Chapter 1
Out of the Box

DataView is a powerful software package, yet one that is also designed to be easy-to-use. This chapter demonstrates that ease of use, by illustrating some straightforward applications. The examples that are shown can be replicated using the demonstration data bank supplied with the program. In addition, you may wish to generate other practice examples as a means of becoming better acquainted with its features and capabilities.

Setting Up for the First Session

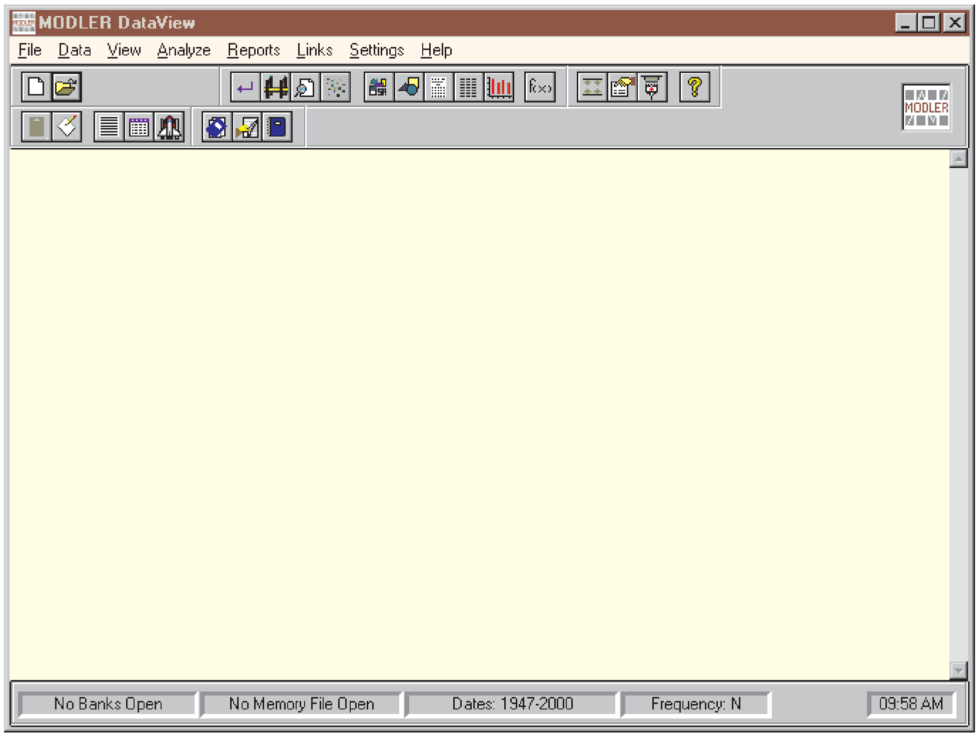
The process of installing DataView for use, and keeping it up-to-date as new versions are released, is not described here but in the separate *Getting Started Guide*. You should have received a copy of both guides, but if the *Getting Started Guide* is not immediately available, you can download a Word-formatted version from www.modler.com. Similarly, when DataView is installed supplementary files, including the demonstration data bank mentioned above and the macro files referred to in later chapters of this user guide, are also automatically copied to the directory (or folder) into which the program is installed. The demonstration data bank, in particular, is provided so that you can immediately begin to operate DataView, without first having to create a data bank; for this reason, the discussion of data bank creation and maintenance is put off until the fourth chapter. However, if these files do not seem to be loaded onto your machine, they too can be downloaded from the website. They are contained in the supplemental program file, DWinSup.EXE.

For present purposes, it is assumed that DataView has been installed on your machine and is properly configured. An indication that this assumption is valid is if the name DataView appears in the list of programs on your machine. In this case, you will often also see a program icon displayed on your desktop.

Executing DataView

Usually, in a Windows environment, DataView is executed either from a prompt or by clicking on the program's icon. In response, you should immediately see its Central Control screen display, as shown in Figure 1-1 on the next page. Considering this screen, there are several obvious features. Near the top is a menu bar, which displays action categories, many of which are likely to be generally familiar, and below which is a tool bar. At the bottom is a status bar, which as shown in Figure 1-1 indicates four things in particular: that there are No banks open, No Memory File open, that the default date range is 1947-2000 and that the observation frequency has not yet been set, the N indicating none. It obviously also shows the time; less obviously, if you click your mouse on the time, DataView will show you the current date. The dominating light-colored space in between the menu and status bars on your monitor is likely to be pale yellow in color. It will be referred to as the *command space* because, although initially blank, it can be used as a place to type commands, which then will be acted on by the program.

As you begin to use the program, you will discover that the screen components just identified are all active program control elements. In particular, if you click on the status bar items, you will find that they are "live." Initially, a click will elicit an offer to open a data bank or permit you to set the date range or specify the observation frequency for your work. Subseqently, clicking on the status bar will display information about the open data banks or other relevant details. Clicking on the menu keywords offers an alternative way to open data banks, set the observation frequency, or specify the date range, but also a wide range of additional options.



No Banks Open | No Memory File Open | Dates: 1947-2000 | Frequency: N | 09:58 AM

Figure 1-1 DataView Opening Screen

The same can of course be said for the tool bar items. For many of the menu elements, there is a corresponding tool bar item. However, these have sometimes been grouped in a slightly different way and the emphasis in their creation is not upon completeness, but instead the likely frequency of use. In general, in this user guide, the tool bar items will not be discussed individually, the presumption being that you will learn to use these as you use the program, simply by knowing that you would like to perform a particular operation and having realized the correspondence between a particular icon button and that operation.

Menu Categories: A Brief Overview

The menu categories generally obey the standard windows conventions. The first of these, **File**, broadly refers to file operations, and clicking on it reveals the dropdown choice list shown in Figure 1-2. As usual, greyed-out choices indicate operations that cannot be selected currently.

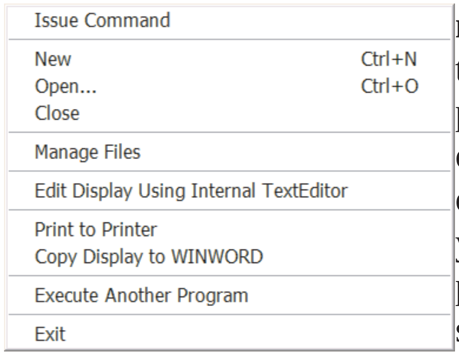


Figure 1-2 File Menu

The first choice, **Issue Command**, discussed later, is used as these words imply. The next three permit you to create new files, open existing files, and close any open files. **Manage Files** allows you to execute a file manager appropriate to your operating system, such as Windows Explorer. **Execute Another Program** obviously executes other Windows or DOS programs. **Exit**, as usual, terminates the program.

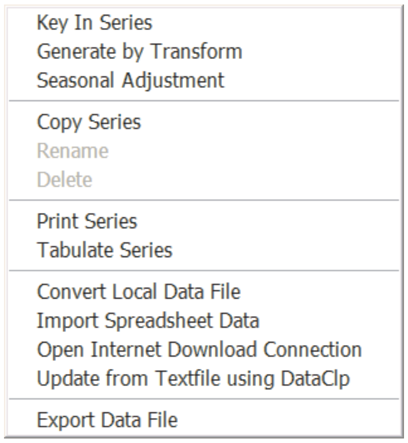


Figure 1-3. Data Management

The menu category, **Data**, is more specific to DataView, and refers generally to Data Management operations, as shown in Figure 1-3. The first three choices provide access to forms that respectively manage the input of time series observations, the creation of series by transformation, and the seasonal adjustment of monthly or quarterly frequency series.

The next three permit time series to be copied from one place to another, or else renamed or deleted. Individual series can be copied, but groups are more commonly copied. Similarly, **Print Series** and **Tabulate Series** are mainly used to display groups of series. The last five, **Convert Local Data File**, **Import Spreadsheet Data**, **Open Internet Connection**,**Update from Textfile using DataClip**, and **Export Data File**, either launch your Internet browser or manage the import and export of data.

