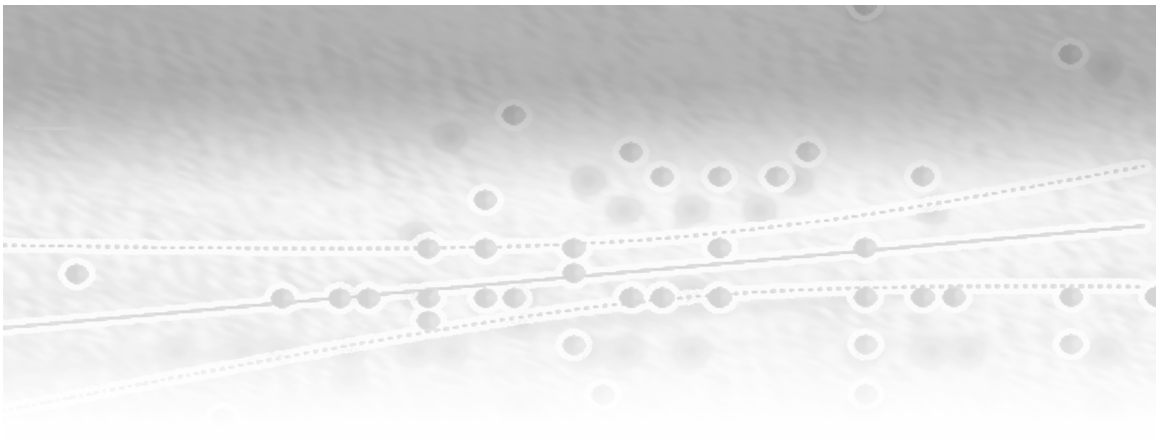


Chapter 1

Introduction to SAS Enterprise Guide

1.1 What Is SAS Enterprise Guide?

SAS is one of the best known and most widely used statistical packages in the world. Although it actually covers much more than statistical analysis, that is the focus of this book. Analyses using SAS are conducted by writing a program in the SAS language, running the program, and inspecting the results. Using SAS requires both a knowledge of programming concepts in general and of the SAS language in particular. One also needs to know what to do when things don't go smoothly; i.e., knowing about error messages, their meanings, and solutions.

SAS Enterprise Guide is a Windows interface to SAS whereby statistical analyses can be specified and run using normal windowing point-and-click style operations and hence without the need for programming or any knowledge of the SAS programming language. As such, SAS Enterprise Guide is ideal for those who wish to use SAS to analyze their data, but do not have the time, or perhaps inclination, to undertake the considerable amount of learning involved in the programming approach. For example, those who have used SAS in the past, but are a bit "rusty" in their programming, may prefer SAS Enterprise Guide. Then again, those who would like to become proficient SAS programmers could start with SAS Enterprise Guide and examine the programs it produces.

It should be born in mind that SAS Enterprise Guide is not an alternative to SAS; rather, it is an *addition* which allows an alternative way of working. SAS itself needs to be present or at least available. The need for SAS to be present is because SAS Enterprise Guide works by translating the user's point-and-click operations into a SAS program. SAS Enterprise Guide then uses SAS to run that program and captures the output for the user.

The computer on which SAS runs is referred to as the *SAS Server*. Usually the SAS Server will be the same computer, referred to as the *Local Computer*, but need not be. We assume that both SAS and SAS Enterprise Guide will have already been set up. The

examples in this book were produced using SAS Enterprise Guide 4.1 and SAS 9.1 under Windows XP Professional. There are some notable differences between version 4.1 and earlier versions, so we would encourage users of earlier versions to upgrade. Such upgrades are available from your local SAS office.

1.2 Using This Book

We assume readers are familiar with the basic operation of Windows and Windows programs; for example, we will use the terms: click, right-click, double-click, and drag to refer to the usual mouse operations without further comment. The description of how to perform a task within SAS Enterprise Guide will usually begin from one of the main menus and typically comprise a sequence of selections from there. For instance, the **File** menu contains the usual **Open** option within it, the use of which leads to a submenu of the kinds of things that can be opened, one of which is **Data**. We abbreviate this sequence to **File➢Open➢Data**. When it seems natural we may extend the sequence to options within the windows that open as a result of the menu selection. Thus, the window that opens following the above sequence (shown in Display 1.5) has two options: **Local Computer** and **SAS Servers**, so the sequence might be extended to **File➢Open➢Data➢Local Computer**. We use the bold, sans-serif font both to distinguish text that appears on screen and forms part of the operation of SAS Enterprise Guide and to distinguish the names of data sets and variables from ordinary text.

Many of our instructions assume that the downloadable files and data sets that accompany this book have been placed in the directory **c:\saseg** and its subdirectories **data** and **sasdata**. If they have been placed elsewhere, the instructions will need to be amended accordingly.

This introductory chapter includes numerous screenshots, whereas subsequent chapters use fewer and rely on the more concise sequences of instructions. It is assumed that the reader will have downloaded the data and will be able to follow the instructions on screen.

In the production of this book, we have altered several settings from their defaults. Readers may wish to use the same settings for comparability between the results shown here and their own results and they can do this, by first make sure settings are at their defaults, by selecting **Tools➢Options➢Reset All**.

Then make the follow changes:

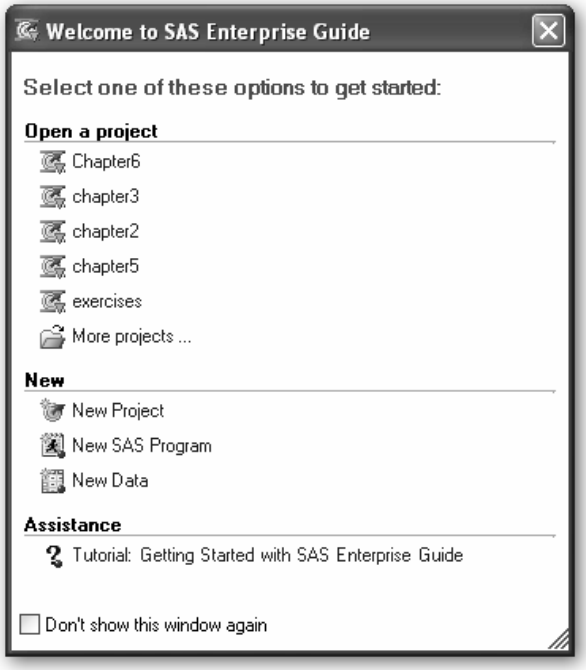
- **Tools➢Options➢Results➢General,** select **RTF** and deselect **HTML**. Click **OK**.
- **Tools➢Options➢Results➢RTF**, select **Theme** as the **Style**. Click **OK**.

- **Tools➢Options➢Tasks➢Tasks General**, delete the **Default footnote text for task output**, and deselect **Include SAS procedure title in results**. Click **OK**.
- **Tools➢Options➢Query**, select the option to **Automatically add columns from input tables to result set of query**. Click **OK**.

1.3 The SAS Enterprise Guide Interface

When SAS Enterprise Guide starts, it first attempts to connect to SAS servers that it knows about. In most cases, connecting to SAS servers simply means that it finds that SAS is installed on the same computer. SAS Enterprise Guide then offers to open one of the projects that have recently been opened or to create a new project as shown in Display 1.1.

