Changing the Trace Level .................................................. 51
Adjusting the SAS Embedded Process Performance .................. 51
Additional Configuration for All Hadoop Distributions ................. 54

PART 3 Administrator’s Guide for Teradata 55

Chapter 6 • In-Database Deployment Package for Teradata ................. 57
Prerequisites ......................................................................... 57
Overview of the In-Database Deployment Package for Teradata .......... 57
Teradata Permissions for Publishing Formats and Scoring Models ........ 58
Documentation for Using In-Database Processing in Teradata .......... 59

Chapter 7 • Deploying the In-Database Deployment Package: Teradata .......... 61
Teradata Installation and Configuration Steps ......................... 61
Upgrading from a Previous Version .................................. 62
Installing the SAS Formats Library and the SAS Embedded Process .... 65
Controlling the SAS Embedded Process ................................ 67

Chapter 8 • SAS Data Quality Accelerator for Teradata ................. 69
Introduction ......................................................................... 69
Installing SAS Data Quality Accelerator Stored Procedures in the Teradata Database 70
Obtaining and Deploying a QKB ........................................ 71
Validating the Accelerator Installation .................................. 74
Troubleshooting the Accelerator Installation .......................... 75
Updating and Customizing a QKB ...................................... 76
Removing SAS Data Quality Accelerator from the Teradata Database ... 77

PART 4 Administrator’s Guides for Aster, DB2, Greenplum, Netezza, Oracle, SAP HANA, and SPD Server 79

Chapter 9 • Administrator’s Guide for Aster .............................. 81
In-Database Deployment Package for Aster .............................. 81
Aster Installation and Configuration ........................................ 82
Validating the Publishing of the SAS_SCORE( ) and the SAS_PUT( ) Functions ... 85
Aster Permissions ................................................................. 85
Documentation for Using In-Database Processing in Aster ............ 86

Chapter 10 • Administrator’s Guide for DB2 ................................ 87
In-Database Deployment Package for DB2 ............................. 87
Function Publishing Process in DB2 ........................................ 88
DB2 Installation and Configuration ......................................... 89
Validating the Publishing of SAS_COMPILEUDF and SAS_DELETEUDF Functions and Global Variables ...................................... 106
DB2 Permissions ................................................................. 106
Documentation for Using In-Database Processing in DB2 ............ 108

Chapter 11 • Administrator’s Guide for Greenplum ...................... 109
In-Database Deployment Package for Greenplum ....................... 109
Greenplum Installation and Configuration .................................. 111
Validation of Publishing Functions ........................................ 123
Controlling the SAS Embedded Process ................................ 124
Semaphore Requirements When Using the SAS Embedded Process for Greenplum . 125
Greenplum Permissions ................................................................. 126
Documentation for Using In-Database Processing in Greenplum ........... 126

Chapter 12 • Administrator’s Guide for Netezza ................................. 127
In-Database Deployment Package for Netezza .................................. 127
Function Publishing Process in Netezza ......................................... 128
Netezza Installation and Configuration ........................................... 129
Netezza Permissions ................................................................. 140
Documentation for Using In-Database Processing in Netezza ............ 141

Chapter 13 • Administrator’s Guide for Oracle ................................. 143
In-Database Deployment Package for Oracle .................................... 143
Oracle Installation and Configuration ............................................. 144
Oracle Permissions ................................................................. 147
Documentation for Using In-Database Processing in Oracle ............... 148

Chapter 14 • Administrator’s Guide for SAP HANA ............................. 149
In-Database Deployment Package for SAP HANA .............................. 149
SAP HANA Installation and Configuration ....................................... 150
Installing the SASLINK AFL Plugins on the Appliance (SPS11) ......... 154
Importing the SAS_EP Stored Procedure ....................................... 155
Auxiliary Wrapper Procedures (SPS11) ......................................... 155
Controlling the SAS Embedded Process ....................................... 156
Semaphore Requirements When Using the SAS Embedded Process for SAP HANA ......................................................... 156
SAP HANA Permissions .............................................................. 157
Documentation for Using In-Database Processing in SAP HANA ....... 158

Chapter 15 • Administrator’s Guide for SPD Server ............................. 159
Installation and Configuration Requirements for the SAS Scoring
Accelerator for SPD Server .......................................................... 159
Where to Go from Here .............................................................. 159

PART 5 Configurations for SAS Model Manager ................................. 161

Chapter 16 • Configuring SAS Model Manager ................................. 163
Preparing a Data Management System for Use with SAS Model Manager ................................................................. 163
Configuring a Database .............................................................. 164
Configuring a Hadoop Distributed File System ................................ 167

PART 6 Upgrading from or Re-installing a Previous Version
of the Hadoop In-Database Deployment Package .............................. 169

Appendix 1 • Upgrading from or Re-installing a Previous Version If Using
SAS Deployment Manager ........................................................... 171
Upgrading from SAS 9.3 .............................................................. 171
Upgrading from SAS 9.4 before the July 2015 Release of SAS 9.4 ............ 172
Upgrading from or Re-installing from the July 2015 Release of SAS 9.4 and Later ................................................................. 173
Appendix 2 • Upgrading from or Re-installing a Previous Version If Using Manual Deployment ................................................................. 177
  Upgrading from or Re-installing from SAS 9.3 ........................................... 177
  Upgrading from or Re-installing from SAS 9.4 before the July 2015 Release of SAS 9.4 .......................................................... 178
  Upgrading from or Re-installing from the July 2015 Release of SAS 9.4 or Later . . . 179

Recommended Reading .............................................................. 181
Index ........................................................................ 183
What’s New in SAS 9.4 In-Database Products: Administrator’s Guide

Overview

In SAS 9.4, the following new features and enhancements were added to expand the capabilities of the SAS In-Database products:

• In the December 2017 release of SAS 9.4, new versions of the SAS Embedded Process for Hadoop and Teradata are available. The SAS Embedded Process for other data sources did not change.

• In the September 2017 release of SAS 9.4, the following changes and enhancements were made:
  • The SAS Deployment Manager can now be run if the cluster manager is enabled with Transport Layer Security (TLS). The SAS Deployment Manager validates if the Certificate of Authority (CA) exists for the host that you are trying to access. If the CA does not exist, a warning message is issued and asks to run another SAS Deployment Manager task to add the CA.

  Note: All discussion of TLS is also applicable to the predecessor protocol, Secure Sockets Layer (SSL).

  • If you license SAS Data Loader for Hadoop, you can now deploy the SAS Embedded Process and the SAS Data Quality Accelerator, the SAS Data Loader for Hadoop Spark Engine, and the SAS Quality Knowledge Base individually, all together, or in any combination at one time.

• In the November 2016 release of SAS 9.4, the following changes and enhancements were made:
  • The installation and configuration of the SAS Embedded Process for Aster, DB2, Greenplum, Netezza, Oracle, and SAP has changed. The in-database deployment package is delivered to the client from the SAS Install Depot. The new process has a smaller client footprint and is a faster install process.

  • If you use the SAS In-Database Code Accelerator for Hadoop, the SPD Engine SerDe can be used to access Hive tables.

  • Several new properties have been added to the SAS Embedded Process for Hadoop cp-config.xml file that enable you to adjust performance.

  • Deploying the SAS Embedded Process for Hadoop no longer requires root or sudo access.

  • Changes have been made to the sasep-admin.sh SAS Embedded Process for Hadoop deployment script.
• SAS in-database processing in Teradata now supports single sign-on (SSO) authentication.

• IBM BigInsights and Pivotal HD now support the Ambari server cluster manager. You can now use the SAS Deployment Manager to install the in-database deployment package for Hadoop on IBM BigInsights and Pivotal HD.

• In the January 2016 release of SAS 9.4, the following changes and enhancements were made:
  • The removal of the SAS Embedded Process stack using Ambari has been simplified. The delete_stack.sh file now enables you to remove the ep-config.xml file, a specific version of the SAS Embedded Process, or all versions of the SAS Embedded Process.
  • A new panel that lists the products being installed by the SAS Deployment Manager has been added. Information that appears on other panels also reflects more closely what is being installed.

• In the July 2015 release of SAS 9.4, the following changes and enhancements were made:
  • The installation and configuration of the SAS Embedded Process for Teradata has changed. The in-database deployment package is delivered to the client from the SAS Install Depot. The new process has a smaller client footprint and is a faster install process.
  • The installation and configuration of the SAS Embedded Process for Hadoop has changed significantly. For Cloudera and Hortonworks, Cloudera Manager and Ambari are used to install the SAS Embedded Process and the SAS Hadoop MapReduce JAR files. For IBM BigInsights, MapR, and Pivotal HD, the in-database deployment package is delivered to the client from the SAS Install Depot. In addition, the SAS Embedded Process and the SAS Hadoop MapReduce JAR files are installed with one script instead of in two separate scripts. The new process has a smaller client footprint and is a faster install process.

• In the August 2014 release of SAS 9.4, the following changes and enhancements were made:
  • Numerous changes were made to the installation and configuration script for the SAS Embedded Process for Hadoop.

• In the April 2014 release of SAS 9.4, documentation enhancements were made in the following areas:
  • Additional information about the installation and configuration of the SAS Embedded Process for Hadoop was added.
  • Added semaphore requirements when using the SAS Embedded Process for Greenplum.

• In the December 2013 release of SAS 9.4, the following changes and enhancements were made:
  • New Hadoop JAR files are now tied to the version of Apache Hadoop that you are using.

• In the June 2013 release of SAS 9.4, the following changes and enhancements were made:
  • In-database scoring for Netezza has been enhanced by the addition of the SAS Embedded Process. The SAS Embedded Process is a SAS server process that runs within Netezza to read and write data.
• The Hadoop scripts that install, control, and provide a status of the SAS Embedded Process have changed. There is now just one script, sasep-server.sh, that installs both the SAS Embedded Process and the Hadoop JAR files.

SAS In-Database Code Accelerator

November 2016 Release of SAS 9.4: Changes and Enhancements

If you use the SAS In-Database Code Accelerator for Hadoop, the SPD Engine SerDe can be used to access Hive tables.

SAS 9.4: Changes and Enhancements

The SAS In-Database Code Accelerator must be licensed at your site.

Aster Changes

November 2016 Release of SAS 9.4: Changes and Enhancements

The installation and configuration of the SAS Embedded Process for Aster has changed. The in-database deployment package is delivered to the client from the SAS Install Depot. The new process has a smaller client footprint and is a faster install process.

DB2 Changes

November 2016 Release of SAS 9.4: Changes and Enhancements

The installation and configuration of the SAS Embedded Process for DB2 has changed. The in-database deployment package is delivered to the client from the SAS Install Depot. The new process has a smaller client footprint and is a faster install process.
Greenplum Changes

November 2016 Release of SAS 9.4: Changes and Enhancements

The installation and configuration of the SAS Embedded Process for Greenplum has changed. The in-database deployment package is delivered to the client from the SAS Install Depot. The new process has a smaller client footprint and is a faster install process.

April 2014 Release of SAS 9.4: Changes and Enhancements

Information about semaphore requirements when using the SAS Embedded Process was added to SAS In-Database Products: Administrator’s Guide.

SAS 9.4: Changes and Enhancements

There are several changes for Greenplum:

• Version 1.2 of the Greenplum Partner Connector (GPPC) is now available and should be installed if you use SAS Embedded Process 9.4.

• A new script, UninstallSASEPFiles.sh, is available. This script stops and uninstalls the SAS Embedded Process on each database host node.

Hadoop Changes

November 2016 Release of SAS 9.4: Changes and Enhancements

In the November 2016 release of SAS 9.4, the following changes and enhancements were made:

• The number of SAS Embedded Process JAR files that are installed during deployment has been reduced from six to one.

• Several new properties have been added to the SAS Embedded Process ep-config.xml file that enable you to adjust performance.

• If you license SAS Data Loader for Hadoop, SAS Data Quality Accelerator for Teradata, or SAS Contextual Analysis In-Database Scoring for Hadoop, the Data Quality Accelerator for Hadoop, the Data Quality Accelerator for Teradata, and the SAS Contextual Analysis In-Database Scoring for Hadoop components, respectively, are included in the in-database deployment package for Hadoop and Teradata.

• Deploying the SAS Embedded Process for Hadoop no longer requires root or sudo access.
• IBM BigInsights and Pivotal HD now support the Ambari server cluster manager. You can now use the SAS Deployment Manager to install the in-database deployment package for Hadoop on IBM BigInsights and Pivotal HD.

• The sasep-admin.sh script has been changed:
  • A new option, -x, that enables you to deploy the SAS Embedded Process for Hadoop without root or sudo access has been added.
  • The -genconfig and -log options have been deleted.

January 2016 Release of SAS 9.4: Changes and Enhancements

In the January 2016 release of SAS 9.4, the following changes and enhancements were made:

• The removal of the SAS Embedded Process stack using Ambari has been simplified. The delete_stack.sh file now enables you to remove the ep-config.xml file, a specific version of the SAS Embedded Process, or all versions of the SAS Embedded Process.

• A new panel that lists the products being installed by the SAS Deployment Manager has been added. Information that appears on other panels also reflects more closely what is being installed.

July 2015 Release of SAS 9.4: Changes and Enhancements

The installation and configuration of the SAS Embedded Process for Hadoop has changed.

• For Cloudera and Hortonworks, Cloudera Manager and Ambari are used to install the SAS Embedded Process and the SAS Hadoop MapReduce JAR files.

• For IBM BigInsights, MapR, and Pivotal HD, the in-database deployment package is delivered to the client from the SAS Install Depot.

• The SAS Embedded Process and the SAS Hadoop MapReduce JAR files are installed with one script instead of in two separate scripts. The new process has a smaller client footprint and is a faster install process.

• The sasep-servers.sh file has changed names to the sasep-admin.sh file. Some of the scripts arguments are no longer needed and have been deleted. Other arguments have been added.

August 2014 Release of SAS 9.4: Changes and Enhancements

In the August 2014 release of SAS 9.4, the following changes and enhancements were made:

• Instead of manually selecting the Hadoop JAR files to the client machine, the SAS Embedded Process determines which version of the JAR files are required and gathers them into a ZIP file for you to copy to the client machine.

• You now have the option whether to automatically start the SAS Embedded Process when the installation is complete.
April 2014 Release of SAS 9.4: Changes and Enhancements

The documentation about the installation and configuration of the SAS Embedded Process was enhanced.

December 2013 Release of SAS 9.4: Changes and Enhancements

In the December 2013 release of SAS 9.4, the following changes and enhancements were made:

• The trace log messages for the SAS Embedded Process are now stored in the MapReduce job log.

• A new option, hdfsuser, is available in the sasep-servers.sh script. hdfsuser specifies the user ID that has Write access to HDFS root directory.

• The Cloudera JAR files for the SAS Embedded Process have been replaced by a set of Apache JAR files. The new JAR files are based on a release of the Apache Hadoop instead of a particular Hadoop distributor.

SAS 9.4: Changes and Enhancements

The Hadoop scripts that install, control, and provide a status of the SAS Embedded Process have changed. There is now just one script, sasep-servers.sh, that installs both the SAS Embedded Process and the Hadoop JAR files. Running this script also enables you to start, stop, and provide a status of the SAS Embedded Process.

Netezza Changes

November 2016 Release of SAS 9.4: Changes and Enhancements

The installation and configuration of the SAS Embedded Process for Netezza has changed. The in-database deployment package is delivered to the client from the SAS Install Depot. The new process has a smaller client footprint and is a faster install process.

July 2015 Release of SAS 9.4: Changes and Enhancements

The SAS Embedded Process for Netezza has a new cartridge file that creates the NZRC database.

SAS 9.4: Changes and Enhancements

In-database scoring for Netezza has been enhanced by the addition of the SAS Embedded Process. The SAS Embedded Process is a SAS server process that runs within Netezza to read and write data. The SAS Embedded Process can be used with the SAS Scoring Accelerator for Netezza to run scoring models.
Oracle Changes

November 2016 Release of SAS 9.4: Changes and Enhancements

The installation and configuration of the SAS Embedded Process for Oracle has changed. The in-database deployment package is delivered to the client from the SAS Install Depot. The new process has a smaller client footprint and is a faster install process.

SAP HANA Changes

November 2016 Release of SAS 9.4: Changes and Enhancements

The installation and configuration of the SAS Embedded Process for SAP has changed. The in-database deployment package is delivered to the client from the SAS Install Depot. The new process has a smaller client footprint and is a faster install process.

July 2015 Release of SAS 9.4: Changes and Enhancements

If you are using SAP HANA SPS9, the SAS Embedded Process for SAP HANA must be manually started. For previous versions, the SAS Embedded Process was automatically started by the SASAFL plug-in. In addition, a different procedure must be used to deploy the SASAFL plug-in.

Teradata Changes

November 2016 Release of SAS 9.4: Changes and Enhancements

SAS in-database processing in Teradata now supports single sign-on (SSO) authentication.

July 2015 Release of SAS 9.4: Changes and Enhancements

The installation and configuration of the SAS Embedded Process for Teradata has changed. The in-database deployment package is delivered to the client from the SAS Install Depot. The new process has a smaller client footprint and is a faster install process.
SAS Deployment Manager

September 2017 Release of SAS 9.4: Changes and Enhancements

In the September 2017 of SAS 9.4, the following changes and enhancements were made:

• The SAS Deployment Manager can now be run if the cluster manager is enabled with Transport Layer Security (TLS). The SAS Deployment Manager validates if the Certificate of Authority (CA) exists for the host that you are trying to access. If the CA does not exist, a warning message is issued and asks to run another SAS Deployment Manager task to add the CA. There is also a SAS Deployment Manager task to remove the CA.

  Note: All discussion of TLS is also applicable to the predecessor protocol, Secure Sockets Layer (SSL).

• If you license SAS Data Loader for Hadoop, you can now deploy the SAS Embedded Process and the SAS Data Quality Accelerator, the SAS Data Loader for Hadoop Spark Engine, and the SAS Quality Knowledge Base individually, all together, or in any combination at one time.
Part 1

Introduction

Chapter 1
Introduction to the Administrator’s Guide
Chapter 1
Introduction to the Administrator’s Guide

Overview of SAS In-Database Products
SAS in-database products integrate SAS solutions, SAS analytic processes, and third-party database management systems. Using SAS in-database technology, you can run scoring models, some SAS procedures, DS2 threaded programs, and formatted SQL queries inside the database. When using conventional processing, all rows of data are returned from the database to SAS. When using SAS in-database technology, processing is done inside the database and thus does not require the transfer of data.

To perform in-database processing, the following SAS products require additional installation and configuration:

• SAS/ACCESS Interface to Aster, SAS/ACCESS Interface to DB2, SAS/ACCESS Interface to Greenplum, SAS/ACCESS Interface to Hadoop, SAS/ACCESS Interface to Netezza, SAS/ACCESS Interface to Oracle, SAS/ACCESS Interface to SAP HANA, and SAS/ACCESS Interface to Teradata

  The SAS/ACCESS interfaces to the individual databases include components that are required for both format publishing to the database and for running some Base SAS procedures inside the database.

• SAS Scoring Accelerator for Aster, SAS Scoring Accelerator for DB2, SAS Scoring Accelerator for Greenplum, SAS Scoring Accelerator for Hadoop, SAS Scoring Accelerator for Netezza, SAS Scoring Accelerator for Oracle, SAS Scoring Accelerator for SAP HANA, and SAS Scoring Accelerator for Teradata

• SAS In-Database Code Accelerator for Greenplum, SAS In-Database Code Accelerator for Hadoop, and SAS In-Database Code Accelerator for Teradata

• SAS Contextual Analysis In-Database Scoring for Hadoop

• SAS Analytics Accelerator for Teradata

• SAS Data Loader for Hadoop

• SAS Data Quality Accelerator for Teradata

• SAS Model Manager In-Database Scoring Scripts

Note: The SAS Scoring Accelerator for SPD Server does not require any additional installation or configuration.
What Is Covered in This Document?

This document provides detailed instructions for installing and configuring the components that are needed for in-database processing using the SAS/ACCESS Interface, the SAS Scoring Accelerator, the SAS Analytics Accelerator, the SAS Data Quality Accelerator for Teradata, and the SAS In-Database Code Accelerator. These components are contained in a deployment package that is specific for your database.

The name and version of the in-database deployment packages are as follows:

- SAS Embedded Process for Aster 9.4
- SAS Formats Library for DB2 3.1
- SAS Embedded Process for DB2 9.4
- SAS Formats Library for Greenplum 3.1
- SAS Embedded Process for Greenplum 9.4
- SAS Embedded Process for Hadoop 9.4
- SAS Formats Library for Netezza 3.1
- SAS Embedded Process for Oracle 9.4
- SAS Embedded Process for SAP HANA 9.4
- SAS Formats Library for Teradata 3.1
- SAS Embedded Process for Teradata 9.4

If you use SAS Model Manager for in-database scoring with DB2, Greenplum, Hadoop, Netezza, or Teradata, additional configuration tasks are needed. This document provides detailed instructions for configuring a database for use with SAS Model Manager.

This document is intended for the system administrator, the database administrator, or both. It is expected that you work closely with the SAS programmers who use these products.

This document is divided by database management systems.
Part 2

Administrator’s Guide for Hadoop

Chapter 2
In-Database Deployment Package for Hadoop ........................... 7

Chapter 3
Deploying the In-Database Deployment Package
Using the SAS Deployment Manager ................................. 13

Chapter 4
Deploying the In-Database Deployment Package Manually .......... 33

Chapter 5
Additional Configuration for the SAS Embedded Process ........... 47
Introduction to the In-Database Deployment Package for Hadoop

The in-database deployment package for Hadoop must be installed and configured on your Hadoop cluster before you can perform the following tasks:

- Run a scoring model in Hadoop Distributed File System (HDFS) using the SAS Scoring Accelerator for Hadoop.
  
  For more information about using the scoring publishing macros, see the *SAS In-Database Products: User’s Guide*.

- Run DATA step scoring programs in Hadoop.
  
  For more information, see the *SAS In-Database Products: User’s Guide*.

- Run DS2 threaded programs in Hadoop using the SAS In-Database Code Accelerator for Hadoop.
  
  For more information, see the *SAS In-Database Products: User’s Guide*.

- Perform data quality operations in Hadoop, transform data in Hadoop, and extract transformed data out of Hadoop for analysis in SAS using the SAS Data Loader for Hadoop.
  
  For more information, see *SAS Data Loader for Hadoop: User’s Guide*.

- Deploy and score text analytic models in Hadoop using SAS Contextual Analysis In-Database Scoring in Hadoop.
Overview of the In-Database Deployment Package for Hadoop

The in-database deployment package for Hadoop includes the SAS Embedded Process and the SAS Hadoop MapReduce JAR files.

If you license either of the following SAS products, additional SAS software components are also included in the in-database deployment package for Hadoop.

<table>
<thead>
<tr>
<th>SAS Product Licensed</th>
<th>SAS Component Installed with the SAS Embedded Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAS Data Loader for Hadoop 3.1</td>
<td>SAS Data Quality Accelerator for Hadoop</td>
</tr>
<tr>
<td>SAS Contextual Analysis In-Database Scoring for Hadoop 14.3</td>
<td>SAS Contextual Analysis In-Database Scoring for Hadoop</td>
</tr>
</tbody>
</table>

*Note:* The deployment instructions that follow refer only to the SAS Embedded Process. However, if you have licensed one or more of the products listed above, the SAS software component is also deployed.

Overview of the SAS Embedded Process

The SAS Embedded Process runs within MapReduce to read and write data. The SAS Embedded Process runs on your Hadoop system where the data lives.

There are two methods by which you can deploy the SAS Embedded Process:

- SAS Deployment Manager
  
  If you are using Cloudera, Hortonworks, IBM BigInsights, or Pivotal HD, you can use the SAS Deployment Manager to deploy the SAS Embedded Process to the Hadoop cluster.

  For more information, see Chapter 3, “Deploying the In-Database Deployment Package Using the SAS Deployment Manager,” on page 13.

- SAS Embedded Process install script
Prerequisites for Installing the In-Database Deployment Package for Hadoop

By default, the SAS Embedded Process install script (sasep-admin.sh) discovers the cluster topology and installs the SAS Embedded Process on all DataNode nodes, including the host node from where you run the script (the Hadoop master NameNode). This occurs even if a DataNode is not present. If you want to add the SAS Embedded Process to new nodes at a later time, you can run the sasep-admin.sh script with the `-host <hosts>` option.

For distributions that are running MapReduce 1, the SAS Hadoop MapReduce JAR files are required in the `hadoop/lib` directory. For distributions that are running MapReduce 2, the SAS Hadoop MapReduce JAR files are in the `EPInstallDir/SASEPHome/jars/` directory.

For more information, see Chapter 4, “Deploying the In-Database Deployment Package Manually,” on page 33.

Prerequisites for Installing the In-Database Deployment Package for Hadoop

The following prerequisites are required before you install and configure the in-database deployment package for Hadoop:

- The required Hadoop JAR and configuration files are available to the SAS client machine.

Depending on your SAS software, there are several ways these JAR and configuration files are gathered. Gathering the JAR and configuration files is a one-time process (unless you are updating your cluster or changing Hadoop vendors). If you have already gathered the Hadoop JAR and configuration files for another SAS component, you do not need to do it again.

For more information on obtaining the JAR and configuration files, see the following documentation, depending on your SAS software:

- SAS Hadoop Configuration Guide for Base SAS and SAS/ACCESS
- SAS Data Loader for Hadoop: Installation and Configuration Guide
- SAS Contextual Analysis In-Database Scoring in Hadoop: Administrator's Guide
- SAS Viya: Deployment Guide
- SAS/ACCESS Interface to Hadoop has been configured.

For more information, see SAS Hadoop Configuration Guide for Base SAS and SAS/ACCESS at SAS 9.4 Support for Hadoop.

- You have working knowledge of the Hadoop vendor distribution that you are using (for example, Cloudera or Hortonworks).

You also need working knowledge of the Hadoop Distributed File System (HDFS), MapReduce 1, MapReduce 2, YARN, Hive, and HiveServer2 services. For more information, see the Apache website or the vendor’s website.

- Ensure that the HCatalog, HDFS, Hive, MapReduce, Oozie, Sqoop, and YARN services are running on the Hadoop cluster. The SAS Embedded Process does not necessarily use these services. However, other SAS software that relies on the SAS Embedded Process might use these various services. This ensures that the appropriate JAR files are gathered during the configuration.
• The SAS in-database and high-performance analytic products require a specific version of the Hadoop distribution. For more information, see the SAS Foundation system requirements documentation for your operating environment.

• The master node needs to connect to the slave nodes using passwordless SSH. For more information, see the Linux manual pages on ssh-keygen and ssh-copy-id.

• You understand and can verify your security setup. This includes Kerberos, Knox, Ranger, and Sentry RecordService.

If your cluster is secured with Kerberos, you need the ability to get a Kerberos ticket. You also need to have knowledge of any additional security policies.

• You have permission to restart the Hadoop MapReduce service (only needed for backward compatibility with SAS 9.4M2 or SAS 9.4M3 and MapReduce 1).

---

**Backward Compatibility**

Starting with the July 2015 release of SAS 9.4, the required location of the SAS Embedded Process JAR files and whether MapReduce service must be restarted during installation of the in-database deployment package for Hadoop depends on what version of the SAS client is being used.

The following table explains the differences.

<table>
<thead>
<tr>
<th>SAS Client Version</th>
<th>What Version of MapReduce is Running?</th>
<th>Where Do My SAS Hadoop MapReduce JAR Files Need to be Located?</th>
<th>Is Restart of MapReduce Required?*</th>
<th>Is Use of -link or -linklib Required During Installation?**</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.4M3</td>
<td>MapReduce 2</td>
<td>SASEPHOME/jars</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>9.4M3</td>
<td>MapReduce 1</td>
<td>hadoop/lib</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>9.4M2</td>
<td>MapReduce 2</td>
<td>hadoop/lib</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>9.4M2</td>
<td>MapReduce 1</td>
<td>hadoop/lib</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

* See Step 7 in “Installing the SAS Embedded Process” on page 36.

** See “SASEP-ADMIN.SH Script” on page 40.

---

**Hadoop Permissions**

The installation of the in-database deployment package for Hadoop involves writing a configuration file to HDFS and deploying files on all data nodes. These tasks require the following permissions:

• Writing the configuration file requires Write permission to HDFS.

• Deploying files across all nodes requires passwordless SSH from the master node to the slave nodes.
For information about using in-database processing in Hadoop, see the following publications:

- FILENAME Statement, Hadoop Access Method in *SAS DATA Step Statements: Reference*
- High-performance procedures in various SAS publications
- PROC HADOOP and PROC HDMD in *Base SAS Procedures Guide*
- SAS/ACCESS Interface to Hadoop in *SAS/ACCESS for Relational Databases: Reference*
- *SAS Contextual Analysis In-Database Scoring in Hadoop: User’s Guide*
- *SAS Data Integration Studio: User’s Guide*
- *SAS Data Loader for Hadoop: User’s Guide*
- *SAS High-Performance Analytics Infrastructure: Installation and Configuration Guide*
- *SAS In-Database Products: User’s Guide*
- *SAS Intelligence Platform: Data Administration Guide*
When to Deploy the SAS In-Database Deployment Package Using the SAS Deployment Manager

You can use the SAS Deployment Manager to deploy the SAS In-Database Deployment Package if the following conditions are met:

- For Cloudera:
  - You are using Cloudera 5.8 or later. For the latest information, see the SAS Foundation system requirements documentation for your operating environment.
  - Cloudera Manager is installed.
  - Your other SAS software, such as Base SAS and SAS/ACCESS Interface to Hadoop, was installed on a UNIX server.

- For Hortonworks, IBM BigInsights, or Pivotal HD:
  - You are using Hortonworks 2.5, IBM BigInsights 4.2, or Pivotal HD 3.0 or later. For the latest information, see the SAS Foundation system requirements documentation for your operating environment.
  - You are using Ambari 2.4 or later.
Your other SAS software, such as Base SAS and SAS/ACCESS Interface to Hadoop, was installed on a UNIX server. Otherwise, you should deploy the SAS In-Database deployment package manually. For more information, see Chapter 4, “Deploying the In-Database Deployment Package Manually,” on page 33.

**CAUTION:**

Once you have chosen a deployment method, you should continue to use that same deployment method when upgrading or redeploying SAS software on the cluster. Otherwise, your SAS software on the cluster can become unusable. For example, if you use the SAS Deployment Manager to deploy the SAS software on the cluster, you should continue to use the SAS Deployment Manager for upgrades or redeployments. You should not use the manual deployment method to upgrade or redeploy. If you do need to change deployment methods, you must first uninstall the SAS software on the cluster using the same method that you used to deploy it. You can then use the other deployment method to install it.

---

**Prerequisites for Using the SAS Deployment Manager to Deploy the In-Database Deployment Package**

The following prerequisites must be met before you can use the SAS Deployment Manager:

- You must have passwordless SSH access from the master node to the slave nodes.
- If your cluster is secured with Kerberos, in addition to having a valid ticket on the client, a Kerberos ticket must be valid on node that is running Hive. This is the node that you specify when using the SAS Deployment Manager.
- If you are using Cloudera, the SSH account must have Write permission to these directories:

  ```
  /opt/cloudera
  /opt/cloudera/csd
  /opt/cloudera/parcels
  ```

- You cannot customize the install location of the SAS Embedded Process on the cluster. By default, the SAS Deployment Manager deploys the SAS Embedded Process in the `/opt/cloudera/parcels` directory for Cloudera and the `/opt/sasep_stack` directory for Hortonworks, IBM BigInsights, and Pivotal HD.
- If you are using Cloudera, the Java JAR and GZIP commands must be available.
- If the following conditions are true, the Ambari server must be restarted after deployment:
  - You are using Hortonworks.
  - The requiretty option is enabled.
  - The SAS Embedded Process is installed using the SAS Deployment Manager.

Otherwise, the SASEP service does not appear in the Ambari list of services. It is recommended that you disable the requiretty option until the deployment is complete.
The following information is required:

- host name and port of the cluster manager
- credentials (account name and password) for the Hadoop cluster manager
- Hive service host name
- Oozie service host name (if required by your software)
- Impala service host name (if required by your software)
- credentials of the UNIX user account with SSH for the Hadoop cluster manager
- If the Hortonworks cluster is Kerberos enabled, the admin principal and password.

**Hadoop Installation and Configuration Steps Using the SAS Deployment Manager**

To install and configure Hadoop using the SAS Deployment Manager, you must follow and complete these steps:

**Note:** If you are running both SAS 9.4 and SAS Viya, only the latest version of the SAS Embedded Process should be installed. For example, assume that you installed the SAS Embedded Process using the sepcorehadp-13.00000-1.sh file. Now you want to install SAS Viya and the in-database deployment package that shipped with SAS Viya, which is sepcorehadp-14.00000-1.sh. In this instance, you would install the in-database deployment package that shipped with SAS Viya because it is a later version (that is, 14.00000 versus 13.00000).

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Where to Go for Information</th>
</tr>
</thead>
</table>
| 1    | Review these topics: | • “Prerequisites for Installing the In-Database Deployment Package for Hadoop” on page 9  
• “Backward Compatibility” on page 10  
• “When to Deploy the SAS In-Database Deployment Package Using the SAS Deployment Manager” on page 13  
• “Prerequisites for Using the SAS Deployment Manager to Deploy the In-Database Deployment Package” on page 14 |
<p>| 2    | If you have not already done so, configure SAS/ACCESS Interface to Hadoop. One of the key tasks in this step is to configure the Hadoop client files. | “Configuring SAS/ACCESS for Hadoop” in SAS Hadoop Configuration Guide for Base SAS and SAS/ACCESS |
| 3    | If you are upgrading from or re-installing a previous release, follow these instructions: | Appendix 1, “Upgrading from or Re-installing a Previous Version If Using SAS Deployment Manager,” on page 171 |
| 4    | Deploy the SAS Embedded Process parcel (Cloudera) or stack (Hortonworks, IBM BigInsights, or Pivotal HD) to the cluster. | For more information, see “Using the SAS Deployment Manager to Deploy the SAS Embedded Process Parcel or Stack to the Cluster” on page 16 |</p>
<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Where to Go for Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Deploy the parcel (Cloudera) or stack (Hortonworks, IBM BigInsights, or Pivotal HD) to all the nodes on the cluster.</td>
<td>For more information see “Deploying the SAS Embedded Process Parcel on Cloudera” on page 28 or “Deploying the SAS Embedded Process Stack on Hortonworks, IBM BigInsights, or Pivotal HD” on page 30.</td>
</tr>
<tr>
<td>6</td>
<td>Review any additional configuration that might be needed depending on your Hadoop distribution.</td>
<td>For more information, see Chapter 5, “Additional Configuration for the SAS Embedded Process,” on page 47.</td>
</tr>
</tbody>
</table>

### Using the SAS Deployment Manager to Deploy the SAS Embedded Process Parcel or Stack to the Cluster

1. Start the SAS Deployment Manager by running sasdm.sh for UNIX. The SAS Deployment Manager script is located in the `/SASHome/SASDeploymentManager/9.4/` directory.

   ```bash
   ./sasdm.sh
   ```

   *Note:* For more information about the SAS Deployment Manager pages, click **Help** on each page.

   The **Choose Language** page opens.

2. Select the language that you want to use to perform the configuration of your software.

   Click **OK**. The **Select SAS Deployment Manager Task** page opens.
3. Under Hadoop Configuration, select **Deploy SAS In-Database Technologies for Hadoop**.

Click **Next** to continue. The **Select Hadoop Distribution** page opens.

**Note:** If you have licensed and downloaded SAS Contextual Analysis In-Database Scoring for Hadoop, the SAS Contextual Analysis In-Database Scoring for Hadoop component is silently deployed at the same time as the SAS Embedded Process for Hadoop.
4. From the drop-down menu, select the distribution of Hadoop that you are using.

   Note: If your distribution is not listed, exit the SAS Deployment Manager and contact SAS Technical Support.

   Click Next. The Hadoop Cluster Manager Information page opens.
5. Enter the following information:

- Enter the host name and port number for your Hadoop cluster manager.

  For Cloudera, enter the location where Cloudera Manager is running. For Hortonworks, IBM BigInsights, or Pivotal, enter the location where the Ambari server is running.

  The port number is set to the appropriate default after Cloudera, Hortonworks, IBM BigInsights, or Pivotal is selected in Step 4 on page 18.

  Note: The host name must be a fully qualified domain name. The port number must be valid, and the cluster manager must be listening. The SAS Deployment Manager validates a constructed URL based on the host name and port number and it issues an error message if the URL is not accessible.

- Indicate whether the cluster manager is enabled with Transport Layer Security (TLS).

  Note: All discussion of TLS is also applicable to the predecessor protocol, Secure Sockets Layer (SSL).

  Note: If you choose Yes and a trusted certificate authority (CA) is not found for the host that you are trying to access, an error occurs. A dialog box that prompts you to run the SAS Deployment Manager Add Certificate to Trusted CA Bundle task is displayed. For more information about how run this task, see “Add Certificate to Trusted CA Bundle” in SAS Deployment Wizard and SAS Deployment Manager: User’s Guide. After you run this task
to add the certificate, you must restart the **Deploy SAS In-Database Technologies for Hadoop** task.

Click **Next**. The **Hadoop Cluster Manager Credentials** page opens.

![Hadoop Cluster Manager Credentials](image)

6. Enter the Cloudera Manager or Ambari administrator account name and password.

   *Note:* Using the credentials of the administrator account to query the Hadoop cluster and to find the Hive node eliminates guesswork and removes the chance of a configuration error. However, the account name does not have to be that of an administrator; it can be a read-only user.

   Click **Next**.

   The **Cluster Manager Host UNIX Account with SSH Information** page opens.
7. Specify whether you want to use a password or a private key file of the UNIX user account with SSH for the cluster manager host.

Click Next.

If you choose Yes, the UNIX User Account with SSH for the Hadoop Cluster Manager Host page opens that prompts you to browse for the private key file.
If you choose No, the **UNIX User Account with SSH for the Hadoop Cluster Manager Host** page opens and prompts you for a password.
8. Perform one of the following actions depending on whether you chose to enter a password or provide a private key file:

- If you chose No on the Cluster Manager Host UNIX Account with SSH Information panel, enter the root SSH account that has access to the cluster manager and password or enter a non-root SSH account if that account can execute sudo without entering a password.

  Note: For Cloudera, the SSH account must have Write permission to the /opt/cloudera directory. Otherwise, the deployment completes with errors.

- If you chose Yes on the Cluster Manager Host UNIX Account with SSH Information panel, click Browse and navigate to the location of the private key file.

Click Next. The Select SAS In-Database Technologies for Hadoop to deploy page opens.
Select which components to deploy on the Hadoop cluster.

You can choose one, two, or all three components. However, the SAS Data Quality Accelerator, the SAS Data Loader for Hadoop Spark Engine, and the SAS Quality Knowledge Base (QKB) are available only if you license SAS Data Loader for Hadoop.

Click Next. A page similar to the Specify the SAS Configuration and Deployment Directories page opens. What actually displays on this page depends on what components you choose to deploy.
10. Enter the location of the SAS configuration and deployment directories:

a. Enter (or navigate to) the location of the /standalone_installs directory. This directory was created when your SAS Software Depot was created by the SAS Download Manager.

   **CAUTION:**
   After installation, do not delete your SAS Software Depot /standalone_installs directory or any of its subdirectories. If hot fixes are made available for your software, they are moved to a subdirectory of the /standalone_installs/
   SAS_Core_Embedded_Process_Package_for_Hadoop/ directory. The SAS Deployment Manager requires that both the initial installation files and the hot fix file exist in a subdirectory of the original SAS Software Depot /standalone_installs/
   SAS_Core_Embedded_Process_Package_for_Hadoop/ directory.

b. (Optional) If you choose to deploy the SAS Quality Knowledge Base, enter (or navigate to) the location of the QKB directory, which can be found in the SAS Software Depot.

   **Note:** You obtain the SAS Quality Knowledge Base from your SAS Administrator. The SAS administrator identifies the SAS Quality Knowledge Base that SAS Data Loader for Hadoop should use for your site. For more information, see *SAS Data Loader for Hadoop: Installation and Configuration Guide.*
Note: A SAS Quality Knowledge Base must be deployed to enable data cleansing directives in SAS Data Loader for Hadoop.

c. Enter (or navigate to) a temporary work directory on the local server where the package or stack is placed. The working directory is removed when the deployment is complete.

Click Next. The Checking System page opens, and a check for locked files and Write permissions is performed.

11. If any files are shown in the text box after the system check, follow the instructions on the Checking System page to fix any problems.

Click Next. The Summary page opens.
12. Click **Start** to begin the configuration.

   **Note:** It takes time to complete the configuration. If your cluster is secured with Kerberos, it could take longer.

   **Note:** The product that appears on this page is the SAS product that is associated with the in-database deployment package for Hadoop. This package includes the SAS Embedded Process and possibly other components. Note that a separate license might be required to use the SAS Embedded Process and these other components.