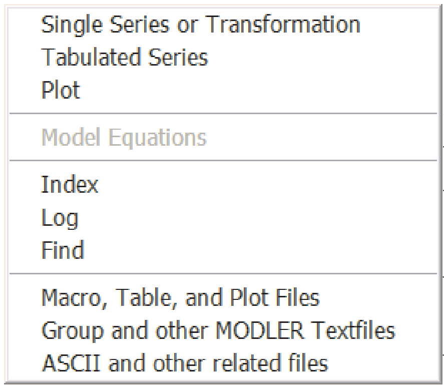

Single Series or Transformation
Tabulated Series
Plot
Model Equations
Index
Log
Find
Macro, Table, and Plot Files
Group and other MODLER Textfiles
ASCII and other related files

Figure 1-4. Display Operations

The next category, **View**, provides evidence that major menu groupings are inevitably arbitrary. The operations under this heading range from the tabular or graphical display of observations on specific series to viewing (and editing) text files. Individual series (or expressions) can be arrayed in tables or plotted as single or multiple series (or expressions) by choosing **Single Series or Transformation**, **Tabulate Series,** or **Plot** respectively.

The middle three choices, **Index**, **Log**, and **Find**, provide access information about series choices

In contrast, **Macro**, **Groupfile**, and **ASCIIfile**, as choices, allow you to view and edit files of these named types, whether or not specifically DataView-related. Macros are used to define batch operations: commands are commonly collected into such files for production runs in order to minimize repetitive work. Groupfiles are textfiles that contain sets of series names; such files are used in various contexts in order to reference series as defined groups. ASCII files are generic text files. A reason to distinguish these file types is that often they are placed in particular directories, to which these commands are keyed. As will be explained later, when any such textfile is viewed, a text editor screen appears, with the characteristics to support text processing.

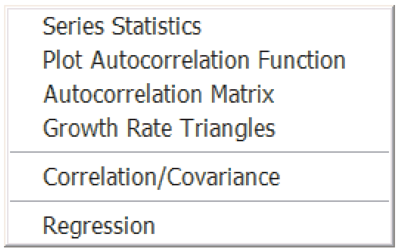


Figure 1-5. Analysis Options

The **Analysis** category, as its name implies, provides access to various "number crunching" operations, which range from performing simple transformations, to displaying growth rate triangles or performing regressions.

Except for regressions, which are the subject of Chapter 5 of this Guide, the various Analysis options are described in some detail in Chapter 3. The options also include those, like seasonal adjustment, that appear in other menu categories.

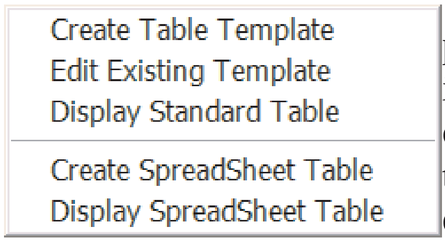


Figure 1-6. Reports Option

The **Reports** category manages the process of developing and producing tabular reports. The options include producing draft tables within the context of DataView and presentation tables in concert with spreadsheet packages such as Excel, Lotus 123, or Quattro Pro.

The production of tables and other reports is described in the second half of Chapter 2 of this Guide. Additional specialty tables are described in later chapters.

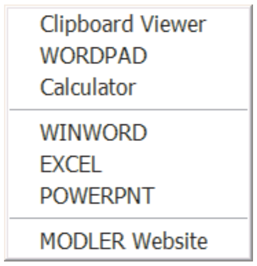


Figure 1-7. Links Option

The **Links** category provides access to a variety of external programs likely to be on your computer, as well as permitting linking to the www.modler.com website via your designated Internet browser. DataView is able to launch a variety of external programs, including the Clipboard viewer, notes editors, word processors, and spreadsheet programs. In addition, it can launch your browser and connect to the DataView.COM and other Internet sites.

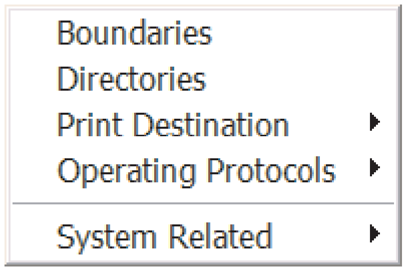


Figure 1-8. Settings

The **Settings** category is used both to establish session settings and general settings that define your working environment. The Boundaries and Print Destination Settings are session related, the latter controlling the effect when the Print button on various forms is pushed; the others are environmental.

The settings associated with **Directories,** *a very important topic*, are briefly considered at the end of this chapter. These are fundamental to the way in which DataView operates. **System Related** settings provide access to the Control Panel and other general Windows facilities.

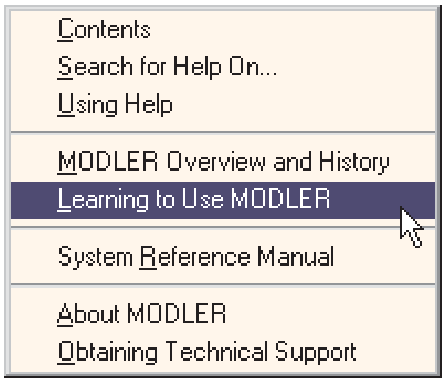


Figure 1-9. Help Facilities

Finally, DataView incorporates an extensive Help facility. **Help** incorporates both context specific help, keyed to particular operations and forms, and a complete reference to all the program's operations. Press the function key F1 at any time to obtain screen specific help.

The Help facility is designed for all users. It supports the new user, offering background and specific learning modules. For the experienced user, it incorporates an encyclopaedic System Reference Manual. For everyone, technical support information, including email and other pertinent addresses, is supplied.

Supporting files, instructional macros, and updated information about DataView are available at all times on the www.modler.com website.

Concepts

The foregoing brief review of DataView's opening screen and menu system is intended to provide an indication of the range of program facilities. This review also incidentally demonstrates that the program is designed to be operated in conformity with the standard "Windows" conventions. An essential characteristic of any "Windows" compatible program is that it should obey the standard conventions of the environment to a relatively high degree. This does not mean that DataView is slavish in its adoption of the Microsoft precepts, for there are necessarily a number of stylistic differences, but where departures occur they have been considered carefully.

You should also be aware that DataView is designed to be a powerful tool, not a black box. It is not intended to foster ignorance. To the contrary, DataView generally requires you to understand the nature of the operations it performs; there are not a lot of "wizards" cloaking the individual program steps. Furthermore, to make best use of the program, you are encouraged to expend effort in order to become familiar with these operations.

It is obviously a virtue if a program like this is easy to learn to use. Yes, as just indicated, this is a qualified characteristic. The assumption that its users will have the appropriate training to use its capabilities means that, on its own, the program is not designed to teach economics nor statistics nor econometrics. Nevertheless it is a DataView design concept that what the program does and how it does it should be relatively intuitive to anyone with the appropriate training. Essentially, the aim is to provide a convenient, comprehensible, and comprehensive productivity tool.

In the Introduction, the point was made that DataView can be viewed as a resource manager, where the essential resource is the available data base. The progam is also very much a facility to apply a series of operators to that data. Among other things, DataView generates Plots and produces Tabular reports. It performs regressions and provides other facilities that permit the creation, maintenance, and use of small or large scale econometric models. These involve specific operations and in a general sense you need to understand what is happening at each stage of the process. The underlying operations that are performed are essentially statistical/mathematical in nature, so that there is little point in trying to hide the fact that the purpose of regression is to estimate the (presumably constant) parameters of an equation, or that a model is made up of sets of equations. Equations consist of constants, named variables and operators, such operators taking the form of both arithmetic operators (+, -, *, / and **) and intrinsic functions (Log, Exp, Sin, Cos, and the like) and even various relational and logical operators.

A further fundamental characteristic of today's versions of DataView is that these are only the latest in a line of development that extends back to 1968. A certain amount of DataView source code dates to that time and the development of the software since has been evolutionary. More to the point, inasmuch as functioning models, table templates, macro files, and data banks still exist that date back to as early as the mid-1980s and before, and a considerable legacy of user models and macros exist from throughout the 1990s, one of the most basic goals in the recent development of DataView is that it should support continuous use with essentially no forced obsolence of its users' earlier work. Fortunately, it has been possible to achieve this goal without limiting the extensive development of the program.

