Shadow ®

Release Notes

For Shadow Client

Version 3, Release 9, Build 366

Release Date: March 19, 2004

These release notes cover the new features, considerations, and problem resolutions offered by Shadow JDBC Connect 3.9.366 as follows:

- Release Summary
  - Shadow JDBC Connect Improvements
  - Shadow JDBC Connect Keyword Enhancements
  - Other Enhancements
- Enhancement and Problem Resolution Details
- Known Issues
- Windows Considerations
- UNIX Considerations

Release Summary

Shadow JDBC Connect 3.9 offers several significant new features and other enhancements, in addition to resolutions to various problems, as follows:

- Shadow JDBC Connect Improvements
- Shadow JDBC Connect Keyword Enhancements
- Other Enhancements

Shadow JDBC Connect Improvements

With the release of Shadow JDBC Connect 3.9, the Type 2 driver implementation has been improved. A summary of changes follows:

- **Type 2 Implementation:** Shadow JDBC Connect no longer has any dependency on ODBC driver files.

NEON SYSTEMS, INC.

NEON Systems, Inc.
14100 SW Freeway, Suite 500
Sugar Land, Texas 77478
Phone: (281) 491-4200
Fax: (281) 242-3880

NEON Systems UK Ltd.
1, High Street
Windsor
Berkshire
SL4 1LD England
tel: +44 1 81 607 9911
fax: +44 1 81 607 9933

- **Data Source Changes:** Data sources for Shadow JDBC Connect will be created and maintained as follows:

  – Data sources are stored in the `SHADOW.INI` configuration file. On Windows, the location of this file is specified in the registry (with a default installation, the location is `C:\Program Files\IBM\Shadow \bin\shadow.ini`). On UNIX, the location is specified by the SHADOW_INI environment variable.

  > ***Note:***
  > Past versions of Shadow JDBC Connect stored data sources in the `ODBC.INI` configuration file.

  Important: **This change implies that users who are currently pointing to data source configuration files with the ODBCINI or ODBC_INI environment variables must now define the SHADOW_INI environment variable, instead.**

  – Existing Shadow JDBC Connect data sources[1] will be automatically copied and migrated for use upon the installation of Shadow JDBC Connect 3.9. In addition, the jConfig tool also offers a feature to allow users to manually select existing data sources to be imported for use with Shadow JDBC Connect.

  > ***Note:***
  > After the data sources are imported, existing data sources present in the `ODBC.INI` will ***not*** be removed; it will be left to the user to delete unused data sources.

  – Data sources are configured using the jConfig tool (not the ODBC Data Source Administrator or the `ODBC.INI` file).

- **jConfig Tool:** The jConfig tool is provided with Shadow JDBC Connect to create and configure data sources. This tool is offered on all supported platforms, giving both Windows and UNIX users a graphical user interface for maintaining data sources.

- **jDemo Program:** The jDemo program is provided as part of the Shadow JDBC Connect utilities to execute SQL queries and obtain data source information using Shadow JDBC Connect. It is an easy-to-use program offered on all supported platforms.

1 "Shadow JDBC Connect data sources" are defined as ODBC/TCPIP data sources with the APPL, DBTY, HOST, and PLAN keywords configured (empty string values are acceptable).

Shadow JDBC Connect Keyword Enhancements

The following enhancements were made regarding Shadow JDBC Connect keywords:

- **ABCN (ADABAS Column Name Support):** The ABCN keyword was added for Shadow JDBC Connect for ADABAS users to enable support for ADABAS column name correlation IDs on a per-connection basis. For details, see “SI-8173” on page 9.
- **DENU Keyword:** The DENU (Decimal as Numeric) Shadow JDBC Connect keyword was added to specify whether DECIMAL and NUMERIC columns should be reported as DECIMAL columns or NUMERIC columns (because both types of columns are treated as the same by Shadow JDBC Connect). For details, see “SI-11560” on page 22.
- **DCTM (Disconnect Timeout Value):** The DCTM keyword was added to allow users to specify a disconnect timeout value to control how many seconds the Shadow Client will wait for a socket to be ready for disconnect operations. For details, see “SI-9242” on page 18.
- **FALG (Fast Logon):** The FALG keyword will now default to YES. For details on the functionality provided by the FALG keyword, see “SI-7575” on page 7.

Other Enhancements

- **Support for SQLESETI and SQLEQRYI:** SQLESETI and SQLEQRYI APIs are now supported. For details on the implementation, see “SI-1544” on page 4.
- **Improved Prepare/Open Optimization:** Prepare/Open Optimization has been modified for improved performance. For details, see “SI-7968” on page 8.
- **Network Installation:** A network installation option for Windows will install the Shadow JDBC Connect drivers (i.e., DLL files) in a common location where they can be easily maintained and upgraded. For details, see “SI-8148” on page 8.
- **LOB Handling:** The handling of NULL and EMPTY LOB result set columns in SQLGetData was corrected to provide performance improvements. For more information, see “LOB Handling” on page 24.
- **XA Support Improvements:** Several enhancements have been added to XA support (two-phase commit support) offered by Shadow JDBC Connect. For details, see “XA Support” on page 24.

Enhancements and Problem Resolution Details

The following enhancements and problem resolutions are offered by Shadow JDBC Connect 3.9.366.

Note:
Support incident (SI) numbers are listed, if applicable.

SI-17-4

Enhancement: If available, the Shadow Server version number will now be displayed in the traces.

Shadow Server Version: This enhancement requires Shadow Server 4.8 SVFX3327 or higher.

SI-1544

Enhancement: Shadow JDBC Connect now supports the SQLESETI (Set Client Information) and SQLEQRYI (Query Client Information) APIs. Two new API functions were added to Shadow JDBC Connect to support this functionality, as follows:

- **SHADOW_SQLESETI():** Maps to SQLESETI.
- **SHADOW_SQLEQRYI():** Maps to SQLEQRYI.

To utilize this new support, the customer must code additional calls, formatted as supported by the SQLESETI and SQLEQRYI APIs.

Currently, the SQLESETI information is only displayed in the ISPF Remote Users application (accessed from the **Shadow Server Primary Option Menu** Option 4) and is written in the Shadow Server subtype 06 SMF records. At this point, it does not appear in the DB2 DISPLAY THREAD command output.

Doc Reference:
For information on the SQLESETI and SQLEQRYI function definitions, review the following:

`http://www-3.ibm.com/cgi-bin/db2www/data/db2/udb/winos2unix/support/v8document.d2w/report?fn=r0001905.htm`

Note:
At this time, Shadow JDBC Connect does not support the following connection settings for the TYPE element of the SQLE-CLIENT-INFO structure, which is used to pass information to the SQLESETI and SQLEQRYI APIs:

- SQLE_CLIENT_INFO_PACKAGEPATH
- SQLE_CLIENT_INFO_PROGRAMID

Shadow Server Version: This enhancement requires Shadow Server 4.8 SVFX3076 or higher.

