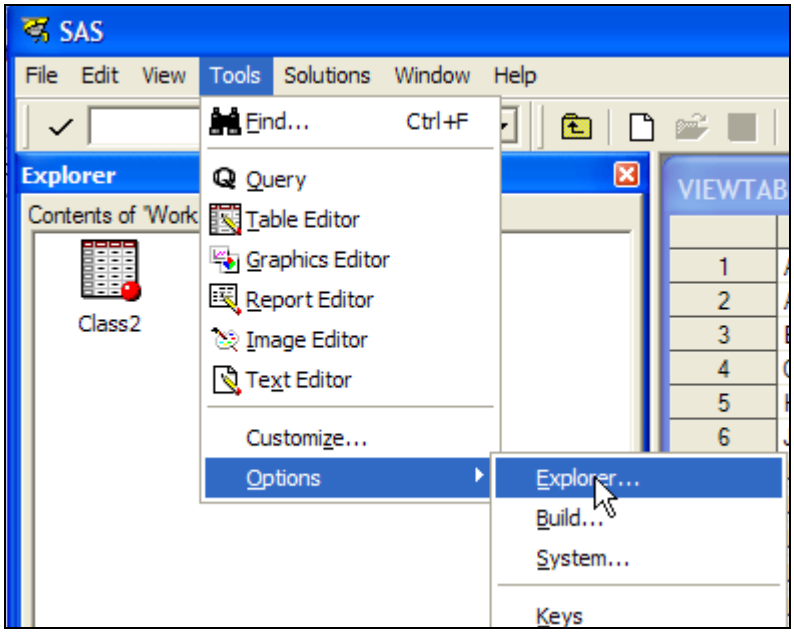
Fall 2008 OASUS Q&A

The following answers are provided to the benefit of the OASUS Users Group and are not meant to replace SAS Technical Support. Also, an Enterprise Guide project is provided as a companion to this document. The project is available on the OASUS web site (www.oasus.ca) under the Spring 2009 meeting since this is when these answers have been presented formally.

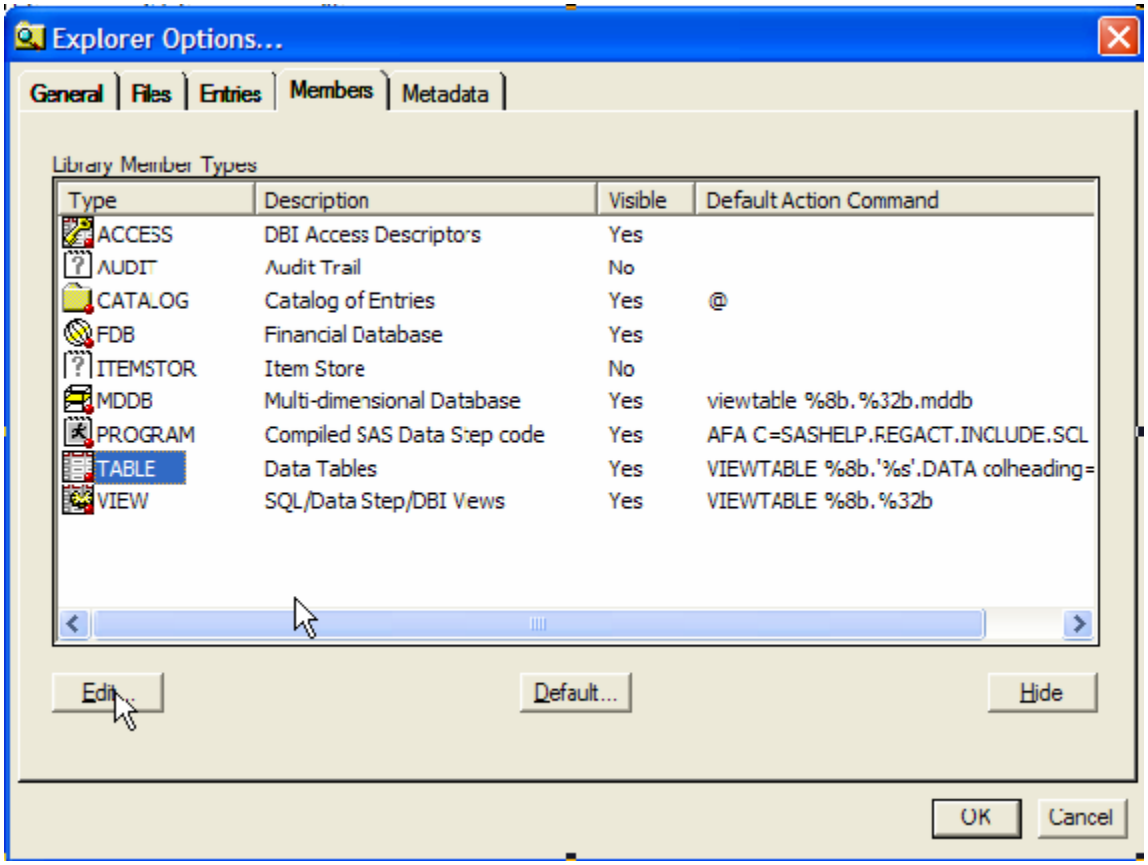
Question 1 : I have used SAS properties to view datasets with column names instead of column labels. Why is it that sometimes it is still the column labels that appear when I open a dataset?

