THE POWER TO KNOW.

SAS® 9.3 In-Database Products
Administrator's Guide

SAS® Documentation

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2011. *SAS® 9.3 In-Database Products: Administrator's Guide*. Cary, NC: SAS Institute Inc.

SAS® 9.3 In-Database Products: Administrator's Guide

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

1st electronic book, July 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

What's New in SAS 9.3 In-Database Products: Administrator's Guide . . . . . *v*
Recommended Reading . . . . . *vii*

Chapter 1 • Introduction to the Administrator's Guide **1**
SAS In-Database Products 1
What Is Covered in This Document? 1

Chapter 2 • Administrator's Guide for Aster nCluster **3**
In-Database Deployment Package for Aster nCluster 3

Chapter 3 • Administrator's Guide for DB2 . . . . . **7**
In-Database Deployment Package for DB2 7

Chapter 4 • Administrator's Guide to Greenplum **21**
In-Database Deployment Package for Greenplum 21

Chapter 5 • Administrator's Guide for Netezza **29**
In-Database Deployment Package for Netezza 29

Chapter 6 • Administrator's Guide for Teradata **39**
In-Database Deployment Package for Teradata 39

Chapter 7 • Configurations for SAS Model Manager **43**
Preparing a Database for Use with SAS Model Manager 43

Index **47**

What's New in SAS 9.3 In-Database Products: Administrator's Guide

Overview

The *SAS In-Database Products: User's Guide* has the following changes and enhancements:

- All in-database publishing macros are compiled now for better security.
- Support for Teradata V2R6 on Linux and Netezza Performance Systems has been dropped.
- Configuration instructions for the SAS Model Manager In-Database Scoring Scripts product have been moved to this book from the *SAS Model Manager: User's Guide*.

Compiled Macros

In SAS 9.3, all publishing macros are compiled for better security. There is no change in the way you run the publishing macros.

Database Systems No Longer Supported

For Teradata, V2R6 on Linux is no longer supported.

For Netezza, the Netezza Performance System (NPS) is no longer supported.

Additional Alias for INDCONN Macro Password Argument

You can now use PASS= for the password argument in the INDCONN macro variable.

Configuration Information for SAS Model Manager In-Database Scoring Scripts

The configuration instructions for the SAS Model Manager In-Database Scoring Scripts product have been moved to this book from the *SAS Model Manager: User's Guide*.

Recommended Reading

Here is the recommended reading list for this title:

- *SAS/ACCESS for Relational Databases: Reference*
- *SAS In-Database Products: User's Guide*
- *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 1

What Is Covered in This Document? 1

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, some SAS 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 In-Database Scoring Scripts

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.

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

- SAS Embedded Process for Aster *n*Cluster, Version 9.3
- SAS Formats Library for DB2, Version 2.1
- SAS Formats Library for Greenplum, Version 2.1
- SAS Formats Library for Netezza, Version 2.1
- SAS Formats Library for Teradata, Version 2.1

Additional configuration tasks are needed if you want to use SAS Model Manager for in-database scoring with DB2, Netezza, or Teradata. This document provides detailed instructions for configuring a database for use with SAS Model Manager.

Note: Administrative tasks for the SAS Analytics Accelerator are currently in the *SAS Analytics Accelerator for Teradata: User's Guide*.

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 nCluster **3**
Prerequisites 3
Overview of the In-Database Deployment Package for Aster nCluster 3
Aster nCluster Installation and Configuration Steps 4
Upgrading from Previous Versions 4
Installing the In-Database Deployment Package Binary Files for Aster nCluster . . . 4
Validating the Publishing of the SAS_SCORE() Function 6
Aster nCluster Permissions 6
Documentation for Publishing SAS Scoring Models in Aster nCluster 6

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

This section describes how to install and configure the in-database deployment package for Aster *n*Cluster (SAS Embedded Process 9.3).

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

The scoring publishing macro is included in the SAS/ACCESS Interface to Aster *n*Cluster. For more information about using the scoring publishing macro, see the *SAS In-Database Products: User's Guide*.

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 so that the SAS scoring files created in Aster *n*Cluster can access the routines within its run-time library.

Aster nCluster Installation and Configuration Steps

1. If you are upgrading from a previous release or installing a maintenance release, follow the instructions in "Upgrading from Previous Versions" on page 4 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 nCluster" on page 4.

Upgrading from Previous Versions

Follow these steps to upgrade from a previous release.

1. Log in 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 directory exists in the partner directory, enter this command to remove an existing installation from the queen.

   ```
   rm -rf SAS
   ```

   If you want to perform a clean install, you will also want to enter these command to remove the SAS directory from all the workers.