SI-1847

Incident Description: When running in Virtual Connection Facility (VCF) TRANSACTION connection mode, if auto-commit is enabled and the SQL statement fails, the connection will not be closed; in fact, the connection remains open indefinitely, even though the client is connecting in TRANSACTION mode.

If auto-commit is enabled and the SQL statement fails, the Shadow Server was not returning a ROLLBACK indicator. In TRANSACTION mode, this prevents Shadow JDBC Connect from sending a CLOSE SYNC or CLOSE ABRT to close the connection.

Resolution: Logic was added to Shadow JDBC Connect to detect such an error condition and send a CLOSE SYNC.

SI-1994

See “SI-1544” on page 4.

SI-2058

Incident Description: When Shadow JDBC Connect is used with prepared statement caching enabled, the following exception would occur if used via a CMP entity bean:

```
[NEON][SCOD32R.DLL]Function called before SQLPrepare
```

The problem resulted when executing a prepared statement after a SQL_ROLLBACK is issued.

However, when used via a servlet or JSP with SQL, the error didn’t occur.

Resolution: This problem is caused by the application having prepared statement caching enabled. Prepared statement caching is not currently supported in Shadow. Shadow JDBC Connect users integrating with application servers should set the statement caching size to zero (0) to disable statement caching.

SI-6626

Enhancement: This enhancement implements changes on Shadow JDBC Connect to support the updated licensing restrictions regarding the use of generic and extended userids. Shadow JDBC Connect no longer checks for Shadow Enterprise Auditing (formerly TLS) support before attempting to set or display the generic and extended userids. Any required verification will be performed on the Shadow Server.

Shadow Server Version: This enhancement requires Shadow Server 4.8 SVFX2200 or higher.

SI-6887

Incident Description: When using Shadow JDBC Connect on UNIX and trying to implement the use of SSL for multi-threaded connections, the following error occurred when trying to connect to the host:

```
[NEON][SCODBC_R DLL]Host communication failed
```

Resolution: There were several threading problems discovered in the SSL support for UNIX, which have been corrected on the platforms supporting SSLeay.

SI-7181

Incident Description: When stress testing a Windows 2000 two-phase commit application, a failure would occur during XA_START processing.

Resolution: The problem that occurred when a new XA_START was processed on an existing connection has been corrected. In addition, all statement handles will now be allocated on the first connection that enlists for an XID. This change improves performance and avoids context switching on the server that was causing DB2 threads to hang in DI state.

SI-7433

Incident Description: When using Shadow JDBC Connect for non-DB2 connections (i.e., when DBTY is set to anything other than DB2), it is required that the user specify SUBSYS=NONE because a DB2 subsystem is not required. However, Shadow JDBC Connect did not enforce this rule. For example, if the user specified DBTY=ADABAS, but mistakenly specified a DB2 subsystem name in the SUBSYS keyword, unexpected errors could occur when ADABAS queries were issued.

Resolution: Code was added so that if DBTY is set to anything other than DB2, Shadow JDBC Connect will automatically set SUBSYS=NONE ***if*** the SUBSYS value is currently not specified.

SI-7560

Incident Description: When using Shadow JDBC Connect for DB2, the following SELECT statement syntax would result in an error:

```
SELECT(COL1 + COL2)
```

> ***Note:***
> Note that there is no space between the word “SELECT” and the parenthetical value. If a space is inserted, the statement executed without a problem.

When using VB Demo to execute the statement, this syntax would cause a SQLCODE -518 (prepare failure). In a program with parameter markers, a system abend X'0C3' would result.

Resolution: The Shadow JDBC Connect parsing used in this situation was rewritten to handle this situation using standard CRT functions.

Shadow Server Version: This resolution requires Shadow Server 4.8 SVFX1643 or higher.

SI-7575

Enhancement: The FALG (Fast Logon) Shadow JDBC Connect keyword will now default to YES. This keyword determines how many network roundtrips it takes to complete the logon. FALG=YES offers a slight performance improvement in connect processing.

With older versions of Shadow JDBC Connect, it was necessary to use two network roundtrips to complete the logon. After Virtual Connection Facility (VCF) was introduced, this requirement went from two network roundtrips to one; however, the customer had to be at Shadow Server v3.1 for this support. Since Shadow Server v3.1 is now over four years old, setting Shadow JDBC Connect to default to FALG=YES should not impact any existing customers when they upgrade.

SI-7699

Incident Description: The OPRW (Optimal Row Count) keyword on Shadow JDBC Connect, which limits the number of rows that will be returned from the host each time a request for rows is made, was not honoring the “OPTIMIZE FOR 1 ROW” clause in the SQL statement if OPRW=0 (the default). Instead, the Shadow Server would fetch 10 rows. Shadow JDBC Connect was changing the optimal row value expressed in “OPTIMIZE FOR X ROWS”; when X = 1, Shadow JDBC Connect used 10 instead.

Resolution: When “OPTIMIZE FOR 1 ROW” is specified, Shadow JDBC Connect now honors the optimal row value. For fetch processing, a check was added to request the maximum number of rows possible when “OPTIMIZE FOR

1 ROW” is specified. This does not affect the first fetch operation; it only affects subsequent fetches.

SI-7777

Incident Description: When using Shadow JDBC Connect for ADABAS, attempts to use `SetLong` to insert data into a field defined as U in ADABAS would result in the following error:

```
SHADOW_ADABAS ERROR HAS OCCURRED RC -2055; NUMERIC VALUE ERROR -
PRECISION EXCEEDED
```

Resolution: Shadow JDBC Connect now allows the client to use `setLong` to insert a value into field defined as U in ADABAS by removing the leading “+” sign (if present) and all 0’s (zeroes) to the left of the first non-zero numeral before passing the value. Basically, for non-DB2 connections, the data sent from Shadow JDBC Connect to the Shadow Server has now been optimized.

Shadow Server Version: This resolution requires Shadow Server 4.8 at any SVFX level.

SI-7968

Enhancement: Support was added to the Shadow JDBC Connect Prepare/Open Optimization feature for deferred prepare, meaning that the SQLPrepare and SQLExecute can be combined into single call that is sent at SQLExecute time. Prepare/Open Optimization minimizes network flow and improves performance. This feature is enabled by setting the DIPO (Disable Prepare/Open Optimization) keyword on Shadow JDBC Connect to NO.

For those client applications that expect the SQLPrepare to be executed as soon as it requested (for example, applications that require the metadata in the SQLCA), Prepare/Open Optimization can be disabled by setting DIPO=YES.

SI-7978

See “SI-1544” on page 4.

SI-8126

Incident Description: When using Shadow JDBC Connect for VSAM for CICS, if using DBTY=VSAMCICS (i.e., the DBMS type was set to VSAMCICS), when trying to execute a VSAM query using parameter markers, the statement is not executed. This affected both the SQLPrepare/SQLExecute and SQLExecDirect code paths.

