模型

1. Django对各种数据库都提供了很好的支持，Django为数据库提供了统一的API，可以根据不同的业务需求选择不同的数据库
2. 配置数据库，工程目录下的\_\_init\_\_.py文件中修改为
   1. import pymysql

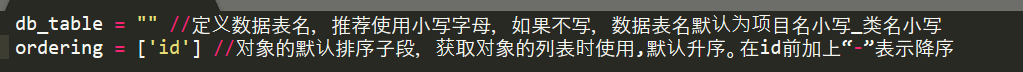
pymysql.install\_as\_MYSQLdb()

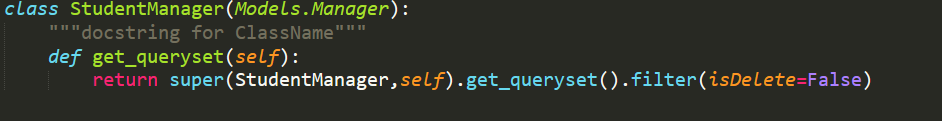
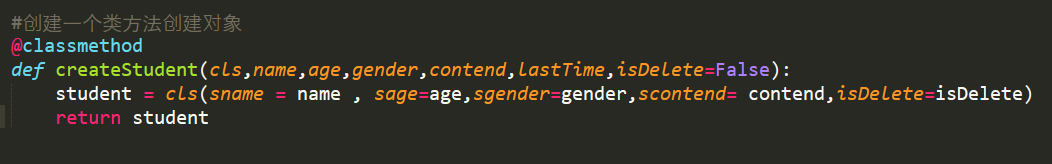
* 1. 修改setting.py文件

1. 开发流程
   1. 配置数据库
   2. 定义模型类
      1. 一个模型类在数据库中对应一张数据表
   3. 生成迁移文件
   4. 生成数据表
   5. 使用模型类进行增删改查
2. ORM
   1. 概述：对象-关系-映射
   2. 任务：
      1. 根据对象的类型生成表结构
      2. 将对象、列表的操作转换为sql语句
      3. 将sql语句查询到的结果转换为对象、列表
   3. 优点：极大的减少了开发者的工作量，而且不需要面对一种操作因为数据库的改变而变更代码
3. 定义模型
   1. 模型、属性、表、字段间的关系
      1. 一个模型类在数据库中对应一张表，在模型类中定义的属性对应模型对应表中的一个字段
   2. 定义属性
      1. 概述：Django根据属性的类型确定一下信息
         1. 当前选择的数据库支持字段的类型
         2. 渲染管理表单时的默认HTML控件
         3. 在管理站点最低限度的验证
         4. Django会为表增加自动增长的主键列，每个模型只能有一个主键列，如果使用选项设置某属性为主键后，则Django不会再生成默认的主键列
         5. 属性命名规则
            1. 不能是python的保留关键字
            2. 由于Django的查询方式，不能使用连续的下划线
      2. 库
         1. 定义属性时，需要字段类型，字段类型定义在django.db.models.files目录下，为了方便使用，被导入到django.db.models中
         2. 使用方式
            1. 导入from django.db import models
            2. 通过models.Filed创建字段类型的对象，赋值给属性
      3. 逻辑删除
         1. 对于重要的数据都做逻辑删除，不做物理删除，实现方法是定义isDelete属性，类型为BooleanFiled，默认值为False
      4. 字段类型
         1. AutoFiled
            1. 一个根据实际ID自动增长的IntegerFiled，通常不指定，一个主键字段将自动增加到模型中
         2. CharFiled（max\_length=字符长度）
            1. 字符串，默认的表单样式是textinput
         3. TextFiled
            1. 大文本字段，一般是超过4000使用，默认的表单控件是Textarea
         4. IntegerFiled
            1. 整数
         5. DecimalFiled（max\_digits=None,decimal\_places=None）
            1. 使用python的decimal实力表示十进制浮点数
            2. 参数说明：第一个表示位数总数，第二个表示小数点后面的数字位数
         6. FloatFiled
            1. 用python的float实例表示浮点数
         7. BooleanFiled
            1. True/False字段，默认控件为CheckBoxInput
         8. NullBooleanFiled
            1. 支持null、True、False
         9. DateFiled[auto\_now = False,auto\_now\_add=false]
            1. 使用python的datetime.date实例表示的日期
            2. 参数说明：

第一个，每次保存对象时，自动设置该字段为当前时间，用于“最后一次修改”的时间戳，他总是使用当前时间，默认为False

当对象创建时自动设置为当前时间，默认为False

* + - 1. TimeFiled，同9
      2. DateTimeFiled
         1. 使用python的datetime
      3. imageFiled
         1. 有效的图片
    1. 字段选项
       1. 概述：通过字段选项，可以实现对字段的约束，在字段对象时通过关键字参数指定
       2. Null
          1. 如果是true，Django将空值以null的形式存储到数据库中，默认值是False
       3. Blanke
          1. 如果为true，表示该字段运行为空白，默认值为False
          2. Null是数据库范畴的概念，blanke是表单验证范畴的
       4. Do\_column
          1. 字段名称，如果未指定，则使用属性的名称
       5. Do\_index
          1. 若值为true，则在表中会为此字段创建索引
       6. Default
          1. 默认值
       7. Primary\_key
          1. 若为true，则该字段会成为模型的主键字段
       8. Unique
          1. 如果值为true，则表示这个字段在表中必须有惟一值
    2. 关系
       1. 分类：
          1. ForeignKey:一对多，将字段定义在多的端中
          2. ManyToManyFiled：多对多，将字段定义在两端中
          3. OneToOneFiled：一对一，字段任意定义在一端中
       2. 用一访问多：
          1. 格式：对象.模型类小写\_set
       3. 用一访问一：
          1. 对象.模型类小写
       4. 访问id
          1. 对象.属性\_id
  1. 创建模型类
  2. 元选项：在模型类中定义一个meta类用于设置元信息

1. 模型成员
   * 1. 类属性
        1. Objects ：是manager类型的对象，作用是与数据库进行交互
           1. 定义模型类时，没有指定管理器，则Django为模型创建名为objects的管理器
        2. 自定义管理器
           1. 
           2. 当我们自定义了管理器时，Django不再生产管理器
        3. 自定义管理器Manager类
           1. 作用：向管理器类中添加额外的方法、修改管理器返回的原始查询集（重写get\_query方法）
           2. 代码示例：
     2. 创建对象
        1. 目的：向数据库中添加数据
        2. 当创建对象时，Django不会对数据库进行操作
        3. 方法：
           1.  在模型类中增加一个类方法
           2. 在自定义管理器中添加一个方法