03_SQLite

March 13, 2020

1 Acesso a bases de dados SQLite com Python

Pedro Cardoso
(ISE/UAlg - pcardoso@ualg.pt)

1.1 Pre-requisitos

1.1.1 Conectores instalados

Verificar que todos os módulos necessários estão instalados, nomeadamente, pode de ter de instalar o `sqlite3` [https://docs.python.org/2/library/sqlite3.html]

(Dependendo do seu sistema poderá ter de substituir pip3 por pip, pip3.7, pip3.8, . . .)

```
In [1]: !pip install pysqlite3
```

```
Collecting pysqlite3
  Using cached https://files.pythonhosted.org/packages/06/1f/b806d2c35c9a587e11a4ec73a531416776
Building wheels for collected packages: pysqlite3
  Running setup.py bdist_wheel for pysqlite3 ... done
  Running setup.py clean for pysqlite3
Failed to build pysqlite3
Installing collected packages: pysqlite3
  Running setup.py install for pysqlite3 ... done
Successfully installed pysqlite3
```

1.2 Estabelecimento de conexão à base de dados usando um Connector/Python

Criar a conexão usando o método `connect`, que tem como parâmetro o caminho para o ficheiro que contém a base de dados

```
In [2]: import sqlite3

        # se não existir a base de dados (ficheiro) exemplo.db será criada
        conn = sqlite3.connect('exemplo.db')

        conn.close()
```

1.3 Criação de uma base de dados

Para a criação das tabelas e relacionamentos podemos construiro o sql ou, como alternativa, podemos usar ferramentas como sejam o MySQL Workbench, o Phpmyadmin, o SQlite Browser, o DataGrip, etc.

Consideremos o caso em que contruímos o sql…

```
In [3]: sql = '''
        create table Location
        (
            idLocation integer
                constraint Location_pk
                    primary key autoincrement,
            name TEXT not null,
            description text not null
        );


        create table Unit
        (
            unit text
                constraint Unit_pk
                    primary key,
            description text not null
        );

        create table Sensor
        (
            idSensor integer
                constraint Sensor_pk
                    primary key,
            idLocation integer not null
                constraint Sensor_Location_idLocation_fk
                    references Location
                        on update cascade on delete cascade,
            name text not null,
            unit text not null
                constraint Sensor_Unit_unit_fk
                    references Unit
                        on update cascade on delete cascade
        );

        create table Reading
        (
            idReading integer
                constraint Reading_pk
                    primary key,
            idSensor integer
```

```
                constraint Reading_Sensor_idSensor_fk
                    references Sensor
                        on update cascade on delete cascade,
            timestamp datetime default CURRENT_TIMESTAMP,
            value real not null
        );


        create table Alert
        (
            idAlert integer
                constraint Alert_pk
                    primary key,
            idSensor integer
                constraint Alert_Sensor_idSensor_fk
                    references Sensor
                        on update cascade on delete cascade,
            description text not null,
            cleared integer

        )
        '''
```

```
In [4]: conn = sqlite3.connect('sensors.db')

        cursor = conn.cursor()

        # executescript is a nonstandard convenience method for executing multiple SQL stateme
        cursor.executescript(sql)

Out[4]: <sqlite3.Cursor at 0x7f648448f650>
```

1.4 Operações CRUD

1.4.1 INSERT

Aberta a conexão em sqlite

```
In [5]: import sqlite3

        cnx = sqlite3.connect('sensors.db')
        cursor = cnx.cursor()
```

inserir uma nova localização na base de dados e obter o id correspondente

```
In [6]: # prepare the sql query for the new location
        sql = '''
        INSERT INTO location
            (name, description)
        VALUES
            (?, ?)
        '''

        data = ('Prometheus Server', 'Prometheus Server @ lab. 163 / ISE /UAlg')

        #execute the sql query and get the new location id
        cursor.execute(sql, data)
        location_id = cursor.lastrowid
        location_id
```

```
Out[6]: 1
```

Quando estamos a usar um sistema transacional, temos de efetuar o `commit` depois de fazer um `INSERT`, `DELETE`, ou `UPDATE`.

