**分组group\_by和过滤分组having:**

**group\_by：**

根据某个字段进行分组。如想要根据年龄进行分组，来统计每个分组分别有多少人

r = session.query(User.age,func.count(User.id)).group\_by(User.age).all()

**having：**

having是对分组查找结果作进一步过滤。如只想要看未成年人的人数，

那么可以首先对年龄进行分组统计人数，然后再对分组进行having过滤。

r = session.query(User.age,func.count(User.id)).group\_by(User.age).having(User.age < 18).all()

**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**)  age = Column(Integer,default=0)  gender = Column(Enum(**"男"**,**"女"**,**"秘密"**),default=**"男"**)  **def** add\_data():  Base.metadata.drop\_all()  Base.metadata.create\_all()  user1 = User(uname=**'李一'**,age=17,gender=**'男'**)  user2 = User(uname=**'李二'**,age=17,gender=**'男'**)  user3 = User(uname=**"张三"**,age=18,gender=**'女'**)  user4 = User(uname=**"张四"**,age=19,gender=**'女'**)  user5 = User(uname=**"莫莫"**,age=20,gender=**'男'**)  session.add\_all([user1,user2,user3,user4,user5])  session.commit()  #查询 每个年龄的人数  **def** oper1():  # user = session.query(User.age,func.count(User.id)).group\_by(User.age)  # print(user) #sql语句  user = session.query(User.age, func.count(User.id)).group\_by(User.age).all()  print(user) # 列表  #查询 每个年龄的人数，要求排除未成年人  **def** oper2():  user = session.query(User.age, func.count(User.id)).group\_by(User.age).having(User.age>=18).all()  print(user) # 列表  #查询 每个年龄的人数，要求未成年人  **def** oper3():  user = session.query(User.age, func.count(User.id)).group\_by(User.age).having(User.age<18).all()  print(user) # 列表  **if** \_\_name\_\_ == **'\_\_main\_\_'**:  # add\_data()  # oper1()  # oper2()  oper3() |