



# **SAS<sup>®</sup> 9.4 In-Database Products: Administrator's Guide, Eighth Edition**

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2016. *SAS® 9.4 In-Database Products: Administrator's Guide, Eighth Edition*. Cary, NC: SAS Institute Inc.

**SAS® 9.4 In-Database Products: Administrator's Guide, Eighth Edition**

Copyright © 2016, SAS Institute Inc., Cary, NC, USA

All Rights Reserved. Produced in the United States of America.

**For a hard copy book:** No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

**For a web download or e-book:** Your use of this publication shall be governed by the terms established by the vendor at the time you acquire this publication.

The scanning, uploading, and distribution of this book via the Internet or any other means without the permission of the publisher is illegal and punishable by law. Please purchase only authorized electronic editions and do not participate in or encourage electronic piracy of copyrighted materials. Your support of others' rights is appreciated.

**U.S. Government License Rights; Restricted Rights:** The Software and its documentation is commercial computer software developed at private expense and is provided with RESTRICTED RIGHTS to the United States Government. Use, duplication, or disclosure of the Software by the United States Government is subject to the license terms of this Agreement pursuant to, as applicable, FAR 12.212, DFAR 227.7202-1(a), DFAR 227.7202-3(a), and DFAR 227.7202-4, and, to the extent required under U.S. federal law, the minimum restricted rights as set out in FAR 52.227-19 (DEC 2007). If FAR 52.227-19 is applicable, this provision serves as notice under clause (c) thereof and no other notice is required to be affixed to the Software or documentation. The Government's rights in Software and documentation shall be only those set forth in this Agreement.

SAS Institute Inc., SAS Campus Drive, Cary, NC 27513-2414

February 2017

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

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

9.4-P1:indbag

With respect to CENTOS third-party technology included with the vApp ("CENTOS"), CENTOS is open-source software that is used with the Software and is not owned by SAS. Use, copying, distribution, and modification of CENTOS is governed by the CENTOS EULA and the GNU General Public License (GPL) version 2.0. The CENTOS EULA can be found at [http://mirror.centos.org/centos/6/os/x86\\_64/EULA](http://mirror.centos.org/centos/6/os/x86_64/EULA). A copy of the GPL license can be found at <http://www.opensource.org/licenses/gpl-2.0> or can be obtained by writing to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02110-1301 USA. The source code for CENTOS is available at <http://vault.centos.org/>.

With respect to open-vm-tools third party technology included in the vApp ("VMTOOLS"), VMTOOLS is open-source software that is used with the Software and is not owned by SAS. Use, copying, distribution, and modification of VMTOOLS is governed by the GNU General Public License (GPL) version 2.0. A copy of the GPL license can be found at <http://opensource.org/licenses/gpl-2.0> or can be obtained by writing to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02110-1301 USA. The source code for VMTOOLS is available at <http://sourceforge.net/projects/open-vm-tools/>.

With respect to VIRTUALBOX third-party technology included in the vApp ("VIRTUALBOX"), VIRTUALBOX is open-source software that is used with the Software and is not owned by SAS. Use, copying, distribution, and modification of VIRTUALBOX is governed by the GNU General Public License (GPL) version 2.0. A copy of the GPL license can be found at <http://opensource.org/licenses/gpl-2.0> or can be obtained by writing to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02110-1301 USA. The source code for VIRTUALBOX is available at <http://www.virtualbox.org/>.

---

# Contents

<i>What's New in SAS 9.4 In-Database Products: Administrator's Guide</i> . . . . .	<i>vii</i>
--	------------

## PART 1 Introduction 1

<b>Chapter 1 • Introduction to the Administrator's Guide</b> . . . . .	<b>3</b>
Overview of SAS In-Database Products . . . . .	3
What Is Covered in This Document? . . . . .	4

## PART 2 Administrator's Guide for Hadoop 5

<b>Chapter 2 • In-Database Deployment Package for Hadoop</b> . . . . .	<b>7</b>
Introduction to the In-Database Deployment Package for Hadoop . . . . .	7
Overview of the In-Database Deployment Package for Hadoop . . . . .	8
Overview of the SAS Embedded Process . . . . .	8
Prerequisites for Installing the In-Database Deployment Package for Hadoop . . . . .	9
Backward Compatibility . . . . .	10
Hadoop Permissions . . . . .	10
Documentation for Using In-Database Processing in Hadoop . . . . .	11
<b>Chapter 3 • Deploying the In-Database Deployment Package Using the SAS Deployment Manager</b> . . . . .	<b>13</b>
When to Deploy the SAS In-Database Deployment Package . . . . .	
Using the SAS Deployment Manager . . . . .	13
Prerequisites for Using the SAS Deployment Manager to Deploy the In-Database Deployment Package . . . . .	14
Hadoop Installation and Configuration Steps Using the SAS Deployment Manager . . . . .	15
Using the SAS Deployment Manager to Deploy the SAS Embedded Process Parcel or Stack to the Cluster . . . . .	16
Deploying the SAS Embedded Process Parcel on Cloudera . . . . .	25
Deploying the SAS Embedded Process Stack on Hortonworks, IBM BigInsights, or Pivotal HD . . . . .	27
<b>Chapter 4 • Deploying the In-Database Deployment Package Manually</b> . . . . .	<b>31</b>
When to Deploy the SAS In-Database Deployment Package Manually . . . . .	31
Hadoop Manual Installation and Configuration Steps . . . . .	32
Unzipping and Copying the In-Database Deployment Package to the Hadoop Cluster . . . . .	33
Installing the SAS Embedded Process . . . . .	34
SASEP-ADMIN.SH Script . . . . .	38
<b>Chapter 5 • Additional Configuration for the SAS Embedded Process</b> . . . . .	<b>45</b>
Overview of Additional Configuration Tasks . . . . .	45
Additional Configuration Needed to Use HCatalog File Formats . . . . .	46
Adding the YARN Application CLASSPATH to the Configuration File for MapR Distributions . . . . .	48

Changing the Trace Level .....	49
Adjusting the SAS Embedded Process Performance .....	49

### PART 3 Administrator's Guide for Teradata 53

<b>Chapter 6 • In-Database Deployment Package for Teradata .....</b>	<b>55</b>
Prerequisites .....	55
Overview of the In-Database Deployment Package for Teradata .....	55
Teradata Permissions for Publishing Formats and Scoring Models .....	57
Documentation for Using In-Database Processing in Teradata .....	57
<b>Chapter 7 • Deploying the In-Database Deployment Package: Teradata .....</b>	<b>59</b>
Teradata Installation and Configuration Steps .....	59
Upgrading from a Previous Version .....	60
Installing the SAS Formats Library and the SAS Embedded Process .....	63
Controlling the SAS Embedded Process .....	65
<b>Chapter 8 • SAS Data Quality Accelerator for Teradata .....</b>	<b>67</b>
Introduction .....	67
Installing SAS Data Quality Accelerator Stored Procedures in the Teradata Database ..	68
Obtaining and Deploying a QKB .....	69
Validating the Accelerator Installation .....	72
Troubleshooting the Accelerator Installation .....	73
Updating and Customizing a QKB .....	74
Removing SAS Data Quality Accelerator from the Teradata Database .....	75

### PART 4 Administrator's Guides for Aster, DB2, Greenplum, Netezza, Oracle, SAP HANA, and SPD Server 77

<b>Chapter 9 • Administrator's Guide for Aster .....</b>	<b>79</b>
In-Database Deployment Package for Aster .....	79
Aster Installation and Configuration .....	80
Validating the Publishing of the SAS_SCORE() and the SAS_PUT() Functions ....	83
Aster Permissions .....	83
Documentation for Using In-Database Processing in Aster .....	84
<b>Chapter 10 • Administrator's Guide for DB2 .....</b>	<b>85</b>
In-Database Deployment Package for DB2 .....	85
Function Publishing Process in DB2 .....	86
DB2 Installation and Configuration .....	87
Validating the Publishing of SAS_COMPILEUDF and SAS_DELETEUDF Functions and Global Variables .....	104
DB2 Permissions .....	104
Documentation for Using In-Database Processing in DB2 .....	106
<b>Chapter 11 • Administrator's Guide for Greenplum .....</b>	<b>107</b>
In-Database Deployment Package for Greenplum .....	107
Greenplum Installation and Configuration .....	109
Validation of Publishing Functions .....	121
Controlling the SAS Embedded Process .....	122
Semaphore Requirements When Using the SAS Embedded Process for Greenplum ..	123

Greenplum Permissions .....	124
Documentation for Using In-Database Processing in Greenplum .....	124
<b>Chapter 12 • Administrator's Guide for Netezza .....</b>	<b>125</b>
In-Database Deployment Package for Netezza .....	125
Function Publishing Process in Netezza .....	126
Netezza Installation and Configuration .....	127
Netezza Permissions .....	138
Documentation for Using In-Database Processing in Netezza .....	139
<b>Chapter 13 • Administrator's Guide for Oracle .....</b>	<b>141</b>
In-Database Deployment Package for Oracle .....	141
Oracle Installation and Configuration .....	142
Oracle Permissions .....	145
Documentation for Using In-Database Processing in Oracle .....	146
<b>Chapter 14 • Administrator's Guide for SAP HANA .....</b>	<b>147</b>
In-Database Deployment Package for SAP HANA .....	147
SAP HANA Installation and Configuration .....	148
Installing the SASLINK AFL Plugins on the Appliance (SPS11) .....	152
Importing the SAS_EP Stored Procedure .....	153
Auxiliary Wrapper Procedures (SPS11) .....	153
Controlling the SAS Embedded Process .....	154
Semaphore Requirements When Using the SAS Embedded Process for SAP HANA .....	154
SAP HANA Permissions .....	155
Documentation for Using In-Database Processing in SAP HANA .....	156
<b>Chapter 15 • Administrator's Guide for SPD Server .....</b>	<b>157</b>
Installation and Configuration Requirements for the SAS Scoring Accelerator for SPD Server .....	157
Where to Go from Here .....	157
 PART 5   Configurations for SAS Model Manager .....	 159
<b>Chapter 16 • Configuring SAS Model Manager .....</b>	<b>161</b>
Preparing a Data Management System for Use with SAS Model Manager .....	161
Configuring a Database .....	162
Configuring a Hadoop Distributed File System .....	165
 PART 6   Upgrading from or Reinstalling a Previous Version of the Hadoop In-Database Deployment Package .....	 167
<b>Appendix 1 • Upgrading from or Reinstalling a Previous Version If Using           SAS Deployment Manager .....</b>	<b>169</b>
Upgrading from SAS 9.3 .....	169
Upgrading from SAS 9.4 before the July 2015 Release of SAS 9.4 .....	170
Upgrading from or Reinstalling from the July 2015 Release of SAS 9.4 and Later ..	171
<b>Appendix 2 • Upgrading from or Reinstalling a Previous Version If Using           Manual Deployment .....</b>	<b>175</b>
Upgrading from or Reinstalling from SAS 9.3 .....	175

Upgrading from or Reinstalling from SAS 9.4 before the July 2015 Release of SAS 9.4 .....	176
Upgrading from or Reinstalling from the July 2015 Release of SAS 9.4 or Later . . . .	177
<b>Recommended Reading</b> .....	<b>179</b>
<b>Index</b> .....	<b>181</b>

# 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 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 Hana 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 ep-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 are included in the in-database deployment package for Hadoop and Teradata, respectively.
- 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, has been added that enables you to deploy the SAS Embedded Process for Hadoop without root or sudo access.
  - 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 Hana 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.



## ***Part 1***

---

# Introduction

### *Chapter 1*

***Introduction to the Administrator's Guide*** ..... 3





## Chapter 1

# Introduction to the Administrator's Guide

---

<b>Overview of SAS In-Database Products .....</b>	<b>3</b>
<b>What Is Covered in This Document? .....</b>	<b>4</b>

---

## 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

<i>Chapter 2</i>	
<b><i>In-Database Deployment Package for Hadoop</i></b> .....	<b><i>7</i></b>
<i>Chapter 3</i>	
<b><i>Deploying the In-Database Deployment Package Using the SAS Deployment Manager</i></b> .....	<b><i>13</i></b>
<i>Chapter 4</i>	
<b><i>Deploying the In-Database Deployment Package Manually</i></b> .....	<b><i>31</i></b>
<i>Chapter 5</i>	
<b><i>Additional Configuration for the SAS Embedded Process</i></b> .....	<b><i>45</i></b>



## Chapter 2

# In-Database Deployment Package for Hadoop

---

<b>Introduction to the In-Database Deployment Package for Hadoop . . . . .</b>	<b>7</b>
<b>Overview of the In-Database Deployment Package for Hadoop . . . . .</b>	<b>8</b>
<b>Overview of the SAS Embedded Process . . . . .</b>	<b>8</b>
<b>Prerequisites for Installing the In-Database Deployment Package for Hadoop . . . .</b>	<b>9</b>
<b>Backward Compatibility . . . . .</b>	<b>10</b>
<b>Hadoop Permissions . . . . .</b>	<b>10</b>
<b>Documentation for Using In-Database Processing in Hadoop . . . . .</b>	<b>11</b>

---

## 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.

For more information, see *SAS Contextual Analysis In-Database Scoring in Hadoop: User's Guide*

- Read and write data to HDFS in parallel for SAS High-Performance Analytics.

*Note:* For deployments that use SAS High-Performance Deployment of Hadoop for the co-located data provider, and access SASHDAT tables exclusively, SAS/ACCESS and SAS Embedded Process are not needed.

*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*.

---

## 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.

SAS Product Licensed	SAS Component Installed with the SAS Embedded Process
SAS Data Loader for Hadoop 3.1	SAS Data Quality Accelerator for Hadoop
SAS Contextual Analysis In-Database Scoring for Hadoop 2.1	SAS Contextual Analysis In-Database Scoring for Hadoop

*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

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 31.

---

## 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/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.

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 2.1** In-database Deployment Package for Hadoop Backward Compatibility

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

\* See Step 7 in “Installing the SAS Embedded Process” on page 34.

\*\* See “SASEP-ADMIN.SH Script” on page 38.

## 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.



---

## Documentation for Using In-Database Processing in Hadoop

For information about using in-database processing in Hadoop, see the following publications:

- FILENAME Statement, Hadoop Access Method in *SAS 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*



## Chapter 3

# Deploying the In-Database Deployment Package Using the SAS Deployment Manager

---

<b>When to Deploy the SAS In-Database Deployment Package Using the SAS Deployment Manager</b> .....	<b>13</b>
<b>Prerequisites for Using the SAS Deployment Manager to Deploy the In-Database Deployment Package</b> .....	<b>14</b>
<b>Hadoop Installation and Configuration Steps Using the SAS Deployment Manager</b> .....	<b>15</b>
<b>Using the SAS Deployment Manager to Deploy the SAS Embedded Process Parcel or Stack to the Cluster</b> .....	<b>16</b>
<b>Deploying the SAS Embedded Process Parcel on Cloudera</b> .....	<b>25</b>
<b>Deploying the SAS Embedded Process Stack on Hortonworks, IBM BigInsights, or Pivotal HD</b> .....	<b>27</b>
Deploying the SAS Embedded Process Stack for the First Time .....	27
Deploying a New Version of the SAS Embedded Process Stack .....	28

---

## 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 31](#).

**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 you are using Hortonworks, the requiretty option is enabled, and the SAS Embedded Process is installed using the SAS Deployment Manager, the Ambari server must be restarted after deployment. 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

## 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 `sepcorehadoop-12.00000-1.sh` file. Now you want to install SAS Viya and the in-database deployment package that shipped with SAS Viya, which is `sepcorehadoop-11.50000-1.sh`. In this instance, you would not install the in-database deployment package that shipped with SAS Viya because it is an earlier version; that is, 11.50000 versus 12.00000.

