

```
In [162]: import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [17]: athlete_data=pd.read_csv('1. Athlete_Events.csv')
region_data=pd.read_csv('2. NOC_Regions.csv')
```

```
In [18]: athlete_data.head(3)
```

```
Out[18]:
```

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	
0	1	A Dijiang	M	24.0	180.0	80.0	China	CHN	1992 Summer	1992	Summer	B
1	2	A Lamusi	M	23.0	170.0	60.0	China	CHN	2012 Summer	2012	Summer	
2	3	Gunnar Nielsen Aaby	M	24.0	NaN	NaN	Denmark	DEN	1920 Summer	1920	Summer	An

```
In [19]: athlete_data.shape
```

```
Out[19]: (271116, 15)
```

Going to work only for Summer Olympics

```
In [20]: df=athlete_data[athlete_data['Season']=='Summer']
```

```
In [21]: df.shape
```

```
Out[21]: (222552, 15)
```

```
In [22]: region_data.head()
```

```
Out[22]:
```

	NOC	region	notes
0	AFG	Afghanistan	NaN
1	AHO	Curacao	Netherlands Antilles
2	ALB	Albania	NaN
3	ALG	Algeria	NaN
4	AND	Andorra	NaN

```
In [23]: df=df.merge(region_data,on='NOC',how='left')
```

In [24]: df

Out[24]:

	ID	Name	Sex	Age	Height	Weight		Team	NOC	Game
0	1	A Dijiang	M	24.0	180.0	80.0		China	CHN	199 Summe
1	2	A Lamusi	M	23.0	170.0	60.0		China	CHN	201 Summe
2	3	Gunnar Nielsen Aaby	M	24.0	NaN	NaN		Denmark	DEN	192 Summe
3	4	Edgar Lindenau Aabye	M	34.0	NaN	NaN	Denmark/Sweden	DEN		190 Summe
4	8	Cornelia "Cor" Aalten (- Strannood)	F	18.0	168.0	NaN		Netherlands	NED	193 Summe
...	
222547	135565	Fernando scar Zylberberg	M	27.0	168.0	76.0		Argentina	ARG	200 Summe
222548	135566	James Francis "Jim" Zylker	M	21.0	175.0	75.0		United States	USA	197 Summe
222549	135567	Aleksandr Viktorovich Zyuzin	M	24.0	183.0	72.0		Russia	RUS	200 Summe
222550	135567	Aleksandr Viktorovich Zyuzin	M	28.0	183.0	72.0		Russia	RUS	200 Summe
222551	135568	Olga Igorevna Zyuzkova	F	33.0	171.0	69.0		Belarus	BLR	201 Summe

222552 rows × 17 columns

In [25]: df['region'].unique().shape

Out[25]: (206,)

```
In [26]: df.isnull().sum()
```

```
Out[26]: ID          0
        Name        0
        Sex         0
        Age       9189
        Height   51857
        Weight   53854
        Team      0
        NOC       0
        Games     0
        Year      0
        Season    0
        City      0
        Sport     0
        Event     0
        Medal   188464
        region    370
        notes    218151
        dtype: int64
```

```
In [27]: df.duplicated().sum()
```

```
Out[27]: 1385
```

```
In [28]: df.drop_duplicates(inplace=True)
```

```
In [29]: df['Medal'].value_counts()
```

```
Out[29]: Medal
        Gold    11456
        Bronze   11409
        Silver   11212
        Name: count, dtype: int64
```

One Hot Encoding on medal tally

```
In [30]: df=pd.concat([df,pd.get_dummies(df['Medal'])],axis=1)
```

```
In [31]: df.groupby('NOC').sum()[['Gold','Silver','Bronze']].sort_values('Gold',ascending
```

Out[31]:

	NOC	Gold	Silver	Bronze
0	USA	2472	1333	1197
1	URS	832	635	596
2	GBR	635	729	620
3	GER	592	538	649
4	ITA	518	474	454
...
225	AHO	0	1	0
226	LBR	0	0	0
227	LCA	0	0	0
228	LES	0	0	0
229	LBA	0	0	0

230 rows × 4 columns

Gold medal analysis for team games

In [32]:

```
df[(df['NOC']=='IND') & (df['Medal']=='Gold')]
```

Out[32]:

