data sets are also called tables, observations are called rows, and variables are called columns.

**Data types:** There are just two data types in SAS: Numeric and Character. The dollar sign (\$) is used to denote character variables.

**Missing data:** Missing numeric data values are represented by periods (.), while missing character values are represented by blanks.

**Size of SAS data sets:** SAS can handle up to 32,767 variables in a single data set. The number of observations is limited by your computer's capacity.

**Rules for SAS names:** When naming your libraries, variables and data sets bear the following rules in mind:

- names must be 32 characters or fewer in length
- names must start with a letter or underscore ( \_ )
- names can contain only letters, numerals, or underscores ( \_ ). No % \$ ! \* & # @, please.
- names can contain upper- and lowercase letters.

Information on SAS data set such as the name, date created, version of SAS used, are stored as part of the data set. Information on the variables such as type, name, length, format and informat are also stored.



## The Two Parts of a SAS Program

A SAS program is made up of two parts a data step and a procedure (proc) step. This can be simple or complex. Examine the **simple program** below:

```
DATA firstdata;  
    quantity=500;  
    price=50;  
    Totalcost=quantity*price;
```

```
PROC PRINT DATA=firstdata;  
RUN;
```

### Data steps:


- begin with DATA statements
- read and modify data
- create a SAS data set

The data step has built-in loops that execute line by line and observation by observation.

### Proc steps:

- begin with PROC statements
- perform specific analysis or function
- produce results or report

## Choosing a Mode for Submitting SAS Programs

A SAS program does nothing until you submit or execute it. Once you have completed your program, there are several ways to submit or execute the program. In this tutorial we recommend you select **Run > Submit** from the menu bar or just click on the submit icon (  ) from the toolbar.

You may have up to three ways to issue commands:

- Menus
- The tool bar
- SAS command bar (command line)

### Reading the SAS Log

SAS writes messages in to the log window each time you submit a program. Always examine the log first before you examine your output if any. There is plenty of important information in the log. You may also find warnings and other type of notes, which sometimes indicate errors and other times just provide useful information.

### Viewing and Printing the SAS Output

As we are using the SAS windowing environment, your output will go to the Output window by default. Listing of the different parts of your output will also be displayed in the Result window.

Note that you can print or save all or just parts of your output.

## Standard and Non-standard Numeric data

Standard numeric data contain only numbers, decimal points, minus signs, and E for scientific notation. Numbers with commas and pound signs are example of non-standard data. To read non-standard data you use SAS informats. To print out non-standard data you use SAS formats. Examples will be highlighted as we proceed through this document.

### Practice 1

Type the simple program below into the Editor window and submit it (via **Run > Submit**). Don't forget the semi colon (;)!

```
DATA firstdata;  
    quantity=500;  
    price=50;  
    Totalcost=quantity*price;
```

```
PROC PRINT DATA=firstdata;  
RUN;
```

Examine the Log and Output Window.

## How to Get Your Data into the SAS System

### SAS Data Libraries

This is used to store and manage data in SAS files. A SAS data library is a collection of one or more SAS files that is recognized by the SAS System. Each file belongs to a particular library. A library reference (libref) is the name you associate with the SAS data library. Each SAS data library is assigned a libref. You reference files in the library by using the corresponding libref followed by the name of the file.

### Temporary and Permanent Data Sets

The SAS System creates two types of data sets:

1. Temporary data set - A temporary SAS data set exists only for the duration of the current SAS session and it is lost forever when you exit the SAS System. The default *libref* for temporary data set is *work*. You may have noticed this library when we created your library earlier called my *MyNewLib*! In other words, the SAS System automatically assigns the libref *work* to the data set name you specify. When working with files in this library, you can specify the libref or ignore it, SAS does not mind.
2. Permanent data set - A permanent SAS data set exists after the end of the current session and until you deletes it. In creating or working with a permanent SAS data set you must specify the libref and the data set name. For example to reference the *Prdsale* data set that you copied into *MyNewLib* you use *MyNewLib.Prdsale* in your SAS program.

