ZMySQLDA Documentation

Release 4.7.dev0

Zope Foundation and Contributors

Mar 04, 2020

Contents

This is the MySQL database adapter product for Zope.

CHAPTER 1

Compatibility

- `Zope2` 2.13.26 and up under Python 2.7 in conjunction with `Products.ZSQLMethods` up to and including 2.13.5.
- `Zope` 4 after 4.0beta5 under Python 2.7 and Python 3.5 and higher.
- MySQL versions 3.22 and later including MariaDB. You need version 4.1 or higher for unicode support.

CHAPTER 2

Narrative documentation

2.1 Connection Strings

The connection string used for Z MySQL Database Connection objects are of the form:

```
[*lock_name][+|-]database[@host[:port]] [user [password [unix_socket]]]
```

or typically just:

```
database user password
```

to use a MySQL server on the local host via the standard UNIX socket.

The components are as follows:

- `*lock_name` at the begining of the connection string leads to pseudo-transactional behavior. When the Zope transaction begins, a lock named after `lock_name` is acquired on the database server. When the Zope transaction commits, the database lock will be released. If the Zope transaction is aborted and restarted, which can happen due to ConflictErrors, you'll get an error in the logs, and inconsistent data. In this respect, it's equivalent to transactions turned off.

  Transactions are highly recommended. Using a named lock in conjunctions with transactions is probably pointless.

- + or −: Integrate database transactions with the Zope transaction machinery. A − in front of the database tells ZMySQLDA to not use Zope's Transaction Manager, even if the server supports transactions. A + in front of the database tells ZMySQLDA that it must use transactions; an exception will be raised if they are not supported by the server. If neither − or + are present, then transactions will be enabled if the server supports them. If you are using non-transaction safe tables (TSTs) on a server that supports TSTs, use −. If you require transactions, use +. If you aren't sure, don't use either.

- `database`: The name of the database to connect to.

- `host`/`port`: Host and port where the database server listens. Only use this if the database server is on a remote system. To use a non-standard port on the local system, use 127.0.0.1 for the host instead of the hostname `localhost`.

- `user/password`: Log into the database with the provided user and password.
- `unix_socket`: If the UNIX socket is in a non-standard location, you can specify the full path to it after the `password`.

2.2 Usage from the Zope ZMI

The database connection object can be manipulated in the Zope ZMI on a series of screens, accessible through named tabs in the main window.

2.2.1 Status

Shows the database connection status and allows the user to open or close the connection.

2.2.2 Properties

Edit the database connection attributes and apply any changes:

- *Title*: An optional title that shows up in the ZMI.
- *Database Connection String*: A string encapsulating how to connect to the database. See *Connection Strings* for details.
- *Connect immediately*: Should the database connection be established immediately or when the first database query is run.
- *Unicode Support*: If set to `True`, values from columns of type `CHAR`, `VARCHAR` and `TEXT` are returned as unicode strings by the database backend.
- *Character set*: Query results will be encoded in the character set specified here:
  - *Not set* will emulate previous releases' behavior on Python 2, which used Latin-1 (ISO 8859-1), but if *Unicode results* is selected, the connection character set switches to UTF-8 and strings in query results are decoded to Unicode. On Python 3, *not set* always defaults to UTF-8.
  - For Python 2, you can force the character set to Latin-1 or UTF-8, regardless of the *Unicode results* setting. This is useful when your application wants to use UTF-8, but cannot deal with unicode return values.
  - **On Python 3, forcing the character set to Latin1 is not supported.**
- *Automatically create database*: If the *Database Connection String* refers to a database that does not yet exist *and* this setting is activated, the ZMySQLDA connector will attempt to create the database.

2.2.3 Test

The Test tab can be used as long as the database connection is connected. You can enter SQL statements into the text field and view the results sent back from the database.

2.2.4 Security

Change the Zope role to permission mappings here.

2.2.5 Undo

If your particular ZODB flavor supports it, you can undo Zope transactions affecting the database connector object here. These transactions don't reflect relational database transactions in the underlying MySQL or MariaDB databases, only ZODB transactions.