In Display Manager:

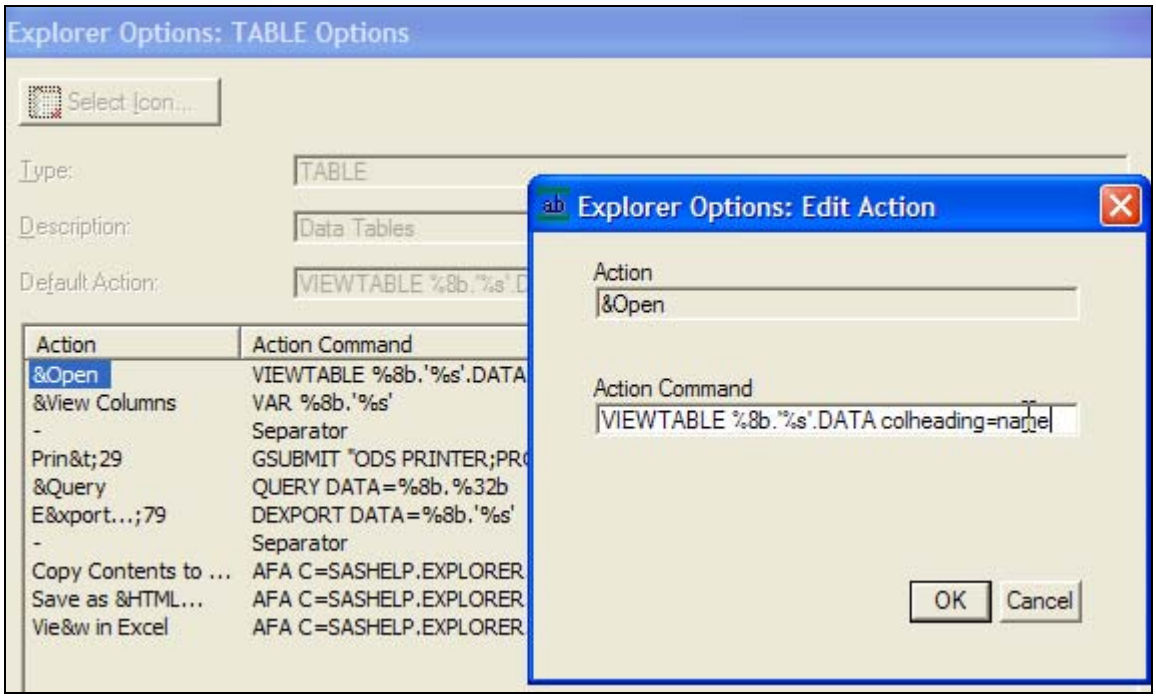
1) Navigate to Explorer Options



2) Edit Table member type under Members tab.



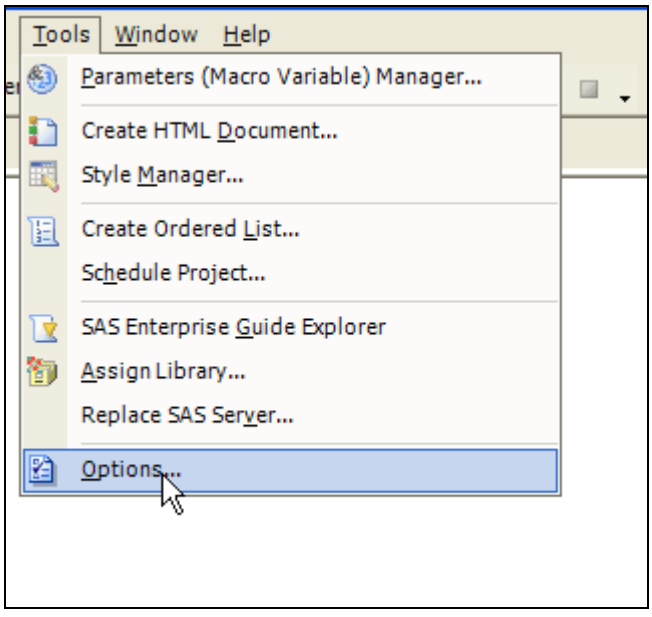
3) Edit &open action and add “colheading=name”



