上 海 财 经 大 学

**Shanghai University of Finance and Economics**

数据科学导论期末项目报告

**《Apex Lengends》全球各赛区战队及队员2019-2023数据分析**

|  |  |
| --- | --- |
| **院（系）：** | **信息管理与工程学院** |
| **班 级：** | **大数据2班** |
| **学 号：** | **2021110616** |
| **姓 名：** | **杨坤** |
| **日 期：** | **2023年5月30日** |

1. 项目背景介绍

“ALGS”全称“Apex Legends Global Series”为大逃杀类FPS游戏《Apex Legends》全球性质的最高级别的职业赛事，由主办方ea即Electronic Atrs举办，旨在推广游戏，发扬竞技精神并保持玩家对于游戏的热情。而“Apex Legends Global Series 2022-2023”为本赛事系列的第三季，随着全球各大赛区区决赛的结束，各赛区所拥有的参与决赛的队伍已经选出，决赛将于今年夏季7月13号开始于伦敦举办。因为游戏本体《Apex Legends》在中国大陆内并没有取得版号，因此纸面上来说“ALGS”是不允许大陆内选手参赛的。而这对于大陆内的热衷于《Apex Legends》的各个选手、主播等要实现参加“ALGS”赛事并且成功晋级决赛是一件非常困难的事情。但是今年，有两只来自CN的队伍(通过注册队伍在香港或是台湾地区)DF(Dream Fire)与MDYW(MDY White)通过亚太南赛区的赛事成功晋级决赛，因此受各位坚持不懈的选手所激励，本项目意在分析2019-2023年《Apex Legends》全球各赛区战队及队员的数据，来对“ALGS”在全球的影响力、各个地区队伍队员数量、总奖金额度等等进行总结与梳理，并简要地对今年“ALGS”决赛的的走向作出大致的预测。

1. 项目内容

本项目基于网站liquipedia中的“Apex Legends”板块展开研究，首先通过python网络爬虫对需要的数据进行爬取，得到Apex目前全球范围内所拥有的队伍以及选手信息，目前个人价值排名前20的选手个人信息以及2019-2023年排名前50的队伍与选手的信息；再通过数据库进行数据的储存；最终进行分析作出对“ALGS”在全球的影响力、各个地区队伍队员数量、总奖金额度等等的总结与梳理，并简要地对今年“ALGS”决赛的的走向作出大致的预测。(Ps:由于liquipedia对标的是WiKi性质的网站，不由官方直接下场运营，因此或多或少在某些地方会出现数据更新的延迟或是纰漏，所以对得到的数据本项目仅以大致的总结以及预测为主)

