[置顶] 采用python中SQLalchemy模块访问数据库

采用python中SQLalchemy模块访问数据库

1. SQLalchemy简介

SQLAlchemy是一个开源的SQL工具包，基本Python编程语言的MIT许可证而发布的对象关系映射器。SQLAlchemy提供了“一个熟知的企业级全套持久性模式，

专为高效率和高性能的数据库访问而设计“。SQLAlchemy的首次发布2006年2月，并已迅速成为最广泛使用的对象关系映射在Python社区的工具之一。

使用ORM等独立SQLAlchemy的一个优势在于其允许开发人员首先考虑数据模型，并能决定稍后可视化数据的方式

2. SQLAlchempy的安装

首先需安装mysql

然后，下载SQLAlchemy-0.7.2，然后打开cmd，在安装包文件目录下，

运行python setup.py install，通过python下输入import sqlalchemy，执行未报错则表示安装成功

3. SQLAlchemy的使用实例

一、完成简单数据表信息查询

```
# 1. 导入模块
from sqlalchemy import *
from sqlclchemy.orm import *
# 2. 建立数据库引擎
mysql_engine = create_engine("$address", echo, module)
#address 数据库://用户名:密码（没有密码则为空）@主机名：端口/数据库名
#echo标识用于设置通过python标准日志模块完成的SQLAlchemy日志系统，当开启日志功能，我们将能看到所有的SQL生成代码
# 3. 建立连接
connection = mysql_engine.connect()
# 4. 查询表信息
result = connection.execute("select name from t_name)
for row in result:
print "name: ", row['name']
# 5. 关闭连接
connection.close()
```

二、插入新的数据表

```
 # 1. 导入模块


 from sqlalchemy import *
```

```
from sqlclchemy.orm import *

# 2. 建立数据库引擎

mysql_engine = create_engine("$address", echo, module)

#address 数据库://用户名:密码（没有密码则为空）@主机名：端口/数据库名

#echo标识用于设置通过python标准日志模块完成的SQLAlchemy日志系统，当开启日志功能，我们将能看到所有的SQL生成代码

# 3. 设置metadata并将其绑定到数据库引擎

metadata = Metadata(mysql_engine)

# 4. 定义需新建的表

users = Table('users', metadata,Column('user_id', Integer, primary_key=True),

Column('name', String(40)),

Column('age', Integer),

Column('password', String),)

#Table实现方式与SQL语言中的CRETE TABLE类似

# 5. 在数据库中创建表

metadata.create_all(mysql_engine)

#向数据库发出CREATE TABLE命令，由此数据库新建名为users的表

#调用时会检查已经存在的表结构，因此可重复调用
```

```
# 6. 创建一个与数据库中的users表匹配的python类

class user():

def __int__(self, name, fullname, password):

self.name = name

self.fullname = fullname

self.passwd = passwd

#python类的属性需与users表的列名一致

# 7. 设置映射

from sqlalchemy.orm import mapper

mapper(user, users)

# mapper()创建一个新的Mapper对象，与定义的类相关联

# 8. 创建session

Session = sessionmaker(bind=mysql_egnine)

session = Session()

#由此我们只需对python的user类的操作，后台数据库的具体实现交由session完成

# 9. 执行
```

```
session.commit()

#实现与数据库的交互

# 10. 查询

usr_info = session.query(user).filter_by(age=12).first()

#返回数据库中年纪12岁的第一条数据
```