**limit、offset、slice使用：**

**1.** limit：可以限制查询的时候只查询前几条数据。 属top-N查询

**2.** offset：可以限制查找数据的时候过滤掉前面多少条。可指定开始查询时的偏移量。

**3.** 切片：可以对Query对象使用切片操作，来获取想要的数据。

可以使用`slice(start,stop)`方法来做切片操作。

也可以使用`[start:stop]`的方式来进行切片操作。

一般在实际开发中，中括号的形式是用得比较多的。

**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** Article(Base):  \_\_tablename\_\_ = **'article'**  id = Column(Integer,primary\_key=**True**,autoincrement=**True**)  title = Column(String(50),nullable=**False**)  create\_time = Column(DateTime,default=datetime.now)  **def** \_\_repr\_\_(self):  **return "<Article(title: %s)>"** % self.title  **def** add\_data():  Base.metadata.drop\_all()  Base.metadata.create\_all()  **for** x **in** range(100):  title = **"title %s"** % x  article = Article(title=title)  session.add(article)  session.commit()  **def** oper1():  # articles = session.query(Article).all()  #limit：可以限制每次查询的时候只查询前几条数据。 属top-N查询  articles = session.query(Article).limit(10).all()  print(articles)  **def** oper2():  #offset：可以限制查找数据的时候过滤掉前面多少条。可指定开始查询时的偏移量。  # articles = session.query(Article).offset(10).limit(10).all()  #场景选型：查看最新的10条新闻  articles = session.query(Article).order\_by(Article.id.desc()).limit(10).all()  print(articles)  #实现分页  **from** sqlalchemy.orm.query **import** Query  **def** oper3():  articles = session.query(Article).order\_by(Article.id.desc()).slice(0,10).all()  print(articles)  **def** oper4():  articles = session.query(Article).order\_by(Article.id.desc())[0:10]  print(articles)  **if** \_\_name\_\_ == **'\_\_main\_\_'**:  # add\_data()  # oper1()  # oper2()  # oper3()  oper4() |