   If the configuration is successful, the page title changes to **Deployment Complete** and a green check mark is displayed beside **SAS In-Database Technologies for Hadoop (64-bit)**.

   **Note:** Part of the configuration process runs SAS code to validate the environment. A green check mark indicates that the SAS Deployment Manager was able to create the SAS Embedded Process parcel or stack and then verify that the parcel or stack was copied to the cluster manager node.

   If warnings or errors occur, fix the issues and restart the configuration.

13. Click **Next** to close the SAS Deployment Manager.

   A log file is written to the %HOME%/SASAppData/SASDeploymentWizard directory on the client machine.

14. Continue the installation process.
Deploying the SAS Embedded Process Parcel on Cloudera

After you run the SAS Deployment Manager to create the SAS Embedded Process parcel, you must distribute and activate the parcel on the cluster. Follow these steps:

*Note:* More than one SAS Embedded Process parcel can be deployed on your cluster, but only one parcel can be activated at one time. Before activating a new parcel, deactivate the old one.

*Note:* If you have licensed and downloaded SAS Data Loader for Hadoop or SAS Contextual Analysis In-Database Scoring for Hadoop, other SAS components are silently deployed at the same time as the SAS Embedded Process for Hadoop. Other configuration is required as noted in step 8. For more information about what components are also deployed, see “Overview of the In-Database Deployment Package for Hadoop” on page 8.

1. Log on to Cloudera Manager.

2. Distribute the parcel to all nodes and create the SASEPHome directory.
   a. From the menu bar, choose **Hosts ⇒ Parcels**.
      The SASEP parcel is located under your cluster. An example name is p0.1.
      *Note:* If the SASEP parcel is missing, run **Check for new parcel**.
   b. On the row for the SASEP parcel, click **Distribute** to copy the parcel to all nodes and create the SASEPHome directory.
      *
   *CAUTION:* The SAS Embedded Process must be distributed on all nodes that are capable of running a MapReduce task (MapReduce 1) or on all nodes that are capable of running a YARN container (MapReduce 2). Hive and HCatalog must be available on all nodes where the SAS Embedded Process is distributed. Otherwise, the SAS Embedded Process does not function properly.
      You can log on to the node and show the contents in the /opt/cloudera/parcel directory.

3. Click **Activate**.
   This step creates a symbolic link to the SAS Hadoop JAR file.
   When prompted, click **Close**.

4. Add the SASEP service and create the SAS Embedded Process configuration file in HDFS.
   a. Navigate to the Cloudera Manager Home.
   b. In Cloudera Manager, select the drop-down arrow next to the name of the cluster, and then select **Add Service**.
      The **Add Service Wizard** page appears.
c. Select the SASEP service and click Continue.

d. On the Add Service Wizard ⇒ Select the set of dependencies for your new service page, select the dependencies for the service. Click Continue.

Note: The dependencies are automatically selected for this service.

e. On the Add Service Wizard ⇒ Customize Role Assignments page, select a node for the service.

Note: If your cluster is secured with Kerberos, in the next step, you must have a valid Kerberos ticket, so be sure to select a node that has a Kerberos ticket for the HDFS user on that node.

Choose any single node that is part of your cluster and where HDFS is a client.
Click OK and then click Continue. The Add a SASEP to Cluster cluster-name page appears.

f. Enter the name of the HDFS user. Click Continue and then click Finish.

Note: The default HDFS user name is hdfs. However, you can enter a custom HDFS user name.

Note: If your cluster is secured with Kerberos, the host that you select must have a valid ticket for the HDFS user.

The ep-config.xml file is created and added to the HDFS /sas/ep/config directory. This task is done in the host that you select.

g. After the SAS Embedded Process ep-config.xml file is created, Cloudera Manager starts the SAS Embedded Process service. This step is not required.
MapReduce is the only service that is required for the SAS Embedded Process.
The SAS Embedded Process service no longer needs to be stopped or started.

5. Verify that the ep-config.xml file exists in the /sas/ep/config directory of the host that you selected in step 4e.

6. Review any additional configuration that might be needed depending on your Hadoop distribution.

For more information, see Chapter 5, “Additional Configuration for the SAS Embedded Process,” on page 47.

7. Validate the deployment of the SAS Embedded Process by running a program that uses the SAS Embedded Process and the MapReduce service. An example is a scoring program.

8. If you have licensed and downloaded the following SAS software, additional configuration is required:

- SAS Contextual Analysis In-Database Scoring for Hadoop
  For more information, see SAS Contextual Analysis In-Database Scoring for Hadoop: Administrator’s Guide.

- SAS Data Loader for Hadoop
  For more information, see SAS Data Loader for Hadoop: Installation and Configuration Guide.

- SAS High-Performance Analytics
  For more information, see SAS High-Performance Analytics Infrastructure: Installation and Configuration Guide.
Deploying the SAS Embedded Process Stack on Hortonworks, IBM BigInsights, or Pivotal HD

**Deploying the SAS Embedded Process Stack for the First Time**

After you run the SAS Deployment Manager to create the SAS Embedded Process stack, you must deploy the stack on the cluster. Follow these steps:

*Note:* If the SAS Embedded Process stack already exists on your cluster, follow the instructions in “Deploying a New Version of the SAS Embedded Process Stack” on page 31.

*Note:* If you have licensed and downloaded SAS Data Loader for Hadoop or SAS Contextual Analysis In-Database Scoring for Hadoop, other SAS components are silently deployed at the same time as the SAS Embedded Process for Hadoop. Other configuration is required as noted in step 14. For more information about what components are also deployed, see “Overview of the In-Database Deployment Package for Hadoop” on page 8.

1. Log on to the machine that is hosting Ambari.
2. Start the Ambari server and log on.
3. If the requiretty option was enabled when you deployed the SAS Embedded Process, you must restart the Ambari server at this time. Otherwise, skip to step 4.
   a. Log on to the cluster.
      ```
      sudo - su
      ```
   b. Restart the Ambari server.
      ```
      ambari-server restart
      ```
   c. Start the Ambari server and log on.
4. Click **Actions** and choose **+ Add Service**.
   The **Add Service Wizard** page appears.
5. Select **Choose Services**.
   The **Choose Services** panel appears.
6. In the **Choose Services** panel, select **SASEP**. Click **Next**.
   The **Assign Slaves and Clients** panel appears.
7. In the **Assign Slaves and Clients** panel, select items under **Client** where you want the stack to be deployed.

**CAUTION:**

The SAS Embedded Process must be deployed on all nodes that are capable of running a MapReduce task (MapReduce 1) or on all nodes that are capable of running a YARN container (MapReduce 2). Hive and HCatalog must be available on all nodes where the SAS Embedded Process is deployed. Otherwise, the SAS Embedded Process does not function properly.

*Note:* You should always select NAMENODE as one of the clients and NAMENODE should have these two client components installed: HDFS_CLIENT and HCAT_CLIENT.
The **Customize Services** panel appears.

The SASEP stack is listed under **activated_version**. An example name is s0.1.

8. Do not change any settings on the **Customize Services** panel.

   **Note:** If your cluster is secured with Kerberos, the **Configure Identities** panel appears. Enter your Kerberos credentials in the **admin_principal** and **admin_password** text boxes.

   Click **Next**. The **Review** panel appears.

9. Review the information about the panel. If everything is correct, click **Deploy**.

   The **Install, Start, and Test** panel appears. After the SAS Embedded Process stack is installed on all nodes, click **Next**.

   The **Summary** panel appears.

10. Click **Complete**. The SAS Embedded Process stack is now installed on all nodes of the cluster.

   The **SASEP** service is displayed on the Ambari dashboard.

11. Verify that the SAS Embedded Process configuration file, ep-config.xml, exists in the **/sas/ep/config** directory.

12. Review any additional configuration that might be needed depending on your Hadoop distribution.

    For more information, see Chapter 5, “Additional Configuration for the SAS Embedded Process,” on page 47.

13. Validate the deployment of the SAS Embedded Process by running a program that uses the SAS Embedded Process and the MapReduce service. An example is a scoring program.

14. If you have licensed and downloaded the following SAS software, additional configuration is required:

    - SAS Contextual Analysis In-Database Scoring for Hadoop
      
      For more information, see *SAS Contextual Analysis In-Database Scoring for Hadoop: Administrator’s Guide*.
    
    - SAS Data Loader for Hadoop
      
      For more information, see *SAS Data Loader for Hadoop: Installation and Configuration Guide*.
    
    - SAS High-Performance Analytics
      
      For more information, see *SAS High-Performance Analytics Infrastructure: Installation and Configuration Guide*.

---

**Deploying a New Version of the SAS Embedded Process Stack**

More than one SAS Embedded Process stack can be deployed on your cluster, but only one stack can be activated at one time. After you run the SAS Deployment Manager to create the SAS Embedded Process stack, follow these steps to deploy an additional SAS Embedded Process stack when one already exists on your cluster.

1. Log on to the machine that is hosting Ambari.
2. Restart the Ambari server and log on to the Ambari manager.
3. Select **SASEP**.
   
   In the **Services** panel, a restart symbol appears next to **SASEP**. The **Configs** tab indicates that a restart is required.

4. Click **Restart**.

5. Click **Restart All**.
   
   After the service is restarted, the previous version of the SAS Embedded Process still appears in the **activated_version** text box on the **Configs** tab. All deployed versions of the SAS Embedded Process stack should appear in the **sasep_allversions** text box.

6. Refresh the browser.
   
   The new version of the SAS Embedded Process should now appear as the **activated_version** text box on the **Configs** tab.

If, at any time, you want to activate another version of the SAS Embedded Process stack, follow these steps:

1. Enter the version number in the **activated_version** text box on the **Configs** tab.

2. Click **Save**.

3. Add a note describing your action (for example, “Changed from version s01.1 to s01.2”), and click **Next**.

4. Click **Restart**.

5. Click **Restart All**.

6. Refresh Ambari.
   
   The new service is activated.

7. Review any additional configuration that might be needed depending on your Hadoop distribution.

   For more information, see Chapter 5, “Additional Configuration for the SAS Embedded Process,” on page 47.

8. Validate the deployment of the SAS Embedded Process by running a program that uses the SAS Embedded Process and the MapReduce service. An example is a scoring program.

9. If you have licensed and downloaded the following SAS software, additional configuration is required:
   
   - **SAS Contextual Analysis In-Database Scoring for Hadoop**
     
     For more information, see *SAS Contextual Analysis In-Database Scoring for Hadoop: Administrator’s Guide*.
   
   - **SAS Data Loader for Hadoop**
     
     For more information, see *SAS Data Loader for Hadoop: Installation and Configuration Guide*.
   
   - **SAS High-Performance Analytics**
     
     For more information, see *SAS High-Performance Analytics Infrastructure: Installation and Configuration Guide*. 
Chapter 4
Deploying the In-Database Deployment Package Manually

When to Deploy the SAS In-Database Deployment Package Manually

You should deploy the SAS In-Database deployment package manually in the following instances:

• Your Hadoop distribution is MapR.

• Your Hadoop distribution is Cloudera and any of the following is true:
  • Cloudera Manager is not installed.
  • You are not using Cloudera 5.8 or later.
  • Your other SAS software, such as Base SAS and SAS/ACCESS Interface to Hadoop, was installed on Windows. The SAS Deployment Manager cannot be used on a Windows client to install the SAS In-Database deployment package.

• Your Hadoop distribution is Hortonworks, IBM BigInsights, or Pivotal HD, and any of the following are true:
  • Ambari is not installed or you are using a version of Ambari prior to 2.4.
  • You are not using Hortonworks 2.5, IBM BigInsights 4.2, or Pivotal 3.0 or later.
  • Your other SAS software, such as Base SAS and SAS/ACCESS Interface to Hadoop, was installed on Windows. The SAS Deployment Manager cannot be used on a Windows client to install the SAS In-Database deployment package.
For more information, see Chapter 3, “Deploying the In-Database Deployment Package Using the SAS Deployment Manager,” on page 13.

CAUTION: Once you have chosen a deployment method, you should continue to use that same deployment method when upgrading or redeploying the SAS Embedded Process. Otherwise, the SAS Embedded Process can become unusable. For example, if you use the SAS Deployment Manager to deploy the SAS software on the cluster, you should continue to use the SAS Deployment Manager for upgrades or redeployments. You should not use the manual deployment method to upgrade or redeploy. If you do need to change deployment methods, you must first uninstall the SAS software on the cluster using the same method that you used to deploy it. You can then use the other deployment method to install it.

Hadoop Manual Installation and Configuration Steps

To install and configure Hadoop manually, you must follow and complete these steps:

Note: If you are running both SAS 9.4 and SAS Viya, only the latest version of the SAS Embedded Process should be installed. For example, assume that you installed the SAS Embedded Process using the sepcorehadp-13.00000-1.sh file. Now you want to install SAS Viya and the in-database deployment package that shipped with SAS Viya, which is sepcorehadp-14.00000-1.sh. In this instance, you would install the in-database deployment package that shipped with SAS Viya because it is a later version (that is, 14.00000 versus 13.00000).

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Where to Go for Information</th>
</tr>
</thead>
</table>
| 1    | Review these topics: | • “Prerequisites for Installing the In-Database Deployment Package for Hadoop” on page 9  
|      |             | • “Backward Compatibility” on page 10  
|      |             | • “When to Deploy the SAS In-Database Deployment Package Manually” on page 33 |
| 2    | If you have not already done so, configure SAS/ACCESS Interface to Hadoop. One of the key tasks in this step is to obtain the required Hadoop JAR and configuration files and make them available to the Hadoop cluster. Depending on your SAS software, there are several ways these JAR and configuration files are gathered. Gathering the JAR and configuration files is a one-time process (unless you are updating your cluster or changing Hadoop vendors). If you have already gathered the Hadoop JAR and configuration files for another SAS component, you do not need to do it again. | “Configuring SAS/ACCESS for Hadoop” in SAS Hadoop Configuration Guide for Base SAS and SAS/ACCESS  
For more information on obtaining the JAR and configuration files, see the following documentation depending on your SAS software:  
• SAS Hadoop Configuration Guide for Base SAS and SAS/ACCESS  
• SAS Data Loader for Hadoop: Installation and Configuration Guide  
• SAS Contextual Analysis In-Database Scoring for Hadoop: Administrator’s Guide |
Unzipping and Copying the In-Database Deployment Package to the Hadoop Cluster

Unzipping the In-Database Deployment Package for Hadoop

The in-database deployment package for Hadoop is contained in a self-extracting archive file named sepcorehadp-14.00000-1.sh. This file is contained in a ZIP file that is put in your SAS Software Depot directory.

To unzip the in-database deployment package for Hadoop, follow these steps:

1. Create a new temporary directory on your client machine, such as /sasep. The new directory is referred to as EPZipDir throughout this section.

2. Navigate to the YourSASDepot/standalone_installs directory.

   This directory was created when your SAS Software Depot was created by the SAS Download Manager.

3. Locate the en_sasexe.zip file. The en_sasexe.zip file is located in the following directory: YourSASDepot/standalone_installs/SAS_Core_Embedded_Process_Package_for_Hadoop/13_0/Hadoop_on_Linux_x64/.

   The sepcorehadp-14.00000-1.sh file is included in this ZIP file.

4. Unzip the en_sasexe.zip file.

   unzip en_sasexe.zip
After the file is unzipped, a `sasexe` directory is created in the same location as the `en_sasexe.zip` file. The `sepcorehadp-14.00000-1.sh` file is in the `sasexe` directory.

```
EPZipDir/sasexe/sepcorehadp-14.00000-1.sh
EPZipDir/sasexe/parcel
EPZipDir/sasexe/stack
```

**CAUTION:**

After installation, do not delete your SAS Software Depot `standalone_installs` directory or any of its subdirectories. If hot fixes are made available for your software, they are moved to a subdirectory of the `/standalone_installs/` directory. The SAS Deployment Manager requires that both the initial installation files and the hot fix file exist in a subdirectory of the original SAS Software Depot `/standalone_installs/` directory.

**Creating the SAS Embedded Process Directory**

Create a new directory on the Hadoop master node that is not part of an existing directory structure, such as `/sasep`.

This path is created on each node in the Hadoop cluster during the SAS Embedded Process installation. We do not recommend that you use existing system directories such as `/opt` or `/usr`. This new directory is referred to as `EPInstallDir`.

**Copying the SAS Embedded Process Install Script to the Hadoop Cluster**

To copy the SAS Embedded Process Install Script to the Hadoop Cluster, follow these steps:

1. Log on to the cluster master node using SSH.
   ```
   ssh username@serverhostname
   ```

2. Copy the `sepcorehadp-14.00000-1.sh` file to your `EPInstallDir` directory. This example uses secure copy.
   ```
   scp sepcorehadp-14.00000-1.sh username@hdpplus1:/EPInstallDir
   ```

   **Note:** The location where you transfer the `sepcorehadp-14.00000-1.sh` file becomes the SAS Embedded Process home and is referred to as `EPInstallDir`.

**Installing the SAS Embedded Process**

To install the SAS Embedded Process and SAS Hadoop Embedded Process JAR file, follow these steps:

**Note:** Permissions are needed to install the SAS Embedded Process and SAS Hadoop Embedded Process JAR file. For more information, see “Hadoop Permissions” on page 10.
1. Navigate to the location on your Hadoop master node where you copied the sepcorehdp-14.00000-1.sh file.
   
   cd /EPInstallDir
   
   For more information, see Step 2 in “Unzipping the In-Database Deployment Package for Hadoop” on page 35.

2. Ensure that both the EPInstallDir folder and the sepcorehdp-14.00000-1.sh file have Read, Write, and Execute permissions (chmod 755 -R).

3. Use the following command to unpack the sepcorehdp-14.00000-1.sh file.
   
   ./sepcorehdp-14.00000-1.sh <--verbose>
   
   Note: The -quiet option is enabled by default. Only error messages are displayed. The -verbose option causes all messages to be displayed that are generated during the installation process. Using verbose messaging can increase the time that is required to perform the installation.

   After this script is run and the files are unpacked, the following directory structure is created where EPInstallDir is the location on the master node from Step 2.

   EPInstallDir/sasexe/SASEPHome
   EPInstallDir/sasexe/sepcorehdp-14.00000-1.sh
   
   Note: During the install process, the sepcorehdp-14.00000-1.sh file is copied to all data nodes. Do not remove or move this file from the EPInstallDir/sasexe directory.

   The SASEPHome directory should have the following structure:

   EPInstallDir/sasexe/SASEPHome/bin
   EPInstallDir/sasexe/SASEPHome/install
   EPInstallDir/sasexe/SASEPHome/jars
   EPInstallDir/sasexe/SASEPHome/misc
   EPInstallDir/sasexe/SASEPHome/sasexe
   EPInstallDir/sasexe/SASEPHome/security
   EPInstallDir/sasexe/SASEPHome/utilities
   
   The EPInstallDir/SASEPHome/jars directory contains the SAS Hadoop Embedded Process JAR file.
   
   EPInstallDir/sasexe/SASEPHome/jars/sas.hadoop2.jar
   
   The EPInstallDir/SASEPHome/install directory contains install scripts for other SAS software that is packaged with the SAS Embedded Process. These files exist only if you have licensed this additional software. For more information about what components are also deployed, see “Overview of the In-Database Deployment Package for Hadoop” on page 8.

   The EPInstallDir/sasexe/SASEPHome/bin directory should contain the following script.
   
   EPInstallDir/sasexe/SASEPHome/bin/sasep-admin.sh

4. If your Hadoop cluster is secured with Kerberos and you have sudo access, the HDFS user must have a valid Kerberos ticket in order to access HDFS. You can obtain a valid Kerberos ticket with the kinit command.
   
   sudo su - root
   su - hdfs | hdfs-userid
   kinit -kt location-of-keytab-file-user-for-which-you-are-requesting-a-ticket principal-name
   exit
Note: For all Hadoop distributions except MapR, the default HDFS user is `hdfs`. For MapR distributions, the default HDFS user is `mapr`. You can specify a different user ID with the `-hdfsuser` argument when you run the `sasep-admin.sh -add` script. If you use a different HDFS superuser, ensure that the user has a home directory in HDFS before you run the `sasep-admin.sh -add` command. For example, if the HDFS superuser is `prodhdfs`, ensure that the `/user/prodhdfs` directory exists in HDFS.

**TIP**

To check the status of your Kerberos ticket on the server, as the HDFS user, run the `klist` command. Here is an example of the command and its output:

```
klist
Ticket cache: FILE/tmp/krb5cc_493
Default principal: hdfs@HOST.COMPANY.COM

Valid starting Expires Service principal
06/20/17 09:51:26 06/27/17 09:51:26 krbtgt/HOST.COMPANY.COM@HOST.COMPANY.COM
  renew until 06/22/17 09:51:26
```

5. Run the `sasep-admin.sh` script to deploy the SAS Embedded Process across all nodes. How you run the script depends on whether you have sudo access.

**Note:** It is recommended that the `sasep-admin.sh` script be run from the `EPInstallDir/sasexe/SASEPHome/bin/` location.

**TIP**

Many options are available for installing the SAS Embedded Process. We recommend that you review the script syntax before running it. For more information, see “SASEP-ADMIN.SH Syntax” on page 41.

- If you have sudo access, run the `sasep-admin.sh` script as follows to deploy SAS Embedded Process on all nodes. Review all of the information in this step and the script syntax before you run the script.

```
cd EPInstallDir/sasexe/SASEPHome/bin/
./sasep-admin.sh -add
```

If you have sudo access, the SAS Embedded Process install script (`sasep-admin.sh`) detects the Hadoop cluster topology and installs the SAS Embedded Process on all DataNode nodes. The install script also installs the SAS Embedded Process on the host node from which you run the script (the Hadoop master NameNode). The SAS Embedded Process is installed even if a DataNode is not present. To add the SAS Embedded Process to new nodes at a later time, run the `sasep-admin.sh` script with the `-host <hosts>` option.

In addition, a configuration file, `ep-config.xml`, is automatically created and written to the `EPInstallDir/SASEPHome/conf` directory and to the HDFS file system in the `/sas/ep/config` directory.

- If you do not have sudo access, follow these steps to deploy the SAS Embedded Process:

  1. Run the `sasep-admin.sh` script as follows to deploy the SAS Embedded Process across all nodes.

```
cd EPInstallDir/SASEPHome/bin/
./sasep-admin.sh -x -add -hostfile host-list-filename | -host "<host-list""
```

**CAUTION:**

The SAS Embedded Process must be installed on all nodes that are capable of running a MapReduce task (MapReduce 1) or on all nodes that are capable of running a YARN container (MapReduce 2). The SAS Embedded Process must also be installed on the host node from which
you run the script (the Hadoop master NameNode). Hive and HCatalog must be available on all nodes where the SAS Embedded Process is installed. Otherwise, the SAS Embedded Process does not function properly.

Note: If you do not have sudo access, you must use the -x option and specify the hosts on which the SAS Embedded Process is deployed with either the -hostfile or -host option. Automatic detection of the Hadoop cluster topology is not available when you run the installation script with the -x option.

The sepcorehadp-14.00000-1.sh file is copied to all nodes that you specify. The configuration file, ep-config.xml, is created and written to the EPInstallDir/SASEPHome/conf directory.

2. Manually copy the ep-config.xml configuration file to the HDFS.

Note: This step must be performed by a user that has Write permission to the HDFS root/ folder. If your Hadoop cluster is secured with Kerberos, the user who copies the configuration file to HDFS must have a valid Kerberos ticket.

i. Log on as your HDFS user or as the user that you use to access HDFS.

ii. Create the /sas/ep/config directory for the configuration file.

```
    hadoop fs -mkdir -p /sas/ep/config
```

iii. Navigate to the EPInstallDir/SASEPHome/conf directory.

```
    cd EPInstallDir/SASEPHome/conf
```

iv. Use the Hadoop copyFromLocal command to copy the ep-config.xml file to HDFS.

```
    hadoop fs -copyFromLocal ep-config.xml /sas/ep/config/ep-config.xml
```

6. Verify that the SAS Embedded Process is installed by running the sasep-admin.sh script with the -check option.

- If you ran the sasep-admin.sh script with sudo access, run the following command. By default, this command verifies that the SAS Embedded Process was installed on all nodes.

```
    cd EPInstallDir/sasexe/SASEPHome/bin/
    ./sasep-admin.sh -check
```

- If you ran the sasep-admin.sh script with the -x argument, run the following command. This command verifies that the SAS Embedded Process was installed on the hosts that you specified.

```
    cd EPInstallDir/sasexe/SASEPHome/bin/
    ./sasep-admin.sh ./sasep-admin.sh -x -check
    -hostfile host-list-filename | -host "*"\host-list<">
```

Note: The sasep-admin.sh -check script does not run successfully if the SAS Embedded Process is not installed.

7. If your distribution is running MapReduce 1, follow these steps. Otherwise, skip to Step 8.

   Note: For more information, see “Backward Compatibility” on page 10.

a. Verify that the sasep.hdp2.jar files are now in the hadoop/lib directory.

For Cloudera, the JAR files are typically located here:

```
    /opt/cloudera/parcels/CDH/lib/hadoop/lib
```
For Hortonworks, the JAR files are typically located here:

/usr/lib/hadoop/lib

b. Restart the Hadoop MapReduce service.

This enables the cluster to load the SAS Hadoop Embedded Process JAR file (sasep.hadp2.jar).

Note: It is preferable to restart the service by using Cloudera Manager or Ambari (for Hortonworks), if available.

8. Verify that the configuration file, ep-config.xml, was written to the HDFS file system.

hadoop fs -ls /sas/ep/config/
hadoop fs -cat /sas/ep/config/ep-config.xml

Note: If your cluster is secured with Kerberos, you need a valid Kerberos ticket in order to access HDFS. Otherwise, you can use the WebHDFS browser.

Note: The /sas/ep/config directory is created automatically when you run the install script with sudo access. If you used the -genconfig option to specify a non-default location, use that location to find the ep-config.xml file. When using a non-default location, a configuration property must be added to the mapred-site.xml configuration file that is used on the client side.

<property>
    <name>sas.ep.config.file</name>
    <value>config-file-location-on-hdfs</value>
</property>

The config-file-location-on-hdfs is the location of the SAS Embedded Process configuration file on HDFS.

---

SASEP-ADMIN.SH Script

**Overview of the SASEP-ADMIN.SH Script**

The sasep-admin.sh script enables you to perform the following actions.

- Install or uninstall the SAS Embedded Process and SAS Hadoop MapReduce JAR file on a single node or a group of nodes.
- Generate a SAS Embedded Process configuration file and write the file to an HDFS location.
- Install a hot fix to the SAS Embedded Process.
- Create a SAS Hadoop MapReduce JAR file symbolic link in the hadoop/lib directory (for backward compatibility only).
- Check whether the SAS Embedded Process is installed correctly.
- Display all live data nodes on the cluster.
- Display the Hadoop configuration environment.
- Display the Hadoop version information for the Hadoop cluster.
- Display the version of the SAS Embedded Process that is installed.
The installation of the SAS Embedded Process for Hadoop involves writing a configuration file to HDFS and deploying files on all data nodes. These two tasks can occur automatically, depending on your Hadoop and HDFS permissions.

If you run the SAS Embedded Process install script (sasep-admin.sh) with sudo access, the script detects the Hadoop cluster topology and installs the SAS Embedded Process on all DataNode nodes. The install script also installs the SAS Embedded Process on the host node on which you run the script (the Hadoop master NameNode). In addition, a configuration file, ep-config.xml, is created and written to the HDFS file system.

If you do not have sudo access, you must specify the hosts on which the SAS Embedded Process is installed. In addition, you must manually copy the ep-config.xml configuration file to the HDFS file system.

Note: The Hadoop master node must be able to connect to the Hadoop slave nodes using passwordless SSH on the cluster where the SAS Embedded Process is installed.

**SASEP-ADMIN.SH Syntax**

**Action Options syntax:**

```
sasep-admin.sh
  <x> -add <link> <epconfig config-filename> <maxsep number-of-copies>
  <hostfile host-list-filename> -host "host-list" <hdfsuser user-id>

sasep-admin.sh
  <x> -genconfig <HDFS-filename> <force>

sasep-admin.sh
  <x> -hotfix hotfix-filename <hostfile host-list-filename> -host "host-list"
  <hdfsuser user-id> <maxsep number-of-copies>

sasep-admin.sh
  <x> -remove <epconfig config-filename> <hostfile host-list-filename>
  -host "host-list" <hdfsuser user-id>

sasep-admin.sh
  <x> -linklib | -unlinklib

**Informational options syntax:**

```
sasep-admin.sh
  <x> <check> <hostfile host-list-filename> -host "host-list"
  <hdfsuser user-id>

sasep-admin.sh
  <env>

sasep-admin.sh
  <hadoopversion>

sasep-admin.sh
  <nodelist>

sasep-admin.sh
  <version>
```

**Action Arguments**
-add
installs the SAS Embedded Process.

Requirement
If you have sudo access, the script automatically retrieves the list of
data nodes from the Hadoop configuration. If you do not have sudo
access, you must use the -x argument and either the -hostfile or -host
argument.

Tip
If you add nodes to the cluster, you can specify the hosts on which
you want to install the SAS Embedded Process by using the -hostfile
or -host option. The -hostfile or -host options are mutually exclusive.

See
-hostfile and -host option on page 45

-genconfig <HDFS-filename> <-force>
generates a new SAS Embedded Process configuration file in the EPInstallDir/
SASEPHome/conf directory of the local file system.

Requirement
If you do not have sudo access, you must use the -x argument.

Interactions
When used without the -x argument, the script creates the ep-
config.xml configuration file and writes it to both the EPInstallDir/
SASEPHome/conf directory on the local file system and the /sas/ep/
config/ directory on HDFS. You can change the filename and HDFS
location by using the HDFS-filename argument. HDFS-filename must
be the fully qualified HDFS pathname where the configuration file is
located.

When used with the -x argument, the script does not write the
configuration file to HDFS. You must manually copy the file to
HDFS.

Note
The -genconfig argument creates two identical configuration files
under EPInstallDir/SASEPHome/conf on the local file system: ep-
config.xml and sasep-site.xml. The sasep-site.xml file might be
copied to the client side under a folder that is in the classpath. When
the sasep-site.xml file is loaded from the classpath, the configuration
file on the HDFS location is not used. However, if sasep-site.xml is
not found in the classpath, a configuration file must exist on HDFS,
either on the default HDFS location /sas/ep/config/ep-config.xml or
in the location that is set in the sas.ep.config.file property.

Tips
Use the -genconfig argument to generate a new SAS Embedded
Process configuration file if you upgrade your Hadoop installation,
you install or upgrade your Hive or HCatalog services, or you
upgrade the JDK or JRE that is used by the Hadoop processes.

This argument generates an updated ep-config.xml file. Use the -force
argument to overwrite the existing configuration file.

Use the HDFS-filename argument to specify another location and
configuration filename. If you decide to generate the configuration
file in a non-default HDFS location, you must set the
sas.ep.config.file property in the mapred-site.xml file to the value that
you specify in the -genconfig option.

See
“-epconfig config-filename” on page 45
-hotfix
distributes a hot fix package.

Requirements
Hot fixes must be installed using the same user ID who performed the initial software installation.

Hot fixes should be installed following the installation instructions provided by SAS Technical Support.

remove
removes the SAS Embedded Process.

CAUTION:
If you are using SAS Data Loader for Hadoop, you should remove the Quality Knowledge Base (QKB) and the SAS Data Management Accelerator for Spark from the Hadoop nodes before removing the SAS Embedded Process. Removing the SAS Embedded Process removes the scripts that are used to remove these products. For more information, see “Removing SAS Software from the Hadoop Cluster” in the SAS Data Loader for Hadoop: Installation and Configuration Guide.

Requirement
If you do not have sudo access, you must use the -x argument and either the -hostfile or -host argument. The -hostfile or -host options are mutually exclusive.

Interactions
When used without the -x argument and you have sudo access, the script automatically retrieves the list of data nodes from the Hadoop configuration. In addition, the script automatically removes the epconfig.xml file from HDFS.

When used with the -x argument, the SAS Embedded Process is removed from all hosts that you specify. However, the ep-config.xml file must be removed manually from HDFS.

See
-hostfile and -host option on page 45

-linklib
creates SAS Hadoop MapReduce JAR file symbolic links in the hadoop/lib folder.

Restriction
This argument should be used only for backward compatibility (that is, when you install the July 2015 release of SAS 9.4 of the SAS Embedded Process on a client that runs SAS 9.4M2).

Requirement
If you use this argument, you must restart the MapReduce service, the YARN service, or both after the SAS Embedded Process is installed.

Interaction
Use the -linklib argument after the SAS Embedded Process is already installed to create the symbolic links. Use the -link argument in conjunction with the -add argument to force the creation of the symbolic links.

See
“Backward Compatibility” on page 10
“-link” on page 45
-unlinklib
removes SAS Hadoop MapReduce JAR file symbolic links in the hadoop/lib folder.

Restriction  This argument should be used only for backward compatibility (that is, when you install the July 2015 release of SAS 9.4 of the SAS Embedded Process on a client that runs SAS 9.4M2).

Requirement  If you use this argument, you must restart the MapReduce service, the YARN service, or both after the SAS Embedded Process is installed.

See  “Backward Compatibility” on page 10

Informational Arguments

-check
checks whether the SAS Embedded Process is installed correctly on all data nodes.

Requirement  If you ran the sasep-admin.sh script with the -x argument, you must specify the hosts for which you want to check the SAS Embedded Process by using the -hostfile or -host option. The -hostfile or -host options are mutually exclusive.

See  -hostfile and -host option on page 45

-env
displays the SAS Embedded Process install script and the Hadoop configuration environment.

-hadoopversion
displays the Hadoop version information for the cluster.

-nodelist
displays all live DataNodes on the cluster.

Requirement  sudo access is required.

-version
displays the version of the SAS Embedded Process that is installed.

Parameters for Action and Informational Arguments

-x
if you do not have sudo access, runs the script solely under the current user’s credential.

Requirements  This option must be the first argument passed to the script.

A list of hosts must be provided with either the -hostfile or -host argument.

If you do not have sudo access, you must use the -x argument.

Interaction  If you use the -x argument to install the SAS Embedded Process, that is, with the -add argument, you must use the -x argument in any other sasep-admin.sh script action that supports it.

See  -hostfile and -host option on page 45
-link
forces the creation of SAS Hadoop MapReduce JAR files symbolic links in the hadoop/lib folder during the installation of the SAS Embedded Process.

Restriction
This argument should be used only for backward compatibility (that is, when you install the July 2015 release of SAS 9.4 of the SAS Embedded Process on a client that runs SAS 9.4M2).

Requirement
If you use this argument, you must restart the MapReduce service, the YARN service, or both after the SAS Embedded Process is installed.

Interactions
Use this argument in conjunction with the -add argument to force the creation of the symbolic links.

Use the -linklib argument after the SAS Embedded Process is already installed to create the symbolic links.

See
“Backward Compatibility” on page 10
“-linklib” on page 43

-epconfig config-filename
generates the SAS Embedded Process configuration file in the specified location.

Default
If the -epconfig argument is not specified, the install script creates the SAS Embedded Process configuration file in the default location /sas/ep/config/ep-config.xml.

Requirement
If the -epconfig argument is not specified a configuration file location must be provided. If you choose a non-default location, you must set the sas.ep.config.file property in the mapred-site.xml file that is on your client machine to the non-default location.

Interaction
Use the -epconfig argument in conjunction with the -add or -remove argument to specify the HDFS location of the configuration file.

-maxscp number-of-copies
specifies the maximum number of parallel copies between the master and data nodes.

Default
10

Interaction
Use this argument in conjunction with the -add or -hotfix argument.

-hostfile host-list-filename
specifies the full path of a file that contains the list of hosts where the SAS Embedded Process is installed or removed.

Requirement
The -hostfile or -host argument is required if you do not have sudo access.

Interaction
Use the -hostfile argument in conjunction with the -add, -hotfix, -check, or -remove arguments.

See
“-hdfsuser user-id” on page 46

Example
-hostfile
**-host <""> host-list <"">**

specifies the target host or host list where the SAS Embedded Process is installed or removed.

**Requirements**

If you specify more than one host, the hosts must be enclosed in double quotation marks and separated by spaces or commas.

The -host or -hostfile argument is required if you do not have sudo access.

**Interaction**

Use the -hostfile argument in conjunction with the -add, -hotfix, -check, or -remove arguments.

**See**

“-hdfsuser user-id” on page 46

**Example**

- host "server1 server2 server3"
- host bluesvr
- host "blue1, blue2, blue2"

**-hdfsuser user-id**

specifies the user ID that has Write access to HDFS root directory.

**Defaults**

hdfs for Cloudera, Hortonworks, Pivotal HD, and IBM BigInsights

mapr for MapR

**Interactions**

This argument has no affect if you use the -x argument.

Use the -hdfsuser argument in conjunction with the -add, -check, or -remove argument to change, check, or remove the HDFS user ID.

**Note**

The user ID is used to copy the SAS Embedded Process configuration files to HDFS.
Chapter 5
Additional Configuration for the SAS Embedded Process

Overview of Additional Configuration Tasks

After you have installed the SAS Embedded Process either manually or by using the SAS Deployment Manager, the following additional configuration tasks must be performed:

- “Additional Configuration Needed to Use HCatalog File Formats” on page 48.
- “Adding the YARN Application CLASSPATH to the Configuration File for MapR Distributions” on page 50.
- “Changing the Trace Level” on page 51.
- “Adjusting the SAS Embedded Process Performance” on page 51.
- “Additional Configuration for All Hadoop Distributions” on page 54.
Additional Configuration Needed to Use HCatalog File Formats

Overview of HCatalog File Types

HCatalog is a table management layer that presents a relational view of data in the HDFS to applications within the Hadoop ecosystem. With HCatalog, data structures that are registered in the Hive metastore, including SAS data, can be accessed through Hadoop code. HCatalog is part of Apache Hive.

The SAS Embedded Process for Hadoop uses HCatalog to process complex, non-delimited file formats.

Prerequisites for HCatalog Support

If you plan to access complex, non-delimited file types such as Avro or Parquet, the following conditions must be met:

- Hive must be installed on all nodes of the cluster.
- HCatalog support depends on the version of Hive that is running on your Hadoop distribution. See the following table for more information.

Note: For MapR distributions, Hive 0.13.0 build: 1501 or later must be installed for access to any HCatalog file type.

<table>
<thead>
<tr>
<th>File Type</th>
<th>Required Hive Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avro</td>
<td>0.14</td>
</tr>
<tr>
<td>ORC</td>
<td>0.11</td>
</tr>
<tr>
<td>Parquet</td>
<td>0.13</td>
</tr>
<tr>
<td>RCFile</td>
<td>0.6</td>
</tr>
</tbody>
</table>

SAS Client Configuration

Note: If you used the SAS Deployment Manager to install the SAS Embedded Process, these configuration tasks are not necessary. They were completed using the SAS Deployment Manager.

The following additional configuration tasks must be performed:

- The hive-site.xml configuration file must be in the SAS_HADOOP_CONFIG_PATH.
- The following Hive or HCatalog JAR files must be in the SAS_HADOOP_JAR_PATH.

hive-hcatalog-core-*.*.jar
hive-webhcat-java-client-*.jar
jdo-api*.jar

- If you are using MapR, the following Hive or HCatalog JAR files must be in the SAS_HADOOP_JAR_PATH.

  hive-hcatalog-hbase-storage-handler-0.13.0-mapr-1408.jar
  hive-hcatalog-server-extensions-0.13.0-mapr-1408.jar
  hive-hcatalog-pig-adapter-0.13.0-mapr-1408.jar
  datanucleus-api-jdo-3.2.6.jar
  datanucleus-core-3.2.10.jar
  datanucleus-rdbms-3.2.9.jar

- To access Avro file types, the avro-1.7.4.jar file must be added to the SAS_HADOOP_JAR_PATH environment variable.

- To access Parquet file types with Cloudera 5.1, the parquet-hadoop-bundle.jar file must be added to the directory defined in the SAS_HADOOP_JAR_PATH environment variable.

- If your distribution is running Hive 0.12, the jersey-client-1.9.jar must be added to the SAS_HADOOP_JAR_PATH environment variable.

For more information about the SAS_HADOOP_JAR_PATH and SAS_HADOOP_CONFIG_PATH environment variables, see the SAS Hadoop Configuration Guide for Base SAS and SAS/ACCESS.

**SAS Server-Side Configuration**

If your distribution is running MapReduce 2 and YARN, the SAS Embedded Process installation automatically sets the HCatalog CLASSPATH in the ep-config.xml file. Otherwise, you must manually include the HCatalog JAR files in either the MapReduce 2 library or the Hadoop CLASSPATH. For Hadoop distributions that run with MapReduce 1, you must also manually add the HCatalog CLASSPATH to the MapReduce CLASSPATH.

Here is an example for a Cloudera distribution.

