
pyAttachSQL Documentation

Release 0.2.0

Andrew Hutchings

March 18, 2015

1	Introduction	3
1.1	What is pyAttachSQL?	3
1.2	Licensing	3
1.3	Installing	6
2	API Documentation	9
2.1	pyAttachSQL Module Basics	9
2.2	Connection Class	12
2.3	Group Class	14
2.4	Query Class	14
2.5	Prepared Statement Class	16
3	DB API Documentation	19
3.1	DB API Module Basics	19
3.2	DB API Connection Class	20
3.3	DB API Cursor Class	20
4	Indices and tables	23
	Python Module Index	25

Release 0.2.0

Date March 18, 2015

Introduction

1.1 What is pyAttachSQL?

pyAttachSQL is a lightweight, high performance, asynchronous library for Python applications designed to connect to MySQL servers. It is a wrapper for the C client library [libAttachSQL](#).

It is Apache 2.0 licensed so that it is possible to use both with Open Source and Commercial applications. It is also designed to provide a relatively easy to use API.

1.2 Licensing

1.2.1 Documentation Content



The pyAttachSQL Documentation is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](#).

1.2.2 pyAttachSQL License

pyAttachSQL is licensed under the [Apache License, Version 2.0](#).

Apache License
Version 2.0, January 2004
<http://www.apache.org/licenses/>

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition,

"control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.
3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable

(except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:
 - (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and
 - (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
 - (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and
 - (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions.

Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.
7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.
8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.
9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

1.3 Installing

1.3.1 Requirements

Before installing pyAttachSQL you will need the following installed:

- libAttachSQL
- tox

It was designed to be used with Python 2.7 and is not yet compatible with Python 3.x.

1.3.2 From Source

To install from source run the following:

```
python setup.py install
```

API Documentation

2.1 pyAttachSQL Module Basics

2.1.1 Exceptions

exception `attachsql.ClientError`

An exception on the client side

exception `attachsql.ServerError`

As exception on the server side

2.1.2 Constants

Return Constants

`attachsql.RETURN_NONE`

No return status

`attachsql.RETURN_NOT_CONNECTED`

The client is not yet connected to the server

`attachsql.RETURN_PROCESSING`

Data processing / transfer is in-progress

`attachsql.RETURN_ROW_DATA`

A row is in the buffer ready for processing

`attachsql.RETURN_ERROR`

An error occurred on the connection

`attachsql.RETURN_EOF`

A query EOF, typically all rows have been returned

Escape Constants

`attachsql.ESCAPE_TYPE_NONE`

Nothing to escape

`attachsql.ESCAPE_TYPE_CHAR`

Escape string data adding quote marks around the string

`attachsql.ESCAPE_TYPE_CHAR_LIKE`

Escape string data without quote marks around the string (for use in LIKE syntax)

`attachsql.ESCAPE_TYPE_INT`

Insert integer data into the query

`attachsql.ESCAPE_TYPE_BIGINT`

Insert 64bit integer data into the query

`attachsql.ESCAPE_TYPE_FLOAT`

Insert float data into the query

`attachsql.ESCAPE_TYPE_DOUBLE`

Insert double data into the query

Option Constants

`attachsql.OPTION_COMPRESS`

Protocol compression option

`attachsql.OPTION_FOUND_ROWS`

Found rows counter for results

`attachsql.OPTION_IGNORE_SIGPIPE`

Ignore sigpipe (not used, sigpipe is ignored anyway)

`attachsql.OPTION_INTERACTIVE`

Client is interactive

`attachsql.OPTION_LOCAL_FILES`

Enable `LOAD LOCAL INFILE`

`attachsql.OPTION_MULTI_STATEMENTS`

Enable multi-statement queries

`attachsql.OPTION_NO_SCHEMA`

Disable the `schema_name.table_name.column_name` syntax (for ODBC)

