

Configuration Guide for SAS[®] 9.2 Foundation for UNIX[®] Environments



Copyright Notice

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *Configuration Guide for SAS® 9.2 Foundation for UNIX® Environments*, Cary, NC: SAS Institute Inc., 2009.

Configuration Guide for SAS® 9.2 Foundation for UNIX® Environments

Copyright © 2009, SAS Institute Inc., Cary, NC, USA.

Some software included in SAS Foundation may display a release number other than 9.2.

All rights reserved. Printed in the United States of America. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, by any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc. Limited permission is granted to store the copyrighted material in your system and display it on terminals, print only the number of copies required for use by those persons responsible for installing and supporting the SAS programming and licensed programs for which this material has been provided, and to modify the material to meet specific installation requirements. The SAS Institute copyright notice must appear on all printed versions of this material or extracts thereof and on the display medium when the material is displayed. Permission is not granted to reproduce or distribute the material except as stated above.

U.S. Government Restricted Rights Notice. Use, duplication, or disclosure of the software by the government is subject to restrictions as set forth in FAR 52.227-19 Commercial Computer Software-Restricted Rights (June 1987).

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries.

® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

Table of Contents

Chapter 1 – Introduction	1
Audience	1
Understanding This Book.....	1
Contacting SAS	1
Accessing Release Documentation	2
Chapter 2 – Restricted Options	3
Global Restrictions	3
Group Restrictions.....	3
User Restrictions	3
Additional information.....	3
Chapter 3 – Post-Installation Configuration for User Authentication and Identification.....	5
Overview	5
Configuring User Authentication	7
Method 1: Using SAS Setup.....	7
Method 2: Using the Command Line	7
Configuring sasauth	7
Using the sasauth LDAP Authentication Method	9
Configuring the sasauth LDAP Authentication Method	10
Configuring PAM Authentication for Use with sasauth.....	11
AIX: Using System LDAP Authentication with sasauth	12
Solaris: LDAP and Numeric User Names	12
Customizing Authentication and Identification.....	13
Chapter 4 - Post-Installation Configuration for Remote Browsing.....	15
Configuring a Host With a Fully Qualified Domain Name	16
Chapter 5 – Supporting 64KB pages on AIX Machines	17
Chapter 6 – Post-Installation Configuration for National Language Support (NLS)	19
Chinese, Japanese, and Korean DBCS Support.....	19
Changing the Default DBCSLANG and DBCSTYPE Option Settings	19
Changing the Configuration File for Unicode Server	20
Setting System Fonts with X Resource Files.....	20
Asian Font Catalogs	21
Specifying the Font Catalog in the Configuration File for Traditional Chinese Fonts	21
Specifying the Font Catalog in a SAS Session for Traditional Chinese Fonts	21
European Language Support	22
Configuring Your System for Locale	22
Changing the Default LOCALE Option Setting.....	22
Running SAS in a Different Locale.....	22
Additional Information	23
Locale Setup on the Remote Server	23
Devmaps and Keymaps for SAS/GRAPH Software.....	24

Chapter 7 – Configuring SAS Analytics Accelerator for Teradata.....	27
SAS Code Files.....	27
Software Components	27
Usage	27
Chapter 8 – Post-Installation Configuration for SAS/ACCESS Software	29
SAS/ACCESS Interface to Aster nCluster Software.....	29
Registering SAS/ACCESS Interface to Aster nCluster	29
Installing and Configuring the ODBC Driver and Bulk Loader.....	29
SAS/ACCESS Interface to DB2 Software.....	30
SAS/ACCESS Interface to Greenplum Software	31
Bulkload.....	34
Registering SAS/ACCESS Interface to Greenplum.....	34
SAS/ACCESS Interface to HP Neoview Software	34
Additional Environment Variables for JNI Transporter on HP-UX for the Itanium Processor Family Architecture	35
SAS/ACCESS Interface to Informix Software	35
SAS/ACCESS Interface to Microsoft SQL Server Software	36
SAS/ACCESS Interface to MySQL Software.....	38
Setting Up the Symbolic Link	38
Setting the Shared Libraries	39
SAS/ACCESS Interface to Netezza Software.....	41
Setting Up the Symbolic Link	41
Setting the Shared Library Path	41
SAS/ACCESS Interface to ODBC Software	42
SAS/ACCESS Interface to Oracle Software	43
Setting Up the Symbolic Link	43
Setting the ORACLE_HOME Variable	43
SAS/ACCESS Interface to R/3 Software	44
SAS/ACCESS Interface to Sybase Software	44
Setting Up the Symbolic Link	44
Installing Sybase Procedure.....	44
Adding Shared Libraries	44
SAS/ACCESS Interface to Sybase IQ Software	45
Registering SAS/ACCESS Interface to Sybase IQ	45
SAS/ACCESS Interface to Teradata Software	45
Access to Shared Libraries.....	45
TTU 8.2 and HP-UX	46
FastExporting	46
MultiLoad	47
Teradata Parallel Transporter.....	47
Chapter 9 – Post-Installation Configuration for SAS/ASSIST Software	49
Adding a Master Profile	49
Chapter 10 – Post-Installation Configuration for SAS/CONNECT Software	51
Storing and Locating SAS/CONNECT Script Files	51
Chapter 11 – Post-Installation Configuration for SAS/GRAPH Software	53
Loading SAS Fonts to Your X Display Server.....	53
Making System Fonts Available to SAS	53

Chapter 12 – Post-Installation Configuration for SAS/IntrNet Software	55
Overview	55
Installing and Configuring SAS/IntrNet Software	56
Install Your Web Server Software	56
Install Your SAS Software	56
CGI Tools Installation Dialogs	56
Installing CGI Tools and SAS Foundation on Machines with Different Operating Systems	58
Test the Web Server	59
Test the Application Broker	59
Configure a Socket Service	60
Starting the Socket Service	60
Testing the Socket Service	60
Configure Additional Services	62
Chapter 13 – Post-Installation Configuration for SAS/SECURE Software	63
SAS/SECURE Client for Windows	63
SAS/SECURE Client for Java	63
Chapter 14 – Post-Installation Configuration for SAS/SHARE Software	65
User Authentication	65
System Configuration for the TCP/IP Communications Method	65
Client Components	65
SAS/SHARE Data Provider	65
SAS ODBC Driver	65
SAS/SHARE Driver for JDBC	66
SAS/SHARE SQL Library for C	66
NLS Information	66
Chapter 15 – Using Host Sort Routines	67
Making Host Sort Routines Available	67
For AIX	67
For Linux and Solaris	67
For HP-UX	68
Using Host Sort Routines in a SAS Session	68

Chapter 1 – Introduction

Audience

This document is intended for the SAS Installation Representative, designated as the person responsible for installing and maintaining SAS software for UNIX systems at your site.

This document describes the configuration instructions for SAS 9.2 Foundation, which is made up of server-side Base SAS and a variety of server-side SAS products (the exact products vary by customer). Information about the configuration of mid-tier and client-side products is available from other sources including the SAS Deployment Wizard and any documentation that it might point you to.

The server-side configuration instructions contained in this document are for the configuration of a generic SAS server. If you wish to configure your server for more specific functions, such as a Workspace Server or Stored Process Server, please refer to the *SAS 9.2 Intelligence Platform: Application Server Administration Guide* located at

<http://support.sas.com/documentation/configuration/index.html>. If you wish to configure your server as an OLAP Server, please refer also to *SAS 9.2 Intelligence Platform: Application Server Administration Guide*, at the same location. If you wish to configure your server as a Metadata Server, please refer to the *SAS 9.2 Intelligence Platform: System Administration Guide*, also at the same location.

Understanding This Book

This document conforms to the following conventions:

Courier	Courier type indicates commands, directory paths, file names, menu items, Internet addresses, etc.
<i>Italics</i>	Italic type indicates documentation references or key notes.
Bold	Bold type indicates important text or concepts.
UPPERCASE	Uppercase type indicates variable and option settings.
Dollar sign \$ Pound sign #	A dollar sign \$ or pound sign # at the beginning of an example indicates a sample UNIX command line.

Contacting SAS

If you need to contact SAS, refer to the *SAS QuickStart Guide* for contact information.

Accessing Release Documentation

The latest versions of the release documentation are available from the Install Center Web page, <http://support.sas.com/installcenter>.

Chapter 2 – Restricted Options

SAS 9.2 Foundation options can be "restricted" by a site administrator so that once they are set by the administrator; they may not be changed by a user. An option can be restricted globally, by group, and by user. To restrict an option it must be added to the appropriate SAS 9.2 Foundation configuration file and this file must have the permissions set by the administrator so that it cannot be updated by users. The option files are processed in the following order: global, group and user. If an option is specified in multiple files then the last occurrence gets used.

Global Restrictions

Create the file !SASROOT/misc/rstropts/rsasv9.cfg and add options to this file in the normal config file format.

Group Restrictions

Create a file of the following format:

```
!SASROOT/misc/rstropts/groups/<groupname>_rsasv9.cfg
```

and add options to this file in the normal config file format.

Example: For user smith in the group staff: the file name would be staff_rsasv9.cfg.

User Restrictions

Create a file of the following format:

```
!SASROOT/misc/rstropts/users/<user ID>_rsasv9.cfg
```

and add options to this file in the normal config file format.

Example:

For user smith, the file name is smith_rsasv9.cfg.

Additional information

To verify that an option has been set correctly follow this example:

1. Assume the option -EMAILSYS=SMTP was specified in one of the restricted configuration files.
2. Submit the following code:

```
proc options restrict; run;
```

The SAS log should then show a message similar to

```
Option Value Information For SAS Option EMAILSYS
Option Value: SMTP
Option Scope: SAS Session
How option value set: Site Administrator Restricted
```

The following describes the process when a user attempts to change the option value.

1. Assume the option -NOTHEADS was specified in one of the restricted configuration files.

2. Submit the following code:

```
options THREADS;
```

The SAS log should then show a message similar to

```
options THREADS;
-----
      36
WARNING 36-12: SAS option THREADS is restricted by your Site
Administrator and cannot be updated.
```

Note: Only one Group Restrictions File will be read during SAS processing. The effective groupid of the SAS process that is running is used in the determination of which Group Restrictions File to use.

Note: If the effective user ID of the SAS process that is running does not have a corresponding entry in the `/etc/passwd` file, then only the global restricted option and the group restricted options files will be read.

Note: If the effective groupid of the SAS process that is running does not have a corresponding entry in the `/etc/group` file, then only the global restricted option and the user restricted options files will be read.

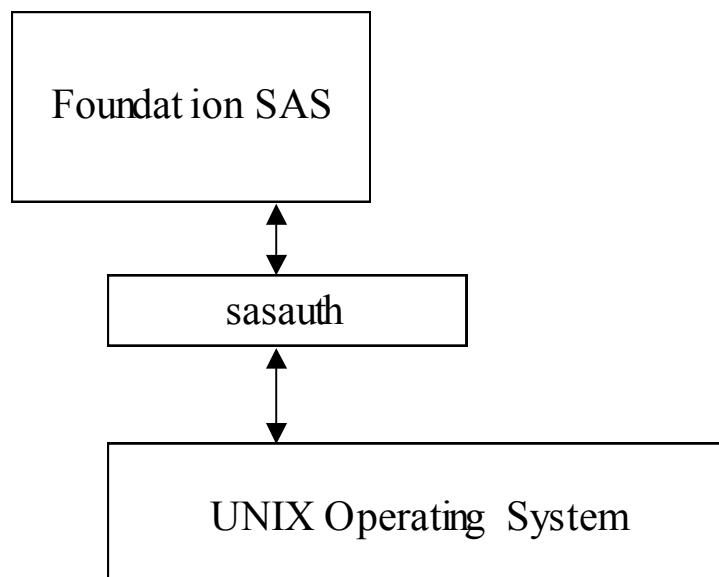
Chapter 3 – Post-Installation Configuration for User Authentication and Identification

Overview

UNIX user security is more than just authentication. User identification is also performed when user credentials are validated. Unlike Windows, UNIX uses an integer value, called the UID, to identify users. Ownership of system resources is then assigned by associating a particular UID with a system resource. User identification determines the UID for a particular user name.

When user credentials are validated, UNIX systems will search a user database for an entry that contains the same user name. Traditionally, the user database is plain file in the file system, but newer security environments may store this in a binary database, or on a server on the network. Most UNIX systems support several other storage methods than the traditional file. Once the user entry is found, the password can be retrieved and matched against an encrypted version of the user provided password (authentication), and the UID is retrieved for the user (identification).

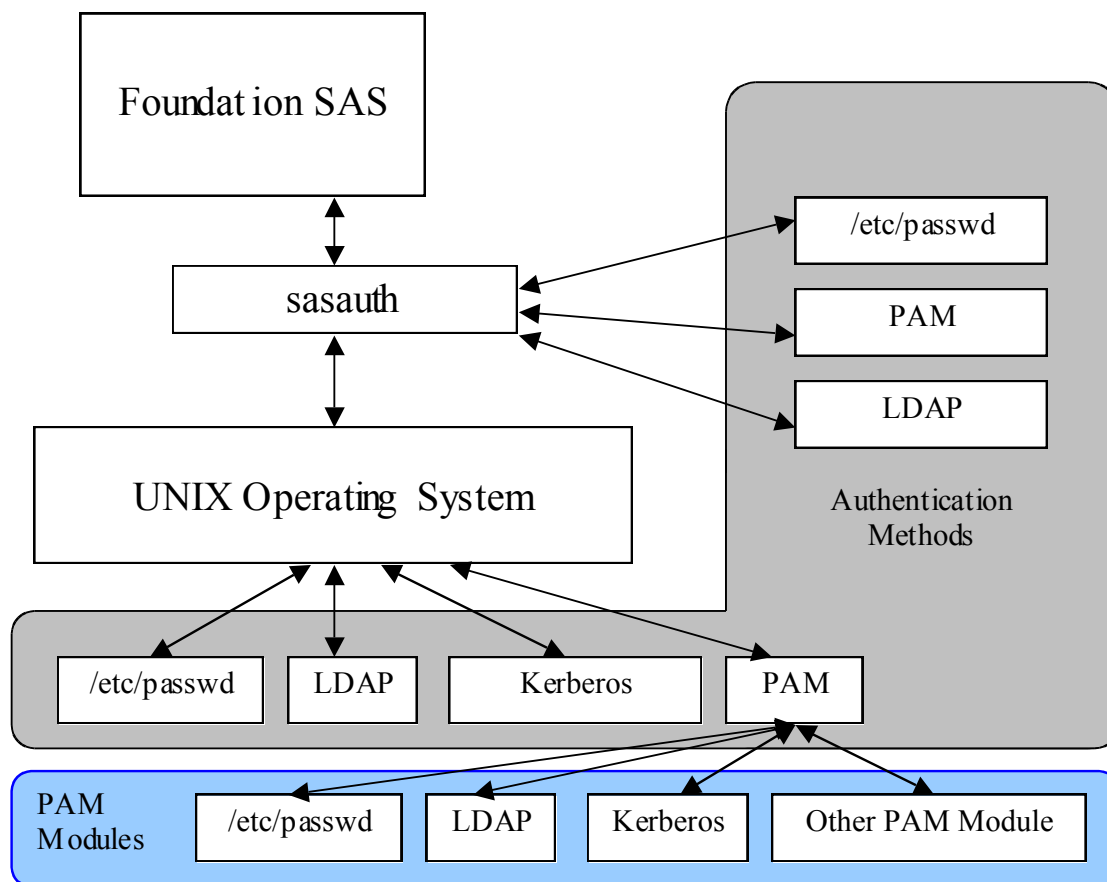
SAS for UNIX systems validates users in the same manner. The user name provides an index into a user database, from which the user is identified and authenticated. Usually superuser permissions are required to read the user database. Since running the entire SAS process with superuser permissions is undesirable – users would have access to files they don't own – an external utility, named `sasauth` (found in `!SASROOT/utilities/bin`), is used to perform authentication. The `sasauth` process runs `setuid` to root, so that it has the appropriate permissions to access the user database.



