03-模型进阶

多表关联

一对多

```
# 一对多
class Grade(db.Model):
    id = db.Column(db.Integer, primary_key=True, autoincrement=True)
    name = db.Column(db.String(16))
    # 定义班级标的一对多关系，不是字段，Student为学生表模型， backref为反向查找名称
    students = db.relationship('Student', backref='grade1', lazy=True)

class Student(db.Model):
    id = db.Column(db.Integer, primary_key=True, autoincrement=True)
    name = db.Column(db.String(16))
    age = db.Column(db.Integer, default=1)
    # 创建外键,关联到班级表的主键，实现一对多关系，班级表中也要有对应操作
    grade = db.Column(db.Integer, db.ForeignKey(Grade.id))

    查:
        # 获取学生的所在班级信息(反向)
    stu = Student.query.get(stuid)
      grade = stu.stus

      # 获取班级的所有学生 (正向)
        grade = Grade.query.get(gradeid)
        students = grade.students
    删:
            # 删除班级后， 学生的grade字段会变为null
            grade = Grade.query.get(id)
            db.session.delete(grade)
            db.session.commit()
```

多对多

用户收藏电影,一个用户可以收藏多部电影，一部电影可以被不同的用户收藏，是一个多对多关系.

```
# 中间表(不是类)
collects = db.Table('collects',
    # user_id为表字段名称，user.id为外键表的id
    db.Column('user_id', db.Integer, db.ForeignKey('user.id'), primary_key=True),
    db.Column('movie_id', db.Integer, db.ForeignKey('movie.id'), primary_key=True)
)

class Movie(db.Model):
    id = db.Column(db.Integer, primary_key=True, autoincrement=True)
```

```
    name = db.Column(db.String(200))

class User(db.Model):
    id = db.Column(db.Integer, primary_key=True, autoincrement=True)
    name = db.Column(db.String(16))
    age = db.Column(db.Integer, default=1)
    # 多对多  关联的学生表格的模型,  中间表的名称,  反向查找
    movies = db.relationship('Movie',  backref='users',
                             secondary=collects, lazy='dynamic')

    lazy属性:
        懒加载,可以延迟在使用关联属性的时候才建立关联
        lazy='dynamic': 会返回一个query对象(查询集), 可以继续使用其他查询方法, 如all().
        lazy='select': 首次访问到属性的时候, 就会全部加载该属性的数据.
          lazy='joined': 在对关联的两个表进行join操作, 从而获取到所有相关的对象
          lazy=True:  返回一个可用的列表对象,同select

查:
  # 查询用户收藏的所有电影
   user = User.query.get(id)
    movies = user.movies

  # 查询电影被哪些用户收藏
    movie = Movie.query.get(id)
    users = movie.users

删:
  # 中间表的数据会被级联删除
    movie = Movie.query.get(id)
    db.session.delete(movie)
    db.session.commit()

增:
  # 用户收藏电影
    user = User.query.get(id)
    movie = Movie.query.get(id)
    user.movies.append(movie)
    db.session.commit()
```

作业:图书馆项目

```
创建一个项目, 用来说明出版社, 书籍和作者的关系。
假定关系: 作者: 书籍 => 1:n   (一本书由一个作者完成,  一个作者可以创作多本书)
    出版社: 书籍 => n:n   (一个出版社可以出版多本书,  一本书可以由多个出版社出版)
要求:
  1.在书籍的book_index.html中有一个"查看所有书籍"的超链接按钮, 点击进入书籍列表book_list.html页面.
```

2.在书籍的book_list.html中显示所有书名，点击书名可以进入书籍详情book_detail.html
3.在书籍book_detail.html中可以点击该书的作者和出版社，进入作者详情的author_detail.html和出版社详情的publisher_detail.html页面

```
# 作者
class Author(db.Model):
    id = db.Column(db.Integer, primary_key=True, autoincrement=True)
    name = db.Column(db.String(20), unique=True)
    age = db.Column(db.Integer, default=1)
    sex = db.Column(db.Boolean, default=True)
    email = db.Column(db.String(200))
    # 关系
    books = db.relationship('Book', backref='author', lazy='dynamic')

# 书籍
class Book(db.Model):
    id = db.Column(db.Integer, primary_key=True, autoincrement=True)
    title = db.Column(db.String(100), unique=True)
    date = db.Column(db.DateTime)
    # 1对多，外键
    author = db.Column(db.Integer, db.ForeignKey(Author.id))

# 中间表（书籍-出版社）
book_publisher = db.Table('book_publisher',
    db.Column('book_id', db.Integer,
              db.ForeignKey('book.id'), primary_key=True),
    db.Column('publisher_id', db.Integer,
              db.ForeignKey('publisher.id'), primary_key=True)
)

# 出版社
class Publisher(db.Model):
    id = db.Column(db.Integer, primary_key=True, autoincrement=True)
    name = db.Column(db.String(20), unique=True)
    address = db.Column(db.String(200))
    city = db.Column(db.String(100))
    province = db.Column(db.String(100))
    country = db.Column(db.String(100))
    website = db.Column(db.String(100))
    # 多对多，关联book表
    books = db.relationship('Book', backref='publishers',
                            secondary=book_publisher, lazy='dynamic')
```