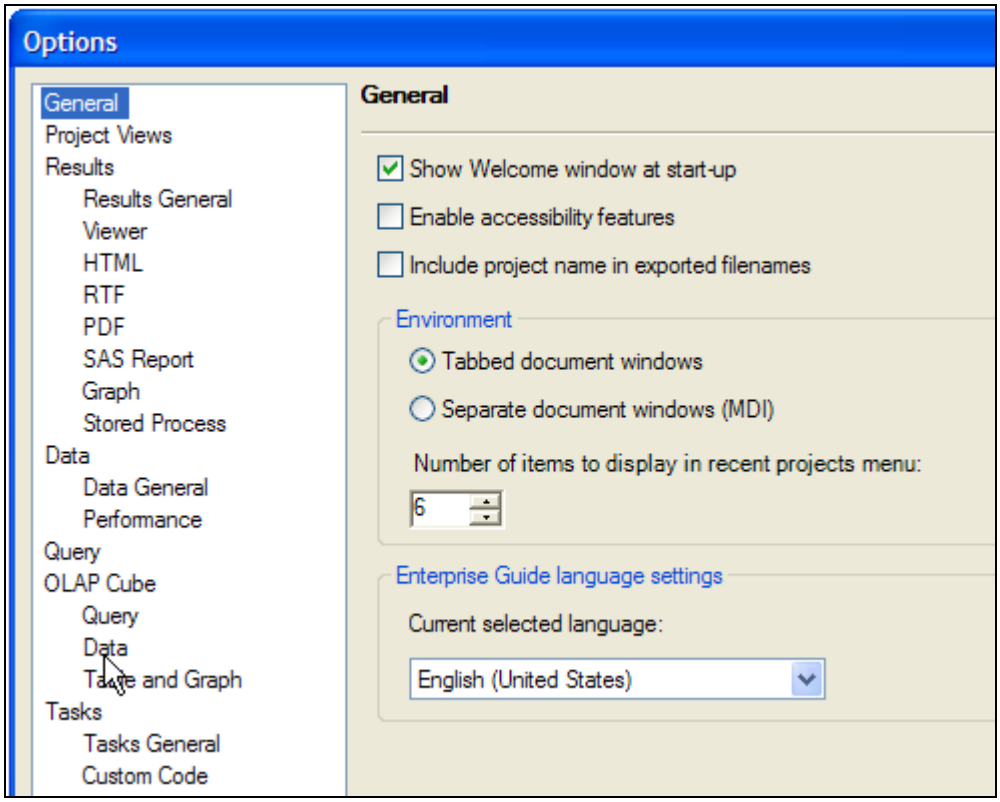
4) You should now be able to browse and/or edit a dataset with the column names!

In Enterprise Guide:

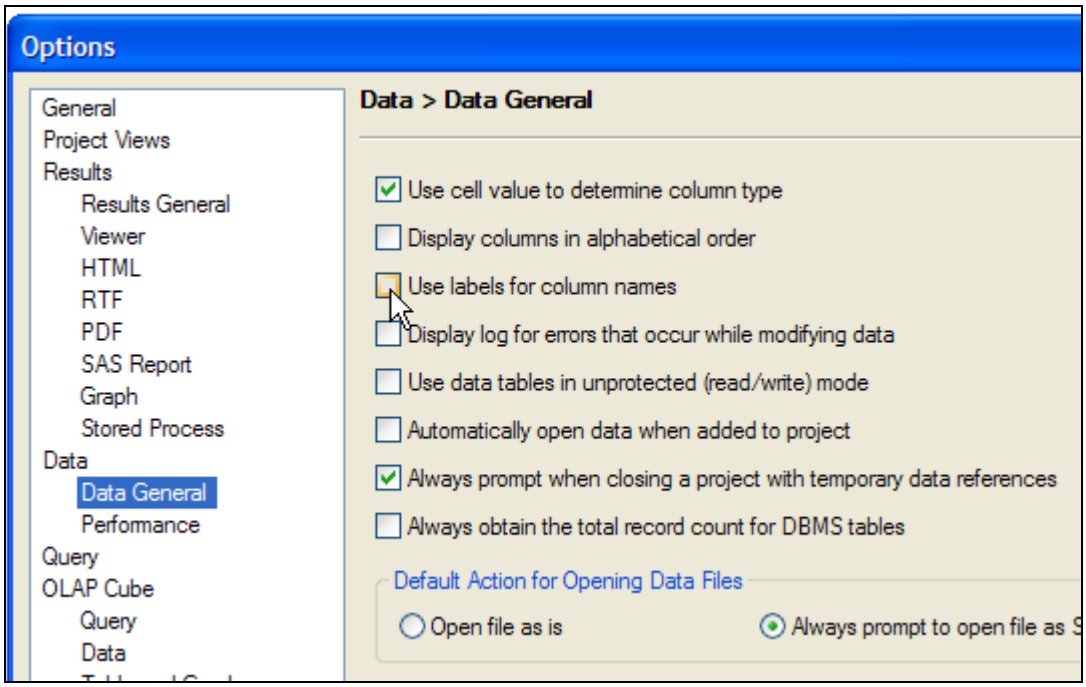
1) Navigate to Tools Options



2) Specifically, go to the Data Options



3) Turn off option ‘Use labels for column names”



