day82 drf

内容回顾

1. restful规范

   ```
   - URL中一般用名词：
       http://www.luffycity.com/article/ (面向资源编程，网络上东西都视为资源)
   - 根据请求不同做不同操作：GET/POST/PUT/DELTE/PATCH
   - 筛选条件，在URL参数中进行传递：
       http://www.luffycity.com/article/?page=1&category=1

   一般传输的数据格式都是JSON
   ```

2. drf组件

   ```
   ...
   ```

3. 潜规则：类的约束
4. drf的配置

   ```
   setting.py

   REST_FRAMEWORK = {
       ...
   }
   ```

5. 分页
   - page
   - offset

今日概要

- 作业：呼啦圈
- 筛选
- 版本管理（源码实现）
- 试图（源码）

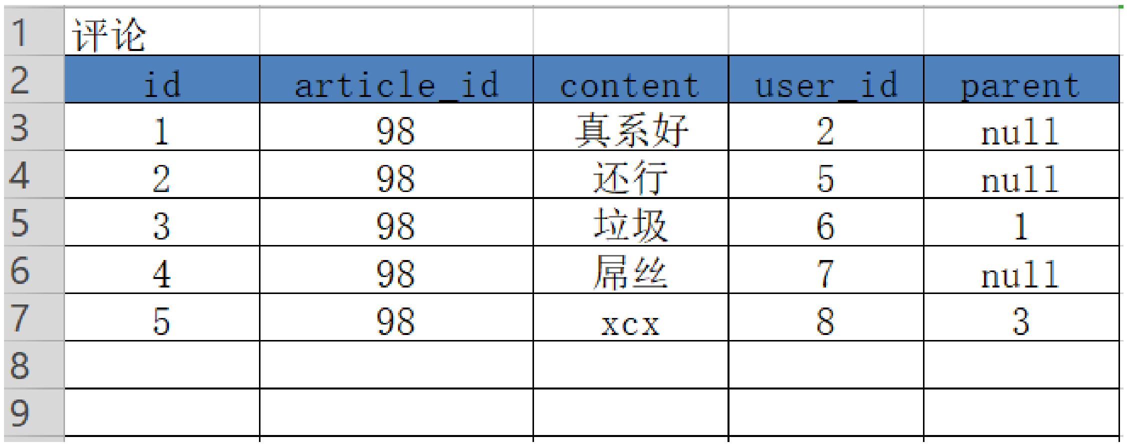
今日详细

1.作业：呼啦圈

1.1 表结构设计

- 不会经常变化的值放在内存：choices形式，避免跨表性能低。

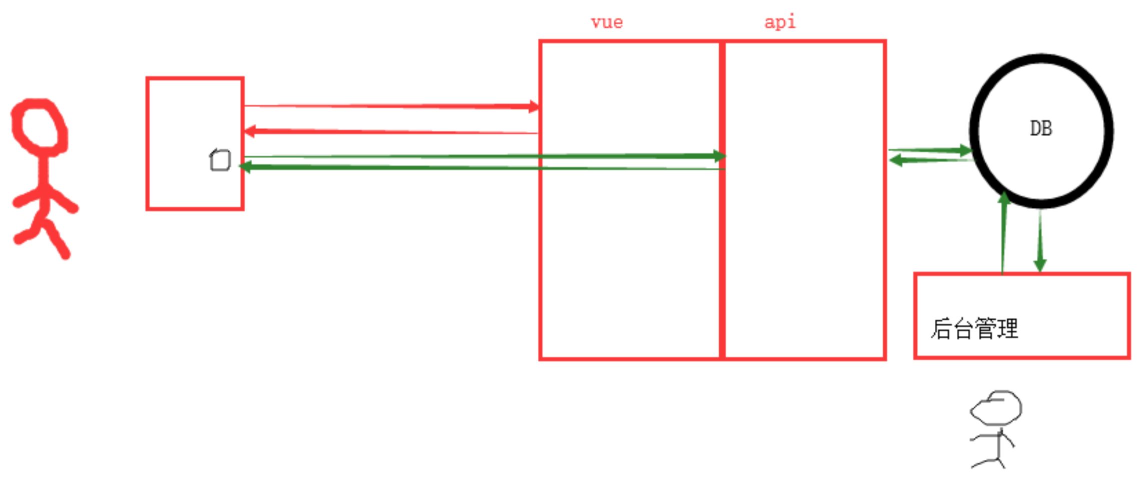
- 分表：如果表中列太多/大量内容可以选择水平分表
- 表自关联



	评论				
2	id	article_id	content	user_id	parent
3	1	98	真系好	2	null
4	2	98	还行	5	null
5	3	98	垃圾	6	1
6	4	98	屌丝	7	null
7	5	98	xcx	8	3
8					
9					

```
from django.db import models

class UserInfo(models.Model):
    """ 用户表 """
    username = models.CharField(verbose_name='用户名',max_length=32)
    password = models.CharField(verbose_name='密码',max_length=64)


class Article(models.Model):
    """ 文章表 """
    category_choices = (
        (1,'咨询'),
        (2,'公司动态'),
        (3,'分享'),
        (4,'答疑'),
        (5,'其他'),
    )
    category = models.IntegerField(verbose_name='分类',choices=category_choices)
    title = models.CharField(verbose_name='标题',max_length=32)
    image = models.CharField(verbose_name='图片路径',max_length=128) #
/media/upload/....
    summary = models.CharField(verbose_name='简介',max_length=255)

    comment_count = models.IntegerField(verbose_name='评论数',default=0)
    read_count = models.IntegerField(verbose_name='浏览数',default=0)

    author = models.ForeignKey(verbose_name='作者',to='UserInfo')
    date = models.DateTimeField(verbose_name='创建时间',auto_now_add=True)

