THE POWER TO KNOW®

SAS® Drivers for Federation Server 3.2
User's Guide

SAS® Documentation

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2013. *SAS® Drivers for Federation Server 3.2: User's Guide*. Cary, NC: SAS Institute Inc.

SAS® Drivers for Federation Server 3.2: User's Guide

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

July 2013

Contents

What's New in SAS® Drivers for Federation Server 3.2

Overview

SAS Drivers for Federation Server has the following changes and enhancements:

- DataFlux Secure installed by default
- new software install location

DataFlux Secure Installed by Default

The DataFlux Secure software is now installed by default when you install the SAS Drivers for the SAS Federation Server. In previous releases, DataFlux Secure was purchased separately.

DataFlux Secure enables the SAS Drivers for the SAS Federation Server to use the same security features that you use on your SAS Federation Server. You can increase the level of encryption for passwords and network traffic. DataFlux Secure also enables compliance with the Federal Information Processing Standard (FIPS) 140-2.

DataFlux Secure is installed in a disabled state, so additional configuration is required only if you want to use a level of encryption other than the default SASPROPRIETARY. If you upgrade to 256-bit AES encryption, you need to encrypt the passwords in your DSNs accordingly.

New Software Install Location

The install location for the SAS Drivers for the SAS Federation Server has changed from:

```
3.1-install-path\SASDataFluxFederationServerClient
```

to:

```
SASHOME\SASFederationServerClient\3.2
```

Documentation Enhancements

The *SAS Drivers for Federation Server 3.2: User's Guide* is a new title for this release. The title for the last release was *DataFlux Drivers for ODBC and JDBC 3.1: User's Guide*.

All of the documentation for DataFlux products is now provided by SAS Customer Support, under `Documentation`, `Install Center`, and `System Requirements`.

Recommended Reading

- *SAS Federation Server: Administrator's Guide*
- *DataFlux Secure Administrator's Guide*

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

SAS Books
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 Overview

Using the Drivers

You can configure your applications to use a SAS ODBC or JDBC driver to access data sources on SAS® Federation Servers. The drivers are implemented as application programming interfaces (APIs).

To use a driver, follow these steps:

1. Define a data source on a SAS Federation Server.
2. Create a data source name (DSN) on your client host to connect to the SAS Federation Server data source.
3. Program your client application to use an ODBC or JDBC driver to access the data source.

About Security

The SAS Drivers for the SAS Federation Server are installed by default with the product DataFlux Secure. The security product enables you to replace the default encryption algorithm SASPROPRIETARY with 256-bit AES encryption.

Encryption is used for all network traffic to and from the SAS Federation Server. You can also encrypt the passwords that are included in your data source names (DSNs), to secure them for storage on disk. The level of encryption that you use for your passwords must match the level of encryption that is configured on your SAS Federation Server.

To learn more about security and encryption, see “Why Encrypt?”.

About ASBATCH

The install package for the SAS Drivers includes the ASBATCH utility. Administrators use ASBATCH to maintain the transactional database on the DataFlux Authentication Server. The use of ASBATCH is described in the *DataFlux Authentication Server Administrator's Guide*.

To maintain security, you should install ASBATCH only as needed by the administrators of the DataFlux Authentication Server.

Chapter 2

Post-Install Configuration

Configure the ODBC Driver on Windows

In the Windows operating environment, no further configuration is needed after you install the SAS ODBC driver for the SAS Federation Server.

Configure the SAS Driver for ODBC on UNIX or Linux

After you install the SAS Drivers for Federation Server on UNIX or Linux, follow these steps:

1. Download the source code for the unixODBC-2.3.0 driver to SAS Federation Server. The source code is located at **`http://www.unixodbc.org/download`**.
2. Before you create the distribution, read the installation steps at **`/opt/unixODBC-2.3.0/INSTALL`**. Also read the installation steps that apply to your operating environment, such as **`/opt/unixODBC-2.3.0/README.SOLARIS`**.
3. If you are on AIX, then set these environment variables as follows, before you create libraries and programs:

   ```
   export OBJECT_MODE=64
   export CFLAGS="-q64 -DBUILD_REAL_64_BIT_MODE
                 DSIZEOF_LONG=8"
   export CC=xlc_r
   export CPPFLAGS=$CFLAGS
   ```

