



CHAPTER 1

Getting Started Using SAS Software

- 1.1 The SAS Language 2
- 1.2 SAS Data Sets 4
- 1.3 The Two Parts of a SAS Program 6
- 1.4 The DATA Step's Built-in Loop 8
- 1.5 Choosing a Mode for Submitting SAS Programs 10
- 1.6 Windows and Commands in the SAS Windowing Environment 12
- 1.7 Submitting a Program in the SAS Windowing Environment 14
- 1.8 Reading the SAS Log 16
- 1.9 Viewing Your Results in the Output Window 18
- 1.10 Creating HTML Output 20
- 1.11 SAS Data Libraries 22
- 1.12 Viewing Data Sets in the Viewtable Window 24
- 1.13 Viewing the Properties of Data Sets with SAS Explorer 26
- 1.14 Using SAS System Options 28

1.1 The SAS Language

Many software applications are either menu driven, or command driven (enter a command—see the result). SAS is neither. With SAS, you use statements to write a series of instructions called a SAS program. The program communicates what you want to do and is written using the SAS language. There are some menu-driven front ends to SAS, for example SAS Enterprise Guide software, which make SAS appear like a point-and-click program. However, these front ends still use the SAS language to write programs for you. You will have much more flexibility using SAS if you learn to write your own programs using the SAS language. Maybe learning a new language is the last thing you want to do, but be assured that although there are parallels between SAS and languages you know (be they English or JAVA), SAS is much easier to learn.

SAS programs A SAS program is a sequence of statements executed in order. A statement gives information or instructions to SAS and must be appropriately placed in the program. An everyday analogy to a SAS program is a trip to the bank. You enter your bank, stand in line, and when you finally reach the teller's window, you say what you want to do. The statements you give can be written down in the form of a program:

```
I would like to make a withdrawal.  
My account number is 0937.  
I would like $200.  
Give me five 20s and two 50s.
```

Note that you first say what you want to do, then give all the information the teller needs to carry out your request. The order of the subsequent statements may not be important, but you must start with the general statement of what you want to do. You would not, for example, go up to a bank teller and say, "Give me five 20s and two 50s." This is not only bad form, but would probably make the teller's heart skip a beat or two. You must also make sure that all the subsequent statements belong with the first. You would not say, "I want the largest box you have" when making a withdrawal from your checking account. That statement belongs with "I would like to open a safe deposit box." A SAS program is an ordered set of SAS statements like the ordered set of instructions you use when you go to the bank.

SAS statements As with any language, there are a few rules to follow when writing SAS programs. Fortunately for us, the rules for writing SAS programs are much fewer and simpler than those for English.

The most important rule is

Every SAS statement ends with a semicolon.

This sounds simple enough. But while children generally outgrow the habit of forgetting the period at the end of a sentence, SAS programmers never seem to outgrow forgetting the semicolon at the end of a SAS statement. Even the most experienced SAS programmer will at least occasionally forget the semicolon. You will be two steps ahead if you remember this simple rule.

Layout of SAS programs There really aren't any rules about how to format your SAS program. While it is helpful to have a neat looking program with each statement on a line by itself and indentions to show the various parts of the program, it isn't necessary.

- ◆ SAS statements can be in upper- or lowercase.
- ◆ Statements can continue on the next line (as long as you don't split words in two).
- ◆ Statements can be on the same line as other statements.
- ◆ Statements can start in any column.

So you see, SAS is so flexible that it is possible to write programs so disorganized that no one can read them, not even you. (Of course, we don't recommend this.)

Comments To make your programs more understandable, you can insert comments into your programs. It doesn't matter what you put in your comments—SAS doesn't look at it. You could put your favorite cookie recipe in there if you want. However, comments are usually used to annotate the program, making it easier for someone to read your program and understand what you have done and why.

There are two styles of comments you can use: one starts with an asterisk (*) and ends with a semicolon (;). The other style starts with a slash asterisk (/*) and ends with an asterisk slash (*//). The following SAS program shows the use of both of these style comments:

```
* Read animals' weights from file;
DATA animals;
    INFILE 'c:\MyRawData\Zoo.dat';
    INPUT Lions Tigers;
PROC PRINT DATA = animals; /* Print the results */
RUN;
```

Since some operating environments interpret a slash asterisk (/*) in the first column as the end of a job, be careful when using this style of comment not to place it in the first column. For this reason, we chose the asterisk-semicolon style of comment for this book.

Programming tips People who are just learning a programming language often get frustrated because their programs do not work correctly the first time they write them. Writing programs should be done in small steps. Don't try to tackle a long complicated program all at once. If you start small, build on what works, and always check your results along the way, you will increase your programming efficiency. Sometimes programs that do not produce errors are still incorrect. This is why it is vital to check your results as you go even when there are no errors. If you do get errors, don't worry. Most programs simply don't work the first time, if for no other reason than you are human. You forget a semicolon, misspell a word, have your fingers in the wrong place on the keyboard. It happens. Often one small mistake can generate a whole list of errors. If you build your programs piece by piece, programs are much easier to correct when something goes wrong.

1.2 SAS Data Sets

Before you run an analysis, before you write a report, before you do anything with your data, SAS must be able to read your data. Before SAS can analyze your data, the data must be in a special form called a SAS data set.¹ Getting your data into a SAS data set is usually quite simple as SAS is very flexible and can read almost any data. Once your data have been read into a SAS data set, SAS keeps track of what is where and in what form. All you have to do is specify the name and location of the data set you want, and SAS figures out what is in it.

Variables and observations Data, of course, are the primary constituent of any data set. In traditional SAS terminology the data consist of variables and observations. Adopting the terminology of relational databases, SAS data sets are also called tables, observations are also called rows, and variables are also called columns. Below you see a rectangular table containing a small data set. Each line represents one observation, while Id, Name, Height, and Weight are variables. The data point Charlie is one of the values of the variable Name and is also part of the second observation.

		Variables (Also Called Columns)			
		Id	Name	Height	Weight
Observations (Also Called Rows)	1	53	Susie	42	41
	2	54	Charlie	46	55
	3	55	Calvin	40	35
	4	56	Lucy	46	52
	5	57	Dennis	44	.
	6	58		43	50

Data types Raw data come in many different forms, but SAS simplifies this. In SAS there are just two data types: numeric and character. Numeric fields are, well, numbers. They can be added and subtracted, can have any number of decimal places, and can be positive or negative. In addition to numerals, numeric fields can contain plus signs (+), minus signs (-), decimal points (.), or E for scientific notation. Character data are everything else. They may contain numerals, letters, or special characters (such as \$ or !) and can be up to 32,767 characters long.

If a variable contains letters or special characters, it must be character data. However, if it contains only numbers, then it may be numeric or character. You should base your decision on how you will use the variable.² Sometimes data that consist solely of numerals make more sense as character data than as numeric. ZIP codes, for example, are made up of numerals, but it just doesn't make sense to add, subtract, multiply, or divide ZIP codes. Such numbers make more sense as character data. In the previous data set, Name is obviously a character variable, and Height and Weight are numeric. Id, however, could be either numeric or character. It's your choice.

¹There are exceptions. If your data are in a format written by another software product, you may be able to read your data directly without creating a SAS data set. For database management systems and spreadsheets, you may be able to use SAS/ACCESS software. See chapter 2 for more information. For SPSS, see appendix B.

²If disk space is a problem, you may also choose to base your decision on storage size. You can use the LENGTH statement, discussed in section 10.15, to control the storage size of variables.

Missing data Sometimes despite your best efforts, your data may be incomplete. The value of a particular variable may be missing for some observations. In those cases, missing character data are represented by blanks, and missing numeric data are represented by a single period (.). In the preceding data set, the value of Weight for observation 5 is missing, and its place is marked by a period. The value of Name for observation 6 is missing and is just left blank.

Size of SAS data sets Prior to SAS 9.1, SAS data sets could contain up to 32,767 variables. Beginning with SAS 9.1, the maximum number of variables in a SAS data set is limited by the resources available on your computer—but SAS data sets with more than 32,767 variables cannot be used with earlier versions of SAS. The number of observations, no matter which version of SAS you are using, is limited only by your computer's capacity to handle and store them.

Rules for SAS names You make up names for the variables in your data and for the data sets themselves. It is helpful to make up names that identify what the data represent, especially for variables. While the variable names A, B, and C might seem like perfectly fine, easy-to-type names when you write your program, the names Sex, Height, and Weight will probably be more helpful when you go back to look at the program six months later. Follow these simple rules when making up names for variables and data set members:

- ◆ Names must be 32 characters or fewer in length.³
- ◆ Names must start with a letter or an underscore (_).
- ◆ Names can contain only letters, numerals, or underscores (_). No %\$!*&#@, please.⁴
- ◆ Names can contain upper- and lowercase letters.

