一．PyMySQL安装

1.使用pypi

pypi即python package index

是python语言的软件仓库

1. 通过pip安装PyMySQL模块

[root@localhost ~]# mkdir ~/.pip/

[root@localhost ~]# vim ~/.pip/pip.conf

[global]

index-url=hKp://pypi.douban.com/simple/

[install]

trusted-host=pypi.douban.com

安装依赖包

[root@localhost packages]# yum install -y gcc

在线安装

[root@localhost packages]# pip3 install pymysql

本地安装

[root@localhost packages]# pip3 install PyMySQL-0.8.0.tar.gz

1. 安装数据库

#yum -y install mariadb-server

# systemctl start mariadb

# systemctl enable mariadb

# mysqladmin password tedu.cn

# mysql -uroot -ptedu.cn

1. 创建库

MariaDB [(none)]> create database tedu default charset='utf8';

MariaDB [(none)]> use tedu;

MariaDB [tedu]> CREATE TABLE departments (dep\_id INT, dep\_name VARCHAR(20), PRIMARY KEY(dep\_id));

MariaDB [tedu]> CREATE TABLE employees (emp\_id INT, emp\_name VARCHAR(20) NOT NULL, gender VARCHAR(6), email VARCHAR(50), dep\_id INT, PRIMARY KEY(emp\_id), FOREIGN KEY(dep\_id) REFERENCES departments(dep\_id));

MariaDB [tedu]> CREATE TABLE salary (auto\_id INT AUTO\_INCREMENT, date DATE, emp\_id INT, basic INT, awards INT, PRIMARY KEY(auto\_id), FOREIGN KEY(emp\_id) REFERENCES employees(emp\_id));

1. 连接数据库
2. 创建游标
3. 插入数据

import pymysql

conn = pymysql.connect( #连接数据库

host = '127.0.0.1',

port = 3306,

user = 'root',

passwd = 'tedu.cn',

db = 'tedu',

charset = 'utf8'

)

cursor = conn.cursor() #游标

insert\_dep1 = 'INSERT INTO departments VALUES(%s, %s)' #插入

#cursor.execute(insert\_dep1,('1','人事部'))

insert\_deps = [(2,'运维部'),(3,'开发部'),(4,'测试部')]

cursor.executemany(insert\_dep1,insert\_deps) #可执行多条语句

conn.commit() #对数据库修改(增删改)，必须要commit

cursor.close()

conn.close()

1. 查询数据

import pymysql

conn = pymysql.connect( #连接数据库

host = '127.0.0.1',

port = 3306,

user = 'root',

passwd = 'tedu.cn',

db = 'tedu',

charset = 'utf8'

)

cursor = conn.cursor()

query1 = 'SELECT \* FROM departments' #查

cursor.execute(query1)

r1 = cursor.fetchone() #fetchone一条语句

print(r1)

print('#' \* 20)

r2 = cursor.fetchmany(2) #fetchmany可查看多条语句

print(r2)

print('#' \* 20)

r3 = cursor.fetchall()

print(r3)

#如果r1、r2、r3同时执行的话，当r1执行完后，r2在执行时，默认会从r1执行完的后面执行，不会从开头执行

cursor.close()

1. 移动游标

MariaDB [tedu]> select \* from departments limit 1,2; #0开始

cursor = conn.cursor()

query1 = 'SELECT \* FROM departments'

cursor.execute(query1)

# cursor.scroll(2,mode='absolute') #以2开始，取后面的值，不包括2

# r1 = cursor.fetchall()

# print(r1)

cursor.scroll(1,mode='absolute') #以开头为起始点移动游标

cursor.fetchone()

cursor.scroll(1,mode='relative') #以当前位置为参考点移动游标

r2 = cursor.fetchall() #取出后续所有内容

print(r2)

cursor.close()

conn.close()

import pymysql

conn = pymysql.connect( #连接数据库

host = '127.0.0.1',

port = 3306,

user = 'root',

passwd = 'tedu.cn',

db = 'tedu',

charset = 'utf8'

