

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

Copyright ©2003 Destiny Corporation. All Rights Reserved.

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.

Copyright ©2003 Destiny Corporation

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.

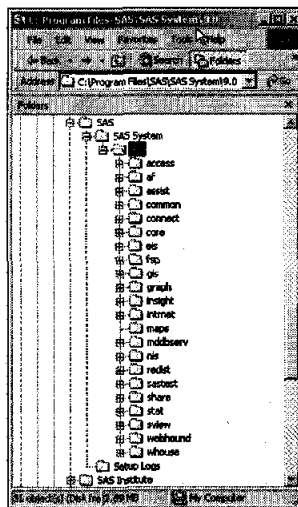
Cross Environment Data Access (CEDA) is still supported on all operating systems.

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.



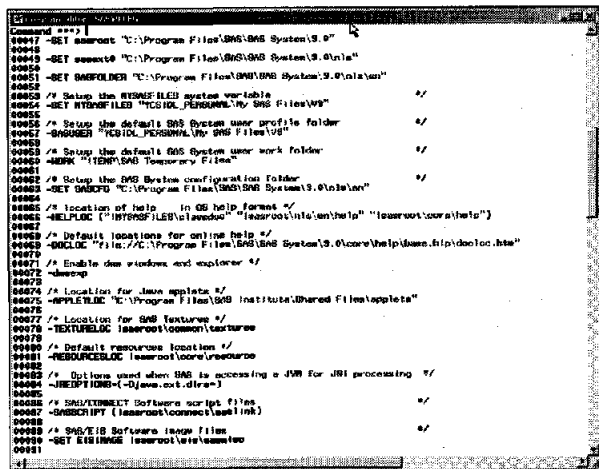
The new directory for Version 9 is SAS. SAS V9 is designed to live alongside SAS V8. Notice the SAS V9 directory and the SAS Institute V8 directory.

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.



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.

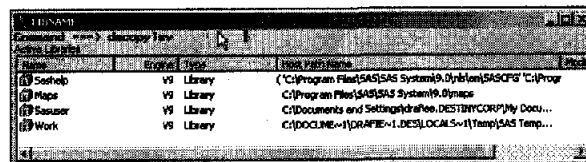
See the SAS installation instructions for space and configuration requirements for your operating system and environment.

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.

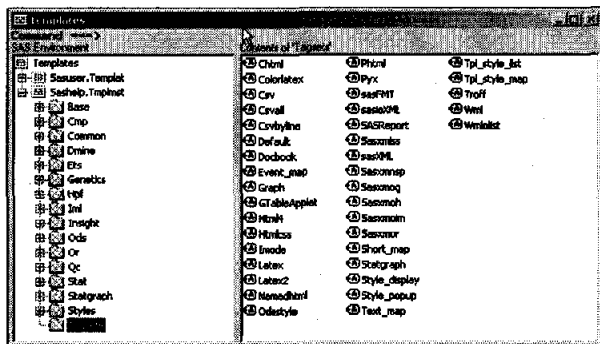


The result is the following information. This has many uses, including documentation configuration documentation.

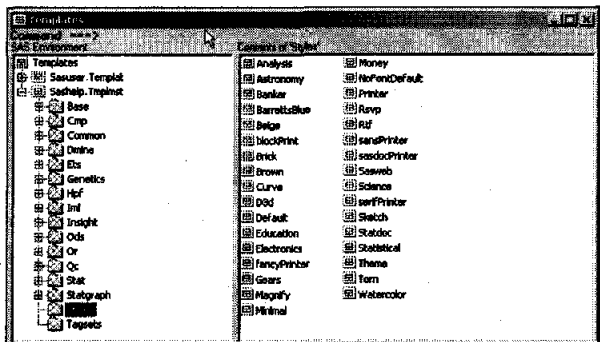
"Name", "Engine", "Type", "Host Path Name", "Modified",
"Sashelp", "V9", "Library", "('C:\Program Files\SAS\SAS
System\9.0\ntl\en\SASCFG' 'C:\Program Files\SAS\SAS
System\9.0\core\sashelp' 'C:\Program Files\SAS\SAS
System\9.0\fts\sashelp' 'C:\Program Files\SAS\SAS
System\9.0\assist\sashelp' 'C:\Program 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\intnet\sashelp' 'C:\Program
Files\SAS\SAS System\9.0\mdbdbserv\sashelp'
'C:\Program Files\SAS\SAS System\9.0\svview\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\drafee.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.

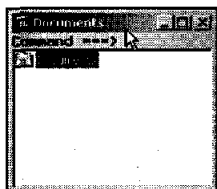


Select Styles to see all of the styles available.



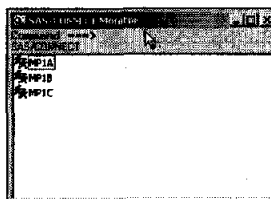
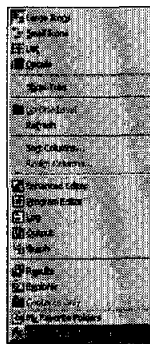
ODS Document Viewer

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.

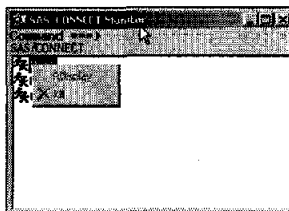


SAS/Connect Monitor

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.

```

Command --->
00001 libname version9 v9 'c:\v9';
00002 libname version8 v8 'c:\v8';
00003 libname version7 v7 'c:\v7';
00004 libname version6 v6 'c:\v6';
  