Authentication databases can be stored in several places. The traditional form is as a text file, `/etc/passwd`, with encrypted passwords stored in `/etc/shadow`. Newer forms utilize client/server architecture to provide network-wide authentication, as in NIS+ and LDAP.

For each of these forms, the operating system or application that performs user credential validation must implement the necessary functionality to access the database. Since each form has a different application interface, it is very difficult to support all authentication forms. PAM, or pluggable authentication modules, is a standard library for performing user authentication (but not identification). PAM uses “modules”, or libraries, to access multiple authentication forms. A system administrator can select the appropriate authentication based on security requirements. Most UNIX systems support PAM in addition to the native operating system authentication.

The following figure shows possible authentication flows.



SAS strongly recommends that the base operating system be configured to use the required authentication/identification form that matches local requirements. For example, if the SAS server is installed at a site where there is a central LDAP repository, the operating system should be configured as an LDAP client for the central repository.

Many sites like to use PAM since it is a widely accepted authentication mechanism and is very flexible. Modules can be obtained for custom authentication mechanisms, such as smart cards, and added to the system without direct application support. But PAM's lack of user identification

is problematic for use with sasauth. The PAM programming libraries will only authenticate a user/password combination. The UID, which is needed by SAS, is not returned. Therefore sasauth will use the standard UNIX authentication calls to obtain the UID, meaning that the system must also be configured to access the same user information as PAM. If you find that your site needs to use PAM for authentication, configuration instructions are provided in subsequent sections of this document.

Configuring User Authentication

Certain SAS products and features employ functionality that require SAS to check user ID authentication and file access authorizations. This in turn necessitates that certain files within your SAS installation have setuid permissions and be owned by root. Configuring user authentication is required for all users of SAS software. You can perform this task using either of the following methods.

Method 1: Using SAS Setup

1. Log in to the root account.
`$ su root`
2. Run SAS Setup from `!SASROOT/sassetup`.
3. Select Run Setup Utilities from the SAS Setup Primary Menu.
4. Select Perform SAS Software Configuration.
5. Select Configure User Authentication.
6. After user authentication has been completed, type `q` to exit the SAS Software Configuration Menu.

Method 2: Using the Command Line

```
$ su root
# cd !SASROOT/utilities/bin
# mv setuid/* .
# chown root elssrv sasauth sasperm
# chmod 4755 elssrv sasauth sasperm
# exit
```

Configuring sasauth

sasauth natively supports system authentication (such as `/etc/passwd`), LDAP repositories, and PAM authentication. It also provides three levels of logging, and user retry lockout, where a user will not be allowed to authenticate for a certain period of time after a certain number of invalid authentication attempts are made. All of these features are configured via a text file, `!SASROOT/utilities/bin/sasauth.conf`.

The sasauth configuration file is a text file consisting of name/value pairs for configuring behavior, one per line. Names and values are case sensitive. A “#” character is used for comments, which extend to the end of the line.

Supported names and values are listed below.

Name: methods

The methods setting specifies what user validation methods should be used. At least one should be specified, though multiple values may be specified, separated by spaces. Authentication then follows the listed methods in order from left to right, which each method attempted until the user identity is found.

Values	Usage
pw	Use system authentication, typically <code>/etc/passwd</code> - <code>/etc/shadow</code> authentication. On some hosts, this also includes protected password databases or OS-provided enhanced security.
pam	Use PAM for authentication. The operating system's user security functions are also used to determine the user's UID and GID. PAM must be configured properly for sasauth, as described in "Configuring PAM Authentication for Use with sasauth" below.
ldap	Use LDAP queries for authentication. See "Using the sasauth LDAP Authentication Method" below.
ext	Use a custom authentication mechanism. This mechanism is built using the sasauth customization kit, which can be found in <code>!SASROOT/utilities/src/auth</code> .

Name: debugLog**Name: accessLog****Name: errorLog**

These settings specify the pathnames for the sasauth logs. sasauth provides three logs:

- error log – Contains error messages.
- access log – Contains transaction information for each user validation request: user name, validation method used, and validation result.
- debug log – Contains verbose debugging information. Useful when troubleshooting initial configuration.

The value should contain the path for the log file. Log files whose paths are unspecified are not generated, with the exception of the error log, which is sent to syslog instead.

For example:

```
#debugLog=
accessLog=/tmp/sasauth.log
#errorLog=
```

will configure sasauth to have no debug log, use `/tmp/sasauth.log` as the access log, and syslog for the error log.

Note: You may need to configure your system's syslog facility to see sasauth messages. Refer to your system documentation for details.

Name: logOwner

Specifies the numeric UID of the owner of the sasauth log files. Defaults to root since sasauth runs as root. Use this setting to allow a user other than root to read the sasauth log files.

Name: debugNoPasswords

When set to "true", passwords will not be written to the log file. Defaults to true.

Name: maxtries**Name: maxtriesPeriod****Name: maxtriesWait**

These settings configure sasauth's maxtries configuration. sasauth will not authenticate a user after a maximum number of attempts are made in a given period of time. The user must then wait for a given wait time before additional authentication requests are validated. When maxtries is activated, information about maxtries failures is logged to the access log. The setting maxtries is the maximum number of attempts that may be made. maxtriesPeriod specifies the number of seconds after which repeated attempts exceeding the maxtries count are not authenticated. maxtriesWait specifies the number seconds the user must wait before the maxtries count is reset and validation requests are then permitted.

For example, these settings:

```
maxtries=5
maxtriesPeriod=60
maxtriesWait=300
```

will cause sasauth to stop authenticating a user for 5 minutes if 5 invalid attempts are made in 1 minute.

To turn off maxtries, remove all three settings from the configuration file by commenting them out.

Using the sasauth LDAP Authentication Method

The sasauth LDAP authentication method ("method=ldap" in the configuration file) provides a direct connection from sasauth to an LDAP database for authentication. sasauth will query user attributes from the database and then authenticate the user based on the returned attributes. sasauth will also query the LDAP database to determine secondary group attributes for the user being authenticated.

Note: *sasauth does not support encrypted communications with the LDAP server. User information will be transmitted in the clear over the network. If your site requires encrypted communications, you must use the operating system's LDAP client by configuring sasauth with "module=pw" and then configuring your system for LDAP-based authentication.*

LDAP repositories that are used for UNIX authentication (including sasauth) must include UNIX/Posix user attributes (such as UID) in the database. Without this information, the LDAP database cannot be used with UNIX. Most LDAP servers provide an LDAP schema that contains this information. Microsoft Active Directory repositories should have Microsoft Services for UNIX (SFU) 2 or 3 installed. Other LDAP databases should conform to the RFC 2307 standard for

including UNIX user attributes in an LDAP database. sasauth requires the following user attributes, listed using their RFC 2307 names:

- uid - user name
- uidnumber - numeric UID
- gidnumber - numeric group number of the user's primary group
- userpassword - encrypted form of user's password. sasauth supports crypt, SHA, and SSHA forms.
- shadowLastChange - date of last password change
- shadowMax - Maximum age of password before change is required
- shadowExpire - Expiration date of account.

Note that password expiration will not be processed by sasauth if the password expiration attributes are not found in the database.

sasauth also requires the following group attributes, listed using their RFC 2307 names:

- group - group name
- gidNumber - numeric ID of the group
- memberUid - user name that is in the group

The memberUid attribute is repeated for each member of the group.

Configuring the sasauth LDAP Authentication Method

Once the LDAP method is added to the list of authentication methods for sasauth (see "Name: methods" above), additional settings will need to be configured for LDAP in sasauth.conf. The names and values are listed below.

Name: LDAP_HOST

Name: LDAP_PORT

Host name and port number of the LDAP server. LDAP_PORT can be omitted, in which case sasauth will use the standard LDAP port number.

Name: LDAP_AUTH_METHOD

Name: LDAP_HOST_DN

Name: LDAP_HOST_PW

sasauth will authenticate user credentials by using bind or match. For bind, sasauth will bind to the server with the user's credentials. If the bind fails, the user is not authenticated. By binding to the server using the user's credentials, the LDAP server does all of the authentication (including applying security rules not supported by sasauth), but sasauth cannot determine the specific cause of failed logins. Users will not know why authentication failed when using bind for authentication.

To use bind authentication, set LDAP_AUTH_METHOD to the value BIND (case-sensitive) in the configuration file.

For match, the user's encrypted password and expiration information are queried from the database and matched with the provided credentials. A mismatch or expiration causes the authentication to fail.

To use match authentication, set `LDAP_AUTH_METHOD` to the value `MATCH`, and set `LDAP_HOST_DN` and `LDAP_HOST_PW` to the user and password for an admin user. An admin user is required because LDAP will not return the encrypted password to a non-administrative user. Since the `sasauth.conf` file will now contain password information, make sure that it is readable only by root (for example, run `chmod 400 sasauth.conf` from the shell).

Name: LDAP_SEARCHBASE

Name: LDAP_USERBASE

These settings provide the search criteria used by `sasauth` when constructing queries to retrieve user identification. For example:

```
LDAP_SEARCHBASE="DC=MYGROUP, DC=MYCOMPANY, DC=COM"
LDAP_USERBASE="ou=People"
```

Set to values appropriate for your organization. Your LDAP administrator can assist with determining these values.

Name: LDAP_SCHEMA

Specifies which schema the server uses. Select from:

- `LDAP_SCHEMA=RFC2307` - for RFC 2307 (e.g. Sun ONE Directory Server),
- `LDAP_SCHEMA=AD2` - for Active Directory with Services for UNIX (SFU) 2
- `LDAP_SCHEMA=AD3` - for Active Directory with Services for UNIX (SFU) 3
- `LDAP_SCHEMA=OTHER` - for a manual configuration. Follow instructions in the config file when using this value.

Configuring PAM Authentication for Use with `sasauth`

PAM is architected such that applications must be registered in order to use authentication services. For `sasauth` to perform authentication, entries must be made in the PAM configuration that describe what authentication services are used when `sasauth` performs an authentication, specifically the “account” and “auth” module types.

Note: *PAM allows configuration of “other,” which permits any application to use authentication services. This is not recommended.*

PAM supports applications that run in both 32-bit and 64-bit environments. Modules used with `sasauth` must match the binary format of the `sasauth` program. For SAS 9.2 on UNIX platforms, `sasauth` is a 64-bit binary, and PAM modules must be 64-bit libraries. The standard system modules are usually provided in both 32-bit and 64-bit versions, with each set stored in a separate directory. `pam.conf` then contains pathnames that are either relative (Solaris and AIX) or contain a symbolic variable (HP-UX) that allows the correct format to be loaded depending on the format of `sasauth`.

On HP-UX, Solaris, and AIX systems, the PAM configuration is stored in `/etc/pam.conf`. For `sasauth` authentication to succeed, entries should be added of the following form:

```
<service_name> <module_type> <control_flag> <module_path> <options>
```

For example, these entries enable `sasauth` to authenticate on Solaris:

```
sasauth auth requisite          pam_authtok_get.so.1
```

```

sasauth auth required      pam_dhkeys.so.1
sasauth auth required      pam_unix_auth.so.1
sasauth account required   pam_unix_account.so.1

```

And for HP/UX:

```

Sasauth account required   /usr/lib/security/$ISA/libpam_unix.so.1
Sasauth auth required      /usr/lib/security/$ISA/libpam_unix.so.1

```

If the system uses an authentication service other than the UNIX password files (such as LDAP or Kerberos), then the entries will have to define what service to use. The manual page for `/etc/pam.d` will help determine these entries.

On Solaris, if LDAP is being used, PAM should also be configured to communicate with the directory server via the `ldapclient(1m)` command. Refer to the `ldapclient` man page for more information.

Note: *AIX systems do not ship with PAM activated. Refer to the IBM document Security Guide – Authentication Module (http://www16.boulder.ibm.com/pseries/en_US/aixbman/security/pam_overview.htm) for instructions on activating PAM on AIX.*

On Linux systems, the directory `/etc/pam.d` contains a file for each program authorized to use PAM. The name of the configuration matches the name of the process making authentication requests. For `sasauth`, the configuration file is `/etc/pam.d/sasauth`.

The configuration file contains entries in the following form:

```
<module_type> <control_flag> <module_path> <options>
```

For example, `/etc/pam.d/sasauth` may contain:

```

#%PAM-1.0
auth      required      pam_unix2.so      nullok
account   required      pam_unix_acct.so

```

Note: *In the SAS Intelligence Platform, PAM is an optional configuration that is useful only in certain circumstances. For guidance and alternatives, see the discussion of authentication in the SAS Intelligence Platform: Security Administration Guide.*

AIX: Using System LDAP Authentication with sasauth

IBM does not provide an LDAP module for PAM. The open source package OpenLDAP can be used to build an LDAP module, but this is not recommended for production environments since it is not a solution supported by IBM. Instead, sites that need LDAP authentication should configure the AIX system for LDAP authentication. Refer to the IBM Redbook *Integrating AIX into Heterogeneous LDAP Environments* for instructions on how to configure AIX as an LDAP client.

Solaris: LDAP and Numeric User Names

The Solaris LDAP client does not treat numeric user names as user names. Instead, Solaris assumes that a user name that is numeric is actually a UID, and converts the user name directly to the UID instead of querying the LDAP database. Since Solaris recommends that user names

begin with an alphabetic character, this is unlikely to change. If your site uses Solaris as an LDAP client, then user names in LDAP cannot be numeric.

Customizing Authentication and Identification

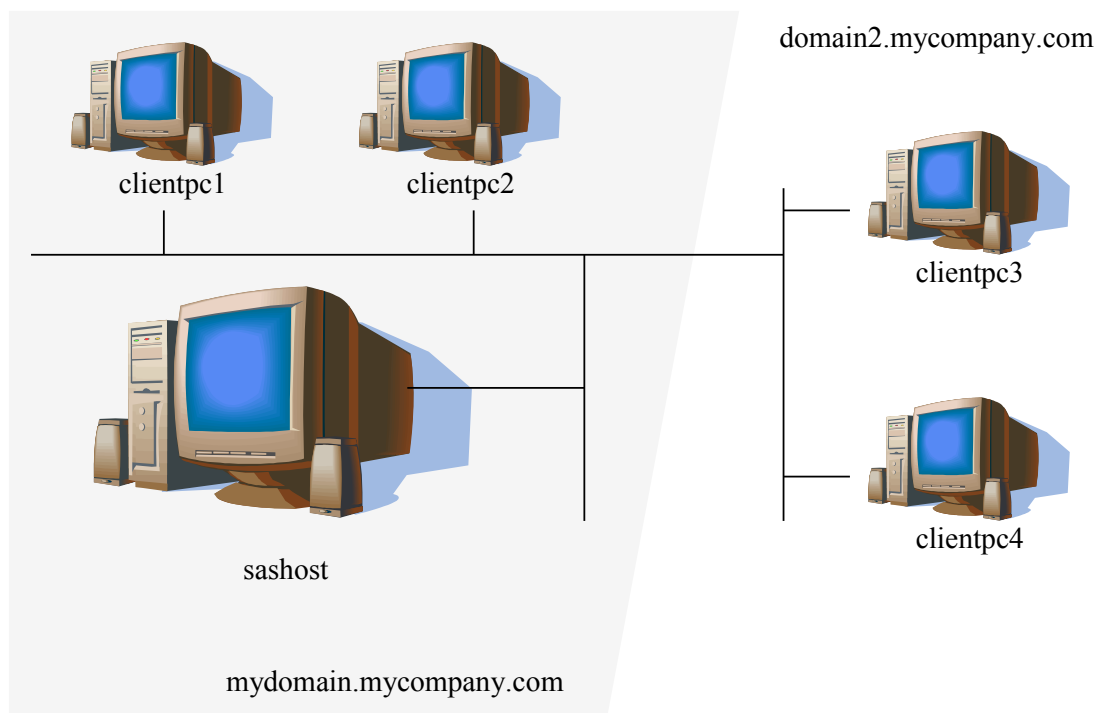
sasauth can be configured to perform authentication in a site-specific manner. The SAS Foundation installation includes the *UNIX Authentication API*, a package for developing site-specific authentication and identification.