```xml
<property>
  <name>mapreduce.application.classpath</name>
  <value>/EPInstallDir/SASEPHome/jars/sas.hadoop.ep.apache205.jar,/EPInstallDir/SASEPHome/jars/sas.hadoop.ep.apache205.nls.jar,/opt/cloudera/parcels/CDH-5.2.0-1.cdh5.2.0.p0.36/bin/../lib/hive/lib/*,/opt/cloudera/parcels/CDH-5.2.0-1.cdh5.2.0.p0.36/lib/hive-hcatalog/libexec/../share/hcatalog/*,/opt/cloudera/parcels/CDH-5.2.0-1.cdh5.2.0.p0.36/lib/hive-hcatalog/libexec/../share/hcatalog/storage-handlers/hbase/lib/*,/SAS_HADOOP_HADOOP_CLASSPATH/*</value>
</property>
```

Here is an example for a Hortonworks distribution.

```xml
<property>
  <name>mapreduce.application.classpath</name>
</property>
```
Additional Configuration for Hadoop 2.6 and SAS Windows Middle Tiers and SAS Windows Servers

If you are using Hadoop 2.6 or later, a Windows server, a Windows middle tier, or both, and HCatalog sources, you must add the HADOOP_HOME environment variable in the Windows server, the Windows middle tier, or both. An example of the value for this option is `HADOOP_HOME=c:\hadoop`.

The directory must contain a subdirectory named `bin`, which must contain the `winutils.exe` file for your distribution. Please contact your distribution vendor for a copy of the `winutils.exe` file.

Adding the YARN Application CLASSPATH to the Configuration File for MapR Distributions

On all of the nodes in your Hadoop cluster that have a `yarn-site.xml` file, two main configuration properties specify the application CLASSPATH: `yarn.application.classpath` and `mapreduce.application.classpath`. If you do not specify the YARN application CLASSPATH, MapR takes the default CLASSPATH. However, if you specify the MapReduce application CLASSPATH, the YARN application CLASSPATH is ignored. The SAS Embedded Process for Hadoop requires both the MapReduce application CLASSPATH and the YARN application CLASSPATH.

To ensure the existence of the YARN application CLASSPATH, you must manually add the YARN application CLASSPATH to the `yarn-site.xml` file. Without the manual definition in the configuration file, the MapReduce application master fails to start a container.

The default YARN application CLASSPATH for Linux is:

```bash
$HADOOP_CONF_DIR,  
$HADOOP_COMMON_HOME/share/hadoop/common/*,  
$HADOOP_COMMON_HOME/share/hadoop/common/lib/*,  
$HADOOP_HDFS_HOME/share/hadoop/hdfs/*,  
$HADOOP_HDFS_HOME/share/hadoop/hdfs/lib/*,  
$HADOOP_YARN_HOME/share/hadoop/yarn/*,  
$HADOOP_YARN_HOME/share/hadoop/yarn/lib/*  
```

The default YARN application CLASSPATH for Windows is:

```bash
%HADOOP_CONF_DIR%,  
%HADOOP_COMMON_HOME%/share/hadoop/common/*,  
%HADOOP_COMMON_HOME%/share/hadoop/common/lib/*,  
%HADOOP_HDFS_HOME%/share/hadoop/hdfs/*,  
%HADOOP_HDFS_HOME%/share/hadoop/hdfs/lib/*,  
%HADOOP_YARN_HOME%/share/hadoop/yarn/*,  
%HADOOP_YARN_HOME%/share/hadoop/yarn/lib/*  
```

**Note:** On MapR, the YARN application CLASSPATH does not resolve the symbols or variables specified in the paths (`$HADOOP_HDFS_HOME`, and so on).

**TIP** Anytime you make a change to the cluster, remember to restart the node managers to pick up the change.
Changing the Trace Level

You can modify the level of tracing by changing the value of the sas.ep.server.trace.level property in the ep-config.xml file. The default value is 4 (TRACE_NOTE).

```xml
<property>
  <name>sas.ep.server.trace.level</name>
  <value>trace-level</value>
</property>
```

The trace-level represents the level of trace that is produced by the SAS Embedded Process. trace-level can be one of the following values:

0
  TRACE_OFF
1
  TRACE_FATAL
2
  TRACE_ERROR
3
  TRACE_WARN
4
  TRACE_NOTE
5
  TRACE_INFO
10
  TRACE_ALL

Note: When using SAS Embedded Process with SAS High-Performance Analytics (HPA), tracing requires that an /opt/SAS directory exist on every node of the cluster where the SAS Embedded Process is installed. If the folder does not exist or does not have Write permission, the SAS Embedded Process job fails.

Adjusting the SAS Embedded Process Performance

You can adjust how the SAS Embedded Process runs by changing or adding its properties. SAS Embedded Process configuration properties can be added to the mapred-site.xml configuration on the client side or the sasep-site.xml file. If you change the properties in the ep-config.xml file located on HDFS, it will affect all SAS Embedded Process jobs. The ep-config.xml file is created when you install the SAS Embedded Process.

When using the SAS Embedded Process, there are a number of properties that you can adjust to improve performance.

- You can specify the number of SAS Embedded Process MapReduce tasks per node by changing the sas.ep.superreader.tasks.per.node property.
The SAS Embedded Process super reader technology does not use the standard MapReduce split calculation. Instead of assigning one split per task, it assigns many. The super reader calculates the splits, groups them, and distributes the groups to a configurable number of mapper tasks based on data locality.

The default number of tasks is 6.

```
<property>
    <name>sas.ep.superreader.tasks.per.node</name>
    <value>number-of-tasks</value>
</property>
```

- You can specify the number of concurrent nodes that are allowed to run High-Performance Analytics output tasks by changing the `sas.ep.hpa.output.concurrent.nodes` property.

  If this property is set to zero, the SAS Embedded Process allocates tasks on all nodes capable of running a YARN container. If this property is set to -1, the number of concurrent nodes equates to the number of High-Performance Analytics worker nodes. If the number of concurrent nodes exceeds the number of available nodes, the property value is adjusted to the number of available nodes. The default value is 0.

```
<property>
    <name>sas.ep.hpa.output.concurrent.nodes</name>
    <value>number-of-nodes</value>
</property>
```

- You can specify the number of High-Performance Analytics output tasks that are allowed to run per node by changing the `sas.ep.hpa.output.tasks.per.node` property.

  The default number of tasks is 1.

```
<property>
    <name>sas.ep.hpa.output.tasks.per.node</name>
    <value>number-of-tasks</value>
</property>
```

- You can specify the number of concurrent input reader threads.

  Each reader thread takes a file split from the input splits queue, opens the file, positions itself at the beginning of the split, and starts reading the records. Each record is stored on a native buffer that is shared with the DS2 container. When the native buffer is full, it is pushed to the DS2 container for processing. When a reader thread finishes reading a file split, it takes another file split from the input splits queue. The default number of input threads is 3.

```
<property>
    <name>sas.ep.input.threads</name>
    <value>number-of-input-threads</value>
</property>
```

- You can specify the number of output writer threads by changing the `sas.ep.output.threads` property.

  The SAS Embedded Process super writer technology improves performance by writing output data in parallel, producing multiple parts of the output file per mapper task. Each writer thread is responsible for writing one part of the output file. The default number of output threads is 2.

```
<property>
    <name>sas.ep.output.threads</name>
    <value>number-of-output-threads</value>
</property>
```
• You can specify the number of compute threads by changing the `sas.ep.compute.threads` property.

Each compute thread runs one instance of the DS2 program inside the SAS Embedded Process. The DS2 code that runs inside the DS2 container processes the records that it receives. At a given point, DS2 flushes output data to native buffers. The super writer threads take the output data from DS2 buffers and writes them to the super writer thread output file on a designated HDFS location. When all file input splits are processed and all output data is flushed and written to HDFS, the mapper task ends. The default number of compute threads is 1.

```xml
<property>
  <name>sas.ep.compute.threads</name>
  <value>number-of-threads</value>
</property>
```

• You can specify the number of buffers that are used for output data by changing the `sas.ep.output.buffers` property.

The number of output buffers should not be less than `sas.ep.compute.threads` plus `sas.ep.output.threads`. The default number of buffers is 3.

```xml
<property>
  <name>sas.ep.output.buffers</name>
  <value>number-of-buffers</value>
</property>
```

• You can specify the number of native buffers that are used to cache input data by changing the `sas.ep.input.buffers` property. The default value is 4. The number of input buffers should not be less than `sas.ep.compute.threads` plus `sas.ep.input.threads`.

```xml
<property>
  <name>sas.ep.input.buffers</name>
  <value>number-of-buffers</value>
</property>
```

• You can specify the optimal size of one input buffer by changing the `sas.ep.optimal.input.buffer.size` property.

The optimal row array size is calculated based on the optimal buffer size. The default value is 1 MB.

```xml
<property>
  <name>sas.ep.optimal.input.buffer.size.mb</name>
  <value>number-in-megabytes</value>
</property>
```

• You can specify the amount of memory in bytes that the SAS Embedded Process is allowed to use with MapReduce 1 by changing the `sas.ep.max.memory` property in the `ep-config.xml` file.

If your Hadoop distribution is running MapReduce 2, this value does not supersede the YARN maximum memory per task. Adjust the YARN container limit to change the amount of memory that the SAS Embedded Process is allowed to use.

The default value is 2147483647 bytes.

```xml
<property>
  <name>sas.ep.max.memory</name>
  <value>number-of-bytes</value>
</property>
```
Note: This property is valid only for Hadoop distributions that are running MapReduce 1.

Additional Configuration for All Hadoop Distributions

Depending on which in-database product that you are running on Base SAS with the SAS Embedded Process, you might get a java.lang.OutOfMemoryError: PermGen space error. This indicates that the Java Virtual Machine PermGen space is exhausted. The PermGen space is not part of the Java Heap memory. It is used to hold metadata that describes the classes and method used by an application.

The PermGen size can be increased to alleviate this issue by adding this line to JREOPTIONS in the SAS configuration file:

-jreoptions "(-XX:MaxPermSize=256m)"
Part 3

Administrator’s Guide for Teradata

Chapter 6

In-Database Deployment Package for Teradata ...................... 57

Chapter 7

Deploying the In-Database Deployment Package: Teradata ....... 61

Chapter 8

SAS Data Quality Accelerator for Teradata ......................... 69
Chapter 6
In-Database Deployment Package for Teradata

Prerequisites

SAS Foundation and the SAS/ACCESS Interface to Teradata must be installed before you install and configure the in-database deployment package for Teradata.

The SAS in-database and high-performance analytic products require a specific version of the Teradata client and server environment. For more information, see the SAS Foundation system requirements documentation for your operating environment.

The SAS Embedded Process installation requires approximately 200MB of disk space in the /opt file system on each Teradata TPA node.

Overview of the In-Database Deployment Package for Teradata

This section describes how to install and configure the in-database deployment package for Teradata (SAS Formats Library for Teradata and SAS Embedded Process). The in-database deployment packages for Teradata must be installed and configured before you can perform the following tasks:

- Use the %INDTD_PUBLISH_FORMATS format publishing macro to publish the SAS_PUT( ) function and to publish user-defined formats as format functions inside the database.
  
  For more information about using the format publishing macros, see the *SAS In-Database Products: User’s Guide*

- Use the %INDTD_PUBLISH_MODEL scoring publishing macro to publish scoring model files or functions inside the database.
For more information about using the scoring publishing macros, see the *SAS In-Database Products: User’s Guide*

- Use the SAS In-Database Code Accelerator for Teradata to execute DS2 thread programs in parallel inside the database.
  
  For more information, see the *SAS DS2 Language Reference*.

- Perform data quality operations in Teradata using the SAS Data Quality Accelerator for Teradata.
  
  For more information, see *SAS Data Quality Accelerator for Teradata: User’s Guide*.

  *Note:* If you are installing the SAS Data Quality Accelerator for Teradata, you must perform additional steps after you install the SAS Embedded Process. For more information, see Chapter 8, “SAS Data Quality Accelerator for Teradata,” on page 69.

- Run SAS High-Performance Analytics when the analytics cluster is using a parallel connection with a remote Teradata data appliance. The SAS Embedded Process, which resides on the data appliance, is used to provide high-speed parallel data transfer between the data appliance and the analytics environment where it is processed.
  
  For more information, see the *SAS High-Performance Analytics Infrastructure: Installation and Configuration Guide*.

The in-database deployment package for Teradata includes the SAS formats library and the SAS Embedded Process.

The SAS formats library is a run-time library that you install on your Teradata system. This installation is done so that the SAS scoring model functions or the SAS_PUT( ) function can access the routines within the run-time library. The SAS formats library contains the formats that are supplied by SAS.

  *Note:* The SAS formats library is not required by the SAS Data Quality Accelerator for Teradata.

The SAS Embedded Process is a SAS server process that runs within Teradata to read and write data. The SAS Embedded Process contains macros, run-time libraries, and other software that is installed on your Teradata system.

  *Note:* If you are performing a system expansion where additional nodes are being added, the version of the SAS formats library and the SAS Embedded Process on the new database nodes must be the same as the version that is being used on already existing nodes.

  *Note:* In addition to the in-database deployment package for Teradata, a set of SAS Embedded Process functions must be installed in the Teradata database. The SAS Embedded Process functions package is downloadable from Teradata. For more information, see “Installing the SAS Embedded Process Support Functions” on page 67.

---

**Teradata Permissions for Publishing Formats and Scoring Models**

Because functions are associated with a database, the functions inherit the access rights of that database. It might be useful to create a separate shared database for the SAS
scoring functions or the SAS_PUT() function so that access rights can be customized as needed.

You must grant the following permissions to any user who runs the scoring or format publishing macros:

CREATE FUNCTION ON database TO userid
DROP FUNCTION ON database TO userid
EXECUTE FUNCTION ON database TO userid
ALTER FUNCTION ON database TO userid

If you use the SAS Embedded Process to run your scoring model, you must grant the following permissions:

SELECT, CREATE TABLE, INSERT ON database TO userid
EXECUTE PROCEDURE ON SAS_SYSFNLIB TO userid
EXECUTE FUNCTION ON SAS_SYSFNLIB TO userid
EXECUTE FUNCTION ON SYSLIB.MonitorVirtualConfig TO userid

Note: If you plan to use SAS Model Manager with the SAS Scoring Accelerator for in-database scoring, additional permissions are required. For more information, see Chapter 16, "Configuring SAS Model Manager," on page 163.
Chapter 7
Deploying the In-Database Deployment Package: Teradata

Teradata Installation and Configuration Steps

1. If you are upgrading from a previous version, follow the instructions in “Upgrading from a Previous Version” on page 62.

2. Install the in-database deployment package.
   For more information, see “Installing the SAS Formats Library and the SAS Embedded Process” on page 65.

3. Install the SAS Embedded Process support functions.
   For more information, see “Installing the SAS Embedded Process Support Functions” on page 67.

Note: If you are using any of the following SAS Software, additional configuration is needed:

- If you plan to use SAS Model Manager with the SAS Scoring Accelerator for in-database scoring, perform the additional configuration tasks provided in Chapter 16, “Configuring SAS Model Manager,” on page 163.

- If you plan to use the SAS Data Quality Accelerator for Teradata, perform the additional configuration tasks provided in Chapter 8, “SAS Data Quality Accelerator for Teradata,” on page 69.
If you plan to use the SAS High-Performance Analytics environment, perform the additional configuration tasks provided in *SAS High-Performance Analytics Infrastructure: Installation and Configuration Guide*.

---

### Upgrading from a Previous Version

#### Upgrading from a Version That Was Installed Prior to SAS 9.4M3 (July 2015)

To upgrade from a previous version of the SAS Formats Library, the SAS Embedded Process, or both, follow these steps:

1. **Check the current installed version of the SAS formats library.**
   
   How you do this depends on the version of the SAS formats library.
   
   - If a SAS 9.2 version of the formats library is currently installed, run this command:
     
     `psh "rpm -qa* | grep jazxfbrs`  
     
     If a previous version is installed, a result similar to this is displayed. The version number might be different.
     
     `jazxfbrs-9.2-1.9`
   
   - If a SAS 9.3 or SAS 9.4 version of the formats library is currently installed, run this command:
     
     `psh "rpm -qa* | grep acc`  
     
     If a previous version is installed, a result similar to this is displayed. The version number might be different.
     
     `accelterafmt-3.1-1.x86_64`
   
   If the library is not installed on the Teradata nodes, no output is displayed. You can continue with the installation steps in “Installing the SAS Formats Library and the SAS Embedded Process” on page 65.

2. **Run this command to check the current installed version of the SAS Embedded Process.**
   
   `psh "rpm -qa* | grep tkindbsrv*`  
   
   If a previous version is installed, a result similar to this is displayed. The version number might be different.
   
   `tkindbsrv-9.42_M1-2.x86_64`
   
   If the SAS Embedded Process is not installed on the Teradata nodes, no output is displayed. You can continue with the installation steps in “Installing the SAS Formats Library and the SAS Embedded Process” on page 65.

3. **If a version of the SAS formats library, the SAS Embedded Process, or both is being installed that has a name that is different from the library that was previously installed, then follow these steps.** An example would be one of these:

   - `accelterafmt-3.1-1` replacing `jazxfbrs-9.2-1.6`
a. If you are upgrading the SAS Formats Library, shut down the Teradata database.
   
   \texttt{tpareset -y -x shutdown\_comment}
   
   This step is required because an older version of the SAS formats library might be loaded in a currently running SAS query.
   
   \textbf{Note:} If you are upgrading only the SAS Embedded Process (tkindbsrv.rpm file), you do not need to shut down the database. You do need to shut down the SAS Embedded Process. For more information about how to shut down the SAS Embedded Process, see “Controlling the SAS Embedded Process” on page 67.

b. Confirm that the database is shut down.
   
   \texttt{pdestate -a}
   
   DOWN/HARDSTOP is displayed if the database is shut down.

c. If the SAS Data Quality Accelerator for Teradata is installed, you must uninstall it before you uninstall the SAS Embedded Process. For more information, see “Removing SAS Data Quality Accelerator from the Teradata Database” on page 77.

d. Remove the old version of the in-database deployment package before you install the updated version.
   
   • To remove the packages from all nodes concurrently, run this command:
     
     \texttt{psh "rpm -e package\_name"}
     
     \textit{package-name} is either jazxfbrs.9.version, accelerafmt-version, or tkindbsrv-version.
     
     For example, to remove \texttt{jazxfbrs}, run the command \texttt{psh "rpm -e jazxfbrs-9.2-1.6"}.
   
   • To remove the package from each node, run this command on each node:
     
     \texttt{rpm -e package\_name}
     
     \textit{package-name} is either jazxfbrs.9.version, accelerafmt-version, or tkindbsrv-version.

4. (Optional) To confirm removal of the package before installing the new package, run this command:
   
   \texttt{psh "rpm -q package\_name"}
   
   \textit{package-name} is either jazxfbrs.9.version, accelerafmt-version, or tkindbsrv-version.
   
   The SAS Formats Library or the SAS Embedded Process should not appear on any node.

5. Continue with the installation steps in “Installing the SAS Formats Library and the SAS Embedded Process” on page 65.

\textbf{Upgrading Versions That Were Installed after SAS 9.4M3 (July 2015)}

To upgrade from a previous version of the SAS Formats Library, the SAS Embedded Process, or both, follow these steps:

1. Run this command to check the current installed version of the SAS formats library.
psh "rpm -qa" | grep acc

If a previous version is installed, a result similar to this is displayed. The version number might be different.

accelterafmt-3.1-1.x86_64

If the library is not installed on the Teradata nodes, no output is displayed. You can continue with the installation steps in “Installing the SAS Formats Library and the SAS Embedded Process” on page 65.

2. Run this command to check the current installed version of the SAS Embedded Process and any hot-fix packages.

psh "rpm -qa | grep sepcoretera*"

If a previous version is installed, a result similar to this is displayed. The version number might be different.

sepcoretera-12.00000-1
sepcoreterahf-12.00000-1

If the SAS Embedded Process is not installed on the Teradata nodes, no output is displayed. You can continue with the installation steps in “Installing the SAS Formats Library and the SAS Embedded Process” on page 65.

3. If the SAS Data Quality Accelerator for Teradata is installed, you must uninstall it before you uninstall the SAS Embedded Process. For more information, see “Removing SAS Data Quality Accelerator from the Teradata Database” on page 77.

4. If a version of the SAS formats library, the SAS Embedded Process, or both is being installed – and has a name that is different from the library that was previously installed – then follow these steps. An example is one of these:

• accelterafmt-3.1-1 replacing jazxfrbs-9.2-1.6
• sepcoretera-14.0000-1 replacing tkindbsrv-9.42_M1-2

Note: Starting with SAS 9.4M5, if the package-name is sepcoretera, the old sepcoretera package does not have to be removed before new sepcoretera package is installed. A sepcoreterahf package must always be removed before installing a new SAS Embedded Process package.

Note: You must remove both the SAS Embedded Process package (if it has a name other than sepcoretera) and any SAS Embedded Process hot-fix package that exists before updating to a new version of the SAS Embedded Process. If a hot fix is installed, it blocks the upgrade to a new SAS Embedded Process package. If you force the upgrade to the new package and then later remove the old hot fix, there is a risk that the old hot fix removal will also remove files delivered by the new SAS Embedded Process installation.

a. If you are upgrading the SAS Formats Library, shut down the Teradata database.

    tparest -y -x shutdown_comment

This step is required because an older version of the SAS formats library might be loaded in a currently running SAS query.

Note: If you are upgrading only the SAS Embedded Process (sepcoretera*.rpm file), you do not need to shut down the database. You do need to shut down the SAS Embedded Process. For more information about how to shut down the SAS Embedded Process, see “Controlling the SAS Embedded Process” on page 67.

b. Confirm that the database is shut down.
psh pdestate -a

DOWN/HARDSTOP is displayed if the database is shut down.

c. Remove the old version before you install the updated version of the in-database deployment package.
   • To remove the packages from all nodes concurrently, run this command:
     
psh "rpm -e package-name"
     
     `package-name` is either accelterafmt-version, tkindbsrv-version, or sepcoreterahf-version.
     
     For example, to remove `tkindbsrv`, run the command `psh "rpm -e tkindbsrv-9.42_M1-2"`.
     
   • To remove the package from each node, run this command on each node:
     
rpm -e package-name
     
     `package-name` is either accelterafmt-version, tkindbsrv-version, or sepcoreterahf-version.

5. (Optional) To confirm removal of the package before installing the new package, run this command:
   
psh "rpm -q package-name"
   
   `package-name` is either accelterafmt-version, sepcoretera-version, or sepcoreterahf-version.
   
The SAS Formats Library or the SAS Embedded Process should not appear on any node.

6. Continue with the installation steps in “Installing the SAS Formats Library and the SAS Embedded Process” on page 65.

---

**Installing the SAS Formats Library and the SAS Embedded Process**

**Moving the SAS Formats Library and the SAS Embedded Process Packages to the Server Machine**

1. Locate the SAS Formats Library for Teradata deployment package file, accelterafmt-3.1-`n`.x86_64.rpm. `n` is a number that indicates the latest version of the file. If this is the initial installation, `n` has a value of 1.

   The accelterafmt-3.1-`n`.x86_64.rpm file is located in the `YourSASDepot/SASFormatsLibraryforTeradata/3.1/TeradataonLinux/` directory.

   *Note:* The SAS formats library is not required by the SAS Data Quality Accelerator for Teradata.

2. Move the package file to your Teradata database server in a location where it is both Read and Write accessible. You need to move this package file to the server machine in accordance with procedures used at your site. Here is an example using secure copy.

   `scp acceterafmt-3.1-`n`.x86_64.rpm root@teramach1:/sasdir/18NOV16`
This package file is readable by the Teradata Parallel Upgrade Tool.

3. Locate the SAS Embedded Process deployment package file, sepcoretera-14.00000-1.x86_64.rpm. Follow these steps:

a. Navigate to the `YourSASDepot/standalone_installs` directory. This directory was created when you created your SAS Software Depot.

b. Locate the en_sasexe.zip file. The en_sasexe.zip file is located in the `YourSASDepot/standalone_installs/SAS_Core_Embedded_Process_Package_for_Teradata/14_0/Teradata_on_Linux/` directory.

The sepcoretera-14.00000-1.x86_64.rpm file is included in this ZIP file.

c. Copy the en_sasexe.zip file to a temporary directory on the server machine. You need to move this package file to the server machine in accordance with procedures used at your site. Here is an example using secure copy.

```
scp en_sasexe.zip root@teramach1:/SomeTempDir
```

d. Log on to the cluster and navigate to the temporary directory in Step 3c.

e. Unzip en_sasexe.zip.

After the file is unzipped, a `sasexe` directory is created in the same location as the en_sasexe.zip file. The sepcoretera-14.00000-1.x86_64.rpm file should be in the `/SomeTempDir/sasexe` directory.

4. Copy the sepcoretera-14.00000-1.x86_64.rpm file to the same location on the server as the accelterafmt-3.1-*.x86_64.rpm file in Step 2.

You need to move this package file to the server machine in accordance with the procedures used at your site. Here is an example using secure copy.

```
scp sepcoretera-14.00000-1.x86_64.rpm root@teramach1:/sasdir/18NOV17
```

This package file is readable by the Teradata Parallel Upgrade Tool.

---

### Installing the SAS Formats Library and the SAS Embedded Process with the Teradata Parallel Upgrade Tool

This installation should be performed by a Teradata systems administrator in collaboration with Teradata Customer Services. A Teradata Change Control is required when a package is added to the Teradata server. Teradata Customer Services has developed change control procedures for installing the SAS in-database deployment package.

The steps assume full knowledge of the Teradata Parallel Upgrade Tool and your environment. For more information about using the Teradata Parallel Upgrade Tool, see the *Parallel Upgrade Tool (PUT) Reference* which is at the Teradata Online Publications site, located at [http://www.info.teradata.com/GenSrch/eOnLine-Srch.cfm](http://www.info.teradata.com/GenSrch/eOnLine-Srch.cfm). On this page, search for “Parallel Upgrade Tool” and download the appropriate document for your system.

The following steps explain the basic steps to install the SAS formats library package by using the Teradata Parallel Upgrade Tool.

**Note:** The Teradata Parallel Upgrade Tool prompts are subject to change as Teradata enhances its software.

1. Locate the SAS Formats Library and the SAS Embedded Process packages on your server machine. They must be in a location where they can be accessed from at least
2. Start the Teradata Parallel Upgrade Tool.

3. Be sure to select all Teradata TPA nodes for installation, including Hot Stand-By nodes.

4. If Teradata Version Migration and Fallback (VM&F) is installed, you might be prompted whether to use VM&F or not. If you are prompted, choose Non-VM&F installation.

5. If the installation is successful, `accelerfmt-3.1-n` or `sepcoretera-14.00000-1.x86_64` is displayed. `n` is a number that indicates the latest version of the file.

   Alternatively, you can manually verify that the installation is successful by running these commands from the shell prompt.

   ```
   psh "rpm -q -a | grep accelerfmt"
   psh "rpm -q -a | grep sepcoretera"
   ```

**Installing the SAS Embedded Process Support Functions**

The SAS Embedded Process support function package (sasepfunc) includes stored procedures that generate SQL to interface with the SAS Embedded Process and functions that load the SAS program and other run-time control information into shared memory. The SAS Embedded Process support functions setup script creates the SAS_SYSFNLIB database and the SAS Embedded Process interface fast path functions in TD_SYSFNLIB.

In addition, the SAS Embedded Process support function package installs grant.sh, which helps an administrator grant permissions for the SAS Embedded Process. An administrator has to run the script from a machine with the Teradata BTEQ utility once for each SAS Embedded Process user.

The SAS Embedded Process support function package is available from the Teradata Software Server. For access to the package that includes the installation instructions, contact your local Teradata account representative or the Teradata consultant supporting your SAS and Teradata integration activities.

*Note:* If you are using SAS 9.4 Data Quality Accelerator, you must contact your Teradata representative to get access to version 15.00-8 or later of the SAS Embedded Process support functions (sasepfunc-15.00-8).

**Controlling the SAS Embedded Process**

The SAS Embedded Process starts when a query is submitted. The SAS Embedded Process continues to run until it is manually stopped or the database is shutdown. You might want to disable or shutdown the SAS Embedded Process without shutting down the database.

The following commands control the SAS Embedded Process.
<table>
<thead>
<tr>
<th>Action performed</th>
<th>Command (by Teradata version)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provides the status of the SAS Embedded Process.</td>
<td>CALL SQLJ.SERVERCONTROL ('SAS', 'status', :A);</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td>Shuts down the SAS Embedded Process.</td>
<td>CALL SQLJ.SERVERCONTROL ('SAS', 'shutdown', :A);</td>
</tr>
<tr>
<td>Note: You cannot shut down until all queries are</td>
<td></td>
</tr>
<tr>
<td>complete.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td>Stops new queries from being started. Queries that</td>
<td>CALL SQLJ.SERVERCONTROL ('SAS', 'disable', :A);</td>
</tr>
<tr>
<td>are currently running continue to run until they are</td>
<td></td>
</tr>
<tr>
<td>complete.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
<tr>
<td>Enables new queries to start running.</td>
<td>CALL SQLJ.SERVERCONTROL ('SAS', 'enable', :A);</td>
</tr>
<tr>
<td></td>
<td>*</td>
</tr>
</tbody>
</table>

* Note that the Languagename parameter, 'SAS', is required and must be uppercase. The Cmd parameter (for example, 'status'), must be lowercase.

Here is the sequence of operations to stop and then restart the SAS Embedded Process:

1. Disable the SAS Embedded Process to stop new queries from being started.
   
   CALL SQLJ.SERVERCONTROL ('SAS', 'disable', :A);

2. Query the status of the SAS Embedded Process until the status returns this message:
   
   Hybrid Server is disabled with no UDFs running.
   
   CALL SQLJ.SERVERCONTROL ('SAS', 'status', :A);

   
   CALL SQLJ.SERVERCONTROL ('SAS', 'shutdown', :A);

4. Perform maintenance on the SAS Embedded Process, for example, install a hot fix or upgrade to a new version.

5. Enable the SAS Embedded Process.
   
   CALL SQLJ.SERVERCONTROL ('SAS', 'enable', :A);

6. Test the SAS Embedded Process. The SAS Embedded Process will start when the next SAS query that uses the SAS Embedded Process is sent to the database.
Chapter 8
SAS Data Quality Accelerator for Teradata

Introduction

In order to use SAS data cleansing functionality inside the Teradata database, you must perform the following tasks after deploying the SAS In-Database Technologies for Teradata (SAS Embedded Process):

- install SAS data quality stored procedures in the Teradata database
- deploy a SAS Quality Knowledge Base (QKB) in the Teradata database

The SAS In-Database Technologies deployment provides shell scripts that enable you to install and manage the data quality stored procedures within the Teradata database. In addition, it contains a shell script that enables you to package the QKB for deployment inside the Teradata database.

The QKB is a collection of files that store data and logic that support data management operations. SAS software products reference the QKB when performing data management operations on your data.

Each Teradata node needs approximately 8 GB for the QKB.
Installing SAS Data Quality Accelerator Stored Procedures in the Teradata Database

Overview

The SAS in-database deployment package for Teradata (sepcoretera) installs three scripts in the `/opt/SAS/SASTKInDatabaseServerforTeradata/<version-number>/install/pgm` directory of the Teradata database server:

- a stored procedure creation script named `dq_install.sh`
- a user authorization script named `dq_grant.sh`
- a stored procedure removal script named `dq_uninstall.sh`

You will need to run the `dq_install.sh` script to create the data quality stored procedures in the Teradata database and the `dq_grant.sh` script to grant users permission to execute the data quality stored procedures.

The `dq_uninstall.sh` script is provided to enable you to remove the data quality stored procedures from the database. You must remove any data quality stored procedures that have already been installed from the Teradata database before upgrading or re-installing either SAS Data Quality Accelerator for Teradata or the SAS Embedded Process.

Note: All three scripts must be run as the root user.

Using the `dq_install.sh` Script

The `dq_grant.sh` shell script is provided to enable the Teradata system administrator to grant users authorization to the data quality stored procedures. The `dq_install.sh` script is located in the `/opt/SAS/SASTKInDatabaseServerforTeradata/<version-number>/install/pgm` directory of the Teradata database server.

The `dq_install.sh` script requires modification before it can be run. The Teradata administrator must edit the shell script to specify the site-specific Teradata server name and DBC user logon credentials for the `DBC_PASS=`, `DBC_SRVR=`, and `DBC_USER=` variables.

Running `dq_install.sh` puts the data quality stored procedures into the `SAS_SYSFNLIB` database and enables the accelerator functionality.

Here is the syntax for executing `dq_install.sh`:

```
./dq_install.sh <-l log-path>
```

**log-path**

specifies an alternative name and location for the `dq_install.sh` log. When this parameter is omitted, the script creates a file named `dq_install.log` in the current directory.

The next step in the installation is to grant users permission to execute the stored procedures.
Using the dq_grant.sh Script

The dq_grant.sh shell script is provided to enable the Teradata system administrator to
grant users authorization to the data quality stored procedures. The dq_grant.sh script is
located in the /opt/SAS/SASTKInDatabaseServerforTeradata/<version-number>/install/pgm
directory of the Teradata database server.

The dq_grant.sh script requires modification before it can be run. The Teradata
administrator must edit the shell script to specify the site-specific Teradata server name
and DBC user logon credentials for the DBC_SRVR=, DBC_USER=, and DBC_PASS=
variables.

Here is the syntax for executing dq_grant.sh:

    ./dq_grant.sh <-l log-path> user-name

log-path

    specifies an alternative name and location for the dq_grant.sh log. When this
    parameter is omitted, the script creates a file named dq_grant.log in the current
directory.

user-name

    is the user name to which permission is being granted. The target user account must
already exist in the Teradata database.

The authorizations granted by dq_grant.sh augment existing authorizations that the target
user account already has in the Teradata database.

You can verify that authorization was granted successfully for a user by logging on to the
database as the user and issuing the following command in BTEQ:

    call sas_sysfnlib.dq_debug();

The command will fail if the user does not have permission. Otherwise, it will have no
effect.

The data quality stored procedures are not yet ready to use. A QKB must be installed in
the Teradata database for the data quality stored procedures to be usable.

Obtaining and Deploying a QKB

Obtaining a QKB

You can obtain a QKB in one of the following ways:

• Run the SAS Deployment Wizard. In the Select Products to Install dialog box, select
  the check box for the SAS Quality Knowledge Base for your order. For step-by-step
guidance on installing a QKB using the SAS Deployment Wizard, see the SAS
Quality Knowledge Base for Contact Information: Installation and Configuration
Guide or the SAS Quality Knowledge Base for Product Data: Installation and
Configuration Guide, as appropriate, on the Quality Knowledge Base for SAS and
DataFlux Documentation site.

• Download a QKB from the SAS Downloads site. Contact your SAS Administrator to
determine which QKB has been licensed before downloading.
Select a QKB, and then follow the installation instructions in the Readme file for your operating environment. To open the Readme file, you must have a SAS profile. When prompted, you can log on or create a new profile.

- Copy a QKB that you already use with other SAS software in your enterprise. Contact your system administrator for its location.

**Packaging the QKB**

Before a QKB can be deployed in the Teradata database, you must package it into an .rpm file. An .rpm file is a file that is suitable for installation on Linux systems that use RPM package management software. SAS Data Quality Accelerator for Teradata provides the qkb_pack script to package the QKB into an .rpm file.

Windows and UNIX versions of qkb_pack are available. You must run the version that is appropriate for the host environment in which your QKB is installed.

qkb_pack is created in the following directories by the SAS Deployment Wizard:

- **Windows**
  
  `<SASHome>`\`SASDataQualityAcceleratorforTeradata\`9.4\`dqacctera\`sasmisc

- **UNIX**
  
  `<SASHome>`/SASDataQualityAcceleratorforTeradata/9.4/install/pgm

You must execute qkb_pack from the `<SASHome>` location.

Here is the syntax for executing qkb_pack:

- **Windows**
  
  `qkb_pack.cmd qkb-dir out-dir`

- **UNIX**
  
  `./qkb_pack.sh qkb-dir out-dir`

**qkb-dir**

specify the path to the QKB. Use the name of the QKB’s root directory. Typically, the root directory is found at the following directories:

- **Windows**
  
  C:\ProgramData\SAS\QKB\product-identifier\product-version (for example, C:\ProgramData\SAS\QKB\CI\28 or C:\ProgramData\SAS\QKB\PD\5).

- **UNIX**
  
  /opt/sas/qkb/share

**Note:** On Windows systems, QKB information exists in two locations: in C:\ProgramData and in C:\Program Files. For the qkb_pack command, you must specify the C:\ProgramData location.

**out-dir**

specify the directory where you want the package file to be created.

Here is an example of a command that you might execute to package a SAS QKB for Contact Information that resides on a Windows computer.

```
    cd c:\Program Files\SASHome\SASDataQualityAcceleratorforTeradata\9.4\dqacctera\sasmisc
    qkb_pack.cmd c:\ProgramData\SAS\QKB\CI\28 c:\temp
```

The default location for the qkb-dir argument is: C:\ProgramData\SAS \SASQualityKnowledgeBase\CI\28.

The package file that is created in C:\temp\ will have a name in the following form:
product
is a two-character product code for the QKB, such as CI (for Contact Information) or PD (for Product Data).

version
is the version number of the QKB.

timestamp
is a UNIX datetime value that indicates when qkb_pack was invoked. A UNIX datetime value is stored as the number of seconds since January 1, 1970.

noarch
indicates that the package file is platform-independent.

Here is an example of an output filename representing the QKB for Contact Information 28:

`sasqkb_ci-28.0-1474057340608.noarch.rpm`

After running qkb_pack, put the sasqkb package file on your Teradata database server in a location where it is available for both reading and writing. The package file must be readable by the Teradata Parallel Upgrade Tool. You need to move this package file to the server machine in accordance with procedures used at your site.

Follow the steps in “Installing the QKB Package File with the Teradata Parallel Upgrade Tool” on page 73 to deploy the QKB package file in the Teradata database.

**Installing the QKB Package File with the Teradata Parallel Upgrade Tool**

This installation should be performed by a Teradata systems administrator in collaboration with Teradata Customer Services. A Teradata Change Control is required when a package is added to the Teradata server. Teradata Customer Services has developed change control procedures for installing the SAS in-database deployment package.

The steps assume full knowledge of the Teradata Parallel Upgrade Tool and your environment. For more information about using the Teradata Parallel Upgrade Tool, see the Parallel Upgrade Tool (PUT) Reference, which is on the Teradata Online Publications site located at [http://www.info.teradata.com/GenSrch/eOnLine-Srch.cfm](http://www.info.teradata.com/GenSrch/eOnLine-Srch.cfm). On this page, search for “Parallel Upgrade Tool” and download the appropriate document for your system.

The following section explains the basic steps to install the sasqkb package file using the Teradata Parallel Upgrade Tool.

**Note:** It is not necessary to stop and restart the Teradata database when you install a QKB. However, if the SAS Embedded Process is running, you must stop it and then re-start it after the QKB is installed. It is also necessary to stop and restart the SAS Embedded Process for QKB updates. See “Controlling the SAS Embedded Process” on page 67 for information about stopping and restarting the SAS Embedded Process.

1. Start the Teradata Parallel Upgrade Tool.
2. Be sure to select all Teradata TPA nodes for installation, including Hot Stand-By nodes.
3. If Teradata Version Migration and Fallback (VM&F) is installed, you might be prompted whether to use VM&F. If you are prompted, choose Non-VM&F installation.

You can verify that the QKB installation was successful by running the following command from the shell prompt on one of the Teradata nodes.

```
psh "rpm -q -a" | grep sasqkb
```

If the installation was successful, the command returns the version number of the sasqkb package. Failure to return an output indicates that a library of that name could not be found.

The QKB is installed in the `/opt/qkb/default` directory of each Teradata node.

You are now ready to validate the data quality stored procedures for use.

---

### Validating the Accelerator Installation

Here is a simple BTEQ program that can be used to verify that the SAS Data Quality Accelerator for Teradata is operational.

The code first lists the locales that are installed in the QKB. Then it creates a table named Dqacceltest and executes the DQ_GENDER() stored procedure on the table. Before running the example, substitute a real value for the `output_table_1`, `output_table_2`, and `locale` variables throughout the program. For `locale`, use one of the values returned by the DQ_LIST_LOCALES() stored procedure. This example assumes that the SAS Data Quality Accelerator for Teradata is using the QKB for Contact Information.

The example also sets the SAS Data Quality Accelerator DQ_OVERWRITE_TABLE option to create temporary output tables in the SAS Data Quality Accelerator session. If you run the example again in the same SAS Data Quality Accelerator session, the new output tables overwrite any existing output tables and the output tables are automatically discarded at the end of the session. The DROP TABLE statement removes table Dqacceltest from your database.

