RA272746_atividades5b

May 4, 2024

Exercício 1 - Cap. 22.4 Install and load the Lahman library. This database includes data related to baseball teams. It includes summary statistics about how the players performed on offense and defense for several years. It also includes personal information about the players.

Listagem dos 10 maiores rebatedores de 2016: reescrever o objeto **top** e exibir playerID, first name, last name, and number of home runs (HR)

[1]:
```
import pandas as pd

base_path = "data/lahman_1871-2023_csv/"
```

[2]:
```
top = pd.read_csv(base_path + "Batting.csv")
top.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 113799 entries, 0 to 113798
Data columns (total 24 columns):
 #   Column     Non-Null Count   Dtype  
---  ------     --------------   -----  
 0   playerID   113799 non-null  object 
 1   yearID     113799 non-null  int64  
 2   stint      113799 non-null  int64  
 3   teamID     113799 non-null  object 
 4   lgID       113062 non-null  object 
 5   G          113799 non-null  int64  
 6   G_batting  1615 non-null    float64
 7   AB         113799 non-null  int64  
 8   R          113799 non-null  int64  
 9   H          113799 non-null  int64  
 10  2B         113799 non-null  int64  
 11  3B         113799 non-null  int64  
 12  HR         113799 non-null  int64  
 13  RBI        113043 non-null  float64
 14  SB         111431 non-null  float64
 15  CS         90257 non-null   float64
 16  BB         113799 non-null  int64  
 17  SO         111699 non-null  float64
 18  IBB        77148 non-null   float64
 19  HBP        110983 non-null  float64
```

```
 20  SH          107731 non-null  float64
 21  SF          77695 non-null   float64
 22  GIDP        88357 non-null   float64
 23  G_old       0 non-null       float64
dtypes: float64(11), int64(10), object(3)
memory usage: 20.8+ MB
```

[3]:
```
top.head()
```

[3]:
```
    playerID  yearID  stint teamID lgID   G  G_batting  AB  R  H  …   SB  \
0  aardsda01    2004      1    SFN   NL  11        NaN   0  0  0  …  0.0
1  aardsda01    2006      1    CHN   NL  45        NaN   2  0  0  …  0.0
2  aardsda01    2007      1    CHA   AL  25        NaN   0  0  0  …  0.0
3  aardsda01    2008      1    BOS   AL  47        NaN   1  0  0  …  0.0
4  aardsda01    2009      1    SEA   AL  73        NaN   0  0  0  …  0.0

    CS  BB   SO  IBB  HBP   SH   SF  GIDP  G_old
0  0.0   0  0.0  0.0  0.0  0.0  0.0   0.0    NaN
1  0.0   0  0.0  0.0  0.0  1.0  0.0   0.0    NaN
2  0.0   0  0.0  0.0  0.0  0.0  0.0   0.0    NaN
3  0.0   0  1.0  0.0  0.0  0.0  0.0   0.0    NaN
4  0.0   0  0.0  0.0  0.0  0.0  0.0   0.0    NaN

[5 rows x 24 columns]
```

[4]:
```
#Filtra pelo ano atual e ordena do maior rebatedor para o menor
top = top[top["yearID"] == 2016].sort_values(by=["HR"], ascending=False)
top
```

[4]:
```
         playerID  yearID  stint teamID lgID    G  G_batting   AB    R    H  \
103772  trumbma01    2016      1    BAL   AL  159        NaN  613   94  157
22463    cruzne02    2016      1    SEA   AL  155        NaN  589   96  169
29385   encared01    2016      1    TOR   AL  160        NaN  601   99  158
24045   daviskh01    2016      1    OAK   AL  150        NaN  555   85  137
27201   doziebr01    2016      1    MIN   AL  155        NaN  615  104  165
…             …     …    …     …    …   …  …          …   …   …   …
47970    hoytja01    2016      1    HOU   AL   22        NaN    0    0    0
47965   hoyinja01    2016      1    TEX   AL   39        NaN   46    8   10
47874   howeljp01    2016      1    LAN   NL   64        NaN    2    0    0
47611   housetj01    2016      1    CLE   AL    4        NaN    0    0    0
113797   zychto01    2016      1    SEA   AL   12        NaN    0    0    0

         …    SB   CS  BB     SO  IBB  HBP   SH   SF  GIDP  G_old
103772   …   2.0  0.0  51  170.0  1.0  3.0  0.0  0.0  14.0    NaN
22463    …   2.0  0.0  62  159.0  5.0  9.0  0.0  7.0  15.0    NaN
29385    …   2.0  0.0  87  138.0  3.0  5.0  0.0  8.0  22.0    NaN
24045    …   1.0  2.0  42  166.0  0.0  8.0  0.0  5.0  19.0    NaN
27201    …  18.0  2.0  61  138.0  6.0  8.0  2.0  5.0  12.0    NaN
```

```
…      …    …   …   ..     …   …   …   …   …    …     …
47970   …   0.0  0.0   0    0.0  0.0  0.0  0.0  0.0   0.0    NaN
47965   …   1.0  0.0   3    8.0  0.0  0.0  0.0  0.0   0.0    NaN
47874   …   0.0  0.0   0    2.0  0.0  0.0  0.0  0.0   0.0    NaN
47611   …   0.0  0.0   0    0.0  0.0  0.0  0.0  0.0   0.0    NaN
113797  …   0.0  0.0   0    0.0  0.0  0.0  0.0  0.0   0.0    NaN

[1483 rows x 24 columns]
```

[5]:
```
#Reduz para os 10 melhores rebatedores
top = top[["playerID", "HR"]][:10]
top
```

[5]:
```
          playerID  HR
103772  trumbma01  47
22463    cruzne02  43
29385   encared01  42
24045   daviskh01  42
27201   doziebr01  42
16151   cartech02  41
2819    arenano01  41
33647   frazito01  40
15320    canoro01  39
12752   bryankr01  39
```

[6]:
```
top.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 10 entries, 103772 to 12752
Data columns (total 2 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   playerID  10 non-null     object
 1   HR        10 non-null     int64
dtypes: int64(1), object(1)
memory usage: 240.0+ bytes
```

[7]:
```
people = pd.read_csv(base_path + "People.csv", encoding="ISO-8859-1")
people
```

[7]:
```
           ID   playerID  birthYear  birthMonth  birthDay   birthCity  \
0           1  aardsda01     1981.0        12.0      27.0      Denver
1           2  aaronha01     1934.0         2.0       5.0      Mobile
2           3  aaronto01     1939.0         8.0       5.0      Mobile
3           4   aasedo01     1954.0         9.0       8.0      Orange
4           5   abadan01     1972.0         8.0      25.0  Palm Beach
…       …         …         …          …       …          …
21005   21006  paysojo99     1903.0         2.0       5.0    New York
```

```
21006  21007  galbrjo99     1897.0         8.0      10.0       Derby
21007  21008  mcshejo99     1944.0         9.0      11.0       Bronx
21008  21009  weyerle99     1936.0         9.0       3.0  Imlay City
21009  21010  palerst99     1949.0        10.0       9.0   Worcester

      birthCountry birthState  deathYear  deathMonth  …   nameLast  \
0              USA         CO        NaN         NaN  …    Aardsma
1              USA         AL     2021.0         1.0  …      Aaron
2              USA         AL     1984.0         8.0  …      Aaron
3              USA         CA        NaN         NaN  …       Aase
4              USA         FL        NaN         NaN  …       Abad
…              …           …         …           …    …         …
21005          USA         NY     1975.0        10.0  …     Payson
21006          USA         OH     1988.0         7.0  …  Galbreath
21007          USA         NY     1996.0         4.0  …   McSherry
21008          USA         MI     1988.0         7.0  …      Weyer
21009          USA         MA     2017.0         5.0  …    Palermo

             nameGiven weight height bats throws       debut    bbrefID  \
0          David Allan  215.0   75.0    R      R  2004-04-06  aardsda01
1          Henry Louis  180.0   72.0    R      R  1954-04-13  aaronha01
2           Tommie Lee  190.0   75.0    R      R  1962-04-10  aaronto01
3       Donald William  190.0   75.0    R      R  1977-07-26   aasedo01
4        Fausto Andres  184.0   73.0    L      L  2001-09-10   abadan01
…                  …    …    …  …    …         …         …
21005     Joan Whitney    NaN    NaN  NaN    NaN         NaN        NaN
21006      John Wilmer    NaN    NaN  NaN    NaN         NaN        NaN
21007     John Patrick  351.0   75.0  NaN    NaN         NaN        NaN
21008       Lee Howard  258.0   78.0  NaN    NaN         NaN        NaN
21009  Stephen Michael  175.0   74.0  NaN    NaN         NaN        NaN

        finalGame   retroID
0      2015-08-23  aardd001
1      1976-10-03  aaroh101
2      1971-09-26  aarot101
3      1990-10-03  aased001
4      2006-04-13  abada001
…           …       …
21005         NaN       NaN
21006         NaN       NaN
21007         NaN  mcshj901
21008         NaN  weyel901
21009         NaN  pales901

[21010 rows x 25 columns]
```