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

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 tkindbsrv-9.3-1_lax.sh. The TAR file is located in the ***SAS-install-directory*/SASTKInDatabaseServer/9.3/AsternClusteronLinuxx64/** directory.

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 in which to put tkindbsrv-9.3-1_lax.sh, execute tkindbsrv-9.3-1_lax.sh,. Then install the SAS_SCORE SQL/MR 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.

```
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.

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

   b. At the SFTP prompt, enter this command to transfer the TAR file.

      ```
      put tkindbsrv-9.3-1_lax.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  tkindbsrv-9.3-1_lax.sh
   ```

7. Unpack the TAR file in the partner directory.

   ```
   ./tkindbsrv-9.3-1_lax.sh
   ```

8. Change to the directory where SAS is installed.

   ```
   cd /home/beehive/partner/SAS/SASTKInDatabaseServerForAster/9.3-1/sasexe
   ```

9. 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
      \remove sas_put.tk.so
      \remove sas_row.tk.so
      \remove sas_partition.tk.so
      ```

   c. Enter the following command 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
      ```

10. Exit the ACT tool.

    ```
    \q
    ```

11. Verify the existence and current date of the tkast-runInCluster and tkeastrmr.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/SAS/SASTKInDatabaseServerForAster/9.3-1 |
      grep tkeastrmr.so"; \
   ssh $ip "ls -al /home/beehive/partner/SAS/SASTKInDatabaseServerForAster/9.3-1
      /utilities/bin/ | grep tkast-runInCluster"; \
done
```

Validating the Publishing of the SAS_SCORE() Function

To validate that the SAS_SCORE() function was installed, run the **\dF** command in the Aster nCluster Client or through 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 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.

For Aster nCluster 4.5, no permissions are needed by the person who runs the scoring publishing macros, because all functions and files are published to the PUBLIC schema.

For Aster nCluster 4.6, the following schema permissions are needed by the person who runs the scoring 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 *yourschemaname* TO *youruserid*

Documentation for Publishing SAS Scoring Models in Aster nCluster

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 3

Administrator's Guide for DB2

In-Database Deployment Package for DB2 7
Prerequisites 7
Overview of the In-Database Deployment Package for DB2 7
Function Publishing Process in DB2 8
DB2 Installation and Configuration Steps 8
Installing the SAS Formats Library and Binary Files 9
Running the %INDB2_PUBLISH_COMPILEUDF Macro 10
Running the %INDB2_PUBLISH_DELETEUDF Macro 14
Validating the Publishing of SAS_COMPILEUDF and SAS_DELETEUDF Functions and Global Variables 17
DB2 Permissions 18
Documentation for Publishing SAS Formats or Scoring Models in DB2 19

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.

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 2.1).

The in-database deployment package for DB2 must be installed and configured before you can perform the following tasks:

- use the %INDB2_PUBLISH_FORMATSformat 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.

The format and scoring publishing macros are included in SAS/ACCESS Interface to DB2. 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 publishing macros.

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 its run-time library.

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.

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*, located at **`http://support.sas.com/techsup/technote/ts800.pdf`**.

DB2 Installation and Configuration Steps

1. 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
   ```

2. Install the SAS formats library and the binary files for the SAS_COMPILEUDF and SAS_DELETEUDF functions.

   For more information, see "Installing the SAS Formats Library and Binary Files" on page 9.

3. 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.

4. 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.

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 7, "Configurations for SAS Model Manager," on page 43.

Installing the SAS Formats Library and Binary Files

Move the Files to DB2

The SAS formats library and the binary files for the SAS_COMPILEUDF and SAS_DELETEUDF functions are contained in a self-extracting TAR file. The TAR file is located in the ***SAS-install-directory*/SASFormatsLibraryforDB2/2.1/DB2on<AIX | Linux64>/** directory.

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:

AIX: acceldb2fmt-2.1-1_r64.sh

Linux(x86_64): acceldb2fmt-2.1-1_lax.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. You must have the appropriate permissions to execute the script and write to the directory.

   ```
   $ cd path_to_tar_file
   $ ./tar_file
   ```

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

   tar_file is either acceldb2fmt-2.1-1_lax.sh or acceldb2fmt-2.1-1_r64.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/SASFormatsLibraryForDB2/2.1-1/bin/
      InstallAccelDB2Fmt.sh
   ```

