Подключение к SQL-базам в Python

Во-первых, для этого пособия мы создаем новую базу данных с именем ****python\_mysql****. Чтобы создать новую базу данных, вы можете запустить ****MySQL Workbench**** или клиентский инструмент ****MySQL**** и использовать оператор ****CREATE DATABASE**** следующим образом:

CREATE DATABASE python\_mysql;

Во-вторых, вам нужно загрузить данные в базу данных **python\_mysql.sql** из файла **python\_mysql.sql**.

**Подключение к базе данных MySQL с помощью функции connect()**

Давайте рассмотрим следующий модуль ****Python (python\_mysql\_connect1.py)****:

import mysql.connector

from mysql.connector import Error

def connect():

""" Connect to MySQL database """

try:

conn = mysql.connector.connect(host='localhost',

database='python\_mysql',

user='root',

password='secret')

if conn.is\_connected():

print('Connected to MySQL database')

except Error as e:

print(e)

finally:

conn.close()

if \_\_name\_\_ == '\_\_main\_\_':

connect()

### Подключение к базе данных MySQL с помощью объекта MySQLConnection

Мы можем создать новый модуль с именем ****python\_mysql\_dbconfig.py****, который считывает конфигурацию базы данных из файла ****config.ini**** и возвращает словарь следующим образом:

from configparser import ConfigParser

def read\_db\_config(filename='config.ini', section='mysql'):

""" Read database configuration file and return a dictionary object

:param filename: name of the configuration file

:param section: section of database configuration

:return: a dictionary of database parameters

"""

# create parser and read ini configuration file

parser = ConfigParser()

parser.read(filename)

# get section, default to mysql

db = {}

if parser.has\_section(section):

items = parser.items(section)

for item in items:

db[item[0]] = item[1]

else:

raise Exception('{0} not found in the {1} file'.format(section, filename))

return db

Обратите внимание, что мы использовали пакет ****ConfigureParser****, чтобы считать файл конфигурации.

Давайте проверим этот модуль в ****REPL****:

>>> from python\_mysql\_dbconfig import read\_db\_config

>>> read\_db\_config()

{'password': '', 'host': 'localhost', 'user': 'root', 'database': 'python\_mysql'}

Он работает, как ожидалось.

Теперь мы можем создать новый модуль ****python\_mysql\_connect2.py****, который использует объект****MySQLConnection**** для подключения к базе данных ****python\_mysql****:

from mysql.connector import MySQLConnection, Error

from python\_mysql\_dbconfig import read\_db\_config

def connect():

""" Connect to MySQL database """

db\_config = read\_db\_config()

try:

print('Connecting to MySQL database...')

conn = MySQLConnection(\*\*db\_config)

if conn.is\_connected():

print('connection established.')

else:

print('connection failed.')

except Error as error:

print(error)

finally:

conn.close()

print('Connection closed.')

if \_\_name\_\_ == '\_\_main\_\_':

connect()