Step	Description	Where to Go for Information
1	Review these topics:	<ul style="list-style-type: none"> <li>• <a href="#">“Prerequisites for Installing the In-Database Deployment Package for Hadoop”</a> on page 9</li> <li>• <a href="#">“Backward Compatibility”</a> on page 10</li> <li>• <a href="#">“When to Deploy the SAS In-Database Deployment Package Using the SAS Deployment Manager”</a> on page 13</li> <li>• <a href="#">“Prerequisites for Using the SAS Deployment Manager to Deploy the In-Database Deployment Package”</a> on page 14</li> </ul>
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.	<a href="#">“Configuring SAS/ACCESS for Hadoop”</a> in <a href="#">SAS Hadoop Configuration Guide for Base SAS and SAS/ACCESS</a>
3	If you are upgrading from or reinstalling a previous release, follow these instructions:	<a href="#">Appendix 1, “Upgrading from or Reinstalling a Previous Version If Using SAS Deployment Manager,”</a> on page 169
4	Deploy the SAS Embedded Process parcel (Cloudera) or stack (Hortonworks, IBM BigInsights, or Pivotal HD) to the cluster.	For more information, see <a href="#">“Using the SAS Deployment Manager to Deploy the SAS Embedded Process Parcel or Stack to the Cluster”</a> on page 16.
5	Deploy the parcel (Cloudera) or stack (Hortonworks, IBM BigInsights, or Pivotal HD) to all the nodes on the cluster.	For more information see <a href="#">“Deploying the SAS Embedded Process Parcel on Cloudera”</a> on page 25 or <a href="#">“Deploying the SAS Embedded Process Stack on Hortonworks, IBM BigInsights, or Pivotal HD”</a> on page 27.

Step	Description	Where to Go for Information
6	Review any additional configuration that might be needed depending on your Hadoop distribution.	For more information, see <a href="#">Chapter 5, “Additional Configuration for the SAS Embedded Process,”</a> on page 45.

---

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

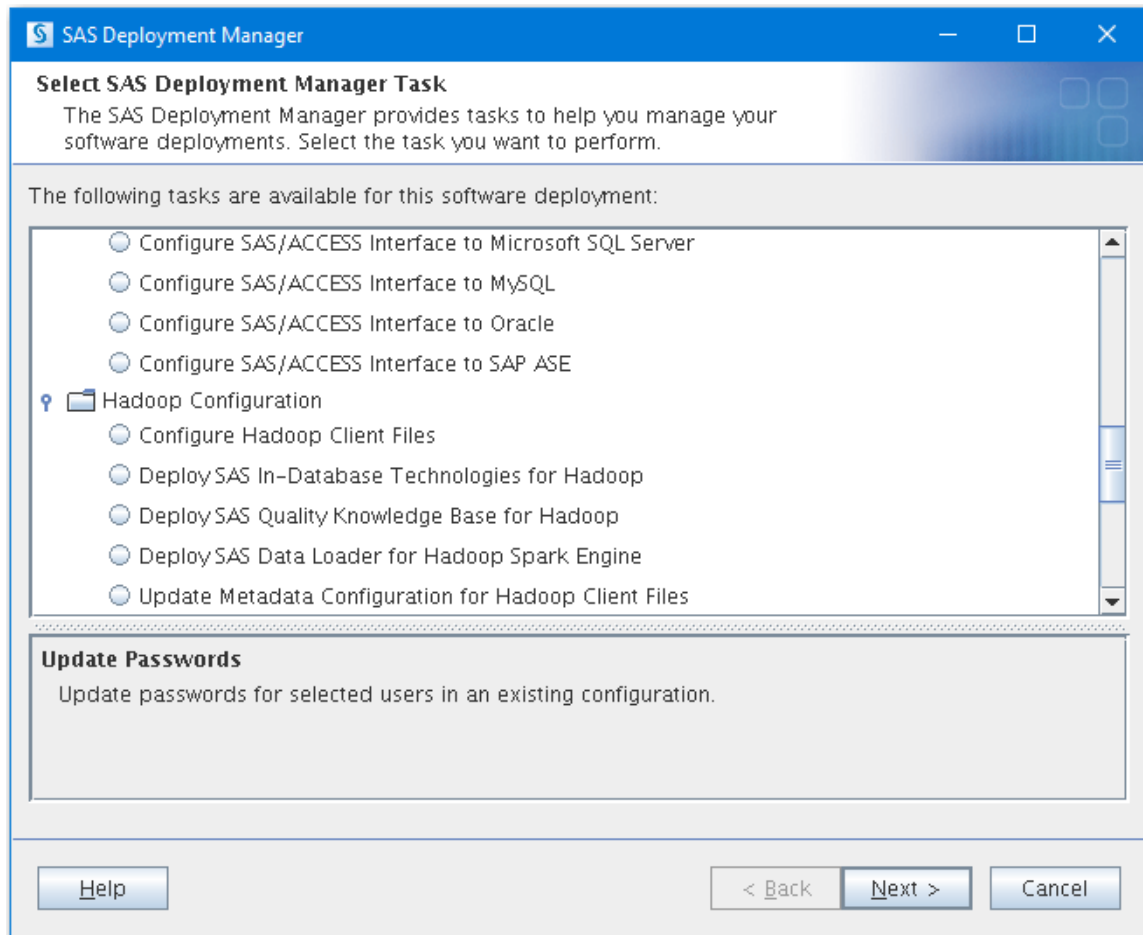
*Note:* For more information about the SAS Deployment Manager pages, click **Help** on each page.

1. Start the SAS Deployment Manager.

```
cd /SASHOME/SASDeploymentManager/9.4  
./sasdm.sh
```

The **Choose Language** page opens.

2. Select the language in which you want to perform the configuration of your software.  
Click **OK**. The **Select SAS Deployment Manager Task** page opens.

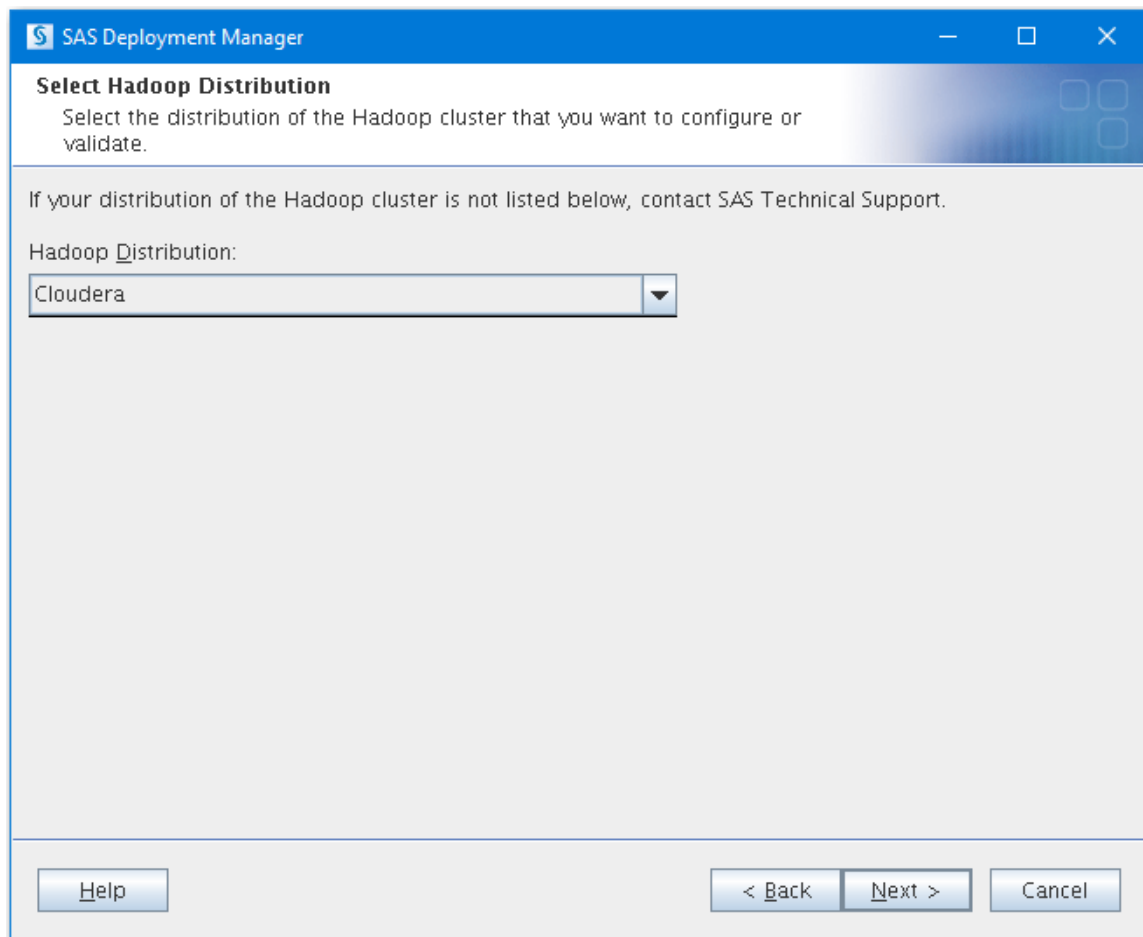


3. Under Hadoop Configuration, select **Deploy SAS In-Database Technologies for Hadoop**.

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

*Note:* If you have licensed and downloaded SAS Data Loader for Hadoop, the SAS Data Quality Accelerator component is silently deployed at the same time as the SAS Embedded Process for Hadoop.

*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.



The screenshot shows a window titled "SAS Deployment Manager" with a sub-header "Hadoop Cluster Manager Information". Below the sub-header is the instruction "Specify the host name and port number of your Hadoop cluster manager." There are two input fields: "Host Name:" which is empty, and "Port Number:" which contains the text "7180". At the bottom of the window are three buttons: "Help", "< Back", and "Next >", followed by a "Cancel" button.

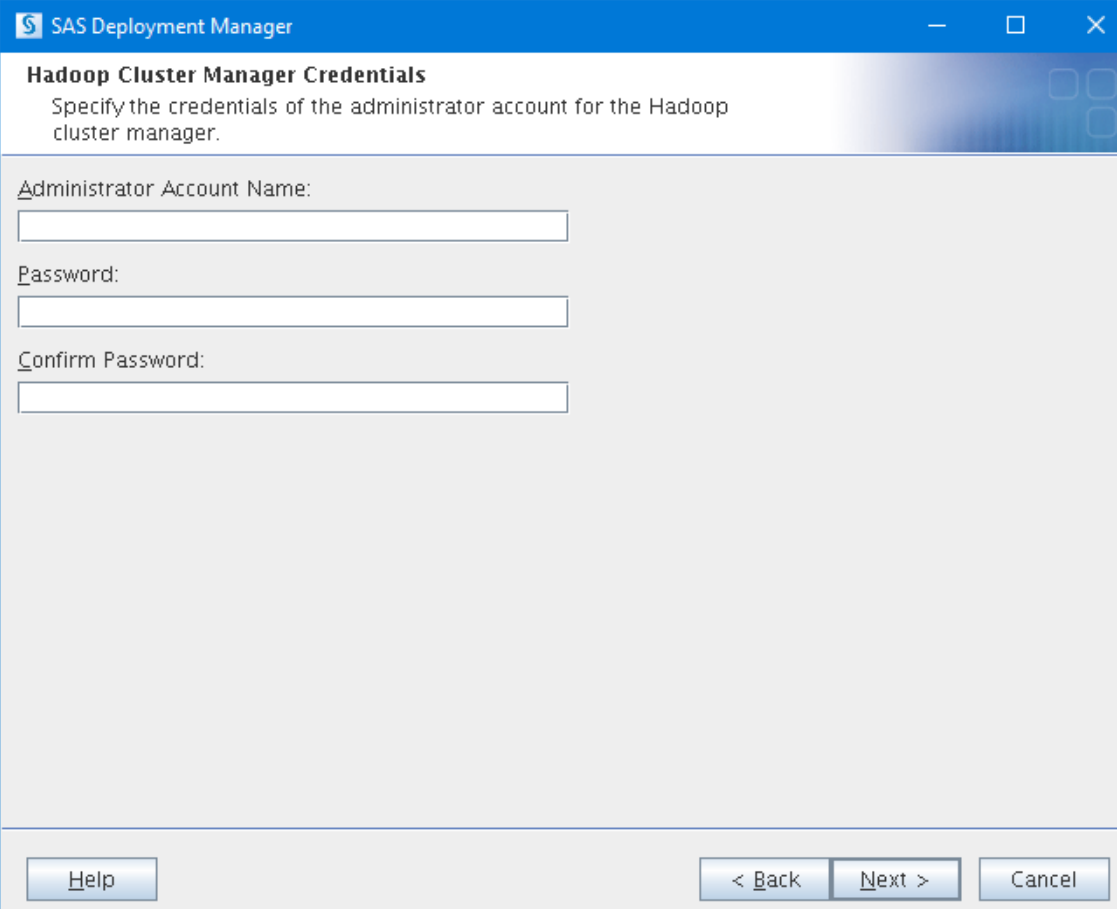
5. Enter the host name and port number for your Hadoop cluster.

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

*Note:* The host name must be a fully qualified domain name. The port number must be valid, and the cluster manager must be listening.

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



The screenshot shows a window titled "SAS Deployment Manager" with a blue header bar. Below the header, the title "Hadoop Cluster Manager Credentials" is displayed in bold. Underneath the title, a subtitle reads: "Specify the credentials of the administrator account for the Hadoop cluster manager." The main area of the dialog contains three text input fields, each preceded by a label: "Administrator Account Name:", "Password:", and "Confirm Password:". At the bottom of the dialog, there are four buttons: "Help" on the left, and "< Back", "Next >", and "Cancel" on the right.

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 **UNIX User Account with SSH for the Hadoop Cluster Manager Host** page opens.

**SAS Deployment Manager**

**UNIX User Account with SSH for the Hadoop Cluster Manager Host**  
Specify the credentials of the UNIX user account with SSH for the Hadoop cluster manager host.

