


ХОЧУ ПОМОЧЬ  
ПРОЕКТУ



mango-office.ru

РЕКЛАМА

Виртуальная АТС Расширенная

2 000 ₽

Узнать больше

С

SQL

тр

яз

Не

пр


Мо

М

В

за

вр



12+

Сроки проведения с 18.09.23 по 30.09.23. Товары и условия их распродажи определяются продавцами таких товаров и отмечены значком «Распродажа». Представлены собирательные образы товаров.

/ Драйвер для баз данных SQLite3

С, которая предоставляет легковесную дисковую базу данных. База данных SQLite не процесса и позволяет получить доступ к базе данных, используя нестандартный вариант

пользовать SQLite для внутреннего хранения данных. Также возможно создать прототип QLite, а затем перенести код в большую базу данных, такую как PostgreSQL или Oracle.

нтерфейс SQL, совместимый со спецификацией DB-API 2.0, описанной в PEP 249.

кали проблемы с разделением соединений между потоками. По этому [модуль sqlite3](#) Python я и курсоры между потоками. Если попытаться это сделать, то получим исключение во

Единственным методом, имеющим смысл вызывать только из другого потока является вызов метода [connect.interrupt\(\)](#).

## Примеры использования:

Чтобы использовать модуль, необходимо сначала создать объект [Connection](#), который представляет базу данных. В примерах, данные будут храниться в файле example.db:

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

Можно также указать специальное имя :memory: для создания базы данных в оперативной памяти.

Получив соединение [Connection](#), можно создать [объект Cursor](#) и вызвать его метод [cursor.execute\(\)](#) для выполнения команд SQL:

```
cursor = conn.cursor()

# Создать таблицу
cursor.execute('''CREATE TABLE stocks
                  (date text, trans text, symbol text, qty real, price real)''')

# Вставить строку данных
cursor.execute("INSERT INTO stocks VALUES ('2006-01-05','BUY','RHAT',100,35.14)")

# Сохранить (зафиксировать) изменения
conn.commit()

# Можно закрыть соединение, если оно больше не нужно.
# Убедитесь, что все изменения были зафиксированы, или они будут потеряны.
conn.close()
```

Данные, которые были сохранены, являются постоянными и доступны в следующих сеансах:

```
import sqlite3
conn = sqlite3.connect('example.db')
cursor = conn.cursor()
```

Вверх

Обычно операции SQL могут использовать значения из переменных в Python. Нет необходимости собирать запрос, используя [строковые операции](#) Python, потому что это небезопасно. Это делает программу уязвимой для атаки SQL-инъекцией.

Вместо этого используйте подстановку параметров DB-API. Поставьте символ '?' в качестве заполнителя везде, где вы хотите использовать значение переменной, а затем предоставьте [кортеж](#) значений в качестве второго аргумента [метода курсора cursor.execute\(\)](#). Другие модули базы данных могут использовать другой заполнитель, такой как '%s' или ':1'.

Например:

```
# Никогда не делай этого - небезопасно!
symbol = 'RHAT'
cursor.execute("SELECT * FROM stocks WHERE symbol = '%s'" % symbol)

# Делайте все время так, как показано ниже.
t = ('RHAT',)
cursor.execute('SELECT * FROM stocks WHERE symbol=?', t)
print(cursor.fetchone())

# Larger example that inserts many records at a time
purchases = [('2020-03-28', 'BUY', 'IBM', 1000, 45.00),
              ('2020-04-05', 'BUY', 'MSFT', 1000, 72.00),
              ('2020-04-06', 'SELL', 'IBM', 500, 53.00),
              ]
cursor.executemany('INSERT INTO stocks VALUES (?,?,?,?,?)', purchases)
```

Чтобы получить данные после выполнения оператора SELECT, можно либо обработать курсор как итератор, вызвать метод курсора [cursor.fetchone\(\)](#), чтобы получить единственную совпадающую строку, либо вызвать [cursor.fetchall\(\)](#), чтобы получить [список](#) совпадающих строк.

В этом примере используется форма [итератора](#):

```
>>> for row in cursor.execute('SELECT * FROM stocks ORDER BY price'):
...     print(row)
...
# ('2020-01-05', 'BUY', 'RHAT', 100, 35.14)
# ('2020-03-28', 'BUY', 'IBM', 1000, 45.0)
# ('2020-04-06', 'SELL', 'IBM', 500, 53.0)
# ('2020-04-05', 'BUY', 'MSFT', 1000, 72.0)
```

### Содержание раздела:

- [КРАТКИЙ ОБЗОР МАТЕРИАЛА.](#)
- [Типы SQLite3 и Python](#)
- [Хранение типов Python в базах данных SQLite3](#)
- [Пример преобразования типов между SQLite и Python](#)
- [Встроенные адаптеры и конвертеры модуля sqlite3](#)
- [Управление транзакциями в модуле sqlite3 Python](#)
- [Эффективное использование sqlite3](#)
- [Функция connect\(\) модуля sqlite3](#)
- [Функция register converter\(\) модуля sqlite3](#)
- [Функция register adapter\(\) модуля sqlite3](#)
- [Методы объекта Connection модуля sqlite3](#)
- [Методы объекта Cursor модуля sqlite3](#)
- [Объект Row модуля sqlite3](#)
- [Сравнение кириллицы в SQLite без учета регистра](#)
- [Функции и константы модуля sqlite3](#)
- [Исключения модуля sqlite3](#)
- [Объект Blob\(\) модуля sqlite3](#)
- [Вверх CSV и XLSX\(XLS\) файла в sqlite3, экспорт данных в CSV](#)

РЕКЛАМА



Вверх