THE POWER TO KNOW.

# SAS® In-Database Products
## Administrator's Guide

SAS® Documentation

The correct bibliographic citation for this manual is as follows: SAS Institute Inc 2010. *SAS® In-Database Products: Administrator's Guide*. Cary, NC: SAS Institute Inc.

**SAS® In-Database Products: Administrator's Guide**



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

1st electronic book, June 2010
2nd electronic book, November 2010
3rd electronic book, December 2010
4th electronic book, February 2011

SAS® Publishing provides a complete selection of books and electronic products to help customers use SAS software to its fullest potential. For more information about our e-books, e-learning products, CDs, and hard-copy books, visit the SAS Publishing Web site at **support.sas.com/publishing** or call 1-800-727-3228.

# Contents

# Recommended Reading

- *SAS/ACCESS for Relational Databases: Reference*
- *SAS Scoring Accelerator for Aster nCluster: User's Guide*
- *SAS Scoring Accelerator for DB2 under UNIX: User's Guide*
- *SAS Scoring Accelerator for Greenplum: User's Guide*
- *SAS Scoring Accelerator for Netezza: User's Guide*
- *SAS Scoring Accelerator for Teradata: User's Guide*

For a complete list of SAS publications, go to support.sas.com/bookstore. If you have questions about which titles you need, please contact a SAS Publishing Sales Representative:

SAS Publishing Sales
SAS Campus Drive
Cary, NC 27513-2414
Phone: 1-800-727-3228
Fax: 1-919-677-8166
E-mail: sasbook@sas.com
Web address: support.sas.com/bookstore

*Chapter 1*

# Introduction to the Administrator's Guide



## SAS In-Database Products

The 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, Base SAS and SAS/STAT procedures, and formatted SQL queries inside the database. When using conventional processing, all rows of data are returned from the database to SAS.

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

- SAS/ACCESS Interface to Aster *n*Cluster, SAS/ACCESS Interface to DB2, SAS/ACCESS Interface to Greenplum, SAS/ACCESS Interface to Netezza, 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 the SAS Scoring Accelerator.
- SAS Scoring Accelerator for Aster *n*Cluster, SAS Scoring Accelerator for DB2, SAS Scoring Accelerator for Greenplum, SAS Scoring Accelerator for Netezza, and SAS Scoring Accelerator for Teradata
- SAS Analytics Accelerator for Teradata
- SAS Model Manager Scoring Utility

## 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 and SAS Scoring Accelerator for your database. These components are contained in a deployment package that is specific for your database.

Two components in these packages are the SAS Accelerator Publishing Agent and the SAS Formats Library. The packages contain these versions of the SAS Accelerator Publishing Agent and SAS Formats Library:

- SAS Accelerator Publishing Agent 2.4
- SAS Formats Library 3.1 for Aster *n*Cluster
- SAS Formats Library 1.8 for DB2 under UNIX
- SAS Formats Library 1.8 for Greenplum
- SAS Formats Library 1.8 for Netezza
- SAS Formats Library 1.9 for Teradata

*Note:* Administrative tasks for the SAS Analytics Accelerator are currently in the *SAS Analytics Accelerator for Teradata: User's Guide*. Administrative tasks for the SAS Model Manager In-Database Scoring are currently in the *SAS Model Manager: Administrator's Guide*. The administration tasks for the SAS Analytics Accelerator and SAS Model Manager In-Database Scoring will be added to this guide in the future.

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.

*Chapter 2*

# Administrator's Guide for Aster *n*Cluster



## In-Database Deployment Package for Aster *n*Cluster

### *Prerequisites*

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

### *Overview of the In-Database Deployment Package for Aster nCluster*

The information in this section describes how to load and configure the in-database deployment package for Aster *n*Cluster.

The in-database deployment package for Aster *n*Cluster enables you to use the scoring publishing macro (%INDAC_PUBLISH_MODEL) to create scoring files inside the database.

The in-database deployment package for Aster *n*Cluster contains macros, run-time libraries, and other software that is installed on your Aster *n*Cluster system.

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

The in-database deployment package binary files for Aster *n*Cluster are contained in a self-extracting tar file named accelastrfmt.sh.

To install the in-database deployment package binary files for Aster *n*Cluster, 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 to put accelastrfmt.sh in, execute accelastrfmt.sh, then install the SAS_SCORE function.

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

1. Log in to the queen node.

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

2. Move to the parent of the partner directory.

   The directories are different between the Gentoo and Unbuntu installations.

   For Gentoo, use the following command:

   ```
   cd /cluster/common/workerimage/home/beehive/
   ```

   For Unbuntu, use the following command:

   ```
   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. Use Secure File Transfer Protocol (SFTP) to transfer the self-extracting TAR file to the partner directory.

   a. Using a method of your choice, start the SFTP client.

      Here is an example of starting SFTP from a command line.

      For Gentoo, you could use the following command.

      ```
      sftp root@name-or-ip-of-queen-node: /cluster/common/workerimage/
         home/beehive/partner
      ```

      For Unbuntu, you could use the following command.

      ```
      sftp root@name-or-ip-of-queen-node:/home/beehive/partner
      ```

   b. For both Gentoo and Unbuntu, enter this command from the SFTP prompt to transfer the TAR file.

      ```
      put accelastrfmt.sh
      ```

6. (Optional) If your SFTP client does not copy the executable attribute from the client machine to the server, change the execute permission on the TAR file.

   ```
   chmod +x accelastrfmt.sh
   ```

7. Create a directory named SAS and unpack the TAR file into it.

   ```
   ./accelastrfmt.sh
   ```

   *Note:* If a SAS directory already exists in the partner directory, you can use **`rm –rf SAS`** to remove it and any previous versions of the in-database deployment package for Aster *n*Cluster.

8. Remove the TAR file from the partner directory.

   ```
   rm accelastrfmt.sh
   ```

9. Change to the directory where SAS is installed.

   ```
   cd SAS/SASAcceleratorForAster/9.2-3.1/sasexe
   ```

10. Enter the following commands to install the SAS_SCORE 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. (Optional) If this is not the first time you have installed the in-database deployment package for Aster *n*Cluster, it is recommended that you remove the existing SQL/MR functions before installing the new ones.

```
\remove sas_score.tk.so
```

c. Enter the following command to install the new SQL/MR functions.

```
\install sas_score.tk.so
```

11. Exit the ACT tool.