2.2.6 Ownership

Information about the Zope user who owns the database connector object. Ownership in the Zope sense confers additional rights.

2.2.7 Interfaces

View and change the Zope Interface assignments for the database connector object.

2.2.8 Browse

You can browse the database tables and columns from the relational database specified in the connection string.

2.3 Development

2.3.1 Getting the source code

The source code is maintained on GitHub. To check out the trunk:

```
$ git clone https://github.com/zms-publishing/Products.ZMySQLDA.git
```

You can also browse the code online at https://github.com/zms-publishing/Products.ZMySQLDA

2.3.2 Bug tracker

For bug reports, suggestions or questions please use the GitHub issue tracker at https://github.com/zms-publishing/Products.ZMySQLDA/issues.

2.3.3 Running the tests using `zc.buildout`

`Products.ZMySQLDA` ships with its own `buildout.cfg` file and `bootstrap.py` for setting up a development buildout:

```
$ python bootstrap.py
...
Generated script '.../bin/buildout'
$ bin/buildout
...
```

Once you have a buildout, the tests can be run as follows:

```
$ bin/test
Running tests at level 1
Running zope.testrunner.layer.UnitTests tests:
  Set up zope.testrunner.layer.UnitTests in 0.000 seconds.
  Running:
..............................................................
  Ran 62 tests with 0 failures and 0 errors in 0.043 seconds.
Tearing down left over layers:
  Tear down zope.testrunner.layer.UnitTests in 0.000 seconds.
```

To run tests for all supported Python versions, code coverage and a PEP-8 coding style checker, you can use `tox` after completing the buildout step above:

```
$ bin/tox
GLOB sdist-make: ...
...
_____________________________________ summary ______________________________________
py27: commands succeeded
py27_zope213: commands succeeded
py35: commands succeeded
py36: commands succeeded
flake8: commands succeeded
coverage: commands succeeded
congratulations :)
```

2.3.4 Running the functional tests

Some tests are hard or even impossible to perform without a real running database backend. During a normal test run they will be skipped, and you will see output like this:

```
Total: 62 tests, 0 failures, 0 errors and 5 skipped in 0.090 seconds.
```

To run those functional tests you need to have a MySQL/MariaDB server running and listening on the standard unix socket, normally located at `/tmp/mysl.sock`. This database server must have a database named `zmysqldatest` that can be accessed by a user `zmysqldatest` with password `zmysqldatest`. To set this up, log into the running database server with an admin user and execute the following statements:

```
mysql> CREATE DATABASE IF NOT EXISTS zmysqldatest;
mysql> CREATE USER 'zmysqldatest'@'localhost' IDENTIFIED BY 'zmysqldatest';
mysql> GRANT ALL PRIVILEGES ON zmysqldatest.* TO 'zmysqldatest'@'localhost';
```

If everything worked you'll see test output like this:

```
Total: 62 tests, 0 failures, 0 errors and 0 skipped in 0.105 seconds.
```

If the functional tests are still skipped, uncomment the `print` call in the `_mySQLNotAvailable` function in the module `Products.ZMySQLDA.tests.base`. It will print any errors emitted by the database server.

2.3.5 Building the documentation using `zc.buildout`

The `Products.ZMySQLDA` buildout installs the Sphinx scripts required to build the documentation, including testing its code snippets:

```
$ cd docs
$ make html
...
build succeeded.

The HTML pages are in _build/html.
```

2.3.6 Making a release

These instructions assume that you have a development sandbox set up using `zc.buildout` as the scripts used here are generated by the buildout.

```
$ bin/buildout -N
$ python setup.py sdist bdist_wheel upload --sign
```

The `bin/buildout` step will make sure the correct package information is used.

2.4 Contributors

The following list of people who have contributed code or documentation makes no claims about completeness.