```

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
.sas7bcatalog	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
.sas7bview	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 --->
00001 libname tranfile 'c:\tranfile';
00002 libname old v8 'c:\data\sas\old';
00003 libname new v9 'c:\v9';
00004 proc cport library=old files=tranfile;
00005 select class98;
00006 run;
00007 proc cimport library=new files=tranfile;
00008 select class98;
00009 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 --->
00001 data work.dates;
00002 infile cards;
00003 input $91 string $ 20.;
00004 extracteddate = input(string,anydtdate32.);
00005 extractedtime = input(string,anydttime32.);
00006 extracteddatetime = input(string,anydtmtime32.);
00007 datalines;
00008 2002-Apr-15
00009 April 15, 2002
00010 April 15 2002
00011 200203
00012 15APR2002
00013 2002110
00014 2002/04/15
00015 2002/04/15:11:22:33
00016 15APR2002:11:22:33
00017 12:04:56
00018 APR2002
00019 15/04/2002
00020 04/15/2002
00021 run;
00022 proc print data=work.dates;
00023 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 ==>
00001 options datestyle=MDY;
00002
00003 data work.dates;
00004 infile cards;
00005 input @1 string $ 20.;
00006 extracteddate = input(string,anydtmm2.);
00007 extractedtime = input(string,anydtmm2.);
00008 extracteddatetime = input(string,anydtmm2.);
00009 datelines;
00010 2002-Apr-15
00011 Apr 11 15, 2002
00012 Apr 11 15, 2002
00013 200202
00014 15APR2002
00015 2002110
00016 2002/04/15
00017 2002/04/15:11:22:33
00018 15APR2002:11:22:33
00019 12:34:56
00020 APR2002
00021 15/04/2002
00022 04/15/2002
00023 run;
00024 proc print data=work.dates;
00025 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 ==>
00001 data work.prx;
00002 length param dash $ 20;
00003 if _N_ = 1 then do;
00004   retain param dash;
00005   param=prxparse('/^([2-9]d\d\d)3[2-9]d\d\d-([2-9]d\d\d)4$/');
00006   dash='';
00007 end;
00008 length first last phone $ 15;
00009 input first last phone $ 40.;
00010 if prxmatch(param, phone) then param = 'NO';
00011 if prxmatch(dash, phone) then param = 'YES';
00012 if prxmatch(dash, phone) then dash = 'YES';
00013
00014 datalines;
00015 S.G. Johnson (900)721-1594
00016 Sally Smith 888-455-1234
00017 Tommy Toad (550) 454-5777
00018 Queen Infobears (705)145-0380
00019 Pauline Kent (900)555-1212
00020 run;
00021 proc print data=work.prx;
00022 run;
```

first	last	phone	param	dash
S.G.	Johnson	(900)721-1594	NO	
Sally	Smith	888-455-1234	YES	
Tommy	Toad	(550) 454-5777	NO	
Queen	Infobears	(705)145-0380	NO	
Pauline	Kent	(900)555-1212	YES	

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

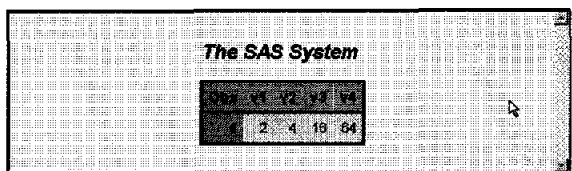
CALL SORTQ is a quick way to sort variable values inside the Data Step. It is not designed to replace PROC SORT. It is a simple way of ordering values of the same structure. For example, if it is used with character variables, they must be the same length.

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

```

Command --->
00001 data work.one;
00002   v1 = 16;
00003   v2 = 2;
00004   v3 = 4;
00005   v4 = 64;
00006   call sortq(v1,v2,v3,v4);
00007 run;
00008 proc print data=work.one;
00009 run;

```



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
FONTSLC	Location of the SAS font file
HELPCMD	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 --->
00001 proc sql;
00002   select a.manager, a.title, a.dept, b.employee
00003   from saved.managers(role=fact) a,
00004        saved.employees(role=dia) b,
00005        where a.dept=b.dept;
00006 quit;

```

MANAGER	TITLE	DEPT	EMPLOYEE
R DONALD	ADMINISTRATOR	301	AL FUNKLIN
R DONALD	ADMINISTRATOR	301	CHRIS MARTENS
R DONALD	ADMINISTRATOR	301	PAT TUESDAY
J JONES	PRESIDENT	401	JIM DIXON
J JONES	PRESIDENT	401	JOHN DOE
J JONES	PRESIDENT	401	JOHN HENNES
S SMITH	VICE PRESIDENT	201	J JONES
R DONALD	ADMINISTRATOR	301	LARRY SMITH
J JONES	PRESIDENT	401	PAUL JONES
J JONES	PRESIDENT	401	PAULIE BLUE
S SMITH	VICE PRESIDENT	201	RICHARD NIXON
J JONES	PRESIDENT	401	BALLT RAY
R DONALD	ADMINISTRATOR	301	STEVE SMITH

Resetting the SAS Date on Output

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

```

Command --->
00001 options date dtreset;
00002 ods rtf file='c:\sasdateupdate.rtf';
00003 proc print data=sashelp.vmacro;
00004 run;
00005 ods rtf close;

```

GLOBAL	SQL089	0 15
GLOBAL	SQL008	0 1501

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

VIEW TABLE: 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

VIEW TABLE: saved.employees		
	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 ===>
00001 data work.hash;
00002 length manager $20 title $50;
00003 if _n_ = 1 then do;
00004   declare associative array aa(dataset: "saved.managers");
00005   aa.defineKey('DEPT');
00006   aa.defineData('MANAGER','TITLE');
00007   aa.defineDone();
00008 end;
00009 set saved.employees;
00010 if aa.find() = 0;
00011 run;
00012 proc print data=work.hash;
00013 run;

```

And yield the following result.

The SAS System			
	DEPT	EMPLOYEE	
1	R OSWALD	ADMINISTRATOR	301
2	R OSWALD	ADMINISTRATOR	301
3	R OSWALD	ADMINISTRATOR	301
4	J JONES	PRESIDENT	401
5	J JONES	PRESIDENT	401
6	J JONES	PRESIDENT	401
7	S SMITH	VICE PRESIDENT	201
8	R OSWALD	ADMINISTRATOR	301
9	J JONES	PRESIDENT	401
10	J JONES	PRESIDENT	401
11	S SMITH	VICE PRESIDENT	201
12	J JONES	PRESIDENT	401
13	R OSWALD	ADMINISTRATOR	301

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/NOTTHREADS) 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.

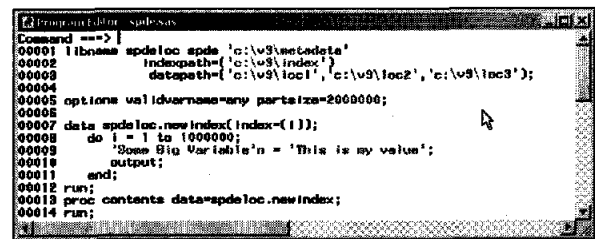
Now consider a 10 lane highway with 10 toll booths.

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.