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year
4186	2699	Shaukat Ali	M	30.0	NaN	NaN	India	IND	1928 Summer	1928
4190	2703	Syed Mushtaq Ali	M	22.0	165.0	61.0	India	IND	1964 Summer	1964
4460	2864	Richard James Allen	M	25.0	172.0	NaN	India	IND	1928 Summer	1928
4461	2864	Richard James Allen	M	30.0	172.0	NaN	India	IND	1932 Summer	1932
4462	2864	Richard James Allen	M	34.0	172.0	NaN	India	IND	1936 Summer	1936
...
191759	117232	Dung Dung Sylvanus	M	31.0	160.0	62.0	India	IND	1980 Summer	1980
193964	118553	Carlyle Carrol Tapsell	M	23.0	182.0	NaN	India	IND	1932 Summer	1932
193965	118553	Carlyle Carrol Tapsell	M	27.0	182.0	NaN	India	IND	1936 Summer	1936
205949	125560	Max "Maxie" Vaz	M	NaN	NaN	NaN	India	IND	1948 Summer	1948
219336	133554	Sayed Muhammad Yusuf	M	NaN	NaN	NaN	India	IND	1928 Summer	1928

131 rows × 20 columns



Solution is quite simple. Remove all the rows with sam Team, NOC, Games, Year and Sport. Count them one time.

In [37]: `medal_tally=df.drop_duplicates(subset=['Team','NOC','Year','Games','Sport','Even`

In [38]: `medal_tally.groupby('NOC').sum()[['Gold','Silver','Bronze']].sort_values('Gold',`

Out[38]:

	NOC	Gold	Silver	Bronze
0	USA	1035	802	708
1	URS	394	317	294
2	GBR	278	317	300
3	GER	235	261	283
4	FRA	234	256	287
...
225	AHO	0	1	0
226	LBR	0	0	0
227	LCA	0	0	0
228	LES	0	0	0
229	LBA	0	0	0

230 rows × 4 columns

```
In [42]: Years=df['Year'].unique().tolist()
```

```
In [45]: Years.sort()
```

```
In [46]: Years.insert(0, 'Overall')
```

```
In [47]: Years
```

```
Out[47]: ['Overall',
1896,
1900,
1904,
1906,
1908,
1912,
1920,
1924,
1928,
1932,
1936,
1948,
1952,
1956,
1960,
1964,
1968,
1972,
1976,
1980,
1984,
1988,
1992,
1996,
2000,
2004,
2008,
2012,
2016]
```

```
In [54]: Country=df['region'].unique().tolist()
```

```
In [55]: Country.sort()
```

```
-----
TypeError                                Traceback (most recent call last)
Cell In[55], line 1
----> 1 Country.sort()

TypeError: '<' not supported between instances of 'float' and 'str'
```

```
In [56]: Country=np.unique(df['region'].dropna().values).tolist()
```

```
In [57]: Country.sort()
```

```
In [58]: Country.insert(0,'Overall')
```

```
In [59]: Country
```

```
Out[59]: ['Overall',
          'Afghanistan',
          'Albania',
          'Algeria',
          'American Samoa',
          'Andorra',
          'Angola',
          'Antigua',
          'Argentina',
          'Armenia',
          'Aruba',
          'Australia',
          'Austria',
          'Azerbaijan',
          'Bahamas',
          'Bahrain',
          'Bangladesh',
          'Barbados',
          'Belarus',
          'Belgium',
          'Belize',
          'Benin',
          'Bermuda',
          'Bhutan',
          'Bolivia',
          'Bosnia and Herzegovina',
          'Botswana',
          'Brazil',
          'Brunei',
          'Bulgaria',
          'Burkina Faso',
          'Burundi',
          'Cambodia',
          'Cameroon',
          'Canada',
          'Cape Verde',
          'Cayman Islands',
          'Central African Republic',
          'Chad',
          'Chile',
          'China',
          'Colombia',
          'Comoros',
          'Cook Islands',
          'Costa Rica',
          'Croatia',
          'Cuba',
          'Curacao',
          'Cyprus',
          'Czech Republic',
          'Democratic Republic of the Congo',
          'Denmark',
          'Djibouti',
          'Dominica',
          'Dominican Republic',
          'Ecuador',
          'Egypt',
          'El Salvador',
          'Equatorial Guinea',
          'Eritrea',
```


