Paper 255-31

The VIEWTABLE Window - A Useful but Underappreciated Tool

Venky Chakravarthy, BioPharma Data Services, Ann Arbor, MI

ABSTRACT

The SAS® VIEWTABLE window is part of SAS/BASE®. This paper focuses on the some of the features of the VIEWTABLE that are easy to overlook. By examining and highlighting some of its capabilities, a rational assessment can be made for the need to license additional products such as SAS/FSP®. The latter, although a very powerful product with many uses, may be unnecessary for some organizations. Hence it is worth while to take a closer look at the VIEWTABLE.

The appropriate audience for this paper is an Intermediate to an advanced level SAS user but a beginner should be able to follow most of this discussion. In any case the VIEWTABLE is covered from the basics.

INTRODUCTION

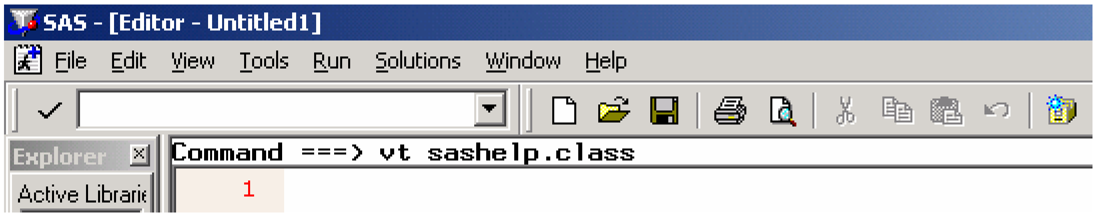
Written mainly with the Microsoft Windows user in mind, it shows how to (1) open a data set using the VIEWTABLE window, (2) create a new data set, (3) view your remote data via a SAS/CONNECT® session, (4) sort your data in place and view the results, (4) filter your data with WHERE clauses, (5) choose the columns you want to display and (6) customize your VIEWTABLE window.

In order to use this facility you should be in an interactive SAS session that uses the Display Manager. Only the BASE product is required for the VIEWTABLE but its reach can be extended with SAS/CONNECT®. This paper has applications in many industries but some illustrations have data that will be most familiar to the Pharmaceutical audience.

1. OPENING A DATA SET IN A VIEWTABLE WINDOW

Let us start with the basic operation of opening a VIEWTABLE window. There are two ways to open the VIEWTABLE window. You can either click on the data set from the SAS Explorer window or type "**VIEWTABLE**" ("**VT**") on a command line (or box). It is convenient to start with the SASHELP library as several datasets already exist. Figure 1, below is a screen capture of the command line method.

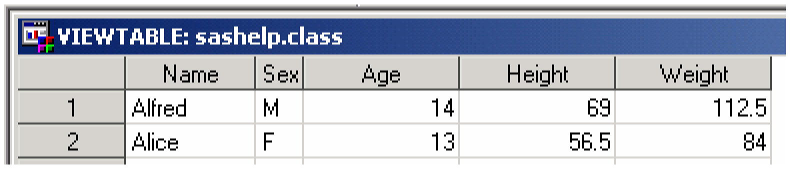
FIGURE 1


SAS - [Editor - Untitled1]
File Edit View Tools Run Solutions Window Help
Explorer
Active Librarie

```
Command ===> vt sashelp.class
1
```

The same command could be typed in the command box that is visible just below the menu and just above the command line. Either method opens the CLASS data set in the SASHELP library (**browse** mode) as seen in part below (see Figure 2).

FIGURE 2



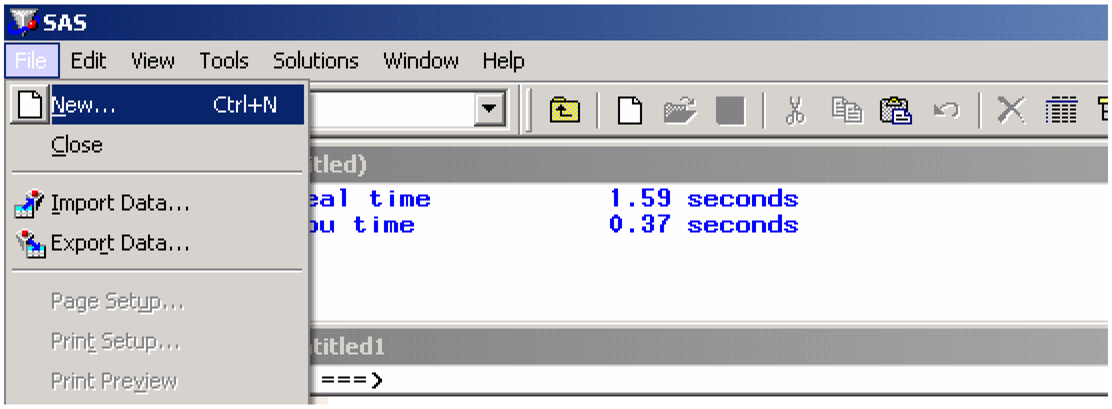
	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69	112.5
2	Alice	F	13	56.5	84

When using the command line method, the dataset opens in BROWSE mode by default. Also by default the Column labels are displayed. There are ways to always display the variable names or you can simply do it for the session. If you have write permissions you can access a dataset in EDIT mode. You could do this even when you opened it in BROWSE mode as you will see a little later.

2. CREATING A NEW DATA SET

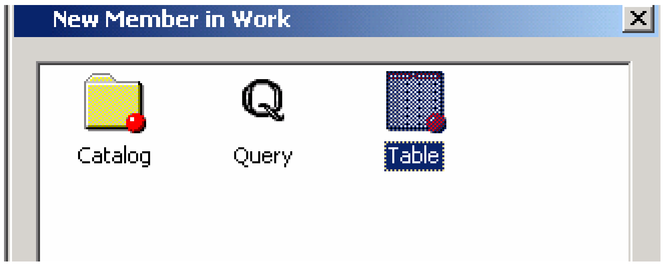
Having seen how you can open an existing data set, let us see how to create a data set using the VIEWTABLE. Once again we can use any of the two methods and this time let us use the **File menu** while keeping the **SAS Explorer** as the active window. First, drill down to the WORK LIBRARY in the Explorer window. You are now ready to create a data set that is available as long as the interactive session lasts. Shown below are some screen captures of the actions necessary to bring up an editable table that can be saved. Remember to make the SAS Explorer the active window when you enter the file menu (see Figure 3).