```
Command Window
00001 libname spdeloc spde 'c:\v9\metadata'
00002 indexpath='c:\v9\index';
00003 datapath='c:\v9\loc1', 'c:\v9\loc2', 'c:\v9\loc3';
00004
00005 options valldvarname=any partsize=2000000;
00006
00007 data spdeloc.newindex(index=1);
00008 do i = 1 to 1000000;
00009   'Some Big Variable'n = 'This is my value';
00010   output;
00011 and;
00012 run;
00013 proc contents data=spdeloc.newindex;
00014 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/NOTHEADS 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 multi-threading by default and uses the THREADS/NOTHEADS options.

PROC SORT

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

The new THREADS/NOTTHREADS 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/NOTHEADS options.

PROC SQL

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

[illegible]

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

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

```

C:\Users\j\Documents> update user test
Command ===>
00001 proc sql;
00002     select a.manager, a.title, a.dept, b.employees
00003     from c:\u01\manager.asa7bdat (role=foot) a
00004     c:\u01\employees.asa7bdat (role=din) b
00005     where a.dept=b.dept;
00006 quit;

```

Leading zeros are supported when using the INTO clause.

```

Command ---> |
00001 proc sql noprint;
00002 select distinct dept into :d01 - :d04
00003 from saved.managers;
00004 quit;
00005 proc print data=sashenv.vsource;
00006 where name in ('D01','D02','D03','D04');
00007 run;

```

PROC TABULATE

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

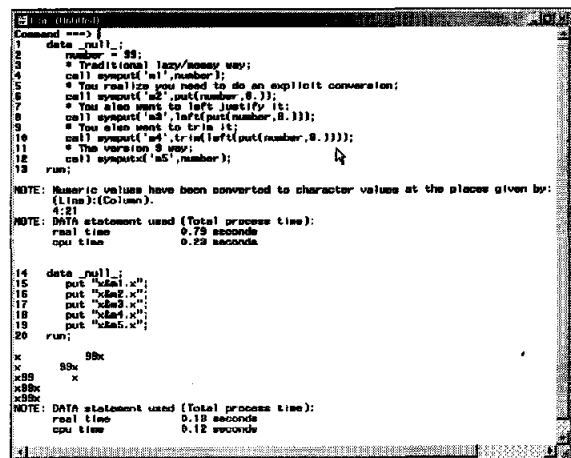
The HISTOGRAM statement now supports the FRONTREF option for displaying reference lines.

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



```
1 data _null_;
2   number = 99;
3   * Traditional lazy/sloppy way;
4   call symput('a1',number);
5   * You realize you need to do an explicit conversion;
6   call symput('a2',put(number,8.));
7   * You also want to left justify it;
8   call symput('a3',left(put(number,8.)));
9   * You also want to trim it;
10  call symput('a4',trim(left(put(number,8.))));
11  * The version 9 way;
12  call symputx('a5',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     0.28 seconds

14 data _null_;
15   put 'a1' x;
16   put 'a2' x;
17   put 'a3' x;
18   put 'a4' x;
19   put 'a5' x;
20 run;

a1      99x
a2      99x
a3      99x
a4      99x
a5      99x

NOTE: DATA statement used (Total process time):
      real time    0.10 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 create Wireless Markup Language for WAP based (phone display) environments with an HREF table of contents.

WMLIST Statement

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

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 ===>
00001 title 'COLUMNS= Example';
00002 ods pdf file='c:\columns.pdf' startpage=no columns=2;
00003 ods rtf file='c:\columns.rtf' startpage=no columns=2;
00004 proc print data=sashelp.vaccro;
00005 run;
00006 ods _all_ close;
```