Display 1.1 Welcome Screen



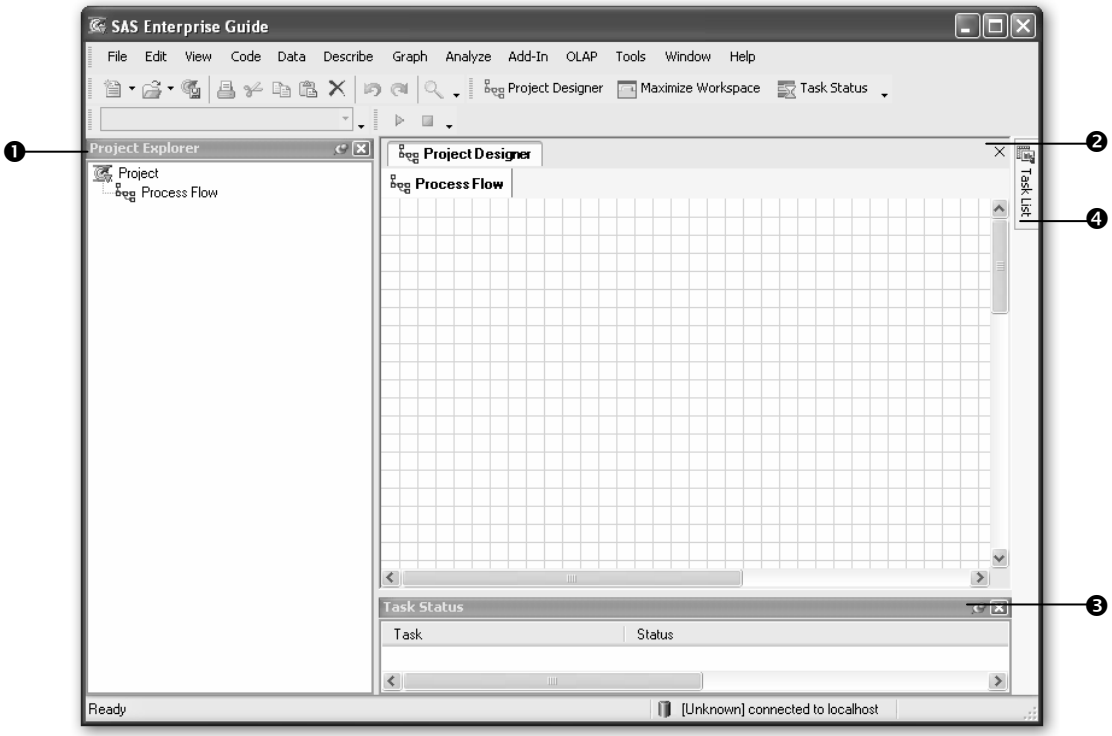
1.3.1 SAS Enterprise Guide Projects

A *project* is the way in which SAS Enterprise Guide stores statistical analyses and their results: it records which data sets were used, what analyses were run, and what the results were. It can also record the user's own notes on what they did and why. In the same way that a word processor loads and saves documents, so SAS Enterprise Guide does with projects. Thus, a project is a piece of statistical analysis in the same way that a document is a piece of writing. In terms of scope, a project might be the user's approach to answering one particular question of interest. It should not be so large or diffuse that it becomes difficult to manage.

1.3.2 The User Interface

The default user interface for SAS Enterprise Guide 4.1 is shown in Display 1.2.

Display 1.2 SAS Enterprise Guide User Interface



The most familiar elements of the interface are the menu bar and toolbar at the top of the window. There are four windows open and visible:

❶ the Project Explorer window

❷ the Project Designer window

❸ the Task Status window

❹ the Task List window

Moving the cursor over the task list causes the task list to scroll to the right.

For the vast majority of the examples in this book, we use only the menus and the Project Designer window. In this way the reader can safely ignore other elements of the interface, or even close them. We give a brief description of them, for completeness sake.

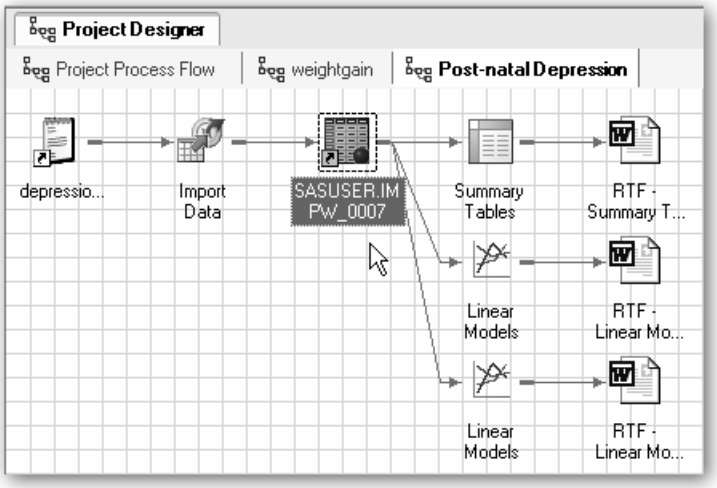
Toolbar and Task List	offer alternative, sometimes quicker, ways to access features of SAS Enterprise Guide.
Task Status window	shows what is happening while SAS Enterprise Guide is using SAS to run a program.
Project Explorer window	offers an alternative view of the project to that presented in the Project Designer window. It tends to show more detail, which can be useful in some cases.

1.3.3 The Process Flow

Within the Project Designer window, we can see an element labeled **Process Flow**, which is another concept central to SAS Enterprise Guide. Essentially, a process flow is a diagram consisting of icons that represent data sets, tasks, and outputs with arrows joining them to indicate how they relate to each other. The general term *tasks* includes not only statistical analyses but data manipulation.

We will begin with some examples of process flow diagrams to give an overview before describing the individual elements in more detail. An example of a Project Designer window is shown in Display 1.3.

Display 1.3 An Example of a Project Designer Window



The first thing to note about this example is that the Project Designer window actually contains three process flows, identified by tabs at the top of the window:

- Project Process Flow (the default name)
- weightgain
- Post-natal Depression

To make a process flow active and bring it to the front, click on the tab. In this case, the Post-natal Depression process flow is the active one, and the title on the tab is bold to indicate that this is the case.

The first three icons in Display 1.3 represent the process of importing some data into a SAS data set. The Import Data task has as its input a raw data file, depressionIQ (**depressio...**), and as its output a SAS data set. The full name of the raw data file is not visible in the process flow; if the cursor is held over the icon, a window pops up with more details, including the full name, path, and location (i.e., which computer it is on). The SAS data set has been automatically given the somewhat arbitrary name **SASUSER.IMPW_0007**. The relationship of a task to its input and output is represented primarily by the arrows, but also by the ordering from left to right—input to the left of the task and output to the right of the task.

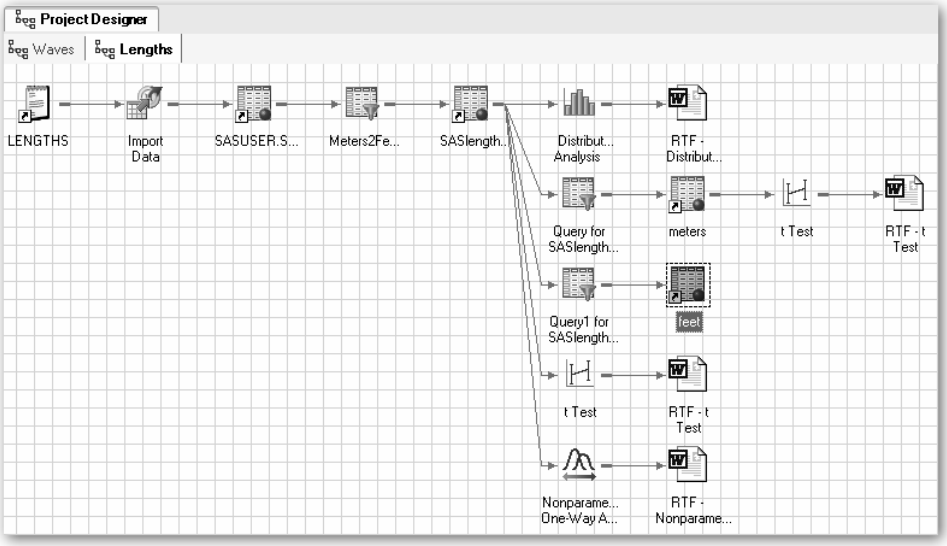
On the right-hand side of the process flow diagram, we can see that the SAS data set is used as input to three tasks: a **Summary Tables** task and two **Linear Models** tasks. The output from each task is an RTF (rich text format) document containing the results. RTF is one of the formats that can be chosen for output and is one particularly suited for reading into a word processor.

1.3.4 The Active Data Set

Two important things to note about Display 1.3 are that the icon for the SAS data set has a dashed line around it and its label is highlighted. The dashed line indicates that the SAS data set has been selected (clicked), and this makes it the active data set. If there are multiple data sets in a project, any tasks selected from the menus will apply to the active data set. It is therefore important to be aware of which data set is active and of how to make a data set active. Each type of object and task in the process flow has its own icon, and a SAS data set can be recognized by the icon (the grid with the red ball in the bottom right corner).

A second example, shown in Display 1.4, contains four SAS data sets. The first data set results from importing some raw data from a file named **LENGTHS**, and the other data sets are derived from it. Generating other data sets is a common situation, where there is an original data set and one or more different versions arise from some modification of the original data. The **feet** data set is the active data set, so any analysis chosen from the menus would apply to that data set.

Display 1.4 A Process Flow Containing Multiple SAS Data Sets



Any of the icons in a process flow diagram can be opened by double-clicking them or right-clicking, and selecting **Open**. For a file, data set, or output, the contents can then be examined, printed, or copied. For a task, the settings can be examined, changed if required, and the task re-run. When a task is re-run, there is the option to replace the output from the previous run or generate new output, keeping the previous version. If the Replace option is taken, a new task icon and output icon will appear in the process flow.

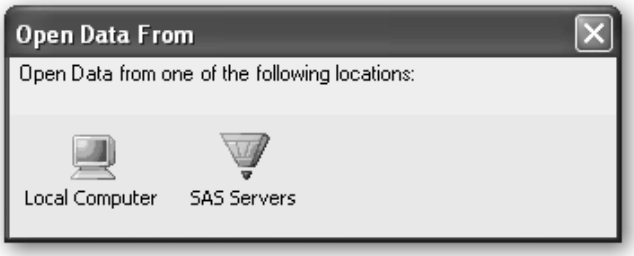
1.4 Creating a Project

The first step in a project is adding the data. In order to be analyzed, data must be in the form of a SAS data set. Data in other formats will need to be converted or imported into a SAS data set. In many cases, the conversion or importation will have already been done.

1.4.1 Opening a SAS Data Set

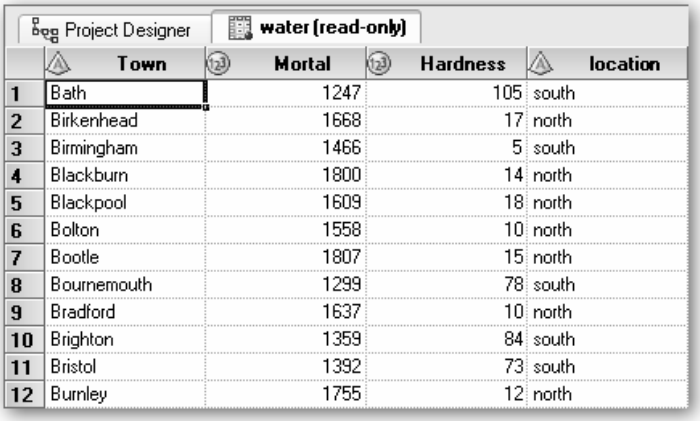
To add a SAS data set to a project, select **File➢Open➢Data**. A window like that shown in Display 1.5 will then appear, prompting a location from which to open the data. **Local Computer** is the user's own computer where SAS Enterprise Guide is being used. **Local Computer** would also be the location for data stored on a network file server mapped to a local drive letter. For example, if the user had data stored on a network drive N: that would also count as stored on the local computer. The alternative, **SAS Servers**, refers to remote computers that have SAS installed and hold SAS data sets. All of the examples in this book use data stored on the local C: drive.

Display 1.5 Data Location Pop Up Window



Having selected **Local Computer** or a **SAS Servers**, browse to the location of the SAS data set, select it, and click **Open**. In our examples, SAS data sets are stored in the directory **c:\saseg\sasdata**. SAS data sets created with version 7 of SAS or a later version have the extension .sas7bdat. Data sets created by earlier versions of SAS are most likely to have the extension .sd2. The SAS data set **water.sas7bdat** contains measures of water hardness and mortality rates for 61 towns in England and Wales. Open that data set and the contents of the data set can then be viewed on screen as shown in Display 1.6.

Display 1.6 The Water Data Set Opened



Project Designer | water (read-only)

	Town	Mortal	Hardness	location
1	Bath	1247	105	south
2	Birkenhead	1668	17	north
3	Birmingham	1466	5	south
4	Blackburn	1800	14	north
5	Blackpool	1609	18	north
6	Bolton	1558	10	north
7	Bootle	1807	15	north
8	Bournemouth	1299	78	south
9	Bradford	1637	10	north
10	Brighton	1359	84	south
11	Bristol	1392	73	south
12	Burnley	1755	12	north

Closing the data set, we see that a SAS data set icon, labeled **water**, has been added to the process flow.

1.4.2 Importing Data

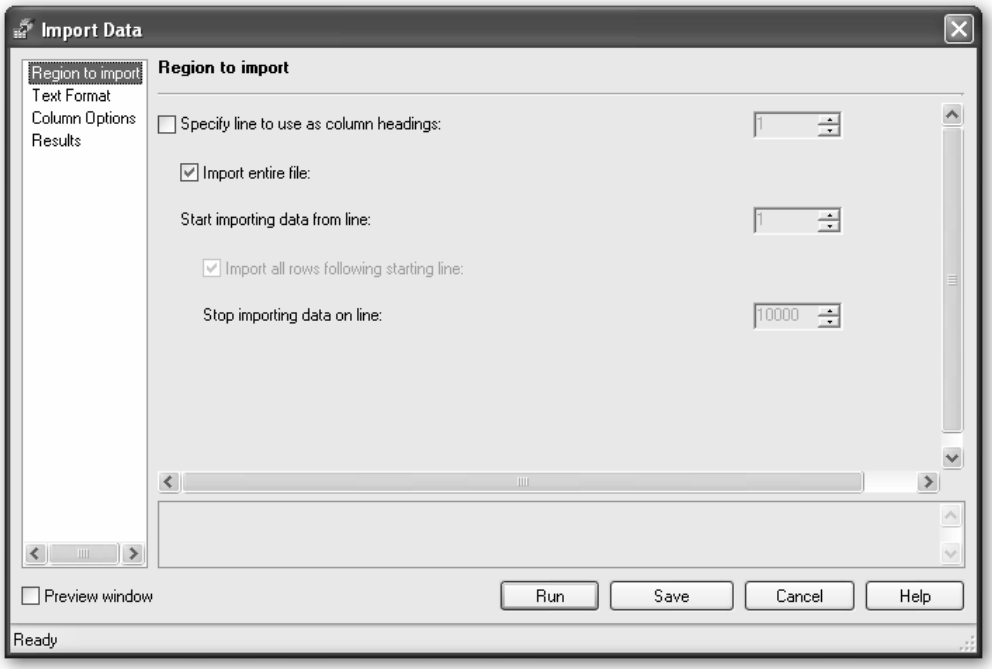
If the data to be analyzed are not already available as a SAS data set, they need to be imported into one, using the Import Data task. We begin with examples of importing raw data files, which are also referred to as text files or ASCII files. Such files contain only the printable characters plus spaces, tabs, and end-of-line characters. The files produced by database programs and spreadsheets are not normally in this format, although the programs usually have an export facility to create raw data files.

The data in a raw data file may be fixed width or delimited. With fixed-width data, the values for each variable are in prespecified columns. With delimited data, the data values are separated by a special character—usually a space, tab, or comma. Tab-separated files and comma-separated files are very common formats. Comma-separated data are sometimes referred to as *comma-separated values* and given the extension .csv. Delimited files may also contain the names of the variables, usually as the first line of the file, with the names separated by the same delimiter as the data values.

There are examples of importing both tab- and comma-delimited data, with and without the variable names, in later chapters (see the index). Here, we illustrate the use of the Import Data task with fixed-width data. The water.dat file contains a slightly different version of the data already available in the SAS data set of the same name. To import them, select **File➢Import Data**.

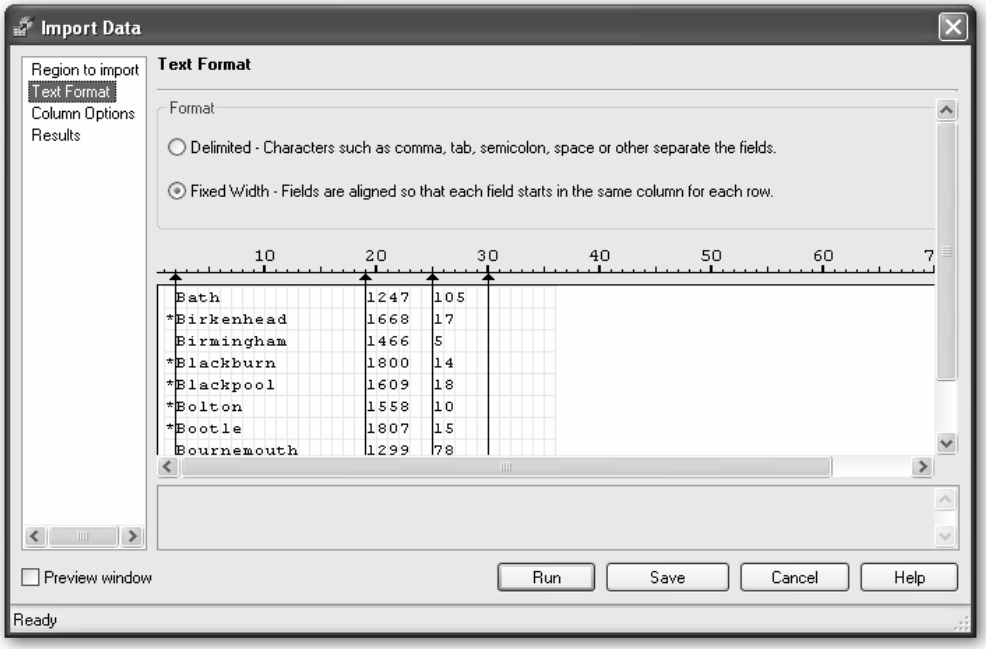
The Import Data task, as with most tasks, consists of a number of panes, each of which allows a set of options to be specified. The initial view is shown in Display 1.7.

