Version 9 Changes and Enhancements

Global Business Intelligence Consulting and Training

Destiny Corporation • 100 Great Meadow Rd Suite 601
Wethersfield, CT 06109-2379
Phone: (860) 721-1684 • 1-800-7TRAINING
Fax: (860) 721-9784
Email: info@destinycorp.com Web Site: www.destinycorp.com

Prepared by Dana Rafiee

Introduction

This document is an excerpt from Destiny Corporation's Version 9 Changes and Enhancement course materials. It is designed to give a brief overview of what Version 9 has to offer.

Version 9 of the SAS System will roll out in several releases. The release date of Version 9.0 is mid 2002 and Version 9.1 will be the beginning of 2003. Some of the enhancements and features discussed in this course will be first available in either 9.0 or 9.1.

The primary goal of Version 9 is to provide support for a new level of computing that supports faster execution of applications, centralized access of data and support of the latest computing technology. These materials are designed as an overview of what is available and new and complement the online documentation that ships with the software.

The new SAS Open Metadata Architecture is designed to allow all registered data in an organization to be centrally managed and accessed. This feature will ship with Version 9.1.

The SAS Management Console will offer one point of control for all SAS servers and applications in the organization.

The new multi-threaded architecture is automatically turned on to support most procedures that sort and summarize data.

The Output Delivery System has been enhanced to support many new styles of markup, along with custom markup tag sets.

SAS also supports the industry standard Application Response Measurement (ARM) protocol for monitoring SAS processes.

SAS V9 is designed to be upwardly compatible with code and data.

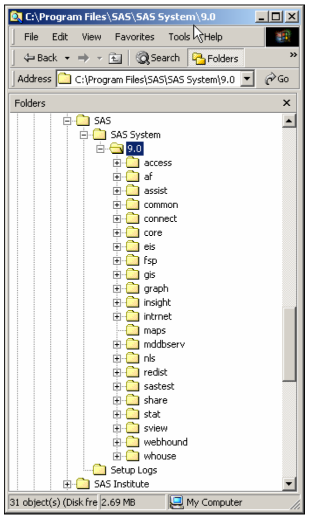
Cross Environment Data Access (CEDA) is still supported on all operating systems.

SAS Version 9 is designed to better support ADA 508 handicapped standards.

This course is designed to discuss as many issues as possible, from the installation of SAS to the finer aspects of operating under the new release.

Installation of SAS

SAS is now installed in a new directory location. The following example shows the tree structure.



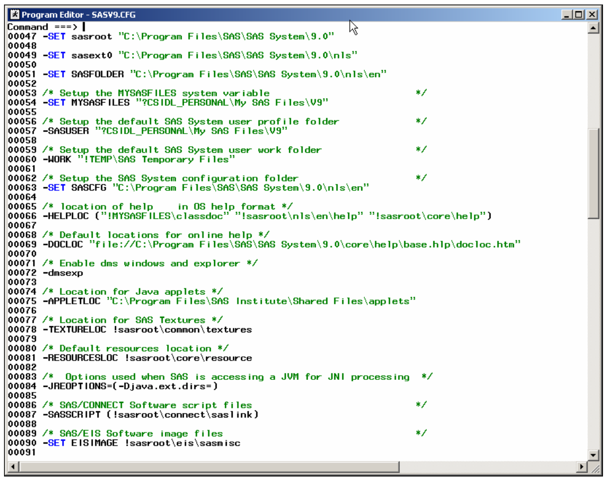
The new directory for Version 9 is SAS. SAS V9 is designed to live alongside SAS V8. Notice the SAS V9 directory and the SAS Institute V8 directory.

Note: SAS V9 and SAS V8 use some of the same shared components like the Enhanced Editor and ActiveX controls. When uninstalling one of the versions, do not uninstall the shared components.

When installing SAS V9, the footprint on disk may be large. You may consider deleting the SAS Video files and Maps that are not needed at your installation.

Configuration File Changes

The configuration file uses pointers to different locations than found in previous versions. It is important to be aware of these changes as it affects local and network installations of the software.



```
Program Editor - SASV9.CFG
Command ===>
-SET sasroot "C:\Program Files\SAS\SAS System\9.0"

-SET sasext0 "C:\Program Files\SAS\SAS System\9.0\nls"

-SET SASFOLDER "C:\Program Files\SAS\SAS System\9.0\nls\en"

/* Setup the MYSASFILES system variable                            */
-SET MYSASFILES "?CSIDL_PERSONAL\My SAS Files\V9"

/* Setup the default SAS System user profile folder                */
-SASUSER "?CSIDL_PERSONAL\My SAS Files\V9"

/* Setup the default SAS System user work folder                   */
-WORK "!TEMP\SAS Temporary Files"

/* Setup the SAS System configuration folder                       */
-SET SASCFG "C:\Program Files\SAS\SAS System\9.0\nls\en"

/* location of help    in OS help format */
-HELPLOC ("!MYSASFILES\classdoc" "!sasroot\nls\en\help" "!sasroot\core\help")

/* Default locations for online help */
-DOCLOC "file://C:\Program Files\SAS\SAS System\9.0\core\help\base.hlp\docloc.htm"

/* Enable dms windows and explorer */
-dmsexp

/* Location for Java applets */
-APPLETLOC "C:\Program Files\SAS Institute\Shared Files\applets"

/* Location for SAS Textures */
-TEXTURELOC !sasroot\common\textures

/* Default resources location */
-RESOURCESLOC !sasroot\core\resource

/*  Options used when SAS is accessing a JVM for JNI processing  */
-JREOPTIONS=(-Djava.ext.dirs=)

/* SAS/CONNECT Software script files                               */
-SASSCRIPT (!sasroot\connect\saslink)

/* SAS/EIS Software image files                                    */
-SET EISIMAGE !sasroot\eis\sasmisc

```

Notice the new locations for sasroot, sasext0 and SASFOLDER, along with MYSASFILES. In addition, there are pointers to Java and graphical directories.

SETINIT Changes

The installation and update of SAS is now easier than ever. SAS Institute will now create a mass production of CDs with all modules. The specific site information of products licensed will now be available off of a file supplied. This file may be shipped with the software or will be available off of the SAS web site, based on each company's site number. See the SAS installation instructions for further detail.

When installing SAS, there is now a selection for selecting all products licensed by the click of a button.

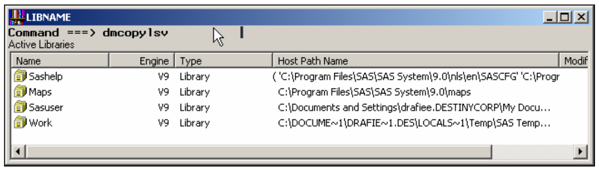
See the SAS installation instructions for space and configuration requirements for your operating system and environment.

Display Manager Changes

In Version 9, there have been several, cosmetic changes to the DMS interface. These changes are designed to make accessing information in this environment easier and more robust.

List View Copy Command – DMCOPYLSV

The new command DMCOPYLSV allows for copying of listing information to the clipboard. The command is used in the following libname window to demonstrate its capabilities.



LIBNAME

Command ===> dmcopylsv

Active Libraries

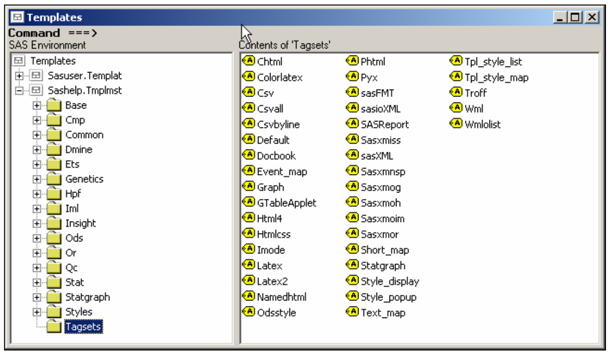
Name	Engine	Type	Host Path Name	Modif
Sashelp	V9	Library	('C:\Program Files\SAS\SAS System\9.0\nls\en\SASCFG' 'C:\Progr	
Maps	V9	Library	C:\Program Files\SAS\SAS System\9.0\maps	
Sasuser	V9	Library	C:\Documents and Settings\drafiee.DESTINYCORP\My Docu...	
Work	V9	Library	C:\DOCUME~1\DRAFIE~1.DES\LOCALS~1\Temp\SAS Temp...	

The result is the following information. This has many uses, including documentation configuration documentation.

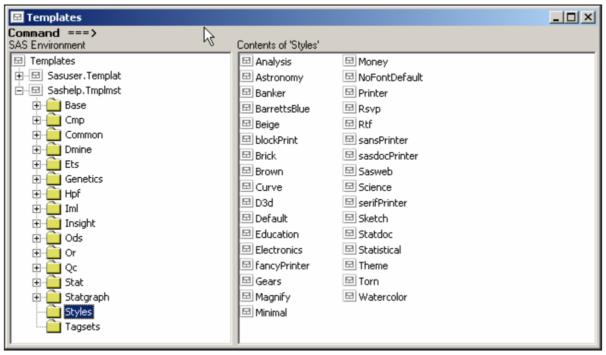
"Name", "Engine", "Type", "Host Path Name", "Modified",
"Sashelp", "V9", "Library", "('C:\Program Files\SAS\SAS System\9.0\nls\en\SASCFG' 'C:\Program Files\SAS\SAS System\9.0\core\sashelp' 'C:\Program Files\SAS\SAS System\9.0\af\sashelp' 'C:\Program Files\SAS\SAS System\9.0\assist\sashelp' 'C:\Program Files\SAS\SAS System\9.0\connect\sashelp' 'C:\Program Files\SAS\SAS System\9.0\eis\sashelp' 'C:\Program Files\SAS\SAS System\9.0\gis\sashelp' 'C:\Program Files\SAS\SAS System\9.0\graph\sashelp' 'C:\Program Files\SAS\SAS System\9.0\insight\sashelp' 'C:\Program Files\SAS\SAS System\9.0\intrnet\sashelp' 'C:\Program Files\SAS\SAS System\9.0\mddbserv\sashelp' 'C:\Program Files\SAS\SAS System\9.0\sview\sashelp' 'C:\Program Files\SAS\SAS System\9.0\stat\sashelp' 'C:\Program Files\SAS\SAS System\9.0\whouse\sashelp' 'C:\Program Files\SAS\SAS System\9.0\webhound\sashelp')", "",
"Maps", "V9", "Library", "C:\Program Files\SAS\SAS System\9.0\maps", "",
"Sasuser", "V9", "Library", "C:\Documents and Settings\drafiee.DESTINYCORP\My Documents\My SAS Files\V9", "",
"Work", "V9", "Library", "C:\DOCUME~1\DRAFIE~1.DES\LOCALS~1\Temp\SAS Temporary Files_TD1488", "",

New Tagsets Available

Select Tagsets from the Templates selection in the Results window to see all of the tagsets written and included with SAS software.