```
\q
```

## *Aster nCluster Permissions*

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

No permissions are needed by the person who runs the scoring publishing macros, because all functions and files are published to the PUBLIC schema.

## *Documentation for Publishing SAS Scoring Models in Aster nCluster*

For information about how to publish SAS scoring models, see the *SAS Scoring Accelerator for Aster nCluster User's Guide* located at **http://support.sas.com/documentation/onlinedoc/scoracclast/index.html**

*Chapter 3*

# Administrator's Guide for DB2



## SAS Accelerator Publishing Agent for DB2

### *Prerequisites*

SAS Foundation and the SAS/ACCESS Interface to DB2 must be installed before you install and configure the SAS Accelerator Publishing Agent for DB2.

### *Overview of the SAS Accelerator Publishing Agent for DB2*

This section describes how to install and configure the SAS Accelerator Publishing Agent for DB2. The SAS 9.2 Formats Library for DB2 is included with the SAS Accelerator Publishing Agent for DB2. This section also describes how to install the SAS 9.2 Formats Library.

The SAS Accelerator Publishing Agent for DB2 enables you to use a scoring publishing macro (%INDB2_PUBLISH_MODEL) to create scoring model functions inside the database.

The SAS Accelerator Publishing Agent for DB2 contains three publishing macros:

- %INDB2_PUBLISH_MODEL, which enables you to create scoring model functions.
- %INDB2_PUBLISH_COMPILEUDF and %INDB2_PUBLISH_DELETEUDF, which register utility functions in the database. The utility functions are called by the scoring publishing macro, %INDB2_PUBLISH_MODEL. You must run these two macros before you run the scoring publishing macro.

### Function Publishing Process in DB2

To publish the SAS scoring model functions on a DB2 server, the publishing macros need to create and transfer the files to the DB2 environment, compile those source files into object files using the appropriate compiler for that system, and then link with the SAS 9.2 Formats Library. After that, the publishing macros register the scoring model functions in DB2 with those object files. If an existing scoring model function is replaced, the publishing macros remove the obsolete object file upon successful compilation and publication of the new scoring model function.

The publishing macros use a SAS FILENAME SFTP statement to transfer the 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*, located at **`http://support.sas.com/techsup/technote/ts800.pdf`**.

### DB2 Installation and Configuration Steps

1. Apply Hot Fix B85002 for SAS Table Server Base Components 9.2_M2. This hot fix can be found at **`http://ftp.sas.com/techsup/dowload/hotfix/HF2/B85.html#B85002`**.
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> userid@machinename
   psftp> ls
   ```

3. Install the SAS 9.2 Formats Library and binary files for the SAS_COMPILEUDF and SAS_DELETEUDF functions.

   For more information, see “Installing the SAS 9.2 Formats Library and Binary Files” on page 9.
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 10.
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 14.

## Installing the SAS 9.2 Formats Library and Binary Files

### Move the Files to DB2

The SAS 9.2 Formats Library and the binary files for the SAS_COMPILEUDF and SAS_DELETEUDF functions are contained in a self-extracting TAR file. The location of the TAR file, the README file, and possibly other files, is in the ordersummary.html file that accompanies the software. Your order e-mail contains the path to the ordersummary.html file. You can use PSFTP, SFTP, or FTP to transfer the TAR file to the DB2 server to be unpacked and compiled.

The file does not have to be downloaded to a specific location, but you need to note where it is downloaded so that it can be executed as the DB2 instance owner at a later time. Choose the TAR file based on the UNIX platform that your DB2 server runs on. For example:

AIX: acceldb2fmt_AIX.sh

Linux(x86_64):acceldb2fmt_Linux_x86_64.sh

List the directory in UNIX to verify that the file has been moved.

### Unpack the Files

After the TAR file has been transferred to the DB2 machine, follow these steps to unpack the files:

1. Log in as the user who owns the DB2 instance from a secured shell, such as SSH.
2. Use the following commands to unpack the appropriate TAR file.

   ```
   $ cd path_to_tar_file
   $ sh tar_file
   ```

   ***path_to_tar_file*** is the location to which you copied the TAR file.

   ***tar_file*** is either acceldb2fmt_Linux_x86_64.sh or acceldb2fmt_AIX.sh depending on your operating system.

   After this script is run and the files are unpacked, 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_tar_file/SAS/SASFormatsLibraryForDB2/9.2-1.8/bin/
      InstallAccelDB2Fmt_AIX.sh

   /path_to_tar_file/SAS/SASFormatsLibraryForDB2/9.2-1.8/bin/CopySASFiles.sh

   /path_to_tar_file/SAS/SASFormatsLibraryForDB2/9.2-1.8/lib/SAS_CompileUDF

   /path_to_tar_file/SAS/SASFormatsLibraryForDB2/9.2-1.8/lib/SAS_DeleteUDF

   /path_to_tar_file/SAS/SASFormatsLibraryForDB2/9.2-1.8/lib/libjazxfbrs.so

   /path_to_tar_file/SAS/SASFormatsLibraryForDB2/9.2 -> ~SAS/
      SASFormatsLibraryForDB2/9.2-1.8
   ```

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

   ```
   $ path_to_tar_file/SAS/SASFormatsLibraryForDB2/9.2/bin/
      CopySASFiles.sh path_to_sqllib_for_db2
   ```

*Note:* If the SAS_CompileUDF,SAS_DeleteUDF, and libjazxfbrs.so files currently exist, you must rename the existing files before you run the CopySASFiles.sh command. Otherwise, the CopySASFiles.sh command does not work, and you get a "Text file is busy" message for each of the three files.

***path_to_sqllib_for_db2*** is the path to the DB2 instance that you want to use (typically ***db2path*/sqllib**).

After this script is run and the files are copied, the target directory should look similar to this.

```
db2path/sqllib/function/SAS/SAS_CompileUDF
db2path/sqllib/function/SAS/SAS_DeleteUDF
db2path/sqllib/function/SAS/libjazxfbrs.so
```

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

   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.

   Before running the DB2SET command, ensure that the DB2 environment is set up correctly. To source the DB2 environment, run the following command.

   ```
   $ path_to_sqllib_for_db2 . . /db2profile
   ```

   Now, run the DB2SET command.

   ```
   $ db2set DB2LIBPATH=path_to_sqllib_for_db2/function/SAS
   ```

   ***path_to_sqllib_for_db2*** is the path to the DB2 instance that you want to use (typically ***db2path*/sqllib**).

5. To verify that DB2LIBPATH was set appropriately, run the DB2SET command without any parameters as follows.

   ```
   $ path_to_sqllib_for_db2/adm/db2set
   ```

   The correct path should be listed if it was set correctly.

## 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_MODEL scoring publishing macro. The SAS_COMPILEUDF function performs the following tasks:

  - compiles the scoring model source files into object files. This compilation occurs through the SQL interface using an appropriate compiler for the system.
  - copies the object files to the ***db2path*/sqllib/function/SAS** directory. You specify the value of **db2path** in the %INDB2_PUBLISH_COMPILEUDF macro syntax.
  - registers those object files in DB2.

  - creates a link to the SAS 9.2 Formats Library for DB2 that is needed for scoring model publishing.
- SASUDF_DB2PATH and SASUDF_COMPILER_PATH global variables

  The SASUDF_DB2PATH and the SASUDF_COMPILER_PATH global variables are used when you publish the scoring model functions.

You have to run the %INDB2_PUBLISH_COMPILEUDF macro only one time.

The SAS_COMPILEUDF function must be published before you run the %INDB2_PUBLISH_DELETEUDF 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 for the SAS Accelerator Publishing Agent" on page 18.

### %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 published.

   You must use "SASLIB" as the schema name for DB2 in-database processing to work correctly.

   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 13.

   The SASLIB schema will contain the SAS_COMPILEUDF and SAS_DELETEUDF functions and the SASUDF_DB2PATH and SASUDF_COMPILER_PATH global variables.

2. Start SAS 9.2 and submit the following commands in the Enhanced Editor or Program Editor:

```
%indb2pc;
%let indconn = server=yourserver user=youruserid password=yourpwd
   database=yourdb;
%indb2_publish_compileudf(
   database=database,
   action=create,
   db2path=yourdb2path/sqllib,
   compiler_path=yourcompilerpath,
   objname=yourobjectfilename,
   outdir=youroutputdirectory);
```

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 17.

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 14.

### %INDB2PC Macro

The %INDB2PC macro is an autocall library that initializes the %INDB2_PUBLISH_COMPILEUDF macro.

### 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*<'>

**SERVER=<'>*server*<'>**
specifies the DB2 server name or the IP address of the server host. If the server name contains spaces or nonalphanumeric characters, you must enclose it in quotation marks.

**Requirement:** The name must be consistent with the way that 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 8.

**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, you must enclose it in quotation marks.

**PASSWORD=<'>*password*<'>**
specifies the password that is associated with your DB2 user ID. If the password contains spaces or nonalphabetic characters, you must enclose it in quotation marks.

**Tip:** You can use only PASSWORD= or PW= for the password argument. Other aliases such as PASS= or PWD= are not supported and cause 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, you must enclose it in quotation marks.

*Note:* 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. The SASLIB schema must be created before publishing the SAS_COMPILEUDF and SAS_DELETEUDF functions.

### %INDB2_PUBLISH_COMPILEUDF Macro Syntax

**%INDB2_PUBLISH_COMPILEUDF**(

DB2PATH=*db2path*/sqllib

, COMPILER_PATH=*compiler-path-directory*

<, DATABASE=*database-name*>

<, ACTION=CREATE | REPLACE | DROP>

<, OBJNAME=*object-file-name*>

<, OUTDIR=*diagnostic-output-directory*>

);

**Arguments**

**DB2PATH=*db2path*/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 scoring model functions.

**Interaction:** *db2path* 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 the Files" on page 9.

**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 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 gcc 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 11.

**Tip:** This argument lets you publish the SAS_COMPILEUDF function to a shared database where other users can access it.

**ACTION=CREATE | REPLACE | DROP**

specifies that the macro performs one of the following actions:

CREATE
creates a new SAS_COMPILEUDF function.

REPLACE
overwrites the current SAS_COMPILEUDF function, if a SAS_COMPILEUDF function by the same name is already registered, or creates a new SAS_COMPILEUDF function if one is not registered.

DROP
causes the SAS_COMPILEUDF function to be dropped from the DB2 database.

**Default:** CREATE

**Tip:** If the SAS_COMPILEUDF function was published previously and you now specify ACTION=CREATE, you receive warning messages from DB2. If the

SAS_COMPILEUDF function was published previously and you specify ACTION=REPLACE, no warnings are issued.

**OBJNAME=*object-file-name***
specifies the object filename that the publishing macro uses to register the SAS_COMPILEUDF function. The object filename is a file system reference to a specific object file, and the value entered for OBJNAME must match the name as it exists in the file system. For example, SAS_CompileUDF is mixed case.

**Default:** SAS_CompileUDF

**Interaction:** If the SAS_COMPILEUDF function is updated, you might want to rename the object file to avoid stopping and restarting the database. If so, the SAS_COMPILEUDF function needs to be reregistered with the new object filename.

**OUTDIR=*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_MODEL scoring publishing macro. The SAS_DELETEUDF function removes existing object files when the scoring publishing macro registers new ones by the same name.

You have to run the %INDB2_PUBLISH_DELETEUDF macro only one time.

The SAS_COMPILEUDF function must be published before you run the %INDB2_PUBLISH_DELETEUDF 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 for the SAS Accelerator Publishing Agent" on page 18.

### %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 published. You must use "SASLIB" as the schema name for DB2 in–database processing to work correctly.

   The SASLIB schema should have been created when 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 11.

2. Start SAS 9.2 and submit the following commands in the Enhanced Editor or Program Editor.

```
%indb2pd;
%let indconn = server=yourserver user=youruserid password=yourpwd
   database=yourdb;
%indb2_publish_deleteudf(
   database=database,
   action=create,
   outdir=youroutputdirectory);
```

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 17.

After the SAS_DELETEUDF function is published, the %INDB2_PUBLISH_MODEL macro can be run to publish the scoring model files.

### %INDB2PD Macro

The %INDB2PD macro is an autocall library that initializes the %INDB2_PUBLISH_DELETEUDF macro.

### 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*<'>

**SERVER=<'>*server*<'>**
specifies the DB2 server name or the IP address of the server host. If the server name contains spaces or nonalphanumeric characters, you must enclose the name in quotation marks.

**Requirement:** The name must be consistent with the way that 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 8.

**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, you must enclose it in quotation marks.

**PASSWORD=<'>*password*<'>**
specifies the password that is associated with your DB2 user ID. If the password contains spaces or nonalphabetic characters, you must enclose it in quotation marks.

**Tip:** You can use only PASSWORD= or PW= for the password argument. Other aliases such as PASS= or PWD= are not supported and cause 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, you must enclose the name in quotation marks.

*Note:* The SAS_DELETEUDF function is published to the SASLIB schema in the specified database. The SASLIB schema must be created 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 14.

**Tip:** This argument lets you publish the SAS_DELETEUDF function to a shared database where other users can access it.

**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_DELTEUDF 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 global variables are created properly, follow these steps.

1. Connect to your DB2 database using Command Line Processor (CLP).
2. Enter the following command.

   ```
   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. Open a UNIX window; validate that the xlc compiler (on AIX) or gcc compiler (on Linux) is in the path that you received as a result.
4. Connect to DB2 using CLP and enter the following command.

   ```
   values(saslib.sasudf_db2path)
   ```

   You should receive a result similar to the following.

   ```
   /users/db2v9/sqllib
   ```

   In this example, **`/users/db2v9`** is the value of *db2path* that was specified during installation and when the SAS_COMPILEUDF function was published.
5. Open a UNIX window; validate that sasudf_db2path is defined as the path that you received as a result.
6. Connect to DB2 using CLP and enter the following command.

   ```
   select funcname, implementation from syscat.functions where
      funcschema='SASLIB'
   ```

   You should receive a result similar to the following.

   ```
   FUNCNAME                          IMPLEMENTATION
   ------------------------------------------------------------
   SAS_DELETEUDF
   /users/db2v9/sqllib/function/SAS/SAS_DeleteUDF!SAS_DeleteUDF
   SAS_COMPILEUDF
   /users/db2v9/sqllib/function/SAS/SAS_CompileUDF!SAS_CompileUDF
   ```

7. Open a UNIX window; validate that the SAS_COMPILEUDF and SAS_DELETEUDF functions are installed in the paths that you received as results.
8. To validate that the SAS_COMPILEUDF and SAS_DELETEUDF functions were built properly for the server box, enter an LDD command from the UNIX command line similar to this one.

   ```
   $ ldd path_to_sqllib_for_db2/function/SAS/SAS_CompileUDF
   ```

   The results should look similar to the following, depending on your operating system.

   ```
   SAS_CompileUDF needs:
        /usr/lib/libc.a(shr_64.o)
        /unix
        /usr/lib/libcrypt.a(shr_64.o)
   ```

### DB2 Permissions for the SAS Accelerator Publishing Agent

There are two sets of permissions involved with the SAS Accelerator Publishing Agent 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<br>*Note:* 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 compiledeletepublisheruserid` |
| 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 compiledeletepublisheruserid` |

The second set of permissions is needed by the person who publishes the scoring models. The person who publishes the 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 scoring model developer. Without these permissions, the publishing of the scoring model functions fails.