PDF Output

COLUMNS= Example

GLOBAL	M1	AUTOMATIC	SYSDTIME
GLOBAL	M2	AUTOMATIC	SYSDM3
GLOBAL	M3	AUTOMATIC	SYSDM4
GLOBAL	M4	AUTOMATIC	SYSDM5
GLOBAL	M5	AUTOMATIC	SYSDM6
AUTOMATIC	APST00	AUTOMATIC	SYSDM7
AUTOMATIC	APST01	AUTOMATIC	SYSDM8
AUTOMATIC	APST02	AUTOMATIC	SYSDM9
AUTOMATIC	APST03	AUTOMATIC	SYSDM10
AUTOMATIC	APST04	AUTOMATIC	SYSDM11
AUTOMATIC	APST05	AUTOMATIC	SYSDM12
AUTOMATIC	APST06	AUTOMATIC	SYSDM13
AUTOMATIC	APST07	AUTOMATIC	SYSDM14
AUTOMATIC	APST08	AUTOMATIC	SYSDM15
AUTOMATIC	APST09	AUTOMATIC	SYSDM16
AUTOMATIC	APST10	AUTOMATIC	SYSDM17
AUTOMATIC	APST11	AUTOMATIC	SYSDM18
AUTOMATIC	APST12	AUTOMATIC	SYSDM19
AUTOMATIC	APST13	AUTOMATIC	SYSDM20
AUTOMATIC	APST14	AUTOMATIC	SYSDM21
AUTOMATIC	APST15	AUTOMATIC	SYSDM22
AUTOMATIC	APST16	AUTOMATIC	SYSDM23
AUTOMATIC	APST17	AUTOMATIC	SYSDM24
AUTOMATIC	APST18	AUTOMATIC	SYSDM25
AUTOMATIC	APST19	AUTOMATIC	SYSDM26
AUTOMATIC	APST20	AUTOMATIC	SYSDM27
AUTOMATIC	APST21	AUTOMATIC	SYSDM28
AUTOMATIC	APST22	AUTOMATIC	SYSDM29
AUTOMATIC	APST23	AUTOMATIC	SYSDM30
AUTOMATIC	APST24	AUTOMATIC	SYSDM31
AUTOMATIC	APST25	AUTOMATIC	SYSDM32
AUTOMATIC	APST26	AUTOMATIC	SYSDM33
AUTOMATIC	APST27	AUTOMATIC	SYSDM34
AUTOMATIC	APST28	AUTOMATIC	SYSDM35
AUTOMATIC	APST29	AUTOMATIC	SYSDM36
AUTOMATIC	APST30	AUTOMATIC	SYSDM37
AUTOMATIC	APST31	AUTOMATIC	SYSDM38
AUTOMATIC	APST32	AUTOMATIC	SYSDM39
AUTOMATIC	APST33	AUTOMATIC	SYSDM40
AUTOMATIC	APST34	AUTOMATIC	SYSDM41
AUTOMATIC	APST35	AUTOMATIC	SYSDM42
AUTOMATIC	APST36	AUTOMATIC	SYSDM43
AUTOMATIC	APST37	AUTOMATIC	SYSDM44
AUTOMATIC	APST38	AUTOMATIC	SYSDM45
AUTOMATIC	APST39	AUTOMATIC	SYSDM46
AUTOMATIC	APST40	AUTOMATIC	SYSDM47
AUTOMATIC	APST41	AUTOMATIC	SYSDM48
AUTOMATIC	APST42	AUTOMATIC	SYSDM49
AUTOMATIC	APST43	AUTOMATIC	SYSDM50
AUTOMATIC	APST44	AUTOMATIC	SYSDM51
AUTOMATIC	APST45	AUTOMATIC	SYSDM52
AUTOMATIC	APST46	AUTOMATIC	SYSDM53
AUTOMATIC	APST47	AUTOMATIC	SYSDM54
AUTOMATIC	APST48	AUTOMATIC	SYSDM55
AUTOMATIC	APST49	AUTOMATIC	SYSDM56
AUTOMATIC	APST50	AUTOMATIC	SYSDM57
AUTOMATIC	APST51	AUTOMATIC	SYSDM58
AUTOMATIC	APST52	AUTOMATIC	SYSDM59
AUTOMATIC	APST53	AUTOMATIC	SYSDM60
AUTOMATIC	APST54	AUTOMATIC	SYSDM61
AUTOMATIC	APST55	AUTOMATIC	SYSDM62
AUTOMATIC	APST56	AUTOMATIC	SYSDM63
AUTOMATIC	APST57	AUTOMATIC	SYSDM64
AUTOMATIC	APST58	AUTOMATIC	SYSDM65
AUTOMATIC	APST59	AUTOMATIC	SYSDM66
AUTOMATIC	APST60	AUTOMATIC	SYSDM67
AUTOMATIC	APST61	AUTOMATIC	SYSDM68
AUTOMATIC	APST62	AUTOMATIC	SYSDM69
AUTOMATIC	APST63	AUTOMATIC	SYSDM70
AUTOMATIC	APST64	AUTOMATIC	SYSDM71
AUTOMATIC	APST65	AUTOMATIC	SYSDM72
AUTOMATIC	APST66	AUTOMATIC	SYSDM73
AUTOMATIC	APST67	AUTOMATIC	SYSDM74
AUTOMATIC	APST68	AUTOMATIC	SYSDM75
AUTOMATIC	APST69	AUTOMATIC	SYSDM76
AUTOMATIC	APST70	AUTOMATIC	SYSDM77
AUTOMATIC	APST71	AUTOMATIC	SYSDM78
AUTOMATIC	APST72	AUTOMATIC	SYSDM79
AUTOMATIC	APST73	AUTOMATIC	SYSDM80
AUTOMATIC	APST74	AUTOMATIC	SYSDM81
AUTOMATIC	APST75	AUTOMATIC	SYSDM82
AUTOMATIC	APST76	AUTOMATIC	SYSDM83
AUTOMATIC	APST77	AUTOMATIC	SYSDM84
AUTOMATIC	APST78	AUTOMATIC	SYSDM85
AUTOMATIC	APST79	AUTOMATIC	SYSDM86
AUTOMATIC	APST80	AUTOMATIC	SYSDM87
AUTOMATIC	APST81	AUTOMATIC	SYSDM88
AUTOMATIC	APST82	AUTOMATIC	SYSDM89
AUTOMATIC	APST83	AUTOMATIC	SYSDM90
AUTOMATIC	APST84	AUTOMATIC	SYSDM91
AUTOMATIC	APST85	AUTOMATIC	SYSDM92
AUTOMATIC	APST86	AUTOMATIC	SYSDM93
AUTOMATIC	APST87	AUTOMATIC	SYSDM94
AUTOMATIC	APST88	AUTOMATIC	SYSDM95
AUTOMATIC	APST89	AUTOMATIC	SYSDM96
AUTOMATIC	APST90	AUTOMATIC	SYSDM97
AUTOMATIC	APST91	AUTOMATIC	SYSDM98
AUTOMATIC	APST92	AUTOMATIC	SYSDM99
AUTOMATIC	APST93	AUTOMATIC	SYSDM100
AUTOMATIC	APST94	AUTOMATIC	SYSDM101