This last point is an important one. SAS is insensitive to case so you can use uppercase, lowercase or mixed case—whichever looks best to you. SAS doesn't care. The data set name heightweight is the same as HEIGHTWEIGHT or HeightWeight. Likewise, the variable name BirthDate is the same as BIRTHDATE and birThDaTe. However, there is one difference for variable names. SAS remembers the case of the first occurrence of each variable name and uses that case when printing results. That is why, in this book, we use mixed case for variable names but lowercase for other SAS names.

Documentation stored in SAS data sets In addition to your actual data, SAS data sets contain information about the data set such as its name, the date that you created it, and the version of SAS you used to create it. SAS also stores information about each variable, including its name, label (if any), type (numeric or character), length (or storage size), and position within the data set. This information is sometimes called the descriptor portion of the data set, and it makes SAS data sets self-documenting.

³Beginning with SAS 9, format names can also be 32 characters long, and informat names can be 31 characters (including the \$ for character values). Prior to SAS 9, format names could be 8 characters while informat names could be 7 characters (also including the \$). Librefs and filerefs must be 8 characters or fewer in length, and member names for versioned data sets must be 28 characters or fewer.

⁴It is possible to use special characters, including spaces, in variable names if you use the system option VALIDVARNames=ANY and a name literal of the form 'variable-name'N. See the SAS Help and Documentation for details.

1.3 The Two Parts of a SAS Program



SAS programs are constructed from two basic building blocks: DATA steps and PROC steps. A typical program starts with a DATA step to create a SAS data set and then passes the data to a PROC step for processing. Here is a simple program that converts miles to kilometers in a DATA step and prints the results with a PROC step:

```

DATA step [ DATA distance;
           [ Miles = 26.22;
           [ Kilometers = 1.61 * Miles;

PROC step [ PROC PRINT DATA = distance;
           [ RUN;

```

DATA and PROC steps are made up of statements. A step may have as few as one or as many as hundreds of statements. Most statements work in only one type of step—in DATA steps but not PROC steps, or vice versa. A common mistake made by beginners is to try to use a statement in the wrong kind of step. You're not likely to make this mistake if you remember that DATA steps read and modify data while PROC steps analyze data, perform utility functions, or print reports.

DATA steps start with the DATA statement, which starts, not surprisingly, with the word DATA. This keyword is followed by a name that you make up for a SAS data set. The DATA step above produces a SAS data set named DISTANCE. In addition to reading data from external, raw data files, DATA steps can include DO loops, IF-THEN/ELSE logic, and a large assortment of numeric and character functions. DATA steps can also combine data sets in just about any way you want, including concatenation and match-merge.

Procedures, on the other hand, start with a PROC statement in which the keyword PROC is followed by the name of the procedure (PRINT, SORT, or MEANS, for example). Most SAS procedures have only a handful of possible statements. Like following a recipe, you use basically the same statements or ingredients each time. SAS procedures do everything from simple sorting and printing to analysis of variance and 3D graphics. Other SAS procedures perform utility functions such as importing data files and data entry.

A step ends when SAS encounters a new step (marked by a DATA or PROC statement), a RUN statement, or, if you are running in batch mode, the end of the program.¹ RUN statements tell SAS to run all the preceding lines of the step and are among those rare, global statements that are not part of a DATA or PROC step. In the program above, SAS knows that the DATA step has ended when it reaches the PROC statement. The PROC step ends with a RUN statement, which coincides with the end of the program.

¹If you use SAS long enough, you may run into an exception. Steps can also terminate with a QUIT, STOP, or ABORT statement.

While a typical program starts with a DATA step to input or modify data and then passes the data to a PROC step, that is certainly not the only pattern for mixing DATA and PROC steps. Just as you can stack building blocks in any order, you can arrange DATA and PROC steps in any order. A program could even contain only DATA steps or only PROC steps.

To review, the table below outlines the basic differences between DATA and PROC steps:

DATA steps	PROC steps
▶ begin with DATA statements	▶ begin with PROC statements
▶ read and modify data	▶ perform specific analysis or function
▶ create a SAS data set	▶ produce results or report

As you read this table, keep in mind that it is a simplification. Because SAS is so flexible, the differences between DATA and PROC steps are, in reality, more blurry. The table above is not meant to imply that PROC steps never create SAS data sets (most do), or that DATA steps never produce reports (they can). Nonetheless, you will find it much easier to write SAS programs if you understand the basic functions of DATA and PROC steps.

1.4 The DATA Step's Built-in Loop

DATA steps read and modify data, and they do it in a way that is flexible, giving you lots of control over what happens to your data. However, DATA steps also have an underlying structure, an implicit, built-in loop. You don't tell SAS to execute this loop: SAS does it automatically. Memorize this:

DATA steps execute line by line and observation by observation.

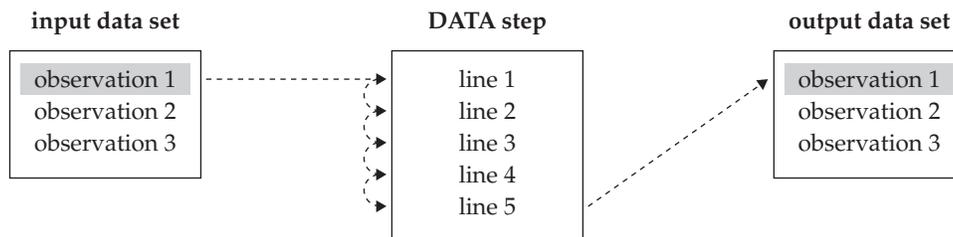
This basic concept is rarely stated explicitly. Consequently, new users often grow into old users before they figure this out on their own.

The idea that DATA steps execute line by line is fairly straightforward and easy to understand. It means that, by default, SAS executes line one of your DATA step before it executes line two, and line two before line three, and so on. That seems common sense, and yet new users frequently run into problems because they try to use a variable before they create it. If a variable named Z is the product of X and Y, then you better make sure that the statements creating X and Y come before the statements creating Z.

What is not so obvious is that while DATA steps execute line by line, they also execute observation by observation. That means SAS takes the first observation and runs it all the way through the DATA step (line by line, of course) before looping back to pick up the second observation. In this way, SAS sees only one observation at a time.

Imagine a SAS program running in slow motion: SAS reads observation number one from your input data set. Then SAS executes your DATA step using that observation. If SAS reaches the end of the DATA step without encountering any serious errors, then SAS writes the current observation to a new, output data set and returns to the beginning of the DATA step to process the next observation. After the last observation has been written to the output data set, SAS terminates the DATA step and moves on to the next step, if there is one. End of slow motion; please return to normal gigahertz.

This diagram illustrates how an observation flows through a DATA step:



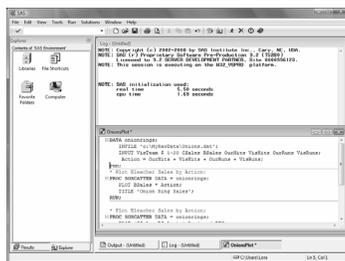
SAS reads observation number one and processes it using line one of the DATA step, then line two, and so on until SAS reaches the end of the DATA step. Then SAS writes the observation in the output data set. This diagram shows the first execution of the line-by-line loop. Once SAS finishes with the first observation, it loops back to the top of the DATA step and picks up observation two. When SAS reaches the last observation, it automatically stops.¹

Here is an analogy. DATA step processing is a bit like voting. When you arrive at your polling place, you stand in line behind other people who have come to vote. When you reach the front of the line you are asked standard questions: “What is your name? Where do you live?” Then you sign your name, and you cast your vote. In this analogy, the people are observations, and the voting process is the DATA step. People vote one at a time (or observation by observation). Each voter’s choices are secret, and peeking at your neighbor’s ballot is definitely frowned upon. In addition, each person completes each step of the process in the same order (line by line). You cannot cast your vote before you give your name and address. Everything must be done in the proper order.

¹ If this seems a bit too structured, don’t worry. You can override the line-by-line and observation-by-observation structure in a number of ways. For example, you can use the RETAIN statement, discussed in section 3.10, to make data from the previous observation available to the current observation. You can also use the OUTPUT statement, discussed in sections 6.9 and 6.10, to control when observations are written to the output data set.

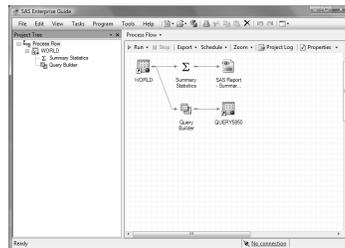
1.5 Choosing a Mode for Submitting SAS Programs

So far we have talked about writing SAS programs, but simply writing a program does not give you any results. Just like writing a letter to your representative in Congress does no good unless you mail it, a SAS program does nothing until you submit or execute it. You can execute a SAS program several ways, but not all methods are available for all operating environments. Check in the SAS Help and Documentation for your operating environment to find out which methods are available to you. The method you choose for executing a SAS program will depend on your preferences and on what is most appropriate for your application and your environment. If you are using SAS at a large site with many users, then ask around and find out which is the most accepted method of executing SAS. If you are using SAS on your own personal computer, then choose the method that suits you.