*Note:* Permissions must be granted for every scoring model publisher and for each database that the scoring model publishing uses. Therefore, you might need to grant these permissions multiple times.

After the DB2 permissions have been set appropriately, the scoring model publishing macro should be called to register the scoring models.

The following table summarizes the permissions that are needed by the person who publishes the scoring models.

| Permission Needed | Authority Required to Grant Permission | Examples |
|---|---|---|
| EXECUTE permission for functions that have been published.<br>This enables the person who publishes the scoring models to execute the SAS_COMPILEUDF and SAS_DELETEUDF functions. | System Administrator or Database Administrator<br>*Note:* 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 scoringpublisheruserid` |
| CREATE_EXTERNAL_ROUTINE permission to the database to create scoring model functions | | `GRANT CREATE_EXTERNAL_ROUTINE ON DATABASE TO scoringpublisheruserid` |
| CREATE_NOT_FENCED_ROUTINE permission to create scoring model functions that are not fenced | | `GRANT CREATE_NOT_FENCED_ROUTINE ON DATABASE TO scoringpublisheruserid` |
| CREATEIN permission for the schema in which the scoring functions are published if the default schema (SASLIB) is not used | | `GRANT CREATEIN ON SCHEMA scoringschema TO scoringpublisheruserid` |
| READ permission to read the SASUDF_COMPILER_PATH and SASUDF_DB2PATH global variables<br>*Note:* 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<br>*Note:* 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 scoringpublisheruserid`<br>`GRANT READ ON VARIABLE SASLIB.SASUDF_COMPILER_PATH TO scoringpublisheruserid` |

## *Documentation for Publishing SAS Scoring Models in DB2*

For information about how to publish SAS scoring models, see the *SAS Scoring Accelerator for DB2 under UNIX: User's Guide* located at **http://support.sas.com/documentation/onlinedoc/scoraccldb2/index.html**.

*Chapter 4*

# Administrator's Guide to Greenplum



## SAS Accelerator Publishing Agent for Greenplum

### *Prerequisites*

SAS Foundation and the SAS/ACCESS Interface to Greenplum must be installed before you install and configure the SAS Accelerator Publishing Agent for Greenplum.

### *Overview of the SAS Accelerator Publishing Agent for Greenplum*

This section describes how to install and configure the SAS Accelerator Publishing Agent for Greenplum. The SAS 9.2 Formats Library for Greenplum is included with the SAS Accelerator Publishing Agent for Greenplum. This section also describes how to install the SAS 9.2 Formats Library.

The SAS Accelerator Publishing Agent for Greenplum enables you to use a scoring publishing macro (%INDGP_PUBLISH_MODEL) to create scoring model functions inside the database.

The SAS 9.2 Formats Library for Greenplum is a run-time library that is installed on your Greenplum system so that the SAS scoring model functions created in Greenplum can access the routines within its run-time library.

The SAS Accelerator Publishing Agent for Greenplum contains two publishing macros:

- %INDGP_PUBLISH_MODEL, which enables you to create scoring model functions.

- %INDGP_PUBLISH_COMPILEUDF, which registers utility functions in the database. The utility functions are called by the scoring publishing macro, %INDGP_PUBLISH_MODEL. You must run this macro before you run the scoring publishing macro.

### Function Publishing Process in Greenplum

To publish the SAS scoring model functions to a Greenplum database, the publishing macros perform the following tasks:

- Create and transfer the source files to the Greenplum server.

  The files are transferred through database tables. 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 a temporary directory on the database server, the source files are converted back to text.

- Compile those source files into object files using the appropriate compiler for the Greenplum system.
- Link with the SAS 9.2 Formats Library.
- Copy the shared object files to *full-path-to-pkglibdir*/SAS. The object files are loaded when the scoring model functions are called.
- Register the scoring model functions in Greenplum with those object files. If an existing scoring model function is replaced, the publishing macros replace the obsolete object file upon successful compilation and publication of the new scoring model function.

### Greenplum Installation and Configuration Steps

1. Apply Hot Fix B85002 for SAS Table Server Base Components 9.2_M2. This hot fix can be found at **`http://ftp.sas.com/techsup/dowload/hotfix/HF2/B85.html#B85002`**.
2. Move and unpack the SAS 9.2 Formats Library and binary files for the publishing macros.

   For more information, see "Moving and Unpacking the SAS 9.2 Formats Library and Binary Files" on page 22.

3. Run the %INDGP_PUBLISH_COMPILEUDF macro.

   For more information, see "Running the %INDGP_PUBLISH_COMPILEUDF Macro" on page 24.

### Moving and Unpacking the SAS 9.2 Formats Library and Binary Files

The SAS 9.2 Formats Library and the binary files for the publishing macros are contained in a self-extracting TAR file on the SAS Software Depot. The location of the TAR file, the README file, and possibly other files, is contained in the ordersummary.html file that accompanies the software. Your order e-mail contains the path to the ordersummary.html file.

To move and unpack the TAR file, follow these steps:

1. Using a method of your choice, transfer the accelgplmfmt.sh file to your Greenplum master node.

   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.

   The accelgplmfmt.sh file is located in the ***install-depot-dir*** **\products\accelgplmfmt_92110_prt_xx_sp0_1** directory.

2. After the accelgplmfmt.sh has been transferred, log in to the Greenplum master node.

3. Move to the directory where the TAR file was downloaded.

4. Use the following command at the UNIX prompt to unpack the TAR file.

   ```
   ./accelgplmfmt.sh
   ```

   *Note:* If you receive a "permissions denied" message, check the permissions on the accelgplmfmt.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_tar_file* is the location to which you copied the TAR file.

   ```
   /path_to_tar_file/SAS/SASFormatsLibraryForGreenplum/9.2-1.8/bin/
       InstallAccelGplmFmt.sh
   /path_to_tar_file/SAS/SASFormatsLibraryForGreenplum/9.2-1.8/bin/
       CopySASFiles.sh
   /path_to_tar_file/SAS/SASFormatsLibraryForGreenplum/9.2-1.8/lib/
       SAS_CompileUDF.so
   /path_to_tar_file/SAS/SASFormatsLibraryForGreenplum/9.2-1.8/lib/
       libjazxfbrs.so
   ```

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

   ```
   /path_to_tar_file/SAS/SASFormatsLibraryForGreenplum/9.2-1.8/bin/
       CopySASFiles.sh
   ```

   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
   ```

   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 7.

6. From the master node, follow these steps to create symbolic links to the SAS 9.2 Formats Library for Greenplum. The symbolic links are created where the library was loaded on each node in the database array including the master and all segments.

   a. Use the following command to determine the full path to where the library was loaded.

      ```
      $ pg_config --libdir
      ```

      This is the path where the symbolic link is created.

   b. Use the following command to determine the SAS In-Database shared library deployment path.

      ```
      $ pg_config --pkglibdir
      ```

This is the path that is linked to and where the SAS 9.2 Formats Library for Greenplum is deployed.

c. Use the following command to create the symbolic link on the master node.

```
$ ln -s path-from-pg_config --pkglibdir/SAS/libjazxfbrs.so
   path-from-pg_config --libdir/libjazxfbrs.so
```

Use the value from Step 6b for *path-from-pg_config --pkglibdir*. Use the value from Step 6a for *path-from-pg_config --libdir*.

d. Use the following commands to connect to each of the segment nodes and create the symbolic links on each of the nodes.