1. 项目组织框架

预测结果

分析对比处理数据

Mysql

将数据储存至数据库中

进行数据的预处理

Apex全球赛区队伍与选手数据以及2019-2023年间数据

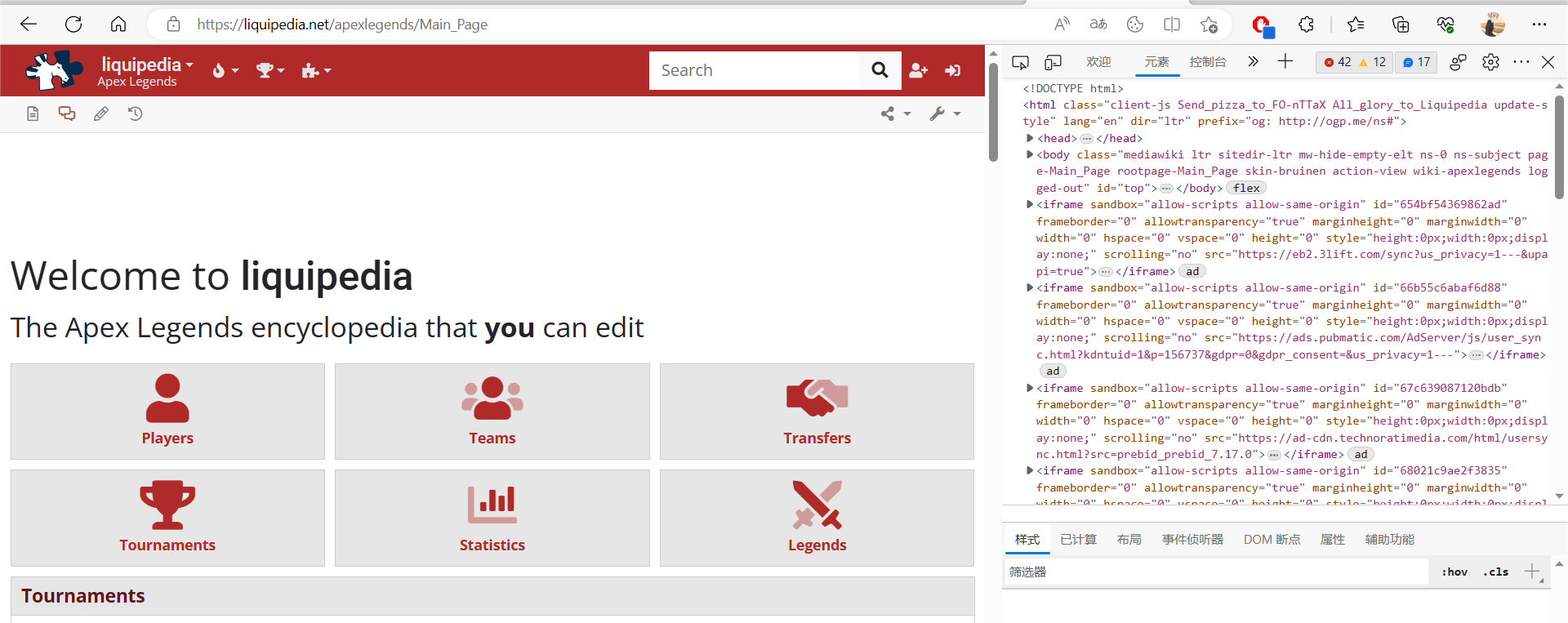
爬取数据

设计网络爬虫

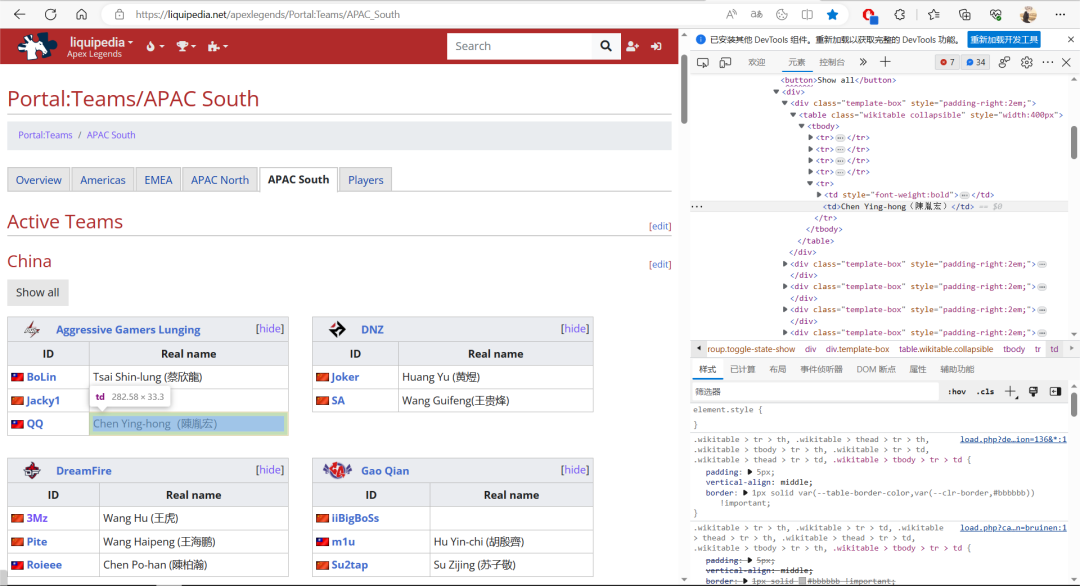
分析

平台liquipedia

1. 数据采集过程分析



首先是liquipedia网站的主界面，从中很清晰地可以得知我们所需要的队伍(Teams)、选手(Players)、2019-2023年间的数据(Statistics)分别位于网站的哪个位置，优先选择的是从Teams开始爬取。



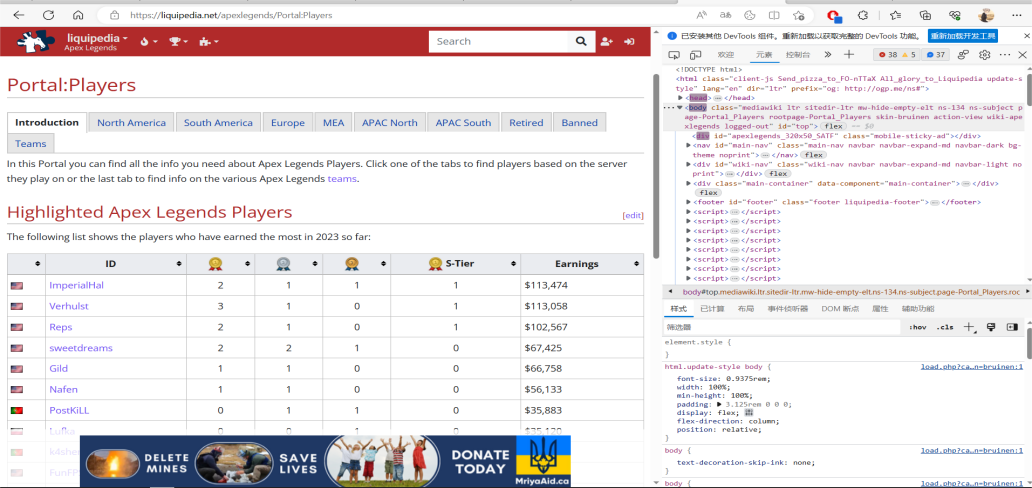
进入Teams界面之后，通过对网页源码的刷新可以得到爬虫所需要的headers，而且从中可以看出在带有[show][hide]标签的部分队伍下可以得到其队伍所拥有的队员ID以及部分选手的真实姓名。

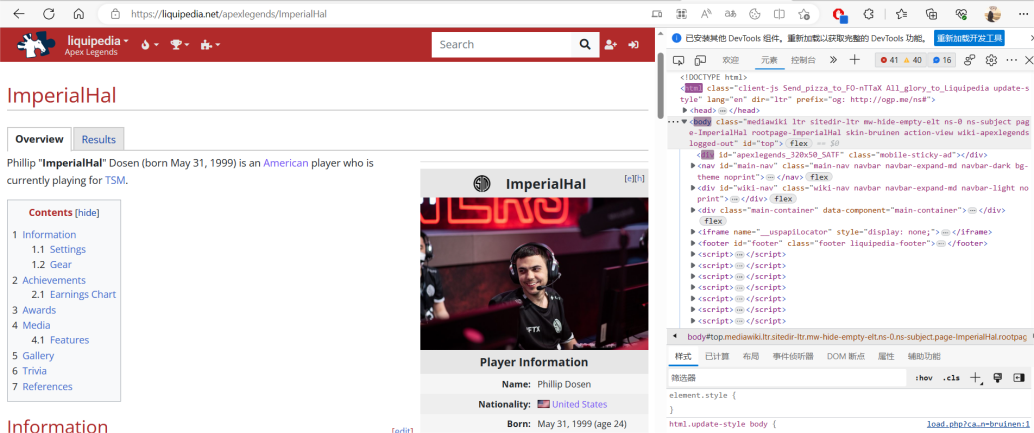
|  |
| --- |
| #爬取全球各赛区战队名称  def getTeam(html):      team\_list = []      soup = BeautifulSoup(html,'html.parser')      team\_list.append((soup.find('li',{'class':'active'})).find('a').string)      div = soup.find\_all('span',{'class':'team-template-text'})      for li in div:          a = li.find('a')          text = a.string          team\_list.append(text)        return team\_list |
| #爬取各战队所拥有队员ID  def getplayerID(html):      Id = []      soup = BeautifulSoup(html,'html.parser')      div = soup.find\_all('span',style="white-space:pre")      for li in div:          a = li.find('a')          text = a.string          Id.append(text)      return Id |
| #爬取各战队所拥有队员真实姓名  def getplayerName(html):      Real\_Name = []      a = re.findall(r'<td>(.\*?)</td>', html, re.I|re.M)      for li in a:          if li == "":              li = "UNKNOWN"          text = li          Real\_Name.append(text)      return Real\_Name |

因此通过对Team、ID、Real\_Name所分别对应的标签编写其对应的函数，其中将真实姓名未知的用“UNKONWN”填充，考虑到后续有将队伍与选手的数据分开进行处理的需求，因此爬取原数据时选择分开爬取。之后采用csv将爬取的数据储存在csv格式的文件‘Apex Legends Global Teams and Players’中。

|  |
| --- |
| csvf = open('Apex Legends Global Teams and Players.csv','w',encoding='utf-8')  fw = csv.writer(csvf)  ......  csvf.close() |

接着选择进入Players界面爬取个人价值排名前20的选手的个人信息，通过选手ID的链接进入到选手个人信息界面中可以发现每个选手所对应的网址均为“https://liquipedia.net/apexlegends/”+选手的ID，因此我们先从Introcution界面爬取Top20选手的ID将其存为一个列表，再构造一个函数使得在20次的循环中每次的url=“https://liquipedia.net/apexlegends/”+ID列表[i]来达到网站每次的变动，但在实际爬取中发现部分选手的ID存在空格使得得到的网址错误，因此采取了repalce的方法将ID中的空格全部转化为下划线\_，最终得到的网址正确。

