MySQL/Python API Documentation

**Summary:**

* The modules, class, and methods describes below were used for communication between Python and MySQL.

**Modules**:

* *sys*

Description: This module provides access to some variables used or maintained by the interpreter and to functions that interact strongly with the interpreter. It is always available.

* *mysql*

Description: This method sets up a connection, establishing a session with the MySQL server.

**Class:**

* *mysql.connection()*

Description: Handles the connection to a PostgreSQL database instance. It encapsulates a database session. Connections are created using the factory function connect(). Connections are thread safe and can be shared among many threads. See Thread and process safety for details.

* *mysql.connection.cursor()*

Description: Allows Python code to execute PostgreSQL command in a database session. Cursors are created by the connection.cursor() method: they are bound to the connection for the entire lifetime and all the commands are executed in the context of the database session wrapped by the connection. Cursors are not thread safe: a multithread application can create many cursors from the same connection and should use each cursor from a single thread. See Thread and process safety for details. Cursors created from the same connection are not isolated, i.e., any changes done to the database by a cursor are immediately visible by the other cursors. Cursors created from different connections can or can’t be isolated, depending on the connections’ isolation level.

**Methods:**

* *sys.exit([arg])*

Exit from Python. This is implemented by raising the SystemExit exception, so cleanup actions specified by finally clauses of try statements are honored, and it is possible to intercept the exit attempt at an outer level.Since exit() ultimately “only” raises an exception, it will only exit the process when called from the main thread, and the exception is not intercepted.

* *mysql.connector.connect()*

A connection with the MySQL server can be established using either the mysql.connector.connect() method or the mysql.connector.MySQLConnection() class. For more information go online to MySQL Connector/Python Developer Guide, see Section 10.2, “connection.MySQLConnection Class”.

* *connection.cursor*(name=None, cursor\_factory=None, scrollable=None, withhold=False)

Return a new cursor object using the connection.If name is specified, the returned cursor will be a server side cursor (also known as named cursor). Otherwise it will be a regular client side cursor. By default a named cursor is declared without SCROLL option and WITHOUT HOLD: set the argument or property scrollable to True/False and or withhold to True to change the declaration

* *connection.commit()*

Commit any pending transaction to the database. By default, Psycopg opens a transaction before executing the first command: if commit() is not called, the effect of any data manipulation will be lost. The connection can be also set in “autocommit” mode: no transaction is automatically open, commands have immediate effect. See Transactions control for details.

* *connection.close()*

Close the connection now (rather than whenever del is executed). The connection will be unusable from this point forward; an InterfaceError will be raised if any operation is attempted with the connection. The same applies to all cursor objects trying to use the connection. Note that closing a connection without committing the changes first will cause any pending change to be discarded as if a ROLLBACK was performed (unless a different isolation level has been selected: see set\_isolation\_level()).

* *cursor.execute(query, vars=None)*

Execute a database operation (query or command). Parameters may be provided as sequence or mapping and will be bound to variables in the operation. Variables are specified either with positional (%s) or named (%(name)s) placeholders. See Passing parameters to SQL queries. The method returns None. If a query was executed, the returned values can be retrieved using fetch\*() methods.

* *insertData()*

Inserts data using the cursor.execute() method. Can change the insert command for any type of insert into the database. If inserted data is not inserted into database then will throw an error to user and will then exit the program.

* *connectDatabase(servername, username, password, database)*

Connects to the database with the mysql.connect(). If the connection is unsuccessful then the user will be thrown an error message and the program will exit.

**Additional information:**

-For connection, cursor classes and methods: <http://initd.org/psycopg/docs/cursor.html>

-For mysql.connector.connect() Method: <https://dev.mysql.com/doc/connector-python/en/connector-python-api-mysql-connector-connect.html>