**Django学习笔记**

一、Django模型最佳实践

1. 正确的为模型和关系字段命名。

2. 设置适当的`related\_name`属性。

3. 用`OneToOneField`代替`ForeignKeyField(unique=True)`。

4. 通过“迁移操作”（migrate）来添加模型。

5. 用NoSQL来应对需要降低范式级别的场景。

6. 如果布尔类型可以为空要使用`NullBooleanField`。

7. 在模型中放置业务逻辑。

8. 用`<ModelName>.DoesNotExists`取代`ObjectDoesNotExists`。

9. 在数据库中不要出现无效数据。

10. 不要对`QuerySet`调用`len()`函数。

11. 将`QuerySet`的`exists()`方法的返回值用于`if`条件。

12. 用`DecimalField`来存储货币相关数据而不是`FloatField`。

13. 定义`\_\_str\_\_`方法。

14. 不要将数据文件放在同一个目录中。

二、模型定义参考

2.1、字段

对字段名称的限制

- 字段名不能是Python的保留字，否则会导致语法错误

- 字段名不能有多个连续下划线，否则影响ORM查询操作

Django模型字段类

| 字段类                |  说明                                                         |

| --------------------- | ------------------------------------------------------------ |

| AutoField             |自增ID字段                                                   |

| BigIntegerField       |64位有符号整数                                               |

| BinaryField           | 存储二进制数据的字段，对应Python的bytes类型                  |

| BooleanField          | 存储True或False                                              |

| CharField             | 长度较小的字符串                                             |

| DateField             | 存储日期，有auto\_now和auto\_now\_add属性                       |

| DateTimeField         | 存储日期和日期，两个附加属性同上                             |

| DecimalField          |存储固定精度小数，有max\_digits（有效位数）和decimal\_places（小数点后面）两个必要的参数 |

| DurationField         |存储时间跨度                                                 |

| EmailField            | 与CharField相同，可以用EmailValidator验证                    |

| FileField             | 文件上传字段                                                 |

| FloatField            | 存储浮点数                                                   |

| ImageField            | 其他同FileFiled，要验证上传的是不是有效图像                  |

| IntegerField          | 存储32位有符号整数。                                         |

| GenericIPAddressField | 存储IPv4或IPv6地址                                           |

| NullBooleanField      | 存储True、False或null值                                      |

| PositiveIntegerField  | 存储无符号整数（只能存储正数）                               |

| SlugField             | 存储slug（简短标注）                                         |

| SmallIntegerField     | 存储16位有符号整数                                           |

| TextField             | 存储数据量较大的文本                                         |

| TimeField             | 存储时间                                                     |

| URLField              | 存储URL的CharField                                           |

| UUIDField             | 存储全局唯一标识符                                           |

2.2、字段属性

①、通用字段属性

| 选项           | 说明                                                         |

| -------------- | ------------------------------------------------------------ |

| null           | 数据库中对应的字段是否允许为NULL，默认为False                |

| blank          | 后台模型管理验证数据时，是否允许为NULL，默认为False          |

| choices        | 设定字段的选项，各元组中的第一个值是设置在模型上的值，第二值是人类可读的值 |

| db\_column      | 字段对应到数据库表中的列名，未指定时直接使用字段的名称       |

| db\_index       | 设置为True时将在该字段创建索引                               |

| db\_tablespace  | 为有索引的字段设置使用的表空间，默认为DEFAULT\_INDEX\_TABLESPACE |

| default        | 字段的默认值                                                 |

| editable       | 字段在后台模型管理或ModelForm中是否显示，默认为True          |

| error\_messages | 设定字段抛出异常时的默认消息的字典，其中的键包括null、blank、invalid、invalid\_choice、unique和unique\_for\_date |

| help\_text      | 表单小组件旁边显示的额外的帮助文本。                         |

| primary\_key    | 将字段指定为模型的主键，未指定时会自动添加AutoField用于主键，只读。 |

| unique         | 设置为True时，表中字段的值必须是唯一的                       |

| verbose\_name   | 字段在后台模型管理显示的名称，未指定时使用字段的名称         |

②、ForeignKey属性

1. limit\_choices\_to：值是一个Q对象或返回一个Q对象，用于限制后台显示哪些对象。

2. related\_name：用于获取关联对象的关联管理器对象（反向查询），如果不允许反向，该属性应该被设置为`'+'`，或者以`'+'`结尾。

3. to\_field：指定关联的字段，默认关联对象的主键字段。

4. db\_constraint：是否为外键创建约束，默认值为True。

5. on\_delete：外键关联的对象被删除时对应的动作，可取的值包括django.db.models中定义的：

   - CASCADE：级联删除。

   - PROTECT：抛出ProtectedError异常，阻止删除引用的对象。

   - SET\_NULL：把外键设置为null，当null属性被设置为True时才能这么做。

   - SET\_DEFAULT：把外键设置为默认值，提供了默认值才能这么做。

③、ManyToManyField属性

1. symmetrical：是否建立对称的多对多关系。

2. through：指定维持多对多关系的中间表的Django模型。

3. throughfields：定义了中间模型时可以指定建立多对多关系的字段。

4. db\_table：指定维持多对多关系的中间表的表名。

2.3、模型元数据选项

| 选项                  | 说明                                                         |

| --------------------- | ------------------------------------------------------------ |

| abstract              | 设置为True时模型是抽象父类                                   |

| app\_label             | 如果定义模型的应用不在INSTALLED\_APPS中可以用该属性指定       |

| db\_table              | 模型使用的数据表名称                                         |

| db\_tablespace         | 模型使用的数据表空间                                         |

| default\_related\_name  | 关联对象回指这个模型时默认使用的名称，默认为<model\_name>\_set |

| get\_latest\_by         | 模型中可排序字段的名称。                                     |

| managed               | 设置为True时，Django在迁移中创建数据表并在执行flush管理命令时把表移除 |

| order\_with\_respect\_to | 标记对象为可排序的                                           |

| ordering              | 对象的默认排序                                               |

| permissions           | 创建对象时写入权限表的额外权限                               |

| default\_permissions   | 默认为`('add', 'change', 'delete')`                          |

| unique\_together       | 设定组合在一起时必须独一无二的字段名                         |

| index\_together        | 设定一起建立索引的多个字段名                                 |

| verbose\_name          | 为对象设定人类可读的名称                                     |

| verbose\_name\_plural   | 设定对象的复数名称                                           |

三、查询参考

按字段查找可以用的条件：

1. exact / iexact：精确匹配/忽略大小写的精确匹配查询

2. contains / icontains / startswith / istartswith / endswith / iendswith：基于`like`的模糊查询

3. in：集合运算

4. gt / gte / lt / lte：大于/大于等于/小于/小于等于关系运算

5. range：指定范围查询（SQL中的`between…and…`）

6. year / month / day / week\_day / hour / minute / second：查询时间日期

7. isnull：查询空值（True）或非空值（False）

8. search：基于全文索引的全文检索

9. regex / iregex：基于正则表达式的模糊匹配查询

Q对象（用于执行复杂查询）的使用：

```Shell

>>> from django.db.models import Q

>>> Emp.objects.filter(

...     Q(name\_\_startswith='张'),

...     Q(sal\_\_gte=5000) | Q(comm\_\_gte=1000)

... ) # 查询名字以“张”开头且工资大于等于5000或补贴大于等于1000的员工

<QuerySet [<Emp: 张三丰>]>

```