1).创建数据库 - webdb

create database webdb default charset utf8 collate utf8\_general\_ci;

2).Django的数据库配置

在settings.py中配置数据库的信息

DATABASES = {

'default':{

'ENGINE':django.db.backends.sqlites,

'NAME':os.path.join(BASE\_DIR,DB.sqlite3),

}

}

1.ENGINE:指定要连接的数据库的驱动程序

连接MYSQL的话可以取值为:

'django.db.backends.mysql'

2.NAME:指定要连接到的数据库的名称

连接mysql的话可以取值为:

数据库名

3.USER:指定登录到数据库管理系统的用户名

连接到mysql的话,可以为:root

4.PASSWORD:指定登录到数据库管理系统的密码

5.HOST:指定要连接到的主机地址

如果是本机的话:127.0.0.1或localhost

6.PORT:指定端口号

mysql的默认为3306

示例:

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.mysql',

'NAME': 'webdb',

'USER': 'root',

'PASSWORD': '123456',

'HOST': 'localhost',

'PORT': 3306,

}

}

3).编写models

在创建的app下的models中编写（不要忘记将app添加到seeting.py中的

INSTALLED\_APPS 中）

1.Models的语法规范

class ClassName(models.Model):

属性 = models.FIELDYPE(FIELD\_OPTIONS)

FIELDTYPE : 字段类型

CharField()

FIELD\_OPTIONS:字段选项(字段说明)

max\_length=30

2. 常见的字段名和字段选项

字段类型 FIELDTYPE

1.BooleanField()

编程语言中使用true 或 false 来表示该列的值

数据库中使用1 或0 来表示具体的值

数据类型为:tinyint

2.CharField()

编程语言中使用字符串来表示该列的值

数据库中也是字符串

数据类型为:varchar

注意:必须要指定max\_length参数值

3.DateField()

编程语言中使用字符串或者Date类型的数据表示该值

数据库中使用的是时间字符串

数据类型为;date

4.DateTimeField()

数据类型为:datetime

5.DecimalField()

编程语言中使用数字(小数)来表示该列的值

数据库中使用小数表示

数据类型:decimal

money = models.DecimalField(max\_digits=7,decimal\_places=2)

6.FloatField()

编程语言中使用数字(小数)来表示该列的值

数据库使用小数

数据类型:float

7.IntegerField()

数据类型:int

8.EmailField()

编程语言中使用字符串来表示一段email地址

数据库中使用字符串表示

数据类型:varchar

9.URLField()

编程语言中使用字符串来表示一段网址

数据库中使用字符串表示

数据类型:varchar

10.ImageField()

目的:存储图片的路径

数据类型:varchar

image = models.ImageField(upload\_to='images/')

字段选项:FIELD\_OPTIONS

1.default

作用:为当前字段指定默认值

2.null

作用:指定当前字段是否为空,默认值为False,不能为空

3.db\_index

作用:指定是否为当前指定索引

4.db\_column

作用:指定当前属性映射到表中的类名,如果不指定则采用属性名称作为类名

# 创建一个实体类 - Publisher(出版社)

# 1.name : 出版社名称(varchar(30))

# 2.address: 出版社所在地址(varchar(200))

# 3.city:出版社所在的城市(varchar(50))

# 4.country:出版社所在的国家(varchar(50))

# 5.website:出版社的网址(varchar(200))

class Publisher(models.Model):

name = models.CharField(max\_length=30)

address = models.CharField(max\_length=200)

city = models.CharField(max\_length=50)

country = models.CharField(max\_length=50)

website = models.URLField()

4） 将数据同步回数据库

先后执行第1 步和第2 步

1. ./manage.py makemigrations

作用:将每个应用下的modeles.py文件生成一个数据库的中间文件,并将中间文件保存

到migrations的目录中

2. /manage.py migrate

作用:将每个应用下的migrations目录中的中间文件同步到数据库中