



UFCD 10794 - Programação avançada em Python

Notebook 11 - CRUD em Bases de dados MySQL - Python

Sílvia Martins

AEFHP 2022

Conexão a uma base de dados MySQL com pymysql

PyMySQL é uma interface para estabelecer uma ligação a um servidor de base de dados MySQL do Python.

métodos da classe Connection :

- `connect(parameters...)` - construtor para a criação da ligação à base de dados

Um `cursor` seleciona um conjunto de dados, busca um registo de cada vez desse conjunto, e modifica o registo atual. Ao terminar a ação requerida no registo, o próximo registo é buscado para ser processado.

Basicamente, um cursor é um ponteiro para uma linha num conjunto de resultados ou uma tabela.

- `cursor()` - Retorna um novo Objeto Cursor usando a ligação à base de dados
- Os cursores são objetos que permitem a interação com bases de dados:
- `class pymysql.cursors.Cursor(connection)`
- `Cursor.next()` devolve a linha seguinte

Instalação das bibliotecas necessárias

In [1]:

```
pip install pymysql
```

Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pymysql in c:\users\sílvia martins\appdata\roaming\python\python39\site-packages (1.0.2)
Note: you may need to restart the kernel to use updated packages.

In [7]:

```
import pymysql.cursors
```

CONEXÃO

In [2]:

```
conexao = pymysql.connect(
    host='127.0.0.1',
    user='root',
    password='',
    db='clientes',
    charset='utf8mb4',
    cursorclass=pymysql.cursors.DictCursor # as consultas terão a forma de um dicionário
)
#cursor = conexao.cursor()

#cursor.close() #fechar
#conexao.close() # fechar a conexão
```

Outros métodos da classe Connection

Função execute

```
execute(query, args=None)
```

Parâmetros:

- query (str) – Query para execução.
- args (tuple, list or dict) – Parametros usados com a query. (opcional)
executemany(query, seq_de_args) , executa vários linhas numa consulta . Este método melhora o desempenho em INSERT e REPLACE de várias linhas. Caso contrário, é equivalente a repetir args com execute(). Método, que busca as linhas da última instrução executada:
- fetchall() , todas as linhas
- fetchone() , a próxima linha
- fetchmany(size=None) , um n.º definido de linhas
- read_next() - método para ler a linha seguinte

commit() - Confirma qualquer transação pendente na base de dados.

LISTAR DADOS

In [3]:

```
with conexao.cursor() as cursor:
    cursor.execute('SELECT * FROM clientes') # para executar o SQL
    result=cursor.fetchall() #para devolver todos os registos
    for i in result:
        print(i) #para imprimir o resultado da query
    for registo in result:
        print(registo['nome'], registo['apelido']) # acesso aos campos nome e apelido da tabela clientes
conexao.close()
```

```
{'id': 1, 'nome': 'Luís', 'apelido': 'Otávio', 'idade': 20, 'peso': 80.0}
{'id': 2, 'nome': 'Maria', 'apelido': 'Inês', 'idade': 50, 'peso': 57.0}
{'id': 3, 'nome': 'Joana', 'apelido': 'Silva', 'idade': 32, 'peso': 62.0}
{'id': 4, 'nome': 'Roberto', 'apelido': 'Oliveira', 'idade': 27, 'peso': 87.0}
{'id': 5, 'nome': 'Fabrício', 'apelido': 'Felix', 'idade': 35, 'peso': 70.0}
Luís Otávio
Maria Inês
Joana Silva
Roberto Oliveira
Fabrício Felix
```

INSERIR UM REGISTO

In []:

```
with conecta() as conexao:
    with conexao.cursor() as cursor:
        sql = 'INSERT INTO clientes (nome, apelido, idade, peso) VALUES ' \
            '(%s, %s, %s, %s)'
        cursor.execute(sql, ('Jack', 'Monroe', 112, 220))
        conexao.commit()
```