UNIX User Account with SSH:

Password:

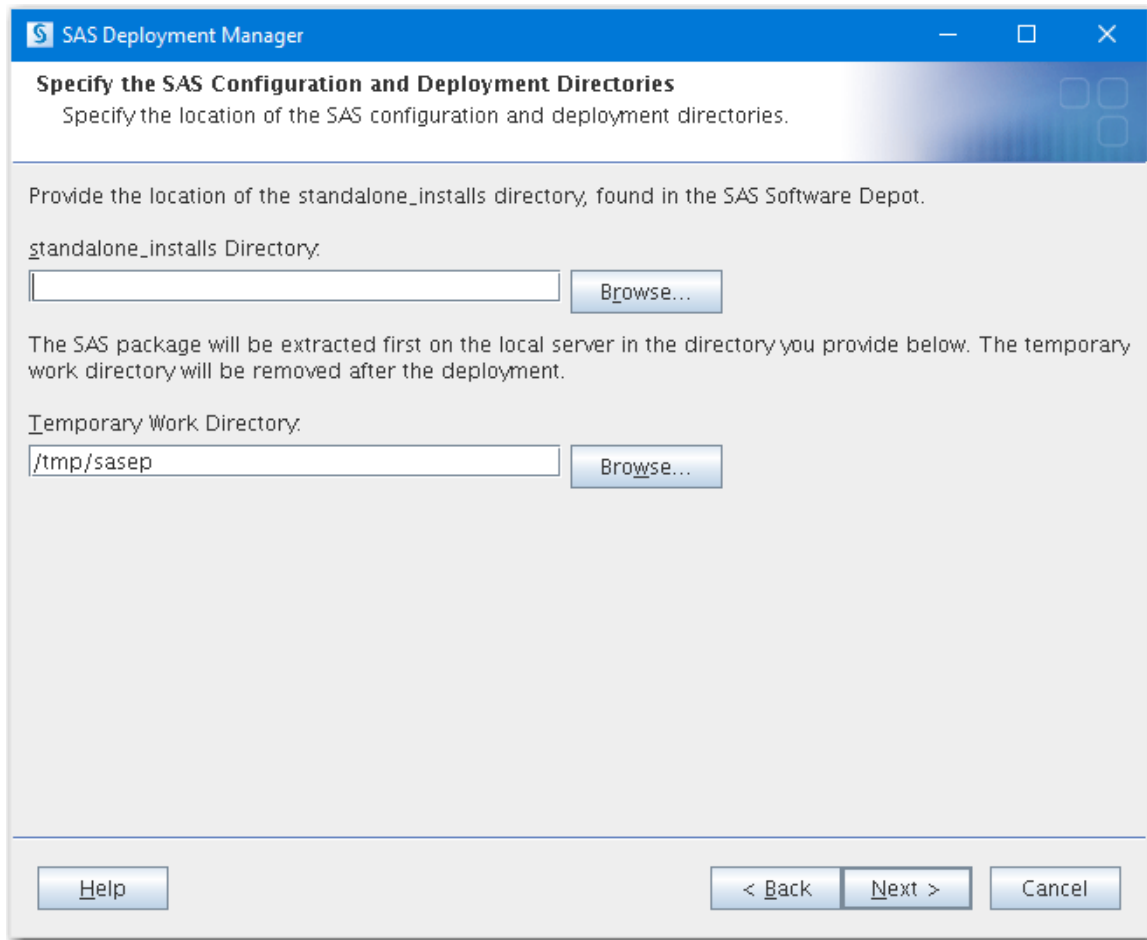
Confirm Password:

[Help](#) [< Back](#) [Next >](#) [Cancel](#)

7. Enter the root SSH account that has access to the cluster manager 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.

Click **Next**. The **Specify the SAS Configuration and Deployment Directories** page opens.



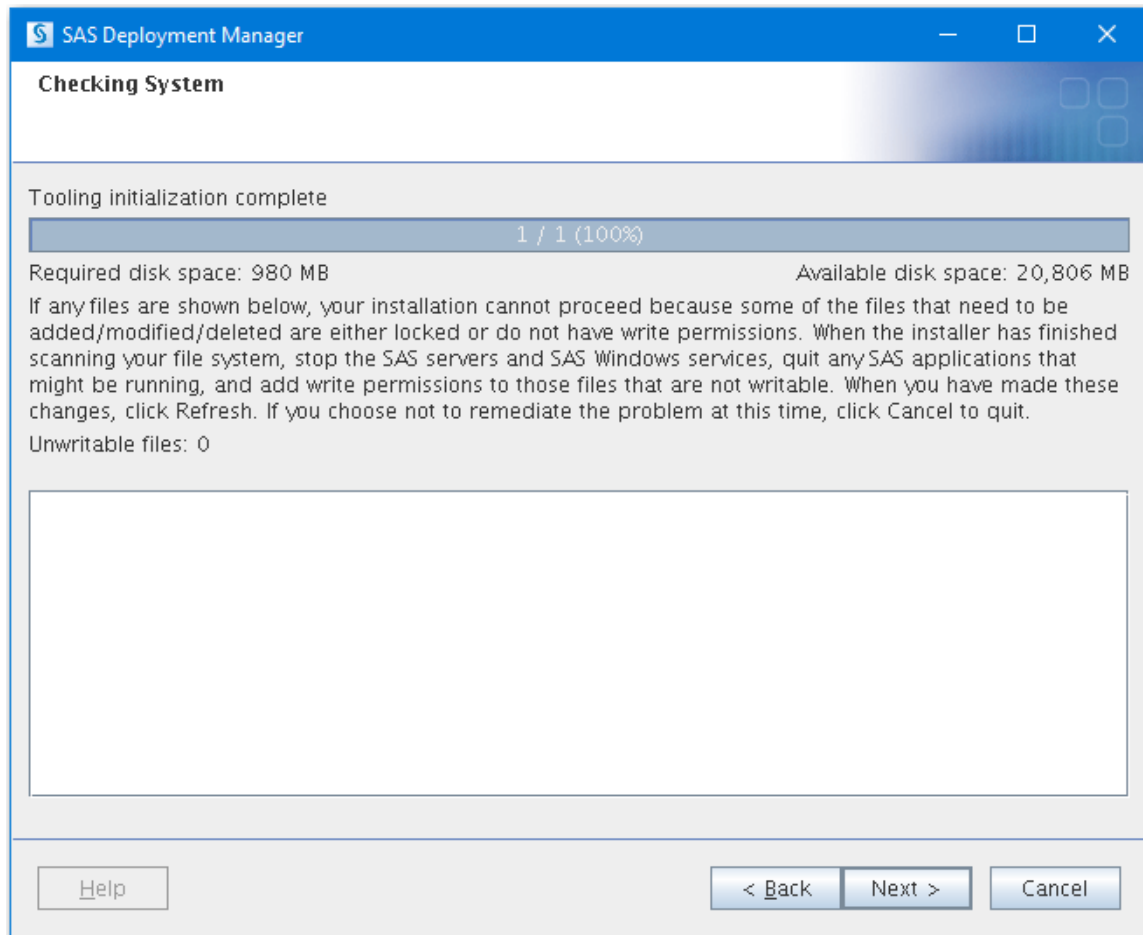
8. Enter the location of the SAS configuration and deployment directories:
  - a. Enter (or navigate to) the location of the `/standalone_installs` directory. This directory was created when your SAS Software Depot was created by the SAS Download Manager.

**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.

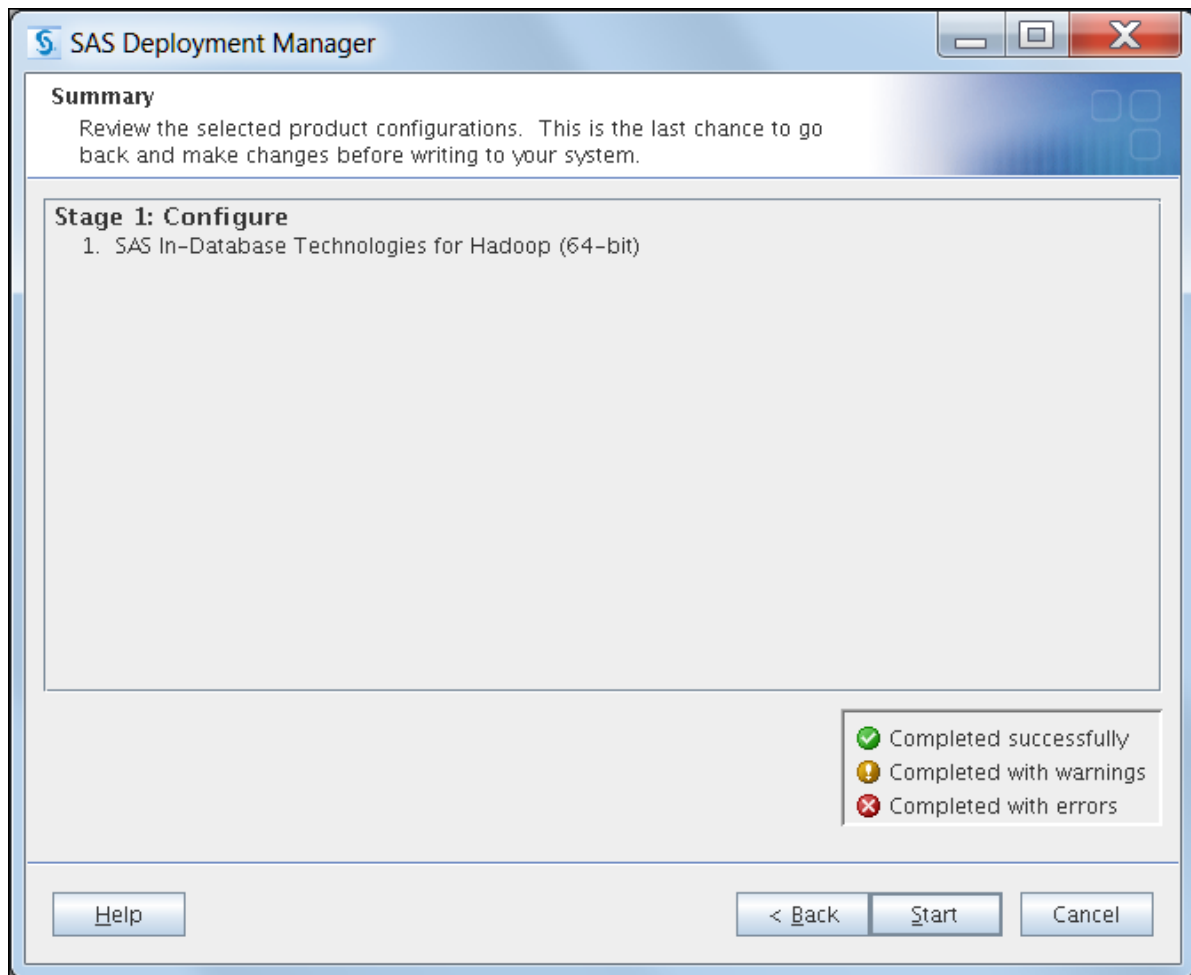
- b. Enter (or navigate to) a working directory on the local server where the package or stack is placed. The working directory is removed when the deployment is complete.

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



9. 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.



10. 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.

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.

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

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

12. Continue the installation process.

For more information, see [“Deploying the SAS Embedded Process Parcel on Cloudera” on page 25](#) or [“Deploying the SAS Embedded Process Stack on Hortonworks, IBM BigInsights, or Pivotal HD” on page 27](#).

---

## 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:

### 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:* 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.
  - b. On the row for the **SASEP** parcel, click **Distribute** to copy the parcel to all nodes and create the SASEPHome directory.  
 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 a 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.

Choose any 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**.

*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. You must stop the SAS Embedded Process service immediately when the task that adds the SAS Embedded Process is finished. 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 45](#).

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:

**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 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 28](#).

*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 3.
  - 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.

*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 45](#).

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 45](#).

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

---

<b>When to Deploy the SAS In-Database Deployment Package Manually . . . . .</b>	<b>31</b>
<b>Hadoop Manual Installation and Configuration Steps . . . . .</b>	<b>32</b>
<b>Unzipping and Copying the In-Database Deployment Package to the Hadoop Cluster . . . . .</b>	<b>33</b>
Unzipping the In-Database Deployment Package for Hadoop . . . . .	33
Creating the SAS Embedded Process Directory . . . . .	34
Copying the SAS Embedded Process Install Script to the Hadoop Cluster . . . . .	34
<b>Installing the SAS Embedded Process . . . . .</b>	<b>34</b>
<b>SASEP-ADMIN.SH Script . . . . .</b>	<b>38</b>
Overview of the SASEP-ADMIN.SH Script . . . . .	38
SASEP-ADMIN.SH Syntax . . . . .	39

---

## 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 `sepcorehdp-12.00000-1.sh` file. Now you want to install SAS Viya and the in-database deployment package that shipped with SAS Viya, which is `sepcorehdp-11.50000-1.sh`. In this instance, you would not install the in-database deployment package that shipped with SAS Viya because it is an earlier version; that is, 11.50000 versus 12.00000.

Step	Description	Where to Go for Information
1	Review these topics:	<ul style="list-style-type: none"> <li>• “Prerequisites for Installing the In-Database Deployment Package for Hadoop” on page 9</li> <li>• “Backward Compatibility” on page 10</li> <li>• “When to Deploy the SAS In-Database Deployment Package Manually” on page 31</li> </ul>
2	<p>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.</p> <p>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.</p>	<p>“Configuring SAS/ACCESS for Hadoop” in <a href="#">SAS Hadoop Configuration Guide for Base SAS and SAS/ACCESS</a></p> <p>For more information on obtaining the JAR and configuration files, see the following documentation depending on your SAS software:</p> <ul style="list-style-type: none"> <li>• <i>SAS Hadoop Configuration Guide for Base SAS and SAS/ACCESS</i></li> <li>• <i>SAS Data Loader for Hadoop: Installation and Configuration Guide</i></li> <li>• <i>SAS Contextual Analysis In-Database Scoring for Hadoop: Administrator's Guide</i></li> </ul>

Step	Description	Where to Go for Information
3	If you are upgrading from or reinstalling a previous release, follow these instructions:	<a href="#">Appendix 2, “Upgrading from or Reinstalling a Previous Version If Using Manual Deployment,” on page 175</a>
4	Unzip the in-database deployment package and copy the SAS Embedded Process install script (sepcorehadp) to the Hadoop master node (the NameNode).	<a href="#">“Unzipping and Copying the In-Database Deployment Package to the Hadoop Cluster” on page 33</a>
5	Install the SAS Embedded Process.	<a href="#">“Installing the SAS Embedded Process” on page 34</a>
6	Review any additional configuration that might be needed depending on your Hadoop distribution.	<a href="#">Chapter 5, “Additional Configuration for the SAS Embedded Process,” on page 45</a>
8 (Optional)	If you are installing other SAS software that uses the SAS Embedded Process, you must perform additional steps after you install the SAS Embedded Process.	<a href="#">SAS Contextual Analysis In-Database Scoring for Hadoop: Administrator’s Guide</a> <a href="#">SAS Data Loader for Hadoop: Installation and Configuration Guide</a> <a href="#">SAS High-Performance Analytics Infrastructure: Installation and Configuration Guide</a>

## 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-12.00000-1.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/12\_0/Hadoop\_on\_Linux\_x64/**.