DataView: the Basics of Operation

As a first step towards learning DataView's operating characteristics it is helpful to consider the nature of specific actions the program performs and why. A typical DataView action is the retrieval of time series data to be used in the subsequent performance of some task. Such data consist of observations that are both dated in time and defined with reference to particular, standard intervals of time that take the form of weeks, months, quarters, years, and the like. The observations also have a conceptual definition that relates to a single or collective economic activity: Gross Domestic Product, Consumption Expenditures on Services, Employment in Durable Manufacturing, and such. Obviously, these data need not be macroeconomic, but in practice many time series are.

In order to be able to instruct a computer program to retrieve particular data series, it is evident that certain information must either be known in advance or easily discoverable. In particular, such information as:

- where the data are, generally and specifically,
- what concepts are represented,
- at what frequency or frequencies, and
- for what overall date range, such as 1947 to 1999.

The issue of data location is obviously a matter of which file and where that file is located, but it is also more specifically a question of the manner in which the observations will be retrieved.

In the context of any software environment, data series must be stored in a particular, usually proprietary way in order to be readily usable. As a general organizational principle, in the DataView environment, data are stored in the form of distinct portable historical collections known as *data banks*, which moreover are specifically configured so as to permit data series to be retrieved rapidly. A single data bank can contain as many as 10,000 series, so that it is reasonable to suppose that you may obtain fairly comprehensive collections of data in this form. Independent data vendors, such as Haver Analytics in the US, FERI GmbH in Germany, and Ibermetrics in Spain, have for many years provided data banks in the MODLER Dataview format, which can of course be used with the program the moment they have been copied to your computing environment. Sometimes these are well-defined, consistent collections, such as the historical Flow of Funds data for the United States from the late 1940s to the present day. Such data banks ordinarily contain 400-600 data series. At other times, they can be large and somewhat more conceptually amorphous, each containing as many as 2000 or more series, and be part of a set of banks defined by name as US monthly, quarterly, and annual frequency data banks that together contain in excess of 6000 series. At a different scale, containing only 60 data series, is the DEMOBANK.BNK data bank that is provided with each new copy of DataView specifically so that you can recreate the examples shown in this User Guide.

DataView data banks are configured so that the size of the bank does not noticeably affect the speed of retrieval of the data it contains. Even if you use a stopwatch, you are unlikely to be able to determine easily whether a series came from a 5000 series bank or a 50 series bank. It would in fact be possible to permit DataView data banks to contain even hundreds of thousands of series with very little, if any performance penalty, considering just the speed of retrieval of a given series. This is a statement of technology, and reflects not only DataView’s algorithms but also the fact that Pentium III microcomputers operating at clock speeds of 500 MHz and more provide the basis for good performance.

Access to Data and Information

There is nevertheless a sense in which bank size does matter:

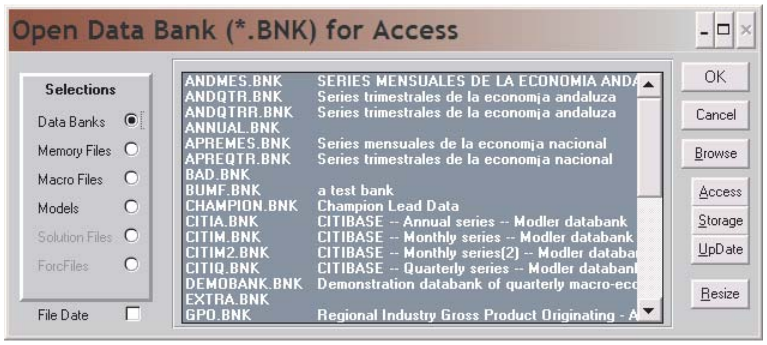
your ability to choose from among many series. That is, the real performance problem is not how to extract a given selected series, but rather how to make it possible for you to select easily the series that should be extracted. A property of any DataView data bank is that it is physically organized to contain not only the observations but also descriptive information about its constituent time series. Clearly, if each data series were to have associated with it a single line of descriptive information, then a bank containing 50 series could display the descriptions using a single high resolution screen's worth of space, or at worst 2 to 3 screens. In contrast, to display the information on a 5000 series data bank would take at least 100 screens. Quite clearly, the more comprehensive the data base you have at your fingertips, the greater the potential difficulty in using it.

But before tackling head on the issue of discovering which series to retrieve, consider first the prior step in this process: opening a given data bank for access. One or more data banks (up to a maximum of 15 at a time) can be opened simply by clicking on **File** and then **Open** progressively. The result of these two clicks together is shown below in Figure 1-10, which lists the available data banks and allows you to make a selection.

Should you need to read the bank descriptions, which may be chopped off by the form, the rightmost edge can be stretched using your mouse; the **Resize** button returns the form to original size. Double click on the bank you wish to open and you should immediately see that the status bar on the main DataView screen tells you that there is at least "1 bank open." Alternatively, you can open a data bank using the Access command word followed by the name of the bank:

ACCESS bankname

although you may need to provide the path as well as the bankname.



```
ANDMES.BNK     SERIES MENSUALES DE LA ECONOMIA ANDA
ANDQTR.BNK     Series trimestrales de la economja andaluza
ANDQTRR.BNK    Series trimestrales de la economja andaluza
ANNUAL.BNK
APREMES.BNK    Series mensuales de la economja nacional
APREQTR.BNK    Series trimestrales de la economja nacional
BAD.BNK
BUMF.BNK       a test bank
CHAMPION.BNK   Champion Lead Data
CITIA.BNK      CITIBASE -- Annual series -- Modler databank
CITIM.BNK      CITIBASE -- Monthly series -- Modler databank
CITIM2.BNK     CITIBASE -- Monthly series(2) -- Modler databa
CITIQ.BNK      CITIBASE -- Quarterly series -- Modler databan
DEMOBANK.BNK   Demonstration databank of quarterly macro-ec
EXTRA.BNK
GPO.BNK        Regional Industry Gross Product Originating - A
```

Figure 1-10 Open Data Bank for Access

Once a bank has been opened, displaying the observations on any of the series it contains is as simple as clicking **View** and then **Series**. DataView should respond by displaying the form shown as Figure 1-11. In principle, all that you then need do is to provide a series name, although as the form tells you, you could instead type in a complete expression, in which case DataView will evaluate it and display the results.

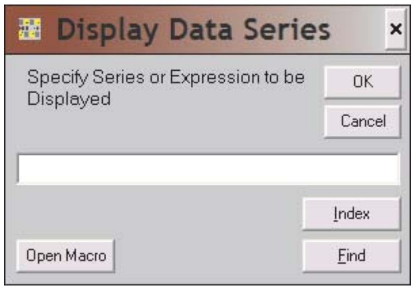
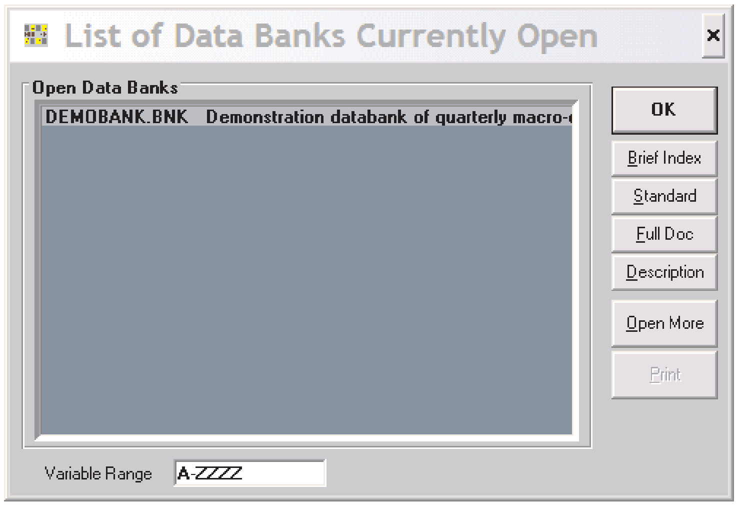


Figure 1-11. Series Display Control Form

But before you type in some series name, look at the form and note the three buttons **Index**, **Find**, and **Open Macro**. In order to provide you with as much information as possible when you need it, whenever a form (or dialogue box) appears that asks you to provide a name of a series or to type an expression you will find at least the first two of these buttons near at hand.

