RogueWave SOFTWARE

Innovate with Confidence

SOURCEPRO® 2018 RELEASE NOTES

Introduction

In this Document

These release notes contain a summary of new features and enhancements, late-breaking product issues, migration guides, and bug fixes. This document covers changes to the product since the SourcePro 2016.3 release.

For change logs or other information on previous releases, please contact Technical Support.

NOTE » Rogue Wave products may contain undocumented interfaces. These interfaces are not supported for general use and may be changed or removed from release to release.

NOTE » The version of this document in the product distribution is a snapshot at the time the product distribution was created. Additional information may be added after that time because of issues found during distribution testing or after the product is released. To be sure you have the most up-to-date information, see the version of this document on the Rogue Wave web site: `http://www.roguewave.com/support/product-documentation/sourcepro.aspx`.

New Features and Enhancements

Updated Platform Support

- Specific platform information is not included in this document. Please see the Support Matrix at `http://www.roguewave.com/products/sourcepro.aspx`. Select *Learn about supported platforms* to see the latest supported platforms.

NOTE » Support for Intel C++ is deprecated and may be removed in a future release. For questions around the future of Intel C++ support in SourcePro, please contact technical support.

NOTE » Support for the SourcePro DB Access Module for Sybase is deprecated and may be removed in a future release. For questions around the future of Sybase support in SourcePro, please contact technical support.

SourcePro Core

Essential Tools Module

- *RWDate* is now implemented as a 32-bit integral type regardless of build type or platform. This provides for a consistent size, range of values, and behavior across platforms.
- Added a specialization of *RWTHash* for *RWDate*. (SP-25757)

SourcePro DB

- Added support for SQL CASE expressions. (SP-8891)
- Added a specialization of *RWTHash* for *RWDBDateTime*. *Note: RWDBDateTime is deprecated. RWDateTime should be used instead.* (SP-25758)

DB Access Module for MySQL

- Added support for prepared statements. (SP-8854)
- Added support for bulk writing via *RWDBBulkInserter*. (SP-25674)
- Added support for executing stored procedures via *RWDBOSql* and *RWDBStoredProc*. (SP-8915)

DB Access Module for Sybase

- Removed requirement for Sybase macros SYB_LP64 and SYB_LLP64 on the compilation line. (SP-26137)

Bug Fixes

SourcePro Core

Essential Tools Module

- Fixed memory allocation strategy in *RWOrdered* to achieve amortized constant time growth. (SP-24287)
- Fixed a bug in *RWTime* that limited the range of values that could be constructed from a `struct tm*`. ***Note: RWTime is deprecated. RWDateTime should be used instead.*** (SP-24997)
- Fixed a bug in *RWTime* where time zone information wasn't correctly handled during construction. ***Note: RWTime is deprecated. RWDateTime should be used instead.*** (SP-24997)
- Fixed bugs in *RWTPtrDeque*, *RWTPtrDlist*, *RWTPtrHashMap*, *RWTPtrHashMultiMap*, *RWTPtrHashMultiSet*, *RWTPtrHashSet*, *RWTPtrMap*, *RWTPtrMultiMap*, *RWTPtrMultiSet*, *RWTPtrOrderedVector*, *RWTPtrSet*, *RWTPtrSlist*, *RWTPtrSortedDlist*, *RWTPtrSortedVector*, and *RWTPtrVector* that prevented fully instantiating on a `const` type. (SP-25606, SP-25607, SP-25608, SP-25609, SP-25510, SP-25611, SP-25612)
- Updated the `apply()` function in *RWTPtrSlist* and *RWTValSlist* to be `const` correct. (SP-25687)
- Fixed a bug in *RWDecimalPortable* where equal values had different hashes. (SP-25762)
- Fixed a performance regression in *RWCollectableString* and other *RWCollectable*-derived types in their `compareTo()` and `isEqual()` functions. See the Migration Guide for additional information on the implications of this change. (SP-25943)

Internationalization Module

- Fixed a memory access error in `RWURegularExpression::setLocale()`. (SP-27023)

SourcePro Net

Secure Communication Module

- Dynamic builds on AIX platforms now use the `-brtl` linker flag. (SP-25711)