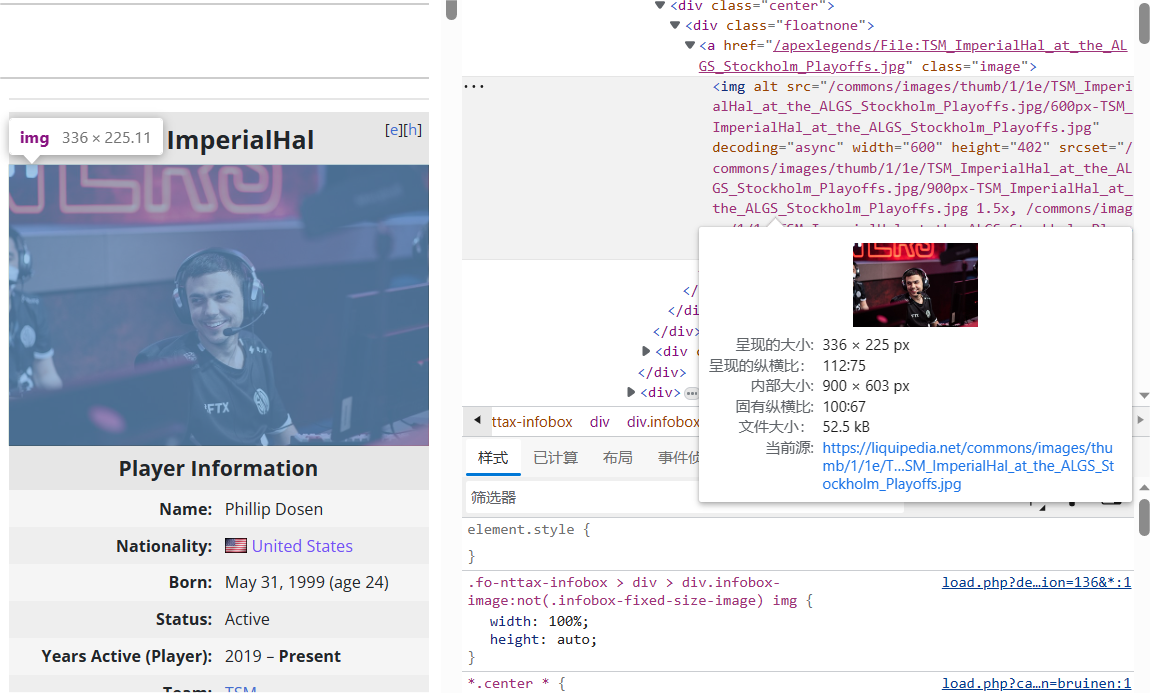




|  |
| --- |
| #爬取个人价值排名前20的选手  def getHighlightedPlayer(html):      HighlightedPlayer\_list = []      soup = BeautifulSoup(html,'html.parser')      div = soup.find\_all('tr',style="line-height:25px")      line = re.findall(r'<td><a[^>]\*>(.\*?)</a></td>', html, re.I|re.M)      for li in line:          HighlightedPlayer\_list.append(li)      return(HighlightedPlayer\_list) |
| #爬取高价值选手的个人信息  def getPlayerInf(html):      HighlightedPlayer = getHighlightedPlayer(html)      Information = []      for i in range(20):          HighlightedPlayer[i] = HighlightedPlayer[i].replace(' ','\_')          url = 'https://liquipedia.net/apexlegends/'+HighlightedPlayer[i]          res = requests.get(url=url,headers=Headers)          soup = BeautifulSoup(res.text,'html.parser')          div1 = soup.find\_all('p')          a = div1[0].text          Information.append(a)      return Information |

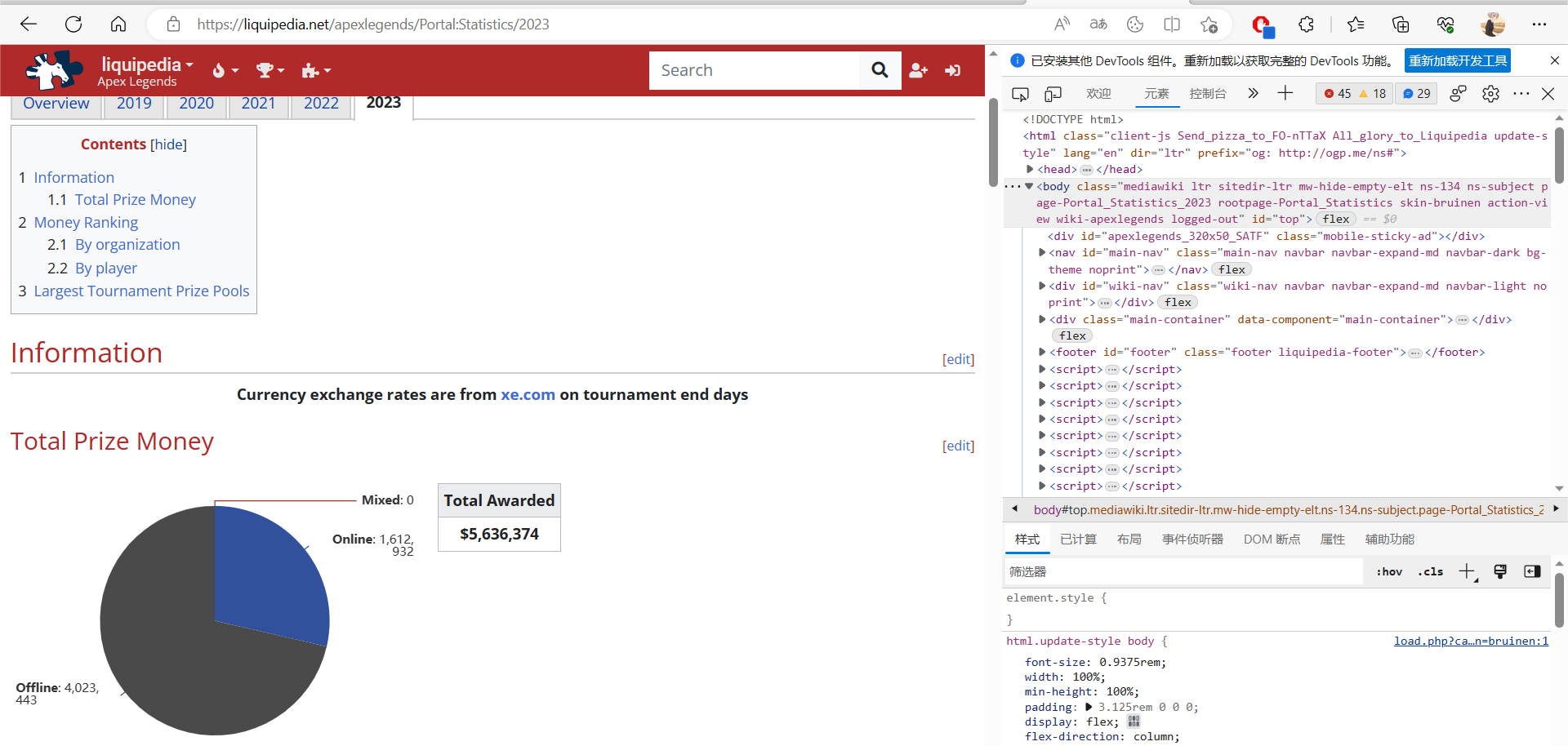
成功获得选手个人信息网址后选择爬取选手的真实姓名、国籍、出生日期、所属队伍、近10次比赛的履历，储存在csv格式的文件‘Top 20 Players in Apex Legends’中。

