

CRUD em MySQL no Python

Importa a base de dados Clientes.sql para o MySQL (Usa o wampServer).

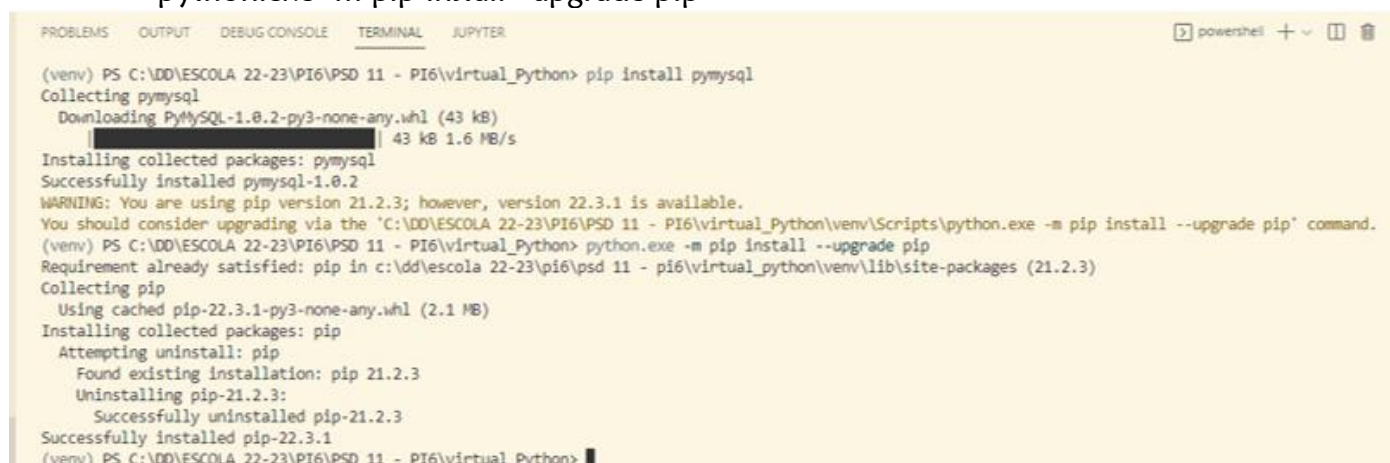
Inicia a Anaconda e depois a IDE VS Code ou Spyder.

1. Abre a pasta onde criaste um ambiente virtual, ou cria um novo.

- Ativa o teu ambiente virtual:
 - No PowerShell do Windows: `.\nome_amb_virtual\Scripts\Activate.ps1` ;
 - No cmd: `.\nome_amb_virtual\Scripts\activate.bat` ;
- Instala a biblioteca pymysql:


```
>> pip install pymysql
```
- Aproveita para fazer um upgrade do pip


```
>> python.exe -m pip install --upgrade pip
```



```
(venv) PS C:\DD\ESCOLA 22-23\PI6\PSD 11 - PI6\virtual_Python> pip install pymysql
Collecting pymysql
  Downloading PyMySQL-1.0.2-py3-none-any.whl (43 kB)
    | 43 kB 1.6 MB/s
Installing collected packages: pymysql
Successfully installed pymysql-1.0.2
WARNING: You are using pip version 21.2.3; however, version 22.3.1 is available.
You should consider upgrading via the 'C:\DD\ESCOLA 22-23\PI6\PSD 11 - PI6\virtual_Python\venv\Scripts\python.exe -m pip install --upgrade pip' command.
(venv) PS C:\DD\ESCOLA 22-23\PI6\PSD 11 - PI6\virtual_Python> python.exe -m pip install --upgrade pip
Requirement already satisfied: pip in c:\dd\escola 22-23\pi6\psd 11 - pi6\virtual_python\venv\lib\site-packages (21.2.3)
Collecting pip
  Using cached pip-22.3.1-py3-none-any.whl (2.1 MB)
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 21.2.3
    Uninstalling pip-21.2.3:
      Successfully uninstalled pip-21.2.3
  Successfully installed pip-22.3.1
(venv) PS C:\DD\ESCOLA 22-23\PI6\PSD 11 - PI6\virtual_Python>
```

- Cria um ficheiro com o nome 'F6Clientes1.py', dentro da pasta do teu ambiente virtual.
 - Reproduz o código de I [2], relativo à criação de uma ligação à base de dados Clientes.sql e impressão de dados.
 - Acrescenta código que te permita imprimir os dados dos 3 clientes mais pesados (ordena-os por ordem decrescente e usa a cláusula LIMIT). Não tre esqueças que estamos a trabalhar com um dicionário!
- Output esperado:

id	nome	apelido	idade	peso
4	Roberto	Oliveira	27	87.0
1	Luís	Otávio	20	80.0
5	Fabrizio	Felix	35	70.0

```
(venv) PS C:\DD\ESCOLA 22-23\PI6\PSD 11 - PI6\virtual_Python>
```

2. Cria um novo programa a partir do anterior com o nome F6Clientes2.py.

- Altera o programa usando o código seguinte, que nos permite usar a função *conecta()*, como um gerador de contexto, sempre que a função for chamada ela deverá devolver a conexão ao programa principal e será fechada de qualquer modo.

(*with* mm *as* nnn)

CÓDIGO (o resto será para manter, exceto o *close*):

```
import pymysql.cursors
from contextlib import contextmanager # funciona como um decorador

@contextmanager
def conecta():
    conexao = pymysql.connect(
        host='127.0.0.1',
        user='root',
        password='',
        db='clientes',
        charset='utf8mb4',
        cursorclass=pymysql.cursors.DictCursor
    )
    try:
        yield conexao
    finally:
        conexao.close()

with conecta() as conexao:
    with conexao.cursor() as cursor:
        cursor.execute('SELECT * FROM clientes order by peso DESC LIMIT...
```

NOTA: @ é que simboliza um decorador -> função que adiciona uma nova funcionalidade a uma função (a estudar posteriormente), neste caso para iteração.