Resolution: The handling of the parameter markers for DBTY=VSAMCICS was corrected.

SI-8148

Enhancement: The Shadow JDBC Connect installation now offers a network installation option for Windows. The network installation will install the Shadow

JDBC Connect drivers (i.e., DLL files) in a common location where they can be easily maintained and upgraded. Individual clients can access the drivers at that location, while installing other files related to Shadow JDBC Connect on their local machine.

SI-8173

Enhancement: A new Shadow JDBC Connect keyword, ABCN (ADABAS Correction Name Support), was added for Shadow JDBC Connect for ADABAS users. This keyword enables support for ADABAS column name correlation IDs. This Shadow JDBC Connect keyword operates on a connection-basis to duplicate the functionality of the ADABASCORRELATIONIDS product parameter on the Shadow Server.

Note:

Column name correlation IDs are only supported for FIND SELECT and SELECT statements.

Setting ABCN=YES to support column name correlation IDs may cause a conflict with earlier versions of Shadow JDBC Connect for ADABAS, which accepted non-standard SQL syntax statements of the following form (note that there are no commas to separate the operands):

```
SELECT AA AB FROM EMPQA1
```

With ABCN=NO (or on earlier versions of Shadow JDBC Connect for ADABAS), this will select two columns, AA and AB. With this parameter set to YES, AB will be considered a correlation name for AA.

To select two columns when this parameter is set to YES, commas must be used to separate the two column names as follows:

```
SELECT AA, AB FROM EMPQA1
```

SI-8192-133

Incident Description: When using Shadow JDBC Connect with LGID=ENU (i.e., the language ID is set to U.S. English) or LGID=ENG (i.e., the language ID is set to U.K. English) , the English pound sign (“£”) was being translated incorrectly. Specifically, “£” was being translated as follows:

- LGID=ENU: “£” was translated to “Â£”.
- LGID=ENG: “£” was translated to “Â$”.

Note:

For a related issue, see “SI-9037” on page 16.

Resolution: The conversion of 8-bit ASCII characters for LGID=ENU and LGIG=ENG was corrected. The call to GetStringUTFChars was removed because this function does not handle graphic ASCII characters properly.

SI-8209

Incident Description: When the NEONTRACE Shadow JDBC Connect keyword was specified within a connection string, if the user set the LOG keyword and then set the JDBCLOG keyword ***after*** the LOG keyword, the following error would result:

```
Assertion failed: 0 && trifda.triffipt && "trace file open error",
file rperfu.c , line 2921
```

For example, the following setting would cause the error:

```
NEONTRACE=INFO THREADID ENVLIST BUFFER LOG=C:\neonlog.txt
JDBCLOG=C:\jdbclog.txt JCALOG=C:\jcalog.txt
```

Shadow JDBC Connect was picking up part of the other specification(s) as part of the LOG file name; thus, it could not open the trace file.

Note:
The JDBCLOG keyword is actually only valid within the NEONTRACE ***environment variable***—not the NEONTRACE Shadow JDBC Connect keyword.

Resolution: When processing the NEONTRACE Shadow JDBC Connect keyword for the connection, Shadow JDBC Connect will check for the JDBCLOG keyword and remove it, if present.

Note:
This doesn't affect the setting of the NEONTRACE environment variable. When setting the NEONTRACE environment variable, if the LOG keyword is specified, any other keywords ***must precede*** the LOG keyword!

SI-8242

Incident Description: The `scjdsm01.java` sample program could not be run from UNIX, as it was returning the following:

```
Can't load libscjd12ts.so, file not found
```

Resolution: There were native C literals that needed to be converted to ASCII.

SI-8254

Incident Description: When using Shadow JDBC Connect for ADABAS, if using DBTY=ADABAS (i.e., the DBMS type was set to ADABAS), there was a problem with inserting literals that contained a single-quote or double-quote into ADABAS fields.

Resolution: Shadow JDBC Connect will now “escape” the internal single-quote or double-quote characters inside literals. In addition, for non-DB2 connections (i.e., when DBTY is set to anything other than DB2), Shadow JDBC Connect will set BYDB=YES, enabling the Bypass Double Quotes keyword.

Shadow Server Version: This resolution requires Shadow Server 4.8 SVFX2711 or higher.

SI-8289

Incident Description: When using the Shadow Client with DBTY=DB2 (i.e., the DBMS type was set to DB2), the function used to determine the number of parameter markers was incorrectly assuming that a statement beginning with “CALL SHADOW_CICS” would include a SQL statement wrapped in the form CALL SHADOW_CICS('EXVS','...some text...'). Thus, attempts to prepare a SQL statement of the form CALL SHADOW_CICS(...some text...) would fail with the incorrect number of parameter markers because the code would skip the first 9 characters of “...some text...”.

Resolution: The assumptions made by the function used to determine the number of parameter markers were modified to handle cases SQL statements of the form CALL SHADOW_CICS(...some text...).

SI-8290

Incident Description: Currently, Shadow JDBC Connect does not support the `ResultSet.first()` method since only forward-cursor result sets are supported. However, if an application issues the `ResultSet.first()` method, Shadow JDBC Connect returned false for this method, which can cause the application to think that there is no result returned for the query. An exception should have been returned instead.

Resolution: If an application issues the `ResultSet.first()` method, Shadow JDBC Connect throws an exception instead of returning false.

The following `ResultSet` methods will also throw an SQLException when called:

- `absolute()`
- `afterLast()`
- `beforeFirst()`
- `first()`
- `last()`
- `previous()`
- `relative()`

SI-8399

Incident Description: In certain situations, when using parameter markers, a host variable substitution error would occur, resulting in corrupt values for certain variables.

This problem occurred because string pointer types of “char *” were being used; thus, any character value exceeding 127 was converted to a negative integer and causing memory problems during string table element allocation.

Resolution: The string pointer types were changed to “unsigned char *” to resolve the problem.

SI-8424

Incident Description: When using Shadow JDBC Connect in certain environments, attempts to use Shadow Enterprise Transactions (two-phase commit) were unsuccessful for the second transaction, resulting in the following error in the Shadow Server trace browse:

```
XID ENQUEUE FAILED - CTX SVCS NEONRRS.RESOURCE.MANAGERSDB3SYSK -
RM ALREADY HAS EXPRESSED INTEREST IN THIS CONTEXT
DSNHLI XA-START - SQLCODE -6, IMPROPER CONTEXT FOR REQUEST -
SDBC1010
```

Resolution: Shadow JDBC Connect was resetting the auto-commit status after XA-END. The auto-commit setting will now be delayed until after XA-COMMIT/XA-ROLLBACK.

Notes:

- For two-phase commit, the `com.neon.jdbc.DataSource` class must be used; the `com.neon.jdbc.Driver` class cannot be used. This is a result of the way the JDBC API is designed.
- `XADataSource` and `DataSource` are now the same. You can use either one, but it is recommended to use `DataSource`.