'Estonia',
'Ethiopia',
'Fiji',
'Finland',
'France',
'Gabon',
'Gambia',
'Georgia',
'Germany',
'Ghana',
'Greece',
'Grenada',
'Guam',
'Guatemala',
'Guinea',
'Guinea-Bissau',
'Guyana',
'Haiti',
'Honduras',
'Hungary',
'Iceland',
'India',
'Individual Olympic Athletes',
'Indonesia',
'Iran',
'Iraq',
'Ireland',
'Israel',
'Italy',
'Ivory Coast',
'Jamaica',
'Japan',
'Jordan',
'Kazakhstan',
'Kenya',
'Kiribati',
'Kosovo',
'Kuwait',
'Kyrgyzstan',
'Laos',
'Latvia',
'Lebanon',
'Lesotho',
'Liberia',
'Libya',
'Liechtenstein',
'Lithuania',
'Luxembourg',
'Macedonia',
'Madagascar',
'Malawi',
'Malaysia',
'Maldives',
'Mali',
'Malta',
'Marshall Islands',
'Mauritania',
'Mauritius',
'Mexico',
'Micronesia',

'Moldova',
'Monaco',
'Mongolia',
'Montenegro',
'Morocco',
'Mozambique',
'Myanmar',
'Namibia',
'Nauru',
'Nepal',
'Netherlands',
'New Zealand',
'Nicaragua',
'Niger',
'Nigeria',
'North Korea',
'Norway',
'Oman',
'Pakistan',
'Palau',
'Palestine',
'Panama',
'Papua New Guinea',
'Paraguay',
'Peru',
'Philippines',
'Poland',
'Portugal',
'Puerto Rico',
'Qatar',
'Republic of Congo',
'Romania',
'Russia',
'Rwanda',
'Saint Kitts',
'Saint Lucia',
'Saint Vincent',
'Samoa',
'San Marino',
'Sao Tome and Principe',
'Saudi Arabia',
'Senegal',
'Serbia',
'Seychelles',
'Sierra Leone',
'Slovakia',
'Slovenia',
'Solomon Islands',
'Somalia',
'South Africa',
'South Korea',
'South Sudan',
'Spain',
'Sri Lanka',
'Sudan',
'Suriname',
'Swaziland',
'Sweden',
'Switzerland',
'Syria',

```

'Taiwan',
'Tajikistan',
'Tanzania',
'Thailand',
'Timor-Leste',
'Togo',
'Tonga',
'Trinidad',
'Tunisia',
'Turkey',
'Turkmenistan',
'UK',
'USA',
'Uganda',
'Ukraine',
'United Arab Emirates',
'Uruguay',
'Uzbekistan',
'Vanuatu',
'Venezuela',
'Vietnam',
'Virgin Islands, British',
'Virgin Islands, US',
'Yemen',
'Zambia',
'Zimbabwe']

```

```
In [80]: medal_df = df.drop_duplicates(subset=['Team', 'NOC', 'Year', 'Games', 'Sport', 'Event'])
```

```
In [95]: def fetch_medal_tally(year, country):
    medal_df = df.drop_duplicates(subset=['Team', 'NOC', 'Year', 'Games', 'Sport', 'Event'])
    flag=0
    if year=='Overall' and country=='Overall':
        temp_df = medal_df

    if year=='Overall' and country!='Overall':
        flag=1
        temp_df = medal_df[medal_df['region']==country]

    if year!='Overall' and country=='Overall':
        temp_df = medal_df[medal_df['Year']==int(year)]

    if year!='Overall' and country!='Overall':
        temp_df = medal_df[(medal_df['Year']==int(year)) & (medal_df['region']==country)]

    if flag==1:
        x=temp_df.groupby('Year').sum()[['Gold', 'Silver', 'Bronze']].sort_values(ascending=False)
    else:
        x=temp_df.groupby('region').sum()[['Gold', 'Silver', 'Bronze']].sort_values(ascending=False)

    x['total'] = x['Gold'] + x['Silver'] + x['Bronze']

    print(x)
```

```
In [97]: fetch_medal_tally(year='1988', country='India')
```

```

region  Gold  Silver  Bronze  total
0  India     0       0       0      0

```

Overall Analysis

- 1. Number of editions
- 2. Number of cities
- 3. Event/Sport
- 4. Number of athletes
- 5. Participating Nations

```
In [106... df.head(2)
```

Out[106...