The sepcorehadp-12.00000-1.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-12.00000-1.sh` file is in the **sasexe** directory.

```
EPZipDir/sasexe/sepcorehadp-12.00000-1.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/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.

## 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 using SSH.

```
ssh username@serverhostname
```

2. Copy the `sepcorehadp-12.00000-1.sh` file to your *EPInstallDir* directory. This example uses secure copy.

```
scp sepcorehadp-12.00000-1.sh username@hdpclus1: /EPInstallDir
```

*Note:* The location where you transfer the `sepcorehadp-12.00000-1.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 `sepcorehadp-12.00000-1.sh` file.

```
cd /EPInstallDir
```

For more information, see [Step 2](#) in “Unzipping the In-Database Deployment Package for Hadoop” on page 33.

2. Ensure that both the `EPInstallDir` folder and the `sepcorehadp-12.00000-1.sh` file have Read, Write, and Execute permissions (`chmod 755 -R`).
3. Use the following command to unpack the `sepcorehadp-12.00000-1.sh` file.

```
./sepcorehadp-12.00000-1.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/sepcorehadp-12.00000-1.sh
```

*Note:* During the install process, the `sepcorehadp-12.00000-1.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/utilities
```

The `EPInstallDir/SASEPHome/jars` directory contains the SAS Hadoop Embedded Process JAR file.

```
EPInstallDir/sasexe/SASEPHome/jars/sas.hadp2.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
06/20/16 09:51:26 06/27/16 09:51:26  krbtgt/HOST.COMPANY.COM@HOST.COMPANY.COM
        renew until 06/22/16 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 39.

- 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 `sepcorehdp-12.00000-1.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 10.

*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>
```

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 > <-maxscp 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> <-maxscp 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 43](#)

**-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 43](#)

**-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 the QKB” and “Removing the SAS Data Management Accelerator for Spark” 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 43](#)

---

**-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 the second maintenance release of SAS 9.4).

**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 43](#)

---

**-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 the second maintenance release of SAS 9.4).

**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 43](#)

**-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 43](#)



**-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.

<b>Restriction</b>	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 the second maintenance release of SAS 9.4).
<b>Requirement</b>	If you use this argument, you must restart the MapReduce service, the YARN service, or both after the SAS Embedded Process is installed.
<b>Interactions</b>	Use this argument in conjunction with the <b>-add</b> argument to force the creation of the symbolic links.  Use the <b>-linklib</b> argument after the SAS Embedded Process is already installed to create the symbolic links.
<b>See</b>	<a href="#">“Backward Compatibility” on page 10</a> <a href="#">“-linklib” on page 41</a>

**-epconfig *config-filename***

generates the SAS Embedded Process configuration file in the specified location.

<b>Default</b>	If the <b>-epconfig</b> argument is not specified, the install script creates the SAS Embedded Process configuration file in the default location <b>/sas/ep/config/ep-config.xml</b> .
<b>Requirement</b>	If the <b>-epconfig</b> argument is not specified a configuration file location must be provided. If you choose a non-default location, you must set the <b>sas.ep.config.file</b> property in the <b>mapred-site.xml</b> file that is on your client machine to the non-default location.
<b>Interaction</b>	Use the <b>-epconfig</b> argument in conjunction with the <b>-add</b> or <b>-remove</b> 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.

<b>Default</b>	10
<b>Interaction</b>	Use this argument in conjunction with the <b>-add</b> or <b>-hotfix</b> 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.

<b>Requirement</b>	The <b>-hostfile</b> or <b>-host</b> argument is required if you do not have sudo access.
<b>Interaction</b>	Use the <b>-hostfile</b> argument in conjunction with the <b>-add</b> , <b>-hotfix</b> , <b>-check</b> , or <b>-remove</b> arguments.
<b>See</b>	<a href="#">“-hdfsuser <i>user-id</i>” on page 44</a>
<b>Example</b>	<b>-hostfile</b>

**-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 44](#)

**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

---

<b>Overview of Additional Configuration Tasks</b> . . . . .	<b>45</b>
<b>Additional Configuration Needed to Use HCatalog File Formats</b> . . . . .	<b>46</b>
Overview of HCatalog File Types . . . . .	46
Prerequisites for HCatalog Support . . . . .	46
SAS Client Configuration . . . . .	46
SAS Server-Side Configuration . . . . .	47
Additional Configuration for Hadoop 2.6 and SAS Windows Middle Tiers and SAS Windows Servers . . . . .	48
<b>Adding the YARN Application CLASSPATH to the Configuration File for MapR Distributions</b> . . . . .	<b>48</b>
<b>Changing the Trace Level</b> . . . . .	<b>49</b>
<b>Adjusting the SAS Embedded Process Performance</b> . . . . .	<b>49</b>

---

## 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 46.](#)
- [“Adding the YARN Application CLASSPATH to the Configuration File for MapR Distributions” on page 48.](#)
- [“Changing the Trace Level” on page 49.](#)
- [“Adjusting the SAS Embedded Process Performance” on page 49.](#)

## 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.

File Type	Required Hive Version
Avro	0.14
ORC	0.11
Parquet	0.13
RCFile	0.6

### 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.

```
<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/*,
  $HADOOP_MAPRED_HOME/*,$HADOOP_MAPRED_HOME/lib/*,$MR2_CLASSPATH</value>
</property>
```

Here is an example for a Hortonworks distribution.

```
<property>
  <name>mapreduce.application.classpath</name>
  <value>/EPInstallDir/SASEPHome/jars/sas.hadoop.ep.apache205.jar,/SASEPHome/
  jars/sas.hadoop.ep.apache205.nls.jar,/usr/lib/hive-hcatalog/libexec/
  ../share/hcatalog/*,/usr/lib/hive-hcatalog/libexec/./share/hcatalog/
  storage-handlers/hbase/lib/*,/usr/lib/hive/lib/*,$HADOOP_MAPRED_HOME/
  share/hadoop/mapreduce/*,$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/
```

```
lib/*</value>
</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:

```
$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:

```
%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`).

```
<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.

```
<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.threadss`. The default number of buffers is 3.

```
<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`.

```
<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.

```
<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.

```
<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.

## Part 3

---

# Administrator's Guide for Teradata

<i>Chapter 6</i>	
<b><i>In-Database Deployment Package for Teradata</i></b> .....	<b>55</b>
<i>Chapter 7</i>	
<b><i>Deploying the In-Database Deployment Package: Teradata</i></b> .....	<b>59</b>
<i>Chapter 8</i>	
<b><i>SAS Data Quality Accelerator for Teradata</i></b> .....	<b>67</b>



## Chapter 6

# In-Database Deployment Package for Teradata

---

Prerequisites .....	55
Overview of the In-Database Deployment Package for Teradata .....	55
Teradata Permissions for Publishing Formats and Scoring Models .....	57
Documentation for Using In-Database Processing in Teradata .....	57

---

## 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 67](#).

- 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. In the fourth maintenance release of SAS 9.4 (November 2016), if you license SAS In-Database Technologies for Teradata, the SAS Data Quality Accelerator for Teradata scripts are also deployed with 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 65](#).

---

## 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 161](#).

---

## Documentation for Using In-Database Processing in Teradata

- [SAS In-Database Products: User's Guide](#)
- [SAS DS2 Language Reference](#)
- [SAS Data Quality Accelerator for Teradata: User's Guide](#)





## Chapter 7

# Deploying the In-Database Deployment Package: Teradata

---

<b>Teradata Installation and Configuration Steps</b> .....	<b>59</b>
<b>Upgrading from a Previous Version</b> .....	<b>60</b>
Upgrading from a Version That Was Installed before the Third Maintenance Release of SAS 9.4 (July 2015) .....	60
Upgrading Versions That Were Installed after the Third Maintenance Release of SAS 9.4 (July 2015) .....	61
<b>Installing the SAS Formats Library and the SAS Embedded Process</b> .....	<b>63</b>
Moving the SAS Formats Library and the SAS Embedded Process Packages to the Server Machine .....	63
Installing the SAS Formats Library and the SAS Embedded Process with the Teradata Parallel Upgrade Tool .....	64
Installing the SAS Embedded Process Support Functions .....	65
<b>Controlling the SAS Embedded Process</b> .....	<b>65</b>

---

## 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 60.
2. Install the in-database deployment package.  
For more information, see [“Installing the SAS Formats Library and the SAS Embedded Process”](#) on page 63.
3. Install the SAS Embedded Process support functions.  
For more information, see [“Installing the SAS Embedded Process Support Functions”](#) on page 65.

*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 161.
- 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 67.

- 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 before the Third Maintenance Release of SAS 9.4 (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 -q -a" | 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 -q -a" | 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 63](#).

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 63](#).

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
- sepcoretera-9.43000-1 replacing tkindbsrv-9.42\_M1-2

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

```
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.

*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 65](#).

- b. Confirm that the database is shut down.

```
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 75](#).

- 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:

```
psh "rpm -e package-name"
```

*package-name* is either *jazxfbrs.9.version*, *accelterafmt-version*, or *tkindbsrv-version*.

For example, to remove **jazxfbrs**, run the command **psh "rpm -e jazxfbrs-9.2-1.6"**.

- To remove the package from each node, run this command on each node:

```
rpm -e package-name
```

*package-name* is either *jazxfbrs.9.version*, *accelterafmt-version*, or *tkindbsrv-version*.

4. (Optional) To confirm removal of the package before installing the new package, run this command:

```
psh "rpm -q package-name"
```

*package-name* is either *jazxfbrs.9.version*, *accelterafmt-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 63](#).

### **Upgrading Versions That Were Installed after the Third Maintenance Release of SAS 9.4 (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 -q -a" | 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 63](#).

2. Run this command to check the current installed version of the SAS Embedded Process.

```
psh "rpm -e sepcoretera*"
```

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

```
sepcoretera-9.43000-1.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 63](#).

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 75](#).
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 jazxfbrs-9.2-1.6
- sepcoretera-12.0000-1 replacing sepcoretera-9.43000-1.x86\_64

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

```
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.

*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 65](#).

- b. Confirm that the database is shut down.

```
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* or *sepcoretera-version*.

For example, to remove **sepcoretera**, run the command **psh "rpm -e sepcoretera-9.43000-1"**.

- To remove the package from each node, run this command on each node:

```
rpm -e package-name
```

*package-name* is either *accelerafmt-version* or *sepcoretera-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 *accelerafmt-version* or *sepcoretera-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 63.

---

## 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, *accelerafmt-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 *accelerafmt-3.1-n.x86\_64.rpm* file is located in the ***SAS-installation-directory/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 accelerafmt-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-12.00000-1.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/12\_0/Teradata\_on\_Linux/*** directory.

The *sepcoretera-12.00000-1.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-12.00000-1.x86\_64.rpm should be in the **/SomeTempDir/sasexe** directory.

- 4. Copy the sepcoretera-12.00000-1.x86\_64.rpm file to the same location on the server as the accelerafmt-3.1-n.x86\_64.rpm file in Step 2.

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 sepcoretera-12.00000-1.x86_64.rpm root@teramach1:/sasdir/18NOV16
```

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>. 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 one of the Teradata nodes. For more information, see “[Moving the SAS Formats Library and the SAS Embedded Process Packages to the Server Machine](#)” on page 63.
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, `accelterfmt-3.1-n` or `sepcoretera-12.00000-1.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 accelterafmt
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.

Action performed	Command (by Teradata version)
Provides the status of the SAS Embedded Process.	<code>CALL SQLJ.SERVERCONTROL ('SAS', 'status', :A); *</code>
Shuts down the SAS Embedded Process.	<code>CALL SQLJ.SERVERCONTROL ('SAS', 'shutdown', :A); *</code>
<i>Note:</i> You cannot shut down until all queries are complete.	

---

Action performed	Command (by Teradata version)
Stops new queries from being started. Queries that are currently running continue to run until they are complete.	<code>CALL SQLJ.SERVERCONTROL ('SAS', 'disable', :A); *</code>
Enables new queries to start running.	<code>CALL SQLJ.SERVERCONTROL ('SAS', 'enable', :A); *</code>

\* Note that the Language 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);`
3. Shutdown the SAS Embedded Process.  
`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

---

<b>Introduction</b> .....	<b>67</b>
<b>Installing SAS Data Quality Accelerator Stored Procedures in the Teradata Database</b> .....	<b>68</b>
Overview .....	68
Using the dq_install.sh Script .....	68
Using the dq_grant.sh Script .....	69
<b>Obtaining and Deploying a QKB</b> .....	<b>69</b>
Obtaining a QKB .....	69
Packaging the QKB .....	70
Installing the QKB Package File with the Teradata Parallel Upgrade Tool .....	71
<b>Validating the Accelerator Installation</b> .....	<b>72</b>
<b>Troubleshooting the Accelerator Installation</b> .....	<b>73</b>
<b>Updating and Customizing a QKB</b> .....	<b>74</b>
<b>Removing SAS Data Quality Accelerator from the Teradata Database</b> .....	<b>75</b>
Overview .....	75
Using the dq_uninstall.sh Script .....	75

---

## 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/12.00000/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 reinstalling 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/12.00000/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/12.00000/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:

```
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 SAS 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\SASQualityKnowledgeBase\product-identifier\product-version (for example, C:\ProgramData\SAS\SASQualityKnowledgeBase\CI\27 or C:\ProgramData\SAS\SASQualityKnowledgeBase\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\27 c:\temp\
```

The default location for the qkb-dir argument is: C:\ProgramData\SAS\SASQualityKnowledgeBase\CI\27.

The package file that is created in C:\temp\ will have a name in the following form:

```
sasqkb_product-version-timestamp.noarch.rpm
```

**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 27:

```
sasqkb_ci-27.0-1474057340608.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 71 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>. 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 65 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 `Dqacctest` 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 `Dqacctest` from your database.

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

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

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

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

call sas_sysfnlib.dq_gender('Name', 'mydb.dqacctest', 'name', 'id_num',
    'mydb.output_table_2', 'locale');

select gender from mydb.output_table_2;
drop table mydb.dqacctest;
```

If the request was successful, the SELECT statement produces an output table that contains this:

```
Gender
-----
M
F
```

---

## 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:

```
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.):

```
TableName
-----
dq_set_qkb
dq_match_parsed
dqi_drop_view_if_exists
dqi_get_option_default
dq_debug
dq_propercase
dqi_tbl_dbname
dqi_drop_tbl_if_exists
dq_set_option
dqt_error
dq_standardize
dq_standardize_parsed
dq_debug2
dqi_invoke_table
dq_lowercase
dq_set_locale
dq_extract
dq_uppercase
dq_list_bindings
dqi_replace_tags
dq_list_defns
dqi_call_ep
dqi_get_bool_option
dqi_gen_toktxt
dqt_codegen
dq_match
```

```

dq_parse
dqt_trace
dq_pattern
dqi_clear_tok_tbls
dqt_tokname_tmp
dq_format
dq_list_locales
dqi_invoke_scalar
dqi_invoke_prepared
dq_bind_token
dq_gender

```

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 SAS 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 70 and “[Installing the QKB Package File with the Teradata Parallel Upgrade Tool](#)” on page 71. 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 75](#).

