**WhoScored爬虫需求**

Whoscored (https://www.whoscored.com)是世界知名的足球数据网站，由于该网站使用了Incapsula反爬机制，我在使用python爬取过程中遇到了如下问题，使用Requests库获取失败，尝试设置header依然失败，之后尝试使用selenium库模拟网页操作爬取数据，连续访问过程中会被Incapsula反爬机制识别为爬虫，见图1，我在每次模拟操作中加入了随机的时间间隔，似乎解决了这个问题，但是耗时太久，而且我只测试爬取了几千个页面不确定稳定性，selenium我认为不是一个好的解决方案。由于足球比赛数据是实时更新的，所以需要爬虫也能实时或者定时爬取最新的比赛数据。

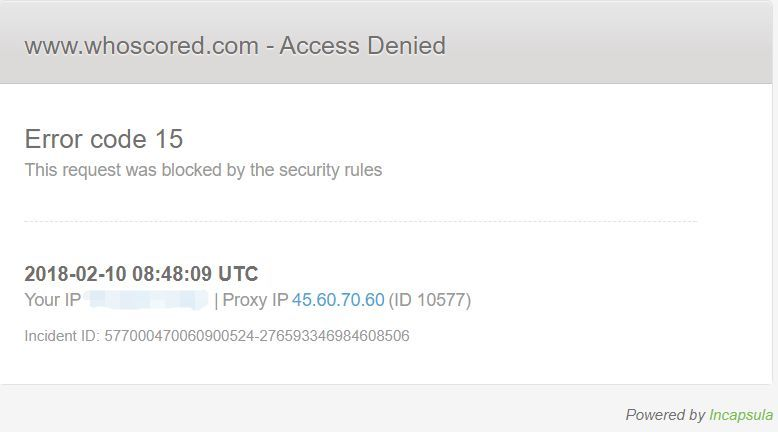


图1

**一．球队数据爬取需求**

进入Whoscored主页https://www.whoscored.com

点击菜单栏LIVE SCORES进入https://www.whoscored.com/LiveScores，见图2

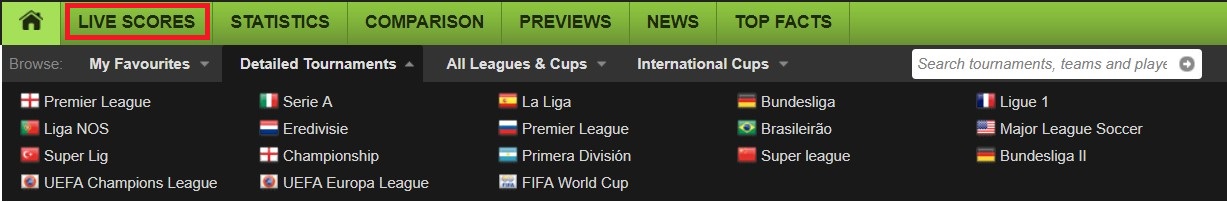


图2

可以看到Football Live Scores表是当天的比赛信息，表头右侧可以选择日期，见图3

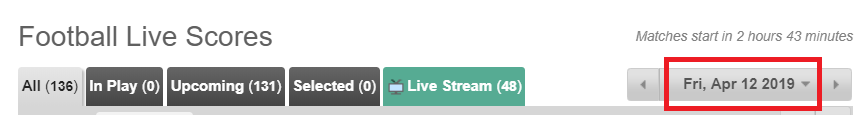


图3

点击日期可以显示日期选择的模块，见图4，左侧是年份可选择2009至2020年，中间是月份1月至12月，右侧是具体日期的选择，需要从2009年1月1日爬取至当前时间换算成伦敦时间的前一天（注意：Whoscored网站默认的日期和时间是伦敦时间）。