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games	Year	Season	
0	1	A Dijiang	M	24.0	180.0	80.0	China	CHN	1992 Summer	1992	Summer	Barcel
1	2	A Lamusi	M	23.0	170.0	60.0	China	CHN	2012 Summer	2012	Summer	Long

◀ [Progress Bar] ▶

```
In [99]: df['Year'].unique()
```

Out[99]: array([1992, 2012, 1920, 1900, 1932, 1952, 2000, 1996, 1912, 1924, 1948, 2008, 2016, 2004, 1984, 1968, 1972, 1988, 1936, 1956, 1960, 1928, 1976, 1980, 1964, 1906, 1904, 1908, 1896], dtype=int64)

```
In [104... df['Year'].unique().shape[0] - 1
```

Out[104... 28

```
In [105... df['City'].unique().shape
```

Out[105... (23,)

```
In [107... df['Sport'].unique().shape
```

Out[107... (52,)

```
In [108... df['Event'].unique().shape
```

Out[108... (651,)

```
In [118... df['Name'].unique().shape
```

Out[118... (116122,)

Participating Nations

```
In [129... nations_over_time=df.drop_duplicates(['Year','region'])['Year'].value_counts().r
```

```
In [144... nations_over_time.rename(columns={'index':'Year','count':'Number of participatin
```

```
In [145... import plotly.express as px
```

```
In [148... fig=px.line(nations_over_time,x='Year', y='Number of participating nations')
```

```
In [149... fig
```

```
In [155... df.drop_duplicates(['Year','Event'])['Year'].value_counts().reset_index().sort_v
```

Out[155...

	Year	count
28	1896	43
26	1900	90
25	1904	95
27	1906	74
23	1908	109
24	1912	107
14	1920	158
20	1924	131
22	1928	122
21	1932	131
18	1936	150
15	1948	153
19	1952	149
16	1956	151
17	1960	150
13	1964	163
12	1968	172
11	1972	193
10	1976	198
9	1980	203
8	1984	221
7	1988	237
6	1992	257
5	1996	271
4	2000	300
3	2004	301
1	2008	302
2	2012	302
0	2016	306

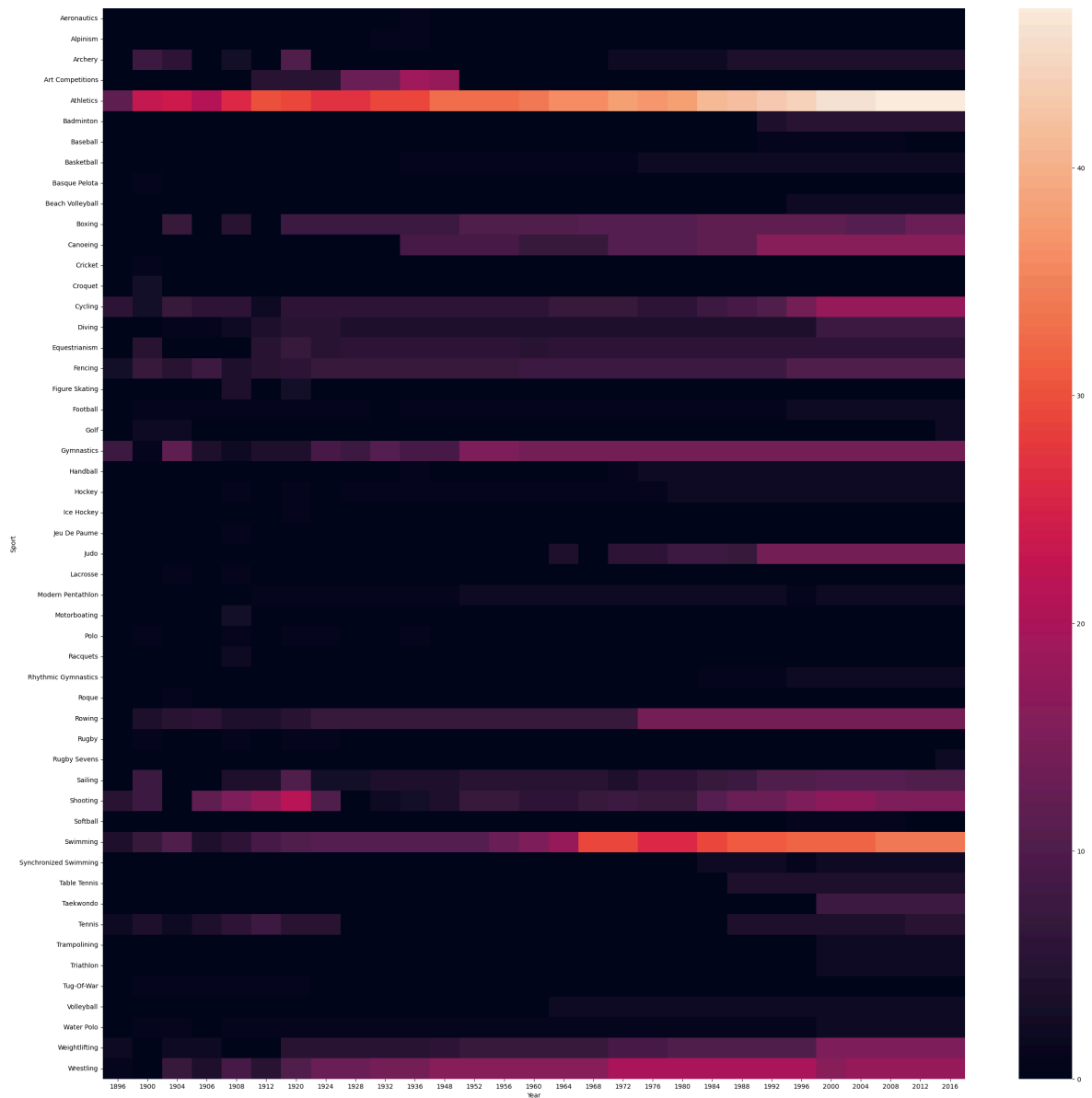
In [166...

```
x=df.drop_duplicates(['Year','Sport','Event'])
```