class ArticleDetail(models.Model):
    article = models.OneToOneField(verbose_name='文章表',to='Article')
    content = models.TextField(verbose_name='内容')


class Comment(models.Model):
    """ 评论表 """
    article = models.ForeignKey(verbose_name='文章',to='Article')
    content = models.TextField(verbose_name='评论')
    user = models.ForeignKey(verbose_name='评论者',to='UserInfo')
    # parent = models.ForeignKey(verbose_name='回复',to='self',
null=True,blank=True)
```

1.2 系统结构（CMS）



1.3 功能实现

1.3.1 增加文章（可以不写）

1.3.2 文章列表

1.3.3 文章详细

1.3.4 评论列表

- 查看评论列表
  访问时：`http://127.0.0.1:8000/hg/comment/?article=2`
- 添加评论

  ```
  http://127.0.0.1:8000/hg/comment/

  {
      article:1,
      content:'xxx'
  }
  ```

  ```
  http://127.0.0.1:8000/hg/comment/?article=1

  {
      content:'xxx'
  }
  ```

2. 筛选

案例：在文章列表时候，添加筛选功能。

```
全部：http://127.0.0.1:8000/hg/article/
筛选：http://127.0.0.1:8000/hg/article/?category=2
```

```
class ArticleView(APIView):
    """ 文章视图类 """

    def get(self,request,*args,**kwargs):
        """ 获取文章列表 """
        pk = kwargs.get('pk')
        if not pk:
            condition = {}
            category = request.query_params.get('category')
            if category:
                condition['category'] = category
            queryset = models.Article.objects.filter(**condition).order_by('-
date')

            pager = PageNumberPagination()
            result = pager.paginate_queryset(queryset,request,self)
            ser = ArticleListSerializer(instance=result,many=True)
            return Response(ser.data)
        article_object = models.Article.objects.filter(id=pk).first()
        ser = PageArticleSerializer(instance=article_object,many=False)
        return Response(ser.data)
```

