Django Meta元数据类属性解析

Model 是 Django ORM 的核心，它有许多特性，比如我们提到过的模型类继承，还有未讲到过的的元数据。每个 Model 都是一个 Python 类，且通常会包含四个部分，它们分别如下：

- 继承自 django.db.model.Model；
- Model 元数据声明；
- Filed类型字段；
- 魔术方法__str__

除了元数据以外，其他三个部分我们在前面的章节都做了相应的介绍，在本节将详细讲解元数据 Meta 类属性。

1. 初识Meta内部类

每个模型类（Model）下都有一个子类 Meta，这个子类就是定义元数据的地方。Meta 类封装了一些数据库的信息，称之为 Model 的元数据。Django 会将 Meta 中的元数据选项定义附加到 Model 中。常见的元数据定义有 db_table（数据表名称）、abstract（抽象类）、ordering（字段排序） 等，Meta 作为内部类，它定义的元数据可以让admin 管理后台对人类更加友好，数据的可读性更高。

Meta 定义的元数据相当于 Model 的配置信息，所以我们可以根据自己的需求进行选择性的添加。当没有需要的时候也可以不定义 Meta，这个时候 Django 会应用默认的 Meta 元数据。

2. Meta类元数据

通过上面的介绍我们知道 Meta 类的作用就是用于定义 Model 的元数据，即不属于 Model 的字段，但是可以用来标识字段一些属性，下面我们介绍 Meta 定义的常见元数据以及如何在 Model 中使用它们。

1) abstract

一个布尔类型的变量。这个属性是定义当前的模型是不是一个抽象类。所谓抽象类是不会对应数据库表的。一般我们用它来归纳一些公共属性字段，然后继承它的子类可以继承这些字段。如果 abstract = True 这个 model 就是一个抽象类。

2) ordering

用于执行获取对象列表时的排序规则。它是一个字符串的列表或元组对象，它的使用格式是由代表字段的字符串和一个表明降序的'-'构成。当字段名前面没有'-'时，将默认使用升序排列。使用'?'将会随机排列。示例如下所示：

```
ordering=["add_time"] #按照升序排序
ordering=["-add_time"]#按照降序
ordering=["?add_time"]#随机排序
#同时指定多个字段来进行排序
ordering=['add_time','-last_login_time']#先按升序，在按降序
```

3) verbose_name_plural

这个元数据主要用在管理后台的展示上，verbose_name_plural 是模型类的复数名 。如果不设置的话，Django 会使用小写的模型名作为默认值，并且在结尾加上 s。通过此项元数据设置名字可以去掉 s。 可参见《Django Admin数据表可视化》一节。

4) db_table

这个字段用于指定数据表的名称，通常没有特别需求，将按照 Django 默认的规则生成 Model 对应的数据库表名。

```
#定义该model在数据库中的表名称
    db_table = 'Students'
#使用自定义的表名，可以通过以下属性
    table_name = 'my_owner_table'
```

5) app_lable

这个选项只在一种使用情形，就是你的模型不在默认的应用程序包下的 models.py 文件中，这时候需要指定你这个模型是哪个应用程序的 app_label = 'app_name'。

6) managed

它是一个布尔类型的变量，默认为 Ture，代表 Django 会管理数据的生命周期，即利用 Django 提供的 syncdb 和 reset 命令可以完成创建和删除数据表。如果为 False，则不会对此模型执行数据库表创建或删除操作。比如数据表之间存在 ManyToMany 的关系，在指定

为 managed=False 的情况下，Django 不会自动创建中间表，需要我们自己手动创建。

7) indexs

它是一个列表类型的元数据项，用来定义 Model 的索引，列表中的每一个元素都是 Index 类型的实例。

Index 引自 django.db.models.indexes.Index

8) default_permissions

Django 默认会给每一个定义的 Model 设置三个权限即添加、更改、删除，它使用格式：default_permissions=('add','change','delete','view')

9) permissions

除了 Django 默认给 Model 添加的三个权限之外，还可以通过 permisssions 给 Model 添加额外的权限。不过 permissions 是一个包含二元组的元组或者列表，所以使用时应该注意格式，即 permissions=[(权限代码，权限名称)]，示例如下所示：

permissions = [(have_read_permission', '有读的权限')]

10) unique_together

这个选项用于下面情形：当你需要通过两个字段保持唯一性时使用。比如用户的姓名（name）和 身份证号码（ID number）两者的组合必须是唯一的，那么需要这样设置：

unique_together = (("first_name", "last_name"),)
一个 ManyToManyField 不能包含在 unique_together 中。如果你需要验证 ManyToManyField 字段的唯一验证，尝试使用 through 属性进行关联。

11) proxy

默认值为为 False， 如果设置成 Ture，则表示为基类、父类的代理模型。这个选项在后续章节还会进行相关介绍，它的主要作用就是创建父模型的代理模型。

12) db_tablespace

表空间，用于优化数据库性能，常用于 Oracle、PostgerSQL 数据库。MySQL 数据库不支持表空间，所以当数据存储后端数据库不支持的时候，Django 会在自动忽略这个元数据选项。

13) get_latest_by

指定一个 DateField 或者 DateTimeField 字段的名字，即 model 的属性名字。使用示例如下：

get_latest_by = "order_date"
这个设置让你在使用模型管理器的 lastest() 方法时，默认使用`order_date` 指定字段来排序。

14) order_with_respect_to

这个选项一般用于多对多的关系中，它指向一个关联对象并将该对象进行排序，使用元数据项后你会得到一个 get_xxx_order() 和 set_xxx_order() 的方法，通过它们你可以设置或者得到排序的对象。

本节给大家介绍了 Meta 类以及类中定义的各项元数据。Meta 类是对 model 模型类的进一步完善以及扩展，所以对于重点的元数据项要学会使用。大家也可以参照官方文档《Model Meta options》进一步学习它，文档中提供了所有的 Meta 类元数据选项。