The files and documentation are installed in !SASROOT/utilities/src/auth. Refer to the file docs.pdf in that directory for detailed development instructions.

Chapter 4 - Post-Installation Configuration for Remote Browsing

The SAS host may need to be configured appropriately for remote browsing. If one or more SAS desktop clients resides outside the DNS domain of the SAS host, then the host must be configured with a hostname that contains the fully qualified domain name (FQDN) of the host.

For example, SAS is installed on the host `sashost.mycompany.com`, two client computers exist in the same domain (`client1.mycompany.com` and `client2.mycompany.com`), and two other clients exist in another domain (`client3.domain2.mycompany.com` and `client4.domain2.mycompany.com`). This relationship is illustrated below.



If the system `sas.mycompany.com` is not configured with a hostname that is the FQDN for the system, then `client1` and `client2` will be able to view HTML content from SAS, but `client3` and `client4` will not. This is because URLs generated for the SAS host will not include the domain, as in `http://sashost:12345/output.html`.

Since `client1` and `client2` are in the same domain as SAS itself, their browser will build valid hostnames from their domain, `sashost.mycompany.com`. But `client3` and `client4`, which are outside the domain of the SAS host, will use their domain names to construct a complete hostname, which results in the invalid name `sashost.domain2.mycompany.com`.

By configuring the SAS host with the system's FQDN, URLs for HTML display are valid from all of clients. From the example, the valid URL for all clients is `http://sashost.mycompany.com:12345/output.html`.

Configuring a Host With a Fully Qualified Domain Name

Note: Superuser privileges are required to make this change.

1. Edit `/etc/hosts`.
2. For the IP address of network interfaces for the host, add the FQDN as the first name in the list. For example (using IPv4 addresses):

```
10.4.86.62          sashost
becomes
10.4.86.62          sashost.mycompany.com sashost
```

Chapter 5 – Supporting 64KB pages on AIX Machines

IBM pSeries servers running AIX 5.3 now support 64KB pages as well as 4KB pages. For SAS executables to take advantage of 64KB pages, you should set and export the environment variables using the following commands:

```
$ LDR_CNTRL="DATAPSIZE=64K@TEXTPSIZE=64K@STACKPSIZE=64K@$LDR_CNTRL"  
$ export LDR_CNTRL
```

Using 64KB pages rather than 4KB pages for a multi-threaded process's data may reduce the maximum number of threads a process can create due to alignment requirements for stack guard pages. Applications that encounter this limit may disable stack guard pages by setting the environment variable `AIXTHREAD_GUARDPAGES` to 0. (Note that this is really only a problem for 32-bit applications that create many threads, because of the 256M segment address limit in PPC 32-bit mode. Real memory is not allocated for guard pages. This is not a problem for 64-bit programs like SAS 9.2.) Use the following commands to set the `AIXTHREAD_GUARDPAGES` variable correctly (note that this setting is not needed for 64-bit programs).

```
$ AIXTHREAD_GUARDPAGES=0  
$ export AIXTHREAD_GUARDPAGES
```


Chapter 6 – Post-Installation Configuration for National Language Support (NLS)

This chapter contains information on post-installation configuration for Asian and European language support.

Important: Before invoking a localized SAS 9.2 Foundation image from a UNIX shell, you must ensure that the UNIX locale environment variable `LANG` is set appropriately for the language of the SAS version you want to run. The exact values to set will vary depending on your operating system support. To list the locales supported on your operating system, enter the following command:

```
$ locale -a
```

For example, to invoke a Japanese version of SAS 9.2 Foundation in the HP-UX Korn shell environment, enter the following command:

```
$ LANG=ja_JP.SJIS; export LANG
```

For more information on setting locale environment variables, consult the documentation for your operating system.

Chinese, Japanese, and Korean DBCS Support

This section explains how to

- change the default settings for the `DBCSLANG` and `DBCSTYPE` system options
- specify Asian font catalogs.

Note: *The `DBCSLANG` and `DBCSTYPE` system options described in the next section should be used to set the DBCS encoding for Asian character sets only. The `LOCALE` and `ENCODING` system options described in the SAS Help System are used to set locale for European languages.*

Also, be aware that full-screen products are NOT supported in 9.2 SAS for the following UNIX platforms and languages:

- HP-UX IPF: Japanese, Korean, Simplified Chinese, and Traditional Chinese
- AIX: Korean, Simplified Chinese, and Traditional Chinese

Changing the Default `DBCSLANG` and `DBCSTYPE` Option Settings

When you install SAS 9.2 Foundation and choose to load NLS language translations, the installation automatically sets default values for the `DBCSLANG` and `DBCSTYPE` system options based on the language selection and platform. For example, if you install Primary Japanese on the Solaris operating system, the configuration file (`!SASROOT/nls/ja/sasv9.cfg`) sets `DBCSLANG` to `JAPANESE` and `DBCSTYPE` to `EUC`.

If you need to change the default settings, edit the configuration file. For example, edit the configuration file to change the `DBCSTYPE` value to `SJIS`.

Changing the Configuration File for Unicode Server

To run the Unicode Server, you need to edit the configuration file for your system with the following changes:

1. Remove the DBCSLANG and the DBCSTYPE options from the configuration file.
2. Add the ENCODING option and set the value to UTF8 (ENCODING=UTF-8).
3. To define a default locale other than English, add the LOCALE option and set the value to your desired locale setting (LOCALE=default-locale).

For more information, please see the white paper on "Processing Multilingual Data with the SAS 9.2 Unicode Server".

Setting System Fonts with X Resource Files

SAS 9.2 Foundation may not have the correct font settings for your locale by default. To ensure that the correct fonts are defined for the SAS System, you must add them to your X Resource files.

Japanese X Resource template files containing DBCS font settings are located in !SASROOT/X11/resource_files, as follows:

- ./Resource_CDE.ja - for the CDE environment
- ./Resource_LNX.ja - for Linux
- ./Resource_Sun.ja - for Solaris
- ./Resource_DEC.ja - for Tru64 UNIX
- ./Resource_HP.ja - for HP-UX
- ./Resource_IBM.ja - for AIX
- ./Resource_ReflX.ja - for ReflectionX users

Simplified Chinese X Resource template files containing DBCS font settings are located in !SASROOT/X11/resource_files, as follows:

- ./Resource_HP.zh - for HP-UX
- ./Resource_LNX.zh - for Linux
- ./Resource_Sun.zh - for Solaris

Traditional Chinese X Resource template files containing DBCS font settings are located in !SASROOT/X11/resource_files, as follows:

- ./Resource_HP.zt - for HP-UX
- ./Resource_HP.zt.euc - for HP-UX
- ./Resource_LNX.zt - for Linux
- ./Resource_Sun.zt - for Solaris
- ./Resource_Sun.zt.big5 - for Solaris

Korean X Resource template files containing DBCS font settings are located in `!SASROOT/X11/resource_files`, as follows:

- `./Resource_HP.ko` - for HP-UX
- `./Resource_LNX.ko` - for Linux
- `./Resource_Sun.ko` - for Solaris

To apply the X Resources in these template files, copy the appropriate template to one of the following locations, renaming it to `SAS` (in all uppercase):

- `/usr/lib/X11/app-defaults` (on most UNIX systems)
- `/usr/openwin/lib/X11/app-defaults` (on Solaris)
- `$HOME` (your home directory)

For example, on a Solaris system, you would use the following `COPY` command:

```
$ cp !SASROOT/X11/resource_files/Resource_CDE.ja /usr/openwin/lib/X11/app-defaults/SAS
```

In the example, `!SASROOT` refers to the root directory of your SAS 9.2 Foundation installation.

For more details, refer to the SAS 9.2 National Language Support (NLS) User's Guide.

Asian Font Catalogs

For SAS 9.2, Simplified and Traditional Chinese have been added to `SASHELP.FONTS`.

Specifying the Font Catalog in the Configuration File for Traditional Chinese Fonts

When you run a Traditional Chinese localization, the configuration file contains the `GFONT` definition for the location of the ZT font catalog in the UNIX DBCS directory. However, when you run the English version with `DBCSLANG=TAIWANESE`, you must either set `GFONT` in your SAS session or you must modify the DBCS configuration file to define the `GFONT` definition for the ZT catalog, as follows

```
-set gfontx !SASROOT/nls/zt/font-name
```

In this statement

`x` represents a value from 0-9

`font-name` represents the name of the font catalog you want to use.

Specifying the Font Catalog in a SAS Session for Traditional Chinese Fonts

To specify the font catalog in a SAS session, submit the following `LIBNAME` statement:

```
libname gfontx !SASROOT/nls/zt/font-name
```

In this statement

x represents a value from 0-9

font-name represents the name of the font catalog you want to use.

European Language Support

Note: The phrase “European Language Support” refers to any language that is not Chinese, Japanese, or Korean.

The following sections explain different methods for configuring your system for locale, describe how to set up your local session to transfer data to a remote session, and provide a list of `devmap` and `keymap` values that match the locales on your operating system. (As mentioned earlier in this chapter, the `LOCALE` and `ENCODING` system options are used to set locale for European languages. These system options are documented in the SAS Help System.)

Configuring Your System for Locale

If you want to configure your SAS session for a locale other than the default locale, you have various methods from which to choose to reconfigure. This section explains those methods.

Changing the Default LOCALE Option Setting

When you install SAS 9.2 Foundation and choose to load NLS language translations, the installation automatically sets the `LOCALE` system option to the default value for the language installed. The `LOCALE` system option is set in the system configuration file for each language installed.

For example, `!SASROOT/nls/fr/sasv9.cfg` sets `LOCALE` to `French_France` by default.

Note: The English version does not set the locale in the configuration file by default.

If you want to change the default locale setting for SAS, you can set the `LOCALE` system option to the appropriate language in your system configuration file.

For example, you can edit `!SASROOT/nls/fr/sasv9.cfg` to change `-locale French_France` to `-locale French_Canada`.

Running SAS in a Different Locale

To set the locale for SAS 9.2 at your site, add the `LOCALE` system option to your configuration file. You can find a list of locale values in the *SAS 9.2 National Language Support (NLS) User's Guide*.

When you read or write a file, SAS expects the data in the external files to be in the session encoding. To specify a different encoding, refer to the documentation for the `ENCODING` system option in the `FILENAME`, `INFILE`, or `FILE` statements in the *SAS 9.2 National Language Support (NLS) User's Guide*.

When the `LOCALE` system option is set, the `ENCODING` system option will be set to an encoding that supports the language for the locale. SAS 9.2 Foundation expects user data to be in the encoding that matches the `ENCODING` option. If you prefer an encoding other than the most

common encoding for the locale, you can also set the ENCODING system option in the configuration file.

When the ENCODING option is set, the TRANTAB option will always be set to match the ENCODING system option. The transport format trantabs (translation tables), set by the TRANTAB option, are used by the CPORT and CIMPORT procedures to transfer SAS data files. These trantabs are also used by the UPLOAD and DOWNLOAD procedures for transferring files and catalogs, remotely submitting code to the server, and returning logs and listings to the client.

The Output Delivery System (ODS) will create output using the encoding that matches the ENCODING system option. If you would like your output created using a different encoding, please refer to the documentation for the Output Delivery System.

For more information, refer to the *Base SAS 9.2 Procedures Guide* for documentation about PROC CPORT and PROC CIMPORT. Refer to the *SAS/CONNECT 9.2 User's Guide* for documentation on PROC UPLOAD and PROC DOWNLOAD.

Additional Information

Depending on the applications you run, additional setup may be required for your system. Refer to the following sections for more information about configuring your system to run with alternate locales.

Locale Setup on the Remote Server

If you are running SAS System 9 as both your client and server sessions, it is not usually necessary to run the %LS () macro to do any further locale setup. The locale of a server should be compatible with the locale of your client session; otherwise, your data may be corrupted.

If your SAS System 9 client is connecting to a session running a release of SAS prior to SAS System 9, you can use the %LS () macro to set up the remote SAS environment for data transfer. As the Locale Setup Window did in previous releases, the %LS () macro copies the host-to-host translation tables from the LOCALE catalog into SASUSER.PROFILE. The %LS () macro does not set the encoding for the SAS session.

If you use SAS/CONNECT to connect to a remote SAS server, you will need to set up the server session for the locale that the SAS client is using. You must set up the server *after* signing on to the remote session from the client.

The following examples show how to set locale for remote connections:

- **Connecting SAS System 9-to-SAS System 9:** Use the LOCALE option at startup. The LOCALE option value of the SAS client and server sessions should be the same. For example,

```
sas -locale Danish_Denmark
```

- **Connecting SAS System 9 and a previous release of SAS:**
 - SAS System 9 receives the data: Use the LOCALE option on the SAS System 9 side at start up.

Example:

```
sas -locale Spanish_Spain
```

- Previous release receives the data: Start SAS System 9 with the LOCALE option at start up.

Example:

```
sas -locale Spanish_Spain
```

Then use the %LS() macro in SAS System 9 to set up the host-to-host translation tables on the previous release after connection is established. **Example** - Submit the following code from the Program Editor:

```
%ls(locale=Spanish_Spain, remote=on);
```

Note that if you are working with the UTF-8 encoding, use sas_u8 in the -locale command line option. For example,

```
sas -locale sas_u8
```

Devmaps and Keymaps for SAS/GRAPH Software

If you are running SAS/GRAPH software and you want to display non-ASCII characters, you will need to set the appropriate devmaps and keymaps to match your current encoding. The devmap and keymap entries are located in the SASHELP.FONTS catalog. To get the correct devmaps and keymaps for your encoding, you should use the %LSGRAPH macro. %LSGRAPH automatically sets up your environment for you by

- copying the devmap and keymap entries that match your encoding to the GFONT0.FONTS catalog
- changing the name of the entry to the name DEFAULT so the devmaps and keymaps will be loaded for you.

