**ORM关系之多对多：**

**1.** 多对多的关系需要通过一张中间表来绑定他们之间的关系。

**2.** 先把两个需要做多对多的模型定义出来

**3.** 使用Table定义一个中间表，中间表一般就是包含两个模型的外键字段就可以了，并且让他们两个来作为一个“复合主键”。

**4.** 在两个需要做多对多的模型中随便选择一个模型，定义一个relationship属性，来绑定三者之间的关系，在使用relationship的时候，需要传入一个secondary=中间表对象名

**代码演示如下：**

|  |
| --- |
| **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  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)()  #表3 中间表  news\_tag = Table(  **"news\_tag"**,  Base.metadata,  Column(**"news\_id"**,Integer,ForeignKey(**"news.id"**),primary\_key=**True**),  Column(**"tag\_id"**,Integer,ForeignKey(**"tag.id"**),primary\_key=**True**)  )  #表1  **class** News(Base):  \_\_tablename\_\_ = **'news'**  id = Column(Integer,primary\_key=**True**,autoincrement=**True**)  title = Column(String(50),nullable=**False**)  #产生关系 写法1  # tags = relationship("Tag",backref="newss",secondary=news\_tag)  **def** \_\_repr\_\_(self):  **return "<News(title:%s)>"** % self.title  #表2  **class** Tag(Base):  \_\_tablename\_\_ = **'tag'**  id = Column(Integer, primary\_key=**True**, autoincrement=**True**)  name = Column(String(50), nullable=**False**)  # 产生关系 写法2  newss = relationship(**"News"**,backref=**"tags"**,secondary=news\_tag)  **def** \_\_repr\_\_(self):  **return "<Tag(name:%s)>"** % self.name  # 1. 先把两个需要做多对多的模型定义出来  # 2. 使用Table定义一个中间表，中间表一般就是包含两个模型的外键字段就可以了，并且让他们两个来作为一个“复合主键”。  # 3. 在两个需要做多对多的模型中随便选择一个模型，定义一个relationship属性，来绑定三者之间的关系，  # 4. 在使用relationship的时候，需要传入一个secondary=中间表对象名。  Base.metadata.drop\_all()  Base.metadata.create\_all()  #添加数据的好处  news1 = News(title=**"世界第一"**)  news2 = News(title=**"世界第二"**)  tag1 = Tag(name=**'要闻'**)  tag2 = Tag(name=**'娱乐'**)  news1.tags.append(tag1)  news1.tags.append(tag2)  news2.tags.append(tag1)  news2.tags.append(tag2)  session.add(news1)  session.add(news2)  session.commit()  #查询数据的好处  news3 = session.query(News).first()  print(news3.tags)  tag = session.query(Tag).first()  print(tag.newss) |