FIGURE 3



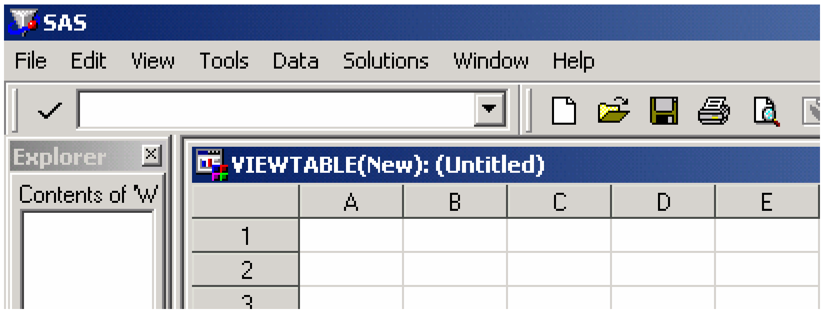
Once you click on **New** you should see a screen as in Figure 4 below:

FIGURE 4



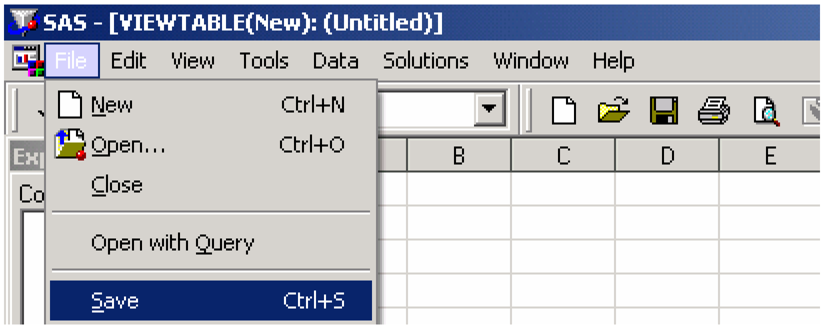
We are not interested in **'Catalog'** or **'Query'**. Click on **Table** and then **OK**. This brings up an Excel like sheet as seen in part below in Figure 5. Now you are ready to define columns and input data.

FIGURE 5


SAS
File Edit View Tools Data Solutions Window Help
Explorer
Contents of 'W
VIEWTABLE(New): (Untitled)
A B C D E
1
2
3

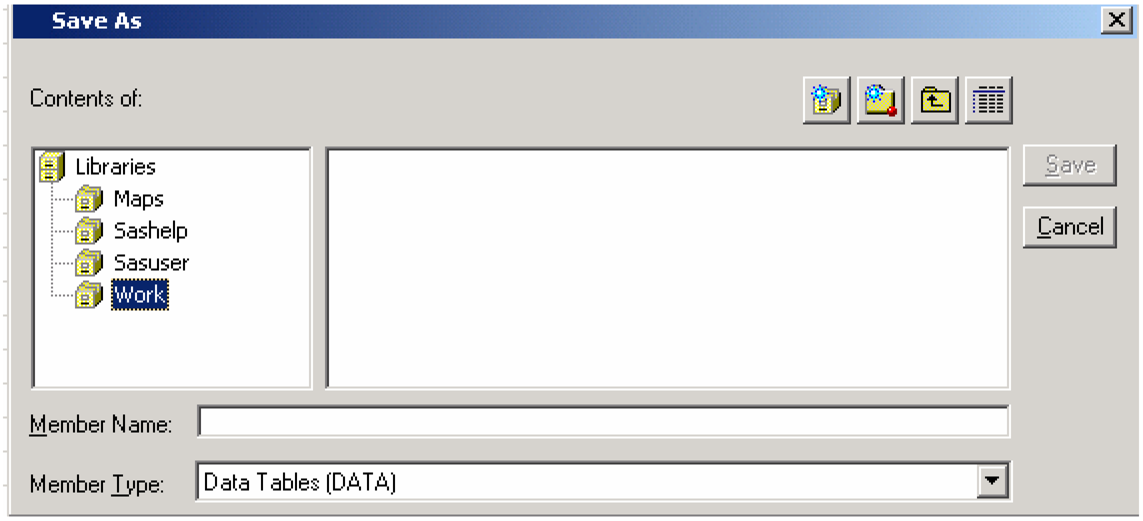
Notice that the menu is available and the columns (variables) are named with alphabets and the rows (observations) with numbers. Also on the VIEWTABLE title bar, **NEW** and **UNTITLED** are in parentheses. This is your prompt to give the data set a name for which we start with the operations in Figure 6.

FIGURE 6


SAS - [VIEWTABLE(New): (Untitled)]
File Edit View Tools Data Solutions Window Help
New Ctrl+N
Open... Ctrl+O
Close
Open with Query
Save Ctrl+S
B C D E

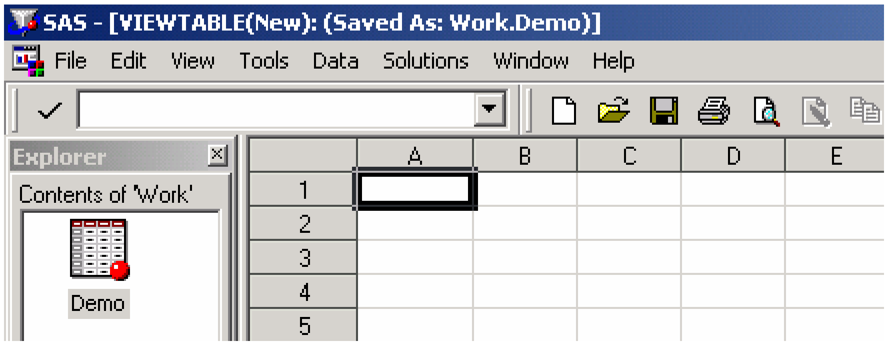
Using the file menu you can save the data set to the library of your choice. The action in Figure 6, brings up a Save As screen (see Figure 7). For our purposes let us save data sets in the WORK library throughout this discussion.

FIGURE 7



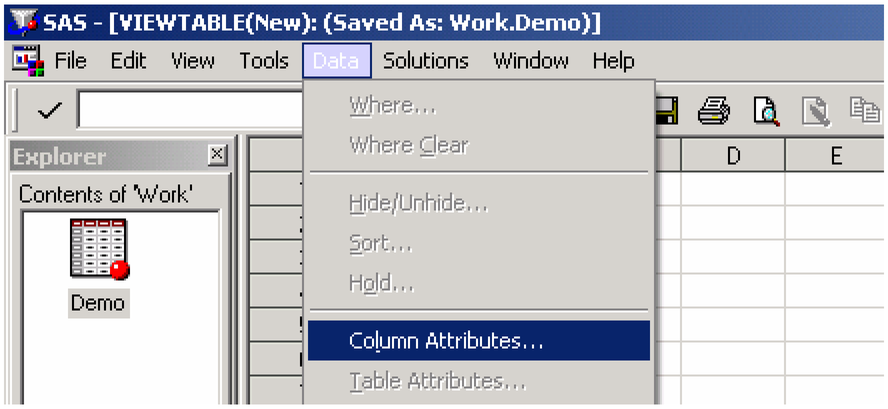
You can type DEMO in the **Member Name** field to un-gray **"Save"**. Once done, the title Bar on the VIEWTABLE window will alter itself. It will still say "New", but now it has a name for the data set (see Figure 8). You can also see the DEMO data set in the Explorer Window under the WORK library.

FIGURE 8


SAS - [VIEWTABLE(New): (Saved As: Work.Demo)]
File Edit View Tools Data Solutions Window Help
Explorer
Contents of 'Work'
Demo
A B C D E
1
2
3
4
5

You have no doubt noticed that the first cell was clicked before the SAVE occurred, so it remains the active column for any action via the pull down menu. Navigate to **'Column Attributes'**, the only Non-Gray item under the **Data** menu (see Figure 9), and click on it.