```
/* Use this command from the master node to connect to each segment node */
$ ssh <segment nodename>
/* Use this commmand on each segment node to create the link */
$ ln -s path-from-pg_config --pkglibdir>/SAS/libjazxfbrs.so
    path-from-pg_config --libdir/libjazxfbrs.so
```

To verify that the link is created correctly, go to the directory that results from running the **`pg_config --libdir`** command and list libjazxfbrs.so.

7. (Optional) If you add new nodes to the Greenplum master node after the initial installation of the SAS Accelerator Publishing Agent for Greenplum, you must copy all the binaries in the *full-path-to-pkglibdir*/SAS directory, including SAS_CompileUDF.so, libjazxfbrs.so, and the binary files for the already published functions, to the new nodes using a method of your choice. In addition, you must repeat Step 6 to create the symbolic links to the SAS 9.2 Formats Library for Greenplum (libjazfbrs.so).

## *Running the %INDGP_PUBLISH_COMPILEUDF Macro*

### *Overview of the %INDGP_PUBLISH_COMPILEUDF Macro*

The %INDGP_PUBLISH_COMPILEUDF macro publishes the following functions to the SASLIB schema in a Greenplum database:

- SAS_COMPILEUDF

  The SAS_COMPILEUDF function facilitates the %INDGP_PUBLISH_MODEL scoring publishing macro. The SAS_COMPILEUDF function performs the following tasks:

  - compiles the scoring model source files into object files. This compilation occurs through the SQL interface using an appropriate compiler for the system.
  - 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. You can use the **`pg_config --pkglibdir`** command to determine the *full-path-to-pkglibdir* directory.
  - creates a link to the SAS 9.2 Formats Library for Greenplum, which is needed for scoring model publishing.

- Three utility functions that are used when the scoring publishing macro transfers source files from the client to the host:

  - SAS_COPYUDF function

    The SAS_COPYUDF function copies the shared libraries to the ***`full-path-to-pkglibdir/SAS`*** path on the whole database array including

the master and all segments. The SAS_COPYUDF function also registers the object files in Greenplum.

- SAS_DIRECTORYUDF function

  The SAS_DIRECTORYUDF function creates and removes a temporary directory that holds the source files on the server.

- SAS_DEHEXUDF function

  The SAS_DEHEXUDF function converts the files from hexadecimal back to text after the files are on the host.

For more information about the file transfer process, see "Function Publishing Process in Greenplum" on page 22.

You have to run the %INDGP_PUBLISH_COMPILEUDF macro only one time in each database.

The SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF functions must be published before you run the %INDGP_PUBLISH_MODEL macro. Otherwise, the scoring model publishing fails.

*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 27.

   The SASLIB schema will contain the SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF functions.

2. Start SAS 9.2 and submit the following commands in the Enhanced Editor or Program Editor:

```
%indgppc;
%let indconn = user=youruserid password=yourpwd dsn=yourdsn;
/* You can use server=yourserver database=yourdb instead of dsn=yourdsn */
%indgp_publish_compileudf(
   objpath=full-path-to-pkglibdir/SAS,
   database=yourdatabase,
   action=create,
   outdir=youroutputdirectory);
```

For more information about the arguments for the %INDGP_PUBLISH_COMPILEUDF macro, see "%INDGP_PUBLISH_COMPILEUDF Macro Syntax" on page 27.

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 28.

### %INDGPPC Macro

The %INDGPPC macro is an autocall library that initializes the %INDGP_PUBLISH_COMPILEUDF macro.

### 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*

USER=<'>*userid*<'> PASSWORD=<'>*password*<'> SERVER=<'>*server*<'> DATABASE=<'>*database*<'>

**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, you must enclose it in quotation marks.

**PASSWORD=<'>*password*<'>**
specifies the password that is associated with your Greenplum user ID. If the password contains spaces or nonalphabetic characters, you must enclose it in quotation marks.

**Tip:** You can use only PASSWORD= or PW= for the password argument. Other aliases such as PASS= or PWD= are not supported and cause 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, you must enclose it 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, you must enclose it 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, you must enclose it in quotation marks.

**Requirement:** You must specify either the DSN= argument or the SERVER= and DATABASE= arguments in the INDCONN macro variable

*Note:* The default port that is specified by Greenplum is 5432.

*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 TAR file. You can use the **`pg_config --pkglibdir`** command to determine the name of the *full-path-to-pkglibdir* directory.

**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.

**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.

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 now specify ACTION=CREATE, you receive warning messages from Greenplum 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=*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 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
   ```

### *Greenplum Permissions for the SAS Accelerator Publishing Agent*

To publish the SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, SAS_DEHEXUDF, and scoring model functions, Greenplum requires that you have superuser permissions to create and execute these functions in the SASLIB schema and in the specified database.

*Chapter 5*

# Administrator's Guide for Netezza



## SAS Accelerator Publishing Agent for Netezza

### *Prerequisites*

SAS Foundation and the SAS/ACCESS Interface to Netezza must be installed before you install and configure the SAS Accelerator Publishing Agent for Netezza.

### *Overview of the SAS Accelerator Publishing Agent for Netezza*

This section describes how to install and configure the SAS Accelerator Publishing Agent for Netezza. The SAS 9.2 Formats Library for Netezza is included with the SAS Accelerator Publishing Agent for Netezza. This section also describes how to install the SAS 9.2 Formats Library.

The SAS Accelerator Publishing Agent and the SAS 9.2 Formats Library are used by both the SAS/ACCESS Interface to Netezza and by the SAS Scoring Accelerator for Netezza.

- In the SAS/ACCESS Interface to Netezza, a format publishing macro, %INDNZ_PUBLISH_FORMATS, creates or publishes the SAS_PUT() function and formats inside the database.
- In the SAS Scoring Accelerator for Netezza, a scoring publishing macro, %INDNZ_PUBLISH_MODEL, creates or publishes scoring model functions.

The SAS 9.2 Formats Library for Netezza is a run-time library that is installed on your Netezza system so that the SAS scoring model functions and the SAS_PUT() function created in Netezza can access the routines within its run-time library.

The SAS Accelerator Publishing Agent contains four publishing macros:

- %INDNZ_PUBLISH_FORMATS
- %INDNZ_PUBLISH_MODEL
- %INDNZ_PUBLISH_JAZLIB (TwinFin systems only)
- %INDNZ_PUBLISH_COMPILEUDF

The %INDNZ_PUBLISH_JAZLIB macro registers the SAS 9.2 Formats Library for TwinFin systems. 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.

### Function Publishing Process in Netezza

To create the SAS scoring model, SAS_PUT() function, and formats on Netezza Performance Server (NPS) and TwinFin systems, the 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 9.2 Formats Library.
- Register those object files with the NPS or TwinFin systems.

### Netezza Installation and Configuration Steps

1. Apply Hot Fix B85002 for SAS Table Server Base Components 9.2_M2. This hot fix can be found at **`http://ftp.sas.com/techsup/dowload/hotfix/HF2/B85.html#B85002`**.
2. Move and unpack the SAS 9.2 Formats Library and binary files for the SAS_COMPILEUDF function.

   For more information, see “Moving and Unpacking the SAS 9.2 Formats Library and Binary Files” on page 31.
3. For TwinFin systems, run the %INDNZ_PUBLISH_JAZLIB macro to publish the SAS 9.2 Formats Library for Netezza as an object.