```
/path_to_tar_file/SAS/SASFormatsLibraryForDB2/2.1-1/bin/CopySASFiles.sh

/path_to_tar_file/SAS/SASFormatsLibraryForDB2/2.1-1/lib/SAS_CompileUDF

/path_to_tar_file/SAS/SASFormatsLibraryForDB2/2.1-1/lib/SAS_DeleteUDF

/path_to_tar_file/SAS/SASFormatsLibraryForDB2/2.1-1/lib/libjazxfbrs.so

/path_to_tar_file/SAS/SASFormatsLibraryForDB2/2.1 -> 2.1-1
```

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

   ```
   $ path_to_tar_file/SAS/SASFormatsLibraryForDB2/2.1-1/bin/
      CopySASFiles.sh db2path/sqllib
   ```

   db2path/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.

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

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

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.

   ```
   $ db2path/sqllib . ./db2profile
   ```

   Now, run the DB2SET command.

   ```
   $ db2set DB2LIBPATH=db2path/sqllib/function/SAS
   ```

   db2path/sqllib is the path to the **sqllib** directory of the DB2 instance that you want to use.

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

   ```
   $ db2path/sqllib/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_FORMATS format publishing macro and the %INDB2_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 that is needed for format and scoring model publishing.
  - copies the object files to the ***db2path*/sqllib/function/SAS** directory. You specify the value of **db2path** 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 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.

   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 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 9.3 and submit the following commands in the Enhanced Editor or Program Editor:

   ```
   %indb2pc;
   %let indconn = server=yourserver user=youruserid password=yourpwd
      database=yourdb schema=saslib;
   ```

   For more information, see "%INDB2PC Macro" on page 12 and "INDCONN Macro Variable" on page 12.

3. Run the %INDB2_PUBLISH_COMPILEUDF macro. For more information, see "%INDB2_PUBLISH_COMPILEUDF Macro Syntax" on page 13.

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* <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, 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=, 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, you must enclose it in quotation marks.

Requirement: The SAS_COMPILEUDF function is created as a Unicode function. If the database is not a Unicode database, then the alternate collating sequence must be configured to use `identity_16bit`.

SCHEMA=SASLIB
specifies SASLIB as the schema name.

Default: SASLIB

Restriction: The SAS_COMPILEUDF function and the two global variables (SASUDF_DB2PATH and SASUDF_COMPILER_PATH) are published to the SASLIB schema in the specified database. If a value other than SASLIB is used, it will be ignored.

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

%INDB2_PUBLISH_COMPILEUDF Macro Syntax

%INDB2_PUBLISH_COMPILEUDF(

DB2PATH=*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 format and 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 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 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.

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_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 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.

The SASLIB schema is used when publishing the %INDB2_PUBLISH_DELETEUDF macro for DB2 in-database processing.

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 9.3 and submit the following commands in the Enhanced Editor or Program Editor.

```
%indb2pd;
%let indconn = server=yourserver user=youruserid password=yourpwd
   database=yourdb schema=saslib;
```

   For more information, see "%INDB2PD Macro" on page 15 and "INDCONN Macro Variable" on page 15.

3. Run the %INDB2_PUBLISH_DELETEUDF macro. For more information, see "%INDB2_PUBLISH_DELETEUDF Macro Syntax" on page 16.

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_FORMATS and the %INDB2_PUBLISH_MODEL macros can be run to publish the format and scoring model functions.

%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* <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, 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=, 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, you must enclose the 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 will be ignored.

Requirement: 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.

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 **`/users/db2v9/sqllib`** is where 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 db2path/sqllib/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

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 *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 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.

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 *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 scoringorfmtpublisherid`
CREATE_EXTERNAL_ROUTINE permission to the database to create format or scoring model functions		`GRANT CREATE_EXTERNAL_ROUTINE ON DATABASE TO scoringorfmtpublisherid`
CREATE_NOT_FENCED_ROUTINE permission to create format or scoring model functions that are not fenced		`GRANT CREATE_NOT_FENCED_ROUTINE ON DATABASE TO scoringorfmtpublisherid`
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 scoringschema TO scoringorfmtpublisherid`
READ permission to read the SASUDF_COMPILER_PATH and SASUDF_DB2PATH global variables *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 *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 scoringorfmtpublisherid` `GRANT READ ON VARIABLE SASLIB.SASUDF_COMPILER_PATH TO scoringorfmtpublisherid`

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 7, "Configurations for SAS Model Manager," on page 43.

Documentation for Publishing SAS Formats or Scoring Models 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 4

Administrator's Guide to Greenplum

In-Database Deployment Package for Greenplum **21**
Prerequisites 21
Overview of the In-Database Deployment Package for Greenplum 21
Function Publishing Process in Greenplum 22
Greenplum Installation and Configuration Steps 22
Moving and Unpacking the SAS Formats Library and Binary Files 22
Running the %INDGP_PUBLISH_COMPILEUDF Macro 24
Validating the Publishing of the SAS_COMPILEUDF, SAS_COPYUDF, SAS_DIRECTORYUDF, and SAS_DEHEXUDF Functions 28
Greenplum Permissions 28
Documentation for Publishing SAS Scoring Models in Greenplum 28

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.

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 2.1).