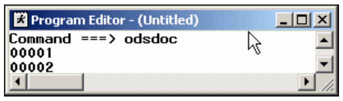


Select Styles to see all of the styles available.

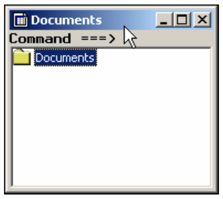


ODS Document Viewer

The new ODS document viewer is now available. It can be used to see stored documents created with the ODS DOCUMENT statement. See the ODS section of this document for further details.

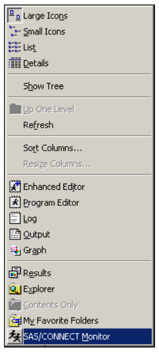


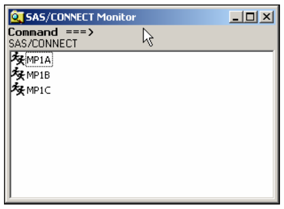
```
odsdoc
```



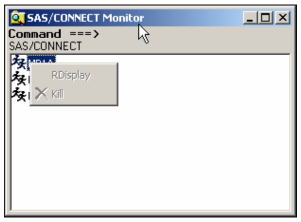
SAS/Connect Monitor

This option is available from the Tools pull down menu. It allows for monitoring of running SAS/Connect sessions.





Right clicking on each session allows for killing the session or viewing output.



Return of the V6 Windows

SAS brought back some of the old Version 6 windows in Display Manager per the request of users.

The commands to access them are:

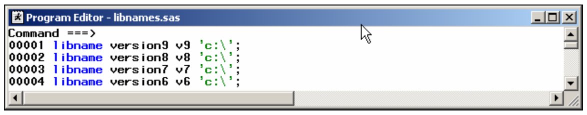
V6CAT
V6LIB
V6DIR
V6VAR

Data Migration

In Version 9, SAS supports formats and informats that are longer than 8 bytes. The only difference in the data set structures is this capability. Version 8 data sets are upward compatible to Version 9 by default. Version 9 data sets are backward compatible if formats and informats conform to Version 8 naming conventions.

Engines

Libname statements offer support for the new V9 engine as well as the V8, V7 and V6 engines.



```
libname version9 v9 'c:\';
libname version8 v8 'c:\';
libname version7 v7 'c:\';
libname version6 v6 'c:\';
```

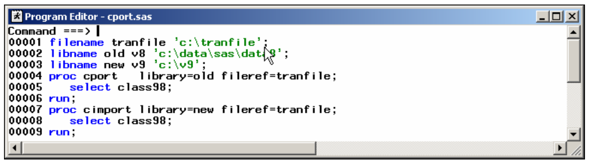
The file extensions are still the same as in Version 8.

File Extension	SAS Member Type	Description
.log	None	Log
.lst	None	Output
.sas	None	SAS Program
.sas7bacs	Access	Access descriptor
.sas7baud	Audit	Audit file
.sas7bcat	Catalog	SAS catalog
.sas7bdat	Table	Data set
.sas7bmdb	MDDB	Multi-dimensional database

.sas7bndx	None	Data set index. Indexes are stored as separate files but are treated by the SAS System as integral parts of the SAS data file.
.sas7bods		Output Delivery System file
.sas7bpgm	Program	Stored DATA step program
.sas7bvew	View	Data set view

Catalog Migration

Note: Version 6 catalogs or Version 8 catalogs on 32 bit systems that are being upgraded to 64 bit systems must be updated from previous versions of SAS to Version 9 with the standing CPORT/CIMPORT procedures.



```
Command ===>
filename tranfile 'c:\tranfile';
libname old v8 'c:\data\sas\data8';
libname new v9 'c:\v9';
proc cport   library=old fileref=tranfile;
   select class98;
run;
proc cimport library=new fileref=tranfile;
   select class98;
run;
```

New Formats

The new length for numeric format names is 32 and for character names is 31. Format names can be associated with a data set. The only difference between SAS V9 and previous versions of data sets is the existence of a format name longer than the traditional 8 bytes.

Version 9 data sets that use 8 byte or shorter format names can be read by Version 8.

$BIDw.

This format is designed to convert a string to be logically or visually ordered. This works with Hebrew and Latin characters.

New Informats

The new length for numeric informat names is 31 and for character names is 30. Informat names can be associated with a data set. The only difference between SAS V9 and previous versions of data sets is the existence of a format name longer than the traditional 8 bytes.

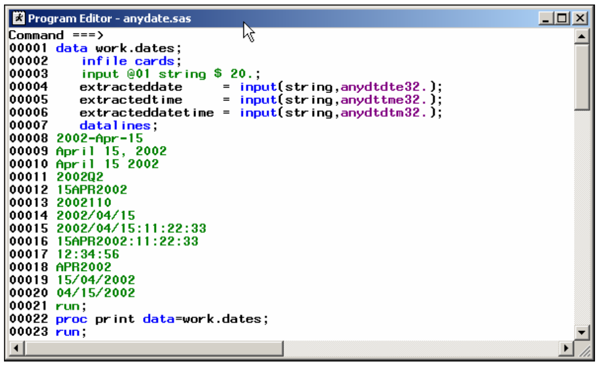
Version 9 data sets that use 8 byte or shorter format names can be read by Version 8.

Any Date Informat

Three new informats are now available to convert various date, time and datetime forms of data into a SAS date or SAS time. They are:

ANYDTDTEw.	To convert to a SAS date value
ANYDTTMEw.	To convert to a SAS time value
ANYDTDTMw.	To convert to a SAS datetime value

These new informats were created to make reading these types of values simpler. It is important to realize that these informats make assumptions on a record by record basis. Ambiguous values can be interpreted in an incorrect fashion.



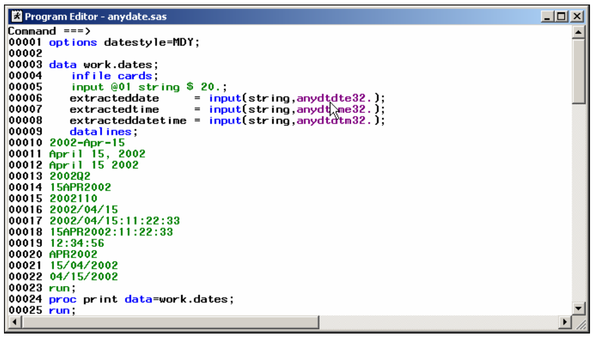
```
Command ===>
data work.dates;
   infile cards;
   input @01 string $ 20.;
   extracteddate     = input(string,anydtdte32.);
   extractedtime     = input(string,anydttme32.);
   extracteddatetime = input(string,anydtdtm32.);
   datalines;
2002-Apr-15
April 15, 2002
April 15 2002
2002Q2
15APR2002
2002110
2002/04/15
2002/04/15:11:22:33
15APR2002:11:22:33
12:34:56
APR2002
15/04/2002
04/15/2002
run;
proc print data=work.dates;
run;
```

Special Interpretations

Timezones with + or – GMT times will be ignored.
The YEARCUTOFF= option interprets two digit years

DATESTYLE Option

The DATESTYLE option can be used when ambiguous values in dates exist. This option sets a default assumption for the date to be either DMY, MDY, or YMD.



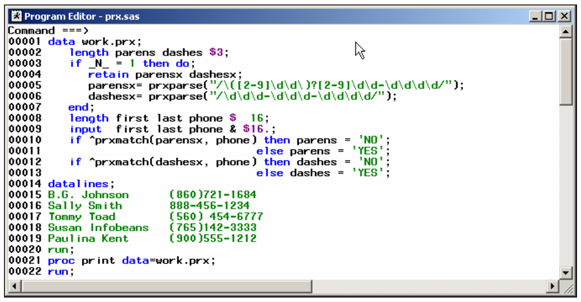
```
Command ===>
options datestyle=MDY;

data work.dates;
   infile cards;
   input @01 string $ 20.;
   extracteddate     = input(string,anydtdte32.);
   extractedtime     = input(string,anydttme32.);
   extracteddatetime = input(string,anydtdtm32.);
   datalines;
2002-Apr-15
April 15, 2002
April 15 2002
2002Q2
15APR2002
2002110
2002/04/15
2002/04/15:11:22:33
15APR2002:11:22:33
12:34:56
APR2002
15/04/2002
04/15/2002
run;
proc print data=work.dates;
run;
```

New Functions

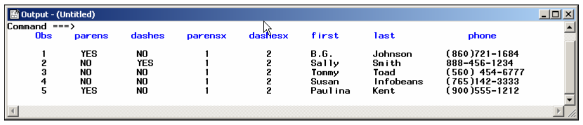
There are several, new functions and call routines available in SAS. Most of these are designed for very specific manipulations of data.

Perl Regular Expressions and Pattern Matching

There are several functions available to perform pattern matching routines on data. The following example shows a typical use for one of these functions. The PRXPARSE function is designed to specify pattern matching. This example sets YES/NO flags as to how phone numbers match a valid pattern.



```
Command ===>
data work.prx;
   length parens dashes $3;
   if _N_ = 1 then do;
      retain parensx dashesx;
      parensx= prxparse("/\([2-9]\d\d\)?[2-9]\d\d-\d\d\d\d/");
      dashesx= prxparse("/\d\d\d-\d\d\d-\d\d\d\d/");
   end;
   length first last phone $  16;
   input  first last phone & $16.;
   if ^prxmatch(parensx, phone) then parens = 'NO';
                                else parens = 'YES';
   if ^prxmatch(dashesx, phone) then dashes = 'NO';
                                else dashes = 'YES';
datalines;
B.G. Johnson      (860)721-1684
Sally Smith       888-456-1234
Tommy Toad        (560) 454-6777
Susan Infobeans   (765)142-3333
Paulina Kent      (900)555-1212
run;
proc print data=work.prx;
run;
```



```
Command ===>
  Obs    parens    dashes    parensx    dashesx    first      last         phone

   1      YES       NO          1          2       B.G.       Johnson      (860)721-1684
   2      NO        YES         1          2       Sally      Smith        888-456-1234
   3      NO        NO          1          2       Tommy      Toad         (560) 454-6777
   4      NO        NO          1          2       Susan      Infobeans    (765)142-3333
   5      YES       NO          1          2       Paulina    Kent         (900)555-1212
```

The valid list of values available for Perl expressions is available from WWW.PERLDOC.COM.