   Change to the output directory and issue the following commands:

   ```
   cd '/opt/unixODBC-2.3.0/aix/lib'
   ar -X64 -x -v libodbc.a
   ar -X64 -x -v libodbccr.a
   ar -X64 -x -v libodbcinst.a
   ```

```
ln -s ./libodbc.so.1     ./libodbc.so
ln -s ./libodbccr.so.1   ./libodbccr.so
ln -s ./libodbcinst.so.1 ./libodbcinst.so
```

4. If you are on HP-UX H6I Itanium, then set environment variables as follows:

   ```
   export CFLAGS="+DD64 -DBUILD_REAL_64_BIT_MODE
                   -DSIZEOF_LONG=8"
   export CPPFLAGS=$CFLAGS
   ```

5. If you are on Solaris, then set environment variables as follows:

   ```
   export CFLAGS="-m64 -DBUILD_REAL_64_BIT_MODE
                  -DSIZEOF_LONG=8"
   export CC=cc
   export CPPFLAGS=$CFLAGS
   ```

6. If you are on Linux, then set environment variables as follows:

   ```
   export CFLAGS="-m64 -DBUILD_REAL_64_BIT_MODE
                  -DSIZEOF_LONG=8"
   export CPPFLAGS=$CFLAGS
   ```

7. Set the link order so that you link to the POSIX thread library before you link to the standard C library. Linking in this order ensures that you load all of the libraries that are required by the SAS Driver for ODBC. The following example command shows the required loading order:

   ```
   /usr/ccs/bin/ld -o UserApplication -u___exit
         -umain logger.o ODBCClient.o -L /usr/local/lib
         -L /usr/lib -L /opt/unixODBC-2.3.0/h6i/lib -lodbc
         -lpthread -lc
   ```

Configure the SAS JDBC Driver

After you install the SAS JDBC Driver for the SAS Federation Server, be sure to add all of the driver's JAR files to your application’s CLASSPATH environment variable. Another alternative is to add the JAR files to the **`-classpath`** option of your application's JVM start-up command.

Chapter 3

Creating DSNs on Windows for the SAS ODBC Driver

Create ODBC DSNs on Windows

Before you create the ODBC data source names in the Windows operating environment, complete the following tasks:

- Install the SAS Driver for ODBC on your Windows application host.
- Configure data sources on the SAS Federation Server using administration DDL or SAS Federation Server Manager. For more information, see the *SAS Federation Server: Administrator's Guide*.

Follow these steps to create ODBC DSNs on Windows:

1. In Windows, start the ODBC Data Source Administrator, as described in `Open the ODBC Data Source Administrator`.
2. Select the **User DSN** or **System DSN** tab.
3. Click **Add**.
4. In the Create New Data Source dialog box, select the driver **SAS 32-bit Federation Server** or **SAS 64-bit Federation Server**.
5. Click **Finish**.
6. In the dialog box SAS Federation Server ODBC Driver Setup, enter the required options and values. To learn about required options, see“Specify Required Options for Data Sources”.

7. Add or change the values of the Advanced Options section as required by your application. To learn about advanced options, see "Specify Advanced Options for ODBC Data Sources".
8. Select **Test Connection** to confirm the validity of your driver setup values.
9. Click **OK** to complete the setup process.

Specify Required Options for Data Sources

The following table defines the required values for ODBC and JDBC data sources. The fields and values apply in the Windows, UNIX, and Linux operating environments.

Table 3.1 *Required Options for Data Sources*

Field	Definition	Example
Data Source Name	The ODBC data source name refers to the collection of information that is used to access a SAS Federation Server data source. The name will likely provide information about the data source and possible options. The maximum length of names is specified in Sqlext.h by SQL_MAX_DSN_LENGTH. The default length is 32 characters. Names are checked by the SQLValidDSN function to ensure that the names do not include the following characters: [ ] { } ( ) , ; ? ! @ \	FedSrvDB2
Description	Provides additional information about the data source.	Federation Server DB2
Server Name	The network name of the target host of the SAS Federation Server.	dev003.orion.com
Port Number	The port number that was assigned to the SAS Federation Server.	21032
Encoding or CEI	The character encoding that is used by the SAS Federation Server.	Win cp1252-latin1 or UTF-8
User ID	The user name that your application uses to connect to the SAS Federation Server. Specify a domain if necessary. The user name appears in plaintext.	HQNT\kehous
Password	The password that is associated with the user ID. On Windows, the password is encrypted and stored in the Windows registry. On UNIX or Linux, the password is stored as plaintext. To learn how to encrypt passwords, see "Encrypt Passwords in DSNs".	Plaintext: 72Hken11Encrypted: {SAS003}1804E6B4DEC1DD52DD9552D39BC82C96AD6E
FSDSN	The SAS Federation Server data source name.	DB2_DSN

Encrypt Passwords in DSNs

Why Encrypt?