- Andy Dustman
- John Eikenberry
- Vincent Pelletier
- Graeme Mathieson
- Federico Schwindt
- Brett Carter
- Mark Van den Borre
- Robert Buchholz
- Jens Vagelpohl

2.5 Products.ZMySQLDA change log

2.5.1 4.7 (unreleased)

2.5.2 4.6 (2020-03-03)

- removed error-prone server version check for savepoint support (#18)
- added aditional unit tests for SQL quoting

2.5.3 4.5 (2019-10-13)

- rely on the Zope 4 branch for tests so we don't lose Python 2 compatibility
- combine Status and Properties ZMI tabs to improve usability

2.5.4 4.4 (2019-06-17)

- add timeout parameter for DatabaseAdapter and database pool classes
- add timeout parameter to Add and Edit Forms
- make sure timeout is None or int (#10)

2.5.5 4.3 (2019-05-28)

- make sure SQL quoting an unencoded string does not change the string type

2.5.6 4.2 (2019-05-21)

- fix wrong use of `charset` for `unicode_literal`

2.5.7 4.1 (2019-04-26)

- fix the Browse tab under Python 3 (#14)
- add more content to the long_description metadata (#13)
- fixes to the Browse ZMI tab

2.5.8 4.0 (2019-03-31)

- fix database version detection for savepoint support (#7)
- remove explicit `setuptools` version pin in `setup.py` (#11)
- more strict flake8 code style compliance
- trove classifier cleanup
- buildout configuration cleanup

2.5.9 4.0b5 (2019-02-20)

- When editing a connection in the ZMI, show an error message and not an exception when the connection fails.
- ZMI usability enhanced by providing feedback when editing a connection
- Specify supported Python versions using `python_requires` in setup.py
- Added support for Python 3.8

2.5.10 4.0b4 (2019-01-24)

- additional compatibility fixes for mysqlclient 14.0 and up

2.5.11 4.0b3 (2019-01-22)

- compatibility fix for mysqlclient 14.0 and up

2.5.12 4.0b2 (2018-12-11)

- added the ability to set the MySQL connection character set separate from the unicode flag.
- make the checkboxes on the add view work correctly
- declare (and test for) Python 3.7 compatibility
- make the checkboxes on the *Properties* ZMI tab work
- added missing `six` dependency declaration
- add some functional tests that require a running database server, see the documentation for how to set it up.

2.5.13 4.0b1 (2018-06-11)

- New maintainers: SNTL PUBLISHING / HOFFMANN+LIEBENBERG GMBH and Jens Vagelpohl
- Moved away from the unsupported `MySQLdb1` to the fork `mysqlclient`, which is Python 3-compatible.
- Added simple buildout configuration with tox integration
- Zope 4 and Python 3 compatibility
- Added unit tests
- Added some `Sphinx`-based documentation and copied any useful items from the old `HelpSys` files.
- Removed the hurt system files.
- Improved the `Browse` tab with more table information.

2.5.14 3.1.1

- #3106015: zope 2.12/plone4 compatibility fix (thanks Mark Van den Borre)
- #3076433: column descriptions always said NOT NULL (thanks Frank Hoffmann)

2.5.15 3.1

- #2357223: Savepoint support

2.5.16 3.0

- Added Setuptools support to create an egg package thanks to Brett Carter.
- Added condition to handle connection getting "out of sync". This can occur when, for instance, you get a stray semicolon in a query. When a connection gets in this state it is hosed and must be closed and reconnected.
- Made some changes to how `use_unicode` and `auto_create_db` are set to better allow for subclassing and extensions of the base classes.
- Added handling of NEWDECIMAL which was added for mysql 5.0.
- Added basic support for procedure calls using `CALL` query.

2.5.17 3.0beta1

- Fixed issue with `sql_quote__` getting called prior to connection being made.
- Fixed bug #1916952. Updating to API change in MySQLdb 1.2.2 ping method.
- Fixed backwards compatibility issue with MySQLdb versions <= 1.2.1.

2.5.18 3.0alpha4

- Fixed pernicious corner case bug with joining a transaction after the transaction has started and been aborted.
- Zope dependency raised to Zope-2.8 or newer.