Both types of SAS data sets have two-level names in the form of *libref.data-set-name*. The sample data sets found in SAS are permanent SAS data sets. For example, some data sets are kept in the *sashelp* library.

### Methods of Getting Your Data into the SAS System

- Entering data directly into SAS data sets, i.e. creating a data set from scratch
- Creating SAS data sets from raw data files
- Converting other software's data files into SAS data sets

- Reading other software's data files directly

## Creating SAS Data Set from scratch

**Internal Raw Data:** Use the **DATALINES** or **CARDS** statement to tell SAS that raw data follows. The following example illustrates this. Type and submit this little program. Examine the log window and then the output window.

### Practice 2

```
DATA seconddata;
    INPUT name $ weight height;
    CARDS;
James 65 1.65
John 70 1.50
Lisa 60 1.70
;
PROC PRINT DATA=seconddata;
RUN;
```

The **DATA** statement tells SAS to create a temporary data set called *seconddata*. The **INPUT** statement tells SAS to create three variables, the first is character (*name*) and the other two are numeric (*weight* and *height*). If a variable name comes before \$ then it is a character variable otherwise numeric. The semicolon after the data indicates end of data line. The **PROC PRINT** statement prints out the output on the screen not on paper.

### How will you modify the program to create a permanent SAS data set?

**External Raw Data Files:** You can read raw data to create a SAS data set using the **INFILE** statement, e.g. the following data in text format could be read into the SAS system using the program below. As an exercise, use NotePad to type in the raw data. Save it as *firsttrial* in the **Temp** directory in drive C. The data values are separated by space. Do not type in the first line.

### Practice 3

#### Raw data firsttrial.txt

```
Lucky 2.3 1.9 . 3.0
Spot 4.6 2.5 3.1 .5
Tubs 7.1 . . 3.8
Hop 4.5 3.2 1.9 2.6
Noisy 3.8 1.3 1.8
1.5
Winner 5.7 . . .
```

Now type the following program in SAS Program Editor window to read the raw data file and create a SAS data set.

#### Program to read raw data

```
* Create a temporary SAS data set named data1;
* Read the data file firsttrial.txt using list input;
DATA data1;
    INFILE 'c:\Temp\firsttrial.txt';
```

```

INPUT ToadName $ Weight Jump1 Jump2 Jump3;
* Print the data to make sure that the file was read correctly;
PROC PRINT DATA = data1;
TITLE 'SAS Data Set Toads';
RUN;

```

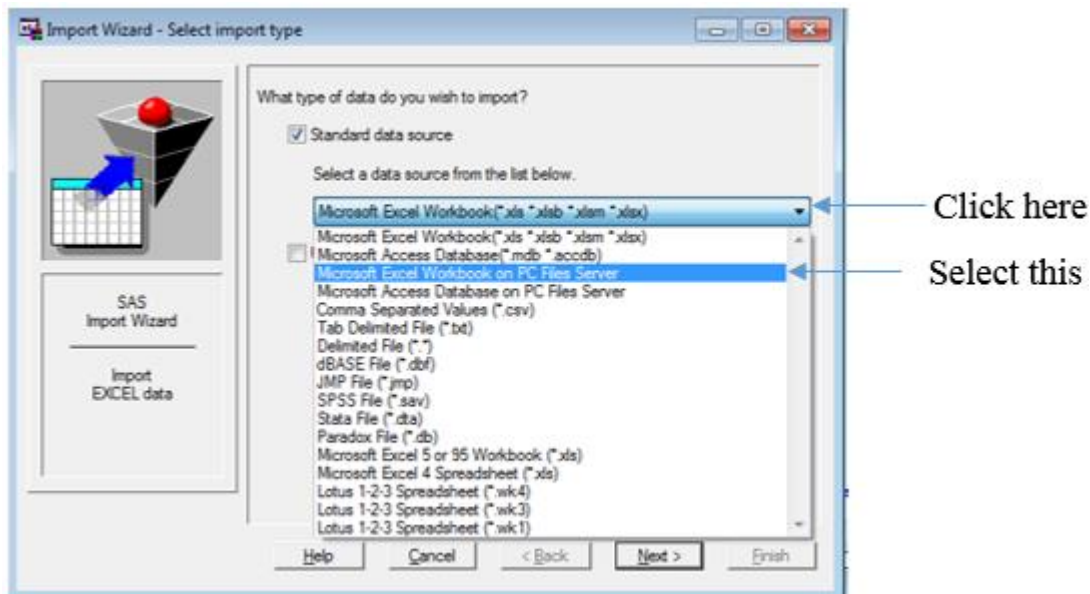
The Input statement tells SAS what variables to read in the order given.