FIGURE 9

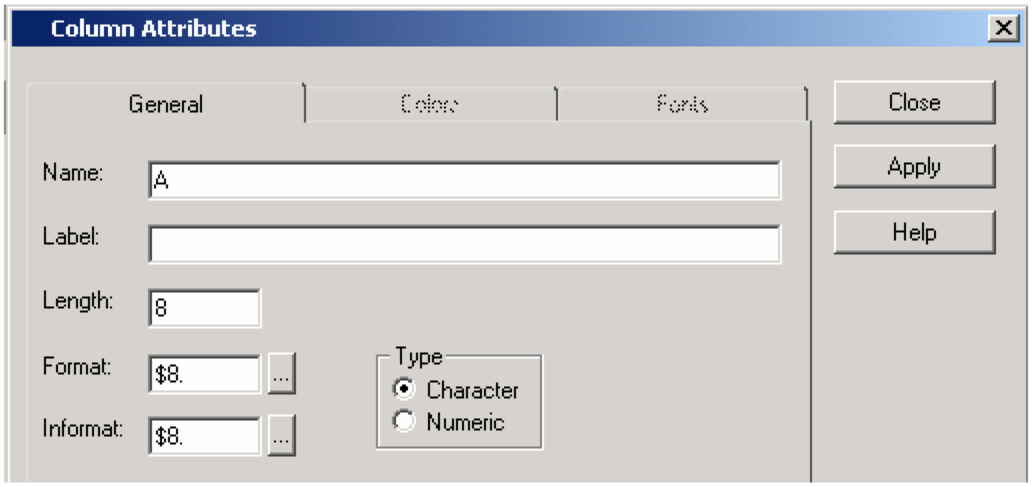


Now this will bring up another screen for action. You will notice a few things that are seemingly at odds with the general way that you expect SAS to work when creating a data set. In a data step, when you do not specify the data type for a variable and use it, SAS assumes it to be a numeric field. To illustrate:

```
5    Data _null_ ;
6      gender = sex ;
7      set sashelp.class ;
ERROR: Variable sex has been defined as both character and
numeric.
8    run ;
```

In line 6, we did not specify the variable TYPE for SEX before we used its value (missing) to create GENDER. SAS treats such variables as NUMERIC. Now, the SET statement compiles and executes after LINE 6. Unfortunately SEX is defined as CHARACTER in the input data, hence the ERROR. Now contrast that to the screen below (see Figure 10) that seemingly shows **CHARACTER** type by default when you use the VIEWTABLE to create data.

FIGURE 10



Name: A

Label:

Length: 8

Format: $8.

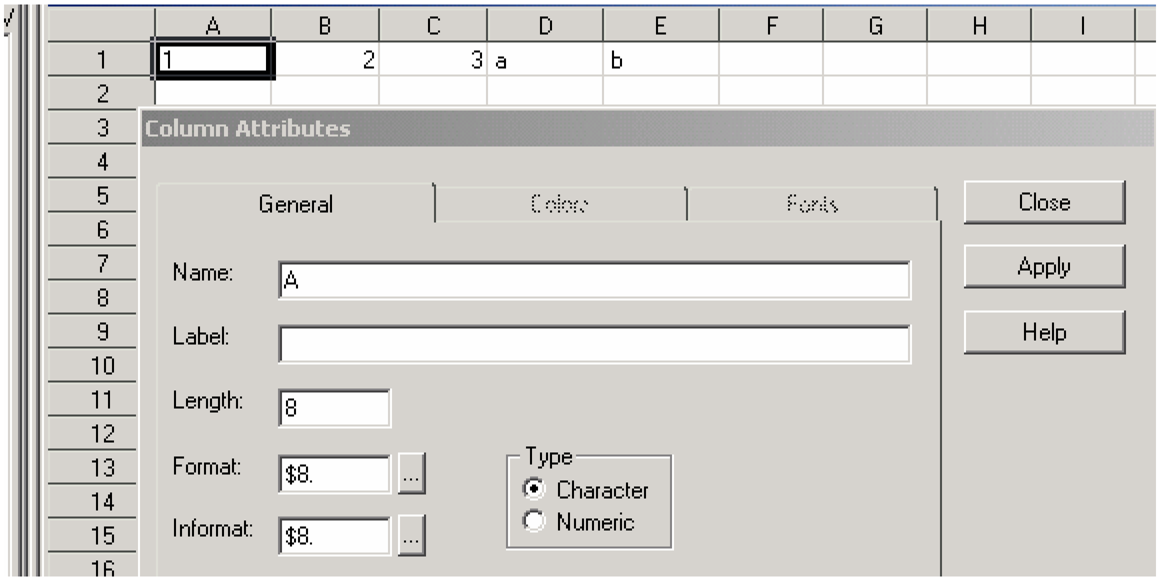
Informat: $8.

Type: [✓ Character] Numeric

When you open the Column Attributes sub window, **before entering any data** in the VIEWTABLE, the columns are **CHARACTER** type by default. Notice the emphasis on “before entering any data”. This is critical. Once you enter the Column Attributes and close it without changing anything, the data type is defined as Character for that column. However, unlike in a data step, you can change the TYPE from CHARACTER to NUMERIC as long as it still remains a new data set. Remember that it is no longer considered “new” when you close and open within the same session. That would be no different than creating a temporary data set in a data step (or SQL) and using it in the very next step. You will not be able to change the data type of existing variables. It becomes true meta data.

In the above paragraph, we emphasized **before entering any data** for a reason. If you have not set the variable type via **Column Attributes** from the menu, the VIEWTABLE **automatically determines** the variable TYPE by considering the input in the first row. You can see how this happens as we type into the columns A through E. Keep in mind that we have already defined column A as Character by simply opening and closing the Column Attributes window with an active cell under column A. So even if you type a number in the first row it is not going to change the variable type as seen below in Figure 11. Contrast that to Figure 12.