The in-database deployment package for Greenplum must be installed and configured before you can use the %INDGP_PUBLISH_MODEL scoring publishing macro to create scoring model functions inside the database.

The scoring publishing macro is included in the SAS/ACCESS Interface to Greenplum. For more information about using the scoring publishing macro, 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 a publishing macro.

The SAS formats library 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 %INDGP_PUBLISH_COMPILEUDF macro 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 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. Move and unpack the SAS formats library and binary files for the publishing macro.

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

2. Run the %INDGP_PUBLISH_COMPILEUDF macro.

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

Moving and Unpacking the SAS Formats Library and Binary Files

The SAS formats library and the binary files for the publishing macro are contained in a self-extracting TAR file. The TAR file is located in the ***`SAS-install-directory`*`/SASFormatsLibraryforGreenplum/2.1/GreenplumonLinux64/`** directory.

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

1. Using a method of your choice, transfer the accelgplmfmt-2.1-1_lax.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.

2. After the accelgplmfmt-2.1-1_lax.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-2.1-1.sh
   ```

   Note: If you receive a "permissions denied" message, check the permissions on the accelgplmfmt-2.1-1_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_tar_file* is the location to which you copied the TAR file.

   ```
   /path_to_tar_file/SAS/SASFormatsLibraryForGreenplum/2.1-1/bin/
       InstallAccelGplmFmt.sh
   /path_to_tar_file/SAS/SASFormatsLibraryForGreenplum/2.1-1/bin/
       CopySASFiles.sh
   /path_to_tar_file/SAS/SASFormatsLibraryForGreenplum/2.1-1/lib/
       SAS_CompileUDF.so
   /path_to_tar_file/SAS/SASFormatsLibraryForGreenplum/2.1-1/lib/
       libjazxfbrs.so
   ```

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

   ```
   ./path_to_tar_file/SAS/SASFormatsLibraryForGreenplum/2.1-1/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 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, 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 follow these steps from the master node to create the symbolic links to the SAS formats library for Greenplum (libjazfbrs.so).

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 formats library 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.

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

- 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.

  - 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 exported 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 9.3 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 */
   ```

   For more information, see "%INDGPPC Macro" on page 25 and "INDCONN Macro Variable" on page 26.

3. Run the %INDGP_PUBLISH_COMPILEUDF macro. For more information, 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=, 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, 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.

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.

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=*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

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.

Documentation for Publishing SAS Scoring Models in Greenplum

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 5

Administrator's Guide for Netezza

In-Database Deployment Package for Netezza **29**
Prerequisites 29
Overview of the In-Database Deployment Package for Netezza 29
Function Publishing Process in Netezza 30
Netezza Installation and Configuration Steps 30
Moving and Unpacking the SAS Formats Library and Binary Files 30
Running the %INDNZ_PUBLISH_JAZLIB Macro 31
Running the %INDNZ_PUBLISH_COMPILEUDF Macro 33
Documentation for Publishing SAS Formats and Scoring Models in Netezza 37

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.

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 2.1).

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.

The format and scoring publishing macros are included in SAS/ACCESS Interface to Netezza. 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 and two additional publishing macros.

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

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.

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.

Netezza Installation and Configuration Steps

1. Move and unpack the SAS formats library and binary files for the SAS_COMPILEUDF function.

   For more information, see "Moving and Unpacking the SAS Formats Library and Binary Files" on page 30.
2. 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 31.
3. Run the %INDNZ_PUBLISH_COMPILEUDF macro.

   For more information, see"Running the %INDNZ_PUBLISH_COMPILEUDF Macro" on page 33.
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 7, "Configurations for SAS Model Manager," on page 43.

Moving and Unpacking 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 TAR file. The TAR file is located in the ***SAS-install-***

***directory*/SASFormatsLibraryforNetezza/2.1/Netezza32bitTwinFin/** directory.

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

1. Using a method of your choice, transfer the accelnetzfmt-2.1-1_lax.sh to your Netezza system.
2. After the accelnetzfmt-2.1-1_lax.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-2.1-1_lax.sh
   path_to_self-extracting_tar_file/accelnetzfmt-2.1-1_lax.sh
   ```

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

   ```
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/2.1-1/bin/InstallAccelNetzFmt.sh
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/2.1-1/lib/SAS_CompileUDF.o_diab_ppc
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/2.1-1/lib/SAS_CompileUDF.o_x86
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/2.1-1/lib/libjazxfbrs_diab_ppc.a
   /nz/extensions/SAS/SASFormatsLibraryForNetezza/2.1-1/lib/libjazxfbrs_x86.a
   ```

   There also is a symbolic link such that **/nz/extensions/SAS/SASFormatsLibraryForNetezza/2.1** points to the latest version.