AUTOMATIC	APST95	AUTOMATIC	SYSDM102
AUTOMATIC	APST96	AUTOMATIC	SYSDM103
AUTOMATIC	APST97	AUTOMATIC	SYSDM104
AUTOMATIC	APST98	AUTOMATIC	SYSDM105
AUTOMATIC	APST99	AUTOMATIC	SYSDM106
AUTOMATIC	APST100	AUTOMATIC	SYSDM107
AUTOMATIC	APST101	AUTOMATIC	SYSDM108
AUTOMATIC	APST102	AUTOMATIC	SYSDM109
AUTOMATIC	APST103	AUTOMATIC	SYSDM110
AUTOMATIC	APST104	AUTOMATIC	SYSDM111
AUTOMATIC	APST105	AUTOMATIC	SYSDM112
AUTOMATIC	APST106	AUTOMATIC	SYSDM113
AUTOMATIC	APST107	AUTOMATIC	SYSDM114
AUTOMATIC	APST108	AUTOMATIC	SYSDM115
AUTOMATIC	APST109	AUTOMATIC	SYSDM116
AUTOMATIC	APST110	AUTOMATIC	SYSDM117
AUTOMATIC	APST111	AUTOMATIC	SYSDM118
AUTOMATIC	APST112	AUTOMATIC	SYSDM119
AUTOMATIC	APST113	AUTOMATIC	SYSDM120
AUTOMATIC	APST114	AUTOMATIC	SYSDM121
AUTOMATIC	APST115	AUTOMATIC	SYSDM122
AUTOMATIC	APST116	AUTOMATIC	SYSDM123
AUTOMATIC	APST117	AUTOMATIC	SYSDM124
AUTOMATIC	APST118	AUTOMATIC	SYSDM125
AUTOMATIC	APST119	AUTOMATIC	SYSDM126
AUTOMATIC	APST120	AUTOMATIC	SYSDM127
AUTOMATIC	APST121	AUTOMATIC	SYSDM128
AUTOMATIC	APST122	AUTOMATIC	SYSDM129
AUTOMATIC	APST123	AUTOMATIC	SYSDM130
AUTOMATIC	APST124	AUTOMATIC	SYSDM131
AUTOMATIC	APST125	AUTOMATIC	SYSDM132
AUTOMATIC	APST126	AUTOMATIC	SYSDM133
AUTOMATIC	APST127	AUTOMATIC	SYSDM134
AUTOMATIC	APST128	AUTOMATIC	SYSDM135
AUTOMATIC	APST129	AUTOMATIC	SYSDM136
AUTOMATIC	APST130	AUTOMATIC	SYSDM137
AUTOMATIC	APST131	AUTOMATIC	SYSDM138
AUTOMATIC	APST132	AUTOMATIC	SYSDM139
AUTOMATIC	APST133	AUTOMATIC	SYSDM140
AUTOMATIC	APST134	AUTOMATIC	SYSDM141
AUTOMATIC	APST135	AUTOMATIC	SYSDM142
AUTOMATIC	APST136	AUTOMATIC	SYSDM143
AUTOMATIC	APST137	AUTOMATIC	SYSDM144
AUTOMATIC	APST138	AUTOMATIC	SYSDM145
AUTOMATIC	APST139	AUTOMATIC	SYSDM146
AUTOMATIC	APST140	AUTOMATIC	SYSDM147
AUTOMATIC	APST141	AUTOMATIC	SYSDM148
AUTOMATIC	APST142	AUTOMATIC	SYSDM149
AUTOMATIC	APST143	AUTOMATIC	SYSDM150
AUTOMATIC	APST144	AUTOMATIC	SYSDM151
AUTOMATIC	APST145	AUTOMATIC	SYSDM152
AUTOMATIC	APST146	AUTOMATIC	SYSDM153
AUTOMATIC	APST147	AUTOMATIC	SYSDM154
AUTOMATIC	APST148	AUTOMATIC	SYSDM155
AUTOMATIC	APST149	AUTOMATIC	SYSDM156
AUTOMATIC	APST150	AUTOMATIC	SYSDM157
AUTOMATIC	APST151	AUTOMATIC	SYSDM158
AUTOMATIC	APST152	AUTOMATIC	SYSDM159
AUTOMATIC	APST153	AUTOMATIC	SYSDM160
AUTOMATIC	APST154	AUTOMATIC	SYSDM161
AUTOMATIC	APST155	AUTOMATIC	SYSDM162
AUTOMATIC	APST156	AUTOMATIC	SYSDM163
AUTOMATIC	APST157	AUTOMATIC	SYSDM164
AUTOMATIC	APST158	AUTOMATIC	SYSDM165
AUTOMATIC	APST159	AUTOMATIC	SYSDM166
AUTOMATIC	APST160	AUTOMATIC	SYSDM167
AUTOMATIC	APST161	AUTOMATIC	SYSDM168
AUTOMATIC	APST162	AUTOMATIC	SYSDM169
AUTOMATIC	APST163	AUTOMATIC	SYSDM170
AUTOMATIC	APST164	AUTOMATIC	SYSDM171
AUTOMATIC	APST165	AUTOMATIC	SYSDM172
AUTOMATIC	APST166	AUTOMATIC	SYSDM173
AUTOMATIC	APST167	AUTOMATIC	SYSDM174
AUTOMATIC	APST168	AUTOMATIC	SYSDM175
AUTOMATIC	APST169	AUTOMATIC	SYSDM176
AUTOMATIC	APST170	AUTOMATIC	SYSDM177
AUTOMATIC	APST171	AUTOMATIC	SYSDM178
AUTOMATIC	APST172	AUTOMATIC	SYSDM179
AUTOMATIC	APST173	AUTOMATIC	SYSDM180
AUTOMATIC	APST174	AUTOMATIC	SYSDM181
AUTOMATIC	APST175	AUTOMATIC	SYSDM182
AUTOMATIC	APST176	AUTOMATIC	SYSDM183
AUTOMATIC	APST177	AUTOMATIC	SYSDM184
AUTOMATIC	APST178	AUTOMATIC	SYSDM185
AUTOMATIC	APST179	AUTOMATIC	SYSDM186
AUTOMATIC	APST180	AUTOMATIC	SYSDM187
AUTOMATIC	APST181	AUTOMATIC	SYSDM188
AUTOMATIC	APST182	AUTOMATIC	SYSDM189
AUTOMATIC	APST183	AUTOMATIC	SYSDM190
AUTOMATIC	APST184	AUTOMATIC	SYSDM191
AUTOMATIC	APST185	AUTOMATIC	SYSDM192
AUTOMATIC	APST186	AUTOMATIC	SYSDM193
AUTOMATIC	APST187	AUTOMATIC	SYSDM194
AUTOMATIC	APST188	AUTOMATIC	SYSDM195
AUTOMATIC	APST189	AUTOMATIC	SYSDM196
