How To Use SQLite3 in Python

How to install SQLite3 for Ubuntu:

```
sudo apt-get install sqlite3 libsqlite3-dev
```

How to Install sqlitebrowser for Ubunt:

```
sudo apt-get install sqlitebrowser
```

How to use:

```
* To start: ubuntu@ubuntu:~$ sqlite3

* To quit:  sqlite> .quit

* To creat: ubuntu@ubuntu:~$  sqlite3 database_name.db

* To check: sqlite> .databases
```

Check SQLITE3 Module & Version

In [1]:

```python
import sqlite3
dir(sqlite3)
```

Out[1]:

```
['Binary',
 'Cache',
 'Connection',
 'Cursor',
 'DataError',
 'DatabaseError',
 'Date',
 'DateFromTicks',
 'Error',
 'IntegrityError',
 'InterfaceError',
 'InternalError',
 'NotSupportedError',
 'OperationalError',
 'OptimizedUnicode',
 'PARSE_COLNAMES',
 'PARSE_DECLTYPES',
 'PrepareProtocol',
```

In [2]:

```
import sqlite3
# sqlite3 module version
print(sqlite3.version)
print(sqlite3.version_info)
# sqlite3 db version
print(sqlite3.sqlite_version)
print(sqlite3.sqlite_version_info)
```

```
2.6.0
(2, 6, 0)
3.8.2
(3, 8, 2)
```

How To Use the sqlite_version function For Checking Version in sqlites

In [3]:

```
import sqlite3
def sqlite_version() :
    try :
        conn = sqlite3.connect('sql_test.db')
        cur = conn.cursor()
        sql = "select sqlite_version() AS 'SQLite Version';"

        cur.execute(sql)
        print(cur.fetchone())

        conn.close()
        print("sqlite_version sucess")
    except Exception as err :
        print('error', err)

sqlite_version()
```

```
(u'3.8.2',)
sqlite_version sucess
```

Running DB Server

DB Connection

In [4]:

```
import sqlite3
conn = sqlite3.connect('sql_test.db')

#####
print(type(conn))
print(dir(conn))
```

```
<type 'sqlite3.Connection'>
['DataError', 'DatabaseError', 'Error', 'IntegrityError', 'InterfaceEr
ror', 'InternalError', 'NotSupportedError', 'OperationalError', 'Progr
ammingError', 'Warning', '__call__', '__class__', '__delattr__', '__do
c__', '__enter__', '__exit__', '__format__', '__getattribute__', '__ha
sh__', '__init__', '__new__', '__reduce__', '__reduce_ex__', '__repr_
_', '__setattr__', '__sizeof__', '__str__', '__subclasshook__', 'clos
e', 'commit', 'create_aggregate', 'create_collation', 'create_functio
n', 'cursor', 'enable_load_extension', 'execute', 'executemany', 'exec
utescript', 'interrupt', 'isolation_level', 'iterdump', 'load_extensio
n', 'rollback', 'row_factory', 'set_authorizer', 'set_progress_handle
r', 'text_factory', 'total_changes']
```

Sqlite3.cursor

In [5]:

```
import sqlite3
conn = sqlite3.connect('sql_test.db')
cur = conn.cursor()
print(type(conn))
for i in dir(cur) :
    if not i.startswith("__") :
        print(i)
cur.close()
```

```
<type 'sqlite3.Connection'>
arraysize
close
connection
description
execute
executemany
executescript
fetchall
fetchmany
fetchone
lastrowid
next
row_factory
rowcount
setinputsizes
setoutputsize
```

cursor.excute

In [6]:

```
# Making student table
import sqlite3
conn = sqlite3.connect('sql_test.db')
cur = conn.cursor()
sql = "create table student(name text, age int)"
cur.execute(sql)
conn.commit()
conn.close()
```

In [7]:

```
# END
```

SQLite - Python Quick Guide

Cotents of Table

Connect To Database

Following Python code shows how to connect to an existing database. If the database does not exist, then it will be created and finally a database object will be returned.

In [8]:

```
import sqlite3

conn = sqlite3.connect('test.db')

print "Opened database successfully";
```

```
Opened database successfully
```

Create a Table

Following Python program will be used to create a table in the previously created database.