[8]:
```
people.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21010 entries, 0 to 21009
Data columns (total 25 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   ID            21010 non-null  int64
 1   playerID      21010 non-null  object
 2   birthYear     20904 non-null  float64
 3   birthMonth    20735 non-null  float64
 4   birthDay      20593 non-null  float64
 5   birthCity     20867 non-null  object
 6   birthCountry  20955 non-null  object
 7   birthState    20464 non-null  object
 8   deathYear     10266 non-null  float64
 9   deathMonth    10265 non-null  float64
 10  deathDay      10264 non-null  float64
 11  deathCountry  10264 non-null  object
 12  deathState    10204 non-null  object
 13  deathCity     10262 non-null  object
 14  nameFirst     20976 non-null  object
 15  nameLast      21010 non-null  object
 16  nameGiven     20973 non-null  object
 17  weight        20165 non-null  float64
 18  height        20245 non-null  float64
 19  bats          19792 non-null  object
 20  throws        19997 non-null  object
 21  debut         20724 non-null  object
 22  bbrefID       20956 non-null  object
 23  finalGame     20724 non-null  object
 24  retroID       20934 non-null  object
dtypes: float64(8), int64(1), object(16)
memory usage: 4.0+ MB
```

[9]:
```
top = pd.merge(top, people, on="playerID", how="left")
top = top[["playerID", "nameFirst", "nameLast", "HR"]]
top
```

[9]:

	playerID	nameFirst	nameLast	HR
0	trumbma01	Mark	Trumbo	47
1	cruzne02	Nelson	Cruz	43
2	encared01	Edwin	Encarnacion	42
3	daviskh01	Khris	Davis	42
4	doziebr01	Brian	Dozier	42
5	cartech02	Chris	Carter	41
6	arenano01	Nolan	Arenado	41
7	frazito01	Todd	Frazier	40
8	canoro01	Robinson	Cano	39

```
 9  bryankr01      Kris       Bryant  39
```

[10]:
```
top.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   playerID   10 non-null     object
 1   nameFirst  10 non-null     object
 2   nameLast   10 non-null     object
 3   HR         10 non-null     int64
dtypes: int64(1), object(3)
memory usage: 448.0+ bytes
```

Exercício 2 - Cap. 22.4 Now use the Salaries data frame to add each player's salary to the table you created in exercise 1. Note that salaries are different every year so make sure to filter for the year 2016, then use right_join. This time show first name, last name, team, HR, and salary.

Usando o objeto criado no exercício anterior, realizar um right_join com os dados de Salaries de cada jogadar. Atentar para utilizar o salário em 2016.

[11]:
```
salaries = pd.read_csv(base_path + "Salaries.csv")
salaries
```

[11]:

	yearID	teamID	lgID	playerID	salary
0	1985	ATL	NL	barkele01	870000
1	1985	ATL	NL	bedrost01	550000
2	1985	ATL	NL	benedbr01	545000
3	1985	ATL	NL	campri01	633333
4	1985	ATL	NL	ceronri01	625000
…	…	…	…	…	…
26423	2016	WAS	NL	strasst01	10400000
26424	2016	WAS	NL	taylomi02	524000
26425	2016	WAS	NL	treinbl01	524900
26426	2016	WAS	NL	werthja01	21733615
26427	2016	WAS	NL	zimmery01	14000000

[26428 rows x 5 columns]

[12]:
```
salaries.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26428 entries, 0 to 26427
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype
```

```
---  ------    --------------  -----
 0   yearID    26428 non-null  int64
 1   teamID    26428 non-null  object
 2   lgID      26428 non-null  object
 3   playerID  26428 non-null  object
 4   salary    26428 non-null  int64
dtypes: int64(2), object(3)
memory usage: 1.0+ MB
```

[13]:
```
salaries = salaries[salaries["yearID"] == 2016]
salaries
```

[13]:

	yearID	teamID	lgID	playerID	salary
25575	2016	ARI	NL	ahmedni01	521600
25576	2016	ARI	NL	barreja01	507500
25577	2016	ARI	NL	brachsi01	509300
25578	2016	ARI	NL	britoso01	508500
25579	2016	ARI	NL	castiwe01	3700000
…	…	…	…	…	…
26423	2016	WAS	NL	strasst01	10400000
26424	2016	WAS	NL	taylomi02	524000
26425	2016	WAS	NL	treinbl01	524900
26426	2016	WAS	NL	werthja01	21733615
26427	2016	WAS	NL	zimmery01	14000000

[853 rows x 5 columns]

[14]:
```
salaries = pd.merge(salaries, top, on="playerID", how="right")
salaries
```

[14]:

	yearID	teamID	lgID	playerID	salary	nameFirst	nameLast	HR
0	2016	BAL	AL	trumbma01	9150000	Mark	Trumbo	47
1	2016	SEA	AL	cruzne02	14250000	Nelson	Cruz	43
2	2016	TOR	AL	encared01	10000000	Edwin	Encarnacion	42
3	2016	OAK	AL	daviskh01	524500	Khris	Davis	42
4	2016	MIN	AL	doziebr01	3000000	Brian	Dozier	42
5	2016	MIL	NL	cartech02	2500000	Chris	Carter	41
6	2016	COL	NL	arenano01	5000000	Nolan	Arenado	41
7	2016	CHA	AL	frazito01	8250000	Todd	Frazier	40
8	2016	SEA	AL	canoro01	24000000	Robinson	Cano	39
9	2016	CHN	NL	bryankr01	652000	Kris	Bryant	39

[15]:
```
salaries = salaries[["nameFirst", "nameLast", "teamID", "HR", "salary"]]
salaries
```

[15]:

	nameFirst	nameLast	teamID	HR	salary
0	Mark	Trumbo	BAL	47	9150000
1	Nelson	Cruz	SEA	43	14250000

```
2     Edwin  Encarnacion    TOR  42  10000000
3     Khris        Davis    OAK  42    524500
4     Brian       Dozier    MIN  42   3000000
5     Chris       Carter    MIL  41   2500000
6     Nolan      Arenado    COL  41   5000000
7      Todd      Frazier    CHA  40   8250000
8  Robinson         Cano    SEA  39  24000000
9      Kris       Bryant    CHN  39    652000
```

Exercício 3 - Cap. 22.4 In a previous exercise, we created a tidy version of the co2 dataset:

```
co2_wide <- data.frame(matrix(co2, ncol = 12, byrow = TRUE)) |>
  setNames(1:12) |>
  mutate(year = 1959:1997) |>
  pivot_longer(-year, names_to = "month", values_to = "co2") |>
  mutate(month = as.numeric(month))
```

We want to see if the monthly trend is changing so we are going to remove the year effects and then plot the results. We will first compute the year averages. Use the group_by and summarize to compute the average co2 for each year. Save in an object called yearly_avg.

