Arrays

Contents

PostgreSQL™ provides robust support for array data types as column types, function arguments and criteria in where clauses. There are several ways to create arrays with pgjdbc.

The java.sql.Connection.createArrayOf(String, Object[]) can be used to create an java.sql.Array from `Object[]` instances (Note: this includes

both primitive and object multi-dimensional arrays). A similar method `org.postgresql.PGConnection.createArrayOf(String, Object)` provides support for primitive array types. The `java.sql.Array` object returned from these methods can be used in other methods, such as PreparedStatement.setArray(int, Array).

Additionally, the following types of arrays can be used in `PreparedStatement.setObject` methods and will use the defined type mapping:

Java Type	Default PostgreSQL™ Type
`short[]`	`int2[]`
`int[]`	`int4[]`
`long[]`	`int8[]`
`float[]`	`float4[]`
`double[]`	`float8[]`
`boolean[]`	`bool[]`
`String[]`	`varchar[]`

- Documentation
- Changelog
- FAQ

Introduction and explanation of symbols

Changes are sorted by "type" and then chronologically with the most recent at the top. These symbols denote the various action types:

= add,

= fix,

= remove,

= update

Version 42.2.5 (2018-08-27)

Notable changes

Changed

- `ssl=true` implies `sslmode=verify-full`, that is it requires valid server certificate cdeeaca4

Added

- Support for `sslmode=allow/prefer/require` cdeeaca4

Fixed

- Security: added server hostname verification for non-default SSL factories in `sslmode=verify-full` (CVE-2018-10936) cdeeaca4
- Updated documentation on SSL configuration fa032732
- Updated Japanese translations PR 1275
- IndexOutOfBounds on prepared multistatement with insert values c2885dd0

Commits by author

Christoph Berg (1):

- chore: remove editor backup files PR 1255 b1b1afb8

Dave Cramer (4):

- Update Contributing.md 1f8ac406
- Add issue templates PR 1263 c66bf710
- move issue template and pull request template into github specific di… PR 1283 b5c19af6
- docs: improve documentation on SSL fa032732

Kazuhiro Sera (1):

- docs: fix typos detected by github.com/client9/misspell PR 1287 9534e9ca

Kyotaro Horiguchi (1):

- fix: Japanese translation PR 1275 993a3beb

Sehrope Sarkuni (1):

- fix: Correct typo in CopyManager comment PR 1285 203a106d

Vladimir Sitnikov (12):

- reflect 42.2.4 release in readme.md b99cea0d
- doc: add SQL_TSI to the known issues for 42.2.3 [ci-skip][2da319a0](https://github.com/pgjdbc/pgjd
- docs: escape underscores in changelog otherwise it produces italic b1507f84
- docs: typo in 42.2.3 changelog "minute->second" 5898cdf1
- refactor: remove unused PgPreparedStatement.adjustIndex PR 1253 178eecc9
- test: run testShortQueryTimeout in PG_VERSION=HEAD Travis job only PR 1270 ecd412e4
- fix: remove POT-Creation-Date from generated .po and .pot files PR 1269 450a496b
- fix: use UTF-8 encoding in generated translation/messages_*.java files 2cbe7b35
- fix: typo in "One ore more ClientInfo failed" error message [ci skip][21e126f4](https://github.com/pgjdbc/pgjdbc/commit/21e126f451df667627c1cc3a0acfb3c38be45
- fix: IndexOutOfBounds on prepared multistatement with insert values PR 1289 c2885dd0
- security: implement SSL hostname verification for non-default (LibPQ-Factory) SSL factories (CVE-2018-10936) cdeeaca4
- Update changelog for 42.2.5 237a89bf

Contributors to this release

We thank the following people for their contributions to this release.

Christoph Berg
Dave Cramer
Kazuhiro Sera
Kyotaro Horiguchi
Sehrope Sarkuni
Vladimir Sitnikov

Version 42.2.4 (2018-07-14)

Notable changes

Changed

- PreparedStatement.setNull(int parameterIndex, int t, String typeName) no longer ignores the typeName argument if it is not null PR 1160

Fixed

- Fix treatment of SQL_TSI_YEAR, SQL_TSI_WEEK, SQL_TSI_MINUTE PR 1250
- Map integrity constraint violation to XA_RBINTEGRITY instead of XAER_RMFAIL PR 1175 f2d1352c

Commits by author

Christian Kotzbauer (1):

- Fixed typo in CHANGELOG.md PR 1249 b20df919

Dave Cramer (3):

- use the correct date for the release 985b63b7
- also fix spelling of error b0162c0a
- fix: setNull for types not in java.sql.Types (e.g. uuid) PR 1160 6287c954

Jan Van den Bergh (1):

- fix: map integrity constraint violation to XA_RBINTEGRITY instead of XAER_RMFAIL PR 1175 f2d1352c

Pavel Raiskup (1):

- packaging: sync RPM spec with Fedora Rawhide cc545d28

Vladimir Sitnikov (2):

- docs: fix broken commit link in 42.2.3 changelog 776e1717
- fix: treatment of SQL_TSI_YEAR, SQL_TSI_WEEK, SQL_TSI_MINUTE PR 1250 4668f43f

Contributors to this release

We thank the following people for their contributions to this release.

Christian Kotzbauer
Dave Cramer
Jan Van den Bergh
Pavel Raiskup
Vladimir Sitnikov

Version 42.2.3 (2018-07-12)

Notable changes

Known issues

- SQL_TSI_YEAR is treated as hour, SQL_TSI_WEEK is treated as hour, SQL_TSI_MINUTE is treated as second (fixed in 42.2.4)

Changed

- Reduce the severity of the error log messages when an exception is re-thrown. The error will be thrown to caller to be dealt with so no need to log at this verbosity by pgjdbc PR 1187
- Deprecate Fastpath API PR 903
- Support parenthesis in JDBC escape syntax PR 1204
- ubenchmark module moved pgjdbc/benchmarks repository due to licensing issues PR 1215
- Include section on how to submit a bug report in CONTRIBUTING.md PR 951

Fixed

- getString for PGObject-based types returned "null" string instead of null PR 1154
- Field metadata cache can be disabled via databaseMetadataCacheFields=0 PR 1052
- Properly encode special symbols in passwords in BaseDataSource PR 1201
- Adjust date, hour, minute, second when rounding nanosecond part of a timestamp PR 1212
- perf: reduce memory allocations in query cache PR 1227
- perf: reduce memory allocations in SQL parser PR 1230, PR 1233
- Encode URL parameters in BaseDataSource PR 1201
- Improve JavaDoc formatting PR 1236

Commits by author

AlBundy33 (1):

- fix: support parenthesis in JDBC escape syntax PR 865 38356e68

AlexElin (3):

- refactor: deprecate Fastpath API PR 903 f8e21b63
- refactor: migrate MultiHostsConnectionTest to JUnit4 PR 886 17a4d6a5
- refactor: simplify PgConnection.java PR 1047 ba360f73

Dave Cramer (3):

- fixed spelling mistake in PostgreSQL PR 1202 b92bd65a
- docs: improve javadocs in PgResultSetMetaData PR 792 825c0924
- minor language updates PR 1241 e19ee7ae

Hari Babu Kommi (2):

- spelling mistake correction PR 1181 e88abd79
- fix: set the loggerName in ConnectionFactoryImpl.log PR 1188 f78a639d

Jesper Pedersen (1):

- perf: guard logging statements PR 1112 7a0b7d65

Jorge Solorzano (6):

- fix: error on Travis build head PR 1186 354d2857
- test: add coverage for extendedCacheEverything PR 1062 f4d503c2
- Update after_n_builds to 10 PR 1193 2f9fed45
- test: drop OpenJ9 CI tests PR 1196 9b6506df
- fix: logger should be generally quiet PR 1187 30f06e14
- docs: improve CONTRIBUTING.md PR 951 38c8845e

KimBisgaardDmi (1):

- fix: getString for PGObject columns returns null PR 1154 bbb6c1f8

Marc Slemko (1):

- fix: allow disabling field metadata cache PR 1052 6ce91721

Michele Mancioppi (1):

- docs: clarify database and username for the build PR 859 83f2e385

Pavel Raiskup (1):

- packaging: fix RPM build requirements 6bb72e69

Sidi Mohamed EL AATIFI (1):

- Fix typos in java8-date-time.md PR 1174 dde8c020

Stephen Nelson (2):

- chore: add missing javadoc tags to avoid warnings PR 1164 e9ced455
- docs: correct the Javadoc and enforce with Checkstyle PR 1236 08631ccd

Vladimir Sitnikov (19):

- reflect 42.2.2 release in readme.md b1581e99
- fix: avoid NPE in PgConnection.finalize PR 1206 03a1441b
- chore: update gettext plugin, sort po files eb5c8fdd
- chore: sort messages in *.po files 10fc2fbb
- chore: remove obsolete translations ed1eab9e
- doc: add Russian translation to "No IOException expected..." eaa0acad
- fix: adjust date, hour, minute, second when rounding timestamp PR 1212 4dc98be8
- chore: remove ubenchmark module due to licensing issues PR 1215 88ec13bb
- chore: remove ubenchmark from packaging/rpm as well a699965a
- fix: support query timeouts exceeding 2147483 seconds (~25 days) PR 1224 b7fd9f3c
- perf: improve performance of replacing JDBC escapes PR 1230 177f63be
- docs: use union merge strategy for CHANGELOG PR 1107 70189203
- chore: use 5432 as default port when running code from IDE 5dc03f63
- docs: use "PR 42" references instead of "PRPR 42" in the changelog (#1239) f4ae60ec
- test: close of replication connection has not been fixed at backend side, so disable the test till 12.1 PR 1243 481460a3
- perf: avoid string allocation for oid/rows parsing in command tag PR 1232 da831de5
- docs: prepare changelog for 42.2.3 release c5f5d8d2
- chore: fetch contributors' URL via GitHub API d7297984
- docs: update site for 42.2.3

bazzargh (1):

- fix: encode url query parameters DataSource PR 1201 9f3838f7

benbenw (2):

- refactor: remove obsolete outParmBeforeFunc PR 1234 71028532
- perf: improve parsing performance of JDBC-style calls PR 1233 435e2f79

benoit (3):

- perf: reduce memory allocations when JDBC escapes () are used 2a1e0910
- refactor: use singleArgumentFunctionCall in EscapedFunctions 191d84eb
- perf: avoid BaseQueryKey.toString in CachedQuery.getSize PR 1227 669fc31e

bpd0018 (1):

- style: rephrase comment on named portals PR 1129 86c46f94

Étienne BERSAC (1):

- docs: fix link to GitHub documentation PR 1191 655b6e70

Contributors to this release

We thank the following people for their contributions to this release.

AlBundy33
AlexElin
bazzargh
benbenw
benoit
bpd0018
Dave Cramer
Étienne BERSAC
Hari Babu Kommi
Jesper Pedersen
Jorge Solorzano
KimBisgaardDmi
Marc Slemko
Michele Mancioppi
Pavel Raiskup
Sidi Mohamed EL AATIFI
Stephen Nelson
Vladimir Sitnikov

Version 42.2.2 (2018-03-15)

Notable changes

Added

- Documentation on server-side prepared statements PR 1135

Fixed

- Avoid failure for `insert ... on conflict...update` for `reWriteBatchedInserts=true` case PR 1130
- fix: allowEncodingChanges should allow set client_encoding=... PR 1125
- Wrong data from Blob/Clob when mark/reset is used PR 971
- Adjust XAException return codes for better compatibility with XA specification PR 782
- Wrong results when single statement is used with different bind typesPR 1137
- Support generated keys for WITH queries that miss RETURNING PR 1138
- Support generated keys when INSERT/UPDATE/DELETE keyword is followed by a comment PR 1138

Commits by author

Dave Cramer (5):

- Update java8-date-time.md 643e5583
- Update about 6fe76a3c
- Update documentation.md 3b3fd8c9
- Update documentation.md a8ef9f96
- test: add Travis configuration to test SSL PR 1095 298683b1

Jorge Solorzano (1):

- fix: improve DatabaseMetaData.getSQLKeywords() PR 940 7a586b6e

Pawel (1):

- Fixes #1096 PR 1097 df4e7fa0

Selene Feigl (1):

- fix: wrong data from Blob/Clob when mark/reset is used PR 971 61e1c300

Simon Stelling (1):

- fix: handle Timestamp values with fractional seconds < 1 microsecond correctly in PreparedStatement arguments PR 1119 8ff2a617

Vladimir Sitnikov (14):

- docs: reflect 42.2.1 release in readme.md 1a4256b9
- chore: make sure TEST_CLIENTS performs regular tests as well aa676bb3
- chore: remove unused variable lastKnownTime in ConnectionFactoryImpl 48b98971
- fix: ArrayIndexOutOfBoundsException when using the same SQL for regular and updateable resultset PR 1123 45c32bc6
- fix: support insert … on conflict…update for reWriteBatchedInserts=true PR 1130 1ca0c586
- fix: allowEncodingChanges should allow set client_encoding=… PR 1125 af64ed2d
- tests: UUID vs setString test PR 1133 5827858b
- fix: UUID test for preferQueryMode=simple 44bb7f8d
- fix: wrong results when a single statement is used with UNSPECIFIED types PR 1137 fcd1ea14
- test: workaround DST issue in StatementTest#testDateFunctions af499625
- docs: improve documentation and tests for server-side prepared statements PR 1135 4204f094
- test: make testAlternatingBindType Java 6-compatible PR 1139 bcdd4273
- fix: better support for RETURNING for WITH queries and queries with comments PR 1138 04e76661
- chore: add contributor links to release script 2568d38e

bpd0018 (3):

- docs: fix spelling and chapter, update sample code PR 1098 0cfffa84
- style: spelling in comment PR 1121 cc219aa7
- docs: fix JavaDoc for getPreferQueryMode() PR 1122 43d80cd4

chalda (1):

- Adjust XAException return codes for better compatibility with XA specification PR 782 e5aab1cd

trtrmitya (1):

- fix: use Locale.Category.DISPLAY (~lc_messages) when selecting resource bundle. PR 1115 0e9dfce4

Contributors to this release

We thank the following people for their contributions to this release.

Dave Cramer
Jorge Solorzano
Pawel
Selene Feigl
Simon Stelling
Vladimir Sitnikov
bpd0018
chalda
trtrmitya

Version 42.2.1 (2018-01-25)

Notable changes

Known issues

- client_encoding has to be UTF8 even with allowEncodingChanges=true

Changed

- socksProxyHost is ignored in case it contains empty string PR 1079

Fixed

- Avoid connection failure when `DateStyle` is set to `ISO` (~PgBouncer) Issue 1080
- Package scram:client classes, so SCRAM works when using a shaded jar PR 1091 1a89290e
- reWriteBatchedInserts=true causes syntax error with ON CONFLICT Issue 1045 PR 1082
- Avoid failure in getPGArrayType when stringType=unspecified PR 1036

Commits by author

AlexElin (1):

- test: check if url is not for PostgreSQL PR 1077 fe463bce

Alexander Kjäll (1):

- feat: add support for fetching 'TIMESTAMP(6) WITHOUT TIME ZONE' as LocalDate to getObject() PR 1083 09af4b23

Dave Cramer (1):

- fix: package scram:client classes, so SCRAM works when using a shaded jar PR 1091 1a89290e

Ivan (2):

- chore: remove braces for LeftCurlyCheck checkstyle PR 1075 c2664b44
- chore: remove additional braces for LeftCurlyCheck checkstyle PR 1076 975aaf5a

JCzogalla (1):

- Fixes issue #1078 PR 1079 0d51370b

Jamie Pullar (1):

- fix: getPGArrayType fails in when stringType=unspecified PR 1036 d5f1cf7c

Jorge Solorzano (1):

- Fix style changelog PR 1089 5ebd2090

Pavel Raiskup (1):

- packaging: update Fedora's CI dcbf70bc

Vladimir Sitnikov (3):

- docs: reflect 42.2.0 release in readme.md 6d02e958
- fix: avoid connection failure when DateStyle is set to ISO PR 1081 e442db1f
- fix: reWriteBatchedInserts=true causes syntax error with ON CONFLICT PR 1082 e133510e

Contributors to this release

We thank the following people for their contributions to this release.

AlexElin
Alexander Kjäll
Dave Cramer
Ivan
JCzogalla
Jamie Pullar
Jorge Solorzano
Pavel Raiskup
Vladimir Sitnikov

Version 42.2.0 (2018-01-17)

Notable changes

Known issues

- SCRAM authentication does not work as scram client classes are not packaged
- client_encoding has to be UTF8 even with allowEncodingChanges=true

Added

- Support SCRAM-SHA-256 for PostgreSQL 10 in the JDBC 4.2 version (Java 8+) using the Ongres SCRAM library. PR 842
- Make SELECT INTO and CREATE TABLE AS return row counts to the client in their command tags. Issue 958 PR 962
- Support Subject Alternative Names for SSL connections. PR 952
- Support isAutoIncrement metadata for PostgreSQL 10 IDENTITY column. PR 1004
- Support for primitive arrays PR 887 3e0491a
- Implement support for get/setNetworkTimeout() in connections. PR 849
- Make GSS JAAS login optional, add an option "jaasLogin" PR 922 see Connecting to the Database

Changed

- Improve behaviour of ResultSet.getObject(int, Class). PR 932

- Parse CommandComplete message using a regular expresion, allows complete catch of server returned commands for INSERT, UPDATE, DELETE, SELECT, FETCH, MOVE, COPY and future commands. PR 962
- Use 'time with timezone' and 'timestamp with timezone' as is and ignore the user provided Calendars, 'time' and 'timestamp' work as earlier except "00:00:00" now maps to 1970-01-01 and "24:00:00" uses the system provided Calendar ignoring the user-provided one PR 1053
- Change behaviour of multihost connection. The new behaviour is to try all secondaries first before trying the master PR 844.
- Avoid reflective access to TimeZone.defaultTimeZone in Java 9+ PR 1002 fixes Issue 986

Fixed

- Make warnings available as soon as they are received from the server. This is useful for long running queries, where it can be beneficial to know about a warning before the query completes. PR 857
- Use 00:00:00 and 24:00:00 for LocalTime.MIN/MAX. PR 992
- Now the DatabaseMetaData.getFunctions() implementation complies with the JDBC docs. PR 918
- Execute autosave/rollback savepoint via simple queries always to prevent "statement S_xx not exists" when autosaving fixes Issue #955
- Received resultset tuples, but no field structure for them" when bind failure happens on 5th execution of a statement Issue 811

Removed

- Drop support for the (insecure) crypt authentication method. PR 1026

Deprecated

- Reintroduce Driver.getVersion for backward compatibility reasons, mark it as deprecated as application should not rely on it (regression since 42.0.0) 50d5dd3e
- slave and preferSlave values for the targetServerType connection property have been deprecated in favour of secondary and preferSecondary respectively.

Commits by author

AlexElin (9):

- docs: fix header in CONTRIBUTING PR 902 38ff0fe
- refactor: remove dead code from PGStream, implement Closeable PR 901 acff949
- refactor: replace some usages of assertTrue PR 957 c759a58
- refactor: state of PGXAConnection as enum PR 966 7618822
- refactor: make PgStream implements Flushable PR 1008 0c3a2fc
- style: add MissingDeprecated into checkstyle PR 1019 d74386d
- chore: update checkstyle PR 1025 69e3b8b
- refactor: simplify methods in ConnectionFactoryImpl PR 1028 ed27c5b
- refactor: replace some usages of initCause PR 1037 0c29823

Álvaro Hernández Tortosa (1):

- Add SCRAM-SHA-256 support PR 842 befea18

Barnabas Bodnar (1):

- fix: don't attempt to read a SQLXML more than once PR 965 8f5e245

Brett Okken (1):

- feat: primitive arrays PR 887 3e0491a

Brett Wooldridge (1):

- Fixes #638 Implement support for get/setNetworkTimeout() PR 849 8a30044

Chen Huajun (1):

- fix: improve multihost connection for preferSlave case (verify expired hosts before connecting to cached master) PR 844 c6fec34

Dave Cramer (11):

- Update thread safety status of the driver to reflect reality; that being that the driver is not thread safe PR 928 ad47aba
- fix: use 00:00:00 and 24:00:00 for LocalTime.MIN/MAX PR 992 f2d8ec5
- fix: support Subject Alternative Names for SSL connections PR 952 2dcb91e
- test: Appveyor configuration PR 1000 059628f

- add test for identity, fix isAutoincrement in postgresql 10 fixes #130 PR 1004 2f6633b
- elaborate on sslmode options PR 1054 aa7a420
- prefer the word secondary over slave PR 1063 2e8c2b6
- Revert "refactor: replace some usages of initCause PR 1037" (#1064) e6a1ecc
- prefer secondary over slave referring to standby or secondary servers PR 1070 32c53902
- first pass at release notes and some fixes to previous notes PR 1041 a8260f5
- Update 2018-01-16-42.2.0-release.md b36867f

Hugh Cole-Baker (1):

- Make GSS JAAS login optional PR 922 d7f0f27

Jeff Klukas (1):

- fix: advance lastReceiveLSN on keepalive messages PR 1038 1be8a9e

Joe Kutner (1):

- fix: Added support for socksNonProxyHosts property PR 975 (#985) 9813c68

Jorge Solorzano (13):

- chore: use mainly Trusty in Travis, reorder CI jobs, and jdk tests PR 939 646a868
- fix: ignore replication test until 11.1 to avoid random failures PR 949 ee6443d
- chore: streamlining jobs PR 959 ed0a398
- docs: move changelog to separate file PR 956 e67e8f9
- docs: improve website front page PR 968 65170f1
- docs: fix test db password in docs PR 984 7df56f8
- test: add openj9 to the matrix PR 974 f187645
- chore: remove testing of the latest Java updates PR 993 0d8fde6
- chore: updates to CHANGELOG.md in release_notes.sh PR 981 bdfc1db
- test: querymode extendedCacheEverything PR 1007 f574285
- fix: first composite query not calling getNativeSql() PR 1020 2cae5a1
- drop old and unused crypt auth PR 1026 405f14e
- chore: collect coverage for Java 7 PR 1030 b629934

Magnus (1):

- fix: make warnings available as soon as they are received PR 857 83dd5fe

Magnus Hagander (1):

- Fix documentation spelling of sslpasswordcallback PR 1021 8ba5841

MichaelZg (1):

- fix: trim trailing zeros in timestamp strings returned in binary mode PR 896 d28deff

Michael Glaesemann (1):

- refactor: use TypeInfo getPGArrayType instead of munging type name PR 913 634e157

Pavel Raiskup (2):

- packaging: rpm_ci: add missing BuildRequires 4e0cdc1
- packaging: rpm_ci: don't shade scram jar into pgjdbc 1fd6c4f

Philippe Marschall (2):

- feat: improve ResultSet#getObject(int, Class) PR 932 fcb28c7
- test: add ubenchmark for UTF-8 decoding PR 988 0d918c3

Piyush Sharma (1):

- doc: Added quotes to URL in '@see' tag over org.postgresql.sspi.NTDSAPI#DsMakeSpnW for syntactic correctness PR 926 29f574a

Sehrope Sarkuni (1):

- feat: parse command complete message via regex PR 962 097db5e

Thach Hoang (2):

- Update ServerVersionTest to actually compare versions PR 1015 cccd6cd

- fix: always return Short[] for java.sql.Array.getArray() on smallint[] [PR 1017](https://github.com/pgjdbc/pgjdbc/pull/1017) 279fb43

Vladimir Sitnikov (23):

- fix: reintroduce Driver.getVersion for backward compatibility reasons PR 905 50d5dd3
- style: make PGReplicationStream, LargeObject implement AutoCloseable for Java 7+ PR 1016 9f07c9a
- fix: prevent statement hang in case close() called when query is in progress PR 1022 04c5dbb
- fix: synchronize Statement#result field access to make #close() more thread-safe 4139248
- fix: avoid reflective access to TimeZone.defaultTimeZone in Java 9+ PR 1002 fd0eeee
- fix: throw TOO_MANY_RESULTS (0100E) instead of "PgResultSet: tuples must be non-null" 0d31d46
- fix: "Received resultset tuples, but no field structure for them" when bind failure happens on 5th execution of a statement PR 811 082d009
- tests: correct assertion to use proper column 63918eb
- fix: add type parameter so code is Java 6/7 compatible 1361c52
- chore: avoid non-blocking IO for stdout to workaround "stdout: write error" in Travis 12bb084
- test: run Travis tests with non-default time zone a3982b4
- fix: execute autosave/rollback savepoint via simple queries always to prevent "statement S_xx not exists" when autosaving PR 955 684a699
- fix: use 'time with time zone' and 'timestamp with time zone' values as is and avoid computation with user-provided/default Calendars e8c43f3
- test: refactor SetObject310Test to use proper assertion messages and use less statements (make it faster) be06946
- refactor: factor out receiveParameterStatus so all the ParameterStatus messages are handled in the same way a94cfea
- fix: add Provide-Capability OSGi manifest PR 1029 236805b
- chore: update version to 42.2.0-SNAPSHOT to reflect the next release version e27ee74
- packaging: add missing maven-clean-plugin dependency a2ed9b5
- chore: introduce release via Travis acb9bdd
- chore: skip CI builds for tags; skip Fedora and extendedCacheEverything jobs when building pull requests 3ba3b63
- fix: avoid NPE from getObject(…, Date.class) and getObject(…, Calendar.class) on null timestamps PR 1071 eb33c4c
- test: add "as" to test queries so they work with PostgreSQL 8.3 71b3c11
- docs: make pgjdbc's javadocs to inherit base Java documentation eb406dc

Zemian Deng (3):

- refactor: use PGProperty enum instead of text ref for targetServerType, hostRecheckSeconds, loadBalanceHosts PR 912 (#915) b0cfc33
- fix: correct javadoc on PGResultSetMetaData.getFormat PR 917 cd77693
- fix: Correct DatabaseMetaData.getFunctions() implementation PR 918 8884202

bpd0018 (3):

- docs - change load.md to reflect current practice PR 1058 90535d9
- docs: fix the URL regex PR 1057 6c5490f
- docs: fix no parameter connect string example PR 1056 bb8a315

djydewang (1):

- style: disallowing user to use incomplete fully qualified Check names in config file PR 961 3286c8c

eperez (1):

- Someone forgot to get the next column PR 973 15aec6a

mjanczykowski (1):

- feat: add setURL method to BaseDataSource PR 999 2277ffb

rnveach (1):

- style: remove deprecated maxLineLength from LeftCurlyCheck PR 904 5f083d1

steinarb (1): * fix: add Provide-Capability org.osgi.service.jdbc.DataSourceFactory to OSGi manifest Issue 1029

zapov (1):

- fix: avoid integer overflow when sending large arguments PR 946 266ed61

Contributors to this release

We thank the following people for their contributions to this release.