Note-se que: - o `commit` confirma a transação atual. Se não se chamar, tudo o que fez desde a última chamada do `commit()` não será visível às outras conexões. - quando a BD é acedida por várias conexões e um dos processos modifica-a, a BD SQLite fica bloqueada até que a transação seja confirmada (*commited*). - podemos desfazer as alterações desde o último `commit` chamando o método `rollback()`

```
In [7]: cnx.commit()
```

Inserir uma nova `Unit`

```
In [8]: sql = '''
        REPLACE INTO Unit
            (unit, description)
        VALUES
            ("percent", "percentage of usage")
        '''

        cursor.execute(sql)

        cnx.commit()
```

Inserir um novo sensor e obter o seu id: - preparar o sql, note-se os *placeholders* com nome usados neste caso - preparar os dados - executar o query

```
In [9]: sql = '''INSERT INTO `sensor` (`idLocation`, `name`, `unit`)
                VALUES (:idLocation, :name, :unit);'''

        data = {
                'idLocation': location_id,
                'name' : 'cpu_sensor_01',
                'unit' : 'percent'
               }

        cursor.execute(sql, data)
        sensor_id = cursor.lastrowid
        cnx.commit()
```

E agora, obter alguns dados e enviar para a base de dados

```
In [10]: import psutil

         sql = '''
         INSERT INTO `reading`
             (`idSensor`, `value`)
         VALUES
             (:idSensor, :value)
         '''

         for _ in range(20):
             data = {
                    'idSensor' : sensor_id,
                     'value' : psutil.cpu_percent(interval=1)
                    }
             cursor.execute(sql, data)
             cnx.commit()
             print('.', end='')
```

```
...
```

```
In [11]: cursor.close()
         cnx.close()
```

1.5 Selecionar dados

```
In [12]: import sqlite3

         cnx = sqlite3.connect('sensors.db')
         cursor = cnx.cursor()
```

```
In [13]: sql = '''
         SELECT idLocation, name, description
         FROM location
         WHERE description LIKE "%163%"'''

         cursor.execute(sql)

         for (idLocation, name, description) in cursor:
           print("id: {}\n\t name: {} \n\t description: {}".format(idLocation, name, descripti
```

```
id: 1
         name: Prometheus Server
         description: Prometheus Server @ lab. 163 / ISE /UAlg
```

```
In [14]: sql = '''
         SELECT idReading, idSensor, timestamp, value
```

```
FROM reading
WHERE value BETWEEN ? and ?
'''
data = (5, 50)

cursor.execute(sql, data)

for (idReading, idSensor, timestamp, value) in cursor:
  print("idReading: {}\n\t idSensor: {} \n\t time: {} \n\t value: {}".format(idReadin
```

```
idReading: 1
         idSensor: 1
         time: 2020-03-13 15:25:59
         value: 14.3
idReading: 2
         idSensor: 1
         time: 2020-03-13 15:26:00
         value: 20.4
idReading: 3
         idSensor: 1
         time: 2020-03-13 15:26:01
         value: 10.6
idReading: 4
         idSensor: 1
         time: 2020-03-13 15:26:03
         value: 16.6
idReading: 5
         idSensor: 1
         time: 2020-03-13 15:26:04
         value: 11.5
idReading: 6
         idSensor: 1
         time: 2020-03-13 15:26:05
         value: 18.1
idReading: 7
         idSensor: 1
         time: 2020-03-13 15:26:06
         value: 9.1
idReading: 8
         idSensor: 1
         time: 2020-03-13 15:26:07
         value: 20.2
idReading: 9
         idSensor: 1
         time: 2020-03-13 15:26:08
         value: 9.9
idReading: 10
         idSensor: 1
         time: 2020-03-13 15:26:09
         value: 9.2
idReading: 11
         idSensor: 1
         time: 2020-03-13 15:26:10
         value: 15.8
idReading: 12
         idSensor: 1
         time: 2020-03-13 15:26:11
         value: 9.3
idReading: 13
         idSensor: 1
         time: 2020-03-13 15:26:12
         value: 8.4
idReading: 14
         idSensor: 1
         time: 2020-03-13 15:26:13
         value: 14.8
idReading: 15
         idSensor: 1
         time: 2020-03-13 15:26:14
         value: 9.7
idReading: 16
         idSensor: 1
         time: 2020-03-13 15:26:15
         value: 17.2
idReading: 17
         idSensor: 1
         time: 2020-03-13 15:26:16
         value: 12.0
idReading: 18
         idSensor: 1
         time: 2020-03-13 15:26:17
         value: 11.6
idReading: 19
         idSensor: 1
         time: 2020-03-13 15:26:18
         value: 10.9
idReading: 20
         idSensor: 1
         time: 2020-03-13 15:26:19
         value: 9.8
```

```
In [15]: sql = '''
             select *
             from Location
                 inner join Sensor S on Location.idLocation = S.idLocation
                 inner join Unit U on S.unit = U.unit
                 inner join Reading R on S.idSensor = R.idSensor
             where value between :low and :high
             order by value
         '''

         data = {
             'low': 5,
             'high': 20
         }

         cursor.execute(sql, data)
```

```
Out[15]: <sqlite3.Cursor at 0x7f64843d3180>
```