Display 1.7 Import Data Task Opening Screen



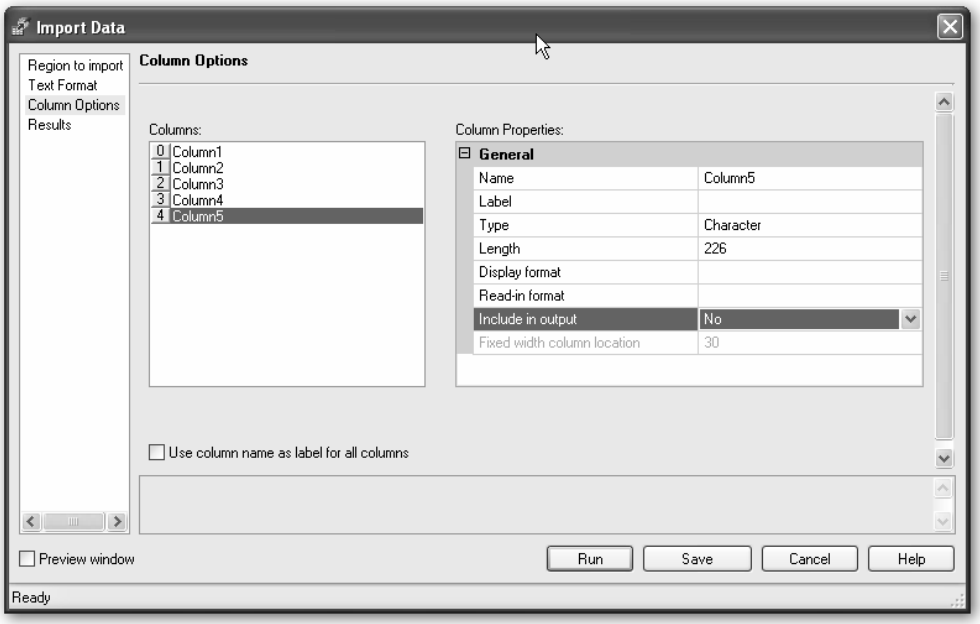
The first pane, **Region to import**, is displayed. Other panes, listed in the left side of the window, are: **Text Format**, **Column Options**, and **Results**. In the **Region to import** pane, **Import entire file** is the default. The option to **Specify line to use as column headings** is for delimited files where the variable names are included in the file, usually in line 1. Hence, 1 is the default value if the option is selected. The **Text Format** pane allows the format to be specified as **Fixed Width** or **Delimited** and, if delimited, what delimiter is used. The default is comma-delimited. Display 1.8 shows the result of selecting **Fixed Width** format with this data file.

Display 1.8 Text Format Pane for Water Data



The pane shows the beginning of the file with a ruler above to indicate which columns the data values are in. Clicking on the ruler specifies where the data fields begin and end. We have put the separators at columns 2, 19, 25, and 30. The Column Options pane is shown in Display 1.9.

Display 1.9 Column Options Pane for Water Data



We see first that five rather than four columns have been defined. Column 5 is the blank remainder of the line after the final delimiter, so we have set the **Include in output** option to **No**. In the pane shown in Display 1.9, we can also give the variables (or columns) more meaningful names. Select **Name** under Column Properties and type a new name. Rename columns 1 to 4 as **flag**, **town**, **Mortality**, and **hardness**, respectively. (We deselected the option to **Use column names as label for all columns** to avoid having to retype these labels as well.)

We also check that other properties of the columns have been correctly assigned. In fact, **Mortality** and **hardness** have been treated as character variables when they should be numeric, but we can change the variable type using the **Type** option under Column Properties.

The final **Results** pane allows the SAS data set being created to be renamed and stored in a particular location. In this case, we leave the default settings and run the task. Display 1.10 shows the results, which are similar to the results shown previously in Display 1.6. The data set has been given an arbitrary name, **SASUSER.IMPW_000A**. At this point, we should scroll through the data to make sure it has all been imported correctly. Having done that, we would close the **water** data set as its contents are in front of the process flow. We could click on the process flow tab (labeled **Project Designer**)

to bring it to the front, but it keeps the workspace tidier if we close data sets and output after we have viewed them.

Display 1.10 Imported Version of Water Data

Project Designer | SASUSER.IMPW_000A (read-only)

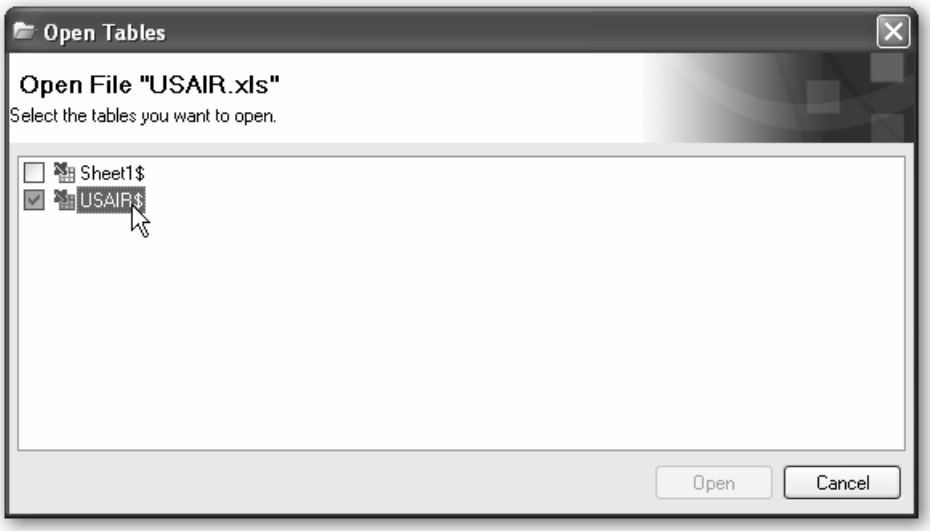
	flag	town	Mortality	hardness
1		Bath	1247	105
2	*	Birkenhead	1668	17
3		Birmingham	1466	5
4	*	Blackburn	1800	14
5	*	Blackpool	1609	18
6	*	Bolton	1558	10
7	*	Bootle	1807	15
8		Bournemouth	1299	78
9	*	Bradford	1637	10
10		Brighton	1359	84
11		Bristol	1392	73
12	*	Burnley	1755	12

In addition to being able to import data from text files, SAS Enterprise Guide can also import data from several popular Windows programs such as Microsoft Excel and Microsoft Access. As a simple example, the file **c:\saseg\data\usair.xls** contains a Microsoft Excel workbook with some data on air pollution in the USA. The data are described more fully in Chapter 6 (Exercise 6.4) but need not concern us here. To import the data:

1. Select **File➢Import Data➢Local Computer**.
2. Browse to **c:\saseg\data**.
3. Select **usair.xls** and **Open**. Because the file contains more than one worksheet and only one can be imported at a time, a window like that in Display 1.11 pops up to select the worksheet to use.
4. Select **USAIR** and then **Open**. The worksheet contains the variable names in the first row. SAS Enterprise Guide has recognized this and set the options under **Region to import** and **Column Options** appropriately, so no changes are needed.
5. Run the task. It is worth noting that the ease of importing the data is due to the fact that the spreadsheet contains only the variable names and the data values. It would be simpler again if the file contained only a single worksheet.

Importing a data table from an Access database would be very similar. It may also be possible to open or import data (**File➢Open➢Data** or **File➢Import Data**) from other proprietory databases, if the appropriate component of SAS (a module of SAS/ACCESS) has been licensed for the computer running SAS.