   For more information, see “Running the %INDNZ_PUBLISH_JAZLIB Macro” on page 31.
4. Run the %INDNZ_PUBLISH_COMPILEUDF macro.

   For more information, see“Running the %INDNZ_PUBLISH_COMPILEUDF Macro” on page 33.

## Moving and Unpacking the SAS 9.2 Formats Library and Binary Files

The SAS 9.2 Formats Library and the binary files for the SAS_COMPILEUDF function are contained in a self-extracting TAR file on the SAS Software Depot. The location of the TAR file, the README file, and possibly other files, is contained in the ordersummary.html file that accompanies the software. Your order e-mail contains the path to the ordersummary.html file. To move and unpack the TAR file, follow these steps:

1. Using a method of your choice, transfer one of these files to your Netezza system:
   - For NPS systems, accelnetzfmt_nps.sh
   - For TwinFin systems, accelnetzfmt_twinfin.sh
2. After the accelnetzfmt_nps.sh or accelnetzfmt_twinfin.sh file has been transferred to the Netezza machine, log in as the user who owns the Netezza software (usually the "nz" ID).
3. Use the following commands at the UNIX prompt to unpack the TAR file.

   ```
   mkdir -p /nz/extensions
   chmod 755 /nz/extensions
   cd /nz/extensions
   chmod 755 accelnetzfmt_nps.sh | /accelnetzfmt_twinfin.sh
   path_to_self-extracting_tar_file/accelnetzfmt_nps.sh | /accelnetzfmt_twinfin.sh
   ```

   After the script runs and the files are unpacked, the target directories should look similar to these.

   For Netezza NPS systems:

   ```
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/9.2-1.8/bin/InstallAccelNetzFmt.sh
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/9.2-1.8/lib/SAS_CompileUDF.o_diab_ppc
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/9.2-1.8/lib/SAS_CompileUDF.o_x86
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/9.2-1.8/lib/libjazxfbrs_diab_ppc.a
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/9.2-1.8/lib/libjazxfbrs_x86.a
   ```

   For Netezza TwinFin systems:

   ```
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/9.2-1.8/bin/InstallAccelNetzFmt.sh
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/9.2-1.8/lib/SAS_CompileUDF.o_spu10
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/9.2-1.8/lib/SAS_CompileUDF.o_x86
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/9.2-1.8/lib/libjazxfbrs_spu10.so
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/9.2-1.8/lib/libjazxfbrs_x86.so
   ```

   There also is a symbolic link such that **`/nz/extensions/SAS/SASFormatsLibraryForNetezza/9.2`** points to the latest version.

## Running the %INDNZ_PUBLISH_JAZLIB Macro

### Overview of Publishing the SAS 9.2 Formats Library

For NPS systems, the SAS 9.2 Formats Library is built as a static object and is linked to the scoring and format functions by the SAS_COMPILEUDF function. No further actions are needed to publish the SAS 9.2 Formats Library.

For TwinFin systems, the SAS 9.2 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 9.2 Formats Library. The %INDNZ_PUBLISH_JAZLIB macro publishes and registers the SAS 9.2 Formats Library for Netezza in the database as the **sas_jazlib** object.

### %INDNZ_PUBLISH_JAZLIB Macro Run Process

To run the %INDNZ_PUBLISH_JAZLIB macro, start SAS 9.2 and submit the following commands in the Enhanced Editor or Program Editor:

```
%indnzpj;
%let indconn=SERVER=yourservername USER=youruserid PW=yourpwd DB=database;
%indnz_publish_jazlib(database=saslib, outdir=c:\fmtlib, action=create);
```

### %INDNZPJ Macro

The %INDNZPJ macro searches the autocall library for the indnzpj.sas file. The indnzpj.sas file contains all the macro definitions that are used in conjunction with the %INDNZ_PUBLISH_JAZLIB macro. The indnzpj.sas file should be in one of the directories listed in the SASAUTOS= system option in your configuration file. If the indnzpj.sas file is not present, the %INDNZPJ macro call (%INDNZPJ; statement) issues the following message:

```
macro indnzpj not defined
```

### 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.

**TIP** The INDCONN macro variable is not passed as an argument to the %INDNZ_PUBLISH_JAZLIB macro. This information can be concealed in your SAS job. For example, you can place it in an autoexec file and apply permissions to the file so others cannot access the user credentials.

Here is the syntax for the value of the INDCONN macro variable for the %INDNZ_PUBLISH_JAZLIB macro:

SERVER=<'>*twinfin-server*<'> USER=<'>*userid*<'> PASSWORD=<'>*password*<'> DATABASE=<'>*database*<'>

**SERVER=<'>*twinfin-server*<'>**
specifies the server name or IP address of the TwinFin 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, you must enclose it 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, you must enclose it in quotation marks.

**PASSWORD=<'>*password*<'>**
specifies the password that is associated with your Netezza user name. If the password contains spaces or nonalphanumeric characters, you must enclose it in quotation marks.

**Tip:** You can use only PASSWORD= or PW= for the password argument. Other aliases such as PASS= or PWD= are not supported and cause errors.

**DATABASE=<'>*database*<'>**
specifies the name of the database on the TwinFin server that contains the tables and views that you want to access. If the database name contains spaces or nonalphanumeric characters, you must enclose it in quotation marks.

### %INDNZ_PUBLISH_JAZLIB Macro Syntax

**%INDNZ_PUBLISH_JAZLIB(**

<DATABASE=*database*>

<, ACTION=CREATE | REPLACE | DROP>

<, OUTDIR=*diagnostic-output-directory*>

);

**Arguments**

**DATABASE=*database***
specifies the name of a Netezza database to which the SAS 9.2 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 9.2 Formats Library is **`sas_jazlib`**

**ACTION=CREATE | REPLACE | DROP**
specifies that the macro performs one of the following actions:

CREATE
creates a new SAS 9.2 Formats Library.

REPLACE
overwrites the current SAS 9.2 Formats Library, if a SAS 9.2 Formats Library by the same name is already registered, or creates a new SAS 9.2 Formats Library if one is not registered.

DROP
causes the SAS 9.2 Formats Library to be dropped from the Netezza database.

**Default:** CREATE

**Tip:** If the SAS 9.2 Formats Library was published previously and you specify ACTION=CREATE or REPLACE, no warning is issued. Also, if you specify ACTION=DROP and the SAS 9.2 Formats Library does not exist, no warning is 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 %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 using an appropriate compiler for the NPS or TwinFin system. After that, the scoring and format publishing macros register those object files with the NPS. For NPS systems, the SAS_COMPILEUDF function also creates a link to the SAS 9.2 Formats Library for Netezza. This link is needed for scoring and format publishing.
- 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 on the host. For more information about the file transfer process, see "Function Publishing Process in Netezza" on page 30.

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 for the SAS Accelerator Publishing Agent" on page 36.

### %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 34.

2. Start SAS 9.2 and submit the following commands in the Enhanced Editor or Program Editor.

```
%indnzpc;
%let indconn = server=yourserver user=youruserid password=yourpwd
   database=database;
%indnz_publish_compileudf(
   database=database,
   action=create,
   outdir=youroutputdirectory);
```

After the SAS_COMPILEUDF function is published, the model or format publishing macros can be run to publish the scoring model or format functions.