SourcePro DB

- Fixed an issue in which nested join expressions were not evaluated in the correct order. (SP-25640)
- Fixed an issue where binding an unknown type would result in an application crash. (SP-25748)
- Fixed an issue where `RWDBDatabase::createProcedure(const RWDBStoredProc&)` did not generate correct SQL. (SP-27009)

DB Access Module for Microsoft SQL Server

- Fixed an issue in which a client API call to fetch data did not return error information when a `SQL_SUCCESS_WITH_INFO` status was returned. (SP-25994)

DB Access Module for MySQL

- Fixed an issue where the *RWDBResult* returned from executing a *RWDBBulkInserter* resulted in an incorrect value from the `rowsAffected()` method. Fixed an issue where `RWDBOSql::rowsAffected()` could return incorrect values. (SP-25749)

DB Access Module for ODBC

- Fixed an issue where `SQLDescribeParam` was called out of sequence in some scenarios. (SP-26322)

DB Access Module for Oracle

- Fixed an issue where temporary LOBs allocated by the database were not being released. (SP-26981)

Migration Guide

SourcePro Core

Essential Tools Module

RWDate serialization incompatible with previous versions of SourcePro

RWDate is now implemented as a 32-bit integral type regardless of build type or platform. This can result in incompatibilities when sharing data with an application using a previous version of SourcePro.

Applications using *RWFile* to share *RWDate* representations should be updated to use virtual streams.

Applications using virtual streams to share *RWDate* representations must set the virtual stream version to ensure compatibility with existing SourcePro applications:

```
const unsigned requested_version = 3;

RWbostream bos(os);

const unsigned default_version = bos.version();
if (default_version < requested_version) {
    throw RWxmsg("Invalid stream version requested");
}

bos.version(requested_version);
```

The requested version number must be the same for both applications, and it should not be greater than the default stream version.

RWDate serialization on RWFile incompatible with previous versions of SourcePro

RWDate objects are stored to *RWFile* as 32-bit integral values regardless of build type or platform. This can result in incompatibilities when the serialized data is shared with an application using a previous version of SourcePro.

To avoid this problem, applications should use the virtual streams classes.

Pointer based template class function overloads using function pointers with value_type arguments have been removed

The following function overloads in pointer-based template classes taking `bool(*)(value_type, void*)` as an argument have been removed, as they prevented instantiation on a `const` type.

```
bool contains(bool(*)(value_type, void*), void*) const
value_type find(bool(*)(value_type, void*), void*) const
size_type index(bool(*)(value_type, void*), void*) const
size_type occurrencesOf(bool(*)(value_type, void*), void*) const
value_type remove(bool(*)(value_type, void*), void*)
size_type replaceAll(bool(*)(value_type, void*), void*, value_type)
size_type removeAll(bool (*)(value_type, void*), void*)
```

Instead, use the corresponding function taking a `bool(*)(const T*, void*)` argument. This affects the following classes: *RWTPtrDeque*, *RWTPtrDlist*, *RWTPtrOrderedVector*, *RWTPtrSlist*, and *RWTPtrVector*.

To modify the value, use `apply(void (*)(reference, void*), void*)`.

apply() functions in RWTPtrSlist and RWTValSlist are now `const` correct

The `const` qualifier has been removed from the `apply()` overloads without a `const` value parameter in the function pointer. If you need to modify the value in the container, you should call `apply()` from a non-const object.

Specialization of RWTHash on RWDate

A specialization of the class template *RWTHash* was added for *RWDate*. Applications that implement this specialization will fail to link.

To avoid this problem, the specialization should be removed from the application code.

RWCollectable-derived types no longer leverage dynamic_cast in compareTo()

RWCollectable-derived classes were modified so that their `compareTo()` and `isEqual()` functions no longer leverage `dynamic_cast` in their implementations. This change was made to address a performance regression from previous versions of the product that did not use a `dynamic_cast`, as well as to avoid throwing an exception from these classes if the parameter was not of the expected type (which could easily break containers that held heterogeneous collections of types).