AUTOMATIC	APST190	AUTOMATIC	SYSDM197
AUTOMATIC	APST191	AUTOMATIC	SYSDM198
AUTOMATIC	APST192	AUTOMATIC	SYSDM199
AUTOMATIC	APST193	AUTOMATIC	SYSDM200
AUTOMATIC	APST194	AUTOMATIC	SYSDM201
AUTOMATIC	APST195	AUTOMATIC	SYSDM202
AUTOMATIC	APST196	AUTOMATIC	SYSDM203
AUTOMATIC	APST197	AUTOMATIC	SYSDM204
AUTOMATIC	APST198	AUTOMATIC	SYSDM205
AUTOMATIC	APST199	AUTOMATIC	SYSDM206
AUTOMATIC	APST200	AUTOMATIC	SYSDM207
AUTOMATIC	APST201	AUTOMATIC	SYSDM208
AUTOMATIC	APST202	AUTOMATIC	SYSDM209
AUTOMATIC	APST203	AUTOMATIC	SYSDM210
AUTOMATIC	APST204	AUTOMATIC	SYSDM211
AUTOMATIC	APST205	AUTOMATIC	SYSDM212
AUTOMATIC	APST206	AUTOMATIC	SYSDM213
AUTOMATIC	APST207	AUTOMATIC	SYSDM214
AUTOMATIC	APST208	AUTOMATIC	SYSDM215
AUTOMATIC	APST209	AUTOMATIC	SYSDM216
AUTOMATIC	APST210	AUTOMATIC	SYSDM217
AUTOMATIC	APST211	AUTOMATIC	SYSDM218
AUTOMATIC	APST212	AUTOMATIC	SYSDM219
AUTOMATIC	APST213	AUTOMATIC	SYSDM220
AUTOMATIC	APST214	AUTOMATIC	SYSDM221
AUTOMATIC	APST215	AUTOMATIC	SYSDM222
AUTOMATIC	APST216	AUTOMATIC	SYSDM223
AUTOMATIC	APST217	AUTOMATIC	SYSDM224
AUTOMATIC	APST218	AUTOMATIC	SYSDM225
AUTOMATIC	APST219	AUTOMATIC	SYSDM226
AUTOMATIC	APST220	AUTOMATIC	SYSDM227
AUTOMATIC	APST221	AUTOMATIC	SYSDM228
AUTOMATIC	APST222	AUTOMATIC	SYSDM229
AUTOMATIC	APST223	AUTOMATIC	SYSDM230
AUTOMATIC	APST224	AUTOMATIC	SYSDM231
AUTOMATIC	APST225	AUTOMATIC	SYSDM232
AUTOMATIC	APST226	AUTOMATIC	SYSDM233
AUTOMATIC	APST227	AUTOMATIC	SYSDM234
AUTOMATIC	APST228	AUTOMATIC	SYSDM235
AUTOMATIC	APST229	AUTOMATIC	SYSDM236
AUTOMATIC	APST230	AUTOMATIC	SYSDM237
AUTOMATIC	APST231	AUTOMATIC	SYSDM238
AUTOMATIC	APST232	AUTOMATIC	SYSDM239
AUTOMATIC	APST233	AUTOMATIC	SYSDM240
AUTOMATIC	APST234	AUTOMATIC	SYSDM241
AUTOMATIC	APST235	AUTOMATIC	SYSDM242
AUTOMATIC	APST236	AUTOMATIC	SYSDM243
AUTOMATIC	APST237	AUTOMATIC	SYSDM244
AUTOMATIC	APST238	AUTOMATIC	SYSDM245
AUTOMATIC	APST239	AUTOMATIC	SYSDM246
AUTOMATIC	APST240	AUTOMATIC	SYSDM247
AUTOMATIC	APST241	AUTOMATIC	SYSDM248
AUTOMATIC	APST242	AUTOMATIC	SYSDM249
AUTOMATIC	APST243	AUTOMATIC	SYSDM250
AUTOMATIC	APST244	AUTOMATIC	SYSDM251
AUTOMATIC	APST245	AUTOMATIC	SYSDM252
AUTOMATIC	APST246	AUTOMATIC	SYSDM253
AUTOMATIC	APST247	AUTOMATIC	SYSDM254
AUTOMATIC	APST248	AUTOMATIC	SYSDM255
AUTOMATIC	APST249	AUTOMATIC	SYSDM256
AUTOMATIC	APST250	AUTOMATIC	SYSDM257
AUTOMATIC	APST251	AUTOMATIC	SYSDM258
AUTOMATIC	APST252	AUTOMATIC	SYSDM259
AUTOMATIC	APST253	AUTOMATIC	SYSDM260
AUTOMATIC	APST254	AUTOMATIC	SYSDM261
AUTOMATIC	APST255	AUTOMATIC	SYSDM262
AUTOMATIC	APST256	AUTOMATIC	SYSDM263
AUTOMATIC	APST257	AUTOMATIC	SYSDM264
AUTOMATIC	APST258	AUTOMATIC	SYSDM265
AUTOMATIC	APST259	AUTOMATIC	SYSDM266
AUTOMATIC	APST260	AUTOMATIC	SYSDM267
AUTOMATIC	APST261	AUTOMATIC	SYSDM268
AUTOMATIC	APST262	AUTOMATIC	SYSDM269
AUTOMATIC	APST263	AUTOMATIC	SYSDM270
AUTOMATIC	APST264	AUTOMATIC	SYSDM271
AUTOMATIC	APST265	AUTOMATIC	SYSDM272
AUTOMATIC	APST266	AUTOMATIC	SYSDM273
AUTOMATIC	APST267	AUTOMATIC	SYSDM274
AUTOMATIC	APST268	AUTOMATIC	SYSDM275
AUTOMATIC	APST269	AUTOMATIC	SYSDM276
AUTOMATIC	APST270	AUTOMATIC	SYSDM277
AUTOMATIC	APST271	AUTOMATIC	SYSDM278
AUTOMATIC	APST272	AUTOMATIC	SYSDM279
AUTOMATIC	APST273	AUTOMATIC	SYSDM280
AUTOMATIC	APST274	AUTOMATIC	SYSDM281
AUTOMATIC	APST275	AUTOMATIC	SYSDM282
AUTOMATIC	APST276	AUTOMATIC	SYSDM283
AUTOMATIC	APST277	AUTOMATIC	SYSDM284
AUTOMATIC	APST278	AUTOMATIC	SYSDM285
AUTOMATIC	APST279	AUTOMATIC	SYSDM286
AUTOMATIC	APST280	AUTOMATIC	SYSDM287
AUTOMATIC	APST281	AUTOMATIC	SYSDM288
AUTOMATIC	APST282	AUTOMATIC	SYSDM289
AUTOMATIC	APST283	AUTOMATIC	SYSD