In [173...

```
plt.figure(figsize=(30,30))
sns.heatmap(x.pivot_table(index='Sport',columns='Year',values='Event',aggfunc='c
```

Out[173... <Axes: xlabel='Year', ylabel='Sport'>



Most successful athletes

Our objective is to create a section in sports which displays the most succesful athlete in their respective game.

```
In [237... def most_successful(df,sport):  
    temp_df=df.dropna(subset=['Medal'])  
  
    if sport != 'Overall':  
        temp_df = temp_df[temp_df['Sport']==sport]  
  
    return temp_df['Name'].value_counts()
```

```
In [239... most_successful(df,'Overall')
```

```
Out[239... Name
Michael Fred Phelps, II 28
Larysa Semenivna Latynina (Diriy-) 18
Nikolay Yefimovich Andrianov 15
Borys Anfiyanovych Shakhlin 13
Takashi Ono 13
..
Joel Isasi Gonzlez 1
Florena Andreea Isrescu 1
Ryoji Isaoka 1
Preben Isaksson 1
rpd Lengyel 1
Name: count, Length: 24545, dtype: int64
```

```
In [304... def most_successful(df,sport):
    temp_df=df.dropna(subset='Medal')

    if sport != 'Overall':
        temp_df = temp_df[temp_df['Sport']==sport]

    return temp_df['Name'].value_counts().reset_index()
```

```
In [305... most_successful(df, 'Overall')
```

```
Out[305...
               Name  count
0  Michael Fred Phelps, II    28
1  Larysa Semenivna Latynina (Diriy-)    18
2    Nikolay Yefimovich Andrianov    15
3    Borys Anfiyanovych Shakhlin    13
4              Takashi Ono    13
...              ...      ...
24540              Joel Isasi Gonzlez    1
24541    Florena Andreea Isrescu    1
24542              Ryoji Isaoka    1
24543    Preben Isaksson    1
24544              rpd Lengyel    1
```

24545 rows × 2 columns

Country-wise Analysis

1. Country wise medal tally per year
2. Countries good at heat map
3. Top 10 most successful athletes

```
In [367... temp_df = df.dropna(subset=['Medal'])
```



```
temp_df.drop_duplicates(subset=['Team', 'NOC', 'Year', 'Games', 'Sport', 'Event', 'Medal'])
```

C:\Users\HP\AppData\Local\Temp\ipykernel_17224\3343483245.py:2: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```
In [377... new_df = temp_df[temp_df['region']=='India']  
final_df = new_df.groupby('Year').count()['Medal'].reset_index()
```