The following example uses %LSGRAPH to set the correct devmap and keymap (LAT2) for a Polish user on a UNIX platform:

```
libname gfont0 'your-font-library';
%lsgraph(LAT2);
```

Here is a list of the devmaps and keymaps that match the locales on your platform:

Locale	Devmap and Keymap Name		Locale	Devmap and Keymap Name
Arabic_Algeria	arab		German_Switzerland	lat9
Arabic_Bahrain	arab		Greek_Greece	grek
Arabic_Egypt	arab		Hebrew_Israel	hebr
Arabic_Jordan	arab		Hungarian_Hungary	lat2
Arabic_Kuwait	arab		Icelandic_Iceland	lat1
Arabic_Lebanon	arab		Italian_Italy	lat9
Arabic_Morocco	arab		Italian_Switzerland	lat9
Arabic_Oman	arab		Latvian_Latvia	lat6

Arabic_Qatar	arab	Lithuanian_Lithuania	lat6
Arabic_SaudiArabia	arab	Norwegian_Norway	lat9
Arabic_Tunisia	arab	Polish_Poland	lat2
Arabic_UnitedArabEmirates	arab	Portuguese_Brazil	lat1
Bulgarian_Bulgaria	cyr1	Portuguese_Portugal	lat9
Byelorussian_Belarus	cyr1	Romanian_Romania	lat2
Croatian_Croatia	lat2	Russian_Russia	Cyr1
Czech_CzechRepublic	lat2	Serbian_Yugoslavia	cyr1
Danish_Denmark	lat9	Slovak_Slovakia	lat2
Dutch_Belgium	lat9	Slovenian_Slovenia	lat2
Dutch_Netherlands	lat9	Spanish_Argentina	lat1
English_Australia	lat1	Spanish_Bolivia	lat1
English_Canada	lat1	Spanish_Chile	lat1
English_HongKong	lat9	Spanish_Colombia	lat1
English_India	lat9	Spanish_CostaRica	lat1
English_Ireland	lat9	Spanish_DominicanRepublic	lat1
English_Jamaica	lat1	Spanish_Ecuador	lat1
English_NewZealand	lat1	Spanish_ElSalvador	lat1
English_Singapore	lat9	Spanish_Guatemala	lat1
English_SouthAfrica	lat1	Spanish_Honduras	lat1
English_UnitedKingdom	lat9	Spanish_Mexico	lat1
English_UnitedStates	lat1	Spanish_Nicaragua	lat1
Estonian_Estonia	lat6	Spanish_Panama	lat1
Finnish_Finland	lat9	Spanish_Paraguay	lat1
French_Belgium	lat9	Spanish_Peru	lat1
French_Canada	lat1	Spanish_PuertoRico	lat1
French_France	lat9	Spanish_Spain	lat9
French_Luxembourg	lat9	Spanish_UnitedStates	lat1
French_Switzerland	lat9	Spanish_Urugay	lat1
German_Austria	lat9	Spanish_Venezuela	lat1
German_Germany	lat9	Swedish_Sweden	lat9
German_Liechtenstein	lat9	Turkish_Turkey	lat5
German_Luxembourg	lat9	Ukrainian_Ukraine	cyr1

Chapter 7 – Configuring SAS Analytics Accelerator for Teradata

This chapter describes the use of updates to the SAS 9.2m3 UDFs (User Defined Functions) for SAS/STAT and SAS Enterprise Miner. These UDF files are used to install SAS functions into the Teradata database. After they are installed, they will enable in-database processing for selected procedures in SAS/STAT 9.22 and SAS Enterprise Miner 6.2.

SAS Code Files

The following four SAS code files are included:

- `udftdem.sas` SAS Enterprise Miner functions
- `udftdets.sas` SAS ETS functions
- `udftdins.sas` Installation functions
- `udftdstt.sas` SAS STAT functions

These files can be executed on any SAS 9.2m3 supported platform except z/OS.

Software Components

The following software components are needed:

- Base SAS 9.2m3
- SAS Analytics Accelerator for Teradata 1.3 (Limited Availability)
- SAS Enterprise Miner 6.2
- SAS/ACCESS Interface to Teradata
- SAS/STAT 9.22
- Teradata 13.0 with updates

Usage

Follow these steps to install SAS functions into the Teradata Data Base:

Note: *The SAS system and user must have credentials for administrator access to the Teradata 13.0 server.*

1. Copy the four SAS code files to your default SAS user directory on your local SAS system. For example, on most Microsoft Windows 7 systems, the directory is:

`C:\users\username`

2. Use SAS language commands to include and run the installation macro, for example:

```
%include 'udftdins.sas' ;
%udftdins( server=my_teradata_server,
           user=teradata_admin,
           password=teradata_admin_password,
           database=SYSLIB
         ) ;
```

For more information, refer to the *SAS Analytics Accelerator 1.3 for Teradata: Guide* documentation at <http://support.sas.com/documentation/cdl/en/anlytaccltdug/63982/PDF/default/anlytaccltdug.pdf>.

Chapter 8 – Post-Installation Configuration for SAS/ACCESS Software

Before beginning your SAS/ACCESS software configuration, you should determine the following information about your DBMS:

- The version or release of the DBMS client shared libraries installed on your operating system. This is important due to potential incompatibilities between DBMS versions or releases.
- The location of the DBMS client shared libraries. This is important so that SAS/ACCESS software can be loaded at execution time.

Refer to the following sections for detailed DBMS-specific instructions on configuring your environment to interface with your SAS/ACCESS software.

SAS/ACCESS Interface to Aster nCluster Software

Registering SAS/ACCESS Interface to Aster nCluster

The following SAS procedure must be run so that the SAS/ACCESS to Aster nCluster product is registered with the SAS system catalog:

```
PROC NICKNAME CAT=sashelp.core engine;  
add nickname=aster module=sasioast desc="SAS/ACCESS to Aster"  
preferred eng;  
quit;
```

Installing and Configuring the ODBC Driver and Bulk Loader

Before configuring the ODBC driver, the bulk loader should be installed in <SASHOME>/SASFoundation/9.2/ or somewhere that is in the PATH environment variable.

The `odbc.ini` system information file contains a list of possible data sources to connect to your Aster nCluster servers. You must configure at least one data source in order to use the SAS/ACCESS Interface to Aster nCluster. A sample `odbc.ini` file may be included with the Aster nCluster ODBC Driver. You will have to edit the `odbc.ini` file with a text editor to configure the data sources. The general format of the `odbc.ini` file is shown below:

```
[ODBC Data Sources]  
nCluster=nCluster ANSI  
  
[ncluster]  
Description           = nCluster Connection  
Driver                = <path to the driver>/libnclusterodbcclient.so  
Trace                 = Yes  
Database              = beehive  
Servername            = 127.0.0.1  
Username              = beehive  
Password              = beehive  
Port                  = 2406  
Protocol              = 7.1  
ReadOnly              = No  
RowVersioning         = No
```

```
ShowSystemTables      = No
ShowOidColumn         = No
FakeOidIndex          = No
ConnSettings          =
```

After you configure your data sources, you must set the ODBCINI environment variable to the location and name of your `odbc.ini`:

- For Bourne Shell


```
ODBCINI=<path to>/odbc.ini
export ODBCINI
```
- For C Shell


```
setenv ODBCINI <path to>/odbc.ini
```

Finally, you must include the full path to the driver manager shared libraries in the shared library path as shown below so that the driver manager can be loaded dynamically at run time.

Linux for Intel Architecture and Linux for x64	
Bourne Shell	<pre>\$ LD_LIBRARY_PATH=<path to driver>/lib:\$LD_LIBRARY_PATH \$ export LD_LIBRARY_PATH</pre>
C Shell	<pre>\$ setenv LD_LIBRARY_PATH <path to driver>/lib:\${LD_LIBRARY_PATH}</pre>

SAS/ACCESS Interface to DB2 Software

The SAS/ACCESS Interface to DB2 executable uses shared libraries, referred to in UNIX as shared objects. You must add the location of the shared libraries to one of the system environment variables, and, if necessary, indicate the DB2 version that you have installed at your site. You must also set the `INSTHOME` environment variable to your DB2 home directory before setting the environment variables as shown in the examples.

AIX	
Bourne Shell	<pre>\$ LIBPATH=\$INSTHOME/lib:\$LIBPATH \$ export LIBPATH</pre>
C Shell	<pre>\$ setenv LIBPATH \$INSTHOME/lib:\$LIBPATH</pre>
HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	<pre>\$ SHLIB_PATH=\$INSTHOME/lib:\$SHLIB_PATH \$ export SHLIB_PATH</pre>
C Shell	<pre>\$ setenv SHLIB_PATH \$INSTHOME/lib:\$SHLIB_PATH</pre>
Linux for Intel Architecture, Linux for x64, Solaris, and Solaris for x64	
Bourne Shell	<pre>\$ LD_LIBRARY_PATH=\$INSTHOME/lib:\$LD_LIBRARY_PATH \$ export LD_LIBRARY_PATH</pre>
C Shell	<pre>\$ setenv LD_LIBRARY_PATH \$INSTHOME/lib:\$LD_LIBRARY_PATH</pre>

SAS/ACCESS Interface to Greenplum Software

SAS/ACCESS Interface to Greenplum uses the DataDirect Technologies Greenplum Wire Protocol ODBC driver component, which should be downloaded from SAS. Please see the section entitled “SAS/ACCESS Interface to Greenplum Software” in the System Requirements documentation for information about the download. When you have received the ODBC driver, use the following steps to unpack it and put it in the appropriate location.

1. CD to the !SASROOT/misc/dbi directory.
2. Copy the <platform>gplm60.tar file to the directory where you want the Greenplum drivers to reside. The <platform> prefix will be unique for each operating system.
3. Change directory to where you placed the tar file and extract the contents by issuing the following command at the UNIX prompt:

```
$ tar -xvf <platform>gplm60.tar
```

Once downloaded and installed, this directory becomes the ODBCHOME directory, which is used to set up the paths to the shared libraries as well as the `odbc.ini` file below. You must set the ODBCHOME environment variable to your ODBC home directory before setting the ODBCINI and shared library environment variables as shown in the examples below.

The `odbc.ini` system information file contains a list of possible data sources to connect to your Greenplum servers. You must configure at least one data source in order to use the SAS/ACCESS Interface to Greenplum. Edit the `odbc.ini` file with a text editor to configure the data sources.

The general format of the `odbc.ini` file is described below:

```
[ODBC Data Sources]
greenplum=SAS ACCESS to Greenplum

[ODBC]
InstallDir=<my install dir>
Trace=0
TraceDll=<my install dir>/lib/odbctrac.so
TraceFile=odbctrace.out

[greenplum]
Driver=<my install dir>/lib/S0gplm<file version>.so
Description=SAS ACCESS to Greenplum
AlternateServers=
ApplicationUsingThreads=1
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
Database=<db>
EnableDescribeParam=1
ExtendedColumnMetadata=0
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
FetchRefCursor=1
FetchTSWTZasTimestamp=0
FetchTWFSasTime=0
HostName=<Greenplum host>
InitializationString=
LoadBalanceTimeout=0
```

```
LoadBalancing=0
LoginTimeout=15
LogonID=
MaxPoolSize=100
MinPoolSize=0
Password=
Pooling=0
PortNumber=<Greenplum server port>
QueryTimeout=0
ReportCodepageConversionErrors=0
TransactionErrorBehavior=1
XMLDescribeType=-10
```

Note that the <driver version> and <file version> specifications describe specific versions of the DataDirect Greenplum driver that is installed with SAS/ACCESS Interface to Greenplum. The <driver version> in your `odbc.ini` will already contain the latest version of the DataDirect driver that SAS ships. Also, the <file version> will contain a two number version that will describe the version of the actual driver library. You will not need to update these two version specifications in your `odbc.ini` file. You should replace all occurrences of <my install dir> in the sample `odbc.ini` with the path and directory to which you installed the Greenplum ODBC driver. This is the same directory that is specified by the ODBC_HOME environment variable that you set earlier in this section. You should also replace <Greenplum host> with the IP address or named server of your Greenplum server, <Greenplum server port> with the port number that your Greenplum server is listening on (typically 5432), and <db> with the name of your Greenplum database.

In the above example, `greenplum` is the name of the configured data source name that is used in the `DSN=` option when assigning a libname to the SAS/ACCESS Interface to Greenplum engine. A sample completed `odbc.ini` is shown below for reference:

```
[ODBC Data Sources]
Greenplum=SAS ACCESS to Greenplum

[ODBC]
InstallDir=/TECHDBI/odbc/gpdrv
Trace=0
TraceDll=/TECHDBI/odbc/gpdrv/lib/odbctrac.so
TraceFile=/tmp/odbctrace.out

[greenplum]
Driver=/TECHDBI/odbc/gpdrv/lib/S0gplm60.so
Description=SAS ACCESS to Greenplum
AlternateServers=
ApplicationUsingThreads=1
ConnectionReset=0
ConnectionRetryCount=0
ConnectionRetryDelay=3
Database=sample
EnableDescribeParam=1
ExtendedColumnMetadata=0
FailoverGranularity=0
FailoverMode=0
FailoverPreconnect=0
```

```
FetchRefCursor=1
FetchTSWTZasTimestamp=0
FetchTWFSasTime=0
HostName=greenplum.unx.sas.com
InitializationString=
LoadBalanceTimeout=0
LoadBalancing=0
LoginTimeout=15
LogonID=
MaxPoolSize=100
MinPoolSize=0
Password=
Pooling=0
PortNumber=5432
QueryTimeout=0
ReportCodepageConversionErrors=0
TransactionErrorBehavior=1
XMLDescribeType=-10
```

After you configure your data sources, you must set the ODBCINI environment variable to the location and name of your odbc.ini:

- For Bourne Shell


```
ODBCINI=$ODBCHOME/odbc.ini
export ODBCINI
```
- For C Shell


```
setenv ODBCINI $ODBCHOME/odbc.ini
```

The DataDirect Greenplum ODBC drivers are ODBC API-compliant shared libraries, referred to in UNIX as shared objects. You must include the full path to the shared libraries in the shared library path as shown below so that the ODBC drivers can be loaded dynamically at run time.

Linux for Intel Architecture and Linux for x64, Solaris, and Solaris for x64	
Bourne Shell	\$ LD_LIBRARY_PATH=\$ODBCHOME/lib:\$LD_LIBRARY_PATH \$ export LD_LIBRARY_PATH
C Shell	\$ setenv LD_LIBRARY_PATH \$ODBCHOME/lib:\${LD_LIBRARY_PATH}
AIX	
Bourne Shell	\$ LIBPATH=\$ODBCHOME/lib:\$LIBPATH \$ export LIBPATH
C Shell	\$ setenv LIBPATH \$ODBCHOME/lib:\${LIBPATH}
HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	\$ SHLIB_PATH=\$ODBCHOME/lib:\$SHLIB_PATH \$ export SHLIB_PATH
C Shell	\$ setenv SHLIB_PATH \$ODBCHOME/lib:\${SHLIB_PATH}

Bulkload

SAS/ACCESS can interface with the Greenplum Client Loader interface for loading large volumes of data. To perform bulkloading, the Greenplum Client Loader Package must be present on the system where you install SAS.

SAS recommends using the “gpfdist” protocol for bulkloading. For this protocol you must set the Gpload_HOME environment variable to point to the location where the Greenplum Client Loader is installed. Further details on how to use the bulkloading feature can be found in the *SAS/ACCESS 9.2 for Relational Databases: Reference* documentation.

Registering SAS/ACCESS Interface to Greenplum

The following SAS procedure can be run so that the SAS/ACCESS Interface to Greenplum product is registered with the SAS system catalog:

```
PROC NICKNAME CAT=sashelp.core engine;
add nickname=greenplm module=sasiogpl desc="SAS/ACCESS to Greenplum"
preferred eng;
quit;
```

SAS/ACCESS Interface to HP Neoview Software

The HP Neoview ODBC driver is ODBC API-compliant shared libraries, referred to in UNIX as shared objects. You must include the full path to the shared libraries in the shared library path as shown below so that the ODBC drivers can be loaded dynamically at run time. You must also include the full path to any additional system shared libraries that may be required by the HP Neoview ODBC driver.

