ماهى PySQLite؟
هى interface ل SQLite من خلال ال Python

للتحميل اضغط هنا

للتستيب مثل اى Lib
python setup.py install

لنبدأ
1- هنحتاج نعمل import لل module
PySQLite هتعملها import ك pysqlite2 ولكن ال lib دى بردو اللى يهمنا فيها هو sub-module بإسم dbapi2

Writing

2- هنحتاج نعمل Connection مع DB تمام ؟ ال db نفسها عبارة عن file عادى جدا -فى حال عدم وجوده هيتم إنشاء file جديد- فلعمل ال Connection هنحتاج نستخدم ال connect method الموجودة بال dbapi2

رمز:

```
#!bin/python

from pysqlite2 import dbapi2 as SQLite
```

نعمل connect ال connect ميثود بتنشئ file فى حال عدم وجوده وإذا موجود هيتعمل return بيه
رمز:

```
dbConnection=SQLite.connect("mydb.sqlite")
```

كدا انشأنا ال connection بنجاح

ملحوظة: تقدر تعمل Quick Access DB على ال Memory

رمز:

```
memConnection=SQLite.connect(":memory:")
```

بعد ماأنشأنا ال Connection محتاجين نعمل Cursor عشان نستخدمه فى التعامل مع ال DB

رمز:

```
cursor=dbConnection.cursor() #gets a cursor object..
عايزين ننشئ Table وليكن بإسم Info 3 ويشمل Fields مثلا
id: integer, primary Key
name: varchar(50)
phone: varchar(10)
```

جميل يبقة هنحتاج SQL Statement
رمز:

```
sqlStmt='CREATE TABLE info (id INTEGER PRIMARY KEY, name VARCHAR(50), phone
VARCHAR(10))'
```

ولتنفيذ ال SQL Statement نستخدم ال execute method الخاصة بال cursor object

رمز:

```
>>> cursor.execute(sqlStmt)
<pysqlite2.dbapi2.Cursor object at 0x0128B230
```

<

ندخل بعض ال داتا

رمز:

```
>>> cursor.execute('INSERT INTO info VALUES(null, "ahmed youssef",
"12345678")')
<pysqlite2.dbapi2.Cursor object at 0x0128B230>

>>> cursor.execute('INSERT INTO info VALUES(null, "3amer mohamed",
"41234114")')
<pysqlite2.dbapi2.Cursor object at 0x0128B230>
```

نقدر ندخل ال fields كالتالى ..

رمز:

```
>>> username="guru"
>>> phone   ="36987452"
```

كل اللى عليك تباصى علامة إستفهام وفى ال 2nd argument تخليها tuple مكونة من ال vars اللى عايز تدخلها ..

رمز:

```
>>> cursor.execute('INSERT INTO info VALUES(null, ?, ?)', (username, phone))
#replaced...
<pysqlite2.dbapi2.Cursor object at 0x0128B230>
```

بعد ماعدلنا او اضفنا لازم نستدعى ال Commit method لحفظ التعديلات دى ..

<<< dbConnection.commit()

ملحوظة: إذا حبيت تخلى التعديلات يتم تنفيذها اوتوماتيك

ضيف فى ال connect ميثود الخاصة بإنشاء ال connection التالى

autocommit=1

فى حالة قيامك بتعديل ما وحبيت ترجع فيه بنستخدم ال rollback method

بعد إنتهائك اقفل ال cursor, connection

رمز:

```
cursor.close()
dbConnection.close();
```

ال Reading

كالعادة لازم نعمل connect على db وننشئ ال connection

ونعمل cursor object بإستخدم cursor ميثود الموجودة بال connection object

ننفذ بعض ال sql statements ولكن هنا هنخليها عبارة عن إستعلامات بسيطة

ننشئ ال connection

رمز:

```
>>> dbConnection=SQLite.connect("mydb.sqlite") #reopen the db..
```

ننشئ cursor

رمز:

```
>>> cursor=dbConnection.cursor()
>>> #let's query the db..
```

sql statement ليتم تنفيذها

رمز:

```
>>> sqlStmt='SELECT * from info'
```

تنفيذ ال sqlStmt

رمز:

```
>>> cursor.execute(sqlStmt)
```

fetchall هى ميثود بتعيد كل ال rows على صورة tuples ف list

رمز:

```
>>> cursor.fetchall()
[(1, u'ahmed youssef', u'12345678'), (2, u'3amer mohamed', u'41234114'), (3,
u'guru', u'36987452')]
```

او تقدر تعمل شئ مشابه لكدا بإنك ت iterate على كل ال rows اللى موجودة بال result

رمز:

```
>>> for row in cursor:
        #id, name, phone
        print "----------"
        print "ID: ", row[0]
        print "Name: ", row[1]
        print "Phone: ", row[2]


----------
ID:  1
Name:  ahmed youssef
Phone:  12345678
----------
ID:  2
Name:  3amer mohamed
Phone:  41234114
----------
ID:  3
Name:  guru
Phone:  36987452
```