FIGURE 11



	A	B	C	D	E	F	G	H	I
1	1	2	3	a	b				

Column Attributes

General | Colors | Fonts

Name: A

Label:

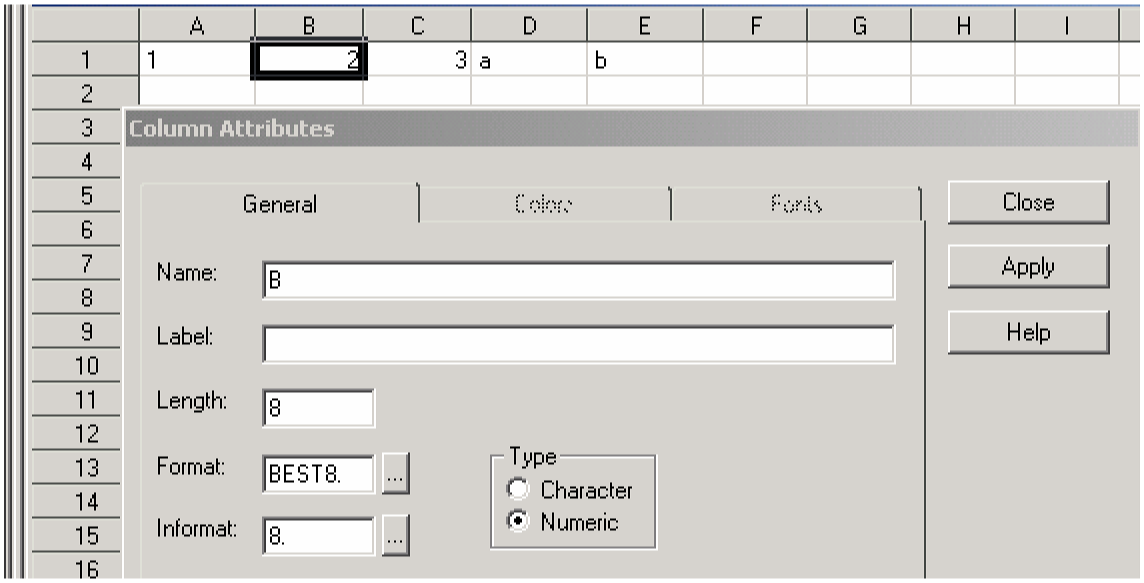
Length: 8

Format: $8.

Informat: $8.

Type: [✓ Character] Numeric

FIGURE 12



	A	B	C	D	E	F	G	H	I
1	1	2	3	a	b				

Column Attributes

General | Colors | Fonts

Name: B

Label:

Length: 8

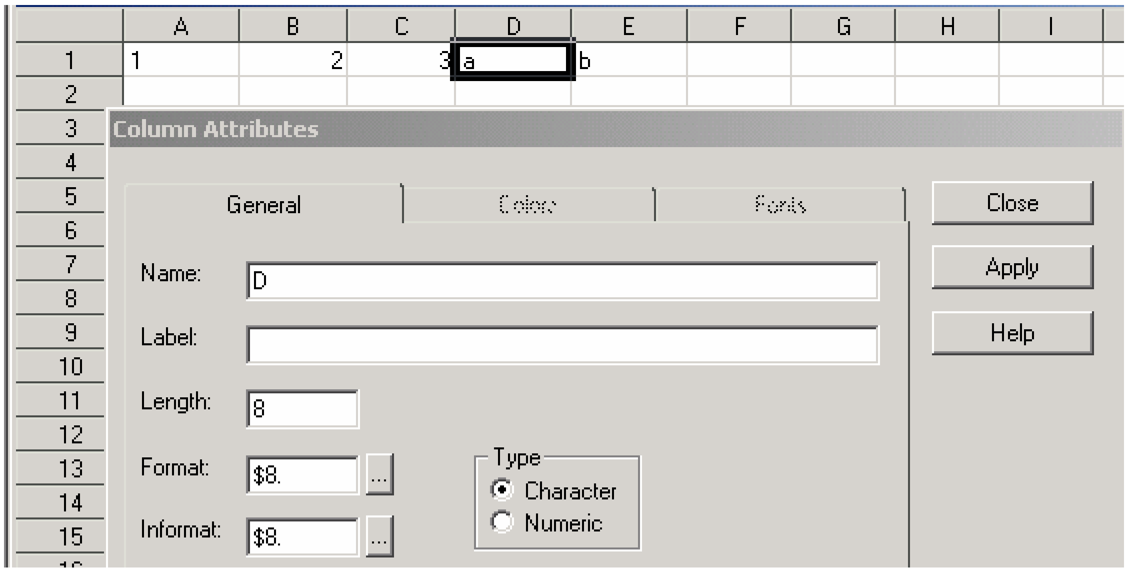
Format: BEST8.

Informat: 8.

Type: Character [✓ Numeric]

The variable type is automatically assigned as NUMERIC. We can also crudely guess from the alignment of the number within the cell. When it is right justified it is NUMERIC and CHARACTER when left justified. The automatic detection is similarly applied for character values as seen below in Figure 13.

FIGURE 13



	A	B	C	D	E
1	1	2	3	a	b

Column Attributes

Name: D

Label:

Length: 8

Format: $8.

Informat: $8.

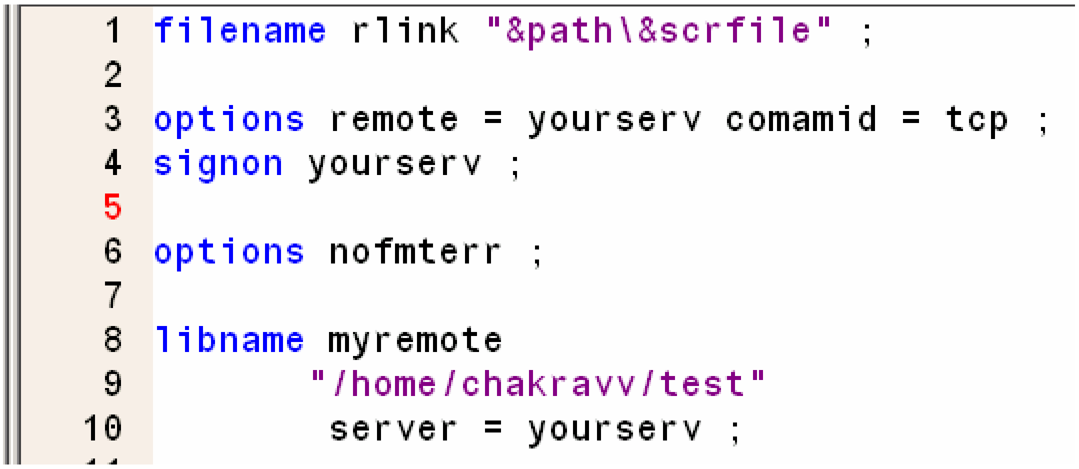
Type: [✓ Character] Numeric

The entry of data is trivial so we will not go into further depth on that. However, there is one thing to keep in mind when entering values requiring special INFORMATS. You will need to set both INFORMAT and FORMAT for easier entry and display. For example, without an INFORMAT you will have to input **date** values as number of days from January 1, 1960.

3. ACCESSING DATA FROM A REMOTE LOCATION

In the above sections we saw how (1) to open an existing data set and (2) to create a new data set. This section extends it to accessing data from a remote location. This will require SAS/CONNECT, an additionally licensed product. All it takes to extend the reach of the VIEWTABLE is to start a CONNECT session as in Figure 14 below, and assign a LIBREF to the remote path. There are some peculiarities to the CONNECT syntax that we will not go into detail. An example is the macro variable YOURSERV whose resolution is not effected with the traditional “&” prefix.