If you are upgrading from SAS Data Quality Accelerator 2.7 for Teradata, you will need to execute the following commands after running the `dq_uninstall.sh` script. The first command locates the SAS Data Quality Accelerator package `sepdqaccta` on the Teradata database server. Specify the output of the first command as the *package-name* in the second command.

```
psh "rpm -q -a" | grep sepdqaccta
psh "rpm -e package-name"
```

The `psh` command executes the commands in parallel on all Teradata database nodes.

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 74](#) and [“Obtaining a QKB” on page 69](#).

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 65](#) 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/12.00000/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

<i>Chapter 9</i>	
<b>Administrator's Guide for Aster</b> .....	79
<i>Chapter 10</i>	
<b>Administrator's Guide for DB2</b> .....	85
<i>Chapter 11</i>	
<b>Administrator's Guide for Greenplum</b> .....	107
<i>Chapter 12</i>	
<b>Administrator's Guide for Netezza</b> .....	125
<i>Chapter 13</i>	
<b>Administrator's Guide for Oracle</b> .....	141
<i>Chapter 14</i>	
<b>Administrator's Guide for SAP HANA</b> .....	147
<i>Chapter 15</i>	
<b>Administrator's Guide for SPD Server</b> .....	157



## Chapter 9

# Administrator's Guide for Aster

---

<b>In-Database Deployment Package for Aster</b> .....	<b>79</b>
Prerequisites .....	79
Overview of the In-Database Deployment Package for Aster .....	79
<b>Aster Installation and Configuration</b> .....	<b>80</b>
Aster Installation and Configuration Steps .....	80
Upgrading from a Previous Version .....	80
Installing the In-Database Deployment Package Binary Files for Aster .....	80
<b>Validating the Publishing of the SAS_SCORE() and the SAS_PUT() Functions</b> .	<b>83</b>
<b>Aster Permissions</b> .....	<b>83</b>
<b>Documentation for Using In-Database Processing in Aster</b> .....	<b>84</b>

---

## 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 80](#) 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 80](#).

### ***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 \
  echo $ip; \
  ssh $ip "rm -r $location/SAS"; \
  ssh $ip "rm -r $location/SASEPHome"; \
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-12.00000-1.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/12\_0/Aster\_on\_Linux\_x64** directory.

The **sepcoreastr-12.00000-1.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-12.00000-1.sh** 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-12.00000-1.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-12.00000-1.sh**, execute **sepcoreastr-12.00000-1.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-12.00000-1.sh** 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-12.00000-1.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-12.00000-1.sh
```

7. Unpack the self-extracting archive file in the partner directory.

```
./sepcoreastr-12.00000-1.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-12.00000-1.sh";
done
```

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

```
/home/beehive/bin/utils/primitives/UpgradeNCluster.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.

```
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 `\dF` 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](http://support.sas.com/documentation/onlinedoc/indbtech/index.html), located at <http://support.sas.com/documentation/onlinedoc/indbtech/index.html>

## Chapter 10

# Administrator's Guide for DB2

---

<b>In-Database Deployment Package for DB2</b> .....	<b>85</b>
Prerequisites .....	85
Overview of the In-Database Deployment Package for DB2 .....	85
<b>Function Publishing Process in DB2</b> .....	<b>86</b>
<b>DB2 Installation and Configuration</b> .....	<b>87</b>
DB2 Installation and Configuration Steps .....	87
Upgrading from a Previous Version .....	87
Installing the SAS Formats Library and Binary Files to DB2 .....	90
Installing the SAS Embedded Process .....	92
Running the %INDB2_PUBLISH_COMPILEUDF Macro .....	97
Running the %INDB2_PUBLISH_DELETEUDF Macro .....	101
<b>Validating the Publishing of SAS_COMPILEUDF and SAS_DELETEUDF Functions and Global Variables</b> .....	<b>104</b>
<b>DB2 Permissions</b> .....	<b>104</b>
<b>Documentation for Using In-Database Processing in DB2</b> .....	<b>106</b>

---

## 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 87](#).
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 90](#).
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 97](#).
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 101](#).
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 161](#).

### **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 88](#).
- If you want to upgrade only the SAS Embedded Process, see [“Upgrading the SAS Embedded Process” on page 89](#).

### 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 89](#).

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 06 - 4:03 db2sasep
db2v10 27983990 16646196 1 08:24:09 pts/10 0:00 grep 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 96](#).

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 before the fourth maintenance release of SAS 9.4, 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 the fourth maintenance release of SAS 9.4, 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 88](#).

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 06 - 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
```

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

*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 96](#).

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 before the fourth maintenance release of SAS 9.4, 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 the fourth maintenance release of SAS 9.4, 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 that you reinstall 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 reinstall 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.

```
cd path_to_sh_file
```

**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.

```
chmod 755 -r acceldb2fmt-3.1-n_r64.sh
```

*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:

```
mv SAS to SAS_OLD /* renames the SAS directory */
rm -fr SAS /* removes the SAS directory */
```

5. Use the following commands to unpack the appropriate self-extracting archive file.

```
./sh_file
```

**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.

```
/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/libjazzfbrs.so
/path_to_sh_file/SAS/SASFormatsLibraryForDB2/3.1 ->3.1-n
```

6. Use the following command to place the files in the DB2 instance:

```
path_to_sh_file/SAS/SASFormatsLibraryForDB2/3.1-n/bin/
CopySASFiles.sh db2instancepath/sqllib
```

**db2instancepath/sqllib** is the path to the **sqllib** 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.

```
db2instancepath/sqllib/function/SAS/SAS_CompileUDF
db2instancepath/sqllib/function/SAS/SAS_DeleteUDF
db2instancepath/sqllib/function/SAS/libjazzfbrs.so
```

*Note:* If the `SAS_CompileUDF`, `SAS_DeleteUDF`, and `libjazzfbrs.so` files currently exist under the target directory, you must rename the existing files before you run

the CopySASFiles.sh script. Otherwise, the CopySASFiles.sh script does not work, and you get a "Text file is busy" message for each of the three files.

7. Use the DB2SET command to tell DB2 where to find the 64-bit formats library.

```
db2set DB2LIBPATH=db2instancepath/sqlllib/function/SAS
```

**db2instancepath/sqlllib** is the path to the **sqlllib** 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/sqlllib/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-12.00000-1.sh (AIX) or sepcoredb2l-12.00000-1.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>/12\_0/DB2on<AIX | Linux\_x64>** directory.

The sepcoredb2a-12.00000-1.sh (AIX) or sepcoredb2l-12.00000-1.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.

- AIX: sepcoredb2a-12.00000-1.sh
- Linux (x86\_64): sepcoredb2l-12.00000-1.sh

### Unpacking the SAS Embedded Process Files for DB2

After the sepcoredb2a-12.00000-1.sh (AIX) or the sepcoredb2l-12.00000-1.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-12.00000-1.sh (AIX) or the sepcoredb2l-12.00000-1.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 before the fourth maintenance release of SAS 9.4, 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 the fourth maintenance release of SAS 9.4 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-12.00000-1.sh or sepcoredb2l-12.00000-1.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 95](#).

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/sqlllib/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 06 - 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.

```
cd instance-home/sqllib/db2dump
ls -al sasep0.log
```

### **DB2SET Command Syntax for the SAS Embedded Process**

The syntax for the DB2SET command is shown below.

```
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.

<b>Note</b>	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.

<b>Default</b>	DIAGPATH dbm configuration value
----------------	----------------------------------

---

<b>Requirement</b>	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

<b>Default</b>	DIAGLEVEL dbm configuration value
----------------	-----------------------------------

---

<b>Range</b>	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 reinstall 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/sqllib/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 104](#).

### **%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 99](#).

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:

```
%let indconn = server=yourserver user=youruserid password=yourpwd
               database=yourdb schema=saslib;
```

For more information, see the [“INDCONN Macro Variable” on page 98](#).

3. Run the %INDB2\_PUBLISH\_COMPILEUDF macro. For more information, see [“%INDB2\\_PUBLISH\\_COMPILEUDF Macro Syntax” on page 99](#).

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 104](#).

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 101](#).

### **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 87.

---

**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.

<b>Default</b>	SASLIB
----------------	--------

<b>Restriction</b>	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.
--------------------	---

<b>Requirement</b>	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/sqllib
, COMPILER_PATH=compiler-path-directory
<, DATABASE=database-name>
<, ACTION=CREATE | REPLACE | DROP>
<, OBJNAME=object-file-name>
<, OUTDIR=diagnostic-output-directory>
);
```

**Arguments****DB2PATH=*db2instancepath/sqllib***

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 91](#).

**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 98](#).

**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.

<b>Default</b>	SAS_CompileUDF
----------------	----------------

<b>Interaction</b>	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 104](#).

### %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 98](#).

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 102](#).

3. Run the %INDB2\_PUBLISH\_DELETEUDF macro. For more information, see [“%INDB2\\_PUBLISH\\_DELETEUDF Macro Syntax” on page 103](#).

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 104](#).

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 87](#).

---

#### **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

**Restriction** 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.

**Requirement** 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 101](#).

**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.

Permission Needed	Authority Required to Grant Permission	Examples
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	System Administrator or Database Administrator  <i>Note:</i> If you have SYSADM or DBADM authority or are the DB2 instance owner, then you have these permissions. Otherwise, contact your database administrator to obtain these permissions.	GRANT CREATEIN ON SCHEMA SASLIB TO <i>compiledeletepublisheruserid</i>
CREATE_EXTERNAL_ROUTINE permission to the database in which the SAS_COMPILEUDF and SAS_DELETEUDF functions are published		GRANT CREATE_EXTERNAL_ROUTINE ON DATABASE TO <i>compiledeletepublisheruserid</i>

- 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.

Permission Needed	Authority Required to Grant Permission	Examples
EXECUTE permission for functions that have been published.  This enables the person who publishes the formats or scoring model functions to execute the SAS_COMPILEUDF and SAS_DELETEUDF functions.	System Administrator or Database Administrator  <i>Note:</i> If you have SYSADM or DBADM authority, then you have these permissions. Otherwise, contact your database administrator to obtain these permissions.	GRANT EXECUTE ON FUNCTION SASLIB.* TO <i>scoringorfmtpublisherid</i>
CREATE_EXTERNAL_ROUTINE permission to the database to create format or scoring model functions		GRANT CREATE_EXTERNAL_ROUTINE ON DATABASE TO <i>scoringorfmtpublisherid</i>
CREATE_NOT_FENCED_ROUTINE permission to create format or scoring model functions that are not fenced		GRANT CREATE_NOT_FENCED_ROUTINE ON DATABASE TO <i>scoringorfmtpublisherid</i>
CREATEIN permission for the schema in which the format or scoring model functions are published if the default schema (SASLIB) is not used		GRANT CREATEIN ON SCHEMA <i>scoringschema</i> TO <i>scoringorfmtpublisherid</i>
CREATE TABLE permission to create the model table used in with scoring and the SAS Embedded Process		GRANT CREATETAB TO <i>scoringpublisherSEPid</i>
READ permission to read the SASUDF_COMPILER_PATH and SASUDF_DB2PATH global variables  <i>Note:</i> The person who ran the %INDB2_PUBLISH_COMPILEUDF macro has these READ permissions and does not need to grant them to himself or herself again.	Person who ran the %INDB2_PUBLISH_COMPILEUDF macro  <i>Note:</i> For security reasons, only the user who created these variables has the permission to grant READ permission to other users. This is true even for the user with administrator permissions such as the DB2 instance owner.	GRANT READ ON VARIABLE SASLIB.SASUDF_DB2PATH TO <i>scoringorfmtpublisherid</i>  GRANT READ ON VARIABLE SASLIB.SASUDF_COMPILER_PATH TO <i>scoringorfmtpublisherid</i>

*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 161.

## Documentation for Using In-Database Processing in DB2

For information about how to publish SAS formats or scoring models, see the [SAS In-Database Products: User's Guide](#), located at <http://support.sas.com/documentation/onlinedoc/indbtech/index.html>.



## Chapter 11

# Administrator's Guide for Greenplum

---

<b>In-Database Deployment Package for Greenplum</b> . . . . .	<b>107</b>
Prerequisites . . . . .	107
Overview of the In-Database Deployment Package for Greenplum . . . . .	108
<b>Greenplum Installation and Configuration</b> . . . . .	<b>109</b>
Greenplum Installation and Configuration Steps . . . . .	109
Upgrading from a Previous Version . . . . .	109
Installing the SAS Formats Library and Binary Files . . . . .	111
Installing the SAS Embedded Process . . . . .	112
Running the %INDGP_PUBLISH_COMPILEUDF Macro . . . . .	114
Running the %INDGP_PUBLISH_COMPILEUDF_EP Macro . . . . .	118
<b>Validation of Publishing Functions</b> . . . . .	<b>121</b>
Validating the Publishing of the SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF Functions . . . . .	121
Validating the Publishing of the SAS_EP Function . . . . .	122
<b>Controlling the SAS Embedded Process</b> . . . . .	<b>122</b>
<b>Semaphore Requirements When Using the SAS Embedded Process for Greenplum</b> . . . . .	<b>123</b>
<b>Greenplum Permissions</b> . . . . .	<b>124</b>
<b>Documentation for Using In-Database Processing in Greenplum</b> . . . . .	<b>124</b>

---

## 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 109](#) 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 111](#).
3. Install the SAS Embedded Process.  
For more information, see [“Installing the SAS Embedded Process” on page 112](#).
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 114](#) or [“Running the %INDGP\\_PUBLISH\\_COMPILEUDF\\_EP Macro” on page 118](#).
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 161](#).

*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.

```
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**

**deleted.** 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 111.

### **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.

Before the fourth maintenance release of SAS 9.4 (November 2016), the UninstallSASEPFiles.sh file is in the `path_to_sh_file` directory where you copied the tkindsrv self-extracting archive file.

Starting with the fourth maintenance release of SAS 9.4 (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 libjazzfbrs.so and sas\_compileudf.so files.
5. In addition to deleting the libjazzfbrs.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 111.

## 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 ***SAS-installation-directory/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 reinstall 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:

```
./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/  
  libjazzfbrs.so
```

5. Use the following command to place the files in Greenplum:

```
./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 `libjazzfbrs.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:

```
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.

*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**, **libjazzfbrs.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-12.00000-1.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/12\_0/Greenplum\_on\_Linux\_x64/** directory.

The **sepcoregplm-12.00000-1.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

The SAS Embedded Process is contained in a self-extracting archive file named **sepcoregplm-12.00000-1.sh**.

To move and unpack the self-extracting archive file, follow these steps:

1. Using a method of your choice, transfer the **sepcoregplm-12.00000-1.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-12.00000-1.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. Change permissions on the file to enable you to execute the script.