RWCollectable-derived types in SourcePro that override the default `compareTo()` functions implement comparisons similar to the following:

```
int DerivedCollectableType::compareTo(const RWCollectable* rhs) {
    const int type_difference = isA() - rhs->isA();
    if (type_difference) {
        return type_difference;
    }

    const DerivedCollectableType* derived_rhs =
            static_cast<const DerivedCollectableType*>(rhs);

    // return the results of type-specific comparisons between this and
    // derived_rhs.
}
```

It is recommended that user-derived *RWCollectable* types follow a similar pattern in order to provide consistent results with Rogue Wave provided *RWCollectable* types.

This approach has drawbacks, particularly when deriving from types that implement `compareTo()`. In those cases, if base classes are compared against derived classes, the `isA()` check will report a difference, and the base class type comparison will not be performed. For those situations, the `compareTo()` function will need to be updated to account for both the base and derived types when determining if they can be compared.

SourcePro Net

Secure Communication Module

AIX dynamic builds now use the `-brtl` flag for linking applications

The Secure Sockets Package linker arguments for dynamic builds have been updated to include the `-brtl` linker flag that allows dependent libraries to be linked as shared libraries. Any application makefiles leveraging the Secure Communication Module should be updated to use the `-brtl` linker option.

SourcePro DB

Specialization of RWTHash on RWDBDateTime

A specialization of the class template *RWTHash* was added for the deprecated class *RWDBDateTime*. Applications that implement this specialization will fail to link.

To avoid this problem, the specialization should be removed from the application code.

Join expression evaluation order

Nested joins now have parentheses placed around them to clarify the order of evaluation to the database.

Previously this code snippet:

```
RWDBJoinExpr join =
    rwdbNaturalLeftOuter(table1, rwdbRightOuter(table2, table3).on(table2["pn"] == table3["pn"]))
```

would have generated this SQL:

```
SELECT * FROM J1 t1  NATURAL LEFT JOIN J2 t2  RIGHT OUTER JOIN  J3 t3  ON ( t2.pn = t3.pn )
```

As formed, this query can be problematic since each database must apply its own join expression evaluation order rules. Parentheses to inner joins now clarify the order in which to evaluate the query, so that the following SQL will now be generated:

```
SELECT * FROM J1 t1  NATURAL LEFT JOIN  (J2 t2  RIGHT OUTER JOIN  J3 t3  ON ( t2.pn = t3.pn ) )
```

To generate a logically equivalent join expression to the original query, update the order in which joins are constructed:

```
RWDBJoinExpr join =
    rwdbRightOuter(rwdbNaturalLeftOuter(table1, table2), table3).on(table2["pn"] == table3["pn"]))
```

which will generate this SQL:

```
SELECT * FROM (J1 t1 NATURAL LEFT JOIN J2 t2 ) RIGHT OUTER JOIN  J3 t3  ON ( t2.pn = t3.pn )
```

Known Limitations

Installation

General

SourcePro should not be installed into an existing directory

Each SourcePro release is a replacement for, not an upgrade to, the previous version. Always perform an installation into a new area, and not into the same install directory as a previous version.

Windows

The user needs to have administrator privileges to run the installer

If you don't have administrator privileges, the installer will prompt you to enter the administrator's password.

UNC installation paths are not supported

For network installs on Windows, using a UNC path to specify the installation location is not supported. The problem is actually in RCB which, in such installs, is unable to create valid makefiles. So instead, create a mapped network drive to the desired network location. For the mapped network drive, be sure to check "Reconnect at Logon" checkbox. If this option is not selected, the mapped network drive will not appear when browsing for an install location.

RCB

Missing Top-Level Build Results HTML Page

Whenever you invoke RCB to build some set of libraries, there should be a build results HTML page at `<buildspace>/records/results/index.html`. However, this occurs only the first time you invoke RCB; for subsequent builds, this file is missing. You can, however, find the same information by looking in the subdirectory for the buildtype of the build just performed. For example, the results for a 12d build would be found at `<buildspace>/records/results/12d/index.html`. (SP-15699)

No spaces in #include path strings

RCB-generated makefiles do not put quotes around include path strings. If the path to the installation directory (local buildspace), export buildspace, or a user-provided third party library contains spaces, then compile failures will result. One way to prevent these failures is to ensure that the paths for the locations in question don't contain spaces. However, this may not be possible in some environments. If it isn't possible to remove all spaces from the paths, select the build action of "just generate makefiles" and then perform the build through RCB. Once RCB has generated the required makefiles, alter the makefiles to quote the path(s) which contain spaces, then execute the relevant `makeall_ <buildtype>` script (located in the root of the buildspace) to perform your build manually. (SP-15689)

SourcePro

General

Optimization issues with Oracle Developer Studio 12.6 on Linux and Solaris/SPARC

Numerous optimization issues have been noted when compiling with `-xO4` with Oracle Developer Studio 12.6 on Linux and Solaris/SPARC.