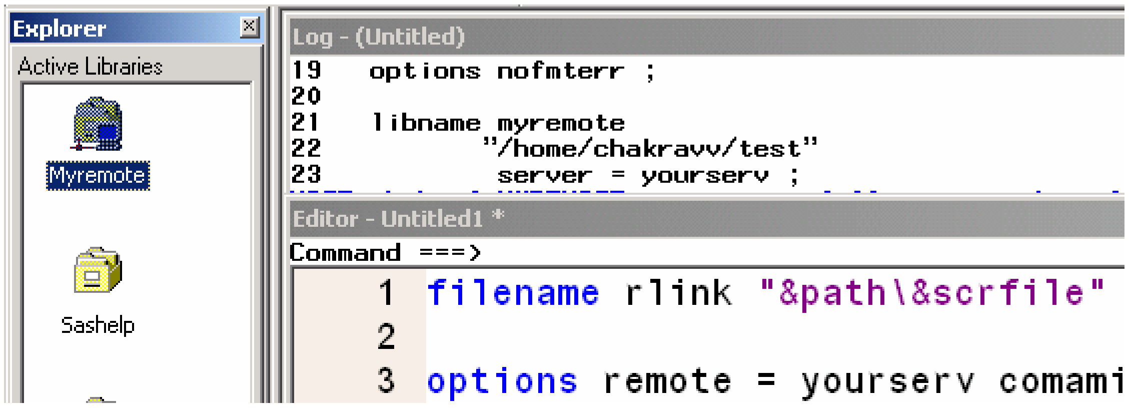
FIGURE 14



```
filename rlink "&path\&scrfile" ;

options remote = yourserv comamid = tcp ;
signon yourserv ;

options nofmterr ;

libname myremote
        "/home/chakravv/test"
         server = yourserv ;
```

After executing the above and filling in the ensuing USERID and PASSWORD prompts, your Local (Client) SAS session tries to start SAS on the remote machine YOURSERV (server). The Client session is the master session that controls the remote operation. Using the Remote Library Services (RLS) we have also assigned a LIBREF to the remote path using the REMOTE engine. In Figure 15 below, the left most section is the Explorer window that displays several LIBREFs. The MYREMOTE libref icon is a little different with a small computer embedded in front of the cabinet. This indicates the use of the REMOTE engine.

FIGURE 15



```
options nofmterr ;

libname myremote
        "/home/chakravv/test"
         server = yourserv ;
```

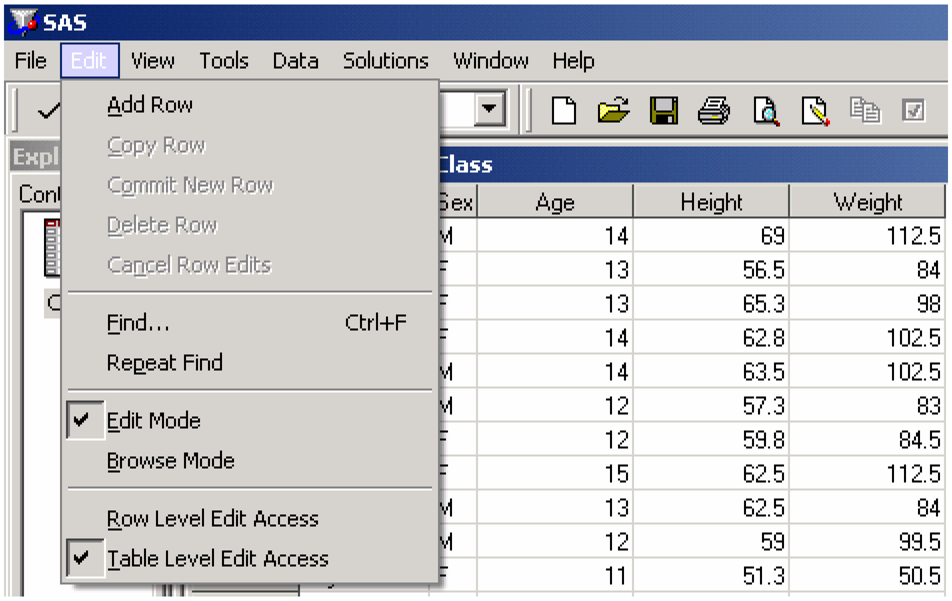
```
filename rlink "&path\&scrfile"

options remote = yourserv comami
```

Now data sets are available for viewing from your remote server. Simply click on the MYREMOTE icon to see a list of the data sets and then click on any of them as you did with the SASHELP library before. Additionally, if you have write access you can update or even create additional data sets.

4. SORTING A DATA SET

So far you have seen how to access data sets (locally and remotely). You have also seen how to create a data set. Now we can go further with what we can do with that data in a VIEWTABLE environment. This section covers how to SORT the data.

FIGURE 16

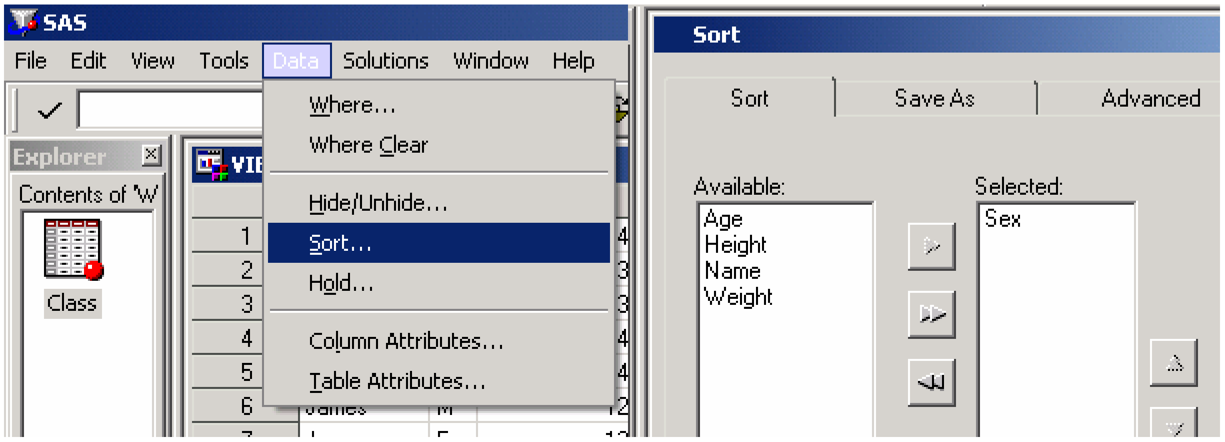


Sex	Age	Height	Weight
M	14	69	112.5
F	13	56.5	84
F	13	65.3	98
F	14	62.8	102.5
M	14	63.5	102.5
M	12	57.3	83
F	12	59.8	84.5
F	15	62.5	112.5
M	13	62.5	84
M	12	59	99.5
F	11	51.3	50.5

In order to SORT the data, edit access is required at the "table level". The actions on Figure 16 or their equivalent are required assuming that a copy of the SASHELP.CLASS data set exists in the WORK library.

Notice that by clicking on Edit mode, the ROW level edit access and TABLE level edit access become un-grayed. Although not applicable to this particular instance, Table Level access, in general, prevents others from accessing the table in a shared environment. Row level access locks only the particular row that you are editing but otherwise the table becomes accessible to others for editing. We need Table Level Edit access for sorting. So check that and proceed to the DATA menu and select SORT and then SEX as in Figure 17.