### %INDNZPC Macro

The %INDNZPC macro is an autocall library that initializes the %INDNZ_PUBLISH_COMPILEUDF macro.

### 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*<'>

**SERVER=<'>*server*<'>**
specifies the server name or IP address of the NPS or TwinFin 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, you must enclose it 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, you must enclose it in quotation marks.

**PASSWORD=<'>*password*<'>**
specifies the password that is associated with your Netezza user name. If the password contains spaces or nonalphanumeric characters, you must enclose it in quotation marks.

**Tip:** You can use only PASSWORD= or PW= for the password argument. Other aliases such as PASS= or PWD= are not supported and cause errors.

**DATABASE=SASLIB | <'>*database*<'>**
specifies the name of the database on the NPS or TwinFin server that contains the tables and views that you want to access. If the database name contains spaces or nonalphanumeric characters, you must enclose it in quotation marks.

**Default:** SASLIB

**Interaction:** 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 the "Documentation for Publishing SAS Scoring Models and Formats in Netezza" on page 38.

### %INDNZ_PUBLISH_COMPILEUDF Macro Syntax

**%INDNZ_PUBLISH_COMPILEUDF (**

<DATABASE=*database-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 "%INDNZ_PUBLISH_COMPILEUDF Macro Run Process" on page 34.

**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 or REPLACE, no warning is issued. Also, if you specify ACTION=DROP and the SAS_COMPILEUDF function does not exist, no warning is 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.

### Netezza Permissions for the SAS Accelerator Publishing Agent

There are two sets of permissions involved with the SAS Accelerator Publishing Agent software.

The first set of permissions is needed to publish the SAS 9.2 Formats Library for Netezza 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 9.2 Formats Library (**`sas_jazlib`** object) | System Administrator or Database Administrator<br>*Note:* 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 fmtlibpublisherid` |
| 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 compileudfpublisherid` |

The second set of permissions is needed by the person who runs the scoring publishing macro, %INDNZ_PUBLISH_MODEL, or the format publishing macro, %INDNZ_PUBLISH_FORMATS. 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 scoring model or format publishing 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 scoring model and format publisher and for each database that the scoring model and format publishing uses. Therefore, you might need to grant the above permissions multiple times. After the Netezza permissions are set appropriately, the scoring model or format 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 scoring or format publishing macro.

| Permission Needed | Authority Required to Grant Permission | Examples |
|---|---|---|
| EXECUTE permission for the SAS 9.2 Formats Library | System Administrator or Database Administrator<br><br>*Note:* 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_JAZLIB TO scoringorfmtpublisherid` |
| EXECUTE permission for the SAS_COMPILEUDF function | | `GRANT EXECUTE ON SAS_COMPILEUDF TO scoringorfmtpublisherid` |
| EXECUTE permission for the SAS_DIRECTORYUDF function | | `GRANT EXECUTE ON SAS_DIRECTORYUDF TO scoringorfmtpublisherid` |
| EXECUTE permission for the SAS_HEXTOTEXTUDF function | | `GRANT EXECUTE ON SAS_HEXTOTEXTUDF TO scoringorfmtpublisherid` |
| CREATE FUNCTION, CREATE TABLE, CREATE TEMP TABLE, and CREATE EXTERNAL TABLE permissions to run the scoring and format publishing macros | | `GRANT CREATE FUNCTION TO scoringorfmtpublisherid`<br><br>`GRANT CREATE TABLE TO scoringorfmtpublisherid`<br><br>`GRANT CREATE TEMP TABLE TO scoringorfmtpublisherid`<br><br>`GRANT CREATE EXTERNAL TABLE TO scoringorfmtpublisherid` |

## *Documentation for Publishing SAS Scoring Models and Formats in Netezza*

For information about how to publish SAS scoring models, see the *SAS Scoring Accelerator for Netezza: User's Guide* located at **http://support.sas.com/documentation/onlinedoc/scornet/index.html**.

For information about how to publish the SAS_PUT() function and formats, see "Deploying and Using SAS Formats in Netezza" in the *SAS/ACCESS 9.2 for Relational Databases: Reference* located at **http://support.sas.com/documentation/onlinedoc/access/index.html**.

*Chapter 6*

# Administrator's Guide for Teradata



## SAS Accelerator Publishing Agent for Teradata

### *Prerequisites*

SAS Foundation and the SAS/ACCESS Interface to Teradata must be installed before you install and configure the SAS Accelerator Publishing Agent for Teradata.

### *Overview of the SAS Accelerator Publishing Agent for Teradata*

The information in this section describes how to install and configure the SAS Accelerator Publishing Agent for Teradata. The SAS 9.2 Formats Library for Teradata is included with SAS Accelerator Publishing Agent for Teradata. This section also describes how to publish the SAS 9.2 Formats Library.

The SAS Accelerator Publishing Agent and the SAS 9.2 Formats Library are used by both the SAS/ACCESS Interface to Teradata and by the SAS Scoring Accelerator for Teradata.

- In SAS/ACCESS Interface to Teradata, a format publishing macro, %INDTD_PUBLISH_FORMATS, creates or publishes the SAS_PUT() function and formats inside the database.
- In the SAS Scoring Accelerator for Teradata, a scoring publishing macro, %INDTD_PUBLISH_MODEL, creates or publishes scoring model functions.

The SAS 9.2 Formats Library for Teradata is a run-time library that is installed on your Teradata system. This allows the SAS scoring model functions or the SAS_PUT() function created in Teradata to have access to routines within its run-time library.

*Note:* If you are performing a system expansion where additional nodes are being added, the version of the SAS 9.2 Formats Library on the new database nodes must be the same as the version that is being used on already existing nodes.

### Teradata Installation and Configuration Steps

1. If you are installing SAS Accelerator Publishing Agent for Teradata during a maintenance update, be sure to run the SAS Deployment Wizard twice as you upgrade to this latest version.

   The first time you run the SAS Deployment Wizard, you apply any maintenance for your software. You must run it a second time to install the update to SAS Accelerator Publishing Agent.

2. Apply Hot Fix B85002 for SAS Table Server Base Components 9.2_M2. This hot fix can be found at **`http://ftp.sas.com/techsup/dowload/hotfix/HF2/B85.html#B85002`**.

3. If your scoring model functions are very large and you are running Teradata V2R6 or 12 on the Linux operating system, install a Teradata patch.

   For more information, see "Increasing the Size of the Scoring Model Functions for Teradata on Linux" on page 40.

4. Install the SAS 9.2 Formats Library for Teradata.

   For more information, see "Installing the SAS 9.2 Formats Library for Teradata" on page 40.

## Increasing the Size of the Scoring Model Functions for Teradata on Linux

You can increase the maximum size of the scoring model functions for Teradata on the Linux operating system by installing a Teradata DBMS patch that adds the No Save (NS) source option to the CREATE/REPLACE FUNCTION statement. This task should be performed before loading the SAS 9.2 Formats Library for Teradata.

The patch is included in the following Teradata DBMS versions:

- Teradata V2R6 version 06.02.02.34 and later
- Teradata 12 version 12.00.01.07 and later

*Note:* This option is used automatically by the scoring model publishing macro if the Teradata system is running Linux and the patch has been applied; no user intervention is required.