```bteq
    call sas_systlib.dq_list_locales('mydb.output_table_1');
    select * from mydb.output_table_1;

    call sas_systlib.dq_set_option('DQ_OVERWRITE_TABLE', '1');

    create table mydb.dqacceltest (id_num integer, name varchar(64))
       unique primary index (id_num);

    insert into mydb.dqacceltest (id_num, name) values (1, 'John Smith');
    insert into mydb.dqacceltest (id_num, name) values (2, 'Mary Jones');

    call sas_systlib.dq_gender('Name', 'mydb.dqacceltest', 'name', 'id_num',
                    'mydb.output_table_2', 'locale');

    select gender from mydb.output_table_2;
    drop table mydb.dqacceltest;
```
If the request was successful, the SELECT statement produces an output table that contains this:

<table>
<thead>
<tr>
<th>Gender</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
</tr>
<tr>
<td>F</td>
</tr>
</tbody>
</table>

**Troubleshooting the Accelerator Installation**

Q. I ran the sample code and the output tables were not created in my user schema. What now?

A. The stored procedures can fail if one or more of the following conditions are true:

- The request specifies an output location to which the user does not have Write permission. Verify that you have access to the database that is specified in the `output_table` parameters.

- The data quality stored procedures are not installed correctly. Verify that the stored procedures are in the SAS_SYSFNLIB database by executing the following command in BTEQ:

  ```sql
  select TableName from dbc.tables where databasename='SAS_SYSFNLIB'
  and tablename like 'dq_%';
  ```

  The command should return a list similar to the following list (This is not a complete list.):

  - `dq_set_qkb`
  - `dq_match_parsed`
  - `dq_match_view_if_exists`
  - `dq_get_option_default`
  - `dq_debug`
  - `dq_propercase`
  - `dq_tbl_dbname`
  - `dq_drop_tbl_if_exists`
  - `dq_set_option`
  - `dqt_error`
  - `dq_standardize`
  - `dq_standardize_parsed`
  - `dq_debug2`
  - `dq_invoke_table`
  - `dq_lowercase`
  - `dq_set_locale`
  - `dq_extract`
  - `dq_uppercase`
  - `dq_listBindings`
  - `dq_replace_tags`
  - `dq_list_defs`
  - `dq_call_ep`
  - `dq_get_bool_option`
  - `dq_gen toktxt`
  - `dq_match`
If the procedures are absent, run the dq_install.sh script again, making sure that you are logged in as Teradata system administrator.

- Permission to the data quality stored procedures is not granted correctly. Verify that the target user name submitted to the dq_grant.sh script is a valid user account in the Teradata database. Verify that the database server and granter information in the dq_grant.sh shell script is correct.

- The QKB is not in the correct location. Look for subdirectories similar to the following in the /opt/qkb/default directory on the Teradata nodes: chopinfo, grammar, locale, phonetx, regexlib, scheme, and vocab.

- Your SQL request does not use the Teradata dialect. The stored procedures are invoked with the CALL keyword from any product that supports the Teradata SQL dialect. When you submit the data quality stored procedures in the SAS SQL procedure using explicit pass-through, the database connection is made in ANSI mode by default. You must specify the MODE= option to switch to Teradata mode. Consult the SAS/ACCESS Interface to Teradata documentation for more information about the MODE= option. Consult appropriate documentation for how to set Teradata mode in other client programs.

---

**Updating and Customizing a QKB**

SAS provides regular updates to the QKB. It is recommended that you update your QKB each time a new one is released. For a listing of the latest enhancements to the QKB, see “What’s New in SAS Quality Knowledge Base.” The What’s New document is available on the Quality Knowledge Base (QKB) for SAS and DataFlux Documentation site on support.sas.com.

Check the What’s New for each QKB to determine which definitions have been added, modified, or deprecated, and to learn about new locales that might be supported. Contact your SAS software representative to order updated QKBs and locales. To deploy a new QKB, follow the steps in “Packaging the QKB” on page 72 and “Installing the QKB Package File with the Teradata Parallel Upgrade Tool” on page 73. The accelerator supports one QKB in the Teradata database.

The standard definitions in the QKB are sufficient for performing most data quality operations. However, you can use the Customize feature of DataFlux Data Management Studio to modify the QKB definitions to meet specific needs.

If you want to customize your QKB, then as a best practice, SAS recommends that you customize your QKB on a local workstation before copying it to the Teradata database for deployment. When updates to the QKB are required, merge your customizations into an updated QKB locally, and copy the updated, customized QKB to the Teradata node.
This enables you to deploy a customized QKB to the Teradata database using the same steps that you would use to deploy a standard QKB. Copying your customized QKB from a local workstation into your cluster also means that you will have a backup of the QKB on your local workstation. See the online Help provided with your SAS Quality Knowledge Base for information about how to merge any customizations that you have made into an updated QKB.

Removing SAS Data Quality Accelerator from the Teradata Database

Overview

Before you can upgrade, re-install, or permanently remove SAS Data Quality Accelerator for Teradata or the SAS Embedded Process, you must remove any existing data quality stored procedures from the Teradata database. The stored procedures are removed from the Teradata database by using the dq_uninstall.sh script. For more information about this script, see “Using the dq_uninstall.sh Script” on page 77.

It is not necessary to remove the QKB when upgrading or re-installing software. QKB deployment steps automatically overwrite an older version of the QKB when you install a new one. For information to replace the QKB, see “Updating and Customizing a QKB” on page 76 and “Obtaining a QKB” on page 71.

When permanently removing SAS Data Quality Accelerator for Teradata from the Teradata database server, follow whatever procedure is appropriate at your site for removing the QKB. The Teradata administrator also needs to remove data quality authorizations from the Teradata database in accordance with site procedures.

Using the dq_uninstall.sh Script

Note: Stop the embedded process by using the instructions at “Controlling the SAS Embedded Process” on page 67 before following these steps. Stopping the SAS Embedded Process ensures that none of the accelerator files are locked when dq_uninstall.sh attempts to remove them.

The accelerator provides the dq_uninstall.sh shell script for removing the data quality stored procedures from the Teradata database. The dq_uninstall.sh script is located in the /opt/SAS/SASTKInDatabaseServerforTeradata/<version-number>/install/pgm directory of the Teradata database server.

The dq_uninstall.sh script requires modification before it can be run. The Teradata administrator must edit the shell script to specify the site-specific Teradata server name and DBC user logon credentials for the DBC_PASS=, DBC_SRVR=, and DBC_USER= variables.

Here is the syntax for executing dq_uninstall.sh:

```
./dq_uninstall.sh <-l log-path>
```

log-path specifies an alternative name and location for the dq_uninstall.sh log. When this parameter is omitted, the script creates a file named dq_uninstall.log in the current directory.

Running dq_uninstall.sh disables the SAS Data Quality Accelerator for Teradata functionality and removes the data quality stored procedures from the database.
Part 4

Administrator’s Guides for Aster, DB2, Greenplum, Netezza, Oracle, SAP HANA, and SPD Server

Chapter 9
Administrator’s Guide for Aster ........................................ 81

Chapter 10
Administrator’s Guide for DB2 ........................................ 87

Chapter 11
Administrator’s Guide for Greenplum .............................. 109

Chapter 12
Administrator’s Guide for Netezza .................................. 127

Chapter 13
Administrator’s Guide for Oracle .................................... 143

Chapter 14
Administrator’s Guide for SAP HANA ............................. 149

Chapter 15
Administrator’s Guide for SPD Server ............................. 159
Chapter 9
Administrator’s Guide for Aster

In-Database Deployment Package for Aster

Prerequisites
SAS Foundation and the SAS/ACCESS Interface to Aster must be installed before you install and configure the in-database deployment package for Aster.

The SAS Scoring Accelerator for Aster requires a specific version of the Aster client and server environment. For more information, see the SAS Foundation system requirements documentation for your operating environment.

Overview of the In-Database Deployment Package for Aster
This section describes how to install and configure the in-database deployment package for Aster (SAS Embedded Process).

The in-database deployment package for Aster must be installed and configured before you can use the %INDAC_PUBLISH_MODEL scoring publishing macro to create scoring files inside the database and the %INDAC_PUBLISH_FORMATS format publishing macro to create user-defined format files.

For more information about using the scoring and format publishing macros, see the SAS In-Database Products: User’s Guide.

The in-database deployment package for Aster includes the SAS Embedded Process. The SAS Embedded Process is a SAS server process that runs within Aster to read and
write data. The SAS Embedded Process contains macros, run-time libraries, and other software that is installed on your Aster system so that the SAS_SCORE( ) and the SAS_PUT( ) functions can access the routines within its run-time libraries.

---

### Aster Installation and Configuration

#### Aster Installation and Configuration Steps

1. If you are upgrading from a previous release, follow the instructions in “Upgrading from a Previous Version” on page 82 before installing the in-database deployment package.

2. Install the in-database deployment package.

   For more information, see “Installing the In-Database Deployment Package Binary Files for Aster” on page 82.

#### Upgrading from a Previous Version

Follow these steps to upgrade from a previous release.

1. Log on to the queen node.

   ```
   ssh -l root name-or-ip-of-queen-node
   ```

2. Move to the partner directory.

   ```
   cd /home/beehive/partner
   ```

3. If a SAS or SASEPHome directory exists in the partner directory, enter these commands to remove the existing installation from the queen and all of the workers.

   ```
   location=/home/beehive/partner/
   hostname -i;
   rm -rf $location/SAS;
   rm -rf $location/SASEPHome;
   for ip in `cat /home/beehive/cluster-management/hosts | grep node |
               awk '{print $3}'`; \
   do \n     echo $ip; \n     ssh $ip "rm -r $location/SAS"; \n     ssh $ip "rm -r $location/SASEPHome"; \n   done
   ```

#### Installing the In-Database Deployment Package Binary Files for Aster

**Unzipping the In-Database Deployment Package for Aster**

The in-database deployment package binary files for Aster are contained in a self-extracting archive file named sepcoreastr-13.00000-1.sh. This file is contained in a ZIP file that is put in your SAS Software Depot directory.

To unzip the in-database deployment package for Aster:
1. Create a new temporary directory on your client machine such as /sasep. The new directory is referred to as EPZipDir throughout this section.

2. Navigate to the YourSASDepot/standalone_installs directory. This directory was created when you created your SAS Software Depot.

3. Locate the en_sasexe.zip file. The en_sasexe.zip file is located in the YourSASDepot/standalone_installs/SAS_Core_Embedded_Process_Package_for_Aster/13_0/Aster_on_Linux_x64 directory.

   The sepcoreastr-13.00000-1.sh file is included in this ZIP file.

4. Copy the en_sasexe.zip file to your EPZipDir temporary directory on the client machine.

   cp en_sasexe.zip /EPZipDir

5. Navigate to your EPZipDir temporary directory and unzip en_sasexe.zip.

   After the file is unzipped, a sasexe directory is created in the same location as the en_sasexe.zip file. The sepcoreastr-13.00000-1.sh file is in the /EPZipDir/sasexe directory.

### Installing the In-Database Deployment Package Binary Files for Aster

The in-database deployment package binary files for Aster are contained in a self-extracting archive file named sepcoreastr-13.00000-1.sh. To install the in-database deployment package binary files for Aster, you need root privileges for the queen node. Once you are logged in to the queen node as root, you need to create a directory in which to put sepcoreastr-13.00000-1.sh, execute sepcoreastr-13.00000-1.sh, and install the SAS_SCORE( ) and the SAS_PUT( ) SQL/MR functions.

Enter these commands to install the SAS System Libraries and the binary files:

1. Log on to the queen node.
   
   ssh -l root name-or-ip-of-queen-node

2. Move to the parent of the partner directory.

   cd /home/beehive/

3. Create a partner directory if it does not already exist.

   mkdir partner

4. Move to the partner directory.

   cd partner

5. From the SAS client machine, copy the sepcoreastr-13.00000-1.sh file from the client EPZipDir directory to the partner directory on the queen node.

   a. Navigate to the /EPZipDir/sasexe directory on the client machine.

   b. Copy the self-extracting archive file to the partner directory on the queen node.

      This example uses secure copy.

      scp sepcoreastr-13.00000-1.sh root@astmach1:/home/beehive/partner

6. If your client does not copy the executable attribute from the client machine to the server, add the EXECUTE permission to the self-extracting archive file after the file is transferred to the server.

   chmod 755 -r sepcoreastr-13.00000-1.sh
7. Unpack the self-extracting archive file in the partner directory.
   
   `/sepcoreastr-13.00000-1.sh`

   This installs the SAS Embedded Process on the queen node. When Aster synchronizes the beehive, the files are copied to all the nodes. This can take a long time.

8. (Optional) There are two methods to copy the files to the nodes right away. You can do either of the following steps:

   • Run the following code to manually move the files across all nodes on the beehive by using secure copy and SSH:

   ```
   location=/home/beehive/partner
   cd $location
   for ip in `cat /home/beehive/cluster-management/hosts | grep node | awk '{print $3}'`; \
   do \ 
   echo $ip; \
   scp -r sepcore* root@$ip"$location"; \
   ssh $ip "cd $location; ./sepcoreastr-13.00000-1.sh";
   done
   ```

   • Run this command to synchronize the beehive and restart the database:

   `/home/beehive/bin/utils/primitives/UpgradeNC Cluster.py -u`

9. Navigate to the directory where the SAS Embedded Process is installed.

   cd `/home/beehive/partner/SASEPHome/sasexe`

10. Install the SAS_SCORE( ), SAS_PUT( ), and other SQL/MR functions.

    a. Start the ACT tool.

       `/home/beehive/clients/act -U db_superuser -w db_superuser-password -d database-to-install-sas_score-into`

    b. Run this command to verify if there are any existing SQL/MR functions installed:

       `\dF *SAS*`

       If you see existing SAS functions installed, it is recommended that you remove them before installing the new ones. Enter these commands to remove the functions:

       `\remove sas_score.tk.so`
       `\remove sas_put.tk.so`
       `\remove sas_row.tk.so`
       `\remove sas_partition.tk.so`

    c. Enter the following commands to install the new SQL/MR functions. The SQL/MR functions need to be installed under the PUBLIC schema.

       `\install sas_score.tk.so`
       `\install sas_put.tk.so`
       `\install sas_row.tk.so`
       `\install sas_partition.tk.so`

11. Exit the ACT tool.

    `\q`
12. Verify the existence and current date of the tkast-runInCluster and tkeastmr.so files. These two binary files are needed by the SAS SQL/MR functions.

```bash
for ip in \
`cat /home/beehive/cluster-management/hosts | grep node | awk '{print $3}'`; \
do \
echo $ip; \
ssh $ip "ls -al /home/beehive/partner/SASEPHome/sasexe/tkeastmr.so"; 
ssh $ip "ls -al /home/beehive/partner/SASEPHome/utilities/bin/ 
tkast-runInCluster"; 

done
```

---

**Validating the Publishing of the SAS_SCORE( ) and the SAS_PUT( ) Functions**

To validate that the SAS_SCORE( ) and the SAS_PUT( ) functions were installed, run the `dp` command in the Aster Client or use any of the following views:

- `nc_all_sqlmr_funcs`, where *all* returns all functions on the system
- `nc_user_sqlmr_funcs`, where *user* returns all functions that are owned by or granted to the user
- `nc_user_owned_sqlmr_funcs`, where *user_owned* returns all functions that are owned by the user

---

**Aster Permissions**

The person who installs the in-database deployment package binary files in Aster needs root privileges for the queen node. This permission is most likely, but not necessarily, needed by the Aster system administrator.

For Aster 6.10 or later, the following schema permissions are needed by the person who runs the scoring and format publishing macros, because all functions and files can be published to a specific schema.

**USAGE permission**

```
GRANT USAGE ON SCHEMA yourschemaname TO youruserid;
```

**INSTALL FILE permission**

```
GRANT INSTALL FILE ON SCHEMA yourschemaname TO youruserid;
```

**CREATE permission**

```
GRANT CREATE ON SCHEMA yourschemaname TO youruserid;
```

**EXECUTE permission**

```
GRANT EXECUTE ON FUNCTION PUBLIC.SAS_SCORE TO youruserid;
GRANT EXECUTE ON FUNCTION PUBLIC.SAS_PUT TO youruserid;
GRANT EXECUTE ON FUNCTION PUBLIC.SAS_ROW TO youruserid;
GRANT EXECUTE ON FUNCTION PUBLIC.SAS_PARTITION TO youruserid;
```
Documentation for Using In-Database Processing in Aster

For information about how to publish SAS formats and scoring models, see the SAS In-Database Products: User’s Guide, located at http://support.sas.com/documentation/onlinedoc/indbtech/index.html
Chapter 10
Administrator’s Guide for DB2

In-Database Deployment Package for DB2

Prerequisites
SAS Foundation and the SAS/ACCESS Interface to DB2 must be installed before you install and configure the in-database deployment package for DB2.

The SAS Scoring Accelerator for DB2 requires a specific version of the DB2 client and server environment. For more information, see the SAS Foundation system requirements documentation for your operating environment.

Overview of the In-Database Deployment Package for DB2
This section describes how to install and configure the in-database deployment package for DB2 (SAS Formats Library for DB2 and SAS Embedded Process).

The in-database deployment package for DB2 must be installed and configured before you can perform the following tasks:
• Use the `%INDB2_PUBLISH_FORMATS` format publishing macro to create or publish the `SAS_PUT( )` function and to create or publish user-defined formats as format functions inside the database.

• Use the `%INDB2_PUBLISH_MODEL` scoring publishing macro to create scoring model functions inside the database.

For more information about using the format and scoring publishing macros, see the *SAS In-Database Products: User's Guide*.

The in-database deployment package for DB2 contains the SAS formats library and the precompiled binary files for two additional utility functions. The package also contains the SAS Embedded Process.

The SAS formats library is a run-time library that is installed on your DB2 system so that the SAS scoring model functions and the `SAS_PUT( )` function created in DB2 can access the routines within the run-time library. The SAS formats library contains the formats that are supplied by SAS.

The two publishing macros, `%INDB2_PUBLISH_COMPILEUDF` and `%INDB2_PUBLISH_DELETEUDF`, register utility functions in the database. The utility functions are called by the format and scoring publishing macros. You must run these two macros before you run the format and scoring publishing macros.

The SAS Embedded Process is a SAS server process that runs within DB2 to read and write data. The SAS Embedded Process contains macros, run-time libraries, and other software that is installed on your DB2 system so that the SAS scoring files created in DB2 can access the routines within the SAS Embedded Process’s run-time libraries.

---

**Function Publishing Process in DB2**

To publish scoring model functions and the `SAS_PUT( )` function on a DB2 server, the publishing macros perform the following tasks:

• Create and transfer the files to the DB2 environment.

• Compile those source files into object files using the appropriate compiler for that system.

• Link with the SAS formats library.

After that, the publishing macros register the format and scoring model functions in DB2 with those object files. If an existing format or scoring model function is replaced, the publishing macros remove the obsolete object file upon successful compilation and publication of the new format or scoring model functions.

The publishing macros use a SAS FILENAME SFTP statement to transfer the format or scoring source files to the DB2 server. An SFTP statement offers a secure method of user validation and data transfer. The SAS FILENAME SFTP statement dynamically launches an SFTP or PSFTP executable, which creates an SSH client process that creates a secure connection to an OpenSSH Server. All conversation across this connection is encrypted, from user authentication to the data transfers.

Currently, only the OpenSSH client and server on UNIX that supports protocol level SSH-2 and the PUTTY client on WINDOWS are supported. For more information about setting up the SSH software to enable the SAS SFTP to work, please see Setting Up SSH Client Software in UNIX and Windows Environments for Use with the SFTP Access Method in SAS 9.2, SAS 9.3, and SAS 9.4, located at http://support.sas.com/techsup/technote/ts800.pdf.
Note: This process is valid only when using publishing formats and scoring functions. It is not applicable to the SAS Embedded Process. If you use the SAS Embedded Process, the scoring publishing macro creates the scoring files and uses the SAS/ACCESS Interface to DB2 to insert the scoring files into a model table.

DB2 Installation and Configuration

DB2 Installation and Configuration Steps

1. If you are upgrading from a previous version, follow the instructions in “Upgrading from a Previous Version” on page 89.

2. Verify that you can use PSFTP from Windows to UNIX without being prompted for a password or cache.

   To do this, enter the following commands from the PSFTP prompt, where userid is the user ID that you want to log on as and machinename is the machine to which you want to log on.

   psftp> open userid@machinename
   psftp> ls

3. Install the SAS formats library, the binary files for the SAS_COMPILEUDF and SAS_DELETEUDF functions, and the SAS Embedded Process.

   For more information, see “Installing the SAS Formats Library and Binary Files to DB2” on page 92.

4. Run the %INDB2_PUBLISH_COMPILEUDF macro to create the SAS_COMPILEUDF function.

   For more information, see “Running the %INDB2_PUBLISH_COMPILEUDF Macro” on page 99.

5. Run the %INDB2_PUBLISH_DELETEUDF macro to create the SAS_DELETEUDF function.

   For more information, see “Running the %INDB2_PUBLISH_DELETEUDF Macro” on page 103.

6. If you plan to use SAS Model Manager with the SAS Scoring Accelerator for in-database scoring, perform the additional configuration tasks provided in Chapter 16, “Configuring SAS Model Manager,” on page 163.

Upgrading from a Previous Version

Overview of Upgrading from a Previous Version

You can upgrade from a previous version of the SAS Formats Library and binary files, the SAS Embedded Process, or both. See the following topics:

- If you want to upgrade from a previous version of the SAS Formats Library, binary files, and the SAS Embedded Process, see “Upgrading the SAS Formats Library, Binary Files, and the SAS Embedded Process” on page 90.

- If you want to upgrade only the SAS Embedded Process, see “Upgrading the SAS Embedded Process” on page 91.
Upgrading the SAS Formats Library, Binary Files, and the SAS Embedded Process

To upgrade a previous version of the SAS Formats Library, binary files, and the SAS Embedded Process, follow these steps.

**Note:** These steps also apply if you want to upgrade only the SAS Formats Library and binary files. If you want to upgrade only the SAS Embedded Process, see “Upgrading the SAS Embedded Process” on page 91.

1. Drop the SAS_COMPILEUDF and SAS_DELETEUDF functions by running the %INDB2_PUBLISH_COMPILEUDF and %INDB2_PUBLISH_DELETEUDF macros with ACTION=DROP.

   Here is an example.
   ```
   %let indconn = user=abcd password=xxxx database=indbdb server=indbsvr;
   %indb2_publish_compileudf(action=drop, db2path=/users/db2v10/sqllib, 
   compiler_path=/usr/vac/bin);
   %indb2_publish_deleteudf(action=drop);
   ```

2. Confirm that the SAS_COMPILEUDF and SAS_DELETEUDF functions were dropped.

   Here is an example.
   ```
   proc sql noerrorstop;
   connect to db2 (user=abcd password=xxxx database=indbdb);
   select * from connection to db2 ( 
   select cast(funcname as char(40)),
   cast(definer as char(20)) from syscat.functions 
   where funcschema='SASLIB' );
   quit;
   ```

   If you are upgrading only the SAS Formats Library and the binary files, skip to Step 6.

3. Enter the following command to see whether the SAS Embedded Process is running.

   ```
   ps -ef | grep db2sasep
   ```

   If the SAS Embedded Process is running, results similar to this are displayed.
   ```
   db2v10 23265382 20840668   0   Oct  6      -  4:03 db2sasep
   ```

4. Stop the DB2 SAS Embedded Process using DB2IDA command.

   Use this command to stop the SAS Embedded Process.
   ```
   db2ida -provider sas -stop
   ```

   If the SAS Embedded Process is still running, an error occurs. Enter this command to force the SAS Embedded Process to stop.
   ```
   db2ida -provider sas -stopforce
   ```

   For more information about the DB2IDA command, see “Controlling the SAS Embedded Process for DB2” on page 98.

5. Remove the SAS directory that contains the SAS Embedded Process binary files from the DB2 instance path.

   - If you are upgrading a version of the SAS Embedded Process prior to SAS 9.4M4, enter the following commands to move to the `db2instancepath` directory and remove the SAS directory. `db2instancepath` is the path to the SAS Embedded Process binary files in the DB2 instance.
cd db2instancepath
rm -fr SAS

- If you are upgrading a version of the SAS Embedded Process starting with SAS 9.4M4, enter the following commands to move to the db2instancepath directory and remove the SAS directory. db2instancepath is the path to the SAS Embedded Process binary files in the DB2 instance.

cd db2instancepath
rm -fr SASEPHome

6. Stop the DB2 instance.
   a. Log on to the DB2 server and enter this command to determine whether there are any users connected to the instance:

      db2 list applications

   b. If any users are connected, enter these commands to force them off before the instance is stopped and clear any background processes:

      db2 force applications all
      db2 terminate

   c. Enter this command to stop the DB2 instance:

      db2stop

7. Enter the following commands to move to the db2instancepath/sqllib/function directory and remove the SAS directory. db2instancepath/sqllib/function is the path to the SAS_COMPILEUDF and SAS_DELETEUDF functions in the DB2 instance.

      cd db2instancepath/sqllib/function
      rm -fr SAS

---

**Upgrading the SAS Embedded Process**

To upgrade the SAS Embedded Process, follow these steps.

*Note:* These steps are for upgrading the SAS Embedded Process. If you want to upgrade the SAS Formats Library and binary files or both the SAS Formats Library and binary files and the SAS Embedded Process, you must follow the steps in “Upgrading the SAS Formats Library, Binary Files, and the SAS Embedded Process” on page 90.

1. Enter the following command to see whether the SAS Embedded Process is running.

   ps -ef | grep db2sasep

   If the SAS Embedded Process is running, results similar to this are displayed.

   db2v10 23265382 20840668   0   Oct  6      -  4:03 db2sasep
   db2v10 27983990 16646196   1 08:24:09 pts/10  0:00 grep db2sasep

2. Enter the following command to determine whether there are any users connected to the instance.

   db2 list applications


   *Note:* If you are upgrading the SAS Embedded Process, you do not need to shut down the database. The DB2IDA command enables you to upgrade only the SAS Embedded Process components without impacting clients already connected to the database. For more information about the DB2IDA command, see “Controlling the SAS Embedded Process for DB2” on page 98.
Use this command to stop the SAS Embedded Process.

```
  db2ida -provider sas -stop
```

If the SAS Embedded Process is still running, an error occurs. Enter this command to force the SAS Embedded Process to stop.

```
  db2ida -provider sas -stopforce
```

4. Remove the SAS directory that contains the SAS Embedded Process binary files from the DB2 instance path.

- If you are upgrading a version of the SAS Embedded Process prior to SAS 9.4M4, enter the following commands to move to the `db2instancepath` directory and remove the SAS directory. `db2instancepath` is the path to the SAS Embedded Process binary files in the DB2 instance.

```
  cd db2instancepath
  rm -fr SAS
```

- If you are upgrading a version of the SAS Embedded Process starting with SAS 9.4M4, enter the following commands to move to the `db2instancepath` directory and remove the SAS directory. `db2instancepath` is the path to the SAS Embedded Process binary files in the DB2 instance.

```
  cd db2instancepath
  rm -fr SASEPHome
```

**Installing the SAS Formats Library and Binary Files to DB2**

**Move the SAS Formats and Binary Files to DB2**

The SAS formats library and the binary files that need to be moved to DB2 are contained in a self-extracting archive file. You can use PSFTP, SFTP, or FTP to transfer the file to the DB2 server to be unpacked and compiled.

The self-extracting archive file contains the SAS formats library and the binary files for the `SAS_COMPILEUDF` and `SAS_DELETEUDF` functions. You need these files when you want to use scoring functions to run your scoring model and when publishing SAS formats.

This self-extracting archive file is located in the `YourSASDepot/SASFormatsLibraryforDB2/3.1/DB2on<AIX|Linux64>/` directory.

Choose the self-extracting archive file based on the UNIX platform that your DB2 server runs on.

- AIX: acceldb2fmt-3.1-`n`-_r64.sh
- Linux(x86_64): acceldb2fmt-3.1-`n`-_lax.sh

`n` is a number that indicates the latest version of the file. If this is the initial installation, `n` has a value of 1. Each time you re-install or upgrade, `n` is incremented by 1.

The file does not have to be downloaded to a specific location. However, you need to note where it is downloaded so that it can be executed as the DB2 instance owner at a later time. It is recommended that you put the acceldb2fmt file somewhere other than the DB2 home directory tree.

List the directory in UNIX to verify that the files have been moved.
Unpack and Install the SAS Formats Library and Binary Files for DB2

After the acceldb2fmt-3.1-\(n\)_lax.sh or acceldb2fmt-3.1-\(n\)_r64.sh self-extracting archive file is transferred to the DB2 machine, follow these steps to unpack the file. \(n\) is a number that indicates the latest version of the file. If this is the initial installation, \(n\) has a value of 1. Each time you re-install or upgrade, \(n\) is incremented by 1.

1. Log on as the user who owns the DB2 instance from a secured shell, such as SSH.
2. Change to the directory where you put the acceldb2fmt file.
   
   \begin{verbatim}
   cd path_to_sh_file
   \end{verbatim}
   
   *path_to_sh_file* is the location to which you copied the self-extracting archive file.
3. Change permissions on the file to enable you to execute the script.
   
   \begin{verbatim}
   chmod 755 -r acceldb2fmt-3.1-n_r64.sh
   \end{verbatim}
   
   *Note:* AIX is the platform that is being used as an example for all the steps in this topic.
4. If there is a previously created self-extracting archive file in the SAS directory, you must either rename or remove the directory. These are examples of the commands that you would use:
   
   \begin{verbatim}
   mv SAS to SAS_OLD /* renames the SAS directory */
   rm -fr SAS /* removes the SAS directory */
   \end{verbatim}
5. Use the following commands to unpack the appropriate self-extracting archive file.
   
   \begin{verbatim}
   ./sh_file
   \end{verbatim}

   *sh_file* is either acceldb2fmt-3.1-\(n\)_lax.sh or acceldb2fmt-3.1-\(n\)_r64.sh depending on your platform.

   After this script is run and the files are unpacked, a SAS tree is built in the current directory. The content of the target directories should be similar to the following, depending on your operating system. Part of the directory path is shaded to emphasize the different target directories that are used.

   \begin{verbatim}
   /path_to_sh_file/SAS/SASFormatsLibraryForDB2/3.1-n/bin/InstallAccelDB2Fmt.sh
   /path_to_sh_file/SAS/SASFormatsLibraryForDB2/3.1-n/bin/CopySASFiles.sh
   /path_to_sh_file/SAS/SASFormatsLibraryForDB2/3.1-n/lib/SAS_CompileUDF
   /path_to_sh_file/SAS/SASFormatsLibraryForDB2/3.1-n/lib/SAS_DeleteUDF
   /path_to_sh_file/SAS/SASFormatsLibraryForDB2/3.1-n/lib/libjazxfbrs.so
   /path_to_sh_file/SAS/SASFormatsLibraryForDB2/3.1 -3.1-n
   \end{verbatim}
6. Use the following command to place the files in the DB2 instance:
   
   \begin{verbatim}
   path_to_sh_file/SAS/SASFormatsLibraryForDB2/3.1-n/bin/CopySASFiles.sh db2instancepath/sqlib
   \end{verbatim}

   *db2instancepath/sqlib* is the path to the *sqlib* directory of the DB2 instance that you want to use.

   After this script is run and the files are copied, the target directory should look similar to this.

   \begin{verbatim}
   db2instancepath/sqlib/function/SAS/SAS_CompileUDF
   db2instancepath/sqlib/function/SAS/SAS_DeleteUDF
   db2instancepath/sqlib/function/SAS/libjazxfbrs.so
   \end{verbatim}

   *Note:* If the SAS_CompileUDF, SAS_DeleteUDF, and libjazxfbrs.so files currently exist under the target directory, you must rename the existing files before you run
7. Use the DB2SET command to tell DB2 where to find the 64-bit formats library.

   db2set DB2LIBPATH=db2instancepath/sqllib/function/SAS

   *db2instancepath/sqllib* is the path to the *sqllib* directory of the DB2 instance that you want to use.

   The DB2 instance owner must run this command for it to be successful. Note that this is similar to setting a UNIX system environment variable using the UNIX EXPORT or SETENV commands. DB2SET registers the environment variable within DB2 only for the specified database server.

8. To verify that DB2LIBPATH was set appropriately, run the DB2SET command without any parameters.

   db2set

   The results should be similar to this one if it was set correctly.

   DB2LIBPATH=db2instancepath/sqllib/function/SAS

---

### Installing the SAS Embedded Process

**Unzipping the SAS Embedded Process for DB2**

The SAS Embedded Process is contained in a self-extracting archive file named `sepcoredb2a-13.00000-1.sh` (AIX) or `sepcoredb2l-13.00000-1.sh` (Linux). This file is contained in a ZIP file that is put in your SAS Software Depot directory. You need this file if you want to run your scoring model.

To unzip the SAS Embedded Process file for DB2:

1. Create a new temporary directory on your client machine such as `/sasep`. The new directory is referred to as `EPZipDir` throughout this section.

2. Navigate to the `YourSASDepot/standalone_installs` directory. This directory was created when you created your SAS Software Depot.

3. Locate the `en_sasexe.zip` file. The `en_sasexe.zip` file is located in the `YourSASDepot/standalone_installs/SAS_Core_Embedded_Process_Package_for_DB2on<AIX|Linux>/13_0/DB2on<AIX|Linux_x64>` directory.

   The `sepcoredb2a-13.00000-1.sh` (AIX) or `sepcoredb2l-13.00000-1.sh` (Linux) file is included in this ZIP file.

4. Copy the `en_sasexe.zip` file to your `EPZipDir` temporary directory on the client machine.

   ```
   cp en_sasexe.zip /EPZipDir
   ```

5. Navigate to your `EPZipDir` temporary directory and unzip `en_sasexe.zip`.

   After the file is unzipped, a `sasexe` directory is created in the same location as the `en_sasexe.zip` file. The self-extracting archive file is in the `/EPZipDir/sasexe` directory.

   The self-extracting archive file depends on the UNIX platform that your DB2 server runs on.
Unpacking the SAS Embedded Process Files for DB2

After the sepcoredb2a-13.00000-1.sh (AIX) or the sepcoredb2l-13.00000-1.sh (Linux) self-extracting archive file has been transferred to the DB2 machine, follow these steps to unpack the file:

1. Log on as the user who owns the DB2 instance from a secured shell, such as SSH.

2. Move the self-extracting archive file to the instance owner’s home directory. This directory is referred to as `path_to_sh_file`.
   a. Move to the `/EPZipDir/sasexe` directory on the client machine where the self-extracting archive file exists.
   b. Use PSFTP, SFTP, or FTP to transfer the self-extracting archive file to the instance owner’s home directory on the DB2 server to be unpacked and compiled.

3. Change to the directory where you put the sepcoredb2a-13.00000-1.sh (AIX) or the sepcoredb2l-13.00000-1.sh (Linux) file.
   ```
   cd path_to_sh_file
   ```
   `path_to_sh_file` is the location to which you copied the self-extracting archive file. This must be the instance owner home directory.
   List the directory in UNIX to verify that the file has been moved.

4. Change permissions on the file to enable you to execute the script.
   ```
   chmod 755 -r sepcoredb2*.sh
   ```

5. If there is a previously created SAS directory, you must either rename or remove the directory. These are examples of the commands that you would use:
   - If you are upgrading from a version of the SAS Embedded Process prior to SAS 9.4M4, here is an example of the commands that you would use:
     ```
     mv SAS to SAS_OLD /* renames the SAS directory */
     rm -fr SAS /* removes the SAS directory */
     ```
   - If you are upgrading from a version of the SAS Embedded Process from SAS 9.4M4 or later, here is an example of the commands that you would use:
     ```
     mv SASEPHome to SAS_OLD /* renames the SASEPHome directory */
     rm -fr SASEPHome /* removes the SASEPHome directory */
     ```

6. Use the following commands to unpack the appropriate self-extracting archive file:
   ```
   ./sh_file
   ```
   `sh_file` is either sepcoredb2a-13.00000-1.sh or sepcoredb2l-13.00000-1.sh, depending on your platform.

After this script is run and the files are unpacked, a SAS tree is built in the current directory. The target directories should be similar to the following, depending on your operating system. Part of the directory path is shaded to emphasize the different target directories that are used.

```
/db2instancepath/SASEPHome/bin
/db2instancepath/SASEPHome/misc
/db2instancepath/SASEPHome/sasexe
/db2instancepath/SASEPHome/utilities
```
7. Use the DB2SET command to enable the SAS Embedded Process in DB2 and to tell the SAS Embedded Process where to find the SAS Embedded Process library files.

```
dbset DB2_SAS_SETTINGS="ENABLE_SAS_EP:true;
LIBRARY_PATH:db2instancepath/SASEPHome/sasexe"
```

The DB2 instance owner must run this command for it to be successful. Note that this is similar to setting a UNIX system environment variable using the UNIX EXPORT or SETENV commands. DB2SET registers the environment variable within DB2 only for the default database instance.

For more information about all of the arguments that can be used with the DB2SET command for the SAS Embedded Process, see “DB2SET Command Syntax for the SAS Embedded Process” on page 97.

8. To verify that the SAS Embedded Process is set appropriately, run the DB2SET command without any parameters.

```
db2set
```

The path should be similar to this one if it was set correctly. Note that the DB2LIBPATH that was set when you installed the SAS Formats Library and binary files is also listed.

```
DB2_SAS_SETTINGS=ENABLE_SAS_EP:true;
LIBRARY_PATH:db2instancepath/SASEPHome/sasexe
DB2LIBPATH= db2instancepath/sqllib/function/SAS
```

9. Stop the database manager instance if it is not stopped already.

```
db2stop
```

A message indicating that the stop was successful displays.

If the database manager instance cannot be stopped because application programs are still connected to databases, use the FORCE APPLICATION command to disconnect all users, use the TERMINATE command to clear any background processes, and then use the DB2STOP command.

```
db2 list applications
db2 force applications all
db2 terminate
db2stop
```

10. (AIX only) Clear the cache.

```
su root
slibclean
exit
```

11. Restart the database manager instance.

```
db2start
```

12. Verify that the SAS Embedded Process started.

```
ps -ef | grep db2sasep
```

If the SAS Embedded Process was started, lines similar to the following are displayed:

```
db2v10 23265382 20840668  0  Oct  6     4:03 db2sasep
db2v10 27983990 16646196  1 08:24:09 pts/10 0:00 grep db2sasep
```

In the DB2 instance, you can also verify if the SAS Embedded Process log file was created in the DB2 instance’s diagnostic directory.
### DB2SET Command Syntax for the SAS Embedded Process

The syntax for the DB2SET command is shown below.