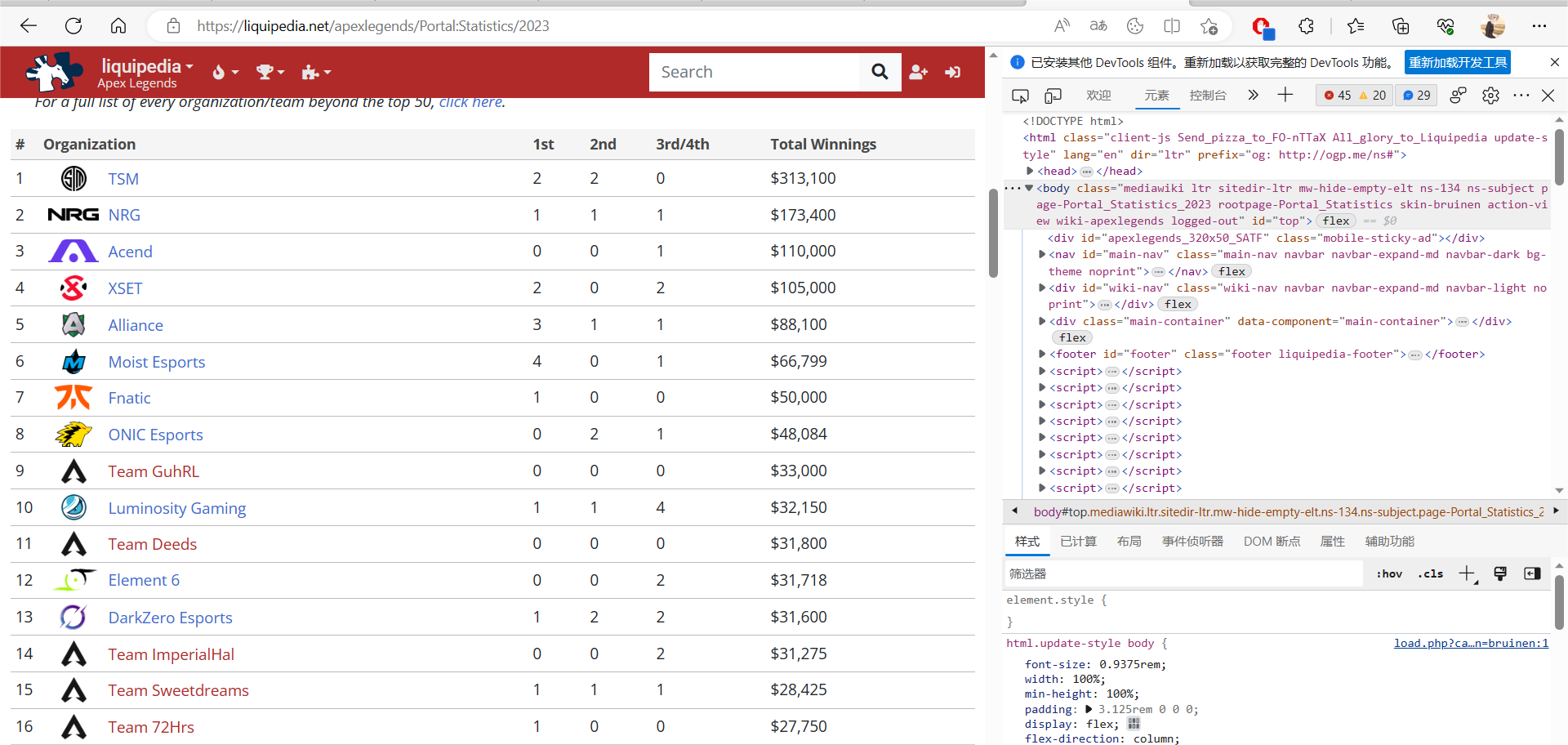
接着对选手个人照片进行爬取，同样按照上述方式得到选手个人网址后找到网页中的<img>标签并得到标签中的<src>内容也就是图片所对应的网址链接，通过分析网址得到图片网址为“https://liquipedia.net/”+src，构造函数爬取图片并分别储存在20张jpg格式的文件中。



|  |
| --- |
| def getsrc(html):      HighlightedPlayer = getHighlightedPlayer(html)      src = []      for i in range(20):          HighlightedPlayer[i] = HighlightedPlayer[i].replace(' ','\_')          url = 'https://liquipedia.net/apexlegends/'+HighlightedPlayer[i]          res = requests.get(url=url,headers=Headers)          soup = BeautifulSoup(res.text,'html.parser')          divp = soup.find('a',{'class':'image'}).find('img').get('src')          src.append(divp)      return src  print(getsrc(response5.text)) |
| #爬取个人价值前20名选手的照片  def getPictures(url,path):      res = requests.get(url=url,headers=Headers)      with open(path,'wb')as f:          for chunk in res.iter\_content(chunk\_size=128):              f.write(chunk)  def getTop20PlayerPictures(html):      src\_list = getsrc(html)      HighlightedPlayer = getHighlightedPlayer(html)      for i in range(20):          urlp = 'https://liquipedia.net'+src\_list[i]          path = HighlightedPlayer[i]+'.jpg'          getPictures(urlp,path) |