AlexElin
Álvaro Hernández Tortosa
Barnabas Bodnar
Brett Okken
Brett Wooldridge
Chen Huajun
Dave Cramer
Hugh Cole-Baker
Jeff Klukas
Joe Kutner
Jorge Solorzano
Magnus
Magnus Hagander
MichaelZg
Michael Glaesemann
Pavel Raiskup
Philippe Marschall
Piyush Sharma
Sehrope Sarkuni
Thach Hoang
Vladimir Sitnikov
Zemian Deng
bpd0018
djydewang
eperez
mjanczykowski
rnveach
zapov

Version 42.1.4 (2017-08-01)

Notable changes

Changed

- Statements with non-zero fetchSize no longer require server-side named handle. This might cause issues when using old PostgreSQL versions (pre-8.4)+fetchSize+interleaved ResultSet processing combo. Issue 869

Commits by author

AlexElin (4):

- test: migrate tests to JUnit 4 PR 738 5b65e2f4
- style: update checkstyle + turn on some rules PR 847 246b759c
- test: migrate tests to Junit4 PR 883 5c12da16
- refactor: remove useless checks in the tests 0221f930

Dave Cramer (2):

- honour PGPORT, PGHOST, PGDBNAME in connection properties PR 862 2951a958
- doc: fix spelling mistakes PR 868 757db625

Michael Glaesemann (4):

- test: assume minimum server version 8.3 testing autosave with ALTER 77ee528d
- test: assume minimum server version 8.3 when testing with uuid ff2717e4
- refactor: remove unused import 8afe856e
- test: assume integer datetimes for timestamp tests PR 873 8287e7f9

Vladimir Sitnikov (7):

- docs: prevent "<!–more–>" from being displayed on the changelog page e1747693
- doc: fix anchors for "contributors to this release" c1d743f2
- test: fix StringTypeParameterTest to skip preferQueryMode=simple beca1692
- chore: install PostgreSQL 9.1 to Trusty builds via apt, and use Precise for Java 6 e960f237
- test: make StringTypeParameterTest 8.3+ since 8.2 misses enum types PR 882 ed0014cc
- fix: named statements were used when fetchSize was non-zero and prepareThreshold=0 PR 870 f0deabf7
- test: skip ConcurrentStatementFetch for PostgreSQL < 8.4 PR 884 5334cb6e

Contributors to this release

We thank the following people for their contributions to this release.

AlexElin
Dave Cramer
Michael Glaesemann
Vladimir Sitnikov

Version 42.1.3 (2017-07-14)

Notable changes

Fixed

- Fix NPE in PreparedStatement.executeBatch in case of empty batch (regression since 42.1.2). PR 867

Commits by author

Vladimir Sitnikov (2):

- doc: ensure changelog uses %Y-%m-%d format, not %Y-%d-%m 5d585aac
- fix: NPE in PreparedStatement.executeBatch in case of empty batch PR 867 7514552d

Contributors to this release

We thank the following people for their contributions to this release.

Vladimir Sitnikov

Version 42.1.2 (2017-07-12)

Notable changes

Changed

- Better logic for *returning* keyword detection. Previously, pgjdbc could be defeated by column names that contain *returning*, so pgjdbc failed to "return generated keys" as it considered statement as already having *returning* keyword PR 824 201daf1d
- Use server-prepared statements for batch inserts when prepareThreshold>0. Note: this enables batch to use server-prepared from the first *executeBatch()* execution (previously it waited for *prepareThreshold executeBatch()* calls) abc3d9d7

Fixed

- Replication API: fix issue in #834 setting statusIntervalUpdate causes high CPU load. PR 835 59236b74

Regressions

- NPE in PreparedStatement.executeBatch in case of empty batch. Fixed in 42.1.3

Commits by author

AlexElin (1):

- refactor: make PSQLState as enum PR 837 fb5df7fe

Dave Cramer (8):

- Initial support of partitioned tables via JDBC metadata API PR 823 9c3471f2
- fix javadoc complaints and some small edits to replication comments PR 832 2d0bfceb
- fix issue #834 setting statusIntervalUpdate causes high CPU load \ PR 835 59236b74
- fix issue #838 make sure we don't get columns that are dropped PR 840 464a2d43
- add missing connection documentation, fix spelling PR 846 cd400f6f
- more spelling mistakes for preferQueryMode PR 850 73bc3c1b
- fix formatting of section on failover, still not perfect but better PR 852 9f722014
- small reformat to clarify read and write connections PR 854 551d71b6

Jorge Solorzano (3):

- test: yet another rename to use "lsn" not "location" in Pg10. PR 822 90228621
- use zulu-9 PR 828 4ac74886
- fix: remove type name from cast exception of getBoolean and setObject PR 781 394b3a2f

Robert 'Bobby' Zenz (1):

- fix: Add fallback to setObject(int, Object) for Number PR 812 5b9edb7d

Vladimir Gordiychuk (1):

- bug: floating logical replcation test PR 829 2d3e8972

Vladimir Sitnikov (7):

- chore: implement a script to stage pgjdbc, pgdjbc-jre7, pgjdbc-jre6 artifacts 15d78839
- doc: fix 42.1.0.jre8->jre6 typo PR 42 88942b58
- fix: use server-prepared statements for batch inserts when prepareThreshold>0 abc3d9d7
- fix: better parsing for returning keyword PR 824 201daf1d
- docs: build index, changelog pages from _posts/… to reduce release overhead d6fe07d7
- chore: make ./release_notes.sh create docs/_posts/$DATE_YMD-$VERS-release.md file e00d4571
- docs: add 42.1.2 release notes 6f127a61

Contributors to this release

We thank the following people for their contributions to this release.

AlexElin
Dave Cramer
Jorge Solorzano
Robert 'Bobby' Zenz
Vladimir Gordiychuk
Vladimir Sitnikov

Version 42.1.1 (2017-05-04)

Notable changes

Fixed

- Fix infinite dates that might be corrupted when transferred in binary for certain JREs. For instance, 5881610-07-11 instead of infinity. 1e5bf563

Commits by author

Vladimir Sitnikov (2):

- fix: infinite dates might be corrupted when transferred in binary for certain JREs 1e5bf563
- chore: print "include notable changes" to the generated changelog 6bc4fe07

Contributors to this release

We thank the following people for their contributions to this release.

Vladimir Sitnikov

Version 42.1.0 (2017-05-03)

Notable changes

Added

- Support fetching a REF_CURSOR using getObject PR 809

Fixed

- Fix data being truncated in setCharacterStream (bug introduced in 42.0.0) PR 802
- Fix calculation of lastReceiveLSN for logical replication PR 801
- Make sure org.postgresql.Driver is loaded when accessing though DataSource interface Issue 768

Regressions

- There's no 42.1.0.jre6 version due to infinity handling bug. Fixed in 42.1.1.jre6

Commits by author

Alexander Kjäll (1):

- documentation typo PR 818 90f5556f

Daniel Migowski (1):

- feat: improve waiting for notifications by providing a timeout option PR 778 a7e0c83b

Dave Cramer (4):

- Update index 4d8b1b38
- Review the documentation for the replication API PR 756 3e1eb34d

- fix callproc escape documentation the specification PR 785 95a3f41d
- Honour setLogStream. If the logStream is set PR 780 b97ad630

Jacques Fuentes (1):

- Make replication docs use PREFER_QUERY_MODE PR 761 bd0497de

James (1):

- fix: use SQLWarning(String reason) constructor for correct DriverManager PR 751 74a426b9

Joe Kutner (1):

- fix: Only resolve hostname if not using a SOCKS proxy PR 774 480b0cf1

Jorge Solorzano (3):

- fix: build site with jekyll 2.2.0 PR 755 773ee679
- test: check that new properties follow correct lower camel case PR 740 3f2a02e1
- refactor: simplify Encoding class PR 765 ef8c6f96

Philippe Marschall (1):

- feat: support fetching a REF_CURSOR using getObject PR 809 4ab5ccb7

Robert Zenz (1):

- fix: function calls without parameters do not require parentheses 2958cc22

Vladimir Gordiychuk (2):

- chore: fix false alarm on check coredump PR 806 3883a846
- bug: fix calculation of lastReceiveLSN for logical replication PR 801 170d9c27

Vladimir Sitnikov (3):

- fix: make sure org.postgresql.Driver is loaded when accessing though DataSource interface PR 768 9c80adc2

- refactor: add encoding, fix expected/actual, use proper constructor 77cace40
- fix: infinity handling for java.time types PR 789 f375701b

slmsbrhgn (1):

- bug: fix data being truncated in setCharacterStream (the bug introduced in 42.0.0) PR 802 28c98418

Contributors to this release

We thank the following people for their contributions to this release.

Alexander Kjäll
Daniel Migowski
Dave Cramer
Jacques Fuentes
James
Joe Kutner
Jorge Solorzano
Philippe Marschall
Robert Zenz
Vladimir Gordiychuk
Vladimir Sitnikov
slmsbrhgn

Version 42.0.0 (2017-02-19)

Notable changes

Added

- Replication protocol API was added: replication API documentation. PR 550
- java.util.logging is now used for logging: logging documentation. PR 722
- Add support for PreparedStatement.setCharacterStream(int, Reader). ee4c4265

Changed

- Version bumped to 42.0.0 to avoid version clash with PostgreSQL version and follow a better sematic versioning. 46634923

- Ensure executeBatch() can be used with pgbouncer. Previously pgjdbc could use server-prepared statements for batch execution even with prepareThreshold=0. Issue 742
- Error position is displayed when SQL has unterminated literals, comments, etc. Issue 688
- Strict handling of accepted values in getBoolean and setObject(BOOLEAN), now it follows PostgreSQL accepted values, only 1 and 0 for numeric types are acepted (previusly !=0 was true). PR 732
- Return correct versions and name of the driver. PR 668

Removed

- Support for PostgreSQL versions below 8.2 was dropped. PR 661

Deprecated

- Deprecated PGPoolingDataSource, instead of this class you should use a fully featured connection pool like HikariCP, vibur-dbcp, commons-dbcp, c3p0, etc. PR 739

Regressions

- Data truncated in setCharacterStream. Fixed in 42.1.0
- No suitable driver found for jdbc:postgresql when using a DataSource implementation. Fixed in 42.1.0

You may have noticed the change in the versioning of the driver, you can read the FAQ for more information.

Commits by author

AlexElin (6):

- refactor: use varargs PR 681 50b7fe0f
- refactor: make HostChooser implement Iterable PR 645 3d37db78
- refactor: migrate to Junit4 PR 682 f4a067cc
- refactor: remove deprecated Utils' methods PR 678 0275d40f
- refactor: migrate tests to junit4 PR 685 faab4998
- refactor: remove checks for jdk version 1.4 (tests) PR 737 ee51dfce

Eric McCormack (1):

- fix: accept server version with more than 3 parts PR 741 8437f6c1

Jordan Lewis (1):

- feat: do not use pg_depend against PostgreSQL 9.0+ PR 689 62e25fba

Jorge Solorzano (22):

- refactor: remove support for postgresql < 8.2 PR 661 14e64be7
- fix: add query to support postgresql 8.2 without t.typarray PR 699 cb3995b5
- test: add CI tests against PostgreSQL 8.3 PR 710 436365b0
- fix: robust castToBoolean for setObject in PreparedStatement PR 714 edc2a14a
- refactor: remove unused V2ReplicationProtocol.java PR 718 7881e41e
- test: ignore tests that don't apply to Pg8.2 and Pg8.3 PR 703 3bc0951e
- style: reorder checkstyle in travis PR 721 ba812fb4
- refactor: remove charset property not used PR 709 f6fd5a5a
- fix: huntbugs on PgDatabaseMetaData, String concatenation in a loop PR 693 3a00ef94
- refactor: fix getDriverVersion, getDriverName and getJDBCMajor/MinorVersion methods PR 668 aa974341
- docs: reword supported versions, include datasources section, compare versions PR 673 b2cdd057
- test: fix test replication on PG_HEAD PR 734 3b406a18
- refactor: deprecated PGPoolingDataSource PR 739 55e2cd16
- fix: strict handling of getBoolean and setObject with postgres accepted values PR 732 4942f7d1
- docs: move docs from www/documentation/head PR 744 70e23c45
- test: fix replication test in Pg10 PR 746 63ed2129
- feat: use java.util.logging PR 722 43e6505e
- perf: short circuit Oid.BOOL in getBoolean PR 745 e69e4a1d
- fix: add isLoggable around parameterized logger PR 752 8b50cfe5
- docs: move www repository to pgjdbc/docs d4e99198
- add syntax highlight to documentation 8c035ade
- add more style 9c510e65

Pavel Raiskup (3):

- fix: sync with latest Fedora PR 637 a29ad80b
- packaging: rpm: update srpm generator 5c3c9239
- packaging: rpm_ci: use curl -L to download rawhide logs c8125cff

Philippe Marschall (3):

- refactor: clean up PgDatabaseMetaData PR 692 d32b077e
- refactor: delete Keyword enum PR 697 677e3c4c
- feat: support microsecond resolution for JSR-310 types PR 691 6b3a1efb

Roman Ivanov (2):

- config: move version of checkstyle to property PR 723 9ef7d6f1
- chore: upgrade checkstyle to 7.4, make checkstyle version configurable via property PR 725 e1a25782

Steve Ungerer (1):

- fix: ensure executeBatch() does not use server-side prepared statements when prepareThreshold=0 PR 690 aca26a07

Trygve Laugstøl (1):

- feat: connect the socket only if the socket factory created an unconnected socket PR 587 f75572be

Vladimir Gordiychuk (4):

- bug: fix not enscaped special symbol that fail build PR 686 b4604cd7
- feat: add replication protocol API PR 550 f48c6bb7
- test: fix drop replication slot on 9.4 for tests PR 696 c1c48bd7
- chore: Gather backtrace from core dump on CI PR 736 da5e4ef1

Vladimir Sitnikov (12):

- fedora: add BuildRequires: classloader-leak-test-framework 64b6750c
- tests: remove Class.forName(..driver..) from test code c99507b5
- test: add CI tests against PostgreSQL 8.2 PR 659 63ee60e2
- feat: display error position when SQL has unterminated literals, comments, etc PR 688 8a95d991
- feat: add support for PreparedStatement.setCharacterStream(int, Reader) PR 671 ee4c4265
- chore: update next version to 42.0.0-SNAPSHOT 46634923
- refactor: add CallableQueryKey#equals override 401c51a1
- doc: correct wording in readme regarding PostgreSQL versions used in pgjdbc regression testing 60391d75
- fix: revert ExpressionProperties to Properties in BaseDataSource 989b4d23

- fix: correct edge cases for setCharacterStream(int, Reader) f1190d8c
- test: add SharedTimerClassLoaderLeakTest to the regular test suite b3553276
- refactor: wrap more log statements with isLoggable check 07d225ec

bd-infor (1):

- docs: clarify handling of loglevel PR 711 6334bac0

Contributors to this release

We thank the following people for their contributions to this release.

AlexElin
bd-infor
Eric McCormack
Jordan Lewis
Jorge Solorzano
Pavel Raiskup
Philippe Marschall
Roman Ivanov
Steve Ungerer
Trygve Laugstøl
Vladimir Gordiychuk
Vladimir Sitnikov

Version 9.4.1212 (2016-11-02)

Notable changes:

- ? can now be used in non-prepared statements (regression was introduced in 1210)

AlexElin (1):

- chore: add coverage badge (#646) 05c2f8d

Jorge Solorzano (6):

- docs: split readme.md to contributing.md (#652) 7270435
- docs: improve readme.md (#666) cf6b836
- docs: change license to BSD-2-Clause (#660) b4c90c8

- style: update the header in the project (#662) 8be516d
- fix: copy src/main/resources (#676) 14f3fce
- style: copy license file to META-INF folder (#677) c518697

Sebastian Utz (1):

- fix: makes escape processing strict according to JDBC spec PR#657 00a8478

Vladimir Sitnikov (4):

- fix: do not convert ?, ? to $1, $2 when statement.executeQuery(String) is used (#644) PR#643 08e1a40
- tests: make CursorFetchTest.testGetRow to use stable order of rows so the test does not depend on actual implementation of UNION 3c8efe4
- fix: avoid ClassLoader leaks caused by SharedTimer PR#664 f52bf7f
- Make successful OSGi activation less noisy PR#672 22b9025

zapov (1):

- fix: Don't break when there is no column metadata (#663) 6d2a53e

Contributors to this release

We thank the following people for their contributions to this release.

AlexElin
Jorge Solorzano
Rikard Pavelic
Sebastian Utz
Vladimir Sitnikov

Version 9.4.1211 (2016-09-18)

Notable changes: * json type is returned as PGObject like in pre-9.4.1210 (fixed regression of 9.4.1210) * 'current transaction is aborted' exception includes the original exception via caused-by chain

Daniel Gustafsson (1):

- doc: fix brand names and links in readme (#636) 56c04d0

Vladimir Sitnikov (4):

- chore: eploy to Travis was silently broken for a couple of days since “test” command was missing 67d7e3c
- fix: json should be returned as PGObject, not as String for backward compatibility reasons (#640) PR#639 beaec3a
- test: add DebugNonSafepoints to FlightRecorderProfiler PR#622 154c463
- feat: include root cause exception in case transaction fails (#628) 51775c1

Contributors to this release

We thank the following people for their contributions to this release.

Daniel Gustafsson
Vladimir Sitnikov

Version 9.4.1210 (2016-09-07)

Notable changes:

- Better support for RETURN*GENERATED*KEYS, statements with RETURNING clause
- Avoid user-visible prepared-statement errors if client uses DEALLOCATE/DISCARD statements (invalidate cache when those statements detected)
- Avoid user-visible prepared-statement errors if client changes search*path (invalidate cache when set search*path detected)
- Support comments when replacing JDBC syntax
- Support for Types.REF_CURSOR

AlexElin (3):

- style: change method names to start with a lower case letter (#615) 22c72b4
- refactor: fix some hintbugs warnings (#616) dd71f6f
- refactor: use varargs in Gt.tr (#629) 07c7902

Chrriis (1):

- perf: cache result set column mapping for prepared statements PR#614 88fbbc5

Marios Trivyzas (1):

- perf: add point type info to Client (#612) ba14509

Mathias Fußenegger (2):

- doc: fix test README link in README (#609) c09be96
- perf: add json type info to Client (#610) 4ad2df3

Pavel Raiskup (1):

- packaging: sync spec file with Fedora package PR#608 dd48911

Philippe Marschall (1):

- feat: support Types.REF_CURSOR PR#635 b5c3f59

Vladimir Gordiychuk (1):

- test: fix Travis job so it actually tests against HEAD PostgreSQL (#630) 9030373

Vladimir Sitnikov (23):

- test: skip json testing on pre-9.2 databases (those do not implement json type) 0a17d82
- chore: make sure JDK9+PG 9.4 Travis job indeed uses PG 9.4 01b65c3
- test: add PostgreSQL HEAD to pgjdbc regression test matrix (#613) PR#561 65a32ff
- test: add benchmark for resultSet.getByName ff4bfda
- chore: use Travis' PostgreSQL 9.5 d1feb58
- docs: add javadoc badge to the project readme 5d7a9bb
- refactor: remove autoCommit argument from QueryExecutor#createSimpleQuery/createParameteriz 756363e
- fix: support cases when user-provided queries have 'returning' PR#488 c3d8571
- feat: support execute statements via simple 'Q' command PR#558 232569c
- feat: show proper error message when connecting to non-UTF-8 database PR#594 ec5fb4f
- test: add tests for Parser.parseDelete/Select/Move/WithKeyword 3b7c7c4
- chore: update to 1.1.0 version of parent pom to upgrade bndlib 2.3.0 -> 2.4.0 4019ed6
- chore: use latest java version for one of Java 8 Travis jobs 0452e79

- refactor: introduce ResultHandlerBase to factor out common error processing logic edcdccd
- feat: reset server-prepared statements on deallocate/discard, ability to autorollback on sqlexception from executing a query PR#451 adc08d5
- fix: add a test case when "deallocate all" detector is turned off aa53fba
- fix: invalidate prepared statement cache when "set search_path=..." query is detected PR#496 17c8afb
- fix: properly account cache size when duplicate entries returned to the cache 191ccf2
- feat: support 10+ version parsing Note: 10.2 means 10 major, 2 minor, that is 10*00*02 9.2 means 9.2 major, that is 9*02*00 PR#631 a639431
- fix: honor comments when replacing {fn curdate()} kind of calls PR#632 2d9b313
- chore: skip "deploy to Central" step when building PRs 9bc194a
- test: add performance test for "reused vs non-reused at client side" prepared statements 2d70385
- chore: improve ./release_notes.sh so it does not require parameters d8736b4

Contributors to this release

We thank the following people for their contributions to this release.

AlexElin
Christopher Deckers
Marios Trivyzas
Mathias Fußenegger
Pavel Raiskup
Philippe Marschall
Vladimir Gordiychuk
Vladimir Sitnikov

Version 9.4.1209 (2016-07-15)

Notable changes:

- BUG: json datatype is returned as java.lang.String object, not as PGObject (fixed in 9.4.1211)
- Many improvements to `insert into .. values(?,?) -> insert .. values(?,?), (?,?)...` rewriter. Give it a try by using `reWriteBatchedInserts=true` connection property. 2-3x improvements for insert batch can be expected
- Full test suite passes against PostgreSQL 9.6, and OpenJDK 9

- Performance optimization for timestamps (~TimeZone.getDefault optimization)
- Allow build-from-source on GNU/Linux without maven repositories, and add Fedora Copr test to the regression suite

AlexElin (2):

- style: code cleanup and Java 5' features 8e8dbab
- style: cleanup and Java 5' features for tests (#602) 28c0c22

Christian Ullrich (4):

- docs: fix test password in readme closes #553 PR#553 07f8610
- test: make tests independent from server LC_MESSAGES (#554) b756a15
- fix: update waffle version PR#555 db1a6e4
- test: add basic tests for SSPI authentication PR#557 16c27b9

Christopher Deckers (2):

- fix: improve insert values(…) batch rewrite PR#580 510e6e0
- perf: cache timezone in statement#setDate, resultSet#getDate, etc PR#588 d2b86a0

Dave Cramer (5):

- use binary trick for tolower, optimize switch, added tests 2200a4d
- fix:bugs with parse 72945b0
- fix: added test for move and fixed checkstyle errors fc59851
- fixed index ASC/DESC for 9.6 (#569) c9e5fc8
- test: DataSource.getConnection().unwrap(PGConnection.class) PR#573 f6b176e

Florin Asăvoaie (1):

- Refactored an unnecessary conversion of List to String[][] when calling the sendStartupPacket. (#544) 2032836

George Kankava (1):

- fix: PgDatabaseMetaData: close statement in finally PR#516 b4c45ca

Jeremy Whiting (3):

- perf: Add optimization to re-write batched insert statements. ac8abf3
- fix: Change optimization to delay parameter and statement re-write to immediately before execution. e591577
- feat: Added to SQL parser to allow inspection of Statement attributes and Command. 4ddb693

Laurenz Albe (1):

- fix: interpretation of empty but set "ssl" connection property PR#528 91f05b4

Minglei Tu (1):

- feat: support java.sql.Types.TIME*WITH*TIMEZONE and java.sql.Types.TIMESTAMP*WITH*TIME in setNull method PR#570 1b73bf6

Pavel Raiskup (3):

- Allow build-from-source on GNU/Linux without maven repositories (#546) 87489a9
- chore: test Fedora packaging CI via Travis job PR#578 1eb4085
- packaging: heal Fedora build (#601) af50d0b

Petro Semeniuk (1):

- fix: use per-connection cache for field metadata (table name, column name, etc) PR#551 dc3bdda

Philippe Marschall (4):

- style: Use more generics PR#519 88e39a0
- style: remove unused code PR#520 f21f168
- refactor: Remove pgTypeName null check PR#525 462928b
- refactor: remove ClassCastException catch PR#527 8bee06b

Tanya Gordeeva (1):

- fix: add a socket timeout on cancel requests PR#603 ab2a6d8

Vladimir Sitnikov (23):

- test: add tests for null::float8[][e6b5bb3](https://github.com/pgjdbc/pgjdbc/commit/e6b5bb3575696
- test: createArrayOf(..., null) test a274321
- test: avoid "The connection attempt failed" in ConnectTimeoutTest.testTimeout PR#531 fbabfa4
- fix: NPE in DatabaseMetaData.getTypeInfo when types are dropped concurrently PR#530 f3b0fd0
- fix: binary timestamptz -> getString should add +XX zone offset to text representation PR#130 1e1f3c4
- refactor: "build without waffle/osgi" PR 766f806
- test: add PostgreSQL 9.6 to CI tests (#560) 3ff47da
- chore: make sure ubenchmark is tested with checkstyle PR#564 f6ed8e6
- chore: add explicit MCENTRAL=Y to Travis jobs that deploy to Maven Central 18ca8f2
- test: prune some Travis jobs to make CI faster c5c1d7c
- chore: propagate java source/target to maven-compiler-plugin d8117d1
- chore: add codecov.yml fb2977f
- refactor: use enum instead of int for PgStatement.statementState to simplify debugging 32d4e08
- test: use TestUtil.getPort() in V3ParameterListTests instead of hardcoded 5432 9a4b296
- chore: make codecov to always wait 7 builds 9f9aa95
- test: add jdk9 Travis job PR#565 bbb0d35
- fix: incorrect binary data format in bind parameter X when using batch execution (#582) 7388dc9
- perf: execute "SET application_name" if name has changed only PR#537 893c1a4
- fix: fix some sonarqube warnings 7311b4e
- fix: revert array naming to pre 1202 behavior (e.g. _int4) PR#595 1d8ebfc
- refactor: rename dmlcommand -> sqlcommand 79db127
- perf: implement fast-path to TimeZone.getDefault if the cache field is accessible through reflection 6b3f2e0
- feat: make connectTimeout=10 (seconds) by default b4d5976

aryabukhin (1):

- feat: add support for url-encoded JDBC URL property values PR#532 4c15f31

Marc Petzold (1):

- feat: support HSTORE in PgPreparedStatement#setObject(int, java.lang.Object, int, int) 785d0c7

goeland86 (1):

- fix: avoid NPE in new PgArray(, null).toString() PR#526 74b4972

Contributors to this release

We thank the following people for their contributions to this release.