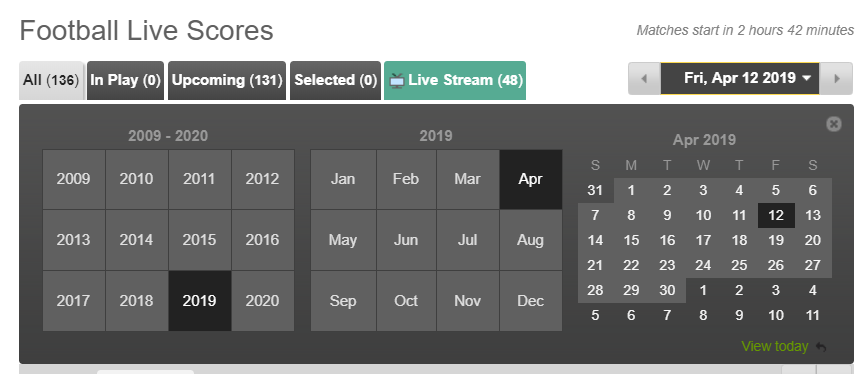


图4

下面我以2019年4月9日作为例子，见图5

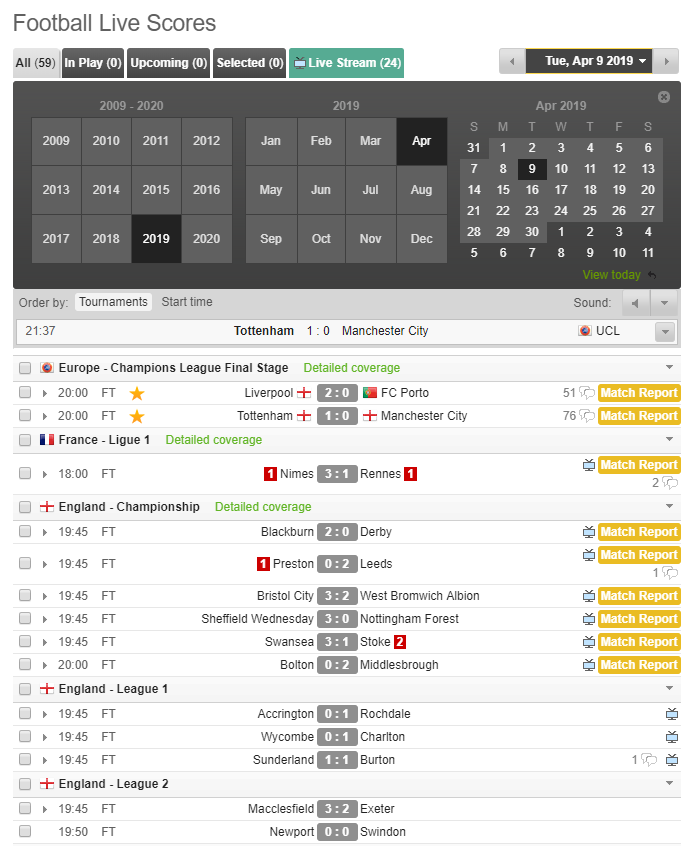


图5

在这个页面需要爬取的有以下数据：（样例数据为图6中2019/4/9 20:00 Liverpool 2:0 FC Porto的比赛数据）

|  |  |  |  |
| --- | --- | --- | --- |
| 中文名 | 字段名 | 样例数据 | 数据来源 |
| 比赛ID | match\_id | 1368027 | 比赛结果 |
| 赛事ID | tournament\_id | 12 | 比赛类型 |
| 赛事名称 | tournament\_name | Champions League Final Stage | 比赛类型 |
| 完成状态 | elapsed | FT | 比赛时间右侧 |
| 比赛日期 | match\_date | 2019/04/09 | 图4日期选择 |
| 比赛时间 | match\_time | 20:00 | 比赛时间 |
| 主队ID | ht\_id | 26 | 主队信息 |
| 主队名称 | ht\_name | Liverpool | 主队信息 |
| 主队进球数 | ht\_goals | 2 | 比赛结果 |
| 客队ID | at\_id | 297 | 客队信息 |
| 客队名称 | at\_name | FC Porto | 客队信息 |
| 客队进球数 | at\_goals | 0 | 比赛结果 |

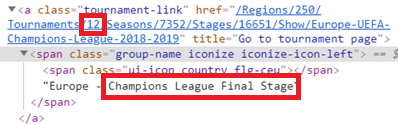


图6

下面具体说明数据来源位置，（样例数据为图6中2019/4/9 20:00 Liverpool 2:0 FC Porto的比赛数据）

1. 比赛类型：

tournament\_id: 12 tournament\_name: Champions League Final Stage



1. 比赛日期：

图4选择的日期

1. 比赛时间：

match\_time: 20:00



1. 主队信息：

ht\_id: 26 ht\_name: Liverpool



1. 比赛结果：

match\_id: 1368027 ht\_goals: 2 at\_goals: 0

可能有取消的比赛，我不确定，elapsed显示不是FT，比赛结果显示为vs，进球保存为空值



1. 客队信息：

at\_id: 297 at\_name: FC Porto



1. 比赛报告标志：

每一行末端有黄色Match Report图标的比赛才有比赛报告

**只需要爬取有比赛报告的比赛**

**二．球员数据爬取需求**

进入每场比赛的球员数据页:

‘https://www.whoscored.com/Matches/’ + match\_id + ‘/LiveStatistics/’