PDF Output

Original Text	Reference text	1. Subject adverb phrase	2. Subject adverb phrase	3. Subject adverb phrase	4. Subject adverb phrase
201 The patient's adverse events were not serious. Collins		The patient's adverse events were not serious Collins	The patient's adverse events were not serious Collins	The patient's adverse events were not serious Collins	The patient's adverse events were not serious Collins
202 The patient's adverse events were not serious. Collins		The patient's adverse events were not serious Collins	The patient's adverse events were not serious Collins	The patient's adverse events were not serious Collins	The patient's adverse events were not serious Collins
203 The patient's adverse events were not serious. Collins		The patient's adverse events were not serious Collins	The patient's adverse events were not serious Collins	The patient's adverse events were not serious Collins	The patient's adverse events were not serious Collins
204 The patient's adverse events were not serious. Collins		The patient's adverse events were not serious Collins	The patient's adverse events were not serious Collins	The patient's adverse events were not serious Collins	The patient's adverse events were not serious Collins

RTF Output

[illegible]

Text Options

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

[illegible]

PDF Output

Test before the app is shut right down the operating information

SQL	ALTERATION	SYSTEM LOGS	IP
SQL	ALTERATION	SYSTEM LOGS	IP

Test after the system, transaction, the table is finished.

Orientation Option

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

```

[1] Command Line: gimpshelp2.exe
Command ==> 1
#####
s0001 options orientation = landscape;
s0002 ods rtf file = "c:\windows\trial.rtf";
s0003 proc print data = sashelp.vsource;
s0004 run;
s0005 options orientation = portrait;
s0006 o Note: ODS RTF that the orientation is now portrait;
s0007 ods rtf;
s0008 proc print data = sashelp.vsource;
s0009 run;
s0010 ods rtf close;
#####

```

[illegible]

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.

```
C:\Program Files\Parsons Systems>
Command ->
00001 ode workspace = '';
00002 title('Page X of Y example.'); % Page {pageno}';
00003 title('This works in RTT only.');
00004 ode rtf file='c:\temp\rtf.rtf';
00005 proc print data=workspace.vectors;
00006 run;
00007 ode rtf close;
```

Page X of Y example.			Page 1 of 2		
Title verbatim RTF only.					
Page	Page	Page	Page	Page	Page
ORIGINAL	SA	0 PP			
ORIGINAL	SA	0 PP			
ORIGINAL	SA	0 PP			
ORIGINAL	SA	0 PP			

New ODS Styles

Decimal Alignment

```

C:\Program Files\IBM\SPSS\PC\spss>
Command ===>
00001 data work.decimal;
00002   input @01 charvar $15.;
00003         @15 numvar $15.;
00004   cards;
00005 129 ( 4.8%) 42.6
00006 45 ( 4.9%) 458.39
00007 5 ( 5.7%)
00008 788 ( 1.6%) 88
00009 19 ( 11.4%) 5.0
00010 run;
00011 ods rtf file='decimal1.rtf';
00012 title Regular Listing;
00013 proc print data=work.decimal;
00014 run;
00015 title Decimal Aligned Listing;
00016 proc print data=work.decimal;
00017   var charvar numvar / style(COLUMN)=c(just=d)
00018 run;
00019 run;
00020 ods rtf close;

```

10/20/2003

Regular Listing

123 (4.8%)	42.60
45 (4.9%)	456.99
6 (3.7%)	
780 (1.6%)	88.00
10 (11.4%)	5.00

Case	Charges	Amount
1	123 (4.8%)	41.60
2	45 (4.5%)	455.99
3	6 (5.7%)	-
4	789 (1.6%)	88.08
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 ===>
00001 /* Global SAS system options to start an MP CONNECT remote session. */
00002 option autosign=yes;
00003 option sascomp="sas";
00004
00005 /* Remote submit first task. */
00006 submit process=task1 wait=no;
00007
00008   libname temp 'c:\v9';
00009   data work.class;
00010     do i = 1 to 100000;
00011       name = 'SAS';
00012       output;
00013     end;
00014 run;
00015 proc sort data=work.class out=temp.class1;
00016   by descending i;
00017 run;
00018
00019 endsubmit;
00020
00021 /* Remote submit second task. */
00022 submit process=task2 wait=no;
00023
00024   libname temp 'c:\v9';
00025   data work.class;
00026     do i = 1 to 100000;
00027       name = 'SAS';
00028       output;
00029     end;
00030 run;
00031 proc sort data=work.class out=temp.class2;
00032   by descending i;
00033 run;
00034
00035 endsubmit;
00036
00037 /* Wait for both tasks to complete. */
00038 waitfor _all_ task1 task2;
00039
```

```
Command ===>
00040 /* Get the remote logs and place them in the current log */
00041 rget task1;
00042 rget task2;
00043
00044 /* Merge the results and continue processing. */
00045 libname temp 'c:\v9';
00046 data work.sorted;
00047   merge temp.class1 temp.class2;
00048 run;
```

```
Command ===>
00001 /* Global SAS system options to start an MP CONNECT remote session. */
00002 option autosign=yes;
00003 option sascomp="sas";
00004
00005 /* Remote submit first task. */
00006 submit process=task1 wait=no;
NOTE: Remote submit to TASK1 commencing (SAS Release 9.00.000P001002).
NOTE: Copyright (c) 2000 by SAS Institute Inc., Cary, NC, USA
NOTE: SAS (r) Proprietary Software: Pre-Production Version 9.00 (SAS 90)
NOTE: Licensed to SAS Institute, Internal Test Release, Site 0000000001.
NOTE: This session is executing on the WIN_P90 platform.