For more information about this option, ask your Teradata support representative and reference Teradata DR 120829.

## Installing the SAS 9.2 Formats Library for Teradata

### Moving the SAS 9.2 Formats Library Package to the Server Machine

1. Locate the SAS 9.2 Formats Library package file, jazxfbrs-9.2-1.9.x86_64.rpm.

   The location of the rpm file, the README file, and possibly other files, is contained in the ordersummary.html file that accompanies the software. Your order e-mail contains the path to the ordersummary.html file.

2. Put the package file on your Teradata database server in a location where it is both read and write accessible.

   The package 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.

### *Installing the SAS 9.2 Formats Library with the Teradata Parallel Upgrade Tool*

Before you install the SAS 9.2 Formats Library package, it must be moved to your server machine. For more information, see "Moving the SAS 9.2 Formats Library Package to the Server Machine" on page 40.

This installation should be performed by a Teradata systems administrator. The steps assume some knowledge of the Teradata Parallel Upgrade Tool (PUT) and your environment. For more information about using the Teradata Parallel Upgrade Tool, see the *Parallel Upgrade Tool (PUT) for UNIX MPRAS and Linux User Guide*, located at **`http://www.info.teradata.com/GenSrch/eOnLine-Srch.cfm`**. Search on this page for "Parallel Upgrade Tool" and download the appropriate document for your system.

The following steps explain how to install the SAS 9.2 Formats Library package by using the Teradata Parallel Upgrade Tool.

*Note:* TDPut prompts are subject to change as Teradata enhances its software.

1. At the Teradata system, start the Teradata Parallel Upgrade Tool and log in to PUT as "root" (super-user).

   The **Welcome to the Parallel Upgrade Tool** dialog box appears.

2. Select **Install/Upgrade Software** from the task list and click **Next**.

   The **Configuration Mode** dialog box appears.

3. Select **Typical Configuration** and click **Next.**

   The **Network Subnet Selection** dialog box appears.

4. Select the correct address for the system to which you are applying from the **Subnet Addresses** list–typically the BYNET network. Click **Next**.

   The **Select Nodes** dialog box appears.

5. Highlight the node or nodes where the SAS 9.2 Formats Library are to be installed in the **Available Nodenames** list on the left.

   You should select all database (TPA) nodes, including any Hot-Stand By (HSN) nodes and any PE-only nodes.

6. Click the right-hand arrow to move your selection to the **Selected Nodename** window. When you have moved all appropriate nodes to the right, click **Next**.

   The **Select Spool Area** dialog box appears.

7. Read the description on the dialog box and take any steps that it describes to choose a location to spool the package on the Teradata nodes. Click **Next**.

   The **Enter Source for New Packages** dialog box appears.

8. Enter the path to the jazxfbrs-9.2-1.9.x86_64.rpm file in the text box. Click **Next**.

   The **Media Source Confirmation** dialog box appears.

9. Select the **No** radio button and the **Keep it** radio button. Click **Next**.

The **Media Source(s) to be Scanned** dialog box appears. This dialog box is a confirmation of the choices that you made in step 8.

10. Verify that the information is correct and click **Next**.

    A warning dialog box might appear. If it does, click **OK** to continue. The **Group Nodes** dialog box appears. You do not need to make any decisions for this dialog box. Note the groups that include all of the Teradata database (TPA) nodes in the Teradata system.

11. Click **Next**.

    The **Select Packages** dialog box appears. Only one package should be listed: jazxfbrs. The **Install on** column is empty.

12. Click the package. A window opens with the header **Modify package (jazxfbrs) version (9.2-1.9) OS type (Linux)**.

    a. Highlight all groups in the left-hand window that include TPA nodes, and then click the right arrow to move those groups to the right-hand window. Click **Next**.

       The **Select Packages** dialog box returns, but the **Install on** column has information in it.

    b. Click **Next**.

       The **Confirm Package Selections** dialog box appears.

13. Verify the information listed in the dialog box and click **Next**.

    Several dialog boxes that do not require input go by fairly quickly, depending on your processor speed. The Teradata Parallel Upgrade Tool is preparing to install. Eventually, the **Begin Installing Packages Now?** dialog box opens.

14. Click **Next** to begin the installation.

    After the package is installed, the **Overall Package Install Results** dialog box appears.

15. Click on the package to view the install status.

    Click **Back** to return to the overall package installation results. Several more dialog boxes that do not require input go by before the **Select Teradata Startup Option** dialog box appears.

16. Select the option that is most appropriate and click **Next**.

    The **Finished!** dialog box opens.

17. Click **Finish** and close the Teradata Parallel Upgrade Tool.

18. If you are performing an update install, reset the database so that all database processes have an opportunity to load the new version of the library.

### *Installing the SAS 9.2 Formats Library Manually*

#### Considerations

If you use the manual process to install the library on your TPA nodes, you must repeat the process for every TPA node in your deployment. This includes repeating the process for any Hot-Stand By (HSN) nodes and any PE-only nodes.

#### Manual Install Process for Teradata V2R6, Teradata 12, or Teradata 13 on Linux

In some cases such as a single-node system installed in a test environment, you might choose to not use the Teradata Parallel Upgrade Tool to install the SAS 9.2 Formats

Library (.rpm file). In those cases, you can manually load the .rpm file to your Teradata system. The manual install should be performed by a Teradata systems administrator.

The following steps explain how to install the SAS 9.2 Formats Library.

1. Log in to the Teradata system as "root" (super-user).
2. Make jazxfbrs-9.2-1.9.x86_64.rpm available to the Teradata system by using network share, CD-ROM, or copy to the file system.
3. [Optional] Copy the file jazxfbrs-9.2-1.9.x86_64.rpm to a package spool directory of your choice in order to make a permanent copy on the Teradata system.
4. Change the directory to the package location.
5. From the shell command prompt, enter the following command.

   ```
   rpm -U jazxfbrs-9.2-1.9.x86_64.rpm
   ```

   No output is displayed.

   If you want to remove the existing package, enter the following command.

   ```
   rpm -e jazxfbrs
   ```

6. To verify a successful installation, run the following command from the shell command prompt:

   ```
   rpm –q jazxfbrs
   ```

   The following line is displayed if the installation is successful:

   ```
   jazxfbrs-9.2-1.9
   ```

   If you want more information, you can use the following command, but the result provides more information than you need to verify your installation.

   ```
   rpm –qi jazxfbrs
   ```

7. If you are performing an update install, reset the database so that all database processes can load the new version of the library.

### *Teradata Permissions for the SAS Accelerator Publishing Agent*

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.

In addition, you must grant the following permissions to any user who runs the scoring or format publishing macros:

CREATE FUNCTION
DROP FUNCTION
EXECUTE FUNCTION
ALTER FUNCTION

### Documentation for Publishing SAS Scoring Models and Formats in Teradata

For information about how to publish SAS scoring models, see the *SAS Scoring Accelerator for Teradata: User's Guide* located at **http://support.sas.com/documentation/onlinedoc/scoraccl/index.html**.

For information about how to publish the SAS_PUT() function and formats, see "Deploying and Using SAS Formats in Teradata" in *SAS/ACCESS 9.2 for Relational Databases Reference* located at **http://support.sas.com/documentation/onlinedoc/access/index.html**.

# Index

## F



## G



## I



## L



## M

## N



## O



## P



## S

## T



## U



## V