[18]:
```
import numpy as np
co2 = pd.read_csv('data/co2.csv')

data = co2['value'].tolist()
data = np.asarray(data)
co2_wide = pd.DataFrame(
    data.reshape(39,12),
    index = range(1959,1998),
    columns = ['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12']
)
co2_wide
```

[18]:

	1	2	3	4	5	6	7	8	9
1959	315.42	316.31	316.50	317.56	318.13	318.00	316.39	314.65	313.68
1960	316.27	316.81	317.42	318.87	319.87	319.43	318.01	315.74	314.00
1961	316.73	317.54	318.38	319.31	320.42	319.61	318.42	316.63	314.83
1962	317.78	318.40	319.53	320.42	320.85	320.45	319.45	317.25	316.11
1963	318.58	318.92	319.70	321.22	322.08	321.31	319.58	317.61	316.05
1964	319.41	320.07	320.74	321.40	322.06	321.73	320.27	318.54	316.54
1965	319.27	320.28	320.73	321.97	322.00	321.71	321.05	318.71	317.66
1966	320.46	321.43	322.23	323.54	323.91	323.59	322.24	320.20	318.48
1967	322.17	322.34	322.88	324.25	324.83	323.93	322.38	320.76	319.10
1968	322.40	322.99	323.73	324.86	325.40	325.20	323.98	321.95	320.18
1969	323.83	324.26	325.47	326.50	327.21	326.54	325.72	323.50	322.22
1970	324.89	325.82	326.77	327.97	327.91	327.50	326.18	324.53	322.93
1971	326.01	326.51	327.01	327.62	328.76	328.40	327.20	325.27	323.20

1972	326.60	327.47	327.58	329.56	329.90	328.92	327.88	326.16	324.68
1973	328.37	329.40	330.14	331.33	332.31	331.90	330.70	329.15	327.35
1974	329.18	330.55	331.32	332.48	332.92	332.08	331.01	329.23	327.27
1975	330.23	331.25	331.87	333.14	333.80	333.43	331.73	329.90	328.40
1976	331.58	332.39	333.33	334.41	334.71	334.17	332.89	330.77	329.14
1977	332.75	333.24	334.53	335.90	336.57	336.10	334.76	332.59	331.42
1978	334.80	335.22	336.47	337.59	337.84	337.72	336.37	334.51	332.60
1979	336.05	336.59	337.79	338.71	339.30	339.12	337.56	335.92	333.75
1980	337.84	338.19	339.91	340.60	341.29	341.00	339.39	337.43	335.72
1981	339.06	340.30	341.21	342.33	342.74	342.08	340.32	338.26	336.52
1982	340.57	341.44	342.53	343.39	343.96	343.18	341.88	339.65	337.81
1983	341.20	342.35	342.93	344.77	345.58	345.14	343.81	342.21	339.69
1984	343.52	344.33	345.11	346.88	347.25	346.62	345.22	343.11	340.90
1985	344.79	345.82	347.25	348.17	348.74	348.07	346.38	344.51	342.92
1986	346.11	346.78	347.68	349.37	350.03	349.37	347.76	345.73	344.68
1987	347.84	348.29	349.23	350.80	351.66	351.07	349.33	347.92	346.27
1988	350.25	351.54	352.05	353.41	354.04	353.62	352.22	350.27	348.55
1989	352.60	352.92	353.53	355.26	355.52	354.97	353.75	351.52	349.64
1990	353.50	354.55	355.23	356.04	357.00	356.07	354.67	352.76	350.82
1991	354.59	355.63	357.03	358.48	359.22	358.12	356.06	353.92	352.05
1992	355.88	356.63	357.72	359.07	359.58	359.17	356.94	354.92	352.94
1993	356.63	357.10	358.32	359.41	360.23	359.55	357.53	355.48	353.67
1994	358.34	358.89	359.95	361.25	361.67	360.94	359.55	357.49	355.84
1995	359.98	361.03	361.66	363.48	363.82	363.30	361.94	359.50	358.11
1996	362.09	363.29	364.06	364.76	365.45	365.01	363.70	361.54	359.51
1997	363.23	364.06	364.61	366.40	366.84	365.68	364.52	362.57	360.24

	10	11	12
1959	313.18	314.66	315.43
1960	313.68	314.84	316.03
1961	315.16	315.94	316.85
1962	315.27	316.53	317.53
1963	315.83	316.91	318.20
1964	316.71	317.53	318.55
1965	317.14	318.70	319.25
1966	317.94	319.63	320.87
1967	319.24	320.56	321.80
1968	320.09	321.16	322.74
1969	321.62	322.69	323.95
1970	322.90	323.85	324.96
1971	323.40	324.63	325.85
1972	325.04	326.34	327.39
1973	327.02	327.99	328.48
1974	327.21	328.29	329.41
1975	328.17	329.32	330.59
1976	328.78	330.14	331.52
1977	330.98	332.24	333.68

```
1978  332.38  333.75  334.78
1979  333.70  335.12  336.56
1980  335.84  336.93  338.04
1981  336.68  338.19  339.44
1982  337.69  339.09  340.32
1983  339.82  340.98  342.82
1984  341.18  342.80  344.04
1985  342.62  344.06  345.38
1986  343.99  345.48  346.72
1987  346.18  347.64  348.78
1988  348.72  349.91  351.18
1989  349.83  351.14  352.37
1990  351.04  352.69  354.07
1991  352.11  353.64  354.89
1992  353.23  354.09  355.33
1993  353.95  355.30  356.78
1994  356.00  357.59  359.05
1995  357.80  359.61  360.74
1996  359.65  360.80  362.38
1997  360.83  362.49  364.34
```

[78]:
```
yearly_avg = pd.DataFrame((round(co2_wide.mean(axis=1), 2)), columns = ['Avg'])
yearly_avg
```

[78]:
```
          Avg
1959  315.83
1960  316.75
1961  317.48
1962  318.30
1963  318.83
1964  319.46
1965  319.87
1966  321.21
1967  322.02
1968  322.89
1969  324.46
1970  325.52
1971  326.16
1972  327.29
1973  329.51
1974  330.08
1975  330.99
1976  331.99
1977  333.73
1978  335.34
1979  336.68
1980  338.52
```