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 9.3 and submit the following commands in the Enhanced Editor or Program Editor:

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

   For more information, see "%INDNZPJ Macro" on page 32 and "INDCONN Macro Variable" on page 32.

2. Run the %INDNZ_PUBLISH_JAZLIB macro. For more information, see "%INDNZ_PUBLISH_JAZLIB Macro Syntax" on page 32.

%INDNZPJ Macro

The %INDNZPJ macro searches the autocall library for the indnzpj.sas file. The indnzpj.sas file needs to be called before calling 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=<'>*server*<'> USER=<'>*userid*<'> PASSWORD=<'>*password*<'> DATABASE=<'>*database*<'>

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, 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=, 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, 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 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`**

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 will receive warning messages that the library already exists and be prompted to use REPLACE. If you specify ACTION=DROP and the SAS formats library does not exist, you will 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 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" 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 9.3 and submit the following commands in the Enhanced Editor or Program Editor.

   ```
   %indnzpc;
   %let indconn = server=yourserver user=youruserid password=yourpwd
      database=database;
   ```

   For more information, see "%INDNZPC Macro" on page 34 and "INDCONN Macro Variable" on page 34.

3. Run the %INDNZ_PUBLISH_COMPILEUDF macro. For more information, see "%INDNZ_PUBLISH_COMPILEUDF Macro Syntax" on page 35.

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 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=, 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, 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 Formats and Scoring Models in Netezza" on page 37.

%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, you will 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 will 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 two 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 (**`sas_jazlib`** object)	System Administrator or Database Administrator *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 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 the above 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 9.3 Formats Library	System Administrator or Database Administrator *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 format and scoring publishing macros		`GRANT CREATE FUNCTION TO` *`scoringorfmtpublisherid`* `GRANT CREATE TABLE TO` *`scoringorfmtpublisherid`* `GRANT CREATE TEMP TABLE TO` *`scoringorfmtpublisherid`* `GRANT CREATE EXTERNAL TABLE TO` *`scoringorfmtpublisherid`*

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 7, “Configurations for SAS Model Manager,” on page 43.

Documentation for Publishing SAS Formats and Scoring Models 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 6

Administrator's Guide for Teradata

In-Database Deployment Package for Teradata **39**
Prerequisites 39
Overview of the In-Database Deployment Package for Teradata 39
Teradata Installation and Configuration Steps 40
Upgrading from Previous Versions 40
Moving and Installing the SAS Formats Library 41
Teradata Permissions 42
Documentation for Publishing SAS Scoring Models and Formats in Teradata 42

In-Database Deployment Package for Teradata

Prerequisites

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

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 2.1).

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

- use the %INDTD_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 %INDTD_PUBLISH_MODEL scoring publishing macro to create or publish scoring model functions inside the database.

The format and scoring publishing macros are included in SAS/ACCESS Interface to Teradata. 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 Teradata contains the SAS formats library.

The SAS formats library is a run-time library that is installed on your Teradata system so that the SAS scoring model functions or the SAS_PUT() function created in Teradata can access the 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 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 the in-database deployment package for Teradata during a maintenance update, 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 the in-database deployment package.

2. If you are upgrading from a previous version, follow the instructions in "Upgrading from Previous Versions" on page 40.

3. Install the in-database deployment package.

   For more information, see "Moving and Installing the SAS Formats Library" on page 41.

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 7, "Configurations for SAS Model Manager," on page 43.

Upgrading from Previous Versions

To upgrade the SAS formats library from a previous version:

1. 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.

2. Confirm that the database is shut down.

   ```
   pdestate -a
   ```

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

3. Check the current installed version of the SAS formats library.

   a. 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.

      ```
      jazxfbrs-9.2-1.6
      ```

   b. If a SAS 9.3 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.

      ```
      accelterafmt-2.1-1
      ```

If the library is not installed on the Teradata nodes, no output is displayed.

4. If a version of the SAS formats library with the same name (for example, **jazxfbrs** or **accelterfmt**), was previously installed, then remove the old version before you install the updated version of the in-database deployment package.

   Remove the old version.

   - To remove the SAS formats library from all nodes concurrently, run this command:

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

     For example, to remove **jazxfbrs**, run the command **psh "rpm –e jazxfbrs"**.

   - To remove the SAS formats library from each node, run this command on each node:

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

     package-name is either jazxfbrs.9.*version* or accelterafmt-2.*version*.

5. (Optional) To confirm removal of the old SAS formats library before installing the new formats library, run this command on all nodes:

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

6. Restart the database.

Moving and Installing the SAS Formats Library

Moving the SAS Formats Library Package to the Server Machine

1. Locate the in-database deployment package file, accelterfmt-2.1-1.x86_64.rpm.

   The RPM file is located in the ***SAS-install-directory*/SASFormatsLibraryforTeradata/2.1/TeradataonLinux/** directory.