`attachsql.OPTION_SSL_NO_VERIFY`

Currently unused

`attachsql.OPTION_SEMI_BLOCKING`

Enable semi-blocking mode

Column Type Constants

`attachsql.COLUMN_TYPE_DECIMAL`

`attachsql.COLUMN_TYPE_TINY`

`attachsql.COLUMN_TYPE_SHORT`

`attachsql.COLUMN_TYPE_LONG`

`attachsql.COLUMN_TYPE_FLOAT`

`attachsql.COLUMN_TYPE_DOUBLE`

`attachsql.COLUMN_TYPE_NULL`

`attachsql.COLUMN_TYPE_TIMESTAMP`

`attachsql.COLUMN_TYPE_LONGLONG`

```
attachsql.COLUMN_TYPE_INT24
attachsql.COLUMN_TYPE_DATE
attachsql.COLUMN_TYPE_TIME
attachsql.COLUMN_TYPE_DATETIME
attachsql.COLUMN_TYPE_YEAR
attachsql.COLUMN_TYPE_VARCHAR
attachsql.COLUMN_TYPE_BIT
attachsql.COLUMN_TYPE_NEWDECIMAL
attachsql.COLUMN_TYPE_ENUM
attachsql.COLUMN_TYPE_SET
attachsql.COLUMN_TYPE_TINY_BLOB
attachsql.COLUMN_TYPE_MEDIUM_BLOB
attachsql.COLUMN_TYPE_LONG_BLOB
attachsql.COLUMN_TYPE_BLOB
attachsql.COLUMN_TYPE_VARSTRING
attachsql.COLUMN_TYPE_STRING
attachsql.COLUMN_TYPE_GEOMETRY
```

2.1.3 Callback Event Constants

```
attachsql.EVENT_CONNECTED
attachsql.EVENT_ERROR
attachsql.EVENT_EOF
attachsql.EVENT_ROW_READY
```

2.1.4 Functions

```
attachsql.connect(hostname, user, password, database, port)
```

Parameters

- **hostname** (*str*) – The hostname to the server
- **user** (*str*) – The user name to connect with
- **password** (*str*) – The password to connect with
- **database** (*str*) – The default database for the connection
- **port** (*int*) – The port to connect on or 0 for a Unix Domain Socket connection

Returns An instance of the `connection`

```
attachsql.get_library_version()
```

Gets the version of libAttachSQL used for pyAttachSQL

Returns A string representation of the version number

Return type str

2.1.5 Callback Function Prototypes

my_callback(events, con, query, unused):

A user defined callback used for `group`

Parameters

- **events** (*int*) – The event which triggered the callback from *Callback Event Constants*
- **con** (*object*) – The connection object which triggered the callback
- **query** (*object*) – The query object which triggered the callback
- **context** (*object*) – The user supplied context

2.2 Connection Class

class attachsql.**connection** (*hostname, user, password, database, port*)

Parameters

- **hostname** (*str*) – The hostname to the server
- **user** (*str*) – The user name to connect with
- **password** (*str*) – The password to connect with
- **database** (*str*) – The default database for the connection
- **port** (*int*) – The port to connect on or 0 for a Unix Domain Socket connection

connection_id()

Returns the connection ID of the connection. If there is no established connection this will return 0.

Returns The ID for the connection

Return type int

connect()

Start connection to the server. `poll()` needs to be called (usually more than once) for the connection to actually occur.

Returns True on success

Return type bool

poll()

Poll a connection to see if there is more data.

Returns A numeric status to be compared with *Return Constants*

Return type int

get_server_version()

Returns the version string for the server

Returns The version string

Return type str

query (*query*[, *parameters*])

Send a query to the server. Parameters can be given to fill in ? markers in a query. This should be in a Python list containing dictionaries formatted as follows as follows:

```
[{'type': attachsql.ESCAPE_TYPE_CHAR, 'data':'hello'}, {'type': attachsql.ESCAPE_TYPE_INT, 'data':1}],
```

For a full list of types see *Escape Constants*

Parameters

- **query** (*str*) – The query statement to send
- **parameters** (*list*) –

Returns A `query` class

Return type query

```
set_ssl (key, cert, ca, capath, cipher[, verify])
```

Sets the SSL certifications and enables SSL connections

Parameters

- **key** (*str*) – The path for the key file
- **cert** (*str*) – The path for the certificate
- **ca** (*str*) – The path for the CA
- **capath** (*str*) – The path containing many CAs
- **cipher** (*str*) – A list of ciphers to allow
- **verify** (*bool*) – Set to verify the SSL connection

Returns True on success

Return type boolean

```
set_option (option[, unused ])
```

Sets a connection option, the list of possible options can be found in *Option Constants*

Parameters

- **option** (*int*) – The option to set
- **unused** – This parameter is for future use

Returns True on success

Return type boolean

```
prepare_statement (statement)
```

Initialize and start sending a prepared statement

Parameters `statement` (*str*) – The statement to send

Returns An instance of `statement`

Return type statement

2.3 Group Class

class `attachsql.group` (*callback_function*, *callback_context*)

Creates a group of connections to execute statements in the same event loop. In this mode events trigger callbacks instead of the normal polling so a callback function is required.