```sql
DB2SET DB2_SAS_SETTINGS="
ENABLE_SAS_EP:TRUE | FALSE;
<LIBRARY_PATH:path>
<COMM_BUFFER_SZ:size;>
<COMM_TIMEOUT:timeout;>
<RESTART_RETRIES:number-of-tries;>
<DIAGPATH:path;>
<DIAGLEVEL:level-number;>"
```

#### Arguments

**ENABLE_SAS_EP:TRUE | FALSE**
- Specifies whether the SAS Embedded Process is started with the DB2 instance.
  - Default: FALSE

**LIBRARY_PATH:path**
- Specifies the path from which the SAS Embedded Process library is loaded.
  - Requirement: The path must be fully qualified.

**COMM_BUFFER_SZ:size**
- Specifies the size in 4K pages of the shared memory buffer that is used for communication sessions between DB2 and SAS.
  - Default: ASLHEAPSZ dbm configuration value
  - Range: 1–32767
  - Requirement: size must be an integer value.

**COMM_TIMEOUT:timeout**
- Specifies a value in seconds that DB2 uses to determine whether the SAS Embedded Process is non-responsive when DB2 and SAS are exchanging control messages.
  - Default: 600 seconds
  - Note: If the time-out value is exceeded, DB2 forces the SAS Embedded Process to stop in order for it to be re-spawned.

**RESTART_RETRIES:number-of-tries**
- Specifies the number of times that DB2 attempts to re-spawn the SAS Embedded Process after DB2 has detected that the SAS Embedded Process has terminated abnormally.
  - Default: 10
  - Range: 1–100
  - Requirement: number-of-tries must be an integer value.
  - Note: When DB2 detects that the SAS Embedded Process has terminated abnormally, DB2 immediately attempts to re-spawn it. This argument
limits the number of times that DB2 attempts to re-spawn the SAS Embedded Process. Once the retry count is exceeded, DB2 waits 15 minutes before trying to re-spawn it again.

**DIAGPATH: path**

specifies the path that indicates where the SAS Embedded Process diagnostic logs are written.

Default: **DIAGPATH dbm configuration value**

Requirement: The path must be fully qualified.

**DIAGLEVEL: level-number**

specifies the minimum severity level of messages that are captured in the SAS Embedded Process diagnostic logs. The levels are defined as follows.

1. **SEVERE**
2. **ERROR**
3. **WARNING**
4. **INFORMATIONAL**

Default: **DIAGLEVEL dbm configuration value**

Range: 1–4

---

### Controlling the SAS Embedded Process for DB2

The SAS Embedded Process starts when a query is submitted. The SAS Embedded Process continues to run until it is manually stopped or the database is shut down.

The DB2IDA command is a utility that is installed with the DB2 server to control the SAS Embedded Process. The DB2IDA command enables you to manually stop and restart the SAS Embedded Process without shutting down the database. You might use the DB2IDA command to upgrade or re-install the SAS Embedded Process library or correct an erroneous library path.

*Note:* DB2IDA requires IBM Fixpack 6 or later.

The DB2IDA command has the following parameters:

- **-provider sas**
  specifies the provider that is targeted by the command. The only provider that is supported is "sas".

- **-start**
  starts the SAS Embedded Process on the DB2 instance if the SAS Embedded Process is not currently running.

  If the SAS Embedded Process is running, this command has no effect.

  *Note:* Once the SAS Embedded Process is started, the normal re-spawn logic in DB2 applies if the SAS Embedded Process is abnormally terminated.

- **-stop**
  stops the SAS Embedded Process if it is safe to do so.

  If the SAS Embedded Process is stopped, this command has no effect.

  If any queries are currently running on the SAS Embedded Process, the `db2ida -stop` command fails and indicates that the SAS Embedded Process is in use and could not be stopped.
Note: DB2 does not attempt to re-spawn the SAS Embedded Process once it has been stopped with the `db2ida -stop` command.

**-stopforce**
forces the SAS Embedded Process to shut down regardless of whether there are any queries currently running on it.

If the SAS Embedded Process is stopped, this command has no effect.
If any queries are currently running on the SAS Embedded Process, those queries receive errors.

Note: DB2 does not attempt to re-spawn the SAS Embedded Process once it has been stopped with the `db2ida -stopforce` command.

Here are some examples of the DB2IDA command:

```
db2ida -provider sas -stopforce

db2ida -provider sas -start
```

---

### Running the `%INDB2_PUBLISH_COMPILEUDF` Macro

#### Overview of the `%INDB2_PUBLISH_COMPILEUDF` Macro

The `%INDB2_PUBLISH_COMPILEUDF` macro publishes the following components to the SASLIB schema in a DB2 database:

- **SAS_COMPILEUDF function**
  
  The SAS_COMPILEUDF function facilitates the `%INDB2_PUBLISH_FORMATS` format publishing macro and the `%INDB2_PUBLISH_MODEL` scoring publishing macro when you use scoring functions to run the scoring model. The SAS_COMPILEUDF function performs the following tasks:

  - compiles the format and scoring model source files into object files. This compilation occurs through the SQL interface using an appropriate compiler for the system.
  - links with the SAS formats library that is needed for format and scoring model publishing.
  - copies the object files to the `db2instancepath/sqlib/function/SAS` directory. You specify the value of `db2instancepath` in the `%INDB2_PUBLISH_COMPILEUDF` macro syntax.

- **SASUDF_DB2PATH and SASUDF_COMPILER_PATH global variables**
  
  The SASUDF_DB2PATH and the SASUDF_COMPILER_PATH global variables are used when you publish the format and scoring model functions.

You have to run the `%INDB2_PUBLISH_COMPILEUDF` macro only one time in a given database.

The SAS_COMPILEUDF function must be published before you run the `%INDB2_PUBLISH_DELETEUDF` macro, the `%INDB2_PUBLISH_FORMATS` macro, and the `%INDB2_PUBLISH_MODEL` macro. Otherwise, these macros fail.

Note: To publish the SAS_COMPILEUDF function, you must have the appropriate DB2 user permissions to create and execute this function in the SASLIB schema and in the specified database. For more information, see “DB2 Permissions” on page 106.
%INDB2_PUBLISH_COMPILEUDF Macro Run Process

To run the %INDB2_PUBLISH_COMPILEUDF macro, follow these steps:

1. Create a SASLIB schema in the database where the SAS.CompileUDF function is to be published.

   The SASLIB schema is used when publishing the %INDB2_PUBLISH_COMPILEUDF macro for DB2 in-database processing. You specify that database in the DATABASE argument of the %INDB2_PUBLISH_COMPILEUDF macro. For more information, see “%INDB2_PUBLISH_COMPILEUDF Macro Syntax” on page 101.

   The SASLIB schema contains the SAS.CompileUDF and SAS_DeleteUDF functions and the SASUDF_DB2PATH and SASUDF_COMPILER_PATH global variables.

2. Start SAS and submit the following command in the Enhanced Editor or Program Editor:

   ```sas
   %let indconn = server=yourserver user=youruserid password=yourpwd
database=yourdb schema=saslib;
   ```

   For more information, see the “INDCONN Macro Variable” on page 100.

3. Run the %INDB2_PUBLISH_COMPILEUDF macro. For more information, see “%INDB2_PUBLISH_COMPILEUDF Macro Syntax” on page 101.

   You can verify that the SAS.CompileUDF function and global variables have been published successfully. For more information, see “Validating the Publishing of SAS.CompileUDF and SAS_DeleteUDF Functions and Global Variables” on page 106.

   After the SAS.CompileUDF function is published, run the %INDB2_PUBLISH_DELETEUDF publishing macro to create the SAS_DeleteUDF function. For more information, see “Running the %INDB2_PUBLISH_DELETEUDF Macro” on page 103.

INDCONN Macro Variable

The INDCONN macro variable provides the credentials to make a connection to DB2. You must specify the server, user, password, and database information to access the machine on which you have installed the DB2 database. You must assign the INDCONN macro variable before the %INDB2_PUBLISH_COMPILEUDF macro is invoked.

The value of the INDCONN macro variable for the %INDB2_PUBLISH_COMPILEUDF macro has this format.

```
SERVER=server USER=userid PASSWORD=password
DATABASE=database <SCHEMA=SASLIB>
```

**SERVER=server** specifies the DB2 server name or the IP address of the server host. If the server name contains spaces or nonalphanumeric characters, enclose the server name in quotation marks.

**Requirement** The name must be consistent with how the host name was cached when PSFTP server was run from the command window. If the full server name was cached, you must use the full server name in the SERVER argument. If the short server name was cached, you must use the short server name. For example, if the long name, `disk3295.unx.comp.com`, is used when PSFTP was run, then `server=disk3295.unx.comp.com` must be specified. If the short name,
disk3295, was used, then server=disk3295 must be specified. For more information, see “DB2 Installation and Configuration Steps” on page 89.

**USER=userid**

specifies the DB2 user name (also called the user ID) that is used to connect to the database. If the user name contains spaces or nonalphanumeric characters, enclose the user name in quotation marks.

**PASSWORD=password**

specifies the password that is associated with your DB2 user ID. If the password contains spaces or nonalphabetic characters, enclose the password in quotation marks.

**Tip** Use only PASSWORD=, PASS=, or PW= for the password argument. PWD= is not supported and causes an error.

**DATABASE=database**

specifies the DB2 database that contains the tables and views that you want to access. If the database name contains spaces or nonalphanumeric characters, enclose the database name in quotation marks.

**Requirement** The SAS_COMPILEUDF function is created as a Unicode function. If the database is not a Unicode database, then the alternate collating sequence must be configured to use identity_16bit.

**SCHEMA=SASLIB**

specifies SASLIB as the schema name.

**Default** SASLIB

**Restriction** The SAS_COMPILEUDF function and the two global variables (SASUDF_DB2PATH and SASUDF_COMPILER_PATH) are published to the SASLIB schema in the specified database. If a value other than SASLIB is used, it is ignored.

**Requirement** The SASLIB schema must be created before publishing the SAS_COMPILEUDF and SAS_DELETEUDF functions.

---

**%INDB2_PUBLISH_COMPILEUDF Macro Syntax**

**%INDB2_PUBLISH_COMPILEUDF**

```
(DB2PATH=db2instancepath/sqlib
, COMPILER_PATH=compiler-path-directory
<, DATABASE=database-name>
<, ACTION=CREATE | REPLACE | DROP>
<, OBJNAME=object-file-name>
<, OUTDIR=diagnostic-output-directory>
);
```

**Arguments**

**DB2PATH=db2instancepath/sqlib**

specifies the parent directory that contains the function/SAS subdirectory, where all the object files are stored and defines the SASUDF_DB2PATH global variable that is used when publishing the format and scoring model functions.
Interaction: `db2instancepath` should be the same path as the path that was specified during the installation of the SAS_COMPILEUDF binary file. For more information, see Step 3 in “Unpack and Install the SAS Formats Library and Binary Files for DB2” on page 93.

Tip: The SASUDF_DB2PATH global variable is defined in the SASLIB schema under the specified database name.

**COMPILER_PATH=**`compiler-path-directory`

specifies the path to the location of the compiler that compiles the source files and defines the SASUDF_COMPILER_PATH global variable that is used when publishing the format and scoring model functions.

Tip: The SASUDF_COMPILER_PATH global variable is defined in the SASLIB schema under the specified database name. The XLC compiler should be used for AIX, and the GGG compiler should be used for Linux.

**DATABASE=**`database-name`

specifies the name of a DB2 database to which the SAS_COMPILEUDF function is published.

Interaction: The database that you specify in the DATABASE= argument takes precedence over the database that you specify in the INDCONN macro variable. For more information, see “%INDB2_PUBLISH_COMPILEUDF Macro Run Process” on page 100.

**ACTION=**`CREATE | REPLACE | DROP`

specifies that the macro performs one of the following actions:

- **CREATE**
  creates a new SAS_COMPILEUDF function.

- **REPLACE**
  overwrites the current SAS_COMPILEUDF function, if a SAS_COMPILEUDF function by the same name is already registered, or creates a new SAS_COMPILEUDF function if one is not registered.

- **DROP**
  causes the SAS_COMPILEUDF function to be dropped from the DB2 database.

Default: CREATE

Tip: If the SAS_COMPILEUDF function was published previously and you now specify ACTION=CREATE, you receive warning messages from DB2. If the SAS_COMPILEUDF function was published previously and you specify ACTION=REPLACE, no warnings are issued.

**OBJNAME=**`object-file-name`

specifies the object filename that the publishing macro uses to register the SAS_COMPILEUDF function. The object filename is a file system reference to a specific object file, and the value entered for OBJNAME must match the name as it exists in the file system. For example, SAS_CompileUDF is mixed case.

Default: SAS_CompileUDF

Interaction: If the SAS_COMPILEUDF function is updated, you might want to rename the object file to avoid stopping and restarting the database. If so, the SAS_COMPILEUDF function needs to be reregistered with the new object filename.
OUTDIR=diagnostic-output-directory
   specifies a directory that contains diagnostic files.

Tip Files that are produced include an event log that contains detailed information about the success or failure of the publishing process.

Running the %INDB2_PUBLISH_DELETEUDF Macro

Overview of the %INDB2_PUBLISH_DELETEUDF Macro
The %INDB2_PUBLISH_DELETEUDF macro publishes the SAS_DELETEUDF function in the SASLIB schema of a DB2 database. The SAS_DELETEUDF function facilitates the %INDB2_PUBLISH_FORMATS format publishing macro and the %INDB2_PUBLISH_MODEL scoring publishing macro. The SAS_DELETEUDF function removes existing object files when the format or scoring publishing macro registers new ones by the same name.

You have to run the %INDB2_PUBLISH_DELETEUDF macro only one time in a given database.

The SAS_COMPILEUDF function must be published before you run the %INDB2_PUBLISH_DELETEUDF macro, the %INDB2_PUBLISH_FORMATS macro, and the %INDB2_PUBLISH_MODEL macro. Otherwise, these macros fail.

Note: To publish the SAS_DELETEUDF function, you must have the appropriate DB2 user permissions to create and execute this function in the SASLIB schema and specified database. For more information, see “DB2 Permissions” on page 106.

%INDB2_PUBLISH_DELETEUDF Macro Run Process
To run the %INDB2_PUBLISH_DELETEUDF macro, follow these steps:

1. Ensure that you have created a SASLIB schema in the database where the SAS_DELETEUDF function is to be published.

   Use the SASLIB schema when publishing the %INDB2_PUBLISH_DELETEUDF macro for DB2 in-database processing.

   The SASLIB schema should have been created before you ran the %INDB2_PUBLISH_COMPILEUDF macro to create the SAS_COMPILEUDF function. The SASLIB schema contains the SAS_COMPILEUDF and SAS_DELETEUDF functions and the SASUDF_DB2PATH and SASUDF_COMPILER_PATH global variables.

   The SAS_COMPILEUDF function must be published before you run the %INDB2_PUBLISH_DELETEUDF macro. The SAS_COMPILEUDF and SAS_DELETEUDF functions must be published to the SASLIB schema in the same database. For more information about creating the SASLIB schema, see “%INDB2_PUBLISH_COMPILEUDF Macro Run Process” on page 100.

2. Start SAS and submit the following command in the Enhanced Editor or Program Editor.

   %let indconn = server=yourserver user=youruserid password=yourpwd database=yourdb schema=saslib;

   For more information, see the “INDCONN Macro Variable” on page 104.

3. Run the %INDB2_PUBLISH_DELETEUDF macro. For more information, see “%INDB2_PUBLISH_DELETEUDF Macro Syntax” on page 105.
You can verify that the function has been published successfully. For more information, see “Validating the Publishing of SAS_COMPILEUDF and SAS_DELETEUDF Functions and Global Variables” on page 106.

After the SAS_DELETEUDF function is published, the %INDB2_PUBLISH_FORMATS and the %INDB2_PUBLISH_MODEL macros can be run to publish the format and scoring model functions.

**INDCONN Macro Variable**

The INDCONN macro variable provides the credentials to make a connection to DB2. You must specify the server, user, password, and database information to access the machine on which you have installed the DB2 database. You must assign the INDCONN macro variable before the %INDB2_PUBLISH_DELETEUDF macro is invoked.

The value of the INDCONN macro variable for the %INDB2_PUBLISH_DELETEUDF macro has this format.

```
SERVER=server USER=userid PASSWORD=password DATABASE=database <SCHEMA=SASLIB>
```

**SERVER=server**

specifies the DB2 server name or the IP address of the server host. If the server name contains spaces or nonalphanumeric characters, enclose the server name in quotation marks.

**Requirement**

The name must be consistent with how the host name was cached when PSFTP server was run from the command window. If the full server name was cached, use the full server name in the SERVER argument. If the short server name was cached, use the short server name. For example, if the long name, `disk3295.unx.comp.com`, is used when PSFTP was run, then `server=disk3295.unx.comp.com` must be specified. If the short name, `disk3295`, was used, then `server=disk3295` must be specified. For more information, see “DB2 Installation and Configuration Steps” on page 89.

**USER=userid**

specifies the DB2 user name (also called the user ID) that is used to connect to the database. If the user name contains spaces or nonalphanumeric characters, enclose the user name in quotation marks.

**PASSWORD=password**

specifies the password that is associated with your DB2 user ID. If the password contains spaces or nonalphabetic characters, enclose the password in quotation marks.

**Tip**

Use only PASSWORD=, PASS=, or PW= for the password argument. PWD= is not supported and causes errors.

**DATABASE=database**

specifies the DB2 database that contains the tables and views that you want to access. If the database name contains spaces or nonalphanumeric characters, enclose the database name in quotation marks.

**SCHEMA=SASLIB**

specifies SASLIB as the schema name.

**Default**

SASLIB
The SAS_DELETEUDF function is published to the SASLIB schema in the specified database. If a value other than SASLIB is used, it is ignored.

Create the SASLIB schema before publishing the SAS_COMPILEUDF and SAS_DELETEUDF functions.

%INDB2_PUBLISH_DELETEUDF Macro Syntax

%INDB2_PUBLISH_DELETEUDF
(<DATABASE=database-name>
,< ACTION=CREATE | REPLACE | DROP>
,< OUTDIR=diagnostic-output-directory>
);

Arguments

DATABASE=database-name
specifies the name of a DB2 database to which the SAS_DELETEUDF function is published.

INTERACTION
The database that you specify in the DATABASE argument takes precedence over the database that you specify in the INDCONN macro variable. For more information, see “Running the %INDB2_PUBLISH_DELETEUDF Macro” on page 103.

ACTION=CREATE | REPLACE | DROP
specifies that the macro performs one of the following actions:

CREATE
creates a new SAS_DELETEUDF function.

REPLACE
overwrites the current SAS_DELETEUDF function, if a SAS_DELETEUDF function by the same name is already registered, or creates a new SAS_DELETEUDF function if one is not registered.

DROP
causes the SAS_DELETEUDF function to be dropped from the DB2 database.

Default
CREATE

Tip
If the SAS_DELETEUDF function was published previously and you specify ACTION=CREATE, you receive warning messages from DB2. If the SAS_DELETEUDF function was published previously and you specify ACTION=REPLACE, no warnings are issued.

OUTDIR=diagnostic-output-directory
specifies a directory that contains diagnostic files.

Tip
Files that are produced include an event log that contains detailed information about the success or failure of the publishing process.
Validating the Publishing of SAS_COMPILEUDF and SAS_DELETEUDF Functions and Global Variables

To validate that the SAS_COMPILEUDF and SAS_DELETEUDF functions and global variables are created properly, follow these steps.

1. Connect to your DB2 database using Command Line Processor (CLP).

2. Enter the following command to verify that the SASUDF_COMPILER_PATH global variable was published.
   ```
   values(saslib.sasudf_compiler_path)
   ```
   You should receive a result similar to one of the following.
   ```
   /usr/vac/bin /* on AIX */
   /usr/bin /* on Linux */
   ```

3. Enter the following command to verify that the SASUDF_DB2PATH global variable was published.
   ```
   values(saslib.sasudf_db2path)
   ```
   You should receive a result similar to the following.
   ```
   /users/db2v10/sqllib
   ```
   In this example, `/users/db2v10` is the value of `db2instancepath` that was specified during installation and `/users/db2v10/sqllib` is also where the SAS_COMPILEUDF function was published.

4. Enter the following command to verify that the SAS_COMPILEUDF and SAS_DELETEUDF functions were published.
   ```
   select funcname, implementation from syscat.functions where funcschema='SASLIB'
   ```
   You should receive a result similar to the following.
   ```
   FUNCNAME                  IMPLEMENTATION
   ------------------------------------------
   SAS_DELETEUDF             /users/db2v10/sqllib/function/SAS/SAS_DeleteUDF!SAS_DeleteUDF
   SAS_COMPILEUDF            /users/db2v10/sqllib/function/SAS/SAS_CompileUDF!SAS_CompileUDF
   ```

DB2 Permissions

There are two sets of permissions involved with the in-database software.

- The first set of permissions is needed by the person who publishes the SAS_COMPILEUDF and SAS_DELETEUDF functions and creates the SASUDF_COMPILER_PATH and SASUDF_DB2PATH global variables.
These permissions must be granted before the `%INDB2_PUBLISH_COMPILEUDF` and `%INDB2_PUBLISH_DELETEUDF` macros are run. Without these permissions, running these macros fails.

The following table summarizes the permissions that are needed by the person who publishes the functions and creates the global variables.

<table>
<thead>
<tr>
<th>Permission Needed</th>
<th>Authority Required to Grant Permission</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATEIN permission for the SASLIB schema in which the SAS_COMPILEUDF and SAS_DELETEUDF functions are published and the SASUDF_COMPILER_PATH and SASUDF_DB2PATH global variables are defined</td>
<td>System Administrator or Database Administrator</td>
<td>GRANT CREATEIN ON SCHEMA SASLIB TO compileddeletepublisheruserid</td>
</tr>
<tr>
<td>CREATE_EXTERNAL_ROUTINE permission to the database in which the SAS_COMPILEUDF and SAS_DELETEUDF functions are published</td>
<td></td>
<td>GRANT CREATE_EXTERNAL_ROUTINE ON DATABASE TO compileddeletepublisheruserid</td>
</tr>
</tbody>
</table>

- The second set of permissions is needed by the person who publishes the format or scoring model functions. The person who publishes the format or scoring model functions is not necessarily the same person who publishes the SAS_COMPILEUDF and SAS_DELETEUDF functions and creates the SASUDF_COMPILER_PATH and SASUDF_DB2PATH global variables. These permissions are most likely needed by the format publishing or scoring model developer. Without these permissions, the publishing of the format or scoring model functions fails.

  Note: Permissions must be granted for every format or scoring model publisher and for each database that the format or scoring model publishing uses. Therefore, you might need to grant these permissions multiple times.

  Note: If you are using the SAS Embedded Process to run your scoring functions, only the CREATE TABLE permission is needed.

After the DB2 permissions have been set appropriately, the format or scoring publishing macro should be called to register the formats or scoring model functions. The following table summarizes the permissions that are needed by the person who publishes the format or scoring model functions.
<table>
<thead>
<tr>
<th>Permission Needed</th>
<th>Authority Required to Grant Permission</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXECUTE permission for functions that have been published.</td>
<td>System Administrator or Database Administrator</td>
<td>GRANT EXECUTE ON FUNCTION SASLIB.* TO scoringormtpublisherid</td>
</tr>
<tr>
<td>This enables the person who publishes the formats or scoring model functions to</td>
<td></td>
<td></td>
</tr>
<tr>
<td>execute the SAS_COMPILEUDF and SAS_DELETEUDF functions.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Note: If you have SYSADM or DBADM authority, then you have these permissions.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Otherwise, contact your database administrator to obtain these permissions.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CREATE_EXTERNAL_ROUTINE permission to the database to create format or scoring</td>
<td>System Administrator or Database Administrator</td>
<td>GRANT CREATE_EXTERNAL_ROUTINE ON DATABASE TO scoringormtpublisherid</td>
</tr>
<tr>
<td>model functions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CREATE_NOT_FENCED_ROUTINE permission to create format or scoring model functions</td>
<td>System Administrator or Database Administrator</td>
<td>GRANT CREATE_NOT_FENCED_ROUTINE ON DATABASE TO scoringormtpublisherid</td>
</tr>
<tr>
<td>that are not fenced</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CREATEIN permission for the schema in which the format or scoring model functions</td>
<td>System Administrator or Database Administrator</td>
<td>GRANT CREATEIN ON SCHEMA scoringschema TO scoringormtpublisherid</td>
</tr>
<tr>
<td>are published if the default schema (SASLIB) is not used</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CREATE TABLE permission to create the model table used in with scoring and the</td>
<td>Person who ran the %INDB2_PUBLISH_COMPILEUDF macro</td>
<td>GRANT CREATETAB TO scoringpublisherSEPid</td>
</tr>
<tr>
<td>SAS Embedded Process</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Note: The person who ran the %INDB2_PUBLISH_COMPILEUDF macro has these READ</td>
<td>Note: For security reasons, only the user who created these variables has the permission to grant READ</td>
<td></td>
</tr>
<tr>
<td>permissions and does not need to grant them to himself or herself again.</td>
<td>permission to other users. This is true even for the user with administrator permissions such as the</td>
<td></td>
</tr>
<tr>
<td>Note: For security reasons, only the user who created these variables has the</td>
<td>DB2 instance owner.</td>
<td></td>
</tr>
<tr>
<td>permission to grant READ permission to other users. This is true even for the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>user with administrator permissions such as the DB2 instance owner.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>READ permission to read the SASUDF_COMPILERT_PATH and SASUDF_DB2PATH global</td>
<td>Person who ran the %INDB2_PUBLISH_COMPILEUDF macro</td>
<td>GRANT READ ON VARIABLE SASLIB.SASUDF_DB2PATH TO scoringormtpublisherid</td>
</tr>
<tr>
<td>variables</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Note: The person who ran the %INDB2_PUBLISH_COMPILEUDF macro has these READ</td>
<td>Note: For security reasons, only the user who created these variables has the permission to grant READ</td>
<td></td>
</tr>
<tr>
<td>permissions and does not need to grant them to himself or herself again.</td>
<td>permission to other users. This is true even for the user with administrator permissions such as the</td>
<td></td>
</tr>
<tr>
<td>Note: For security reasons, only the user who created these variables has the</td>
<td>DB2 instance owner.</td>
<td></td>
</tr>
<tr>
<td>permission to grant READ permission to other users. This is true even for the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>user with administrator permissions such as the DB2 instance owner.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:** If you plan to use SAS Model Manager with the SAS Scoring Accelerator for in-database scoring, additional permissions are required. For more information, see Chapter 16, “Configuring SAS Model Manager,” on page 163.

### Documentation for Using In-Database Processing in DB2

# Chapter 11
Administrator’s Guide for Greenplum

## In-Database Deployment Package for Greenplum

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prerequisites</td>
<td>109</td>
</tr>
<tr>
<td>Overview of the In-Database Deployment Package for Greenplum</td>
<td>109</td>
</tr>
<tr>
<td>Greenplum Installation and Configuration</td>
<td>111</td>
</tr>
<tr>
<td>Greenplum Installation and Configuration Steps</td>
<td>111</td>
</tr>
<tr>
<td>Upgrading from a Previous Version</td>
<td>111</td>
</tr>
<tr>
<td>Installing the SAS Formats Library and Binary Files</td>
<td>113</td>
</tr>
<tr>
<td>Installing the SAS Embedded Process</td>
<td>114</td>
</tr>
<tr>
<td>Running the <code>%INDGP_PUBLISH_COMPILEUDF</code> Macro</td>
<td>116</td>
</tr>
<tr>
<td>Running the <code>%INDGP_PUBLISH_COMPILEUDF_EP</code> Macro</td>
<td>120</td>
</tr>
<tr>
<td>Validation of Publishing Functions</td>
<td>123</td>
</tr>
<tr>
<td>Validating the Publishing of the SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF Functions</td>
<td>123</td>
</tr>
<tr>
<td>Validating the Publishing of the SAS_EP Function</td>
<td>124</td>
</tr>
<tr>
<td>Controlling the SAS Embedded Process</td>
<td>124</td>
</tr>
<tr>
<td>Semaphore Requirements When Using the SAS Embedded Process for Greenplum</td>
<td>125</td>
</tr>
<tr>
<td>Greenplum Permissions</td>
<td>126</td>
</tr>
<tr>
<td>Documentation for Using In-Database Processing in Greenplum</td>
<td>126</td>
</tr>
</tbody>
</table>

### In-Database Deployment Package for Greenplum

#### Prerequisites

SAS Foundation and the SAS/ACCESS Interface to Greenplum must be installed before you install and configure the in-database deployment package for Greenplum.

The SAS Scoring Accelerator for Greenplum requires a specific version of the Greenplum client and server environment and the Greenplum Partner Connector (GPPC) API. For more information, see the SAS Foundation system requirements documentation for your operating environment.
Overview of the In-Database Deployment Package for Greenplum

This section describes how to install and configure the in-database deployment package for Greenplum (SAS Formats Library for Greenplum and the SAS Embedded Process).

The in-database deployment package for Greenplum must be installed and configured before you can perform the following tasks:

- Use the `%INDGP_PUBLISH_FORMATS` format publishing macro to create or publish the `SAS_PUT( )` function and to create or publish user-defined formats as format functions inside the database.
- Use the `%INDGP_PUBLISH_MODEL` scoring publishing macro to create scoring files and functions inside the database.
- Use the SAS In-Database Code Accelerator for Greenplum to execute DS2 thread programs in parallel inside the database.

For more information, see the SAS DS2 Language Reference.

- Run SAS High-Performance Analytics when the analytics cluster is co-located with the Greenplum data appliance or when the analytics cluster is using a parallel connection with a remote Greenplum data appliance. The SAS Embedded Process, which resides on the data appliance, is used to provide high-speed parallel data transfer between the data appliance and the analytics environment where it is processed.

For more information, see the SAS High-Performance Analytics Infrastructure: Installation and Configuration Guide.

For more information about using the format and scoring publishing macros, see the SAS In-Database Products: User’s Guide.

The in-database deployment package for Greenplum contains the SAS formats library and precompiled binary files for the utility functions. The package also contains the SAS Embedded Process.

The SAS formats library is a run-time library that is installed on your Greenplum system. This installation is done so that the SAS scoring model functions and the `SAS_PUT( )` function created in Greenplum can access the routines within the run-time library. The SAS formats library contains the formats that are supplied by SAS.

The `%INDGP_PUBLISH_COMPILEUDF` macro registers utility functions in the database. The utility functions are called by the format and scoring publishing macros: `%INDGP_PUBLISH_FORMATS` and `%INDGP_PUBLISH_MODEL`. You must run the `%INDGP_PUBLISH_COMPILEUDF` macro before you run the format and scoring publishing macros.

The SAS Embedded Process is a SAS server process that runs within Greenplum to read and write data. The SAS Embedded Process contains the `%INDGP_PUBLISH_COMPILEUDF_EP` macro, run-time libraries, and other software that is installed on your Greenplum system. The `%INDGP_PUBLISH_COMPILEUDF_EP` macro defines the SAS_EP table functions to the Greenplum database. You use the SAS_EP table function to produce scoring models after you run the `%INDGP_PUBLISH_MODEL` macro to create the SAS scoring files and publish them to the scoring model table. The SAS Embedded Process accesses the SAS scoring files when a scoring operation is performed. You also use the SAS_EP table function for other SAS software that requires it, such as SAS High-Performance Analytics.
Greenplum Installation and Configuration

Greenplum Installation and Configuration Steps

1. If you are upgrading from a previous release, follow the instructions in “Upgrading from a Previous Version” on page 111 before installing the in-database deployment package.

2. Install the SAS formats library and the binary files.
   For more information, see “Installing the SAS Formats Library and Binary Files” on page 113.

3. Install the SAS Embedded Process.
   For more information, see “Installing the SAS Embedded Process” on page 114.

4. Run the %INDGP_PUBLISH_COMPILEUDF macro if you want to publish formats or use scoring functions to run a scoring model. Run the %INDGP_PUBLISH_COMPILEUDF_EP macro if you want to use the SAS Embedded Process to run a scoring model or other SAS software that requires it.
   For more information, see “Running the %INDGP_PUBLISH_COMPILEUDF Macro” on page 116 or “Running the %INDGP_PUBLISH_COMPILEUDF_EP Macro” on page 120.

5. If you plan to use SAS Model Manager with the SAS Scoring Accelerator for in-database scoring, perform the additional configuration tasks provided in Chapter 16, “Configuring SAS Model Manager,” on page 163.

   Note: If you are installing the SAS High-Performance Analytics environment, there are additional steps to be performed after you install the SAS Embedded Process. For more information, see SAS High-Performance Analytics Infrastructure: Installation and Configuration Guide.

Upgrading from a Previous Version

Upgrading the 9.3 SAS Formats Library and SAS Embedded Process
To upgrade from the SAS 9.3 version, follow these steps:

1. Delete the full-path-to-pkglibdir/SAS directory that contains the SAS Formats Library and the SAS Embedded Process.

   Note: You can use the following command to determine the full-path-to-pkglibdir directory.

   \[ \text{pg_config --pkglibdir} \]

   If you did not perform the Greenplum install, you cannot run the pg_config --pkglibdir command. The pg_config --pkglibdir command must be run by the person who performed the Greenplum installation.

   CAUTION:

   If you delete the SAS directory, all the scoring models that you published using scoring functions and all user-defined formats that you published are
If you previously published scoring models using scoring functions or if you previously published user-defined formats, you must republish your scoring models and formats. If you used the SAS Embedded Process to publish scoring models, the scoring models are not deleted.

It is a best practice to delete the SAS directory when you upgrade from a previous version. Doing so ensures that you get the latest version of both the SAS Formats Library and the SAS Embedded Process.

2. Continue the installation instructions in “Installing the SAS Formats Library and Binary Files” on page 113.

Upgrading the 9.4 SAS Formats Library and SAS Embedded Process

To upgrade from the SAS 9.4 version, follow these steps. If you upgrade or install the SAS Formats Library and the SAS Embedded Process in this manner, you do not delete any scoring models or formats that were previously published.

1. Log on to the Greenplum master node as a superuser.
2. Run the UninstallSASEPFiles.sh file.

   ./UninstallSASEPFiles.sh

   This script stops the SAS Embedded Process on each database host node. The script deletes the /SAS/SASTKInDatabaseServerForGreenplum directory and all its contents from each database host node.

   Prior to SAS 9.4M4 (November 2016), the UninstallSASEPFiles.sh file is in the path_to_sh_file directory where you copied the tkindbsrv self-extracting archive file.

   Starting with SAS 9.4M4 (November 2016) and later, the UninstallSASEPFiles.sh file is in the path_to_sh_file/SASEPHome/bin directory.

   **CAUTION:**
   The timing option must be off for the UninstallSASEPFiles.sh scripts to work. Put \timing off in your .psqlrc file before running this script.

3. Move to the directory where the SAS Formats Library is installed.

   The directory path is full-path-to-pkglibdir/SAS/.

   **Note:** You can use the following command to determine the full-path-to-pkglibdir directory.

   pg_config --pkglibdir

   If you did not perform the Greenplum install, you cannot run the pg_config --pkglibdir command. The pg_config --pkglibdir command must be run by the person who performed the Greenplum install.

4. Delete the libjazxfbrs.so and sas_compileudf.so files.

5. In addition to deleting the libjazxfbrs.so and sas_compileudf.so files on the master node, you must log on to each host node and delete the files on these nodes.

6. Continue the installation instructions in “Installing the SAS Formats Library and Binary Files” on page 113.
**Installing the SAS Formats Library and Binary Files**

*Move and Install the SAS Formats Library and Binary Files*

The SAS formats library and the binary files for the publishing macros are contained in a self-extracting archive file. The self-extracting archive file is located in the `YourSASDepot/SASFormatsLibraryforGreenplum/3.1/GreenplumonLinux64/` directory.

To move and unpack the self-extracting archive file, follow these steps:

1. Using a method of your choice, transfer the `accelgplmfmt-3.1-n_lax.sh` file to your Greenplum master node. `n` is a number that indicates the latest version of the file. If this is the initial installation, `n` has a value of 1. Each time you re-install or upgrade, `n` is incremented by 1.

   The file does not have to be downloaded to a specific location. However, you should note where the file is downloaded so that it can be executed at a later time.

2. After the `accelgplmfmt-3.1-n_lax.sh` has been transferred, log on to the Greenplum master node as a superuser.

3. Move to the directory where the self-extracting archive file was downloaded.

4. Use the following command at the UNIX prompt to unpack the self-extracting archive file:

   ```bash
   ./accelgplmfmt-3.1-n_lax.sh
   ```

   *Note:* If you receive a “permissions denied” message, check the permissions on the `accelgplmfmt-3.1-n_lax.sh` file. This file must have EXECUTE permissions to run.

   After the script runs and the files are unpacked, the content of the target directories should look similar to these where `path_to_sh_file` is the location to which you copied the self-extracting archive file.

   ```
   /path_to_sh_file/SAS/SASFormatsLibraryForGreenplum/3.1-1/bin/
   InstallAccelGplmFmt.sh
   /path_to_sh_file/SAS/SASFormatsLibraryForGreenplum/3.1-1/bin/
   CopySASFiles.sh
   /path_to_sh_file/SAS/SASFormatsLibraryForGreenplum/3.1-1/lib/
   SAS_CompileUDF.so
   /path_to_sh_file/SAS/SASFormatsLibraryForGreenplum/3.1-1/lib/
   libjazxfbrs.so
   ```

5. Use the following command to place the files in Greenplum:

   ```bash
   ./path_to_sh_file/SAS/SASFormatsLibraryForGreenplum/3.1-1/bin/
   CopySASFiles.sh
   ```

   **CAUTION:**

   The timing option must be off for the `CopySASFiles.sh` script to work. Put `\timing off` in your `.psqlrc` file before running this script.

   This command replaces all previous versions of the `libjazxfbrs.so` file.

   All the SAS object files are stored under `full-path-to-pkglibdir/SAS`. The files are copied to the master node and each of the segment nodes.

   *Note:* You can use the following command to determine the `full-path-to-pkglibdir` directory:
If you did not perform the Greenplum install, you cannot run the `pg_config --pkglibdir` command. The `pg_config --pkglibdir` command must be run by the person who performed the Greenplum install.

*Note:* If you add new nodes at a later date, you must copy all the binary files to the new nodes. For more information, see Step 6.

6. (Optional) If you add new nodes to the Greenplum master node after the initial installation of the SAS formats library and publishing macro, you must copy all the binaries in the `full-path-to-pkglibdir/SAS` directory to the new nodes using a method of your choice such as `scp /SAS`. The binary files include `SAS_CompileUDF.so`, `libjazxfbrs.so`, and the binary files for the already published functions.

### Installing the SAS Embedded Process

#### Unzipping the SAS Embedded Process for Greenplum

The SAS Embedded Process is contained in a self-extracting archive file named `sepcoregplm-13.00000-1.sh`. This file is contained in a ZIP file that is put in your SAS Software Depot directory. You need this file if you want to run your scoring model.

To unzip the SAS Embedded Process file for Greenplum, follow these steps:

1. Create a new temporary directory on your client machine such as `/sasep`. The new directory is referred to as `EPZipDir` throughout this section.

2. Navigate to the `YourSASDepot/standalone_installs` directory. This directory was created when you created your SAS Software Depot.

3. Locate the `en_sasexe.zip` file. The `en_sasexe.zip` file is located in the `YourSASDepot/standalone_installs/SAS_Core_Embedded_Process_Package_for_Greenplum/13_0/Greenplum_on_Linux_x64/` directory.

   The `sepcoregplm-13.00000-1.sh` file is included in this ZIP file.

4. Copy the `en_sasexe.zip` file to your `EPZipDir` temporary directory on the client machine.

   ```
   cp en_sasexe.zip /EPZipDir
   ```

5. Navigate to your `EPZipDir` temporary directory and unzip `en_sasexe.zip`.

   After the file is unzipped, a `sasexe` directory is created in the same location as the `en_sasexe.zip` file. The self-extracting archive file is in the `/EPZipDir/sasexe` directory.

#### Moving and Installing the SAS Embedded Process


To move and unpack the self-extracting archive file, follow these steps:

1. Using a method of your choice, transfer the `sepcoregplm-13.00000-1.sh` file to your Greenplum master node. This directory is referred to as the `path_to_sh_file`.

   The file does not have to be downloaded to a specific location. However, you need to note where it is downloaded so that it can be executed at a later time.
2. After the sepcoregplm-13.00000-1.sh file has been transferred, log on to the Greenplum master node as a superuser.

3. Move to the directory where the self-extracting archive file was downloaded.

4. Change permissions on the file to enable you to execute the script.
   
   ```sh
cmd 755 -r sepcoregplm-13.00000-1.sh
   ```

5. Use the following command at the UNIX prompt to unpack the self-extracting archive file.
   
   ```sh
./sepcoregplm-13.00000-1.sh
   ```

   After the script runs and the files are unpacked, the contents of the target directories should look similar to these. `path_to_sh_file` is the location to which you copied the self-extracting archive file in Step 1.

   ```
   /path_to_sh_file/SASEPHome/bin
   /path_to_sh_file/SASEPHome/misc
   /path_to_sh_file/SASEPHome/sasexe
   /path_to_sh_file/SASEPHome/utilities
   ```

   The `path_to_sh_file/SASEPHome/bin/` directory contains several scripts:

   ```
   /path_to_sh_file/SASEPHome/bin/InstallSASEPFiles.sh
   /path_to_sh_file/SASEPHome/bin/UninstallSASEPFiles.sh
   /path_to_sh_file/SASEPHome/bin/StartupSASEP.sh
   /path_to_sh_file/SASEPHome/bin/ShutdownSASEP.sh
   /path_to_sh_file/SASEPHome/bin/ShowSASEPStatus.sh
   ```

   The `InstallSASEPFiles.sh` file installs the SAS Embedded Process. The next step explains how to run this file. The `StartupSASEP.sh` and `ShutdownSASEP.sh` files enable you to manually start and stop the SAS Embedded Process. For more information about running these two files, see “Controlling the SAS Embedded Process” on page 124.

   The `UninstallSASEPFiles.sh` file uninstalls the SAS Embedded Process. The `ShowEPFilesStatus.sh` file shows the status of the SAS Embedded Process on each host.

6. Use the following commands at the UNIX prompt to install the SAS Embedded Process to the master node and all host nodes.

   The `InstallSASEPFiles.sh` file must be run from the `path_to_sh_file/SASEPHome/bin/` directory.

   ```sh
cd /path_to_sh_file/SASEPHome/bin
./InstallSASEPFiles.sh <-quiet>
   ```

   **CAUTION:**

   The timing option must be off for the `InstallSASEPFiles.sh` script to work.
   
   Put `\timing off` in your `.psqlrc` file before running this script.

   **Note:** `-verbose` is on by default and enables you to see all messages that are generated during the installation process. Specify `-quiet` to suppress messages.

   The installation also creates a `full-path-to-pkglibdir/SASEPHome/` directory. This directory is created on the master node and each host node.

   The installation also copies the SAS directories and files from Step 4 across every node.

   The contents of the `full-path-to-pkglibdir/SASEPHome` directory should look similar to these.
full-path-to-pkglibdir/SASEPHome/bin
full-path-to-pkglibdir/SASEPHome/misc
full-path-to-pkglibdir/SASEPHome/sasexe
full-path-to-pkglibdir/SASEPHome/utilities

Note: You can use the following command to determine the
full-path-to-pkglibdir directory:

```
pg_config --pkglibdir
```

If you did not perform the Greenplum install, you cannot run the `pg_config --pkglibdir` command. The `pg_config --pkglibdir` command must be run by the person who performed the Greenplum install.

This is an example of the SAS Embedded Process install location:

```
usr/local/greenplum-db-4.2.6.1/lib/postgresql/SASEPHome
```

Running the %INDGP_PUBLISH_COMPILEUDF Macro

**Overview of the %INDGP_PUBLISH_COMPILEUDF Macro**

Use the %INDGP_PUBLISH_COMPILEUDF macro if you want to use scoring functions to run scoring models.

Note: Use the %INDGP_PUBLISH_COMPILEUDF_EP macro if you need to use the SAS Embedded Process. For more information, see “Running the %INDGP_PUBLISH_COMPILEUDF_EP Macro” on page 120.

The %INDGP_PUBLISH_COMPILEUDF macro publishes the following functions to the SASLIB schema in a Greenplum database:

- **SAS_COMPILEUDF function**
  
  This function facilitates the %INDGP_PUBLISH_FORMATS format publishing macro and the %INDGP_PUBLISH_MODEL scoring publishing macro. The SAS_COMPILEUDF function performs the following tasks:
  
  - compiles the format and scoring model source files into object files. This compilation occurs through the SQL interface using an appropriate compiler for the system.
  
  - links with the SAS formats library.
  
  - copies the object files to the full-path-to-pkglibdir/SAS directory. All the SAS object files are stored under full-path-to-pkglibdir/SAS.

  Note: You can use the following command to determine the full-path-to-pkglibdir directory:

  ```
  pg_config --pkglibdir
  ```

  If you did not perform the Greenplum install, you cannot run the `pg_config --pkglibdir` command. The `pg_config --pkglibdir` command must be run by the person who performed the Greenplum install.

- Three utility functions that are used when the scoring publishing macro transfers source files from the client to the host:
  
  - **SAS_COPYUDF function**
This function copies the shared libraries to the

**full-path-to-pkglibdir/SAS** path on the whole database array including
the master and all segments.

- **SAS_DIRECTORYUDF function**

  This function creates and removes a temporary directory that holds the source
  files on the server.

- **SAS_DEHEXUDF function**

  This function converts the files from hexadecimal back to text after the files are
  exported on the host.

You have to run the `%INDGP_PUBLISH_COMPILEUDF` macro only one time in each
database.

**Note:** The SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and
SAS_DEHEXUDF functions must be published before you run the
%INDGP_PUBLISH_FORMATS or the %INDGP_PUBLISH_MODEL macro. Otherwise, these macros fail.

**Note:** To publish the SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF,
and SAS_DEHEXUDF functions, you must have superuser permissions to create and
execute these functions in the SASLIB schema and in the specified database.

**%INDGP_PUBLISH_COMPILEUDF Macro Run Process**

To run the `%INDGP_PUBLISH_COMPILEUDF` macro, follow these steps:

**Note:** To publish the SAS_COMPILEUDF function, you must have superuser
permissions to create and execute this function in the SASLIB schema and in the
specified database.

1. Create a SASLIB schema in the database where the SAS_COMPILEUDF,
   SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF functions are
   published.

   You must use “SASLIB” as the schema name for Greenplum in-database processing
to work correctly.

   You specify that database in the DATABASE argument of the
   %INDGP_PUBLISH_COMPILEUDF macro. For more information, see
   “%INDGP_PUBLISH_COMPILEUDF Macro Syntax” on page 119.

   The SASLIB schema contains the SAS_COMPILEUDF, SAS_COPYUDF,
   SAS_DIRECTORYUDF, and SAS_DEHEXUDF functions.

2. Start SAS 9.4 and submit the following command in the Enhanced Editor or Program
   Editor:

   ```
   %let indconn = user=youruserid password=yourpwd dsn=yourdsn;
   /* You can use server=yourserver database=yourdb instead of dsn=yourdsn */
   
   For more information, see the “INDCONN Macro Variable” on page 118.
   