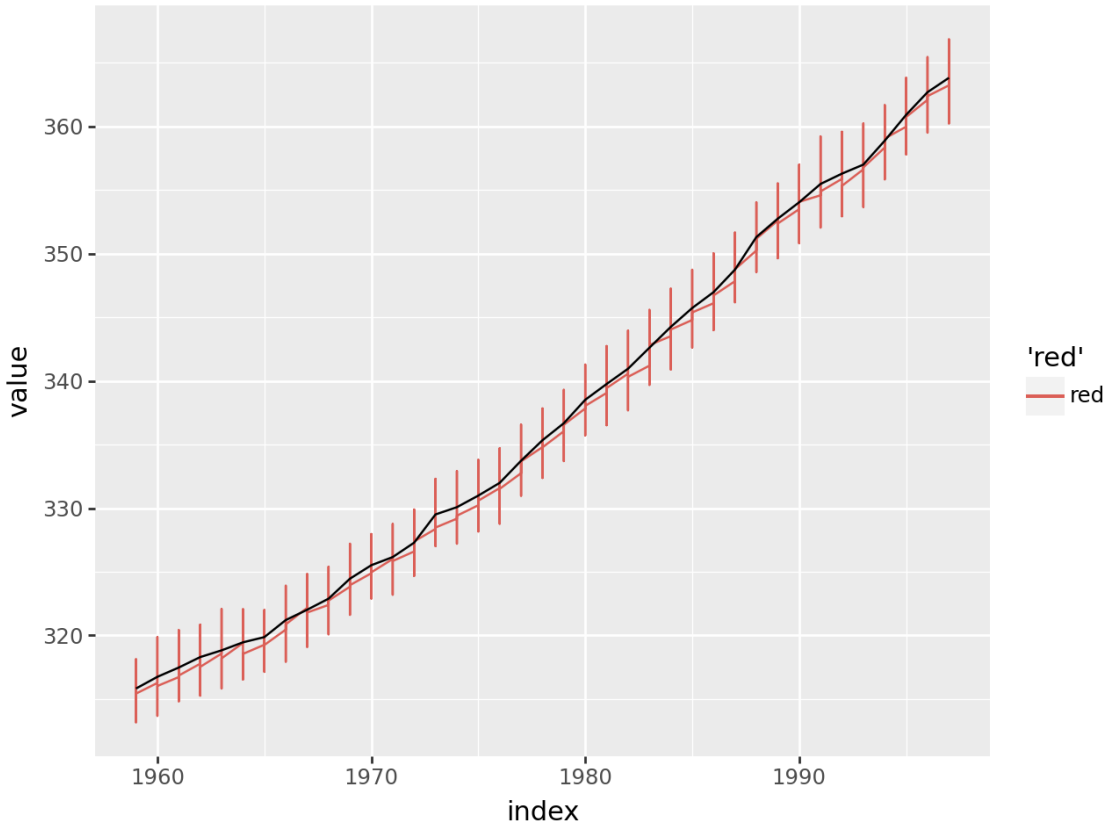
```
1981  339.76
1982  340.96
1983  342.61
1984  344.25
1985  345.73
1986  346.98
1987  348.75
1988  351.31
1989  352.75
1990  354.04
1991  355.48
1992  356.29
1993  357.00
1994  358.88
1995  360.91
1996  362.69
1997  363.82
```

[65]:
```
co2_melted = pd.melt(co2_wide.reset_index(), id_vars='index',␣
 ↪value_vars=[str(i) for i in range(1,13)] )
co2_melted
```

[65]:
```
     index variable   value
0     1959        1  315.42
1     1960        1  316.27
2     1961        1  316.73
3     1962        1  317.78
4     1963        1  318.58
..     …       …     …
463   1993       12  356.78
464   1994       12  359.05
465   1995       12  360.74
466   1996       12  362.38
467   1997       12  364.34

[468 rows x 3 columns]
```

[99]:
```
(
    ggplot()
    + geom_line(data=co2_melted, mapping=aes(x="index", y="value",␣
 ↪color="'red'"))
    + geom_line(data=yearly_avg, mapping=aes(x=range(1959, 1998), y="Avg"))
)
```



O gráfico acima foi plotado tentando representar a variação de valores das medições mensais (linha vermelha) e a média anual representada pela linha na cor preta.

A inspiração para este gráfico foi obtida através de pesquisas para entender sobre o conjunto de dados e onde encontrei a página a seguir: https://gml.noaa.gov/ccgg/trends/

Exercício 4 - Cap. 22.4 Now use the left_join function to add the yearly average to the co2_wide dataset. Then compute the residuals: observed co2 measure - yearly average.

[21]:
```
co2_wide = co2_wide.join(yearly_avg, how = 'left')
co2_wide
```

[21]:

	1	2	3	4	5	6	7	8	9	\
1959	315.42	316.31	316.50	317.56	318.13	318.00	316.39	314.65	313.68	
1960	316.27	316.81	317.42	318.87	319.87	319.43	318.01	315.74	314.00	
1961	316.73	317.54	318.38	319.31	320.42	319.61	318.42	316.63	314.83	
1962	317.78	318.40	319.53	320.42	320.85	320.45	319.45	317.25	316.11	
1963	318.58	318.92	319.70	321.22	322.08	321.31	319.58	317.61	316.05	
1964	319.41	320.07	320.74	321.40	322.06	321.73	320.27	318.54	316.54	
1965	319.27	320.28	320.73	321.97	322.00	321.71	321.05	318.71	317.66	

1966	320.46	321.43	322.23	323.54	323.91	323.59	322.24	320.20	318.48
1967	322.17	322.34	322.88	324.25	324.83	323.93	322.38	320.76	319.10
1968	322.40	322.99	323.73	324.86	325.40	325.20	323.98	321.95	320.18
1969	323.83	324.26	325.47	326.50	327.21	326.54	325.72	323.50	322.22
1970	324.89	325.82	326.77	327.97	327.91	327.50	326.18	324.53	322.93
1971	326.01	326.51	327.01	327.62	328.76	328.40	327.20	325.27	323.20
1972	326.60	327.47	327.58	329.56	329.90	328.92	327.88	326.16	324.68
1973	328.37	329.40	330.14	331.33	332.31	331.90	330.70	329.15	327.35
1974	329.18	330.55	331.32	332.48	332.92	332.08	331.01	329.23	327.27
1975	330.23	331.25	331.87	333.14	333.80	333.43	331.73	329.90	328.40
1976	331.58	332.39	333.33	334.41	334.71	334.17	332.89	330.77	329.14
1977	332.75	333.24	334.53	335.90	336.57	336.10	334.76	332.59	331.42
1978	334.80	335.22	336.47	337.59	337.84	337.72	336.37	334.51	332.60
1979	336.05	336.59	337.79	338.71	339.30	339.12	337.56	335.92	333.75
1980	337.84	338.19	339.91	340.60	341.29	341.00	339.39	337.43	335.72
1981	339.06	340.30	341.21	342.33	342.74	342.08	340.32	338.26	336.52
1982	340.57	341.44	342.53	343.39	343.96	343.18	341.88	339.65	337.81
1983	341.20	342.35	342.93	344.77	345.58	345.14	343.81	342.21	339.69
1984	343.52	344.33	345.11	346.88	347.25	346.62	345.22	343.11	340.90
1985	344.79	345.82	347.25	348.17	348.74	348.07	346.38	344.51	342.92
1986	346.11	346.78	347.68	349.37	350.03	349.37	347.76	345.73	344.68
1987	347.84	348.29	349.23	350.80	351.66	351.07	349.33	347.92	346.27
1988	350.25	351.54	352.05	353.41	354.04	353.62	352.22	350.27	348.55
1989	352.60	352.92	353.53	355.26	355.52	354.97	353.75	351.52	349.64
1990	353.50	354.55	355.23	356.04	357.00	356.07	354.67	352.76	350.82
1991	354.59	355.63	357.03	358.48	359.22	358.12	356.06	353.92	352.05
1992	355.88	356.63	357.72	359.07	359.58	359.17	356.94	354.92	352.94
1993	356.63	357.10	358.32	359.41	360.23	359.55	357.53	355.48	353.67
1994	358.34	358.89	359.95	361.25	361.67	360.94	359.55	357.49	355.84
1995	359.98	361.03	361.66	363.48	363.82	363.30	361.94	359.50	358.11
1996	362.09	363.29	364.06	364.76	365.45	365.01	363.70	361.54	359.51
1997	363.23	364.06	364.61	366.40	366.84	365.68	364.52	362.57	360.24