SAS windowing environment If you type SAS at your system prompt, or click on the SAS icon, you will most likely get into the SAS windowing environment. In this interactive environment, you can write and edit SAS programs, submit programs for processing, and view and print your results. In addition, there are many SAS windows for performing different tasks such as managing SAS files, customizing the interface, accessing SAS Help, and importing or exporting data. Exactly what your windowing environment looks like depends on the type of computer you

are using, the operating environment on the computer, and what options are in effect when you start up SAS. If you are using a personal computer, then the SAS windowing environment will look similar to other programs on your computer, and many of the features will be familiar to you.



SAS Enterprise Guide If you have SAS Enterprise Guide software,¹ which runs only under Windows, you may choose to submit your programs from within SAS Enterprise Guide. To do this, open a Code window where you can enter your SAS program or open an existing SAS program. Then you can choose to run your code on the local machine, or on a remote server where SAS is installed. To run your SAS program on a remote server, you must have SAS Integration Technologies software installed. Also, SAS Enterprise Guide can write SAS

code for you through its extensive menu system.



Noninteractive mode Noninteractive mode is where your SAS program statements are in a file on your system, and you start up SAS specifying that you want to execute that file. SAS immediately starts to process your file and ties up your computer, or window, until it is finished. The results are usually placed in a file or files, and you are returned to your system prompt.

¹ Beginning with SAS 9, SAS Enterprise Guide software is included with Base SAS software for Windows. SAS Enterprise Guide software is also available with SAS Version 8, but is licensed separately.

Noninteractive mode is useful in many situations. This mode is good if you want your program to execute immediately, but you do not want to or cannot use a windowing environment. Noninteractive mode is usually started by typing SAS at your system prompt (shown here as \$), followed by the filename containing your program statements:

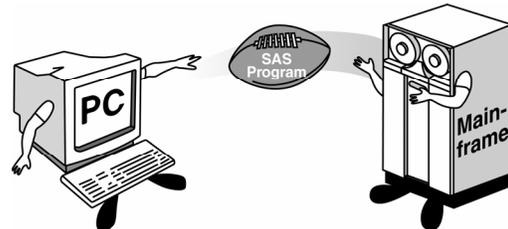
```
$ SAS MyFile.sas
```

Batch or background mode With batch or background mode, your SAS program is in a file. You submit the file for processing with SAS. Your SAS program may start executing immediately, or it could be put in a queue behind other jobs. Batch processing is used a lot on mainframe computers. You can continue to work on your computer while your job is being processed, or better yet, you can go to the baseball game and let the computer work in your absence. Batch processing is usually less expensive than other methods and is especially good for large jobs which can be set up to execute during off-hours when the rates are at their lowest. When your job is complete, the results will be placed in a file or files, which you can display or print at any time.



To find out how to submit SAS programs for batch processing, check the SAS Help and Documentation for your operating environment or with other SAS users at your site. Even sites with the same operating environment may have different ways of submitting jobs in batch mode.

Remote submit If you have SAS/CONNECT software, it is possible to write and develop your SAS programs on one system, then submit them for processing on another. Using this method, you write your program on your local machine, establish a connection to the remote machine, and run the program on the remote machine. Then the results are delivered back to your local machine. You might want to do this if your remote machine is much more powerful than your local machine, and you are running very large programs. Also, you might need to access large or shared data files on the remote machine.



Interactive line mode This mode is mentioned only because you might see it in the SAS documentation, and you might get into it by accident. In interactive line mode, you are prompted for SAS statements one line at a time. There is no easy way to correct mistakes once you have entered them, so unless you are an excellent typist, and an excellent programmer, interactive line mode is exceedingly frustrating.



If you do find yourself in this mode (you will know when you get a 1? as a prompt), you can get out by typing ENDSAS; and pressing ENTER. For example

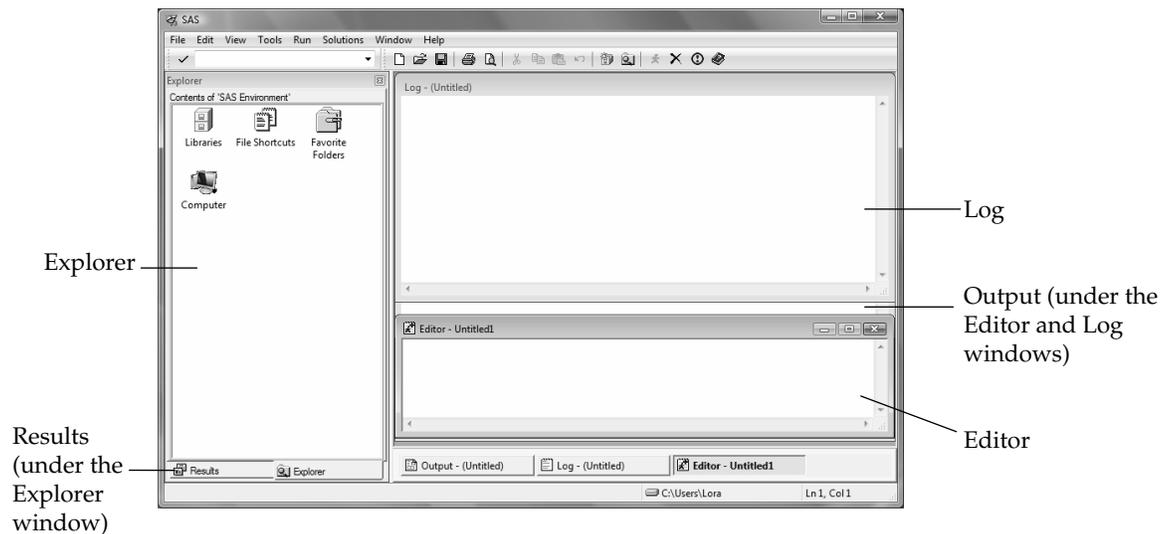
```
1? ENDSAS;
```

1.6 Windows and Commands in the SAS Windowing Environment

It used to be that SAS looked pretty much the same on all platforms, and you couldn't change its appearance. But now SAS adopts the look and feel of your operating environment, and there are many ways in which you can customize your SAS environment. This is good for you because many aspects of the SAS windowing environment will be familiar, and if you don't like the default view, you can change it. It makes writing about it more difficult, because we can't tell you exactly what your SAS session will look like and how it will behave. However, there are many common elements between the various operating environments, and you will probably already be familiar with those elements which are different.

The SAS Windows

There are five basic SAS windows: the Results and Explorer windows, and three programming windows: Editor, Log, and Output. It is possible to bring up SAS without all these windows, and sometimes the windows are not immediately visible (for example, in the Windows operating environment, the Output window comes up behind the Editor and Log windows), but all these windows do exist in your SAS session. There are also many other SAS windows that you may use for tasks such as getting help, changing SAS system options, and customizing your SAS session. The following figure shows the default view for a Microsoft Windows SAS session, with pointers to the five main SAS windows.



Editor This window is a text editor. You can use it to type in, edit, and submit SAS programs as well as edit other text files such as raw data files. In Windows operating environments, the default editor is the Enhanced Editor. The Enhanced Editor is syntax sensitive and color codes your programs making it easier to read them and find mistakes. The Enhanced Editor also allows you to collapse and expand the various steps in your program. For other operating environments, the default editor is the Program Editor whose features vary with the version of SAS and operating environment.

Log The Log window contains notes about your SAS session, and after you submit a SAS program, any notes, errors, or warnings associated with your program as well as the program statements themselves will appear in the Log window.

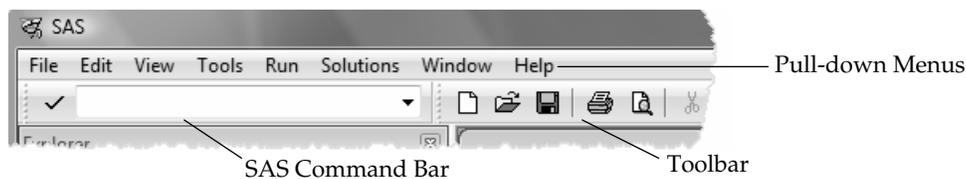
Output If your program generates any printable results, then they will appear in the Output window.

Results The Results window is like a table of contents for your Output window; the results tree lists each part of your results in an outline form.

Explorer The Explorer window gives you easy access to your SAS files and libraries.

The SAS Commands

There are SAS commands for performing a variety of tasks. Some tasks are probably familiar, such as opening and saving files, cutting and pasting text, and accessing Help. Other commands are specific to the SAS System, such as submitting a SAS program, or starting up a SAS application. You may have up to three ways to issue commands: menus, the toolbar, or the SAS command bar (or command line). The following figure shows the location of these three methods of issuing SAS commands in the Windows operating environment default view.



