

# 专栏：008：MySQLdb及其模拟转账

## 用理工科思维看待这个世界

### 系列爬虫专栏

崇尚的学习思维是：输入，输出平衡，且平衡点不断攀升。

曾经有大神告诫说：没事别瞎写文章；所以，很认真的写的是能力范围内的，看客要是看不懂，不是你的问题，问题在我，得持续输入，再输出。

今天的主题是：MySQLdb及其模拟转账

## 1：框架

序号	内容	说明
01	概念及其工具介绍	—
02	SQL语句	—
03	实例演示数据库操作	—
04	银行转账操作演示	—
05	参考及其说明	—

## 2：概念，工具介绍

- MySQL：关系型数据库  
MySQL（官方发音为英语发音：/maɪˌɛskjuːˈɛl/“My S-Q-L”[1]，但也经常读作英语发音：/maɪˈsiːkwəl/“My Sequel”）原本是一个开放源代码的关系数据库管理系统，原开发者为瑞典的MySQL AB公司，该公司于2008年被昇阳微系统（Sun Microsystems）收购。2009年，甲骨文公司（Oracle）收购昇阳微系统公司，MySQL成为Oracle旗下产品。
- MySQL 服务器的 下载：[官网](#)
- python MySQL 驱动 安装: [下载安装](#)
- python MySQL 开发环境

01 : Python 集成开发环境编写SQL语句  
02 : python客户端 + python MySQL 驱动  
03 : MySQL 服务器 : MySQL客户端工具

本地需要先开启MySQL服务，python编写SQL语句，驱动完成python和MySQL之间的联系。这样就可以实现数据库数据的增删改查。

## 3 : SQL语句

- SQL语句  
SQL是用于访问和处理数据库的标准的计算机语言：  
[教程w3school](#)  
基本操作：增删改查
- 查：  
SELECT 列名称 FROM 表名称  
一个名为"Persons"的表：提取LastName列：`select LastName from Persons`

Id	LastName	FirstName	Address	City
1	Adams	John	Oxford Street	London
2	Bush	George	Fifth Avenue	New York
3	Carter	Thomas	Changan Street	Beijing

结果：

LastName
Adams
Bush
Carter

- 改：  
UPDATE 表名称 SET 列名称 = 新值 WHERE 列名称 = 某值
- 增：  
01 : INSERT INTO table\_name (列1, 列2,...) VALUES (值1, 值2,...) 02 : INSERT INTO 表名称 VALUES (值1, 值2,...)
- 删：  
DELETE FROM 表名称 WHERE 列名称 = 值

还有好些高级的用法....[点这里](#)

这里主要讲解在python中的使用。

## 4 : 代码演示

- MySQLdb

## 01：创建连接对象

序号	方法	使用说明
01	MySQLdb.Connet()	创建数据库连接对象

参数	类型	说明
host	字符串	MySQL服务器地址，本地：localhost
port	数字	MySQL服务器端口号
user	字符串	用户名，默认root
passwd	字符串	密码，默认空
db	字符串	数据库名称
charset	字符串	连接编码

## 02：连接对象支持的方法

方法名	说明
cursor()	游标对象：用于查询和获取结果
commit()	提交当前事务
rollback()	回滚当前事务
close()	关闭连接

## 03：游标对象 所支持的方法

方法	说明
execute()	执行一个数据库查询和命令
fetchone()	取结果集的下一个
fetchmany(size)	获取结果集下几行
fetchall()	获取剩下的所有
rowcount	最近一次execute影响的行数
close()	关闭游标对象

## 04：使用MySQLdb的方法流程

- connection:建立数据库连接
- cursor: 执行SQL,获取数据库数据
- 提交事务或者回滚操作
- 关闭cursor , 关闭connection连接对象

数据库可视化工具：SQLyog (有好些，看你自己喜好了)

在本地已经手工创建了一个 `db = exercise` 的数据库，表名为：`persons` 如图：

Id	LastName	FirstName	Address	City
1	Adams	John	Oxford Street	London
2	Bush	George	Fifth Avenue	New York
3	Carter	Thomas	Changan Street	Beijing

```
# 获取第一行
import MySQLdb
connector = MySQLdb.connect(
    user="root",
    host="localhost",
    port=3306,
    passwd="123456",
    db="exercise",
    charset="utf8") # 创建连接对象
cur = connector.cursor() # 创建游标对象
sql = 'select * from persons WHERE id =1' # 编写的Sql语句
cur.execute(sql) # 使用游标对象的执行方法
print(cur.fetchone()) # 使用右边对象的获取方法
# output
(1, 'Adams', 'John', 'Oxford Street', 'London')
```

```
# 第一行的FirstName 更改成 'xiexiaolu'
sql2 = "UPDATE persons SET FirstName = 'xiexiaolu' WHERE id = 1"
cur.execute(sql2)

# 表数据变成
(1, 'Adams', 'xiexiaolu', 'Oxford Street', 'London')
(2, 'Bush', 'George', 'Fifth Avenue', 'New York')
(3, 'Carter', 'Thomas', 'Changan Street', 'Beijing')
```

```
# 增加一条数据
sql3 ="INSERT INTO persons(Id, LastName,FirstName,Address,City)VALUES (4,
'xiaolu', 'xie','zhabei','shanghai')"

# output
(1, 'Adams', 'John', 'Oxford Street', 'London')
(2, 'Bush', 'George', 'Fifth Avenue', 'New York')
(3, 'Carter', 'Thomas', 'Changan Street', 'Beijing')
(4, 'xiaolu', 'xie', 'zhabei', 'shanghai')
```