AlexElin
Christian Ullrich
Christopher Deckers
Dave Cramer
Florin Asăvoaie
George Kankava
Jeremy Whiting
Laurenz Albe
Minglei Tu
Pavel Raiskup
Petro Semeniuk
Philippe Marschall
Tanya Gordeeva
Vladimir Sitnikov
aryabukhin
Marc Petzold
goeland86

Version 9.4.1208 (2016-02-16)

John Harvey (1):

- chore: rework update-translations.sh into pom file as a profile e2fa9ec

Dave Cramer (3):

- fix:regression from previous behaviour where setObject(index,object,VARHCAR) should call getString if it can't cast it to a string PR#482 9d6389c
- fix:correct comment PR#482 bfd73c7
- Update README.md PR#482 5a956a3

George Kankava (3):

- fix: squid:S1206 -equals(Object obj) and hashCode() should be overridden in pairs PR#515 a07d6d3

- fix: squid:S2325 - private methods that don't access instance data should be static PR#514 7aac95c
- fix: squid:S1488 - Local Variables should not be declared and then immediately returned or thrown PR#513 299c9f5

Gilles Cornu (1):

- docs: update travis-ci badge PR#475 f9405db

Jeremy Whiting (1):

- fix: load the ssl configuration property when defined through Properties. a6a61be

Markus KARG (1):

- docs: fix broken link to testing README PR#481 0a11144

Philippe Marschall (7):

- style: use java generics instead of raw types PR#463 c10acf0
- refactor: do not synchronize on ConcurrentHashMap PR#482 ea4cadd
- perf: add guards around debug log statements PR#469 d77aa41
- fix: Do not swallow security exceptions PR#471 beab720
- feat: implement resultSet.getObject(col, Class) with JSR-310 support PR#482 9aa3142
- feat: implement JSR-310 support in setObject e52f7e3
- fix: support local date times not in time zone 61384ec

Rikard Pavelic (1):

- perf: cache result of parsing server_version PR#464 9c43d27

Vladimir Sitnikov (24):

- doc: update current versions in readme.md PR#482 4f5d57c
- refactor: move implementation of Prepared and Callable statements to PgPreparedStatement and PgCallableStatement PR#459 8fca8b4
- style: add import order check PR#482 77a188c
- style: align import order with style convention PR#482 00c7eb3
- doc: add gitter chat link to readme PR#482 53190c9
- style: enable more checkstyle verifications PR#482 b0225f6

- style: reformat code to fix checkstyle violations PR#482 8f1c9d7
- style: reformat with Eclipse PR#467 2d5e7fa
- Update IDEA config PR#482 7083008
- docs: fix typos in code comments PR#472 5d43712
- style: update translations 484eafd
- doc: update russian translations 92011d7
- test: add insert .. on conflict tests for PostgreSQL 9.5 d622a9f
- fix: improve handling of DATE columns around DST dates in binary transfer 642b48a
- test: improve insertBatch test aea9383
- test: track code coverage PR#494 0f979c3
- test: PostgreSQL 8.4 and 9.5 databases, XA configuration PR#499 8c9898a
- fix: make sure executeBatch returns error response for rows that would not get into database PR#502 d6e3b17
- test: add benchmarks for insert via copy and insert via array of structs 880244e
- fix: PgArray returning null for binary arrays PR#504 b225535
- doc: backend protocol, wanted features PR#478 ee6118e
- fix: OSGi require-capability manifest entry PR#497 a3e2045
- fix: make sure {fn now()} jdbc translation is not performed in dollar-quoted strings PR#511 9109451

mtran (1):

- feat: ability to customize socket factory (e.g. for unix domain sockets) PR#457 dc1844c

Contributors to this release

We thank the following people for their contributions to this release.

John Harvey
Dave Cramer
George Kankava
Gilles Cornu
Markus KARG
Rikard Pavelic
Jeremy Whiting
Vladimir Sitnikov
Stephen Nelson
Philippe Marschall
mtran

Version 9.4.1207 (2015-12-23)

Jeremy Whiting (3):

- Changed property loader to avoid using a default when property not already set. Allows the caller to detect when unset. Thus allowing connection Cto use programmatic log level value. Added test cases to check PGProperty method working properly. PR#438 ecf4a2c
- Update testcase to set properties where TestUtil methods look them up. PR#438 b86e07a
- Restore boot time ssl property after test cases complete. Revert change to assertion test. Safely remove existing ssl property. PR#438 274f382

Philippe Marschall (2):

- Fix null check in getTablePrivileges PR#453 244ce59
- Useless condition in parseQuery PR#454 dd229d5

Stephen Nelson (1):

- chore: migrate the build to Maven PR#322 f470f05

Vladimir Sitnikov (12):

- doc: add coding guidelines to readme PR#442 af62c6d
- chore: remove docbkx/pgjdbc.xml from reposiory PR#435 92b65f0
- chore: move files to "pgjdbc" maven module to prepare for pre-processor fix PR#435 f95b44e
- chore: use java comment preprocessor to build jre6/jre7/jre8 jars from the same sources PR#435 42c2e3b
- doc: fix javadocs warnings when building via java 8 PR#435 3b10873
- test: improve CopyLargeFileTest performance by dropping non required index and foreign key PR#435 410a662
- test: speedup DriverTest.testConnectFailover by adding a connect timeout of 5 seconds PR#435 ed1a916
- chore: skip gpg signature when deploying snapshot artifacts via Travis PR#449 d196cf4
- chore: use custom settings.xml when deploying snapshots via Travis PR#449 cf40cf2
- test: add test for insert batch that changes server-prepared statement names PR#449 e5ac899
- fix: prepared statement "S_2" does not exist in batch executions PR#449 fa310e0

- doc: add note on how to skip tests when building from source PR#460 823e124

Yao Chunlin (1):

- Fix bug when call XAResource.start with TMJOIN flag, the old localAutoCommitMode lost. PR#434 df09e2b

Contributors to this release

We thank the following people for their contributions to this release.

Jeremy Whiting
Vladimir Sitnikov
Stephen Nelson
Philippe Marschall Yao Chunlin

Version 9.4-1206 (2015-11-25)

Andrea Catalucci (1):

- Fixed typo in Driver.java.in PR#428 2589020

Dave Cramer (3):

- make sure logs actually get written PR#422 2ad367e
- fix: Binary handling of empty arrays. The type of the empty array was not properly read as a result getting the type of the array caused an NPE fixes Issue #421 reported by Juha Komulainen PR#422 38d8488
- fix:building on java 1.6 PR#422 9133012
- fix test case to actually check the value don't mask the exception in the event that it is thrown PR#422 e975c07
- add an explicit message for the test case PR#422 270b1a3

Laurenz Albe (3):

- Replace question marks only in PreparedStatements PR#427 3d30a4c

- Allow both single and double question marks in simple statements PR#427 39d510c
- Prettify code and add another regression test PR#427 2f8a67f

Vladimir Sitnikov (3):

- chore: fix dos end-of-lines PR#418 63d1dd3
- test: add tests for getBigDecimal of int4 field in both text and binary modes PR#426 faac288
- fix: fix invalid values when receiving int2, int4, int8 via getBigDecimal() PR#424 e6f1beb

John K. Harvey(1)

- fix: update translations to handle Russian characters PR#430 this hasn't been committed yet but the message class files were updated

Contributors to this release

We thank the following people for their contributions to this release.

John K Harvey
Andrea Catalucci
Laurenz Albe
Vladimir Sitnikov

Version 9.4-1205 (2015-11-03)

Chapman Flack (2):

- fix: recover cs, de, fr, it translations PR#409 1793454
- fix: redo spelling changes in unclobbered .po PR#409 5bfb26d

Dave Cramer (10):

- use UTF-8 instead of US-ASCII for initial encoding PR#398 8dae0ae
- fix: coerce array type names to lower case PR#402 f1a5cc4
- fix:getColumns should return columns for anything that looks like a table PR#405 f9f55d6
- revert: undo changes to getTypeInfo which removed core types PR#406 779ce18
- fix: tests PR#406 7736b8d

- reformat: tests PR#406 4acf7ae
- fix:Do not set the loglevel when instantiating a connection it is set when the Driver starts PR#407 84b28eb
- fix: japanese translation 8aa478b
- updated translation class files 2fa59d1
- build:incremented for 1205 PR#410 fa0d7d5

Michael Paquier (2):

- Remove non-ASCII character in code PR#400 9f08e46
- Add missing entry in lib/.gitignore PR#401 358aef1

Rikard Pavelic (1):

- fix: Improve type detection with casing issues. PR#404 0de91a5

Vladimir Sitnikov (9):

- fix: avoid memory leak when redeploying pgjdbc PR#394 d7cab7b
- test: add {?= call mysum(?, ?)} kind of test for CallableStatement PR#410 45c8368
- perf: add BigDecimal test performance test to ProcessResultSet PR#411 6dc4183
- perf: optimize ResultSet.getObject PR#411 567e268
- perf: optimize getBigDecimal PR#411 33904ef
- test: add statement.getQueryTimeoutMs, so millisecond-scale timeouts can be tested PR#413 892d7af
- fix: statement.cancel and statement.setQueryTimeout should be thread-safe PR#412 bc3d848
- chore: print markdown links in release notes script PR#416 a5b8ea7
- chore: add .gitattributes to ensure automatic end-of-line storage of text files is used PR#417 254b739

Contributors to this release

We thank the following people for their contributions to this release.

Chapman Flack
Michael Paquier
Rikard Pavelic
Vladimir Sitnikov

Version 9.4-1204 (2015-10-09)

Alexey Mozhenin

- fix: Make sure copy manager completesPR #378 (7c420ed)
- test: update for CopyLargeFileTest test to fit into 10 minutes limit PR #378 (39cd851)

Dave Cramer

- fix: implemented getFunctions in jdbc4 (03483e1)
- fix: error in getFunctionColumns PR #376 (5fafb86)
- fix: filter DatabaseMetaData.getColumns by tables PR #386 (0c95126)
- fix: abort connections after IOException instead of close PR #392 (3e68a70)

Vladimir Sitnikov

- pref: improve executeBatch by avoiding statement-by-statement execution PR #380 (92a9f30)
- perf: improve setTimestamp, setTime, setDate performance PR #379 (5a03a6e)
- chore: update jmh, drop irrelevant benchmark Parser#unmarkDoubleQuestion PR #384 (0d1237c)
- chore: add more benchmarks, add missing FlightRecorderProfiler PR #385 (613a641)
- feat: ignore empty sub-queries in composite queries PR #386 (3fb8046)
- fix: binary processing of Date/Time/Timestamps PR #387 (5ec0ac3)
- test: add tests for parsing of empty queries separated by semicolons PR #388 (d9310ce)
- fix: ConcurrentModificationException when calling PreparedStatement.close from a concurrent thread PR #392 (40bcc01)

Contributors to this release

We thank the following people for their contributions to this release.

Alexey Mozhenin
Vladimir Sitnikov

Version 9.4-1203 (2015-09-17)

Author: Dave Cramer

- fix: Implemented getFunctions
- fix: changed getProcedureColumns to getFunctionColumns
- fix: CopyManager fails to copy a file, reading just part of the data #366

Author: Lonny Jacobson

- add: Added PGTime/PGTimestamp

Author: Patric Bechtel

- fix: setObject(int parameterIndex, Object x, int targetSqlType) as it will set scale of BigDecimal 'x' to 0 as default, resulting in rounded whole values (!). PR #353 (24312c6)
- fix: round to correct amount test: add test for BigDecimal rounding behaviour in setObject(index,Object,targetSqlType) and setObject(index,Object,targetSqlType,scale) PR #353 (ff14f62)

Contributors to this release

We thank the following people for their contributions to this release.

Lonny Jacobson
Patric Bechtel
Alexey Mozhenin

Version 9.4-1202 (2015-08-27)

Author: Alexis Meneses

- ResultSet positioning methods in some particular cases PR #296 (282536b)

Author: Craig Ringer

- Disable binary xfer on batches returning generated keys PR #273 (763ae84)
- Add a new test case demonstrating a bug in returning support PR #273 (4d2b046)

- Always Describe a query in a batch that returns generated keys PR #273 (a6bd36f)

Author: Dave Cramer

- chore: fix build.xml to allow releasing to maven PR #262 (34f9361)
- fix: BlobInputStream ignores constructor parameters #263 PR #273 (c1c6edc)
- don't reset forceBinary transfer if we setPreparedThreshold (937a11c)
- Revert "perf: Remove expensive finalize methods from Statement and Connection" PR #293 (a0d3997)
- updated copyright PR #312 (263375c)
- Revert "Issue 250 – Adding setURL/getURL to BaseDataSource.java" PR #312 (a1ac380)
- fixed mailing list href PR #326 (c3e86a6)
- increment driver version PR #346 (b8ee75d)

Author: David R. Bild

- feat: add equality support to PSQLState PR #277 (7698cd9)

Author: David Schlosnagle

- Improve version checking PR #355 (f7a84db)

Author: Eugene Koontz

- Add support within "private Object buildArray (PgArrayList input, int index, int count)" for array elements whose type is jsonb PR #349 (d313138)

Author: Jeremy Whiting

- Added setter method for logging level. The method exactly matches property name in documentation. PR #282 (d9595d1)
- Added getter method. PR #282 (65759f0)
- Adding XML catalog to help unit tests not remote entity resolution. PR #284 (cb87067)
- Added support to locally resolve dtd or entity files. PR #284 (017970d)
- Disable verbose logging of the catalog resolver. PR #284 (fcc34f5)

Author: Kris Jurka

- Improve error message for failure to update multicolumn primary key RSs. PR #284 (05ff811)
- Remove all JDBC3 code as JDK 1.4/1.5 are no longer supported. PR #284 (f9a956b)
- Add preliminary support for JDBC4.2. PR #284 (bd05fd2)

Author: Lonny Jacobson

- Added setURL/getURL methods. (fcc8f75)
- Added a unit test for setURL PR #309 (5fa405b)

Author: Markus KARG

- perf: use shared PGBoolean instances PR #321 (159fed6)
- docs: parameter "database" is optional PR #332 (9a9d03f)
- refactor: binary transfer for setObject(int, Object, int) PR #351 (3ff2129)

Author: Michael Paquier

- Update entries in lib/.gitignore PR #262 (8cd15a9)

Author: Phillip Ross

- Fix for issue https://github.com/pgjdbc/pgjdbc/issues/342 - Modifications to PGline to store coefficients and constant value for linear equation representations used by postgresql for native line datatype. PR #343 (0565416)
- Fix for issue https://github.com/pgjdbc/pgjdbc/issues/342 - Removed extra copyright comments. PR #343 (5f21a18)
- Fix for issue https://github.com/pgjdbc/pgjdbc/issues/342 - Handle vertical lines. PR #343 (3918b24)
- Fix for issue https://github.com/pgjdbc/pgjdbc/issues/342 - Added test method for testing PGline PR #343 (1a50585)
- Fix for issue https://github.com/pgjdbc/pgjdbc/issues/342 - Modifications to PGline test method to only attempt database access if the postgresql version supports it (v9.4+). PR #343 (15eedb5)

Author: Rikard Pavelic

- feat: Improved composite/array type support and type naming changes. PR #333 (cddcd18)

Author: Robert J. Macomber

- Deadlock after IO exception during copy cancel PR #363 (d535c13)

Author: Sehrope Sarkuni

- style: clean up newline whitespace PR #273 (1b77b4c)
- style: clean up whitespace in .travis.yml PR #274 (3ee5bbf)
- fix: correct incorrect PG database version in .travis.yml matrix PR #274 (74b88c6)
- style: reorder jdk versions in .travis.yml PR #274 (21289e7)
- feat: add PostgreSQL 9.4 to .travis.yml matrix PR #274 (9e94f35)
- feat: add escapeLiteral(…) and escapeIdentifier(…) to PGConnection PR #275 (096241f)

Author: Stephen Nelson

- Replace for loops with Java 5-style for loops. Replace String.indexOf with String.contains. Replace StringBuffer with StringBuilder. Remove boxing/unboxing of primitives. PR #245 (206a542)

Author: Vladimir Gordiychuk

- feat: Customize default fetchSize for statements PR #287 (093a4bc)
- feat: Customize default fetchSize for statements PR #287 (519bfe1)
- perf: Read test only property "org.postgresql.forceBinary" spend many time when creating statements PR #291 (e185a48)

Author: Vladimir Sitnikov

- perf: Remove expensive finalize method from Statement Finalize method on Statement is moved to a separate class that is lazily created if user sets "autoCloseUnclosedConnections"="true". This dramatically improves performance of statement instantiation and reduces garbage collection overhead on several wildly used JMVs. PR #290 (eb83210)
- docs: fix misleading statement on "can't use jdk5 features" in README.md PR #298 (5b91aed)
- feat: implement micro-benchmark module for performance testing PR #297 (48b79a3)
- feat: add benchmark for Parser.unmarkDoubleQuestion PR #297 (e5a7e4e)
- feat: improve sql parsing performance PR #301 (fdd9249)

- perf: Remove AbstractJdbc2Statement.finalize() PR #299 (b3a2f80)
- test: add test for prepare-fetch-execute performance PR #303 (d23306c)
- perf: improve performance of preparing statements PR #303 (7c0655b)
- test: add test for utf8-encoder performance PR #307 (6345ab1)
- perf: improve performance of UTF-8 encoding PR #307 (f2c175f)
- perf: skip instantiation of testReturn and functionReturnType for non-callable statements PR #323 (8eacd06)
- perf: parse SQL to a single string, not a array of fragments PR #319 (4797114)
- perf: cache parsed statement across .prepareStatement calls PR #319 (5642abc)
- refactor: cleanup constructors of JDBC4 and JDBC42 connections/statements PR #318 (a4789c0)
- refactor: use Dequeue<…> instead of raw ArrayList in v3.QueryExecutorImpl PR #314 (787d775)
- perf: SimpleParameterList.flags int[] -> byte[] PR #325 (f5bceda)
- perf: cut new byte[1] from QueryExecutorImpl.receiveCommandStatus PR #326 (0ae1968)
- perf: avoid useBinary(field) check for each sendBind PR #324 (45269b8)
- refactor: cleanup Parser and NativeQuery after #311 PR #346 (a1029df)
- refactor: cleanup Parser and CallableQueryKey after #319 PR #346 (5ec7dea)
- perf: skip caching of very large queries to prevent statement cache pollution PR #346 (126b60c)
- use current_schema() for Connection#getSchema PR #356 (ffda429)
- chore: simple script to compose release notes PR #357 (341ff8e)
- chore: teach release_notes.sh to identify PR ids out of merge commits PR #358 (f3214b1)

Contributors to this release

We thank the following people for their contributions to this release.

Alexis Meneses
Craig Ringer
Jeremy Whiting
Sehrope Sarkuni
Mikko Tiihonen
Dave Cramer
Stephen Nelson
Vladimir Gordiychuk
Vladimir Sitnikov
David Schlosnagle
Eugene Koontz

Markus KARG
Phillip Ross
Rikard Pavelic
Robert J. Macomber
Michael Paquier
Lonny Jacobson
Kris Jurka
David R. Bild

Other committers

Version 9.4-1201 (2015-02-25)

Author: Alexis Meneses

- Do not pull a concrete implementation of lf4j in maven pom. Fixes #251 PR #252 (6c2848c)
- Declare all maven dependencies as optional in the pom PR #252 (1d02876)
- Automatically load maven-ant-tasks to remove the need of "-lib lib" when building the project using ant PR #253 (7e80fca)
- ant clean target delete maven dependencies added into "lib" directory PR #253 (cb7964e)
- travis uses build process simplification PR #253 (e09032c)
- regression of `ssl` parameter processing when used in additional Properties PR #260 (33d55af)

Author: Dave Cramer davec@postgresintl.com

- Make junit optional in the pom PR #254 (9530047)
- Added commit message readme PR #255 (c6312eb)
- setUrl method of BaseDataSource. fixes issue #258 PR #358 (52a8d91)

Mikko Tiihonen (1):

- setUrl method of BaseDataSource. Add test to make sure it does not break again PR #257 (477c7c8)

Contributors to this release

We thank the following people for their contributions to this release.

Alexis Meneses
Sehrope Sarkuni

Mikko Tiihonen
Dave Cramer

Other committers

Version 9.4-1200 (2015-01-02)

Author: Alexis Meneses

- Support for setBinaryStream with unknown length PR #220

- Improved support for BLOBS PR #219
- Added the support for very large objects (4TB) when backend is at least 9.3 using lo_xxx64 functions (see release 9.3 notes).
- Added support for various JDBC4 methods related to Blobs:

- setBlob with input stream in PreparedStatement

- getBinaryStream with position and offset in Blob (also helps a lot handling very large objects)
- Fix for setStringType in DataSource PR #221
- Set search path in startup packet PR #216
- Connection.isValid() should have no impact on transaction state PR #218 fixes issue #214
- Replace StringBuffer with StringBuilder for performance PR #243
- Make Pgjdbc an OSGi bundle PR #241
- Fixing the Travis-CI integration

Author: Sehrope Sarkuni

- Fix Timer thread classloader leak PR #197
- Escape search_path in Connection.setSchema PR #207
- Add SSL factory SingleCertValidatingFactory PR #88
- Speed up connection creation on 9.0+ PR #144

Author: Mikko Tiihonen

- Enhance connection fail-over with master/slave restriction and loadbalancing PR #209 Based on work by chenhj@cn.fujitsu.com

Author: Minglei Tu

- add ?-contained operator support PR#227

Author: Martin Simka

- GSS: fall back to old authentication when Subject doesn't contain instance of GssCredentials PR#228
- gssapi: Re-use existing Subject and GssCredentials PR#201

Author: bryonv

- Honor stringtype=unspecified when also saving null values

- Currently, saving a string with setString applies oid.UNSPECIFIED, but saving null with setNull(index, java.sql.Types.VARCHAR) saves with oid.VARCHAR.
- For consistency, if the user requested that we treat string types as unspecified, we should do so in all cases.

Author: Craig Ringer

- Add native SSPI authentication support on Windows using JNA via Waffle PR #212
- Add limited support for returning generated columns in batches PR #204. Fixes issues #194 and #195

Contributors to this release

We thank the following people for their contributions to this release.

Alexis Meneses
Sehrope Sarkuni
Minglei Tu
Martin Simka
Mikko Tiihonen
bryonv
Craig Ringer craig@2ndquadrant.com
Dave Cramer davec@postgresintl.com

Other committers

Version 9.3-1103 (2015-01-02)

Author: Ancoron ancoron.luciferis@gmail.com

```
Backport PGXAConnection.equals() fix from master
```

Author: Ancoron ancoron.luciferis@gmail.com

```
Fix connection URL generation for 'stringtype' parameter in BaseDataSource
```

Author: Heikki Linnakangas heikki.linnakangas@iki.fi

```
Fix equals-method of the wrapper returned by PGXAConnection.getConnection()
Patch by Florent Guillaume
```

Author: Alexis Meneses alexismeneses@users.noreply.github.com

```
Fixes on get/set/current-schema (9.3 branch)
```

Author: Alexis Meneses alexismeneses@users.noreply.github.com

```
Add a TestSuite for JDBC 4.1
```

Author: Alexis Meneses alexismeneses@users.noreply.github.com

```
Unescape/unquote result of getSchema
```

Author: Alexis Meneses alexismeneses@users.noreply.github.com

```
Setting the search_path from currentSchema property is done in startup packet (v3 protocol or
```

Author: Alexis Meneses alexismeneses@users.noreply.github.com

```
Add tests for schema name containing special characters
```

Author: Alexis Meneses alexismeneses@users.noreply.github.com

```
Escape schema name when setting search_path
```

Author: Damiano Albani damiano.albani@gmail.com

```
Add support for "currentSchema" connection property.
```

Author: Dave Cramer davec@postgresintl.com

```
Fixed timezone test as per Tom Lane's suggestion. …
Now using Europe/Helsinki as the exemplar.
```

Author: Dave Cramer davec@postgresintl.com

```
Clob will now use the connection encoding
pull #121 from brekka/clob_encoding
```

Author: Dave Cramer davec@postgresintl.com

```
backpatched Statement.isClosed() implementation of PGPoolingDataSource with some performanc
```

Author: nicolas-f github@nettrader.fr

```
Implement hashcode in PGObject 9.3
Handle null value
```

Author: Dave Cramer davec@postgresintl.com Date: Mon Aug 18 12:30:48 2014 +0000

```
NPE fix in org.postgresql.Driver.getPropertyInfo #176 \
from Sergey Ignatov
```

Version 9.3-1102 (2014-07-10)

Author:epgrubmair bug #161

```
fix copyOut close hanging bug #161 from epgrubmair
```

Author:romank0

```
backpatch exception during close of fully read stream from romank0
```

Author:Christophe Canovas

```
Added caching for ResultSetMetaData  complete commit
```

Author:Elizabeth Chatman

```
NullPointerException in AbstractJdbc2DatabaseMetaData.getUDTs
```

```
setNull, setString, setObject may fail if a specified type cannot be transferred in a binary
```

```
backpatch fix for changing datestyle before copy
```

Author:TomonariKatsumata

```
binary transfer fixes new feature -1 for forceBinaryTransfer
```

Author:Sergey Chernov

```
connectTimeout property support backpatch
```

Author:Naoya Anzai

```
fix prepared statement ERROR due to EMPTY_QUERY defined as static.
```

Version 9.3-1101 (2014-02-14)

Author:Jeremy Whiting jwhiting@redhat.com

```
Added feature to disable column name sanitiser with a new property.
disableColumnSanitiser= boolean
remove toLower calls for performance
```

Author:Craig Ringer craig@2ndquadrant.com

```
Add a MainClass that tells the user they can't just run the JDBC driver
After one too many reports of
   "Failed to load Main-Class manifest attribute from postgresql-xxx.jar"
I'm submitting a dummy main-class that tells the user what they should
do instead.

The message looks like:

------------
PostgreSQL x.y JDBC4.1 (build bbbb)
Found in: jar:file:/path/to/postgresql-x.y-bbbb.jdbc41.jar!/org/postgresql/Driver.class

The PgJDBC driver is not an executable Java program.
```

```
You must install it according to the JDBC driver installation instructions for your applicati
    jdbc:postgresql://
or using an application specific method.

See the PgJDBC documentation: http://jdbc.postgresql.org/documentation/head/index

This command has had no effect.
------------


fixed bug PreparedStatement.getMetaData failed if result set was closed
reported by Emmanuel Guiton
```

Author: cchantep chantepie@altern.org Date: Thu Dec 12 15:54:55 2013 +0100

```
Base table more useful than "" as basic table name


fixed driver fails to find foreign tables fix from plalg@hotmail.com
```

Author: Heikki Linnakangas heikki.linnakangas@iki.fi

```
Fix various setQueryTimeout bugs.

1. If you call setQueryTimeout(5), wait 10 seconds, and call execute(), the
cancel timer has already expired, and the statement will be allowed to run
forever. Likewise, if you call setQueryTimeout(5), wait 4 seconds, and call
execute(), the statement will be canceled after only 1 second.

2. If you call setQueryTimeout on a PreparedStatement, and execute the same
statement several times, the timeout only takes affect on the first
statement.

3. If you call setQueryTimeout on one Statement, but don't execute it, the
timer will still fire, possible on an unrelated victim Statement.

The root cause of all of these bugs was that the timer was started at the
setQueryTimeout() call, not on the execute() call.

Also, remove the finally-block from the cancellation task's run-method,
because that might erroneously cancel the timer for the next query, if a
new query is started using the same statement fast enough.
```

Author: Heikki Linnakangas heikki.linnakangas@iki.fi

Use StringBuffer to construct a string.

This piece of code isn't performance-critical at all, but silences a Coverity complaint.

Author: Heikki Linnakangas heikki.linnakangas@iki.fi

Avoid integer overflow.

The function returns long, but does the calculation first in int. If someone sets the timeout to 600 hours in the URL, it will overflow, even though the return value of the function is long and hence could return a larger value.

To silence a Coverity complaint.

Author: Heikki Linnakangas heikki.linnakangas@iki.fi

Plug some Statement leaks in metadata queries.

These are fairly harmless, nobody calls these metadata methods frequently enough for the leaks to matter, and a regular Statement doesn't hold onto any server resources anyway. But let's appease Coverity.

Author: Heikki Linnakangas heikki.linnakangas@iki.fi

Make sure file is closed on exception.

The system will eventually close the file anyway, and this read is highly unlikely to throw an IOException in practice.

Also, use RandomAccessFile.readFully(byte[]) to slurp the file into byte array, rather than FileInputStream.read(byte[]). The latter would need to be called in a loop to protect from short reads.

Both issues were complained of by Coverity.

Author: Stephen Nelson stephen@eccostudio.com

Generate a non-Maven JAR filename using build-time version and JDBC API level.

Author: Stephen Nelson stephen@eccostudio.com

Build script changes to allow packaging and deployment to Maven central using maven-ant-tasks

Updated build.properties to contain the sonatype urls. Updated build.xml so that gpg signing

Author: Craig Ringer craig@2ndquadrant.com

```
NonValidatingFactory should be included in both JDBC3 and JDBC4
```

Author: Michael McCaskill michael@team.shoeboxed.com

```
Use proper System property

Using 'path.separator' results in malformed paths such as:
/default/dir:./postgresql/root.crt
This corrects the problem.
```

Author: Dave Cramer davecramer@gmail.com

```
reset interrupted bit and throw unchecked exception if we get interrupted while trying to con
```

Author: Dave Cramer davecramer@gmail.com

```
add functions to allow LargeObjectMaager to commit on close from Marc Cousin
```

Author: tminglei tmlneu@gmail.com

```
add uuid array support (rename ArrayElementBuilder to ArrayAssistant)
```

Author: Nick White nwhite@palantir.com

```
allow the first query to be binary
```

Author: Dave Cramer davec@postgresintl.com

```
Merge pull request #107 from cchantep/rs-basic-tblname

Basic table name for resultset metadata
```

Author: Dave Cramer davecramer@gmail.com

```
fixed bug PreparedStatement.getMetaData failed if result set was closed
reported by Emmanuel Guiton #bugfix#
```

Version 9.3-1100 (2013-11-01)

Author: Dave Cramer davecramer@gmail.com Date: Thu Oct 17 08:29:07 2013 -0400

```
reset interrupted bit and throw unchecked exception if we get interrupted while trying to con
```

Author: Dave Cramer davecramer@gmail.com Date: Tue Oct 15 06:51:45 2013 -0400

```
add functions to allow LargeObjectMaager to commit on close from Marc Cousin
```

Author: halset halset@ecc.no Date: Mon Sep 9 12:12:26 2013 +0200

```
fix for setBlob with large blob
```

Author: Dave Cramer davecramer@gmail.com Date: Tue Sep 10 09:00:37 2013 -0400

```
fixed DatabaseMetaDataTest as per Sylvain Cuaz
```

Author: Dave Cramer davecramer@gmail.com Date: Mon Jul 29 09:43:20 2013 -0400

```
fixed sort order for DatabaseMetaData
```

Author: Dave Cramer davec@postgresintl.com Date: Sun Jul 21 13:18:26 2013 +0000

```
backpatched canceltimer bug reported by Andri-Redko
```

Author: Dave Cramer davec@postgresintl.com Date: Wed May 22 04:03:58 2013 -0700

```
Merge pull request #60 from davecramer/REL9_2_STABLE

backpatch BaseDataSource
```

Author: Dave Cramer davec@postgresintl.com Date: Tue May 21 21:03:21 2013 -0400

```
check for null before appending to url
```

Author: Dave Cramer davecramer@gmail.com Date: Tue May 21 20:23:00 2013 -0400

```
initialize binaryTranferEnable to null
```

Author: Dave Cramer davecramer@gmail.com Date: Tue May 21 20:08:27 2013 -0400

```
fixed tcpkeepalive as per Rui Zhu
pass stringtype on to url through properties from Mike O'Toole
```

Author: Dave Cramer davecramer@gmail.com

Date: Wed Feb 20 06:58:59 2013 -0500

```
avoid NullPointerException from Derrik Hudson on User Defined Types
```

Author: Kris Jurka jurka@ejurka.com Date: Tue Mar 26 05:33:45 2013 -0700

```
Lookup correct array delimiter in Connection.createArrayOf.

The old code had hardcoded a comma, but that's not true for all
datatypes.

Identification and fix by sumo in pull request #49, testcase by me.
```

Author: Dave Cramer davecramer@gmail.com Date: Thu Nov 1 11:24:34 2012 -0400

```
fix toString to handle binary integer and float types
```

Author: Dave Cramer davecramer@gmail.com Date: Wed Oct 31 10:23:23 2012 -0400

```
Fix performance regression introduced by using InetSocketAddress.getHostName()
Patch provided by Scott Harrington, improved upon by Kris Jurka
```

Author: Dave Cramer davec@postgresintl.com Date: Thu Oct 18 07:44:06 2012 -0400

```
removed testSetObjectFromJavaArray out of jdbc2 tests
```

Author: Dave Cramer davec@postgresintl.com Date: Wed Oct 17 14:22:03 2012 -0400

```
added BR translation class file
```

Author: Dave Cramer davecramer@gmail.com Date: Thu Sep 27 09:38:25 2012 -0400

```
fixed missing isValid function
```

Author: Dave Cramer davecramer@gmail.com Date: Fri Sep 21 14:58:00 2012 -0400

```
fixed attacl for servers below 8.3
```

Author: Craig Ringer ringerc@ringerc.id.au Date: Thu Sep 20 17:51:39 2012 +0800

```
Add some info to the README with contributor info

Discuss:
- Bug reporting
- Submitting patches
- Test matrix
- GitHub
```

Author: Craig Ringer ringerc@ringerc.id.au Date: Thu Sep 20 13:47:26 2012 +0800

```
Update URLs in README to refer to the Oracle page locations

Oracle has been doing a very poor job of maintaining old SUN URLs,
and it's likely that these will break at some point, so best update
them to reflect Oracle's control of Java now.

Added a link to the JDBC tutorial in the process.
```

Author: Craig Ringer ringerc@ringerc.id.au Date: Thu Sep 20 14:02:18 2012 +0800

```
Allow testing to continue when local host name doesn't resolve

It seems to be a common and default configuration on some Linux systems
```

```
for the local hostname not to resolve to the loopback IP address. This
causes testTimeoutOccurs(org.postgresql.test.jdbc2.LoginTimeoutTest)
to fail. I'm seeing this on Fedora 17 among others.

While it's best to fix such systems, not causing an easily avoided
and spurious failure in PgJDBC's test suite is probably worthwhile.
Spit out a warning and continue.
```

Author: Dave Cramer davecramer@gmail.com Date: Fri Aug 24 14:24:31 2012 -0400

```
Fixed build to not delete pgjdbc, which breaks the website build
```

Author: Dave Cramer davec@postgresintl.com Date: Thu Sep 13 07:48:52 2012 -0400

```
updated translations
```

Author: Dave Cramer davec@postgresintl.com Date: Thu Sep 13 07:43:40 2012 -0400

```
Added explicit test case for array handling from Craig Ringer
```

Author: Dave Cramer davec@postgresintl.com Date: Thu Sep 13 07:46:55 2012 -0400

```
added docs for send/recv buffer size
```

Author: Dave Cramer davec@postgresintl.com Date: Wed Sep 12 05:11:02 2012 -0400

```
added test case for send/recv buffers sizes
```

Author: Dave Cramer davec@postgresintl.com Date: Tue Sep 11 20:55:41 2012 -0400

```
patch from Bernd Helme for send recev buffer sizes
```

Author: Dave Cramer davec@postgresintl.com Date: Tue Sep 11 20:12:13 2012 -0400

```
new translation from Euler Taveira
```

Author: Craig Ringer ringerc@ringerc.id.au Date: Thu Sep 20 16:07:43 2012 +0800

```
Fix breakage of JDBC3 builds with JDK5 by bddc05f939

Fixes:
  commit bddc05f939ac9227b682e85d1ba0a9b902da814c
  simple connection failover from Mikko Tiihonen

See:
  https://github.com/pgjdbc/pgjdbc/issues/6

These fixes only affect builds of the JDBC3 driver. The JDBC4 driver
appears fine.
```

Author: Dave Cramer davec@postgresintl.com Date: Thu Sep 13 07:43:40 2012 -0400

```
added explicit test case for array handling from Craig Ringer
```

Author: Dave Cramer davecramer@gmail.com Date: Mon Jun 4 08:56:38 2012 -0400

```
simple connection failover from Mikko Tiihonen
```

Author: Dave Cramer davecramer@gmail.com Date: Sun Jun 3 07:49:37 2012 -0400

```
implemented setBinaryStream by Johann Oskarsson
```

Author: Dave Cramer davecramer@gmail.com Date: Fri Jun 1 17:57:31 2012 -0400

```
fixed docs from Mikko Tiihonen
```

Author: Dave Cramer davecramer@gmail.com Date: Fri Jun 1 17:18:05 2012 -0400

```
fixed urls in docs  from Mikko Tiihonen
```

Author: Dave Cramer davecramer@gmail.com Date: Fri Jun 1 17:16:49 2012 -0400

```
fixed docs for loading in java 6.0 from Mikko Tiihonen
```

Author: Dave Cramer davecramer@gmail.com Date: Tue May 15 20:35:51 2012 -0400

```
rest of Add hstore patch
```

Author: Dave Cramer davecramer@gmail.com Date: Tue May 15 06:47:09 2012 -0400

```
Add support for hstore from Mikko Tiihonen
```

Author: Dave Cramer davecramer@gmail.com Date: Tue May 15 06:24:34 2012 -0400

```
change Hashtable with Map from Mikko Tiihonen
```

Author: Dave Cramer davecramer@gmail.com Date: Tue May 15 06:23:24 2012 -0400

```
change vector to list from Mikko Tiihonen
```

Author: Dave Cramer davec@postgresintl.com Date: Wed May 2 14:12:17 2012 -0400

```
setProtocolVersion argument mis-spelled from Mikko Tiihonen
```

Author: Dave Cramer davec@postgresintl.com Date: Wed May 2 14:09:28 2012 -0400

```
fix to build for java 1.8 from Mikko Tiihonen
```

Author: Dave Cramer davecramer@gmail.com Date: Fri Apr 27 09:54:47 2012 -0400

```
check for array bounds before accessing the array
```

Author: Kris Jurka jurka@ejurka.com Date: Mon Mar 12 17:57:55 2012 -0700

```
Fix setQueryTimeout test.

When setQueryTimeout was fixed to use the correct units, I
neglected to adjust the test at the same time.
```

Author: Kris Jurka jurka@ejurka.com Date: Mon Mar 12 17:43:58 2012 -0700

```
Use a Set instead of a BitSet for tracking which types support
binary transfer.

A BitSet is a compact representation if we're only considering
builtin types that will have low oids, but if any user defined
types are enabled, all bets are off.  Once the oid counter exceeds
INT_MAX, database connections were failing outright even if no
high oid types used binary transfer because we represent these
oids with negative values that a BitSet cannot handle.
```

Author: Kris Jurka jurka@ejurka.com Date: Mon Mar 12 17:33:40 2012 -0700

```
Fix ResultSetMetaData retrieval when the oid counter exceeds INT_MAX.

Since Java doesn't have unsigned ints we retrieve the values as long
and then truncate to int, so it may have a negative value.

As reported by Owen Tran.
```

Author: Kris Jurka jurka@ejurka.com Date: Mon Mar 12 17:33:27 2012 -0700

```
Fix ResultSetMetaData retrieval when the oid counter exceeds INT_MAX.

Since Java doesn't have unsigned ints we retrieve the values as long
and then truncate to int, so it may have a negative value.

As reported by Owen Tran.
```

Author: Dave Cramer davecramer@gmail.com Date: Mon Feb 13 16:43:57 2012 -0500

```
resolve ACL getTablePriveledges for later servers
```

Author: Kris Jurka jurka@ejurka.com Date: Fri Feb 10 01:13:48 2012 -0800

```
Fix bugs in setQueryTimeout.

Setting a timeout of zero seconds should disable the timeout.
The timeout should be in seconds, but was implemented as milliseconds.
```

Author: Kris Jurka jurka@ejurka.com Date: Fri Feb 10 00:34:59 2012 -0800

```
Move logic out of concrete JDBC version classes.

The concrete JDBC implementation classes should do as little work
as possible.  They exist solely to connect the right set of concrete
implemention classes, leaving all the real work to the Abstract
versions.  Some logic had started to creep out into these concrete
classes which is bad because it duplicates code in paths that aren't
likely to be tested by a developer who is working primarily with a
single JDK version.
```

Author: Kris Jurka jurka@ejurka.com Date: Thu Feb 9 23:59:41 2012 -0800

```
Cache a copy of ResultSetMetaData in the ResultSet.

This solves a major performance problem for ResultSetMetaData users
which did not cache the ResultSetMetaData object.  One of the users
is the driver's own implementation of updatable ResultSets, so this
can't be worked around solely in end user code.

In the 9.0 and earlier releases, the Field objects were used to hold
database lookup results and these were longer lived than the
ResultSetMetaData object.  Now that ResultSetMetaData is holding
these database lookups we must hold onto the object to avoid
repeating the database queries.

Reported as bug #6293, fix by Steven Schlansker.
```

Author: Kris Jurka books@ejurka.com Date: Mon Feb 6 13:09:48 2012 -0800

```
Convert .cvsignore files to .gitignore files.
```

Author: Kris Jurka books@ejurka.com Date: Mon Feb 6 13:01:36 2012 -0800

```
Remove PostgreSQL CVS keyword expansion tags.
```

commit a57743f980a9de37877aa42a29e9c57514d5550e Author: Kris Jurka books@ejurka.com Date: Mon Feb 6 13:01:28 2012 -0800

```
Remove PostgreSQL CVS keyword expansion tags.
```

Author: Dave Cramer davec@fastcrypt.com Date: Thu Jan 19 20:00:51 2012 +0000

```
added isValid implementation from Luis Flores
```

Author: Dave Cramer davec@fastcrypt.com Date: Thu Jan 19 12:06:44 2012 +0000

```
SSPI authentication support from Christian Ullrich
```

Author: Dave Cramer davec@fastcrypt.com Date: Mon Jan 16 21:00:51 2012 +0000

```
removed ssl tests for buildfarm
```

Author: Dave Cramer davec@fastcrypt.com Date: Thu Jan 5 01:12:44 2012 +0000

```
removed quotes around language specifier server no longer supports this
```

Author: Dave Cramer davec@fastcrypt.com Date: Tue Jan 3 15:27:55 2012 +0000

```
remove MakeSSL.java this is now generated
```

Author: Dave Cramer davec@fastcrypt.com Date: Mon Nov 28 11:24:19 2011 +0000

```
removed Override to compile with java 1.4 added extra docs for ssl from Mikko Tiihonen
```

Author: Dave Cramer davec@fastcrypt.com Date: Thu Nov 17 11:45:21 2011 +0000

```
docs for ssl from Andras Bodor
```

Author: Dave Cramer davec@fastcrypt.com Date: Thu Nov 17 11:27:51 2011 +0000

```
SSL implementation from Andras Bodor more closely follow libpq
```

Author: Dave Cramer davec@fastcrypt.com Date: Tue Oct 4 08:33:43 2011 +0000

```
stack overflow fix from Mike Fowler
```

Author: Dave Cramer davec@fastcrypt.com Date: Fri Sep 30 10:08:17 2011 +0000

```
small fixes to binary transfer code and unit tests from Mikko Tiihonen
```

Author: Dave Cramer davec@fastcrypt.com Date: Tue Sep 27 11:15:23 2011 +0000

```
more jdk 1.4 compatibility issues fixed from Mike Fowler
```

Author: Dave Cramer davec@fastcrypt.com Date: Mon Sep 26 15:16:05 2011 +0000

```
patch from Mike Fowler to fix broken builds
```

Author: Dave Cramer davec@fastcrypt.com Date: Mon Sep 26 12:52:31 2011 +0000

```
Mikko Tiihonen patches for binary types
```

Author: Dave Cramer davec@fastcrypt.com Date: Thu Sep 22 12:53:26 2011 +0000

```
binary protocol implementation from Mikko Tiihonen
```

Author: Dave Cramer davec@fastcrypt.com Date: Tue Sep 20 15:58:49 2011 +0000

```
forgot driver.java.in for cancel query implementation
```

Author: Dave Cramer davec@fastcrypt.com Date: Tue Sep 20 14:57:13 2011 +0000

```
implemented query timeout
```

Author: Dave Cramer davec@fastcrypt.com Date: Tue Sep 20 14:43:44 2011 +0000

```
changed name of table from testmetadata to metadatatest, test was failing due to confusion wi
```

Author: Kris Jurka books@ejurka.com Date: Sun Sep 11 01:39:32 2011 +0000

```
Open HEAD for 9.2 development.
```

Version 9.2-1004 (2013-10-31)

Author: Dave Cramer davec@postgresintl.com Date: Tue Oct 29 14:32:44 2013 +0000

```
move default port back to 5432
```

Author: Dave Cramer davec@postgresintl.com Date: Tue Oct 29 12:41:09 2013 +0000

```
resolved conflict
```

Author: Dave Cramer davec@postgresintl.com Date: Tue Oct 29 05:38:39 2013 -0700

```
Merge pull request #96 from lordnelson/maven-to-9.2-branch

Build script changes to allow packaging and deployment to Maven central ...
```

Author: Stephen Nelson stephen@eccostudio.com Date: Sat Apr 13 00:20:36 2013 +0100

```
Build script changes to allow packaging and deployment to Maven central using maven-ant-task

Updated build.properties to contain the sonatype urls. Updated build.xml so that gpg signing

Use 1.4 version of gpg plugin for signing Maven upload.
```

Author: Craig Ringer craig@2ndquadrant.com Date: Fri Oct 25 01:27:23 2013 -0700

```
Merge pull request #93 from ringerc/REL9_2_STABLE

Fix #92, missing NonValidatingFactory in JDBC3 driver
```

Author: Craig Ringer craig@2ndquadrant.com Date: Fri Oct 25 16:18:10 2013 +0800

```
Fix #92, missing NonValidatingFactory in JDBC3 driver

See https://github.com/pgjdbc/pgjdbc/issues/92

You need this patch if attempts to use a URL like
```

```
   jdbc:postgresql://ipaddress:port/dbname?ssl=true&sslfactory=org.postgresql.ssl.NonVali
```

```
fails with:
```

```
   java.lang.ClassNotFoundException: org.postgresql.ssl.NonValidatingFactory
```

```
in the stack trace.
```

Author: Dave Cramer davecramer@gmail.com Date: Thu Oct 17 08:29:07 2013 -0400

```
reset interrupted bit and throw unchecked exception if we get interrupted while trying to con
```

Author: Dave Cramer davecramer@gmail.com Date: Tue Oct 15 06:51:45 2013 -0400

```
add functions to allow LargeObjectMaager to commit on close from Marc Cousin
```

Author: Dave Cramer davec@postgresintl.com Date: Tue Sep 10 07:25:06 2013 -0700

```
Merge pull request #87 from davecramer/REL9_2_STABLE

Fixed setBlob with large blob from Tore Halset
```

Author: halset halset@ecc.no Date: Mon Sep 9 12:12:26 2013 +0200

```
fix for setBlob with large blob
```

Author: Dave Cramer davec@postgresintl.com Date: Tue Sep 10 06:20:52 2013 -0700

```
Merge pull request #84 from davecramer/REL9_2_STABLE

Fixed sort order for DatabaseMetaData from Sylvain Cuaz
```

Author: Dave Cramer davecramer@gmail.com Date: Tue Sep 10 09:00:37 2013 -0400

```
fixed DatabaseMetaDataTest as per Sylvain Cuaz
```

Author: Dave Cramer davecramer@gmail.com Date: Mon Jul 29 09:43:20 2013 -0400

```
fixed sort order for DatabaseMetaData
```

Author: Dave Cramer davec@postgresintl.com Date: Sun Jul 21 13:18:26 2013 +0000

```
backpatched canceltimer bug reported by Andri-Redko
```

Author: Dave Cramer davec@postgresintl.com Date: Sun Jun 23 06:22:09 2013 -0700

```
Merge pull request #66 from davecramer/REL9_2_STABLE

fixed compile mistake
```

Author: Dave Cramer davec@postgresintl.com Date: Sun Jun 23 09:17:27 2013 -0400

```
fixed compile mistake
```

Author: Dave Cramer davec@postgresintl.com Date: Mon May 27 15:23:14 2013 -0700

```
Merge pull request #62 from davecramer/REL9_2_STABLE

incremented version to fix pushing a 1.7 build for maven
```

Version 9.2-1003 (2013-07-08)

Author: Dave Cramer Date: Mon Jul 8 03:23:25 2013 -0700

```
Merge pull request #68 from tomdcc/setobject-varchar-stringtype

Make PreparedStatement.setObject(pos, value, Types.VARCHAR) respect stringtype=unspecified
```

Author: Tom Dunstan Date: Sun Jul 7 16:20:41 2013 +0930

```
Make PreparedStatement.getObject() for an enum type return a string rather than a PGObject
```

Author: Tom Dunstan Date: Sun Jul 7 12:53:43 2013 +0930

```
Make PreparedStatement.setObject(pos, value, Types.VARCHAR) respect stringtype=unspecified
```

Author: Dave Cramer Date: Wed Jul 3 03:42:36 2013 -0700

```
Merge pull request #67 from njwhite/hstore

the driver will always return Maps for hstore columns
```

Author: Dave Cramer Date: Wed Jul 3 03:42:11 2013 -0700

```
Merge pull request #35 from fionatay/translations

Correct spelling in error messages and translations
```

Author: Nick White Date: Fri Jun 28 21:44:53 2013 -0400

```
the driver will always return Maps for hstore columns
```

Author: Dave Cramer Date: Tue Jun 25 08:31:34 2013 -0700

```
Merge pull request #64 from polarislabs/respect_ant_srcdir

Respect ant srcdir
```

Author: Bryan Varner Date: Fri Jun 21 11:39:35 2013 -0400

```
Respect the $ property for all source file references in ant script.

This makes it possible to restructure the build (in the future?) so that source and artifact f
```

Author: Bryan Varner Date: Fri Jun 21 11:31:11 2013 -0400

```
Ignore netbeans and IDEA projects files.
```

Author: Dave Cramer Date: Mon Jun 10 09:19:51 2013 -0700

```
Merge pull request #52 from valgog/master

Consider search_path when looking up type OIDs in TypeInfoCache
```

Author: Dave Cramer Date: Wed May 22 08:05:11 2013 -0700

```
Merge pull request #61 from davecramer/master

expose URL in BaseDataSource
```

Author: Dave Cramer Date: Wed May 22 10:56:55 2013 -0400

```
expose URL property in BaseDataSource
make sure stringtype gets into url from properties
```

Author: Dave Cramer Date: Mon May 20 16:57:37 2013 -0700

```
Merge pull request #59 from davecramer/master

support for materialized views from Thomas Kellerer
```

Author: Dave Cramer Date: Mon May 20 19:56:30 2013 -0400

```
support for materialized views from Thomas Kelllerer
```

Author: Dave Cramer Date: Mon May 20 11:46:52 2013 -0700

```
Merge pull request #58 from davecramer/master

pgbouncer transaction patch
```

Author: Valentine Gogichashvili Date: Mon Apr 15 11:16:23 2013 +0200

```
Added another test case for searching objects using search_path
```

Author: Valentine Gogichashvili Date: Fri Apr 12 17:05:00 2013 +0200

```
search_path support should be working correctly even for complex cases
```

Author: Valentine Gogichashvili Date: Fri Apr 12 03:31:40 2013 +0200

```
Test is checking search_path usage dirctly on TypeInfo methods
```

Author: Valentine Gogichashvili Date: Fri Apr 12 02:35:27 2013 +0200

```
Consider search_path when resolving type names to OIDs

In case when types with the same name existed in several schemas,
TypeInfoCache did not consider the current search_path and was choosing
an OID of a type not deterministically. These change will make
the type from the current schema to be chosen. Also this change remains
backwards compatible with the previous implementation, still being anble
to find a type, that is not included into the current search_path.

Provided test fails now, as it does not TypeInfoCache
directly. So more work is to be done to make this test work.
```

Author: Kris Jurka Date: Tue Mar 26 05:33:45 2013 -0700

```
Lookup correct array delimiter in Connection.createArrayOf.

The old code had hardcoded a comma, but that's not true for all
datatypes.

Identification and fix by sumo in pull request #49, testcase by me.
```

Author: Kris Jurka Date: Tue Mar 26 05:27:06 2013 -0700

```
Remove plaintext README in favor of Markdown version.

Having two copies is just going to invite drift.
```

Author: Dave Cramer Date: Wed Mar 20 02:30:46 2013 -0700

```
Merge pull request #48 from ChenHuajun/master

Fix a simple mistake in Driver.getPropertyInfo()
```

Author: chj Date: Wed Mar 20 14:52:49 2013 +0800

```
fix a simple miss in getPropertyInfo()
```

Author: Dave Cramer Date: Wed Feb 27 04:27:55 2013 -0800

```
Merge pull request #45 from fathomdb/fix_default_password

Change default password to 'test'
```

Author: Dave Cramer Date: Wed Feb 27 04:27:26 2013 -0800

```
Merge pull request #44 from fathomdb/support_wrappers

Support for JDBC4 isWrapperFor & unwrap methods
```

Author: Justin Santa Barbara Date: Mon Feb 25 08:12:46 2013 -0800

```
Change default password to 'test'

./org/postgresql/test/README says the default password for unit tests is 'test',
but the default was actually 'password'
```

Author: Justin Santa Barbara Date: Mon Feb 25 07:57:24 2013 -0800

```
Added unit test for wrapper functions
```

Author: Justin Santa Barbara Date: Sat Feb 23 11:19:49 2013 -0800

```
Support for JDBC4 isWrapperFor & unwrap methods
```

Author: Dave Cramer Date: Wed Feb 20 09:32:12 2013 -0500

```
removed compile error with double ,
```

Author: Dave Cramer Date: Wed Feb 20 04:00:46 2013 -0800

```
Merge pull request #42 from davecramer/master

Avoid NPE on user defined types which have a value of null provided by Derrick Hudson
```

Author: Dave Cramer Date: Wed Feb 20 06:58:59 2013 -0500

```
avoid NullPointerException from Derrik Hudson on User Defined Types
```

Author: Dave Cramer Date: Wed Feb 20 03:42:10 2013 -0800

```
Merge pull request #41 from davecramer/master

Added constant for turning logging OFF
```

Author: Dave Cramer Date: Wed Feb 20 06:40:54 2013 -0500

```
added Loglevel.OFF to complete the settings
```

Author: lordnelson Date: Tue Feb 19 14:13:16 2013 +0000

```
Markdown version of the README
```

Author: Dave Cramer Date: Thu Feb 7 06:00:11 2013 -0800

```
Merge pull request #38 from davecramer/master

logging did not work properly when using a datasource, also many properties were not copied t
```

Author: Dave Cramer Date: Thu Feb 7 08:55:06 2013 -0500

```
log can not be output when using DataSource
property settings were not being copied to the datasource
these included logLevel, binaryTranfer, sslfactory, applicationName
patch provided by Chen Huajun
```

