**通过一个内嵌类 "class Meta" 给你的 model 定义元数据, 类似下面这样:**

class Foo(models.Model):

bar = models.CharField(maxlength=30)

class Meta:

# ...

Model 元数据就是 "不是一个字段的任何数据" -- 比如排序选项, admin 选项等等.

下面是所有可能用到的 Meta 选项. 没有一个选项是必需的. 是否添加 class Meta 到你的 model 完全是可选的.

**app\_label**

app\_label这个选项只在一种情况下使用，就是你的模型类不在默认的应用程序包下的models.py文件中，这时候你需要指定你这个模型类是那个应用程序的。比如你在其他地方写了一个模型类，而这个模型类是属于myapp的，那么你这是需要指定为：

app\_label='myapp'

**db\_table**

db\_table是用于指定自定义数据库表名的。Django有一套默认的按照一定规则生成数据模型对应的数据库表名，如果你想使用自定义的表名，就通过这个属性指定，比如：

table\_name='my\_owner\_table'

若不提供该参数, Django 会使用 app\_label + '\_' + module\_name 作为表的名字.

若你的表的名字是一个 SQL 保留字, 或包含 Python 变量名不允许的字符--特别是连字符 --没关系. Django 会自动在幕后替你将列名字和表名字用引号引起来.

**db\_tablespace**

有些数据库有数据库表空间，比如Oracle。你可以通过db\_tablespace来指定这个模型对应的数据库表放在哪个数据库表空间。

**get\_latest\_by**

由于Django的管理方法中有个lastest()方法，就是得到最近一行记录。如果你的数据模型中有 DateField 或 DateTimeField 类型的字段，你可以通过这个选项来指定lastest()是按照哪个字段进行选取的。

一个 DateField 或 DateTimeField 字段的名字. 若提供该选项, 该模块将拥有一个 get\_latest() 函数以得到 "最新的" 对象(依据那个字段):

get\_latest\_by = "order\_date"

**managed**

由于Django会自动根据模型类生成映射的数据库表，如果你不希望Django这么做，可以把managed的值设置为False。

默认值为True,这个选项为True时Django可以对数据库表进行 migrate或migrations、删除等操作。在这个时间Django将管理数据库中表的生命周期

如果为False的时候，不会对数据库表进行创建、删除等操作。可以用于现有表、数据库视图等，其他操作是一样的。

**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。就是说关联对象找到这个对象后它是经过排序的。指定这个属性后你会得到一个get\_XXX\_order()和set\_XXX\_order（）的方法,通过它们你可以设置或者回去排序的对象。

举例来说, 如果一个 PizzaToppping 关联到一个 Pizza 对象, 这样做:

order\_with\_respect\_to = 'pizza'

...就允许 toppings 依照相关的 pizza 来排序.

**ordering**

这个字段是告诉Django模型对象返回的记录结果集是按照哪个字段排序的。比如下面的代码：

ordering=['order\_date']

# 按订单升序排列

ordering=['-order\_date']

# 按订单降序排列，-表示降序

ordering=['?order\_date']

# 随机排序，？表示随机

ordering = ['-pub\_date', 'author']

# 对 pub\_date 降序,然后对 author 升序

**需要注意的是:不论你使用了多少个字段排序, admin 只使用第一个字段**

**permissions**

permissions主要是为了在Django Admin管理模块下使用的，如果你设置了这个属性可以让指定的方法权限描述更清晰可读。

要创建一个对象所需要的额外的权限. 如果一个对象有 admin 设置, 则每个对象的添加,删除和改变权限会人(依据该选项)自动创建.下面这个例子指定了一个附加权限: can\_deliver\_pizzas:

permissions = (("can\_deliver\_pizzas", "Can deliver pizzas"),)

这是一个2-元素 tuple 的tuple或列表, 其中两2-元素 tuple 的格式为:(permission\_code, human\_readable\_permission\_name).

**unique\_together**

unique\_together这个选项用于：当你需要通过两个字段保持唯一性时使用。这会在 Django admin 层和数据库层同时做出限制(也就是相关的 UNIQUE 语句会被包括在 CREATE TABLE 语句中)。比如：一个Person的FirstName和LastName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"),)

**verbose\_name**

verbose\_name的意思很简单，就是给你的模型类起一个更可读的名字：

verbose\_name = "pizza"

若未提供该选项, Django 则会用一个类名字的 munged 版本来代替: CamelCase becomes camel case.

**verbose\_name\_plural**

这个选项是指定，模型的复数形式是什么，比如：

verbose\_name\_plural = "stories"

若未提供该选项, Django 会使用 verbose\_name + "s".