Function	Definition
CALL PRXCHANGE	Matches and substitutes values
CALL PRXDEBUG	Debugs PRXs for problem solving
CALL PRXFREE	Frees PRX memory
CALL PRXNEXT	Starting position of a substring pattern match
CALL PRXPOSN	Starting position and length for the buffer
CALL PRXSUBSTR	Starting position and length of a substring
PRXCOUNT	Finds the final bracket match for a pattern
PRXMATCH	Starting position of a pattern match
PRXPARSE	Compiles a PRX for character matching

Numeric Functions

Function	Definition
ARCOSH	Finds the inverse hyperbolic cosine
ARSINH	Finds the inverse hyperbolic sine
ARTANH	Finds the inverse hyperbolic tangent
BETA	Finds the beta value
CALL ALLPERM	Permutation creation of several variables
CALL COMPCOST	Prepares costs of operations
CALL RANPERK	Random permutation creation of several variables
CALL RANPERM	Random permutation creation of several values
CALL SORTQ	Sorts values of variables in the Data Step
CALL STDIZE	Standardizes variable values
CALL SYMPUTX	Left justifies, trims and converts numeric to character for macros
CALL VNEXT	Lists the PDV variable names, types and lengths
CEILZ	Uses zero fuzzing to find the smallest integer GE a value
COMPARE	Finds the first character where two strings do not match
FLOORZ	Uses zero fuzzing to find the largest integer LE a value
INTZ	Uses zero fuzzing to find the integer part of a value
IQR	Finds the inter-quartile range
KCVT	DBCS data to 2 byte code converter
LARGEST	Finds the nth largest non-missing value
LOGBETA	Finds the log of the beta value
MAD	Finds the median absolute deviation of the median
MEDIAN	Finds the median value
MODZ	Uses zero fuzzing to find the remainder
PCTL	Finds the percentile
ROUNDE	Rounds the first value to the closest multiple of the second value
ROUNDZ	Uses zero fuzzing to round the first value to the closest multiple of the second value
SMALLEST	Finds the nth smallest non-missing value
VVALUE	Extracts the formatted value of a variable
VVALUEX	Extracts the formatted value of a value

Character Functions

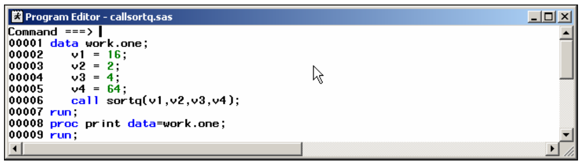
Function	Definition
ANYALNUM	Finds the first occurrence of an alphanumeric character
ANYALPHA	Finds the first occurrence of an alphabetic character
ANYCNTRL	Finds the first occurrence of a control character
ANYDIGIT	Finds the first occurrence of a digit
ANYFIRST	Finds the first occurrence of an _, upper, or lower case alphanumeric character of the name of a SAS variable
ANYGRAPH	Finds the first occurrence of a graphical character
ANYLOWER	Finds the first occurrence of a lower case character
ANYNAME	Finds the first occurrence of an _, upper, or lower case alphanumeric character
ANYPRINT	Finds the first occurrence of a printable character
ANYPUNCT	Finds the first occurrence of a punctuation character
ANYSPACE	Finds the first occurrence of a white space character

ANYUPPER	Finds the first occurrence of an upper case character
ANYXDIGIT	Finds the first occurrence of a digit in a hex character
CALL SORTQ	Sorts values of variables in the Data Step
CAT, CATS, CATT, CATX	Concatenate passed parameters without the need for using TRIM, LEFT and PUT functions.
COMPARE	Finds the first occurrence of where two strings differ
COMPGED	Uses generalized edit distance to compare two strings
COMPLEV	Uses Levenshtein edit distance to compare two strings
COUNT	Returns the number of occurrences of a character in a string
COUNTC	Returns the number of occurrences of a character in a string that exist or do not exist
FIND	Like INDEX function searches, but not case or direction sensitive
FINDC	Like INDEXC function searches, but not case or direction sensitive
LENGTHC	Finds the length of a character value including trailing blanks
LENGTHM	Finds the length used by a character value in memory
LENGTHN	Finds the length of a character value excluding trailing blanks
NINVALID	Determines if the string is a valid SAS name
NLITERAL	Converts a string to a literal if not a valid SAS name
NOTALNUM	Finds the first occurrence of a non alphabetic character
NOTCNTRL	Finds the first occurrence of a non control character
NOTDIGIT	Finds the first occurrence of a non digit
NOTFIRST	Finds the first position of a character value that could be used as the beginning of a SAS name
NOTGRAPH	Finds the first occurrence of a non graphical character
NOTLOWER	Finds the first occurrence of a non lower case character
NOTNAME	Finds the first occurrence of a non valid character for a SAS name
NOTPRINT	Finds the first occurrence of a non printable character
NOTPUNCT	Finds the first occurrence of a non punctuation character
NOTSPACE	Finds the first occurrence of a non white space character
NOTUPPER	Finds the first occurrence of a non upper case character
NOTXDIGIT	Finds the first occurrence of a non digit in a hex value
NVALID	Determines if a string can be used as a SAS name
SCANQ	Finds a word in a string while ignoring delimiters inside quotes
SUBSTRN	Allows for doing a substring where the length may be zero

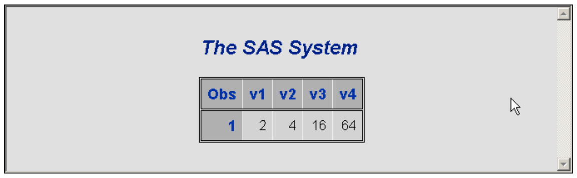
CALL SORTQ Example

CALL SORTQ is a quick way to sort variable values inside the Data Step. It is not designed to replace PROC SORT. It is a simple way of ordering values of the same structure. For example, if it is used with character variables, they must be the same length.

Consider the following example. Four variables are loaded into a Data Step. We want them ordered smallest to largest to process on.



```
Command ===>
data work.one;
   v1 = 16;
   v2 = 2;
   v3 = 4;
   v4 = 64;
   call sortq(v1,v2,v3,v4);
run;
proc print data=work.one;
run;
```



The SAS System

Obs	v1	v2	v3	v4
1	2	4	16	64

New System Options

There are several new system options available in Version 9.0.

Option	Definition
ARMAGENT=	Lists a vendor's executable ARM agent
ARMLOC=	Location of the ARM log
ARMSUBSYS=	Starts and stops the SAS ARM subsystems
BYSORTED	Lists the way information is sorted in the data set
CPUCOUNT	Limits the number of processors used in threading procedures
DATESTYLE	Used with ANYDATE informats to specify default Month Day Year
DBSLICEPARM=	Used with SAS/ACCESS when reading databases. It changes the way DBMS data is read by allowing SAS to read ahead as a table is loaded.
DMSSYNCHK	Allows for syntax checking in Display Manager
DOCINDEX	Index file location for SAS online documentation

DOCTOC	Table of contents files for SAS online documentation
DTRESET	Updates the date and time in the log and listing file
EMAILAUTHPROTOCOL	Sets up SMTP email authentication protocol
EMAILID=	Sets the default email ID
EMAILPW	Sets the default email password
ERRORBYABEND	Determines how SAS responds to a By group error
FONTSLOC	Location of the SAS font file
HELPENCMD	Tell SAS to use the English index for command line requests
HELPINDEX	Sets the location of the help index files
HELPTOC	Sets the location of the help table of contents files
LOGPARM	SAS log file control options
PAGEBREAKINITIAL	Used primarily for Unix to insert page breaks for the log and listing
QUOTELENMAX=	Writes a string maximum length warning to the log in quotes
SORTEQUALS=	Specifies to maintain order of identical values when sorting. SAS performs best when this is set to NO. There is a big degradation of performance when this is set to YES.
SORTSIZE=	The amount of memory allocated for sorting
TERMSTMT=	Opposite of INITCMD. This specifies the SAS statements to be executed at the end of a SAS session
TEXTURELOC=	ODS style location for textures and images
THREADS NOTHREADS	This option turns thread usage on or off. It is on by default.
TOOLSMENU NOTOOLSMENU	This turns the tools menu on and off.
VALIDFMTNAME=	Specifies V9 or previous conventions for format name lengths
VIEWMENU NOVIEWMENU	This option turns the SAS menu on and off.

There are several new system options available in Version 9.1. These options primarily relate to the SAS Open Metadata Server. This server will be used as a one stop metadata location for all organization wide data registered.

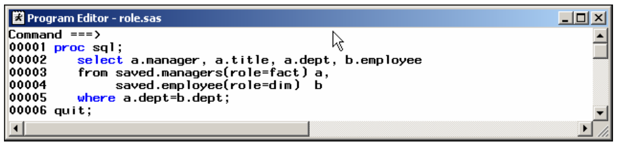
Option	Definition
METAID=	Specifies the installation of SAS on the Open Metadata Server
METAPASS=	Specifies the default password for OMS
METAPORT=	Specifies the TCP/IP port for OMS
METAPROTOCOL=	Specifies the network protocol for OMS
METAREPOSITORY=	Specifies the metadata repository for OMS
METASERVER=	Specifies the OMS
METAUSER=	Specifies the default user for OMS

New Data Set Options

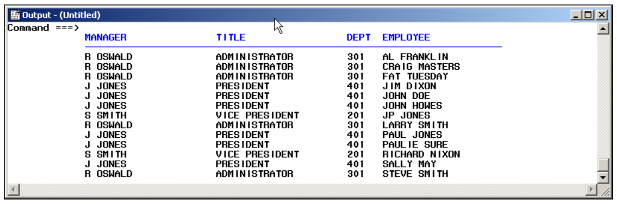
Option	Definition
ENCODING=	This is designed for multinational language support, typically used with SAS/SHARE environments where data is shared between different countries. One data set can be referenced in one character set for one country and a different character set for a different country. See the table below for the encoding values supported.
ROLE=	When the data set is being used in a star schema style join, this table can be labeled FACT or DIMENSION. This can speed up processing if the appropriate tables are labeled. SAS will use the role identification during SQL joins.
SORTSEQ=	Determines the collating sequencing during sorting.
SPILL=NO/YES	This is an option on Data Step Views that tells SAS to produce or not produce spill files when a view is opened for two pass mode. This reduces the amount of disk space required for a spill file. See SAS Documentation for further information on this efficiency and when to use it.