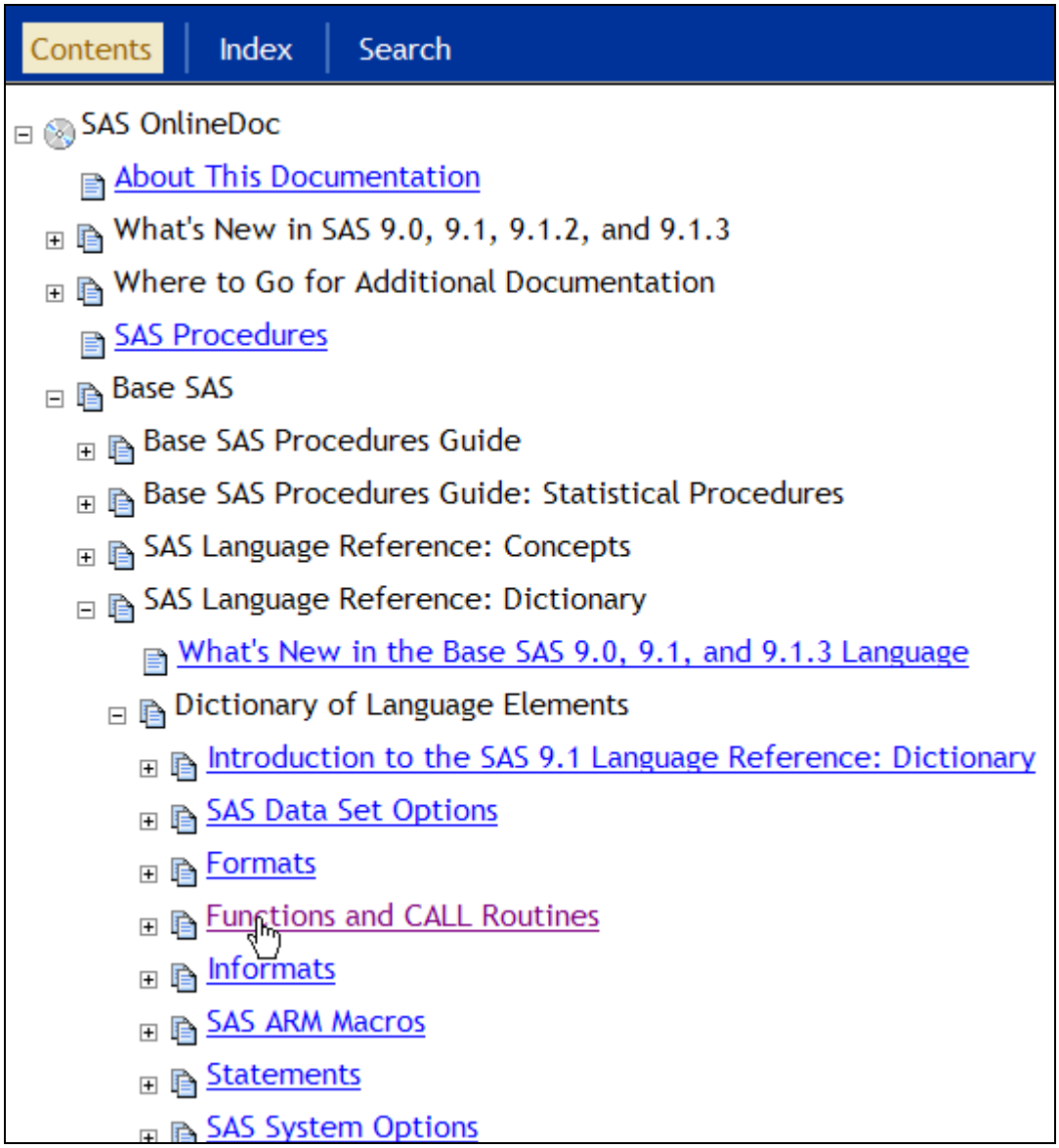
4) You can now browse and/or edit datasets with the column names. It is that simple!

Question 2 : Is there a dictionary/directory of all SAS functions? If so, where?