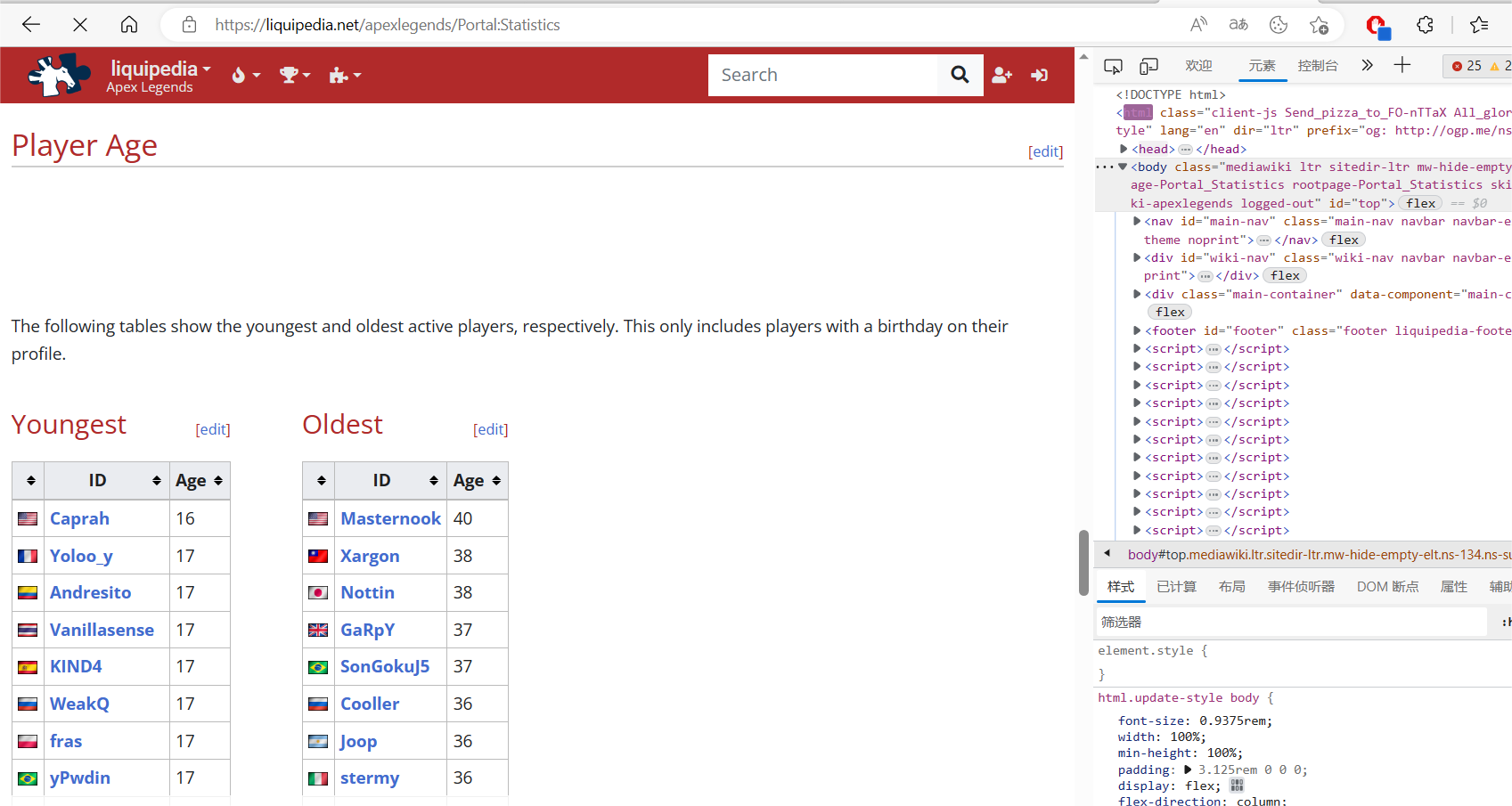
而后爬取的是2019-2023年间的数据，发现2019-2023年间数据对应的网页为“https://liquipedia.net/apexlegends/Portal:Statistics/”+年份，因此采用format(i)方法令i = 2019~2023得到各年份对应的网址，而后每年份各列出的Top50的队伍以及选手，将队伍/选手,1st,2nd,3rd/4th,(S-Tier),TotalWinnings(Earnings)信息存入到一个列表之中，爬取信息，分别储存到csv格式的文件‘2019-2023 Team Data’‘2019-2023 Player Data’中。





|  |
| --- |
| #爬取2019-2023年Top50的战队及队员的近年数据  def getTeamData():      TeamData = []      PlayerData = []      for i in range(2019,2024):          urldata = 'https://liquipedia.net/apexlegends/Portal:Statistics/{}'.format(i)          res = requests.get(url=urldata,headers=Headers)          soup = BeautifulSoup(res.text,'html.parser')          divt = soup.find\_all('div',{'class':"divRow"})          for li in divt:              team = li.find('span',{'class':"team-template-text"}).find('a')              \_1st = li.find\_all('div',{'class':"divCell"})[2].text              \_2nd = li.find\_all('div',{'class':"divCell"})[3].text              \_3rd\_4th = li.find\_all('div',{'class':"divCell"})[4].text              Totalwinnings = li.find\_all('div',{'class':"divCell"})[5].text              TeamData.append(team.string+" owns ["+\_1st+" 1st] ["+\_2nd+" 2nd] ["+\_3rd\_4th+" 3rd/4th] ["+Totalwinnings+" Totalwinnings]")      return TeamData  def getPlayerData():      PlayerData = []      for i in range(2019,2024):          urldata = 'https://liquipedia.net/apexlegends/Portal:Statistics/{}'.format(i)          res = requests.get(url=urldata,headers=Headers)          soup = BeautifulSoup(res.text,'html.parser')          divp = soup.find\_all('tr',style="line-height:25px")          for li in divp:              player = li.find\_all('td')[1].find('a')              \_1st = li.find\_all('td',style="text-align:center")[0].text              \_2nd = li.find\_all('td',style="text-align:center")[1].text              \_3rd\_4th = li.find\_all('td',style="text-align:center")[2].text              \_S\_Tier = li.find\_all('td',style="text-align:center")[3].text              Earnings = li.find\_all('td')[6].string              PlayerData.append(player.string+" owns ["+\_1st+" 1st] ["+\_2nd+" 2nd] ["+\_3rd\_4th+" 3rd/4th] ["+\_S\_Tier+" S-Tier] ["+Earnings+" Earnings]")      return PlayerData |

最后爬取的是网站中所有队伍、选手总体的信息表，以及目前最年轻与最年长的各20位选手。



|  |
| --- |
| #爬取所有队伍与选手信息  def getTeamOverall():      TeamOverall = []      urldata = 'https://liquipedia.net/apexlegends/Portal:Statistics/Team\_earnings'      res = requests.get(url=urldata,headers=Headers)      soup = BeautifulSoup(res.text,'html.parser')      divt = soup.find('div',{'class':"mw-parser-output"}).find('tbody').find\_all('tr')      i = 0      for li in divt:          if i != 0:              team = li.find('span',{'class':"team-template-text"}).find('a').string              \_1st = li.find\_all('td',style="text-align:center")[0].text              \_2nd = li.find\_all('td',style="text-align:center")[1].text              \_3rd\_4th = li.find\_all('td',style="text-align:center")[2].text              \_S\_Tier = li.find\_all('td',style="text-align:center")[3].text              Totalwinnings = li.find\_all('td')[5].text              TeamOverall.append(team+" owns ["+\_1st+" 1st] ["+\_2nd+" 2nd] ["+\_3rd\_4th+" 3rd/4th] ["+\_S\_Tier+" S-Tier] ["+Totalwinnings+" Totalwinnings]")          i += 1      return TeamOverall |