In [9]:

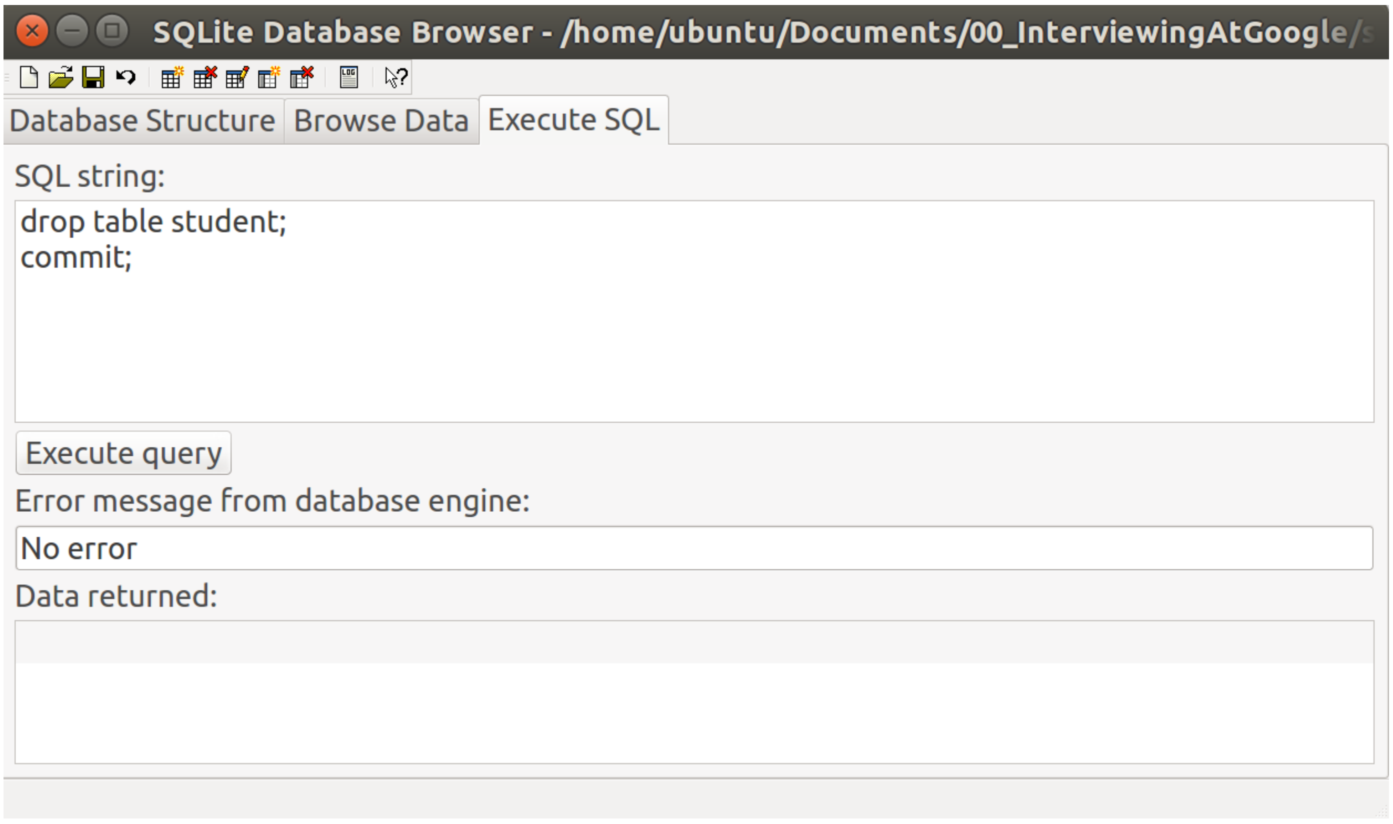
```
import sqlite3

conn = sqlite3.connect('test.db')
print "Opened database successfully";

conn.execute('''CREATE TABLE COMPANY
         (ID INT PRIMARY KEY     NOT NULL,
         NAME           TEXT    NOT NULL,
         AGE            INT     NOT NULL,
         ADDRESS        CHAR(50),
         SALARY         REAL);''')
print "Table created successfully";

conn.close()
```

```
Opened database successfully
Table created successfully
```

Delete a Table

```
DROP TABLE database_name.table_name;
```

- (Or) We can also use at SQLiteBrowser.