FIGURE 17



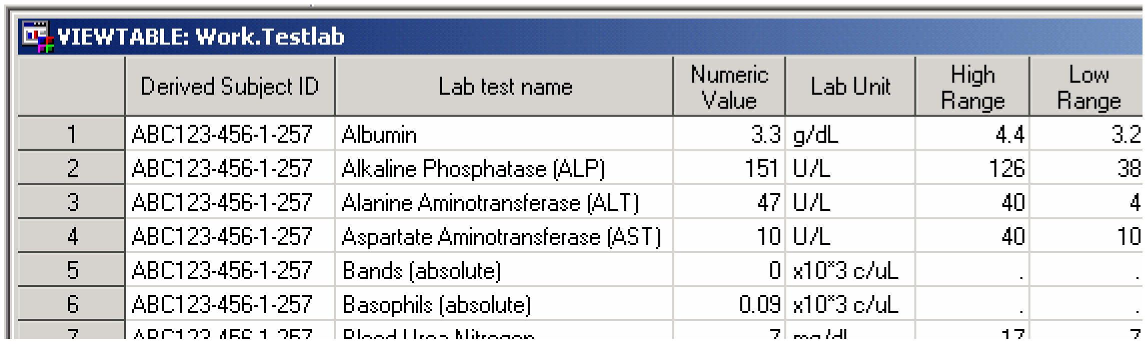
One small note if you are in BROWSE mode and you enter the SORT menu. You will be prompted to save in another location or with another name in the same directory. Once done, the saved data set automatically becomes available in EDIT mode.

5. SUBSETTING OBSERVATIONS WITH A WHERE CLAUSE

One of the most important features of the VIEWTABLE is the ability to apply filters. There are two types of data reduction techniques. One reduces the number of rows and the other reduces the number of columns. In this section we examine how to reduce the number of observations to examine only those of interest to us from a clinical standpoint within a Pharmaceutical human clinical trials environment.

There is a lot of information collected to demonstrate the safety of the drug to get an approval from the Food and Drug Administration (FDA) to market it. One of the data sets assembled in the process, contains the various lab values collected from patients for each visit to the various trial centers. Most lab measures have a normal range. For example in a Cancer trial, within the Hematology group of lab tests, the Normal range for Hemoglobin is 11.2 - 15.2 in g/dL. Study Scientists may be interested in patients whose measures fall below 11.2. Also of interest may be missing lab values. A partial LAB data set with some key variables is displayed in Figure 18.

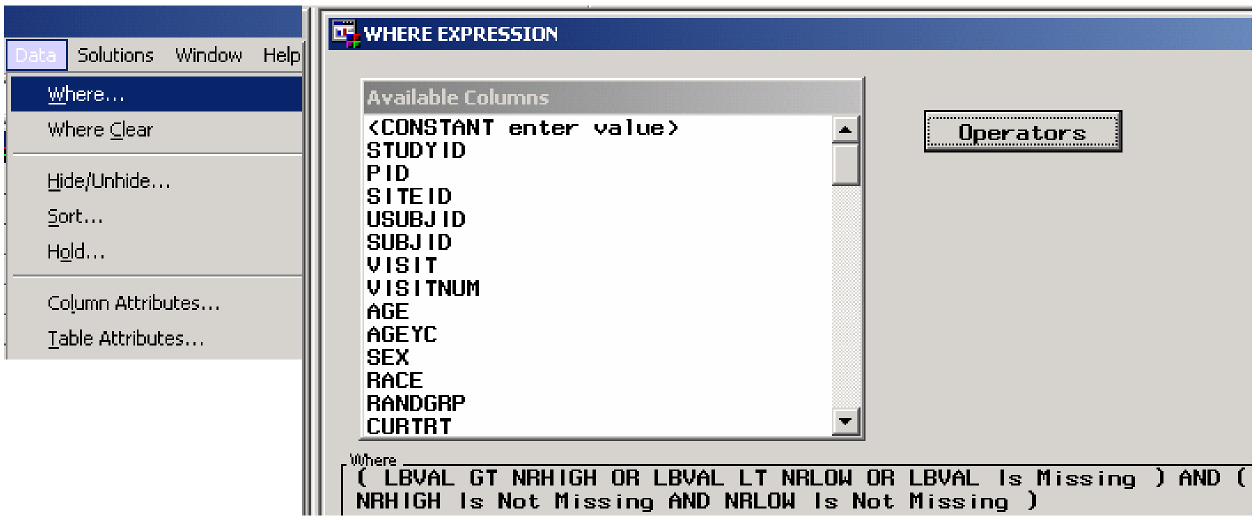
FIGURE 18


VIEWTABLE: Work.Testlab

	Derived Subject ID	Lab test name	Numeric Value	Lab Unit	High Range	Low Range
1	ABC123-456-1-257	Albumin	3.3	g/dL	4.4	3.2
2	ABC123-456-1-257	Alkaline Phosphatase (ALP)	151	U/L	126	38
3	ABC123-456-1-257	Alanine Aminotransferase (ALT)	47	U/L	40	4
4	ABC123-456-1-257	Aspartate Aminotransferase (AST)	10	U/L	40	10
5	ABC123-456-1-257	Bands (absolute)	0	x10*3 c/uL	.	.
6	ABC123-456-1-257	Basophils (absolute)	0.09	x10*3 c/uL	.	.
7	ABC123-456-1-257	Blood Urea Nitrogen	7	mg/dL	17	7

The ALP and ALT tests for this patient returned higher than normal values. So this is of interest. Missing values, although not seen here, are also of interest as we may want to investigate why they are missing. Is it because data were not entered? Or were the data not collected in the first place even though the study protocol mandated it? If these are observed in the early stages, the Study Managers may take this up with the offending trial center and ensure that they are collected for future visits. Now, going back to our table in Figure 18, it only has partial values and only for a single patient. There are hundreds of patients and there are more lab tests in a study. So it would make it easier if the view is restricted to only such patients with missing, above normal and below normal values. In short a WHERE clause is needed to filter the observations.

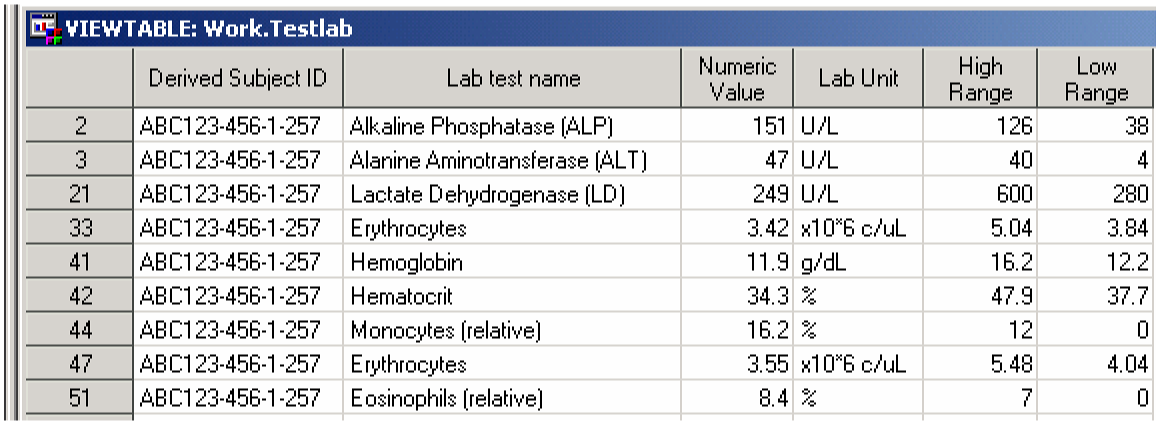
FIGURE 19



```
( LBVAL GT NRHIGH OR LBVAL LT NRLOW OR LBVAL Is Missing ) AND (
NRHIGH Is Not Missing AND NRLOW Is Not Missing )
```

At the bottom right corner of Figure 19 we can see the WHERE expression that was populated by clicking on a combination of Available columns and operators. Upon execution, we get the desired results displayed in Figure 20.