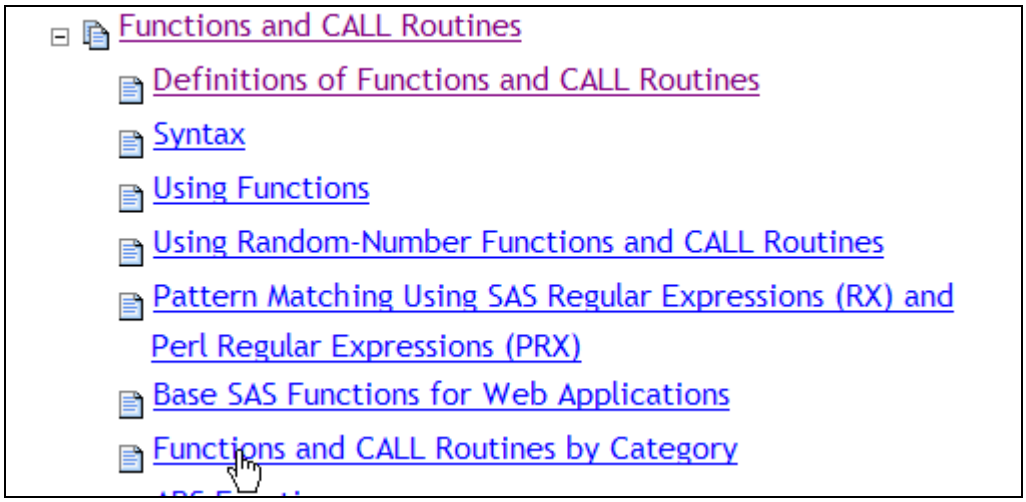
According to the SAS Online Documentation, a SAS function is defined as followed:

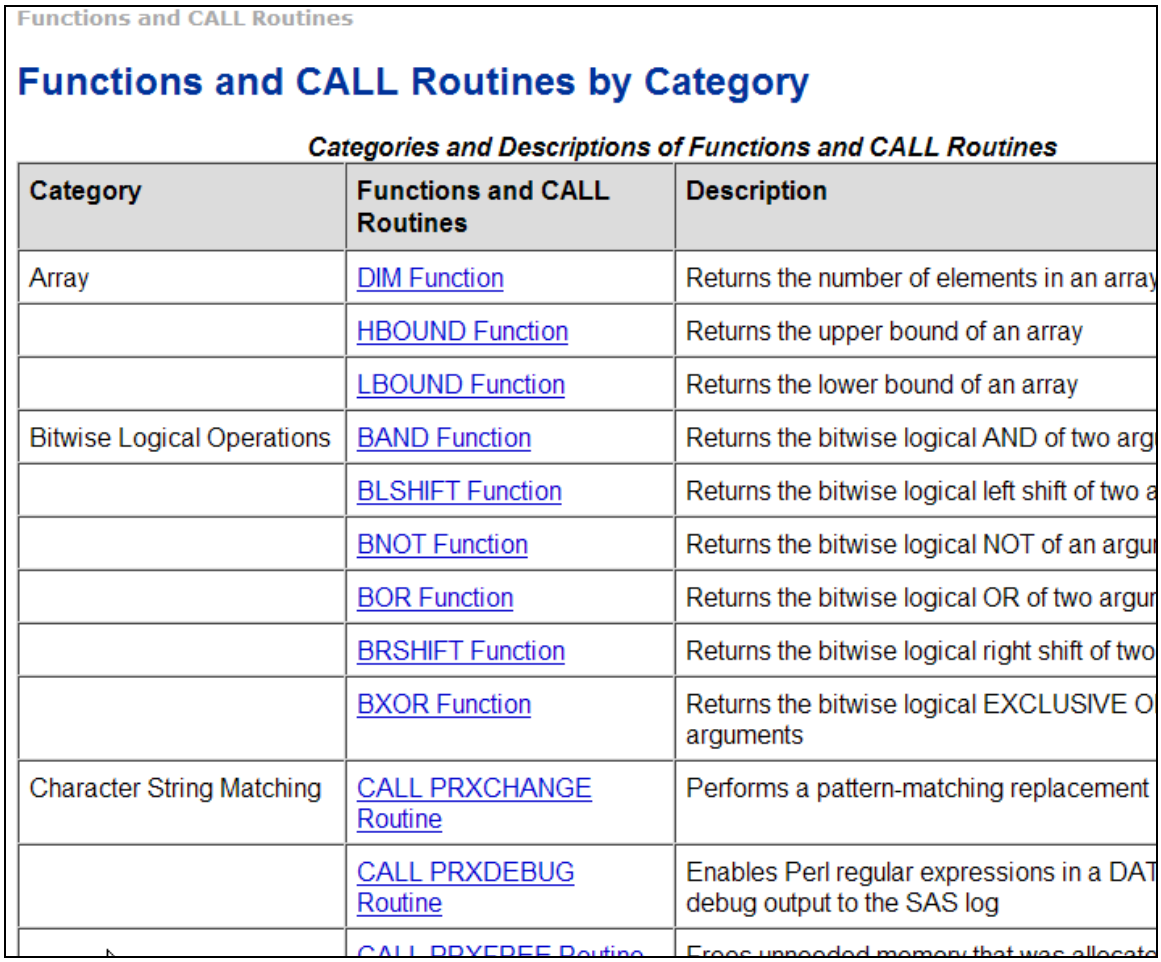
> "A SAS function performs a computation or system manipulation on arguments and returns a value. Most functions use arguments supplied by the user, but a few obtain their arguments from the operating environment."