Note: The HP Neoview ODBC driver may require additional operating system libraries, libgcc version 3.4.3 or later and libstdc++ version 6.0 or later. Please contact HP Neoview for details.

Linux for Intel Architecture and Solaris	
Bourne Shell	\$ LD_LIBRARY_PATH=Neoview_ODBC_driver_install_directory /lib:Additional_system_library_directory:\$LD_LIBRARY_PATH \$ export LD_LIBRARY_PATH
C Shell	\$ setenv LD_LIBRARY_PATH Neoview_ODBC_driver_install_directory /lib:Additional_system_library_directory:\${LD_LIBRARY_PATH}
HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	\$ SHLIB_PATH=Neoview_ODBC_driver_install_directory /lib:.\$SHLIB_PATH \$ export SHLIB_PATH
C Shell	\$ setenv SHLIB_PATH Neoview_ODBC_driver_install_directory /lib:\${SHLIB_PATH}

AIX	
Bourne Shell	<pre>\$ LIBPATH=Neoview_ODBC_driver_install_directory /lib: Additional_system_library_directory:\$LIBPATH \$ export LIBPATH</pre>
C Shell	<pre>\$ setenv LIBPATH Neoview_ODBC_driver_install_directory /lib: Additional_system_library_directory:\${LIBPATH}</pre>

Additional Environment Variables for JNI Transporter on HP-UX for the Itanium Processor Family Architecture

The following environment variables are required for customers using the JNI Transporter with SAS/ACCESS Interface to HP Neoview on HP-UX for the Itanium Processor Family Architecture:

```
export JAVA_HOME=/opt/java1.5/jre
export NVTHOME=<Transporter install directory>
export SHLIB_PATH=<Neoview ODBC driver install directory>/
lib:$NVTHOME/lib
export JNVT_SPAWN=Y
```

Note that the `JAVA_HOME` should NOT point to the SAS-installed Java components since SAS uses a 32-bit JVM internally, but the Transporter layer requires a 64-bit JVM. Also, SAS typically specifies a `JAVA_HOME` setting in the `!SASROOT/bin` environment scripts, `sasenv` and `sasenv_local`. Customers wishing to use Transporter should comment out those lines in the `sasenv` and `sasenv_local` scripts so that the shell setting of `JAVA_HOME` takes precedence.

The `JNVT_SPAWN=Y` is a Transporter environment variable that causes Transporter to be launched in a separate process. This is needed since SAS needs to use a 32-bit JVM internally but Transporter requires a 64-bit JVM. Failing to set this variable can lead to errors such as the following:

```
/usr/lib/hpux64/dld.so: Unsatisfied data symbol 'UseSIGUSR2' in load
module '/opt/java6/jre/lib/IA64W/native_threads/libhpi.so'.
/usr/lib/hpux64/dld.so: Unsatisfied data symbol
'doCloseWithReadPending' in load module
'/opt/java6/jre/lib/IA64W/native_threads/libhpi.so'.

HPI shl_load failed: Unresolved external There was an error trying
to initialize the HPI library.

Please check libhpi in your java installation.
```

SAS/ACCESS Interface to Informix Software

For SAS 9.1 or higher, SAS/ACCESS Interface to Informix software uses an ODBC interface to access Informix.

You may have to edit the `.odbc.ini` file in your home directory with a text editor to configure data sources. Some ODBC driver vendors may allow system administrators to maintain a centralized copy by setting the environment variable `ODBCINI`. Please refer to your ODBC driver's vendor documentation to find more specific information.

The ODBC drivers are ODBC API-compliant shared libraries, referred to in UNIX as shared objects. You must add the location of the shared libraries to one of the system environment

variables so that ODBC drivers can be loaded dynamically at run time. You must also set the `InformixDIR` environment variable to your Informix home directory before setting the environment variables as shown in the examples.

AIX	
Bourne Shell	<code>\$ LIBPATH = \$InformixDIR/lib/cli:\$InformixDIR/lib/esql:\$LIBPATH</code> <code>\$ LIBPATH</code>
C Shell	<code>\$ setenv LIBPATH</code> <code>\$InformixDIR/lib/cli:\$InformixDIR/lib/esql:\$LIBPATH</code>
HP-UX and HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	<code>\$ SHLIB_PATH=\$InformixDIR/lib/cli:\$InformixDIR/lib/esql:\$SHLIB_PATH</code> <code>\$ export SHLIB_PATH</code>
C Shell	<code>\$ setenv SHLIB_PATH</code> <code>\$ InformixDIR/lib/cli:\$InformixDIR/lib/esql:\$SHLIB_PATH</code>
Linux for x64 and Solaris	
Bourne Shell	<code>\$ LD_LIBRARY_PATH=\$InformixDIR/lib/cli:\$InformixDIR/lib/esql:\$LD_LIBRARY_PATH</code> <code>\$ export LD_LIBRARY_PATH</code>
C Shell	<code>\$ setenv LD_LIBRARY_PATH \$InformixDIR/lib/cli:\$InformixDIR/lib/esql:\$LD_LIBRARY_PATH</code>

SAS/ACCESS Interface to Microsoft SQL Server Software

Before you can use the SAS/ACCESS Interface to Microsoft SQL Server, the following products are required:

Base SAS software

- ☐ SAS/ACCESS Interface to Microsoft SQL Server
- ☐ Microsoft SQL Server Version 7.0 or later

The SAS/ACCESS product contains the DataDirect Technologies Microsoft SQL Server ODBC driver component that is unloaded during the SAS/ACCESS Software Configuration phase of the installation. The setup/configuration procedures are described below.

During the SAS/ACCESS Software Configuration for MS SQL Server, you entered a directory in which the DataDirect Technologies driver components were unloaded. This directory becomes the `ODBCHOME` directory, which is used to set up the paths to the shared libraries as well as the `odbc.ini` file below. You must set the `ODBCHOME` environment variable to your ODBC home directory before setting the `ODBCINI` and shared library environment variables as shown in the examples below.

The `odbc.ini` system information file contains a list of possible data sources to connect to your Microsoft SQL Server servers. You must configure at least one data source in order to use the SAS/ACCESS Interface to Microsoft SQL Server. A sample `odbc.ini` file is located in the `ODBCHOME` directory as `odbc.ini.sample`. You will have to edit the `odbc.ini` file with a

text editor to configure the data sources. The general format of the `odbc.ini` file is shown below:

```
[ODBC Data Sources]
sqlserver=DataDirect <driver version>  SQL Server Wire Protocol

[sqlserver]
Driver=<my install dir>/lib/S0msss<file version>.so
Description=DataDirect <driver version>  SQL Server Wire Protocol
Address=<SQLServer host>,<SQLServer server port>
AnsiNPW=Yes
Database=<db>
LogonID=
Password=
QuotedId=yes

[ODBC]
InstallDir=<my install dir>
Trace=0
TraceDll=<my install dir>/lib/odbctrac.so
TraceFile=odbctrace.out
```

Note that the `<driver version>` and `<file version>` specifications describe specific versions of the DataDirect Microsoft SQL Server driver that is installed with SAS/ACCESS to Microsoft SQL Server. The `<driver version>` in your `odbc.ini` will already contain the latest version of the DataDirect driver that SAS ships. Also, the `<file version>` will contain a two number version that will describe the version of the actual driver library. You will not need to update these two version specifications in your `odbc.ini`.

You should replace all occurrences of `<my install dir>` in the sample `odbc.ini` with the path and directory name you specified during the SAS/ACCESS Software Configuration for MS SQL Server. This is the same directory that is specified by the `ODBCHOME` environment variable that you set earlier in this section.

You should also replace `<SQLServer host>` with the IP address or named server of your SQL Server machine, `<SQLServer server port>` with the port number that your SQL Server is listening on (typically 1433), and `<db>` with the name of your SQL Server database.

In the above example, `sqlserver` is the name of the configured data source name that is used in the `DSN=` option when assigning a `libname` to the SAS/ACCESS to MS SQL Server engine.

A sample completed `odbc.ini` is shown below for reference:

```
[ODBC Data Sources]
sqlserver=DataDirect 4.20 SQL Server Wire Protocol

[sqlserver]
Driver=/install/sas/driver/lib/S0msss19.so
Description=DataDirect 4.20 SQL Server Wire Protocol
Address=199.255.255.255,1433
AnsiNPW=Yes
Database=users
LogonID=
Password=
QuotedId=yes
```

```
[ODBC]
InstallDir=/install/sas/driver
Trace=0
TraceDll=/install/sas/driver/lib/odbcdrac.so
TraceFile=odbcdrac.out
```

After you configure your data sources, you must set the ODBCINI environment variable to the location and name of your `odbc.ini`:

- For Bourne Shell


```
ODBCINI=$ODBCHOME/odbc.ini
export ODBCINI
```
- For C Shell


```
setenv ODBCINI $ODBCHOME/odbc.ini
```

The DataDirect Microsoft SQL Server ODBC drivers are ODBC API-compliant shared libraries, referred to in UNIX as shared objects. You must include the full path to the shared libraries in the shared library path as shown below so that the ODBC drivers can be loaded dynamically at run time.

Linux for Intel Architecture, Linux for x64, Solaris, and Solaris for x64	
Bourne Shell	\$ LD_LIBRARY_PATH=\$ODBCHOME/lib:\$LD_LIBRARY_PATH \$ export LD_LIBRARY_PATH
C Shell	\$ setenv LD_LIBRARY_PATH \$ODBCHOME/lib:\${LD_LIBRARY_PATH}
AIX	
Bourne Shell	\$ LIBPATH=\$ODBCHOME/lib:\$LIBPATH \$ export LIBPATH
C Shell	\$ setenv LIBPATH \$ODBCHOME/lib:\${LIBPATH}
HP-UX and HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	\$ SHLIB_PATH=\$ODBCHOME/lib:\$SHLIB_PATH \$ export SHLIB_PATH
C Shell	\$ setenv SHLIB_PATH \$ODBCHOME/lib:\${SHLIB_PATH}

SAS/ACCESS Interface to MySQL Software

Setting Up the Symbolic Link

Use the following procedure to correctly link `sasmyl`:

1. Run SAS Setup from `!SASROOT/sassetup`. Make sure you have the appropriate privilege to update files in `!SASROOT`.
2. Select Run Setup Utilities from the SAS Setup Primary Menu.
3. Select Perform Product-Specific Configuration.
4. Select SAS/ACCESS Configuration.
5. Select the SAS/ACCESS Interface to MySQL from the selection screen. SAS Setup will set up the symbolic link.

Setting the Shared Libraries

The SAS/ACCESS Interface to MySQL executable uses shared libraries, referred to in UNIX as shared objects. You must add the location of the MySQL shared libraries to the shared library path environment variable specific to your operating system. Due to the way MySQL Client libraries are distributed, on most operating systems you will need to link the SAS ACCESS to MySQL module against the MySQL client libraries on your system using the following process.

1. Set the following environment variables:

`MYSQL_CLIENT_DIR`= directory location for the newly linked MySQL client module; needs to be set on your PATH so SAS can load the module (variable not needed on Linux for Intel Architecture and Linux for Itanium-based Systems)

`MYSQL_LIBDIR`= MySQL Client Install location (use location of dynamic client libraries – by default, `/usr/lib`, on Linux for Intel Architecture and Linux for Itanium-based Systems)

`SASROOT`= SAS installed directory location (variable not needed on Linux for Intel Architecture and Linux for Itanium-based Systems)

For example, if you have installed MySQL in the directory `/usr/local/mysql/5.0`, and SAS is installed in `/usr/local/sas` then you would need to do the following:

AIX, HP-UX, HP-UX for the Itanium Processor Family Architecture, Solaris, and Solaris for x64	
Bourne Shell	<pre>\$ MYSQL_LIBDIR=/usr/local/mysql/5.0/lib \$ export MYSQL_LIBDIR \$ SASROOT=/usr/local/sas \$ export SASROOT \$ MYSQL_CLIENT_DIR=/usr/local/sas_lib \$ export MYSQL_CLIENT_DIR</pre>
C Shell	<pre>\$ setenv MYSQL_LIBDIR /usr/local/mysql/5.0/lib \$ setenv SASROOT /usr/local/sas \$ setenv MYSQL_CLIENT_DIR /usr/local/sas lib</pre>
Linux for Intel Architecture and Linux for x64	
Bourne Shell	<pre>\$ MYSQL_LIBDIR=/usr/lib \$ export MYSQL_LIBDIR</pre>
C Shell	<pre>\$ setenv MYSQL_LIBDIR /usr/lib</pre>

2. Modify the shared library variable based on the host and shell you are using, according to the table below.

AIX	
Bourne Shell	<pre>\$ LIBPATH=\$MYSQL_CLIENT_DIR:\$LIBPATH \$ export LIBPATH</pre>
C Shell	<pre>\$ setenv LIBPATH \$MYSQL_CLIENT_DIR:\$LIBPATH</pre>

HP-UX and HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	\$ SHLIB_PATH=\$MYSQL_CLIENT_DIR:\$SHLIB_PATH \$ export SHLIB_PATH
C Shell	\$ setenv SHLIB_PATH \$MYSQL_CLIENT_DIR:\$SHLIB_PATH
Solaris and Solaris for x64	
Bourne Shell	\$ LD_LIBRARY_PATH=\$MYSQL_CLIENT_DIR:\$MYSQL_LIBDIR:\$LD_LIBRARY_PATH \$ export LD_LIBRARY_PATH
C Shell	\$ setenv LD_LIBRARY_PATH \$MYSQL_CLIENT_DIR:\$MYSQL_LIBDIR:\$LD_LIBRARY_PATH
Linux for Intel Architecture and Linux for x64	
Bourne Shell	\$ LD_LIBRARY_PATH=\$MYSQL_LIBDIR:\$LD_LIBRARY_PATH \$ export LD_LIBRARY_PATH
C Shell	\$ setenv LD_LIBRARY_PATH \$MYSQL_LIBDIR:\$LD_LIBRARY_PATH

- Once you have set up the environment variables you need to run the link command, as necessary, based on your operating system. This command will produce the file name `libmysqlclient_<os>` in the directory that was set as the `MYSQL_CLIENT_DIR`. You must have write permissions to this directory. You can confirm that the link ran successfully by bringing up SAS and assigning a libname to MySQL. If you get the following error message then you will need to double-check your link command.

```
ERROR: The SAS/ACCESS Interface to MySQL cannot be loaded. The
libmysqlclient
      code appendage could not be loaded.
ERROR: Error in the LIBNAME statement.
```

AIX

```
ld -berok -b64 -bM:SRE -bexpall -e _nostart
-o $MYSQL_CLIENT_DIR/libmysqlclient_aix $SASROOT/misc/dbi/obj/r64myl.o
-lpthread -L$MYSQL_LIBDIR -lmysqlclient -lm -lc
```

Linux for Itanium-based Systems

A link is not required for Linux for Itanium-based Systems because it supports the MySQL dynamic library.

Linux for Intel Architecture

A link is not required for Linux for Intel Architecture because it supports the MySQL dynamic library.