Display 1.11 Table Selection Window



1.5 Modifying Data

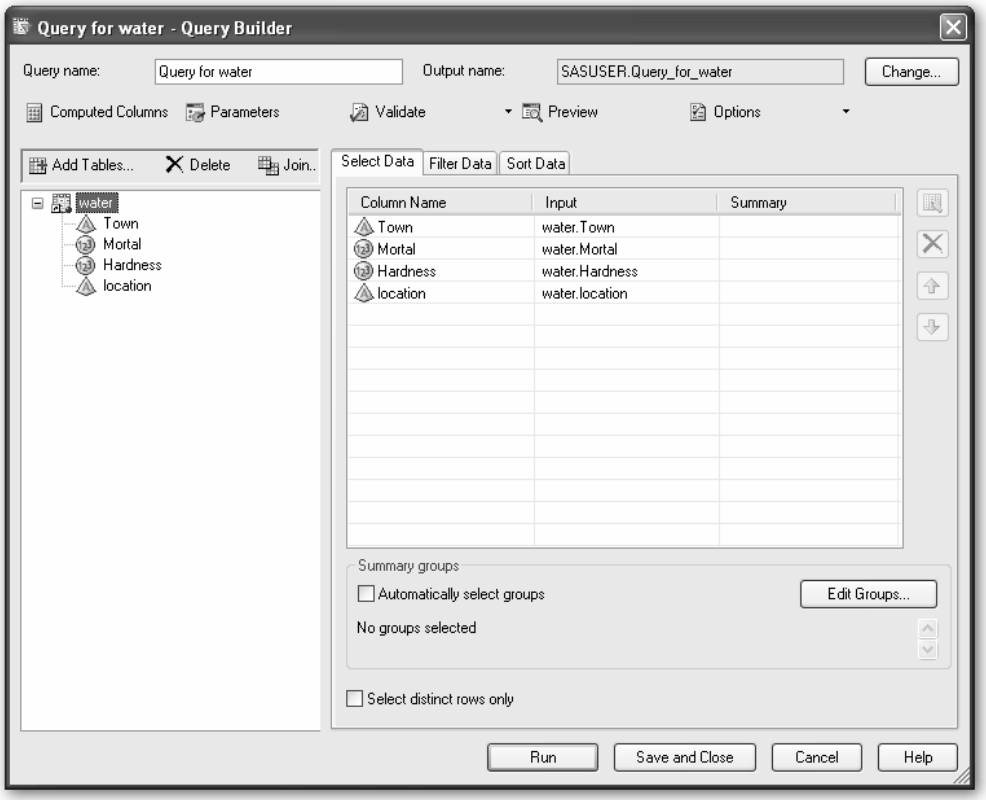
After adding data to a project, it may be necessary to modify the data before it is ready to be analyzed. The Filter and Query task can be used to modify a SAS data set in a variety of ways.

1.5.1 Modifying Variables: Using Queries

We begin with an example of creating a new variable from an existing variable. One common reason for creating a new variable is when a transform of an existing variable is considered necessary. The **hardness** variable in the **water** data set is somewhat skewed, so a log transformation might be appropriate.

1. Click on the **water** data set to make it active. There are two icons in the process flow both named water. The SAS data set that we wish to use is distinguished by its icon—the text file of the same name has a notepad icon. They can also be distinguished by holding the cursor over them, which reveals additional details of each.
2. Select the SAS data set.
3. Select **Data➢Filter and Query**. The opening screen should look like Display 1.12.

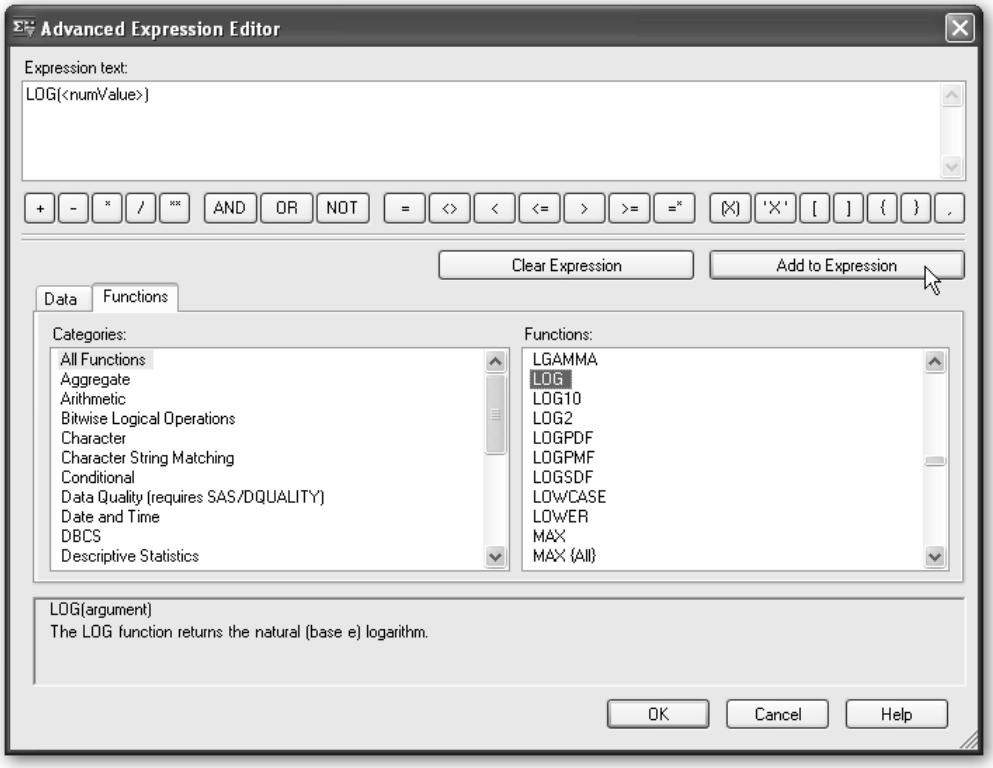
Display 1.12 Query Builder Window



The four variables in the input data set also appear in the **Select Data** pane because we have set the option to **Automatically add columns from input tables to result set of query** under **Tools➢Options➢Query**. Otherwise, variables from the input data set would need to be dragged across. It is worth noting in passing that the variables have icons that indicate whether they are character or numeric.

4. To create a new variable, select **Computed Columns➢New➢Build Expression**. This brings up the **Advanced Expression Editor** window as shown in Display 1.13.

Display 1.13 Advanced Expression Editor



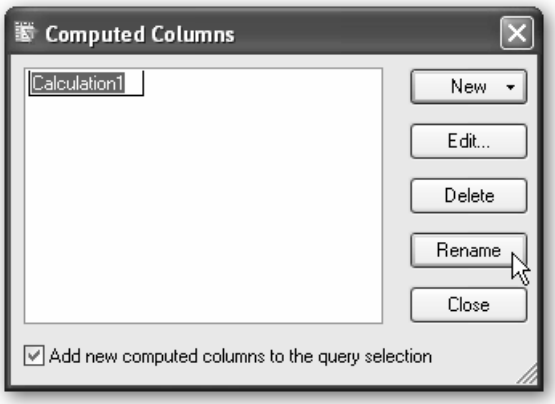
The expression text specifies how the new variable is to be calculated. It can either be typed into the pane or constructed using the buttons and menus. Selecting the **Functions** tab shows a list of function categories with **All Functions** as the default. The right hand pane shows the functions by name, with a brief description of the highlighted function below.

5. Scroll down this list, click on **LOG** and **Add to Expression**. **LOG(<numValue>)** appears in the expression text. The **<numValue>** part indicates that the log function takes a numeric argument.

6. Because we want the log of the **hardness** variable, replace **<numValue>** with **hardness** either by simply typing **hardness** in or by using the Data tab. If the Data tab is used, the variable name will be prefixed with the name of the data set.

7. Clicking **OK** returns us to the Computed Columns window as shown in Display 1.14. The new variable is simply called **Calculation1**, by default, but can be renamed by selecting it, clicking **Rename**, and typing in a more meaningful name, such as **loghardness**.

Display 1.14 Computed Columns Window



Running the task adds an icon for the query and a new SAS data set to the process flow. The new data set contains the **loghardness** variable in addition to the original four variables.

1.5.2 Recoding Variables

Another common modification is to classify a continuous variable like **hardness** into a number of groups. Rather than create another Filter and Query task, we can re-open the existing one and add to that.

1. Open the task by double-clicking on its icon, or by **right-click➢Open**.
2. Select **Computed Columns➢New➢Recode a column**.
3. Select **hardness** and **Continue**. The **Recode Column** window opens.
4. Click on the **Add** button.
5. Select the **Replace a range** tab.
6. Use these to replace the ranges 0–15 with 1, 16–60 with 2, and 61–138 with 3. The actual values of **hardness** contained in the data are available to view via the drop-down boxes for the start and end of the ranges. The **Recode Column** window

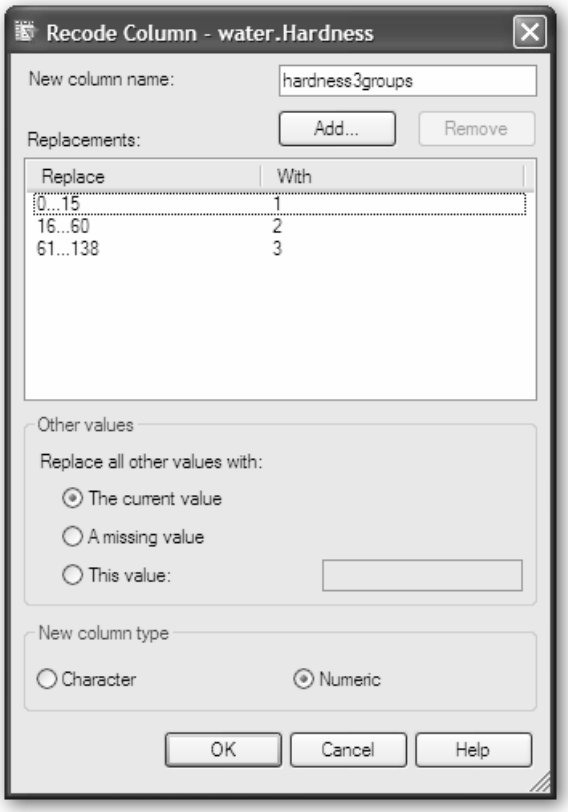
should now look like Display 1.15. Change the **New column name** to **hardness3groups** as shown.

7. Click **OK**, **Close**, and **Run**.

8. Reply **Yes** to **Would you like to replace the results from the previous run?** The **Recode Column** option within the Filter and Query task can also be used to reduce the number of categories a categorical variable has, for instance when combining categories which have too few members in. Such recoding can be done with both numeric and character variables.

   Including multiple data modifications in the one Filter and Query task helps to keep the process flow diagrams simple and clear.

Display 1.15 Recode Column Window



To modify the value of a variable for some observations and not others, or to make different modifications for different groups of observations, use the Advanced Expression Editor to build a query with a conditional function. A simple example is given in Chapter 2, Section 2.3.1.

1.5.3 Splitting Data Sets: Using Filters

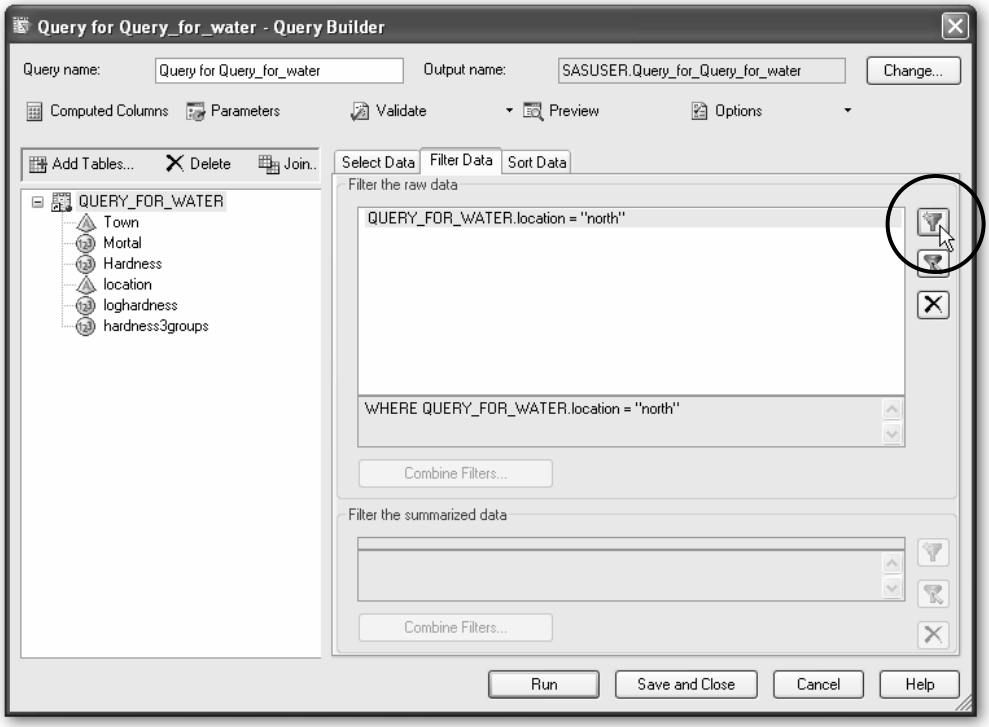
So far we have looked at using the Filter and Query task to create and modify the values of variables and we used queries for the purpose. We now turn to the use of filters to produce subsets of the observations in a data set. We might want to form a subset of the observations in order to discard observations that have errors, or because we wish to focus our analysis on one particular group of observations. Take the **water** data set as an example where we want to look only at the northerly towns. Normally we would want to include the newly derived variables, and so we would use the data set calculated with the query described above.

1. Click on the **water** data set to make it the active data set.
2. Select **Data➢Filter and Query**.
3. Click on the **Filter Data** tab.
4. **Location** is the variable we want to filter on, so we drag and drop that into the **Filter Data** pane. The **Edit Filter** window pops up.
5. The value of location that we want to select is **north**. We could simply type that into the value box, but it would be safer to use the drop-down button and select **Get Values**.

The reason for preferring **Get Values** is that filters which use character variables are case sensitive: **North** is different from **north**, so if both occurred in the data set, the filter would need to include both. Using **Get Values** would give us the correct spelling and case as well as alerting us to any misspellings that there might be in the data set.

In our example here, the situation is straightforward and the Query Builder window should look like Display 1.16. A more complex filter can be constructed by clicking the new filter button (circled in Display 1.16) and selecting **New Advanced Filter**, which brings up the Advanced Expression Editor seen earlier. Another example of using filters to split the data set for separate analyses is given in Chapter 2, Section 2.2.2, and the process flow is reproduced in Display 1.4 above.

Display 1.16 Query Builder Window Filtering the Water Data Set



1.5.4 Concatenating and Merging Data Sets: Appends and Joins

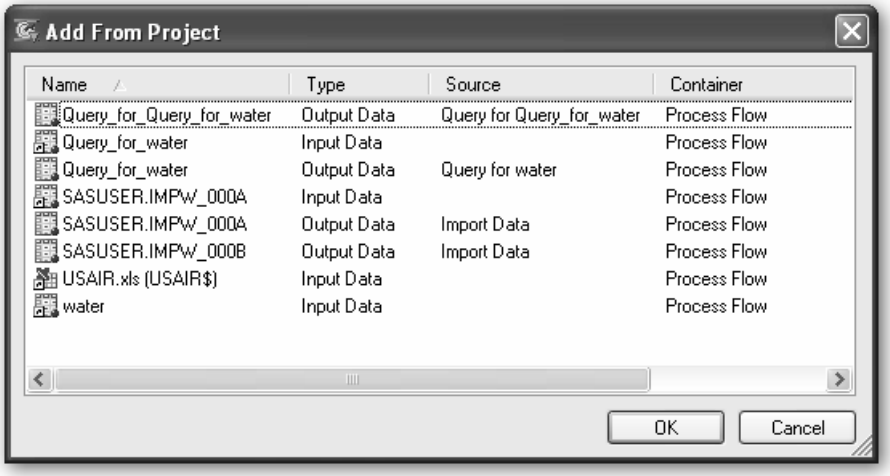
Where two or more data sets contain the same variables (or mostly the same) but different observations, they can be combined into a single data set using **Data➢Append Table** and specifying the table(s) to be concatenated with the active data set. Concatenation is essentially the converse of the process of splitting data sets described above.

Where two data sets contain mostly the same observations but different variables, they can be combined to create a data set with all the variables using a join. Joins are yet another function of the Filter and Query task. We will illustrate a join again using the **water** data set. The original **water** data set has a variable, **location**, with values **north** and **south**. The version imported from the raw data has a variable, **flag**, where the value

'*' indicates the more northerly towns. To check that the two variables do in fact correspond, we will merge the data sets to produce one that has both variables.

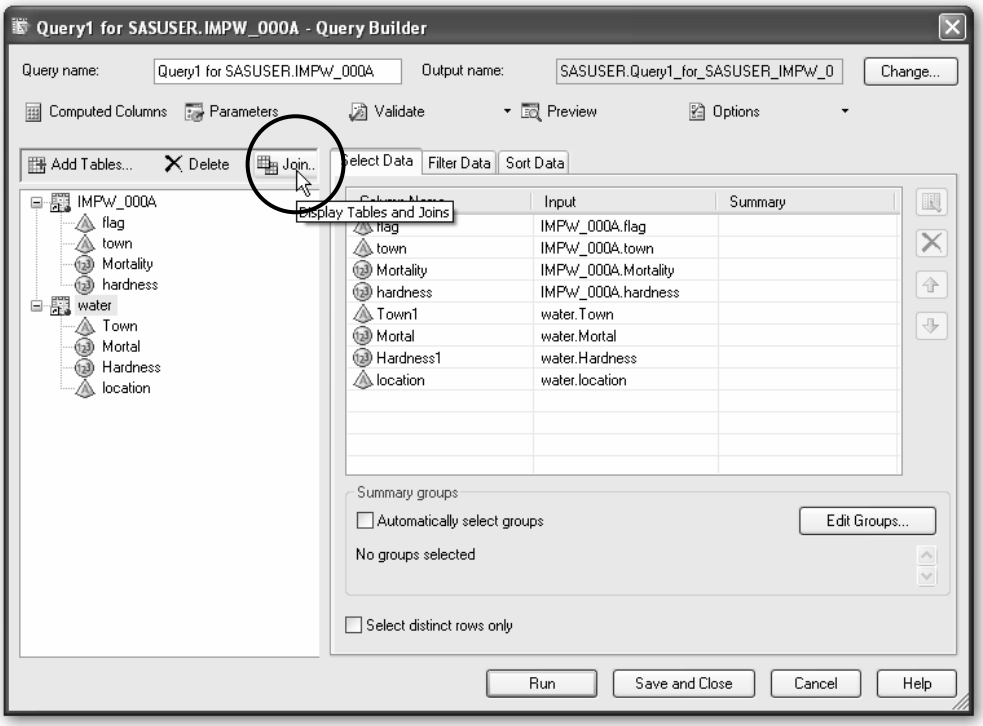
1. Make the imported data set the active data set.
2. Select **Data➢Filter and Query**.
3. Click **Add tables**.
4. Select **project** as the location to open the data from. The list of similarly named data sets shown in Display 1.17 illustrates the potential value of giving output data sets explicit and more meaningful names. In this instance, the one simply labeled **water** is the one we need.