drf的组件：内置了筛选的功能

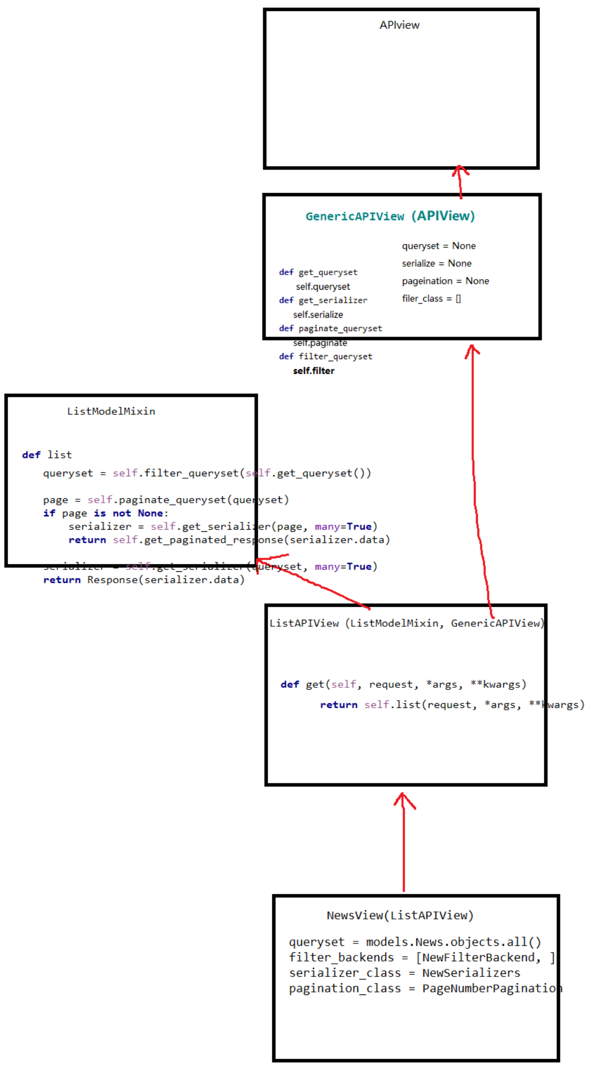
```
from django.shortcuts import render
from rest_framework.views import APIView
from rest_framework.response import Response
from . import models

from rest_framework.filters import BaseFilterBackend

class MyFilterBackend(BaseFilterBackend):

    def filter_queryset(self, request, queryset, view):
        val = request.query_params.get('cagetory')
        return queryset.filter(category_id=val)


class IndexView(APIView):

    def get(self,request,*args,**kwargs):
        # http://www.xx.com/cx/index/
        # models.News.objects.all()

        # http://www.xx.com/cx/index/?category=1
        # models.News.objects.filter(category=1)

        # http://www.xx.com/cx/index/?category=1
        # queryset = models.News.objects.all()
        # obj = MyFilterBackend()
        # result = obj.filter_queryset(request,queryset,self)
        # print(result)

        return Response('...')
```

3.视图

- APIView，感觉没提供功能。
- GenericAPIView，桥梁，内部定义：get_queryset/get_serilizer/get_page...

  ```
  ListAPIView,CreateAPIView,RetrieveAPIView,UpdateAPIView,DestroyAPIView
  ```

```
class TagSer(serializers.ModelSerializer):
    class Meta:
        model = models.Tag
        fields = "__all__"

class TagView(ListAPIView,CreateAPIView):
    queryset = models.Tag.objects.all()
    serializer_class = TagSer

    def get_serializer_class(self):
        # self.request
        # self.args
        # self.kwargs
        if self.request.method == 'GET':
            return TagSer
        elif self.request.method == 'POST':
            return OtherTagSer
    def perform_create(self,serializer):
        serializer.save(author=1)

class TagDetailView(RetrieveAPIView,UpdateAPIView,DestroyAPIView):
    queryset = models.Tag.objects.all()
    serializer_class = TagSer
```



```
GenericAPIView (APIView)

queryset = None
serialize = None
pageination = None
filer_class = []

def get_queryset
    self.queryset
def get_serializer
    self.serialize
def paginate_queryset
    self.paginate
def filter_queryset
    self.filter
```

```
ListModelMixin

def list
    queryset = self.filter_queryset(self.get_queryset())

    page = self.paginate_queryset(queryset)
    if page is not None:
        serializer = self.get_serializer(page, many=True)
        return self.get_paginated_response(serializer.data)

    serializer = self.get_serializer(queryset, many=True)
    return Response(serializer.data)
```

```
ListAPIView (ListModelMixin, GenericAPIView)

 def get(self, request, *args, **kwargs)
       return self.list(request, *args, **kwargs)
```

```
NewsView(ListAPIView)

queryset = models.News.objects.all()
filter_backends = [NewFilterBackend, ]
serializer_class = NewSerializers
pagination_class = PageNumberPagination
```

用户发来请求：
  get请求 http://127.0.0.1:8000/cx/news/
  1. 去NewView中找get方法
        obj = NewView()
        obj.get(....)

作业：实现呼啦圈（参考今日答案）

```
ListAPIView,CreateAPIView,RetrieveAPIView,UpdateAPIView,DestroyAPIView
+
定义钩子方法
```