You encrypt the passwords in your DSNs to protect those values when they are stored on disk. Network connections between your client application and the SAS Federation Server are encrypted by default.

Encryption Tools

To encrypt passwords, use the encryption tool **`dfs_crypt`** (in Windows) or **`dfsadmin`** (in UNIX or Linux). For information about using the encryption tools, refer to *'Utilities for SAS Federation Server'* in the *SAS Federation Server: Administrator's Guide*.

Encryption Level

The level (or algorithm) of encryption that you use for your passwords must match the level of encryption that is used by the SAS Federation Server. The default encryption level is SASPROPRIETARY. If you upgrade the level of encryption on the SAS Federation Server, then you need to update your existing DSNs. The passwords need to be encrypted again using the encryption algorithm that is used by your SAS Federation Server.

Entering Passwords into Your Client Application

If your client application requires users to enter passwords to access data sources, then it might be necessary for those users to enclose their password entries in single quotation marks. The following example shows how single quotation marks are required by the Microsoft ODBC Test Tool. In the SQLDriverConnect statement, the password value for inConnectionString requires single quotation marks:

```
DSN=ORACLE_GENERIC;UID=Local/dsnadm;
     PWD='{SAS003}1804E6B4DEC1DD52DD9552D39BC82C96AD6E';
```

If quotation marks are expected but not provided, you will receive a parsing error message that references the encrypted password value.

Note that quotation marks are not used around encrypted password values in DSNs. Quotation marks are not used in the Windows registry, the Windows ODBC Administrator, or in the UNIX or Linux file that defines DSNs (typically dfodbc.ini.)

Specify Advanced Options for ODBC Data Sources

Introduction

You can specify advanced options in your ODBC DSNs to configure the SQL cursor, parameter set behavior, and transcode errors.

Advanced Options

The following table defines advanced options for Federation Server ODBC data sources. These values apply in the Windows, UNIX, and Linux operating environments.

An example of an application that requires advanced options is the OpenOffice application, which requires that you enable the options Cursor Concurrency and Cursor Library. Enabling these options prevents SQLFetchScroll from reporting SQL_ERROR with SQL State of HY106, which indicates that a fetch type is out of range.

Note: In the following table, the Keyword value is the name of the option as it appears in the data source definition.

Table 3.2 *Advanced Options for ODBC Data Sources*

Option	Description
Cursor Concurrency	Values: false or true Default: false Keyword: CONCURR_RO Description: This option specifies the allowed values for the SQL_ATTR_CONCURRENCY attribute. This attribute is set using the SQLSetStmtAttr ODBC API. When this option is disabled, the ODBC driver places no limits on the values for the SQL_ATTR_CONCURRENCY attribute. When this option is enabled, the driver limits SQL_ATTR_CONCURRENCY to only SQL_CONCUR_READ_ONLY. Attempts to set SQL_ATTR_CONCURRENCY to values other than SQL_CONCUR_READ_ONLY , such as SQL_CONCUR_LOCK, result in an SQL state of HYC00.
Cursor Library	Values: false or true Default: false Keyword: CUR_FO Description: This option specifies the allowed values for the SQL_ATTR_CONCURRENCY attribute. This attribute is set using the SQLSetStmtAttr ODBC API. When this option is disabled, the ODBC driver places no limits on values for the SQL_ATTR_ODBC_CURSORS attribute. When this option is enabled, the driver explicitly sets SQL_ATTR_ODBC_CURSORS to SQL_CUR_USE_IF_NEEDED just prior to connection.