To avoid these issues, we've reduced the optimization level from `-xO4` to `-xO3` for all SourcePro modules. (SP-25946)

SourcePro Analysis

General

Optimization errors with Oracle Developer Studio 12.6 on x86 (64-bit only)

Optimization issues have been noted when using Oracle Developer Studio 12.6 to build 64-bit binaries on x86 platforms.

To avoid these issues, we've changed the compiler target architecture from `-xarch=sse3` to `-xarch=sse2` for the Essential Math, Linear Algebra, and Business Analysis modules. (SP-24293, SP-25937, SP-25507)

Essential Math Module

Intel's Math Kernel Library 11.3 links incorrectly library libcpmt.lib

When building static SourcePro libraries with Intel++ and the Intel Math Kernel Library, the library `libcpmt.lib` is incorrectly linked. This may result in warnings that contain the message:

```
'MT_StaticRelease' does not match value 'MD_DynamicRelease'
```

This has been reported to Intel and should be fixed in its next release of the MKL. You can suppress this warning with the linker option: `/NODEFAULTLIB:libcpmt.lib`.

Currency Module

Inexact exception for results with insignificant trailing zeros

Multiplication where the result contains insignificant trailing zeros may cause an inexact exception to be thrown, even if the result is represented exactly. In the following example, in which the class *RWDecimal*<*RWMultiPrecisionInt*<*2*>> can represent numbers with 18 decimal digits, an inexact exception will be generated:

```
RWDecimal< RWMultiPrecisionInt<2> > valA = "5282.80977864";
RWDecimal< RWMultiPrecisionInt<2> > valB = "1000000000";
RWDecimal< RWMultiPrecisionInt<2> > valMul = valA * valB;
cout << "valA : " << valA << endl;
cout << "valB : " << valB << endl;
cout << "valMul: " << valMul << endl;
```

The program creates the following output:

```
[MULINEXACT] RWDecimal: inexact result in multiplication
```

```
valA : 5282.80977864
valB : 1000000000
valMul : 5282809778640.00000 // 18 digit result still exact
```

In one sense, the inexact exception is correct, since a certain number of significant digits are lost in the calculation. It just happens that those digits are all zeros. (SP-10262)

`-fp-model precise` recommended when compiling with Intel C++

Intel C++ compiler optimizations include storing floating point variables in registers. On some machines this results in greater precision than a double should have.

According to the Intel C++ compiler User's Guide:

```
-fp-model option or /fp:precise (Windows) Tells the compiler to strictly adhere to
value-safe optimization when implementing floating-point calculations.  It disables
optimizations that can change the result of floating-point calculations, which is
required for strict ANSI conformance
```

This advice applies to portions of the Currency Module code.

We strongly recommend that you add `-fp-model precise` (Linux) or `/fp:precise` (Windows) to the command line of any non-debug programs built using the Currency Module and the Intel C++ compiler.

`-ffloat-store` recommended when compiling with GCC

GCC compiler optimizations include storing floating point variables in registers. On some machines this results in greater precision than a double should have.

According to the GCC man page:

```
"For most programs, the excess precision does only good, but a few programs rely
on the precise definition of IEEE floating point.  Use '-ffloat-store' for such
programs."
```

This advice applies to portions of the Currency Module code, and we have disabled this particular optimization for non-debug library builds and example-program builds by adding the `-ffloat-store` option to the compiler command line.

We strongly recommend that you add `-ffloat-store` to the command line of any non-debug programs built using the Currency Module and the GCC compiler.