You can see a complete list of the functions available at the SAS Functions and Call Routines Dictionary online documentation page.



The list of functions is very long. If you don't know the name of the function that you are looking, it is recommended to start with "Functions and CALL Routines by category".





	RXMATCH Function	Finds the beginning of a substring that matches a pattern
	RXPARSE Function	Parses a pattern
Character	ANYALNUM Function	Searches a character string for an alphanumeric character and returns the first position at which it is found
	ANYALPHA Function	Searches a character string for an alphabetic character and returns the first position at which it is found
	ANYCNTRL Function	Searches a character string for a control character and returns the first position at which it is found
	ANYDIGIT Function	Searches a character string for a digit and returns the first position at which it is found
	ANYFIRST Function	Searches a character string for a character that is valid as the first character in a SAS variable name under

Functions and CALL Routines

ANYALPHA Function

Searches a character string for an alphabetic character and returns the first position at which it is found

Category: Character

Syntax
Arguments
Details
Comparisons
Examples
See Also

Syntax

ANYALPHA(*string* <,*start*>)

Arguments

string

is the character constant, variable, or expression to search.

start

is an optional integer that specifies the position at which the search should start and the direction in which to search.

Finally, an excellent book on SAS functions is the one from Ron Cody, simply titled "SAS Functions by Example". More than 180 of the most useful SAS language functions are clearly defined and illustrated with fully annotated working programs.

Question 3 : If I am using one dataset and use PROC MEANS to calculate the mean of a variable. I need this value to be passed to different procedures in the program.

Many “report oriented” procedures such as PROC REPORT, PROC TABULATE and PROC MEANS are capable of producing data files as output (rather than a report). This is a useful feature as you can “chain” PROC steps and DATA steps to create highly sophisticated results. It is particularly true for PROC MEANS. If you use the OUTPUT statement and provide the name of an output dataset, then PROC MEANS will write the requested statistics directly in the output file. You can then post-process those statistics to meet some specific requirements.

```
/* Run PROC MEANS and output to data set */
proc means data=Sashelp.Class
      noprint  nway;
      var Age;
      class Sex ;
      output out=meanout  mean()= / autoname    ;
run;

/* Do something with the output of PROC MEANS*/
data meanout22(Keep=sex Age_Mean);
      set meanout;
run;
```

Question 4 : We are using Web Report Studio (WRS) and in the dataset we have a numerical variable that we wanted to change to a string, but the SAS programmer said that this was not possible with the SAS version 9.1 but would be able to with SAS 9.2. Is there a way around this using the SAS 9.1 version?

That is correct; you can not change the character type within WRS. WRS is meant to do query and reporting on the web, so it can not change the actual value types. It is suggested that you speak to your data developers. They can use Data Integration Studio (DIS) to recode this value or just recode this data in Base SAS and then bring it back into an Information Map.

Data Integration Studio:

You can create a calculated field in DIS. When doing the calculation, you can use SAS functions such as PUT and INPUT to change the type of a variable. Then you can map the calculations to your target data.

Some of the Data transformations even allow you to use the Expression Builder Window to create your calculations. You have access to a long list of SAS functions inside the Expression Builder window.

Base SAS example:

```
data temp;
   length Char4 $ 4;
   input numeric char4;

   /* convert character to numeric */
   New_Num=input(Char4,best4.);

   /* convert numeric to character */
   New_Char=put(Numeric,4.0);

cards;
789 1234
009 0009
1 9999
;;

proc print;
run;
```

*Question 5 : How to migrate all the data & objects in SQL*Server 2000 into SAS?*

This can be accomplished by using the SAS ACCESS engine for SQL*Server. You can find some examples that might help at the following location: SAS products>SAS Access>Relational databases>DBMS specific reference>SAS Access for Microsoft SQL Server. Note that many SQL*Server objects (for example, stored procedures) can not be migrated because there is no direct equivalent in SAS.

Question 6 : How to connect a Unix server (with SAS/SHARE) from EG to browse data or run SAS programs?

Before answering the question, lets review what is SAS/SHARE and under what circumstances it should be used.