   3. Run the `%INDGP_PUBLISH_COMPILEUDF` macro.

   For more information, see “%INDGP_PUBLISH_COMPILEUDF Macro Syntax” on
   page 119.
   
   You can verify that the SAS_COMPILEUDF, SAS_COPYUDF,
   SAS_DIRECTORYUDF, and SAS_DEHEXUDF functions have been published
   successfully. For more information, see “Validating the Publishing of the
The INDCONN macro variable provides the credentials to make a connection to Greenplum. You must specify the user, password, and either the DSN or server and database information to access the machine on which you have installed the Greenplum database. You must assign the INDCONN macro variable before the %INDGP_PUBLISH_COMPILEUDF macro is invoked.

The value of the INDCONN macro variable for the %INDGP_PUBLISH_COMPILEUDF macro has one of these formats:

```
USER=<userid>  PASSWORD=<password>  DSN=<dsnname>
<PORT=<port-number>>

USER=<userid>  PASSWORD=<password>  SERVER=<server>
DATABASE=<database>  <PORT=<port-number>>
```

- **USER=** specifies the Greenplum user name (also called the user ID) that is used to connect to the database. If the user name contains spaces or nonalphanumeric characters, enclose the user name in quotation marks.
- **PASSWORD=** specifies the password that is associated with your Greenplum user ID. If the password contains spaces or nonalphabetic characters, enclose the password in quotation marks.
  
  **Tip**  Use only PASSWORD=, PASS=, or PW= for the password argument. PWD= is not supported and causes an error.

- **DSN=** specifies the configured Greenplum ODBC data source to which you want to connect. If the DSN name contains spaces or nonalphabetic characters, enclose the DSN name in quotation marks.
  
  **Requirement**  You must specify either the DSN= argument or the SERVER= and DATABASE= arguments in the INDCONN macro variable.

- **SERVER=** specifies the Greenplum server name or the IP address of the server host. If the server name contains spaces or nonalphanumeric characters, enclose the server name in quotation marks.
  
  **Requirement**  You must specify either the DSN= argument or the SERVER= and DATABASE= arguments in the INDCONN macro variable.

- **DATABASE=** specifies the Greenplum database that contains the tables and views that you want to access. If the database name contains spaces or nonalphabetic characters, enclose the database name in quotation marks.
  
  **Requirement**  You must specify either the DSN= argument or the SERVER= and DATABASE= arguments in the INDCONN macro variable.

- **PORT=** specifies the psql port number.
Default 5432

Requirement The server-side installer uses psql, and psql default port is 5432. If you want to use another port, you must have the UNIX or database administrator change the psql port.

**Note:** The SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF functions are published to the SASLIB schema in the specified database. The SASLIB schema must be created before publishing the SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF functions.

%%INDGP_PUBLISH_COMPILEUDF Macro Syntax

%%INDGP_PUBLISH_COMPILEUDF

(OBJPATH=full-path-to-pkglibdir/SAS
  <, DATABASE=database-name>
  <, ACTION=CREATE | REPLACE | DROP>
  <, OUTDIR=diagnostic-output-directory>
  );

**Arguments**

**OBJPATH=full-path-to-pkglibdir/SAS** specifies the parent directory where all the object files are stored.

**Tip** The full-path-to-pkglibdir directory was created during installation of the InstallAccelGplmFmt.sh self-extracting archive file. You can use the following command to determine the full-path-to-pkglibdir directory:

```
pg_config --pkglibdir
```

If you did not perform the Greenplum install, you cannot run the pg_config --pkglibdir command. The pg_config --pkglibdir command must be run by the person who performed the Greenplum install.

**DATABASE=database-name** specifies the name of a Greenplum database to which the SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF functions are published.

**Restriction** If you specify DSN= in the INDCONN macro variable, do not use the DATABASE argument.

**ACTION=CREATE | REPLACE | DROP** specifies that the macro performs one of the following actions:

**CREATE**

creates a new SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF function.

**REPLACE**

overwrites the current SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF functions, if a function by the same name is already registered, or creates a new SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF function if one is not registered.

**Requirement** If you are upgrading the SAS Formats Library, run the %%INDGP_PUBLISH_COMPILEUDF macro with
ACTION=REPLACE. The CopySASFiles.sh install script replaces existing versions of most files. However, you need to replace the existing SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF functions after you run the CopySASFiles.sh install script. For more information, see “Upgrading from a Previous Version” on page 111 and “Move and Install the SAS Formats Library and Binary Files” on page 113.

DROP causes the SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF functions to be dropped from the Greenplum database.

Default CREATE

Tip If the SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF functions were published previously and you specify ACTION=CREATE, you receive warning messages that the functions already exist and you are prompted to use REPLACE. If the SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF functions were published previously and you specify ACTION=REPLACE, no warnings are issued.

OUTDIR=diagnostic-output-directory specifies a directory that contains diagnostic files.

Tip Files that are produced include an event log that contains detailed information about the success or failure of the publishing process.

Running the %INDGP_PUBLISH_COMPILEUDF_EP Macro

Overview of the %INDGP_PUBLISH_COMPILEUDF_EP Macro
Use the %INDGP_PUBLISH_COMPILEUDF_EP macro if you want to use the SAS Embedded Process to run scoring models or other SAS software that requires it.

Note: Use the %INDGP_PUBLISH_COMPILEUDF macro if you want to use scoring functions to run scoring models. For more information, see “Running the %INDGP_PUBLISH_COMPILEUDF Macro” on page 116.

The %INDGP_PUBLISH_COMPILEUDF_EP macro registers the SAS_EP table functions in the database.

You have to run the %INDGP_PUBLISH_COMPILEUDF_EP macro only one time in each database where scoring models are published.

The %INDGP_PUBLISH_COMPILEUDF_EP macro must be run before you use the SAS_EP function in an SQL query.

Note: To publish the SAS_EP function, you must have superuser permissions to create and execute this function in the specified schema and database.

%INDGP_PUBLISH_COMPILEUDF_EP Macro Run Process
To run the %INDGP_PUBLISH_COMPILEUDF_EP macro, follow these steps:

Note: To publish the SAS_EP function, you must have superuser permissions to create and execute this function in the specified schema and database.
1. Create a schema in the database where the SAS_EP function is published.

   Note: You must publish the SAS_EP function to a schema that is in your schema search path.

   You specify the schema and database in the INDCONN macro variable. For more information, see “INDCONN Macro Variable” on page 121.

2. Start SAS 9.4 and submit the following command in the Enhanced Editor or Program Editor:

   `%let indconn = user=youruserid password=yourpwd dsn=yourdsn <schema=yourschema>;`  
   /* You can use server=yourserver database=yourdb instead of dsn=yourdsn */

   For more information, see the “INDCONN Macro Variable” on page 121.

3. Run the `%INDGP_PUBLISH_COMPILEUDF_EP` macro. For more information, see “%INDGP_PUBLISH_COMPILEUDF_EP Macro Syntax” on page 122.

   You can verify that the SAS_EP function has been published successfully. For more information, see “Validating the Publishing of the SAS_EP Function” on page 124.

**INDCONN Macro Variable**

The INDCONN macro variable provides the credentials to make a connection to Greenplum. You must specify the user, password, and either the DSN or server and database information to access the machine on which you have installed the Greenplum database. You must assign the INDCONN macro variable before the `%INDGP_PUBLISH_COMPILEUDF_EP` macro is invoked.

The value of the INDCONN macro variable for the `%INDGP_PUBLISH_COMPILEUDF_EP` macro has one of these formats:

USER=<user><PASSWORD=<password> DSN=<dsname>  
<SCHEMA=<schema> <PORT=<port-number>>

USER=<user><PASSWORD=<password> SERVER=<server>  
DATABASE=<database> <SCHEMA=<schema>>  
<PORT=<port-number>>

USER=<user>

specifies the Greenplum user name (also called the user ID) that is used to connect to the database. If the user name contains spaces or nonalphanumeric characters, enclose the user name in quotation marks.

PASSWORD=<password>

specifies the password that is associated with your Greenplum user ID. If the password contains spaces or nonalphabetic characters, enclose the password in quotation marks.

Tip Use only PASSWORD=, PASS=, or PW= for the password argument. PWD= is not supported and causes an error.

DSN=<datasource>

specifies the configured Greenplum ODBC data source to which you want to connect. If the DSN name contains spaces or nonalphabetic characters, enclose the DSN name in quotation marks.

Requirement You must specify either the DSN= argument or the SERVER= and DATABASE= arguments in the INDCONN macro variable.
SERVER=<server>
spcifies the Greenplum server name or the IP address of the server host. If the server name contains spaces or nonalphanumeric characters, enclose the server name in quotation marks.

Requirement You must specify either the DSN= argument or the SERVER= and DATABASE= arguments in the INDCONN macro variable.

DATABASE=<database>
spcifies the Greenplum database that contains the tables and views that you want to access. If the database name contains spaces or nonalphanumeric characters, enclose the database name in quotation marks.

Requirement You must specify either the DSN= argument or the SERVER= and DATABASE= arguments in the INDCONN macro variable.

SCHEMA= <schema>
spcifies the name of the schema where the SAS_EP function is defined.

Default SASLIB

Requirements You must create the schema in the database before you run the %INDGP_PUBLISH_COMPILEUDF_EP macro.

You must publish the SAS_EP function to a schema that is in your schema search path.

PORT=<port-number>
spcifies the psql port number.

Default 5432

Requirement The server-side installer uses psql, and psql default port is 5432. If you want to use another port, you must have the UNIX or database administrator change the psql port.

%INDGP_PUBLISH_COMPILEUDF_EP Macro Syntax
%INDGP_PUBLISH_COMPILEUDF_EP
(<OBJPATH=full-path-to-pkglibdir/SAS>
<, DATABASE=database-name>
<, ACTION=CREATE | REPLACE | DROP>
<, OUTDIR=diagnostic-output-directory>
);

Arguments

OBJPATH=full-path-to-pkglibdir/SAS
spcifies the parent directory where all the object files are stored.

Tip The full-path-to-pkglibdir directory was created during installation of the InstallSASEP.sh self-extracting archive file. You can use the following command to determine the full-path-to-pkglibdir directory:

```
pg_config --pkglibdir
```

If you did not perform the Greenplum install, you cannot run the `pg_config --pkglibdir` command. The `pg_config --pkglibdir` command must be run by the person who performed the Greenplum install.
**DATABASE=database-name**
specifies the name of a Greenplum database where the SAS_EP function is defined.

Restriction  
If you specify DSN= in the INDCONN macro variable, do not use the DATABASE argument.

**ACTION=CREATE | REPLACE | DROP**
specifies that the macro performs one of the following actions:

CREATE  
creates a new SAS_EP function.

REPLACE  
overwrites the current SAS_EP function, if a function by the same name is already registered, or creates a new SAS_EP function if one is not registered.

Requirement  
If you are upgrading the SAS Embedded Process, run the %INDGP_PUBLISH_COMPILEUDF_EP macro with ACTION=REPLACE. The InstallSASEPFfiles.sh install script replaces existing versions of most files. However, you need to replace the existing SAS_EP function after you run the InstallSASEPFfiles.sh install script. For more information, see “Upgrading from a Previous Version” on page 111 and “Moving and Installing the SAS Embedded Process” on page 114.

DROP  
causes the SAS_EP function to be dropped from the Greenplum database.

Default  
CREATE

Tip  
If the SAS_EP function was defined previously and you specify ACTION=CREATE, you receive warning messages that the functions already exist and you are prompted to use REPLACE. If the SAS_EP function was defined previously and you specify ACTION=REPLACE, no warnings are issued.

**OUTDIR=diagnostic-output-directory**
specifies a directory that contains diagnostic files.

Tip  
Files that are produced include an event log that contains detailed information about the success or failure of the publishing process.

---

**Validation of Publishing Functions**

**Validating the Publishing of the SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF Functions**

To validate that the SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF functions are registered properly under the SASLIB schema in the specified database, follow these steps.

1. Use psql to connect to the database.
Validating the Publishing of the SAS_EP Function

To validate that the SAS_EP function is registered properly under the specified schema in the specified database, follow these steps.

1. Use psql to connect to the database.

```bash
psql -d databasename
```

You should receive the following prompt.

`databasename=#`

2. At the prompt, enter the following command.

```sql
select prosrc from pg_proc f, pg_namespace s where f.pronamespace=s.oid
and upper(s.nspname)='SASLIB';
```

You should receive a result similar to the following:

- SAS_CompileUDF
- SAS_CopyUDF
- SAS_DirectoryUDF
- SAS_DehexUDF

3. Exit psql.

```bash
\q
```

Controlling the SAS Embedded Process

The SAS Embedded Process starts when a query is submitted using the SAS_EP function. It continues to run until it is manually stopped or the database is shut down.

*Note:* Starting and stopping the SAS Embedded Process has implications for all scoring model publishers.
Note: Manually starting and stopping the SAS Embedded Process requires superuser permissions and must be done from the Greenplum master node.

When the SAS Embedded Process is installed, the ShutdownSASEP.sh and StartupSASEP.sh scripts are installed in the following directory. For more information about these files, see “Moving and Installing the SAS Embedded Process” on page 114.

```
/path_to_sh_file/SASEPHome/bin
```

Use the following command to shut down the SAS Embedded Process:

```
/path_to_sh_file/SASEPHome/bin/ShutdownSASEP.sh
```

When invoked from the master node, ShutdownSASEP.sh shuts down the SAS Embedded Process on each database node. The -verbose option is on by default and provides a status of the shutdown operations as they occur. You can specify the -quiet option to suppress messages. This script should not be used as part of the normal operation. It is designed to be used to shut down the SAS Embedded Process prior to a database upgrade or re-install.

Use the following command to start the SAS Embedded Process:

```
/path_to_sh_file/SASEPHome/bin/StartupSASEP.sh
```

When invoked from the master node, StartupSASEP.sh manually starts the SAS Embedded Process on each database node. The -verbose option is on by default and provides a status of the installation as it occurs. You can specify the -quiet option to suppress messages. This script should not be used as part of the normal operation. It is designed to be used to manually start the SAS Embedded Process and only after consultation with SAS Technical Support.

**CAUTION:**

The timing option must be off for any of the .sh scripts to work. Put \texttt{timing off} in your .psqrlc file before running these scripts.

---

**Semaphore Requirements When Using the SAS Embedded Process for Greenplum**

Each time a query using a SAS_EP table function is invoked to execute a score, it requests a set of semaphore arrays (sometimes referred to as semaphore "sets") from the operating system. The SAS Embedded Process releases the semaphore arrays back to the operating system after scoring is complete.

The number of semaphore arrays required for a given SAS Embedded Process execution is a function of the number of Greenplum database segments that are engaged for the query. The Greenplum system determines the number of segments to engage as part of its query plan based on a number of factors, including the data distribution across the appliance.

The SAS Embedded Process requires five semaphore arrays per database segment that is engaged. The maximum number of semaphore arrays required per database host node per SAS Embedded Process execution can be determined by the following formula:

```
maximum_number_semaphore_arrays = 5 \times number_database_segments
```

Here is an example. On a full-rack Greenplum appliance configured with 16 host nodes and six database segment servers per node, a maximum of 30 (5 \times 6) semaphore arrays...
are required on each host node per concurrent SAS Embedded Process execution of a score. If the requirement is to support the concurrent execution by the SAS Embedded Process of 10 scores, then the SAS Embedded Process requires a maximum of 300 (5* 6 * 10) semaphore arrays on each host node.

SAS recommends that you configure the semaphore array limit on the Greenplum appliance to support twice the limit that is configured by default on the appliance. For example, if the default limit is 2048, double the default limit to 4096.

Note: The semaphore limit discussed here is the limit on the number of "semaphore arrays", where each semaphore array is allocated with an application-specified number of semaphores. For the SAS Embedded Process, the limit on the number of semaphore arrays is distinct from the limit on the "maximum number of semaphores system wide". The SAS Embedded Process requests semaphore arrays with two or fewer semaphores in each array. The limit on the maximum semaphores system wide should not need to be increased. The Linux $ ipcs -sl command output shows the typical default semaphore-related limits set on a Greenplum appliance:

```
------- Semaphore Limits -------
max number of arrays = 2048
max semaphores per array = 250
max semaphores system wide = 512000
max ops per semop call = 100
semaphore max value = 32767
```

Greenplum Permissions

To publish the utility (SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, SAS_DEHEXUDF, SAS_EP), format, and scoring model functions, Greenplum requires that you have superuser permissions to create and execute these functions in the SASLIB (or other specified) schema and in the specified database.

In addition to Greenplum superuser permissions, you must have CREATE TABLE permission to create a model table when using the SAS Embedded Process.

If you plan to use SAS Model Manager with the SAS Scoring Accelerator for in-database scoring, additional permissions are required. For more information, see Chapter 16, “Configuring SAS Model Manager,” on page 163.

Documentation for Using In-Database Processing in Greenplum


For information about how to use the SAS In-Database Code Accelerator, see the SAS DS2 Language Reference, located at http://support.sas.com/documentation/ onlinedoc/base/index.html.
In-Database Deployment Package for Netezza

Prerequisites

SAS Foundation and the SAS/ACCESS Interface to Netezza must be installed before you install and configure the in-database deployment package for Netezza.

The SAS Scoring Accelerator for Netezza and the SAS Embedded Process require a specific version of the Netezza client and server environment. For more information, see the SAS Foundation system requirements documentation for your operating environment.

Overview of the In-Database Deployment Package for Netezza

This section describes how to install and configure the in-database deployment package for Netezza (SAS Formats Library for Netezza and SAS Embedded Process).

The in-database deployment package for Netezza must be installed and configured before you can perform the following tasks:
• Use the %INDNZ_PUBLISH_FORMATS format publishing macro to create or publish the SAS_PUT( ) function and to create or publish user-defined formats as format functions inside the database.

• Use the %INDNZ_PUBLISH_MODEL scoring publishing macro to create scoring model functions inside the database.

For more information about using the format and scoring publishing macros, see the SAS In-Database Products: User's Guide.

The in-database deployment package for Netezza contains the SAS formats library, two pre-compiled binaries for utility functions, and the SAS Embedded Process.

The SAS formats library is a run-time library that is installed on your Netezza server. This installation is made so that the SAS scoring model functions and the SAS_PUT( ) function can access the routines within the run-time library. The SAS formats library contains the formats that are supplied by SAS.

The %INDNZ_PUBLISH_JAZLIB macro registers the SAS formats library. The %INDNZ_PUBLISH_COMPILEUDF macro registers a utility function in the database. The utility function is then called by the format and scoring publishing macros. You must run these two macros before you run the format and scoring publishing macros.

The SAS Embedded Process is a SAS server process that runs within Netezza to read and write data. The SAS Embedded Process contains macros, run-time libraries, and other software that is installed on your Netezza server. These installations are done so that the SAS scoring files created in Netezza can access routines within the SAS Embedded Process run-time libraries.

---

Function Publishing Process in Netezza

To publish the SAS scoring model functions, the SAS_PUT( ) function, and format functions on Netezza systems, the format and scoring publishing macros perform the following tasks:

• Create and transfer the files, using the Netezza External Table interface, to the Netezza server.

  Using the Netezza External Table interface, the source files are loaded from the client to a database table through remote ODBC. The source files are then exported to files (external table objects) on the host. Before transfer, each source file is divided into 32K blocks and converted to hexadecimal values to avoid problems with special characters, such as line feed or quotation marks. After the files are exported to the host, the source files are converted back to text.

• Compile those source files into object files using a Netezza compiler.

• Link with the SAS formats library.

• Register those object files with the Netezza system.

Note: This process is valid only when using publishing formats and scoring functions. It is not applicable to the SAS Embedded Process. If you use the SAS Embedded Process, the scoring publishing macro creates the scoring files and uses the SAS/ACCESS Interface to Netezza to insert the scoring files into a model table.
Netezza Installation and Configuration

Netezza Installation and Configuration Steps

1. If you are upgrading from a previous version, follow the instructions in “Upgrading from a Previous Version” on page 129.

2. Install the in-database deployment package.
   
   For more information, see “Installing the SAS Formats Library, Binary Files, and the SAS Embedded Process” on page 130.

3. Run the %INDNZ_PUBLISH_JAZLIB macro to publish the SAS formats library as an object.

   For more information, see “Running the %INDNZ_PUBLISH_JAZLIB Macro” on page 134.

4. Run the %INDNZ_PUBLISH_COMPILEUDF macro.

   For more information, see “Running the %INDNZ_PUBLISH_COMPILEUDF Macro” on page 136.

5. If you plan to use SAS Model Manager with the SAS Scoring Accelerator for in-database scoring, perform the additional configuration tasks in Chapter 16, “Configuring SAS Model Manager,” on page 163.

Upgrading from a Previous Version

Overview of Upgrading from a Previous Version

You can upgrade from a previous version of the SAS Formats Library and binary files, the SAS Embedded Process, or both. See the following topics:

- If you want to upgrade from a previous version of the SAS Formats Library and binary files, see “Upgrading the SAS Formats Library and Binary Files” on page 129.

- If you want to upgrade from a previous version of the SAS Embedded Process, see “Upgrading the SAS Embedded Process” on page 130.

Upgrading the SAS Formats Library and Binary Files

To upgrade from a previous version of the SAS Formats Library and binary files, follow these steps.

Note: These steps apply if you want to upgrade only the SAS Formats Library and binary files. If you want to upgrade only the SAS Embedded Process, see “Upgrading the SAS Embedded Process” on page 130.

1. Run the %INDNZ_PUBLISH_JAZLIB macro with ACTION=DROP to remove the SAS formats library as an object.

   For more information, see “Running the %INDNZ_PUBLISH_JAZLIB Macro” on page 134.
2. Run the %INDNZ_PUBLISH_COMPILEUDF macro with ACTION=DROP to remove the SAS_COMPILEUDF, SAS_DIRECTORYUDF, and SAS_HEXTOTEXTUDF functions.

For more information, see “Running the %INDNZ_PUBLISH_COMPILEUDF Macro” on page 136.

3. Navigate to the /nz/extensions/SAS directory and delete the SASFormatsLibraryForNetezza directory.

   Note: Under the SAS directory, the installer for the SAS Formats Library and binary files and the SAS Embedded Process installer both create a directory under the SAS directory. These directories are named SASFormatsLibraryForNetezza and SASTKInDatabaseServerForNetezza, respectively. If you delete everything under the SAS directory, the SAS Embedded Process, the SAS Formats Library, and the binary files are removed. If you want to remove only one, then you must leave the other directory.

4. If you are also upgrading the SAS Embedded Process, continue the installation instructions in “Upgrading the SAS Embedded Process” on page 130. Otherwise, continue the installation instructions in “Installing the SAS Formats Library, Binary Files, and the SAS Embedded Process” on page 130.

**Upgrading the SAS Embedded Process**

To upgrade the SAS Embedded Process, follow these steps.

   Note: These steps are for upgrading only the SAS Embedded Process. If you want to upgrade the SAS Formats Library and binary files, you must follow the steps in “Upgrading the SAS Formats Library and Binary Files” on page 129.

1. Check the current installed version of the SAS Embedded Process.

   nzcm --installed

2. Enter these commands to unregister and uninstall the SAS Embedded Process.

   nzcm -u SASTKInDatabaseServerForNetezza
   nzcm -e SASTKInDatabaseServerForNetezza

3. Navigate to the /nz/extensions/SASTKInDatabaseServerForNetezza directory and verify that the directory is empty.

   Note: Under the SAS directory, the installer for the SAS Formats Library and binary files and the SAS Embedded Process installer both create a directory under the SAS directory. These directories are named SASFormatsLibraryForNetezza and SASTKInDatabaseServerForNetezza, respectively. If you delete everything under the SAS directory, the SAS Embedded Process, the SAS Formats Library, and the binary files are removed. If you want to remove only one, then you must leave the other directory.


**Installing the SAS Formats Library, Binary Files, and the SAS Embedded Process**

**Moving and Installing the SAS Formats Library and Binary Files**

The SAS formats library and the binary files for the SAS_COMPILEUDF function are contained in a self-extracting archive file. The self-extracting archive file is located in
the YourSASDepot/SASFormatsLibraryforNetezza/3.1/
Netezza32bitTwinFin/ directory.

To move and unpack the self-extracting archive file, follow these steps:

1. Using a method of your choice, transfer the accelnetzfmt-3.1-n_lax.sh to a location on your Netezza server.

   \( n \) is a number that indicates the latest version of the file. If this is the initial installation, \( n \) has a value of 1. Each time you re-install or upgrade, \( n \) is incremented by 1.

2. After the accelnetzfmt-3.1-n_lax.sh file has been transferred to the Netezza server, log on as the user who owns the Netezza appliance (usually the “nz” ID).

3. Use the following commands at the UNIX prompt to unpack the self-extracting archive file.

   ```
   mkdir -p /nz/extensions
   chmod 755 /nz/extensions
   cd /nz/extensions
   chmod 755 path_to_sh_file/accelnetzfmt-3.1-n_lax.sh
   path_to_sh_file is the location to which you copied the self-extracting archive file in Step 1.
   ```

   After the script runs and the files are unpacked, the target directories should look similar to these.

   ```
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/3.1-n/bin/InstallAccelNetzFmt.sh
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/3.1-n/lib/SAS_CompileUDF_o_spu10
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/3.1-n/lib/SAS_CompileUDF_o_x86
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/3.1-n/lib/libjazxfbrs_spu10.so
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/3.1-n/lib/libjazxfbrs_x86.so
   ```

   There also is a symbolic link such that /nz/extensions/
SAS/SASFormatsLibraryForNetezza/3.1 points to the latest version.

---

**Moving and Installing the SAS Embedded Process**

The SAS Embedded Process is contained in a self-extracting archive file named sepcorenetz-13.00000-1.sh. This file is contained in a ZIP file that is put in your SAS Software Depot directory.

To unzip and unpack the self-extracting archive file and create a Netezza cartridge file, follow these steps:

1. Create a new temporary directory on your client machine such as /sasep. The new directory is referred to as EPZipDir throughout this section.

2. Navigate to the YourSASDepot/
   SAS_Core_Embedded_Process_Package_for_Netezza/13_0/
   Netezza_TwinFin_64bit_host directory. This directory was created when you created your SAS Software Depot.

3. Locate the en_sasexe.zip file. The en_sasexe.zip file is located in the
   YourSASDepot/SAS_Core_Embedded_Process_Package_for_Netezza/13_0/Netezza_TwinFin_64bit_host directory.

   The sepcorenetz-13.00000-1.sh file is included in this ZIP file.

4. Copy the en_sasexe.zip file to your EPZipDir temporary directory on the client machine.
5. Navigate to your \textit{EPZipDir} temporary directory and unzip \texttt{en_sasexe.zip}.

After the file is unzipped, a \texttt{sasexe} directory is created in the same location as the \texttt{en_sasexe.zip} file. The self-extracting archive file is in the \texttt{/EPZipDir/sasexe} directory.

6. Using a method of your choice, transfer the \texttt{sepcorenetz-13.00000-1.sh} file to any directory on the Netezza server. This directory is referred to as \texttt{path_to_sh_file}.

7. After the \texttt{sepcorenetz-13.00000-1.sh} file has been transferred to the Netezza server, log on as the user who owns the Netezza appliance (usually the “nz” ID).

8. If you have a database named SAS_EP, you should rename it.

When you unpack the self-extracting archive file, a SAS_EP database that contains the SAS Embedded Process cartridge file is created. The creation of the SAS_EP database overwrites any existing database that is named SAS_EP.


a. Change to the directory where you put the \texttt{sepcorenetz-13.00000-1.sh} file.

\begin{verbatim}
cd path_to_sh_file
\end{verbatim}

\texttt{path_to_sh_file} is the location to which you copied the self-extracting archive file in Step 6.

b. Change permissions on the file to enable you to execute the script.

\begin{verbatim}
chmod 755 -r sepcorenetz-13.00000-1.sh
\end{verbatim}

c. Use the following command at the UNIX prompt to unpack the self-extracting archive file:

\begin{verbatim}
./sepcorenetz-13.00000-1.sh
\end{verbatim}

After the script runs, the \texttt{sepcorenetz-13.00000-1.sh} file goes away, and the \texttt{SASTKInDatabaseServerForNetezza-13.0.0.0.nzc} Netezza cartridge file is created in its place.

10. Use the following \texttt{nzcm} commands to install and register the \texttt{sas_ep} cartridge:

\begin{verbatim}
nzcm -i sas_ep
nzcm -r sas_ep
\end{verbatim}

\textit{Note:} The \texttt{sas_ep} cartridge creates the NZRC database. The NZRC database contains remote controller functions that are required by the SAS Embedded Process. The \texttt{sas_ep} cartridge is available on the Netezza website. For access to the \texttt{sas_ep} cartridge, contact your local Netezza representative.

11. Use the following \texttt{nzcm} commands to install and register the SAS Embedded Process:

\begin{verbatim}
nzcm -i SASTKInDatabaseServerForNetezza-13.0.0.0.nzc
nzcm -r SASTKInDatabaseServerForNetezza
\end{verbatim}

\textit{Note:} The installation of the SAS Embedded Process is dependent on the \texttt{sas_ep} cartridge that is supplied by Netezza.

For more NZCM commands, see “NZCM Commands for the SAS Embedded Process” on page 133.
### NZCM Commands for the SAS Embedded Process

The following table lists and describes the NZCM commands that you can use with the SAS Embedded Process.

<table>
<thead>
<tr>
<th>Command</th>
<th>Action performed</th>
</tr>
</thead>
<tbody>
<tr>
<td>nzcm -help</td>
<td>Displays help for NZCM commands</td>
</tr>
<tr>
<td>nzcm --installed nzcm - i</td>
<td>Displays the filename (SASTKInDatabaseServerForNetezza) and the version number that is installed</td>
</tr>
<tr>
<td>nzcm --registered nzcm - r</td>
<td>Displays the filename (SASTKInDatabaseServerForNetezza) and the version number that is registered</td>
</tr>
<tr>
<td>nzcm --unregister SASTKInDatabaseServerForNetezza nzcm - u SASTKInDatabaseServerForNetezza</td>
<td>Unregisters the SAS Embedded Process</td>
</tr>
<tr>
<td>nzcm --unregister sas_ep nzcm - u sas_ep</td>
<td>Unregisters the sas_ep cartridge</td>
</tr>
<tr>
<td>Note: The sas_ep cartridge is installed only once. It does not need to be unregistered or uninstalled when the SAS Embedded Process is upgraded or re-installed. The sas_ep cartridge needs to be unregistered and uninstalled only when Netezza changes the cartridge version.</td>
<td></td>
</tr>
<tr>
<td>nzcm -uninstall SASTKInDatabaseServerForNetezza nzcm - e SASTKInDatabaseServerForNetezza</td>
<td>Uninstalls the SAS Embedded Process</td>
</tr>
<tr>
<td>nzcm --uninstall sas_ep nzcm - e sas_ep</td>
<td>Uninstalls the sas_ep cartridge</td>
</tr>
<tr>
<td>Note: The sas_ep cartridge is installed only once. It does not need to be unregistered or uninstalled when the SAS Embedded Process is upgraded or re-installed. The sas_ep cartridge needs to be unregistered and uninstalled only when Netezza changes the cartridge version.</td>
<td></td>
</tr>
<tr>
<td>nzcm --install SASTKInDatabaseServerForNetezza-13.0.0.0.nzc nzcm -i SASTKInDatabaseServerForNetezza-13.0.0.0.nzc</td>
<td>Installs the SAS Embedded Process</td>
</tr>
<tr>
<td>nzcm --install sas_ep nzcm -i sas_ep</td>
<td>Installs the sas_ep cartridge</td>
</tr>
<tr>
<td>nzcm --register SASTKInDatabaseServerForNetezza nzcm -r SASTKInDatabaseServerForNetezza</td>
<td>Registers the SAS Embedded Process</td>
</tr>
<tr>
<td>nzcm --register sas_ep nzcm -register sas_ep</td>
<td>Registers the sas_ep cartridge</td>
</tr>
</tbody>
</table>
Running the %INDNZ_PUBLISH_JAZLIB Macro

Overview of Publishing the SAS Formats Library
The SAS formats library is a shared library and must be published and registered as an object in the Netezza database. The library is linked to the scoring and format publishing macros through a DEPENDENCIES statement when the scoring model functions or formats are created.

You must run the %INDNZ_PUBLISH_JAZLIB macro to publish and register the SAS formats library. The %INDNZ_PUBLISH_JAZLIB macro publishes and registers the SAS formats library in the database as the sas_jazlib object.

%INDNZ_PUBLISH_JAZLIB Macro Run Process
To run the %INDNZ_PUBLISH_JAZLIB macro, follow these steps:

1. Start SAS and submit the following command in the Enhanced Editor or Program Editor:
   ```sas
   %let indconn=SERVER=yourservername USER=youruserid PW=yourpwd DB=database;
   ```
   For more information, see the “INDCONN Macro Variable” on page 134.

2. Run the %INDNZ_PUBLISH_JAZLIB macro. For more information, see “%INDNZ_PUBLISH_JAZLIB Macro Syntax” on page 135.

INDCONN Macro Variable
The INDCONN macro variable is used to provide credentials to connect to Netezza. You must specify server, user, password, and database information to access the machine on which you have installed the Netezza data warehouse. You must assign the INDCONN macro variable before the %INDNZ_PUBLISH_JAZLIB macro is invoked.

The value of the INDCONN macro variable for the %INDNZ_PUBLISH_JAZLIB macro has this format:

```
SERVER=<'server'><' USER=<'userid'><' PASSWORD=<'password'><' DATABASE=<'database'><' SCHEMA=<'schema-name'><'
```

- **SERVER=<'server'><'** specifies the server name or IP address of the server to which you want to connect. This server accesses the database that contains the tables and views that you want to access. If the server name contains spaces or nonalphanumeric characters, enclose the server name in quotation marks.

- **USER=<'userid'><'** specifies the Netezza user name (also called the user ID) that you use to connect to your database. If the user name contains spaces or nonalphanumeric characters, enclose the user name in quotation marks.

- **PASSWORD=<'password'><'** specifies the password that is associated with your Netezza user name. If the password contains spaces or nonalphanumeric characters, enclose the password in quotation marks.

Tip Use only PASSWORD=, PASS=, or PW= for the password argument. PWD= is not supported and causes an error.
**DATABASE=**<database> specifies the name of the database on the server that contains the tables and views that you want to access. If the database name contains spaces or nonalphanumeric characters, enclose the database name in quotation marks.

**Interaction** The database that is specified by the %INDNZ_PUBLISH_JAZLIB macro’s DATABASE= argument takes precedence over the database that you specify in the INDCONN macro variable. If you do not specify a value for DATABASE= in either the INDCONN macro variable or the %INDNZ_PUBLISH_JAZLIB macro, the default value of SASLIB is used. For more information, see “%INDNZ_PUBLISH_JAZLIB Macro Syntax” on page 135.

**Tip** The object name for the SAS formats library is **sas_jazlib**.

**SCHEMA=**<schema-name> specifies the name of the schema where the SAS formats library is published.

**Restriction** This argument is supported only on Netezza v7.0.3 or later.

**Interaction** The schema that is specified by the %INDNZ_PUBLISH_JAZLIB macro’s DBSCHEMA= argument takes precedence over the schema that you specify in the INDCONN macro variable. If you do not specify a schema in the DBSCHEMA= argument or the INDCONN macro variable, the default schema for the target database is used.

---

**%INDNZ_PUBLISH_JAZLIB Macro Syntax**

%INDNZ_PUBLISH_JAZLIB

(\<DATABASE=database> \\
<, DBSCHEMA=schema-name> \\
<, ACTION=CREATE | REPLACE | DROP> \\
<, OUTDIR=diagnostic-output-directory> \\
);

**Arguments**

**DATABASE=**<database>

specifies the name of a Netezza database to which the SAS formats library is published as the **sas_jazlib** object.

**Default** SASLIB

**Interaction** The database that is specified by the DATABASE= argument takes precedence over the database that you specify in the INDCONN macro variable.

**Tip** The object name for the SAS formats library is **sas_jazlib**.

**DBSCHEMA=**<schema-name>

specifies the name of a Netezza schema to which the SAS formats library is published.

**Restrictions** This argument is supported only on Netezza v7.0.3 or later.

This argument is supported only on Netezza v7.0.3 or later.
Interaction  The schema that is specified by the DBSCHEMA= argument takes precedence over the schema that you specify in the INDCONN macro variable. If you do not specify a schema in the DBSCHEMA= argument or the INDCONN macro variable, the default schema for the target database is used.

ACTION=CREATE | REPLACE | DROP
specifies that the macro performs one of the following actions:

CREATE  creates a new SAS formats library.

REPLACE  overwrites the current SAS formats library, if a SAS formats library by the same name is already registered, or creates a new SAS formats library if one is not registered.

DROP  causes the SAS formats library to be dropped from the Netezza database.

