python数据库API简介

# 全局变量

python推荐支持DB API2.0的数据库模块都应该提供如下3个全局变量。

apilevel: 数据库模块的API版本号

threadsafety

paramstyle: sql语句重使用哪种风格的参数

## 数据库API的核心类

数据库连接：用于获取游标，控制事务

cursor

commit

rollback

close

isolation\_level

in\_transaction

游标：执行各种sql语句

execute

executemany

executescript

fetchone

fetchmany

fetchall

close

rowcount

lastrowid

arraysize

description

connection

## sqlite3

apt install sqlite -y

sqlite> create table my\_test(\_id integer primary key, name, pass, gender);

sqlite> insert into my\_test values(1, "zhangsan", "123", "male");

sqlite> insert into my\_test values(2, "zcw", "1234", "male");

sqlite> insert into my\_test values(3, "wangwu", "12345", "female");

### 命令行

sqlite> .help

sqlite> .databases

sqlite> .tables

root@2004a:/home/test# cat test.py

import sqlite3

print(sqlite3.apilevel)

print(sqlite3.threadsafety)

print(sqlite3.paramstyle)

### 建表

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

c = conn.cursor()

c.execute('''create table user\_tb(\_id integer primary key AUTOINCREMENT, name text, pass text, gender text)''')

c.execute('''create table order\_tb(\_id integer primary key AUTOINCREMENT, item\_name text, item\_price text,

item\_number real,

user\_id integer,

foreign key(user\_id) references user\_tb(\_id))''')

c.close()

conn.close()

### 插入

import sqlite3

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

c=conn.cursor()

c.execute('insert into user\_tb values(null,?,?,?)', ('sunwukong', '123456', 'male'))

c.execute('insert into order\_tb values(null,?,?,?,?)', ('mouse', '34.2', '3', 1))

conn.commit()

c.close()

conn.close()

### 批量更新

import sqlite3

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

c=conn.cursor()

c.executemany('insert into user\_tb values(null,?,?,?)',

(('sun', '123456', 'male'), ('bai', '123456', 'female'),('zhu', '123456', 'male'),('niu', '123456', 'male'),('tang', '123456', 'male') ))

conn.commit()

c.close()

conn.close()

root@2004a:/home/test# cat test3.py

import sqlite3

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

c = conn.cursor()

c.executemany('update user\_tb set name=? where \_id=?',

(('xiao sun', 2), ('xiao bai', 3), ('xiao zhu', 4), ('xiao niu', 5),('xiao tang', 6),))

print("受影响的记录条数: ", c.rowcount)

conn.commit()

c.close()

conn.close()

root@2004a:/home/test#

root@2004a:/home/test# python3 test3.py

受影响的记录条数: 4

### 查询

import sqlite3

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

c = conn.cursor()

c.execute('select \* from user\_tb where \_id > ?', (2,))

print("查询结果返回的记录数:", c.rowcount)

for col in (c.description):

print(col[0], end='\t')

print('\n----------------------')

while True:

row = c.fetchone()

if not row:

break

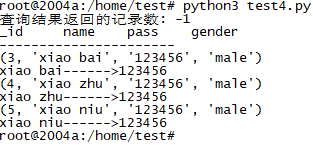
print(row)

print(row[1] + '------>' + row[2])

#conn.commit)

c.close()

conn.close()



### 批量查询

import sqlite3

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

c = conn.cursor()

c.execute('select \* from user\_tb where \_id > ?', (2,))

print("查询结果返回的记录数:", c.rowcount)

for col in (c.description):

print(col[0], end='\t')

print('\n----------------------')

while True:

rows = c.fetchmany(3)

if not rows:

break

for r in rows:

print(r)

#conn.commit)

c.close()

conn.close()

### 事务控制

事务的ACID特性

原子性

一致性：原子性保证一致性。一致性是指事务执行的结果，以转账为例，不能A的账户还未划扣，但B的账户已经进账了。

隔离性：各个事务的执行互不干扰

持续性：持久性，保存到物理数据库

### 执行sql脚本

import sqlite3

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

c=conn.cursor()

c.executescript('''

insert into user\_tb values(null,'wusong', '3444', 'male');

insert into user\_tb values(null,'linchong', '44444', 'male');

create table item\_tb(id integer primary key autoincrement, name, price); ''')

conn.commit()

c.close()

conn.close()

### 自定义函数

import sqlite3

def myfunc(st):

return '[' + st[::-1] + ']' #字符串反转

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

conn.create\_function('enc', 1, myfunc)

c = conn.cursor()

c.execute('insert into user\_tb values(null, ?, enc(?), ?)', ('jiabaoyu', '12456', 'male'))

conn.commit()

c.close()

conn.close()