SQLite Database Browser - /home/ubuntu/Documents/00_InterviewingAtGoogle/s
Database Structure
Browse Data
Execute SQL
SQL string:
Execute query
Error message from database engine:
Data returned:

```
drop table student;
commit;
```

No error

In [10]:

```
# Deleting the student table in Python
import sqlite3
conn = sqlite3.connect('sql_test.db')
cur = conn.cursor()
sql = "DROP TABLE student;"
cur.execute(sql)
conn.commit()
conn.close()
```

INSERT Operation

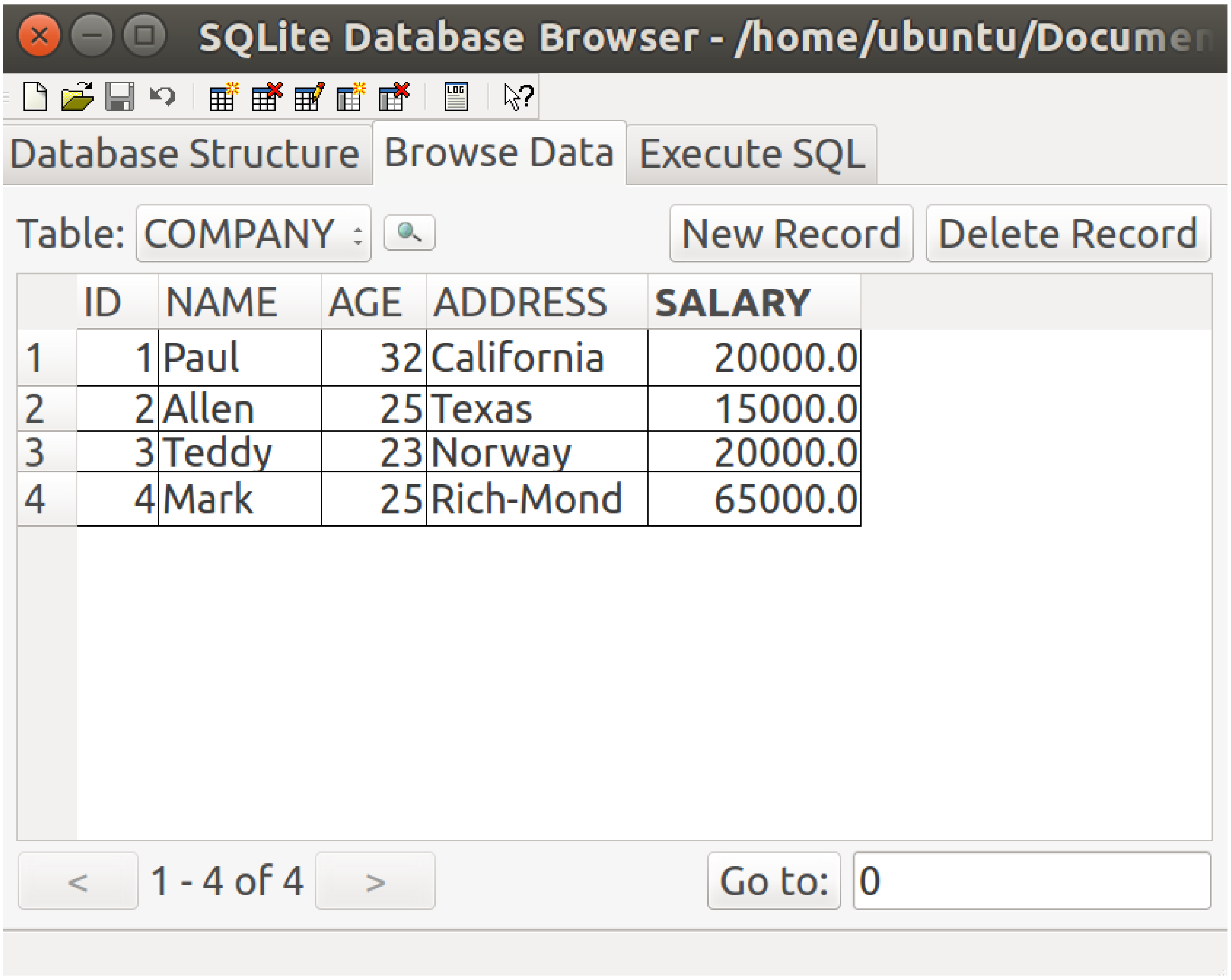
Following Python program shows how to create records in the COMPANY table created in the above example.