Option	Description
Parameter Set Behavior	Values: emulate, direct, or prepare Default: prepare Keyword: PSB Description: This option specifies the manner in which FedSQL emulates parameterized Insert, Delete, and Update statements. emulate indicates that FedSQL always emulates parameterized Insert, Delete, and Update statements, using BulkOps. Parameter arrays should always work, provided that the underlying driver supports BulkOps. However, because the user has the option of setting the PARAMSET_SIZE after calling Prepare, some parameterized statements that could have been pushed down are emulated instead. direct indicates that FedSQL attempts to support the emulation of parameterized Insert, Delete, and Update statements. When parameterized statements are attempted, if the underlying driver cannot support those statements, then those statements fail. prepare indicates that FedSQL limits PARAMSET_SIZE prior to Prepare. Attempts to set the PARAMSET_SIZE generate errors. FedSQL determines at Prepare-time whether to off-load the parameterized Insert, Delete, or Update statement to the underlying driver or to attempt to emulate that statement via BulkOps.
Transcode Errors Behavior	Values: error, warning, or ignore Default: ignore Keyword: XCODE Description: This option determines how to report errors to applications when the SAS ODBC driver is unable to transform character data (Transcode) to or from the data source. These errors typically occur when a character cannot be represented in both the application and the data source. The values specified by this option are used to set the TKTS_ATTR_XCODE_WARN attribute using the TKTSSetStmtAttr API provided by the Federation Server. error reports transcode errors at the error level. warning reports transcode errors at the warning level. ignore does not report transcode errors..

Example Registry Entry

The following example depicts a registry entry for FedServerDSN1, which connects to the Federation Server with an Oracle base data source named ORACLE_DSN.

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\ODBC\ODBC.INI\FedServerDSN1]
"Driver"="C:\Program Files (x86)\SASHOME\bin\dfodbc.dll"
"Description"="" "Server"="myserver.us.orion.com" "Port"="21032" "Protocol"="BRIDGE"
"UID"="SAS\myuser" "FSDSN"="ORACLE_DSN" "Cei"="Win cp1252-latin1"
"PWD"="{SAS002}9C943B705636DF691286BEA34C6547D1" "CONCURR_RO"="false"
"CUR_FO"="false" "PSB"="prepare" "XCODE"="ignore" "FSCONSTR"=""
"TRACEFILE"="client.log" "TRACEFLAGS"="" "TRACELEVEL"="off"
```

```
"PARMS_FILE"="dfodbcJournal.log" "PARMS_TRACE"="off"
```

Usage Note for Microsoft Master Data Services

When you use Microsoft Excel to access data sources on a SAS Federation Server, be sure to configure your data service as case-insensitive. If your data service is case-sensitive, you might receive an error message that states that you submitted an invalid schema name.

The requirement for case-insensitivity applies whenever you use Microsoft Master Data Services to access data sources on SAS Federation Servers.

Chapter 4

Creating DSNs on UNIX or Linux for the SAS ODBC Driver

Overview

To enable your application to connect to ODBC data sources on SAS Federation Servers, you define data sources in a file and set environment variables on your application host.

Set Environment Variables

Set the following environment variables in the shell of your application:

LD_LIBRARY_PATH
Set or append the location of the unixODBC binaries and also set or append the library for the ODBC driver, in ***install-path*/lib**. The following example command appends the two paths, using typical values:

```
export LD_LIBRARY_PATH=/wire/develop/odbc/2.3.0/lax/lib:

        /opt/SASHOME/fedclient/lib
```

ODBCINI
Set the name and location of the file that defines ODBC data sources on your Federation Servers. Set a value such as the following:**/opt/SASHOME/fedclient/lib/dfodbc.ini**.

TKPATH
Set or append the location of the libraries that are used by the SAS Driver for ODBC. Set or append a value such as the following: **/opt/SASHOME/fedclient/lib/**.

The library path specified for TKPATH reflects the fact that the SAS Driver for ODBC can be used only with that version of the SAS Threaded Kernel library. Applications that use other SAS threaded kernel libraries cannot use the SAS Driver for ODBC.

The following example uses the script dfsenv to illustrate how to update the TKPATH and LD_LIBRARY_PATH environment variables. If your application runs in a sh, ksh, or bash shell, then execute the following command to update the environment variables:

```
eval './bin/dfsenv sh'
```

If your application runs in a csh or tcsh shell, then execute the following command:

```
eval './bin/dfsenv csh'
```

Define Data Sources

After you install the SAS Driver for ODBC on a UNIX or Linux client host, you create the file that defines ODBC data sources. Begin by opening a new file using the name and location that is specified in the environment variable ODBCINI.

To define data sources in the file, refer to the following example, which contains values for a DSN named FS_ORACLE.

Note: To add a comment line, begin the line with a **#** character.

To learn about the options and values that you can add to your data source definitions, see see "Specify Required Options for Data Sources", and "Specify Advanced Options for ODBC Data Sources".