Shadow Server Version: This resolution requires Shadow Server 4.8 SVFX2872 or higher.

SI-8425

Incident Description: In certain environments, attempts to use stored procedures that have input parameters of type DECIMAL or TIMESTAMP would cause errors because SQLProcedureColumns was returning the wrong attributes for the data type of DECIMAL and TIMESTAMP columns.

Resolution: The attributes for the DECIMAL and TIMESTAMP column data types returned by SQLProcedureColumns have been corrected so that it returns a data type of 3 for DECIMAL columns and a data type of 11 for TIMESTAMP columns.

SI-8451

Incident Description: When using Shadow JDBC Connect on UNIX with LGID=DEU (i.e., the language ID is set to German), the Euro symbol was not translated correctly.

Resolution: The Euro symbol was added to the German conversion tables on UNIX.

SI-8461

Incident Description: When attempting to retrieve results using using stored procedures that have input parameters in certain environments, the following error was returned:

```
[NEON][SCOD32TS.DLL]Numeric literal conversion error
```

The problem was the result of converting long integer literal values into 4-byte integers, which generated an overflow error.

Resolution: Long integer literal values (greater than or equal to 10 bytes) will now be converted into decimals instead of integers to avoid overflow problems.

SI-8476

Incident Description: When using Shadow JDBC Connect with DBTY=VSAM (i.e., the DBMS type was set to VSAM), if the SQL statement included the “CALL SHADOW_VSAM” wrapper, the following error would result in SQLDescribeParam:

```
'Driver does not support this function'
```

Note:
Setting the DAOP (SQLDescribeParam Support) Shadow JDBC Connect keyword to NO did ***not*** work around the problem.

Resolution: Shadow JDBC Connect now takes into consideration cases where the “CALL SHADOW_VSAM” wrapper is used when DBTY=VSAM.

SI-8523

Incident Description: When using Shadow JDBC Connect in certain environments and defining the JDBC connection pool, errors would result.

This problem was due to the code that was placed into Shadow JDBC Connect to enable binary tracing.

Resolution: The issues with the binary tracing code have been corrected.

SI-8543

Incident Description: When the LGPA (Convert Strings to Params) keyword on Shadow JDBC Connect is set to YES, quotes in the literal string sent to DB2 would cause the following error:

```
[NEON][SCODBC_R.DLL][DB2]DSNT408I SQLCODE = -102, ERROR: LITERAL
STRING IS TOO LONG
```

DB2 determines the length of a literal ***before*** processing embedded quotes that are escaped (two for one); however, Shadow JDBC Connect determined the literal length ***after*** quote processing, which prevented literals that are too long from DB2's perspective to be changed to parameters.

Resolution: The literal length determination has been updated to count embedded quote characters that will be removed, to match the processing by DB2.

SI-8566

Incident Description: When using Shadow JDBC Connect and saving the time in a DB2 time or timestamp column using a specific format, that format was not returned when the column was viewed.

Resolution: The time zone conversion logic in `setTime`, `setTimestamp`, `getTime`, and `getTimestamp` was corrected. For the set functions, the time is in local time and the calendar object is the time zone of the database. The set functions convert the local time to the database time zone. For the get functions, the calendar object is the time zone of the database and the returned time is local time. The get functions convert the database time to the local time zone.

SI-8665

Incident Description: For non-DB2 connections (i.e., when DBTY is set to anything other than DB2) with AUST=YES (to enable auto-static), Shadow JDBC Connect would incorrectly issue statements in some cases, causing the statement to fail. For example, with the Shadow JDBC Connect for ADABAS, when using DBTY=ADABAS, issuing a CONNECT statement to connect to ADABAS would result in an error.

Resolution: Because auto-static support only applies for DB2 connections, for non-DB2 connections, a check was added for AUST=YES; if found, the value is set to NO and an informational message is logged.

Shadow Server Version: This resolution requires Shadow Server 4.8 SVFX3021 or higher.

SI-8711

Incident Description: When using Shadow JDBC Connect for non-DB2 and non-ADABAS connections (i.e., when DBTY is set to anything other than DB2 or ADABAS), the autocommit clauses (SET AUTO-ON and SET AUTO-OFF) were

incorrectly being treated as SQL statements and were being sent to the Shadow Server in CALL statement wrappers; thus, the statements were not being executed.

Resolution: For non-DB2 and non-ADABAS connections, Shadow JDBC Connect now recognizes the autocommit clauses (SET AUTO-ON and SET AUTO-OFF) and passes them “as is” (unwrapped), so they are properly executed.

SI-8733

Incident Description: When using Shadow JDBC Connect on AIX 5.2 in certain environments, attempts to run a JSP would cause Shadow JDBC Connect to “hang”.

Resolution: The problem was resolved by replacing `getuser()` with `getprocs()`.

SI-8755

Incident Description: Attempts to use Shadow Enterprise Transactions (two-phase commit) were unsuccessful, resulting in the following error in the Shadow Server trace browse:

```
DSNHLI XA-START - SQLCODE -6, IMPROPER CONTEXT FOR REQUEST -
SDBC1010
```

Resolution: XA commands have an associated transaction management flag (TMFLAG). For improved diagnostics, the XA TMFLAG text was added to the Shadow Client Trace Facility output. Informational text was added to the trace entry that explained the hex value.

Required PTFs: The resolution requires that the PTFs for APAR OA02556 be installed to fix an RRS problem.

SI-8798

Incident Description: When using Shadow JDBC Connect with LGID=ESN or LGID=ESP (i.e., the language ID is set to Modern Spanish or Castilian Spanish), the special characters “ñ” and “Ñ” were not translated correctly.

Resolution: The ASCII-to-EBCDIC and EBCDIC-to-ASCII translation tables for LGID=ESN and LGID=ESP were corrected to match the following:

	EBCDIC	ASCII
Ñ	7B	D1
ñ	6A	F1

SI-8810

Incident Description: When trying to pass “null” (in lowercase) as the value to a CHAR parameter in a stored procedure, the stored procedure failed with the following error:

```
[NEON][SCOD32.DLL]Invalid characters after numeric literal
```

However, the same stored procedure call succeeds if “null” is changed to “NULL” (in uppercase) in the call statement.

Resolution: The comparison being performed by Shadow JDBC Connect was corrected so that all mixed cases of the word “null” are now supported.

SI-8843

Incident Description: When attempting to use the DB2 ROUND function (which is available in DB2 v6 and higher) with Shadow JDBC Connect, the query would fail with the following error:

```
Unsupported scalar function in escape syntax
```

Resolution: ROUND was added was a recognized DB2 scalar function.

SI-8965

Incident Description: The Shadow JDBC Connect version number was displayed in different formats in various locations.

Resolution: The Shadow JDBC Connect version number will now be displayed in “major_version.minor_version.build_number” format without any reference to the JDK version (for example, the current Shadow JDBC Connect version would be displayed as “3.9.366”).