Author: Kris Jurka Date: Thu Jan 31 16:40:41 2013 -0800

```
Expose enhanced error message fields from constraint violations.

The server now provides the schema, table, column, datatype, and
constraint names for certain errors.  So expose that to users so
they don't have to go rummaging through the error text trying
to find it.

The server doesn't always provide all the fields and doesn't cover
all the error messages, but it's a good start.  In the future it
would be good to expose this information in a PGXXX class instead
of the supposedly private ServerErrorMessage.
```

Author: Dave Cramer Date: Tue Jan 29 02:36:13 2013 -0800

```
Merge pull request #37 from davecramer/master

fix loading of driver so that it checks for beginning of postgresql url before parsing anythi
```

Author: Dave Cramer Date: Mon Jan 28 16:52:56 2013 -0500

```
make sure driver doesn't parse anything if the url isn't for us, also catch other possible er
```

Author: Fiona Tay Date: Sun Jan 20 23:46:31 2013 -0800

```
Fix spelling of occurred in error message
- An error occurred while setting up the SSL connection
```

Author: Fiona Tay Date: Sun Jan 20 23:45:26 2013 -0800

```
Fix spelling of occurred in error message
- Something unusual has occurred to cause the driver to fail
```

Author: Fiona Tay Date: Sun Jan 20 23:44:02 2013 -0800

```
Fix spelling of occurred in error message
 - An I/O error occurred while sending to the backend.
```

Author: Dave Cramer Date: Fri Jan 11 11:38:17 2013 -0800

```
Merge pull request #33 from davecramer/master

fix bug where update_count not updated correctly
```

Author: Dave Cramer Date: Fri Jan 11 14:36:48 2013 -0500

```
fixed mistake where update_count not updated properly
```

Author: Dave Cramer Date: Fri Jan 11 10:25:55 2013 -0800

```
Merge pull request #32 from davecramer/master

Allow ParseException to be thrown
```

Author: Dave Cramer Date: Fri Jan 11 13:21:56 2013 -0500

```
Fixed my patch to deal with update counts over 2^32
Check to see if the update count is greater than Integer.MAX_VALUE before
setting the update_count to Statement.SUCCESS_NO_INFO
NumberFormatException will still be thrown if there is a problem
parsing it.
```

Author: Dave Cramer Date: Fri Jan 11 08:57:05 2013 -0800

```
Merge pull request #31 from davecramer/master

Bug reference 7766 reported by Zelaine Fong
```

Author: Dave Cramer Date: Fri Jan 11 11:54:20 2013 -0500

```
Bug reference 7766 reported by Zelaine Fong
if an insert or update or delete statement affects more than 2^32 rows
we now return Statement.SUCCESS_NO_INFO
```

Author: Dave Cramer Date: Fri Jan 11 08:38:06 2013 -0800

```
Merge pull request #30 from davecramer/master

Fix cancel timer bug reported by Andriy Redko
```

Author: Dave Cramer Date: Fri Jan 11 11:34:44 2013 -0500

```
fix cancelTimer bug reported by Andri Redko.
now cancel timer when connection is closed
make sure timer is cancelled if there is an exception in execute
```

Author: Dave Cramer Date: Fri Jan 11 06:11:41 2013 -0800

```
Merge pull request #29 from davecramer/master

DbKeyStoreSocketFactory was in wrong package
```

Author: Dave Cramer Date: Fri Jan 11 09:09:59 2013 -0500

```
changed package to org.postgresql.ssl
```

Author: Dave Cramer Date: Fri Jan 11 05:57:43 2013 -0800

```
Merge pull request #28 from davecramer/master

Japanese Translation spelling fixed by Tomonari Katsumata
```

Author: Dave Cramer Date: Fri Jan 11 08:56:22 2013 -0500

```
Japanese translation spelling corrected provided by Tomonari Katsumata
```

Author: Dave Cramer Date: Fri Jan 11 05:43:41 2013 -0800

```
Merge pull request #26 from stevenschlansker/read-only

Allow driver to set read-only based on a connection parameter.
```

Author: Dave Cramer Date: Fri Jan 11 05:43:24 2013 -0800

```
Merge pull request #25 from rkrzewski/backend_pid

Expose PID of the backend process serving a particular JDBC connection
```

Author: Dave Cramer Date: Fri Jan 11 05:41:51 2013 -0800

```
Merge pull request #27 from davecramer/master

DbKeyStoreFactory
```

Author: Dave Cramer Date: Fri Jan 11 08:37:06 2013 -0500

```
SSL client certificate via Keystore from a Resource file provided by Brendan Jurd

It seems that the most common way to deal with this situation is to
specify the keystore file and the password via system properties
(javax.net.ssl.keyStore et. al.), but that wasn't suitable in my case.
 I needed to be able to load the keystore from a Resource file
embedded in the compiled JAR.

The class I came up with is attached.  It builds on the WrappedFactory
provided in jdbc-postgres.  All the implementer needs to do is
override the two abstract methods to provide an InputStream of the key
store, and the password to access it.  The InputStream could be a
FileInputStream, or an InputStream returned by getResource(), or
whatever.

This class uses the same keystore for KeyManager (selecting the
key/cert to send as the client) and for TrustManager (verifying the
server's certificate against trusted CAs).  It could easily be
extended to allow for two separate keystores by adding another couple
of methods.
```

Author: Steven Schlansker Date: Sun Dec 30 11:06:43 2012 -0800

```
Allow driver to set read-only based on a connection parameter.
```

Contributors to this release

We thank the following people for their contributions to this release.

This is a list of all people who participated as committers:

Dave Cramer (davec), Kris Jurka.

Version 9.2-1002 (2012-11-14)

- Fix Statement.toString to handle binary integer and Float types Committed by davec.

Contributors to this release

We thank the following people for their contributions to this release.

This is a list of all people who participated as committers:

Dave Cramer (davec).

Version 9.2-1001 (2012-10-31)

- Fix performance regression introduced by using InetSocketAddress.getHostName Committed by davec. Thanks to Scott Harrington, Kris Jurka.

Contributors to this release

We thank the following people for their contributions to this release.

This is a list of all people who participated as committers:
Dave Cramer (davec).

This is a list of other contributors:
Scott Harrington, Kris Jurka.

Version 9.2-1000 (2012-09-27)

- Implemented query timeout Committed by davec. Thanks to davec.
- First pass implementation of binary protocol Committed by davec. Thanks to Mikko Tiihonen.
- SSPI authentication support Committed by davec. Thanks to Christian Ullrich.
- isValid implementation Committed by davec. Thanks to Louis Flores.
- Add support for hstore Committed by davec. Thanks to Mikko Tiihonen.
- Implemented JDBC4 setBinaryStream Committed by davec. Thanks to Johann Oskarsson.
- Implementation of simple connection failover Committed by davec. Thanks to Mikko Tiihonen.
- Allow changing of send recv buffer sizes Committed by davec. Thanks to Bernd Helme.
- Fixed broken jdk 1.4 builds Committed by davec. Thanks to Mike Fowler.
- Stack overflow fix Committed by davec. Thanks to Mike Fowler.
- A BitSet is a compact representation if we're only considering builtin types that will have low oids, but if any user defined types are enabled, all bets are off. Once the oid counter exceeds INT_MAX, database connections were failing outright even if no high oid types used binary transfer because we represent these oids with negative values that a BitSet cannot handle. Committed by jurka.
- Fix attacl for servers below 8.3 Committed by davec.
- SSL implementation to mimic libpq more closely Committed by davec. Thanks to Andras Bodor.
- fix to build for java 1.8 Committed by davec. Thanks to Mikko Tiihonen.
- Updated Brazilian Portuguese translation. Committed by davec. Thanks to Euler Taveira de Oliveira.
- Documentation updates Committed by ringerc.

Contributors to this release

We thank the following people for their contributions to this release.

This is a list of all people who participated as committers:

Dave Cramer (davec), Kris Jurka (jurka), Craig Ringer (ringerc).

This is a list of other contributors:

Andras Bodor, Bernd Helme, Christian Ullrich, davec, Euler Taveira de Oliveira, Johann Oskarsson, Louis Flores, Mike Fowler, Mikko Tiihonen.

Version 9.1-902 (2011-04-18)

- Fix ResultSetMetaData retrieval when the oid counter exceeds INT_MAX. Since Java doesn't have unsigned ints we retrieve the values as long and then truncate to int, so it may have a negative value. Committed by jurka. Thanks to Owen Tran .
- Cache a copy of ResultSetMetaData in the ResultSet. This solves a major performance problem for ResultSetMetaData users which did not cache the ResultSetMetaData object. One of the users is the driver's own implementation of updatable ResultSets, so this can't be worked around solely in end user code. In the 9.0 and earlier releases, the Field objects were used to hold database lookup results and these were longer lived than the ResultSetMetaData object. Now that ResultSetMetaData is holding these database lookups we must hold onto the object to avoid repeating the database queries. Committed by jurka. Thanks to Steven Schlansker.

Contributors to this release

We thank the following people for their contributions to this release.

This is a list of all people who participated as committers:
Kris Jurka (jurka).

This is a list of other contributors:
Owen Tran Steven Schlansker.

Version 9.1-901 (2011-04-18)

- Set version to 901 for release Committed by jurka. Thanks to jurka.

Contributors to this release

We thank the following people for their contributions to this release.

This is a list of all people who participated as committers:
Kris Jurka (jurka).

This is a list of other contributors:

jurka.

Version 9.1dev-900 (2011-04-18)

- Add support for setting application_name on both connection startup and later through Connection.setClientInfo. Committed by jurka.
- Fetch all metadata for the ResultSet in one query instead of making a trip for each attribute of each column. Committed by jurka.
- Bring getSchemas up to JDBC 4 compliance. Return the additional TABLE_CATALOG column that was added in JDBC 3. Additionally support the getSchemas method added in JDBC 4 which filters the returned schemas. Committed by jurka.
- Bring getProcedures/getProcedureColumns up to JDBC 4 compliance. Both methods have added a SPECIFIC_NAME column that can be used to differentiate between overloaded functions. getProcedureColumns has added some other additional columns to describe the datatype being returned. Committed by jurka. Thanks to Thor Michael Store.
- Allow the driver to support setObject with Types.DISTINCT. We report metadata indicating that values are of this type, so we'd better accept it coming back in. Committed by jurka. Thanks to Vitalii Tymchyshyn.
- Support building with the 1.7 JDK. Committed by jurka.
- Support returning generated keys from batch statement execution. Unfortunately we need to disable the actual batching that the driver does behind the scenes because now that it is returning potentially large result values we must avoid a deadlock. Committed by jurka.
- Report permission metadata for a table with no permissions correctly. Committed by jurka. Thanks to danap.
- Ensure that an XAConnection throws SQLExceptions appropriately even though it's a proxy class. Before this change it was throwing an InvocationTargetException instead of the actual cause. Committed by jurka. Thanks to Yaocl.
- Make updatable ResultSets work with SQLXML data. Committed by jurka. Thanks to Michael Musset.
- If a domain has a not null constraint, report that information in the metadata for both DatabaseMetaData.getColumns and ResultSetMetaData.isNullable. Committed by jurka. Thanks to Thomas Kellerer.
- In DatabaseMetaData.getSchemas, return the user's own temp schemas, but no others. Previously it wasn't returning the users own temp schema, but was showing all toast temp schemas. Committed by jurka. Thanks to Thomas Kellerer.
- Change ResultSetMetaData to return information on serial datatypes in getColumnTypeName to match up with the behaviour of DatabaseMetaData.getColumns. Committed by jurka.
- Fix literals that are injected into a SQL query to contain the PG specific E" marker if they are using the non-standard-conforming-strings backslash escaping. This will get rid of the warnings from escape*string*warning.

Committed by jurka.

- Clear the generated keys associated with a Statement at the next execution. If the next execution doesn't want generated keys, we don't want to leave the old keys around. Committed by jurka.
- The 9.1 server canonicalizes the client_encoding setting, so if we ask for unicode we get utf8. This confused the driver because previous versions just echo back what we asked for. Ask for the canonical name now. Committed by jurka. Thanks to Mike Fowler.
- When running tests, don't assume that we know the server's default transaction isolation level. Committed by jurka. Thanks to Kevin Grittner.
- Update default permissions to account for changes in different server versions. 8.2 removed the rule permission while 8.4 added a trunctate permission. Committed by jurka.
- Newer server versions (9.0+) allow extra*float*digits to be set to 3 instead of the old limit of 2 to get the maximum precision of floating point values out of the server. Committed by jurka.
- Update the date tests for changes in the 1.6 JVM. Older versions allowed five digit years and single digit days and months. The latest code only allows a strict yyyy-mm-dd. This changed somewhere between 1.6.0*11 and 1.6.0*21. Committed by jurka. Thanks to Mike Fowler.
- Use slightly different SQL State error codes for the different types of connection setup failures to indicate which can be retried and which cannot. Committed by jurka. Thanks to Donald Fraser and Kevin Grittner.

Contributors to this release

We thank the following people for their contributions to this release.

This is a list of all people who participated as committers:

Kris Jurka (jurka).

This is a list of other contributors:

danap, Donald Fraser and Kevin Grittner, Kevin Grittner, Michael Musset, Mike Fowler, Thomas Kellerer, Thor Michael Store, Vitalii Tymchyshyn, Yaocl.

Version 9.0-801 (2010-09-20)

- Implement returning ASC/DESC order information in getIndexInfo. Committed by jurka.
- Support PreparedStatement.setObject(int, Character). Committed by jurka. Thanks to Vitalii Tymchyshyn.

- Work around a bug in the server's implementation of the binary copy protocol. Committed by jurka. Thanks to Matthew Wakeling.
- Make ResultSetMetaData.getColumnType match the results of DatabaseMetaData.getColumns.DATA_TYPE for domain and composite types. Committed by jurka. Thanks to Thomas Kellerer.
- Fix DatabaseMetaData.getColumns for 7.2 servers. This was accidentally broken in the previous release. Committed by jurka.
- Fix a minor concurrency issue during the setup for processing escape functions. Committed by jurka. Thanks to Pierre Queinnec.
- Fix DatabaseMetaData routines that return index information for a change in the 9.0 server that no longer renames the pg_attribute entries for the index, but only adjust the table's attributes on a column rename. Committed by jurka. Thanks to Adam Rauch.
- Track the tail of the SQLWarning chain so we can quickly add a new element to it instead of having to walk the entire chain from the head. This is important for the performance of handling plpgsql functions which do a ton of RAISE NOTICES which get translated into warnings. Committed by jurka. Thanks to Altaf Malik.

Contributors to this release

We thank the following people for their contributions to this release.

This is a list of all people who participated as committers:

Kris Jurka (jurka).

This is a list of other contributors:

Adam Rauch, Altaf Malik, Matthew Wakeling, Pierre Queinnec, Thomas Kellerer, Vitalii Tymchyshyn.

Version 9.0-dev800 (2010-05-11)

- Support reading the new hex escaped bytea format. Committed by jurka.
- Add support for returning the new TRUNCATE privilege, that was added in 8.4, to the list of known table privileges. Committed by jurka. Thanks to Thomas Kellerer.
- Add the partial index constraint to the FILTER_CONDITION column returned by DatabaseMetaData.getIndexInfo. Committed by jurka. Thanks to Mark Kirkwood.
- Japanese translation of error messages. Committed by jurka. Thanks to Hiroshi Saito.

- Bulgarian translation of error messages. Committed by jurka. Thanks to Viktor Usunov.
- Add some more specific types to the return value for DatabaseMetaData.getTables. Return composite types, temporary views, and temporary sequences with TABLE_TYPE values specifically for them. Committed by jurka. Thanks to Thomas Kellerer.
- Add loglevel and protocolversion options to DataSources. Committed by jurka.
- Remove an unused Sun specific import that prevented compilation on non-Sun JDKs. Committed by jurka. Thanks to Tom Lane.
- Change the processing of Statement.executeUpdate to complain if any of the results of a multi-statement query string return a ResultSet. Previously we were only checking the first result which resulted in silent partial execution of later SELECT statements. Committed by jurka. Thanks to Joseph Shraibman.
- Check that a Connection hasn't been closed before allowing any operations on it. Committed by jurka. Thanks to Kevin Grittner.
- Don't allow rollback or commit when a Connection is in autocommit mode. Committed by jurka. Thanks to Kevin Grittner.
- Change the SQLStates reported for using a closed Connection and closed ResultSet to be more consistent. Report connection*does*not*exist (08003) for a closed Connection and object*not*in*state (55000) for a ResultSet. Committed by jurka.
- When a COPY operation is the first statement issued in a transaction, it was not sending the BEGIN necessary to start the transaction. Committed by jurka. Thanks to Maciek Sakrejda.
- The 8.4 release added some code to avoid re-describing a statement if we already had the type information available by copying the resolved type information from the query to the parameters. Its goal was just to overwrite parameters without a type (unknown), but it was actually overwriting all types which could change the query's desired behaviour. Committed by jurka. Thanks to Hiroshi Saito.
- Fix the ORDINAL*POSITION in the DatabaseMetaData.getColumns. Previously we were returning simply pg*attribute.attnum, but that doesn't work in the presence of dropped columns because later columns don't get their attnum decremented if a preceding column is dropped. Instead use the row_number window function for 8.4 and later servers to figure out the live column position. Committed by jurka.
- Always specify an XA error code when creating an XAException. Otherwise a transaction manager won't know what to do with the error and may have to assume the worst. Committed by jurka. Thanks to Heikki Linnakangas, Justin Bertram.
- LOB truncation didn't allow truncating to zero length because it was improperly using the positioning length checks which don't allow a zero length. Committed by jurka. Thanks to Simon Kissane.

- Protocol sync was lost when a batch statement parameter had an embedded null byte. Committed by jurka. Thanks to Pierre Queinnec.
- Fix a problem using the Copy API to copy data to the server from a Reader. After reading data out of the Reader and into a buffer, we were sending the entire buffer on to the server, not just the subset of it that was filled by the read operation. Committed by jurka. Thanks to Leonardo F.
- A XA transaction should not change the autocommit setting of a Connection. Ensure that we restore this property correctly after the XA transaction completes. Committed by jurka. Thanks to Heikki Linnakangas, Achilleas Mantzios.
- PoolingDataSources were not picking up all of the properties that were set for them. Notably it would not give you a SSL connection when asked. Committed by jurka. Thanks to Eric Jain.
- When setNull is called with a TIME or TIMESTAMP type we cannot pass that type information on to the backend because we really don't know whether it is with or without a time zone. For a NULL value it doesn't matter, but we can't establish a type because a later call with a non-null value using the same PreparedStatement can potentially end up using a specific type that is incorrect. Committed by jurka. Thanks to Martti Jeenicke.

Contributors to this release

We thank the following people for their contributions to this release.

This is a list of all people who participated as committers:

Kris Jurka (jurka).

This is a list of other contributors:

Eric Jain, Heikki Linnakangas, Achilleas Mantzios, Heikki Linnakangas, Justin Bertram, Hiroshi Saito, Joseph Shraibman, Kevin Grittner, Leonardo F, Maciek Sakrejda, Mark Kirkwood, Martti Jeenicke, Pierre Queinnec, Simon Kissane, Thomas Kellerer, Tom Lane, Viktor Usunov.

All Committers

This is a list of all people who have ever participated as committers on this project.

- Kris Jurka (jurka)

- Oliver Jowett (oliver)
- Dave Cramer (davec)
- Barry Lind (blind)
- Craig Ringer (ringerc)

To use the driver, the JAR archive named `postgresql.jar` if you built from source, otherwise it will likely be (named with the following convention: `postgresql-*[server version]*.*[build number]*.jdbc*[JDBC version]*.jar`, for example `postgresql-8.0-310.jdbc3.jar`) needs to be included in the class path, either by putting it in the `CLASSPATH` environment variable, or by using flags on the **java** command line.

For instance, assume we have an application that uses the JDBC driver to access a database, and that application is installed as `/usr/local/lib/myapp.jar`. The PostgreSQL™ JDBC driver installed as `/usr/local/pgsql/share/java/postgresql.jar`. To run the application, we would use:

```
export CLASSPATH=/usr/local/lib/myapp.jar:/usr/local/pgsql/share/java/postgresql.jar:.
java MyApp
```

Loading the driver from within the application is covered in Chapter 3, Initializing the Driver.

With JDBC, a database is represented by a URL (Uniform Resource Locator). With PostgreSQL, this takes one of the following forms:

- jdbc:postgresql:*database*
- jdbc:postgresql:/
- jdbc:postgresql://*host/database*
- jdbc:postgresql://*host/*
- jdbc:postgresql://*host:port/database*
- jdbc:postgresql://*host:port/*

The parameters have the following meanings:

- *host*

  The host name of the server. Defaults to `localhost`. To specify an IPv6 address your must enclose the `host` parameter with square brackets, for example:

  jdbc:postgresql://[::1]:5740/accounting

- *port*

  The port number the server is listening on. Defaults to the PostgreSQL™ standard port number (5432).

- *database*

  The database name. The default is to connect to a database with the same name as the user name.

To connect, you need to get a `Connection` instance from JDBC. To do this, you use the `DriverManager.getConnection()` method:

`Connection db = DriverManager.getConnection(url, username, password);`

Connection Parameters

In addition to the standard connection parameters the driver supports a number of additional properties which can be used to specify additional driver behaviour specific to PostgreSQL™. These properties may be specified in either the connection URL or an additional `Properties` object parameter to `DriverManager.getConnection`. The following examples illustrate the use of both methods to establish a SSL connection.

```
String url = "jdbc:postgresql://localhost/test";
Properties props = new Properties();
props.setProperty("user","fred");
props.setProperty("password","secret");
props.setProperty("ssl","true");
Connection conn = DriverManager.getConnection(url, props);
```

```
String url = "jdbc:postgresql://localhost/test?user=fred&password=secret&ssl=true";
Connection conn = DriverManager.getConnection(url);
```

- **user** = String

  The database user on whose behalf the connection is being made.

- **password** = String

  The database user's password.

- **ssl** = boolean

  Connect using SSL. The driver must have been compiled with SSL support. This property does not need a value associated with it. The mere presence of it specifies a SSL connection. However, for compatibility with future versions, the value "true" is preferred. For more information see Chapter 4, *Using SSL*.

- **sslfactory** = String

  The provided value is a class name to use as the `SSLSocketFactory` when establishing a SSL connection. For more information see the section called "Custom SSLSocketFactory".

- **sslfactoryarg** (deprecated) = String

  This value is an optional argument to the constructor of the sslfactory class provided above. For more information see the section called "Custom SSLSocketFactory".

- **sslmode** = String

  possible values include `disable`, `allow`, `prefer`, `require`, `verify-ca` and `verify-full` . `require`, `allow` and `prefer` all default to a non validating SSL factory and do not check the validity of the certificate or the host name. `verify-ca` validates the certificate, but does not verify the hostname. `verify-full` will validate that the certificate is correct and verify the host connected to has the same hostname as the certificate.

  Setting these will necessitate storing the server certificate on the client machine see "Configuring the client" for details.

- **sslcert** = String

  Provide the full path for the certificate file. Defaults to /defaultdir/postgresql.crt

  Note: defaultdir is ${user.home}/.postgresql/ in *nix systems and %appdata%/postgresql/ on windows

- **sslkey** = String

  Provide the full path for the key file. Defaults to /defaultdir/postgresql.pk8

- **sslrootcert** = String

  File name of the SSL root certificate. Defaults to defaultdir/root.crt

- **sslhostnameverifier** = String

  Class name of hostname verifier. Defaults to using `org.postgresql.ssl.PGjdbcHostnameVerifier`

- **sslpasswordcallback** = String

  Class name of the SSL password provider. Defaults to `org.postgresql.ssl.jdbc4.LibPQFactory.Co`

- **sslpassword** = String

  If provided will be used by ConsoleCallbackHandler

- **sendBufferSize** = int

  Sets SO_SNDBUF on the connection stream

- **recvBufferSize** = int

  Sets SO_RCVBUF on the connection stream

- **protocolVersion** = int

  The driver supports the V3 frontend/backend protocols. The V3 protocol was introduced in 7.4 and the driver will by default try to connect using the V3 protocol.

- **loggerLevel** = String

  Logger level of the driver. Allowed values: OFF, DEBUG or TRACE. This enable the java.util.logging.Logger Level of the driver based on the following mapping of levels: DEBUG -> FINE, TRACE -> FINEST. This property is intended for debug the driver and not for general SQL query debug.

- **loggerFile** = String

  File name output of the Logger. If set, the Logger will use a java.util.logging.FileHandler to write to a specified file. If the parameter is not set or the file can't be created the java.util.logging.ConsoleHandler will be used instead. This parameter should be use together with loggerLevel.

- **allowEncodingChanges** = boolean

  When using the V3 protocol the driver monitors changes in certain server configuration parameters that should not be touched by end users. The `client_encoding` setting is set by the driver and should not be altered. If the driver detects a change it will abort the connection. There is one legitimate exception to this behaviour though, using the `COPY` command on a file residing on the server's filesystem. The only means of specifying the encoding of this file is by altering the `client_encoding` setting. The JDBC team considers this a failing of the `COPY` command and hopes to provide an alternate means of specifying the encoding in the future, but for now there is this URL parameter. Enable this only if you need to override the client encoding when doing a copy.

- **logUnclosedConnections** = boolean

  Clients may leak `Connection` objects by failing to call its `close()` method. Eventually these objects will be garbage collected and the `finalize()` method will be called which will close the `Connection` if caller has neglected to do this himself. The usage of a finalizer is just a stopgap solution. To help developers detect and correct the source of these leaks the `logUnclosedConnections` URL parameter has been added. It captures a stacktrace at each `Connection` opening and if the `finalize()` method is reached without having been closed the stacktrace is printed to the log.

- **autosave** = String

  Specifies what the driver should do if a query fails. In `autosave=always` mode, JDBC driver sets a savepoint before each query, and rolls back to that savepoint in case of failure. In `autosave=never` mode (default), no

savepoint dance is made ever. In `autosave=conservative` mode, savepoint is set for each query, however the rollback is done only for rare cases like 'cached statement cannot change return type' or 'statement XXX is not valid' so JDBC driver rollsback and retries

The default is `never`

- **binaryTransferEnable** = String

  A comma separated list of types to enable binary transfer. Either OID numbers or names.

- **binaryTransferDisable** = String

  A comma separated list of types to disable binary transfer. Either OID numbers or names. Overrides values in the driver default set and values set with binaryTransferEnable.

- **prepareThreshold** = int

  Determine the number of `PreparedStatement` executions required before switching over to use server side prepared statements. The default is five, meaning start using server side prepared statements on the fifth execution of the same `PreparedStatement` object. More information on server side prepared statements is available in the section called "Server Prepared Statements".

- **preparedStatementCacheQueries** = int

  Determine the number of queries that are cached in each connection. The default is 256, meaning if you use more than 256 different queries in `prepareStatement()` calls, the least recently used ones will be discarded. The cache allows application to benefit from "Server Prepared Statements" (see `prepareThreshold`) even if the prepared statement is closed after each execution. The value of 0 disables the cache.