Menus Most operating environments will have pull-down menus located either at the top of each window, or at the top of your screen. If your menus are at the top of your screen, then the menus will change when you activate the different windows (usually by clicking on them). You may also have, for each window, context-sensitive pop-up menus that appear when you press the right or center button of your mouse.

Toolbar The toolbar, if you have one, gives you quick access to commands that are already accessible through the pull-down menus. Not all operating environments have a toolbar.

SAS command bar The command bar is a place that you can type in SAS commands. In some operating environments the command bar is located with the toolbar (as shown here); in other operating environments you may have a command line with each of the SAS windows (usually indicated by `Command=>`). Most of the commands that you can type in the command bar are also accessible through the pull-down menus or the toolbar.

Controlling your windows The Window pull-down menu gives you choices on how the windows are placed on your screen. You can also activate any of the programming windows by selecting it from the Window pull-down menu, typing the name of the window in the command line area of your SAS session, or simply clicking on the window.

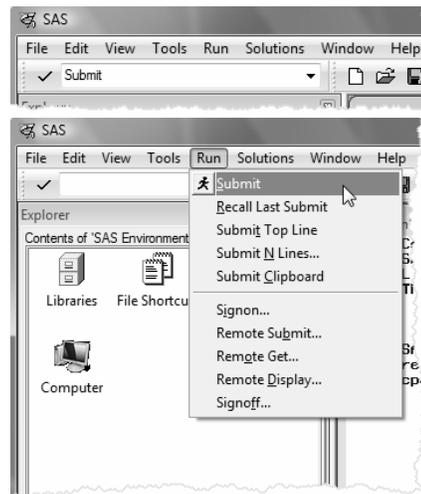
1.7 Submitting a Program in the SAS Windowing Environment

Naturally after going to the trouble of writing SAS programs, you want to see some results. As we have already discussed, there are several ways of submitting SAS programs. If you use the SAS windowing environment, then you can edit and submit programs, and see results all within the windowing environment.

Getting your program into the editor The first thing you need to do is get your program into the Editor window. You can either type your program into the editor, or you can bring the program into the Editor window from a file. The commands for editing in the editor and for opening files should be familiar. SAS tries to follow conventions that are common for your operating environment. For example, to open a file in the editor, you can select *Open* from the *File* pull-down menu. For some operating environments you may have an *Open* icon on the toolbar, and you may also have the option of pasting your file into the editor from the clipboard.

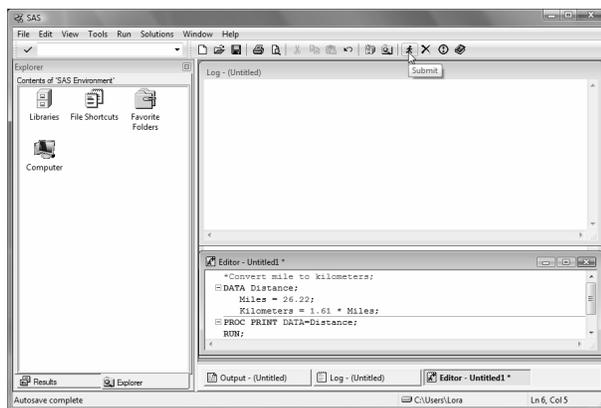
Submitting your program Once your program appears in the editor, you execute it using the *SUBMIT* command. Depending on your operating environment, you have a few choices on how to execute the *SUBMIT* command.

 Use the *Submit* icon on the toolbar.



Make the Editor window active and enter *SUBMIT* in the command line area of your SAS session.

Make the Editor window active and select *Submit* from the *Run* pull-down menu.

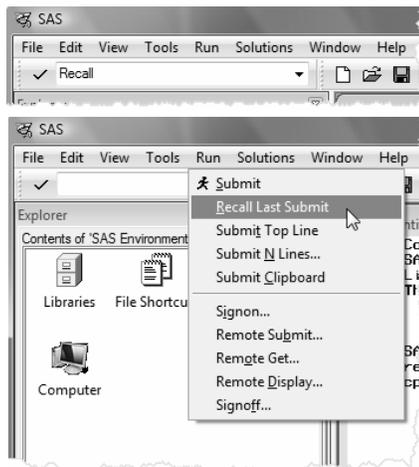
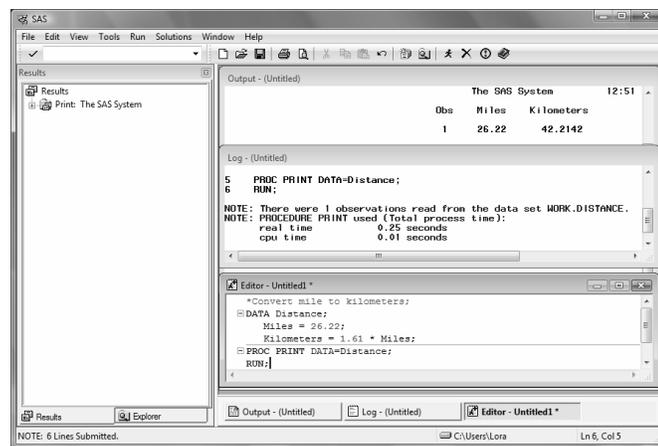


The figure to the right shows a program in the Enhanced Editor in the Windows operating environment ready to be submitted using the *Submit* icon on the toolbar.

Viewing the SAS Log and Output If you are using the Enhanced Editor (Windows operating environment), after you submit your program, the program remains in the Enhanced Editor window and the results of your program go into the Log and Output windows. If you are using the Program Editor (all other operating environments) then your results also go into the Log and Output windows, but your program disappears from the Program Editor window. At first it may be a shock for you to see your program disappear in front of your eyes. Don't worry; the program you spent so long writing is not gone forever. If your program produced any output, then you will also get new entries in the Results window. The Results window is like a table of contents for your SAS output and is discussed in more detail in section 1.9. This figure is an example of what your screen might look like after you submit a program from the Enhanced Editor.

You may not see all three of the programming windows (Editor, Log, and Output) at the same time. In some operating environments, the windows are placed one on top of the other. You can bring a window to the top by clicking on it, typing its name in the command line area, or selecting it from the Window menu.

Getting your program back Unfortunately for most of us, our programs do not run perfectly every time. If you have an error in your program, you will most likely want to edit the program and run it again. If you are using the Enhanced Editor, then your program will remain in the window after you submit it. However, if you are using the Program Editor window, you will need to get your program back in the Program Editor window using the RECALL command. You have two choices for executing the RECALL command.



Make the Program Editor the active window, then enter RECALL in the command line area of your SAS session.

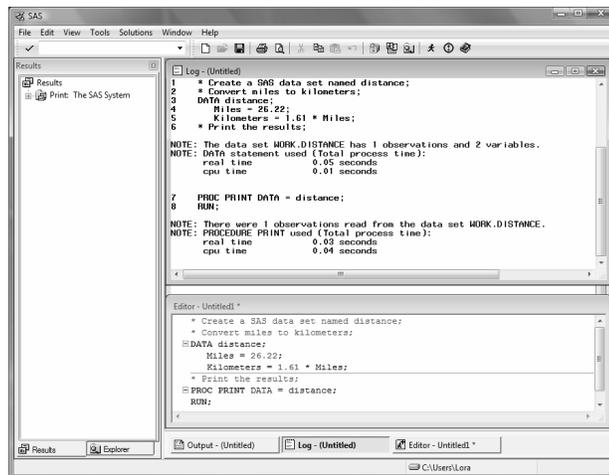
Make the Program Editor the active window, then select Recall Last Submit from the Run pull-down menu.

The RECALL command will bring back the last block of statements you submitted. If you use the RECALL command again, it will insert the block of statements submitted before the last one, and so on and so on, until it retrieves all the statements you submitted.

1.8 Reading the SAS Log

Every time you run a SAS job, SAS writes messages in your log. Many SAS programmers ignore the SAS log and go straight to the output. That's understandable, but dangerous. It is possible—and sooner or later it happens to all of us—to get bogus results that look fine in the output. The only way to know they are bad is to check the SAS log. Just because it runs doesn't mean it's right.

Where to find the SAS log The location of the SAS log varies depending on the operating environment you use, the mode you use (SAS windowing environment, noninteractive, or batch), and local settings. If you submit a program in the windowing environment, you will, by default, see the SAS log in your Log window as in the following figure.



If you submit your program in batch or noninteractive mode, the log will be written to a file that you can view or print using your operating environment's commands for viewing and printing. The name given to the log file is generally some permutation of the name you gave the original program. For example, if you named your SAS program *Marathon.sas*, then it is a good bet that your log file will be *Marathon.log*. At some installations the log and output files are written to a single file, so don't be surprised if you find them together.