```
In [378... final_df
```

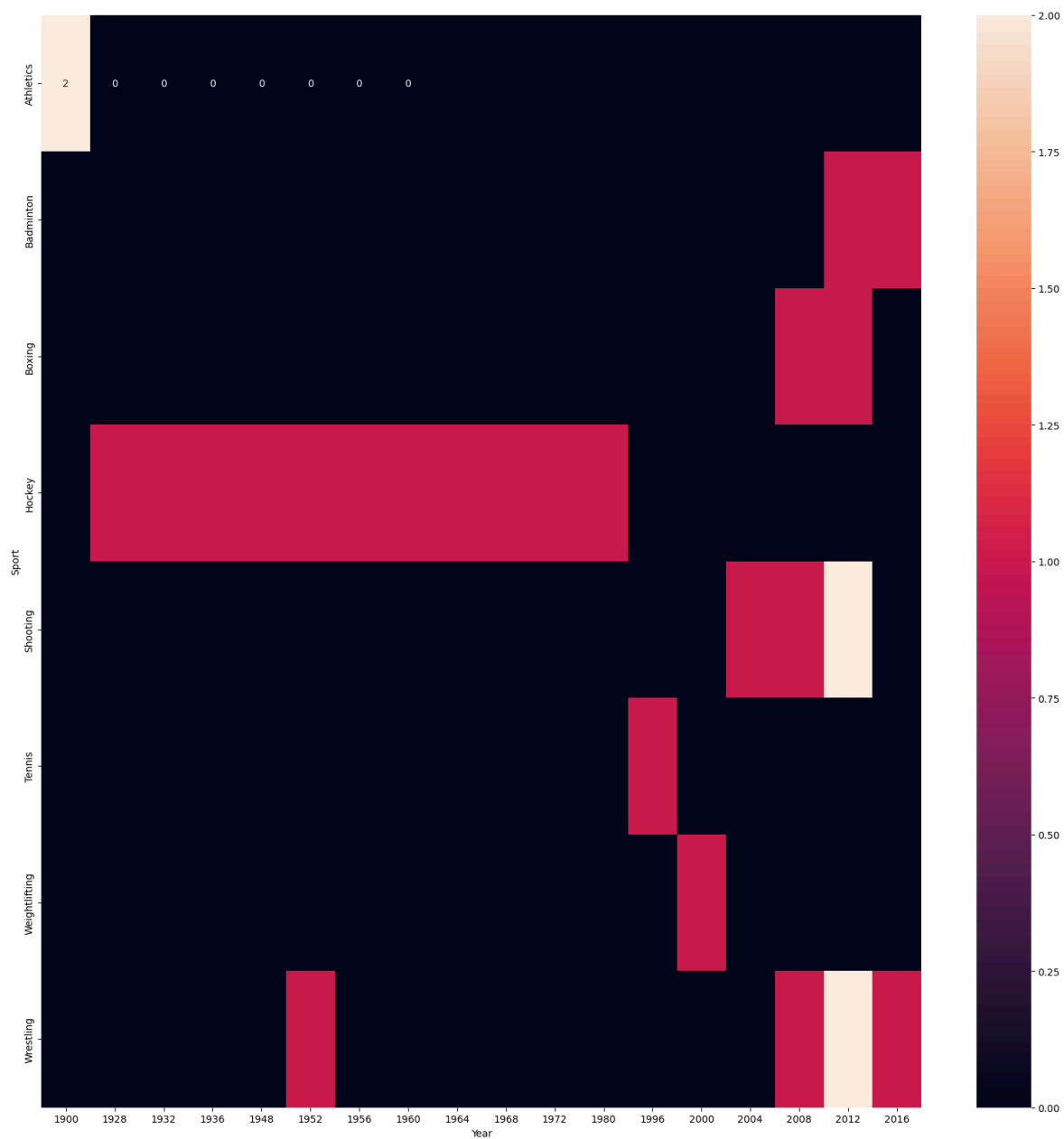
```
Out[378...      Year  Medal  
0  1900      2  
1  1928      1  
2  1932      1  
3  1936      1  
4  1948      1  
5  1952      2  
6  1956      1  
7  1960      1  
8  1964      1  
9  1968      1  
10 1972      1  
11 1980      1  
12 1996      1  
13 2000      1  
14 2004      1  
15 2008      3  
16 2012      6  
17 2016      2
```

```
In [379... fig=px.line(final_df, x='Year', y='Medal')
```

```
In [381... fig
```

```
In [382... new_df = temp_df[temp_df['region']=='India']  
plt.figure(figsize=(20,20))  
sns.heatmap(new_df.pivot_table(index='Sport',columns='Year',values='Event',aggfu
```

```
Out[382... <Axes: xlabel='Year', ylabel='Sport'>
```



In []: