

HuanBlog开发注解-Part2

models.py与数据库

温故知新

数据类型字段

- `CharField()` 字符字段 适用小文本 比如用户名、账号、密码等字段
- `TextField()` 文本字段 适用于大量文本 比如发布博客文章
- `DateField()` 日期字段 通过now方法自动获取当前时间 适用于博客文章时间戳
- `IntegerField()`，`int`类型数字字段
- `URLField()`，URL地址字段 可以是网站网址也可以是路径地址
- `BooleanField()` 布尔逻辑字段 比如是否被禁言 1为真 被禁言 0为假 未禁言
- `FileField()` - 文件字段
- `ImageField()` 图片字段

关系字段

- `OneToOneField(to, on_delete=xxx, options)` - 单对单关系
- `ForeignKey(to, on_delete=xxx, options)` - 单对多关系
- `ManyToManyField(to, options)` - 多对多关系

模型方法

以下三个方法是Django模型自带的三个标准方法：

- `def __str__()`：给单个模型对象实例设置人为可读的名字(可选)。
- `def save()`：重写save方法(可选)。
- `def get_absolute_url()`：为单个模型实例对象生成独一无二的url(可选)

用户模型

```
class User(models.Model):
    username = models.CharField(max_length=50) # 用户名 CharField字段 最大长度为50
    password = models.CharField(max_length=200) # 密码
    nickname = models.CharField(max_length=50, default='匿名用户')
    email = models.EmailField
    created_time = models.CharField(max_length=50, default=now)
    comment_num = models.PositiveIntegerField(verbose_name='评论条数', default=0)
    avatar = models.ImageField(upload_to='media', default="media/cat.png")

    # 默认方法 给单个模型对象实例设置人为可读的名字(可选)。
    def __str__(self):
        return self.username

    # 发表评论
    def comment(self):
        self.comment_num += 1
        self.save(update_fields=['comment_num'])

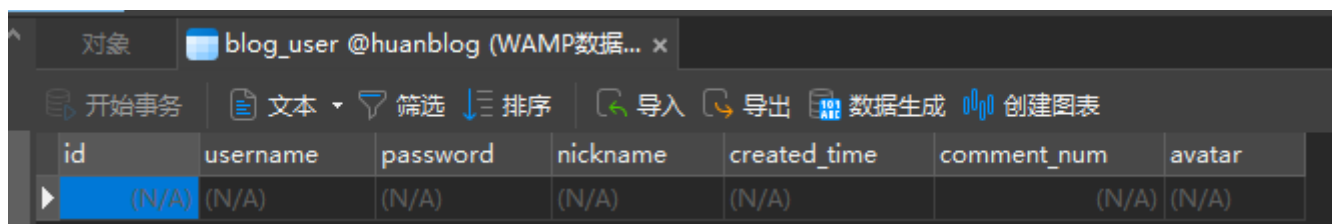
    # 删除评论
    def comment_del(self):
```

```
self.comment_num -= 1
self.save(update_fields=['comment_num'])
```

后台管理系统相关 指明名称以及数据表名

```
class Meta:
    verbose_name = '用户' # 指定后台显示模型名称
    verbose_name_plural = '用户' # 指定后台显示模型复数名称
    db_table = "blog_user"
```

- id-主键 自带的
- username 用户名 CharField
- password 密码
- nickname 昵称
- email: 邮箱地址
- created_time: 用户创建日期
- comment_num: 评论数
- avatar:头像



id	username	password	nickname	created_time	comment_num	avatar
(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)

文章模型

```
class Article(models.Model):
    STATUS_CHOICES = (
        ('a', '草稿'),
        ('b', '发表'),
    )

    article_id = models.CharField(verbose_name='标号', max_length=100)
    title = models.CharField(verbose_name='标题', max_length=100)
    content = models.TextField(verbose_name='正文', blank=True, null=True)
    status = models.CharField(verbose_name='状态', max_length=1, choices=STATUS_CHOICES, default='b')
    views = models.PositiveIntegerField(verbose_name='浏览量', default=0)
    created_time = models.DateTimeField(verbose_name='创建时间', default=now)
    category = models.ForeignKey(Category, verbose_name='分类', on_delete=models.CASCADE, blank=False, null=False)
    tags = models.ManyToManyField(Tag, verbose_name='标签集合', blank=True)

    def __str__(self):
        return self.title

    # 更新文章浏览量
    def viewed(self):
        self.views += 1
        self.save(update_fields=['views'])

    # 下一篇
    def next_article(self):
        # id比当前id大 状态为已经发布 且发布时间需合法
        return Article.objects.filter(id__lt=self.id, status='b', pub_time__isnull=False).first()

    class Meta:
        ordering = ['-created_time'] # 按照文章创建日期进行降序排列
```

```

verbose_name = '文章' # 指定后台显示模型名称
verbose_name_plural = '文章列表' # 指定后台显示模型复数名称
db_table = 'article'
get_latest_by = 'created_time'

```

表	视图	函数	用户	其它	查询	备份	自动运行	模型	图表
对象	blog_user @huanblog (WAMP数据库...)	article @huanblog (WAMP数据库学习...)							
开始事务	文本	筛选	排序	导入	导出	数据生成	创建图表		
id	article_id	title	content	status	views	created_time	category_id		
(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)		

文章评论模型

```

class ArticleComment(models.Model):
    body = models.TextField()
    username = models.CharField(max_length=50)
    usingimg = models.CharField(max_length=70)
    nickname = models.CharField(max_length=50, default='匿名用户')
    createtime = models.DateTimeField(verbose_name='创建时间', default=now)
    article = models.CharField(max_length=50)
    title = models.CharField(max_length=50)

    def __str__(self):
        return self.article

    class Meta:
        ordering = ['-createtime']
        verbose_name = '评论' # 后台显示
        verbose_name_plural = '评论列表'
        db_table = "comment" # 表名

```

对象	blog_user @huanblog (WAMP数据库...)	article @huanblog (WAMP数据库...)	comment @huanblog (WAMP数据库...)						
开始事务	文本	筛选	排序	导入	导出	数据生成	创建图表		
id	body	username	usingimg	nickname	createtime	article	title		
(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)		

文章标签模型

```

class Tag(models.Model):
    name = models.CharField(verbose_name='标签名', max_length=64)

    def __str__(self):
        return self.name

    class Meta:
        ordering = ['name']
        verbose_name = '标签名称'
        verbose_name_plural = '标签列表'
        db_table = "tag" # 数据库表名

```

开始事务			文本	筛选	排序	导入	导出	数据生成	创建图表
id	article_id	tag_id							
(N/A)	(N/A)	(N/A)							

文章分类模型

```
class Category(models.Model):
    name = models.CharField(verbose_name='类别名称',max_length=64)

    class Meta:
        ordering = ['name']
        verbose_name = '类别名称'
        verbose_name_plural = '分类列表'
        db_table = 'category' # 数据库表名

    def __str__(self):
        return self.name
```

对象

blog_user @huan...

article @huanblo...

comment @huan...

article_tags @hua...

category @huanb...

开始事务

文本

筛选

排序

导入

导出

数据生成

创建图表

id	name
(N/A)	(N/A)