What the log contains People tend to think of the SAS log as either a rehash of their program or as just a lot of gibberish. OK, we admit, there is some technical trivia in the SAS log, but there is also plenty of important information. Here is a simple program that converts miles to kilometers and prints the result:

```

* Create a SAS data set named distance;
* Convert miles to kilometers;
DATA distance;
  Miles = 26.22;
  Kilometers = 1.61 * Miles;
RUN;
* Print the results;
PROC PRINT DATA = distance;
RUN;

```

If you run this program, SAS will produce a log similar to this:

```

① NOTE: Copyright (c) 2002-2008 by SAS Institute Inc., Cary, NC, USA.
NOTE: SAS (r) Proprietary Software Version 9.2 (TS1M0)
      Licensed to XYZ Inc., Site 0099999001.
NOTE: This session is executing on the W32_VSPRO platform.

NOTE: SAS initialization used:
      real time          1.40 seconds
      cpu time           0.96 seconds

② 1  * Create a SAS data set named distance;
   2  * Convert miles to kilometers;
   3  DATA distance;
   4      Miles = 26.22;
   5      Kilometers = 1.61 * Miles;
   6  RUN;

③ NOTE: The data set WORK.DISTANCE has 1 observations and 2 variables.
④ NOTE: DATA statement used (Total process time):
      real time          0.03 seconds
      cpu time           0.03 seconds

② 7  * Print the results;
   8  PROC PRINT DATA = distance;
   9  RUN;

NOTE: There were 1 observations read from the data set WORK.DISTANCE
④ NOTE: PROCEDURE PRINT used (Total process time):
      real time          0.01 seconds
      cpu time           0.00 seconds

```

The SAS log above is a blow-by-blow account of how SAS executes the program.

- ① It starts with notes about the version of SAS and your SAS site number.
- ② It contains the original program statements with line numbers added on the left.
- ③ The DATA step is followed by a note containing the name of the SAS data set created (WORK.DISTANCE), and the number of observations (1) and variables (2). A quick glance is enough to assure you that you did not lose any observations or accidentally create a lot of unwanted variables.
- ④ Both DATA and PROC steps produce a note about the computer resources used. At first you probably won't care in the least. But if you run on a multi-user system or have long jobs with large data sets, these statistics may start to pique your interest. If you ever find yourself wondering why your job takes so long to run, a glance at the SAS log will tell you which steps are the culprits.

If there were error messages, they would appear in the log, indicating where SAS got confused and what action it took. You may also find warnings and other types of notes which sometimes indicate errors and other times just provide useful information.

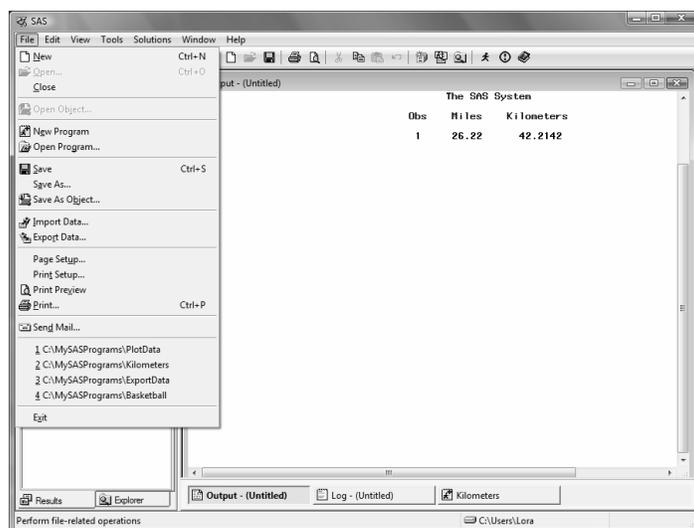
1.9 Viewing Your Results in the Output Window

How you view or print your output depends on how you submit your program. If you submit your program in the SAS windowing environment, then your output will, by default, go to the Output window. If you choose another way to submit your program, either batch or noninteractive, then your output will probably be in a file on your computer. Use your operating environment's commands to view and print the output file (also called the listing). For example, if you execute your SAS program in noninteractive mode on a UNIX system, then your output will be in a file with an extension .lst. To view the file, you can use either the `cat` or `more` commands, and to print the file you would use your system's command for printing files (usually you would type either `lp` or `lpr`).

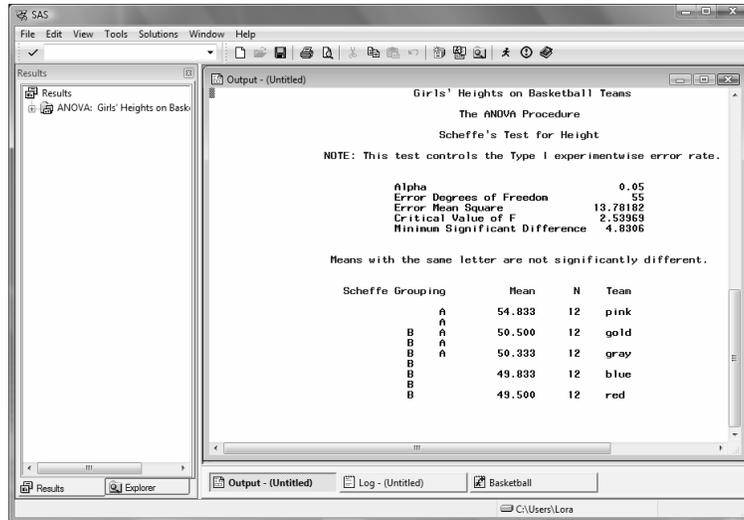
The Output window After submitting your program in the SAS windowing environment, your results will go to the Output window. If you have the SAS Explorer option turned on (some operating environments have this turned on by default, while others do not), then you will also see a listing of the different parts of your output in your Results window. The following figure shows what your Output window might look like after submitting a simple program under Windows.

Printing or saving the contents of the Output window

If you want to print or save the entire contents of the Output window, first make the Output window active by clicking in it, then select either **Print** or **Save As** from the File pull-down menu. If you are not using a personal computer, then your environment may not be set up for printing from within SAS. If you cannot print from within SAS, then save the output to a file and use your system's command for printing files.

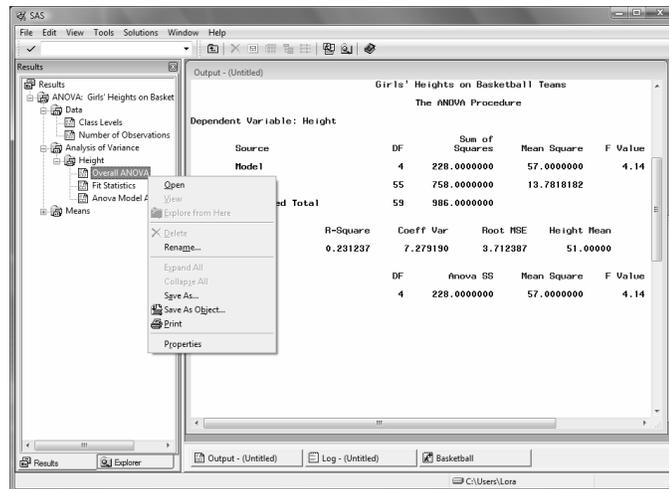


The Results window When you have a lot of output, the Results window can be very helpful. The Results window is like a table of contents for your output. It lists each procedure that produces output, and if you open, or expand, the procedure in the Results tree, you can see each part of the procedure output. The following figure shows what your screen might look like if you ran the ANOVA (Analysis of Variance) procedure.



There is one entry in the Results window for the ANOVA procedure. Notice that in the Output window, you see the end of the procedure's output. If you expand the ANOVA procedure in the results tree, by clicking the plus (+) signs, then you will see all the different parts of the ANOVA output. Double-click the output you want to see, and it will appear at the top of the Output window. The following figure shows what your Output window would look like after you double-click on the Overall ANOVA item in the Results window.

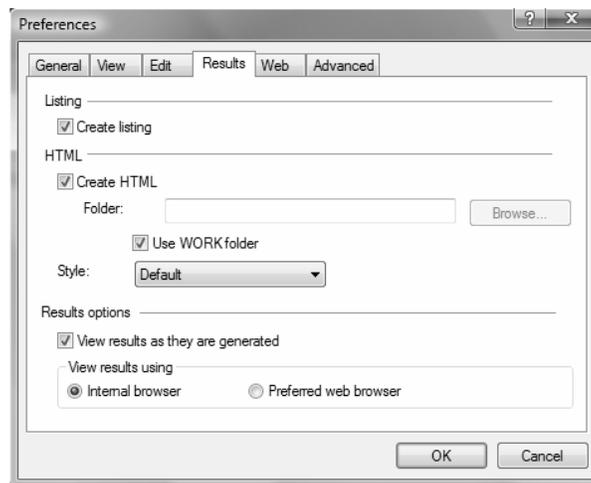
Printing or saving parts of the output Using the Results window, it is possible to print or save just the parts of the output you want. First highlight the item you want in the Results window, then bring up the context-sensitive menu. In the Windows operating environment you do this with the right mouse button; in other operating environments, it may be the middle or right mouse button. Then select either Print or Save As from the pop-up menu. You may also be able to print or save from the File pull-down menu once you highlight the output part you want. If your SAS environment is not set up for printing from within SAS, then save your results to a file and use your operating environment's command for printing files.