SourcePro Core

Essential Tools Module

Using the Solaris SunPro Compiler and the native C++ Standard Library

When using the Solaris SunPro compiler and the native C++ Standard Library, the number of bytes counted by an *RWAuditStreamBuffer* cannot be printed without forcing the compiler to invoke the member conversion operator. The following code demonstrates the issue:

```
RWAuditStreamBuffer buf;
RWbostream bcount(&buf);
std::cout << buf << std::endl;
```

This problem affects the Essential Tools examples `manual/auditbuf` and `manual/auditbufx`. The problem is not observed when using the GCC C++ Standard Library. (SP-9247)

HP aC++ Standard Library does not support locales other than "C"

The C++ Standard Library that comes with the HP aC++ compiler has problems with locales other than the "C" locale. This can cause failures in `RWAnsiLocale::stringToDate()`. (SP-10843 / HP CR# QXCR1000990079)

Internationalization Module

ICU binaries and Oracle Developer Studio with the GCC C++ Standard Library

ICU binaries built with Oracle Developer Studio and the GCC C++ Standard Library will have a runtime dependency on the GCC libraries bundled with the compiler. The runtime shared library path must be set to include the GCC libraries. (SP-24358)

Threads Module

User-defined stack attributes are not supported on Linux

The *Threads Platform Guide* documents that user-defined stack attributes are supported on Linux. This is not currently the case. We continue to investigate providing stack attribute support on this platform; however, we are unable to do so at this time. (SP-13419)

RWTOnlyPointer casts away `const` in copy constructor and assignment operator

The *RWTOnlyPointer* copy constructor and assignment operator take `const` *RWTOnlyPointer* references, but casts away `const` on the parameter in order to invalidate it.

The parameters should be changed to non-`const` references to be correct, but this could potentially break user code which depends on being able to pass a `const` parameter. This issue will be fixed in the future.

Do not use *RWTOnlyPointer* instances in containers that expect assignability and copy constructability. This could cause issues because the value passed to the copy constructor or assignment operator will be invalidated. (SP-10222)

Thread attribute support inconsistent on Solaris 11

Oracle Solaris 11 changed the behavior of various functions related to setting and retrieving thread attributes. These changes are incompatible with assumptions the SourcePro Threads Module makes and may lead to unexpected exceptions, silent failures, or other errors. For this release, it is recommended that users avoid thread attributes related to priority, scheduling or inheritance on Solaris 11. This behavior will be reviewed in a future release. (SP-14298)

SourcePro DB

DB Access Module for DB2 CLI

Linking DB2 10.5 fixpack 5 on Solaris requires explicit `-lsocket`

The DB2 client library is unable to load the socket libary on Solaris. When this happens the database returns:

```
SQL1042C An unexpected system error occurred. SQLSTATE=58004
```

The workaround is to add `-lsocket` to the link line of the executable. (SP-21466)

DB Access Module for MySQL

RWDBDatabase and multithreaded applications.

In multithreaded applications, producing *RWDBDatabase* objects using an `RWDBManager::database()` call in each thread simultaneously results in a segmentation fault. This is due to a MySQL native call that is not multithread-safe.

To work around this problem, serialize the production of the first *RWDBDatabase* object. Alternatively, call MySQL native API `mysql_server_init()` (`mysql_library_init()` for MySQL 5.x) before accessing the DB Access Module for MySQL. (SP-8553)

DB Access Module for ODBC - Informix Driver

Return values from stored procedures returned as result sets

ODBC allows the driver implementer to decide how to handle return values from stored procedures. A common convention is to support a bound return value. In this scenario, SourcePro DB allows retrieving the return value in the following manner:

```
RWDBDatabase db = ...
RWDBConnection conn = db.getConnection();
RWDBStoredProc sproc = db.storedProc("_procedurename_", conn);

// execute the stored procedure
sproc.execute(conn);

// retrieve the return value
RWDBValue val = sproc.returnValue();
```

The Informix ODBC driver supports multiple return values, and as a result follows an alternate convention, in which the return values are returned as a result set for the function. In that scenario, the return value can be retrieved from the result set, like so:

```
// execute the stored procedure
RWDBResult res = sproc.execute(conn);

// create a reader for the result set.
RWDBReader reader = res.table().reader();
RWDBValue val;

reader() >> val;  // retrieve the return value
```

This behavior is consistent with the legacy Informix Access Module.

Calling stored procedures with no parameters

ODBC defines the call semantics for a stored procedure with no parameters as:

```
{ call procedurename }
```

Informix does not support this syntax and instead uses an alternate syntax in this scenario:

```
{ call procedurename() }
```

The Informix syntax is interpreted by conforming ODBC implementations as calling the procedure procedurename with one argument, where that argument has a default value.