ROLE= Option for Optimized Joins

The following code is a simple example of how we can define particular tables as FACT or DIMENSION files.



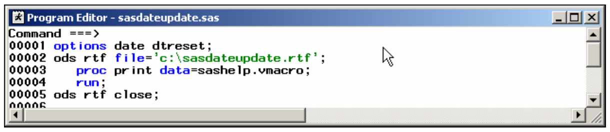
```
Command ===>
proc sql;
   select a.manager, a.title, a.dept, b.employee
   from saved.managers(role=fact) a,
        saved.employee(role=dim)  b
   where a.dept=b.dept;
quit;
```



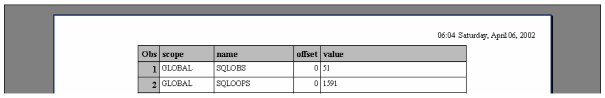
```
Command ===>
          MANAGER              TITLE               DEPT  EMPLOYEE
          ------------------------------------------------------------
          R OSWALD             ADMINISTRATOR       301   AL FRANKLIN
          R OSWALD             ADMINISTRATOR       301   CRAIG MASTERS
          R OSWALD             ADMINISTRATOR       301   FAT TUESDAY
          J JONES              PRESIDENT           401   JIM DIXON
          J JONES              PRESIDENT           401   JOHN DOE
          J JONES              PRESIDENT           401   JOHN HOWES
          S SMITH              VICE PRESIDENT      201   JP JONES
          R OSWALD             ADMINISTRATOR       301   LARRY SMITH
          J JONES              PRESIDENT           401   PAUL JONES
          J JONES              PRESIDENT           401   PAULIE SURE
          S SMITH              VICE PRESIDENT      201   RICHARD NIXON
          J JONES              PRESIDENT           401   SALLY MAY
          R OSWALD             ADMINISTRATOR       301   STEVE SMITH
```

Resetting the SAS Date on Output

SAS introduces the DTRESET system option for resetting the date on SAS output.



```
Command ===>
options date dtreset;
ods rtf file='c:\sasdateupdate.rtf';
   proc print data=sashelp.vmacro;
   run;
ods rtf close;
```



06:04 Saturday, April 06, 2002

Obs	scope	name	offset	value
1	GLOBAL	SQLOBS	0	51
2	GLOBAL	SQLOOPS	0	1591

Putlog Statement

The new PUTLOG statement is designed to always write information to the SAS log, no matter where the FILEREF points to. This can be handy for debugging. It is similar to the PUT statement, but does not reference the FILEREF destination.

Object Dot Syntax

DECLARE Statement

The Data Step is being extended. SAS has introduced the concept of Object Dot Syntax. This is similar to the concept of Dot Notation as applied to Version 8 of SAS Component Language (SCL).

The DECLARE statement has been added to the data step, along with this syntax. This allows declaration of an object. The subsequent syntax allows for methods to be called on that object: Object.method()

Hash Table for Lookups

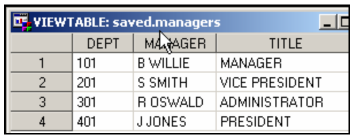
The first use of this new syntax has been introduced through Hash tables. Hash tables are a way of performing a table lookup by loading key variables and values into an array in memory and then matching those values to values being read from a data set.

The beauty of this feature is that the key information lives in memory and not on disk. This is another way of performing lookups that creates a similar result to using:

Proc format
Macro variables
Arrays
SQL Joins
Indexing
Data Step Merging

The hash table grows in memory based on the size of the data loaded. Consider the following example.

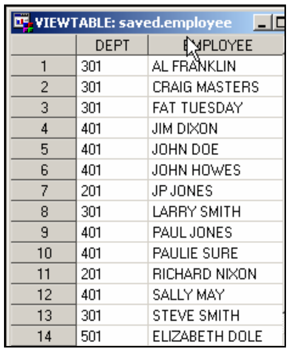
Managers Data Set



VIEWTABLE: saved.managers

	DEPT	MANAGER	TITLE
1	101	B WILLIE	MANAGER
2	201	S SMITH	VICE PRESIDENT
3	301	R OSWALD	ADMINISTRATOR
4	401	J JONES	PRESIDENT

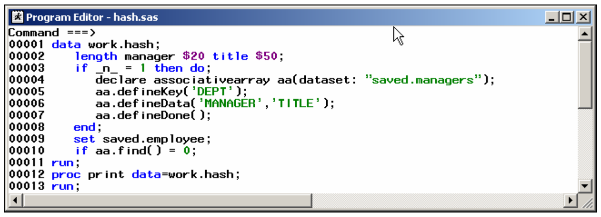
Employee Data Set



VIEWTABLE: saved.employee

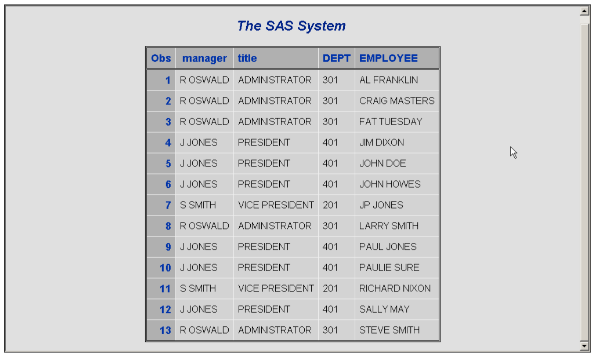
	DEPT	EMPLOYEE
1	301	AL FRANKLIN
2	301	CRAIG MASTERS
3	301	FAT TUESDAY
4	401	JIM DIXON
5	401	JOHN DOE
6	401	JOHN HOWES
7	201	JP JONES
8	301	LARRY SMITH
9	401	PAUL JONES
10	401	PAULIE SURE
11	201	RICHARD NIXON
12	401	SALLY MAY
13	301	STEVE SMITH
14	501	ELIZABETH DOLE

The following syntax will read both files.



```
Command ===>
data work.hash;
   length manager $20 title $50;
   if _n_ = 1 then do;
      declare associativearray aa(dataset: "saved.managers");
      aa.defineKey('DEPT');
      aa.defineData('MANAGER','TITLE');
      aa.defineDone();
   end;
   set saved.employee;
   if aa.find() = 0;
run;
proc print data=work.hash;
run;
```

And yield the following result.



The SAS System

Obs	manager	title	DEPT	EMPLOYEE
1	R OSWALD	ADMINISTRATOR	301	AL FRANKLIN
2	R OSWALD	ADMINISTRATOR	301	CRAIG MASTERS
3	R OSWALD	ADMINISTRATOR	301	FAT TUESDAY
4	J JONES	PRESIDENT	401	JIM DIXON
5	J JONES	PRESIDENT	401	JOHN DOE
6	J JONES	PRESIDENT	401	JOHN HOWES
7	S SMITH	VICE PRESIDENT	201	JP JONES
8	R OSWALD	ADMINISTRATOR	301	LARRY SMITH
9	J JONES	PRESIDENT	401	PAUL JONES
10	J JONES	PRESIDENT	401	PAULIE SURE
11	S SMITH	VICE PRESIDENT	201	RICHARD NIXON
12	J JONES	PRESIDENT	401	SALLY MAY
13	R OSWALD	ADMINISTRATOR	301	STEVE SMITH

Multi-Threaded Architecture

One of the biggest enhancements with SAS software in Version 9 is its ability to support multi-threaded access to files for use in the Data Step and certain procedures. The concept is simple. Instead of using the traditional 'serial' approach to either sorting or summarizing data, SAS now breaks up the data into smaller chunks, performs the operation and then puts the result back together.

Sorting Analogy

Consider the following analogy for sorting. There are four decks of playing cards. The goal is to order them. One person can try to order all four decks together to produce an ordered result. Another way of doing this is to get four people, each tasked with ordering one deck. The ordered four decks are then combined for a final result.

Summarizing Analogy

The goal is to get a total of 100 numbers. One person can sit down with a pencil and paper and total up the 100 numbers to produce a result. Another way of doing this is to get four people to each take 25 numbers and produce 4 totals. The 4 totals are then added up to produce the number.

The division of labor/tasks in these examples demonstrates why a multi-threaded architecture makes sense when possible.

Multi-threading is supported for:

PROC SORT
PROC SUMMARY
PROC MEANS
PROC REPORT
PROC TABULATE
PROC SQL
PROC REG
PROC GLM
PROC ROBUSTREG

By default, multi-threading is turned on in Version 9 for all of these procedures. Therefore, there is a new option available with each procedure (THREADS/NOTHREADS) to optionally turn this feature off.

CPUCOUNT Option

It is important to realize that multi-threading works best in a multiple processor environment. For example, if SAS is running on a four processor server, it will attempt to utilize all of the CPUs available. For large SAS jobs, this may impact performance of other applications or programs running on that server.

The default value of CPUCOUNT is set to the maximum number of CPUs found. This tells SAS to go and use as many processors as it can access. In some situations, it may be wise to limit the number of CPUs accessed by SAS by setting this value to a maximum CPU number. CPUCOUNT is a way to throttle back the number of processors used in multiple processor environments.

Benchmarking

In threading environments, it is possible that the CPU time used to process may actually be larger than real time/wall time. This may be a consideration when scheduling large production jobs.

Scalable Performance Data Architecture

Since Version 6 of SAS, the Scalable Performance Data Server has been available as a separate product that allows for breaking apart data storage to speed up I/O and avoid the traditional 'serial based' I/O architecture. SPDS Software also supported a very intense security model for access to this data. The ideal storage of a data source would be split across several storage locations that were usually separate disks with separate disk controllers.

In Version 9, this entire architecture, except for the security model, is included.

Scalable Performance Data Engine SPDE

This is a new engine that is part of Base SAS software. It is designed to be used on a libname statement for much quicker access to data stored on disk. The ideal environment is the same as listed above.

Each data location is stored on a separate disk
Each disk has a separate disk controller
Each metadata repository is stored in a separate location
Each index is stored in a separate location

Scalable Analogy

Consider a two lane highway with a two toll booth. Traffic can proceed through those toll booths at a particular speed.

Now consider a two lane highway with 10 toll booths. People split apart to pay the tolls and then merge back together back into a two lane highway.