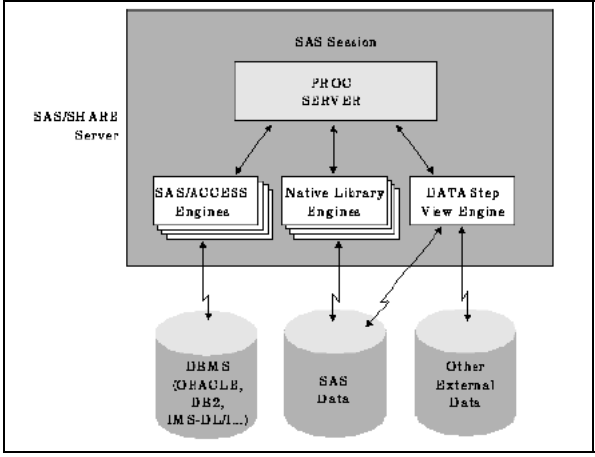
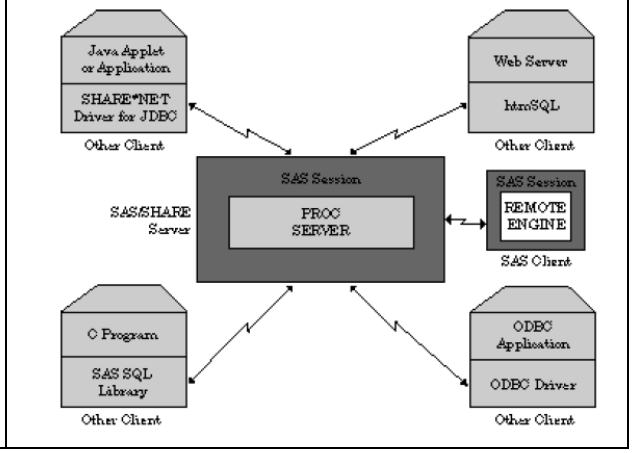
SAS/SHARE is a multi-user data server that allows concurrent update to one or many SAS data files and views. It is also a data hub that can be used to access files on a server without having to use a separate SAS/CONNECT session for every user.

The use of SAS/SHARE involves 3 distinct roles:

- *Server administrator*: the administrator starts the SHARE server (with PROC SERVER) and manages the SHARE server (with PROC OPERATE).
- *Applications developer*: the developer use the LIBNAME statement and, if necessary, various locking mechanisms to enable the applications read/update/write data through the SHARE server.
- *End users*: end users use an application that let him/her read, add, or update data in one or multiple files.

Although there are 3 roles involved in the use of SAS/SHARE, it is not required that 3 distinct persons be involved. Often the end user and the application developer are the same person, whereas the SAS/SHARE administrator is often the operating server administrator (UNIX or Windows for example).

SAS/SHARE provides a path to remote data and the engine used by a client application to access data on SHARE server is called the REMOTE engine and is triggered by using the SERVER option in the LIBNAME statement.

Data sources for a SAS/SHARE Server	**SAS/SHARE Server Clients**
	

Security is obviously important for a data server. A SAS/SHARE server can be made very secure by controlling access as followed:

- *With an administrator password*: to control access to administration commands issued via PROC OPERATE.
- *With a user password*: to control all connections from users. This password must be supplied whenever a client connects to the server.
- *Limiting the libraries a server can access*: the NOALLOC option in the PROC SERVER statement limits users to access only the libraries that are predefined.
- *SAS datasets password*: when creating a SAS dataset, it is possible to assign passwords to control the various operations (read, write and alter).
- *File system protections*: a SAS/SHARE server runs and accesses data under the same account on behalf of the clients who send data access requests. The server accesses that data through a file system which in turn validate the server's authority to read and write that data. It is therefore a good practice to limit the server's access to only the data that is required through file system permissions.
- *Users authentication*: the PROC SERVER option AUTHENTICATE=REQUIRED forces the users to provide a valid userid and password for the operating environment under which the server is running.

Here are some code examples for starting, managing and using a SAS/SHARE server:

To start a SAS/SHARE server

```
/* Start server sasshr and pre-allocate a library */
libname demo (work);
proc server id=sasshr authenticate=opt noalloc;
run;
```

To Manage a server

```
/* Allocate a library on the server */
proc operate id=&ServerName;
  allocate library test '/temp/test';
run;
/* Display allocated libraries */
proc operate id=&ServerName;
  display library _ALL_;
run;
/* Free a library previously allocated on a server */
proc operate id=&ServerName;
  free library test;
run;
```

To access a remote library previously allocated on the server.

```
/* Assign a libref to a library manage by a server */
Libname demo server=&ServerName;
```

A SAS/SHARE server is started using PROC SERVER. Depending on the communication access method being used, some set-up prior to invoking PROC SERVER might be necessary. In most cases, TCP/IP is being used and requires that a TCP port be defined for SAS/SHARE in the services file for both the server and each of

the clients. It is possible to make do without this definition, but that requires specifying the port number (using the syntax __<port number>.