1.10 Creating HTML Output

If you are using the SAS windowing environment, then you can create output in Hypertext Markup Language (HTML) format with just a few clicks of your mouse.¹

The Preferences window To turn on HTML output (in Windows, UNIX, or OpenVMS²), select `Options-Preferences` from the Tools menu. This opens the Preferences window. Click the Results tab to bring it to the front. Here is what the Results portion of the Preferences window looks like in Windows.



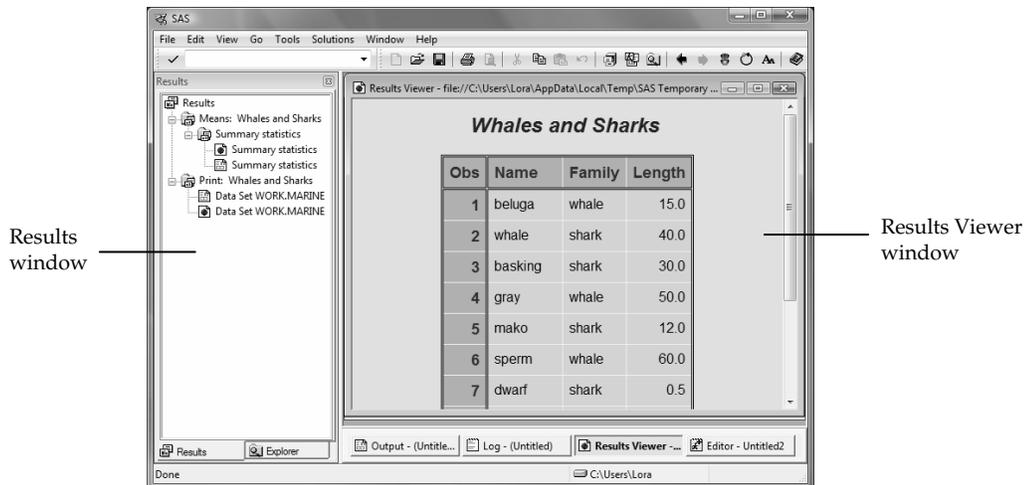
When you first open this window, you will see a check next to `Create Listing`. Listing is the default type of output, and it is what you see in the Output window if you are using the SAS windowing environment, or in the output or listing file if you are running in batch mode. You can turn on HTML output by clicking the box next to `Create HTML`. To turn off the listing or HTML output, just click to un-check it.

In the Preferences window, you can also select a style for HTML output by clicking the arrow next to the Style box and scrolling through the list of styles provided with SAS. When you are done with the Preferences window, click the OK button.

The Results Viewer and Results windows Once you have turned on HTML output, then every time you run a program, your output will automatically appear in the Results Viewer window. The following figure shows what you see after running two simple procedures: MEANS and PRINT. Two windows are showing: the Results Viewer window displaying the HTML output, and the Results window listing all the pieces of output in tree form.

¹ If you are not using the SAS windowing environment, you can still produce HTML output by using ODS statements (see chapter 5).

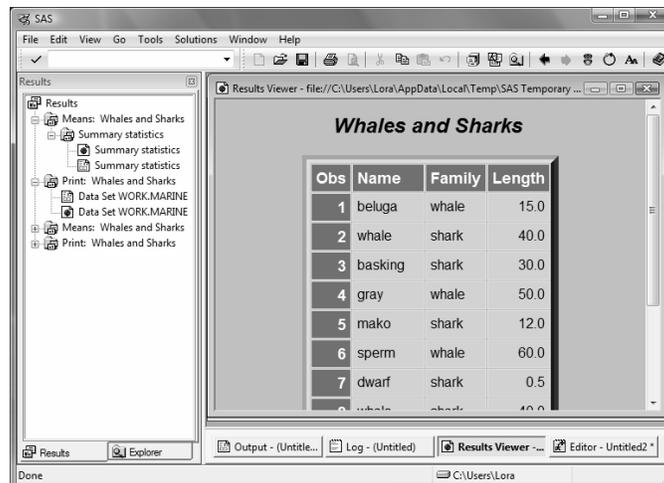
² If you are using z/OS, you will need to modify your registry settings in order to generate HTML interactively. Contact your site's SAS Support Personnel for more information.



The Results Viewer window shows you only one piece of output at a time, but you can tell that SAS ran both procedures by looking at the list in the Results window. You can expand the list by clicking the plus (+) signs, or collapse it by clicking the minus (-) signs. Since both the listing and HTML output were turned on, each procedure produced two pieces of output: one for listing, and one for HTML. You can display any piece of output by double-clicking its name in the Results window.

To save a piece of output in a file, make the Results Viewer window active by clicking it, then click the File menu, and choose *Save As...* To print a piece of output, select *Print* from the File menu.

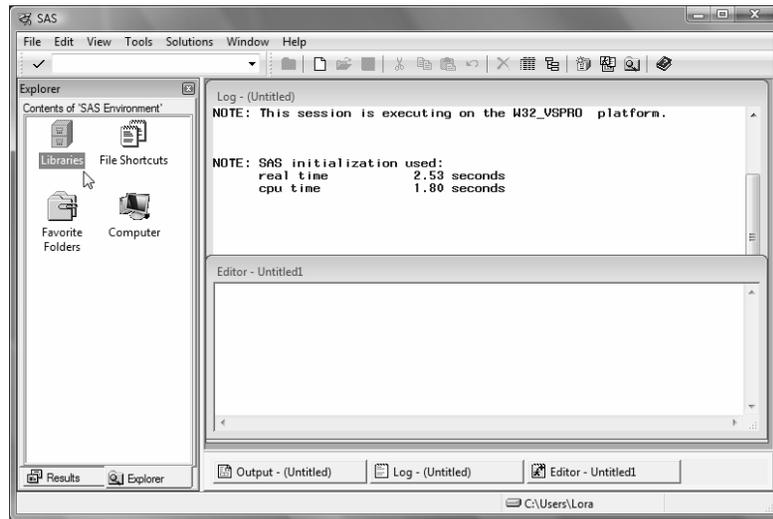
The preceding screen used the DEFAULT style which is the default for HTML output. To see the same output with a different style, just choose a different style in the Preferences window, and re-run your program. Here is the output from the same program using the D3D style.



1.11 SAS Data Libraries

Before you can use a SAS data set, you have to tell SAS where to find it. You do that by setting up a SAS library. A SAS library is simply a location where SAS data sets (as well as other types of SAS files) are stored. Depending on your operating environment, a SAS library might be a folder or directory on your computer, or it might be a physical location like a hard drive, flash drive, or CD. To set up a library, all you have to do is make up a name for your library and tell SAS where it is. There are several ways to do this including using the LIBNAME statement (covered in sections 2.19 to 2.20) and using the New Library window in the SAS windowing environment.

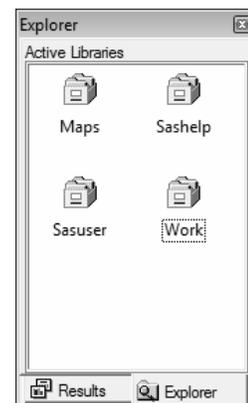
When you start the SAS windowing environment, you see the basic SAS windows including the SAS Explorer window on the left. (If the Explorer window is under the Results window, click its tab to bring it forward.) If you double-click the Libraries icon, Explorer will open the Active Libraries window showing all the libraries that are currently defined. To



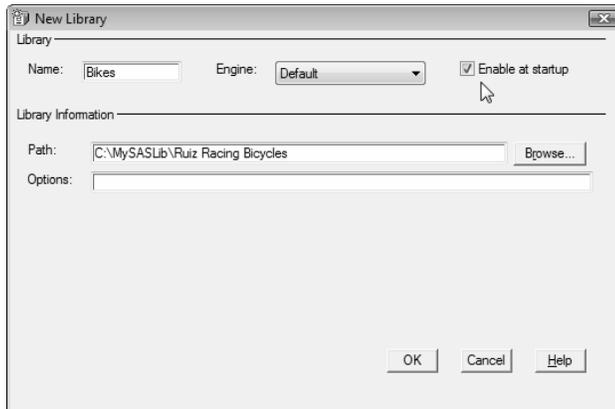
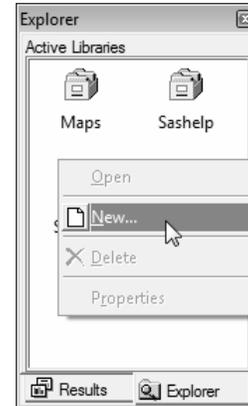
go back to the previous window within Explorer, choose `Up one level` from the View menu,

or click in the Explorer window to make it active and then click the Up One Level button  on the toolbar.