Parameters

- **callback_function** (*object*) – The function name to use for the callback
- **callback_context** (*object*) – An object of arbitrary data to send to the callback function

create_connection (*hostname*, *user*, *password*, *database*, *port*)

Creates a new connection in the connection group and returns the connection object. The connection itself is not made at this time.

Parameters

- **hostname** (*str*) – The hostname to the server
- **user** (*str*) – The user name to connect with
- **password** (*str*) – The password to connect with
- **port** (*int*) – The port to connect on or 0 for a Unix Domain Socket connection

Returns A connection class which is attached to this group

Return type `connection`

run ()

Runs a single iteration of the event loop. If an event is triggered the `py:function:my_callback` will be fired before this returns.

Returns `None`

2.4 Query Class

class `attachsql.query`

column_count ()

Returns the number of columns in a query result

Returns The number of columns

Return type `int`

row_get ()

Returns a Python tuple containing strings of the row data for the current row

Returns The row data

Return type `tuple`

row_next ()

Start retrieval of the next row

Returns `None`

last_insert_id ()

Returns the last insert ID from the previous query

Returns The insert ID

Return type long

affected_rows ()

Returns the number of affected rows from the previous query

Returns The number of affected rows

Return type long

warning_count ()

Returns the number of warnings generated by the last query

Returns The warning count

Return type long

info ()

Returns the info from the last query

Returns The query info

Return type str

row_count ()

Returns the row count for a buffered query

Returns The row count

Return type long

next_result ()

Start retrieval of the next result set in a multi statement / result query

Returns `RETURN_PROCESSING` for more results, `RETURN_EOF` for no more results

Return type int

buffer_rows ()

Enable row buffering for connection

Returns `True` on success

Return type bool

buffer_row_get ()

Return the next row from a buffered result set in a similar way to `row_get ()`

Returns The row data

Return type tuple

row_get_offset (offset)

Return a specific row from a buffered result set in a similar way to `row_get ()`

Parameters *offset (long)* – The offset row

Returns The row data

Return type tuple

2.5 Prepared Statement Class

`class attachsql.statement`

execute()

Start execution a prepared statement

Returns True on success

Return type boolean

reset()

Reset a prepared statement

Returns True on success

Return type boolean

send_long_data(*param_no*, *data*)

Send a long data packet as a server parameter

Parameters

- **param_no** (*int*) – The parameter to set (starting from 0)
- **data** (*str*) – The data to set

Returns True on success

Return type boolean

param_count()

The number of parameters in the query to be set

Returns The number of parameters

Return type int

set_int(*param_no*, *data*[, *is_unsigned*])

Sets a parameter as an integer

Parameters

- **param_no** (*int*) – The parameter to set (starting from 0)
- **data** (*int*) – The data to set
- **is_unsigned** (*boolean*) – Whether or not the data is unsigned (False by default)

Returns True on success

Return type boolean

set_bigint(*param_no*, *data*[, *is_unsigned*])

Sets a parameter as a bigint

Parameters

- **param_no** (*int*) – The parameter to set (starting from 0)
- **data** (*longlong*) – The data to set
- **is_unsigned** (*boolean*) – Whether or not the data is unsigned (False by default)

Returns True on success

Return type boolean

set_float (*param_no*, *data*)

Sets a double precision floating point number

Parameters

- **param_no** (*int*) – The parameter to set (starting from 0)
- **data** (*float*) – The data to set

Returns True on success

Return type boolean

set_string (*param_no*, *data*)

Sets a string parameter

Parameters

- **param_no** (*int*) – The parameter to set (starting from 0)
- **data** (*str*) – The data to set

Returns True on success

Return type boolean

set_null (*param_no*)

Sets a parameter to NULL

Parameters **param_no** (*int*) – The parameter to set (starting from 0)

Returns True on success

Return type boolean

set_datetime (*param_no*, *data*)

Sets a parameter to a datetime.datetime or datetime.date object

Parameters

- **param_no** (*int*) – The parameter to set (starting from 0)
- **data** (*datetime*) – The date / datetime to set

Returns True on success

Return type boolean

set_time (*param_no*, *data*)

Sets a parameter to a datetime.time object

Parameters

- **param_no** (*int*) – The parameter to set (starting from 0)
- **data** (*time*) – The time to set

Returns True on success

Return type boolean

row_get ()

Retrieve a ready row from the buffer

Returns True on success

Return type boolean

get_int (*column_no*[, *get_unsigned*])

Get an int from the row

Parameters

- **column_no** (*int*) – The column number to get (starting from 0)
- **get_unsigned** (*boolean*) – Whether or not to get the data as unsigned (False by default)

Returns The integer data

Return type long

get_bigint (*column_no*[, *get_unsigned*])

Get a bigint from the row

Parameters

- **column_no** (*int*) – The column number to get (starting from 0)
- **get_unsigned** (*boolean*) – Whether or not to get the data as unsigned (False by default)

Returns The bigint data

Return type longlong

get_float (*column_no*)

Get a float from the row

Parameters **column_no** (*int*) – The column number to get (starting from 0)

Returns The float data

Return type float

get_char (*column_no*)

Get a character string from the row

Parameters **column_no** (*int*) – The column number to get (starting from 0)

Returns The char data

Return type str

get_column_type (*column_no*)

Get the native column type for a column, types are to be compared with *Column Type Constants*

Parameters **column_no** (*int*) – The column number to get (starting from 0)

Returns The column type

Return type int

row_next ()

Start retrieving the next row of the results

Returns None

get_column_count ()

Get the number of columns in the results

Returns The number of columns

Return type int

DB API Documentation

3.1 DB API Module Basics

3.1.1 Exceptions

exception `attachdb.Warning`

A warning exception, a subclass of `StandardError`

Note: currently unused

exception `attachdb.Error`

A base error exception, a subclass of `StandardError`

exception `attachdb.InterfaceError`

An error in the interface, a subclass of `Error`

Note: currently unused

exception `attachdb.DatabaseError`

A base exception for errors in the database, a subclass of `Error`

exception `attachdb.DataError`

An error in the processed data, a subclass of `DatabaseError`

Note: currently unused

exception `attachdb.OperationalError`

An error in the database server, a subclass of `DatabaseError`

exception `attachdb.IntegrityError`

An error in foreign key, a subclass of `DatabaseError`

Note: currently unused

exception `attachdb.InternalError`

An internal error in the database server, a subclass of `DatabaseError`

Note: currently unused

exception `attachdb.ProgrammingError`

An error in the client side application, a subclass of `DatabaseError`

exception `attachdb.NotSupportedError`

Feature not supported error, a subclass of `DatabaseError`

Note: currently unused

3.1.2 Functions

`attachdb.connection` (*host*, *port*=3306, *user*='', *password*='', *database*='', *autocommit*=False)

An alias to create an instance of `Connection`

3.2 DB API Connection Class

class `attachdb.Connection` (*host*, *port*=3306, *user*='', *password*='', *database*='', *autocommit*=False)

Parameters

- **host** (*str*) – The hostname to the server or path to Unix Domain Socket
- **port** (*int*) – The port number for the server or 0 for Unix Domain Socket
- **user** (*str*) – The username for the server
- **password** (*str*) – The password for the server
- **database** (*str*) – The default database for the connection
- **autocommit** (*bool*) – Enable/disable autocommit (None for server default)

autocommit (*setting*)

Enables/disables autocommit for the connection

Parameters *setting* (*bool*) – True to enable, False to disable

commit ()

Commit a transaction

rollback ()

Roll back a transaction

close ()

Close the connection

cursor ()

Create a new cursor for the connection to execute queries

Returns The new cursor

Return type Cursor

3.3 DB API Cursor Class

class `attachdb.Cursor`

arraysize

An integer representing the number of rows to be returned by `fetchmany()` if not specified by the function call. Defaults to 1

callproc (*name*, *parameters=None*)

Call a stored procedure

Parameters

- **name** (*str*) – the name of the stored procedure
- **parameters** (*tuple*) – parameters for the stored procedure

close ()

Close the cursor. This will read through to the end of the result set first if there is one active

execute (*query*, *parameters=[]*)

Execute a query with optional prepared statement style parameters. Parameter markers should be question marks (?).

Parameters

- **query** (*str*) – the query to execute
- **parameters** (*list*) – parameters for the query

executemany (*operation*, *sequence*)

Execute a query multiple times with each of a sequences of parameters. For use with INSERT or UPDATE queries.

Parameters

- **operation** (*str*) – the query to execute
- **sequence** (*list*) – a list containing lists of parameters

fetchone ()

Fetch the next result from the result set. Returns `None` if there are now rows left.

Returns The next row in the result set (or `None`)

Return type tuple

fetchmany (*size=None*)

Fetch many rows from a result set. The maximum amount to return is set by the `size` parameter or `arraysize` if this is not set.

Parameters **size** (*int*) – the amount of rows to return

Returns a list of row tuples

Return type list

fetchall ()

Fetch all the rows in a result set.

Returns a list of row tuples

Return type list

nextset ()

Start receiving the next result set in a multiple result set query

Returns `True` if successful, `None` if there is no more result sets

Return type boolean

setinputsizes (*sizes*)

Not implemented

setoutputsizes (*size*, *column=None*)

Not implemented

Indices and tables

- *genindex*
- *modindex*
- *search*

a

`attachdb`, [19](#)

`attachsql`, [16](#)

A

affected_rows() (attachsql.query method), 15
arraysize (attachdb.Cursor attribute), 20
attachdb (module), 19, 20
attachsql (module), 9, 12, 14, 16
autocommit() (attachdb.Connection method), 20

B

buffer_row_get() (attachsql.query method), 15
buffer_rows() (attachsql.query method), 15

C

callproc() (attachdb.Cursor method), 21
ClientError, 9
close() (attachdb.Connection method), 20
close() (attachdb.Cursor method), 21
column_count() (attachsql.query method), 14
COLUMN_TYPE_BIT (in module attachsql), 11
COLUMN_TYPE_BLOB (in module attachsql), 11
COLUMN_TYPE_DATE (in module attachsql), 11
COLUMN_TYPE_DATETIME (in module attachsql), 11
COLUMN_TYPE_DECIMAL (in module attachsql), 10
COLUMN_TYPE_DOUBLE (in module attachsql), 10
COLUMN_TYPE_ENUM (in module attachsql), 11
COLUMN_TYPE_FLOAT (in module attachsql), 10
COLUMN_TYPE_GEOMETRY (in module attachsql), 11
COLUMN_TYPE_INT24 (in module attachsql), 11
COLUMN_TYPE_LONG (in module attachsql), 10
COLUMN_TYPE_LONG_BLOB (in module attachsql), 11
COLUMN_TYPE_LONGLONG (in module attachsql), 10
COLUMN_TYPE_MEDIUM_BLOB (in module attachsql), 11
COLUMN_TYPE_NEWDECIMAL (in module attachsql), 11
COLUMN_TYPE_NULL (in module attachsql), 10
COLUMN_TYPE_SET (in module attachsql), 11
COLUMN_TYPE_SHORT (in module attachsql), 10

COLUMN_TYPE_STRING (in module attachsql), 11
COLUMN_TYPE_TIME (in module attachsql), 11
COLUMN_TYPE_TIMESTAMP (in module attachsql), 10
COLUMN_TYPE_TINY (in module attachsql), 10
COLUMN_TYPE_TINY_BLOB (in module attachsql), 11
COLUMN_TYPE_VARCHAR (in module attachsql), 11
COLUMN_TYPE_VARSTRING (in module attachsql), 11
COLUMN_TYPE_YEAR (in module attachsql), 11
commit() (attachdb.Connection method), 20
connect() (attachsql.connection method), 12
connect() (in module attachsql), 11
Connection (class in attachdb), 20
connection (class in attachsql), 12
connection() (in module attachdb), 20
connection_id() (attachsql.connection method), 12
create_connection() (attachsql.group method), 14
Cursor (class in attachdb), 20
cursor() (attachdb.Connection method), 20