)

cursor = conn.cursor() #创建游标，相当于打开文件返回文件对象

insert\_dep1 = 'UPDATE departments SET dep\_name=%s WHERE dep\_name=%s' #改

cursor.execute(insert\_dep1,('人力资源部','人事部'))

delete1 = 'DELETE FROM departments WHERE dep\_name=%s'

cursor.execute(delete1,('测试部',)) #删

conn.commit()

cursor.close()

conn.close()

1. SQLAlchemy基础

2.数据库对象管理

1)连接mysql

• 通过create\_engine实现数据库的连接

]> create database tarena default charset = 'utf8';

2)

from sqlalchemy import create\_engine

from sqlalchemy.ext.declarative import declarative\_base

from sqlalchemy import Column,Integer,String,Date,ForeignKey

from sqlalchemy.orm import sessionmaker

engine = create\_engine(

'mysql+pymysql://root:tedu.cn@localhost/tarena?charset=utf8',

#连接数据库,?charset=utf8可支持中文

encoding = 'utf8',echo=True

)

#mysql+pymysql://用户名：密码@主机名/库名

#/echo=True表示将日志输出到终端屏幕,默认为False

Base = declarative\_base() #创建ORM所需的基类

Session = sessionmaker(bind=engine)

class Deparments(Base):

\_\_tablename\_\_ = 'departments' #库中表名

dep\_id = Column(Integer,primary\_key=True) #integer是int类型，为主健，为字段

dep\_name = Column(String( ), unique=True) #dep\_name是字段

def \_\_str\_\_(self): #把对象转换成列表,把对象返回id和name

return '[%s: %s]' % (self.dep\_id,self.dep\_name)

class Employees(Base):

\_\_tablename\_\_ = 'employees'

emp\_id = Column(Integer,primary\_key=True)

emp\_name = Column(String(20),nullable=False)

gender = Column(String(6))

birth\_date = Column(Date)

email = Column(String(50))

dep\_id = Column(Integer,ForeignKey('departments.dep\_id'))

def \_\_str\_\_(self):

return '[%s: %s]' % (self.emp\_id,self.emp\_name)

class Salary(Base):

\_\_tablename\_\_ = 'salary'

auto\_id = Column(Integer, primary\_key=True)

date = Column(Date)

emp\_id = Column(Integer, ForeignKey('employees.emp\_id'))

basic = Column(Integer)

awards = Column(Integer)

if \_\_name\_\_ == '\_\_main\_\_':

Base.metadata.create\_all(engine) #没有时创建，已有不创建

from dbconn import Deparments,Session,Employees,Salary

# hr = Deparments(dep\_name='hr')

# print(hr.dep\_id) # 此时还没有在数据库中创建记录，所以是None

# print(hr.dep\_name)

#添加一个

# op = Deparments(dep\_id=2,dep\_name = '运维部')

# session = Session()

# session.add(op)

# session = Session() # 建立到数据库的会话连接

# session.add(hr) #真正向数据库写入记录

# session.commit()

# print(hr.dep\_id)

# session.close()

#添加多个

# dev = Deparments(dep\_id=3,dep\_name='开发部')

# qa = Deparments(dep\_id=4,dep\_name='测试部')

# session = Session()

# session.add\_all([dev,qa])

#

# session.commit()

# session.close()

#

# bob = Employees(emp\_id=1,emp\_name='Bob',gender='male',birth\_date='1995-08-23',email='bob@tedu.cn',dep\_id='1')

# john = Employees(emp\_id=2, emp\_name='John', gender='male', birth\_date='1992-06-2', email='john@tedu.cn', dep\_id='1')

# alice = Employees(emp\_id=3, emp\_name='Alice', gender='female', birth\_date='1997-02-2', email='alice@tedu.cn', dep\_id='2')

# jane = Employees(emp\_id=4, emp\_name='Jane', gender='female', birth\_date='1988-09-23', email='jane@tedu.cn', dep\_id='2')

# tom = Employees(emp\_id=5, emp\_name='Tom', gender='male', birth\_date='1994-04-6', email='tom@tedu.cn', dep\_id='3')

# session = Session()

# session.add\_all([bob,john,alice,jane,tom])

#

# session.commit()

# session.close()

三．SQLAlchemy进阶

1.查询操作

from dbconn import Deparments,Session,Salary

session = Session()

qset1 = session.query(Deparments).order\_by(Deparments.dep\_id)