Yield é um gerador* – usa-se em listas que têm valores muito grandes e que não queremos armazenar em memória. Enquanto que, numa função, usar **return** devolve uma lista completa e a armazena em memória, **yield** devolve um elemento de cada vez, sem necessidade de se guardar os valores em memória (a estudar posteriormente).

3. Inserção de dados:

Guarda o ficheiro anterior como *F6Clientes3.py*.

Acrescenta, antes do código correspondente à listagem dos dados da query, o seguinte código que irá acrescentar um registo à tabela clientes.

```
# INSERE UM REGISTO NA BASE DE DADOS
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()
```

4. Cria um novo programa com o nome F6Clientes4.py.

Tendo por base o código do notebook NB11. 10794 – CRUD em Base de dados MySQL, escreve as instruções necessárias para:

- Criar uma função para abrir uma conexão à base de dados Clientes;
- Criar funções para : inserir um registo, listar os registos, apagar um registo e atualizar um registo;
- Pedir os dados necessários ao utilizador;
- Criar um menu de opções para escolher o que se pretende executar na base de dados, através de definição de um dicionário.
- Repetir as instruções até que seja inserido uma opção que termina o ciclo.
- Deve ser limpo o terminal depois de executada cada opção.

Exemplo de saída:

<pre>=====MENU:===== 1. Listar 2. Inserir 3. Apagar 4. Alterar 5. Sair Insira a opção->1 id nome apelido idade peso 5 Fabrício Felix 35 99.0 3 Joana Silva 32 78.0 1 Luiz Otávio 20 100.0 2 Maria Inês 50 57.0 4 Roberto Oliveira 27 87.0 █</pre>	<pre>=====MENU:===== 1. Listar 2. Inserir 3. Apagar 4. Alterar 5. Sair Insira a opção->2 Nome->Ana Apelido->Martins Idade->45 Peso->50 █</pre>
<pre>=====MENU:===== 1. Listar 2. Inserir 3. Apagar 4. Alterar 5. Sair Insira a opção->4 N.º do id atualizar->3 Novo nome ->Lara █</pre>	<pre>1. Listar 2. Inserir 3. Apagar 4. Alterar 5. Sair Insira a opção->1 id nome apelido idade peso 38 Ana Martins 45 50.0 5 Fabrício Felix 35 99.0 3 Lara Silva 32 78.0 1 Luiz Otávio 20 100.0 2 Maria Inês 50 57.0 4 Roberto Oliveira 27 87.0 █</pre>
<pre>=====MENU:===== 1. Listar 2. Inserir 3. Apagar 4. Alterar 5. Sair Insira a opção->3 qual o nome a apagar->Lara █</pre>	

5. Conexão a uma base de dados usando o mysqlconector

Instala o conetor com:

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

ou

```
python -m pip install mysql-connector-python
```

5. Cria um programa com o nome F6clientes5.py com o conteúdo seguinte.

```
import mysql.connector
from mysql.connector import Error
# LISTAR DADOS

try:
    conn=mysql.connector.connect(host='localhost', database = 'clientes', user='root',password='')
    db_info = conn.get_server_info()
    print("ligado ao servidor MySQL versão ",db_info)
    cursor = conn.cursor()
    cursor.execute("select database();")
    linha = cursor.fetchone()
    print("Ligado à BD ",linha)

    print("Nº de registos lidos: ", cursor.rowcount,"\n")
    cursor = conn.cursor()
    sql = "select * from clientes"
    cursor.execute(sql)
    registo =cursor.fetchall()
    for linha in registo:
        print(linha)
    print(f"{'Id':<10}{'nome':<10}{'apelido':<10}{'idade':<10}{'peso':<8}\n")
    for registo in registo:
        print(f"{registo[0]:<10}{registo[1]:<10}{registo[2]:<10}{registo[3]:<10}{registo[4]:<8}")

except mysql.connector.Error as erro:
    print(f"Erro: {erro}")
finally:
    cursor.close()
    conn.close()
    print("Conexão ao MySQL encerrada!")
```

6. Tendo como base o código seguinte insere 2 registos na base de dados, sendo os dados fornecidos pelo utilizador (podes acrescentar o código no ficheiro criado anteriormente).

```
import mysql.connector
from mysql.connector import Error
# Inserir registos em um banco de dados MySQL
print("Rotina para cadastro de produtos no banco de dados")
print("Entre com os dados conforme solicitado")

idProd = input("ID do Produto: ")
nomeProd = input("Nome do Produto: ")
precoProd = input("Preço: ")
quantProd = input("Quantidade: ")

dados = idProd + ',' + nomeProd + ',' + precoProd + ',' + quantProd + ','
declaracao = """INSERT INTO tbl_produtos
(IdProduto, NomeProduto, Preço, Quantidade)
VALUES ("""
sql = declaracao + dados
```

*Exemplo de uma função que devolve uma lista(esq.) e de um gerador (yield) (dta.):

<pre>def lista (n): l=[] for i in range (n+1): l.append(i) return l print(lista(10))</pre>	<pre>def lista_it(n): for i in range(n+1): yield i for i in lista(10): print(i)</pre>
--	--

Bibliografia:

<https://horadecodar.com.br/2020/04/27/python-para-que-serve-o-yield-quando-utilizar/>
<https://snky.io/advisor/python/PyMySQL/functions/pymysql.err.ProgrammingError>