The **Index** button, if pressed, immediately causes the display of a separate form like that shown in Figure 1-12. This one provides



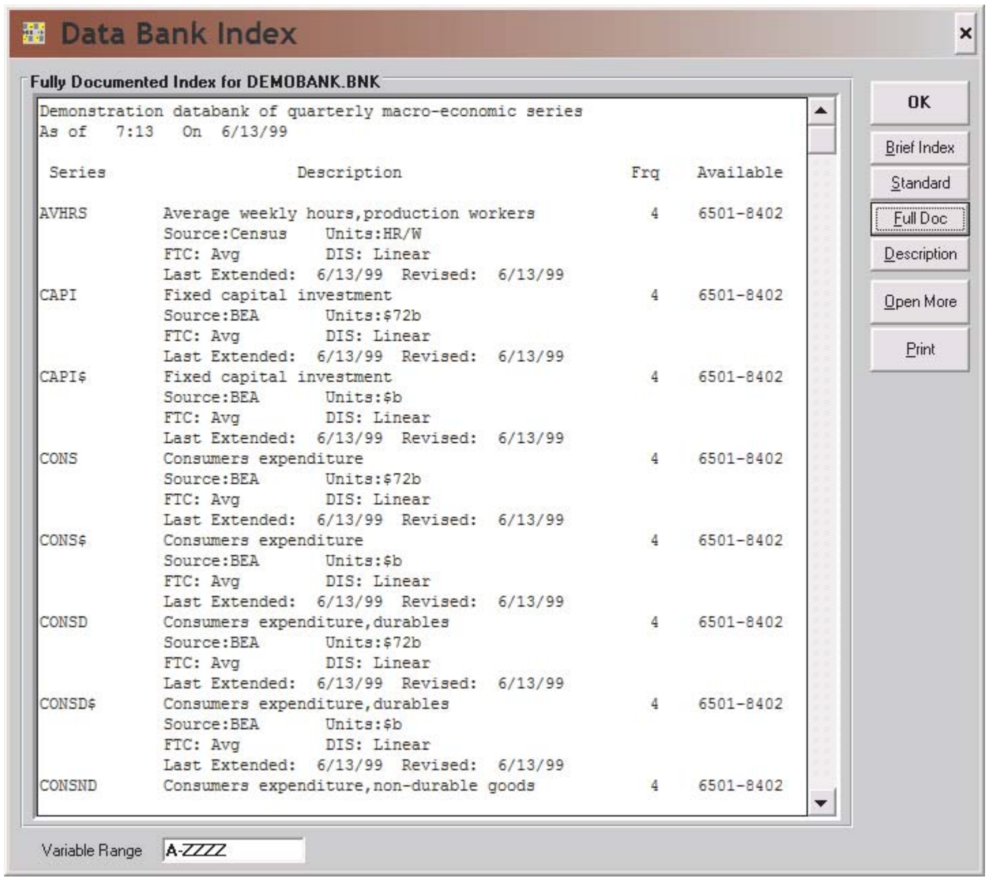
DEMOBANK.BNK Demonstration databank of quarterly macro-

A-ZZZZ

Figure 1-12. Index Control Form

the names of all open data banks and, as you click on each, gives you the choice of displaying for it any of three types of indexes: a brief index, a standard index, or a fully documented (Full Doc) index. The brief index consists of simply the names of the series in the bank, arranged in alphabetic order; it is obviously most useful when you already know by sight the series in the bank, but may not remember precisely these names. The standard and Full Doc indexes provide detail on each series in the bank, organized alphabetically.

An example of a Full Doc index is shown as Figure 1-13 on the next page. The standard index differs by not including the frequency transformation codes for the series, nor the latest dates of revision and extension of those series. The form also provides access to a general description of the bank, obtained by pressing **Description**. An additional characteristic of the form shown in Figure 1-13 is that if you double click on the name of any series and then press the OK button, that series name will be transferred to the command space of the original form. Pressing the OK button on the original form then causes the operation to be performed that uses the identified variable. The aim of course is to provide you not only with the information you require but also to minimize effort. Once you have selected the series you wish to use, it is



```
Demonstration databank of quarterly macro-economic series
As of   7:13   On  6/13/99

 Series                   Description                          Frq    Available

AVHRS       Average weekly hours,production workers            4    6501-8402
            Source:Census    Units:HR/W
            FTC: Avg         DIS: Linear
            Last Extended:  6/13/99  Revised:  6/13/99
CAPI        Fixed capital investment                           4    6501-8402
            Source:BEA       Units:$72b
            FTC: Avg         DIS: Linear
            Last Extended:  6/13/99  Revised:  6/13/99
CAPI$       Fixed capital investment                           4    6501-8402
            Source:BEA       Units:$b
            FTC: Avg         DIS: Linear
            Last Extended:  6/13/99  Revised:  6/13/99
CONS        Consumers expenditure                              4    6501-8402
            Source:BEA       Units:$72b
            FTC: Avg         DIS: Linear
            Last Extended:  6/13/99  Revised:  6/13/99
CONS$       Consumers expenditure                              4    6501-8402
            Source:BEA       Units:$b
            FTC: Avg         DIS: Linear
            Last Extended:  6/13/99  Revised:  6/13/99
CONSD       Consumers expenditure,durables                     4    6501-8402
            Source:BEA       Units:$72b
            FTC: Avg         DIS: Linear
            Last Extended:  6/13/99  Revised:  6/13/99
CONSD$      Consumers expenditure,durables                     4    6501-8402
            Source:BEA       Units:$b
            FTC: Avg         DIS: Linear
            Last Extended:  6/13/99  Revised:  6/13/99
CONSND      Consumers expenditure,non-durable goods            4    6501-8402
```

Figure 1-13. Fully Documented Index

easy for DataView to move the relevant mnemonic to the command space; there is no need for you to type the name. The mnemonic is moved to the place your cursor is at the time you press the Index button, so that in fact the placement is precise, thus allowing you to insert it exactly where you wish.

A data bank Index is obviously useful when you know generally what you are looking for among the data series contained in a bank. However, when you first access a new bank, you may know roughly what sort of series it contains but you may not know the exact contents or the series naming conventions. The alphabetic ordering of an index is of course both its strength and its weakness. The benefit, because of the alphabetization, is that you know where to find the name of a particular series in the index. The drawback, as with any dictionary, is both that you must know the name, to a high degree of approximation, and that you must go to the point in the index where the name is.

In contrast, the **Find** button offers the capability to discover one or more series in the currently open data bank(s) that have some identifiable characteristic, whatever the name and wherever located. Procedurally, when this button is pressed, you are asked to type a character string, which could be a complete word, such as “income” or “rate” or “price.” It also can be some fragment such as “consum” or “inves.” DataView searches both the alphabetic index and the series descriptions, looking for a case-insensitive match.

Note that the longer the keyword string, the more restrictive the choice: for instance, the string “inv” can refer to both inventories and investment, whereas “inves” refers only to the latter. You should also be aware that DataView will find embedded strings, as well; as in: disINVEStment.

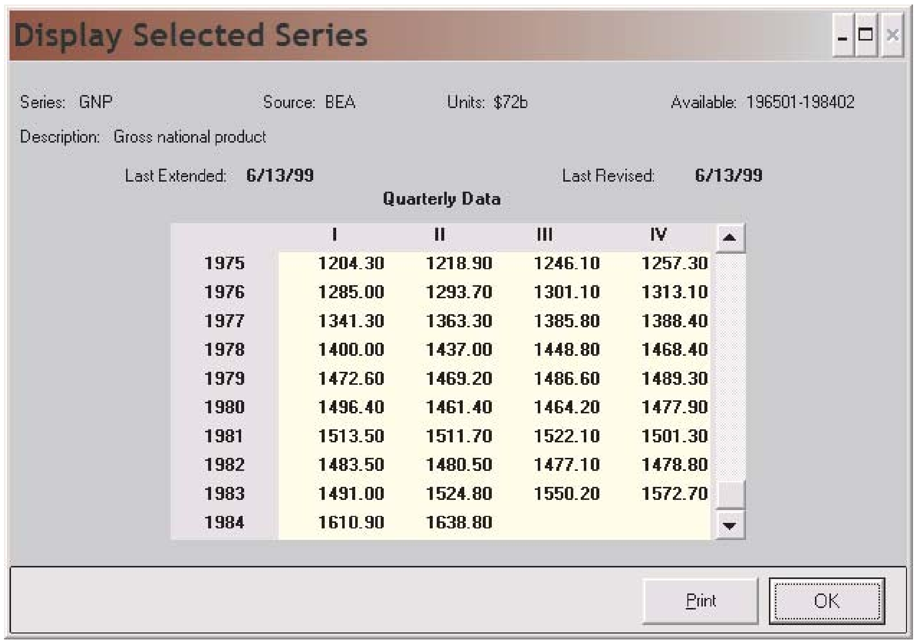
Once Find has found a series you wish to use, the series’ name can be moved automatically to the command space in much the same way as in the case of **Index**: simply doubleclick on the series mnemonic and it will be copied.