	10	11	12	Avg
1959	313.18	314.66	315.43	315.83
1960	313.68	314.84	316.03	316.75
1961	315.16	315.94	316.85	317.48
1962	315.27	316.53	317.53	318.30
1963	315.83	316.91	318.20	318.83
1964	316.71	317.53	318.55	319.46
1965	317.14	318.70	319.25	319.87
1966	317.94	319.63	320.87	321.21
1967	319.24	320.56	321.80	322.02
1968	320.09	321.16	322.74	322.89
1969	321.62	322.69	323.95	324.46
1970	322.90	323.85	324.96	325.52
1971	323.40	324.63	325.85	326.16

```
1972  325.04  326.34  327.39  327.29
1973  327.02  327.99  328.48  329.51
1974  327.21  328.29  329.41  330.08
1975  328.17  329.32  330.59  330.99
1976  328.78  330.14  331.52  331.99
1977  330.98  332.24  333.68  333.73
1978  332.38  333.75  334.78  335.34
1979  333.70  335.12  336.56  336.68
1980  335.84  336.93  338.04  338.52
1981  336.68  338.19  339.44  339.76
1982  337.69  339.09  340.32  340.96
1983  339.82  340.98  342.82  342.61
1984  341.18  342.80  344.04  344.25
1985  342.62  344.06  345.38  345.73
1986  343.99  345.48  346.72  346.98
1987  346.18  347.64  348.78  348.75
1988  348.72  349.91  351.18  351.31
1989  349.83  351.14  352.37  352.75
1990  351.04  352.69  354.07  354.04
1991  352.11  353.64  354.89  355.48
1992  353.23  354.09  355.33  356.29
1993  353.95  355.30  356.78  357.00
1994  356.00  357.59  359.05  358.88
1995  357.80  359.61  360.74  360.91
1996  359.65  360.80  362.38  362.69
1997  360.83  362.49  364.34  363.82
```

[22]:
```
res_co2 = pd.DataFrame(index = range(1959,1998), columns = ['1', '2', '3', '4',␣
 ↪'5', '6', '7', '8', '9', '10', '11', '12'])

for i in range(1,13):
  res_co2[f'{i}'] = co2_wide[f'{i}'] - co2_wide['Avg']

res_co2
```

[22]:
```
          1     2     3     4     5     6     7     8     9    10    11    12
1959 -0.41  0.48  0.67  1.73  2.30  2.17  0.56 -1.18 -2.15 -2.65 -1.17 -0.40
1960 -0.48  0.06  0.67  2.12  3.12  2.68  1.26 -1.01 -2.75 -3.07 -1.91 -0.72
1961 -0.75  0.06  0.90  1.83  2.94  2.13  0.94 -0.85 -2.65 -2.32 -1.54 -0.63
1962 -0.52  0.10  1.23  2.12  2.55  2.15  1.15 -1.05 -2.19 -3.03 -1.77 -0.77
1963 -0.25  0.09  0.87  2.39  3.25  2.48  0.75 -1.22 -2.78 -3.00 -1.92 -0.63
1964 -0.05  0.61  1.28  1.94  2.60  2.27  0.81 -0.92 -2.92 -2.75 -1.93 -0.91
1965 -0.60  0.41  0.86  2.10  2.13  1.84  1.18 -1.16 -2.21 -2.73 -1.17 -0.62
1966 -0.75  0.22  1.02  2.33  2.70  2.38  1.03 -1.01 -2.73 -3.27 -1.58 -0.34
1967  0.15  0.32  0.86  2.23  2.81  1.91  0.36 -1.26 -2.92 -2.78 -1.46 -0.22
1968 -0.49  0.10  0.84  1.97  2.51  2.31  1.09 -0.94 -2.71 -2.80 -1.73 -0.15
1969 -0.63 -0.20  1.01  2.04  2.75  2.08  1.26 -0.96 -2.24 -2.84 -1.77 -0.51
```

```
1970 -0.63  0.30  1.25  2.45  2.39  1.98  0.66 -0.99 -2.59 -2.62 -1.67 -0.56
1971 -0.15  0.35  0.85  1.46  2.60  2.24  1.04 -0.89 -2.96 -2.76 -1.53 -0.31
1972 -0.69  0.18  0.29  2.27  2.61  1.63  0.59 -1.13 -2.61 -2.25 -0.95  0.10
1973 -1.14 -0.11  0.63  1.82  2.80  2.39  1.19 -0.36 -2.16 -2.49 -1.52 -1.03
1974 -0.90  0.47  1.24  2.40  2.84  2.00  0.93 -0.85 -2.81 -2.87 -1.79 -0.67
1975 -0.76  0.26  0.88  2.15  2.81  2.44  0.74 -1.09 -2.59 -2.82 -1.67 -0.40
1976 -0.41  0.40  1.34  2.42  2.72  2.18  0.90 -1.22 -2.85 -3.21 -1.85 -0.47
1977 -0.98 -0.49  0.80  2.17  2.84  2.37  1.03 -1.14 -2.31 -2.75 -1.49 -0.05
1978 -0.54 -0.12  1.13  2.25  2.50  2.38  1.03 -0.83 -2.74 -2.96 -1.59 -0.56
1979 -0.63 -0.09  1.11  2.03  2.62  2.44  0.88 -0.76 -2.93 -2.98 -1.56 -0.12
1980 -0.68 -0.33  1.39  2.08  2.77  2.48  0.87 -1.09 -2.80 -2.68 -1.59 -0.48
1981 -0.70  0.54  1.45  2.57  2.98  2.32  0.56 -1.50 -3.24 -3.08 -1.57 -0.32
1982 -0.39  0.48  1.57  2.43  3.00  2.22  0.92 -1.31 -3.15 -3.27 -1.87 -0.64
1983 -1.41 -0.26  0.32  2.16  2.97  2.53  1.20 -0.40 -2.92 -2.79 -1.63  0.21
1984 -0.73  0.08  0.86  2.63  3.00  2.37  0.97 -1.14 -3.35 -3.07 -1.45 -0.21
1985 -0.94  0.09  1.52  2.44  3.01  2.34  0.65 -1.22 -2.81 -3.11 -1.67 -0.35
1986 -0.87 -0.20  0.70  2.39  3.05  2.39  0.78 -1.25 -2.30 -2.99 -1.50 -0.26
1987 -0.91 -0.46  0.48  2.05  2.91  2.32  0.58 -0.83 -2.48 -2.57 -1.11  0.03
1988 -1.06  0.23  0.74  2.10  2.73  2.31  0.91 -1.04 -2.76 -2.59 -1.40 -0.13
1989 -0.15  0.17  0.78  2.51  2.77  2.22  1.00 -1.23 -3.11 -2.92 -1.61 -0.38
1990 -0.54  0.51  1.19  2.00  2.96  2.03  0.63 -1.28 -3.22 -3.00 -1.35  0.03
1991 -0.89  0.15  1.55  3.00  3.74  2.64  0.58 -1.56 -3.43 -3.37 -1.84 -0.59
1992 -0.41  0.34  1.43  2.78  3.29  2.88  0.65 -1.37 -3.35 -3.06 -2.20 -0.96
1993 -0.37  0.10  1.32  2.41  3.23  2.55  0.53 -1.52 -3.33 -3.05 -1.70 -0.22
1994 -0.54  0.01  1.07  2.37  2.79  2.06  0.67 -1.39 -3.04 -2.88 -1.29  0.17
1995 -0.93  0.12  0.75  2.57  2.91  2.39  1.03 -1.41 -2.80 -3.11 -1.30 -0.17
1996 -0.60  0.60  1.37  2.07  2.76  2.32  1.01 -1.15 -3.18 -3.04 -1.89 -0.31
1997 -0.59  0.24  0.79  2.58  3.02  1.86  0.70 -1.25 -3.58 -2.99 -1.33  0.52
```

Exercício 5 - Cap. 22.4 Make a plot of the seasonal trends by year but only after removing the year effect.

