数据库后端

一.后端的开发与设计

使用Python作为开发语言，通过Django库进行后台框架搭建。主要对三个文件进行设计：

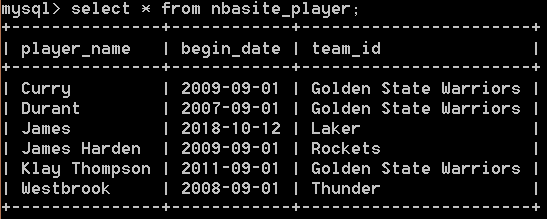
①通过Urls.py来解析前端发送过来的Http请求，并执行相应的方法

②在Views.py里完成各种方法的实现，并进行对数据库的操作

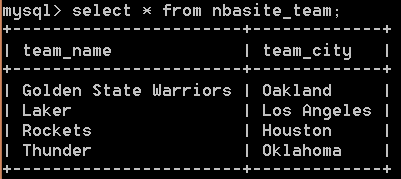
③在Models.py里实现数据库模型的构建，以供Views.py使用；或者同样使用MySQLdb进行数据库的建立，两种方式均可。

二.数据库表的定义和展示

数据库的定义较为简单，包括两个表，球员信息表和球队信息表。其中球员数据属性包括球员姓名（Prime Key 此次作业先把球员名当做主码使用）、开始比赛年份、所在球队（外码），球队属性包括球队名（Prime Key）、球队所在城市。此次数据库仅供实现基本操作，所以较为简单。



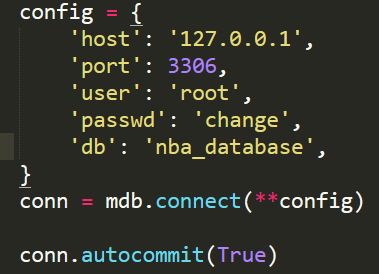
球员表



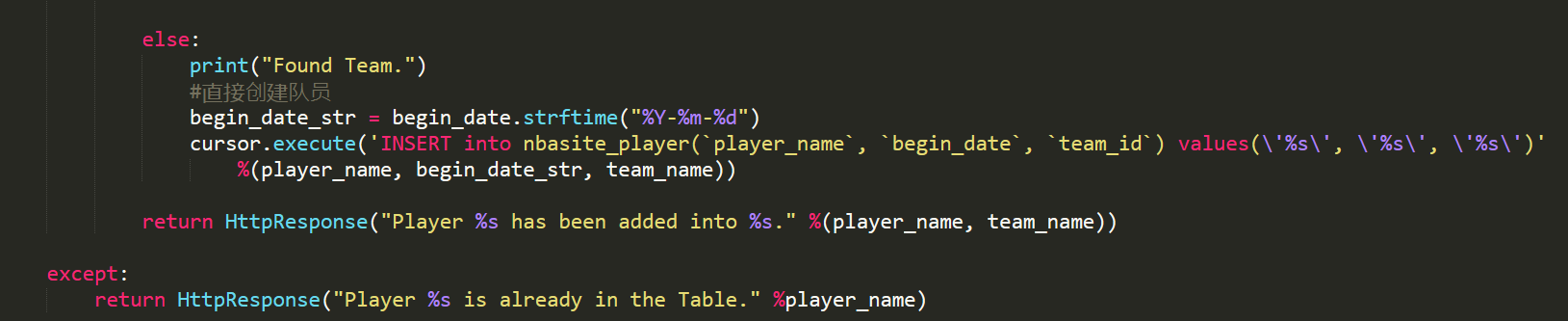
球队表

三.数据库的连接方法与基本操作

①使用Python的MySQLdb的connect进行数据库连接，并使用cursor游标进行操作。



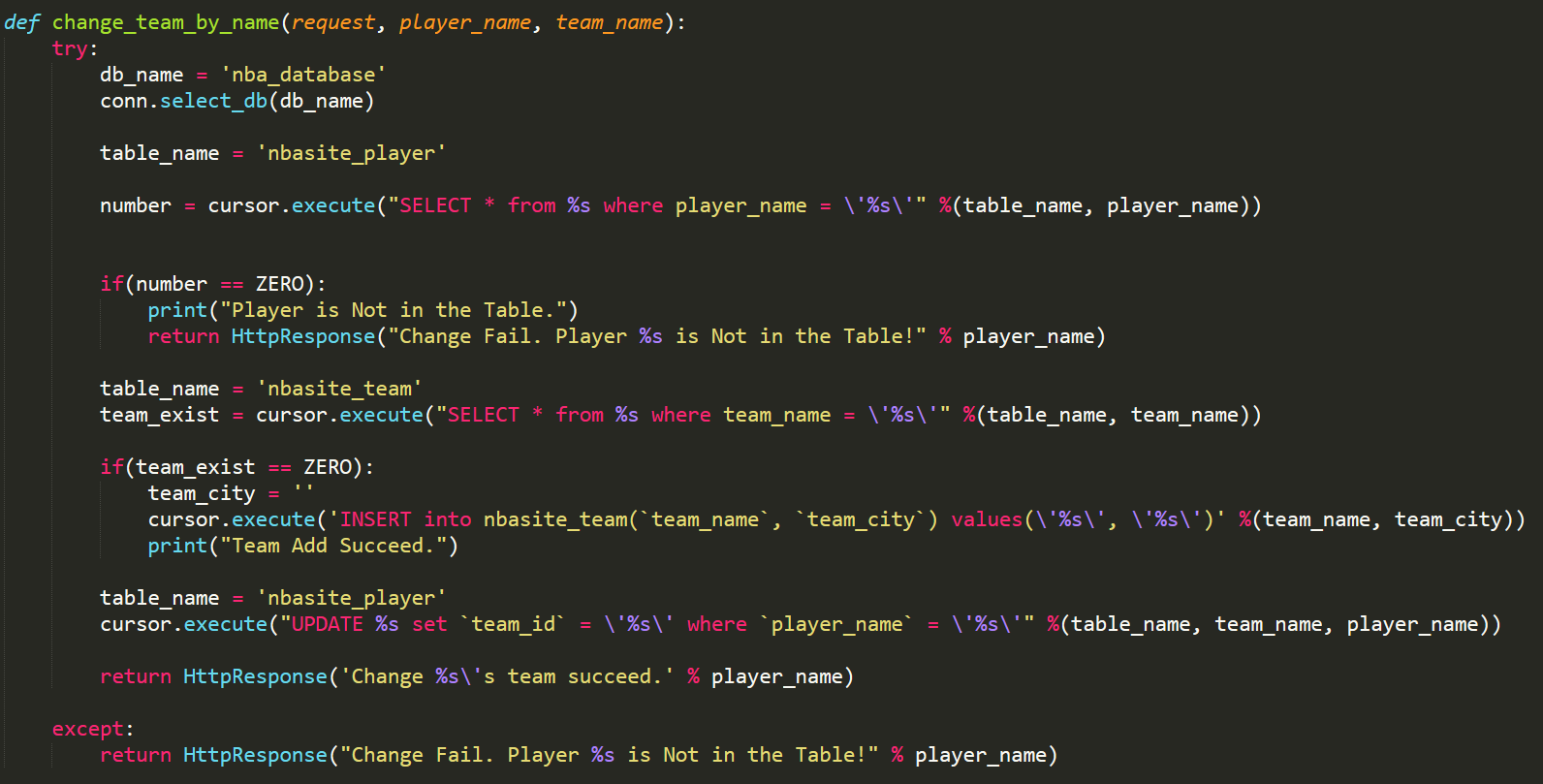
②基本操作的实现：



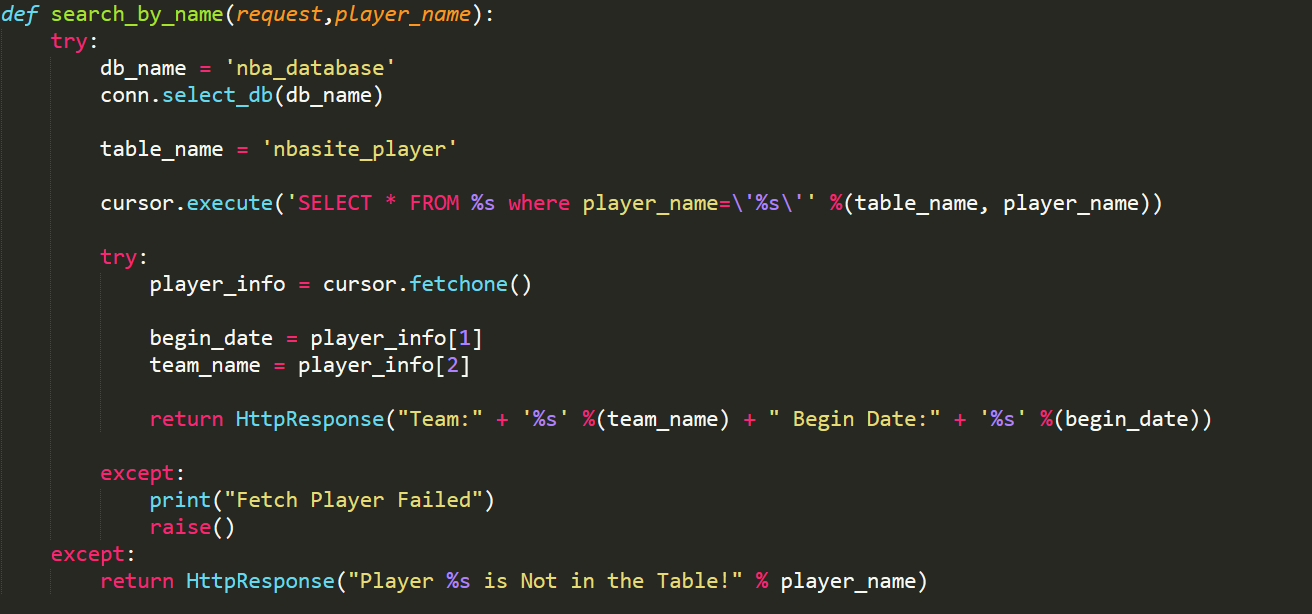
1.“增加球员”操作使用MySQL的INSERT，具体实现中为添加球员信息，先查找是否已经有了同名球员，如果有了返回相应信息，如果没有，再查找是否有该球队信息。如果没有球队信息则先创建新球队，然后再根据新球队名最为外码创建新球员。