```
chmod 755 -r sepcoregplm-12.00000-1.sh
```

5. Use the following command at the UNIX prompt to unpack the self-extracting archive file.

```
./sepcoregplm-12.00000-1.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 122](#).

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.

```
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 118](#).

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 117](#).

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 116](#).

3. Run the %INDGP\_PUBLISH\_COMPILEUDF macro.

For more information, see [“%INDGP\\_PUBLISH\\_COMPILEUDF Macro Syntax” on page 117](#).

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](#)

[SAS\\_COMPILEUDF, SAS\\_COPYUDF, SAS\\_DIRECTORYUDF, and SAS\\_DEHEXUDF Functions](#)” on page 121.

### **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 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=<'>userid<'>**

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<'>**

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=<'>database<'>**

specifies 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.

---

**PORT=<'>port-number<'>**

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 109](#) and [“Move and Install the SAS Formats Library and Binary Files” on page 111](#).

---

#### 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 114](#).

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 119](#).

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 119](#).

3. Run the %INDGP\_PUBLISH\_COMPILEUDF\_EP macro. For more information, see [“%INDGP\\_PUBLISH\\_COMPILEUDF\\_EP Macro Syntax” on page 120](#).

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 122](#).

### **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=<'>userid<'> PASSWORD=<'>password<'> DSN=<'>dsname <'>
<SCHEMA=<'>schema<'>> <PORT=<'>port-number<'>>
```

```
USER=<'>userid<'> PASSWORD=<'>password<'> SERVER=<'>server<'>
DATABASE=<'>database<'> <SCHEMA=<'>schema<'>>
<PORT=<'>port-number<'>>
```

**USER=<'>userid<'>**

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 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.

**DSN=<'>datasource<'>**

specifies the configured Greenplum ODBC data source to which you want to connect. If the DSN name contains spaces or nonalphanumeric 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<'>**

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=<'>database<'>**

specifies 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 <'>**

specifies 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<'>**

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.

## **%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**

specifies 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 InstallSASEPFiles.sh install script replaces existing versions of most files. However, you need to replace the existing SAS\_EP function after you run the InstallSASEPFiles.sh install script. For more information, see [“Upgrading from a Previous Version” on page 109](#) and [“Moving and Installing the SAS Embedded Process” on page 112](#).

**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.

```
psql -d databasename
```

You should receive the following prompt.

```
databasename=#
```

2. At the prompt, enter the following command.

```
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
```

### 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.

```
psql -d databasename
```

You should receive the following prompt.

```
databasename=#
```

2. At the prompt, enter the following command.

```
select proname, prosrc, probin, nspname FROM pg_catalog.pg_namespace n
JOIN pg_catalog.pg_proc p ON pronamespace = n.oid WHERE proname = 'sas_ep'
or proname = 'sas_ep_describe';
```

You should receive a result similar to the following:

proname   nspname	prosrc	probin
-----+-----+-----		
+-----		
sas_ep_describe	SAS_EP_Describe	\$libdir/SASEPHome/sasexe/sasep_tablefunc.so
<schema>		
sas_ep	SAS_EP	\$libdir/SASEPHome/sasexe/sasep_tablefunc.so
<schema>		

3. Exit psql.

```
\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 112](#).

```
/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
<-quiet>
```

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
<-quiet>
```

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 `\timing off` in your `.psqlrc` 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:

$$\text{maximum\_number\_semaphore\_arrays} = 5 * \text{number\_database\_segments}$$

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 \* 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 161.

---

## Documentation for Using In-Database Processing in Greenplum

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>.

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>.

## Chapter 12

# Administrator's Guide for Netezza

---

<b>In-Database Deployment Package for Netezza</b> .....	<b>125</b>
Prerequisites .....	125
Overview of the In-Database Deployment Package for Netezza .....	125
<b>Function Publishing Process in Netezza</b> .....	<b>126</b>
<b>Netezza Installation and Configuration</b> .....	<b>127</b>
Netezza Installation and Configuration Steps .....	127
Upgrading from a Previous Version .....	127
Installing the SAS Formats Library, Binary Files, and the SAS Embedded Process .....	128
Running the %INDNZ_PUBLISH_JAZLIB Macro .....	132
Running the %INDNZ_PUBLISH_COMPILEUDF Macro .....	134
<b>Netezza Permissions</b> .....	<b>138</b>
<b>Documentation for Using In-Database Processing in Netezza</b> .....	<b>139</b>

---

## 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 127](#).
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 128](#).
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 132](#).
4. Run the %INDNZ\_PUBLISH\_COMPILEUDF macro.  
For more information, see [“Running the %INDNZ\\_PUBLISH\\_COMPILEUDF Macro” on page 134](#).
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 161](#).

## 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 127](#).
- If you want to upgrade from a previous version of the SAS Embedded Process, see [“Upgrading the SAS Embedded Process” on page 128](#).

### 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 128](#).

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 132](#).

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 134](#).

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 128](#). Otherwise, continue the installation instructions in [“Installing the SAS Formats Library, Binary Files, and the SAS Embedded Process” on page 128](#).

### **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 127](#).

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.

4. Continue the installation instructions in [“Installing the SAS Formats Library, Binary Files, and the SAS Embedded Process” on page 128](#).

## **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 `accelnetzfnt-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 reinstall or upgrade, *n* is incremented by 1.

2. After the `accelnetzfnt-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/accelnetzfnt-3.1-n_lax.sh
path_to_sh_file/accelnetzfnt-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/libjzxfbrs_spu10.so
/nz/extensions/SAS/SASFormatsLibraryForNetezza/3.1-n/lib/libjzxfbrs_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-12.00000-1.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/12\_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/12\_0/Netezza\_TwinFin\_64bit\_host*** directory.

The `sepcorenetz-12.00000-1.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.

6. Using a method of your choice, transfer the *sepcorennetz-12.00000-1.sh* to any directory on the Netezza server. This directory will be referred to as ***path\_to\_sh\_file***
7. After the *sepcorennetz-12.00000-1.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.

9. Unpack the self-extracting archive file and create a Netezza cartridge file.

- a. Change to the directory where you put the *sepcorennetz-12.00000-1.sh* file.

```
cd path_to_sh_file
```

***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.

```
chmod 755 -r sepcorennetz-12.00000-1.sh
```

- c. Use the following command at the UNIX prompt to unpack the self-extracting archive file:

```
./sepcorennetz-12.00000-1.sh
```

After the script runs, the *sepcorennetz-12.00000-1.sh* file goes away, and the *SASTKInDatabaseServerForNetezza-12.0.0.0.nzc* Netezza cartridge file is created in its place.

10. Use the following **nzcm** commands to install and register the *sas\_ep* cartridge:

```
nzcm -i sas_ep
nzcm -r sas_ep
```

*Note:* The *sas\_ep* cartridge creates the NZRC database. The NZRC database contains remote controller functions that are required by the SAS Embedded Process. The *sas\_ep* cartridge is available on the Netezza website. For access to the *sas\_ep* cartridge, contact your local Netezza representative.

11. Use the following **nzcm** commands to install and register the SAS Embedded Process:

```
nzcm -i SASTKInDatabaseServerForNetezza-12.0.0.0.nzc
nzcm -r SASTKInDatabaseServerForNetezza
```

*Note:* The installation of the SAS Embedded Process is dependent on the *sas\_ep* cartridge that is supplied by Netezza.

For more NZCM commands, see “[NZCM Commands for the SAS Embedded Process](#)” on page 131.



## 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.

Command	Action performed
<code>nzcm -help</code>	Displays help for NZCM commands
<code>nzcm --installed</code> <code>nzcm -i</code>	Displays the filename (SASTKInDatabaseServerForNetezza) and the version number that is installed
<code>nzcm --registered</code> <code>nzcm -r</code>	Displays the filename (SASTKInDatabaseServerForNetezza) and the version number that is registered
<code>nzcm --unregister SASTKInDatabaseServerForNetezza</code> <code>nzcm -u SASTKInDatabaseServerForNetezza</code>	Unregisters the SAS Embedded Process
<code>nzcm --unregister sas_ep</code> <code>nzcm -u sas_ep</code>	Unregisters the sas_ep cartridge <i>Note:</i> 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 reinstalled. The sas_ep cartridge needs to be unregistered and uninstalled only when Netezza changes the cartridge version.
<code>nzcm -uninstall SASTKInDatabaseServerForNetezza</code> <code>nzcm -e SASTKInDatabaseServerForNetezza</code>	Uninstalls the SAS Embedded Process
<code>nzcm --uninstall sas_ep</code> <code>nzcm -e sas_ep</code>	Uninstalls the sas_ep cartridge <i>Note:</i> 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 reinstalled. The sas_ep cartridge needs to be unregistered and uninstalled only when Netezza changes the cartridge version.
<code>nzcm --install SSASTKInDatabaseServerForNetezza-12.0.0.0.nzc</code> <code>nzcm -i SSASTKInDatabaseServerForNetezza-12.0.0.0.nzc</code>	Installs the SAS Embedded Process
<code>nzcm --install sas_ep</code> <code>nzcm -i sas_ep</code>	Installs the sas_ep cartridge
<code>nzcm --register SASTKInDatabaseServerForNetezza</code> <code>nzcm -r SASTKInDatabaseServerForNetezza</code>	Registers the SAS Embedded Process
<code>nzcm --register sas_ep</code> <code>nzcm -register sas_ep</code>	Registers the sas_ep cartridge

## 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:

```
%let indconn=SERVER=yourservername USER=youruserid PW=yourpwd DB=database;
```

For more information, see the “[INDCONN Macro Variable](#)” on page 132.

2. Run the %INDNZ\_PUBLISH\_JAZLIB macro. For more information, see “[%INDNZ\\_PUBLISH\\_JAZLIB Macro Syntax](#)” on page 133.

### 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 133](#).

**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 126](#).

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 138](#).

### **%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 135](#).

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 135](#).

3. Run the %INDNZ\_PUBLISH\_COMPILEUDF macro. For more information, see [“%INDNZ\\_PUBLISH\\_COMPILEUDF Macro Syntax” on page 136](#).

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.

<b>Default</b>	SASLIB
----------------	--------

<b>Interactions</b>	<p>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 <a href="#">“%INDNZ_PUBLISH_COMPILEUDF Macro Syntax” on page 136</a>.</p>
---------------------	--

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 139](#).

**SCHEMA=<'>schema-name<'>**

specifies the name of the schema where the SAS\_COMPILEUDF, SAS\_DIRECTORYUDF, and SAS\_HEXTOTEXTUDF functions are published.

<b>Restriction</b>	This argument is supported only on Netezza v7.0.3 or later.
--------------------	---

<b>Interaction</b>	<p>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.</p>
--------------------	--

**%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 INDCONN macro variable. For more information, see [“INDCONN Macro Variable” on page 135](#).

### **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 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\_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.

Permission Needed	Authority Required to Grant Permission	Examples
CREATE LIBRARY permission to run the %INDNZ_PUBLISH_JAZLIB macro that publishes the SAS formats library ( <b>sas_jazlib</b> object)	System Administrator or Database Administrator  <i>Note:</i> If you have SYSADM or DBADM authority, then you have these permissions. Otherwise, contact your database administrator to obtain these permissions.	GRANT CREATE LIBRARY TO <i>fmtlibpublisherid</i>
CREATE FUNCTION permission to run the %INDNZ_PUBLISH_COMPILEUDF macro that publishes the SAS_COMPILEUDF, SAS_DIRECTORYUDF, and the SAS_HEXTOTEXTUDF functions		GRANT CREATE FUNCTION TO <i>compileudfpublisherid</i>

- 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.



Permission Needed	Authority Required to Grant Permission	Examples
EXECUTE permission for the SAS Formats Library	System Administrator or Database Administrator	GRANT EXECUTE ON SAS_JAZLIB TO <i>scoringorfmtpublisherid</i>
EXECUTE permission for the SAS_COMPILEUDF function	<i>Note:</i> If you have SYSADM or DBADM authority, then you have these permissions. Otherwise, contact your database administrator to obtain these permissions.	GRANT EXECUTE ON SAS_COMPILEUDF TO <i>scoringorfmtpublisherid</i>
EXECUTE permission for the SAS_DIRECTORYUDF function		GRANT EXECUTE ON SAS_DIRECTORYUDF TO <i>scoringorfmtpublisherid</i>
EXECUTE permission for the SAS_HEXTOTEXTUDF function		GRANT EXECUTE ON SAS_HEXTOTEXTUDF TO <i>scoringorfmtpublisherid</i>
CREATE FUNCTION, CREATE TABLE, CREATE TEMP TABLE, and CREATE EXTERNAL TABLE permissions to run the format and scoring publishing macros		GRANT CREATE FUNCTION TO <i>scoringorfmtpublisherid</i>
		GRANT CREATE TABLE TO <i>scoringorfmtpublisherid</i>
		GRANT CREATE TEMP TABLE TO <i>scoringorfmtpublisherid</i>
		GRANT CREATE EXTERNAL TABLE TO <i>scoringorfmtpublisherid</i>
		GRANT UNFENCED TO <i>scoringorfmtpublisherid</i>

- The third set of permissions is needed by the person who runs the SAS Embedded Process to create scoring files.

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.

```
/nz/export/ae/utilities/bin/create_inza_db_user.sh user-name database-name
/nz/export/ae/utilities/bin/create_inza_db.sh database-name
```

*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 161.

---

## 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

---

<b>In-Database Deployment Package for Oracle</b> .....	<b>141</b>
Prerequisites .....	141
Overview of the In-Database Package for Oracle .....	141
<b>Oracle Installation and Configuration</b> .....	<b>142</b>
Installing and Configuring Oracle .....	142
Upgrading from a Previous Version .....	142
Installing the In-Database Deployment Package for Oracle .....	143
Creating Users and Objects for the SAS Embedded Process .....	144
<b>Oracle Permissions</b> .....	<b>145</b>
<b>Documentation for Using In-Database Processing in Oracle</b> .....	<b>146</b>

---

## 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 142](#) 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 143](#).

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 144](#).

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 161](#).

*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 143](#).

## 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-12.00000-1.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/12_0/Oracle_on_Linux/` directory.

