1. 修改数据库的步骤：
   1. 编辑 models.py 文件，改变模型。
   2. 运行 python manage.py makemigrations 为模型的改变生成迁移文件，此时改动还没有在数据库中生效。
   3. 运行 python manage.py migrate 来应用数据库迁移，此时数据库将发生改动。
2. 时区设置可以参考如下网站：

<https://en.wikipedia.org/wiki/List_of_tz_database_time_zones>

使用“TZ\*”那一列的名称填入”TIME\_ZONE”中，如TIME\_ZONE = 'Asia/Shanghai'表示北京时间。

1. URL中的name参数的意义：

urlpatterns = patterns('',

url(r'^article$','news\_index' ，name="news\_index"),

)

Templates里的html为:

<a href="{*%u*rl news\_index%}">资讯</a>

在模板中直接使用函数就能使a标签直接跳转到对应的网址，不需要一个个修改。

1. 通用视图

分类：

* 展示对象列表（比如所有用户，所有文章）- **ListView**
* 展示某个对象的详细信息（比如用户资料，比如文章详情) - **DetailView**
* 通过表单创建某个对象（比如创建用户，新建文章）- **CreateView**
* 通过表单更新某个对象信息（比如修改密码，修改文字内容）- **UpdateView**
* 用户填写表单后转到某个完成页面 - **FormView**
* 删除某个对象 - **DeleteView**

视图中常用方法：

* get\_queryset()

如果没有在视图中重新该方法的话默认将返回全部的model查询结果，即不添加过滤条件（此时需要填写model属性）。重新该方法后只返回该方法查询到的数据（此时不需要填写model属性）。

class IndexView(ListView):  
  
    template\_name = 'blog/article\_list.html'  
    context\_object\_name = 'latest\_articles'  
  
    def get\_queryset(self):  
        return Article.objects.filter(author = self.request.user).order\_by('-pub\_date')

* **get\_context\_data()**

可以用于给模板传递模型以外的内容或参数，非常有用。例如现在的时间并不属于Article模型。如果你想把现在的时间传递给模板，你还可以通过重写get\_context\_data方法。

class IndexView(ListView):  
  
    queryset = Article.objects.all() #等同于model = Article  
    template\_name = 'blog/article\_list.html'  
    context\_object\_name = 'latest\_articles'  
  
    def get\_context\_data(self, \*\*kwargs):  
        context = super().get\_context\_data(\*\*kwargs)  
        context['now'] = timezone.now() #只有这行代码有用  
        return context

* **get\_object()**

重新该函数可以在获取数据库对象阶段对数据进行处理，比如你希望一个用户只能查看或编辑自己发表的文章对象，当用户查看别人的对象时，返回http 404错误。这时候你可以通过更具体的get\_object()方法来返回一个更具体的对象。代码如下:

class ArticleDetailView(DetailView):  
  
    queryset = Article.objects.all().order\_by("-pub\_date")  
    template\_name = 'blog/article\_detail.html'  
    context\_object\_name = 'article'  
  
    def get\_object(self, queryset=None):  
        obj = super().get\_object(queryset=queryset)  
        if obj.author != self.request.user:  
            raise Http404()  
        return obj