In [11]:

```
import sqlite3

conn = sqlite3.connect('test.db')
print "Opened database successfully";

conn.execute("INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) \
      VALUES (1, 'Paul', 32, 'California', 20000.00 )");

conn.execute("INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) \
      VALUES (2, 'Allen', 25, 'Texas', 15000.00 )");

conn.execute("INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) \
      VALUES (3, 'Teddy', 23, 'Norway', 20000.00 )");

conn.execute("INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) \
      VALUES (4, 'Mark', 25, 'Rich-Mond ', 65000.00 )");

conn.commit()
print "Records created successfully";
conn.close()
```

```
Opened database successfully
Records created successfully
```

You can also check at DBbrowser.


SQLite Database Browser - /home/ubuntu/Documen

Database Structure | Browse Data | Execute SQL

Table: COMPANY | New Record | Delete Record

	ID	NAME	AGE	ADDRESS	SALARY
1	1	Paul	32	California	20000.0
2	2	Allen	25	Texas	15000.0
3	3	Teddy	23	Norway	20000.0
4	4	Mark	25	Rich-Mond	65000.0

< 1 - 4 of 4 > Go to: 0

SELECT Operation

Following Python program shows how to fetch and display records from the COMPANY table created in the above example.

In [12]:

```
import sqlite3

conn = sqlite3.connect('test.db')
print "Opened database successfully";

cursor = conn.execute("SELECT id, name, address, salary from COMPANY")
for row in cursor:
   print "ID = ", row[0]
   print "NAME = ", row[1]
   print "ADDRESS = ", row[2]
   print "SALARY = ", row[3], "\n"

print "Operation done successfully";
conn.close()
```

```
Opened database successfully
ID =  1
NAME =  Paul
ADDRESS =  California
SALARY =  20000.0

ID =  2
NAME =  Allen
ADDRESS =  Texas
SALARY =  15000.0

ID =  3
NAME =  Teddy
ADDRESS =  Norway
SALARY =  20000.0

ID =  4
NAME =  Mark
ADDRESS =  Rich-Mond
SALARY =  65000.0

Operation done successfully
```

UPDATE Operation

Following Python code shows how to use UPDATE statement to update any record and then fetch and display the updated records from the COMPANY table.

In [13]:

```
import sqlite3

conn = sqlite3.connect('test.db')
print "Opened database successfully";

conn.execute("UPDATE COMPANY set SALARY = 40000.00 where ID = 1")
conn.commit
print "Total number of rows updated :", conn.total_changes

cursor = conn.execute("SELECT id, name, address, salary from COMPANY")
for row in cursor:
   print "ID = ", row[0]
   print "NAME = ", row[1]
   print "ADDRESS = ", row[2]
   print "SALARY = ", row[3], "\n"

print "Operation done successfully";
conn.close()
```

```
Opened database successfully
Total number of rows updated : 1
ID =  1
NAME =  Paul
ADDRESS =  California
SALARY =  40000.0

ID =  2
NAME =  Allen
ADDRESS =  Texas
SALARY =  15000.0

ID =  3
NAME =  Teddy
ADDRESS =  Norway
SALARY =  20000.0

ID =  4
NAME =  Mark
ADDRESS =  Rich-Mond
SALARY =  65000.0

Operation done successfully
```

DELETE Operation

Following Python code shows how to use DELETE statement to delete any record and then fetch and display the remaining records from the COMPANY table.

In [14]:

```python
import sqlite3

conn = sqlite3.connect('test.db')
print "Opened database successfully";

conn.execute("DELETE from COMPANY where ID = 2;")
conn.commit()
print "Total number of rows deleted :", conn.total_changes

cursor = conn.execute("SELECT id, name, address, salary from COMPANY")
for row in cursor:
   print "ID = ", row[0]
   print "NAME = ", row[1]
   print "ADDRESS = ", row[2]
   print "SALARY = ", row[3], "\n"

print "Operation done successfully";
conn.close()
```

```
Opened database successfully
Total number of rows deleted : 1
ID =  1
NAME =  Paul
ADDRESS =  California
SALARY =  20000.0

ID =  3
NAME =  Teddy
ADDRESS =  Norway
SALARY =  20000.0

ID =  4
NAME =  Mark
ADDRESS =  Rich-Mond
SALARY =  65000.0

Operation done successfully
```

You can also check at DBbrowser.