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 Formats Library with the Teradata Parallel Upgrade Tool

This installation should be performed by a Teradata systems administrator. The steps assume some knowledge of the Teradata Parallel Upgrade Tool (TDPut) and your environment. For more information about using the Teradata Parallel Upgrade Tool, see the *Parallel Upgrade Tool (PUT) Reference Release 3.05.01B035–5713–011K January 2011*, located at **http://www.info.teradata.com/edownload.cfm?itemid=110550001**. Search on this page 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: TDPut prompts are subject to change as Teradata enhances its software.

1. Move the SAS Formats Library package to your server machine where it can be accessed from at least one of the Teradata nodes.For more information, see "Moving the SAS Formats Library Package to the Server Machine" on page 41.
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 or not. If you are prompted, choose a Non-VM&F installation.
4. Verify that the install is successful. Run this command from the shell command prompt.

   ```
   psh "rpm -q accelterafmt"
   ```

   If the install is successful, *accelterfmt-2.1-1* is displayed.
5. Restart the server so that all database processes can load the new version of the library.

   ```
   /etx/init.d/tpa start
   ```
6. Verify that the correct version of the SAS formats library is installed and active on all nodes.

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

   You should see a line with *accelterafmt-2.1-1* displayed for each Teradata node.

Teradata Permissions

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

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 7, "Configurations for SAS Model Manager," on page 43.

Documentation for Publishing SAS Scoring Models and Formats in Teradata

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 7

Configurations for SAS Model Manager

Preparing a Database for Use with SAS Model Manager **43**
Prerequisites 43
Overview of Preparing a Database for Use with SAS Model Manager 43
Configuring a Database 44
Finding the JDBC JAR Files 45

Preparing a Database for Use with SAS Model Manager

Prerequisites

SAS Foundation, the SAS/ACCESS Interface, and the in-database deployment package for the database must be installed and configured, before you can prepare a database for use with SAS Model Manager. For more information, see the chapter for your type of database in this guide. Here are the databases that can be used with SAS Model Manager:

- DB2
- Netezza
- Teradata

Overview of Preparing a Database for Use with SAS Model Manager

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. Additional configuration steps are required to prepare the database for publishing and scoring in SAS Model Manager.

During the installation and configuration of SAS 9.3 products, the SAS Model Manager In-Database Scoring Scripts product is installed on the middle-tier server.

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`**

In the script installation directory, includes a directory that specifies the version of SAS Model Manager, which is currently 3.1. 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.

Configuring a Database

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.
   a. GRANT CREATE, DROP, EXECUTE, and ALTER permissions for functions and procedures. For more information about permissions for the specific database, see the following topics:
      - "DB2 Permissions" on page 18
      - "Netezza Permissions" on page 36
      - "Teradata Permissions" on page 42
   b. GRANT SELECT, INSERT, UPDATE, and DELETE permissions for SAS Model Manager metadata tables.
   c. GRANT SELECT permission for the following views to validate the scoring function names:
      - syscat.functions for DB2
      - 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 those tables or views.
3. Navigate to the ***sasinstalldir*\SASModelManagerInDatabaseScoringScripts\3.1\Utilities** directory to find the Create Tables and Drop Tables scripts for your database, and then perform the following 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:
      - Teradata SQL scripts: createTablesTD.sql and dropTablesTD.sql
      - DB2 SQL scripts: createTablesDB2.sql and dropTablesDB2.sql
      - Netezza SQL scripts: createTablesNetezza.sql and dropTablesNetezza.sql
   b. Execute the Create Tables script for a specific type of database.
4. 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 Web application servers are the following:

   JBOSS
   ***JBoss_Home*\server\SASServer1\lib**

An example of the directory path is the following: **C:\JBoss4.3.0.GA\server\SASServer1\lib**

WebLogic

***Weblogic_HOME*\server\lib**

An example of the directory path is the following: **C:\bea10.3.2\wlserver_10.3\server\lib**

WebSphere

***WebSphere_HOME*\lib**

An example of the directory path is the following: **C:\Program Files\IBM\WebSphere7\AppServer\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 45.

Finding the JDBC JAR Files

The Teradata JDBC JAR files are **terajdbc4.jar** and **tdgssconfig.jar**. The Teradata JDBC JAR files can be found on the Teradata Web site at http://www.teradata.com. Select **Support & Downloads** ⇨ **Downloads** ⇨ **Teradata JDBC Driver**.

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 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**.

For more information about the database versions that are supported, see the **SAS Scoring Accelerator System Requirements**.