The ODBC Access Module's *RWDBStoredProcedure* implementation uses the first calling convention when invoking a stored procedure with no parameters. In order to call an Informix stored procedure with no parameters, use the *RWDBOSql* interface instead. For example:

```
RWDBDatabase db = /* ... */
RWDBConnection conn = db.getConnection();

const RWCString callStmt = "{ call procedurename() }";
RWDBOSql osql(callStmt, RWDBOSql::Procedure);

osql.execute(conn);
```

Dropping columns

The SQL syntax for dropping a column allows for an optional `COLUMN` keyword. The syntax is:

```
ALTER TABLE <table> DROP [COLUMN] <column>
```

Some databases require the optional `COLUMN` keyword, others allow it, and Informix prohibits it. The ODBC Access Module emits the `COLUMN` keyword, causing a conflict when connecting to Informix. To resolve this, use the *RWDBOSql* interface. For example:

```
RWDBDatabase db = /* ... */

RWDBTable table = db.table("table");
// ...

RWDBColumn col;
col.type(RWDBValue::Int).name("col");

// ...
table.dropColumn(col);
```

This last line could be rewritten to use `RWDBOSql` as:

```
char buf [1024];
snprintf(buf, sizeof buf, "ALTER TABLE %s DROP %s",
         table.name().data(), col.name().data());
RWDBOSql osql(buf);

osql.execute(con);
```

Operations involving a TEXT column

If you use RWDBBulkInserter to insert an array of strings with a length of 255 or less into a `TEXT` column, the insertion will fail and the Informix Database will return the error:

```
HY000 Illegal attempt to convert Text/Byte blob type.
```

Resolve this by specifying the width parameter while constructing the *RWDBTBuffer* that is bound against the `TEXT` column. For example:

```
// Bulk inserting 100 rows of strings with width of 256
RWDBTBuffer<RWCString> buff(100, 256);
buff[0] = "abc";

// ...
```

If you encounter the same Informix database error from any other operation, use the *RWDBOSql* interface along with *RWDBTBuffer*. For example:

```
RWDBDatabase db = /* ... */

RWDBInserter inserter = db.inserter("table");

RWCString str1;
RWDBBoundExpr expr1(&str1, &nullIndicator1);

inserter << expr1;
str1 = "abc";

inserter.execute();
```

Here's an example using *RWDBOSql* and specifying the width parameter of the *RWDBTBuffer* with the SQL statement obtained by using the *RWDBTracer* interface:

```
RWDBDatabase db = /* ... */

RWDBOSql sql("INSERT INTO table VALUES (?)");
RWDBTBuffer<RWCString> str1(1, 256);

sql << str1;
str1[0] = "abc";

sql.execute();
```

Clustered indexes

The SQL syntax for creating a clustered index is slightly different for Informix than other databases. The normal syntax is:

```
CREATE UNIQUE CLUSTERED INDEX <index> ON <table>(<column>)
```

Informix uses `CLUSTER` instead of `CLUSTERED`. The ODBC Access Module emits `CLUSTERED`, causing a conflict when connecting to Informix. Resolve this using the *RWDBOSql* interface. For example:

```
RWDBDatabase db = /* ... */

RWDBTable table = db.table("table");

// ...

RWDBColumn col;
col.type(RWDBValue::Int).name("col");

// ...

table.createIndex("index", schema, conn, true, true);
```

Rewrite the last line to use *RWDBOSql* as:

```
char buf [1024];

snprintf(buf, sizeof buf, "CREATE UNIQUE CLUSTERED INDEX %s ON %s(%s)",
         "index", table.name().data(), col.name().data());

RWDBOSql osql(buf);

osql.execute(con);
```

Inserting multibyte string literals not supported

Informix does not correctly handle inserting multibyte string literal values. Attempted insertions may result in either data corruption or an application crash, in some cases. Inserting multibyte strings via bound values works correctly. (SP-16804, IT02280)

Incorrect value returned by RWDBOSql::rowsAffected() when inserting with data-at-execution

`RWDBOSql::rowsAffected()` may return an incorrect value when a row is inserted using data-at-execution. Inserting a single row at a time avoids this issue. (SP-16579, IT02184)

Reading BIGINT values into a long long type is not portable

On some platforms, reading `BIGINT` data in to a `long long` datatype may result in the return of an incorrect value in which the upper and lower 32-bit values are swapped in the resulting output. This issue was observed with 32-bit Solaris x86 builds, but may affect other platforms as well. To ensure portability, it is recommended that values not be read in to the `long long` or `unsigned long long` datatypes. (SP-17188, IT02278)

Unable to describe default values for columns in a table schema

Attempting to describe a column that has a default value may result in incorrect or incomplete information. This issue affects only describing a column with a default value, creating a column with a default value and leveraging default values when inserting data are unaffected. (SP-16582, IT02230)

Binding 64-bit integral types not supported with input parameters

Binding a 64-bit integral type as an input parameter results in incorrect data being passed to the database. This affects only bound values; literal values are unaffected. (SP-17293, IT02552)

Trailing whitespace removed when inserting into a VARCHAR column

When strings are inserted into a `VARCHAR` column, trailing whitespace is removed. This behavior is not observed when inserting strings into a `TEXT` column. (SP-17187, IT01262)

Unable to set login timeout on Linux

On Linux, setting the login timeout (`RWDBODBCLibEnvironmentHandle::timeout`) has no effect. This is caused by a conflict in the handling of this value between unixODBC and the Informix driver. (SP-17187, IT02279)

Delimited names not supported

Informix requires delimited table names to be quoted when passed to ODBC API functions such as `SQLColumns` and `SQLProcedures`. This conflicts with the ODBC Access Module and other ODBC drivers that expect those names to be unquoted. Due to this limitation, delimited names are not currently supported with Informix. (SP-16743, IT01153)

DB Access Module for Sybase

Binding 8-byte long and unsigned long data

The Sybase Client-Library `CS_INT` datatype is strictly 4 bytes, even on 64-bit platforms. Binding a `long` or `unsigned long` on a platform with an 8-byte `long` results in data truncation. Use the `int` datatype instead. (SP-14608)

Limitations in cursor API when handling decimal columns

This library has problems handling decimal columns with `NULL` values through cursors. On some platforms, deleting a row with null values in decimal columns through a cursor causes a core dump. In some platforms, it returns non-null values for `NULL` columns. Sybase has identified this as a bug. (Sybase #94310 / SP-7765)

RWDBOSql limitation for RWCString data

If a user attempts to bind *RWCString* data having length greater than 16384 bytes into a column of type `TEXT`, the data gets truncated to 16384 bytes. This is due to the page-size limitation in Sybase.

A workaround is to insert the data as a literal rather than a placeholder. All DB Data Manipulation Classes will put *RWCString* data greater than 16384 bytes as literal in the SQL sent for execution. However, while using class *RWDBOSql*, the user must perform this before passing the statement to the *RWDBOSql* instance. (Sybase case #10832821 / SP-7789)

Bulk Insertion Limitations

The DB Access Module for Sybase supports bulk writing through the array interface provided by the Sybase Bulk-Library function `blk_bind()`, which is part of the Open Client and Open Server common libraries. However, Sybase Bulk-Library does not support insertion with column-list. Here are some other specifics on using bulk insertion with SourcePro DB and Sybase:

- The *RWDBBulkInserter* associated with the DB Access Module for Sybase can accept *RWDBTBuffer*<*RWDBBlob*> with elements up to 64K in width. Elements of width greater than 64K may corrupt the database.
- Bulk insertion is implemented based on the Sybase Bulk-Library, which may impact the recoverability of the database in case of failures. See the Sybase documentation on "Bulk-Library client programming" for full details.
- The Sybase Bulk-Library call `blk_bind()` doesn't report errors when `NULLS` are inserted into a column that doesn't accept `NULL` values.

Intermittent segmentation fault if Sybase vendor libraries are not linked

Executables using Sybase Access Module multithreaded dynamic libraries on Solaris and Linux need to link the Sybase vendor libraries. Not linking the Sybase vendor libraries to the executable has been observed to produce intermittent segmentation faults. The Sybase vendor libraries can be linked directly or by linking the Sybase Access Module library to the executable. The Sybase Access module examples link the Sybase Access Module library.