### 聚合函数

sqlite3内置了一些聚合函数

sum()

avg()

count()

max()

min()

要定义自己的聚合函数，需要注册一个自定义聚集类

该类必须实现step(self, params…), finalize(self)

step对于查询所返回的每条记录各执行一次

finalize(self)方法只在最后执行一次，将该方法的返回值作为聚集函数的终值返回

import sqlite3

class MinLen:

def \_\_init\_\_(self):

self.min\_len = None

def step(self, value):

if self.min\_len is None:

self.min\_len = value

return

if len(self.min\_len) > len(value):

self.min\_len = value

def finalize(self):

return self.min\_len

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

conn.create\_aggregate('min\_len', 1, MinLen)

c = conn.cursor()

c.execute('select min\_len(pass) from user\_tb')

print(c.fetchone()[0])

conn.commit()

c.close()

conn.close()

### 比较函数

root@2004a:/home/test# cat test7.py

import sqlite3

def myfunc(s1, s2):

if s1[1:-1] == s2[1:-1]: #去掉首字符和末尾字符

return 0

elif s1[1:-1] > s2[1:-1]:

return 1

else:

return -1

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

#conn.create\_aggregate('min\_len', 1, MinLen)

conn.create\_collation('sub\_cmp', myfunc) #自定义比较函数

c = conn.cursor()

c.execute('select \* from user\_tb order by pass collate sub\_cmp') #根据密码进行排序

for row in c:

print(row)

#conn.commit()

c.close()

conn.close()

root@2004a:/home/test# python3 test7.py

(1, 'sun', '123456', 'male')

(2, 'xiao sun', '123456', 'female')

(3, 'xiao bai', '123456', 'male')

(4, 'xiao zhu', '123456', 'male')

(5, 'xiao niu', '123456', 'male')

(6, 'wusong', '3444', 'male')

(7, 'linchong', '44444', 'male')

(8, 'jiabaoyu', '[65421]', 'male')

## mysql

python自带了一个pip工具用来查看，管理所安装的各种模块

pip show packagename

apt install python3-pip -y

pip install mysql-connector-python

apt-get update

apt install mysql-server -y

systemctl status mysql

root@2004a:/home/test# cat test.py

import mysql.connector

print(mysql.connector.apilevel)

print(mysql.connector.threadsafety)

print(mysql.connector.paramstyle)

root@2004a:/home/test# python3 test.py

2.0

1

pyformat

root@2004a:/home/test#

### 修改密码及创建数据库

root@2004a:/home/test# mysql

Server version: 8.0.28-0ubuntu0.20.04.3 (Ubuntu)

mysql> alter user 'root'@'localhost' identified with mysql\_native\_password by '123456';

Query OK, 0 rows affected (0.01 sec)

mysql> flush privileges;

Query OK, 0 rows affected (0.00 sec)

mysql> create database python;

Query OK, 1 row affected (0.01 sec)

mysql> use python

Database changed

mysql -u root -p

### 建表(DDL语句)

root@2004a:/home/test# cat test.py

import mysql.connector

print(mysql.connector.apilevel)

print(mysql.connector.threadsafety)

print(mysql.connector.paramstyle)

conn=mysql.connector.connect(user='root', password='123456', host='localhost', port='3306', database='python', use\_unicode=True)

c=conn.cursor()

c.execute('''create table if not exists user\_tb(user\_id int primary key auto\_increment,

name varchar(255),

pass varchar(255),

gender varchar(255))''')

c.execute('''create table if not exists order\_tb(order\_id int primary key auto\_increment,

item\_name varchar(255),

item\_price double,

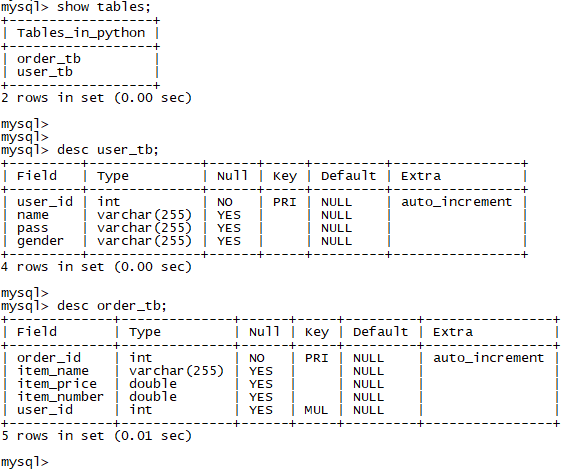
item\_number double,

user\_id int,

foreign key(user\_id) references user\_tb(user\_id))''')

c.close()

conn.close()