  N.B.Each connection has its own statement cache.

- **preparedStatementCacheSizeMiB** = int

  Determine the maximum size (in mebibytes) of the prepared queries cache (see `preparedStatementCacheQueries`). The default is 5, meaning if you happen to cache more than 5 MiB of queries the least recently used ones will be discarded. The main aim of this setting is to prevent `OutOfMemoryError`. The value of 0 disables the cache.

- **preferQueryMode** = String

  Specifies which mode is used to execute queries to database: simple means ('Q' execute, no parse, no bind, text mode only), extended means always use bind/execute messages, extendedForPrepared means extended for prepared statements only, extendedCacheEverything means use extended protocol and try cache every statement (including

Statement.execute(String sql)) in a query cache. extended | extendedForPrepared | extendedCacheEverything | simple

The default is extended

- **defaultRowFetchSize** = int

  Determine the number of rows fetched in `ResultSet` by one fetch with trip to the database. Limiting the number of rows are fetch with each trip to the database allow avoids unnecessary memory consumption and as a consequence `OutOfMemoryException`.

  The default is zero, meaning that in `ResultSet` will be fetch all rows at once. Negative number is not available.

- **loginTimeout** = int

  Specify how long to wait for establishment of a database connection. The timeout is specified in seconds.

- **connectTimeout** = int

  The timeout value used for socket connect operations. If connecting to the server takes longer than this value, the connection is broken. The timeout is specified in seconds and a value of zero means that it is disabled.

- **socketTimeout** = int

  The timeout value used for socket read operations. If reading from the server takes longer than this value, the connection is closed. This can be used as both a brute force global query timeout and a method of detecting network problems. The timeout is specified in seconds and a value of zero means that it is disabled.

- **cancelSignalTimeout** = int

Cancel command is sent out of band over its own connection, so cancel message can itself get stuck. This property controls "connect timeout" and "socket timeout" used for cancel commands. The timeout is specified in seconds. Default value is 10 seconds.

- **tcpKeepAlive** = boolean

  Enable or disable TCP keep-alive probe. The default is `false`.

- **unknownLength** = int

  Certain postgresql types such as `TEXT` do not have a well defined length. When returning meta-data about these types through functions like `ResultSetMetaData.getColumnDisplaySize` and `ResultSetMetaData.getPrecision` we must provide a value and various client tools have different ideas about what they would like to see. This parameter specifies the length to return for types of unknown length.

- **stringtype** = String

  Specify the type to use when binding `PreparedStatement` parameters set via `setString()`. If `stringtype` is set to `VARCHAR` (the default), such parameters will be sent to the server as varchar parameters. If `stringtype` is set to `unspecified`, parameters will be sent to the server as untyped values, and the server will attempt to infer an appropriate type. This is useful if you have an existing application that uses `setString()` to set parameters that are actually some other type, such as integers, and you are unable to change the application to use an appropriate method such as `setInt()`.

- **kerberosServerName** = String

  The Kerberos service name to use when authenticating with GSSAPI. This is equivalent to libpq's PGKRBSRVNAME environment variable and defaults to "postgres".

- **jaasApplicationName** = String

  Specifies the name of the JAAS system or application login configuration.

- **jaasLogin** = boolean

  Specifies whether to perform a JAAS login before authenticating with GSSAPI. If set to `true` (the default), the driver will attempt to obtain GSS credentials using the configured JAAS login module(s) (e.g. `Krb5LoginModule`) before authenticating. To skip the JAAS login, for example if the native GSS implementation is being used to obtain credentials, set this to `false`.

- **ApplicationName** = String

  Specifies the name of the application that is using the connection. This allows a database administrator to see what applications are connected to the server and what resources they are using through views like pg_stat_activity.

- **gsslib** = String

  Force either SSPI (Windows transparent single-sign-on) or GSSAPI (Kerberos, via JSSE) to be used when the server requests Kerberos or SSPI authentication. Permissible values are auto (default, see below), sspi (force SSPI) or gssapi (force GSSAPI-JSSE).

  If this parameter is auto, SSPI is attempted if the server requests SSPI authentication, the JDBC client is running on Windows, and the Waffle libraries required for SSPI are on the CLASSPATH. Otherwise Kerberos/GSSAPI via JSSE is used. Note that this behaviour does not exactly match that of libpq, which uses Windows' SSPI libraries for Kerberos (GSSAPI) requests by default when on Windows.

gssapi mode forces JSSE's GSSAPI to be used even if SSPI is available, matching the pre-9.4 behaviour.

On non-Windows platforms or where SSPI is unavailable, forcing sspi mode will fail with a PSQLException.

Since: 9.4

- **sspiServiceClass** = String

  Specifies the name of the Windows SSPI service class that forms the service class part of the SPN. The default, POSTGRES, is almost always correct.

  See: SSPI authentication (Pg docs) Service Principal Names (MSDN), DsMakeSpn (MSDN) Configuring SSPI (Pg wiki).

  This parameter is ignored on non-Windows platforms.

- **useSpnego** = boolean

  Use SPNEGO in SSPI authentication requests

- **sendBufferSize** = int

  Sets SO_SNDBUF on the connection stream

- **receiveBufferSize** = int

  Sets SO_RCVBUF on the connection stream

- **readOnly** = boolean

  Put the connection in read-only mode

- **disableColumnSanitiser** = boolean

  Setting this to true disables column name sanitiser. The sanitiser folds columns in the resultset to lowercase. The default is to sanitise the columns (off).

- **assumeMinServerVersion** = String

  Assume that the server is at least the given version, thus enabling to some optimization at connection time instead of trying to be version blind.

- **currentSchema** = String

  Specify the schema to be set in the search-path. This schema will be used to resolve unqualified object names used in statements over this connection.

- **targetServerType** = String

  Allows opening connections to only servers with required state, the allowed values are any, master, slave, secondary, preferSlave and preferSecondary. The master/slave distinction is currently done by observing if the server allows writes. The value preferSecondary tries to connect to secondary if any are available, otherwise allows falls back to connecting also to master.

- **hostRecheckSeconds** = int

  Controls how long in seconds the knowledge about a host state is cached in JVM wide global cache. The default value is 10 seconds.

- **loadBalanceHosts** = boolean

  In default mode (disabled) hosts are connected in the given order. If enabled hosts are chosen randomly from the set of suitable candidates.

- **socketFactory** = String

  The provided value is a class name to use as the `SocketFactory` when establishing a socket connection. This may be used to create unix sockets instead of normal sockets. The class name specified by `socketFactory` must extend `javax.net.SocketFactory` and be available to the driver's classloader. This class must have a zero argument constructor or a single argument constructor taking a String argument. This argument may optionally be supplied by `socketFactoryArg`.

- **socketFactoryArg** (deprecated) = String

  This value is an optional argument to the constructor of the socket factory class provided above.

- **reWriteBatchedInserts** = boolean

  This will change batch inserts from insert into foo (col1, col2, col3) values (1,2,3) into insert into foo (col1, col2, col3) values (1,2,3), (4,5,6) this provides 2-3x performance improvement

- **replication** = String

Connection parameter passed in the startup message. This parameter accepts two values; "true" and `database`. Passing `true` tells the backend to go into walsender mode, wherein a small set of replication commands can be issued instead of SQL statements. Only the simple query protocol can be used in walsender mode. Passing "database" as the value instructs walsender to connect to the database specified in the dbname parameter, which will allow the connection to be used for logical replication from that database.

Parameter should be use together with `assumeMinServerVersion` with parameter >= 9.4 (backend >= 9.4)

Connection Fail-over

To support simple connection fail-over it is possible to define multiple endpoints (host and port pairs) in the connection url separated by commas. The driver will try to once connect to each of them in order until the connection succeeds. If none succeed, a normal connection exception is thrown.

The syntax for the connection url is:

```
jdbc:postgresql://host1:port1,host2:port2/database
```

The simple connection fail-over is useful when running against a high availability postgres installation that has identical data on each node. For example streaming replication postgres or postgres-xc cluster.

For example an application can create two connection pools. One data source is for writes, another for reads. The write pool limits connections only to master node:

`jdbc:postgresql://node1,node2,node3/accounting?targetServerType=master`.

And read pool balances connections between slaves nodes, but allows connections also to master if no slaves are available:

```
jdbc:postgresql://node1,node2,node3/accounting?targetServerType=preferSlave&loadBalanceH
```

If a slave fails, all slaves in the list will be tried first. If the case that there are no available slaves the master will be tried. If all of the servers are marked as "can't connect" in the cache then an attempt will be made to connect to all of the hosts in the URL in order.

- Community
- Mailing List
- Contributors

Maintainers

Please do not contact the maintainers directly unless you have a specific need to contact just them. Please use the pgsql-jdbc@postgresql.org mailing list if at all possible.

- [davec] - Dave Cramer (pg.at.fastcrypt.com)
- [jurka] - Kris Jurka (jurka.at.ejurka.com)
- [oliver] - Oliver Jowett (oliver.at.opencloud.com)

Previous Maintainers

The original JDBC driver was written by Adrian Hall. Peter Mount and Barry Lind have maintained it in the past, but have since moved on to other things.

Developers

People who have contributed significant code to the project since the 7.4 release. A careful historical study has not been commissioned, but if you've done something valuable, we'd be happy to recognize you for it. Just let us know.

- Jan Andre le Roux
  - ResultSetMetaData information based on the V3 protocol
- Jaroslaw J. Pyszny
  - Improve MetaData regarding the serial datatype
- Ulrich Meis
  - Allow users to customize the SSL connection
- Xavier Poinsard
  - Standard escaped functions {fn …() }
- Oliver Siegmar
  - Support for infinity in the timestamp datatype
  - Make PGInterval able to decode and manipulate interval data
- Michael Barker
  - Blob write and position methods.
- Andras Kadinger
  - Support asynchronous notification retrieval.
- Heikki Linnakangas
  - XADataSource implementation.
- Luis Vilar Flores
  - Reduce memory usage retrieving bytea data.
- Michael Paesold
  - Correctly parse dollar quotes and comments.
  - Work with standard_conforming_strings = on.
- Mikko Tiihonen
  - Improve speed of parsing ResultSet data.
- Marek Lewczuk
  - Support multi-dimensional arrays and NULL array elements.

Translators

- cs - Czech
  - Petr Dittrich
- de - German
  - Andre Bialojahn
- es - Spanish
  - Diego A. Gil
- fr - French
  - Xavier Poinsard
- it - Italian
  - Giuseppe Sacco
- pl - Polish
  - Jaroslaw Pyszny
- pt_BR - Brazilian Portuguese
  - Euler Taveira de Oliveira
- ru - Russian
  - Serguei Mokhov
- sr - Serbian
  - Bojan Skaljac
- tr - Turkish
  - Devrim Gunduz
  - Nicolai Tufar
- zh_CN - Simplified Chinese
  - Weiping
  - Kuo ChaoYi
- zh_TW - Traditional Chinese
  - Zhenbang Wei
  - Kuo ChaoYi

To create, modify or drop a database object like a table or view you use the `execute()` method. This method is similar to the method `executeQuery()`, but it doesn't return a result. Example 5.4, "Dropping a Table in JDBC illustrates the usage.

Example 5.4. Dropping a Table in JDBC

This example will drop a table.

```
Statement st = conn.createStatement();
st.execute("DROP TABLE mytable");
st.close();
```

PostgreSQL™ includes one implementation of `ConnectionPoolDataSource` named `org.postgresql.ds.PGConnectionPoolDataSource`.

JDBC requires that a `ConnectionPoolDataSource` be configured via JavaBean properties, shown in Table 11.1, "`ConnectionPoolDataSource` Configuration Properties", so there are get and set methods for each of these properties.

Table 11.1. `ConnectionPoolDataSource` Configuration Properties

Property

Type

Description

serverName

STRING

PostgreSQL™ database server host name

databaseName

STRING

PostgreSQL™ database name

portNumber

INT

TCP port which the PostgreSQL™

database server is listening on (or 0 to use the default port)

user

STRING

User used to make database connections

password

STRING

Password used to make database connections

ssl

BOOLEAN

If `true`, use SSL encrypted connections (default `false`)

sslfactory

STRING

Custom `javax.net.ssl.SSLSocketFactory` class name (see the section called "Custom SSLSocketFactory")

defaultAutoCommit

BOOLEAN

Whether connections should have autocommit enabled or disabled when they are supplied to the caller. The default is `false`, to disable autocommit.

Many application servers use a properties-style syntax to configure these properties, so it would not be unusual to enter properties as a block of text. If the application server provides a single area to enter all the properties, they might be listed like this:

```
serverName=localhost
databaseName=test
user=testuser
password=testpassword
```

Or, if semicolons are used as separators instead of newlines, it could look like this:

```
serverName=localhost;databaseName=test;user=testuser;password=testpassword
```

PostgreSQL™ includes two implementations of `DataSource`, as shown in Table 11.2, "`DataSource` Implementations". One that does pooling and the other that does not. The pooling implementation does not actually close connections when the client calls the `close` method, but instead returns the connections to a pool of available connections for other clients to use. This avoids any overhead of repeatedly opening and closing connections, and allows a large number of clients to share a small number of database connections.

The pooling data-source implementation provided here is not the most feature-rich in the world. Among other things, connections are never closed until the pool itself is closed; there is no way to shrink the pool. As well, connections requested for users other than the default configured user are not pooled. Its error handling sometimes cannot remove a broken connection from the pool. In

general it is not recommended to use the PostgreSQL™ provided connection pool. Check your application server or check out the excellent jakarta commons DBCP project.

Table 11.2. `DataSource` Implementations

Pooling

Implementation Class

No

`org.postgresql.ds.PGSimpleDataSource</td> </tr> <tr> <td>Yes</td> <td>`org.postgresql.ds.PGPoolingDataSource

Both implementations use the same configuration scheme. JDBC requires that a `DataSource` be configured via JavaBean properties, shown in Table 11.3, "`DataSource` Configuration Properties", so there are get and set methods for each of these properties.

Table 11.3. `DataSource` Configuration Properties

Property

Type

Description

serverName

STRING

PostgreSQL™ database server host name

databaseName

STRING

PostgreSQL™ database name

portNumber

INT

TCP port which the PostgreSQL™ database server is listening on (or 0 to use the default port)

user

STRING

User used to make database connections

password

STRING

Password used to make database connections

ssl

BOOLEAN

If true, use SSL encrypted connections (default false)

sslfactory

STRING

Custom javax.net.ssl.SSLSocketFactory class name (see the section called "Custom SSLSocketFactory")

The pooling implementation requires some additional configuration properties, which are shown in Table 11.4, "Additional Pooling `DataSource` Configuration Properties".

Table 11.4. Additional Pooling `DataSource` Configuration Properties

Property

Type

Description

dataSourceName

STRING

Every pooling DataSource must have a unique name.

initialConnections

INT

The number of database connections to be created when the pool is initialized.

maxConnections

INT

The maximum number of open database connections to allow. When more connections are requested, the caller will hang until a connection is returned to the pool.

Example 11.1, "`DataSource` Code Example" shows an example of typical application code using a pooling `DataSource`.

Example 11.1. `DataSource` Code Example

Code to initialize a pooling `DataSource` might look like this:

```
PGPoolingDataSource source = new PGPoolingDataSource();
source.setDataSourceName("A Data Source");
source.setServerName("localhost");
source.setDatabaseName("test");
source.setUser("testuser");
```

```
source.setPassword("testpassword");
source.setMaxConnections(10);
```

Then code to use a connection from the pool might look like this. Note that it is critical that the connections are eventually closed. Else the pool will "leak" connections and will eventually lock all the clients out.

```
Connection conn = null;
try
{
    conn = source.getConnection();
    // use connection
}
catch (SQLException e)
{
    // log error
}
finally
{
    if (con != null)
    {
        try { conn.close(); } catch (SQLException e) {}
    }
}
```

The JDBC specification defines functions with an escape call syntax : `{fn function_name(arguments)}`. The following tables show which functions are supported by the PostgreSQL™ driver. The driver supports the nesting and the mixing of escaped functions and escaped values. The appendix C of the JDBC specification describes the functions.

Some functions in the following tables are translated but not reported as supported because they are duplicating or changing their order of the arguments. While this is harmless for literal values or columns, it will cause problems when using prepared statements. For example "`{fn right(?,?)}`" will be translated to "`substring(?  from  (length(?)+1-?))`". As you can see the translated SQL requires more parameters than before the translation but the driver will not automatically handle this.

Table 8.1. Supported escaped numeric functions

function

reported as supported

translation

comments

abs(arg1)
yes
abs(arg1)

acos(arg1)
yes
acos(arg1)

asin(arg1)
yes
asin(arg1)

atan(arg1)
yes
atan(arg1)

atan2(arg1,arg2)
yes
atan2(arg1,arg2)

ceiling(arg1)
yes
ceil(arg1)

cos(arg1)
yes
cos(arg1)

cot(arg1)
yes
cot(arg1)

degrees(arg1)
yes
degrees(arg1)

exp(arg1)
yes
exp(arg1)

floor(arg1)
yes
floor(arg1)

log(arg1)
yes
ln(arg1)

log10(arg1)
yes
log(arg1)

mod(arg1,arg2)
yes
mod(arg1,arg2)

pi(arg1)
yes
pi(arg1)

power(arg1,arg2)
yes

pow(arg1,arg2)

radians(arg1)

yes

radians(arg1)

rand()

yes

random()

rand(arg1)

yes

setseed(arg1)*0+random()

The seed is initialized with the given argument and a new randow value is returned.

round(arg1,arg2)

yes

round(arg1,arg2)

sign(arg1)

yes

sign(arg1)

sin(arg1)

yes

sin(arg1)

sqrt(arg1)

yes

sqrt(arg1)

tan(arg1)

yes

tan(arg1)

truncate(arg1,arg2)

yes

trunc(arg1,arg2)

Table 8.2. Supported escaped string functions

function

reported as supported

translation

comments

ascii(arg1)

yes

ascii(arg1)

char(arg1)

yes

chr(arg1)

concat(arg1,arg2…)

yes

(arg1||arg2…)

The JDBC specification only require the two arguments version, but supporting more arguments was so easy…

insert(arg1,arg2,arg3,arg4)

no

overlay(arg1 placing arg4 from arg2 for arg3)

This function is not reported as supported since it changes the order of the arguments which can be a problem (for prepared statements by example).

lcase(arg1)

yes

lower(arg1)

left(arg1,arg2)

yes

substring(arg1 for arg2)

length(arg1)

yes

length(trim(trailing from arg1))

locate(arg1,arg2)

no

position(arg1 in arg2)

locate(arg1,arg2,arg3)

no

(arg2*sign(position(arg1 in substring(arg2 from arg3)+position(arg1 in substring(arg2 from arg3))

Not reported as supported since the three arguments version duplicate and change the order of the arguments.

ltrim(arg1)

yes

trim(leading from arg1)

repeat(arg1,arg2)

yes

repeat(arg1,arg2)

replace(arg1,arg2,arg3)

yes

replace(arg1,arg2,arg3)

Only reported as supported by 7.3 and above servers.

right(arg1,arg2)

no

substring(arg1 from (length(arg1)+1-arg2))

Not reported as supported since arg2 is duplicated.

rtrim(arg1)

yes

trim(trailing from arg1)

space(arg1)

yes

repeat(‘’,arg1)

substring(arg1,arg2)

yes

substr(arg1,arg2)

substring(arg1,arg2,arg3)

yes

substr(arg1,arg2,arg3)

ucase(arg1)

yes

upper(arg1)

soundex(arg1)

no

soundex(arg1)

Not reported as supported since it requires the fuzzystrmatch contrib module.

difference(arg1,arg2)

no

difference(arg1,arg2)

Not reported as supported since it requires the fuzzystrmatch contrib module.

Table 8.3. Supported escaped date/time functions

function

reported as supported

translation

comments

curdate()

yes

current_date

curtime()

yes

current_time

dayname(arg1)

yes

to_char(arg1,'Day')

dayofmonth(arg1)

yes

extract(day from arg1)

dayofweek(arg1)

yes

extract(dow from arg1)+1

We must add 1 to be in the expected 1-7 range.

dayofyear(arg1)

yes

extract(doy from arg1)

hour(arg1)
yes
extract(hour from arg1)

minute(arg1)
yes
extract(minute from arg1)

month(arg1)
yes
extract(month from arg1)

monthname(arg1)
yes
to_char(arg1,‘Month’)

now()
yes
now()

quarter(arg1)
yes
extract(quarter from arg1)

second(arg1)
yes
extract(second from arg1)

week(arg1)
yes
extract(week from arg1)

year(arg1)

yes

extract(year from arg1)

timestampadd(argIntervalType,argCount,argTimeStamp)

yes

(‘(interval according to argIntervalType and argCount)’+argTimeStamp)

an argIntervalType value of SQL_TSI_FRAC_SECOND is not implemented since backend does not support it

timestampdiff(argIntervalType,argTimeStamp1,argTimeStamp2)

not

extract((interval according to argIntervalType) from argTimeStamp2-argTimeStamp1)

only an argIntervalType value of SQL_TSI_FRAC_SECOND, SQL_TSI_FRAC_MINUTE, SQL_TSI_FRAC_HOUR or SQL_TSI_FRAC_DAY is supported

Table 8.4. Supported escaped misc functions

function

reported as supported

translation

comments

database()

yes

current_database()

Only reported as supported by 7.3 and above servers.

ifnull(arg1,arg2)

yes

coalesce(arg1,arg2)

user()

yes

user

The JDBC specification defines escapes for specifying date, time and timestamp values which are supported by the driver.

> date > `{d 'yyyy-mm-dd'}` which is translated to `DATE 'yyyy-mm-dd'`
>
> time > `{t 'hh:mm:ss'}` which is translated to `TIME 'hh:mm:ss'`
>
> timestamp > `{ts 'yyyy-mm-dd hh:mm:ss.f...'}` which is translated to `TIMESTAMP 'yyyy-mm-dd hh:mm:ss.f'` > The fractional seconds (.f…) portion of the TIMESTAMP can be omitted. — title: 'Extras' draft: false —

Introduction

Here is a collection of things that may be useful to you that are not part of the core driver. For assistance or support you should contact the channels specified in each item, instead of the core driver lists/team.

Statement Caching Wrapper

The Statement Caching Wrapper wraps the core driver to enable caching of PreparedStatement objects. Not all application servers support this functionality in their connection pools and not all applications are deployed onto application servers.

Frequently Asked Questions

---

1. New versioning policy

1.1. Why the versioning change from 9.4.xxxx to 42.x.x?

We have three issues we are trying to address here.

a) We do not want to be tied to the server release schedule.

Previously the version was based on the server release to declare some kind of compatibility, from 9.4.xxxx this was no longer the case and the increments was just in the last 4 digits, this leads us to the second issue.

b) Avoid confusion as to which version to use with which server version.

The naming scheme previously has 9.4 in it which leads people to believe it is for server version 9.4 only, when in fact it support PostgreSQL 8.2 and higher. That means that some users looking for PostgreSQL 9.5 were asking what is the version to use, and some users that still use PostgreSQL 8.4 were using the JDBC driver 8.4 Build 703.

The driver is version agnostic for the most point so there is no reason to tie it to a specific server version. Unless you have unusual requirements (running old applications or JVMs), this is the driver you should be using.

c) The previous version policy don’t leave room for differentiate from bug fixes releases and feature releases.

The new version policy will allow us to use more or less Semantic Versioning, and have a more clear understanding of the versions.

1.2. Why the number 42?

42 was more or less chosen at random. But it is large enough to avoid any future conflicts with the server. Given current server project policies, server version 42 should come out in 2049, plus or minus a bit.

Some say that "The answer to the ultimate question of life, the universe and everything is 42."

1.3. What is not the 42.0.0 release?

This release is not a rewrite of the driver, is not using a new architecture, nor is using something special, it's the continuation of the same driver following a better versioning policy.

2. XA

2.1. Does the driver have XA support?

Yes, starting with the 8.1dev-403 driver. However, transaction interleaving is not supported.

2.2. What is "transaction interleaving"?

Transaction interleaving means that one database connection can be used for multiple transactions at the same time, switching between the transactions.

Transaction interleaving is mostly useless, but it's a required part of the JTA specification. Some application servers use it to allow a bit more concurrency without allocating a bigger jdbc connection pool.

Few JDBC drivers support transaction interleaving properly. Some fake it by issuing early prepare commands, risking transaction integrity, some give strange error messages, some fail in other, subtle ways. The PostgreSQL JDBC driver does it's best to detect interleaving and throws a proper error message when it can't do what's requested.

Because of the lack of driver support, all of the popular application servers provide options to work around it, or don't use it at all. Therefore, lack of transaction interleaving shouldn't affect your application or data integrity.

See the JTA specification, section 3.4.4, or search the pgsql-jdbc mailing list archives for more information.

3. Problems

3.1. executeBatch hangs without error Possible solutions

This is related to batched queries and synchronous TCP.

The thing to look at is setting the network buffer sizes to use to large values to avoid the deadlock. The default values are machine dependent which also explains it working or not on different machines (credit to Kris Jurka)

3.2. I upgraded from 7.x to 8.x. Why did my application break?

By default, 8.x versions of the driver use protocol version 3 when communicating with servers 7.4 or higher. This protocol allows for more efficient query execution and enables true server-side prepared statements, but also places some additional restrictions on queries. Problems with upgrading the driver generally fall into one of two categories:

Parameter Typing. Previous versions of the driver sent all PreparedStatement parameters to the server as untyped strings, and allowed the server to infer their types as appropriate. When running protocol version 3 however, the driver specifies the type of each parameter as it is being sent. The upshot of this is that code which was previously able to call (for example): PreparedStatement.setObject(1, "5") to set an integer parameter now breaks, because setting a String value for an integer parameter is not allowed.

Parameter Position. Previous versions of the driver emulated PreparedStatements by performing string replacements each time the query was executed. Newer drivers using protocol 3 however actually use server-side prepared statements with placeholders for the positional parameters. The upshot of this is that '?' positional parameters are now only allowed where the PostgreSQL back-end allows parameters.

In situations where it is difficult to modify the Java code and/or queries to work with the newer protocol version, it is possible to force the driver to use an older protocol version to restore the old behavior. Look in the documentation for the protocolVersion connection parameter.

PostgreSQL™ has a set of data types that can store geometric features into a table. These include single points, lines, and polygons. We support these types in Java with the org.postgresql.geometric package. Please consult the Javadoc

mentioned in Chapter 13, *Further Reading* for details of available classes and features.

