**query函数可传递的参数一共有3种：**

**1.** 模型名。指定查找这个模型中所有的属性（对应查询表为全表查询）。

**2.** 模型中的属性。可以指定只查找某个模型的其中几个属性。

**3.** 聚合函数。

\* func.count：统计行的数量。

\* func.avg：求平均值。

\* func.max：求最大值。

\* func.min：求最小值。

\* func.sum：求和。

`func`上，其实没有任何聚合函数。但是因为他底层做了一些魔术，只要mysql中有的聚合函数，都可以通过func调用。

**代码演示如下：**

|  |
| --- |
| **from** sqlalchemy **import** create\_engine,Column,Integer,String,Float,Enum,Boolean,DECIMAL,Text,\  Date,DateTime,Time,func  **from** sqlalchemy.ext.declarative **import** declarative\_base  **from** sqlalchemy.dialects.mysql **import** LONGTEXT  **from** sqlalchemy.orm **import** sessionmaker  **import** random  **import** enum  **from** datetime **import** date  **from** datetime **import** datetime  **from** datetime **import** time  #准备数据库的一堆信息 ip port user pwd 数据库的名称 按要求组织格式  HOSTNAME = **'127.0.0.1'**  PORT = **'3306'**  DATABASE = **'first\_sqlalchemy'**  USERNAME = **'root'**  PASSWORD = **'root'**  #dialect+driver://username:password@host:port/database?charset=utf8  #按照上述的格式来 组织数据库信息  DB\_URI =**"mysql+pymysql://{username}:{password}@{host}:{port}/{db}?charset=utf8"**.\  format(username=USERNAME,password=PASSWORD,host=HOSTNAME,port=PORT,db=DATABASE)  #创建数据库引擎  engine = create\_engine(DB\_URI)  #创建会话对象  session = sessionmaker(engine)()  #创建一个ORM模型  Base = declarative\_base(engine)  **class** News(Base):  \_\_tablename\_\_=**'news'**  id = Column(Integer,primary\_key=**True**,autoincrement=**True**)  title =Column(String(50),nullable=**False**)  price = Column(Integer)  **def** \_\_repr\_\_(self):  **return "<title:%s,price:%s>"**%(self.title,self.price)  # Base.metadata.drop\_all()  # Base.metadata.create\_all()  #新增测试数据  # for x in range(1,6):  # a = News(title="标题%s"%x,price =random.randint(1,100))  # session.add(a)  # session.commit()  #query()函数中能写什么参数 3种  #1.模型名  # results = session.query(News).all()  # print(results)  #2.模型名中的属性。 返回的列表中的元素是 元组类型数据  # results = session.query(News.title,News.price).all()  # print(results)  #3.mysql聚合函数  # r = session.query(func.count(News.id)).first()  # print(r)  # r = session.query(func.max(News.price)).first()  # print(r)  # r = session.query(func.min(News.price)).first()  # print(r)  # r = session.query(func.avg(News.price)).first()  # print(r)  r = session.query(func.sum(News.price)).first()  print(r) |