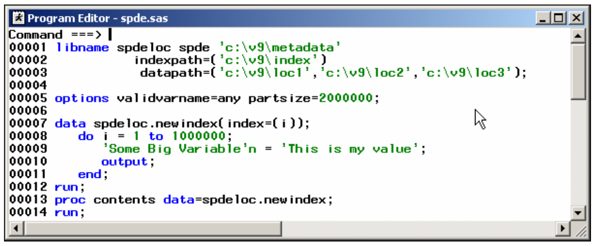
Now consider a 10 lane highway with 10 toll booths.

Now consider a 10 lane highway with no toll booths.

Which one would be fastest?

SPDE performs best when everything is separated into its specific tasks with no stopgaps to slow down processes e.g. a shared disk controller (bad). Performance is best on a 10 lane highway with no toll booths.

Consider the following example to demonstrate the syntax.



```
libname spdeloc spde 'c:\v9\metadata'
           indexpath=('c:\v9\index')
            datapath=('c:\v9\loc1','c:\v9\loc2','c:\v9\loc3');

options validvarname=any partsize=2000000;

data spdeloc.newindex(index=(i));
   do i = 1 to 1000000;
      'Some Big Variable'n = 'This is my value';
      output;
   end;
run;
proc contents data=spdeloc.newindex;
run;
```

This form of the libname statement uses the SPDE engine.

There is a separate location for:

Metadata

Datapath1
Datapath2
Datapath3
Indexes

This example is for syntax only. In an ideal situation, the locations would all reside on separate disks with different disk controllers.

Conclusion

Large amounts of data can be processed effectively with this architecture. The location of data is now split and must be maintained. This is a consideration when moving data. However, the performance gained may be worth the additional housekeeping.

For additional information on SPDE, see

http://www.destinycorp.com/documentation/spde/index.html

PROC CONTENTS

The Contents procedure has been updated to display the native environment of the file, the encoding value and the generation data group.

PROC COPY

The FORCE option has been added to allow MOVEing a data set with audit trails.

The data set attributes are now copied using the CLONE option.

PROC EXPORT

The EXPORT procedure now supports writing to Microsoft Access and Excel 2002 files.

The EXPORT procedure also supports SHEET destinations with the new SHEET= option.

PROC FORMAT

The FORMAT procedure now supports numeric format names that are 32 bytes long and character format names 31 bytes long.

The FORMAT procedure now supports numeric informat names that are 31 bytes long and character informat names 30 bytes long.

PROC FREQ

Multi-threading is not supported for this procedure.

PROC IMPORT

The IMPORT procedure now supports reading to Microsoft Access and Excel 2002 files.

PROC MEANS

This procedure supports mutli-threading by default and uses the THREADS/NOTHREADS options.

PROC PRTDEF

This procedure is useful for batch definitions of printers, especially under the UNIX environment.

PROC PRTEXP

This procedure allows for writing attributes of printers to SAS data sets for easy distribution of printer settings.

PROC REGISTRY

The new LISTREG options presents registry information to the SAS Log.

Registries of two different SAS installations can be compared by using the COMPAREREG1 and COMPAREREG2 options.

PROC REPORT

The BEST12. format is now the default numeric format.

This procedure supports mutli-threading by default and uses the THREADS/NOTHREADS options.

PROC SORT

The new DATECOPY option uses the original date and time stamp for the new SAS data set.

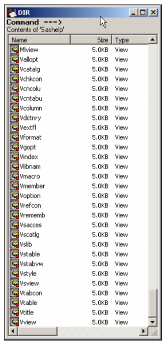
The new THREADS/NOTHREADS options turns sorting in a multi threaded mode on and off on the procedure level.

PROC SUMMARY

This procedure supports mutli-threading by default and uses the THREADS/NOTHREADS options.

PROC SQL

New Dictionary tables are now supported. The complete list is below.



DIR

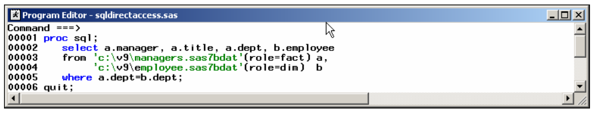
Command ===>

Contents of 'Sashelp'

Name	Size	Type
Mlview	5.0KB	View
Vallopt	5.0KB	View
Vcatalg	5.0KB	View
Vchkcon	5.0KB	View
Vcncolu	5.0KB	View
Vcntabu	5.0KB	View
Vcolumn	5.0KB	View
Vdctnry	5.0KB	View
Vextfl	5.0KB	View
Vformat	5.0KB	View
Vgopt	5.0KB	View
Vindex	5.0KB	View
Vlibnam	5.0KB	View
Vmacro	5.0KB	View
Vmember	5.0KB	View
Voption	5.0KB	View
Vrefcon	5.0KB	View
Vrememb	5.0KB	View
Vsacces	5.0KB	View
Vscatlg	5.0KB	View
Vslib	5.0KB	View
Vstable	5.0KB	View
Vstabvw	5.0KB	View
Vstyle	5.0KB	View
Vsview	5.0KB	View
Vtabcon	5.0KB	View
Vtable	5.0KB	View
Vtitle	5.0KB	View
Vview	5.0KB	View

PROC SQL now has a THREADS/NOTHREADS option to turn mutli-threading on and off.

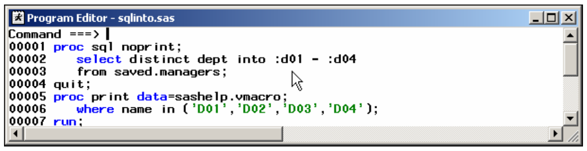
A SAS data set can be reference by the real, physical location.



Program Editor - sqldirectaccess.sas

```
Command ===>
proc sql;
   select a.manager, a.title, a.dept, b.employee
   from 'c:\v9\managers.sas7bdat'(role=fact) a,
        'c:\v9\employee.sas7bdat'(role=dim)  b
   where a.dept=b.dept;
quit;
```

Leading zeros are supported when using the INTO clause.



Program Editor - sqlinto.sas

```
Command ===>
proc sql noprint;
   select distinct dept into :d01 - :d04
   from saved.managers;
quit;
proc print data=sashelp.vmacro;
   where name in ('D01','D02','D03','D04');
run;
```

PROC TABULATE

This procedure supports mutli-threading by default and uses the THREADS/NOTHREADS options.

The TABULATE procedure now supports:

Upper confidence limits with the ALPHA= option
Lower confidence limits with the ALPHA= option
Kurtosis
Skewness

PROC TIMEPLOT

Labels can now be split on multiple lines when using the SPLIT= option.

PROC UNIVARIATE

Multi-threading is not supported for this procedure.

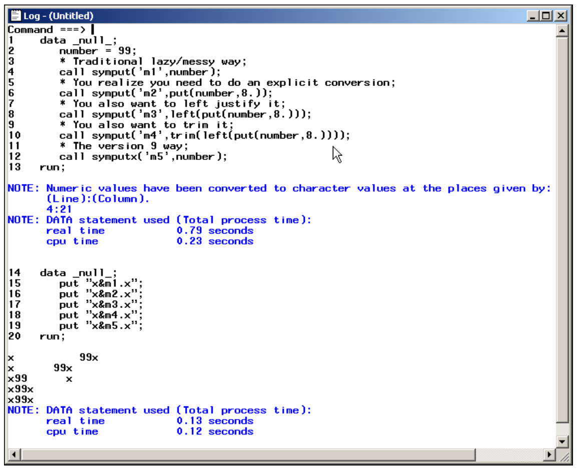
The HISTOGRAM statement now supports the FRONTREF option for displaying reference lines.

The HISTOGRAM statement now supports lower and upper bounds for fitted kernel density curves. The new options are LOWER= and UPPER=.

Call Symputx Macro Statement

This is a new statement that creates a macro variable at execution time in the data step by:

left justifying the value
trimming trailing blanks
automatically converting numeric values to character



Log - (Untitled)

```
Command ===>
1     data _null_;
2        number = 99;
3        * Traditional lazy/messy way;
4        call symput('m1',number);
5        * You realize you need to do an explicit conversion;
6        call symput('m2',put(number,8.));
7        * You also want to left justify it;
8        call symput('m3',left(put(number,8.)));
9        * You also want to trim it;
10       call symput('m4',trim(left(put(number,8.))));
11       * The version 9 way;
12       call symputx('m5',number);
13    run;

NOTE: Numeric values have been converted to character values at the places given by:
      (Line):(Column).
      4:21
NOTE: DATA statement used (Total process time):
      real time           0.79 seconds
      cpu time            0.23 seconds


14    data _null_;
15       put "x&m1.x";
16       put "x&m2.x";
17       put "x&m3.x";
18       put "x&m4.x";
19       put "x&m5.x";
20    run;

x          99x
x      99x
x99      x
x99x
x99x
NOTE: DATA statement used (Total process time):
      real time           0.13 seconds
      cpu time            0.12 seconds
```

Application Response Measurement (ARM) Macros

These macros have been included with the SAS System and enhanced for Version 9. They support monitoring of SAS processes. Information is written out in an industry standard form for ARM applications to read. These macros can be used by IT for production SAS application monitoring and by developers and programmers for ad hoc benchmarking of SAS programs.

XML Enhancements

As XML matures in the industry, we see that many vendors and systems support XML as the ideal transport form for data. SAS has always supported the creation or reading of XML in its basic form. However, XML can be a very customized structure and is quite often dependent on each vendor's or industry's specifications. For example, the Financial industry may have its own version of XML that would be very different from the Pharmaceutical industry.

SAS has responded to this by creating what is called the XMLMAP. This uses a 'MAP' file to interpret non-standard XML data for use in SAS.

This is a public sample of NHL hockey teams in XML form. It has a special structure. Our goal is to read it into SAS.

We create a file called a MAP. This file is designed to interpret the layout of this XML file to something SAS can read.

Atlas Utility

The process of creating a map can be time consuming and tedious. SAS has created a utility to make this simple. It is called Atlas. Through the use of a GUI, it can help you map out the data in the XML file to a MAP file for use by SAS.

New ODS Statements

CHTML Statement

This statement creates the simplest HTML possible without using styles.

CSV Statement

This statement is designed to create a comma delimited CSV file of table information. These types of files are typically imported into Excel.

CSVALL Statement

The statement is designed to create a CSV file while preserving titles, notes and bylines.

Formatted Excel Tip

Output from SAS can create formatted data that Excel can read. Consider creating an HTML file with ODS, an XLS extension and then opening it up in Excel. See the following example.

DOCBOOK Statement

This statement creates XML files and supports the DocBook DTD format from Oasis.