[133]:
```
res_co2_melted = pd.melt(res_co2.reset_index(), id_vars='index',␣
↪value_vars=[str(i) for i in range(1,13)] )
res_co2_melted = res_co2_melted.rename(columns={"index": "Year", "variable":␣
↪"Month"})
res_co2_melted['Month'] = res_co2_melted['Month'].str.zfill(2)
res_co2_melted
```

[133]:

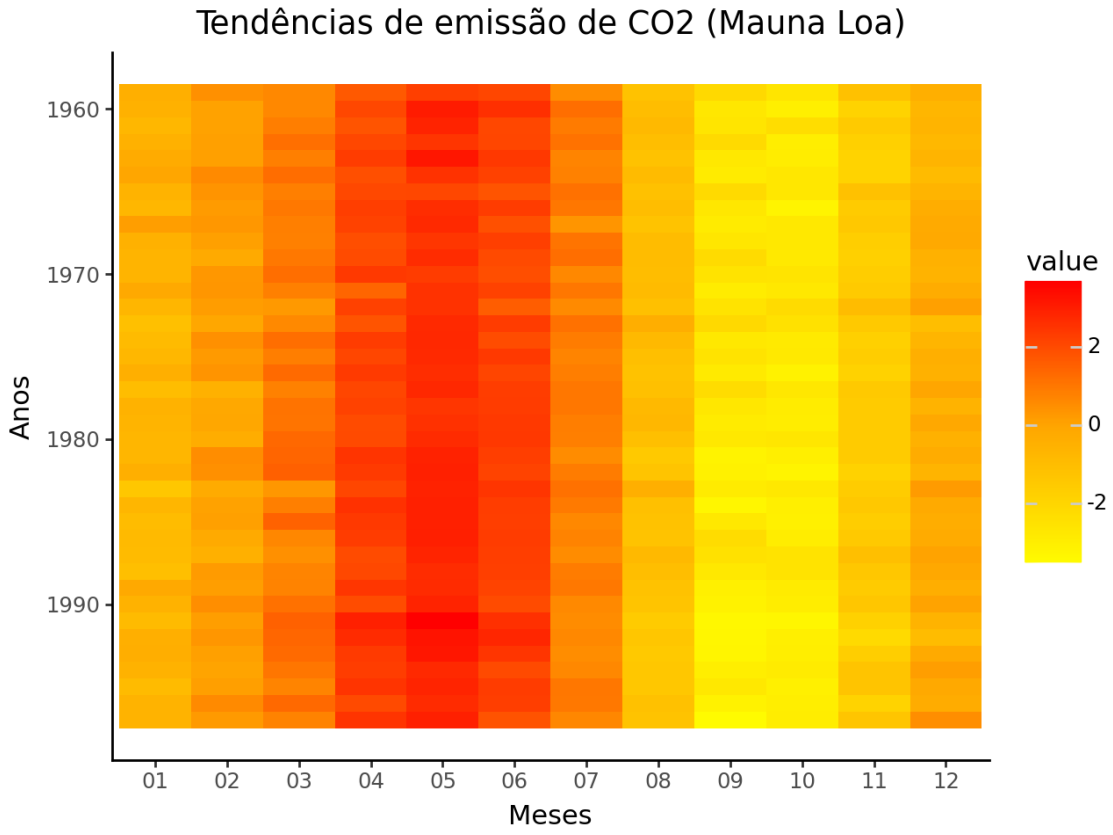
	Year	Month	value
0	1959	01	-0.41
1	1960	01	-0.48
2	1961	01	-0.75
3	1962	01	-0.52
4	1963	01	-0.25

```
..    …    …    …
463  1993    12  -0.22
464  1994    12   0.17
465  1995    12  -0.17
466  1996    12  -0.31
467  1997    12   0.52

[468 rows x 3 columns]
```

[167]:
```
from plotnine import *

(
    ggplot(data=res_co2_melted, mapping=aes(x="factor(Month)", y="Year",␣
 ↪fill="value"))
    + geom_tile()
    + scale_fill_gradient2(low="yellow", mid="orange", high="red")
    + scale_y_reverse()
    + theme_classic()
    + labs(title = "Tendências de emissão de CO2 (Mauna Loa)", x="Meses",␣
 ↪y="Anos")
)
```



Exercício: 13. Advanced: extract the titles of the movies that won Best Picture from this website: https://m.imdb.com/chart/bestpicture/

Minha abordagem: A página apresenta somente 50 títulos por vez. E para exibição dos próximos itens é necessário clicar sobre um botão que realiza uma chamada assincrona obtendo mais resultados. Seria possível implementar um web scraping utilizando Selenium porém não conseguiria demonstrar aqui por notebook.

Percebi que o site possibilitava reordenar os registros por ano gerando uma nova URL, contendo parâmetros para a ordenação, como demonstrado abaixo:

URL após a ordenação por ano do mais antigo para o mais recente: https://m.imdb.com/search/title/?groups=best_picture_winner&sort=year,asc

URL após a ordenação por ano do mais recente para o mais antigo: https://m.imdb.com/search/title/?groups=best_picture_winner&sort=year,desc

[1]:
```
import requests
import re
from bs4 import BeautifulSoup

#Link original: "https://m.imdb.com/chart/bestpicture/"

links = [
    "https://m.imdb.com/search/title/?groups=best_picture_winner&sort=year,asc",
    "https://m.imdb.com/search/title/?groups=best_picture_winner&sort=year,desc"
]


"""
Função para carregamento do conteúdo de uma URL informada

:param site_url: Link (URL) da página a ser carregada
:return: retorna um objeto contendo os dados da resposta da requisição
"""
def load_site(site_url):
    headers = {
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/
 ↪537.36 (KHTML, like Gecko) Chrome/109.0.0.0 Safari/537.36',
    }

    response = requests.get(link, headers=headers)

    print(f'URL requisitada: {link}')

    if response.status_code == 200 :
```

```
        print(f"Sucesso: código de retorno {response.status_code}")
        return response
    else :
        print(f'Falha na requisição - código do erro: {response.status_code}')


pages = []

for link in links:
    response = load_site(link)
    pages.append(BeautifulSoup(response.text, 'html.parser'))
```

```
URL requisitada:
https://m.imdb.com/search/title/?groups=best_picture_winner&sort=year,asc
Sucesso: código de retorno 200
URL requisitada:
https://m.imdb.com/search/title/?groups=best_picture_winner&sort=year,desc
Sucesso: código de retorno 200
```

Abaixo obtenho os elementos HTML que contém a lista de títulos (de ambas URLS). Estes objetos são armazenados em uma lista chamada **titles**

[2]:
```
movies_titles = []
for page in pages:
    movies_titles.append(page.find_all("li",␣
 ↪class_="ipc-metadata-list-summary-item"))
```

Agora para cada uma das listas de títulos de filmes extraio o conteúdo de texto contento do nome do título do filme. Estas duas listas serão a base para criação de DataFrames, onde farei o merge e a eliminação de possíveis elementos duplicados.