The `sepcoreorcl-12.00000-1.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-12.00000-1.sh` 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-12.00000-1.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/pochome/SAS`. This directory is referred to as *path\_to\_sh\_file*.
2. Copy the `sepcoreorcl-12.00000-1.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

V9.4M4 16w48 Defects S1279019, and S1285593 Updated new sepcore package method which replaces tkindsrv KMM 10/21/16

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-12.00000-1.sh` file is located.
2. Change permissions on the file to enable you to execute the script.

```
chmod 755 -r sepcoreorcl-12.00000-1.sh
```

3. Use this command to unpack the self-extracting archive file.

```
./sepcoreorcl-12.00000-1.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

V9.4M4 16w48 Defects S1279019, and S1285593. Updated path. KMM 10/21/16

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 161](#).

---

## Documentation for Using In-Database Processing in Oracle

For information about how to publish SAS scoring models, see the [SAS In-Database Products: User's Guide](#), located at <http://support.sas.com/documentation/onlinedoc/indbtech/index.html>.



## Chapter 14

# Administrator's Guide for SAP HANA

---

<b>In-Database Deployment Package for SAP HANA</b> . . . . .	<b>147</b>
Prerequisites . . . . .	147
Overview of the In-Database Deployment Package for SAP HANA . . . . .	147
<b>SAP HANA Installation and Configuration</b> . . . . .	<b>148</b>
SAP Hana Installation and Configuration Steps . . . . .	148
Upgrading from a Previous Version . . . . .	149
Installing the In-Database Deployment Package for SAP HANA . . . . .	150
<b>Installing the SASLINK AFL Plugins on the Appliance (SPS11)</b> . . . . .	<b>152</b>
<b>Importing the SAS_EP Stored Procedure</b> . . . . .	<b>153</b>
<b>Auxiliary Wrapper Procedures (SPS11)</b> . . . . .	<b>153</b>
<b>Controlling the SAS Embedded Process</b> . . . . .	<b>154</b>
<b>Semaphore Requirements When Using the SAS Embedded Process for SAP HANA</b> . . . . .	<b>154</b>
<b>SAP HANA Permissions</b> . . . . .	<b>155</b>
<b>Documentation for Using In-Database Processing in SAP HANA</b> . . . . .	<b>156</b>

---

## 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 155](#).

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 154](#).

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 **scriptserver** 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 149](#) 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 150](#).
6. Install the SASLINK Application Function Library.  
For more information, see [“Installing the SASLINK AFL Plugins on the Appliance \(SPS11\)” on page 152](#).
7. Import the SAS\_EP Stored Procedure.  
For more information, see [“Importing the SAS\\_EP Stored Procedure” on page 153](#).
8. Review the requirement for Auxiliary Wrapper Procedures.  
For more information, see [“Auxiliary Wrapper Procedures \(SPS11\)” on page 153](#).
9. Start the SAS Embedded Process.
  - 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 161](#).

*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 the fourth maintenance release of SAS 9.4 (November 2016), the script deletes the `/SAS/SASTKInDatabaseServerForSAPHANA` directory and all its contents.

For versions starting with the fourth maintenance release of SAS 9.4 (November 2016), the script deletes the `/SASEPHome/bin` directory and all its contents.

4. Reinstall the SAS Embedded Process.

For more information, see [“Installing the In-Database Deployment Package for SAP HANA” on page 150](#).

## 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-12.00000-1.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/12_0/SAP_HANA_on_Linux_x64` directory.

The `sepcorehana-12.00000-1.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-12.00000-1.sh` 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-12.00000-1.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-12.00000-1.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-12.00000-1.sh username@hana:/EPInstallDir
```

*Note:* The `sepcorehana-12.00000-1.sh` file and the **EPInstallDir** directory require Read and Execute permissions for the database administrator.

3. After the `sepcorehana-12.00000-1.sh` has been transferred, log on to the SAP HANA server as the “owner” of the SAS Embedded Process installation directory.

```
ssh sas-owner@server-name
```

4. Navigate to the directory where the self-extracting archive file was downloaded in Step 1.

```
cd /EPInstallDir
```

5. Change permissions on the file to enable you to execute the script.

```
chmod 755 -r sepcorehana-12.00000-1.sh
```

6. Use the following command at the UNIX prompt to unpack the self-extracting archive file:

```
./sepcorehana-12.00000-1.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_EP_sas.com.tgz
/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 154](#).

7. Navigate to the directory where the SAS Embedded Process script files were installed.

```
cd /EPInstallDir/SASEPHome/bin
```

8. Use the following command at the UNIX prompt to install the SAS Embedded Process.

```
./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.

HDB start

---

## Importing the SAS\_EP Stored Procedure

The SAS\_EP Stored Procedure is used by the %INDHN\_RUN\_MODEL macro to run the scoring model.

The SAS\_EP stored procedure is contained in a delivery unit named SAS\_EP\_sas.com.tgz. The SAS\_EP\_sas.com.tgz package was installed in the *EPInstallDir/SASEPHome/misc* directory when the sepcorehana-12.00000-1.sh file was unpacked.

*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:

*Note:* Permissions and roles are required to import the procedure package. For more information, see “[SAP HANA Permissions](#)” on page 155.

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 155.

---

## 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 150](#).

`/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 -sl` command shows the typical default semaphore-related limits set on SAP HANA:

```
----- 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

```

## 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 153](#).

Task	Permission Needed
Unpack the self-extracting archive file	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.
Install or uninstall the SAS Embedded Process (run InstallSASEPFiles.sh or UninstallSASEPFiles.sh script)	root authority
Import the SAS_EP procedure package	A user on the SAP HANA server that has at least the CONTENT_ADMIN role or its equivalent
Install AFL plugins (requires starting and stopping the database)	root authority and database administrator

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 161](#).

---

## Documentation for Using In-Database Processing in SAP HANA

For information about using in-database processing in SAP HANA, see the *SAS In-Database Products: User's Guide*, located at <http://support.sas.com/documentation/onlinedoc/indbtech/index.html>.

## Chapter 15

# Administrator's Guide for SPD Server

---

<b>Installation and Configuration Requirements for the SAS</b>	
<b>Scoring Accelerator for SPD Server</b> .....	<b>157</b>
Prerequisites .....	157
SPD Server Permissions .....	157
<b>Where to Go from Here</b> .....	<b>157</b>

---

## 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** ..... 161



## Chapter 16

# Configuring SAS Model Manager

---

<b>Preparing a Data Management System for Use with SAS Model Manager . . . .</b>	<b>161</b>
Prerequisites . . . . .	161
Overview of Preparing a Data Management System for Use with SAS Model Manager . . . . .	162
<b>Configuring a Database . . . . .</b>	<b>162</b>
SAS Embedded Process Publish Method . . . . .	162
Scoring Function Publish Method . . . . .	163
Finding the JDBC JAR Files . . . . .	164
<b>Configuring a Hadoop Distributed File System . . . . .</b>	<b>165</b>

---

## 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”](#)



- [“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 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 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 `\sasinstalldir\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\Levn\Web\WebAppServer\SASServer1_1\lib`

This is an example of the directory path: `C:\SAS\Config\Lev1\Web\WebAppServer\SASServer1_1\lib`

multiple server install and configuration

`\sasconfigdir\Levn\Web\WebAppServer\SASServer11_1\lib`

This is an example of the directory path: `C:\SAS\Config\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 164](#).

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>. 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\Levn\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\Levn\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 Reinstalling a Previous Version of the Hadoop In-Database Deployment Package

### *Appendix 1*

**Upgrading from or Reinstalling a Previous Version If Using SAS Deployment Manager** ..... 169

### *Appendix 2*

**Upgrading from or Reinstalling a Previous Version If Using Manual Deployment** ..... 175



## Appendix 1

# Upgrading from or Reinstalling a Previous Version If Using SAS Deployment Manager

---

Upgrading from SAS 9.3 .....	169
Upgrading from SAS 9.4 before the July 2015 Release of SAS 9.4 .....	170
Upgrading from or Reinstalling from the July 2015 Release of SAS 9.4 and Later .....	171
Overview .....	171
Removing the SAS Embedded Process Parcel Using Cloudera Manager .....	171
Removing the SAS Embedded Process Stack Using Ambari .....	173

---

## Upgrading from SAS 9.3

To upgrade from SAS 9.3, follow these steps:

1. Stop the SAS Embedded Process.

```
EPInstallDir/SAS/SASTKInDatabaseServerForHadoop/9.35/bin/sasep-stop.all.sh
```

*EPInstallDir* is the master node where you installed the SAS Embedded Process.

2. Delete the SAS Embedded Process from all nodes.

```
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 ***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 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. 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/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
```

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 Reinstalling 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 reinstalling. 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 171 or [“Removing the SAS Embedded Process Stack Using Ambari”](#) on page 173.

### 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

**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 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.  
`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 -rmr /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 `$ASHOME/$ASHadoopConfigurationLibraries/2.4/Config/Deployment/stacks/sasep` directory on the client where the SAS software is downloaded and installed.

```
cd $ASHOME/$ASHadoopConfigurationLibraries/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 Reinstalling a Previous Version If Using Manual Deployment

---

Upgrading from or Reinstalling from SAS 9.3 .....	175
Upgrading from or Reinstalling from SAS 9.4 before the July 2015 Release of SAS 9.4 .....	176
Upgrading from or Reinstalling from the July 2015 Release of SAS 9.4 or Later	177

---

## Upgrading from or Reinstalling from SAS 9.3

To upgrade or reinstall from SAS 9.3, follow these steps:

1. Stop the Hadoop SAS Embedded Process using the 9.3 sasep-stop.all.sh script.

```
EPInstallDir/SAS/SASTKInDatabaseServerForHadoop/9.35/bin/sasep-stop.all.sh
```

*EPInstallDir* is the master node where you installed the SAS Embedded Process.

2. Delete the Hadoop SAS Embedded Process from all nodes.

```
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 **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. Restart the MapReduce service to clear the SAS Hadoop MapReduce JAR files from the cache.
5. Continue the installation process.

If reinstalling 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 33.

## Upgrading from or Reinstalling 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 reinstall from a version of SAS 9.4 before the July 2015 release of SAS 9.4, follow these steps:

1. Stop the Hadoop SAS Embedded Process using the 9.4 `sasep-servers.sh -stop` script.

```
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/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. 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/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
```

5. Continue the installation process.

If you are reinstalling 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 33.

---

## Upgrading from or Reinstalling 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 reinstall 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.
  3. Run this command to remove the SASEPHome directories from the master node.

```
rm -rf SASEPHome
```

4. Continue the installation process.

If you are reinstalling 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 33.



# 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*

For a complete list of SAS publications, go to [sas.com/store/books](http://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](mailto:sasbook@sas.com)  
Web address: [sas.com/store/books](http://sas.com/store/books)



# Index

---

## Special Characters

%INDAC\_PUBLISH\_FORMATS macro  
     79  
 %INDAC\_PUBLISH\_MODEL macro 79  
 %INDB2\_PUBLISH\_COMPILEUDF  
     macro 97  
     running 98  
     syntax 99  
 %INDB2\_PUBLISH\_DELETEUDF  
     macro 101  
     running 101  
     syntax 103  
 %INDB2\_PUBLISH\_FORMATS macro  
     85  
 %INDB2\_PUBLISH\_MODEL macro 85  
 %INDGP\_PUBLISH\_COMPILEUDF  
     macro 114  
     running 115  
     syntax 117  
 %INDGP\_PUBLISH\_COMPILEUDF\_E  
     P macro 118  
     running 118  
     syntax 120  
 %INDGP\_PUBLISH\_FORMATS macro  
     108  
 %INDGP\_PUBLISH\_MODEL macro  
     108  
 %INDHN\_PUBLISH\_MODEL macro  
     147  
 %INDNZ\_PUBLISH\_COMPILEUDF  
     macro 134  
     running 135  
     syntax 136  
 %INDNZ\_PUBLISH\_FORMATS macro  
     125  
 %INDNZ\_PUBLISH\_JAZLIB macro 132  
     running 132  
     syntax 133  
 %INDNZ\_PUBLISH\_MODEL macro  
     125  
 %INDOR\_PUBLISH\_MODEL macro  
     141  
 %INDTD\_PUBLISH\_FORMATS macro  
     55

%INDTD\_PUBLISH\_MODEL macro 55

## A

ACTION= argument  
     %INDB2\_PUBLISH\_COMPILEUDF  
         macro 100  
     %INDB2\_PUBLISH\_DELETEUDF  
         macro 103  
     %INDGP\_PUBLISH\_COMPILEUDF  
         macro 117  
     %INDGP\_PUBLISH\_COMPILEUDF\_  
         EP macro 121  
     %INDNZ\_PUBLISH\_COMPILEUDF  
         macro 137  
     %INDNZ\_PUBLISH\_JAZLIB macro  
         134  
 Ambari  
     deploy SAS Embedded Process stack  
         16, 27  
     remove SAS Embedded Process stack  
         173  
 Aster  
     documentation for publishing formats  
         and scoring models 84  
     in-database deployment package 79  
     installation and configuration 80  
     permissions 83  
     SAS Embedded Process 79  
     SAS/ACCESS Interface 79  
     SQL/MR functions 80  
     authorization for stored procedures 69

## B

binary files  
     for Aster 80  
     for DB2 functions 91  
     for Greenplum functions 111  
     for Netezza functions 128  
 BTEQ 72

**C**

Cloudera  
     Hadoop installation and configuration  
         using the SAS Deployment Manager 13  
         manual Hadoop installation and configuration 31  
 Cloudera Manager  
     deploy SAS Embedded Process parcel 16, 25  
     remove SAS Embedded Process stack 171  
 COMPILER\_PATH= argument  
     %INDB2\_PUBLISH\_COMPILEUDF macro 100  
 configuration  
     Aster 79  
     DB2 87  
     Greenplum 109  
     HDFS for Model Manager 165  
     IBM BigInsights 31  
     MapR 31  
     Model Manager database 162  
     Netezza 127  
     Oracle 141  
     Pivotal HD 31  
     SAP HANA 148  
     SPD Server 157  
     Teradata 59  
 configuration file  
     when to regenerate 40  
 customizing the QKB 74

**D**

data quality stored procedures  
     See [stored procedures](#)  
 DATABASE= argument  
     %INDB2\_PUBLISH\_COMPILEUDF macro 100  
     %INDB2\_PUBLISH\_DELETEUDF macro 103  
     %INDGP\_PUBLISH\_COMPILEUDF macro 117  
     %INDGP\_PUBLISH\_COMPILEUDF\_EP macro 121  
     %INDNZ\_PUBLISH\_COMPILEUDF macro 137  
     %INDNZ\_PUBLISH\_JAZLIB macro 133  
 DataFlux Data Management Studio  
     customizing the QKB 74  
 DB2  
     documentation for publishing formats or scoring models 106  
     function publishing process 86

installation and configuration 87  
 JDBC Driver 164  
 permissions 104  
 preparing for SAS Model Manager use 161  
 SAS Embedded Process 85  
 SAS/ACCESS Interface 85  
 unpacking self-extracting archive files 91, 93  
 DB2IDA command 96  
 DB2PATH= argument  
     %INDB2\_PUBLISH\_COMPILEUDF macro 100  
 DB2SET command 92  
     syntax 95  
 DBSCHEMA= argument  
     %INDNZ\_PUBLISH\_COMPILEUDF macro 137  
     %INDNZ\_PUBLISH\_JAZLIB macro 133  
 documentation  
     for in-database processing in Hadoop 11  
     for in-database processing in SAP HANA 156  
     for publishing formats and scoring models in Aster 84  
     for publishing formats and scoring models in DB2 106  
     for publishing formats and scoring models in Greenplum 124  
     for publishing formats and scoring models in Netezza 139  
     for publishing formats and scoring models in Teradata 57  
     for publishing scoring models in Oracle 146

dq\_grant.sh script 68, 69  
 dq\_install.sh script 68  
 dq\_uninstall script 68  
 dq\_uninstall.sh script 75

