**aliased的函数\_高级查询之别名使用：**

当多表关联查询的时候，

有时候同一个表要用到多次，

这时候用别名就可以方便的解决命名冲突的问题了

**代码演示：**

|  |
| --- |
| **from** sqlalchemy **import** create\_engine,Column,Integer,Float,Boolean,DECIMAL,Enum,\  Date,DateTime,Time,String,Text,func,or\_,and\_,ForeignKey,Table  **from** sqlalchemy.dialects.mysql **import** LONGTEXT  **from** sqlalchemy.ext.declarative **import** declarative\_base  **from** sqlalchemy.orm **import** sessionmaker,relationship,backref  **import** random,time  **from** datetime **import** datetime  HOSTNAME = **'127.0.0.1'**  PORT = **'3306'**  DATABASE = **'first\_sqlalchemy'**  USERNAME = **'root'**  PASSWORD = **'root'**  DB\_URI =**"mysql+pymysql://{username}:{password}@{host}:{port}/{db}?charset=utf8"**.format(username=USERNAME,password=PASSWORD,host=HOSTNAME,port=PORT,db=DATABASE)  engine = create\_engine(DB\_URI)  Base = declarative\_base(engine)  session = sessionmaker(engine)()  **class** User(Base):  \_\_tablename\_\_ = **'user'**  id = Column(Integer,primary\_key=**True**,autoincrement=**True**)  uname = Column(String(50),nullable=**False**)  city = Column(String(50),nullable=**False**)  age = Column(Integer,default=0)  **def** \_\_repr\_\_(self):  **return "<User(username: %s)>"** % self.uname  **def** add\_data():  Base.metadata.drop\_all()  Base.metadata.create\_all()  user1 = User(uname=**'一哥'**,city=**"贵阳"**,age=18)  user2 = User(uname=**'王二'**,city=**"贵阳"**,age=18)  user3 = User(uname=**'张三'**,city=**"北京"**,age=18)  user4 = User(uname=**'赵四'**,city=**"贵阳"**,age=20)  session.add\_all([user1,user2,user3,user4])  session.commit()  **from** sqlalchemy.orm **import** aliased  a1 = aliased(User)  a2 = aliased(User)  #别名  **def** oper1():  **for** uname, age1, age2 **in** session.query(User.uname,a1.age, a2.age).join(  a1,User.id==a1.id).join(a2,a1.id==a2.id).all():  print(uname, age1, age2)  **if** \_\_name\_\_ == **'\_\_main\_\_'**:  # add\_data()  oper1() |