专栏：012：没时间解释了，快使用sqlalchemy

E – 爬虫技术 爬虫技术

> **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。

今天的主题是：sqlalchmey的使用(这是一篇没有真正实战的博文)

0：框架

序号	内容	说明
01	概念解释	是什么？
02	代码解释	怎么做？
03	总结	如何做的？

1：概念

- ORM
  对象关系映射（英语：Object Relational Mapping，简称ORM，或O/RM，或O/R mapping），是一种程序设计技术，用于实现面向对象编程语言里不同类型系统的数据之间的转换。

图片显示：对象和数据库之间的映射

```python
class Test1Data(object):
    def __init__(self, client_id, expires, salt, user_id):
        self.client_id = client_id
        self.expires = expires
        self.salt = salt
        self.user_id = user_id
```

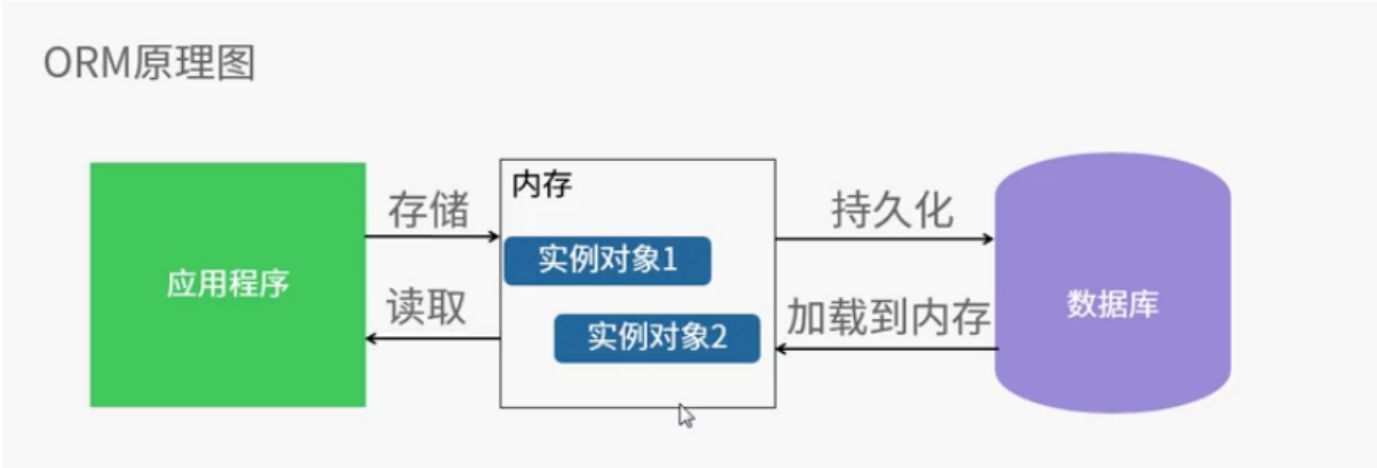
O

M

client_id	expires	salt	user_id
1	1429459171	12297302746346	1001
2	1429459171	52154829157439	1002
3	1429459171	91118614224180	1003

R

原理：



- SQLAlchemy
  是python的一款开源软件，提供了SQL工具包及对象关系映射(ORM)工具。(需要安装第三方库)
  官方文档
- 为什么会出现这种技术？
  一句话解释：为了避免写繁复的sql语句.（隐藏数据库，良好的数据接口，动态的数据映射，引入缓存）

2：代码解释

一般步骤：

- 创建连接
- 声明映射文件

- 创建模式
- 初始化映射类实例
- 创建回话
- 持久化实例对象

1：创建连接

```
from sqlalchemy import create_engine
engine = create_engine("mysql://root:123456@localhost:3306/test?charset=utf8", echo = True)

# echo= True 会打印操作数据库的信息
```

Database_urls 的组成形式：
`dialect+driver://username:password@host:port/database`
举例：

1. `mysql://root:123456@localhost:3306/test`
2. `postgresql://scott:tiger@localhost/mydatabase`
3. `oracle://scott:tiger@127.0.0.1:1521/sidname`
4. `sqlite:///foo.db`
   具体参见：点我点我

2：声明映射文件

```
from sqlalchemy.ext.declarative import declarative_base
Base = declarative_base()
from sqlalchemy import Column, Integer, String
class User(Base):
    __tablename__ = 'users'
    id = Column(Integer, primary_key=True)
    name = Column(String(10))
    password = Column(String(10))

    def __repr__(self):
        return "<User(name='%s', name='%s', password='%s')>" % (self.name, self.name, self.password)

# 以上声明了表文件的形式：表名为：users，包含3列：id，name，password,且定义了数据类型
```