Index

Special Characters

%INDAC_PUBLISH_MODEL macro 3
%INDB2_PUBLISH_COMPILEUDF macro 10
  running 11
  syntax 13
%INDB2_PUBLISH_DELETEUDF macro 14
  running 14
  syntax 16
%INDB2_PUBLISH_FORMATS 7
%INDB2_PUBLISH_MODEL macro 7
%INDB2PC macro 12
%INDB2PD macro 15
%INDGP_PUBLISH_COMPILEUDF macro 24
  running 25
  syntax 27
%INDGP_PUBLISH_MODEL macro 21
%INDGPPC macro 25
%INDNZ_PUBLISH_COMPILEUDF macro 33
  running 34
  syntax 35
%INDNZ_PUBLISH_FORMATS macro 29
%INDNZ_PUBLISH_JAZLIB macro 31
  running 31
  syntax 32
%INDNZ_PUBLISH_MODEL macro 29
%INDNZPC macro 34
%INDNZPJ macro 32
%INDTD_PUBLISH_FORMATS macro 39
%INDTD_PUBLISH_MODEL macro 39

A

ACTION= argument
  %INDB2_PUBLISH_COMPILEUDF macro 13
  %INDB2_PUBLISH_DELETEUDF macro 16
  %INDGP_PUBLISH_COMPILEUDF macro 27
  %INDNZ_PUBLISH_COMPILEUDF macro 35
  %INDNZ_PUBLISH_JAZLIB macro 33
Aster nCluster
  documentation for publishing scoring models 6
  in-database deployment package 3
  installation and configuration 4
  permissions 6
  SAS/ACCESS Interface 3

B

binary files
  for Aster nCluster 4
  for DB2 functions 9
  for Greenplum functions 22
  for Netezza functions 30

C

COMPILER_PATH= argument
  %INDB2_PUBLISH_COMPILEUDF macro 13
configuration
  Aster nCluster 3
  DB2 8
  Greenplum 22
  Netezza 30
  Teradata 40

D

DATABASE= argument
  %INDB2_PUBLISH_COMPILEUDF macro 13

%INDB2_PUBLISH_DELETEUDF macro 16
%INDGP_PUBLISH_COMPILEUDF macro 27
%INDNZ_PUBLISH_COMPILEUDF macro 35
%INDNZ_PUBLISH_JAZLIB macro 33
DB2
creating tables 44
documentation for publishing formats or scoring models 19
function publishing process 8
in-database deployment package 7
installation and configuration 8
JDBC Driver 45
permissions 18
preparing for SAS Model Manager use 43
SAS/ACCESS Interface 7
unpacking TAR files 9
DB2PATH= argument
%INDB2_PUBLISH_COMPILEUDF macro 13
DB2SET command 10
documentation
for publishing formats and scoring models in DB2 19
for publishing formats and scoring models in Netezza 37
for publishing formats and scoring models in Teradata 42
for publishing scoring models in Aster nCluster 6
for publishing scoring models in Greenplum 28

F
formats library
DB2 installation 9
Grenplum installation 22
Netezza installation 30
Teradata installation 41
function publishing process
DB2 8
Greenplum 22
Netezza 30
functions
SAS_COMPILEUDF (DB2) 9, 11, 17
SAS_COMPILEUDF (Greenplum) 22, 24, 28
SAS_COMPILEUDF (Netezza) 30, 33
SAS_COPYUDF (Greenplum) 28
SAS_DEHEXUDF (Greenplum) 28
SAS_DELETEUDF (DB2) 9, 14, 17
SAS_DIRECTORYUDF (Greenplum) 28
SAS_DIRECTORYUDF (Netezza) 33
SAS_HEXTOTEXTUDF (Netezza) 33
SAS_PUT() (Netezza) 30
SAS_PUT() (Teradata) 39
SAS_SCORE() (Aster nCluster) 4

G
global variables
See variables
Greenplum
documentation for publishing scoring models 28
function publishing process 22
in-database deployment package 21
installation and configuration 22
permissions 28
SAS/ACCESS Interface 21
unpacking TAR files 22

I
in-database deployment package for Aster nCluster
overview 3
prerequisites 3
in-database deployment package for DB2
overview 7
prerequisites 7
in-database deployment package for Greenplum
overview 21
prerequisites 21
in-database deployment package for Netezza
overview 29
prerequisites 29
in-database deployment package for Teradata
overview 39
prerequisites 39
INDCONN macro variable 12, 15, 26, 32, 34
installation
Aster nCluster 3
DB2 8
Greenplum 22
Netezza 30
SAS formats library 9, 22, 30, 41
Teradata 40