举个例子，2019/4/9 20:00 Liverpool 2:0 FC Porto的比赛球员数据页为：

<https://www.whoscored.com/Matches/1368027/LiveStatistics/>

可以看到两队的球员数据，以主队利物浦为例，见图7，表头有四个选项，分别对应了总览数据、防守数据、进攻数据和传球数据，Rating为球员评分，评分为’-‘的为未上场球员，**未上场球员数据不需要爬取**。

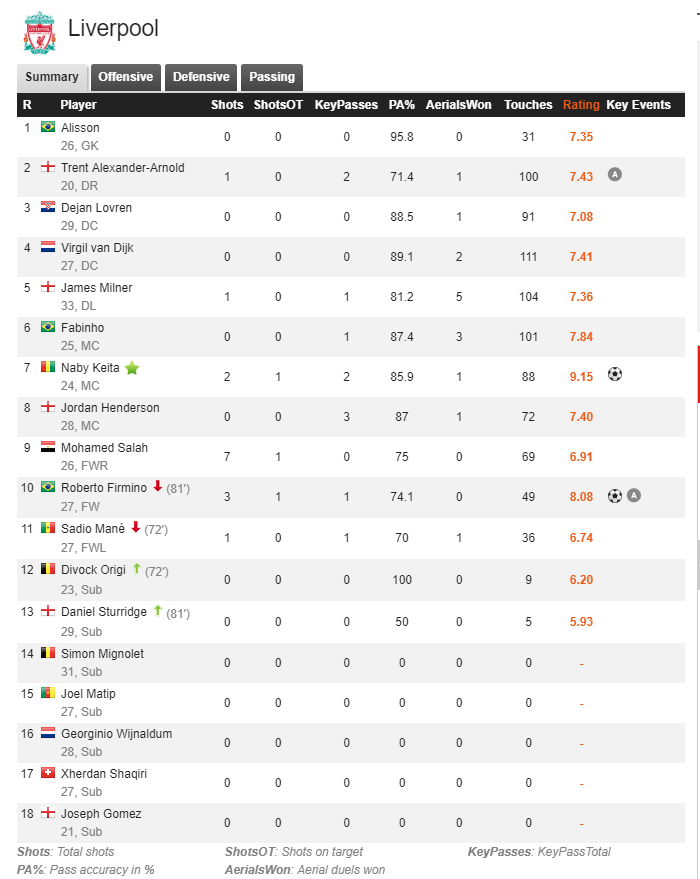


图7

在这个页面需要爬取的有以下数据：（样例数据为图6中2019/4/9 20:00 Liverpool 2:0 FC Porto的比赛数据中主队Liverpool的球员Roberto Firmino）

|  |  |  |  |
| --- | --- | --- | --- |
| 中文名 | 字段名 | 样例数据 | 数据来源 |
| 球员ID | player\_id | 96182 | Player |
| 球员姓名 | player\_name | Roberto Firmino | Player |
| 比赛ID | match\_id | 1368027 | 已知（就是根据match\_id来爬取每场比赛的） |
| 球队ID | team\_id | 26 | 表头球队图标 |
| 球队名称 | team\_name | Liverpool | 表头球队图标 |
| 场上位置 | position | FW | Player |
| 年龄 | age | 27 | Player |
| 下场标志 | off\_flag | 1 | Player |
| 下场时间 | off\_time | 81 | Player |
| 上场标志 | on\_flag | 0 | Player |
| 上场时间 | on\_time | 0 | Player |
| 进球 | goals | 1 | Key Events |
| 助攻 | assists | 1 | Key Events |
| 点球射进 | penalties\_goaled | 0 | Key Events |
| 点球射失 | penalties\_missed | 0 | Key Events |
| 黄牌 | yel | 0 | Key Events |
| 红牌 | red | 0 | Key Events |
| 乌龙球 | owng | 0 | Key Events |
| 争顶成功 | aerialswon | 0 | Summary |
| 触球 | touches | 49 | Summary |
| 评分 | ratings | 8.08 | Summary |
| 射门 | shots | 3 | Offensive |
| 射正 | shotsot | 1 | Offensive |
| 关键传球 | keypasses | 1 | Offensive |
| 突破过人 | dribbles | 2 | Offensive |
| 被侵犯 | fouled | 0 | Offensive |
| 越位 | offsides | 0 | Offensive |
| 受迫丢球 | Disp | 2 | Offensive |
| 非受迫丢球 | unstouches | 3 | Offensive |
| 抢断 | totaltackles | 3 | Defensive |
| 拦截 | interceptions | 0 | Defensive |
| 解围 | clearances | 0 | Defensive |
| 阻挡射门 | blockedshots | 0 | Defensive |
| 犯规 | fouls | 0 | Defensive |
| 传球 | passes | 27 | Passing |
| 传球成功率 | PA | 74.1 | Passing |
| 传中球 | crosses | 1 | Passing |
| 成功传中球 | acccrosses | 0 | Passing |
| 长传球 | lb | 0 | Passing |
| 成功长传球 | acclb | 0 | Passing |
| 直塞球 | thb | 0 | Passing |
| 成功直塞球 | accthb | 0 | Passing |

具体说明部分字段：

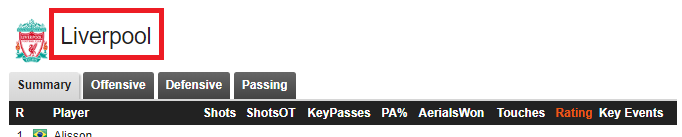
1. 球员ID：

取Player球员姓名的链接中’…Players/’和’/Show…’中间的数字



2. 球队ID、球队名称：

通过表头的图标获取





3. 上、下场标志、时间：

红色向下的箭头为下场标志，有则off\_flag为1，off\_time为括号内时间，没有则off\_flag为0，off\_time为90



绿色向上的箭头为上场标志，有则on\_flag为1，on\_time为括号内时间，没有则on\_flag为0，on\_time为0



4. Key Events：（reference为含有该图标的比赛链接，除红黄牌外其他可能同时有多个）

Key Events记录了比赛中的关键事件，包括进球、助攻、黄牌、红牌、乌龙球、点球等

1. C:\Users\congl\AppData\Local\Temp\1555380151(1).png:有一个该图标代表一个进球，goals + 1

reference: https://www.whoscored.com/Matches/1368027/LiveStatistics/

1. C:\Users\congl\AppData\Local\Temp\1555380224(1).png：有一个该图标代表一个助攻，assists + 1

reference: https://www.whoscored.com/Matches/1368027/LiveStatistics/

1. C:\Users\congl\AppData\Local\Temp\1555380383(1).png：有一个该图标代表一个点球进球，penalties\_goaled + 1 goals + 1

reference: https://www.whoscored.com/Matches/1362735/LiveStatistics/

1. C:\Users\congl\AppData\Local\Temp\1555380410(1).png：有一个该图标代表一个点球未进球，penalties\_missed + 1

reference: https://www.whoscored.com/Matches/1362723/LiveStatistics/

1. C:\Users\congl\AppData\Local\Temp\1555380537(1).png：有一个该图标代表一个乌龙球，owng + 1

reference: https://www.whoscored.com/Matches/1362735/LiveStatistics/

1. C:\Users\congl\AppData\Local\Temp\1555380588(1).png：该图标代表一张红牌，red + 1

reference: https://www.whoscored.com/Matches/1362749/ LiveStatistics/

1. C:\Users\congl\AppData\Local\Temp\1555380804(1).png：该图标代表一张黄牌，yel + 1

reference: https://www.whoscored.com/Matches/1368027/LiveStatistics/

1. C:\Users\congl\AppData\Local\Temp\1555380732(1).png：该图标代表两张红牌变一张红牌，red + 1 yel – 1

reference: https://www.whoscored.com/Matches/1362753/LiveStatistics/

5. 其他字段在表中都很容易找到

**三．每日更新数据需求**

之前的历史数据是从2009年1月1日爬取至当前时间换算成伦敦时间的前一天

***举例：***

当前时间现在是2019/4/16 11:11

伦敦时间是2019/4/16 4:11

前一天时2019/4/15

历史数据爬取2009年1月1日至2019年4月15日

每日更新数据设定在伦敦时间每天凌晨2；30自动爬取

***举例：***

伦敦时间2019/4/17 2:30（北京时间2019/4/17 9:30）开始爬取2019/4/16日的数据

每日更新数据爬取的数据内容和历史数据完全相同，见一、二部分

**四．数据存储需求**

首次爬取历史数据保存到PostgreSQL里（其他数据库，如MySQL，也可以，这一部分代码备注好我自己改）

每日更新数据填加到PostgreSQL里，同时保存为.csv文件，球队数据文件名为日期+’\_teams’，球员数据文件名为日期+’\_players’

***举例：***

球队数据 20190416\_teams.csv

球员数据 20190416\_players.csv

每日更新数据.csv文件保留近7天的数据，即保存第8天的数据时删除第1天的数据

**再次提醒：**

1. **Whoscored网站默认为伦敦时间，需要换算时间**
2. **球队数据只需要爬取有比赛报告的数据，即有黄色Match Report标志的数据**
3. **球员数据只需要爬取上场球员的数据，即Rating有数字不为’-’的数据**