There are a few caveats. In order for the Find facility to work properly, data banks must be properly documented. If series are not described, or are badly described, then what you will find is anyone’s guess and may be misleading. Note also that one of the aspects of a well documented data base is consistency in abbreviation, and that each user knows how abbreviations are made. Ideally, the Find facility is very powerful, but in practice you may also be glad to have the Index button even if you are working with large data banks.

Finally, **Open Macro**, the third button, permits you to gain access to previously created command files. This option is generally of little use if all you wish to do is to view a series. It is of greatest use if you wish to view a previously created complex expression, for if you highlight a string of code in an opened macro this code will be copied immediately to the command space on the form shown earlier in Figure 1-11.

Simple Data Display

The foregoing discussion has been focused upon facilities that can aid in discovery, but once you have determined the variable to display, by the process of clicking on **View**, then **Series**, having specified its name, and then either pressed the <Enter> key or **OK**, the resulting display itself will generally be similar to that shown in Figure 1-14.



Series: GNP Source: BEA Units: $72b Available: 196501-198402

Description: Gross national product

Last Extended: 6/13/99 Last Revised: 6/13/99

Quarterly Data

	I	II	III	IV
1975	1204.30	1218.90	1246.10	1257.30
1976	1285.00	1293.70	1301.10	1313.10
1977	1341.30	1363.30	1385.80	1388.40
1978	1400.00	1437.00	1448.80	1468.40
1979	1472.60	1469.20	1486.60	1489.30
1980	1496.40	1461.40	1464.20	1477.90
1981	1513.50	1511.70	1522.10	1501.30
1982	1483.50	1480.50	1477.10	1478.80
1983	1491.00	1524.80	1550.20	1572.70
1984	1610.90	1638.80		

Figure 1-14. Simple Time Series Display

Note the various aspects of this display, including the description, the specification of the dates for which the series is available in the bank from which it has been retrieved, as well as the capability to print the series, as implied by the Print button. As with most such DataView displays, you can increase the range of observations displayed (when relevant) either by stretching the display top or bottom or by pressing the maximize button at theupper right hand corner of the form, and of course you can scroll line by line.

The purpose of this type of display is to provide, in an economical form, the maximum amount of information about the series displayed. Note in particular, the Last Revised and Last Extended dates. Last Revised refers to the last time observations within the date range of the series were changed, presumably revised. Last Extended refers to a change in the earliest or latest date of availability. These dates can provide you with important information concerned the currency of the observations displayed.

An important DataView capability is implicit in this display, which is that the program will automatically adjust the default date range to suit the operation performed, whenever the date range set exceeds that for which observations are available. Thus, for instance, if the default date range is set for the period from 1947 through 2000, the fact that the latest observation date is the second quarter of 1984 implies that there are no observations on the series since that date—at least to the degree that the bank is regularly and carefully updated. You will find that this type of date range adjustment occurs generally in DataView, and one of its benefits is that you do not need to know in advance more than the general date range for your work.

What is not obvious from this display, but is an equally important DataView characteristic, is the search order for series when two or more banks have been opened for access. The bank opened first is the first searched, and if a series name is found that matches the one you have provided, the series displayed will be from this bank. If no such series is found in the first bank, the second bank is searched, and so on, until the first match occurs.

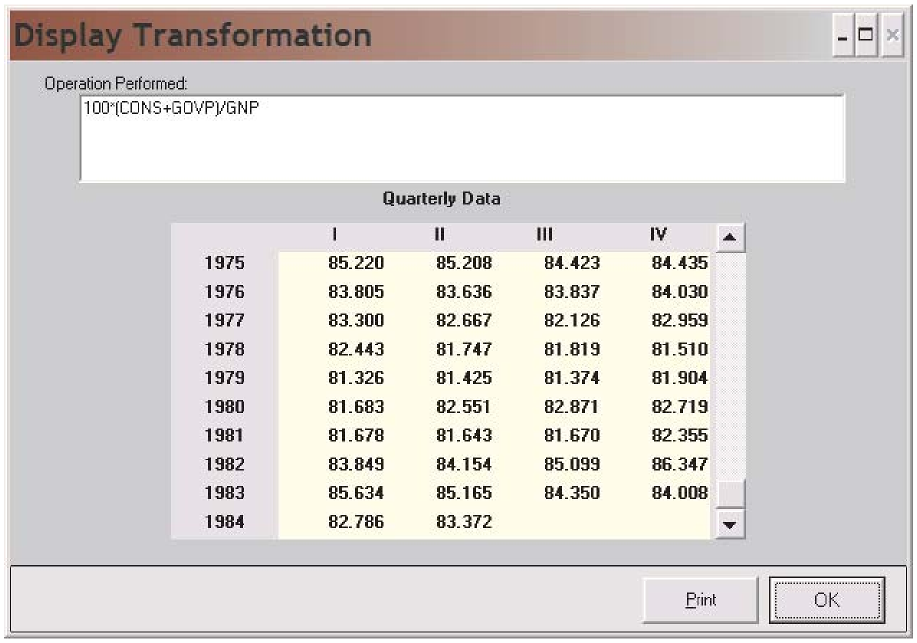
There are exceptions to this selection rule: for instance, in the case that a particular data bank is being updated, as described in Chapter 4, or in the case that a Memory File is open, as described in Chapter 3. But you need to be aware that, in all cases, DataView searches data banks for series in a specific order. In addition, as will be discussed in Chapter 3, should you have stored data in the Memory File, which is a temporary storage location, it will usually be searched before *any* data bank. Understanding the default search order is therefore important.

Transformations can also be displayed easily. To display a transformation, simply type it into the command space of the form shown in Figure 1-12. The dropdown menu title **Series or Transformation**, as shown in Figure 1-4, indicates that although this facility is used to display series, nothing precludes your specifying an expression instead. For example, try:

```
100*(CONS+GOVP)/GNP
```

Assuming of course that the appropirate data bank is open for accesss, this command on its own will cause observations to be retrieved on the three variables CONS, GOVP and GNP, the specified transformation to be performed, and the results to be displayed. This outcome is illustrated by Figure 1-15.

The transformation will be performed for the mutually longest period of data availability of all the variables, within the bounds of the default date range. The rule that DataView follows for all transformations is that, for any observation, if any constituent variable lacks a valid observation, the result for that date will be recorded immediately as NA, Not Available. When transforma-



100*(CONS+GOVP)/GNP

Quarterly Data

	I	II	III	IV
1975	85.220	85.208	84.423	84.435
1976	83.805	83.636	83.837	84.030
1977	83.300	82.667	82.126	82.959
1978	82.443	81.747	81.819	81.510
1979	81.326	81.425	81.374	81.904
1980	81.683	82.551	82.871	82.719
1981	81.678	81.643	81.670	82.355
1982	83.849	84.154	85.099	86.347
1983	85.634	85.165	84.350	84.008
1984	82.786	83.372		

Figure 1-15. Diplay of a Transformation

tions are performed, it is consequently possible to find the results displayed for a shorter period than that for any series used in that transformation. Certain result observations within the overall data range may also be NA. Furthermore, given the search rules described earlier, it is also obviously quite possible to perform transformations for which the variables used have been retrieved from different data banks; this is normally desirable, but you need to be aware that the possibility also opens the door to error if the same named series appear in two or more banks yet are different, or if not all copies of the same series, in two or more banks, are equally up-to-date in their revision cycles.