Podemos obter os nomes das colunas

```
In [16]: cursor.description
```

```
Out[16]: (('idLocation', None, None, None, None, None, None),
          ('name', None, None, None, None, None, None),
          ('description', None, None, None, None, None, None),
          ('idSensor', None, None, None, None, None, None),
          ('idLocation', None, None, None, None, None, None),
          ('name', None, None, None, None, None, None),
          ('unit', None, None, None, None, None, None),
          ('unit', None, None, None, None, None, None),
          ('description', None, None, None, None, None, None),
          ('idReading', None, None, None, None, None, None),
          ('idSensor', None, None, None, None, None, None),
          ('timestamp', None, None, None, None, None, None),
          ('value', None, None, None, None, None, None))
```

```
In [17]: lista_de_colunas = [linha[0] for linha in cursor.description]
         lista_de_colunas
```

```
Out[17]: ['idLocation',
          'name',
          'description',
          'idSensor',
          'idLocation',
          'name',
          'unit',
          'unit',
          'description',
          'idReading',
          'idSensor',
          'timestamp',
          'value']
```

```
In [18]: for linha in cursor:
             print('\t'.join([f'|{coluna}: {valor}' for valor, coluna  in zip(linha, lista_de_
```

```
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
|idLocation: 1          |name: Prometheus Server        |description: Prometheus Server @ lab. 1
```

Usando o comando `fetchall` podemos obter todos os resultados de uma única vez como uma lista de tuplos

```
In [19]: # é necessario voltar a correr o select pois o cursor foi esvaziado
         cursor.execute(sql, data)

         cursor.fetchall()
```

```
Out[19]: [(1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           13,
           1,
           '2020-03-13 15:26:12',
           8.4),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           7,
           1,
           '2020-03-13 15:26:06',
           9.1),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           10,
           1,
           '2020-03-13 15:26:09',
           9.2),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           12,
           1,
           '2020-03-13 15:26:11',
           9.3),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           15,
           1,
           '2020-03-13 15:26:14',
           9.7),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           20,
           1,
           '2020-03-13 15:26:19',
           9.8),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           9,
           1,
           '2020-03-13 15:26:08',
           9.9),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           3,
           1,
           '2020-03-13 15:26:01',
           10.6),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           19,
           1,
           '2020-03-13 15:26:18',
           10.9),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           5,
           1,
           '2020-03-13 15:26:04',
           11.5),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           18,
           1,
           '2020-03-13 15:26:17',
           11.6),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           17,
           1,
           '2020-03-13 15:26:16',
           12.0),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           1,
           1,
           '2020-03-13 15:25:59',
           14.3),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           14,
           1,
           '2020-03-13 15:26:13',
           14.8),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           11,
           1,
           '2020-03-13 15:26:10',
           15.8),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           4,
           1,
           '2020-03-13 15:26:03',
           16.6),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           16,
           1,
           '2020-03-13 15:26:15',
           17.2),
          (1,
           'Prometheus Server',
           'Prometheus Server @ lab. 163 / ISE /UAlg',
           1,
           1,
           'cpu_sensor_01',
           'percent',
           'percent',
           'percentage of usage',
           6,
           1,
           '2020-03-13 15:26:05',
           18.1)]
```

Podemos também converter para um dicionário mas **nosso caso NÃO é boa ideia** pois duas colunas “têm o mesmo nome” (e.g., `nome`), pelo que se perdem colunas.

```
In [20]: # é necessario voltar a correr o select pois o cursor foi esvaziado
         cursor.execute(sql, data)

         for linha in cursor:
             print({coluna: valor for valor, coluna  in zip(linha, lista_de_colunas)})
```

```
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
```

```
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
{'idLocation': 1, 'name': 'cpu_sensor_01', 'description': 'percentage of usage', 'idSensor': 1
```

```
In [21]: cursor.close()
         cnx.close()
```

```
In [ ]:
```