## 更多操作

# 5：银行转账操作演示

模拟银行两个账户之间的资金流动：  
流程：

- 检查账户是否存在
- 检查账户是否存在足够的资金
- A账户 - 资金
- B账户 + 资金

表名为：`bank`:完成 `zhangsan` 向 `lisi` 转账100元

id	money
zhangsan	110
lisi	10

```
# 检查账户
def check_account(self, account_id):
    cur = self.con.cursor()
    sql = 'select * from bank WHERE id = "%s"' % (str(account_id))
    cur.execute(sql)
    if cur.rowcount == 1:
        print("%s existes."%(account_id))
    else:
        print("%s is wrong" % account_id)
```

# 检查是否有足够的钱

```
def check_money(self, values, account_id):
    cur = self.con.cursor()
    sql = 'select * from bank WHERE id="%s" AND money>%s' % (account_id, values)
    cur.execute(sql)
    print(cur.rowcount)
    if cur.rowcount == 1:
        print("Enough money!")
        return True
    else:
        print("No enough money!")
        return False
```

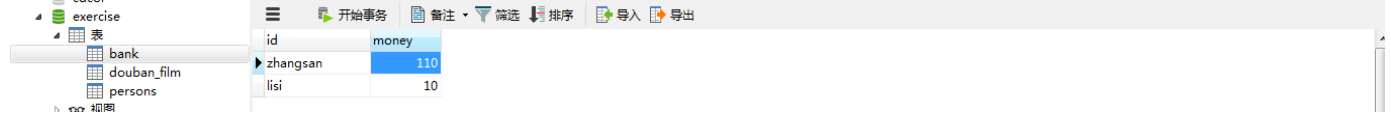
# 减款操作

```
def subtract(self, accout_id, tranfer_money, Flag):
    cur = self.con.cursor()
    if Flag:
        try:
            sql = 'update bank SET money = money - %s where id = "%s"' % (tranfer_money, accout_id)
            cur.execute(sql)
            print(cur.rowcount)
            if cur.rowcount == 1:
                print(u"减款成功")
        finally:
            cur.close()
    else:
        print(u"操作不成功.")
```

#加款操作

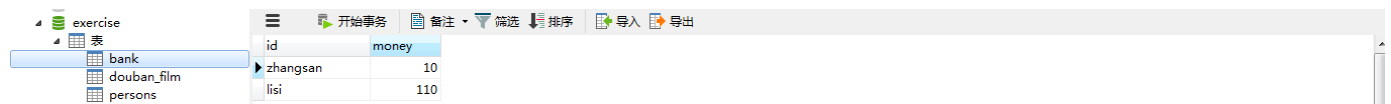
```
def add(self, accout_id, tranfer_money, Flag):
    cur = self.con.cursor()
    if Flag:
        try:
            sql = 'update bank SET money = money + %s WHERE id = "%s"' % (tranfer_money, accout_id)
            cur.execute(sql)
            print(cur.rowcount)
            if cur.rowcount == 1:
                print(u"加款成功")
        finally:
            cur.close()
    else:
        print(u"操作不成功.")
```

效果显示：



id	money
zhangsan	110
lisi	10

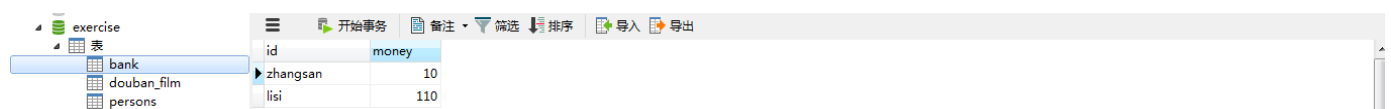
执行后：



id	money
zhangsan	10
lisi	110

```
zhangsan exists.
lisi exists.
Enough money!
减款成功
加款成功
```

第二次执行的初始状态：



id	money
zhangsan	10
lisi	110

```
zhangsan exists.
lisi exists.
0
No enough money!
操作不成功.
操作不成功.
```

完整版代码：[完整代码](#)

数据库的学习是为了完成爬虫数据的储存。

## 6：参考及其说明

参考列表：

01: [参考1](#)

02: [参考2](#)

03: [参考：慕课网](#)

Github: [github](#)

关于本人：

国内小硕，半路出家的IT学习者。

兴趣领域：爬虫，数据科学

本人正在构建一个共同成长爬虫小型社群。有兴趣私信。

文档及代码托管在Github上。

下期预告：爬取豆瓣电影，并存储在数据库中...(为啥都爬豆瓣来着)

---