Note also in Figures 1-14 and 1-15 that the observation frequency is stated prominently in the display. A specific aspect of a transformation is that you must generally specify the observation frequency in advance. If only a series is displayed, DataView can infer the observation frequency, if none is given, from the characteristics of the series as it is retrieved; the documentation for the series includes its original frequency. In contrast, a transformation can begin with an explicit constant value, as in the example used, which provides no indication of the frequency of observation of the variables; in this case, if the frequency has never been set DataView will request that you set it. If the frequency is incorrect for the series retrieved, then by default DataView will note the "error."

It is also possible of course that you may wish to display a series, or a transformation, at an observation frequency different from that of the series as retrieved. DataView permits on-the-fly frequency conversion, provided that you have indicated that you wish this to occur. Click on **Settings**, then **Boundaries** (or, more directly, click on the status bar item labelled "Frequency") Then, on the resulting form, check "Automatic Frequency Conversion," if you wish to be able to convert automatically from higher to lower frequency (for instance, months to quarters). The other check box labeled "Automatic Distribution of Observations" permits lower to higher frequency conversion (for instance, years to months). But, whenever you do this, you should be aware of two things in particular. First, that in both cases the conversion method is determined series by series from the data bank documentation for that series, although there is a way to override this convention. Second, that conversion from lower to higher frequency involves interpolation and thus a new source of measurement error for any series so converted.

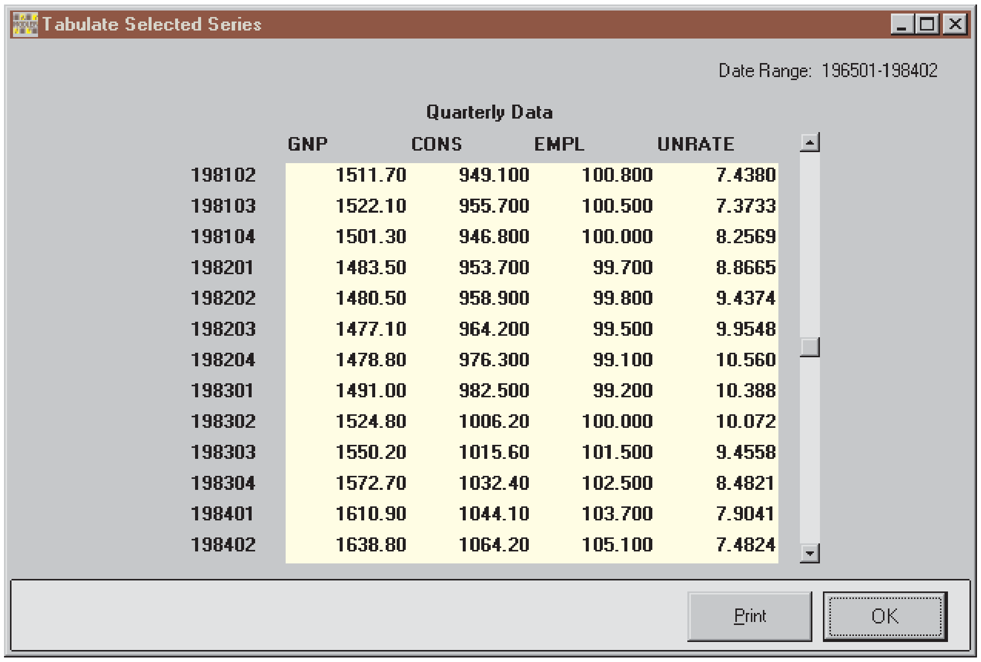
Tabulate Series

Tables produced by DataView generally take one of two forms: specialty tables and user-designed. The specialty tables, such as the default model solution table, are produced automatically by the program, and are designed to be produced quickly in particular contexts, with minimum effort on your part. For instance, when you are in the process of first solving a new model, you are unlikely to want to stop and spend alot of time designing a table; at that stage, you will not even be sure that you do not wish to make further major changes in the model, once you have seen the first solution. So the idea behind DataView's specialty tables is not so much beauty as simply being able to produce a reasonable table quickly.

In contrast, the user-designed tables are as fancy as you wish to make them, but require you to develop table templates. This template development process can be quite automated, and may involve the creation of a spreadsheet based table, but will nevertheless take a bit of time. It may take a good bit of time. The benefit is that you will be able to control the format, including the line descriptions. Furthermore, once you have developed one or more table templates, these can be used repeatedly to produce the tables, every week, every month, every quarter, or each year. The second half of the next chapter, Chapter 3, introduces this subject.

However, it is also possible to produce simple, vertically aligned tables using essentially the same degree of effort required to display a single series. Tabulated Series tables are produced by clicking on the **View** menu item of the Central Control screen, then choosing **Tabulated Series**. The initial effect is to cause the display of a dialogue box very similar to Figure 1-13, into which (in the simpliest case) you can type one or more series names, then press OK or the <Enter> key. DataView will then generate a display like that shown in Figure 1-16.

Note that this scrollable, columnar table vertically aligns the series by date. One use of such a table is to allow you to compare the values of series, since comparable observations appear in the same table row. However, you are not restricted simply to series. Instead of entering series names in the dialogue box referred to earlier, you can type expressions, separated by commas. Thus you can compare expressions, observation by observation. Of course, you can also mix series and expressions. And should you attempt to display more series or expressions than can fit into the width of your screen, the display will automatically scroll, or pan,



Date Range: 196501-198402

Quarterly Data

	GNP	CONS	EMPL	UNRATE
198102	1511.70	949.100	100.800	7.4380
198103	1522.10	955.700	100.500	7.3733
198104	1501.30	946.800	100.000	8.2569
198201	1483.50	953.700	99.700	8.8665
198202	1480.50	958.900	99.800	9.4374
198203	1477.10	964.200	99.500	9.9548
198204	1478.80	976.300	99.100	10.560
198301	1491.00	982.500	99.200	10.388
198302	1524.80	1006.20	100.000	10.072
198303	1550.20	1015.60	101.500	9.4558
198304	1572.70	1032.40	102.500	8.4821
198401	1610.90	1044.10	103.700	7.9041
198402	1638.80	1064.20	105.100	7.4824

Figure 1-16. Tabulated Series or Expressions

left and right in order to allow you to view all the columns of the resulting table.

Macro Files

DataView commands may be entered interactively by making menu selections, as described. However, alternatively, a set of commands such as those typed into the command space of the Central Control screen can be formed into a command file, called a Macro File. Such files individually or collectively can be used to perform repetitive operations—or operations that involve the use of a particular sequence of commands. For example, if each month you wished to make a large number of data transformations using both historical data and the most recently released values, these can be included in a macro file and executed monthly, rather than having to retype them each time. Macros can also be used to produce a set of regressions, tables, and other types of output, perhaps as a follow on to the execution of the series of transformations performed after updating one or more data banks.

If, as you have read this chapter, you have tried the examples, you will have seen that DataView automatically displays commands in the command space of the Central Control screen even when you have used the menus or clicked the status bar. Thus, in all cases, working with DataView results in a set of displayed commands. As you may have discovered, whenever commands are visible in this command space, if you double click with your left mouse button while pointing at the space with your cursor, DataView will respond with the question, “Do you wish to create a text file from the contents of this screen?” If you answer “Yes”, you will immediately be presented with a form superimposed on the Central Control screen that describes itself as “DataView Text and Data File Editor: Untitled”. What has occurred is that DataView has shifted into text edit mode, and become for the moment a text editor. Initially the file being edited is “Untitled.”

Using this DataView facility, you can edit the commands as you see fit and at the end, when you click on **File** and then either **Save As** or **Close**, the program will permit you to create a macro file (or update an existing one) using the results. Incidentally, when you click on **Close** (or **Quit**, for that matter), two things will occur. First, that file operation will be performed. Second, DataView will hide the text editor screen, returning you to the set of commands displayed when you first left-clicked with your mouse.

Note also that the file you will have created or updated will contain only the text of the commands, not the data that may have been used earlier in the DataView session. But, if in a later session you reopen any relevant data banks or other data sources, and if you then execute the macro (using **File | Open** or the explicit DataView command **Run**, *typed into the Central Control screen command space*) the program will re-execute the commands in the macro producing once again the same results, *ceteris paribus*.

A macro file so created generally will contain the same commands as you might type in the command space interactively or that are generated automatically by DataView: in the simpliest case, a macro is just a batch of ordinary DataView commands. However, there are also specific commands for use in macros; these are described in the section of DataView's help file that discusses the creation of macros. These commands include those that allow you to do such things as call one macro from another or deal with the possibility that a macro may fail for some reason during execution or else cause a table to begin printing at the top of a page.