FIGURE 20


VIEWTABLE: Work.Testlab

	Derived Subject ID	Lab test name	Numeric Value	Lab Unit	High Range	Low Range
2	ABC123-456-1-257	Alkaline Phosphatase (ALP)	151	U/L	126	38
3	ABC123-456-1-257	Alanine Aminotransferase (ALT)	47	U/L	40	4
21	ABC123-456-1-257	Lactate Dehydrogenase (LD)	249	U/L	600	280
33	ABC123-456-1-257	Erythrocytes	3.42	x10^6 c/uL	5.04	3.84
41	ABC123-456-1-257	Hemoglobin	11.9	g/dL	16.2	12.2
42	ABC123-456-1-257	Hematocrit	34.3	%	47.9	37.7
44	ABC123-456-1-257	Monocytes (relative)	16.2	%	12	0
47	ABC123-456-1-257	Erythrocytes	3.55	x10^6 c/uL	5.48	4.04
51	ABC123-456-1-257	Eosinophils (relative)	8.4	%	7	0

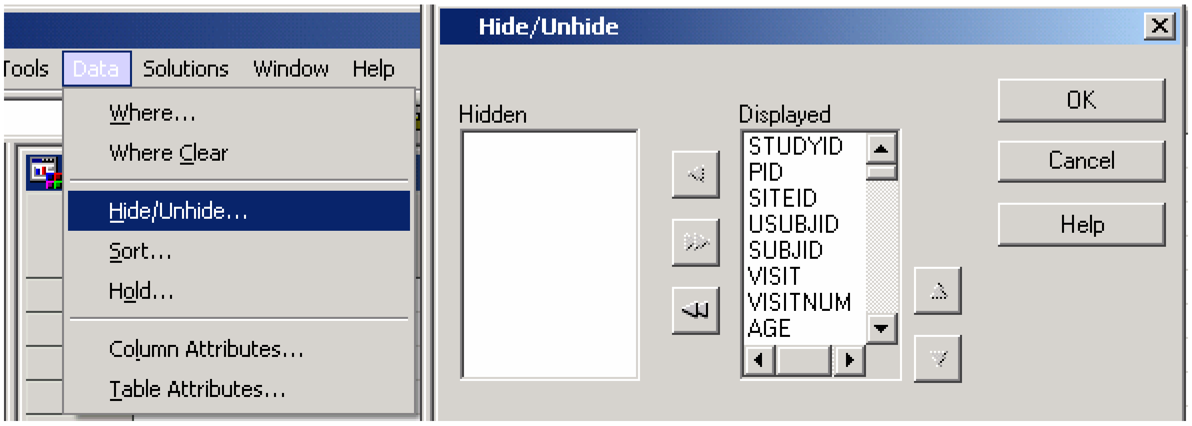
The WHERE expression acts on the data set only for presentation purposes and does not alter the data set. even in Edit mode. The SAVE option is disabled and if this needs to be saved it can be done separately to a different data set. One of the other options is to Save as HTML. It is quite useful to publish it in an internal site for other team members to view the selected observations.

6. SELECTING COLUMNS WITH HIDE/UNHIDE

We have covered one type of data reduction using the VIEWTABLE, i.e. reducing the number of observations. In this section, we will cover the other type of data reduction - reducing the number of variables. Pharmaceutical industry datasets are often wide. Presentation is a problem in a single screen if we do not choose our variables of interest carefully.

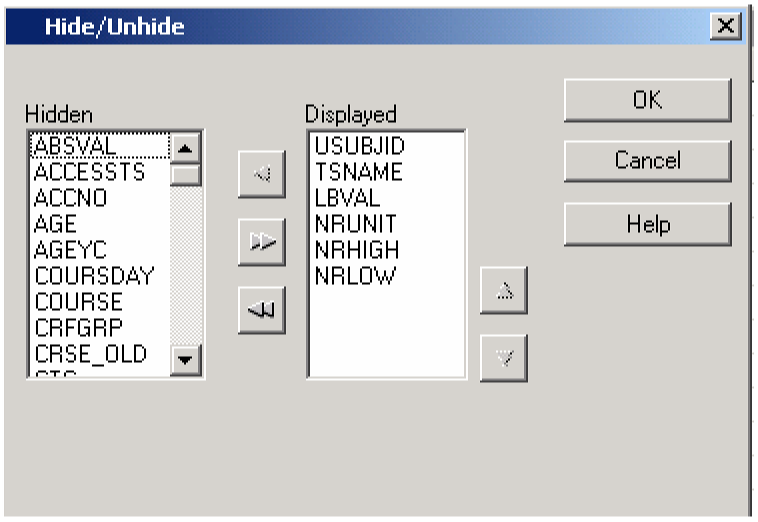
Let us go back to the LAB data set used in the earlier example (see figure 20). It exemplifies the clarity of the data because all the noise columns are thrown out. However, in its raw form this data set contains 80 columns out of which we chose to display only six. Even if we scroll horizontally it would take a while before we find some variables of interest. This, of course, is clumsy. Now how did we go about selecting only the columns that we needed to display and also display them in the order of our liking from Left to Right?

FIGURE 21



We already know that we have 80 variable and only 6 are needed. So it is first convenient to move all the variables from “Displayed” to “Hidden” in Figure 21. We do this with the double arrow button pointing toward Hidden that is between Displayed and Hidden. After that we can choose the six columns of interest. Notice that there are two arrow buttons to the right of the “Displayed” variables. We use them to move the variables up or down in the list. The top most variable is the left most in the VIEWTABLE window. Once done, the Hide/Unhide Screen appears as in Figure 22 below which of course displays the table as in Figure 20.