```
[ODBC Data Sources]
[FS_ORACLE]
Description = SAS Federation Server DSN
FSDSN = ORACLE_DSN
Driver = /opt/SASHOME/lib/dfodbc.so
SERVER = testFed01.us.orion.com
PORT = 21032
PROTOCOL = BRIDGE
UID = appConnect
PWD = 25o7xWd0gJzK
CEI = UTF-8
CONCURR_RO = false
CUR_FO = false
PSB = prepare
XCODE = ignore
```

Chapter 5 Creating DSNs for the SAS JDBC Driver

Java Class Name

The name of the Java class that implements **`java.sql.Driver`** in the SAS Federation Server is: **`com.dataflux.fs.jdbc.driver.FSJDBCDriver`**. Use this class name when registering the driver or when configuring software to use the SAS JDBC Driver.

Specify Properties for the SAS JDBC Driver

Introduction

You can specify properties for the SAS JDBC Driver in a URL or in the object java.util.Properties.

URL

The URL syntax for the SAS JDBC Driver on the SAS Federation Server is as follows:

```
jdbc:sastkts://fed-server-host:fed-server-port
      ?property-name1=property-value1
      &property-name2=property-value2...
      &property-nameX=property-valueX
```

The following example demonstrates the syntax:

```
jdbc:sastkts://devtest001.orion.com:2171?constring=(DSN=DB2DSN1)
      &userName=local\\user1&password=user1password
```

There is no default value for ***`fed-server-host`*** or ***`fed-server-port`***.

To connect to a SAS Federation Server that runs on the same host as your client, use the value **`localhost`** or **`127.0.0.1`** for ***`fed-server-host`***.

In a long JDBC URL, it is easy to become confused between JDBC connection properties and the SAS Federation Server driver's connection options. For example, consider this URL:

```
jdbc:sastkts://localhost:2171?cursorType=TKTS_CUR_USE_DRIVER
        &constring=(DRIVER=FIREBIRD;CATALOG=fbdetail;
        DATABASE='c:\fbdata\fbtest.fdb';)
```

In this case, **`cursorType`** and **`constring`** are JDBC connection properties.

java.util.Properties

To set connection properties in the java.util.Properties object, add key-value pairs to the object, and pass the object to DriverManager.getConnection() or Driver.connect(), or use the set*() methods on a Datasource.

Table 5.1 *java.util.Properties for the Statement Pool Manager*

Property	Definition
constring	Connection string.
cursorType	TKTS_CUR_USE_TKTS, TKTS_CUR_USE_DRIVER, or TKTS_CUR_USE_IF_NEEDED
defaultFetchSize	Specifies the default fetch size for the statements created by the connection.
IDLE_TIMEOUT	Specifies the interval, in seconds, after which an idle pooled statement handle is closed and discarded. The default value is 180.
IS_ALLOCATE_OVER_THE_LIMIT	Determines whether the Statement Pool Manager allocates new statements after the maximum number of statements in the pool has been reached. The default value is TRUE.
IS_POOLED_STATEMENT_ENABLED	Determines whether the Statement Pool Manager is used in the current connection. The default value is FALSE.
IS_RECORD_STATISTICS	Determines whether the Statement Pool Manager displays statistics when the connection is closed. The default value is FALSE.
MAX_STATEMENT_POOL_SIZE	Specifies the maximum number of statement handles that can be stored in the pool. There is no wait if the pool reaches its maximum size. A request either immediately allocates a new statement handle if IS_ALLOCATE_OVER_THE_LIMIT is set to TRUE, or it returns an error. The pool never exceeds the MAX_STATEMENT_POOL_SIZE. The Statement Pool Manager is at liberty to dispose of a previously pooled statement handle and replace it with a new one. The maintenance task is free to clean out as many pooled statements as it sees fit. The default value is 100.

Property	Definition
password	The password that accompanies the userName to authenticate the client application for access to the Federation Server. The login must exist as a user definition in the appropriate Authentication Server. Note that you might need to supply additional authentication parameters to connect to a relational database through the SAS Federation Server. To encrypt the password, see "Encrypt Passwords in DSNs".
POOL_MAINTENANCE_INTERVAL	Specifies the interval, in seconds, between executions of the statement pool maintenance task. The default value is 1.
spn	Sets the service principal name to use when connecting to the Federation Server. This property is valid only when the Federation Server uses Kerberos or Negotiate.
STATISTICS_OUTPUT_DIRECTORY	Specifies the directory that contains a file of statistic records. The default value is the default JVM temporary directory, as specified in the system property java.io.tmpdir. The filename has the following format: stmtPooling_*YYMMDD_HHMMSSmmm_pool-Hash-Code.log*.
userName	The user name of the login that is used to authenticate the client application for access to the SAS Federation Server. The login must exist on the associated Authentication Server. Note that you might need to supply additional authentication parameters to connect to a relational database through the SAS Federation Server.
usesspi	Determines whether the driver uses SSPI authentication. Valid values are none, kerberos, ntlm, or negotiate.

Example JDBC Connection Using a DSN

The following example accesses an employee table in a sample database using a DSN:

```
package com.sas.fs.doc.example;

import java.sql.*;
import java.util.Properties;

public class DocTest
{
    static {
        try {
             Class.forName("com.dataflux.fs.jdbc.driver.FSJDBCDriver");
        } catch (ClassNotFoundException e) {
          e.printStackTrace();
        }
    }

  public static void main(String argv[])
    {
       try {
            Properties props = new Properties();
            props.put("user", "fedserveruser");
```

```
                props.put("password","{SAS003}5F5F60A0CEC0B921B503926075B4B6EBD08F");
          props.put("constring", "DSN=MYTRANDSN");
              Connection connection =
                     DriverManager.getConnection
                   ("jdbc:sastkts://mymachine.orion.com:2171", props);
             Statement statement = connection.createStatement();
                ResultSet result = statement.executeQuery("SELECT * FROM EMPLOYEE");
             ResultSetMetaData rsmd = result.getMetaData();
                 int numberOfColumns = rsmd.getColumnCount();

                 for (int i = 1; i <= numberOfColumns; i++) {
                   System.out.print(rsmd.getColumnLabel(i) + " ");
                 }
                 System.out.println("");
                 while (result.next()) {
                      for (int i = 1; i <= numberOfColumns; i++) {
                     System.out.print(result.getString(i) + " ");
                  }

                  System.out.println("");
               }
          statement.close();
          connection.close();
        }

        catch (Exception e) {
           System.out.println("error " + e);
              }
       }
   }
```

Using the Statement Pool Manager

About the Statement Pool Manager

The Statement Pool Manager of the SAS JDBC Driver manages a list of named handles that apply to a set of frequently used SQL statements.

To enable and configure the Statement Pool Manager, set JDBC connection properties, as described in “Specify Properties for the SAS JDBC Driver”.

You can set properties as Java Virtual Machine arguments or as TKTS connection properties. Properties defined at the JVM level are the default for all TKTS connections. To override the JVM properties, applications must define the properties at connection time.

When it is initialized, the Statement Pool Manager starts a pool maintenance task. At intervals specified by the property POOL_MAINTENANCE_INTERVAL, the maintenance task closes and discards idle statements. Statements are removed if they have been idle for the interval specified by the property IDLE_TIMEOUT.

To retrieve information about the statement pool, you can specify static methods in the StatementPoolManager class. For information about the static methods, see “Interact with the Statement Pool Manager”.

About Statement Attributes

To use the Statement Pool Manager effectively, it is important to understand that the manager does not retain changes to statement attributes. Attribute changes are reset when you return a statement to the pool. To avoid confusion, be sure to use separate handles for statements that have different attributes, even if the statements are otherwise alike.

About Statement Handles

It is possible for the Statement Pool Manager to cache separate statement handles that have identical SQL statements. For an example, imagine an application that executes multiple data selection statements against a single table, using different parameters, such as the following:

```
select * from WORLD_SALES where region = ?
```

To generate a report, if the application executes the statement once for each region, then the Statement Pool Manager needs to maintain separate statement handles for each region parameter. With separate statement handles, each region parameter generates a separate and unique result set. In this example, to create separate statement handles, you would submit separate statements with explicit region values. You could then reuse those handles as needed.

About the Operational Sequence

When your application requests a statement, the Statement Pool Manager responds as follows:

1. If the statement is in the pool, then the manager removes it from the pool and makes it available to your application.
2. If the statement is not in the pool and the pool is not full, then the manager provides your application with a new statement handle. When the newly created statement is closed by your application, it is put into the pool.
3. The manager throws an exception StatementPoolFullException to your application if all of the following conditions are true:
   - The statement is not in the pool.
   - The pool is already full.
   - The property IS_ALLOCATE_OVER_THE_LIMIT is set to FALSE.
4. The manager provides your application with a new statement handle if the following conditions are true:
   - The statement is not in the pool.
   - The pool is already full.
   - The property IS_ALLOCATE_OVER_THE_LIMIT is set to TRUE.

Set Properties with TKTS

The following example shows how to set properties for the Statement Pool Manager using TKTS at connection time.

For option definitions, see “Interact with the Statement Pool Manager”.

```
Properties connectionProperties = new Properties();
connectionProperties.put(StatementPoolManager.PROP_IS_POOLED_STATEMENT_ENABLED,
      "true");
connectionProperties.put(StatementPoolManager.PROP_MAX_STATEMENT_POOL_SIZE,
      "100");

connectionProperties.put(StatementPoolManager.PROP_IS_ALLOCATE_OVER_THE_LIMIT,
      "true");

connectionProperties.put(StatementPoolManager.PROP_POOL_MAINTENANCE_INTERVAL,
      "1");

connectionProperties.put(StatementPoolManager.PROP_IDLE_TIMEOUT, "60");

Connection conn = DriverManager.getConnection("jdbc:sastkts://hostName:2100,
      connectionProperties);
```

Set Properties with JVM Arguments

To set properties for the Statement Pool Manager using JVM arguments, use the following syntax to add options to your Java command.

For option definitions, see "Interact with the Statement Pool Manager".

```
-DIS_POOLED_STATEMENT_ENABLED=TRUE | FALSE

-DMAX_STATEMENT_POOL_SIZE=number-of-statements

-DIS_ALLOCATE_OVER_THE_LIMIT=TRUE | FALSE

-DPOOL_MAINTENANCE_INTERVAL=time-in-seconds

-DIDLE_TIMEOUT=time-in-seconds

-DIS_RECORD_STATISTICS=TRUE | FALSE

-D STATISTICS_OUTPUT_DIRECTORY=absolute-path
```

Interact with the Statement Pool Manager

Use the following static methods in the class StatementPoolManager to retrieve information about the statement pool. Each method receives a connection as its input parameter:

`public static long getCurrentPoolSize(Connection conn)`
The current number of statements in the pool.

`public static int getIdleTimeout(Connection conn)`
The interval, in seconds, after which an idle pooled statement handle is closed and discarded. Example:

```
Boolean b = StatementPoolManager.isPooledStatementEnabled(connection);

System.out.println("Is Pooled Statement Enabled   = " + b);
```

`public static int getMaxStatementPoolSize(Connection conn)`
The maximum number of statement handles that can be stored in the pool.

`public static long getPoolFullHits(Connection conn)`
The number of times the pool manager tried to create a new statement but the pool was full.

`public static long getPoolHits(Connection conn)`
The number of times a statement was delivered from the pool.

`public static int getPoolMaintenanceInterval(Connection conn)`
The interval, in seconds, between executions of the statement pool maintenance task.

`public static long getPoolMisses(Connection conn)`
The number of times the pool manager did not find an available statement handle in the pool.

`public static long getPoolPurges(Connection conn)`
The number of times the maintenance task purged a pooled statement.

`public static boolean isAllocateOverTheLimit(Connection conn)`
The flag indicating if the pool manager will allocate a new statement after the maximum number of statements in the pool has been reached.

`public static boolean isPooledStatementEnabled(Connection conn)`
The flag indicating if statement pooling is being used in the current connection.