Default  CREATE

Tip  If the SAS formats library was published previously and you specify ACTION=CREATE, you receive warning messages that the library already exists. You are prompted to use REPLACE. If you specify ACTION=DROP and the SAS formats library does not exist, you receive an error message.

OUTDIR=diagnostic-output-directory
specifies a directory that contains diagnostic files.

Tip  Files that are produced include an event log that contains detailed information about the success or failure of the publishing process.

Running the %INDNZ_PUBLISH_COMPILEUDF Macro

Overview of the %INDNZ_PUBLISH_COMPILEUDF Macro

The %INDNZ_PUBLISH_COMPILEUDF macro creates three functions:

- SAS_COMPILEUDF. This function facilitates the scoring and format publishing macros. The SAS_COMPILEUDF function compiles the scoring model and format source files into object files. This compilation uses a Netezza compiler and occurs through the SQL interface.

- SAS_DIRECTORYUDF and SAS_HEXTOTEXTUDF. These functions are used when the scoring and format publishing macros transfer source files from the client to the host using the Netezza External Tables interface. SAS_DIRECTORYUDF creates and deletes temporary directories on the host. SAS_HEXTOTEXTUDF converts the files from hexadecimal back to text after the files are exported on the host. For more information about the file transfer process, see “Function Publishing Process in Netezza” on page 128.

You have to run the %INDNZ_PUBLISH_COMPILEUDF macro only one time.

The SAS_COMPILEUDF, SAS_DIRECTORYUDF, and SAS_HEXTOTEXTUDF functions must be published before the %INDNZ_PUBLISH_FORMATS or %INDNZ_PUBLISH_MODEL macros are run. Otherwise, these macros fail.
Note: To publish the SAS_COMPILEUDF, SAS_DIRECTORYUDF, and SAS_HEXTOTEXTUDF functions, you must have the appropriate Netezza user permissions to create these functions in either the SASLIB database (default) or in the database that is used in lieu of SASLIB. For more information, see “Netezza Permissions” on page 140.

%INDNZ_PUBLISH_COMPILEUDF Macro Run Process
To run the %INDNZ_PUBLISH_COMPILEUDF macro to publish the SAS_COMPILEUDF, SAS_DIRECTORYUDF, and SAS_HEXTOTEXTUDF functions, follow these steps:

1. Create either a SASLIB database or a database to be used in lieu of the SASLIB database.
   This database is where the SAS_COMPILEUDF, SAS_DIRECTORYUDF, and SAS_HEXTOTEXTUDF functions are published. You specify this database in the DATABASE argument of the %INDNZ_PUBLISH_COMPILEUDF macro. For more information about how to specify the database that is used in lieu of SASLIB, see “%INDNZ_PUBLISH_COMPILEUDF Macro Run Process” on page 137.

2. Start SAS and submit the following command in the Enhanced Editor or Program Editor.
   `%let indconn = server=yourserver user=youruserid password=yourpwd database=database;`
   For more information, see the “INDCONN Macro Variable” on page 137.

3. Run the %INDNZ_PUBLISH_COMPILEUDF macro. For more information, see “%INDNZ_PUBLISH_COMPILEUDF Macro Syntax” on page 138.

   After the SAS_COMPILEUDF function is published, the model or format publishing macros can be run to publish the scoring model or format functions.

INDCONN Macro Variable
The INDCONN macro variable provides the credentials to make a connection to Netezza. You must specify the server, user, password, and database information to access the machine on which you have installed the Netezza database. You must assign the INDCONN macro variable before the %INDNZ_PUBLISH_COMPILEUDF macro is invoked.

The value of the INDCONN macro variable for the %INDNZ_PUBLISH_COMPILEUDF macro has this format.

```
SERVER=<>server<> USER=<>userid<> PASSWORD=<>password<> DATABASE=SASLIB | <>database<> SCHEMA=<>schema-name<>
```

- **SERVER=<>server<>** specifies the server name or IP address of the server to which you want to connect. This server accesses the database that contains the tables and views that you want to access. If the server name contains spaces or nonalphanumeric characters, enclose the server name in quotation marks.

- **USER=<>userid<>** specifies the Netezza user name (also called the user ID) that you use to connect to your database. If the user name contains spaces or nonalphanumeric characters, enclose the user name in quotation marks.
PASSWORD=<'password'>
specifies the password that is associated with your Netezza user name. If the password contains spaces or nonalphanumeric characters, enclose the password in quotation marks.

Tip  Use only PASSWORD=, PASS=, or PW= for the password argument. PWD= is not supported and causes an error.

DATABASE=SASLIB | <'database'>
specifies the name of the database on the server that contains the tables and views that you want to access. If the database name contains spaces or nonalphanumeric characters, enclose the database name in quotation marks.

Default       SASLIB

Interactions The database that is specified by the %INDNZ_PUBLISH_COMPILEUDF macro’s DATABASE= argument takes precedence over the database that you specify in the INDCONN macro variable. If you do not specify a value for DATABASE= in either the INDCONN macro variable or the %INDNZ_PUBLISH_COMPILEUDF macro, the default value of SASLIB is used. For more information, see “%INDNZ_PUBLISH_COMPILEUDF Macro Syntax” on page 138.

If the SAS_COMPILEUDF function is published in a database other than SASLIB, then that database name should be used instead of SASLIB for the DBCOMPILE argument in the %INDNZ_PUBLISH_FORMATS and %INDNZ_PUBLISH_MODEL macros. Otherwise, the %INDNZ_PUBLISH_FORMATS and %INDNZ_PUBLISH_MODEL macros fail when calling the SAS_COMPILEUDF function during the publishing process. If a database name is not specified, the default is SASLIB. For documentation on the %INDNZ_PUBLISH_FORMATS and %INDNZ_PUBLISH_MODEL macros, see “Documentation for Using In-Database Processing in Netezza” on page 141.

SCHEMA=<'schema-name'>
specifies the name of the schema where the SAS_COMPILEUDF, SAS_DIRECTORYUDF, and SAS_HEXTOTEXTUDF functions are published.

Restriction This argument is supported only on Netezza v7.0.3 or later.

Interaction The schema that is specified by the %INDNZ_PUBLISH_COMPILEUDF macro’s DBSCHEMA= argument takes precedence over the schema that you specify in the INDCONN macro variable. If you do not specify a schema in the DBSCHEMA= argument or the INDCONN macro variable, the default schema for the target database is used.

%INDNZ_PUBLISH_COMPILEUDF Macro Syntax
%INDNZ_PUBLISH_COMPILEUDF
(<DATABASE=database-name>
 <, DBSCHEMA=schema-name>
 <, ACTION=CREATE | REPLACE | DROP>
 <, OUTDIR=diagnostic-output-directory>
 );
Arguments

**DATABASE=**`database-name`
specifies the name of a Netezza database to which the SAS_COMPILEUDF is published.

Default: SASLIB

Interaction: The database that is specified by the DATABASE= argument takes precedence over the database that you specify in the INDCCONN macro variable. For more information, see “INDCONN Macro Variable” on page 137.

**DBSCHEMA=**`schema-name`
specifies the name of a Netezza schema to which the SAS_COMPILEUDF function is published.

Restriction: This argument is supported only on Netezza v7.0.3 or later.

Interaction: The schema that is specified by the DBSCHEMA= argument takes precedence over the schema that you specify in the INDCCONN macro variable. If you do not specify a schema in the DBSCHEMA= argument or the INDCCONN macro variable, the default schema for the target database is used.

**ACTION=CREATE | REPLACE | DROP**
specifies that the macro performs one of the following actions:

**CREATE**
creates a new SAS_COMPILEUDF function.

**REPLACE**
overwrites the current SAS_COMPILEUDF function, if a SAS_COMPILEUDF function by the same name is already registered, or creates a new SAS_COMPILEUDF function if one is not registered.

**DROP**
causes the SAS_COMPILEUDF function to be dropped from the Netezza database.

Default: CREATE

Tip: If the SAS_COMPILEUDF function was published previously and you specify ACTION=CREATE, you receive warning messages that the function already exists and be prompted to use REPLACE. If you specify ACTION=DROP and the SAS_COMPILEUDF function does not exist, you receive an error message.

**OUTDIR=**`diagnostic-output-directory`
specifies a directory that contains diagnostic files.

Tip: Files that are produced include an event log that contains detailed information about the success or failure of the publishing process.
Netezza Permissions

There are three sets of permissions involved with the in-database software.

- The first set of permissions is needed by the person who publishes the SAS formats library and the SAS_COMPILEUDF, SAS_DIRECTORYUDF, and SAS_HEXTOTEXTUDF functions. These permissions must be granted before the %INDNZ_PUBLISH_JAZLIB and %INDNZ_PUBLISH_COMPILEUDF macros are run. Without these permissions, running these macros fails.

The following table summarizes the permissions that are needed by the person who publishes the formats library and the functions.

<table>
<thead>
<tr>
<th>Permission Needed</th>
<th>Authority Required to Grant Permission</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATE LIBRARY permission to run the %INDNZ_PUBLISH_JAZLIB macro that publishes the SAS formats library (sas_jazlib object)</td>
<td>System Administrator or Database Administrator</td>
<td>GRANT CREATE LIBRARY TO fmtlibpublisherid</td>
</tr>
<tr>
<td>CREATE FUNCTION permission to run the %INDNZ_PUBLISH_COMPILEUDF macro that publishes the SAS_COMPILEUDF, SAS_DIRECTORYUDF, and the SAS_HEXTOTEXTUDF functions</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- The second set of permissions is needed by the person who runs the format publishing macro, %INDNZ_PUBLISH_FORMATS, or the scoring publishing macro, %INDNZ_PUBLISH_MODEL. The person who runs these macros is not necessarily the same person who runs the %INDNZ_PUBLISH_JAZLIB and %INDNZ_PUBLISH_COMPILEUDF macros. These permissions are most likely needed by the format publishing or scoring model developer. Without these permissions, the publishing of the scoring model functions and the SAS_PUT( ) function and formats fails.

Note: Permissions must be granted for every format and scoring model publisher and for each database that the format and scoring model publishing uses. Therefore, you might need to grant these permissions multiple times. After the Netezza permissions are set appropriately, the format and scoring publishing macros can be run.

Note: When permissions are granted to specific functions, the correct signature, including the sizes for numeric and string data types, must be specified.

The following table summarizes the permissions that are needed by the person who runs the format or scoring publishing macro.
<table>
<thead>
<tr>
<th>Permission Needed</th>
<th>Authority Required to Grant Permission</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXECUTE permission for the SAS Formats Library</td>
<td>System Administrator or Database Administrator</td>
<td>GRANT EXECUTE ON SAS_JAZLIB TO scoringorfmtpublisherid</td>
</tr>
<tr>
<td></td>
<td>Note: If you have SYSADM or DBADM authority, then you have these permissions. Otherwise, contact your database administrator to obtain these permissions.</td>
<td>GRANT EXECUTE ON SAS_COMPILEUDF TO scoringorfmtpublisherid</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GRANT EXECUTE ON SAS_DIRECTORYUDF TO scoringorfmtpublisherid</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GRANT EXECUTE ON SAS_HEXTOTEXTUDF TO scoringorfmtpublisherid</td>
</tr>
<tr>
<td>EXECUTE permission for the SAS_COMPILEUDF function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXECUTE permission for the SAS_DIRECTORYUDF function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXECUTE permission for the SAS_HEXTOTEXTUDF function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CREATE FUNCTION, CREATE TABLE, CREATE TEMP TABLE, and CREATE EXTERNAL TABLE permissions to run the format and scoring publishing macros</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• The third set of permissions is needed by the person who runs the SAS Embedded Process to create scoring files.</td>
<td>The SAS Embedded Process has a dependency on the IBM Netezza Analytics (INZA) utility. You must grant the user and database permissions using these commands.</td>
<td>Note: If you plan to use SAS Model Manager with the SAS Scoring Accelerator for in-database scoring, additional permissions are required. For more information, see Chapter 16, “Configuring SAS Model Manager,” on page 163.</td>
</tr>
</tbody>
</table>

Documentation for Using In-Database Processing in Netezza

For information about how to publish SAS formats, the SAS_PUT( ) function, and scoring models, see the SAS In-Database Products: User’s Guide, located at http://support.sas.com/documentation/onlinedoc/indbtech/index.html.
Chapter 13
Administrator’s Guide for Oracle

In-Database Deployment Package for Oracle

Prerequisites

SAS Foundation and the SAS/ACCESS Interface to Oracle must be installed before you install and configure the in-database deployment package for Oracle.

The SAS Scoring Accelerator for Oracle requires a specific version of the Oracle client and server environment. For more information, see the SAS Foundation system requirements documentation for your operating environment.

Overview of the In-Database Package for Oracle

This section describes how to install and configure the in-database deployment package for Oracle (SAS Embedded Process).

The in-database deployment package for Oracle must be installed and configured before you perform the following tasks:

- Use the %INDOR_PUBLISH_MODEL scoring publishing macro to create scoring files inside the database.
- Run SAS High-Performance Analytics when the analytics cluster is using a parallel connection with a remote Oracle Exadata appliance. The SAS Embedded Process, which resides on the data appliance, is used to provide high-speed parallel data
transfer between the data appliance and the analytics environment where it is processed.

For more information, see the SAS High-Performance Analytics Infrastructure: Installation and Configuration Guide.

For more information about using the scoring publishing macros, see the SAS In-Database Products: User’s Guide.

The in-database deployment package for Oracle includes the SAS Embedded Process. The SAS Embedded Process is a SAS server process that runs within Oracle to read and write data. The SAS Embedded Process contains macros, run-time libraries, and other software that are installed on your Oracle system. The software is installed so that the SAS scoring files created in Oracle can access the routines within the SAS Embedded Process’s run-time libraries.

Oracle Installation and Configuration

Installing and Configuring Oracle

To install and configure Oracle, follow these steps:

1. If you are upgrading from a previous version, follow the instructions in “Upgrading from a Previous Version” on page 144 before installing the in-database deployment package.

2. Install the in-database deployment package.

   For more information, see “Installing the In-Database Deployment Package for Oracle” on page 145.

3. Create the required users and objects in the Oracle server.

   For more information, see “Creating Users and Objects for the SAS Embedded Process” on page 146.

4. If you plan to use SAS Model Manager with the SAS Scoring Accelerator for in-database scoring, perform the additional configuration tasks provided in Chapter 16, “Configuring SAS Model Manager,” on page 163.

   Note: If you are installing the SAS High-Performance Analytics environment, there are additional steps to be performed after you install the SAS Embedded Process. For more information, see SAS High-Performance Analytics Infrastructure: Installation and Configuration Guide.

Upgrading from a Previous Version

You can upgrade from a previous version of the SAS Embedded Process. Before installing the In-Database Deployment Package for Oracle, have the database administrator (DBA) notify the user community that there will be an upgrade of the SAS Embedded Process. The DBA should then alter the availability of the database by restricting access, or by bringing the database down. Then, follow the steps outlined in “Installing the In-Database Deployment Package for Oracle” on page 145.
Installing the In-Database Deployment Package for Oracle

Unzipping the In-Database Deployment Package for Oracle
The in-database deployment package binary files for Oracle are contained in a self-extracting archive file named `sepcoreorcl-13.00000-1.sh`. This file is contained in a ZIP file that is put in your SAS Software Depot directory.

To unzip the in-database deployment package for Oracle, follow these steps:

1. Create a new temporary directory on your client machine such as `/sasep`. The new directory is referred to as `EPZipDir` throughout this section.
2. Navigate to the `YourSASDepot/standalone_installs` directory. This directory was created when you created your SAS Software Depot.
3. Locate the `en_sasexe.zip` file. The `en_sasexe.zip` file is located in the `YourSASDepot/standalone_installs/SAS_Core_Embedded_Process_Package_for_Oracle/13_0/Oracle_on_Linux/` directory.
   The `sepcoreorcl-13.00000-1.sh` file is included in this ZIP file.
4. Copy the `en_sasexe.zip` file to your `EPZipDir` temporary directory on the client machine.
   ```
cp en_sasexe.zip /EPZipDir
```
5. Navigate to your `EPZipDir` temporary directory and unzip `en_sasexe.zip`.
   After the file is unzipped, a `sasexe` directory is created in the same location as the `en_sasexe.zip` file. The `sepcoreorcl-13.00000-1.sh` file is in the `/EPZipDir/sasexe` directory.

Move the SAS Embedded Process Package to the Oracle Server
To move and copy the Oracle in-database deployment package, follow these steps:

1. Using a method of your choice (for example, PSFTP, SFTP, SCP, or FTP), move the `sepcoreorcl-13.00000-1.sh` file to a directory of your choice. It is recommended that you create a SAS directory under your home directory. An example is `/u01/pohome/SAS`. This directory is referred to as `path_to_sh_file`.
2. Copy the `sepcoreorcl-13.00000-1.sh` file onto each of the RAC nodes using a method of your choice (for example, DCLI, SFTP, SCP, or FTP).
   
   Note: This might not be necessary. For RAC environments with a shared Oracle Home, you can also use one of these methods:
   
   - Copy the extracted directories from a single node.
   - Copy the self-extracting archive file to a directory common to all the nodes.
   
   Note: As a best practice, you should always deploy from an empty directory.
   - If the file system is not a database file system (DBFS), extract the file in one location for the whole appliance.

Unpack the SAS Embedded Process Files
For each node, log on as the owner user for the Oracle software using a secured shell, such as SSH. Follow these steps:

1. Change to the directory where the `sepcoreorcl-13.00000-1.sh` file is located.
2. Change permissions on the file to enable you to execute the script.
   
   `chmod 755 -r sepcoreorcl-13.00000-1.sh`

3. Use this command to unpack the self-extracting archive file.
   
   `./sepcoreorcl-13.00000-1.sh`

   After this script is run and the files are unpacked, a SAS tree is built in the current directory. The content of the target directories should be similar to the following, depending on the path to your self-extracting archive file. Part of the directory path is shaded to emphasize the different target directories that are used.

   ```
   /path_to_sh_file/SASEPHome/admin
   /path_to_sh_file/SASEPHome/bin
   /path_to_sh_file/SASEPHome/logs
   /path_to_sh_file/SASEPHome/misc
   /path_to_sh_file/SASEPHome/sasexe
   /path_to_sh_file/SASEPHome/utilities
   ```

4. On non-shared Oracle home systems, update the contents of the $ORACLE_HOME/hs/admin/extproc.ora file on each node. On shared Oracle home systems, you can update the file in one location that is accessible by all nodes.

   **Note:** Ask your DBA if the ORACLE_HOME environment variable is not set.

   a. Make a backup of the current extproc.ora file.
   b. Add the following settings to the file making sure to override any previous settings.
      
      ```
      SET EXTPROC_DLLS=ANY
      SET EPPATH=/path_to_sh_file/SASEPHome/
      SET TKPATH=/path_to_sh_file/SASEPHome/sasexe
      ```

5. On non-shared Oracle home systems, update the contents of the $ORACLE_HOME/network/admin/sqlnet.ora file on each node.

   a. Make a backup of the current sqlnet.ora file. If the file does not exist, create one.
   b. Add the following setting to the file.
      
      ```
      DIAG_ADR_ENABLED=OFF
      ```

---

**Creating Users and Objects for the SAS Embedded Process**

After the In-Database Deployment Package for Oracle is installed, the DBA must create the users and grant user privileges. The DBA needs to perform these tasks before the SAS administrator can create the objects for the Oracle server. The users and objects are required for the SAS Embedded Process to work.

**Note:** SQLPLUS or an equivalent SQL tool can be used to submit the SQL statements in this topic.

1. Create a SASADMIN user.

   To create the user accounts for Oracle, the DBA must perform the following steps:

   a. Change the directory to `/path_to_sh_file/SASEPHome/admin`.
   b. Connect as SYS, using the following command:
      
      ```
      sqlplus sys/<password> as sysdba
      ```
   c. Create and grant user privileges for the SASADMIN user.
Here is an example of how to create a SASADMIN user.

CREATE USER SASADMIN IDENTIFIED BY <password>
    DEFAULT TABLESPACE <tablespace-name>
    TEMPORARY TABLESPACE <tablespace-name>;
    GRANT UNLIMITED TABLESPACE TO SASADMIN;

d. Submit the following SQL script to grant the required privileges to the SASADMIN user.

SQL>@sasadmin_grant_privs.sql

e. Log off from the SQLPLUS session using “Quit” or close your SQL tool.

2. Create the necessary database objects.

To create the objects and the SASEPunzipFUNC table function that are needed to run the scoring model, the SAS administrator (SASADMIN) must perform the following steps:

a. Change the current directory to /path_to_sh_file/SASEPHome/admin (if you are not already there).

b. Connect as SASADMIN, using the following command:

sqlplus sasadmin/<password>

c. Submit the following SQL statement:

@create_sasepfunc.sql;

Note: You can ignore the following errors:

ORA-00942: table or view does not exist
ORA-01432: public synonym to be dropped does not exist

---

Oracle Permissions

The person who runs the %INDOR_CREATE_MODELTABLE needs CREATE permission to create the model table. Here is an example.

GRANT CREATE TABLE TO userid

The person who runs the %INDOR_PUBLISH_MODEL macro needs INSERT permission to load data into the model table. This permission must be granted after the model table is created. Here is an example.

GRANT INSERT ON modeltablename TO userid

Note: The RESOURCE user privilege that was granted in the previous topic includes the permissions for CREATE, DELETE, DROP, and INSERT.

If you plan to use SAS Model Manager with the SAS Scoring Accelerator for in-database scoring, additional permissions are required. For more information, see Chapter 16, “Configuring SAS Model Manager,” on page 163.
Documentation for Using In-Database Processing in Oracle

Chapter 14
Administrator’s Guide for SAP HANA

In-Database Deployment Package for SAP HANA

Prerequisites
SAS Foundation and the SAS/ACCESS Interface to SAP HANA must be installed before you install and configure the in-database deployment package for SAP HANA.

The SAS Scoring Accelerator for SAP HANA and the SAS Embedded Process require a specific version of the SAP HANA client and server environment. For more information, see the SAS Foundation system requirements documentation for your operating environment.

Overview of the In-Database Deployment Package for SAP HANA
This section describes how to install and configure the in-database deployment package for SAP HANA (SAS Embedded Process).
The in-database deployment package for SAP HANA must be installed and configured before you can perform the following tasks:

- Use the `%INDHN_PUBLISH_MODEL` scoring publishing macro to insert a model into the model table in the SAP HANA database.
- Run SAS High-Performance Analytics when the analytics cluster is using a parallel connection with a remote SAP HANA appliance. The SAS Embedded Process, which resides on the data appliance, is used to provide high-speed parallel data transfer between the data appliance and the analytics environment where it is processed.

For more information, see the *SAS High-Performance Analytics Infrastructure: Installation and Configuration Guide*.

For more information about using the scoring publishing macros, see the *SAS In-Database Products: User's Guide*.

The SAS Embedded Process is a SAS server process that runs within SAP HANA to read and write data. The SAS Embedded Process contains macros, run-time libraries, and other software that is installed on your SAP HANA system. These installations are done so that the SAS scoring files created in SAP HANA can access routines within the SAS Embedded Process run-time libraries.

---

**SAP HANA Installation and Configuration**

**SAP HANA Installation and Configuration Steps**

To install and configure SAP HANA, follow these steps:

1. Review the permissions required for installation.
   
   For more information, see “SAP HANA Permissions” on page 157.

2. Review the number of semaphore arrays configured for the SAP HANA server.
   
   It is recommended that the SAP HANA server that runs the SAS Embedded Process be configured with a minimum of 1024 to 2048 semaphore arrays. For more information, see “Semaphore Requirements When Using the SAS Embedded Process for SAP HANA” on page 156.

3. Enable the SAP HANA Script Server process as SYSTEM in the SAP HANA Studio.
   
   The SAP HANA script server process must be enabled to run in the HANA instance. The script server process can be started while the SAP HANA database is already running.

   To start the Script Server, follow these steps:

   a. Open the **Configuration** tab page in the SAP HANA Studio.
   
   b. Expand the daemon.ini configuration file.
   
   c. Expand the **scriptserv** section.
   
   d. Change the **instances** parameter from 0 to 1 at the system level.

   A value of 1 indicates you have enabled the server.

   **Note:** For more information, see SAP Note 1650957.
4. If you are upgrading from a previous version, follow the instructions in “Upgrading from a Previous Version” on page 151 before installing the in-database deployment package.

5. Install the SAS Embedded Process.
   For more information, see “Installing the In-Database Deployment Package for SAP HANA” on page 152.

6. Install the SASLINK Application Function Library.
   For more information, see “Installing the SASLINK AFL Plugins on the Appliance (SPS11)” on page 154.

   For more information, see “Importing the SAS_EP Stored Procedure” on page 155.

8. Review the requirement for Auxiliary Wrapper Procedures.
   For more information, see “Auxiliary Wrapper Procedures (SPS11)” on page 155.

   a. Log on to the SAP HANA server as the database administrator or change the user to the database administrator.
      
      You can use one of these commands:
      
      ```
      su - SIDadm
      ssh SIDadm@server-name
      ```
   
   b. Run the StartupSASEP.sh script.
      
      ```
      ./StartupSASEP.sh
      ```

10. If you plan to use SAS Model Manager with the SAS Scoring Accelerator for in-database scoring, perform the additional configuration tasks provided in Chapter 16, “Configuring SAS Model Manager,” on page 163.

**Note:** If you are installing the SAS High-Performance Analytics environment, you must perform additional steps after you install the SAS Embedded Process. For more information, see *SAS High-Performance Analytics Infrastructure: Installation and Configuration Guide*.

---

**Upgrading from a Previous Version**

To upgrade from a previous version, follow these steps.

1. Log on to the SAP HANA system as root.
   
   You can use `su` or `sudo` to become the root authority.

2. Navigate to the directory that contains the UninstallSASEPFiles.sh script.
   
   ```
   cd /EPInstallDir/SASEPHome/bin
   ```
   
   **Note:** You can find the location of `EPInstallDir` by using the following command:
   
   ```
   ls -l /usr/local/bin/tkhnmain
   ```

3. Run the UninstallSASEPFiles.sh file.
   
   ```
   ./UninstallSASEPFiles.sh
   ```
   
   This script stops the SAS Embedded Process on the server.
For versions prior to SAS 9.4M4 (November 2016), the script deletes
the /SAS/SASTKInDatabaseServerForSAP_HANA directory and all its contents.
For versions starting with SAS 9.4M4 (November 2016), the script deletes
the /SASEPHome/bin directory and all its contents.

4. Re-install the SAS Embedded Process.

For more information, see “Installing the In-Database Deployment Package for SAP
HANA” on page 152.

Installing the In-Database Deployment Package for SAP HANA

Unzipping the In-Database Deployment Package for SAP HANA

The in-database deployment package for SAP HANA is contained in a self-extracting
archive file named sepcorehana-13.00000-1.sh. This file is contained in a ZIP file that is
put in your SAS Software Depot directory.

To unzip the in-database deployment package for SAP HANA, follow these steps:

1. Create a new temporary directory on your client machine such as /sasep. The new
directory is referred to as EPZipDir throughout this section.

2. Navigate to the YourSASDepot/standalone_installs directory. This
directory was created when you created your SAS Software Depot.

3. Locate the en_sasexe.zip file. The en_sasexe.zip file is located in the
YourSASDepot/standalone_installs/
SAS_Core_Embedded_Process_Package_for_SAP_HANA/13_0/
SAP_HANA_on_Linux_x64 directory.

The sepcorehana-13.00000-1.sh file is included in this ZIP file.

4. Copy the en_sasexe.zip file to your EPZipDir temporary directory on the client
machine.

cp en_sasexe.zip /EPZipDir

5. Navigate to your EPZipDir temporary directory and unzip en_sasexe.zip.

After the file is unzipped, a sasexe directory is created in the same location as the
en_sasexe.zip file. The sepcorehana-13.00000-1.sh file is in
the /EPZipDir/sasexe directory.

Installing the In-Database Deployment Package for SAP HANA

The in-database deployment package for SAP HANA is contained in a self-extracting
archive file named sepcorehana-1300000-1.sh.

To install the self-extracting archive file, follow these steps:

1. Create a directory on the SAP HANA appliance. This directory is referred to as
EPInstallDir throughout this section.

2. Using a method of your choice, transfer the sepcorehana-13.00000-1.sh file to the
EPInstallDir on the SAP HANA appliance.

This example uses secure copy, and EPInstallDir is the location where you want to
install the SAS Embedded Process.

scp sepcorehana-13.00000-1.sh username@hana:/EPInstallDir
Note: The sepcorehana-13.00000-1.sh file and the EPInstallDir directory require Read and Execute permissions for the database administrator.

3. After the sepcorehana-13.00000-1.sh file has been transferred, log on to the SAP HANA server as the “owner” of the SAS Embedded Process installation directory.

```bash
ssh sas-owner@server-name
```

4. Navigate to the directory where the self-extracting archive file was downloaded in Step 1.

```bash
cd /EPInstallDir
```

5. Change permissions on the file to enable you to execute the script.

```bash
chmod 755 -r sepcorehana-13.00000-1.sh
```

6. Use the following command at the UNIX prompt to unpack the self-extracting archive file:

```bash
./sepcorehana-13.00000-1.sh
```

After the script runs, a SASEPHome directory is created where the EP files are installed. The contents of the /EPInstallDir/SASEPHome directory should look similar to this. Directories and files of interest are shaded.

```
/EPInstallDir/SASEPHome/bin
/EPInstallDir/SASEPHome/bin/InstallsASEPFiles.sh
/EPInstallDir/SASEPHome/bin/ShowSASEPStatus.sh
/EPInstallDir/SASEPHome/bin/ShutdownSASEP.sh
/EPInstallDir/SASEPHome/bin/StartupSASEP.sh
/EPInstallDir/SASEPHome/bin/UninstallSASEPFiles.sh
/EPInstallDir/SASEPHome/misc
/EPInstallDir/SASEPHome/misc/sas_saslink_installer.tgz
/EPInstallDir/SASEPHome/sasexe
/EPInstallDir/SASEPHome/utilities
```

The InstallsASEPFiles.sh file installs the SAS Embedded Process. The next step explains how to run this file.

The UninstallSASEPFiles.sh file uninstalls the SAS Embedded Process. The ShowSASEPStatus.sh file shows the status of the SAS Embedded Process on each instance. The StartupSASEP.sh and ShutdownSASEP.sh files enable you to manually start and stop the SAS Embedded Process. For more information about running these two files, see “Controlling the SAS Embedded Process” on page 156.

7. Navigate to the directory where the SAS Embedded Process script files were installed.

```bash
cd /EPInstallDir/SASEPHome/bin
```

8. Use the following command at the UNIX prompt to install the SAS Embedded Process.

```bash
./InstallsASEPFiles.sh
```

Note: To execute this script you need root authority. Either use the su command to become the root or use the sudo command to execute this script to install the SAS Embedded Process.

Note: -verbose is on by default and enables you to see all messages that are generated during the installation process. Specify -quiet to suppress messages.
Installing the SASLINK AFL Plugins on the Appliance (SPS11)

The SASLINK Application Function Library (AFL) files are included in an installer that is packaged as a tarball (TAR file) and that is provided when the SAS Embedded Process self-extracting archive file is unpacked.

Note: The SID referenced in these instructions is the SAP HANA system identifier (for example, HDB).

To install the SASLINK AFL plugins on the appliance (HANA SPS11), follow these steps:

1. Log on to the SAP HANA server as the database administrator or change the user to the database administrator.
   
   You can use one of these commands.
   
   su - SIDadm
   
   ssh SIDadm@$server-name

2. If the SAS Embedded Process is running, run the ShutdownSASEP.sh script to stop the process.
   
   ./ShutdownSASEP.sh

   Alternatively, you can shutdown the SAS Embedded Process by removing its PID file.
   
   rm /var/tmp/tkhnmain.pid

3. Stop the SAP HANA database if it is running.
   
   HDB stop

4. Use one of these commands to change the user to the root authority.
   
   su - root
   
   sudo su -

5. Copy the TAR file to the /tmp directory.
   
   cp /EPInstallDir/SASEPHome/misc/sas_saslink_installer.tgz /tmp

6. Unpack the TAR file.
   
   cd /tmp
   
   tar -xvzf sas_saslink_installer.tgz

7. Run the HANA install utility from the directory where the TAR file was unpacked. Specify the system ID of the HANA instance when prompted by the install utility.
   
   cd /tmp/sas_saslink_installer/installer
   
   ./hdbinst

8. Use one of these commands to change the user back to the database administrator or change the user to the database administrator.
   
   su - SIDadm
   
   exit

9. Restart the SAP HANA database.
Importing the SAS_EP Stored Procedure

The SAS_EP Stored Procedure is used by the %INDHN_RUN_MODEL macro to run the scoring model.


Note: You can find the location of EPInstallDir by using the following command:

```
ls -l /usr/local/bin/tkhnmain
```

To import the delivery unit into SAP HANA, follow these steps:

1. Navigate to the EPInstallDir/SASEPHome/misc directory.
2. Copy the SAS_EP_sas.com.tgz package to a client machine on which the SAP HANA Studio client is installed.
3. Import the delivery unit.

There are several methods of importing the .tgz file. Examples are SAP HANA Studio or the Lifecycle Manager. To import the delivery unit using SAP HANA Studio, follow these steps:

a. Ensure that you have a connection to the target SAP HANA back end from your local SAP HANA Studio.

b. Select File ⇒ Import.

c. Select SAP HANA Content ⇒ Delivery Unit and click Next.

d. Select the target system and click Next.

e. In the Import Through Delivery Unit window, select the Client check box and select the SAS_EP_sas.com.tgz file.

f. Select the Overwrite inactive versions and Activate object check boxes.

The list of objects is displayed under Object import simulation.

g. Click Finish to import the delivery unit.

Auxiliary Wrapper Procedures (SPS11)

Operation of the SASLINK AFL and the SAS Embedded Process requires wrapper procedures that are already installed in the SAP HANA catalog on the server. There is no need to manually install these procedures.

However, an additional permission, AFLPM_CREATOR_ERASER_EXECUTE, is required. For more information, see “SAP HANA Permissions” on page 157.
Controlling the SAS Embedded Process

The SAS Embedded Process starts when you run the StartupSASEP.sh script. It continues to run until it is manually stopped or the database is shut down.

Note: Starting and stopping the SAS Embedded Process has implications for all scoring model publishers.

Note: Manually starting and stopping the SAS Embedded Process requires HANA database administrator user permissions.

When the SAS Embedded Process is installed, the ShutdownSASEP.sh and StartupSASEP.sh scripts are installed in the following directory. For more information about these files, see “Installing the In-Database Deployment Package for SAP HANA” on page 152.

/EPInstallDir/SASEPHome/bin

Note: You can find the location of EPInstallDir by using the following command:

`ls -l /usr/local/bin/tkhnmain`

Use the following command to start the SAS Embedded Process:

`. /StartupSASEP.sh`

Note: The -verbose option is on by default and provides a status of the start-up operations as they occur. You can specify the -quiet option to suppress messages.

ShutdownSASEP.sh shuts down the SAS Embedded Process. It is designed to be used to shut down the SAS Embedded Process prior to a database upgrade or re-install. This script should not be used as part of the normal operation.

Use the following command to shut down the SAS Embedded Process.

`. /ShutdownSASEP.sh`

Note: The -verbose option is on by default and provides a status of the shutdown operations as they occur. You can specify the -quiet option to suppress messages.

Semaphore Requirements When Using the SAS Embedded Process for SAP HANA

Each time a query using the SAS_EP stored procedure is invoked to execute a score, it requests a set of semaphore arrays (sometimes referred to as semaphore "sets") from the operating system. The SAS Embedded Process releases the semaphore arrays back to the operating system after scoring is complete.

The SAP HANA server that runs the SAS Embedded Process should be configured with a minimum of 1024 to 2048 semaphore arrays.

Note: The semaphore limit on the “maximum number of arrays” is distinct from the semaphore limit on the “maximum number of semaphores system wide”. The Linux `ipcs -s1` command shows the typical default semaphore-related limits set on SAP HANA:

```
------- Semaphore Limits -------
```
SAP HANA Permissions

The following permissions are needed by the person who installs the in-database deployment package:

**Note:** Some of the permissions listed below cannot be granted until the Auxiliary Wrapper Generator and Eraser Procedures are installed. For more information, see “Auxiliary Wrapper Procedures (SPS11)” on page 155.

<table>
<thead>
<tr>
<th>Task</th>
<th>Permission Needed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unpack the self-extracting archive file</td>
<td>owner of the SAS Embedded Process install directory. The SAS Embedded Process install directory must have permissions that allow Read and Execute permission by the database administrator user.</td>
</tr>
<tr>
<td>Install or uninstall the SAS Embedded Process (run InstallSASEPFiles.sh or UninstallSASEPFiles.sh script)</td>
<td>root authority</td>
</tr>
<tr>
<td>Import the SAS_EP procedure package</td>
<td>A user on the SAP HANA server that has at least the CONTENT_ADMIN role or its equivalent</td>
</tr>
<tr>
<td>Install AFL plugins (requires starting and stopping the database)</td>
<td>root authority and database administrator</td>
</tr>
</tbody>
</table>

The following permissions are needed by the person who runs the scoring models. Without these permissions, the publishing of the scoring models fails:

**SAP HANA SPS11:**

- AFLPM_CREATOR_ERASER_EXECUTE to `userid | role`;
- EXECUTE, SELECT, INSERT, UPDATE, and DELETE on the schema that is used for scoring

In addition, the roles of `sas.ep::User` and `AFL_SYS_AFL_SASLINK_AREA_EXECUTE` must be assigned to any user who wants to perform in-database processing. The `sas.ep::User` role is created when you import the SAS_EP stored procedure. The `AFL_SYS_AFL_SASLINK_AREA_EXECUTE` role is created when the AFL wrapper generator is created.

**Note:** If you plan to use SAS Model Manager with the SAS Scoring Accelerator for in-database scoring, additional permissions are required. For more information, see Chapter 16, “Configuring SAS Model Manager,” on page 163.
Documentation for Using In-Database Processing in SAP HANA

Installation and Configuration Requirements for the SAS Scoring Accelerator for SPD Server

Prerequisites

The SAS Scoring Accelerator for SPD Server requires SAS Scalable Performance Data Server 5.1 and SAS 9.4.

If you have a model that was produced by SAS Enterprise Miner, an active SPD Server, and a license for the SAS Scoring Accelerator for SPD Server, you have everything that you need to run scoring models in the SPD Server. Installation of an in-database deployment package is not required.

SPD Server Permissions

You must have permissions for the domains that you specify in the INDCONN and INDDATA macro variables when you execute the publish and run macros.

You also need regular Read, Write, and Alter permissions when writing files to the OUTDIR directory in the %INDSP_RUN_MODEL macro.

Where to Go from Here

For more information about using the SAS Scoring Accelerator for SPD Server, see the SAS In-Database Products: User's Guide.
Part 5

Configurations for SAS Model Manager

Chapter 16

Configuring SAS Model Manager ............................ 163
Chapter 16
Configuring SAS Model Manager

Preparing a Data Management System for Use with SAS Model Manager . . . 163
Prerequisites ................................................................. 163
Overview of Preparing a Data Management System for Use
with SAS Model Manager ........................................... 164
Configuring a Database .................................................. 164
SAS Embedded Process Publish Method ................................ 164
Scoring Function Publish Method ..................................... 165
Finding the JDBC JAR Files ............................................. 166
Configuring a Hadoop Distributed File System ....................... 167

Preparing a Data Management System for Use
with SAS Model Manager

Prerequisites

SAS Foundation, SAS/ACCESS, and the in-database deployment package for the
database must be installed and configured before you can prepare a data management
system (database or file system) for use with SAS Model Manager. For more
information, see the chapter for your type of database or file system in this guide. Here
are the databases and file systems that can be used with SAS Model Manager:

- DB2
- Greenplum
- Hadoop
- Netezza
- Oracle
- SAP HANA
- Teradata
Overview of Preparing a Data Management System for Use with SAS Model Manager

Additional configuration steps are required to prepare a data management system (database or file system) for publishing and scoring in SAS Model Manager if you plan to use the scoring function (MECHANISM=STATIC) publish method or the SAS Embedded Process (MECHANISM=EP) publish method. If you want to store the scoring function metadata tables in the database, then the SAS Model Manager In-Database Scoring Scripts product must be installed before the database administrator (DBA) can prepare a database for use with SAS Model Manager.

During the installation and configuration of SAS 9.4 products, the SAS Model Manager In-Database Scoring Scripts product is installed on the middle-tier server or another server tier if it is included in the custom plan file.

The location of the SAS installation directory is specified by the user. Here is the default installation location for the SAS Model Manager In-Database Scoring Scripts product on a Microsoft Windows server: C:\Program Files\SASHome\SASModelManagerInDatabaseScoringScripts

The script installation directory includes a directory that specifies the version of SAS Model Manager (currently 14.2). The files and subdirectories that are needed to prepare a database for use by SAS Model Manager are located in the version directory. The Utilities subdirectory contains two SQL scripts for each type of database: a Create Tables script and a Drop Tables script. The DBA needs these SQL scripts to create the tables needed by the SAS Model Manager to publish scoring functions.