The Active Libraries window When you open the Active Libraries window, you will see at least three libraries: SASHELP, SASUSER, and WORK. You may have other libraries for specific SAS products (such as the MAPS library for SAS/GRAPH software), or libraries that have been set up by you or someone you work with. The SASHELP library contains information that controls your SAS session along with sample SAS data sets. The WORK library is a temporary storage location for SAS data sets. It is also the default library. If you create a SAS data set without specifying a library, SAS will put it in the WORK library, and then delete it when you end your session. If you make changes to the default settings for the SAS windowing environment, this information will be stored in the SASUSER library. You can also store SAS data sets, SAS programs, and other SAS files in the SASUSER library. However, many people prefer to create a new library for their SAS files.



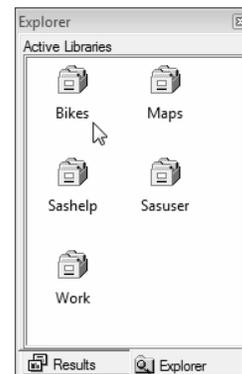
Creating a new library You can create new SAS libraries using the New Library window. To open this window, either left-click in the Active Libraries window (to make it active) and choose **New** from the File menu, or right-click in the Active Libraries window and choose **New** from the pop-up menu.



In the New Library window, type the name of the library you want to create. This name is called a libref which is short for library reference. A libref must be 8 characters or fewer; start with a letter or underscore; and contain only letters, numerals, or underscores. In this window, the name BIKES has been typed in as the libref. In the Path field, enter the complete path to the folder or directory where you want your data sets to be stored, or choose the Browse... button to navigate to the location. If you don't

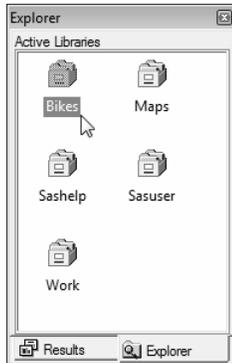
want to define your library reference every time you start up SAS, then check the Enable at startup box. Click OK and then your new library reference will appear in the Active Libraries window.

Here is the Active Libraries window showing the newly created BIKES library.



1.12 Viewing Data Sets in the Viewtable Window

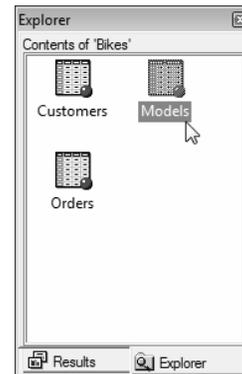
In addition to listing your current libraries and creating new libraries, you can also use SAS Explorer to open SAS data sets for viewing in Viewtable. When you are writing programs it is always a good idea to check the data sets you create to make sure they are correct. Viewtable is one way you can look at your SAS data sets.



Start by double-clicking the Libraries icon in the Explorer window as shown in the previous section. This will open the Active Libraries window showing all the libraries that are currently defined on your system. If you double-click a library icon, SAS will open a Contents window showing you all the SAS files in that particular library.

To go back to the previous window within Explorer, choose **Up one level** from the **View** menu, or click in the Explorer window to make it active and then click the **Up One Level** button  on the toolbar.

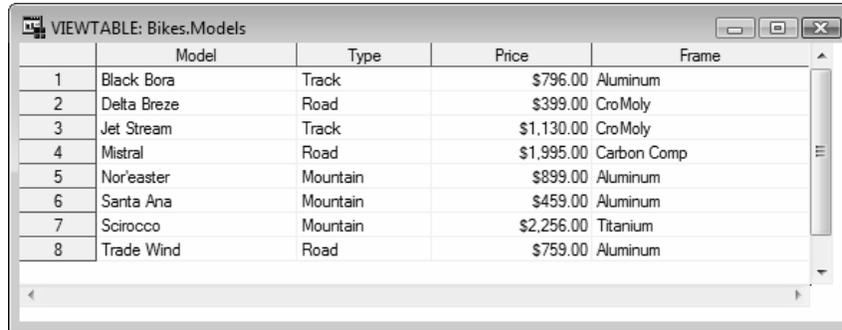
The Contents window This window shows the contents of a library. SAS data sets are represented by an icon showing a little table of data and a red ball, so the library shown on the right contains three data sets named CUSTOMERS, MODELS, and ORDERS. If you double-click a data set, SAS will open a Viewtable window showing that data set. (If you don't yet have any SAS data sets of your own, you can view sample data sets that are provided with SAS in the SASHELP library. The CLASS data set in the SASHELP library is a good one to view.)



The Viewtable window This window allows you to create, browse, and edit data sets. When you first open SAS data sets, the data are in browse mode so you cannot make any changes. To switch to edit mode, select **Edit Mode** from the **Edit** menu. Creating and editing data sets using Viewtable is discussed in more detail in section 2.2. This picture shows the data set named MODELS from the BIKES library.

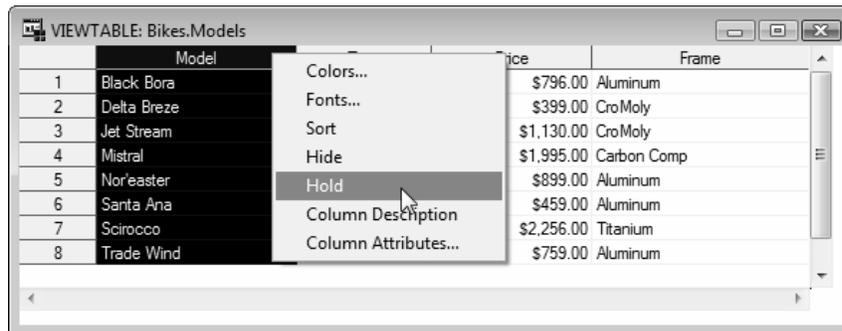
	Model Name	Type of Bicycle	List Price	Frame Composition
1	Black Bora	Track	\$796.00	Aluminum
2	Delta Breze	Road	\$399.00	CroMoly
3	Jet Stream	Track	\$1,130.00	CroMoly
4	Mistral	Road	\$1,995.00	Carbon Comp
5	Nor'easter	Mountain	\$899.00	Aluminum
6	Santa Ana	Mountain	\$459.00	Aluminum
7	Scirocco	Mountain	\$2,256.00	Titanium
8	Trade Wind	Road	\$759.00	Aluminum

Changing column headings By default, Viewtable uses variable labels for column headings or, if a variable does not have a label, the variable name is displayed. Sometimes you may want to see the actual variable names instead of the labels. To do this, click the Viewtable window to make it active, then select **Column Names** from the View menu. Here is the MODELS SAS data set showing the column (also called variable) names instead of the labels.



	Model	Type	Price	Frame
1	Black Bora	Track	\$796.00	Aluminum
2	Delta Breze	Road	\$399.00	CroMoly
3	Jet Stream	Track	\$1,130.00	CroMoly
4	Mistral	Road	\$1,995.00	Carbon Comp
5	Nor'easter	Mountain	\$899.00	Aluminum
6	Santa Ana	Mountain	\$459.00	Aluminum
7	Scirocco	Mountain	\$2,256.00	Titanium
8	Trade Wind	Road	\$759.00	Aluminum

Column options If you right-click a column heading, several options will appear in the pop-up menu. You can control colors, fonts, and view the column attributes. You can choose to sort the data by the values in the column. If you are not in edit mode, then you are given the option of creating a new data set containing the sorted data. You can also hide or hold columns. If you choose to hide a column, the data will not be visible in the current Viewtable session. To unhide a column, select **Hide/Unhide** from the Data menu item to open the Hide/Unhide window. In this window you can change the visibility of all columns. When you choose to hold a column, it and every column to the left of it will always be visible, even when you scroll to the right.



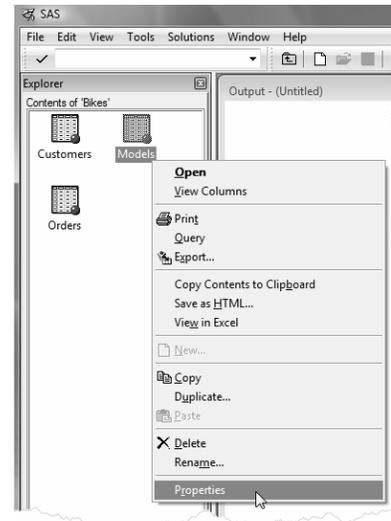
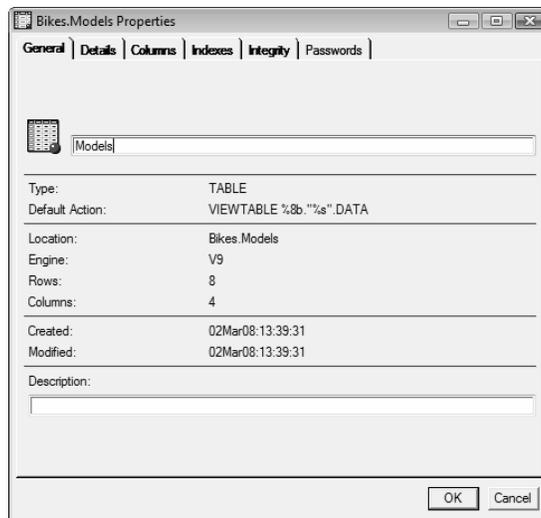
	Model	Type	Price	Frame
1	Black Bora	Track	\$796.00	Aluminum
2	Delta Breze	Road	\$399.00	CroMoly
3	Jet Stream	Track	\$1,130.00	CroMoly
4	Mistral	Road	\$1,995.00	Carbon Comp
5	Nor'easter	Mountain	\$899.00	Aluminum
6	Santa Ana	Mountain	\$459.00	Aluminum
7	Scirocco	Mountain	\$2,256.00	Titanium
8	Trade Wind	Road	\$759.00	Aluminum

- Colors...
- Fonts...
- Sort
- Hide
- Hold
- Column Description
- Column Attributes...