2.5.19 3.0alpha3

- Unicode support now works!
- Unicode support reworked to use MySQLdb's unicode support instead of its own half-baked layer.
- Minor cleanups and extensions to database introspection methods.
- Minor cleanups/fixes to dtml.
- Removed a few unnecessary thread locks.
- Changed failed query logging entries from errors to warnings.

2.5.20 3.0alpha2

- Moved DBPool instantiation from factory() to connect() to better facilitate API backwards compatibility.
- Changed all default values on keyword aruments for the auto create db feature. They all now default to True.
- Left in a bit of debugging code that disabled the new create_db functionality. Removed it.

2.5.21 3.0alpha1

- New maintainer: John Eikenberry
- Note that there are some changes in the internal API. So if you have subclassed you should double check compatibility.

Features:

- Experimental Unicode support has been added. It is hardcoded to UTF-8 and has had limited testing at this point. Adapted from patches made by Graeme Mathieson.
- New optional feature of automatically creating the database provided in the connection string. The mysql `user` used for the connection must have CREATE permission. It defaults to on to encourage more testing.
- Database connection not created until first use instead of when the object is first loaded. Ie. connection created at `connect()` call instead of `__setstate__()` call. This helps conserve system resources and makes debugging connection issues a bit easier. It is also needed for the new db pool implementation (see below).

Bugs:

- Automatically reopens connections closed by client timeouts.
- Fixed major deadlock causing bug that can occur with versions of Zope greater than 2.8. It was caused by the use of the volatile attribute `_v_` to keep the reference to the existing connection. Volatile attributes can go away mid-transaction which would cause a deadlock when used with a transactional engine (eg. innodb). The fix involves a fixed pool of adapters and db connections. This also allowed for the elimination of many of the locks. Adapted from patches made by Vincent Pelletier.
- #670137: missing `sortKey()` fixed in Zope
- #814378: infinite reconnect recursion fixed
- #1560557: missing import
- #1242842: missing `MULTI_STATEMENTS`
- #1226690: missing `close()` method

2.5.22 2.0.9

- Allow the connection string to work without a specified database.
- Wrap queries with a lock to prevent multiple threads from using the connection simultaneously (this may or may not be happening). If transactional, then there is an additional transaction lock, acquired at the beginning of the transaction and released when either finished or aborted.
- A named lock can be specified by inserting `*LOCKNAME` at the start of the connection string. This is probably best used only if you must use non-transactional tables.
- Some stuff will be logged as an error when bad things happen during the transaction manager hooks.

2.5.23 2.0.8

- More information about columns is available from the table browser. This is primarily to support SQL Blender.
- `DECIMAL` and `NUMERIC` columns now returned as floating-point numbers (was string). This has also been fixed in MySQLdb-0.9.1, but the fix is included here just in case you don't upgrade. Upgrading is a good idea anyway, because some memory-related bugs are fixed, particularly if using Zope 2.4 and Python 2.1.

2.5.24 2.0.7

- Transaction support tweaked some more. A plus (+) or minus (−) at the beginning of the connection string will force transactions on or off respectively. By default, transactions are enabled if the server supports them. Beware: If you are using non-TST tables on a server that supports transactions, you should probably force transactions off.

2.5.25 2.0.6

- This version finally should have all the transaction support working correctly. If your MySQL server supports transactions, i.e. it has at least one transaction-safe table (TST) handler, transactions are enabled automatically. If transactions are enabled, rollbacks (aborts) fail if any non-TST tables were modified.

2.5.26 2.0.5

- Transactions don't really work right in this and prior versions.

2.5.27 2.0.4

- `INT` columns, whether `UNSIGNED` or not, are returned as Python long integers to avoid overflows. Python-1.5.2 adds an `L` to the end of long integers when printing. Later versions do not. As a workaround, use affected columns with a format string, i.e. `<dtml-var x fmt="%d">`.

2.5.28 2.0.0

- This is the first version of the database adapter using MySQLdb for Zope. This database adapter is based on the Z DCOracle DA version 2.2.0.

CHAPTER 3

API documentation

3.1 Products.ZMySQLDA.DA.Connection

CHAPTER 4

Indices and tables

- genindex
- modindex
- search
- glossary