NOTE: SAS initialization used:
      real time      4.26 seconds
      cpu time       0.52 seconds

NOTE: Remote submit to TASK1 complete.
NOTE: Background remote submit to TASK1 in progress.
00007
00008 /* Remote submit second task. */
00009 submit process=task2 wait=no;
NOTE: Remote submit to TASK2 commencing (SAS Release 9.00.000P001002).
NOTE: Copyright (c) 2000 by SAS Institute Inc., Cary, NC, USA
NOTE: SAS (r) Proprietary Software: Pre-Production Version 9.00 (SAS 90)
NOTE: Licensed to SAS Institute, Internal Test Release, Site 0000000001.
NOTE: This session is executing on the WIN_P90 platform.

NOTE: SAS initialization used:
      real time      1.34 seconds
      cpu time       0.58 seconds

NOTE: Remote submit to TASK2 complete.
NOTE: Background remote submit to TASK2 in progress.
00010
00011 /* Wait for both tasks to complete. */
00012 waitfor _all_ task1 task2;
00013
00014 /* Get the remote logs and place them in the current log */
00015 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.

```
Command ===>
00001 /* Global SAS system options to start an MP CONNECT remote session. */
00002 option autosign=yes;
00003 option sascomp="sas";
00004
00005 /* Remote submit first task. */
00006 submit process=task1 wait=no;
00007
00008   libname pipe1 sascomp "pipe1";
00009   data pipe1.class1;
00010     do i = 1 to 100000;
00011       name = 'SAS';
00012       output;
00013     end;
00014 run;
00015
00016 endsubmit;
00017
00018 libname pipe2 sascomp "pipe2";
00019 proc sort data=pipe1.class1 out=temp.class1;
00020   by descending i;
00021 run;
00022
00023 submit process=task2 wait=no;
00024
00025   libname pipe2 sascomp "pipe2";
00026   data pipe2.class2;
00027     do i = 1 to 100000;
00028       name = 'SAS';
00029       output;
00030     end;
00031 run;
00032
00033 endsubmit;
00034
00035 libname pipe2 sascomp "pipe2";
00036 proc sort data=pipe2.class2 out=temp.class2;
00037   by descending i;
00038 run;
00039
00040 /* Wait for both tasks to complete. */
00041 waitfor _all_ task1 task2;
00042
00043 /* Get the remote logs and place them in the current log */
00044 rget task1;
00045 rget task2;
00046
00047 /* Merge the results and continue processing. */
00048 data work.sorted;
00049   merge temp.class1 temp.class2;
00050 run;
```

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

#services: Subnet			
File Edit Format Help			
radacct	1813/udp		#RADIUS accounting
protocol			
nfsd	2049/udp	nfs	#NFS server
knetd	2033/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.

MAPIIMPORT

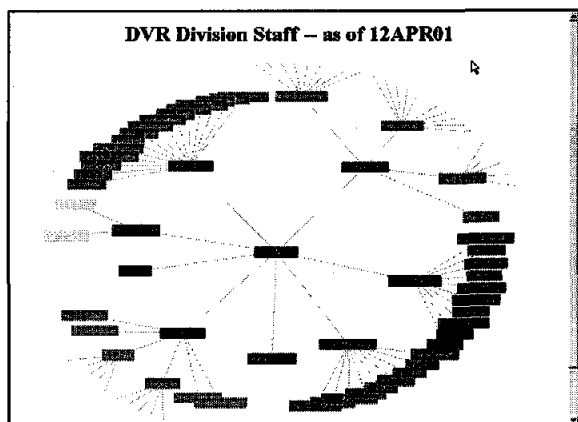
This procedure imports map information.

Java Applets Supported by SAS/Graph

Treeview Java Applet

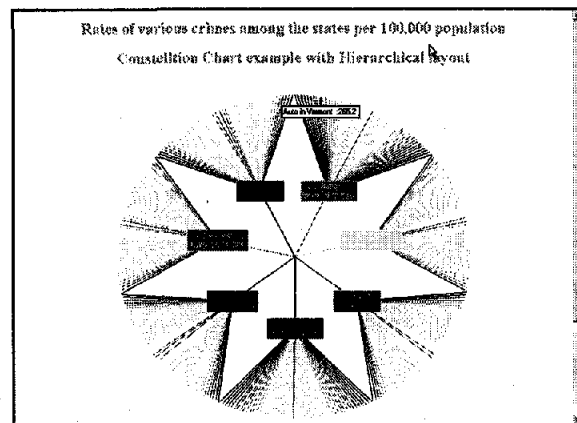
The new Treeview macro has been introduced to create a Java Applet for an interactive 'tree' style report. This creates a JAR file that is publishable via the web. The following code is a snapshot of the beginning of the macro.

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.



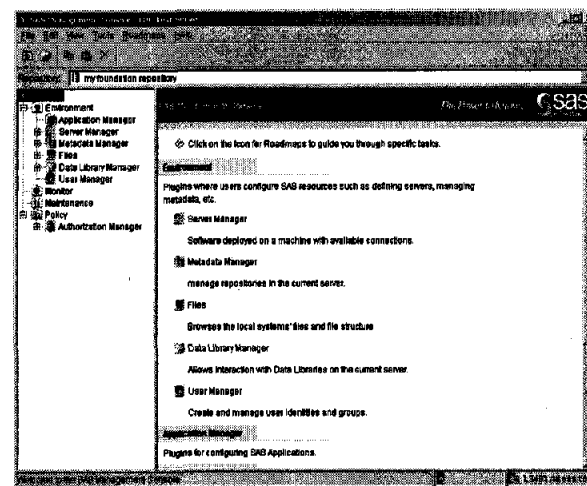
SAS Management Console

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



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.