INSERIR VÁRIOS REGISTOS

In []:

```
with conecta() as conexao:
    with conexao.cursor() as cursor:
        sql = 'INSERT INTO clientes (nome, sobrenome, idade, peso) VALUES ' \
            '(%s, %s, %s, %s)'

        dados = [
            ('MURIEL', 'FIGUEIREDO', 19, 55),
            ('ROSE', 'FIGUEIREDO', 19, 55),
            ('JOSE', 'FIGUEIREDO', 19, 55),
        ]

        cursor.executemany(sql, dados)
        conexao.commit()
```

ELIMINAR UM REGISTO

In []:

```
with conecta() as conexao:
    with conexao.cursor() as cursor:
        sql = 'DELETE FROM clientes WHERE id = %s'
        cursor.execute(sql, (6,))
        conexao.commit()
```

ELIMINAR REGISTOS

In []:

```
with conecta() as conexao:
    with conexao.cursor() as cursor:
        sql = 'DELETE FROM clientes WHERE id IN (%s, %s, %s)'
        cursor.execute(sql, (7, 8, 9))
        conexao.commit()
```

ELIMINAR REGISTOS ENTRE INTERVALO DE VALORES

In []:

```
with conecta() as conexao:
    with conexao.cursor() as cursor:
        sql = 'DELETE FROM clientes WHERE id BETWEEN %s AND %s'
        cursor.execute(sql, (10, 12))
        conexao.commit()
```

ATUALIZAR UM REGISTO

In []:

```
with conecta() as conexao:
    with conexao.cursor() as cursor:
        sql = 'UPDATE clientes SET nome=%s WHERE id=%s'
        cursor.execute(sql, ('JOANA', 5))
        conexao.commit()
```

Conexão a uma base de dados MySQL com MySQLConnector

MySQLConnector é uma interface/driver para estabelecer uma ligação a um servidor de base de dados MySQL do Python.

Instalação das bibliotecas necessárias

In []:

```
pip install mysql-connector-p
```

In []:

```
import mysql.connector
```

CONEXÃO

In []:

```
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    password="yourpassword",
    database="mydatabase"
)
```

LISTAR DADOS

In []:

```
mycursor = mydb.cursor()

mycursor.execute("SELECT * FROM customers")

myresult = mycursor.fetchall()

for x in myresult:
    print(x)

mydb.close()
```

(1, 'John', 'Highway 21') (2, 'Peter', 'Lowstreet 27') (3, 'Amy', 'Apple st 652') (4, 'Hannah', 'Mountain 21') (5, 'Michael', 'Valley 345') (6, 'Sandy', 'Ocean blvd 2')

INSERIR UM REGISTO

In []:

```
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    password="yourpassword",
    database="mydatabase"
)

mycursor = mydb.cursor()

sql = "INSERT INTO customers (name, address) VALUES (%s, %s)"
val = ("John", "Highway 21")
mycursor.execute(sql, val)

mydb.commit()

print(mycursor.rowcount, "record inserted.")
mydb.close()
```

INSERIR VÁRIOS REGISTOS, ELIMINAR E ALTERAR REGISTOS É EFETUADO DE MODO SEMELHANTE AOS EXEMPLOS ANTERIORES

Bibliografia

<https://pymysql.readthedocs.io/en/latest/modules/cursors.html> (<https://pymysql.readthedocs.io/en/latest/modules/cursors.html>)
<https://stackoverflow.com/questions/231767/what-does-the-yield-keyword-do> (<https://stackoverflow.com/questions/231767/what-does-the-yield-keyword-do>)
Vasconcelos, J. (2015). Python - Algoritmia e Programação Web. Lisboa:FCA
<https://www.udemy.com/course/python-3-do-zero-ao-avancado> (<https://www.udemy.com/course/python-3-do-zero-ao-avancado>)
https://www.w3schools.com/python/python_mysql_getstarted.asp (https://www.w3schools.com/python/python_mysql_getstarted.asp)
<https://peps.python.org/pep-0249/#commit> (<https://peps.python.org/pep-0249/#commit>)