DOCUMENT Statement

This statement is designed to change the order or type of display of any output through ODS without having to rerun the procedures.

Notice the following program that creates the Tabulate and Freq ODS Documents and then replays them in RTF and PDF form.

HTMLCSS Statement

This statement is designed to create a Cascading Style Sheet document from an existing SAS style sheet, alongside HTML. This statement will also use an existing Cascading Style sheet if specified.

IMODE Statement

This statement produces HTML in a column form that is separated by lines.

LATEX Statement

This statement produces LaTeX output for high quality typesetting systems.

MARKUP Statement

This is an example that allows for custom tag set creation. Any customized form of tag sets can be created, registered and used in ODS.

The following example is supplied by SAS Institute and demonstrates how background colors can be changed on every other row of output. The tag set is created and then used.

PCL Statement

This statement is designed to create information for HP LaserJet emulation.

TROFF Statement

This statement creates Troff markup language for high quality laser printing and typesetting.

WML Statement

This statement is designed to created Wireless Markup Language for WAP based (phone display) environments with an HREF table of contents.

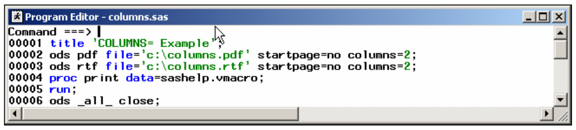
WMLOLIST Statement

This statement is designed to also create Wireless Markup Language with a table of contents option list.
New ODS Options

There are several new options in ODS. They are typically designed for better control and formatting of output.

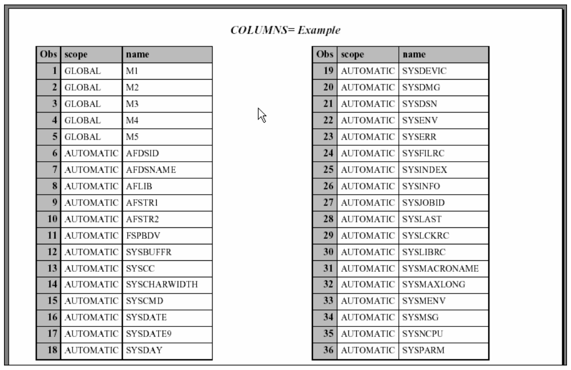
Columns Option

This option is designed to create multiple columns in output.



```
Command ===>
title 'COLUMNS= Example';
ods pdf file='c:\columns.pdf' startpage=no columns=2;
ods rtf file='c:\columns.rtf' startpage=no columns=2;
proc print data=sashelp.vmacro;
run;
ods _all_ close;
```

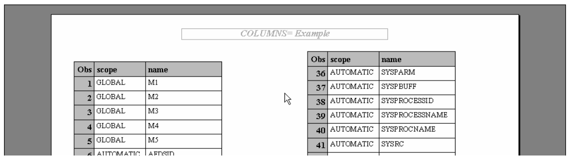
PDF Output



COLUMNS= Example

Obs	scope	name	Obs	scope	name
1	GLOBAL	M1	19	AUTOMATIC	SYSDEVIC
2	GLOBAL	M2	20	AUTOMATIC	SYSDMG
3	GLOBAL	M3	21	AUTOMATIC	SYSDSN
4	GLOBAL	M4	22	AUTOMATIC	SYSENV
5	GLOBAL	M5	23	AUTOMATIC	SYSERR
6	AUTOMATIC	AFDSID	24	AUTOMATIC	SYSFILRC
7	AUTOMATIC	AFDSNAME	25	AUTOMATIC	SYSINDEX
8	AUTOMATIC	AFLIB	26	AUTOMATIC	SYSINFO
9	AUTOMATIC	AFSTR1	27	AUTOMATIC	SYSJOBID
10	AUTOMATIC	AFSTR2	28	AUTOMATIC	SYSLAST
11	AUTOMATIC	FSPBDV	29	AUTOMATIC	SYSLCKRC
12	AUTOMATIC	SYSBUFFR	30	AUTOMATIC	SYSLIBRC
13	AUTOMATIC	SYSCC	31	AUTOMATIC	SYSMACRONAME
14	AUTOMATIC	SYSCHARWIDTH	32	AUTOMATIC	SYSMAXLONG
15	AUTOMATIC	SYSCMD	33	AUTOMATIC	SYSMENV
16	AUTOMATIC	SYSDATE	34	AUTOMATIC	SYSMSG
17	AUTOMATIC	SYSDATE9	35	AUTOMATIC	SYSNCPU
18	AUTOMATIC	SYSDAY	36	AUTOMATIC	SYSPARM

RTF Output



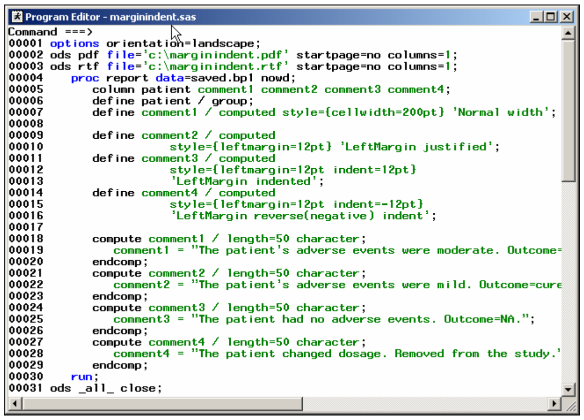
COLUMNS= Example

Obs	scope	name
1	GLOBAL	M1
2	GLOBAL	M2
3	GLOBAL	M3
4	GLOBAL	M4
5	GLOBAL	M5
6	AUTOMATIC	AFDSID

Obs	scope	name
36	AUTOMATIC	SYSPARM
37	AUTOMATIC	SYSPBUFF
38	AUTOMATIC	SYSPROCESSID
39	AUTOMATIC	SYSPROCESSNAME
40	AUTOMATIC	SYSPROCNAME
41	AUTOMATIC	SYSRC

Margin and Indent Options

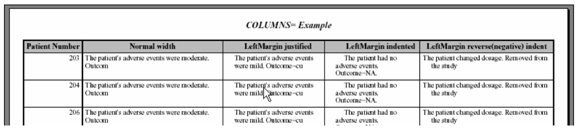
This option is designed to control margins and indentations in output.



Program Editor - marginindent.sas

```
Command ===>
options orientation=landscape;
ods pdf file='c:\marginindent.pdf' startpage=no columns=1;
ods rtf file='c:\marginindent.rtf' startpage=no columns=1;
   proc report data=saved.bp1 nowd;
      column patient comment1 comment2 comment3 comment4;
      define patient / group;
      define comment1 / computed style={cellwidth=200pt} 'Normal width';

      define comment2 / computed
                style={leftmargin=12pt} 'LeftMargin justified';
      define comment3 / computed
                style={leftmargin=12pt indent=12pt}
                'LeftMargin indented';
      define comment4 / computed
                style={leftmargin=12pt indent=-12pt}
                'LeftMargin reverse(negative) indent';

      compute comment1 / length=50 character;
         comment1 = "The patient's adverse events were moderate. Outcome=
      endcomp;
      compute comment2 / length=50 character;
         comment2 = "The patient's adverse events were mild. Outcome=cure
      endcomp;
      compute comment3 / length=50 character;
         comment3 = "The patient had no adverse events. Outcome=NA.";
      endcomp;
      compute comment4 / length=50 character;
         comment4 = "The patient changed dosage. Removed from the study.'
      endcomp;
   run;
ods _all_ close;
```

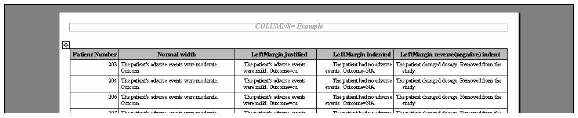
PDF Output



COLUMNS= Example

Patient Number	Normal width	LeftMargin justified	LeftMargin indented	LeftMargin reverse(negative) indent
203	The patient's adverse events were moderate. Outcom	The patient's adverse events were mild. Outcome=cu	The patient had no adverse events. Outcome=NA.	The patient changed dosage. Removed from the study
204	The patient's adverse events were moderate. Outcom	The patient's adverse events were mild. Outcome=cu	The patient had no adverse events. Outcome=NA.	The patient changed dosage. Removed from the study
206	The patient's adverse events were moderate. Outcom	The patient's adverse events were mild. Outcome=cu	The patient had no adverse events.	The patient changed dosage. Removed from the study

RTF Output

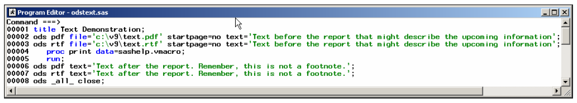


COLUMNS= Example

Patient Number	Normal width	LeftMargin justified	LeftMargin indented	LeftMargin reverse(negative) indent
203	The patient's adverse events were moderate. Outcom	The patient's adverse events were mild. Outcome=cu	The patient had no adverse events. Outcome=NA.	The patient changed dosage. Removed from the study
204	The patient's adverse events were moderate. Outcom	The patient's adverse events were mild. Outcome=cu	The patient had no adverse events. Outcome=NA.	The patient changed dosage. Removed from the study
206	The patient's adverse events were moderate. Outcom	The patient's adverse events were mild. Outcome=cu	The patient had no adverse events. Outcome=NA.	The patient changed dosage. Removed from the study

Text Options

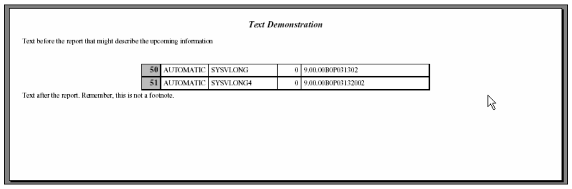
This option is designed to allow placement of text in any location around ODS output.



Program Editor - odstext.sas

```
Command ===>
title Text Demonstration;
ods pdf file='c:\v9\text.pdf' startpage=no text='Text before the report that might describe the upcoming information';
ods rtf file='c:\v9\text.rtf' startpage=no text='Text before the report that might describe the upcoming information';
   proc print data=sashelp.vmacro;
   run;
ods pdf text='Text after the report. Remember, this is not a footnote.';
ods rtf text='Text after the report. Remember, this is not a footnote.';
ods _all_ close;
```

PDF Output



Text Demonstration

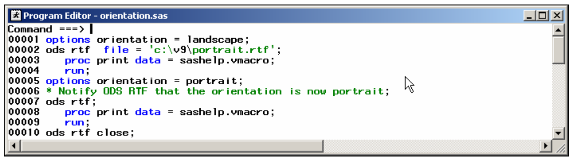
Text before the report that might describe the upcoming information

50	AUTOMATIC	SYSVLONG	0	9.00.00B0P031302
51	AUTOMATIC	SYSVLONG4	0	9.00.00B0P03132002

Text after the report. Remember, this is not a footnote.

