In [1]:
```
import pandas as pd
from bs4 import BeautifulSoup as bs
import requests
```

Get every NBA Team

In [2]:
```
url="https://www.basketball-reference.com/"
```

In [3]:
```
east=pd.read_html(url)[0]
west=pd.read_html(url)[1]
```

In [4]:
```
east=east[["East","W","L"]]
west=west[["West","W","L"]]
```

In [5]:
```
east["East"]=east["East"].str[:3]
west["West"]=west["West"].str[:3]
```

In [6]:
```
east, west
```

Out[6]:
```
(   East   W   L
 0   MIA  53  29
 1   BOS  51  31
 2   MIL  51  31
 3   PHI  51  31
 4   TOR  48  34
 5   CHI  46  36
 6   BRK  44  38
 7   CLE  44  38
 8   ATL  43  39
 9   CHO  43  39
 10  NYK  37  45
 11  WAS  35  47
 12  IND  25  57
 13  DET  23  59
 14  ORL  22  60,
    West   W   L
 0   PHO  64  18
 1   MEM  56  26
 2   GSW  53  29
 3   DAL  52  30
 4   UTA  49  33
 5   DEN  48  34
 6   MIN  46  36
 7   LAC  42  40
 8   NOP  36  46
 9   SAS  34  48
 10  LAL  33  49
 11  SAC  30  52
 12  POR  27  55
 13  OKC  24  58
 14  HOU  20  62)
```

Get teams info from links

In [7]:
```
url="https://www.basketball-reference.com/"
html=requests.get(url).content
soup=bs(html,"html.parser")


roster=pd.DataFrame()
totals=pd.DataFrame()

for i in ["E","W"]:

    hrefs=soup.select("div[id='all_confs_standings_"+i+"'] tbody tr th a")

    for href in hrefs:
        url="https://www.basketball-reference.com/"+href["href"]

        df1=pd.read_html(url,match="Roster")[0]
        df1["Team"]=href.text
        roster=pd.concat([roster,df1],ignore_index=True)

        df2=pd.read_html(url,match="Totals")[0]
        df2["Team"]=href.text
        totals=pd.concat([totals,df2],ignore_index=True)
```

Clean Data

In [8]:
```
import datetime as dt

roster.columns=roster.columns.str.strip().str.replace(" ","")
roster=roster.loc[:,roster.columns.isin(["Unnamed:6"])==False]
roster["BirthDate"]=pd.to_datetime(roster.BirthDate)

totals.columns=totals.columns.str.strip().str.replace(" ","")
totals=totals.loc[:,totals.columns.isin(["Unnamed:1"])==False]
totals.loc[:,totals.columns.isin(["Team"])==False]=totals.loc[:,totals.columns.isin(["Team"])==False].astype(float)
```

In [9]:
```
pd.set_option("display.max_columns",100)
```

Dataframes

In [10]:
```
roster.head(5)
```

Out[10]:

	No.	Player	Pos	Ht	Wt	BirthDate	Exp	College	Team
0	55	Duncan Robinson	SG	6-7	215	1994-04-22	3	Williams, Michigan	MIA
1	17	P.J. Tucker	PF	6-5	245	1985-05-05	10	Texas	MIA
2	31	Max Strus	SF	6-5	215	1996-03-28	2	Lewis (IL), DePaul	MIA
3	2	Gabe Vincent	PG	6-3	200	1996-06-14	2	UC Santa Barbara	MIA
4	21	Dewayne Dedmon	C	7-0	245	1989-08-12	8	USC	MIA

In [12]:
```
totals.head(5)
```

Out[12]:

	Rk	Age	G	GS	MP	FG	FGA	FG%	3P	3PA	3P%	2P	2PA	2P%	eFG%	FT	FTA	FT%	ORB	DRB	TRB	AST	STL	BLK	TOV	PF	PTS	Team
0	1.0	22.0	66.0	10.0	2151.0	501.0	1122.0	0.447	175.0	439.0	0.399	326.0	683.0	0.477	0.525	190.0	219.0	0.868	32.0	297.0	329.0	263.0	44.0	8.0	174.0	95.0	1367.0	MIA
1	2.0	35.0	63.0	63.0	2133.0	276.0	627.0	0.440	144.0	382.0	0.377	132.0	245.0	0.539	0.555	148.0	174.0	0.851	33.0	249.0	282.0	474.0	67.0	17.0	168.0	177.0	844.0	MIA
2	3.0	27.0	79.0	68.0	2043.0	290.0	726.0	0.399	232.0	624.0	0.372	58.0	102.0	0.569	0.559	51.0	61.0	0.836	24.0	179.0	203.0	129.0	42.0	14.0	60.0	201.0	863.0	MIA
3	4.0	36.0	71.0	70.0	1981.0	207.0	428.0	0.484	80.0	193.0	0.415	127.0	235.0	0.540	0.577	45.0	61.0	0.738	100.0	287.0	387.0	149.0	58.0	15.0	66.0	161.0	539.0	MIA
4	5.0	32.0	57.0	57.0	1931.0	398.0	829.0	0.480	27.0	116.0	0.233	371.0	713.0	0.520	0.496	396.0	455.0	0.870	102.0	234.0	336.0	312.0	94.0	27.0	121.0	88.0	1219.0	MIA

Upload data to MySQL database

In [17]:
```
import mysql.connector

mydb=mysql.connector.connect(host="localhost",user="root",password="password")
mycursor=mydb.cursor()

mycursor.execute("drop database if exists nba")
mycursor.execute("create database nba")

mydb=mysql.connector.connect(host="localhost",user="root",password="password",database="nba")
mycursor=mydb.cursor()
```

In [19]:
```
dfs=[roster,totals]
for df,n in zip(dfs,range(1,len(dfs)+1)):


    df = df.where(pd.notnull(df), 0)
    df.columns=df.columns.str.strip().str.lower().str.replace(" |:","",regex=True).str.replace("no.","no",regex=True).str.replace("%","percent",regex=True)   # different dataframes need different approach

    columns1 = ' varchar(255), '.join([str(elem) for elem in df.columns])

    drop_table="drop table if exists df"+str(n)
    create_table="create table "+"df"+str(n)+"(ID int auto_increment primary key, "+columns1+" varchar(255))"

    mycursor.execute(drop_table)
    mycursor.execute(create_table)

    columns2 = ', '.join([str(elem) for elem in df.columns])
    values=len(df.columns)*"%s,"
    insert_into="insert into "+"df"+str(n)+"("+columns2+")"+" values("+values[:-1]+")"
    columns3=[]
    for i in df.columns:
        i="row."+str(i)
        columns3.append(i)
    for row in df.itertuples():
        mycursor.execute(insert_into,(columns3))
mydb.commit()
```