**F**

formats library  
     DB2 installation 90, 92, 112  
     Greenplum installation 111  
     Netezza installation 128  
     Teradata installation 63  
 function publishing process  
     DB2 86  
     Netezza 126  
 functions  
     SAS\_COMPILEUDF (DB2) 90, 92, 97, 104, 112

SAS\_COMPILEUDF (Greenplum) 111, 114, 121  
 SAS\_COMPILEUDF (Netezza) 128, 134  
 SAS\_COPYUDF (Greenplum) 121  
 SAS\_DEHEXUDF (Greenplum) 121  
 SAS\_DELETEUDF (DB2) 90, 92, 101, 104, 112  
 SAS\_DIRECTORYUDF (Greenplum) 121  
 SAS\_DIRECTORYUDF (Netezza) 134  
 SAS\_EP (Greenplum) 122  
 SAS\_HEXTOTEXTUDF (Netezza) 134  
 SAS\_PUT( ) (Aster) 79  
 SAS\_PUT( ) (DB2) 85  
 SAS\_PUT( ) (Greenplum) 108  
 SAS\_PUT( ) (Netezza) 125, 126  
 SAS\_PUT( ) (Teradata) 55  
 SAS\_SCORE( ) (Aster) 80  
 SQL/MR (Aster) 80

## G

global variables  
     *See* [variables](#)  
 Greenplum  
     documentation for publishing formats and scoring models 124  
     in-database deployment package 107  
     installation and configuration 109  
     JDBC Driver 165  
     permissions 124  
     preparing for SAS Model Manager use 161  
     SAS Embedded Process 122  
     SAS/ACCESS Interface 107  
     semaphore requirements 123  
     unpacking self-extracting archive files 111

## H

Hadoop  
     backward compatibility 10  
     configuring HDFS using Model Manager 165  
     in-database deployment package 9  
     installation and configuration for IBM BigInsights 31  
     installation and configuration for MapR 31  
     installation and configuration for Pivotal HD 31  
     overview of configuration steps using the SAS Deployment Manager 15

    overview of manual configuration steps 32  
     permissions 10  
     preparing for SAS Model Manager use 161  
     SAS/ACCESS Interface 9  
     unpacking self-extracting archive files 34

## HCatalog

    prerequisites 46  
     SAS client configuration 46  
     SAS Embedded Process configuration 46  
     SAS server-side configuration 47

## Hortonworks

    Hadoop installation and configuration using the SAS Deployment Manager 13  
     manual Hadoop installation and configuration 31

## I

### IBM BigInsights

    Hadoop installation and configuration 31  
     Hadoop installation and configuration using the SAS Deployment Manager 13  
     in-database deployment package for Aster  
         overview 79  
         prerequisites 79  
     in-database deployment package for DB2  
         overview 85  
         prerequisites 85  
     in-database deployment package for Greenplum  
         overview 108  
         prerequisites 107  
     in-database deployment package for Hadoop  
         introduction 7  
         overview 8  
         prerequisites 9  
     in-database deployment package for Netezza  
         overview 125  
         prerequisites 125  
     in-database deployment package for Oracle  
         overview 141  
         prerequisites 141  
     in-database deployment package for SAP HANA  
         overview 147  
         prerequisites 147

in-database deployment package for  
 Teradata  
 overview 55  
 prerequisites 55  
 INDCONN macro variable 98, 102, 116,  
 119, 132, 135  
 installation  
 Aster 79  
 DB2 87  
 Greenplum 109  
 IBM BigInsights 31  
 MapR 31  
 Netezza 127  
 Oracle 141  
 Pivotal HD 31  
 SAP HANA 148  
 SAS Contextual Analysis In-Database  
 Scoring in Hadoop:User's Guide 7  
 SAS Data Quality Accelerator (Hadoop)  
 7  
 SAS Embedded Process (Aster) 79  
 SAS Embedded Process (DB2) 86, 90  
 SAS Embedded Process (Greenplum)  
 108, 111  
 SAS Embedded Process (Hadoop) 7, 34  
 SAS Embedded Process (Netezza) 126,  
 128, 129  
 SAS Embedded Process (Oracle) 141  
 SAS Embedded Process (Teradata) 56  
 SAS formats library 64, 90, 111, 128  
 SAS Hadoop MapReduce JAR files 34  
 scripts 68  
 SPD Server 157  
 Teradata 59  
 troubleshooting 73  
 verifying 72  
 installation, manual  
 SAS In-Database Deployment Package  
 33

## J

JDBC Driver  
 DB2 164  
 Greenplum 165  
 Netezza 165  
 Teradata 165  
 JDBC JAR file locations 164

## M

macro variables  
 See [variables](#)  
 macros  
 %INDAC\_PUBLISH\_FORMATS 79  
 %INDAC\_PUBLISH\_MODEL 79

%INDB2\_PUBLISH\_COMPILEUDF  
 97, 99  
 %INDB2\_PUBLISH\_DELETEUDF  
 101, 103  
 %INDB2\_PUBLISH\_FORMATS 85  
 %INDB2\_PUBLISH\_MODEL 85  
 %INDGP\_PUBLISH\_COMPILEUDF  
 114, 117  
 %INDGP\_PUBLISH\_COMPILEUDF\_  
 EP 120  
 %INDGP\_PUBLISH\_FORMATS 108  
 %INDGP\_PUBLISH\_MODEL 108  
 %INDHN\_PUBLISH\_MODEL 147  
 %INDNZ\_PUBLISH\_COMPILEUDF  
 134, 136  
 %INDNZ\_PUBLISH\_FORMATS 125  
 %INDNZ\_PUBLISH\_JAZLIB 132,  
 133  
 %INDNZ\_PUBLISH\_MODEL 125  
 %INDOR\_PUBLISH\_MODEL 141  
 %INDTD\_PUBLISH\_FORMATS 55  
 %INDTD\_PUBLISH\_MODEL 55

manual installation

SAS In-Database Deployment Package  
 33

MapR

additional configuration 48  
 Hadoop installation and configuration  
 31  
 YARN application CLASSPATH 48

Model Manager

configuration 161  
 configuring a database 162  
 configuring HDFS 165  
 creating tables 163  
 JDBC Driver 164

## N

Netezza

documentation for publishing formats  
 and scoring models 139  
 function publishing process 126  
 in-database deployment package 125  
 installation and configuration 127  
 JDBC Driver 165  
 permissions 138  
 preparing for SAS Model Manager use  
 161  
 publishing SAS formats library 132  
 SAS Embedded Process 125  
 sas\_ep cartridge 130  
 SAS/ACCESS Interface 125

**O**

OBJNAME= argument  
 %INDB2\_PUBLISH\_COMPILEUDF  
 macro 100

OBJPATH= argument  
 %INDGP\_PUBLISH\_COMPILEUDF  
 macro 117  
 %INDGP\_PUBLISH\_COMPILEUDF\_  
 EP macro 120

Oracle  
 documentation for publishing formats  
 and scoring models 146  
 in-database deployment package 141  
 permissions 145  
 preparing for SAS Model Manager use  
 161  
 SAS Embedded Process 141  
 SAS/ACCESS Interface 141

OUTDIR= argument  
 %INDB2\_PUBLISH\_COMPILEUDF  
 macro 101  
 %INDB2\_PUBLISH\_DELETEUDF  
 macro 103  
 %INDGP\_PUBLISH\_COMPILEUDF  
 macro 118  
 %INDGP\_PUBLISH\_COMPILEUDF\_  
 EP macro 121  
 %INDNZ\_PUBLISH\_COMPILEUDF  
 macro 137  
 %INDNZ\_PUBLISH\_JAZLIB macro  
 134

**P**

permissions  
 for Aster 83  
 for DB2 104  
 for Greenplum 124  
 for Hadoop 10  
 for Netezza 138  
 for Oracle 145  
 for SAP HANA 155  
 for SPD Server 157  
 for Teradata 57

Pivotal  
 Hadoop installation and configuration  
 31

Pivotal HD  
 Hadoop installation and configuration  
 using the SAS Deployment Manager  
 13

PSFTP (DB2) 87

publishing  
 Aster permissions 83  
 DB2 permissions 104  
 functions in DB2 86

functions in Netezza 126  
 Greenplum permissions 124  
 Hadoop permissions 10  
 Netezza permissions 138  
 Oracle permissions 145  
 SAP HANA permissions 155  
 SPD Server permissions 157  
 Teradata permissions 57

**Q**

QKB  
 customizing 74  
 packaging for deployment 70  
 updates 74  
 qkb\_pack script 70

**R**

reinstalling a previous version  
 Hadoop 175  
 removing stored procedures 75  
 RPM file (Teradata) 63

**S**

SAP HANA  
 documentation for in-database  
 processing 156  
 in-database deployment package 147  
 installation and configuration 148  
 permissions 155  
 SAS Embedded Process 147, 154  
 SAS/ACCESS Interface 147  
 semaphore requirements 154

SAS Deployment Manager  
 using to deploy Hadoop in-database  
 deployment package 13  
 using to deploy the Teradata in-database  
 deployment package 59

SAS Embedded Process  
 adjusting performance 49  
 Aster 79  
 check status (DB2) 96  
 check status (Teradata) 65  
 configuration for HCatalog file formats  
 46  
 controlling (DB2) 96  
 controlling (Greenplum) 122  
 controlling (Hadoop) 38  
 controlling (SAP HANA) 154  
 controlling (Teradata) 65  
 DB2 85  
 disable or enable (DB2) 96  
 disable or enable (Teradata) 65  
 Greenplum 122

- Hadoop 9
- Netezza 125, 129
- Oracle 141
- overview 8
- SAP HANA 147, 154
- shutdown (DB2) 96
- shutdown (Teradata) 65
- support functions (Teradata) 65
- Teradata 55
- upgrading from a previous version (Aster) 80
- upgrading from a previous version (DB2) 87
- upgrading from a previous version (Netezza) 127
- upgrading from a previous version (Oracle) 142
- upgrading from a previous version (Teradata) 60
- SAS FILENAME SFTP statement (DB2) 86
- SAS formats library
  - DB2 90, 92, 112
  - Greenplum 111
  - Netezza 128, 132
  - Teradata 63
  - upgrading from a previous version (Greenplum) 109
  - upgrading from a previous versions (DB2) 87
  - upgrading from a previous versions (Netezza) 127
  - upgrading from a previous versions (Teradata) 60
- SAS Foundation 9, 55, 79, 107, 125, 141, 147
- SAS Hadoop MapReduce JAR files 34
- SAS In-Database products 4
- SAS\_COMPILEUDF function
  - actions for DB2 97
  - actions for Greenplum 114
  - actions for Netezza 134
  - binary files for DB2 90, 92, 112
  - binary files for Greenplum 111
  - binary files for Netezza 128
  - validating publication for DB2 104
  - validating publication for Greenplum 121
- SAS\_COPYUDF function 114
  - validating publication for Greenplum 121
- SAS\_DEHEXUDF function 115
  - validating publication for Greenplum 121
- SAS\_DELETEUDF function
  - actions for DB2 101
  - binary files for DB2 90, 92, 112
  - validating publication for DB2 104
- SAS\_DIRECTORYUDF function 115, 134
  - validating publication for Greenplum 121
- sas\_ep cartridge 130
- SAS\_EP function
  - validating publication for Greenplum 122
- SAS\_HEXTOTEXTUDF function 134
- SAS\_PUT( ) function
  - Aster 79
  - DB2 86
  - Greenplum 108
  - Netezza 126
  - Teradata 55
- SAS\_SCORE( ) function
  - publishing 80
  - validating publication for Aster 83
- SAS\_SYSFNLIB (Teradata) 65
- SAS/ACCESS Interface to Aster 79
- SAS/ACCESS Interface to Greenplum 107
- SAS/ACCESS Interface to Hadoop 9
- SAS/ACCESS Interface to Netezza 125
- SAS/ACCESS Interface to Oracle 141
- SAS/ACCESS Interface to SAP HANA 147
- SAS/ACCESS Interface to Teradata 55
- sasep-admin.sh script
  - overview 38
  - syntax 39
- sasepfunc function package 65
- SASLIB database (Netezza) 135
- SASLIB schema
  - DB2 98, 101
  - Greenplum 115
- SASUDF\_COMPILER\_PATH global variable 97
- SASUDF\_DB2PATH global variable 97
- scoring functions in SAS Model Manager 162
- scripts for installation 68
- self-extracting archive files
  - unpacking for Aster 80
  - unpacking for DB2 91, 93
  - unpacking for Greenplum 111
  - unpacking for Hadoop 34
- semaphore requirements
  - Greenplum 123
  - SAP HANA 154
- SFTP statement 86
- SPD Server
  - in-database deployment package 157
  - permissions 157



SQL/MR functions (Aster) 80  
 SSH software (DB2) 86  
 stored procedures  
   creating 68  
   removing from database 75

## T

tables  
   creating for SAS Model Manager 163  
 Teradata  
   BTEQ 72  
   documentation for publishing formats  
     and scoring models 57  
   in-database deployment package 55  
   installation and configuration 59  
   JDBC Driver 165  
   Parallel Upgrade Tool 64  
   permissions 57  
   preparing for SAS Model Manager use  
     161  
   SAS Embedded Process 55  
   SAS Embedded Process support  
     functions 65  
   SAS/ACCESS Interface 55  
   sasfunc function package 65  
 troubleshooting installation 73

## U

unpacking self-extracting archive files  
   for Aster 80  
   for DB2 91, 93  
   for Greenplum 111

  for Hadoop 34  
 upgrading from a previous version  
   Aster 80  
   DB2 87  
   Greenplum 109  
   Hadoop 175  
   Netezza 127  
   Oracle 142  
   SAP HANA 149  
   Teradata 60  
 user authorization for stored procedures  
   69

## V

validating publication of functions and  
   variables for DB2 104  
 validating publication of functions for  
   Aster 83  
 validating publication of functions for  
   Greenplum 121, 122  
 variables  
   INDCONN macro variable 98, 102,  
     116, 119, 132, 135  
   SASUDF\_COMPILER\_PATH global  
     variable 97  
   SASUDF\_DB2PATH global variable  
     97  
 verifying installation 72

## Y

YARN application CLASSPATH for  
   MapR 48





# Gain Greater Insight into Your SAS® Software with SAS Books.

Discover all that you need on your journey to knowledge and empowerment.



SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademarks of their respective companies. © 2013 SAS Institute Inc. All rights reserved. S107969US.0613