Generally, separately from DataView, macros can be created and revised using a text editor, such as Notepad or Wordpad. The limiting restriction on any third party text editor is that to be useful it *must* produce an ASCII file without control characters. Some word processors always embed control characters for formatting or selecting type faces and these must not be used.

The PROFILE Macro

The discussion in this chapter has successively considered a series of operations corresponding to commands that open data banks, set the observation frequency, allow on-the-fly frequency transformations, and the like. As a group, these commands are called *environment* commands, since they establish the environment in which other commands are acted upon. As you work day-to-day, you are likely to find that you tend to use the same or a similar group of environment commands.

An example of a set of environment commands is:

ACCess demobank
SET FREQuency=Quarterly, **DATES**=47-2000
AUTOFREQ on

where the essential portions of the command words are in capital letters, although in fact you can issue the commands in any combination of upper and lower case letters.

Such commands can be issued manually at the beginning of each DataView session, either explicitly or using the menu options. However, they do not need to be. Particularly once you have established a routine, you can put commands to SET FREQUENCY, SET DATES and ACCESS one or more Data Banks into a file called:

PROFILE.MAC

To be recognized by DataView, you must put this file in your Macro directory; this directory type is described briefly at the end of this chapter and in more detail in the DataView *Getting Started Guide*, which you would be well advised to read carefully.

Then whenever you start DataView subsequently, the PROFILE macro will be executed automatically as soon as DataView starts. If you organize your work into projects, as described below, you can create a specific PROFILE.MAC for each project. Note that PROFILE.MAC plays an analogous role with reference to DataView as AUTOEXEC.BAT played to MS-DOS.

File Protection

Short of drastic exits (such as simply turning off the machine) DataView is designed to provide a high level of data and file protection. Data Banks and model files will almost always survive crashes. Even the Memory File (the temporary workspace file) generates a *Crashfile*, which will be preserved in the Home directory. However, in order to recover this crashfile after a crash, before you re-execute DataView it is imperative that you rename it to some other name than its default, CRASH.MFL. Once this renaming has been done, you can re-start DataView, load the renamed Crashfile, and in most cases begin exactly where you left off previously. To load, use either **File | Open** or the explicit command:

LOADMF name-of-file

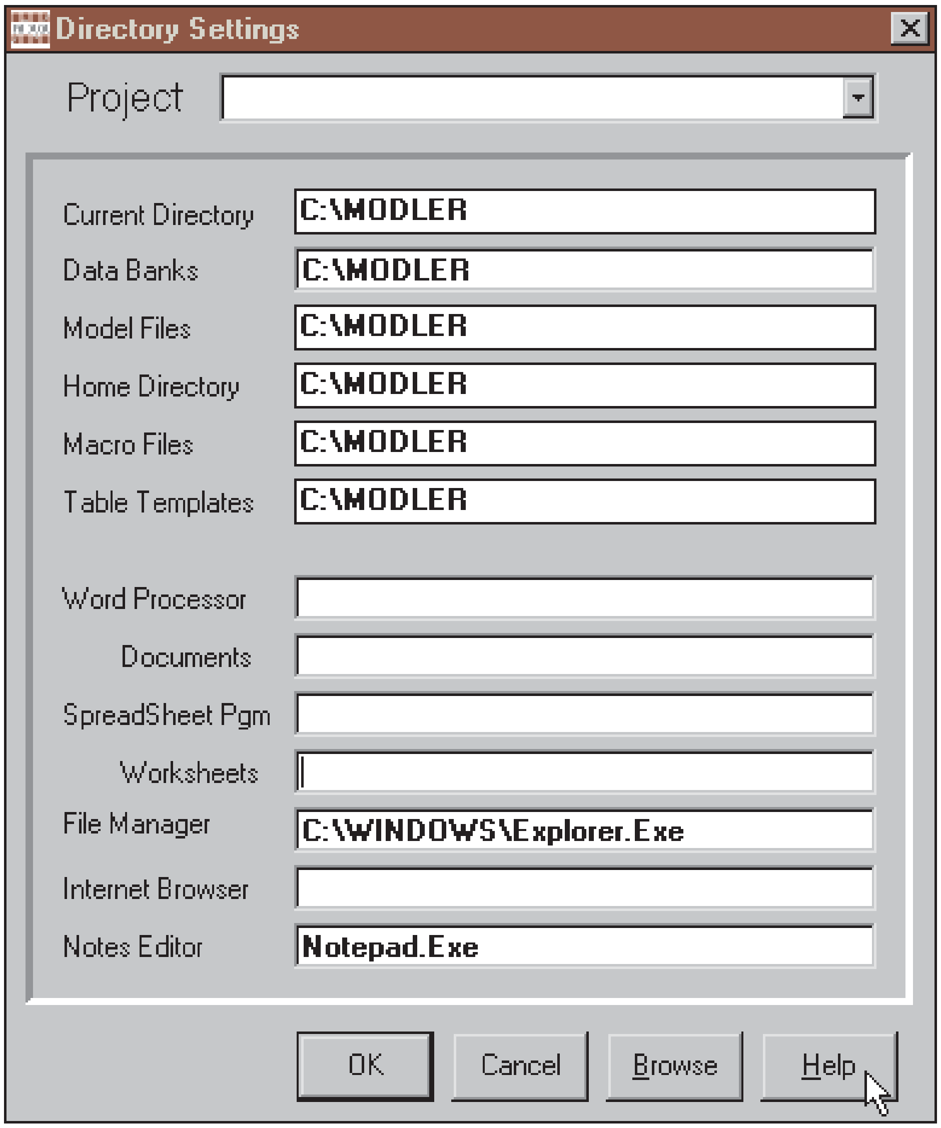
where "name-of-file" is of course the name of the renamed CRASH.MFL.

Organizing Your Work

DataView provides a default setting in most instances, so that generally you can begin without any elaborate program preconfiguration. However, certain settings need to be made for effective operation of the program. If you have not previously read the DataView *Getting Started Guide* and followed the instructions there, you should now press **Settings** and then click on **Directories**. In response, your screen will display the form shown as Figure 1-17 on the next page.

A possible difference between exactly what is shown in this Figure and what your screen shows is that yours may display a different directory than C:\DataView in each of the successive categories. But if you have not previous used DataView for Windows, your display should be similar inasmuch as a single directory will be displayed, possibly C:\WINMOD; whatever its name, this directory is likely to be the default start up directory, the directory into which you installed DataView.

Notice also that this form naturally subdivides into two sections, a top section that relates to the organization of the directories (or folders) that contain variaous types of DataView related files and



Project	
Current Directory	C:\MODLER
Data Banks	C:\MODLER
Model Files	C:\MODLER
Home Directory	C:\MODLER
Macro Files	C:\MODLER
Table Templates	C:\MODLER
Word Processor	
Documents	
SpreadSheet Pgm	
Worksheets	
File Manager	C:\WINDOWS\Explorer.Exe
Internet Browser	
Notes Editor	Notepad.Exe

Figure 1-17 Directory Settings

a bottom section that refers to external programs. Let us consider these two sections in reverse order.

Configuring External Programs for DataView

Focus initially on the bottom half of the form shown in Figure 1-17. In general, the Directories option of Settings has two related, but distinct roles. The bottom half of the Directories Form allows you to identify the word processor, spreadsheet program, file manager, Internet browser, and notes editor that you wish to use to perform DataView-related operations pertinent to these functions; for the word processor and spreadsheet programs, and browser, these specifications include the default directories for associated files.

Looking at the form, you will see immediately that it contains text spaces reserved for the paths and executable file names of programs that can be used with DataView. For example, in the case of a Word Processor, what DataView expects is a specification something like:

C:\MSOFFICE\WINWORD\WINWORD.EXE

where the last portion (WINWORD.EXE) provides the name of the program executable and the first (C:\MSOFFICE\WINWORD\) the path, or directory location, of this executable. The path will of course be specific to your machine and can change, so that from time to time you may need to respecify.