SI-9034

Incident Description: When executing DB2 stored procedures in certain environments, the following error was received:

```
[NEON][SCOD32TS.DLL]Parameter SQLDA build in buffer failed
```

Resolution: A check that caused zero length data to appear as erroneous was corrected. Now, the defined length of a CHAR input parameter will be changed from 0 to 1 and one (1) byte of NULL data will be sent to prevent SQLERROR -804 from occurring.

SI-9037

Incident Description: When using Shadow JDBC Connect on Windows XP with LGID=ENG (i.e., the language ID is set to U.K. English), the English pound sign (“£”) was being incorrectly converted to a space and not being sent correctly.

The C function `isspace()` was being called incorrectly for byte values larger than 0x7f (127); this led to the English pound sign ("£") being incorrectly identified as a space character.

Resolution: A cast of the value passed to `isspace()` was added to "usigned char", so now it is correctly identified.

SI-9119

Enhancement: This enhancement introduces a way to completely eliminate base tracing by setting LOG=NULL as follows:

```
NEONTRACE=INFO FLUSH THREADID JDBCLOG=/tmp/jdbclog.txt LOG=NULL
```

Note:

In the past, users could partially disable base tracing as follows:

```
NEONTRACE=INFO FLUSH THREADID
JDBCLOG=/tmp/jdbclog.txt LOG=/dev/null
```

However, this did not completely disable the tracing, it simply eliminated the I/O to the trace file; thus, significant overhead could still be incurred.

SI-9136

Incident Description: The CRIN (Create Table Index Automatically) Shadow JDBC Connect keyword functionality was not operating properly. When functioning properly, a flag would be set for "CREATE TABLE" SQL statements so that after execution, a routine would called that searched the statement to determine what indexes need to be created. However, if the statement was committed, the flag was being cleared, so if Shadow JDBC Connect was running in auto-commit mode, the indexes were not being created automatically.

Resolution: The functionality of the CRIN (Create Table Index Automatically) Shadow JDBC Connect keyword was re-implemented. This keyword controls whether an index is automatically created when you create a table with a primary key or unique constraint.

SI-9175

Incident Description: Attempts to insert BLOB data using Shadow JDBC Connect would result in the following error:

```
[NEON][SCOD32TS.DLL]C binary data has been truncated
```

Resolution: When SQLGetData is called the first time with cbValueMax = 0 and the resulting column is one of the variable length types, data conversion will be skipped so that just the length of the data in pcbValue is returned.

SI-9197

Incident Description: Problems would result when using Shadow JDBC Connect on UNIX platforms if the `/etc/NeonShadow.conf` file existed. Although this file is not required, it was being installed with all Shadow JDBC Connect installations.

Users affected by this issue may experience the following:

- "Unexpected Signal" warnings.
- Core dumps.

Resolution: This file will no longer be installed.

SI-9220

Incident Description: When using Shadow JDBC Connect to execute a stored procedure that contains brackets within a literal string, a SQLError would occur (however, no error message would be returned).

Resolution: A parsing problem with escape sequences was found and corrected.

SI-9221

Incident Description: When attempting to use a .NET application, the following error would result:

```
SQLCODE = -519, ERROR: THE PREPARE STATEMENT IDENTIFIES THE SELECT
STATEMENT OF THE OPENED CURSOR SQL_CUR51
```

This problem occurred because cursors were not being closed properly. The cursor open flags were being erroneously cleared, preventing close cursor commands from being sent to the host.

Resolution: When a COMMIT is done for auto-commit, the cursor open flags will no longer be cleared, thus allowing close cursor commands to be sent to the host.

SI-9242

Incident Description: When using Shadow JDBC Connect in certain environments, problems occurred during SQLDisconnect.

Resolution: The following changes were made to resolve the problem:

- The CLOSE THREAD call in SQLDisconnect will not be performed if the connection is broken.

- A new Shadow JDBC Connect keyword, DCTM (Disconnect Timeout Value), was added to allow users to specify a disconnect timeout value to control how many seconds Shadow JDBC Connect will wait for a socket to be ready for disconnect operations. This keyword will will solve certain SQLDisconnect hang problems that sometimes occur due to a broken socket or at times when

the Shadow Server is being recycled. The default value is 30 seconds. Setting the field to zero disables the timeout; however, this value should never be negative.

SI-9299

Enhancement: For Shadow JDBC Connect for ADABAS users, the SQLPREPARE prefix is now supported for the READ verb when SQLPrepare is called in situations where the Prepare/Open Optimization Feature is enabled (i.e., the DIPO (Disable Prepare/Open Optimization) Shadow JDBC Connect keyword is set to NO).

SI-9321

Incident Description: A javacore dump would result if the connection string exceeded the length limit.

Resolution: Shadow JDBC Connect was modified to handle longer connection strings.

SI-9403

Incident Description: When using Shadow JDBC Connect's XA support (two-phase commit support) in certain environments, a "hang" would occur due to an XA-RECOVER issue.

Resolution: The XA support offered by Shadow JDBC Connect was completely reworked.

SI-9450

Incident Description: On Windows, although the `neonlog.bin` binary logging file is not required, it was being installed with all Shadow JDBC Connect installations. This could result in WRITE permission problems for the file.

Resolution: The `neonlog.bin` file will no longer be installed.

SI-9453

Incident Description: If Virtual Connection Facility (VCF) TRANSBLOCK mode (i.e., the CNMD (Connection Mode) Shadow JDBC Connect keyword is set to TRANSBLOCK) was used in conjunction with the Fast Logon feature (i.e., the FALG (Fast Logon) Shadow JDBC Connect keyword is set to YES), the following problems would result:

- All batched operations would be lost, leading to various problems, including blocked statements being discarded instead of being executed
- SQLErrors were not being returned to the application because errors were not recorded properly.

Resolution: The interaction problem with CNMD=TRANSBLOCK and FALG=YES was corrected so that batched operations are properly executed. In

addition, code was added to retrieve and store DSNHLI errors so SQLErrors would be appropriately returned.

SI-9479

Incident Description: CICS COBOL program fields defined as S9(4)V9(2) COMP-3 packed decimal fields were not being translating correctly to Java BigDecimal fields and the following error was being returned:

```
[NEON][SCOD32TR.DLL][DB2]CALL CICSEX ERROR HAS OCCURRED RC -1180;
PARAMETER 033 IS TOO LONG
```

Resolution: PRAD support, which controls the precision adjustment Shadow JDBC Connect makes when passing values in a remote procedure call (RPC), was extended to statement parameters. Thus, during execution time, the precision for decimal parameters in an RPC are now adjusted to odd numbers to resolve the issue.

SI-9544

Incident Description: Queries that contained a LIKE clause and a character string failed to retrieve data; however, when the query was submitted using the primary key database, it was successful.