Display 1.17 List of Project Data Sets



5. Select the **water** data set.
6. Click **OK**. A Query Builder window like that shown in Display 1.18 opens.

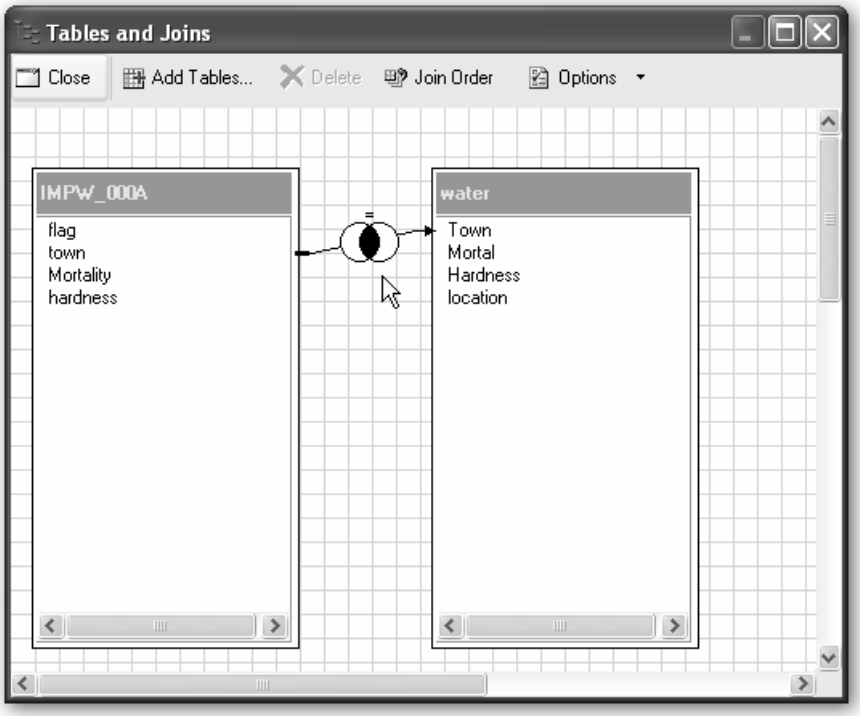
Display 1.18 Query Builder Window for Join of Two Versions of the Water Data Set



All the variables from the **water** data set have been added and, where they had the same name, the names have been suffixed with a 1 to make them distinct.

7. Click on **Join**. The join is displayed, as in Display 1.19, and can be modified if necessary.

Display 1.19 Join of Two Versions of the Water Data Set

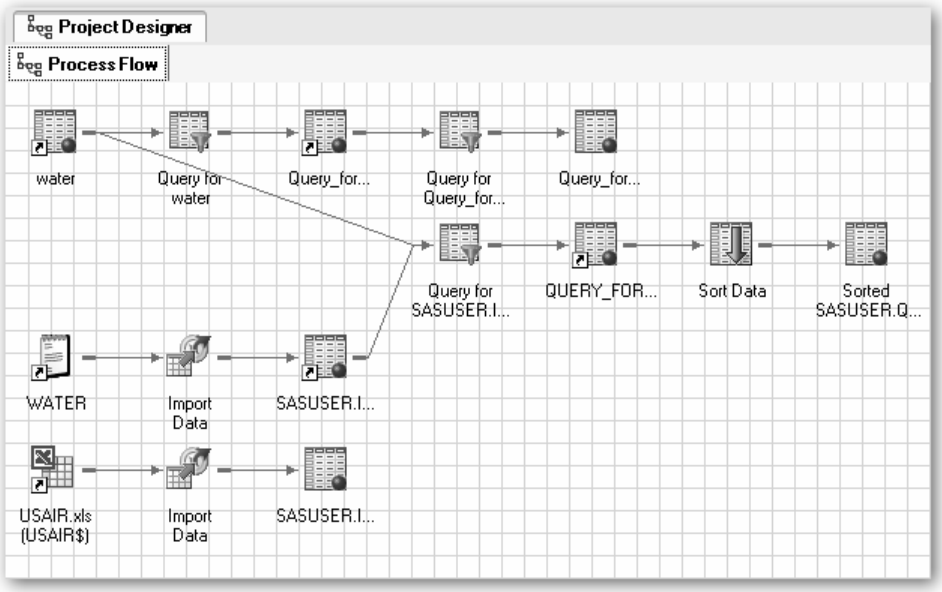


The program has recognized that both data sets contain the variable **town**, which uniquely identifies each observation and can therefore be used to match them. The Venn diagram in the arrow connecting them shows that an inner join will be used. Right-clicking on the Venn diagram and selecting **Modify Join** lists the different types of joins and explains them. A choice will need to be made if the two data sets contain different observations. Here, the two data sets contain the same observations, so the type of join makes no difference.

8. Close the **Tables and Joins** window.
9. Use the buttons on the right of the **Select Data** pane to delete **Town1**, **Mortal**, and **Hardness1**, and to move **flag** next to **location**.
10. Run the query.
11. Sort the resulting data set by location (**Data➢Sort Data** and **Sort by location**). Scrolling down the results confirms that **flag** and **location** do indeed correspond.

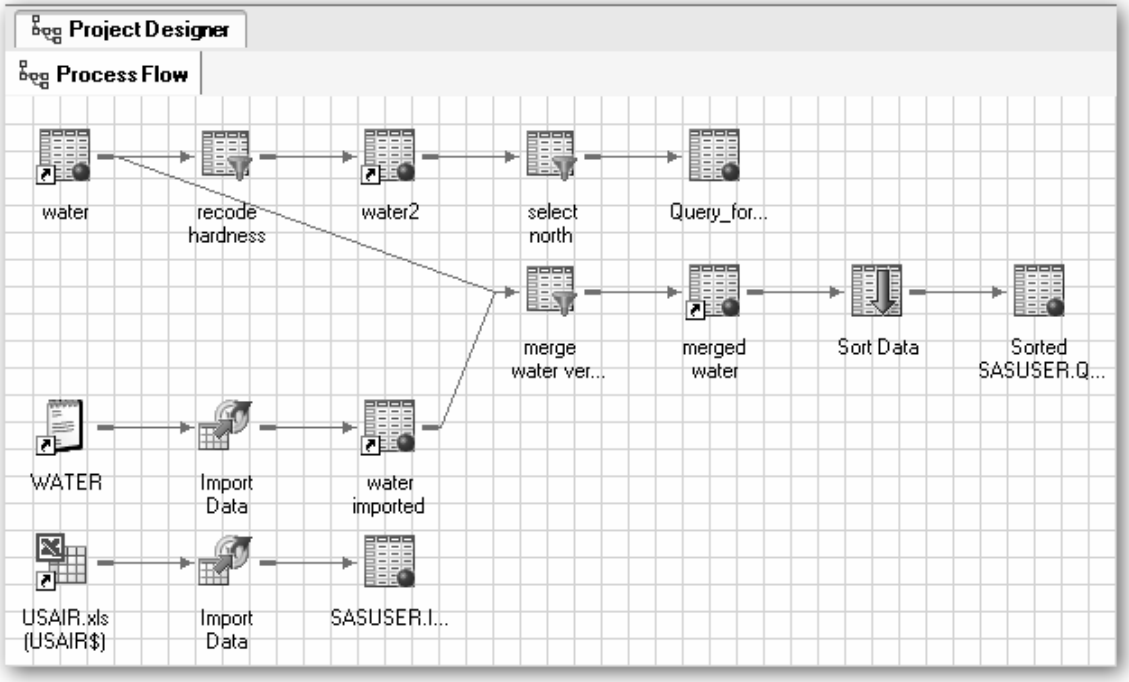
The process flow should now resemble Display 1.20. It is beginning to look a bit confusing. Several tasks and data sets have similar names (beginning with "Query") which do not give much idea of their purpose or contents.

Display 1.20 Process Flow with Default Names



Some of the tasks and data sets could be renamed (**right-click➢Rename**) to make this clearer. Display 1.21 shows an example.

