**Pandas 读写sqlite数据**，本章将学习使用[python](https://www.coolcou.com/python)内置的SQLite数据库sqlite3。SQLite3工具实现了简单、轻量级的DBMS SQL，因此可以内置于用python语言实现的任何应用。若想使用数据库的所有功能而又不想安装真正的数据库，这个工具就是最佳选择。若想在使用真正的数据库之前练习数据库操作，或在单一程序中使用数据库存储数据而无需考虑接口，SQLite3都是不错的选择。



**使用Pandas库提供的I/O API**

如下所示，创建一个DataFrame对象，我们将用它在SQLite3数据库新建一张表。如[上章](http://coolcou.com/pandas/pandas-read-write/pandas-database-operations.html)所述，使用create\_engine()连接sqlite3数据库，再使用to\_sql()转换为数据库表，如下所示：

import pandas as pd

import numpy as np

from sqlalchemy import create\_engine

frame = pd.DataFrame(np.arange(20).reshape(4,5),

columns=['white', 'red', 'blue', 'black', 'green'])

engine= create\_engine('sqlite:///foo.db')

print(frame)

frame.to\_sql('colors', engine)

Python

Copy

输出结果如下：  


反之，读取数据库，则需要使用read\_sql()函数，参数为表名和engine实例，如下所示：

import pandas as pd

from sqlalchemy import create\_engine

engine= create\_engine('sqlite:///foo.db')

frame = pd.read\_sql('colors', engine)

print(frame)

Python

Copy

输出结果如下:  


如上所示，由于pandas库提供了I/O API，数据库写操作也变得非常简单。

**不使用Pandas库提供的I/O API**

我们尝试不使用I/O API的情况下，应该如何实现相同的操作，这样可以让我们理解为什么pandas是读写数据库的好工具。

首先，连接数据库，创建数据表，正确定义数据类型，数据类型应与要加载的数据对得上。

import sqlite3

query = """

CREATE TABLE test

(a VARCHAR(20), b VARCHAR(20),

c REAL, d INTEGER);

"""

con = sqlite3.connect(":memory:")

print(con.execute(query))

con.commit()

# 使用SQL INSERT语句插入数据

data = [('white', 'up', 1, 3),

('black', 'down', 2, 8),

('green', 'up', 4, 4),

('red', 'down', 5, 5)]

stmt = "INSERT INTO test VALUES(?,?,?,?)"

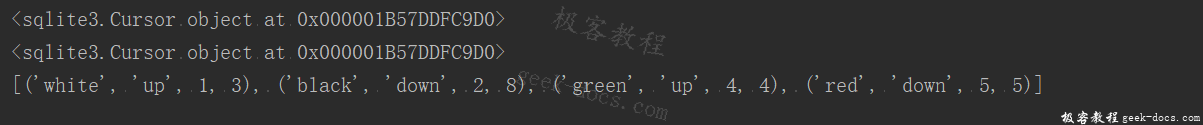
print(con.executemany(stmt, data))

con.commit()

print(data)

Python

Copy

输出结果如下:  


下面从数据库查找刚才插入的数据，可以使用SQL SELECT语句，如下所示：

import sqlite3

query = """

CREATE TABLE test

(a VARCHAR(20), b VARCHAR(20),

c REAL, d INTEGER);

"""

con = sqlite3.connect(":memory:")

print(con.execute(query))

con.commit()

# 使用SQL INSERT语句插入数据

data = [('white', 'up', 1, 3),

('black', 'down', 2, 8),

('green', 'up', 4, 4),

('red', 'down', 5, 5)]

stmt = "INSERT INTO test VALUES(?,?,?,?)"

print(con.executemany(stmt, data))

con.commit()

# 使用SQL select语句查询数据

cursor = con.execute("select \* from test")

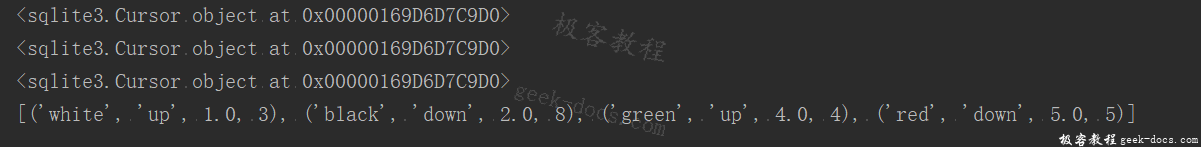
print(cursor)

rows = cursor.fetchall()

print(rows)

Python

Copy

输出结果如下:  


把元组列表传给**DataFrame**的构造函数，如需要列表名称，可以用游标的description属性来获取，如下所示：

import sqlite3

import pandas as pd

query = """

CREATE TABLE test

(a VARCHAR(20), b VARCHAR(20),

c REAL, d INTEGER);

"""

con = sqlite3.connect(":memory:")

print(con.execute(query))

con.commit()

# 使用SQL INSERT语句插入数据

data = [('white', 'up', 1, 3),

('black', 'down', 2, 8),

('green', 'up', 4, 4),

('red', 'down', 5, 5)]

stmt = "INSERT INTO test VALUES(?,?,?,?)"

print(con.executemany(stmt, data))

con.commit()

# 使用SQL select语句查询数据

cursor = con.execute("select \* from test")

print(cursor)

rows = cursor.fetchall()

# 使用 description 获取属性

print(cursor.description)

print('------------')

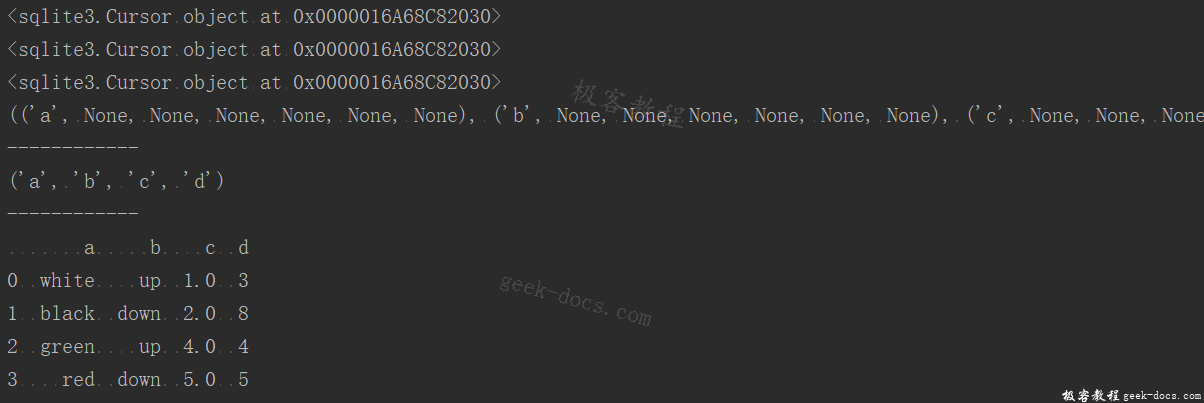
print(list(zip(\*cursor.description))[0])

print('------------')

print(pd.DataFrame(rows, columns=list(zip(\*cursor.description))[0]))

Python

Copy

输出结果如下：  


如上所示，不使用pandas库I/O API也可以达到目的，可是**操作起来就太费劲**啦。