**Note:** The database tables store SAS Model Manager metadata about scoring functions.

If your system is configured for Kerberos authentication for Hadoop or Teradata, each user must have a valid Kerberos ticket. You must also complete post-installation configuration steps to enable users to publish models from the SAS Model Manager web application. For more information, see [SAS Model Manager: Administrator’s Guide](#).

Configuring a Database

**SAS Embedded Process Publish Method**

To enable users to publish scoring model files to a database from SAS Model Manager using the SAS Embedded Process, follow these steps:

1. Create a separate database where the tables can be stored.
2. Set the user access permissions for the database.

For information about the permissions that are required for specific databases, see the following topics:

- “DB2 Permissions”
- “Greenplum Permissions”
- “Netezza Permissions”
- “Oracle Permissions”
- “SAP HANA Permissions”
3. SAS Model Manager requires that the following database and table permissions be set at minimum. Verify that the permissions that you have set on your database include these permissions.

a. GRANT CREATE, DROP, EXECUTE, and ALTER permissions for functions and procedures.

b. GRANT CREATE and DROP permissions for tables. With these permissions, users can validate the scoring results when publishing scoring model files using SAS Model Manager.

c. Run the database-specific macro to create a table in the database to store the published model scoring files. The value of the MODELTABLE= argument in the macro should match the specification of the In-Database Options for SAS Model Manager in SAS Management Console. For more information, see In-Database Options.

   If the Use model manager table option is set to No, then the model-table-name should be `sas_model_table`. Otherwise, it should be `sas_mdlmgr_ep`.

   Here is an example of the create model table macro for Teradata:

   ```
   %INDTD_CREATE_MODELTABLE(DATABASE=database-name, MODELTABLE=model-table-name, ACTION=CREATE);
   ```

   For more information about creating a table for a specific database, see the SAS In-Database Products: User's Guide.

---

**Scoring Function Publish Method**

To enable users to publish scoring functions to a database from SAS Model Manager, follow these steps:

1. Create a separate database where the tables can be stored.

2. Set the user access permissions for the database.

   For information about the permissions that are required for specific databases, see the following topics:

   - “DB2 Permissions”
   - “Greenplum Permissions”
   - “Netezza Permissions”
   - “Teradata Permissions for Publishing Formats and Scoring Models”

3. SAS Model Manager requires that the following database and table permissions be set at minimum. Verify that the permissions that you have set on your database include these permissions.

   a. GRANT CREATE, DROP, EXECUTE, and ALTER permissions for functions and procedures.

   b. GRANT CREATE and DROP permissions for tables. With these permissions, users can validate the scoring results when publishing a scoring function using SAS Model Manager.

   c. GRANT SELECT, INSERT, UPDATE, and DELETE permissions for SAS Model Manager metadata tables.
d. GRANT SELECT permission for the following views to validate the scoring function names:

- syscat.functions for DB2
- pg_catalog.pg_proc for Greenplum
- dbc.functions for Teradata
- _v_function for Netezza

Note: If scoring input tables, scoring output tables, or views exist in another database, then the user needs appropriate permissions to access those tables or views.

4. Navigate to the `\sasinstall\SASModelManagerInDatabaseScoringScripts\14.2\Utilities` directory to find the Create Tables and Drop Tables scripts for your database. Then, follow these steps:

a. Verify the statements that are specified in the Create Tables script. Here are the names of the scripts for each type of database:

- DB2 SQL scripts: createTablesDB2.sql and dropTablesDB2.sql
- Greenplum SQL scripts: createTablesGreenplum.sql and dropTablesGreenplum.sql
- Netezza SQL scripts: createTablesNetezza.sql and dropTablesNetezza.sql
- Teradata SQL scripts: createTablesTD.sql and dropTablesTD.sql

b. Execute the Create Tables script for a specific type of database.

5. Download the JDBC driver JAR files and place them in the `\lib` directory on the web application server where the SAS Model Manager web application is deployed. The default directory paths for the SAS Web Application Server are the following:

   single server install and configuration
   `\sasconfigdir\Lev1\Web\WebAppServer\SASServer1_1\lib`
   
   multiple server install and configuration
   `\sasconfigdir\Lev1\Web\WebAppServer\SASServer11_1\lib`

   Note: You must have Write permission to place the JDBC driver JAR files in the `\lib` directory. Otherwise, you can have the server administrator download them for you.

   For more information, see “Finding the JDBC JAR Files” on page 166.

6. Restart the SAS servers on the web application server.

**Finding the JDBC JAR Files**

The DB2 JDBC JAR files are `db2jcc.jar` and `db2jcc_license_cu.jar`. The DB2 JDBC JAR files can be found on the server on which the database client was
installed. For example, the default location for Windows is `C:\Program Files\IBM\SQLLIB\java`.

The Greenplum database uses the standard PostgreSQL database drivers. The PostgreSQL JDBC JAR file can be found on the PostgreSQL – JDBC Driver site at https://jdbc.postgresql.org/download.html. An example of a JDBC driver name is `postgresql-9.2-1002.jdbc4.jar`.

The Netezza JDBC JAR file is `nzjdbc.jar`. The Netezza JDBC JAR file can be found on the server on which the database client was installed. For example, the default location for Windows is `C:\JDBC`.

The Teradata JDBC JAR files are `terajdbc4.jar` and `tdgssconfig.jar`. The Teradata JDBC JAR files can be found on the Teradata website at [http://www.teradata.com](http://www.teradata.com). Select Support ⇒ Downloads ⇒ Developer Downloads, and then click JDBC Driver in the table.

For more information about the database versions that are supported, see the SAS Foundation system requirements documentation for your operating environment.

---

**Configuring a Hadoop Distributed File System**

To enable users to publish scoring model files to a Hadoop Distributed File System (HDFS) from SAS Model Manager using the SAS Embedded Process, follow these steps:

1. Create an HDFS directory where the model files can be stored.

   *Note:* The path to this directory is used when a user publishes a model from the SAS Model Manager user interface to Hadoop.

2. Grant users Write access permission to the HDFS directory. For more information, see “Hadoop Permissions” on page 10.

3. Add this line of code to the autoexec_usermods.sas file that is located in the Windows directory `\SAS-configuration-directory\Lev\SASApp\WorkspaceServer\`:

   ```
   %let HADOOP_Auth = Kerberos or blank;
   ```

   **UNIX Specifics**
   
   The location of the autoexec_usermods.sas file for UNIX is `/SAS-configuration-directory/Levn/SASApp/WorkspaceServer/`.

   If your Hadoop server is configured with Kerberos, set the HADOOP_Auth variable to Kerberos. Otherwise, leave it blank.

4. (Optional) If you want users to be able to copy the publish code and execute it using Base SAS, then this line of code must be added to the sasv9.cfg file that is located in the Windows directory `\SASHome\SASFoundation\9.4\`:

   ```
   -AUTOEXEC '\SAS-configuration-directory\Lev\SASApp\WorkspaceServer\autoexec_usermods.sas'
   ```

   **UNIX Specifics**
   
   The location of the sasv9.cfg file for UNIX is `/SASHome/SASFoundation/9.4/`.

5. (Optional) If your Hadoop distribution is using Kerberos authentication, each user must have a valid Kerberos ticket to access SAS Model Manager. You must also complete additional post-installation configuration steps to enable users to publish
models to a Hadoop Distributed File System (HDFS) from SAS Model Manager. For more information, see SAS Model Manager: Administrator’s Guide.
Part 6

Upgrading from or Re-installing a Previous Version of the Hadoop In-Database Deployment Package

Appendix 1
Upgrading from or Re-installing a Previous Version If Using SAS Deployment Manager .................. 171

Appendix 2
Upgrading from or Re-installing a Previous Version If Using Manual Deployment ......................... 177
Appendix 1

Upgrading from or Re-installing a Previous Version If Using SAS Deployment Manager

Upgrading from SAS 9.3 .......................................................... 171
Upgrading from SAS 9.4 before the July 2015 Release of SAS 9.4 .......... 172
Upgrading from or Re-installing from the July 2015 Release of SAS 9.4 and Later ................................................................. 173
Overview .................................................................................. 173
Removing the SAS Embedded Process Parcel Using Cloudera Manager ...... 173
Removing the SAS Embedded Process Stack Using Ambari ..................... 175

Upgrading from SAS 9.3

To upgrade from SAS 9.3, follow these steps:

1. Stop the SAS Embedded Process.

   \texttt{EPInstallDir}/SAS/SASTKInDatabaseServerForHadoop/9.35/bin/sasep-stop.all.sh

   \texttt{EPInstallDir} is the master node where you installed the SAS Embedded Process.

2. Delete the SAS Embedded Process from all nodes.

   \texttt{EPInstallDir}/SAS/SASTKInDatabaseServerForHadoop/9.35/bin/sasep-delete.all.sh

3. Verify that the sas.hadoop.ep.distribution-name.jar files have been deleted.

   The JAR files are located at \texttt{HadoopHome}/lib.

   For Cloudera, the JAR files are typically located here:
   /opt/cloudera/parcels/CDH/lib/hadoop/lib

   For Hortonworks, the JAR files are typically located here:
   /usr/lib/hadoop/lib

4. Continue the installation process.

   For more information, see “Using the SAS Deployment Manager to Deploy the SAS Embedded Process Parcel or Stack to the Cluster” on page 16.
Upgrading from SAS 9.4 before the July 2015 Release of SAS 9.4

CAUTION:
If you are using SAS Data Loader for Hadoop, you should remove the Quality Knowledge Base (QKB) and the SAS Data Management Accelerator for Spark from the Hadoop nodes before removing the SAS Embedded Process.

Removing the SAS Embedded Process removes the scripts that are used to remove these products. For more information, see “Removing the QKB” and “Removing the SAS Data Management Accelerator for Spark” in the SAS Data Loader for Hadoop: Installation and Configuration Guide.

To upgrade from SAS 9.4 before the July 2015 release of SAS 9.4, follow these steps:

1. Stop the SAS Embedded Process.

   EPInstallDir/SAS/SASTKInDatabaseServerForHadoop/9.*/bin/sasep-servers.sh
   -stop -hostfile host-list-filename | -host <"host-list<">

   EPInstallDir is the master node where you installed the SAS Embedded Process.

   For more information, see the SAS In-Database Products: Administrator’s Guide that came with your release.

2. Remove the SAS Embedded Process from all nodes.

   EPInstallDir/SAS/SASTKInDatabaseForServerHadoop/9.*/bin/sasep-servers.sh
   -remove -hostfile host-list-filename | -host "host-list<"
   -mrhome dir

   Note: This step ensures that all old SAS Hadoop MapReduce JAR files are removed.

   For more information, see the SASEP-SERVERS.SH syntax section of the SAS In-Database Products: Administrator’s Guide that came with your release.

3. Verify that the sas.hadoop.ep.apache*.jar files have been deleted.

   The JAR files are located at HadoopHome/lib.

   For Cloudera, the JAR files are typically located here:

   /opt/cloudera/parcels/CDH/lib/hadoop/lib

   For Hortonworks, the JAR files are typically located here:

   /usr/lib/hadoop/lib

   Note: If all the files have not been deleted, then you must manually delete them.

   Open-source utilities are available that can delete these files across multiple nodes.

4. Verify that all the SAS Embedded Process directories and files have been deleted on all nodes except the node from which you ran the sasep-servers.sh -remove script.

   The sasep-servers.sh -remove script removes the file everywhere except on the node from which you ran the script.

   Note: If all the files have not been deleted, then you must manually delete them.

   Open-source utilities are available that can delete these files across multiple nodes.
5. Manually remove the SAS Embedded Process directories and files on the node from which you ran the script. Open-source utilities are available that can delete these files across multiple nodes.

The sasep-servers.sh -remove script removes the file everywhere except on the node from which you ran the script. The sasep-servers.sh -remove script displays instructions that are similar to the following example.

```
localhost WARN: Apparently, you are trying to uninstall SAS Embedded Process for Hadoop from the local node.
The binary files located at
  local_node/SAS/SASTKInDatabaseServerForHadoop/local_node/
SAS/SASACCESSstoHadoopMapReduceJARFiles will not be removed.
localhost WARN: The init script will be removed from /etc/init.d and the
  SAS Map Reduce JAR files will be removed from /usr/lib/hadoop-mapreduce/lib.
localhost WARN: The binary files located at local_node/SAS
  should be removed manually.
```

**TIP** You can use this command to find the location of any instance of the SAS Embedded Process:

```
ps -ef | grep depserver
```

6. Continue the installation process.

For more information, see “Using the SAS Deployment Manager to Deploy the SAS Embedded Process Parcel or Stack to the Cluster” on page 16.

---

**Upgrading from or Re-installing from the July 2015 Release of SAS 9.4 and Later**

**Overview**

The version number of the parcel or stack is calculated by the SAS Deployment Manager with the actual version of the installed product that you selected to deploy. You cannot deploy a parcel or stack that has the same version number as a parcel or stack that was previously deployed. The SAS Deployment Manager assigns a new version number or you can specify your own.

You can either deactivate the existing parcel or stack or remove it before upgrading or re-installing. If you want to deactivate the existing parcel or stack, use your cluster manager to deactivate the parcel or stack, and continue with the installation instructions in “Using the SAS Deployment Manager to Deploy the SAS Embedded Process Parcel or Stack to the Cluster” on page 16.

If you want to remove the existing parcel or stack, see either “Removing the SAS Embedded Process Parcel Using Cloudera Manager” on page 173 or “Removing the SAS Embedded Process Stack Using Ambari” on page 175.

**Removing the SAS Embedded Process Parcel Using Cloudera Manager**

**CAUTION:**

If you are using SAS Data Loader for Hadoop, you should remove the Quality Knowledge Base (QKB) and the SAS Data Management Accelerator for Spark
Removing the SAS Embedded Process removes the scripts that are used to remove these products. For more information, see “Removing the QKB” and “Removing the SAS Data Management Accelerator for Spark” in the *SAS Data Loader for Hadoop: Installation and Configuration Guide*.

To remove the SAS Embedded Process Parcel using Cloudera Manager, follow these steps:

1. Start Cloudera Manager.
2. Stop the SAS_EP service:
   a. On the Home page, click the down arrow next to SASEP service.
   b. Under SAS EPActions, select Stop, and click Stop.
   c. Click Close.
3. Delete the SASEP service from Cloudera Manager:
   a. On the Home page, click the down arrow next to SASEP service.
   b. Click Delete.
   c. Click Close.

   The SASEP service should not appear on the Home ⇒ Status tab.
4. Deactivate the SASEP parcel:
   a. Navigate to the Hosts ⇒ Parcels tab.
   b. Select Actions ⇒ Deactivate.
   You are asked to restart the cluster.
   c. Click Cancel.

   Note: Restarting the cluster is not required. If you want to restart and a rolling restart is available on your cluster, you can choose to perform a rolling restart instead of a full restart. For instructions about performing a rolling restart, see the Cloudera Manager documentation.
   d. Click OK to continue the deactivation.
5. Remove the SASEP parcel:
   a. Select Activate ⇒ Remove from Hosts.
   b. Click OK to confirm.
6. Delete the SASEP parcel.
7. Select Distribute ⇒ Delete.
8. Click OK to confirm.

   This step deletes the parcel files from the /opt/cloudera/parcel directory.
9. Manually remove the ep-config.xml file:

   **CAUTION:**
   If you fail to remove the ep-config.xml file, the SAS Embedded Process still appears to be available for use. Any software that uses the SAS Embedded Process fails.
   a. Log on to HDFS.

   ```bash
   sudo su - root
   ```
su - hdfs | hdfs-userid

Note: If your cluster is secured with Kerberos, the HDFS user must have a valid Kerberos ticket to access HDFS. This can be done with kinit.

b. Navigate to the /sas/ep/config/ directory on HDFS.
c. Locate the ep-config.xml file.
   hadoop fs -ls /sas/ep/config/ep-config.xml
d. Delete the directory.
   hadoop fs -rm r /sas/ep/

10. Continue the installation process.

   For more information, see “Using the SAS Deployment Manager to Deploy the SAS Embedded Process Parcel or Stack to the Cluster” on page 16.

Removing the SAS Embedded Process Stack Using Ambari

CAUTION:

If you are using SAS Data Loader for Hadoop, you should remove the Quality Knowledge Base (QKB) and the SAS Data Management Accelerator for Spark from the Hadoop nodes before removing the SAS Embedded Process.

Removing the SAS Embedded Process removes the scripts that are used to remove these products. For more information, see “Removing the QKB” and “Removing the SAS Data Management Accelerator for Spark” in the SAS Data Loader for Hadoop: Installation and Configuration Guide.

Note: You need root or passwordless sudo access to remove the stack.

To remove the SAS Embedded Process stack using Ambari, follow these steps:

1. Navigate to the SASHOME/SASHadoopConfigurationLibraries/2.4/Config/Deployment/stacks/sasep directory on the client where the SAS software is downloaded and installed.
   cd SASHOME/SASHadoopConfigurationLibraries/2.4/Config/Deployment/stacks/sasep
   The delete_stack.sh file should be in this directory.

2. Copy the delete_stack.sh file to a temporary directory where the cluster manager server is located. Here is an example using secure copy.
   scp delete_stack.sh user@cluster-manager-host:/mydir

3. Use this command to run the delete script.
   ./delete_stack.sh <Ambari-Admin-User-Name>

4. Enter the Ambari administrator password at the prompt.

   A message appears that offers options for removal. Enter one of the options as appropriate:
   • Enter 1 to remove only the ep-config.xml file.
   • Enter 2 to remove a specific version of the SAS Embedded Process. If you enter 2, a list of all the versions that are available for removal appears. You can then enter any of the versions to be deleted.
   • Enter 3 to remove all versions of the SAS Embedded Process.
5. You are prompted to restart the Ambari server in order to complete the removal of the **SASEP** service. To remove the SAS Embedded Process, you must restart the Ambari server.

6. Enter `y` to restart the Ambari server. The **SASEP** service is no longer listed on the Ambari dashboard user interface.

7. Continue the installation process.

    For more information, see “Using the SAS Deployment Manager to Deploy the SAS Embedded Process Parcel or Stack to the Cluster” on page 16.
Appendix 2
Upgrading from or Re-installing a Previous Version If Using Manual Deployment

Upgrading from or Re-installing from SAS 9.3

To upgrade or re-install from SAS 9.3, follow these steps:

1. Stop the Hadoop SAS Embedded Process using the 9.3 sasep-stop.all.sh script.
   
   \texttt{EPInstallDir/SAS/SASTKInDatabaseServerForHadoop/9.35/bin/sasep-stop.all.sh}
   
   \textit{EPInstallDir} is the master node where you installed the SAS Embedded Process.

2. Delete the Hadoop SAS Embedded Process from all nodes.
   
   \texttt{EPInstallDir/SAS/SASTKInDatabaseServerForHadoop/9.35/bin/sasep-delete.all.sh}

3. Verify that the sas.hadoop.ep.\textit{distribution-name}.jar files have been deleted.

   The JAR files are located at \textit{HadoopHome}/\textit{lib}.

   For Cloudera, the JAR files are typically located here:
   
   /opt/cloudera/parcels/CDH/lib/hadoop/lib

   For Hortonworks, the JAR files are typically located here:
   
   /usr/lib/hadoop/lib

4. Restart the MapReduce service to clear the SAS Hadoop MapReduce JAR files from the cache.

5. Continue the installation process.

   If re-installing from SAS 9.3, refer to your SAS 9.3 documentation.

   For instructions on how to upgrade to the latest release of SAS 9.4, see “Unzipping and Copying the In-Database Deployment Package to the Hadoop Cluster” on page 35.
Upgrading from or Re-installing from SAS 9.4 before the July 2015 Release of SAS 9.4

**CAUTION:**
If you are using SAS Data Loader for Hadoop, you should remove the Quality Knowledge Base (QKB) and the SAS Data Management Accelerator for Spark from the Hadoop nodes before removing the SAS Embedded Process.

Removing the SAS Embedded Process removes the scripts that are used to remove these products. For more information, see “Removing the QKB” and “Removing the SAS Data Management Accelerator for Spark” in the *SAS Data Loader for Hadoop: Installation and Configuration Guide*.

To upgrade or re-install from a version of SAS 9.4 before the July 2015 release of SAS 9.4, follow these steps:


   ```
   EPInstallDir/SAS/SASTKInDatabaseServerForHadoop/9.*/bin/sasep-servers.sh -stop -hostfile host-list-filename | -host "host-list"
   ```

   *EPInstallDir* is the master node where you installed the SAS Embedded Process.

   For more information, see the SASEP-SERVERS.SH syntax section of the *SAS In-Database Products: Administrator’s Guide* that came with your release.

2. Remove the SAS Embedded Process from all nodes.

   ```
   EPInstallDir/SAS/SASTKInDatabaseServerForHadoop/9.*/bin/sasep-servers.sh -remove -hostfile host-list-filename | -host "host-list"
   -mrhome dir
   ```

   *Note:* This step ensures that all old SAS Hadoop MapReduce JAR files are removed.

   For more information, see the SASEP-SERVERS.SH syntax section of the *SAS In-Database Products: Administrator’s Guide* that came with your release.

3. Restart the MapReduce service to clear the SAS Hadoop MapReduce JAR files from the cache.

4. Verify that all files associated with the SAS Embedded Process have been removed.

   *Note:* If all the files have not been deleted, then you must manually delete them. Open-source utilities are available that can delete these files across multiple nodes.

   a. Verify that the sas.hadoop.ep.apache*.jar files have been deleted.

      The JAR files are located at *HadoopHome/lib*.

      For Cloudera, the JAR files are typically located here:

      ```
      /opt/cloudera/parcels/CDH/lib/hadoop/lib
      ```

      For Hortonworks, the JAR files are typically located here:

      ```
      /usr/lib/hadoop/lib
      ```

   b. Verify that all the SAS Embedded Process directories and files have been deleted on all nodes except the node from which you ran the sasep-servers.sh -remove script. The sasep-servers.sh -remove script removes the file everywhere except on the node from which you ran the script.
c. Manually remove the SAS Embedded Process directories and files on the master node (EPInstallDir) from which you ran the script.

The sasep-servers.sh -remove script removes the file everywhere except on the node from which you ran the script. The sasep-servers.sh -remove script displays instructions that are similar to the following example.

```
localhost WARN: Apparently, you are trying to uninstall SAS Embedded Process for Hadoop from the local node.
The binary files located at
  local_node/SAS/SASTKInDatabaseServerForHadoop/local_node/
SAS/SASACCESStoHadoopMapReduceJARFiles will not be removed.
localhost WARN: The init script will be removed from /etc/init.d and the
  SAS Map Reduce JAR files will be removed from /usr/lib/hadoop-mapreduce/lib.
localhost WARN: The binary files located at local_node/SAS
should be removed manually.
```

Tip: You can use this command to find the location of any instance of the SAS Embedded Process:

```
ps -ef | grep depserver
```

5. Continue the installation process.

If you are re-installing from the version of SAS 9.4 prior to July 2015, see the SAS 9.4 documentation for that release.

For instructions on how to upgrade to the latest release of SAS 9.4, see “Unzipping and Copying the In-Database Deployment Package to the Hadoop Cluster” on page 35.

---

**Upgrading from or Re-installing from the July 2015 Release of SAS 9.4 or Later**

**CAUTION:**

If you are using SAS Data Loader for Hadoop, you should remove the Quality Knowledge Base (QKB) and the SAS Data Management Accelerator for Spark from the Hadoop nodes before removing the SAS Embedded Process.

Removing the SAS Embedded Process removes the scripts that are used to remove these products. For more information, see “Removing the QKB” and “Removing the SAS Data Management Accelerator for Spark” in the *SAS Data Loader for Hadoop: Installation and Configuration Guide*.

To upgrade or re-install from the July 2015 release of SAS 9.4 or later, follow these steps:

1. Locate the sasep-admin.sh file.

   This file is in the EPInstallDir/SASEPHome/bin directory. EPInstallDir is where you installed the SAS Embedded Process.

   One way to find the EPInstallDir directory is to look at the sas.ep.classpath property in the ep-config.xml file. The ep-config.xml file is located on HDFS in the /sas/ep/config/ directory.

a. Enter this Hadoop command to read the ep-config.xml file on HDFS.

```
hadoop fs -cat /sas/ep/config/ep-config.xml
```
b. Search for the sas.ep.classpath property.

c. Copy the directory path.

   The path should be `EPInstallDir/SASEPHome/` where `EPInstallDir` is where you installed the SAS Embedded Process.

d. Navigate to the `EPInstallDir/SASEPHome/bin` directory.

2. Run `sasep-admin.sh -remove` script.

   This script removes the SAS Embedded Process from the data nodes.

3. Run this command to remove the SASEPHome directories from the master node.

   `rm -rf SASEPHome`

4. Continue the installation process.

   If you are re-installing from the version of SAS 9.4 after July 2015, see the SAS 9.4 documentation for that release.

   For instructions on how to upgrade to the latest release of SAS 9.4, see “Unzipping and Copying the In-Database Deployment Package to the Hadoop Cluster” on page 35.
Recommended Reading

Here is the recommended reading list for this title:

- *SAS/ACCESS for Relational Databases: Reference*
- *SAS Data Loader for Hadoop: User’s Guide*
- *SAS Data Quality Accelerator for Teradata: User’s Guide*
- *SAS DS2 Language Reference*
- *SAS Hadoop Configuration Guide for Base SAS and SAS/ACCESS*
- *SAS High-Performance Analytics Infrastructure: Installation and Configuration Guide*
- *SAS In-Database Products: User’s Guide*
- *SAS Model Manager: Administrator’s Guide*
- *SAS Viya Deployment Guide*

For a complete list of SAS publications, go to sas.com/store/books. If you have questions about which titles you need, please contact a SAS Representative:

SAS Books
SAS Campus Drive
Cary, NC 27513-2414
Phone: 1-800-727-0025
Fax: 1-919-677-4444
Email: sasbook@sas.com
Web address: sas.com/store/books
Index

**Special Characters**
- %INDAC_PUBLISH_FORMATS macro 81
- %INDAC_PUBLISH_MODEL macro 81
- %INDB2_PUBLISH_COMPILEUDF macro 99
  - running 100
  - syntax 101
- %INDB2_PUBLISH_DELETEUDF macro 103
  - running 103
  - syntax 105
- %INDB2_PUBLISH_FORMATS macro 87
  - %INDB2_PUBLISH_MODEL macro 87
- %INDGP_PUBLISH_COMPILEUDF macro 116
  - running 117
  - syntax 119
- %INDGP_PUBLISH_COMPILEUDF_E P macro 120
  - running 120
  - syntax 122
- %INDGP_PUBLISH_FORMATS macro 110
  - %INDGP_PUBLISH_MODEL macro 110
- %INDHN_PUBLISH_MODEL macro 149
- %INDNZ_PUBLISH_COMPILEUDF macro 136
  - running 137
  - syntax 138
- %INDNZ_PUBLISH_FORMATS macro 127
- %INDNZ_PUBLISH_JAZLIB macro 134
  - running 134
  - syntax 135
- %INDNZ_PUBLISH_MODEL macro 127
- %INDOR_PUBLISH_MODEL macro 143
- %INDTD_PUBLISH_FORMATS macro 57
- %INDTD_PUBLISH_MODEL macro 57

**A**
- ACTION= argument
  - %INDB2_PUBLISH_COMPILEUDF macro 102
  - %INDB2_PUBLISH_DELETEUDF macro 105
  - %INDGP_PUBLISH_COMPILEUDF macro 119
  - %INDGP_PUBLISH_COMPILEUDF_EP macro 123
  - %INDNZ_PUBLISH_COMPILEUDF macro 139
  - %INDNZ_PUBLISH_JAZLIB macro 136
- Ambari deploy SAS Embedded Process stack 16, 30
  - remove SAS Embedded Process stack 175
- Aster documentation for publishing formats and scoring models 86
  - in-database deployment package 81
  - installation and configuration 82
  - permissions 85
  - SAS Embedded Process 81
  - SAS/ACCESS Interface 81
  - SQL/MR functions 82
  - authorization for stored procedures 71

**B**
- binary files
  - for Aster 82
  - for DB2 functions 93
  - for Greenplum functions 113
  - for Netezza functions 130
- BTEQ 74
Cloudera
Hadoop installation and configuration using the SAS Deployment Manager 13
manual Hadoop installation and configuration 33
Cloudera Manager
deploy SAS Embedded Process parcel 16, 28
remove SAS Embedded Process stack 173
COMPILER_PATH= argument
%INDB2_PUBLISH_COMPILEUDF macro 102
configuration
Aster 81
DB2 89
Greenplum 111
HDFS for Model Manager 167
IBM BigInsights 33
MapR 33
Model Manager database 164
Netezza 129
Oracle 143
Pivotal HD 33
SAP HANA 150
SPD Server 159
Teradata 61
configuration file when to regenerate 42
customizing the QKB 76

data quality stored procedures
See stored procedures
DATABASE= argument
%INDB2_PUBLISH_COMPILEUDF macro 102
%INDB2_PUBLISH_DELETEUDF macro 105
%INDGP_PUBLISH_COMPILEUDF macro 119
%INDGP_PUBLISH_COMPILEUDF_EP macro 123
%INDNZ_PUBLISH_COMPILEUDF macro 139
%INDNZ_PUBLISH_JAZLIB macro 135
DataFlux Data Management Studio customizing the QKB 76
DB2
documentation for publishing formats or scoring models 108
function publishing process 88
installation and configuration 89
JDBC Driver 166
permissions 106
preparing for SAS Model Manager use 163
SAS Embedded Process 87
SAS/ACCESS Interface 87
unpacking self-extracting archive files 93, 95
DB2IDA command 98
DB2PATH= argument
%INDB2_PUBLISH_COMPILEUDF macro 101
DB2SET command 94
syntax 97
DBSCHEMA= argument
%INDNZ_PUBLISH_COMPILEUDF macro 139
%INDNZ_PUBLISH_JAZLIB macro 135
documentation
for in-database processing in Hadoop 11
for in-database processing in SAP HANA 158
for publishing formats and scoring models in Aster 86
for publishing formats and scoring models in DB2 108
for publishing formats and scoring models in Greenplum 126
for publishing formats and scoring models in Netezza 141
for publishing formats and scoring models in Teradata 59
for publishing scoring models in Oracle 148
dq_grant.sh script 70, 71
dq_install.sh script 70
dq_uninstall script 70
dq_uninstall.sh script 77
formats library
DB2 installation 92, 94, 114
Greenplum installation 113
Netezza installation 130
Teradata installation 65
function publishing process
DB2 88
Netezza 128
functions
SAS_COMPILEUDF (DB2) 92, 94, 99, 106, 114
SAS_COMPILEUDF (Greenplum) 113, 116, 123
SAS_COMPILEUDF (Netezza) 130, 136
SAS_COPYUDF (Greenplum) 123
SAS_DEHExUDF (Greenplum) 123
SAS_DELETEUDF (DB2) 92, 94, 103, 106, 114
SAS_DIRECTORYUDF (Greenplum) 123
SAS_DIRECTORYUDF (Netezza) 136
SAS_EP (Greenplum) 124
SAS_HEXTOTEXTUDF (Netezza) 136
SAS_PUT( ) (Aster) 81
SAS_PUT( ) (DB2) 87
SAS_PUT( ) (Greenplum) 110
SAS_PUT( ) (Netezza) 127, 128
SAS_PUT( ) (Teradata) 57
SAS_SCORE( ) (Aster) 82
SQL/MR (Aster) 82
overview of manual configuration steps 34
permissions 10
preparing for SAS Model Manager use 163
SAS/ACCESS Interface 9
unpacking self-extracting archive files 36

H
Hadoop
backward compatibility 10
configuring HDFS using Model Manager 167
in-database deployment package 9
installation and configuration for IBM BigInsights 33
installation and configuration for MapR 33
installation and configuration for Pivotal HD 33
overview of configuration steps using the SAS Deployment Manager 15
overview of manual configuration steps 34
permissions 10
preparing for SAS Model Manager use 163
SAS/ACCESS Interface 9
unpacking self-extracting archive files 36

I
IBM BigInsights
Hadoop installation and configuration 33
Hadoop installation and configuration using the SAS Deployment Manager 13
in-database deployment package for Aster overview 81
prerequisites 81
in-database deployment package for DB2 overview 87
prerequisites 87
in-database deployment package for Greenplum overview 110
prerequisites 109
in-database deployment package for Hadoop introduction 7
overview 8
prerequisites 9
in-database deployment package for Netezza overview 127
prerequisites 127
in-database deployment package for Oracle overview 143
prerequisites 143
in-database deployment package for SAP HANA overview 149
prerequisites 149

G
global variables
See variables
Greenplum
documentation for publishing formats and scoring models 126
in-database deployment package 109
installation and configuration 111
JDBC Driver 167
permissions 126
preparing for SAS Model Manager use 163
SAS Embedded Process 124
SAS/ACCESS Interface 109
semaphore requirements 125
unpacking self-extracting archive files 113
in-database deployment package for Teradata overview 57 prerequisites 57
INDCONN macro variable 100, 104, 118, 121, 134, 137

J
JDBC Driver DB2 166 Greenplum 167 Netezza 167 Teradata 167 JDBC JAR file locations 166

M
macro variables See variables macros %INDAC_PUBLISH_FORMATS 81 %INDAC_PUBLISH_MODEL 81

%INDB2_PUBLISH_COMPILEUDF 99, 101 %INDB2_PUBLISH_DELETEUDF 103, 105 %INDB2_PUBLISH_FORMATS 87 %INDB2_PUBLISH_MODEL 87 %INDGP_PUBLISH_COMPILEUDF 116, 119 %INDGP_PUBLISH_COMPILEUDF_EP 122 %INDGP_PUBLISH_FORMATS 110 %INDGP_PUBLISH_MODEL 110 %INDHN_PUBLISH_MODEL 149 %INDNZ_PUBLISH_COMPILEUDF 136, 138 %INDNZ_PUBLISH_FORMATS 127 %INDNZ_PUBLISH_JAZLIB 134, 135 %INDNZ_PUBLISH_MODEL 127 %INDOR_PUBLISH_MODEL 143 %INDTD_PUBLISH_FORMATS 57 %INDTD_PUBLISH_MODEL 57 manual installation SAS In-Database Deployment Package 35

MapR additional configuration 50 Hadoop installation and configuration 33 YARN application CLASSPATH 50 Model Manager configuration 163 configuring a database 164 configuring HDFS 167 creating tables 165 JDBC Driver 166

N
Netezza documentation for publishing formats and scoring models 141 function publishing process 128 in-database deployment package 127 installation and configuration 129 JDBC Driver 167 permissions 140 preparing for SAS Model Manager use 163 publishing SAS formats library 134 SAS Embedded Process 127 sas_ep cartridge 132 SAS/ACCESS Interface 127
Oracle documentation for publishing formats and scoring models 148
in-database deployment package 143
preparing for SAS Model Manager use 163
SAS Embedded Process 143
SAS/ACCESS Interface 143

OUTDIR= argument
%IND2_PUBLISH_COMPILEUDF macro 102
%IND2_PUBLISH_DELETEUDF macro 103
%IND2_PUBLISH_COMPILEUDF_ EP macro 122

P
permissions
for Aster 85
for DB2 106
for Greenplum 126
for Hadoop 10
for Netezza 140
for Oracle 147
for SAP HANA 157
for SPD Server 159
for Teradata 58
Pivotal
Hadoop installation and configuration 33
Pivotal HD
Hadoop installation and configuration using the SAS Deployment Manager 13
PSFTP (DB2) 89
publishing
Aster permissions 85
DB2 permissions 106
functions in DB2 88
functions in Netezza 128
Greenplum permissions 126
Hadoop permissions 10
Netezza permissions 140
Oracle permissions 147
SAP HANA permissions 157
SPD Server permissions 159
Teradata permissions 58
Q
QKB
customizing 76
packaging for deployment 72
updates 76
qkb_pack script 72
R
re-installing a previous version
Hadoop 177
removing stored procedures 77
RPM file (Teradata) 65
S
SAP HANA
documentation for in-database processing 158
in-database deployment package 149
installation and configuration 150
permissions 157
SAS Embedded Process 149, 156
SAS/ACCESS Interface 149
semaphore requirements 156
SAS Deployment Manager
using to deploy Hadoop in-database deployment package 13
using to deploy the Teradata in-database deployment package 61
SAS Embedded Process
adjusting performance 51
Aster 81
check status (DB2) 98
check status (Teradata) 67
Cloudera 5.5 54
configuration for HCatalog file formats 48
controlling (DB2) 98
controlling (Greenplum) 124
controlling (Hadoop) 40
controlling (SAP HANA) 156
controlling (Teradata) 67
DB2 87
disable or enable (DB2) 98
disable or enable (Teradata) 67
Greenplum 124
Hadoop 9
Netezza 127, 131
Oracle 143
overview 8
SAP HANA 149, 156
shutdown (DB2) 98
shutdown (Teradata) 67
support functions (Teradata) 67
Teradata 57
upgrading from a previous version
(Aster) 82
upgrading from a previous version
(DB2) 89
upgrading from a previous version
(Netezza) 129
upgrading from a previous version
(Oracle) 144
upgrading from a previous version
(Teradata) 62
SAS FILENAME SFTP statement (DB2)
88
SAS formats library
DB2 92, 94, 114
Greenplum 113
Netezza 130, 134
Teradata 65
upgrading from a previous version
(Greenplum) 111
upgrading from a previous versions
(DB2) 89
upgrading from a previous versions
(Netezza) 129
upgrading from a previous versions
(Teradata) 62
SAS Foundation 9, 57, 81, 109, 127, 143, 149
SAS Hadoop MapReduce JAR files 36
SAS In-Database products 4
SAS_COMPILEUDF function
actions for DB2 99
actions for Greenplum 116
actions for Netezza 136
binary files for DB2 92, 94, 114
binary files for Greenplum 113
binary files for Netezza 130
validating publication for DB2 106
validating publication for Greenplum 123
SAS_COPYUDF function 116
validating publication for Greenplum 123
SAS_DEHEXUDF function 117
validating publication for Greenplum 123
SAS_DELETEUDF function
actions for DB2 103
binary files for DB2 92, 94, 114
validating publication for DB2 106
SAS_DIRECTORYUDF function 117, 136
validating publication for Greenplum 123
sas_ep cartridge 132
SAS_EP function
validating publication for Greenplum 124
SAS_HEXTOTEXTUDF function 136
SAS_PUT( ) function
Aster 81
DB2 88
Greenplum 110
Netezza 128
Teradata 57
SAS_SCORE( ) function
publishing 82
validating publication for Aster 85
SAS_SYSLNLIB (Teradata) 67
SAS/ACCESS Interface to Aster 81
SAS/ACCESS Interface to Greenplum 109
SAS/ACCESS Interface to Hadoop 9
SAS/ACCESS Interface to Netezza 127
SAS/ACCESS Interface to Oracle 143
SAS/ACCESS Interface to SAP HANA 149
SAS/ACCESS Interface to Teradata 57
sasep-admin.sh script
overview 40
syntax 41
sasepfunc function package 67
SASLIB database (Netezza) 137
SASLIB schema
DB2 100, 103
Greenplum 117
SASUDF_COMPILER_PATH global variable 99
SASUDF_DB2PATH global variable 99
scoring functions in SAS Model Manager 164
scripts for installation 70
self-extracting archive files
unpacking for Aster 82
unpacking for DB2 93, 95
unpacking for Greenplum 113
unpacking for Hadoop 36
semaphore requirements
Greenplum 125
SAP HANA 156
SFTP statement 88
SPD Server
in-database deployment package 159
permissions 159
SQL/MR functions (Aster) 82
SSH software (DB2) 88
stored procedures
creating 70
removing from database 77

T
 tables
creating for SAS Model Manager 165
Teradata
BTEQ 74
documentation for publishing formats and scoring models 59
in-database deployment package 57
installation and configuration 61
JDBC Driver 167
Parallel Upgrade Tool 66
permissions 58
preparing for SAS Model Manager use 163
SAS Embedded Process 57
SAS Embedded Process support functions 67
SAS/ACCESS Interface 57
sasepfunc function package 67
troubleshooting installation 75

U
unpacking self-extracting archive files
for Aster 82
for DB2 93, 95
for Greenplum 113

V
validating publication of functions and variables for DB2 106
validating publication of functions for Aster 85
validating publication of functions for Greenplum 123, 124
variables
INDCONN macro variable 100, 104, 118, 121, 134, 137
SASUDF_COMPILER_PATH global variable 99
SASUDF_DB2PATH global variable 99
verifying installation 74

Y
YARN application CLASSPATH for MapR 50
Gain Greater Insight into Your SAS® Software with SAS Books.

Discover all that you need on your journey to knowledge and empowerment.

support.sas.com/bookstore for additional books and resources.