Resolution: SQL_CHAR parameters (fixed length character fields) were being padded with spaces, causing problems with the LIKE clause. A change was implemented so that SQL_CHAR parameters will only be padded when the data length is zero, thus correcting the problem. In addition, support was added for a null pointer as the pcbValueLen of SQLBindParameter. The pcbValueLen, which is the last argument, is the value size indicator, which specifies how to determine the length of actual data.

SI-9628

Enhancement: As part of the Shadow JDBC Connect type 2 improvements, JDBC dynamic tracing is now enabled by setting the JDBCLOG keyword of the NEONTRACE environment variable to a colon (“:”) value, as follows:

```
NEONTRACE= INFO JDBCLOG=:
```

SI-9772

Incident Description: The `dbmd.getSchemas()` call does not work properly for different data sources. Instead of returning the schema names available in the database, ordered by schema name, as suggested in the JDBC specification, Shadow JDBC Connect was incorrectly returning the schema name for each table in the database.

Resolution: For DB2 connections (i.e., DBTY=DB2), the functionality of `JStringToCString()` was corrected, which was causing a problem for

`SQLTables()`. For non-DB2 connections (i.e., when DBTY is set to anything other than DB2), the `dbmd.getSchemas()` call will return the following:

- **For DBTY=ADABAS:** Return “ADABAS” as the only schema.
- **For DBTY=VSAM:** Return “VSAM” as the only schema.
- **For DBTY=VSAMCICS:** Return “VSAMCICS” as the only schema.

Note:
The `SQLTables()` call does not work for non-DB2 connections.

SI-9976

Incident Description: The SQLDescribeParam call was causing excessive activity against the SYSIBM.SYSPARMS table in the DB2 catalog.

Resolution: The query used by Shadow JDBC Connect for SQLDescribeParam was optimized as follows:

- SELECT statements in SQLDescribeParam now use SPECIFICNAME instead of NAME. This improves performance when the catalog is SYSIBM because SPECIFICNAME is indexed, whereas NAME is not indexed.
- The ORDINAL predicate now uses either of the following:
  - IN list
  - ORDINAL <= number of parameters

SI-10027

Incident Description: Repeating SQLGetData calls for variable length columns were not returning the correct results. When a repeated SQLGetData call was made to a variable length column, instead of fetching the remaining data with subsequent calls, Shadow JDBC Connect was starting over at the beginning of the column with subsequent calls.

Resolution: Repeating SQLGetData calls for variable length columns will now allow applications to correctly retrieve the column data in pieces. After the data is completely retrieved, the next SQLGetData call will retrieve the data from the beginning again.

Note:
LOB type columns cannot be retrieved repeatedly.

SI-10400

Incident Description: SQLGetInfo with SQL_TXN_ISOLATION_OPTION was not returning the correct value for the SDBU1010 plan, which is bound with Uncommitted Read and should return SQL_TXN_READ_UNCOMMITTED.

Resolution: SQLGetInfo with SQL_TXN_ISOLATION_OPTION was corrected to return the following:

- SQL_TXN_READ_COMMITTED if the plan name (such as SDBC1010) indicates that the plan was bound with Cursor Stability (CS).
- SQL_TXN_SERIALIZABLE if the plan name (such as SDBR1010) indicates that the plan was bound with Repeatable Read (RR).
- SQL_TXN_REPEATABLE_READ if the plan name (such as SDBS1010) indicates that the plan was bound with Read Stability (RS).
- SQL_TXN_READ_UNCOMMITTED is the plan name (such as SDBU1010) indicates that the plan was bound with Uncommitted Read (UR).

SI-11525

Incident Description: When using Shadow JDBC Connect for non-DB2 connections (i.e., when DBTY is set to anything other than DB2), the use of prepared statements would result in the following error message because Shadow JDBC Connect was not allocating enough space for processing the input parameter values:

```
[NEON][SCOD32R.DLL]Not enough space for literal
```

Resolution: A correction was made in the function that replaces parameter markers in SQL statements from non-DB2 connections.

SI-11560

Incident Description: In previous versions of Shadow JDBC Connect, DECIMAL columns were returned as NUMERIC, but with Shadow Client 3.9, DECIMAL columns were returned as DECIMAL to conform with the Sun JDBC specification for java.sql.Types; however, this change caused problems with certain existing applications.

Resolution: To minimize the impact to existing applications, a new Shadow JDBC Connect keyword, DENU (Decimal as Numeric) was introduced to specify whether DECIMAL and NUMERIC columns should be reported as DECIMAL columns or NUMERIC columns (because both types of columns are treated as the same by Shadow JDBC Connect). Possible values are as follows:

- YES: All DECIMAL and NUMERIC columns will be reported as NUMERIC. This option restores the previous behavior of Shadow JDBC Connect (prior to 3.9).
- NO: (Default) All DECIMAL and NUMERIC columns will be reported as DECIMAL. This behavior conforms to the Sun JDBC specification for java.sql.Types.

SI-12332

Incident Description: When using jDemo, graphic columns were not displayed properly.

Resolution: The mapping of JDBC data types in Shadow JDBC Connect was corrected.

Shadow Server Version: GRAPHIC data type support requires Shadow Server 4.8 SVFX3824 or higher.

SI-12393

Incident Description: When using Shadow JDBC Connect for IMS/DB via SQL access, if the SUBSYS (Subsystem) Shadow JDBC Connect keyword was set to NONE, the metadata calls were not properly sent to the Shadow Server.

Resolution: Shadow JDBC Connect will check for connections with the DBTY (DBMS Type) Shadow JDBC Connect keyword set to DB2 and the CPFX (Catalog Prefix) Shadow JDBC Connect keyword set to IMS, which indicates a connection for Shadow JDBC Connect for IMS/DB via SQL access. In such cases, the connection will send metadata calls to the Shadow Server for SQLTables, SQLColumns, SQLStatistics, SQLSpecialColumns, SQLPrimaryKeys, and SQLForeignKeys.

SI-13608

Incident Description: When issuing a SQL call that returns LONGVARCHAR columns, an exception error would sometimes occur during the SGLGetData call. The problem only occurred when using SQLGetData to process multiple rows of data if SQLGetData was called against a VARCHAR or LONGVARCHAR column without retrieving all of the data available.

Resolution: A problem with passing a parameter incorrectly to the free allocated storage function was corrected.

SI-13737

Incident Description: When using the single signon support with RPCs, a SQLCode -805 would be returned, as follows:

```
-805, ERROR: DBRM OR PACKAGE NAME ... OPRXSQ ...
```

The problem was caused by a SET-SQLID (Set Current SQLID) call being issued incorrectly.

Resolution: For single signon support, Shadow JDBC Connect was modified to prevent sending a secondary userid value if one was not specified in the data source settings or in the connection string.

SI-14201

Incident Description: When deploying Shadow JDBC Connect on application servers on Linux, certain problems such as core dumps would sometimes occur.

Resolution: On Linux, all the global variables and functions defined in the STUB were sharing a single copy. This created various problems. Declaring them to be “static” solved the problems.