D

DatabaseError, 19
DataError, 19

E

Error, 19
ESCAPE_TYPE_BIGINT (in module attachsql), 10
ESCAPE_TYPE_CHAR (in module attachsql), 9
ESCAPE_TYPE_CHAR_LIKE (in module attachsql), 9
ESCAPE_TYPE_DOUBLE (in module attachsql), 10
ESCAPE_TYPE_FLOAT (in module attachsql), 10
ESCAPE_TYPE_INT (in module attachsql), 10
ESCAPE_TYPE_NONE (in module attachsql), 9
EVENT_CONNECTED (in module attachsql), 11
EVENT_EOF (in module attachsql), 11
EVENT_ERROR (in module attachsql), 11
EVENT_ROW_READY (in module attachsql), 11
execute() (attachdb.Cursor method), 21
execute() (attachsql.statement method), 16

executemany() (attachdb.Cursor method), 21

F

fetchall() (attachdb.Cursor method), 21
 fetchmany() (attachdb.Cursor method), 21
 fetchone() (attachdb.Cursor method), 21

G

get_bigint() (attachsql.statement method), 18
 get_char() (attachsql.statement method), 18
 get_column_count() (attachsql.statement method), 18
 get_column_type() (attachsql.statement method), 18
 get_float() (attachsql.statement method), 17
 get_int() (attachsql.statement method), 17
 get_library_version() (in module attachsql), 11
 get_server_version() (attachsql.connection method), 12
 group (class in attachsql), 14

I

info() (attachsql.query method), 15
 IntegrityError, 19
 InterfaceError, 19
 InternalError, 19

L

last_insert_id() (attachsql.query method), 14

N

next_result() (attachsql.query method), 15
 nextset() (attachdb.Cursor method), 21
 NotSupportedError, 20

O

OperationalError, 19
 OPTION_COMPRESS (in module attachsql), 10
 OPTION_FOUND_ROWS (in module attachsql), 10
 OPTION_IGNORE_SIGPIPE (in module attachsql), 10
 OPTION_INTERACTIVE (in module attachsql), 10
 OPTION_LOCAL_FILES (in module attachsql), 10
 OPTION_MULTI_STATEMENTS (in module attachsql), 10
 OPTION_NO_SCHEMA (in module attachsql), 10
 OPTION_SEMI_BLOCKING (in module attachsql), 10
 OPTION_SSL_NO_VERIFY (in module attachsql), 10

P

param_count() (attachsql.statement method), 16
 poll() (attachsql.connection method), 12
 prepare_statement() (attachsql.connection method), 13
 ProgrammingError, 19

Q

query (class in attachsql), 14

query() (attachsql.connection method), 12

R

reset() (attachsql.statement method), 16
 RETURN_EOF (in module attachsql), 9
 RETURN_ERROR (in module attachsql), 9
 RETURN_NONE (in module attachsql), 9
 RETURN_NOT_CONNECTED (in module attachsql), 9
 RETURN_PROCESSING (in module attachsql), 9
 RETURN_ROW_DATA (in module attachsql), 9
 rollback() (attachdb.Connection method), 20
 row_count() (attachsql.query method), 15
 row_get() (attachsql.query method), 14
 row_get() (attachsql.statement method), 17
 row_get_offset() (attachsql.query method), 15
 row_next() (attachsql.query method), 14
 row_next() (attachsql.statement method), 18
 run() (attachsql.group method), 14

S

send_long_data() (attachsql.statement method), 16
 ServerError, 9
 set_bigint() (attachsql.statement method), 16
 set_datetime() (attachsql.statement method), 17
 set_float() (attachsql.statement method), 16
 set_int() (attachsql.statement method), 16
 set_null() (attachsql.statement method), 17
 set_option() (attachsql.connection method), 13
 set_ssl() (attachsql.connection method), 13
 set_string() (attachsql.statement method), 17
 set_time() (attachsql.statement method), 17
 setinputsizes() (attachdb.Cursor method), 21
 setoutputsizes() (attachdb.Cursor method), 22
 statement (class in attachsql), 16

W

Warning, 19
 warning_count() (attachsql.query method), 15