FIGURE 22

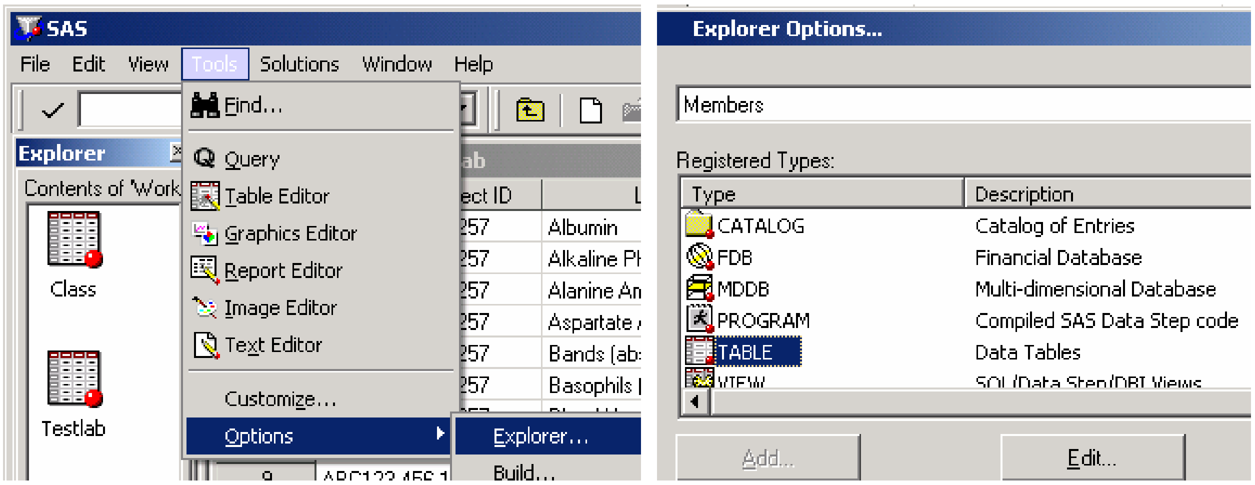


7. DISPLAYING COLUMN NAMES

You may have noticed that the “Displayed” field in Figure 22 has variable names whereas we had labels for them in Figure 20. How could we possibly guess that TSNAME is the variable containing the names of the lab tests? It is somewhat non-intuitive. Would it not be more useful to display the variable names in the VIEWTABLE so that WHERE expressions and HIDE/UNHIDE features can be done more easily?

We can choose to do this from either the Menu or the command line but that would be temporary and last only as long as the session. However, it is possible to permanently change the default display from Column Labels to Column names through the following actions in Figure 23. It alters the registry so the settings are saved for all future sessions.

FIGURE 23

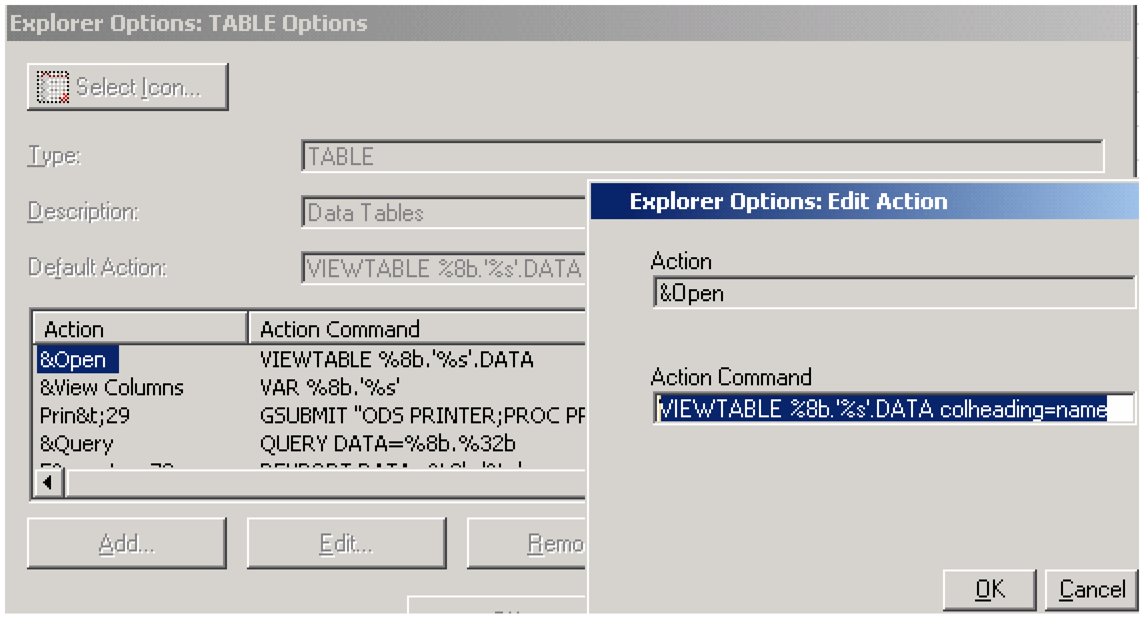


Registered Types:

Type	Description
CATALOG	Catalog of Entries
FDB	Financial Database
MDDB	Multi-dimensional Database
PROGRAM	Compiled SAS Data Step code
TABLE	Data Tables
VIEW	SQL/Data Step/DBI Views

Click on Edit and again Edit after selecting “&Open”. The following screens appear as shown in figure 24. In the final screen, add COLHEADING=NAME to make the name of the variable as the DEFAULT action when a VIEWTABLE window is opened.

FIGURE 24


Explorer Options: TABLE Options
Select Icon...
Explorer Options: Edit Action
Add...
Edit...
Remo
OK
Cancel

Type: TABLE

Description: Data Tables

Default Action: VIEWTABLE %8b.'%s'.DATA

Action	Action Command
&Open	VIEWTABLE %8b.'%s'.DATA
&View Columns	VAR %8b.'%s'
Prin&t;29	GSUBMIT "ODS PRINTER;PROC PF
&Query	QUERY DATA=%8b.%32b

Action: &Open

Action Command: VIEWTABLE %8b.'%s'.DATA colheading=name

CONCLUSION

This is not intended to be an all exhaustive discussion of the capabilities of the VIEWTABLE window. Rather, its purpose is to arouse your curiosity to seriously consider the VIEWTABLE as a legitimate tool. Logical errors in code can be detected by using the window. It is worth investigating whether the full suite of FSP products need to be licensed. It may well be the case that your organization has a legitimate need. However, by examining the capabilities of the humble VIEWTABLE window, this decision could be more easily justified.

REFERENCES

SAS Institute Inc. (2005), *SAS ONLineDoc®*, Version 9.1.3, Cary, NC, USA: SAS Institute Inc.

Curley, Lynn (2005). *Customize the data set browser for the SAS Explorer window*. Cary, NC, USA: SAS Institute Inc.

SAS Institute Inc. (1999), *SAS System Help*, Version 8.2, Cary, NC, USA: SAS Institute Inc.

RECOMMENDED READING

Fain, T. and Gareleck, C. (2002). *Mouse Clicking Your Way Viewing and Manipulating Data with Version 8 of the SAS® System*. Proceedings of the Twenty-Seventh Annual SAS Users Group International Conference, Orlando, FL. http://www2.sas.com/proceedings/sugi27/p054-27.pdf.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Name	Venky Chakravarthy
Enterprise	BioPharma Data Services
E-mail:	venky.chakravarthy@biopharmadataservices.com
Web:	http://www.biopharmadataservices.com