SI-14962

Incident Description: When running in XA mode for two-phase commit support and using auto-static (i.e., the AUST (Change Dynamic SQL to Auto-Static) Shadow JDBC Connect keyword is set to YES), various failures would sometimes occur.

Resolution: If the XAEN (X/OPEN XA Support) Shadow JDBC Connect keyword is set to TWO-PHASE, indicating XA mode, Shadow JDBC Connect will automatically disable auto-static by setting the AUST (Change Dynamic SQL to Auto-Static) Shadow JDBC Connect keyword to NO.

Other Updates

BINARY Column Handling

Using Shadow JDBC Connect to call `ResultSet.getString()` on a BINARY column would cause binary data to be treated as ASCII and converted accordingly; however, in some cases, this resulted in column data that could not be read. Thus, when calling `ResultSet.getString()` on a BINARY column, the data will now be converted to hexadecimal code. In addition, to accommodate the size of the hex representation of the data, SQL_COLUMN_DISPLAY_SIZE will be used to allocate buffers passed to SQLGetData.

LOB Handling

The handling of NULL and EMPTY LOB result set columns in SQLGetData was corrected. The change should provide performance improvements with NULL and EMPTY LOB columns because an unnecessary network round-trip has been removed.

XA Support

The following aspects of XA support (two-phase) commit support have been updated:

- JTS support.
- Auto-commit for two-phase commit transactions.
- XID hash code.

JTS Support

The following changes were made regarding JTS support, which is used by Shadow JDBC Connect users utilizing transaction managers compatible with the Java Transaction API (JTA) for two-phase commit.

Note:
To specify such a transaction manager, the XAOP (X/OPEN XA Transaction Manager) keyword on Shadow JDBC Connect must be set to JTS.

- **UNIX Platforms:** JTS support has been added to all UNIX platforms on which Shadow JDBC Connect is supported.
- **Auto-Commit:** Auto-commit is now supported for JTS.
- **Active Transaction Mode:** Under JTS, active transaction mode flags will now be cleared when transactions are completed. Previously, the active transaction flags were not cleared, which prevented Shadow JDBC Connect from starting new transactions.

Auto-Commit for Two-Phase Commit Transactions

Many changes were made to auto-commit support for two-phase commit transactions, as follows:

- Auto-commit will now be disabled after the XA_START succeeds. Shadow JDBC Connect used to send an extra call to the Shadow Server to turn off the auto-commit feature. This call has now been eliminated because auto-commit will be turned off automatically after the XA_START succeeds.
- After the XA transaction completes, auto-commit will be disabled.
- Instead of keeping track of the XA transaction status and delaying the setting of auto-commit until after `Connection.close()` is called, Shadow JDBC Connect will call `PooledConnectionListeners` before setting auto-commit. This is because upon being called, the listeners (application servers) will clean up all uncommitted XA transactions.
- A retry counter was added for XARMCreate.
- Various `Connection` class hierarchies were reorganized.

XID Hash Code Issue

In the past, XA operations would fail randomly. The problem was that XID hash code was not calculated correctly so the same XID could be treated as different. A correction was made to zero out the XID buffer so that Shadow JDBC Connect can come up with a consistent hash code.

Auto-Static Support

The following updates were made to Shadow JDBC Connect auto-static support:

- **Failure Handling:** When auto-static fails, Shadow JDBC Connect will now restore the original SQL statement, the number of parameters, and the number of literals.
- **Serialization:** Mutex was added for auto-static serialization.

Shadow JDBC Connect Version Information

The `DriverInfo.class`, which is used to return the name, version, build date, and other relevant information for Shadow JDBC connect has been updated as follows:

- **Syntax Update:** The `DriverInfo.class` syntax is now as follows:

  ```
  java DriverInfo -jdbc
  ```

- **Correction in ProxyLocal Use:** The `DriverInfo.class` was creating an instance of ProxyLocal when it should have used the existing one. Creating an extra instance of ProxyLocal caused an extra environment handle to be created, resulting in an error.

Known Issues

SI-8820

Incident Description: When using Shadow JDBC Connect on AIX under CICS/COBOL, running a volume test to gauge performance resulted in the following:

```
IWZ993W Insufficient storage. Cannot find space for ...
```

At this point, the CICS regions on the machines have essentially run out of memory.

When multiple environment handles are allocated, SQLFreeEnv incorrectly deallocates mutexes when more than one environment handle is still allocated, causing SQLDisconnect to fail.

Workaround: Setting the AUST (Change Dynamic SQL to Auto-Static) Shadow JDBC Connect keyword to NO solves the problem.

Changes Implemented: Although the complete resolution has not been implemented, the following changes have been made:

- When more than one environment handle is allocated, Shadow JDBC Connect will now bypass deallocating mutexes.
- All mutex related flags will be cleared after they are freed.
- The count of the number of allocated environment handles has now been serialized.

SI-9485

Incident Description: After installing Shadow JDBC Connect on Windows using silent mode, attempts to install result in the following error message:

```
'Can't launch executable'
```

This error only occurs when using an `installer.properties` file with the USER_INSTALL_DIR property explicitly set to a value. InstallAnywhere has a problem handling the forward slashes in the path (which it actually requires for the path specification); thus, the uninstall program will not be able to locate the JRE that was installed, which is necessary to run the uninstall.

Workaround: To solve the problem, you can manually add the following to the user's PATH prior to running the uninstall:

```
<INSTALL_DIR>\jre\bin
```

Where `<INSTALL_DIR>` is the full path specified via the USER_INSTALL_DIR property.

Also, if you install in silent mode and ***do not*** specify the USER_INSTALL_DIR property (i.e., you accept the default) the uninstaller will run without a problem.

SI-9197

Incident Description: Multi-threading issues exist for Shadow JDBC Connect on Solaris and Linux platforms where the `/etc/NeonShadow.conf` file exists. This file is not required.

Workaround: You can remove or rename the `/etc/NeonShadow.conf` file in order for the multi-threading to work.

Windows Considerations

Users should be aware of the following when installing and using Shadow JDBC Connect 3.9 on Windows systems:

- Installation program requirements.
- Upgrading from previous versions.

Installation Program Requirements

When installing Shadow JDBC Connect on Windows systems, users should be aware that InstallAnywhere requires the following:

- A JVM 1.2 or higher to run the installation file; thus the Shadow JDBC Connect installation includes Java™ 2 Runtime Environment (JRE), Standard Edition, Version 1.3.1. The JRE that is installed as part of the installation is "private"—it is put into the folder specific for Shadow JDBC Connect and it is not registered on the system (in the registry, path, classpath, etc.), so it is not actually seen on the machine by any application other than the Shadow JDBC Connect installation/uninstallation program.

- A color depth on the target system of at least 256 colors.

Upgrading from Previous Versions

JDBC Type 2 improvements result in important changes for users upgrading from previous versions:

- Data sources are now stored in the `SHADOW.INI` configuration file.