Example 9.1. Using the CIRCLE datatype JDBC

```
import java.sql.*;

import org.postgresql.geometric.PGpoint;
import org.postgresql.geometric.PGcircle;

public class GeometricTest {
    public static void main(String args[]) throws Exception {
        String url = "jdbc:postgresql://localhost:5432/test";
        try (Connection conn = DriverManager.getConnection(url, "test", "")) {
            try (Statement stmt = conn.createStatement()) {
                stmt.execute("CREATE TEMP TABLE geomtest(mycirc circle)");
            }
            insertCircle(conn);
            retrieveCircle(conn);
        }
    }

    private static void insertCircle(Connection conn) throws SQLException {
        PGpoint center = new PGpoint(1, 2.5);
        double radius = 4;
        PGcircle circle = new PGcircle(center, radius);
        try (PreparedStatement ps = conn.prepareStatement("INSERT INTO geomtest(mycirc) V
            ps.setObject(1, circle);
            ps.executeUpdate();
        }
    }

    private static void retrieveCircle(Connection conn) throws SQLException {
        try (Statement stmt = conn.createStatement()) {
            try (ResultSet rs = stmt.executeQuery("SELECT mycirc, area(mycirc) FROM geom
                while (rs.next()) {
                    PGcircle circle = (PGcircle)rs.getObject(1);
                    double area = rs.getDouble(2);

                    System.out.println("Center (X, Y) = (" + circle.center.x + ", " + ci
                    System.out.println("Radius = " + circle.radius);
                    System.out.println("Area = " + area);
                }
            }
        }
    }
```

```
}
```

- Development
- GIT
- Translations
- Website
- Todo
- Private API

GIT Web Interface

A web view of the GIT repository is available here.

Retrieving the Code

Clone the PostgreSQL JDBC repository from Github:

```
git clone git://github.com/pgjdbc/pgjdbc.git
```

This will create a subdirectory “pgjdbc” containing the driver source code.

PostgreSQL JDBC Driver

27 August 2018

PostgreSQL JDBC Driver 42.2.5 Released

Notable changes

Changed

- `ssl=true` implies `sslmode=verify-full`, that is it requires valid server certificate cdeeaca4

Added

- Support for `sslmode=allow/prefer/require` cdeeaca4

Fixed

- Security: added server hostname verification for non-default SSL factories in `sslmode=verify-full` (CVE-2018-10936) cdeeaca4
- Updated documentation on SSL configuration fa032732
- Updated Japanese translations PR 1275
- IndexOutOfBounds on prepared multistatement with insert values c2885dd0

See full changelog for 42.2.5

---

14 July 2018

PostgreSQL JDBC Driver 42.2.4 Released

Notable changes

Changed

- PreparedStatement.setNull(int parameterIndex, int t, String typeName) no longer ignores the typeName argument if it is not null PR 1160

Fixed

- Fix treatment of SQL_TSI_YEAR, SQL_TSI_WEEK, SQL_TSI_MINUTE PR 1250
- Map integrity constraint violation to XA_RBINTEGRITY instead of XAER_RMFAIL PR 1175 f2d1352c

See full changelog for 42.2.4

12 July 2018

PostgreSQL JDBC Driver 42.2.3 Released

Notable changes

Known issues

- SQL_TSI_YEAR is treated as hour, SQL_TSI_WEEK is treated as hour, SQL_TSI_MINUTE is treated as second (fixed in 42.2.4)

Changed

- Reduce the severity of the error log messages when an exception is re-thrown. The error will be thrown to caller to be dealt with so no need to log at this verbosity by pgjdbc PR 1187
- Deprecate Fastpath API PR 903
- Support parenthesis in JDBC escape syntax PR 1204
- ubenchmark module moved pgjdbc/benchmarks repository due to licensing issues PR 1215
- Include section on how to submit a bug report in CONTRIBUTING.md PR 951

Fixed

- getString for PGObject-based types returned "null" string instead of null PR 1154
- Field metadata cache can be disabled via databaseMetadataCacheFields=0 PR 1052
- Properly encode special symbols in passwords in BaseDataSource PR 1201
- Adjust date, hour, minute, second when rounding nanosecond part of a timestamp PR 1212
- perf: reduce memory allocations in query cache PR 1227
- perf: reduce memory allocations in SQL parser PR 1230, PR 1233
- Encode URL parameters in BaseDataSource PR 1201
- Improve JavaDoc formatting PR 1236

See full changelog for 42.2.3

The PostgreSQL JDBC group would like to thank YourKit for graciously providing licenses to the project.

Latest Releases

42.2.5 · 27 Aug 2018 · Notes
42.2.4 · 14 Jul 2018 · Notes
42.2.3 · 12 Jul 2018 · Notes
42.2.2 · 15 Mar 2018 · Notes
42.2.1 · 25 Jan 2018 · Notes
Downloads | Snapshots

Shortcuts

- GitHub project
- Documentation
- Mailing list
- Report a bug
- FAQ

The PostgreSQL™ JDBC driver implements native support for the Java 8 Date and Time API (JSR-310) using JDBC 4.2.

Table 5.1. Supported Java 8 Date and Time classes

PostgreSQL™

Java SE 8

DATE

LocalDate

TIME [WITHOUT TIME ZONE]

LocalTime

TIMESTAMP [WITHOUT TIME ZONE]

LocalDateTime

TIMESTAMP WITH TIME ZONE

OffsetDateTime

This is closely aligned with tables B-4 and B-5 of the JDBC 4.2 specification. Note that `ZonedDateTime`, `Instant` and `OffsetTime / TIME WITH TIME ZONE` are not supported. Also note that all `OffsetDateTime` instances will have be in UTC (have offset 0). This is because the backend stores them as UTC.

Example 5.2. Reading Java 8 Date and Time values using JDBC

```
Statement st = conn.createStatement();
ResultSet rs = st.executeQuery("SELECT * FROM mytable WHERE columnfoo = 500");
while (rs.next())
{
    System.out.print("Column 1 returned ");
    LocalDate localDate = rs.getObject(1, LocalDate.class));
    System.out.println(localDate);
}
rs.close();
st.close();
```

For other data types simply pass other classes to `#getObject`. Note that the Java data types needs to match the SQL data types in table 7.1.

Example 5.3. Writing Java 8 Date and Time values using JDBC

```
LocalDate localDate = LocalDate.now();
PreparedStatement st = conn.prepareStatement("INSERT INTO mytable (columnfoo) VALUES (?)
st.setObject(1, localDate);
st.executeUpdate();
st.close();
```

All the `ConnectionPoolDataSource` and `DataSource` implementations can be stored in JNDI. In the case of the nonpooling implementations, a new instance will be created every time the object is retrieved from JNDI, with the same settings as the instance that was stored. For the pooling implementations, the same instance will be retrieved as long as it is available (e.g., not a different JVM retrieving the pool from JNDI), or a new instance with the same settings created otherwise.

In the application server environment, typically the application server's `DataSource` instance will be stored in JNDI, instead of the PostgreSQL™ `ConnectionPoolDataSource` implementation.

In an application environment, the application may store the `DataSource` in JNDI so that it doesn't have to make a reference to the `DataSource` available to all application components that may need to use it. An example of this is shown in Example 11.2, "`DataSource` JNDI Code Example".

Example 11.2. `DataSource` JNDI Code Example

Application code to initialize a pooling `DataSource` and add it to JNDI might look like this:

```
PGPoolingDataSource source = new PGPoolingDataSource();
source.setDataSourceName("A Data Source");
source.setServerName("localhost");
source.setDatabaseName("test");
source.setUser("testuser");
source.setPassword("testpassword");
source.setMaxConnections(10);
new InitialContext().rebind("DataSource", source);
```

Then code to use a connection from the pool might look like this:

```
Connection conn = null;
try
{
    DataSource source = (DataSource)new InitialContext().lookup("DataSource");
    conn = source.getConnection();
    // use connection
}
catch (SQLException e)
{
    // log error
}
catch (NamingException e)
{
    // DataSource wasn't found in JNDI
}
finally
{
    if (con != null)
    {
        try { conn.close(); } catch (SQLException e) {}
    }
}
```

Large objects are supported in the standard JDBC specification. However, that interface is limited, and the API provided by PostgreSQL™ allows for random access to the objects contents, as if it was a local file.

The org.postgresql.largeobject package provides to Java the libpq C interface's large object API. It consists of two classes, `LargeObjectManager`, which deals with creating, opening and deleting large objects, and `LargeObject` which deals with an individual object. For an example usage of this API, please see Example 7.1, "Processing Binary Data in JDBC". — title: 'License' draft: false —

BSD 2-clause “Simplified” License

The PostgreSQL JDBC Driver is distributed under the BSD-2-Clause License. The simplest explanation of the licensing terms is that you can do whatever you want with the product and source code as long as you don’t claim you wrote it or sue us. You should give it a read though, it’s only half a page.

Listen and Notify provide a simple form of signal or interprocess communication mechanism for a collection of processes accessing the same PostgreSQL™ database. For more information on notifications consult the main server documentation. This section only deals with the JDBC specific aspects of notifications.

Standard `LISTEN`, `NOTIFY`, and `UNLISTEN` commands are issued via the standard `Statement` interface. To retrieve and process retrieved notifications the `Connection` must be cast to the PostgreSQL™ specific extension interface `PGConnection`. From there the `getNotifications()` method can be used to retrieve any outstanding notifications.

Note

A key limitation of the JDBC driver is that it cannot receive asynchronous notifications and must poll the backend to check if any notifications were issued. A timeout can be given to the poll function, but then the execution of statements from other threads will block.

Example 9.2. Receiving Notifications

```
import java.sql.*;

public class NotificationTest
{
    public static void main(String args[]) throws Exception
    {
        Class.forName("org.postgresql.Driver");
        String url = "jdbc:postgresql://localhost:5432/test";

        // Create two distinct connections, one for the notifier
        // and another for the listener to show the communication
        // works across connections although this example would
        // work fine with just one connection.

        Connection lConn = DriverManager.getConnection(url,"test","");
        Connection nConn = DriverManager.getConnection(url,"test","");

        // Create two threads, one to issue notifications and
        // the other to receive them.

        Listener listener = new Listener(lConn);
        Notifier notifier = new Notifier(nConn);
        listener.start();
```

```
        notifier.start();
    }
}

class Listener extends Thread
{
    private Connection conn;
    private org.postgresql.PGConnection pgconn;

    Listener(Connection conn) throws SQLException
    {
        this.conn = conn;
        this.pgconn = conn.unwrap(org.postgresql.PGConnection.class);
        Statement stmt = conn.createStatement();
        stmt.execute("LISTEN mymessage");
        stmt.close();
    }

    public void run()
    {
        try
        {
            while (true)
            {
                org.postgresql.PGNotification notifications[] = pgconn.getNotifications()

                // If this thread is the only one that uses the connection, a timeout ca
                // receive notifications immediately:
                // org.postgresql.PGNotification notifications[] = pgconn.getNotificatio

                if (notifications != null)
                {
                    for (int i=0; i < notifications.length; i++)
                        System.out.println("Got notification: " + notifications[i].getNa
                }

                // wait a while before checking again for new
                // notifications

                Thread.sleep(500);
            }
        }
        catch (SQLException sqle)
        {
            sqle.printStackTrace();
        }
```

```
        catch (InterruptedException ie)
        {
            ie.printStackTrace();
        }
    }
}

class Notifier extends Thread
{
    private Connection conn;

    public Notifier(Connection conn)
    {
        this.conn = conn;
    }

    public void run()
    {
        while (true)
        {
            try
            {
                Statement stmt = conn.createStatement();
                stmt.execute("NOTIFY mymessage");
                stmt.close();
                Thread.sleep(2000);
            }
            catch (SQLException sqle)
            {
                sqle.printStackTrace();
            }
            catch (InterruptedException ie)
            {
                ie.printStackTrace();
            }
        }
    }

}
```

Applications do not need to explicitly load the org.postgresql.Driver class because the pgjdbc driver jar supports the Java Service Provider mechanism. The driver will be loaded by the JVM when the application connects to PostgreSQL™ (as long as the driver's jar file is on the classpath).

Note

Prior to Java 1.6, the driver had to be loaded by the application - either by calling

```
Class.forName("org.postgresql.Driver");
```

or by passing the driver class name as a JVM parameter.

```
java -Djdbc.drivers=org.postgresql.Driver example.ImageViewer
```

These older methods of loading the driver are still supported but they are no longer necessary.

- Community
- Mailing List
- Contributors

Before Mailing Anyone

Before posting a question or problem to the mailing list, please first look at the following resources:

- FAQ
- Documentation
- Mailing list archives, (Google interface to newer messages.)
- Consider upgrading your JDBC driver to the latest version. It should be backwards compatible and may fix your problem.

General List - pgsql-jdbc@postgresql.org

This mailing list is for all PostgreSQL Java/JDBC related discussions. This is the place for bug reports, feature requests, help with problem solving, and pretty much anything involving Java and PostgreSQL. Messages from people who are not subscribed to the list will be held for moderator approval in an effort to cut down on spam. Archives and subscription information is available on the main PostgreSQL site.

When asking for help or reporting a problem please try to include as much information as possible. Good things to note are:

- JDBC driver build number
- Server version
- Exact error message and stacktrace
- What you were doing, ideally in code form

Commit Messages - jdbc-commits@pgfoundry.org

This mailing list is for people interested in carefully monitoring the development process. The mailing list was active until the code base was transferred to GitHub in late 2012. Every commit to this earlier CVS repository sent out an email with the log message and links to diffs. So the archive of this list, pgfoundry site, holds the history of activity with the driver prior to 2013.

Currently activity on commits is best observed directly from the git repository hosted with GitHub. The console tool, gitk, available with a git installation is an excellent GUI tool to observe commits.

You can specify outer joins using the following syntax: `{oj table (LEFT|RIGHT|FULL) OUTER JOIN (table | outer-join) ON search-condition }`

For example :

```
rs = stmt.executeQuery( "select * from {oj a left outer join b on (a.i=b.i)} ");
```

Because Java does not support using unix sockets the PostgreSQL™ server must be configured to allow TCP/IP connections. Starting with server version 8.0 TCP/IP connections are allowed from `localhost`. To allow connections to other interfaces than the loopback interface, you must modify the `postgresql.conf` file's `listen_addresses` setting.

For server versions prior to 8.0 the server does not listen on any interface by default, and you must set `tcpip_socket = true` in the `postgresql.conf` file.

Once you have made sure the server is correctly listening for TCP/IP connections the next step is to verify that users are allowed to connect to the server. Client authentication is setup in `pg_hba.conf`. Refer to the main PostgreSQL™ documentation for details. The JDBC driver supports the `trust`, `ident`, `password`, `md5`, and `crypt` authentication methods.

Overview

Postgres 9.4 (released in December 2014) introduced a new feature called logical replication. Logical replication allows changes from a database to be streamed in real-time to an external system. The difference between physical replication and logical replication is that logical replication sends data over in a logical format whereas physical replication sends data over in a binary format. Additionally logical replication can send over a single table, or database. Binary replication replicates the entire cluster in an all or nothing fashion; which is to say there is no way to get a specific table or database using binary replication

Prior to logical replication keeping an external system synchronized in real time was problematic. The application would have to update/invalidate the appropriate cache entries, reindex the data in your search engine, send it to your analytics system, and so on. This suffers from race conditions and reliability problems. For example if slightly different data gets written to two different datastores (perhaps due to a bug or a race condition),the contents of the datastores will gradually drift apart — they will become more and more inconsistent over time. Recovering from such gradual data corruption is difficult.

Logical decoding takes the database's write-ahead log (WAL), and gives us access to row-level change events: every time a row in a table is inserted, updated or deleted, that's an event. Those events are grouped by transaction, and appear in the order in which they were committed to the database. Aborted/rolled-back transactions do not appear in the stream. Thus, if you apply the change events in the same order, you end up with an exact, transactionally consistent copy of the database. It's looks like the Event Sourcing pattern that you previously implemented in your application, but now it's available out of the box from the PostgreSQL database.

For access to real-time changes PostgreSQL provides the streaming replication protocol. Replication protocol can be physical or logical. Physical replication protocol is used for Master/Secondary replication. Logical replication protocol can be used to stream changes to an external system.

Since the JDBC API does not include replication `PGConnection` implements the PostgreSQL API

Configure database

Your database should be configured to enable logical or physical replication

postgresql.conf

- Property `max_wal_senders` should be at least equal to the number of replication consumers

- Property `wal_keep_segments` should contain count wal segments that can't be removed from database.
- Property `wal_level` for logical replication should be equal to `logical`.
- Property `max_replication_slots` should be greater than zero for logical replication, because logical replication can't work without replication slot.

pg_hba.conf

Enable connect user with replication privileges to replication stream.

```
local   replication   all                     trust
host    replication   all   127.0.0.1/32      md5
host    replication   all   ::1/128           md5
```

Configuration for examples

postgresql.conf

```
max_wal_senders = 4             # max number of walsender processes
wal_keep_segments = 4           # in logfile segments, 16MB each; 0 disables
wal_level = logical             # minimal, replica, or logical
max_replication_slots = 4       # max number of replication slots
```

pg_hba.conf

```
# Allow replication connections from localhost, by a user with the
# replication privilege.
local   replication   all                     trust
host    replication   all   127.0.0.1/32      md5
host    replication   all   ::1/128           md5
```

Logical replication

Logical replication uses a replication slot to reserve WAL logs on the server and also defines which decoding plugin to use to decode the WAL logs to the required format, for example you can decode changes as json, protobuf, etc. To demonstrate how to use the pgjdbc replication API we will use the `test_decoding` plugin that is include in the `postgresql-contrib` package, but you can use your own decoding plugin. There are a few on github which can be used as examples.

In order to use the replication API, the Connection has to be created in replication mode, in this mode the connection is not available to execute SQL commands, and can only be used with replication API. This is a restriction imposed by PostgreSQL.

Example 9.4. Create replication connection.

```
String url = "jdbc:postgresql://localhost:5432/postgres";
Properties props = new Properties();
PGProperty.USER.set(props, "postgres");
PGProperty.PASSWORD.set(props, "postgres");
PGProperty.ASSUME_MIN_SERVER_VERSION.set(props, "9.4");
PGProperty.REPLICATION.set(props, "database");
PGProperty.PREFER_QUERY_MODE.set(props, "simple");

Connection con = DriverManager.getConnection(url, props);
PGConnection replConnection = con.unwrap(PGConnection.class);
```

The entire replication API is grouped in `org.postgresql.replication.PGReplicationConnection` and is available via `org.postgresql.PGConnection#getReplicationAPI`.

Before you can start replication protocol, you need to have replication slot, which can be also created via pgjdbc API.

Example 9.5. Create replication slot via pgjdbc API

```
replConnection.getReplicationAPI()
    .createReplicationSlot()
    .logical()
    .withSlotName("demo_logical_slot")
    .withOutputPlugin("test_decoding")
    .make();
```

Once we have the replication slot, we can create a ReplicationStream.

Example 9.6. Create logical replication stream.

```
PGReplicationStream stream =
    replConnection.getReplicationAPI()
        .replicationStream()
        .logical()
        .withSlotName("demo_logical_slot")
        .withSlotOption("include-xids", false)
        .withSlotOption("skip-empty-xacts", true)
        .start();
```

The replication stream will send all changes since the creation of the replication slot or from replication slot restart LSN if the slot was already used for replication. You can also start streaming changes from a particular LSN position,in that case LSN position should be specified when you create the replication stream.

Example 9.7. Create logical replication stream from particular position.

```
LogSequenceNumber waitLSN = LogSequenceNumber.valueOf("6F/E3C53568");

PGReplicationStream stream =
    replConnection.getReplicationAPI()
        .replicationStream()
        .logical()
        .withSlotName("demo_logical_slot")
        .withSlotOption("include-xids", false)
        .withSlotOption("skip-empty-xacts", true)
        .withStartPosition(waitLSN)
        .start();
```

Via `withSlotOption` we also can specify options that will be sent to our output plugin, this allows customize decoding. For example I have my own output plugin that has a property `sensitive=true` which will include changes by sensitive columns to change event.

Example 9.8. Example output with include-xids=true

```
BEGIN 105779
table public.test_logic_table: INSERT: pk[integer]:1 name[character varying]:'previous valu
COMMIT 105779
```

Example 9.9. Example output with include-xids=false

```
BEGIN
table public.test_logic_table: INSERT: pk[integer]:1 name[character varying]:'previous valu
COMMIT
```

During replication the database and consumer periodically exchange ping messages. When the database or client do not receive ping message within the configured timeout, replication has been deemed to have stopped and an exception will be thrown and the database will free resources. In PostgreSQL the ping timeout is configured by the property `wal_sender_timeout` (default = 60 seconds). Replication stream in pgjdc can be configured to send feedback(ping) when required or by time interval. It is recommended to send feedback(ping) to the database more often than configured `wal_sender_timeout`. In production I use value equal to `wal_sender_timeout / 3`. It's avoids a potential problems with networks and changes to be streamed without disconnects by timeout. To specify the feedback interval use `withStatusInterval` method.

Example 9.10. Replication stream with configured feedback interval equal to 20 sec

```
PGReplicationStream stream =
    replConnection.getReplicationAPI()
        .replicationStream()
        .logical()
        .withSlotName("demo_logical_slot")
        .withSlotOption("include-xids", false)
        .withSlotOption("skip-empty-xacts", true)
        .withStatusInterval(20, TimeUnit.SECONDS)
        .start();
```

After create `PGReplicationStream`, it's time to start receive changes in real-time. Changes can be received from stream as blocking(`org.postgresql.replication.PGReplicationStr` or as non-blocking(`org.postgresql.replication.PGReplicationStream#readPending`). Both methods receive changes as a `java.nio.ByteBuffer` with the payload from the send output plugin. We can't receive part of message, only the full message that was sent by the output plugin. ByteBuffer contains message in format that is defined by the decoding output plugin, it can be simple String, json, or whatever the plugin determines. That why pgjdbc returns the raw ByteBuffer instead of making assumptions.

Example 9.11. Example send message from output plugin.

```
OutputPluginPrepareWrite(ctx, true);
appendStringInfo(ctx->out, "BEGIN %u", txn->xid);
OutputPluginWrite(ctx, true);
```

Example 9.12. Receive changes via replication stream.

```
while (true) {
  //non blocking receive message
  ByteBuffer msg = stream.readPending();

  if (msg == null) {
    TimeUnit.MILLISECONDS.sleep(10L);
    continue;
  }

  int offset = msg.arrayOffset();
  byte[] source = msg.array();
  int length = source.length - offset;
  System.out.println(new String(source, offset, length));
}
```

As mentioned previously, replication stream should periodically send feedback to the database to prevent disconnect via timeout. Feedback is automatically sent

when `read` or `readPending` are called if it's time to send feedback. Feedback can also be sent via `org.postgresql.replication.PGReplicationStream#forceUpdateStatus()` regardless of the timeout. Another important duty of feedback is to provide the server with the Logial Sequence Number (LSN) that has been successfully received and applied to consumer, it is necessary for monitoring and to truncate/archive WAL's that that are no longer needed. In the event that replication has been restarted, it's will start from last successfully processed LSN that was sent via feedback to database.

The API provides the following feedback mechanism to indicate the successfully applied LSN by the current consumer. LSN's before this can be truncated or archived. `org.postgresql.replication.PGReplicationStream#setFlushedLSN` and `org.postgresql.replication.PGReplicationStream#setAppliedLSN`. You always can get last receive LSN via `org.postgresql.replication.PGReplicationStream#getLastRe`

Example 9.13. Add feedback indicating a successfully process LSN

```
while (true) {
  //Receive last successfully send to queue message. LSN ordered.
  LogSequenceNumber successfullySendToQueue = getQueueFeedback();
  if (successfullySendToQueue != null) {
    stream.setAppliedLSN(successfullySendToQueue);
    stream.setFlushedLSN(successfullySendToQueue);
  }

  //non blocking receive message
  ByteBuffer msg = stream.readPending();

  if (msg == null) {
    TimeUnit.MILLISECONDS.sleep(10L);
    continue;
  }

  asyncSendToQueue(msg, stream.getLastReceiveLSN());
}
```

Example 9.14. Full example of logical replication

```
String url = "jdbc:postgresql://localhost:5432/test";
Properties props = new Properties();
PGProperty.USER.set(props, "postgres");
PGProperty.PASSWORD.set(props, "postgres");
PGProperty.ASSUME_MIN_SERVER_VERSION.set(props, "9.4");
PGProperty.REPLICATION.set(props, "database");
PGProperty.PREFER_QUERY_MODE.set(props, "simple");
```

```
Connection con = DriverManager.getConnection(url, props);
PGConnection replConnection = con.unwrap(PGConnection.class);

replConnection.getReplicationAPI()
    .createReplicationSlot()
    .logical()
    .withSlotName("demo_logical_slot")
    .withOutputPlugin("test_decoding")
    .make();

//some changes after create replication slot to demonstrate receive it
sqlConnection.setAutoCommit(true);
Statement st = sqlConnection.createStatement();
st.execute("insert into test_logic_table(name) values('first tx changes')");
st.close();

st = sqlConnection.createStatement();
st.execute("update test_logic_table set name = 'second tx change' where pk = 1");
st.close();

st = sqlConnection.createStatement();
st.execute("delete from test_logic_table where pk = 1");
st.close();

PGReplicationStream stream =
    replConnection.getReplicationAPI()
        .replicationStream()
        .logical()
        .withSlotName("demo_logical_slot")
        .withSlotOption("include-xids", false)
        .withSlotOption("skip-empty-xacts", true)
        .withStatusInterval(20, TimeUnit.SECONDS)
        .start();

while (true) {
  //non blocking receive message
  ByteBuffer msg = stream.readPending();

  if (msg == null) {
    TimeUnit.MILLISECONDS.sleep(10L);
    continue;
  }

  int offset = msg.arrayOffset();
  byte[] source = msg.array();
  int length = source.length - offset;
```

```
      System.out.println(new String(source, offset, length));

      //feedback
      stream.setAppliedLSN(stream.getLastReceiveLSN());
      stream.setFlushedLSN(stream.getLastReceiveLSN());
    }
```

Where output looks like this, where each line is a separate message.

```
BEGIN
table public.test_logic_table: INSERT: pk[integer]:1 name[character varying]:'first tx char
COMMIT
BEGIN
table public.test_logic_table: UPDATE: pk[integer]:1 name[character varying]:'second tx cha
COMMIT
BEGIN
table public.test_logic_table: DELETE: pk[integer]:1
COMMIT
```

Physical replication

API for physical replication looks like the API for logical replication. Physical replication does not require a replication slot. And ByteBuffer will contain the binary form of WAL logs. The binary WAL format is a very low level API, and can change from version to version. That is why replication between different major PostgreSQL versions is not possible. But physical replication can contain many important data, that is not available via logical replication. That is why pgjdc contains an implementation for both.

Example 9.15. Use physical replication

```
LogSequenceNumber lsn = getCurrentLSN();

Statement st = sqlConnection.createStatement();
st.execute("insert into test_physic_table(name) values('previous value')");
st.close();

PGReplicationStream stream =
    pgConnection
        .getReplicationAPI()
        .replicationStream()
        .physical()
        .withStartPosition(lsn)
        .start();

ByteBuffer read = stream.read();
```

The following must be considered when using the `ResultSet` interface:

- Before reading any values, you must call `next()`. This returns true if there is a result, but more importantly, it prepares the row for processing.
- You must close a `ResultSet` by calling `close()` once you have finished using it.
- Once you make another query with the `Statement` used to create a `ResultSet`, the currently open `ResultSet` instance is closed automatically. — layout: default_docs title: Server Prepared Statements header: Chapter 9. PostgreSQL™ Extensions to the JDBC API resource: media previoustitle: Listen / Notify previous: listennotify nexttitle: Physical and Logical replication API next: replication —

Motivation

The PostgreSQL™ server allows clients to compile sql statements that are expected to be reused to avoid the overhead of parsing and planning the statement for every execution. This functionality is available at the SQL level via PREPARE and EXECUTE beginning with server version 7.3, and at the protocol level beginning with server version 7.4, but as Java developers we really just want to use the standard `PreparedStatement` interface.