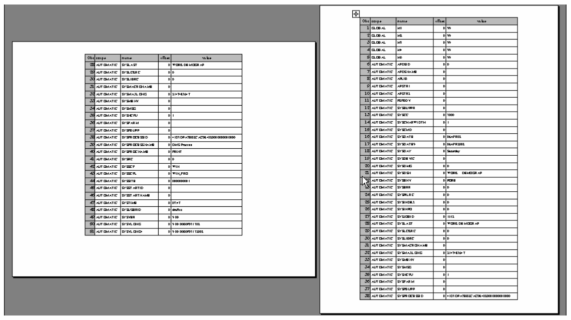
Orientation Option

This option allows the changing of output orientation. Notice the placement of the system option.



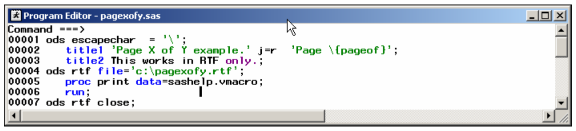
Program Editor - orientation.sas

```
Command ===>
options orientation = landscape;
ods rtf  file = 'c:\v9\portrait.rtf';
   proc print data = sashelp.vmacro;
   run;
options orientation = portrait;
* Notify ODS RTF that the orientation is now portrait;
ods rtf;
   proc print data = sashelp.vmacro;
   run;
ods rtf close;
```



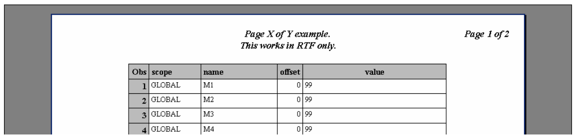
Page X of Y Support

SAS now supports the ability to put page numbers on output with the total pages. This is currently supported for RTF.



Program Editor - pagexofy.sas

```
Command ===>
ods escapechar  = '\';
   title1 'Page X of Y example.' j=r  'Page \{pageof}';
   title2 This works in RTF only.;
ods rtf file='c:\pagexofy.rtf';
   proc print data=sashelp.vmacro;
   run;
ods rtf close;
```



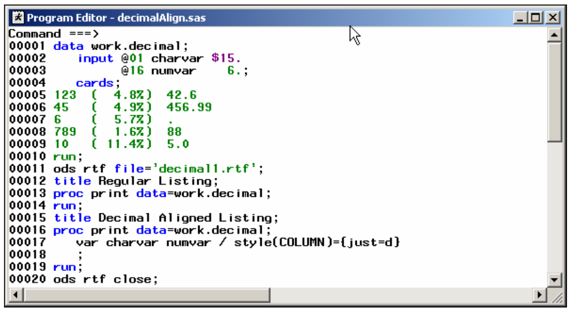
Page X of Y example.
This works in RTF only.

Page 1 of 2

Obs	scope	name	offset	value
1	GLOBAL	M1	0	99
2	GLOBAL	M2	0	99
3	GLOBAL	M3	0	99
4	GLOBAL	M4	0	99

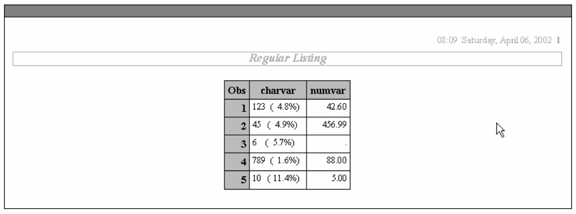
New ODS Styles

Decimal Alignment



Program Editor - decimalAlign.sas

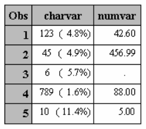
```
Command ===>
data work.decimal;
   input @01 charvar $15.
         @16 numvar    6.;
   cards;
123  (   4.8%)  42.6
45   (   4.9%)  456.99
6    (   5.7%)  .
789  (   1.6%)  88
10   (  11.4%)  5.0
run;
ods rtf file='decimal1.rtf';
title Regular Listing;
proc print data=work.decimal;
run;
title Decimal Aligned Listing;
proc print data=work.decimal;
   var charvar numvar / style(COLUMN)={just=d}
   ;
run;
ods rtf close;
```



08:09 Saturday, April 06, 2002 1

Regular Listing

Obs	charvar	numvar
1	123 (4.8%)	42.60
2	45 (4.9%)	456.99
3	6 (5.7%)	.
4	789 (1.6%)	88.00
5	10 (11.4%)	5.00



Obs	charvar	numvar
1	123 (4.8%)	42.60
2	45 (4.9%)	456.99
3	6 (5.7%)	.
4	789 (1.6%)	88.00
5	10 (11.4%)	5.00

Cross Environment Data Access (CEDA)

CEDA is still supported in Version 9. CEDA is designed for cross operating system access of data files without the need for PROC UPLOAD/DOWNLOAD or creating a transport file.

In Version 9, the SAS log lists messages if CEDA is being used.

CEDA automatically converts 32 bit and 64 bit data on different operating systems.

Libref Inheritance

Client session defined librefs are now inherited and do not have to be reassigned in multiple server session environments. They are read/write.

%SYSLPUT CONNECTREMOTE= Option

Macro variables can be created on a remote session. Because multiple remote sessions may be available, the CONNECTREMOTE= option allows specification of the session name.

Remote Library Services

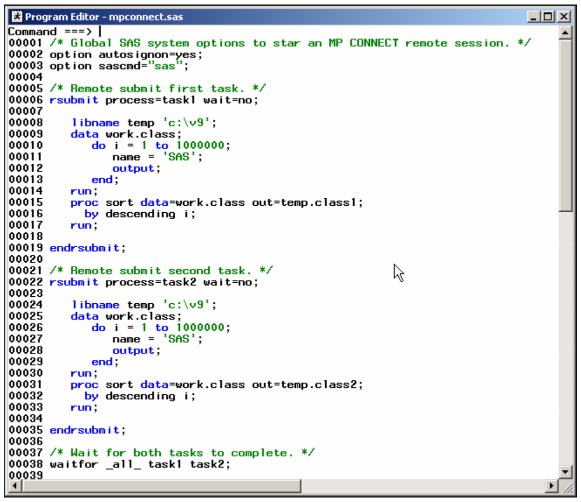
Version 9 clients do not connect to Version 6 remote sessions using this feature anymore.

MP CONNECT

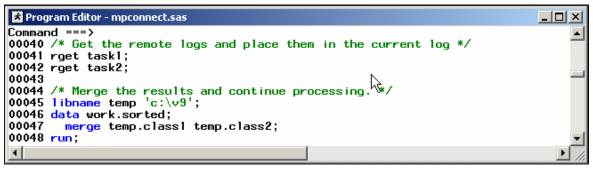
This enhances traditional SAS/Connect software with asynchronous processing that is now production. Asynchronous processing is available on remotely submitted servers or processes on multiple processor systems. MP CONNECT processing can increase the performance of processes.

Traditional Example

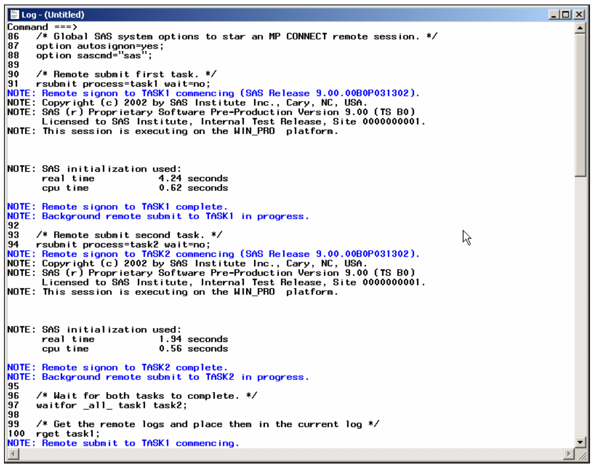
This is a simple example of how MP CONNECT can work.



```
Command ===> |
/* Global SAS system options to star an MP CONNECT remote session. */
option autosignon=yes;
option sascmd="sas";

/* Remote submit first task. */
rsubmit process=task1 wait=no;

   libname temp 'c:\v9';
   data work.class;
      do i = 1 to 1000000;
         name = 'SAS';
         output;
      end;
   run;
   proc sort data=work.class out=temp.class1;
     by descending i;
   run;

endrsubmit;

/* Remote submit second task. */
rsubmit process=task2 wait=no;

   libname temp 'c:\v9';
   data work.class;
      do i = 1 to 1000000;
         name = 'SAS';
         output;
      end;
   run;
   proc sort data=work.class out=temp.class2;
     by descending i;
   run;

endrsubmit;

/* Wait for both tasks to complete. */
waitfor _all_ task1 task2;

```



```
Command ===>
/* Get the remote logs and place them in the current log */
rget task1;
rget task2;

/* Merge the results and continue processing. */
libname temp 'c:\v9';
data work.sorted;
  merge temp.class1 temp.class2;
run;
```



```
Command ===>
86    /* Global SAS system options to star an MP CONNECT remote session. */
87    option autosignon=yes;
88    option sascmd="sas";
89
90    /* Remote submit first task. */
91    rsubmit process=task1 wait=no;
NOTE: Remote signon to TASK1 commencing (SAS Release 9.00.00B0P031302).
NOTE: Copyright (c) 2002 by SAS Institute Inc., Cary, NC, USA.
NOTE: SAS (r) Proprietary Software Pre-Production Version 9.00 (TS B0)
      Licensed to SAS Institute, Internal Test Release, Site 0000000001.
NOTE: This session is executing on the WIN_PRO  platform.



NOTE: SAS initialization used:
      real time           4.24 seconds
      cpu time            0.62 seconds

NOTE: Remote signon to TASK1 complete.
NOTE: Background remote submit to TASK1 in progress.
92
93    /* Remote submit second task. */
94    rsubmit process=task2 wait=no;
NOTE: Remote signon to TASK2 commencing (SAS Release 9.00.00B0P031302).
NOTE: Copyright (c) 2002 by SAS Institute Inc., Cary, NC, USA.
NOTE: SAS (r) Proprietary Software Pre-Production Version 9.00 (TS B0)
      Licensed to SAS Institute, Internal Test Release, Site 0000000001.
NOTE: This session is executing on the WIN_PRO  platform.



NOTE: SAS initialization used:
      real time           1.94 seconds
      cpu time            0.56 seconds

NOTE: Remote signon to TASK2 complete.
NOTE: Background remote submit to TASK2 in progress.
95
96    /* Wait for both tasks to complete. */
97    waitfor _all_ task1 task2;
98
99    /* Get the remote logs and place them in the current log */
100   rget task1;
NOTE: Remote submit to TASK1 commencing.
```