[ ]:
```
"""
"""
def get_movie_titles(movie_list):
    titles = []
    for item in movie_list:
        titles.append({
            'title' : re.sub("\d+. ", "", item.find("h3",␣
 ↪class_="ipc-title__text").getText()),
            'release'  : item.find_all("span",␣
 ↪class_="dli-title-metadata-item")[0].getText(),
            'runtime'  : item.find_all("span",␣
 ↪class_="dli-title-metadata-item")[1].getText()
        })
    return titles

#Dicionário contendo a lista de títulos final e tratada
titles = ()
```

Extraindo os títulos da lista ordenada de forma decrescente (título mais recente para o mais antigo) [Qtde. de títulos: 50]

[ ]:
```
titles_list_1 = get_movie_titles(movies_titles[1])
titles_list_1
```

[ ]:
```
[{'title': 'Oppenheimer', 'release': '2023', 'runtime': '3h'},
 {'title': 'Tudo em Todo o Lugar ao Mesmo Tempo',
  'release': '2022',
  'runtime': '2h 19m'},
 {'title': 'No Ritmo do Coração', 'release': '2021', 'runtime': '1h 51m'},
 {'title': 'Nomadland', 'release': '2020', 'runtime': '1h 47m'},
 {'title': 'Parasita', 'release': '2019', 'runtime': '2h 12m'},
 {'title': 'Green Book: O Guia', 'release': '2018', 'runtime': '2h 10m'},
 {'title': 'A Forma da Água', 'release': '2017', 'runtime': '2h 3m'},
 {'title': 'Moonlight: Sob a Luz do Luar',
  'release': '2016',
  'runtime': '1h 51m'},
 {'title': 'Spotlight: Segredos Revelados',
  'release': '2015',
  'runtime': '2h 9m'},
 {'title': 'Birdman ou (A Inesperada Virtude da Ignorância)',
  'release': '2014',
  'runtime': '1h 59m'},
 {'title': 'Anos de Escravidão', 'release': '2013', 'runtime': '2h 14m'},
 {'title': 'Argo', 'release': '2012', 'runtime': '2h'},
 {'title': 'O Artista', 'release': '2011', 'runtime': '1h 40m'},
 {'title': 'O Discurso do Rei', 'release': '2010', 'runtime': '1h 58m'},
 {'title': 'Quem Quer Ser um Milionário?', 'release': '2008', 'runtime': '2h'},
 {'title': 'Guerra ao Terror', 'release': '2008', 'runtime': '2h 11m'},
 {'title': 'Onde os Fracos Não Têm Vez',
  'release': '2007',
  'runtime': '2h 2m'},
 {'title': 'Os Infiltrados', 'release': '2006', 'runtime': '2h 31m'},
 {'title': 'Crash: No Limite', 'release': '2004', 'runtime': '1h 52m'},
 {'title': 'Menina de Ouro', 'release': '2004', 'runtime': '2h 12m'},
 {'title': 'O Senhor dos Anéis: O Retorno do Rei',
  'release': '2003',
  'runtime': '3h 21m'},
 {'title': 'Chicago', 'release': '2002', 'runtime': '1h 53m'},
 {'title': 'Uma Mente Brilhante', 'release': '2001', 'runtime': '2h 15m'},
 {'title': 'Gladiador', 'release': '2000', 'runtime': '2h 35m'},
 {'title': 'Beleza Americana', 'release': '1999', 'runtime': '2h 2m'},
 {'title': 'Shakespeare Apaixonado', 'release': '1998', 'runtime': '2h 3m'},
 {'title': 'Titanic', 'release': '1997', 'runtime': '3h 14m'},
 {'title': 'O Paciente Inglês', 'release': '1996', 'runtime': '2h 42m'},
```

```
{'title': 'Coração Valente', 'release': '1995', 'runtime': '2h 58m'},
{'title': 'Forrest Gump: O Contador de Histórias',
 'release': '1994',
 'runtime': '2h 22m'},
{'title': 'A Lista de Schindler', 'release': '1993', 'runtime': '3h 15m'},
{'title': 'Os Imperdoáveis', 'release': '1992', 'runtime': '2h 10m'},
{'title': 'O Silêncio dos Inocentes', 'release': '1991', 'runtime': '1h 58m'},
{'title': 'Dança com Lobos', 'release': '1990', 'runtime': '3h 1m'},
{'title': 'Conduzindo Miss Daisy', 'release': '1989', 'runtime': '1h 39m'},
{'title': 'Rain Man', 'release': '1988', 'runtime': '2h 13m'},
{'title': 'O Último Imperador', 'release': '1987', 'runtime': '2h 43m'},
{'title': 'Platoon', 'release': '1986', 'runtime': '2h'},
{'title': 'Entre Dois Amores', 'release': '1985', 'runtime': '2h 41m'},
{'title': 'Amadeus', 'release': '1984', 'runtime': '2h 40m'},
{'title': 'Laços de Ternura', 'release': '1983', 'runtime': '2h 12m'},
{'title': 'Gandhi', 'release': '1982', 'runtime': '3h 11m'},
{'title': 'Carruagens de Fogo', 'release': '1981', 'runtime': '2h 5m'},
{'title': 'Gente como a Gente', 'release': '1980', 'runtime': '2h 4m'},
{'title': 'Kramer vs. Kramer', 'release': '1979', 'runtime': '1h 45m'},
{'title': 'O Franco Atirador', 'release': '1978', 'runtime': '3h 3m'},
{'title': 'Noivo Neurótico, Noiva Nervosa',
 'release': '1977',
 'runtime': '1h 33m'},
{'title': 'Rocky, um Lutador', 'release': '1976', 'runtime': '2h'},
{'title': 'Um Estranho no Ninho', 'release': '1975', 'runtime': '2h 13m'},
{'title': 'O Poderoso Chefão II', 'release': '1974', 'runtime': '3h 22m'}]
```

Extraindo os títulos da lista ordenada de forma crescente (título mais antigo para o mais recente) [Qtde. de títulos: 50] Observações: * A lista irá sobrepor alguns título da lista anterior, para remover vou utilizar um set (conjunto) * É necessário reverter a ordem da lista para dicar igual à lista anterior (do mais recente para o mais antigo)

[ ]:
```
titles_list_2 = get_movie_titles(movies_titles[0])
titles_list_2.reverse()
titles_list_2
```