Note also that, for both the Word Processor and Spreadsheet programs, an entry should be made for the default directories for documents and worksheets respectively; these entries should consist only of the path specification; for example:

C:\MYDOCS

without a trailing \. All directory specifications and filenames must be the short form that all versions of Windows actually directly use—not the long form that Windows 9.x and later versions of this operating system provide as indirect references. Furthermore, the directories specified must already exist for DataView to allow you to include them here. Note that the Browse button can be used to discover these directories.

The Internet Browser you may wish to use with DataView is specified in much the same way as the Word Processor and Spreadsheet; however the File Manager and Notes Editor are handled differently. As Figure 1-18 implies, DataView on its own will initially select what it "thinks" is the most advanced file manager for your particular operating system, but if you wish you can choose another, including a third party file manager. Similarly, DataView initially selects Notepad.Exe for your working notes, but you can change this to Wordpad or some other simple editor if you wish.

Incidentally, a conceptual distinction between a Word Processor and the Notes Editor is essentially the speed with which they can be invoked and the degree of complexity of operation; in general, both Notepad and Wordpad are much simpler programs, much more ready to be put into a corner of your screen in order to make notes as you work than are programs such as WordPro, Word, or WordPerfect.

Organization of DataView Files: Multiple Projects Support

Looking once again at the form in Figure 1-18, its top half obviously controls the directories into which DataView puts files of various types; filling it in appropriately therefore provides you with the capability to organize your work in a variety of different ways. A fundamental choice, when specifying the directories to use with DataView, is between organizing your work into distinct "projects" or not.

Looking at Figure1-18, you will see that near the top the word **Project** appears, followed by an (initially blank) dropdown textbox. If your needs are simple, you can ignore this option. Moreover, later, should you feel that you need to organize your work into projects, you can come back and do this. However, do not use this facility just because it is there: once you have begun to organize along project lines it can be difficult to simplify again, although there is a brute force method to achieve this: deleting the file DataView.INI from the main Windows directory, which will re-initialize all elements of the Directories Form and wipe out all central references to projects—but if you take this drastic step, you should first save DataView.INI to another directory or backup medium before deleting it from the Windows directory, just in case.

Simply Specifying Directories

The files associated with DataView consist of data banks, model files, and macro files, to cite only the major categories. One option is simply to put all files into a single directory or folder. An alternative, which DataView just as easily supports, is to specify a particular directory for data banks, another for model files, and yet another for macros; as you wish, these can be set up as independent directories—or subdirectories of some master directory (for instance, \DataView). However, in addition to files that you will consciously create, such as these, there are a variety of files that DataView creates, more or less automatically, as needed for special purposes. For example, the Memory File is automatically created whenever the program needs a place to put a transformation or generated variable. There are in addition more esoteric temporary files that DataView uses and creates as needed.

DataView usually places these automatically-created files in what it calls a *Home Directory*. A characteristic of such a directory is that it generally does not change during a session, no matter how much you move around your hard disk. By default, the current directory when a DataView session begins is adopted as the Home Directory. Note that the Home Directory is therefore not necessarily a physically separate and distinct directory: it is conceptually any directory that DataView uses in this particular way. It is only a separate directory when you uniquely specify a particular directory to be the Home Directory. And it only changes during a session if you explicitly change it.

But be aware that these protocols are simply organizing principles. It is up to you to put them into effect. One option is to use a single directory for everything. If you do nothing, this is the way DataView will organize your work and, generally, the single directory that the program will choose will either be the directory you selected when you first installed the program, or else the first directory ever used for a DataView session. DataView is designed to operate in a make-do fashion in whatever environment it finds itself, but this is not a recommendation that you leave your destiny to the caprices of a machine.

Assuming that you wish to be captain of your fate, begin your first session by clicking **Settings** on the main menu of the Central Control screen and then **Directories**. You will probably find that DataView has used a single directory, as just described and as illustrated in Figure 1-19. However, whatever you discover, you will also see that the Directories Form exhibits an implicit organization by file type. Note the categories:

Current Directory
Data Banks
Home Directory
Macro Files
Table Templates

considering only the top half of the form. As mentioned earlier, at the top, above this list, will be the word Project with a blank drop-down textbox to the right; for the moment, ignore it.

The files types associated with DataView can be categorized by extent, and on this basis consist of 15 different types, not counting various data transfer files that may be used to import data. Obviously, if you use DataView intensively, leaving all these files in a single directory may soon yield a mess. One way to organize your work is to establish a parent directory and then a set of subdirectories that identify the types of files. If you wish to work this way, you will first need to create this directory structure and then the subdirectory paths. Once you have created these, fill in the Directories Form during your next DataView session. At the end of that session, as you exit normally, the program will create (or, subsequently, edit) the textfile called DataView.INI located in the main Windows directory. Thereafter, when you begin each subsequent session, DataView will immediately load the specifications from DataView.INI so as to re-establish this structure; note that DataView.INI must be located in this particular directory.

If, when working interactively, you use DataView commands to open data banks, run macros, attach models, and so on, a disadvantage of this organization may be that in some cases you will need to specify path directories as well as filenames. However, if you use the menus, or if you issue command words followed by <Enter> (so that DataView displays available files of the relevant type), this organization may simplify your work over time. It will certainly organize it better.

Organizing Your Work By Project

If your work involves the use of distinct sets of files at distinct times, with little or no overlapping use, then organizing your work by project may make sense. If you click on the dropdown Project textbox at the top of the Directories Form, you will immediately see that it has one entry <Add New Project>. If you make this choice an Input Box will appear, asking you to provide the name of a Project, and giving you space for as many as 80 characters. If and when you provide a name and click OK, two things will happen. First, a Project of that name will be created and, second, DataView will place in the currently-specified Home Directory a file it creates; this file is always named PROJECT.INI. At the same time, DataView will amend the DataView.INI file (found in your main Windows directory) to include both the project name and the path for the just-mentioned Home Directory. As you add projects, the PROJECTS section of the DataView.INI file will grow, and the designated Home Directory for each project will each receive a PROJECTS.INI file. In the future, as appropriate, these files will be dynamically updated whenever you make changes using the Directories Form.

Actually, during the project creation process, nothing permanent happens until you both identify the project and click the OK button at the bottom of the Directories Form. So that once you have provided a project name, and before you click that OK button, you have a chance to set up your various directories appropriately, including the Home Directory. Or, you can set up the directories first, and then click OK. If you are curious, you can inspect both the DataView.INI file and the PROJECT.INI file, once you have crossed this Rubicon. These are both simple text files, but be careful not to change them unintentionally. It also might be a good idea, periodically, to save backup copies of these files, carefully organized of course, since the name of the PROJECT.INI file is common across projects.

Note that once you start working, whenever you open a project and then change some directory or program specification in the Directories Form, these changes will be recorded in the Data-View.INI file and the appropriate PROJECT.INI file. As mentioned earlier, these files are dynamically updated. If you wish to change the name of a project, you need only to change the name in the dropdown text box at the top of the Directories Form, and then click OK at the bottom. However, do not be mislead by this simplicity into thinking that these INI files control the actual, physical organization of directories on your machine. They do not: they simply point to existing directories (and programs) and provide a conceptual organizing structure. When you change a project's specifications you do not affect the actual directory(ies) involved. Its also true that if and when you arbitrarily change the actual directory structure of your machine (or, in a network context, change the organization of the network) affecting your projects, you will usually need to make the appropriate changes in your project files, using the Directories Form.

To delete a particular project, by the way, simply erase the project name in the drop-down Project text box at the top of the Directories Form. Then immediately press the OK button at the bottom of the form. DataView will respond by asking you to confirm that you wish to delete that project. If you answer Yes the deletion will occur provisionally, although no files will yet be affected. It is then best to exit from DataView immediately and re-start the program. What occurs during the project delete operation is that the list of projects found in DataView.INI at the start of the session is re-written to omit the deleted project, but this re-writing of DataView.INI takes place only when the session ends.

As a fail-safe, if you delete a project, DataView does not delete nor change any files other than DataView.INI. In particular, the PROJECT.INI file in the Home Directory of the deleted project is not changed in any way. Consequently, if you previously saved the DataView.INI file from the main Windows directory, after exiting DataView, you can actually undo the effects of the deletion simply by copying the saved DataView.INI into the main Windows directory over top the new one generated as the result of the deletion.