爬取思路大致相同，需要注意的是每次爬取过程中循环中第一个爬到的数据是NONE，因此采用i作为标记来跳过。爬取完数据后将数据存储为csv格式的文件“TeamOverall”“PlayerOverall”“Player Age”

爬取完数据之后将上述数据全部整理至‘(MySQL)Apex Legends Global Teams and Players’中并分为十四张表，通过pymysql以及xlrd库对数据进行储存，创建名为Apex的数据库并根据每张表创建对应的数据表以及各个字段，其中2019-2023年间的数据以每50为分界线分别储存。

|  |
| --- |
| for year in range(2019,2024):      sqlT = 'drop table if exists Apex\_Team'+str(year)      sqlP = 'drop table if exists Apex\_Player'+str(year)      cursor.execute(sqlT)      cursor.execute(sqlP)      sqlT = 'create table if not exists Apex\_Team'+str(year)+' (Team varchar(255),1st int,2nd int,3rd\_4th int,Totalwinnings varchar(255))'      sqlP = 'create table if not exists Apex\_Player'+str(year)+' (ID varchar(255),1st int,2nd int,3rd\_4th int,S\_Tier int,Earnings varchar(255))'      cursor.execute(sqlT)      cursor.execute(sqlP) |
| def getExcelYearTeam(year,sheet):      for row in range(5\*(year-2019),5\*(year-2019)+50):          Team = sheet.cell(row,0).value          \_1st = sheet.cell(row,1).value          \_2nd = sheet.cell(row,2).value          \_3rd\_4th = sheet.cell(row,3).value          Totalwinnings = sheet.cell(row,4).value          query = 'insert into Apex\_Team'+str(year)+'(Team,1st,2nd,3rd\_4th,Totalwinnings) values (%s,%s,%s,%s,%s)'          values = (Team,\_1st,\_2nd,\_3rd\_4th,Totalwinnings)          cursor.execute(query,values)  def getExcelYearPlayer(year,sheet):      for row in range(5\*(year-2019),5\*(year-2019)+50):          ID = sheet.cell(row,0).value          \_1st = sheet.cell(row,1).value          \_2nd = sheet.cell(row,2).value          \_3rd\_4th = sheet.cell(row,3).value          \_S\_Tier = sheet.cell(row,4).value          Earnings = sheet.cell(row,5).value          query = 'insert into Apex\_Player'+str(year)+'(ID,1st,2nd,3rd\_4th,S\_Tier,Earnings) values (%s,%s,%s,%s,%s,%s)'          values = (ID,\_1st,\_2nd,\_3rd\_4th,\_S\_Tier,Earnings)          cursor.execute(query,values) |

其中在储存Top20选手个人信息中的出生日期时发现xlsx类型中的时间格式均为float无法储存到数据库中的date格式之中，因此调用了datetime库来解决这个问题。

|  |
| --- |
| ......  Born = (sheet.cell(row,3).value)          Dt = xlrd.xldate\_as\_datetime(Born,0)  ...... |

1. 数据介绍与预处理

预处理过程中我们一共得到四张表：

‘(预处理后)Apex Legends Global Teams and Players’共有630支队伍与426名选手的信息，并且根据四个Americas,EMMA,APAC North,APAC South地区进行了分组；

‘(预处理后)Top 20 Players in Apex Legends’20名选手的ID、真实姓名、国籍、出生日期、所属队伍以及最近的赛事履历；

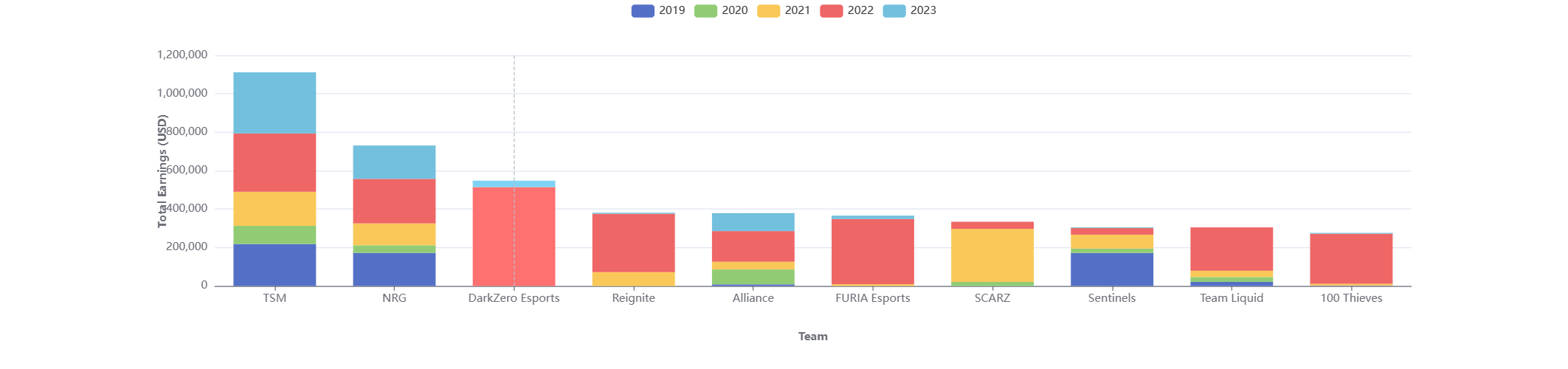
‘(MySQL)Apex Legends Global Teams and Players’用于专门储存至SQL数据库中，并且包含了各队伍队员、队员个人信息等等对应关系；