When dealing with long records, include the LRECL=option in the INFILE statement, e.g. `Infile 'c:\stk\Training\sas\parks.txt' LRECL=2000;` This ensure that data are not truncated.

### Importing Data into SAS

You can also import data directly into SAS. Follow these instructions.

- Make sure that the **Explorer** window is your active window then select **File -> Import Data....**We want to import an **Excel** file.
- On the displayed dialogue box select **Microsoft Excel Workbook on PC File Server** from the dropdown arrow (see diagram).



- Click **Next** and select **Browse...**
- Under **File name:** type `\\campus\software\dept\spss`. Then click **Open** and select **Employee data**.
- Click **Open** again. Notice that the full path of the file appears next to **Workbook:**. So if you knew the full path of your file you could have type it in.
- Click **OK**. Then click **Next** and select *Mynewlib* from the library dropdown menu.
- Under **Member** type the name of your file e.g. *myfirstimport*.
- Click **Next** and click **Finish**. Check the **Log** window to make sure that the file was imported successfully. Open and examine the file. Make sure you understand the data file as you will be using it in the next section.

## Working with your Data

Once you have successfully included your data in the SAS software, there is a lot that you can do with it. This section of the document gives you examples of some of the things that you can do. You can create new variables from existing ones, use SAS functions on variables, request various types of statistics, build a model, etc.

## Basic Statistical Procedures

### Examining the Distribution of Data with PROC UNIVARIATE

This is a good procedure to help you explore your data before you do any formal statistical testing. Type this simple program into the data editor and submit it:

```
proc univariate data=mynewlib.myfirstimport plot normal;  
var salary salbegin;  
run;
```

The **proc univariate** of the statement tells SAS which procedure to use, the **data=** part tells SAS which data set to use. The **normal** option produces tests of normality while the **plot** option produces three plots of your data (stem-and-leaf, box plot and normal probability plot). The **var** statement tells SAS which columns (variables) to analyse.

Examine the output. What conclusion can you draw?

**Hint: Skewness** indicates how symmetric the distribution is while **Kurtosis** indicates how flat or peaked the distribution is. The **normal distribution** has values of zero for both Skewness and Kurtosis. If the **mean** (average) is very different from the **median** this indicates that the data is not normal.

### Producing Statistics with PROC MEANS

Proc means produce the descriptive statistics that you produce with proc Univariate. Univariate prints out all these statistics by default. But if you know you want only a few of these statistics then proc Means is better. Type this simple program into the data editor and submit it:

```
proc means data= mynewlib.myfirstimport n mean median std mode;  
var salary salbegin;  
run;
```

This program will generate only the statistics specified [sample size (n), mean, median, standard deviation (std) and mode].

### Testing Categorical Data with PROC FREQ

Produces many statistics for categorical data. The best known of which is chi-square. Type this simple program into the data editor and submit it:

```
proc format;  
value $gender 'm'='male'  
              'f'='Female';  
value group 1='Clerical'
```

```
2='Custodial'  
3='Manager';
```

```
proc freq data= mynewlib.myfirstimport;  
format gender $gender. jobcat group.;  
tables gender jobcat gender*jobcat / chisq measures;  
run;
```

Examine the output. What conclusion can you draw?