1.13 Viewing the Properties of Data Sets with SAS Explorer

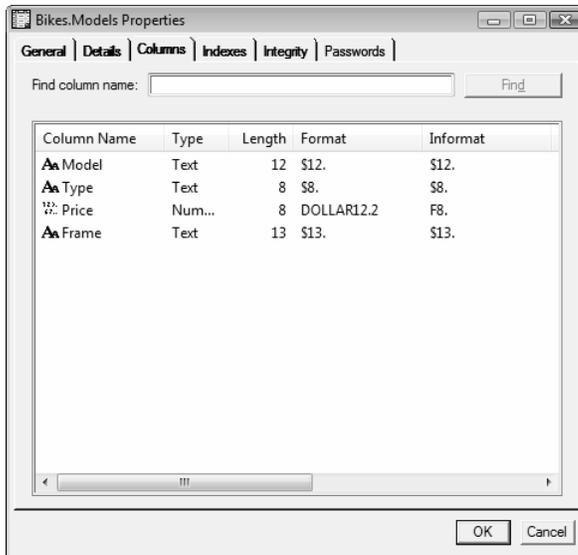
The Properties window for a SAS data set contains some very useful information, such as the date and time the data set was created, the number of observations, all the variable names, and the attributes of the variables. The Properties window contains information similar to the output produced by the CONTENTS procedure described in section 2.22.

Opening the Properties window To open the Properties window, start by double-clicking the Libraries icon in the Explorer window and then double-clicking the library containing the SAS data set. SAS will display the contents of the library in the Explorer window. Right-click the icon for the data set, and select **Properties** from the pop-up menu. This opens the Properties window with the General tab on top.¹

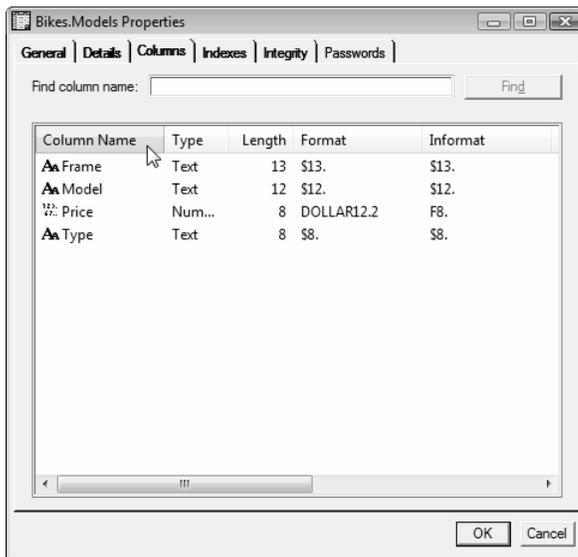


The General tab This window displays information about the data set such as the date it was created and the number of rows (or observations) and columns (or variables).

¹ The Properties windows shown here are from SAS 9.2 in the Windows operating environment. If you are using a different version of SAS, or if you are using a different operating environment, your windows may have a different look.



The Columns tab If you click the Columns tab, SAS displays information about the columns (or variables) in that data set. The variable name, type, and length are displayed along with any formats or informats assigned to the variable. The variable labels are also displayed in this window, but to see them, you need to scroll to the right.



You can also sort any of these columns alphabetically by clicking the column heading, or use the Find feature to look for specific variables. This window shows the variables sorted by name. If you have lots of variables in your data set, using the Sort and Find features can make your work easier.

1.14 Using SAS System Options

System options are parameters you can change that affect SAS—how it works, what the output looks like, how much memory is used, error handling, and a host of other things. SAS makes many assumptions about how you want it to work. This is good. You do not want to specify every little detail each time you use SAS. However, you may not always like the assumptions SAS makes. System options give you a way to change some of these assumptions.

Not all options are available for all operating environments. A list of options specific to your operating environment appears in the SAS Help and Documentation. You can see a list of system options and their current values by opening the SAS System Options window or by using the OPTIONS procedure. To use the OPTIONS procedure, submit the following SAS program and view the results in the SAS log:

```
PROC OPTIONS;
RUN;
```

There are four ways to specify system options. Some options can be specified using only some of these methods. The SAS Help and Documentation for your operating environment tells you which methods are valid for each system option:

1. Your system administrator (this could be you if you are using a PC) can create a SAS configuration file which contains settings for the system options. This file is accessed by SAS every time SAS is started.
2. Specify system options at the time you start up SAS from your system's prompt (called the invocation).
3. Change selected options in the SAS System Options window if you are using the SAS windowing environment.
4. Use the OPTIONS statement as a part of your SAS program.

The methods are listed here in order of increasing precedence; method 2 will override method 1, method 3 will override method 2, and so forth. If you are using the SAS windowing environment, methods 3 and 4, the SAS System Options window and OPTIONS statement, will override each other—so whichever was used last will be in effect. Only the last two methods are covered here. The first two methods are very system dependent; to find out more about these methods see the SAS Help and Documentation for your operating environment.

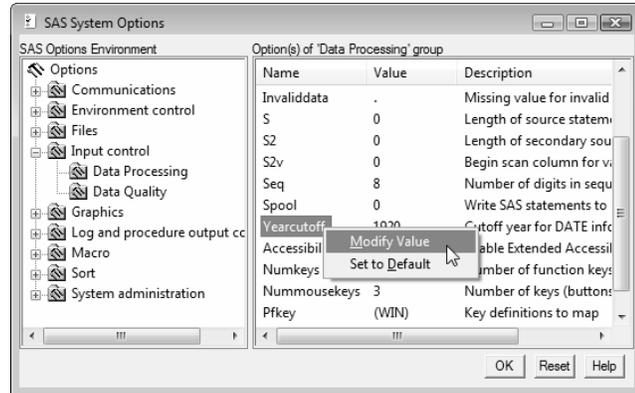
OPTIONS statement The OPTIONS statement is part of a SAS program and affects all steps that follow it. It starts with the keyword OPTIONS and follows with a list of options and their values. For example

```
OPTIONS LINESIZE = 80 NODATE;
```

The OPTIONS statement is one of the special SAS statements which do not belong to either a PROC or a DATA step. This global statement can appear anywhere in your SAS program, but it usually makes the most sense to let it be the first line in your program. This way you can easily see which options are in effect. If the OPTIONS statement is in a DATA or PROC step, then it affects that step and the following steps. Any subsequent OPTIONS statements in a program override previous ones.

The SAS System Options

window You can view and change SAS system options through the SAS System Options window. Open it either by typing `OPTIONS` in the command line area on your screen, or by selecting it from the Tools pull-down menu. To change the value of an option, first locate the option by clicking on the appropriate category on the left side of the screen. A list of options and their current values will appear on the right side of the screen. Right-click the option itself to modify the value or set it to the default.



Common options The following are some common system options you might want to use:

`CENTER` | `NOCENTER`

Controls whether output is centered or left-justified.
Default: `CENTER`.

`DATE` | `NODATE`

Controls whether or not today's date will appear at the top of each page of output. Default: `DATE`.

`LINESIZE = n`

Controls the maximum length of output lines.
Possible values for n are 64 to 256. Default varies.

`NUMBER` | `NONUMBER`

Controls whether or not page numbers appear on each page of SAS output. Default: `NUMBER`.

`ORIENTATION = PORTRAIT`
`ORIENTATION = LANDSCAPE`

Specifies the orientation for printing output.
Default: `PORTRAIT`

`PAGENO = n`

Starts numbering output pages with n . Default is 1.

`PAGESIZE = n`

Controls the maximum number of lines per page of output.
Possible values for n are 15 to 32767. Default varies.

`RIGHTMARGIN = n`

`LEFTMARGIN = n`

`TOPMARGIN = n`

`BOTTOMMARGIN = n`

Specifies size of margin (such as 0.75in or 2cm) to be used for printing output. Default: 0.00in.

`YEARCUTOFF = yyyy`

Specifies the first year in a hundred-year span for interpreting two-digit dates. Default: 1920.