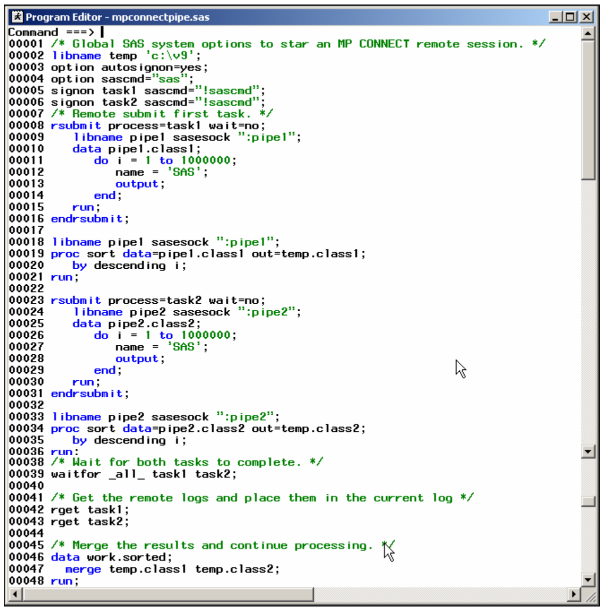
Notice the RGET statements to allow the local SAS log to see the remote SAS log information.

Notice the WAITFOR statements which must be used at strategic points in processing.

Piping Example

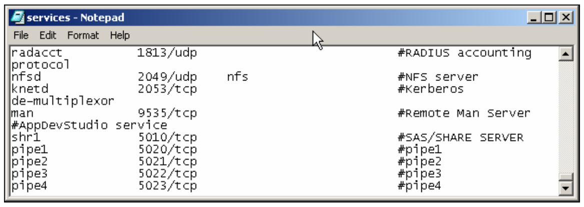
Piping is a new methodology in SAS where the output of one step is automatically being fed to the input of a subsequent step. When the process is appropriate, for example, the creation of a data set with a data step feeding directly into a subsequent proc sort, performance increases are possible. The output of the first process is not written to disk. The input of the second process does not read from disk. All information is passed through memory.

This is a simple example of how piping might be used. Notice the use of real physical storage to pass data between sessions.



```
/* Global SAS system options to star an MP CONNECT remote session. */
libname temp 'c:\v9';
option autosignon=yes;
option sascmd="sas";
signon task1 sascmd="!sascmd";
signon task2 sascmd="!sascmd";
/* Remote submit first task. */
rsubmit process=task1 wait=no;
   libname pipe1 sasesock ":pipe1";
   data pipe1.class1;
      do i = 1 to 1000000;
         name = 'SAS';
         output;
      end;
   run;
endrsubmit;

libname pipe1 sasesock ":pipe1";
proc sort data=pipe1.class1 out=temp.class1;
   by descending i;
run;

rsubmit process=task2 wait=no;
   libname pipe2 sasesock ":pipe2";
   data pipe2.class2;
      do i = 1 to 1000000;
         name = 'SAS';
         output;
      end;
   run;
endrsubmit;

libname pipe2 sasesock ":pipe2";
proc sort data=pipe2.class2 out=temp.class2;
   by descending i;
run;
/* Wait for both tasks to complete. */
waitfor _all_ task1 task2;

/* Get the remote logs and place them in the current log */
rget task1;
rget task2;

/* Merge the results and continue processing. */
data work.sorted;
  merge temp.class1 temp.class2;
run;
```

In addition, the pipe locations must be listed in the services file. Notice pipe1 – pipe4 in the file listed below.



```
radacct           1813/udp                         #RADIUS accounting
protocol
nfsd              2049/udp    nfs                  #NFS server
knetd             2053/tcp                         #Kerberos
de-multiplexor
man               9535/tcp                         #Remote Man Server
#AppDevStudio service
shr1              5010/tcp                         #SAS/SHARE SERVER
pipe1             5020/tcp                         #pipe1
pipe2             5021/tcp                         #pipe2
pipe3             5022/tcp                         #pipe3
pipe4             5023/tcp                         #pipe4
```

New Graph Procedures

Three new procedures now exist in SAS/GRAPH software.

GBARLINE

This procedure creates vertical bar charts with a plot line.

GAREABAR

This procedure creates bar charts where the width of the bar represents a prescribed value.

MAPIMPORT

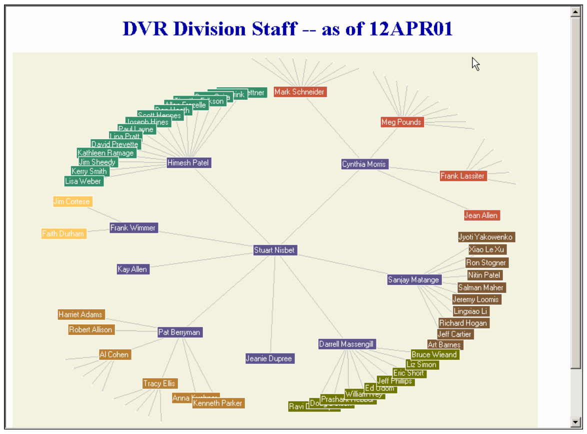
This procedure imports map information.

Java Applets Supported by SAS/Graph

Treeview Java Applet

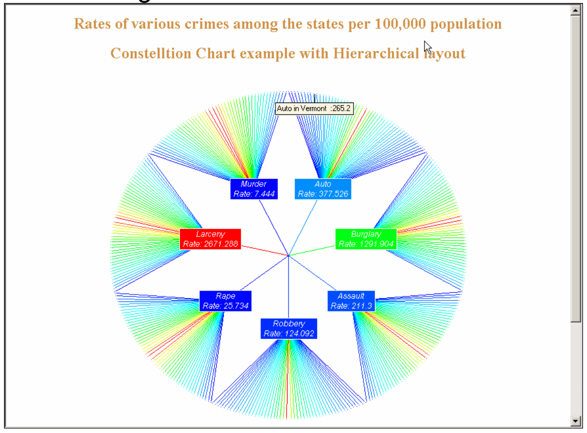
The new Treeview macro has been introduced to create a Java Applet for an interactive 'tree' style report. This creates a JAR file that is publishable via the web. The following code is a snapshot of the beginning of the macro.

This macro and all subsequent macros can be found in the following directory (under Windows). In a web browser, the resulting JAR file yields the following, interactive display.



Constellation Java Applet

Another macro exists to create a constellation JAR file. The result of running this macro yields an interactive JAR file like the following.



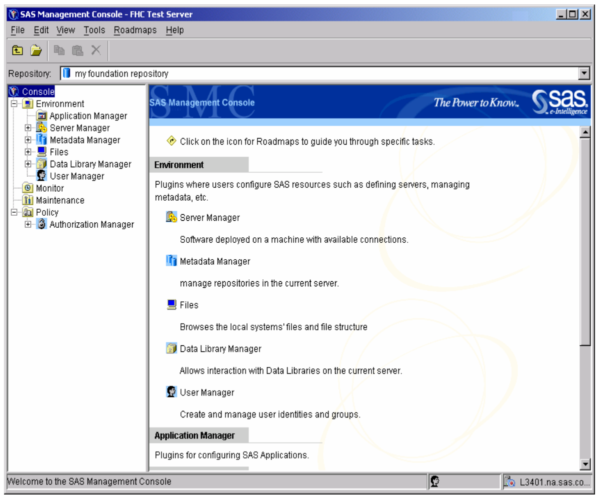
SAS Management Console

The SAS Management Console is designed to allow for managing and monitoring metadata, servers, libraries of data, servers and security from one, central point in an organization.

This application also supports third part plug ins.

Aspects of the management console include:

Server Manager – manage SAS servers
Metadata Manager –interact with running repository servers
Application Manager – SAS application plug-ins
SAS Library Manager – define and manage SAS libraries
User Manager – define SAS Users and Groups
Authorization Manager - administer authorization policy for accessing SAS Metadata and OLAP



Click on the icon for Roadmaps to guide you through specific tasks.

Environment

Plugins where users configure SAS resources such as defining servers, managing metadata, etc.

Server Manager

Software deployed on a machine with available connections.

Metadata Manager

manage repositories in the current server.

Files

Browses the local systems' files and file structure

Data Library Manager

Allows interaction with Data Libraries on the current server.

User Manager

Create and manage user identities and groups.

Application Manager

Plugins for configuring SAS Applications.

Welcome to the SAS Management Console

The benefits of the SAS Management Console are:

- Simplify administrative tasks by using the same tools for all SAS products and solutions.
- Reduce staff training and support time.
- Build standard and repeatable processes for SAS operations.
- Define and manage connections to servers:
- Application
- Database
- SAS
- IOM Bridge
- Others
- Define and manage SAS users and groups.
- Policy descriptions: set values for user and group attributes (such as read, write, etc.)
- Manage SAS Library definitions.
- Manage Database Schemas.
- Administer authorization policy for accessing SAS Metadata and OLAP:
- Permission Creation
- Access Control Templates
- Resource Authorization Definitions
- Supports plug-ins to administer SAS applications, or user-written SAS applications.
- Manage SAS licenses (SETINIT):
- The SAS License Manager (SLM) displays SAS installed software information.
- Monitor SAS processes:
- Determine where SAS is running
- Identify user of a SAS process
- Identify "orphans"
- Interface with resource managers

Operating Systems Supported

SAS V9 is supported under all of the Windows operating systems, including NT, 2000 and XP, but not supported under Windows 95, 98 and Me.

Operating System	Size
Windows NT4.0/2000/XP (WNT/W2K/WXP)	32 bit
OpenVMS Alpha 7.2	64 bit
Compaq's Digital UNIX 5.1	64 bit
HP HP-UX 64bit 11.0	64 bit
Solaris 64bit Solaris8	64 bit
AIX 64bit 5.1	64 bit
RedHat Linux 7.2 on Intel	32 bit
OS/390 (MVS) V2R10	32 bit

Conclusion

As you can see, SAS Version 9 offers a whole new way of processing that yields more choices to take your programming to the next level.