3：创建数据库表

```
User.metadata.create_all(engine)
# 运行后会在本地mysql数据库中创建这个数据库表

---
结果显示：
2016-05-09 20:52:38,062 INFO sqlalchemy.engine.base.Engine SHOW VARIABLES L
IKE 'sql_mode'
2016-05-09 20:52:38,091 INFO sqlalchemy.engine.base.Engine ()
2016-05-09 20:52:38,094 INFO sqlalchemy.engine.base.Engine SELECT DATABASE
()
2016-05-09 20:52:38,094 INFO sqlalchemy.engine.base.Engine ()
2016-05-09 20:52:38,114 INFO sqlalchemy.engine.base.Engine show collation w
here `Charset` = 'utf8' and `Collation` = 'utf8_bin'
2016-05-09 20:52:38,114 INFO sqlalchemy.engine.base.Engine ()
2016-05-09 20:52:38,116 INFO sqlalchemy.engine.base.Engine SELECT CAST('tes
t plain returns' AS CHAR(60)) AS anon_1
2016-05-09 20:52:38,117 INFO sqlalchemy.engine.base.Engine ()
2016-05-09 20:52:38,127 INFO sqlalchemy.engine.base.Engine SELECT CAST('tes
t unicode returns' AS CHAR(60)) AS anon_1
2016-05-09 20:52:38,128 INFO sqlalchemy.engine.base.Engine ()
2016-05-09 20:52:38,158 INFO sqlalchemy.engine.base.Engine SELECT CAST('tes
t collated returns' AS CHAR CHARACTER SET utf8) COLLATE utf8_bin AS anon_1
2016-05-09 20:52:38,158 INFO sqlalchemy.engine.base.Engine ()
2016-05-09 20:52:38,160 INFO sqlalchemy.engine.base.Engine DESCRIBE `users`
2016-05-09 20:52:38,160 INFO sqlalchemy.engine.base.Engine ()
2016-05-09 20:52:38,168 INFO sqlalchemy.engine.base.Engine ROLLBACK
2016-05-09 20:52:38,170 INFO sqlalchemy.engine.base.Engine
CREATE TABLE users (
    id INTEGER NOT NULL AUTO_INCREMENT,
    name VARCHAR(10),
    password VARCHAR(10),
    PRIMARY KEY (id)
)


2016-05-09 20:52:38,171 INFO sqlalchemy.engine.base.Engine ()
2016-05-09 20:52:38,655 INFO sqlalchemy.engine.base.Engine COMMIT
```

数据库test中生成表数据：

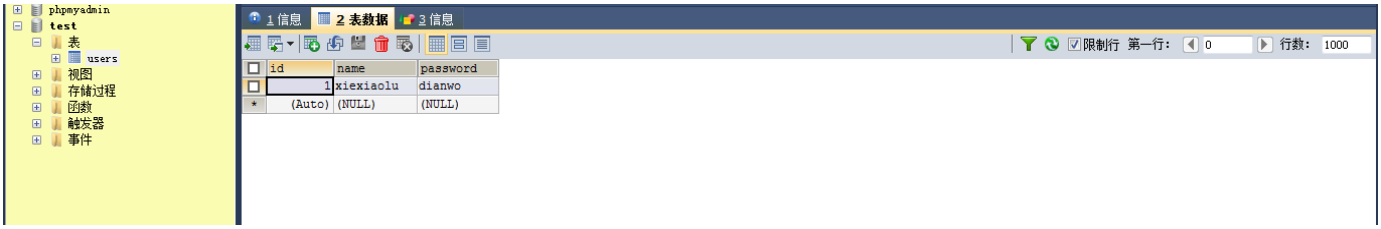
4：创建会话

```
from sqlalchemy.orm import sessionmaker
Session = sessionmaker(bind=engine)
session = Session()
```

5：持久化实例对象

```
ed_user = User(id=1, name='xiexiaolu', password='dianwo')
session.add(ed_user)
session.commit()
```

数据显示：插入一条数据




```
# 查询操作
user=session.query(User).filter(User.id ==1).one()
print(user.name)
print(user.password)
# output: id =1 的数据的 name 和 password
xiexiaolu
dianwo
```

创建数据表的其他方式

```
from sqlalchemy import Table, MetaData, create_engine

engine = create_engine("mysql+mysqlconnector://root:123456@localhost:3306/t
est")
metadata = MetaData()

t1 = Table('users',
metadata,
Column('id',INT, primary_key=True),
Column('name', String(20)),
Column('fullname', String(50)),
Column('password', String(20))
)
t2 = Table('address',
metadata,
Column('id',INT, primary_key = True),
Column('email_address',String(50), nullable=False),
Column('user_id', INT, ForeignKey('users.id'))
)
metadata.create_all(engine)
# 创建两个数据表分别为users 和 address
```

插入数据的其他方式

```
# 在建立的会话基础上执行sql语句
session.execute('insert into users values(2,"Bob","budian")')
session.commit()

# 使用映射类成员变量的数据
user = User(id="3", name="alice", password="hgf")
session.add(user)
session.commit()
```

查询操作

```
users = session.query(User).all()# 返回数据表所有数据
```

修改数据

```
# 在会话的基础上执行sql语句
session.execute('update addresses set user_id = 1 where id = 2')
session.commit()

# 使用映射类成员变量的数据
session.query(Address).filter(Address.id == 2).update({"user_id": 1})
```

3：总结

参考文献：
参考文献
参考文献
参考文献

> 一切为了精进.
> 新搭建的博客：Xie-xiaolu
> 预览界面：