HP-UX

```
ld -b -o $MYSQL_CLIENT_DIR/libmysqlclient_hp
$SASROOT/misc/dbi/obj/h64myl.o
-L$MYSQL_LIBDIR -lmysqlclient -lc -lm
```

HP-UX for the Itanium Processor Family

```
ld -b -o $MYSQL_CLIENT_DIR/libmysqlclient_hpi
$SASROOT/misc/dbi/obj/h6imyl.o
-L$MYSQL_LIBDIR -lmysqlclient -lc -lm -lz
```

Solaris and Solaris for x64

You must include the location of the linker in the LD_LIBRARY_PATH environment variable (this is usually found in the libraries listed in the example, below):

Bourne Shell	\$ LD_LIBRARY_PATH=/usr/lib/???:/usr/ucblib/???:\$LD_LIBRARY_PATH \$ export LD_LIBRARY_PATH
C Shell	\$ setenv LD_LIBRARY_PATH /usr/lib/???:/usr/ucblib/???:\$LD_LIBRARY_PATH

The variable ??? should be filled based on the hardware your Solaris system is running on. If you are using a SPARC system, use sparcv9. If you are using Solaris for x64 on an Intel system, use 64. If you are using Solaris on x64 on an AMD system, use amd64.

Then use the link command:

(for Solaris)

```
ld -64 -G -o $MYSQL_CLIENT_DIR/libmysqlclient_sun
$SASROOT/misc/dbi/obj/s64myl.o -L$MYSQL_LIBDIR -lmysqlclient
```

(for Solaris on x64)

```
ld -64 -G -o $MYSQL_CLIENT_DIR/libmysqlclient_sax
$SASROOT/misc/dbi/obj/saxmyl.o -L$MYSQL_LIBDIR -lmysqlclient
```

SAS/ACCESS Interface to Netezza Software**Setting Up the Symbolic Link**

In order for your SAS software to correctly link to the image, you must identify the version of Netezza software you are using. Use the following procedure to do this:

1. Run SAS Setup from !SASROOT/sassetup. Make sure you have the appropriate privilege to update files in !SASROOT.
2. Select Run Setup Utilities from the SAS Setup Primary Menu.
3. Select Perform Product-Specific Configuration.
4. Select SAS/ACCESS Configuration.
5. Select the SAS/ACCESS Interface to Netezza from the selection screen.
6. Select the version of your Netezza software.

Setting the Shared Library Path

The Netezza ODBC drivers are ODBC API-compliant shared libraries, referred to in UNIX as shared objects. You must include the full path to the shared libraries in the shared library path as shown below so that the ODBC drivers can be loaded dynamically at run time.

Linux for Intel Architecture, Linux for x64, and Solaris	
Bourne Shell	\$ LD_LIBRARY_PATH=\$ODBCHOME/lib:\$LD_LIBRARY_PATH \$ export LD_LIBRARY_PATH
C Shell	\$ setenv LD_LIBRARY_PATH \$ODBCHOME/lib:\${LD_LIBRARY_PATH}
AIX	
Bourne Shell	\$ LIBPATH=\$ODBCHOME/lib:\$LIBPATH \$ export LIBPATH
C Shell	\$ setenv LIBPATH \$ODBCHOME/lib:\${LIBPATH}
HP-UX and HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	\$ SHLIB_PATH=\$ODBCHOME/lib:\$SHLIB_PATH \$ export SHLIB_PATH
C Shell	\$ setenv SHLIB_PATH \$ODBCHOME/lib:\${SHLIB_PATH}

SAS/ACCESS Interface to ODBC Software

You may have to edit the `.odbc.ini` file in your home directory with a text editor to configure data sources. Some ODBC driver vendors may allow system administrators to maintain a centralized copy by setting the environment variable `ODBCINI`. Please refer to your ODBC driver's vendor documentation to find more specific information.

The ODBC drivers are ODBC API-compliant shared libraries, referred to in UNIX as shared objects. You must add the location of the shared libraries to one of the system environment variables so that ODBC drivers can be loaded dynamically at run time. You must also set the `ODBCHOME` environment variable to your ODBC home directory before setting the environment variables as shown in the examples.

Linux for Intel Architecture and Linux for x64	
Bourne Shell	\$ LD_LIBRARY_PATH=\$ODBCHOME/lib:\$LD_LIBRARY_PATH \$ export LD_LIBRARY_PATH
C Shell	\$ setenv LD_LIBRARY_PATH \$ODBCHOME/lib:\$LD_LIBRARY_PATH
Solaris and Solaris for x64	
Bourne Shell	\$ LD_LIBRARY_PATH=\$ODBCHOME/lib:\$LD_LIBRARY_PATH \$ export LD_LIBRARY_PATH
C Shell	\$ setenv LD_LIBRARY_PATH \$ODBCHOME/lib:\${LD_LIBRARY_PATH}
AIX	
Bourne Shell	\$ LIBPATH=\$ODBCHOME/lib:\$LIBPATH \$ export LIBPATH
C Shell	\$ setenv LIBPATH \$ODBCHOME/lib:\${LIBPATH}

HP-UX and HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	\$ SHLIB_PATH=\$ODBCHOME/lib:\$SHLIB_PATH \$ export SHLIB_PATH
C Shell	\$ setenv SHLIB_PATH \$ODBCHOME/lib:\${SHLIB_PATH}

SAS/ACCESS Interface to Oracle Software

Setting Up the Symbolic Link

In order for your SAS software to correctly link sasora to the image, you must identify the version of Oracle software you are using. Use the following procedure to do this:

1. Run SAS Setup from !SASROOT/sassetup. Make sure you have the appropriate privilege to update files in !SASROOT.
2. Select Run Setup Utilities from the SAS Setup Primary Menu.
3. Select Perform Product-Specific Configuration.
4. Select SAS/ACCESS Configuration.
5. Select the SAS/ACCESS Interface to Oracle from the selection screen.
6. Select the version of your Oracle software.

Setting the ORACLE_HOME Variable

In order to use SAS/ACCESS Interface to Oracle software, you must set the ORACLE_HOME environment variable. In addition, you must make sure that the shared library path variable (the name of this variable is operating system dependent) points to where the Oracle shared libraries are located. This is required since the SAS/ACCESS Interface to Oracle executable uses Oracle shared libraries and needs to know where they are located at your site.

The following are examples for the various operating systems:

AIX	
Bourne Shell	\$ LIBPATH=\$ORACLE_HOME/lib:\$LIBPATH \$ export LIBPATH
C Shell	\$ setenv LIBPATH=\$ORACLE_HOME/lib:\$LIBPATH
HP-UX and HP-UX for the Itanium Processor Family Architecture	
Bourne Shell	\$ SHLIB_PATH=\$ORACLE_HOME/lib:\$SHLIB_PATH \$ export SHLIB_PATH
C Shell	\$ setenv SHLIB_PATH \$ORACLE_HOME/lib:\$SHLIB_PATH
Linux for Intel Architecture, Linux for Itanium-based Systems, Solaris, and Solaris for x64	
Bourne Shell	\$ LD_LIBRARY_PATH=\$ORACLE_HOME/lib:\$LD_LIBRARY_PATH \$ export LD_LIBRARY_PATH
C Shell	\$ setenv LD_LIBRARY_PATH \$ORACLE_HOME/lib:\$LD_LIBRARY_PATH

SAS/ACCESS Interface to R/3 Software

SAS/ACCESS Interface to R/3 software requires extensive post-installation configuration before it can be used. Refer to the *Configuration Instructions for SAS/ACCESS 4.3 Interface to R/3* on Install Center (<http://support.sas.com/idsearch?ct=200000>) for detailed information.

SAS/ACCESS Interface to Sybase Software

For users of Sybase Open Client 15, in order to correctly copy the Sybase libraries for use with SAS/ACCESS Interface to Sybase, you must have read/write authority for `$SYBASE/OCS-15_0/lib` and `$SYBASE/OCS-15_0/devlib` in order to run `$SYBASE/OCS-15_0/scripts/lnsybllib`. Instructions for copying the Sybase libraries are in the header comments in the `lnsybllib` file.

Setting Up the Symbolic Link

Use the following procedure to correctly link `sassyb`:

1. Run `SAS Setup` from `!SASROOT/sassetup`. Make sure you have the appropriate privilege to update files in `!SASROOT`.
2. Select `Run Setup Utilities` from the `SAS Setup Primary Menu`.
3. Select `Perform Product-Specific Configuration`.
4. Select `SAS/ACCESS Configuration`.
5. Select the `SAS/ACCESS Interface to Sybase` from the selection screen.
6. Select the version of your Sybase software.

Installing Sybase Procedure

In SAS 9.2, the administrator or user must install a Sybase-stored procedure on the target Sybase server. Two files have been included in the `!SASROOT/misc/dbi` directory to assist in the installation:

- `sas-spcp.txt` is a text file containing instructions on how to do the installation.
- `sas-spdf.txt` is the actual stored procedure script.

The process utilizes two Sybase facilities, `defncopy` and `isql`.

Adding Shared Libraries

Finally, the SAS/ACCESS Interface to Sybase executable uses shared libraries, referred to in UNIX as shared objects. You must add the location of the shared libraries to one of the system environment variables and, if necessary, indicate the Sybase version that you have installed at your site. You must also set the `Sybase` environment variable to your Sybase home directory before setting the environment variables as shown in the examples.

AIX	
Bourne Shell	\$ LIBPATH=\$INSTHOME/lib:\$LIBPATH \$ export LIBPATH
C Shell	\$ setenv LIBPATH \$INSTHOME/lib:\$LIBPATH
HP-UX and HP-UX for the Itanium Processor Family	
Bourne Shell	\$ SHLIB_PATH=\$Sybase/lib:/lib:\$SHLIB_PATH \$ export SHLIB_PATH
C Shell	\$ setenv SHLIB_PATH \$Sybase/lib:/lib:\$SHLIB_PATH
Linux for Intel Architecture, Linux for x64, Solaris, and Solaris for x64	
Bourne Shell	\$ LD_LIBRARY_PATH=\$Sybase/lib:/lib:\$LD_LIBRARY_PATH \$ export LD_LIBRARY_PATH
C Shell	\$ setenv LD_LIBRARY_PATH \$Sybase/lib:/lib:\$LD_LIBRARY_PATH

SAS/ACCESS Interface to Sybase IQ Software

After your installation is complete, you will need to run a script that will set up your environment, including the path to the shared library. This script is located in the installation directory for your Sybase IQ software. If you are using Sybase IQ 12.7, the script is named `ASIQ-12_7.sh` (or `.csh`). If you are using Sybase IQ 15 or later, the script is named after version. For example for Sybase IQ 15, the script is named `IQ-15_0.sh`.

Registering SAS/ACCESS Interface to Sybase IQ

The following SAS procedure must be run so that the SAS/ACCESS to Sybase IQ product is registered with the SAS system catalog:

```
PROC NICKNAME CAT=sashelp.core engine;
add nickname=sybaseiq module=sasiosiq desc="SAS/ACCESS to Sybase IQ"
preferred eng;
quit;
```

SAS/ACCESS Interface to Teradata Software

Access to Shared Libraries

The SAS/ACCESS Interface to Teradata executable uses shared libraries, referred to in UNIX as shared objects. These shared objects typically reside in `/usr/lib`. You must add the location of the shared libraries to one of the system environment variables.

AIX	
Bourne Shell	\$ LIBPATH=TERADATA-CLIENT-LOCATION:\$LIBPATH \$ export LIBPATH
C Shell	\$ setenv LIBPATH TERADATA-CLIENT-LOCATION:\$LIBPATH

HP-UX	
Bourne Shell	\$ SHLIB_PATH=TERADATA-CLIENT-LOCATION:\$SHLIB_PATH \$ export SHLIB_PATH
C Shell	\$ setenv SHLIB_PATH TERADATA-CLIENT-LOCATION:\$SHLIB_PATH
HP-UX for the Itanium Processor Family	
Bourne Shell	\$ SHLIB_PATH=TERADATA-CLIENT-LOCATION:\$SHLIB_PATH \$ export SHLIB_PATH \$ LD_PRELOAD=/usr/lib/hpux64/libpthread.so.1 \$ export LD_PRELOAD
C Shell	\$ setenv SHLIB_PATH TERADATA-CLIENT-LOCATION:\$SHLIB_PATH \$ setenv LD_PRELOAD /usr/lib/hpux64/libpthread.so.1
Linux for Intel Architecture, Linux for x64, Solaris, and Solaris for x64	
Bourne Shell	\$ LD_LIBRARY_PATH=TERADATA-CLIENT-LOCATION:\$LD_LIBRARY_PATH \$ export LD_LIBRARY_PATH
C Shell	\$ setenv LD_LIBRARY_PATH TERADATA-CLIENT-LOCATION:\$LD_LIBRARY_PATH

TTU 8.2 and HP-UX

HP-UX users with TTU 8.2 must create two symbolic links from the /usr/lib/pa20_64 directory with the following commands:

```
$ ln -s /usr/lib/pa20_64/libicudatatd.sl libicudatatd.sl.34
$ ln -s /usr/lib/pa20_64/libicuuctd.sl libicuuctd.sl.34
```

FastExporting

For optimal reads of large tables, SAS/ACCESS can perform FastExporting. To perform FastExporting, the Teradata FastExport Utility must be present on the system where you install SAS.

As needed, modify your library path environment variable to include the directory containing sasaxsm.sl (HP-UX) or sasaxsm.so (Linux, Solaris, and AIX). These shared objects are delivered in the \$SASROOT/sasexe directory. You may copy these modules where you wish, but ensure that the directory you copy them into is in the appropriate shared library path environment variable.

On Solaris and Linux, the library path variable is LD_LIBRARY_PATH. On HP-UX, it is SHLIB_PATH. On AIX, it is LIBPATH. Also, make sure that the Teradata FastExport utility, fexp, has its directory included in the PATH environment variable. This utility is usually installed in the /usr/bin directory.

The FastExport Utility is not required; SAS/ACCESS reads large tables quite efficiently without it. For further information, see the DBSLICEPARM option in your SAS/ACCESS to Teradata documentation. Contact Teradata if you want to obtain the Teradata FastExport Utility.

MultiLoad

SAS/ACCESS can interface with MultiLoad for loading large volumes of data. To perform MultiLoading, the Teradata MultiLoad Utility must be present on the system where you install SAS.

As needed, modify your library path environment variable to include the directory containing the shared objects `sasmlam.sl` and `sasmlne.sl` (HP-UX) or `sasmlam.so` and `sasmlne.so` (Linux, Solaris, HP-UX for the Itanium Processor Family and AIX). These shared objects are delivered in the `$SASROOT/sasexe` directory. You may copy these modules where you wish, but ensure that the directory you copy them into is in the appropriate shared library path environment variable. On Solaris and Linux, the library path variable is `LD_LIBRARY_PATH`. On HP-UX and HP-UX for the Itanium Processor Family, it is `SHLIB_PATH`. On AIX, it is `LIBPATH`. Also, make sure that the Teradata MultiLoad utility, `mload`, has its directory included in the `PATH` environment variable. This utility is usually installed in the `/usr/bin` directory.

The MultiLoad Utility is not required; SAS/ACCESS provides other options for loading tables. For further information, see the `MULTISTMT` option in your SAS/ACCESS Interface to Teradata documentation. Contact Teradata if you want to obtain the Teradata MultiLoad Utility.

Teradata Parallel Transporter

SAS/ACCESS supports the Teradata parallel transporter API for loading data using Multiload, Fastload, and multi-statement inserts. The API also supports reading data using FastExport.

Note: The Teradata Parallel Transporter API is not required; SAS/ACCESS provides other options for loading and reading data.

If you plan to use the Teradata parallel transporter API, the following two requirements must be met:

1. The API must be installed on the system where SAS is installed.
2. The path system variable must include the location of the Teradata Parallel transporter API libraries (specifically the location of `libtelapi.*`). It may be necessary to set other environment variables depending on the type of UNIX environment. Some of these variables may have already been set correctly when Teradata parallel transporter was installed.

