In [ ]:
```
import time
from pybaseball import batting_stats
import pandas as pd
import numpy as np
```

In [ ]:
```
# loop through all seasons from 1903 to 2022
# and download players stats for each season
startYear = 1903
endYear = 2022
for i in np.arange(startYear,endYear+1):
    stats = batting_stats(start_season=i,
                          end_season=i,
                          league='all', # both leagues
                          qual=0, # no minimum number of at bats
                          ind=1,
                          stat_columns = ['SO','AB']) # only get strikeouts and at bats
    if(i==startYear): # write to file to ensure file is 'new'
        stats.to_csv('battingStatsRaw.csv',header=True,index=False,mode='w')
    else: # append instead of overwriteing.  Also dont include header
        stats.to_csv('battingStatsRaw.csv',header=False,index=False,mode='a')
    if(i%20==0): #print out when each decade is complete
        print(f'Done with {i}')
```

```
Done with 1920
Done with 1940
Done with 1960
Done with 1980
Done with 2000
Done with 2020
```

In [ ]:
```
df = pd.read_csv('battingStatsRaw.csv')[['Season','AB','SO']]
```

In [ ]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 87467 entries, 0 to 87466
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Season  87467 non-null  int64
 1   AB      87467 non-null  int64
 2   SO      83794 non-null  float64
dtypes: float64(1), int64(2)
memory usage: 2.0 MB
```

In [ ]:
```
df.head(5)
```

Out[ ]:

	Season	AB	SO
0	1903	3	NaN
1	1903	0	NaN
2	1903	0	NaN
3	1903	0	NaN
4	1903	6	NaN

In [ ]:
```
df.isna().sum()
```

Out[ ]:
```
Season       0
AB           0
SO        3673
dtype: int64
```

lotta Null values for strikeouts. Lets see when they appear

In [ ]:
```
# checking which seasons have null values
seasons = []
nulls = []
for season in df['Season'].unique():
    seasonData = df.loc[df['Season'] == season]
    seasons.append(season)
    nulls.append(seasonData['SO'].isna().sum())
numNull = pd.DataFrame({'Season':seasons,'Null':nulls})
numNull = numNull.loc[numNull['Null']!=0]
numNull.sort_values(by='Season',ascending=True)
```

Out[ ]:

	Season	Null
0	1903	362
1	1904	360
2	1905	385
3	1906	409
4	1907	411
5	1908	428
6	1909	497
7	1910	249
8	1911	269
9	1912	303

The seasons 1903 - 1912 have a lot of missing values for strikeouts.

I will download the season totals for those years manually https://www.baseball-reference.com/leagues/majors/bat.shtml

In [ ]:
```
reduced = df.loc[df['Season']>=1913]
reduced = reduced.astype(int)
```

In [ ]:
```
reduced.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 83028 entries, 4439 to 87466
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Season  83028 non-null  int32
 1   AB      83028 non-null  int32
 2   SO      83028 non-null  int32
dtypes: int32(3)
memory usage: 1.6 MB
```

In [ ]:
```
reduced.to_csv('battingStatsClean1913to2022.csv',index=False)
```

Now I need to get the season level totals for 1913-2022.

In [ ]:
```
summary = reduced.groupby(['Season']).sum().reset_index()
```

In [ ]:
```
summary.head(5)
```

Out[ ]:

	Season	AB	SO
0	1913	81213	9282
1	1914	122489	14743
2	1915	121704	14020
3	1916	81923	9534
4	1917	82055	8680

Now I will load in the data for 1903-1912.

This was downloaded manually and saved to a csv file

In [ ]:
```
summary1912 = pd.read_csv('1903to1912.csv')
```

In [ ]:
```
summary1912.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Season  10 non-null     int64
 1   AB      10 non-null     int64
 2   SO      10 non-null     int64
dtypes: int64(3)
memory usage: 368.0 bytes
```

In [ ]:
```
summary1912.tail(5)
```

Out[ ]:

	Season	AB	SO
5	1908	80679	9078
6	1909	80613	9377
7	1910	81551	9677
8	1911	82259	9871
9	1912	82039	9684

In [ ]:
```
allSeasonsSummary = pd.concat([summary1912, summary])
```

In [ ]:
```
allSeasonsSummary
```

Out[ ]:

	Season	AB	SO
0	1903	75439	7899
1	1904	82488	9299
2	1905	81842	9523
3	1906	80061	9110
4	1907	80304	8836
...	...	...	...
105	2018	165432	41207
106	2019	166651	42823
107	2020	59030	15586
108	2021	161941	42145
109	2022	163465	40812

120 rows × 3 columns

In [ ]:
```
# calculate strikeouts per 100 at bats
allSeasonsSummary['Rate'] = round((allSeasonsSummary['SO'] / allSeasonsSummary['AB'])*100,2)
```

In [ ]:
```
allSeasonsSummary.tail()
```

Out[ ]:

	Season	AB	SO	Rate
105	2018	165432	41207	24.91
106	2019	166651	42823	25.70
107	2020	59030	15586	26.40
108	2021	161941	42145	26.02
109	2022	163465	40812	24.97

In [ ]:
```
allSeasonsSummary.to_csv('seasonSummaries.csv', index=False)
```