As these failures are intermittent, if you do not observe these failures, you do not need to link these libraries to your executables. (SP-8505/SP-9426/SP-9472)

DB Access Module for Oracle OCI

Oracle Heterogeneous Services requires explicit transaction management

When using Oracle Heterogeneous Services for SQL execution, you need to use explicit Transaction Management using `beginTransaction()` and `commitTransaction()`. By default, this module uses `OCI_COMMIT_ON_SUCCESS`, whereas heterogeneous services require transactions.

Oracle and LONG columns

When a result set fetched from Oracle has a `LONG` column in it and no buffer is bound for the `LONG` column, then Oracle returns either `ORA-03106: fatal two-task communication protocol` or `ORA-00127 dispatcher s does not exist`. (Oracle bug #4919068 / SP-8671)

Breaking out of polling operations with OCIBreak

Calling `OCIBreak` to terminate polling operations doesn't work. This results in ORA-03127 when the next operation is attempted on the connection. `OCIBreak` is called when:

- Reading into a *RWDBTBuffer*<*char*> when the buffer is not large enough to hold the data.
- Returning false or throwing an exception from `RWDBDataCallback::onFetch()`. This has been identified as Oracle bug #10606993.

Using SYS.XMLTYPE datatype

The Oracle datatypes `SYS.XMLTYPE` is supported by SourcePro DB, but extra steps are required in some cases. When inserting data using *RWDBInserter*, to use the best transfer mechanism the DB Interface Module needs to know that the underlying table has a `SYS.XMLTYPE` column. The easiest way to do this is by calling the `RWDBTable::fetchSchema()` method before creating your *RWDBInserter* instance. Here's an example that demonstrates this:

```
// CREATE TABLE foo(c NUMBER(9), xmlcol SYS.XMLTYPE);
RWDBTable foo = myDb.table("foo");
foo.fetchSchema(myConn);

RWDBInserter ins = foo.inserter();
int c;
RWCString xml;

// fill in data ...
ins << c << xml;
ins.execute(myConn);
```

Application crash on HP-UX if first connection is established within a spawned thread

Applications using the Oracle Access Module on HP-UX 11i v3 crash with a bus error if the first connection established by the application is inside a child thread. Increasing the thread stack size using `thread_attr_setstacksize()` resolves the issue. (SP-9006)

Library conflict between Oracle vendor libraries and Intel C++ on Linux

Using the Intel C++ compiler may result in unresolved symbols errors. Supplying the path to the lib subdirectory of the compiler installation as the first path for the linker to search for libraries (for example: `-L<path to compiler>/lib`) resolves the errors. (SP-8484)

In addition, the path to the Intel libraries should also appear before the path to the Oracle libraries in the `LD_LIBRARY_PATH` environment variable. (SP-8585)

SourcePro Net

Secure Communication Module

SSLv2 fallback support no longer supported by default

Clients using SSLv2 may no longer be able to connect to servers using TLSv1WithFallback. This is because the default cipher list used by OpenSSL 1.0.0 and later no longer includes the necessary ciphers for this fallback to work. While it is strongly encouraged to avoid using SSLv2, fallback support for SSLv2 can be enabled by enabling the necessary ciphers on a context. The following code snippet demonstrates.

```
RWSecureSocketContext context;
context.setCipherList("ALL:!NULL");
```

Documentation

Inter-document Hyperlinks do not Work in the PDF Documents

The PDF documents shipped with SourcePro are provided so customers can print the documents if they wish. These documents are not intended as primary access to the product documentation, and may not be as fully functional as the HTML documentation. One known issue is that hyperlinks between these documents do not work, and other issues may exist.

Windows 8 PDF Reader May Behave Inconsistently

The native Windows 8 PDF Reader may behave inconsistently; it has been observed that hypertext links are not processed correctly. Instead, use Adobe's free Acrobat Reader.

Rogue Wave provides software development tools for mission-critical applications. Our trusted solutions address the growing complexity of building great software and accelerates the value gained from code across the enterprise. Rogue Wave's portfolio of complementary, cross-platform tools helps developers quickly build applications for strategic software initiatives. With Rogue Wave, customers improve software quality and ensure code integrity, while shortening development cycle times.