With this background on SAS/SHARE, we should be able to answer the specific question. Note that SAS/SHARE is not meant to run SAS programs but to access data on a remote server. To run SAS program remotely on a UNIX host from Enterprise Guide, you should use either SAS/CONNECT or IOM to execute SAS programs on a remote host. Also, there is nothing special about UNIX. Once the SAS/SHARE server is up and running, the flavour of operating system does not matter.

Accessing data on a SAS/SHARE server from Enterprise Guide can be done in one of two ways:

- With an explicit LIBNAME statement run from within an Enterprise Guide project. This will assign a libref on the workspace server that uses the remote engine to access data on the SAS/SHARE server. This is more or less equivalent to using the Display Manager.
- A SAS/SHARE server can also be part of a SAS Intelligence Platform. In this case, the SAS/SHARE server is defined in a Metadata repository along with any data sources that need to be shared. In this scenario, EG users access the shared libraries like any other library defined in a Metadata repository – directly from the server list window.

Question 7 : What are the suggestions for tuning SAS/SHARE to improve the performance?

The first thing to be mentioned is that a SAS/SHARE server always introduces some overhead. The data simply flows faster when an application accesses data directly. So, before using a server, one should understand that performance will never be optimum and that there must good reasons for using a server. In the case of SAS/SHARE, either concurrent update is required or file access must be undertaken by SAS on the remote host (for examples: the unavailability of a network file system, the requirement to log data access for audit purposes, etc.).

Once it is decided that SAS/SHARE must be used, there are 3 computer resources that must be considered when trying to optimize the performance: the CPU, the I/O and the memory. Both the client and the server run as a SAS session and consume those 3 resources. Optimizing any of those should improve performance.

It is also critical to analyse the type of access that needs to be supported and the size/type of files that will be accessed. Reading sequentially a large SAS file is quite different than updating frequently a small SAS file.

The following are some suggestions to help improve performance (by maximizing I/O, CPU and memory):

- *Clean-up the data*: remove unused variables and observations. This will reduce the size of the data files and the number of I/O operations.
- *Subset the data*: If you subset the data with a WHERE clause, the good news is that the subsetting takes place on the server and will reduce network traffic. The bad news is that it can consume large amount of I/O and memory on the server. Using indexes can help in that regard.
- *PROC SERVER options:*
    - *TBUFSIZE option*: specifies the suggested size of a buffer that the server uses to transmit data to or for receiving data from the client.
    - *TOOBSNO option*: controls the number of observations of individual dataset that accessed through the server for transmission.
    - *LRPYIELD option*: sets the rate at which server processes yield control so that others can use the server.
    - *WORKTASKS option*: specifies the initial and maximum work tasks for the SHARE server to execute. More work tasks enable the server to service more asynchronous requests.
- *SAS System options*:
    - *BUFSIZE option*: when a SAS file is created this option can be use to set the page size. SAS reads one page at a time so fine tuning this option can bring interesting benefits. Depending how the file is intended to be used

(sequentially, random access, etc.) the default page size, which is suitable for sequential access, could be changed..

- *COMPRESS option*: this option causes the SAS file to be stored in a compressed format. A compressed data file can be much smaller that the uncompressed equivalent thus reducing substantially the number of I/O operations. However, more CPU will be consumed to uncompress the file.

- *Multiple SAS/SHARE Servers*: if a SHARE server is simply overloaded, there is nothing wrong about adding more servers to balance the workload. SAS provides a number of macros to help manage SAS files access through multiple servers.

Question 8 : How to use SAS to do data warehousing?

With the advent of SAS 9, Data Warehousing is done in SAS using the SAS Data Integration Server (for the creation of OLAP cubes and other integrated data) and Enterprise BI server (for the consumption of the cubes and other integrated data). For more information, please visit www.sas.com and look under the SAS Intelligence Platform.

Question 9 : I was told that SAS/AF is not going to be available in future years as part of SAS. What is SAS planning to replace this with?

SAS/AF is still fully supported by SAS. Here is a recent statement from SAS regarding the future of SAS/AF:

> "Is there a future for SAS/AF software? Absolutely! We have heard from many customers that SAS/AF is particularly useful for the development of interactive applications that can be executed with as little as an installation of Base SAS. In response to this feedback, SAS wants to demonstrate its support for SAS/AF, ensuring that SAS/AF will continue to be a useful product for many years to come. Accordingly, SAS/AF and the VIEWTABLE window are being enhanced to include support of long format names. Two classes with experimental status, the Dual Selector Control and the Tree View Control, are being enhanced and promoted to production status."

Question 10 : Is it possible to have a SAS JMP presentation?(OASUS)

Yes! We are hoping to have a SAS JMP presentation in the near future.

Question 11 : Do we have a blog for Q&A for OASUS? (OASUS)

We do not have a blog for the Q&A for OASUS at the moment.