Sqlite数据库操作

SQLite是一个软件库，实现了自给自足的、无服务器的、零配置的、事务性的SQL 数据库引擎。SQLite 是在世界上最广泛部署的SQL数据库引擎，可以直接在本地以一个文件的形式保存这个数据库。Python3已经自带了Sqlite数据库，可以直接使用。

代码整理：谢作如

In [1]:

```
#导入sqlite3数据库
import sqlite3 as sq3
```

Sqlite的常用数据类型

1）NULL 值是一个 NULL 值。 2）INTEGER 值是一个带符号的整数，根据值的大小存储在 1、2、3、4、6 或 8 字节中。 3）REAL 值是一个浮点值，存储为8字节的IEEE浮点数字。 4）TEXT 值是一个文本字符串，使用数据库编码（UTF-8、UTF-16BE 或 UTF-16LE）存储。 5）BLOB 值是一个blob数据，完全根据它的输入存储。

创建数据库和表

In [2]:

```
path  = r'./'
#在同一目录下，新建一个数据库文件
#connect 连接一个数据库
con =  sq3.connect(path+'data.db')
```

In [3]:

```
#创建一个表，并说明表内容的数据类型(列名 列类型,)
con.execute('CREATE TABLE numbs (Date date,No1 real,No2 real)')
#执行完都要提交
con.commit()
```

现在，请查看一下这个文件夹，是不是多了一个“data.db”文件？这就是新建的数据库文件。这个数据库有一个叫做“numbs”的表，字段分别为Date，No1和No2。其中Date的类型是“date”，其他都是“real”。

说明：ls是linux的列目录命令，类似windows中的dir。在命令请加上“！”，是告诉jupyter，这是linux命令。

In [4]:

```
!ls
```

```
data.db                    肺炎疫情数据的获取和呈现.ipynb    爬虫之requests
库.pdf
Sqlite数据库操作.ipynb     爬虫之requests库.ipynb
```

我们来查看一下numbs表的结构。

In [5]:

```
d=con.execute('PRAGMA table_info(numbs)')
d.fetchall()
```

Out[5]:

```
[(0, 'Date', 'date', 0, None, 0),
 (1, 'No1', 'real', 0, None, 0),
 (2, 'No2', 'real', 0, None, 0)]
```

写入数据

In [6]:

```
import datetime as dt
con.execute('INSERT INTO numbs VALUES(?,?,?)',(dt.datetime.now(),0.15,8.9))
```

Out[6]:

```
<sqlite3.Cursor at 0x7fac3658f0>
```

“INSERT INTO”是典型的sql语法，表示插入一条记录。下面用“select”语句读出。是不是看到了一条记录?

In [7]:

```
d=con.execute('select * from numbs')
d.fetchall()
```

Out[7]:

```
[('2020-02-16 22:12:00.875679', 0.15, 8.9)]
```

d.fetchall()得到的是一个元组，获得所有记录。如果有多条记录，用d.fetchone()得到第一条记录，再使用就得到第二条记录，以此类推。

In [8]:

```
#继续写入多条数据，用numpy生成随机数。
import numpy as np
data = np.random.standard_normal((1000,2)).round(5) #取1000行2列的标准正态分布的随机数
for row in data:
    con.execute('INSERT INTO numbs VALUES(?,?,?)',(dt.datetime.now(),row[0],row[1]))
con.commit()
```

查询语句

前面用con.execute('select * from numbs').fetchone()，得到了第一条记录，用fetchmany(10)表示得到前10条。

In [9]:

```
con.execute('SELECT * FROM numbs').fetchmany(10) #获取前10条
```

Out[9]:

```
[('2020-02-16 22:12:00.875679', 0.15, 8.9),
 ('2020-02-16 22:12:02.547587', 0.46266, -1.27232),
 ('2020-02-16 22:12:02.547914', 1.47302, -1.09923),
 ('2020-02-16 22:12:02.548213', -1.19066, 1.64815),
 ('2020-02-16 22:12:02.548460', -0.84103, 1.02713),
 ('2020-02-16 22:12:02.548641', -0.44841, 0.02644),
 ('2020-02-16 22:12:02.548858', -1.73114, -0.53738),
 ('2020-02-16 22:12:02.549059', -0.20276, 1.30281),
 ('2020-02-16 22:12:02.549339', -1.43802, 1.09714),
 ('2020-02-16 22:12:02.549522', 0.1495, -0.06341)]
```

In [10]:

```
# 可以用这样的方式，循环读出5条。
d = con.execute('SELECT * FROM numbs')
for i in range(5):
    print(d.fetchone())
```

```
('2020-02-16 22:12:00.875679', 0.15, 8.9)
('2020-02-16 22:12:02.547587', 0.46266, -1.27232)
('2020-02-16 22:12:02.547914', 1.47302, -1.09923)
('2020-02-16 22:12:02.548213', -1.19066, 1.64815)
('2020-02-16 22:12:02.548460', -0.84103, 1.02713)
```

修改记录

修改之前插入的第一条记录，即No1=0.15，No2=8.9的那一条。

In [11]:

```
sql = 'update numbs set No1=2.5 where No2=8.9'
con.execute(sql)
```

Out[11]:

```
<sqlite3.Cursor at 0x7fa443ec00>
```

读出来看一下，是不是已经修改?

In [12]:

```
d=con.execute('select * from numbs where No2=8.9')
d.fetchone()
```

Out[12]:

```
('2020-02-16 22:12:00.875679', 2.5, 8.9)
```

也许sql语句采用这样的写法，看起来会更加舒服一些。

In [13]:

```
sql = 'update numbs set No1=? where No2=?'
con.execute(sql,(3.0,8.9))
```

Out[13]:

```
<sqlite3.Cursor at 0x7fa443edc0>
```

删除记录

In [14]:

```
sql = 'delete from numbs where No2=8.9'
con.execute(sql)
```

Out[14]:

```
<sqlite3.Cursor at 0x7fa443ef10>
```

还是要读出来看一下，是不是已经删除了。

In [15]:

```
d=con.execute('select * from numbs')
d.fetchone()
```

Out[15]:

```
('2020-02-16 22:12:02.547587', 0.46266, -1.27232)
```

In [16]:

```
# 最后要关闭数据库连接con
con.close()
```

使用游标对象来操作

使用con.cursor()获取游标对象，查询数据库的效率会更高。下面提供的是完整代码。

In [17]:

```
import sqlite3 as sq3
path  = r'./'
con =  sq3.connect(path+'data.db')
# 创建一个名为student的表
con.execute("CREATE TABLE student (pname TEXT,age INTEGER)")
con.commit()
# 获取cursor对象
cur = con.cursor()
sql = 'insert into student(pname,age) values(?,?)'
try:
    cur.execute(sql,('张三',23))
    con.commit()
    print('插入成功')
except Exception as e:
    print(e)
    print('插入失败')
    con.rollback()
finally:
    # 关闭游标
    cur.close()
    # 关闭连接
    con.close()
```

插入成功

In [18]:

```
import sqlite3 as sq3
path  = r'./'
con =  sq3.connect(path+'data.db')
con.commit()
cur = con.cursor()
sql = 'select * from student'
try:
    cur.execute(sql)
    # 获取所有数据
    person_all = cur.fetchall()
    # print(person_all)
    # 遍历
    for p in person_all:
        print(p)
except Exception as e:
    print(e)
    print('查询失败')
finally:
    # 关闭游标
    cur.close()
    # 关闭连接
    con.close()
```

('张三', 23)

In [ ]:

```

```