Display 1.21 Process Flow with Renamed Tasks and Data Sets



1.5.5 Names of Data Sets and Variables in SAS and SAS Enterprise Guide

Renaming some data sets and tasks in the process flow, as we did for Display 1.21, actually changed their *labels* rather than their *names*. Data sets, variables, and tasks all have labels as well as names, but there are different rules for creating names and labels.

The SAS rules for names of variables and data sets:

- Names are limited to 32 characters or less.
- Names start with a letter or underscore (_) and include only letters, numbers, and underscores. Names should not contain spaces.

Although SAS Enterprise Guide has more flexibility in its naming, we recommend keeping to the SAS rules for variables and data sets.

Labels, in contrast, can contain spaces and other characters and can be up to 256 characters long. However, when there is any doubt about which is being changed, it would be safer to leave spaces out and keep to the rules for SAS names.

1.5.6 Storing SAS Data Sets: Libraries

The SAS data sets created so far have been left with default names and locations. Some data set labels were altered to make the process flow easier to read. In most cases, it is not necessary to alter names and locations. When you want to control where project data sets are stored, use *libraries*. Essentially, a library is a folder where SAS data sets are stored. Rather than refer to the folder explicitly, the folder is assigned an alias: the library name. For example, the data sets created by the Import Data task were automatically given names like **SASUSER.IMPW_xxxx**. The part of the name before the period, **SASUSER**, is the library name and is an alias for **c:\My SAS Files\9.1** on our system (it may vary depending on how SAS Enterprise Guide was set up). To store data sets in a particular folder:

1. Assign a library name for that folder using the **Assign Library** wizard (**Tools➢Assign Library**).
2. Type in a name, which should follow the rules for data set names but be eight characters or less; e.g., **ch1**.
3. Add a description if required.
4. When prompted, browse to the path of the folder; e.g., **c:\saseg\libraries\ch1**.
5. Continue through the wizard accepting defaults and an **Assign Library** icon should be added to the process flow.

   This needs to be run before the library can be used in the project, so it is best to set up the libraries at the beginning of the project. Having set up the library, any data set that is given a name beginning with **ch1**., such as **ch1.water**, will be stored in the folder **c:\saseg\libraries\ch1**.

All SAS data sets are stored in a library. If a data set name is not prefixed with a library name, it has the implicit library name of WORK which, like SASUSER, is one of the libraries assigned automatically by SAS Enterprise Guide. However, WORK is a temporary library which means that data sets stored in it will be deleted and removed from the project when SAS Enterprise Guide is closed, although the option to move the data sets to another library is offered at that point.

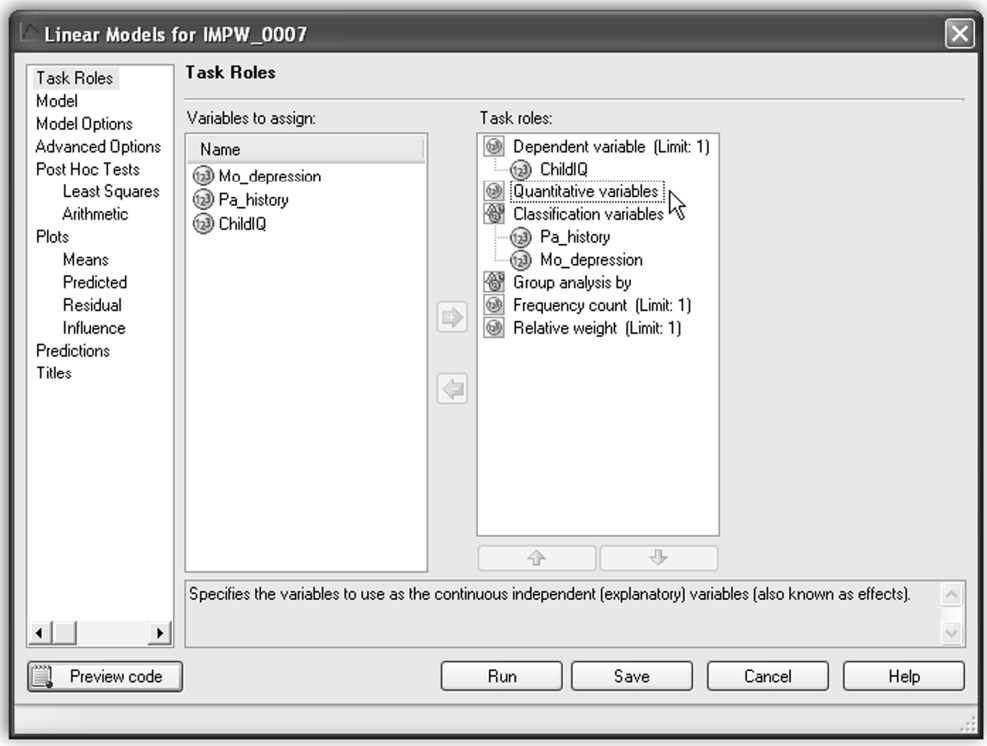
1.6 Statistical Analysis Tasks

Once data in a SAS data set have been added to a project, whether directly or by importing raw data, the analysis can begin. Individual tasks are described in detail in subsequent chapters. Here, we describe some general features of the analysis tasks.

One point to bear in mind is that not all tasks that might be considered as *analysis* are under the **Analyze** menu. Several are accessed from the **Describe** menu, and some of the tasks under the **Data** menu could form part of an analysis.

A typical analysis task consists of a number of panes, each of which allows some aspect of the analysis or set of options to be specified. We begin by looking at an example taken from Chapter 5. The process flow diagram is shown in Display 1.3. Opening the first of the Linear Models tasks gives the screen shown in Display 1.22.

Display 1.22 Linear Models Task Opening Window



The panes are listed down the left: **Task Roles**, **Model**, **Model Options**, etc.

The **Task Roles** pane, which is selected, is where the variables that are to be used in the analysis are selected and their roles in the analysis specified. The available variables are listed in the central section, and they can be dragged from there to the specific roles in the right-hand section. The available roles vary depending on the task, but some of the most common are included here:

- The **Dependent variable** is the response variable, the one whose values we are modeling. The numeric icon to the left indicates that only numeric variables can be assigned this role and (**Limit: 1**) to the right indicates that only one response variable can be included in the model. The variable **ChildIQ** has been assigned this role.
- **Quantitative variables** are also numeric. The dashed line around it shows that it has been selected (clicked on) and a description of the role appears in the box below, explaining that these are continuous explanatory variables. There are no variables assigned to this role.
- **Classification variables** are discrete explanatory variables. They can be numeric or character. If they are numeric, classification variables will tend to have relatively few distinct values. **Pa_history** and **Mo_depression** are both assigned this role.
- **Group analysis by** variables are also discrete, numeric, or character—variables which define groups in the data. When a variable is assigned this role, the analysis is repeated for each group defined by the variable. For example, if a variable, **sex**, with values **male** and **female** was assigned this role, the analysis would be repeated for males and females separately. We saw earlier how to use Filter and Query to split or subset a data set. If the reason for doing this is to apply the same analysis to separate groups of observations, then using **Group Analysis by** with a suitable variable could be both simpler and more efficient.
- **Frequency count** variables are used with grouped data, where each observation represents a number of individuals. The **frequency count** variable is the one which specifies how many individuals the observation pertains to. The most common use is in analysing tabulated data. Examples are given in Chapter 3, Sections 3.4.3 and 3.4.4.
- The **relative weight** role is for weighted analysis.

Task panes like **Model, Model Options**, and **Advanced Options**, as their names imply, specify what model is to be fitted and how. They will be dealt with in detail in later chapters as they arise.

Many analysis tasks also produce plots of data values, predicted values, residuals, etc., each of which may be specified in the **Plots** pane(s).

1.7 Graphs

SAS Enterprise Guide also makes the powerful graphics facilities of SAS much easier to use. Some of these graphic facilities are available within analysis tasks and others are accessed from the **Graph** menu. A wide range of plots and charts are described in later chapters. Rather than describe the graph tasks here, the interested reader is referred to the index.

One point to note, however, is that the graphs produced are dependent both on the format of the results and the graph format. Both formats are specified under **Tools➢Options➢Results➢Results General** and **Tools➢Options➢Results➢Graph**. One major difference is that, when the output format is RTF, the graphs are included in the same file as the textual output and tables; when HTML output is chosen, each graph appears in a separate file with its own icon in the process flow.

1.8 Running Parts of the Process Flow

So far, we have described running individual tasks. It is also possible to run a branch of the process flow or the whole process flow. If we right-click on any task within a process flow, we will have the option to run that task or to run the branch from that task. The branch is everything to the right of the task which is directly or indirectly connected to it by the arrows. To run the whole process flow, right-click on its tab and select **Run**.