#order\_by排序

print(qset1)

for dep in qset1:

print(dep)

for dep in qset1:

print('%s: %s' % (dep.dep\_id,dep.dep\_name))

###############################################

qset2 = session.query(Deparments.dep\_id,Deparments.dep\_name)

print(qset2)

for did,dname in qset2:

print(did,dname)

# ###########切片############################

qset3 = session.query(Deparments)[1:3] #返回的不是sql语句

print(qset3)

for dep in qset3:

print(dep.dep\_name)

##################过滤######################

qset4 = session.query(Deparments.dep\_name).\

filter(Deparments.dep\_id ==2 )

print(qset4)

for dep in qset4:

print(dep.dep\_name)

###############################################

qset5 = session.query(

Salary.date, Salary.emp\_id,Salary.basic + Salary.awards

)

print(qset5)

for date,emp\_id, sal in qset5:

print('%s:%s:%s' % (date,emp\_id,sal))

#####################常用过滤操作符###########

#in

qset6 = session.query(Deparments.dep\_id). \

filter(Deparments.dep\_name.in\_(['运维部','开发部']))

print(qset6)

for did in qset6:

print(did)

################################################not in 取反

qset7 = session.query(Deparments.dep\_id).filter \

(~Deparments.dep\_name.in\_(['运维部','开发部'])) #取反

print(qset7)

for did in qset7:

print(did)

######################常用过滤操作符##########

#多重条件and

from sqlalchemy import and\_,or\_

from dbconn import Employees

qset8 = session.query(Employees).\

filter(and\_(Employees.gender=='male',Employees.dep\_id ==3))

print(qset8)

for emp in qset8:

print(emp.emp\_name)

###############################################

#多重条件or

qset9 = session.query(Employees).\

filter(or\_(Employees.gender=='female',Employees.dep\_id==3))

print(qset9)

for emp in qset9:

print(emp.emp\_name)

#############查询对象返回值##################

#all()返回列表

# first()返回结果中的第一条记录

# one()取出所有记录,如果不是一条记录则抛出异常

# scalar()调用one(),返回第一列的值

qset10 = session.query(Deparments).order\_by(Deparments.dep\_id)

print(qset10.all()) #返回所有查到的结果，组成列表

print(qset10.first()) #只返回查询到的第一个结果

#print(qset10.one()) #报错，one返回的结果只有一个

qset11 = session.query(Deparments.dep\_id, Deparments.dep\_name).\

filter(Deparments.dep\_id ==1)

print(qset11.one())

print(qset11.scalar()) #调用one(),返回第一列

###############################################

#统计一共有部门

qset12 = session.query(Deparments).count()

print(qset12)

###############################################

#得到每个员工在那个部门，部门使用名字，不用ID

qset13 = session.query(Employees.emp\_name,Deparments.dep\_name).\

join(Deparments,Employees.dep\_id==Deparments.dep\_id)

print(qset13.all())

# 注意query()中先写Employees.emp\_name，join()中就要先用Departments

qset14 = session.query(Deparments.dep\_name,Employees.emp\_name).\

join(Employees,Employees.dep\_id==Deparments.dep\_id)

print(qset14)

#deparments是Employees外健，一般deparments写在前面，可以更好的看出来，如果两者位置换了过来，返回的值相当于是对象，不容易看出来

1. 修改操作

###############################################

#修改操作

# 第一种修改记录的方法，通过查询语句的update方法

hr = session.query(Deparments).filter(Deparments.dep\_name=='hr')

print(hr)

hr.update({'dep\_name': '人力资源部'})

session.commit()

session.close()

###############################################hr = session.query(Deparments).get(1) #获取主健是1的实列

print(hr)

hr.dep\_name = '人事部'

session.commit()

session.close()

################################################删除ID号为5的员工记录

tom = session.query(Employees).get(5)

session.delete(tom)

session.commit()

session.close()