لاحظ إنك تقدر تتعامل معاها ب .next لأنها iterator

رمز:

```
>>> cursor.next()
(1, u'ahmed youssef', u'12345678')
>>> cursor.next()
(2, u'3amer mohamed', u'41234114')
```

(fetchmany(num

بتعيد num معين من ال rows

رمز:

```
>>> ret=cursor.fetchmany(2)
>>> ret
[(1, u'ahmed youssef', u'12345678'), (2, u'3amer mohamed', u'41234114')]
```

fetchone()

بتعيد row واحد

رمز:

```
>>> one=cursor.fetchone()
>>> one
(3, u'guru', u'36987452')
```

جميل جدا .. طب وإذا حبيت اخزن user defined type ؟
بكل بساطة اعمل ال class بتاعك الأول

رمز:

```
class Person(object):

    def __init__(self, name, phone):
        self.name=name
        self.phone=phone
```

ننشئ connection و cursor ولكن ننبه ال database انها تعمل parse لل declared types زى ال Person مثلا .. هنغير شوية ونتعامل مع ال memory

رمز:

```
#create a connection.
memConnection=SQLite.connect(':memory:',
detect_types=SQLite.PARSE_DECLTYPES)
```

رمز:

```
#cursor
cursor=memConnection.cursor()
```

الوقتى ننشئ table بحيث إنه ياخد 2 fields وهم ال ID, information

رمز:

```
cursor.execute("CREATE TABLE test (id INTEGER PRIMARY KEY, p person)")
```

جميل جدا .. ناقص إننا نحدد إزاى ال object بتاعنا يتحول ل string وازاى نجمع ال data بتاعته تانى من ال string دا
ملحوظة: إحنا بنتكلم على مجرد text بإستخدم toString method مثلا ..مش serializing objects او Pickling

رمز:

```
def adaptPerson(person):
    return "%s;%s" %(person.name, person.phone)
```

وكيفية التجميع .. بكل بساطة إحنا حولنا ال fields بتاعت ال Person object ل string ودمجناهم ب ;.. يبقة نقدر نجمعهم بإننا نفصل ال ; ونباصى ال قيم الخاصة بال fields دى لل Constructor وننشئ object منها

رمز:

```
def convToPerson(text):
    name, phone=map(str, text.split(";"))
    return Person(name, phone)
```

بعد ماعملنا الميثودز الخاصة بالتحويل والتجميع .. كل اللى ناقص اننا نبلغ SQLite بكدا

رمز:

```
SQLite.register_adapter(Person, adaptPerson)
SQLite.register_converter("person", convToPerson)
```

ننشئ شوية objects

رمز:

```
p1=Person("ahmed", "12345678")
p2=Person("rul3z", "89745632")
```

ونضيفهم لل Table

رمز:

```
cursor.execute('INSERT INTO test VALUES(null, ?)', (p1, ))
cursor.execute('INSERT INTO test VALUES(null, ?)', (p2, ))
```

نجرب نستعلم عن الموجودين

رمز:

```
#select..
cursor.execute('SELECT * from test')
for row in cursor:
    print row
```

اقتباس:

```
#output:
(1, (ahmed;12345678))
(2, (rul3z;89745632))
```

نقفل ال cursor, connection

رمز:

```
#clean-up
cursor.close()
memConnection.close()
```

الكود النهائى

رمز:

```
#!bin/python

from pysqlite2 import dbapi2 as SQLite


#dbName='myobjDBTest.sqlite'
#create a connection.
#dbConnection=SQLite.connect(dbName, detect_types=SQLite.PARSE_DECLTYPES)
memConnection=SQLite.connect(':memory:',
detect_types=SQLite.PARSE_DECLTYPES)
#cursor
cursor=memConnection.cursor()


class Person(object):

    def __init__(self, name, phone):
        self.name=name
        self.phone=phone
```

```
    def __repr__(self):
        return "(%s;%s)" %(self.name, self.phone)

#define a method to register it..

def adaptPerson(person):
    return "%s;%s" %(person.name, person.phone)

def convToPerson(text):
    name, phone=map(str, text.split(";"))
    return Person(name, phone)

SQLite.register_adapter(Person, adaptPerson)
SQLite.register_converter("person", convToPerson)

p1=Person("ahmed", "12345678")
p2=Person("rul3z", "89745632")


#create a test table..
cursor.execute("CREATE TABLE test (id INTEGER PRIMARY KEY, p person)")

#add
cursor.execute('INSERT INTO test VALUES(null, ?)', (p1, ))
cursor.execute('INSERT INTO test VALUES(null, ?)', (p2, ))

#select..
cursor.execute('SELECT * from test')
for row in cursor:
    print row

#clean-up
cursor.close()
memConnection.close()
```

وللمزيد راجع التالى :

http://www.devshed.com/c/a/Python/Us...ite-in-Python/
http://www.initd.org/tracker/pysqlite/wiki/basicintro
http://www.initd.org/pub/software/py...ined-functions

ملحوظة:
من الإصدار 2.5 و بايثون بتأتى مع ال Sqlite3