登录

2019年05月04日 阅读 412

关注

Flask框架从入门到精通之模型关系（十七）

> 知识点：1、模型关系参照

一、概况

在数据库中，我们知道数据关系大概有如下几种：一对一、一对多、多对多、自关联等。我们模型已经描述过了一对多，那么下面我们在用模型把其它关系也写出来。

关系

- 一对一模型 案例：一篇文章只对应一个内容

复制代码

```
# 文章模型
class Article(db.Model):
    # 表名
    __tablename__ = 'tbl_article'

    # 数据库真正存在的字段
    id = db.Column(db.Integer, primary_key=True)  # 主键
    title = db.Column(db.String(128), unique=True)  # 名字

    # 方便查找, 数据并不存在的字段
    content = db.relationship('Acontent', backref='article', uselist=False) #一对一需要把uselist设置为False


# 内容模型
class Acontent(db.Model):
    # 表名
    __tablename__ = 'tbl_acontent'

    # 数据库真正存在的字段
    id = db.Column(db.Integer, primary_key=True)  # 主键
    content = db.Column(db.Text(4000))  # 名字
    article_id = db.Column(db.Integer, db.ForeignKey('tbl_article.id'))
```

tbl_article

id	title
1	Flask入门

id	title
2	Flask模型

tbl_acontent

id	content	article_id
1	Flask是轻量级框架	1
2	Flask是模型入门	2

- 一对多模型 案例：一个分类下有很多文章

复制代码

```
# 分类模型
class Category(db.Model):
    # 表名
    __tablename__ = 'tbl_category'

    # 数据库真正存在的字段
    id = db.Column(db.Integer, primary_key=True)  # 主键
    name = db.Column(db.String(32), unique=True)  # 名字
    # 方便查找, 数据并不存在的字段
    article = db.relationship('Article', backref='category')


# 文章模型
class Article(db.Model):
    # 表名
    __tablename__ = 'tbl_article'

    # 数据库真正存在的字段
    id = db.Column(db.Integer, primary_key=True)  # 主键
    title = db.Column(db.String(128), unique=True)  # 名字
    category_id = db.Column(db.Integer, db.ForeignKey('tbl_category.id'))  # 分类id
    # 方便查找, 数据并不存在的字段
    content = db.relationship('Acontent', backref='article', uselist=False)  # 一对一需要把uselist设置为False
```

tbl_category

id	name
1	框架
2	模型

tbl_article

id	title	category_id
1	Flask是轻量级框架	1
2	Flask是模型入门	2

id	title	category_id
3	Flask是模型查询	2

- **多对多模型 案例：一个标签对应很多文章，一篇文章也对应很多标签**

复制代码

```
# 辅助表
tbl_tags = db.Table('tbl_tags',
                db.Column('tag_id', db.Integer, db.ForeignKey('tbl_tag.id')),
                db.Column('article_id', db.Integer, db.ForeignKey('tbl_article.id'))
                )

# 标签模型
class Tag(db.Model):
    # 表名
    __tablename__ = 'tbl_tag'

    # 数据库真正存在的字段
    id = db.Column(db.Integer, primary_key=True)  # 主键
    name = db.Column(db.String(32), unique=True)  # 名字


# 文章模型
class Article(db.Model):
    # 表名
    __tablename__ = 'tbl_article'

    # 数据库真正存在的字段
    id = db.Column(db.Integer, primary_key=True)  # 主键
    title = db.Column(db.String(128), unique=True)  # 名字
    category_id = db.Column(db.Integer, db.ForeignKey('tbl_category.id'))  # 分类id
    # 方便查找, 数据并不存在的字段
    content = db.relationship('Acontent', backref='article', uselist=False)  # 一对一需要把uselist设置为False
    tags = db.relationship('Tag', secondary=tbl_tags, backref='articles')
```

tbl_tag

id	name
1	python
2	后端
3	mysql

tbl_article

id	title	category_id
1	Flask是轻量级框架	1
2	Flask是模型入门	2
3	Flask是模型查询	2

tbl_tags

tag_id	article_id
1	1
1	1
2	1
2	2

- **自关联模型 案例：地区**

复制代码

```
# 地区模型
class Area(db.Model):
    # 表名
    __tablename__ = "tbl_area"
    # 数据库真正存在的字段
    id = db.Column(db.Integer, primary_key=True)  # 主键
    name = db.Column(db.Text, nullable=False)  # 地区名字
    parent_id = db.Column(db.Integer, db.ForeignKey("tbl_area.id"))  # 父评论id

    # 方便查找, 数据并不存在的字段
    parent = db.relationship("Area", remote_side=[id])  # 自关联需要加remote_side
```

id	name	parent_id
1	北京市	Null
2	黑龙江	Null
3	哈尔滨	2
4	南岗区	3
5	北京市	1
6	朝阳区	5

欢迎关注我的公众号：

文章分类

后端

文章标签

Flask

相关推荐