The **proc format** statement instructs SAS to create two formats for that are used to display the actual values used in the data sets. For example, instead of displaying **m** SAS will display **male**. Notice how the created formats are used in the **format** statement to associate it to the correct variables. The **tables** statement tells SAS to generate 3 tables one for gender, one for jobcat and a contingency table for gender and jobcat.

### Examining Correlations with PROC CORR

A correlation coefficient measures the relationship between two variables. Before producing the correlation coefficient between two interval variables, it is advisable to produce a scatter plot first. Type this simple program into the data editor and submit it:

```
proc gplot data= mynewlib.myfirstimport;  
plot salary*salbegin = gender;  
run;  
proc corr data= mynewlib.myfirstimport;  
var salary salbegin;  
run;
```

Examine the output. What conclusion can you draw?

### Using PROC ANOVA for One-Way Analysis of Variance

The ANOVA procedure is one of several in the SAS System that perform analysis of variance. PROC ANOVA is specifically designed for balanced data – data where there are equal numbers of observations (cases or subjects) in each classification. If your data is not balanced use the GLM procedure (see next example). To do this example you need to first import the data file from [\\campus\software\dept\spss](#). The file is an SPSS file and is called ‘*number of words recalled*’. So you must tell SAS that you want to import an SPSS file in the Import Window (File -> Import Data...). Import the file into your library *Mynewlib* and give it a member name *words*. Once the data is imported, you will be able to analyse it. Type this simple program into the data editor and submit it:

```
proc anova data= mynewlib.words;  
class group;  
model score=group;  
means group / scheffe;  
run;  
quit; *the quit statement is necessary because proc anova will still be running;
```

Examine the output. What conclusion can you draw?

### Using PROC GLM

Use this when your data is unbalance. Type this simple program into the data editor and submit it:

```
proc glm data= mynewlib.myfirstimport;
    class jobcat;
    model salary=jobcat;
    means jobcat / scheffe;
run;
quit;
```

Examine the output. What conclusion can you draw?

### Using PROC T TEST

You can use this procedure for group means comparison, one-sample comparison or paired sample comparison. It is assumed that the data is normally distributed.

#### Group Means Comparison (Independent samples)

Type this simple program into the data editor and submit it:

```
proc ttest data=stk.employeeedata;
    class gender;
    var salary;
run;
```

Examine the output. What conclusion can you draw?

#### One-Sample Comparison

Is it true that the average salary of people working in this company is \$30,000? Type this simple program into the data editor and submit it:

```
proc ttest data=stk.employeeedata h0=30000;
    var salary;
run;
```

Examine the output. What conclusion can you draw?

#### Paired Sample Comparison

Some common examples of paired samples are:

- pre- and post-test scores for a student receiving tutoring
- fuel efficiency readings of two fuel types observed on the same automobile
- sunburn scores for two sunblock lotions, one applied to the individual's right arm, one to the left arm
- political attitude scores of husbands and wives

Type this simple program into the data editor and submit it:

```
proc ttest data=stk.employeedata;
  paired Salary*Salbegin;
run;
```

Examine the output. What conclusion can you draw?

### Nonparametric Analysis

Most formal statistical analysis assumes that your data is normally distributed. If your data fails this assumption you must use nonparametric analysis. We have seen that salary is not normally distributed. Type this simple program into the data editor and submit it:

### Independent samples

```
ods graphics on;
proc npar1way data= mynewlib.myfirstimport wilcoxon median;
  class gender;
  var salary;
run;
ods graphics off;
```

Examine the output. What conclusion can you draw?

### Using PROC REG

You use proc reg to build linear regression model. Type this simple program into the data editor and submit it:

```
ods graphics on;
proc reg data= mynewlib.myfirstimport;
  model salary=salbegin prevexp;
run;
ods graphics off;
```