AIX:

```
LIBPATH=TPT-API-LIBRARY-LOCATION:$LIBPATH
NLSPATH=TPT-API-MESSAGE-CATALOG-LOCATION
LC__FASTMSG=false           // Note: There are two underscores
```

HP-UX and HP-UX for the Itanium Processor Family:

```
SHLIB_PATH=TPT-API-LIBRARY-LOCATION:$SHLIB_PATH
NLSPATH=TPT-API-MESSAGE-CATALOG-LOCATION
```

Linux for Intel Architecture, Linux for x64, and Solaris for x64:

```
LD_LIBRARY_PATH=TPT-API-LIBRARY-LOCATION:$LD_LIBRARY_PATH
NLSPATH=TPT-API-MESSAGE-CATALOG-LOCATION
```


Chapter 9 – Post-Installation Configuration for SAS/ASSIST Software

This chapter describes how to add a master profile to SAS/ASSIST software. You can use a master profile to override the default SAS settings. This allows you to provide a customized setup for SAS/ASSIST software. With the master profile, you can control the profile options of all SAS/ASSIST users from one central place. For information on the profile options, refer to the *SAS/ASSIST Software System Administrator's Guide*.

Adding a Master Profile

Complete the following steps to add a master profile to SAS/ASSIST software.

1. Specify the location of the master profile by creating a new directory to which all users of SAS/ASSIST software will have read access.

All users with write access to this directory will automatically have write access to the master profile in SAS/ASSIST software. Select a name that conforms to the naming conventions of your installation. The name of this new directory must be stored in an entry in the `SASHELP` library. This requires that you have write access to the `SASHELP` library.

On line 1 of the `Program Editor` window of the SAS Display Manager System, type the physical pathname of the master profile directory. Execute the `Save` command to store this pathname in the `SASHELP.QASSIST` catalog. Save it as `SASHELP.QASSIST.PARMS.SOURCE`. The location of the master profile will now be known by SAS/ASSIST software.

2. Create the master profile.

The first time SAS/ASSIST software is started, a master profile is created if `SASHELP.QASSIST.PARMS.SOURCE` contains the name of an existing physical pathname, and the person who starts SAS/ASSIST software has write access to this physical pathname.

3. Customize the master profile by starting SAS/ASSIST software and selecting:

```
Setup, then  
Profiles, and then  
Master/group ...
```

If you have write access to the SAS library containing the master profile, you can specify default values. New users will use these default values when they start SAS/ASSIST software.

Note: *If you restrict values by typing `R` in `Status`, users will not be allowed to change the values you define.*

You can run SAS/ASSIST software in two different styles - Workplace or Block Menu. The Block Menu can be New style or Old style. You can control this using the profile options below.

Run Workplace:

```
SAS/Assist style:           Workplace
```

Run Block Menu New style:

```
SAS/Assist style:      Block Menu
Save selections on end: Yes
Menu Style:           New
```

Run Block Menu Old style:

```
SAS/Assist style:      Block Menu
Save selections on end: Yes
Menu Style:           Old
```

By setting the default values in the master profile, you can control if users should use the New or Old style of SAS/ASSIST software. In addition, there are many other profile options. For more information on these options, refer to the *SAS/ASSIST Software System Administrator's Guide*.

4. Create group profiles.

From the master profile, it is possible to create group profiles to allow groups of users to have different setups. The master profile controls group profiles and user profiles when a user is not a member of any group. All users are indirectly controlled by the master profile when option values are set to a restricted status.

```
Select Setup...Master/Group
then Tools...Create Group Profile.
```

To add users to a group profile, select Tools...Update User Group.

By default, the user ID is found in the macro variable `&SYSJOBID`. This value is set in the option `Userid` in the master profile (option type `System Administration`). Change the value if your site uses another variable to keep the user ID. If the value name starts with `&`, it is a macro variable; otherwise, it is an environment variable, which is set before the start of SAS 9.2.

Chapter 10 – Post-Installation Configuration for SAS/CONNECT Software

TCP/IP is the access method supported for UNIX environments and their derivatives. Refer to the publication *Communications Access Methods for SAS/CONNECT and SAS/SHARE Software* for information on the access methods supported by other systems. This document can be found at <http://support.sas.com/documentation/onlinedoc>.

Storing and Locating SAS/CONNECT Script Files

SAS/CONNECT software ships several sample script files that are used to establish a connection to a remote SAS session. The `SASSCRIPT` configuration option points to the location of the SAS/CONNECT script files. The `SASSCRIPT` option is used by SAS/ASSIST software and can be used by user-written SCL applications.

The script files are installed into the `!SASROOT/misc/connect` directory by default. The following line has been included in the `sasv9.cfg` file in order to define the default script file location:

```
-SASSCRIPT !SASROOT/misc/connect
```

If you want to move the script files to another directory, you must edit the `sasv9.cfg` file and update the `SASSCRIPT` option with the new directory location.

Chapter 11 – Post-Installation Configuration for SAS/GRAPH Software

Loading SAS Fonts to Your X Display Server

Many SAS/GRAPH procedures and devices now support ODS styles in all destinations, including the LISTING destination. By default, all colors, fonts, symbols, and graph sizes are derived from the current style. The default fonts in these styles are the TrueType fonts provided by SAS. Devices that use FreeType rendering are able to find these fonts by default and render them in an environment without a DISPLAY set or valid Xdisplay available. For devices like XCOLOR that use host-rendering, the fonts must be registered with the display in order for them to work. You may override the default font setting by using the FTEXT option on the GOPTIONS statement or by creating a modified style sheet. However, SAS recommends you make the TrueType fonts available to the display device to take advantage of their benefits.

Refer to your vendor user documentation for your X display server for the instructions for making the SAS fonts available to it. SAS's fonts are located at `$SASROOT/misc/fonts`.

Making System Fonts Available to SAS

One of the main advantages of using FreeType rendering is that TrueType and other hardware fonts that produce high quality text are available in an environment without a DISPLAY set. The graphics devices that use FreeType rendering will only recognize fonts that have been registered in SAS.

If you wish to register additional fonts to SAS, including system or display fonts, use the FONTREG procedure to update the SAS registry to include these fonts. For full information about the use and syntax of the FONTREG procedure, please see the appropriate chapter in the *Base SAS 9.2 Procedures Guide*, available from <http://support.sas.com>.

Chapter 12 – Post-Installation Configuration for SAS/IntrNet Software

This chapter has information for your SAS/IntrNet installation. It will help you install, configure, and test your SAS/IntrNet components.

The procedures for installing SAS software using the SAS Deployment Wizard are described in other documentation and not available from this chapter. Furthermore, the installation of your Web server is your responsibility and not described in SAS documentation.

When the SAS/IntrNet software has been installed, configured and tested using the procedures described in this chapter, review the latest version of the SAS/IntrNet product documentation online at

<http://support.sas.com/documentation/onlinedoc/IntrNet/index.html>. The “What’s New” page at this Web site lists any recent changes to the product or documentation.

Overview

All SAS/IntrNet installations are made up of two components:

1. The SAS/IntrNet server (also referred to as the Application Server). This is where SAS Foundation is installed.
2. CGI Tools (also referred to as the Broker). This is where the `broker.cfg` file and its supporting files are installed.

When you install SAS/IntrNet, choose between two installation configurations:

Type A - The SAS/IntrNet server and CGI Tools components are both installed on the same system machine. The Web server **must** be installed before starting the SAS installation.

Type B -The SAS/IntrNet server component is installed on one system machine and the CGI Tools component is installed on a different system machine. The Web server **must** be installed on the CGI Tools system machine prior to installing CGI Tools.

Type A and Type B require different installation steps:

Type A Installation Steps	Type B Installation Steps
Confirm that the Web server software (IIS, Apache etc.) is on the same system machine as your SAS/IntrNet software.	Confirm that the Web server software (IIS, Apache, etc.) is on the system machine where you will install CGI Tools.
Install your SAS products. Check “CGI Tools for the Web Server” in the “Select Products to Install” dialog.	On your application server system machine, start your SAS installation. Uncheck “CGI Tools for the Web Server” in the “Select Products to Install” dialog.
	On your Web server system machine, start your SAS installation. Uncheck all products except “CGI Tools for the Web Server” in the “Select Products to Install” dialog.
	You can optionally check the IntrNet Monitor or Connect Drivers

Test the Broker
Configure a Socket Service
Start the Socket Service
Test the Socket Service

Test the Broker
Configure a Socket Service
Start the Socket Service
Test the Socket Service

The steps are described more thoroughly in the sections that follow.

Installing and Configuring SAS/IntrNet Software

Install Your Web Server Software

Refer to your Web server's documentation for its installation procedures.

Install Your SAS Software

Refer to your *QuickStart Guide* for a description of how to start your SAS software installation.

If you are performing a Type A installation (as described in the "Overview" above), confirm that your Web server software is installed before starting your SAS software installation. Check "CGI Tools for the Web Server" in the **Select Products to Install** dialog.

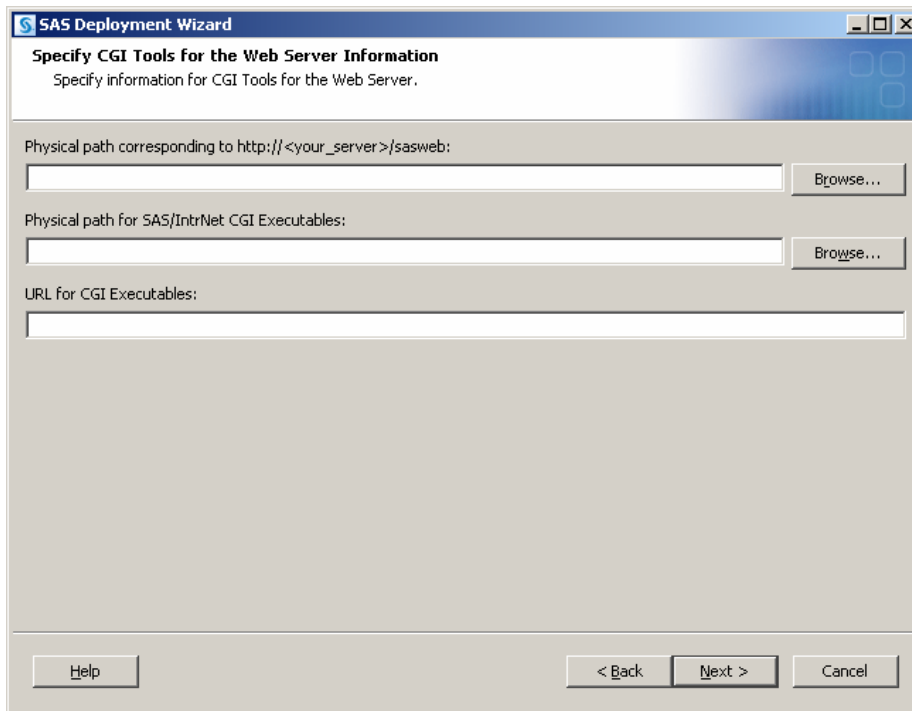
If you are performing a Type B installation (as described in the "Overview" above), do **both** of the following:

- Install the SAS software on the SAS System side, unchecking "CGI Tools for the Web Server" and "SAS/GRAPH Java Applets" in the **Select Products to Install** dialog.
- Start the SAS software install on the Web server and check "CGI Tools for the Web Server" and "SAS/GRAPH Java Applets" in the **Select Products to Install** dialog. SAS/IntrNet Monitor and SAS/CONNECT Driver for Java are optional selections. Uncheck everything else.

CGI Tools Installation Dialogs

The following screens appear for CGI Tools for the Web Server for all installations. Click **Help** on any dialog for information about the fields.

Customary entries are documented following each screen shown below. Customize the entries according to your environment.



The following are examples of common entries for popular Web servers. Customize your entries according to your own web Server environment. These fields will tell SAS where your Web server software is located.

Physical path corresponding to `http://<your_server>/sasweb`:

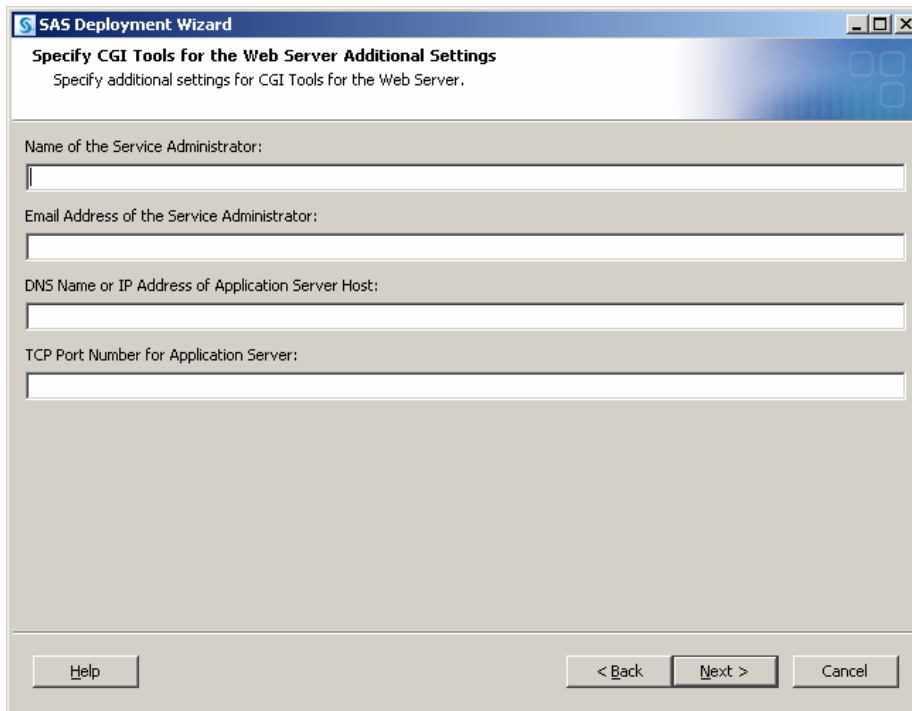
IIS:	<code>C:\Inetpub\wwwroot\sasweb</code>
Apache (Windows):	<code>C:\program files\Apache Software Foundation\Apache2.2\htdocs\sasweb</code>
Apache (UNIX):	<code>/usr/local/apache2/htdocs/sasweb</code>

Physical path for SAS/IntrNet CGI Executables:

IIS:	<code>C:\Inetpub\scripts</code>
Apache (Windows):	<code>C:\program files\Apache Software Foundation\Apache2.2\cgi-bin</code>
Apache (UNIX):	<code>/usr/local/apache2/cgi-bin</code>

URL for CGI Executables:

IIS:	<code>http://<web_servername>/scripts</code> Example: <code>http://abcserver.comp.com/scripts</code>
Apache (Windows):	<code>http:// <web_servername>/cgi-bin</code> Example: <code>http://abcserver.comp.com/cgi-bin</code>
Apache (UNIX):	<code>http://<web_servername>/cgi-bin</code> Example: <code>http://abcserver.comp.com/cgi-bin</code>



SAS Deployment Wizard

Specify CGI Tools for the Web Server Additional Settings
Specify additional settings for CGI Tools for the Web Server.

Name of the Service Administrator:

Email Address of the Service Administrator:

DNS Name or IP Address of Application Server Host:

TCP Port Number for Application Server:

Note that your entries for this dialog are added to the `broker.cfg` file. The `broker.cfg` file is a text file that can be edited after the installation is complete.

Name of the Service Administrator:

(optional) Enter the name of the administrator (for example, John Doe).