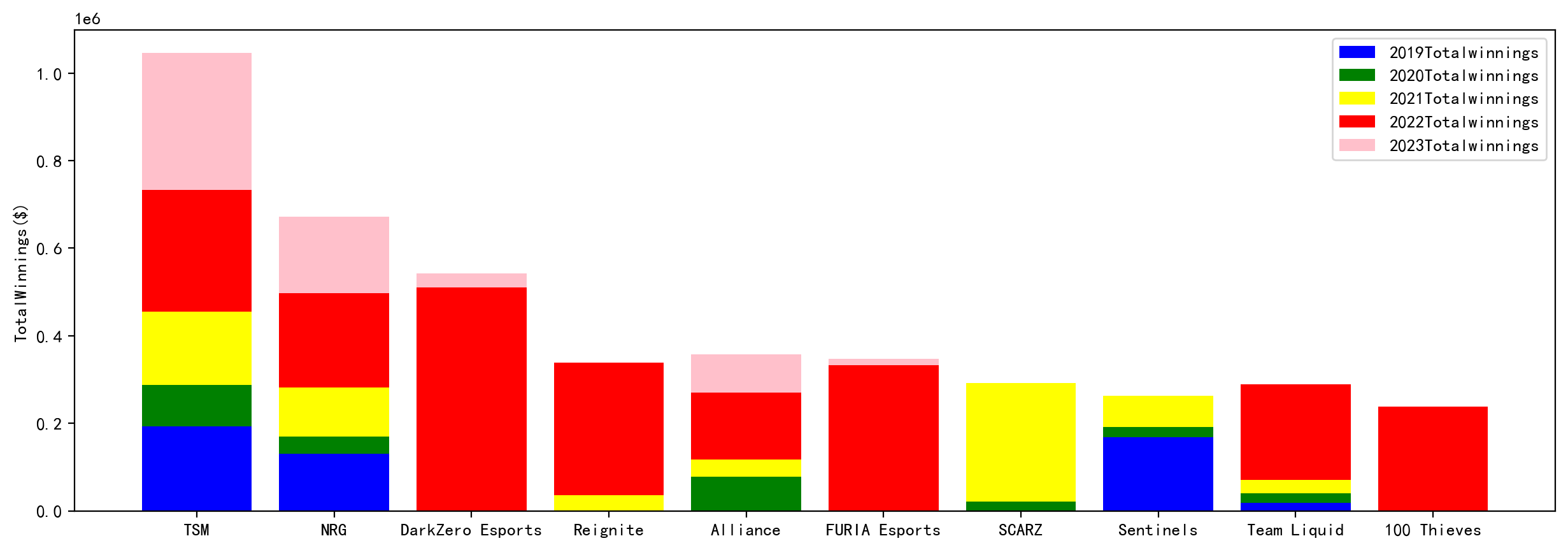
‘(预处理后)2019-2023 Data’2019-2023年间所有队伍与选手所获得奖项以及奖金等等信息。

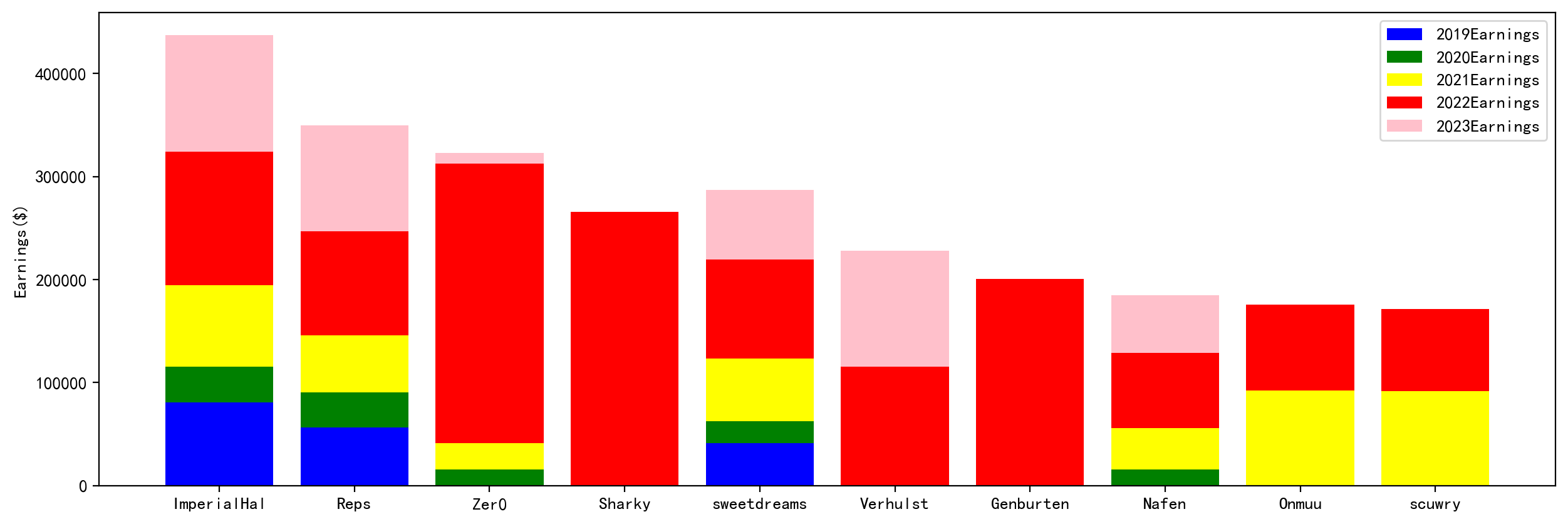
预处理过程中删去了重复的数据并填补上了漏掉的信息，并且在各赛区分组过程中将网站中没有记录队员的队伍剔除，得到了将近三千条的数据。