- Users must use the jConfig tool, provided with Shadow JDBC Connect, to create and configure data sources.

For details, see "Shadow JDBC Connect Improvements" on page 1.

UNIX Considerations

When installing and using Shadow JDBC Connect 3.9.366 on UNIX systems, users should be aware of the following:

- Installation program requirements.
- Upgrading from previous versions.
- General considerations.
  - File permissions.
  - GUI installation considerations.
  - Setting of environment variables
  - Common UNIX errors.
  - Uninstalling Shadow JDBC Connect.

Installation Program Requirements

Shadow JDBC Connect offers an installation program using InstallAnywhere, which offers an easy-to-use GUI that runs with X Windows; however, it can be specified to run via the console for users without X Windows.

InstallAnywhere requires the following:

- JVM 1.2 or higher to run the installation file; thus most Shadow JDBC Connect installations (with the exception of Linux/390) include Java™ 2 Runtime Environment (JRE), Standard Edition, Version 1.3.1. The JRE that is installed as part of the installation is "private"—it is put into the folder specific for Shadow JDBC Connect and it is not registered on the system (in the registry, path, classpath, etc.), so it is not actually seen on the machine by any application other than the Shadow JDBC Connect installation/uninstallation program.

  

  Note:
  Because the installation for Linux/390 does not include a JVM; users installing Shadow JDBC Connect on that platform must have JVM 1.2 or greater available to run the installation file.

- A color depth on the target system of at least 256 colors.

Upgrading from Previous Versions

JDBC Type 2 improvements result in important changes for users upgrading from previous versions:

- Data sources are now stored in the `SHADOW.INI` configuration file.
- Users must set the SHADOW_INI environment variable to specify the location of the `SHADOW.INI` configuration file. The ODBCINI and ODBC_INI environment variables are no longer supported. Thus, if users reference the ODBCINI or ODBC_INI environment variables in any script that uses JDBC or J2CA access (for example, startup scripts for application servers), they need to change the script to set the SHADOW_INI environment variable.
- Users can use the jConfig tool, provided with Shadow JDBC Connect, to create and configure data sources.

For details, see "Shadow JDBC Connect Improvements" on page 1.

General Considerations

File Permissions

Before attempting to install Shadow JDBC Connect on UNIX, the file permissions for the installation file must be updated to allow execution of the file, as follows:

```
chmod 777 IBMShadowClient_3_9_nnn_xxx.bin
```

Where nnn is the build number and xxx is the suffix reflecting the UNIX platform.

GUI Installation Considerations

If you will be using the GUI installation of Shadow JDBC Connect via X Windows, you must ensure that your DISPLAY environment variable is set appropriately.

Setting of Environment Variables

The Shadow JDBC Connect installation on Linux/390 will not set any environment variables, so all applicable environment variables must be set by the user.

Common UNIX Errors

GUI Installation Failure

Symptom: An attempt to launch the Shadow JDBC Connect installation in GUI mode fails with the following error:

```
$ ./IBMShadowClient_3_9_nnn_xxx.bin
Preparing to install...
Extracting the JRE from the installer archive...
Unpacking the JRE...
Extracting the installation resources from the installer
archive...
Configuring the installer for this system's environment...

Launching installer...

Invocation of this Java Application has caused an
InvocationTargetException. This application will now exit. (LAX)

Stack Trace:
java.lang.NoClassDefFoundError
        at java.lang.Class.forName0(Native Method)
        at java.lang.Class.forName(Class.java:115)
        at java.awt.GraphicsEnvironment.getLocalGraphics
        Environment(GraphicsEnvironment.java:53)
        at java.awt.Window.<init>(Window.java:183)
        at java.awt.Frame.<init>(Frame.java:310)
```

```
        at java.awt.Frame.<init>(Frame.java:257)
        at com.zerog.ia.installer.Main.c(Unknown Source)
        at com.zerog.ia.installer.Main.main(Unknown Source)
        at java.lang.reflect.Method.invoke(Native Method)
        at com.zerog.lax.LAX.launch(Unknown Source)
        at com.zerog.lax.LAX.main(Unknown Source)
GUI-
```

Description: This error indicates that the DISPLAY environment variable is not set appropriately.

Internal Error in scodbcco.c

Symptom: A message warning of an internal error detected in `scodbcco.c`, such as the following:

```
Wed May 14 16:41:25 2003 internal error detected: file scodbcco.c
line 3416 rc = 0 from yp_get_default_domain
Wed May 14 16:41:25 2003 internal error detected: file scodbcco.c
line 957 rc = -1 from scinnsck
```

Description: This message indicates that the Shadow Client Trace Facility is not configured properly; tracing has not been set to a valid log file. For example, in the following configuration, an appropriate path name is not specified:

```
NEONTRACE="INFO JDBCLOG=jdbc.log log=odbc.log"
```

Client Trace Initialization Failed

Symptom: A message warning that client trace initialization failed, such as the following (see the line in red, bold text):

```
Wed May 14 11:39:57 2003 SQLSetEnvAttr entered
Wed May 14 11:39:57 2003 SQLSetEnvAttr exited
Wed May 14 11:39:57 2003 Initializing Client Trace Environment
Wed May 14 11:39:57 2003 Client Trace Initialization failed
Wed May 14 11:39:57 2003 ODBC.INI file mutex created
```

Description: This message indicates one of the following, related to internal tracing performed by Shadow JDBC Connect:

- There is no `/etc/NeonShadow.conf` file present. This file configures the internal tracing and should be created automatically during the installation process.
- There is no `/opt/NeonShadow` directory present. This directory is the default location for the internal tracing.
- The specified tracing file (by default `/opt/NeonShadow/neonlog.bin`) is unreachable due to folder or file permissions.

Uninstalling the Shadow Client

When uninstalling Shadow JDBC Connect on UNIX, a message similar to the following will be returned upon the completion of the process:

```
./Uninstall_IBM_Shadow_Connect_Client_Install_3.9.nnn.0:
/opt/userdirectory/IBM/Shadow/UninstallerData: does not exist
```

Note:
This message is normal and expected.

Where:

/opt/userdirectory/NEON/Shadow
Indicates the directory in which the Shadow JDBC Connect components were installed. This depends on what was specified during the installation process.

This error message is a result of the fact that to perform the uninstall, you have used a directory（`/opt/userdirectory/IBM/Shadow/UninstallerData`) that ends up being removed during the uninstall process. During the uninstall, all directories for Shadow JDBC Connect will be removed; thus, when the uninstall is complete and the shell takes over again, it tries to verify the current location (which has been removed), so the error message indicates that the current directory does not exist.

To prevent this message from occurring, run the uninstall as follows:

```
cd /opt/userdirectory
```

```
./NEON/Shadow/UninstallerData/Uninstall_IBM_Shadow_Connect_Client
_Install_3.9.nnn.0
```