J
JDBC Driver for DB2 45

JDBC Driver for Netezza 45
JDBC Driver for Teradata 45

M

macro variables
  See variables
macros
  %INDB2_PUBLISH_COMPILEUDF 11, 13
  %INDB2_PUBLISH_DELETEUDF 14, 16
  %INDB2_PUBLISH_FORMATS 7
  %INDB2_PUBLISH_MODEL 7
  %INDB2PC 12
  %INDB2PD 15
  %INDGP_PUBLISH_COMPILEUDF 24, 27
  %INDGP_PUBLISH_MODEL 21
  %INDGPPC 25
  %INDNZ_PUBLISH_COMPILEUDF 33, 35
  %INDNZ_PUBLISH_FORMATS 29
  %INDNZ_PUBLISH_JAZLIB 31, 32
  %INDNZ_PUBLISH_MODEL 29
  %INDNZPC 34
  %INDNZPJ 32
  %INDTD_PUBLISH_FORMATS 39
  %INDTD_PUBLISH_MODEL 39

N

Netezza
  creating tables 44
  documentation for publishing formats and scoring models 37
  function publishing process 30
  in-database deployment package 29
  installation and configuration 30
  JDBC Driver 45
  permissions 36
  preparing for SAS Model Manager use 43
  publishing SAS formats library 31
  SAS/ACCESS Interface 29
  unpacking TAR files 30

O

OBJNAME= argument
  %INDB2_PUBLISH_COMPILEUDF macro 14
OBJPATH= argument
  %INDGP_PUBLISH_COMPILEUDF macro 27
OUTDIR= argument
  %INDB2_PUBLISH_COMPILEUDF macro 14
  %INDB2_PUBLISH_DELETEUDF macro 17
  %INDGP_PUBLISH_COMPILEUDF macro 27
  %INDNZ_PUBLISH_COMPILEUDF macro 36
  %INDNZ_PUBLISH_JAZLIB macro 33

P

permissions
  for Aster nCluster 6
  for DB2 18
  for Greenplum 28
  for Netezza 36
  for Teradata 42
PSFTP 8
publishing
  Aster nCluster permissions 6
  DB2 permissions 18
  functions in DB2 8
  functions in Greenplum 22
  functions in Netezza 30
  Greenplum permissions 28
  Netezza permissions 36
  SAS formats library (Netezza) 31
  Teradata permissions 42

R

rpm file for Teradata 41

S

SAS_COMPILEUDF function
  actions for DB2 11
  actions for Greenplum 24
  actions for Netezza 33
  binary files for DB2 9
  binary files for Greenplum 22
  binary files for Netezza 30
  validating publication for DB2 17
  validating publication for Greenplum 28
SAS_COPYUDF function 24
  validating publication for Greenplum 28
SAS_DEHEXUDF function 24
  validating publication for Greenplum 28
SAS_DELETEUDF function
  actions for DB2 14
  binary files for DB2 9

validating publication for DB2 17
SAS_DIRECTORYUDF function 24, 33
validating publication for Greenplum 28
SAS_HEXTOTEXTUDF function 33
SAS_PUT() function
DB2 8
Netezza 30
Teradata 39
SAS_SCORE() function
publishing 4
validating publication for Aster nCluster 6
SAS Deployment Wizard 40
SAS FILENAME SFTP statement 8
SAS formats library
DB2 9
Greenplum 22
installing with Teradata Parallel Upgrade Tool 41
Netezza 30, 31
Teradata 41
upgrading from previous versions (Teradata) 40
SAS Foundation 3, 7, 21, 29, 39
SAS In-Database products 1
SAS/ACCESS Interface to Aster nCluster 3
SAS/ACCESS Interface to DB2 7
SAS/ACCESS Interface to Greenplum 21
SAS/ACCESS Interface to Netezza 29
SAS/ACCESS Interface to Teradata 39
SASLIB database 34
SASLIB schema (DB2) 11, 14
SASLIB schema (Greenplum) 25
SASUDF_COMPILER_PATH global variable 11
SASUDF_DB2PATH global variable 11
scoring functions in SAS Model Manager 43
SFTP statement 8
SSH software 8

T

tables
creating for Teradata 44
TAR files
unpacking for Aster nCluster 4
unpacking for DB2 9
unpacking for Greenplum 22
unpacking for Netezza 30
Teradata
creating tables 44
documentation for publishing formats and scoring models 42
in-database deployment package 39
installation and configuration 40
JDBC Driver 45
permissions 42
preparing for SAS Model Manager use 43
SAS/ACCESS Interface 39
Teradata Parallel Upgrade Tool 41

U

unpacking TAR files
for Aster nCluster 4
for DB2 9
for Greenplum 22
for Netezza 30

V

validating publication of functions and variables for DB2 17
validating publication of functions for Aster nCluster 6
validating publication of functions for Greenplum 28
variables
INDCONN macro variable 12, 15, 26, 32, 34
SASUDF_COMPILER_PATH global variable 11
SASUDF_DB2PATH global variable 11