Email Address of the Service Administrator:

(optional) Enter the e-mail address of the administrator (for example, NetAdmin@comp.com).

DNS Name or IP Address of Application Server Host:

Enter the DNS name or IP address of the application server host where SAS Foundation is located.

TCP Port Number for Application Server:

The customary default port number is *5001*, but you can use any valid available port on your system between 256 – 65535.

Installing CGI Tools and SAS Foundation on Machines with Different Operating Systems

Your SAS Foundation system's operating system might be different than your CGI Tools system's operating system. For example, your SAS Foundation might be installed on a Windows system and your CGI Tools might be installed on a UNIX system. The CGI Tools install from the SDW will detect the destination operating system and install the appropriate operating system-specific software.

There are two methods to make the SAS Software Depot available to the installer on the destination CGI Tools system. The method you choose is dependent on the facilities available at

your site. To access a SAS Software Depot on the destination CGI Tools system, do one of the following:

1. Launch the set-up from a SAS Software Depot residing on a remote system. You might need to use a cross-platform file access method, such as NFS or SAMBA, to connect the two systems.
2. Create media from an existing depot using the SAS Deployment Wizard and use that media on the host machine. This process is described more thoroughly in the *SAS Deployment Wizard Users Guide*, available from Install Center (<http://support.sas.com/documentation/installcenter/92/documents/index.html>).

Note: *SAS/IntrNet operation requires TCP/IP connectivity between the SAS Foundation system and the CGI Tools system regardless of which operating systems these components are installed on.*

Test the Web Server

To determine if the Web server is running, launch the Web server's browser and enter `http://localhost`. This will return a Web page if the Web server is running.

If you do not receive a web page, you must debug or reinstall your Web server before continuing.

Test the Application Broker

To verify that CGI Tools was installed correctly and can access the `broker.cfg` file, point your Web browser to the following URL:

Windows:

IIS - `http://your_webserver/scripts/broker.exe`

Apache - `http://your_webserver/cgi-bin/broker.exe`

Other hosts:

`http://your_webserver/cgi-bin/broker`

Replace `your_webserver` with the name of the Web server. The URL path might also need to be changed if you installed CGI Tools to a different directory. You should see a Web page similar to the following:

SAS/IntrNet Application Dispatcher

Application Broker Version 9.2 (Build 1495)

[Application Dispatcher Administration](#)

[SAS/IntrNet Samples](#)

[SAS/IntrNet Documentation](#) - requires Internet access

If you do not receive this page, you must debug your Web server installation before continuing. Verify that your Web server is enabled for CGI execution in the directory where you installed the Application Broker (`broker.exe` and `broker.cfg` files). This directory was determined by what was entered for **Physical path for SAS/IntrNet CGI Executables** during the CGI Tools installation above.

Configure a Socket Service

On UNIX platforms, the configuration utility is a Perl script. Perform the following steps to create and start the default service:

1. From a system prompt, submit the following command:

```
!SASROOT/utilities/bin/inetcfg.pl
```

where !SASROOT is the path to the SAS root directory.

As the configuration utility runs, you are prompted for information about the service that you are creating.
2. Press **Return** to accept the default value, which names the service *default*.
3. The next prompt asks for the name of the directory where all of the service control files should be stored. Press **Return** to accept the suggested value, or type the desired directory name and then press **Return**.
4. Type **S** and press **Return** to define a socket service.
5. Press **Return** to select one server.
6. Type the TCP/IP port number that you reserved for this service and press **Return**.
7. Press **Return** to skip entering an administrator password. You can add an administrator password later if you use this service for production applications.
8. Verify the displayed information and press **Return** to create the service. Note the path for the service directory.
9. The configuration utility created a start.pl file to start the default Application Server. Change to the service directory path and start the server by submitting the following command:

```
./start.pl
```

Starting the Socket Service

As stated above, change to the service directory path and start the server by submitting the following command:

```
./start.pl
```

Testing the Socket Service

1. To make sure that the service was installed and started correctly, point your Web browser to this URL:

Windows:

IIS - `http://your_webserver/scripts/broker.exe`

Apache - `http://your_webserver/cgi-bin/broker.exe`

Other hosts:

`http://your_webserver/cgi-bin/broker`

Replace `your_webserver` with the name of the Web server. The URL path might also need to be changed if you installed the Application Broker to a different directory. You should see the following Web page:

SAS/IntrNet Application Dispatcher

Application Broker Version 9.2 (Build 1495)

[Application Dispatcher Administration](#)

[SAS/IntrNet Samples](#)

[SAS/IntrNet Documentation](#) - requires Internet access

- Click on the **Application Dispatcher Administration** link to see if the Application Broker can read the `broker.cfg` file. The Application Dispatcher Services Web page should open.
- Verify connectivity between the Application Server and the Web server. Click on the **Application Dispatcher Administration** link and then click on the **ping** link under **SocketService default** heading. If the ping is successful, you should see:

Ping. The Application Server <hostname>:<port_number> is functioning properly.

- To complete installation testing, type this URL in your browser address line:

Windows:

IIS - `http://your_webserver/scripts/broker.exe?_service=default&_program=sample.webhello.sas`

Apache - `http://your_webserver/cgi-bin/broker.exe?_service=default&_program=sample.webhello.sas`

Other hosts:

`http://your_webserver/cgi-bin/broker?_service=default&_program=sample.webhello.sas`

You should see the string "Hello World!" in large bold type in your browser. If you do not, add the debug option to create a log:

Windows:

IIS - `http://your_webserver/scripts/broker.exe?_service=default&_program=sample.webhello.sas&_debug=131`

Apache - `http://your_webserver/cgi-bin/broker.exe?_service=default&_program=sample.webhello.sas&_debug=131`

Other hosts:

`http://yourWebserver/cgi-bin/broker?_service=default&_program=sample.webhello.sas&_debug=131`

Save the log screen on the browser for SAS Technical Support.

Configure Additional Services

This chapter only describes how to configure a simple default Application Dispatcher service. There are many reasons you may want to configure additional services, including segregating applications by security or performance requirements and implementing more scalable servers. See the “Using Services” section of the SAS/IntrNet Application Dispatcher documentation at <http://support.sas.com/documentation/onlinedoc/intrnet/index.html> for information on configuring additional services, using the Load Manager, and adding pool services.

Chapter 13 – Post-Installation Configuration for SAS/SECURE Software

SAS/SECURE software includes client components that non-SAS System client applications can use to communicate with a SAS server in a secure environment. To use encryption between a non-SAS System client and a SAS server with SAS/SECURE software licensed, you must install the SAS/SECURE client components on the client machine.

Note: This install is not necessary if the SAS System is your client. The SAS System installs the components that it needs as part of the SAS System install process.

SAS/SECURE Client for Windows

The SAS/SECURE components needed by Windows clients can be installed by running the SAS Deployment Wizard.

SAS/SECURE Client for Java

The SAS/SECURE components for Java clients provide encryption support for Java applications. You can incorporate this support into applications that are written using the following components:

- SAS/SHARE driver for JDBC,
- SAS/CONNECT driver for Java, and
- IOM Bridge for Java.

The SAS/SECURE components needed by Java clients can be installed by running the SAS Deployment Wizard. The SECUREJAVA folder contains two JAR files that enable Java clients to use the CryptoAPI algorithms:

- `sas.rutil.jar` - should be copied to the location where the client you are running gets started.
- `sas.core.jar` - included in case you do not already have one however, this will most likely not be needed.

Chapter 14 – Post-Installation Configuration for SAS/SHARE Software

User Authentication

You are required to complete the steps from the section “Configuring User Authentication” in the *Installation Instructions for the SAS 9.2 Foundation for UNIX Environments*. This allows SAS/SHARE software to authenticate a client’s identity and check a client’s authority to access resources.

System Configuration for the TCP/IP Communications Method

We suggest each SAS/SHARE server that runs on a network node be defined as a service in the file `/etc/services` or `/etc/inet/services` on that node. Each entry in this file associates a service name with the port number and protocol used by that service. An entry for a SAS/SHARE server has the following form:

```
<server name>      <port number>/tcp      # <comments>
```

The server name must be one to eight characters in length. The first character must be a letter or underscore; the remaining seven characters can include letters, digits, underscores, the dollar \$ sign, or the at @ sign.

An entry for a server whose name is `MKTSERV` might look like the following:

```
mktserv      5000/tcp      # SAS/SHARE server for Marketing and Sales
```

The server name is specified with the `SERVER=` option in the `LIBNAME` statement, in the `OPERATE`, and in the `SERVER` procedure. If a server name is not defined in the services file, you must specify “__<port#>”, two consecutive underscores followed by the port number (e.g., `server=__5012`).

Client Components

SAS/SHARE software includes client components that are used outside of your SAS 9.2 Foundation installation. The 9.2 SAS/SHARE client components are available from the SAS 9.2 Software Download site. The SAS/SHARE client components are described below.

SAS/SHARE Data Provider

The SAS/SHARE Data Provider enables you to access, update, and manipulate SAS data using OLE DB- and ADO-compliant applications on Windows platforms.

SAS ODBC Driver

The SAS ODBC Driver enables you to access, update, and manipulate SAS data from ODBC-compliant applications on Windows platforms.

SAS/SHARE Driver for JDBC

The SAS/SHARE Driver for JDBC enables you to write applets, applications, and servlets that access and update SAS data. The Java Tools package that includes the SAS/SHARE driver for JDBC also includes the SAS/CONNECT driver for Java. If you are writing Java programs using these interfaces, you may also want to use the tunnel feature. This optional feature can be used with the Java applets you write to solve some common configuration problems.

SAS/SHARE SQL Library for C

The SAS/SHARE SQL Library for C provides an application programming interface (API) that enables your applications to send SQL queries and statements through a SAS/SHARE server to data on remote hosts.

NLS Information

Sites that develop or support international applications that use SAS/SHARE software should refer to Chapter 9, “Post-Installation Configuration for National Language Support (NLS).”

Chapter 15 – Using Host Sort Routines

This chapter provides instructions for making host sort routines available to SAS 9.2. The only supported host sort routine is SyncSort. To use host sort routines with SAS 9.2, complete the following steps:

1. Install the host sort library on your system by following the instructions provided by the vendor. Ensure that the host sort routine works outside of SAS 9.2.
2. Make the host sort library available to SAS 9.2 by following the instructions in the following section, “Making Host Sort Routines Available.”
3. Submit an options statement in a SAS session to specify the host sort routine by following the instructions in the section “Using Host Sort Routines in a SAS Session.”

Note: For information on using host sort routines in a SAS session once they are available, please refer to the SAS 9.2 Companion for UNIX Environments.

Making Host Sort Routines Available

This section describes the system-specific instructions for making host sort routines available to SAS 9.2.

For AIX

Set the environment variable \$LIBPATH to the directory containing the host sort library. For example, if the directory is /usr/local/syncsort/lib, then add these lines to both !SASROOT/bin/sasenv_local and !SASROOT/bin/sasenv_local.ksh:

```
LIBPATH=/usr/local/syncsort/lib:$LIBPATH
export LIBPATH
```

Add this line to !SASROOT/bin/sasenv_local.csh:

```
setenv LIBPATH /usr/local/syncsort/lib:$LIBPATH
```

For Linux and Solaris

Set the environment variable \$LD_LIBRARY_PATH to the directory containing the host sort library. For example, if the directory is /usr/local/syncsort/lib, then add these lines to both !SASROOT/bin/sasenv_local and !SASROOT/bin/sasenv_local.ksh:

```
LD_LIBRARY_PATH=/usr/local/syncsort/lib:$LIBPATH
export LD_LIBRARY_PATH
```

Add this line to !SASROOT/bin/sasenv_local.csh:

```
setenv LD_LIBRARY_PATH /usr/local/syncsort/lib:$LIBPATH
```

For HP-UX

Set the environment variable \$SHLIB_PATH to the directory containing the host sort library. For example, if the directory is /usr/local/syncsort/lib, then add these lines to both !SASROOT/bin/sasenv_local and !SASROOT/bin/sasenv_local.ksh:

```
SHLIB_PATH=/usr/local/syncsort/lib:$LIBPATH
export SHLIB_PATH
```

Add this line to !SASROOT/bin/sasenv_local.csh:

```
setenv SHLIB_PATH /usr/local/syncsort/lib:$LIBPATH
```

Using Host Sort Routines in a SAS Session

Note: The options statements throughout this section specify the syntax to submit to the SAS System. You can also specify these options as command line options and options in the sasv8.cfg file. Refer to the SAS Companion for UNIX Environments for more information on setting options.

Use the SORTNAME option to tell the SAS System which host sort routine should be used. Submit one of the following options statements in a SAS session:

- To use SyncSort (the default):

```
OPTIONS SORTNAME=SYNCSORT;
```
- To use CoSORT:

```
OPTIONS SORTNAME=COSORT;
```

Once the host sort routine is available, use the SORTPGM=HOST or SORTPGM=BEST options statements to tell the SAS System when to use the host sort routine.

Submit one of the following options statements in a SAS session:

- ```
OPTIONS SORTPGM=HOST;
```

  
tells the SAS System to always use the host sort routine made available.
- ```
OPTIONS SORTPGM=BEST;
```


tells the SAS System to choose the best sorting method in a given situation, the SAS System sort or the host sort.

There are two options that define how the SAS System chooses the “best” sort algorithm. The following examples use the syntax of an options statement that needs to be submitted to the SAS System:

- ```
-sortcut <n>
```

, where *n* specifies a number of observations.  

```
OPTIONS SORTPGM=BEST SORTCUT=500;
```

```
-sortcut
```

 tells the SAS System to choose the host sort routine if the number of observations is greater than the number you specify, and to use the SAS System sort if the number of observations is equal to or less than the number specified.
- ```
-sortcutp <size>[kKmM]
```

, where *<size>* specifies a file size in either kilobytes or megabytes.

```
OPTIONS SORTPGM=BEST SORTCUTP=40M;
```

`-sortcutp` tells the SAS System to choose the host sort routine if the size of the data being sorted exceeds the size you specify, and to use the SAS System sort if the size of the data is equal to or smaller than the size you specify.

If these options are not defined or these options are set to zero, the SAS System chooses the SAS System sort routine. If you specify both options and either condition is met, the SAS System chooses the host sort routine.

You can change the work directory used for temporary sort files by using the option `sortdev <dir>`, where `<dir>` is the directory in which you want the temporary files to be created. For example, submit the following statement if you want the temporary files to be created in `/tmp`:

```
OPTIONS SORTPGM=BEST SORTCUT=500 sortdev="/tmp";
```

You can specify the host sort option `sortanom t` to print timing and resource information to the SAS log after each phase of a sort. The following is an example of this option:

```
OPTIONS SORTPGM=HOST SORTANOM=t;
```

You can specify the host sort option `sortanom v` to print to the SAS log the arguments passed to the sort, which may be useful for tuning or debugging:

```
OPTIONS SORTPGM=HOST SORTANOM=v;
```

You can attempt to increase your sort performance by increasing the values of the `sortsize` and `memsize` SAS options. However, make sure that `sortsize` is at least 4M less than `memsize`.

You can see other SAS performance statistics in the SAS log using the `FULLSTIMER` option:

```
OPTIONS FULLSTIMER;
```




support.sas.com

SAS is the world leader in providing software and services that enable customers to transform data from all areas of their business into intelligence. SAS solutions help organizations make better, more informed decisions and maximize customer, supplier, and organizational relationships. For more than 30 years, SAS has been giving customers around the world The Power to Know®. Visit us at **www.sas.com**.