2.“删除球员”操作使用MySQLdb的DELETE，具体实现为，先在表中找同名球员，如果没有给出相应的反馈，如果有则将其从数据库中删除并给出回应。

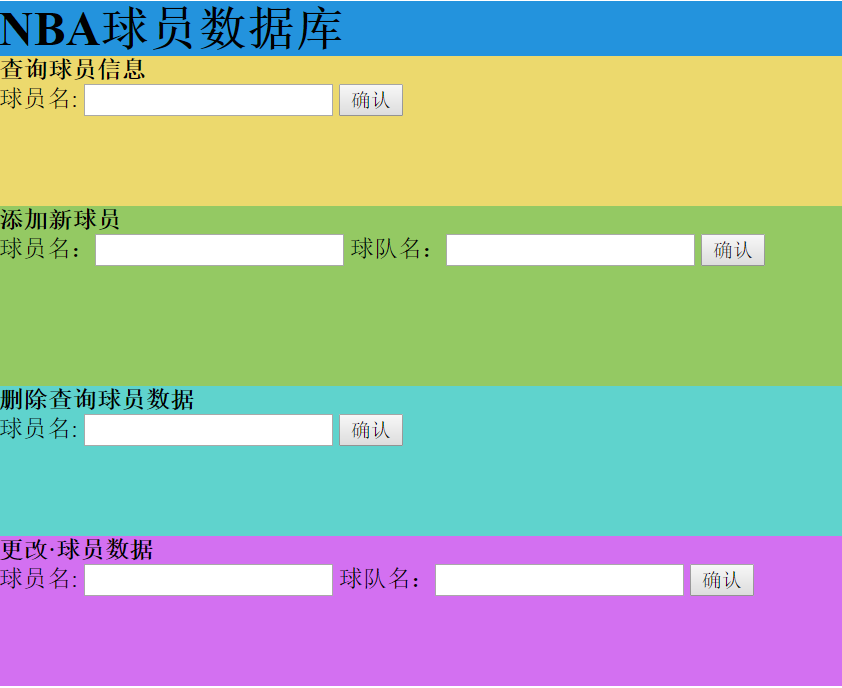


3.“更改队伍”操作使用MySQLdb的UPDATE，具体实现为，先在表中找同名球员，如果没有则报错并返回消息，如果有则在表中查找同名球队。如果没有则先建立该球队，如果有则将球员的球队信息更新。

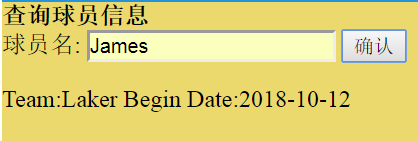


4.“查询球员”操作使用MySQLdb的SELECT，查找同名球员，如果存在，再根据外部键找出球队信息，并将球员和球队信息一起输出。如果不存在，则返回查询失败。

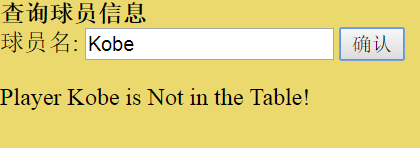
③整体效果



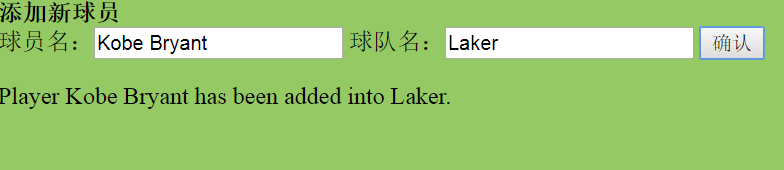
④展示效果



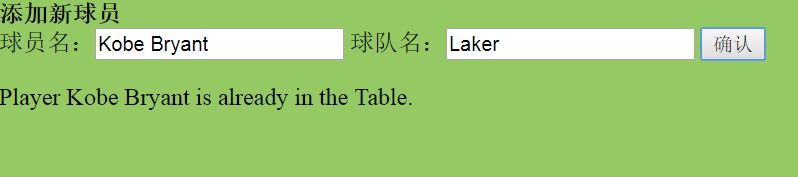
1.查询操作存在



2.查找球员不存在



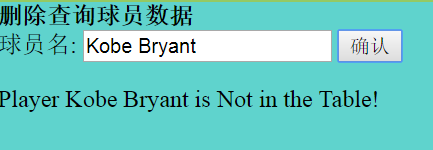
3.添加新球员成功



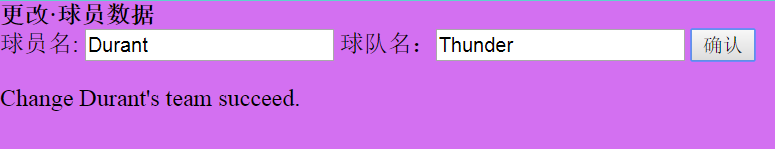
4.添加已存在球员失败



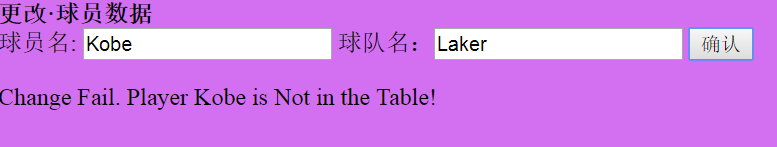
5.删除球员成功



6.删除不存在球员失败



7.更换球员球队成功



8.更换不存在球员失败