> PostgreSQL 9.2 release notes: prepared statements used to be optimized once, without any knowledge of the parameters' values. With 9.2, the planner will use specific plans regarding to the parameters sent (the query will be planned at execution), except if the query is executed several times and the planner decides that the generic plan is not too much more expensive than the specific plans.

Server side prepared statements can improve execution speed as 1. It sends just statement handle (e.g. `S_1`) instead of full SQL text 1. It enables use of binary transfer (e.g. binary int4, binary timestamps, etc); the parameters and results are much faster to parse 1. It enables the reuse server-side execution plan 1. The client can reuse result set column definition, so it does not have to receive and parse metadata on each execution

Activation

> Previous versions of the driver used PREPARE and EXECUTE to implement server-prepared statements. This is supported on all server versions beginning with 7.3, but produced application-visible changes in query results, such as missing ResultSet metadata and row update counts. The current driver uses the V3 protocol-level equivalents which avoid these changes in query results.

The driver uses server side prepared statements **by default** when `PreparedStatement` API is used. In order to get to server-side prepare, you need to execute the query 5 times (that can be configured via `prepareThreshold` connection property). An internal counter keeps track of how many times the statement has been executed and when it reaches the threshold it will start to use server side prepared statements.

It is generally a good idea to reuse the same `PreparedStatement` object for performance reasons, however the driver is able to server-prepare statements automatically across `connection.prepareStatement(...)` calls.

For instance:

```
PreparedStatement ps = con.prepareStatement("select /*test*/ ?::int4");
ps.setInt(1, 42);
ps.executeQuery().close();
ps.close();

PreparedStatement ps = con.prepareStatement("select /*test*/ ?::int4");
ps.setInt(1, 43);
ps.executeQuery().close();
ps.close();
```

is less efficient than

```
PreparedStatement ps = con.prepareStatement("select /*test*/ ?::int4");
ps.setInt(1, 42);
ps.executeQuery().close();

ps.setInt(1, 43);
ps.executeQuery().close();
```

however pgjdbc can use server side prepared statements in both cases.

Note: the `Statement` object is bound to a `Connection`, and it is not a good idea to access the same `Statement` and/or `Connection` from multiple concurrent threads (except `cancel()`, `close()`, and alike cases). It might be safer to just `close()` the statement rather than trying to cache it somehow.

Server-prepared statements consume memory both on the client and the server, so pgjdbc limits the number of server-prepared statements per connection. It can be configured via `preparedStatementCacheQueries` (default `256`, the number of queries known to pgjdbc), and `preparedStatementCacheSizeMiB` (default `5`, that is the client side cache size in megabytes per connection). Only a subset of `statement cache` is server-prepared as some of the statements might fail to reach `prepareThreshold`.

Deactivation

There might be cases when you would want to disable use of server-prepared statements. For instance, if you route connections through a balancer that is incompatible with server-prepared statements, you have little choice.

You can disable usage of server side prepared statements by setting `prepareThreshold=0`

Corner cases

DDL V3 protocol avoids sending column metadata on each execution, and BIND message specifies output column format. That creates a problem for cases like

```
SELECT * FROM mytable;
ALTER mytable ADD column ...;
SELECT * FROM mytable;
```

That results in `cached plan must not change result type` error, and it causes the transaction to fail.

The recommendation is: 1. Use explicit column names in the SELECT list 1. Avoid column type alters

DEALLOCATE ALL, DISCARD ALL There are explicit commands to deallocate all server side prepared statements. It would result in the following server-side error message: `prepared statement name is invalid`. Of course it could defeat pgjdbc, however there are cases when you need to discard statements (e.g. after lots of DDLs)

The recommendation is: 1. Use simple `DEALLOCATE ALL` and/or `DISCARD ALL` commands, avoid nesting the commands into pl/pgsql or alike. The driver does understand top-level DEALLOCATE/DISCARD commands, and it invalidates client-side cache as well 1. Reconnect. The cache is per connection, so it would get invalidated if you reconnect

set search_path=... PostgreSQL allows to customize `search_path`, and it provides great power to the developer. With great power the following case could happen:

```
set search_path='app_v1';
SELECT * FROM mytable;
set search_path='app_v2';
SELECT * FROM mytable; -- Does mytable mean app_v1.mytable or app_v2.mytable here?
```

Server side prepared statements are linked to database object IDs, so it could fetch data from "old" `app_v1.mytable` table. It is hard to tell which behaviour is expected, however pgjdbc tries to track `search_path` changes, and it invalidates prepare cache accordingly.

The recommendation is: 1. Avoid changing `search_path` often, as it invalidates server side prepared statements 1. Use simple `set search_path...` commands, avoid nesting the comands into pl/pgsql or alike, otherwise pgjdbc won't be able to identify `search_path` change

Re-execution of failed statements It is a pity that a single `cached plan must not change result type` could cause the whole transaction to fail. The driver could re-execute the statement automatically in certain cases.

1. In case the transaction has not failed (e.g. the transaction did not exist before execution of the statement that caused `cached plan...` error), then pgjdbc re-executes the statement automatically. This makes the application happy, and avoids unnecessary errors.
2. In case the transaction is in a failed state, there's nothing to do but rollback it. pgjdbc does have "automatic savepoint" feature, and it could automatically rollback and retry the statement. The behaviour is controlled via `autosave` property (default `never`). The value of `conservative` would auto-rollback for the errors related to invalid server-prepared statements. Note: `autosave` might result in **severe** performance issues for long transactions, as PostgreSQL backend is not optimized for the case of long transactions and lots of savepoints.

Replication connection PostgreSQL replication connection does not allow to use server side prepared statements, so pgjdbc uses simple queries in the case where `replication` connection property is activated.

Use of server-prepared statements for con.createStatement() By default, pgjdbc uses server-prepard statements for `PreparedStatement` only, however you might want to activate server side prepared statements for regular `Statement` as well. For instance, if you execute the same statement through `con.createStatement().executeQuery(...)`, then you might improve performance by caching the statement. Of course it is better to use `PreparedStatements` explicitly, however the driver has an option to cache simple statements as well.

You can do that by setting `preferQueryMode` to `extendedCacheEverything`. Note: the option is more of a diagnostinc/debugging sort, so be careful how you use it .

Bind placeholder datatypes The database optimizes the execution plan for given parameter types. Consider the below case:

```
-- create table rooms (id int4, name varchar);
-- create index name__rooms on rooms(name);
PreparedStatement ps = con.prepareStatement("select id from rooms where name=?");
ps.setString(1, "42");
```

It works as expected, however what would happen if one uses `setInt` instead?

```
ps.setInt(1, 42);
```

Even though the result would be identical, the first variation (`setString` case) enables the database to use index `name__rooms`, and the latter does not. In case the database gets `42` as integer, it uses the plan like `where cast(name as int4) = ?`.

The plan has to be specific for the (`SQL text`; `parameter types`) combination, so the driver has to invalidate server side prepared statements in case the statement is used with different parameter types.

This gets especially painful for batch operations as you don't want to interrupt the batch by using alternating datatypes.

The most typical case is as follows (don't ever use this in production):

```
PreparedStatement ps = con.prepareStatement("select id from rooms where ...");
if (param instanceof String) {
    ps.setString(1, param);
} else if (param instanceof Integer) {
    ps.setInt(1, ((Integer) param).intValue());
} else {
    // Does it really matter which type of NULL to use?
   // In fact, it does since data types specify which server-procedure to call
    ps.setNull(1, Types.INTEGER);
}
```

As you might guess, `setString` vs `setNull(..., Types.INTEGER)` result in alternating datatypes, and it forces the driver to invalidate and re-prepare server side statement.

Recommendation is to use the consistent datatype for each bind placeholder, and use the same type for `setNull`. Check out `org.postgresql.test.jdbc2.PreparedStatementTest.testAl` example for more details.

Debugging In case you run into `cached plan must not change result type` or `prepared statement \"S_2\" does not exist` the following might be helpful to debug the case.

1. Client logging. If you add `loggerLevel=TRACE&loggerFile=pgjdbc-trace.log`, you would get trace of the messages send between the driver and the backend
2. You might check `org.postgresql.test.jdbc2.AutoRollbackTestSuite` as it verifies lots of combinations

Example 9.3. Using server side prepared statements

```
import java.sql.*;

public class ServerSidePreparedStatement
{

    public static void main(String args[]) throws Exception
    {
        Class.forName("org.postgresql.Driver");
        String url = "jdbc:postgresql://localhost:5432/test";
        Connection conn = DriverManager.getConnection(url,"test","");

        PreparedStatement pstmt = conn.prepareStatement("SELECT ?");

        // cast to the pg extension interface
        org.postgresql.PGStatement pgstmt = pstmt.unwrap(org.postgresql.PGStatement.class

        // on the third execution start using server side statements
        pgstmt.setPrepareThreshold(3);

        for (int i=1; i<=5; i++)
        {
            pstmt.setInt(1,i);
            boolean usingServerPrepare = pgstmt.isUseServerPrepare();
            ResultSet rs = pstmt.executeQuery();
            rs.next();
            System.out.println("Execution: "+i+", Used server side: " + usingServerPrepa
            rs.close();
        }

        pstmt.close();
        conn.close();
    }
}
```

Which produces the expected result of using server side prepared statements upon the third execution.

```
Execution: 1, Used server side: false, Result: 1
Execution: 2, Used server side: false, Result: 2
Execution: 3, Used server side: true, Result: 3
Execution: 4, Used server side: true, Result: 4
Execution: 5, Used server side: true, Result: 5
```

The example shown above requires the programmer to use PostgreSQL™ specific code in a supposedly portable API which is not ideal. Also it sets the threshold only for that particular statement which is some extra typing if we wanted to use that threshold for every statement. Let's take a look at the other ways to set the threshold to enable server side prepared statements. There is already a hierarchy in place above a `PreparedStatement`, the `Connection` it was created from, and above that the source of the connection be it a `Datasource` or a URL. The server side prepared statement threshold can be set at any of these levels such that the value will be the default for all of it's children.

```
// pg extension interfaces
org.postgresql.PGConnection pgconn;
org.postgresql.PGStatement pgstmt;

// set a prepared statement threshold for connections created from this url
String url = "jdbc:postgresql://localhost:5432/test?prepareThreshold=3";

// see that the connection has picked up the correct threshold from the url
Connection conn = DriverManager.getConnection(url,"test","");
pgconn = conn.unwrap(org.postgresql.PGConnection.class);
System.out.println(pgconn.getPrepareThreshold()); // Should be 3

// see that the statement has picked up the correct threshold from the connection
PreparedStatement pstmt = conn.prepareStatement("SELECT ?");
pgstmt = pstmt.unwrap(org.postgresql.PGStatement.class);
System.out.println(pgstmt.getPrepareThreshold()); // Should be 3

// change the connection's threshold and ensure that new statements pick it up
pgconn.setPrepareThreshold(5);
PreparedStatement pstmt = conn.prepareStatement("SELECT ?");
pgstmt = pstmt.unwrap(org.postgresql.PGStatement.class);
System.out.println(pgstmt.getPrepareThreshold()); // Should be 5
```

There are a number of connection parameters for configuring the client for SSL.
See SSL Connection parameters

The simplest being `ssl=true`, passing this into the driver will cause the driver to validate both the SSL certificate and verify the hostname (same as `verify-full`). **Note** this is different than libpq which defaults to a non-validating SSL connection.

In this mode, when establishing a SSL connection the JDBC driver will validate the server's identity preventing "man in the middle" attacks. It does this by checking that the server certificate is signed by a trusted authority, and that the host you are connecting to is the same as the hostname in the certificate.

If you **require** encryption and want the connection to fail if it can't be encrypted then set `sslmode=require` this ensures that the server is configured to accept SSL connections for this Host/IP address and that the server recognizes the client certificate. In other words if the server does not accept SSL connections or the client certificate is not recognized the connection will fail. **Note** in this mode we will accept all server certificates.

If `sslmode=verify-ca`, the server is verified by checking the certificate chain up to the root certificate stored on the client.

If `sslmode=verify-full`, the server host name will be verified to make sure it matches the name stored in the server certificate.

The SSL connection will fail if the server certificate cannot be verified. `verify-full` is recommended in most security-sensitive environments.

In the case where the certificate validation is failing you can try `sslcert=` and LibPQFactory will not send the client certificate. If the server is not configured to authenticate using the certificate it should connect.

The location of the client certificate, client key and root certificate can be overridden with the `sslcert`, `sslkey`, and `sslrootcert` settings respectively. These default to /defaultdir/postgresql.crt, /defaultdir/postgresql.pk8, and /defaultdir/root.crt respectively where defaultdir is ${user.home}/.postgresql/ in *nix systems and %appdata%/postgresql/ on windows

Finer control of the SSL connection can be achieved using the `sslmode` connection parameter. This parameter is the same as the libpq `sslmode` parameter and the currently SSL implements the following

sslmode

Eavesdropping Protection

MITM Protection

disable

No

No

I don't care about security and don't want to pay the overhead for encryption

allow

Maybe

No

I don't care about security but will pay the overhead for encryption if the server insists on it

prefer

Maybe

No

I don't care about encryption but will pay the overhead of encryption if the server supports it

require

Yes

No

I want my data to be encrypted, and I accept the overhead. I trust that the network will make sure I always connect to the server I want.

verify-ca

Yes

Depends on CA policy

I want my data encrypted, and I accept the overhead. I want to be sure that I connect to a server that I trust.

verify-full

Yes

Yes

I want my data encrypted, and I accept the overhead. I want to be sure that I connect to a server I trust, and that it's the one I specify.

Note

If you are using Java's default mechanism (not LibPQFactory) to create the SSL connection you will need to make the server certificate available to Java, the first step is to convert it to a form Java understands.

```
openssl x509 -in server.crt -out server.crt.der -outform der
```

From here the easiest thing to do is import this certificate into Java's system truststore.

```
keytool -keystore $JAVA_HOME/lib/security/cacerts -alias postgresql
-import -file server.crt.der
```

The default password for the cacerts keystore is `changeit`. The alias to postgresql is not important and you may select any name you desire.

If you do not have access to the system cacerts truststore you can create your own truststore.

```
keytool -keystore mystore -alias postgresql -import -file
server.crt.der
```

When starting your Java application you must specify this keystore and password to use.

```
java -Djavax.net.ssl.trustStore=mystore -Djavax.net.ssl.trustStorePassword=mypassword
com.mycompany.MyApp
```

In the event of problems extra debugging information is available by adding `-Djavax.net.debug=ssl` to your command line.

Using SSL without Certificate Validation

In some situations it may not be possible to configure your Java environment to make the server certificate available, for example in an applet. For a large scale deployment it would be best to get a certificate signed by recognized certificate authority, but that is not always an option. The JDBC driver provides an option to establish a SSL connection without doing any validation, but please understand the risk involved before enabling this option.

A non-validating connection is established via a custom `SSLSocketFactory` class that is provided with the driver. Setting the connection URL parameter `sslfactory=org.postgresql.ssl.NonValidatingFactory` will turn off all SSL validation.

PostgreSQL™ provides a way for developers to customize how a SSL connection is established. This may be used to provide a custom certificate source or other extensions by allowing the developer to create their own `SSLContext` instance. The connection URL parameters `sslfactory` allow the user to specify which custom class to use for creating the `SSLSocketFactory`. The class name specified by `sslfactory` must extend `javax.net.ssl.SSLSocketFactory` and be available to the driver's classloader.

This class must have a zero argument constructor or a single argument constructor preferentially taking a `Properties` argument. There is a simple `org.postgresql.ssl.DefaultJavaSSLFactory` provided which uses the default java SSLFactory.

Information on how to actually implement such a class is beyond the scope of this documentation. Places to look for help are the JSSE Reference Guide and the source to the `NonValidatingFactory` provided by the JDBC driver.

The following must be considered when using the `Statement` or `PreparedStatement` interface:

- You can use a single `Statement` instance as many times as you want. You could create one as soon as you open the connection and use it for the connection's lifetime. But you have to remember that only one `ResultSet` can exist per `Statement` or `PreparedStatement` at a given time.
- If you need to perform a query while processing a `ResultSet`, you can simply create and use another `Statement`.
- If you are using threads, and several are using the database, you must use a separate `Statement` for each thread. Refer to Chapter 10, *Using the Driver in a Multithreaded or a Servlet Environment* if you are thinking of using threads, as it covers some important points.
- When you are done using the `Statement` or `PreparedStatement` you should close it.
- In JDBC, the question mark (`?`) is the placeholder for the positional parameters of a `PreparedStatement`. There are, however, a number of PostgreSQL operators that contain a question mark. To keep such question marks in a SQL statement from being interpreted as positional parameters, use two question marks (`??`) as escape sequence. You can also use this escape sequence in a `Statement`, but that is not required. Specifically only in a `Statement` a single (`?`) can be used as an operator.

- Development
- GIT
- Translations
- Website
- Todo
- Private API

Branch - 91

bg

cs

de

es

fr

it

ja

nl

pl

pt_BR

ru

sr

tr

zh_CN

zh_TW

Template - Messages: 224

95

42

74

9

74

70

97

3

32

82

21

82

82

55

55

- Development
- GIT
- Translations

- Website
- Todo
- Private API

Known Bugs

- [**bugs**] Deallocating large numbers of server side statements can break the connection by filling network buffers. This is a very, very low probability bug, but it is still possible. ref →

Compliance

- [**JDBC1**] Implement Statement.setQueryTimeout. →
- [**JDBC2**] Sort DatabaseMetaData.getTypeInfo properly (by closest match). →
- [**JDBC2**] Implement SQLInput and SQLOutput to allow composite types to be used. →
- [**JDBC3**] Implement Statement.getGeneratedKeys. ref2 →
- [**JDBC3**] The JDBC 3 DatabaseMetaData methods sometimes return additional information. Currently we only return JDBC 2 data for these methods. ref →
- [**JDBC3**] Implement Clob write/position methods. →

Performance

- [] Add statement pooling to take advantage of server prepared statements. →
- [] Allow scrollable ResultSets to not fetch all results in one batch. →
- [] Allow refcursor ResultSets to not fetch all results in one batch. →
- [] Allow binary data transfers for all datatypes not just bytea. →

PG Extensions

- [] Allow configuration of GUC parameters via the Connection URL or Datasource. The most obvious example of usefulness is search_path. ref →

Other

- [**test**] Pass the JDBC CTS (Sun's test suite). →
- [**code**] Allow SSL to use client certificates. This can probably be done with our existing SSLSocketFactory customization code, but it would be good to provide an example or other wrapper so a non-expert can set it up. ref1, ref2 →
- [**code**] Currently the internal type cache is not schema aware. →
- [**code**] Need a much better lexer/parser than the ad hoc stuff in the driver. ref2 →

Ideas

- [] Allow Blob/Clob to operate on bytea/text data. ref →
- [] Allow getByte/getInt/… to work on boolean values ref →
- [] Add a URL parameter to make the driver not force a rollback on error for compatibility with other dbs. The driver can wrap each statement in a Savepoint. ref →
- [] Combine DatabaseMetaData efforts with pl/java. ref →
- [] ResultSetMetaData calls that run queries are cached on a per column basis, but it seems likely that they're going to be called for all columns, so try to issue one query per ResultSet, not per column. →
- [] Make PGConnection, PGStatement, … extend java.sql.XXX ref →

Documentation

- [] The PGResultSetMetaData interface is not mentioned. →
- [] Timestamp +/- Infinity values are not mentioned. →
- [] Async notifies are more async now. ref →

Website

- [] Setup a cron job somewhere to build and deploy the sight on a daily basis to keep API changes and translations up to date. →
- [] Add a daily git snapshot build to make the latest updates available. →

Note

The postgresql.jar file must be placed in $CATALINA_HOME/common/lib in both Tomcat 4 and 5.

The absolute easiest way to set this up in either tomcat instance is to use the admin web application that comes with Tomcat, simply add the datasource to the context you want to use it in.

Setup for Tomcat 4 place the following inside the <Context> tag inside conf/server.xml

```
<Resource name="jdbc/postgres" scope="Shareable" type="javax.sql.DataSource"/>
<ResourceParams name="jdbc/postgres">
    <parameter>
        <name>validationQuery</name>
        <value>select version();</value>
    </parameter>
    <parameter>
        <name>url</name>
        <value>jdbc:postgresql://localhost/davec</value>
    </parameter>
    <parameter>
        <name>password</name>
        <value>davec</value>
    </parameter>
    <parameter>
        <name>maxActive</name>
        <value>4</value>
    </parameter>
    <parameter>
        <name>maxWait</name>
        <value>5000</value>
```

```
    </parameter>
    <parameter>
        <name>driverClassName</name>
        <value>org.postgresql.Driver</value>
    </parameter>
    <parameter>
        <name>username</name>
        <value>davec</value>
    </parameter>
    <parameter>
        <name>maxIdle</name>
        <value>2</value>
    </parameter>
</ResourceParams>
```

Setup for Tomcat 5, you can use the above method, except that it goes inside the <DefaultContext> tag inside the <Host> tag. eg. <Host> ... <DefaultContext> ...

Alternatively there is a conf/Catalina/hostname/context.xml file. For example http://localhost:8080/servlet-example has a directory $CATALINA_HOME/conf/Catalina/localhost/servl example.xml file. Inside this file place the above xml inside the <Context> tag

Then you can use the following code to access the connection.

```
import javax.naming.*;
import javax.sql.*;
import java.sql.*;
public class DBTest
{

    String foo = "Not Connected";
    int bar = -1;

    public void init()
    {
        try
        {
            Context ctx = new InitialContext();
            if(ctx == null )
                throw new Exception("Boom - No Context");

            // /jdbc/postgres is the name of the resource above
            DataSource ds = (DataSource)ctx.lookup("java:comp/env/jdbc/postgres");
```

```
            if (ds != null)
            {
                Connection conn = ds.getConnection();

                if(conn != null)
                {
                    foo = "Got Connection "+conn.toString();
                    Statement stmt = conn.createStatement();
                    ResultSet rst = stmt.executeQuery("select id, foo, bar from testdata

                    if(rst.next())
                    {
                        foo=rst.getString(2);
                        bar=rst.getInt(3);
                    }
                    conn.close();
                }
            }
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }

    public String getFoo() { return foo; }

    public int getBar() { return bar;}
}
```

Overview

Translations of driver specific error messages are available in a number of languages. The following information is about how to update an existing translation or to provide a translation to a new language.

The translations are done using the GNU gettext tools. These are not required to build or use the driver, only translators and maintainers need them.

Developers

When writing code for the driver you need to specially mark strings for translation so they are picked up by the tools. In general any user visible message should be made available for translation. Strings are marked using the GT.tr method. The name means "gettext translate", but a shorter name was wanted because this shows up in a lot of places.

To provide context sensitive information the standard Java MessageFormat syntax is used in the error messages. Consider, for example, the error message for calling ResultSet.getInt() with an invalid column number, we want to helpfully report the column asked for and the number of columns in the ResultSet.

```
if (column < 1 || column > fields.length) {
    String err = GT.tr(
        "The column index requested:  is out of range.",
        new Integer(column)
        );
   throw new PSQLException(err, PSQLState.INVALID_PARAMETER_VALID);
}
```

Translators

Check the current translation status page to see if an existing translation exists for your language, so you can update that instead of starting from scratch. To start a new translation you can download the template file and work on that instead.

Editing the .po file is a pretty straightforward process and a number of tools exist to aid you in the process:

- GNU Emacs and XEmacs have PO editing modes.
- KBabel is a KDE-based editing tool.
- poEdit is another tool which can run on Windows.

Once you feel the translation is accurate and complete (or you get tired), verify that the file by running msgfmt.

```
msgfmt -c -v -o /dev/null pofile
```

If everything checks out send the po file on over to the pgsql-jdbc@postgresql.org mailing list. This list does have a size limit of 30k, so you will need to compress the po file before sending it.

Maintainers

To avoid requiring the gettext tools to compile the driver the decision has been made to directly check in the compiled message catalogs to the git repository. When you get a new or updated translation, first ensure that it is valid by running the msgfmt command mentioned in the translators section. If this looks correct drop it into the src/org/postgresql/translation directory and run the update-translations.sh script in the top level directory. This will produce the compiled class file that contains the translated messages. Then simply check in both the .po and .class files. Be sure to only commit changes to the translations you've modified because the update-translations.sh script modifies all the translations.

To change data (perform an `INSERT`, `UPDATE`, or `DELETE`) you use the `executeUpdate()` method. This method is similar to the method `executeQuery()` used to issue a `SELECT` statement, but it doesn't return a `ResultSet`; instead it returns the number of rows affected by the `INSERT`, `UPDATE`, or `DELETE` statement. Example 5.3, "Deleting Rows in JDBC" illustrates the usage.

Example 5.3. Deleting Rows in JDBC

This example will issue a simple `DELETE` statement and print out the number of rows deleted.

```
int foovalue = 500;
PreparedStatement st = conn.prepareStatement("DELETE FROM mytable WHERE columnfoo = ?");
st.setInt(1, foovalue);
int rowsDeleted = st.executeUpdate();
System.out.println(rowsDeleted + " rows deleted");
st.close();
```

- Development
- GIT
- Translations
- Website
- Todo
- Private API

Building the Website

The website is produced with Jekyll. It allows you to build a reasonably good looking website that is easy to maintain and modular in nature. Templates are used from the _layout and _includes directories which are then used in conjunction with content that is created with Markdown, Textile, or just standard HTML for input. Using Markdown or Textile allows the content to be generated with simple rules that allow a more free flowing process of writing without worrying about coding for the HTML.

To get started please read the Jekyll website for installation instructions for that tool. After installing Jekyll you need to get the website project. This is available from the same git repository that the main source code is, it's just a different module, www. Checkout this module and then within the top level directory of the module simply run jekyll build. This should produce the website in the _site subdirectory.

Adding a Page

To add a page the easiest thing to do is copy an existing one from the site like about/about and then rip out its contents. Once new content is created and you have saved the new page you will need to add the new file to the menu system of the appropriate _includes subdirectory submenu.

A good place to look for example content is Jekyll's wiki site where there is a listing of real sites that use Jekyll. Many of these sites will have their repositories available for review.

When creating a database to be accessed via JDBC it is important to select an appropriate encoding for your data. Many other client interfaces do not care what data you send back and forth, and will allow you to do inappropriate things, but Java makes sure that your data is correctly encoded. Do not use a database that uses the `SQL_ASCII` encoding. This is not a real encoding and you will have problems the moment you store data in it that does not fit in the seven bit ASCII character set. If you do not know what your encoding will be or are otherwise unsure about what you will be storing the `UNICODE` encoding is a reasonable default to use.