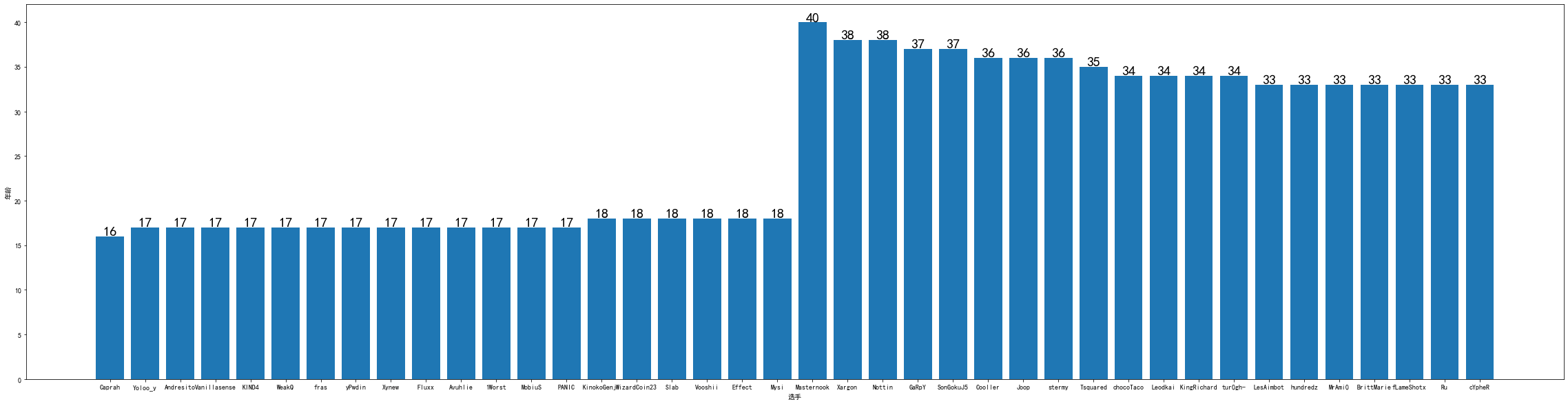
1. 数据结果可视化

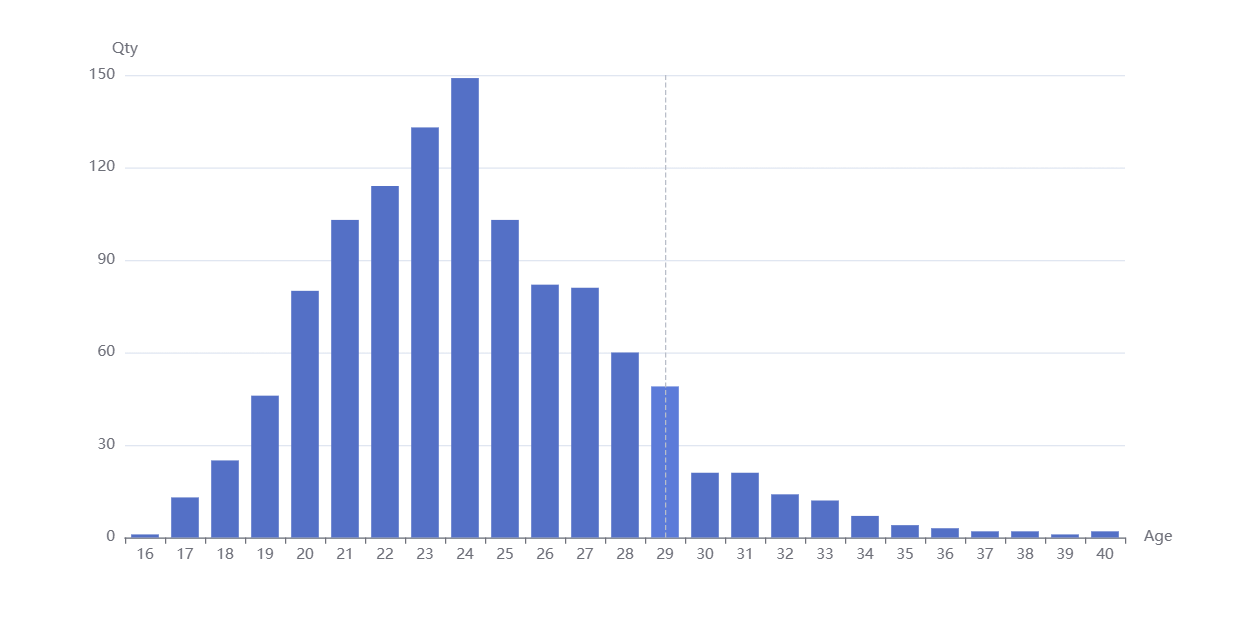
生成了2019-2023年间所有队伍中总奖金排行前10 的队伍与所有选手中总奖金排行前10的队伍的堆叠柱状图，以及最年轻和最年长的选手的年龄分布柱状图，并爬取了网站中的2019-2023年总体前10队伍奖金的分析图以及所有选手的年龄分布图用于对比。











1. 项目结果

从此次的项目中我们得到了得到Apex目前全球范围内所拥有的队伍以及选手信息，赛事的安排、频次、奖金的情报，目前个人价值排名前20的选手个人信息以及2019-2023年排名前50的队伍与选手的信息等等。从对全球各赛区的分析中，发现北美南美赛区的职业队伍一共有205支，北美占了141支南美占了64支；EMMA即欧洲与非洲中东赛区的职业队伍一共有186支，而欧洲赛区就有169支；亚太北赛区的职业队伍一共有120支，其中日本占了86支，韩国占了34支；亚太南的职业队伍一共有119支，其中中国队伍占有21支，东南亚占有78支，南亚仅有1支，而澳洲有19支。由此可以看出“ALGS”在美洲、欧洲等地明显拥有的队伍数量更多，发展的更为完善，这跟他们拥有较为完整的电竞俱乐部系统密切相关。但是北美赛区相较于南美赛区队伍数量前者超越后者两倍不止，以及EMMA赛区几乎被欧洲赛区垄断，从中甚至可以看出不同地区之间发展水平的差异。在亚太北赛区日本拥有的队伍数量也更多，说明游戏在日本的热度要高于在韩国。在亚太南赛区拥有不到完整3人队员的情况也相较于欧洲、美洲赛区更多，其中有转会频繁，没有正规俱乐部支持等种种原因。并且由于大陆地区没有版号中国的队伍明显数量较少，澳洲则是因为人口密度小因此比起其他赛区队伍数量少，大部分队伍都来自于东南亚。

2019年至2023年APEX全年间各比赛上总共花费的奖金额度分别为2019 $3138211 ，2020 $2,541,135， 2021 $6435785 ，2022 $7241189 ，2023半年间 $5636374，总计$21853773，总体上每年的金额呈现向上的趋势，并且是一笔不小的数目，足以见得目前电竞行业所具有的活力。而分析2019-2023年间队伍与选手的数据，能够发现几支队伍如TSM、NRG几乎一直保持前列，他们获得的奖金总数也远超其他队伍，而DZ(DarkZero)在2022年异军突起，一方面能够得知DZ这支队伍于2022年成立，另一方面说明其实力不容小觑；而排名Top20的选手如[ImperialHal](https://liquipedia.net/apexlegends/ImperialHal" \o "ImperialHal)、[Verhulst](https://liquipedia.net/apexlegends/Verhulst" \o "Verhulst)、[sweetdreams](https://liquipedia.net/apexlegends/Sweetdreams" \o "Sweetdreams)均来自TSM、NRG等战队、好的选手造就了好的战队，其中TSM更是上届“ALGS 21-22”的冠军。目前在全球的赛区中存在着明显的分水岭，放眼今年上半年已经举行的比赛，基本上靠前的名次与奖项都被强队所垄断，而此次入围的DF与MDYW相较于这些强队仍有差距，尤其是MDYW是初次进入世界赛。因此大致预测此次ALGS决赛前10仍然会在如TSM,NRG,DZ,Alliance,100 T,MST,Acend,XEST等强队之中，而冠军由于大逃杀类游戏赛点制的性质充满未知因素不好预测，但我仍倾向于TSM,NRG,或是Alliance能够摘得桂冠。而分析DF往年的数据来看DF成功在“ALGS 21-22”中进入最终的决赛，并且目前在所有队伍之中排名第24，队员中的三位选手也都有5年往上的比赛经验，因此客观认为DF此次有望竞争胜者组的行列。而MDYW因为初次进入世界赛必然缺乏经验，并且队伍处于中游的位置，感性上对他肯定是寄以最大的希望但理性上认为大概率还是会到败者组，但会不会被“一轮游”直接淘汰无法确定。

再看最年轻选手与最年长选手的数据，年龄分布在16-40岁之间，而年轻选手平均年龄为17.25岁，年长选手平均年龄为35岁，而所有选手的平均年龄为23.88岁，大于30岁的选手只有61人仅仅占所有选手的5.41%，大部分选手的年龄集中在20-27岁之间。这也正是电竞对于年龄上客观的严苛程度的表现。