### 插入（DML语句）

|  |
| --- |
| 先插入user\_tb，看看自动生成的user\_id是啥，这里假设user\_id是6 |
| import mysql.connector  print(mysql.connector.apilevel)  print(mysql.connector.threadsafety)  print(mysql.connector.paramstyle)  conn=mysql.connector.connect(user='root', password='123456', host='localhost', port='3306', database='python', use\_unicode=True)  c=conn.cursor()  c.execute('insert into user\_tb values(null, %s, %s, %s)', ('sunwukong', '123456', 'male'))  conn.commit()  c.close()  conn.close() |
| 再插入order\_tb |
| import mysql.connector  print(mysql.connector.apilevel)  print(mysql.connector.threadsafety)  print(mysql.connector.paramstyle)  conn=mysql.connector.connect(user='root', password='123456', host='localhost', port='3306', database='python', use\_unicode=True)  c=conn.cursor()  c.execute('insert into order\_tb values(null, %s, %s, %s, %s)',  ('mouse', '34.2', '3', 6))  conn.commit()  c.close()  conn.close() |

import mysql.connector

print(mysql.connector.apilevel)

print(mysql.connector.threadsafety)

print(mysql.connector.paramstyle)

conn=mysql.connector.connect(user='root', password='123456', host='localhost', port='3306', database='python', use\_unicode=True)

c=conn.cursor()

#c.execute('insert into user\_tb values(null, %s, %s, %s)',

# ('sunwukong', '123456', 'male'))

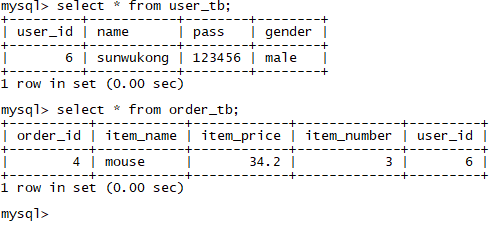
c.execute('insert into order\_tb values(null, %s, %s, %s, %s)',

('mouse', '34.2', '3', 6))

conn.commit()

c.close()

conn.close()



### 批量插入

root@2004a:/home/test# cat test2.py

import mysql.connector

conn=mysql.connector.connect(user='root', password='123456', host='localhost', port='3306', database='python', use\_unicode=True)

c=conn.cursor()

c.executemany('insert into user\_tb values(null, %s, %s, %s)',

(('sun', '123456', 'male'),

('bai', '123456', 'female'), #不能是tab键

('zhu', '123456', 'male'),

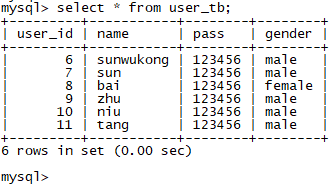
('niu', '123456', 'male'),

('tang', '123456', 'male')))

conn.commit()

c.close()

conn.close()



### autocommit

mysql的连接有各自动提交的属性

不需要显示调用conn.commit()

root@2004a:/home/test# cat test3.py

import mysql.connector

conn=mysql.connector.connect(user='root', password='123456', host='localhost', port='3306', database='python', use\_unicode=True)

c=conn.cursor()

conn.autocommit = True

#conn.commit()

c.close()

conn.close()

### 查询

root@2004a:/home/test# cat test4.py

import mysql.connector

conn=mysql.connector.connect(user='root', password='123456', host='localhost', port='3306', database='python', use\_unicode=True)

c=conn.cursor()

c.execute('select \* from user\_tb where user\_id > %s', (2,))

for col in (c.description):

print(col[0], end='\t')

for row in c:

print(row)

print(row[1] + '---->' + row[2])

c.close()

conn.close()

root@2004a:/home/test#

root@2004a:/home/test# python3 test4.py

user\_id name pass gender (6, 'sunwukong', '123456', 'male')

sunwukong---->123456

(7, 'sun', '123456', 'male')

sun---->123456

(8, 'bai', '123456', 'female')

bai---->123456

(9, 'zhu', '123456', 'male')

zhu---->123456

(10, 'niu', '123456', 'male')

niu---->123456

(11, 'tang', '123456', 'male')

tang---->123456

### 存储过程

存储过程是mysql语句编写的一段脚本

可以当函数使

mysql>

mysql> use python

Database changed

mysql>

mysql> delimiter //

mysql> create procedure add\_proc(a int, b int, out sum int)

-> begin

-> set sum=a+b;

-> end;

-> //

Query OK, 0 rows affected (0.02 sec)

root@2004a:/home/test# cat test5.py

import mysql.connector

conn=mysql.connector.connect(user='root', password='123456', host='localhost', port='3306', database='python', use\_unicode=True)

c=conn.cursor()

res = c.callproc('add\_proc', (5,6,0))

print(res)

print(res[2])

conn.commit()

c.close()

conn.close()

root@2004a:/home/test# python3 test5.py

(5, 6, 11)

11

root@2004a:/home/test#