[ ]:
```
[{'title': 'Rocky, um Lutador', 'release': '1976', 'runtime': '2h'},
 {'title': 'Um Estranho no Ninho', 'release': '1975', 'runtime': '2h 13m'},
 {'title': 'O Poderoso Chefão II', 'release': '1974', 'runtime': '3h 22m'},
 {'title': 'Golpe de Mestre', 'release': '1973', 'runtime': '2h 9m'},
 {'title': 'O Poderoso Chefão', 'release': '1972', 'runtime': '2h 55m'},
 {'title': 'Operação França', 'release': '1971', 'runtime': '1h 44m'},
 {'title': 'Patton, Rebelde ou Herói?',
  'release': '1970',
  'runtime': '2h 52m'},
 {'title': 'Perdidos na Noite', 'release': '1969', 'runtime': '1h 53m'},
 {'title': 'Oliver!', 'release': '1968', 'runtime': '2h 33m'},
 {'title': 'No Calor da Noite', 'release': '1967', 'runtime': '1h 50m'},
```

```
{'title': 'O Homem que Não Vendeu sua Alma',
 'release': '1966',
 'runtime': '2h'},
{'title': 'A Noviça Rebelde', 'release': '1965', 'runtime': '2h 52m'},
{'title': 'Minha Bela Dama', 'release': '1964', 'runtime': '2h 50m'},
{'title': 'As Aventuras de Tom Jones', 'release': '1963', 'runtime': '2h 9m'},
{'title': 'Lawrence da Arábia', 'release': '1962', 'runtime': '3h 38m'},
{'title': 'Amor, Sublime Amor', 'release': '1961', 'runtime': '2h 33m'},
{'title': 'Se Meu Apartamento Falasse',
 'release': '1960',
 'runtime': '2h 5m'},
{'title': 'Ben-Hur', 'release': '1959', 'runtime': '3h 32m'},
{'title': 'Gigi', 'release': '1958', 'runtime': '1h 55m'},
{'title': 'A Ponte do Rio Kwai', 'release': '1957', 'runtime': '2h 41m'},
{'title': 'A Volta ao Mundo em Dias', 'release': '1956', 'runtime': '2h 47m'},
{'title': 'Marty', 'release': '1955', 'runtime': '1h 30m'},
{'title': 'Sindicato de Ladrões', 'release': '1954', 'runtime': '1h 48m'},
{'title': 'A um Passo da Eternidade', 'release': '1953', 'runtime': '1h 58m'},
{'title': 'O Maior Espetáculo da Terra',
 'release': '1952',
 'runtime': '2h 32m'},
{'title': 'Sinfonia de Paris', 'release': '1951', 'runtime': '1h 54m'},
{'title': 'A Malvada', 'release': '1950', 'runtime': '2h 18m'},
{'title': 'A Grande Ilusão', 'release': '1949', 'runtime': '1h 50m'},
{'title': 'Hamlet', 'release': '1948', 'runtime': '2h 34m'},
{'title': 'A Luz é para Todos', 'release': '1947', 'runtime': '1h 58m'},
{'title': 'Os Melhores Anos de Nossa Vida',
 'release': '1946',
 'runtime': '2h 50m'},
{'title': 'Farrapo Humano', 'release': '1945', 'runtime': '1h 41m'},
{'title': 'O Bom Pastor', 'release': '1944', 'runtime': '2h 6m'},
{'title': 'Rosa de Esperança', 'release': '1942', 'runtime': '2h 14m'},
{'title': 'Casablanca', 'release': '1942', 'runtime': '1h 42m'},
{'title': 'Como Era Verde o Meu Vale',
 'release': '1941',
 'runtime': '1h 58m'},
{'title': 'Rebecca, a Mulher Inesquecível',
 'release': '1940',
 'runtime': '2h 10m'},
{'title': '…E o Vento Levou', 'release': '1939', 'runtime': '3h 58m'},
{'title': 'Do Mundo Nada se Leva', 'release': '1938', 'runtime': '2h 6m'},
{'title': 'A Vida de Emile Zola', 'release': '1937', 'runtime': '1h 56m'},
{'title': 'Ziegfeld, o Criador de Estrelas',
 'release': '1936',
 'runtime': '2h 56m'},
{'title': 'O Grande Motim', 'release': '1935', 'runtime': '2h 12m'},
{'title': 'Aconteceu Naquela Noite', 'release': '1934', 'runtime': '1h 45m'},
```

```
{'title': 'Cavalgada', 'release': '1933', 'runtime': '1h 52m'},
{'title': 'Grande Hotel', 'release': '1932', 'runtime': '1h 52m'},
{'title': 'Cimarron', 'release': '1931', 'runtime': '2h 3m'},
{'title': 'Sem Novidade no Front', 'release': '1930', 'runtime': '2h 32m'},
{'title': 'Melodia da Broadway', 'release': '1929', 'runtime': '1h 40m'},
{'title': 'Asas', 'release': '1927', 'runtime': '2h 24m'},
{'title': 'Aurora', 'release': '1927', 'runtime': '1h 34m'}]
```

Junção das listas em um set (conjunto)

[ ]:
```
df1 = pd.DataFrame(titles_list_1)
df1
```

[ ]:

	title	release	runtime
0	Oppenheimer	2023	3h
1	Tudo em Todo o Lugar ao Mesmo Tempo	2022	2h 19m
2	No Ritmo do Coração	2021	1h 51m
3	Nomadland	2020	1h 47m
4	Parasita	2019	2h 12m
5	Green Book: O Guia	2018	2h 10m
6	A Forma da Água	2017	2h 3m
7	Moonlight: Sob a Luz do Luar	2016	1h 51m
8	Spotlight: Segredos Revelados	2015	2h 9m
9	Birdman ou (A Inesperada Virtude da Ignorância)	2014	1h 59m
10	Anos de Escravidão	2013	2h 14m
11	Argo	2012	2h
12	O Artista	2011	1h 40m
13	O Discurso do Rei	2010	1h 58m
14	Quem Quer Ser um Milionário?	2008	2h
15	Guerra ao Terror	2008	2h 11m
16	Onde os Fracos Não Têm Vez	2007	2h 2m
17	Os Infiltrados	2006	2h 31m
18	Crash: No Limite	2004	1h 52m
19	Menina de Ouro	2004	2h 12m
20	O Senhor dos Anéis: O Retorno do Rei	2003	3h 21m
21	Chicago	2002	1h 53m
22	Uma Mente Brilhante	2001	2h 15m
23	Gladiador	2000	2h 35m
24	Beleza Americana	1999	2h 2m
25	Shakespeare Apaixonado	1998	2h 3m
26	Titanic	1997	3h 14m
27	O Paciente Inglês	1996	2h 42m
28	Coração Valente	1995	2h 58m
29	Forrest Gump: O Contador de Histórias	1994	2h 22m
30	A Lista de Schindler	1993	3h 15m
31	Os Imperdoáveis	1992	2h 10m
32	O Silêncio dos Inocentes	1991	1h 58m
33	Dança com Lobos	1990	3h 1m

```
34                          Conduzindo Miss Daisy    1989  1h 39m
35                                      Rain Man    1988  2h 13m
36                         O Último Imperador    1987  2h 43m
37                                       Platoon    1986      2h
38                          Entre Dois Amores    1985  2h 41m
39                                      Amadeus    1984  2h 40m
40                           Laços de Ternura    1983  2h 12m
41                                       Gandhi    1982  3h 11m
42                          Carruagens de Fogo    1981   2h 5m
43                          Gente como a Gente    1980   2h 4m
44                           Kramer vs. Kramer    1979  1h 45m
45                           O Franco Atirador    1978   3h 3m
46                Noivo Neurótico, Noiva Nervosa    1977  1h 33m
47                           Rocky, um Lutador    1976      2h
48                        Um Estranho no Ninho    1975  2h 13m
49                        O Poderoso Chefão II    1974  3h 22m
```

[ ]:
```
df2 = pd.DataFrame(titles_list_2)
df2
```

[ ]:
```
                          title release runtime
0                Rocky, um Lutador    1976      2h
1             Um Estranho no Ninho    1975  2h 13m
2             O Poderoso Chefão II    1974  3h 22m
3                  Golpe de Mestre    1973   2h 9m
4                O Poderoso Chefão    1972  2h 55m
5                  Operação França    1971  1h 44m
6         Patton, Rebelde ou Herói?    1970  2h 52m
7                Perdidos na Noite    1969  1h 53m
8                          Oliver!    1968  2h 33m
9                No Calor da Noite    1967  1h 50m
10  O Homem que Não Vendeu sua Alma    1966      2h
11                 A Noviça Rebelde    1965  2h 52m
12                  Minha Bela Dama    1964  2h 50m
13        As Aventuras de Tom Jones    1963   2h 9m
14               Lawrence da Arábia    1962  3h 38m
15               Amor, Sublime Amor    1961  2h 33m
16      Se Meu Apartamento Falasse    1960   2h 5m
17                          Ben-Hur    1959  3h 32m
18                             Gigi    1958  1h 55m
19             A Ponte do Rio Kwai    1957  2h 41m
20         A Volta ao Mundo em Dias    1956  2h 47m
21                           Marty    1955  1h 30m
22             Sindicato de Ladrões    1954  1h 48m
23         A um Passo da Eternidade    1953  1h 58m
24      O Maior Espetáculo da Terra    1952  2h 32m
25                Sinfonia de Paris    1951  1h 54m
```

```
26                        A Malvada    1950  2h 18m
27                  A Grande Ilusão    1949  1h 50m
28                           Hamlet    1948  2h 34m
29                A Luz é para Todos    1947  1h 58m
30   Os Melhores Anos de Nossa Vida    1946  2h 50m
31                    Farrapo Humano    1945  1h 41m
32                      O Bom Pastor    1944   2h 6m
33                Rosa de Esperança    1942  2h 14m
34                       Casablanca    1942  1h 42m
35        Como Era Verde o Meu Vale    1941  1h 58m
36   Rebecca, a Mulher Inesquecível    1940  2h 10m
37                …E o Vento Levou    1939  3h 58m
38             Do Mundo Nada se Leva    1938   2h 6m
39              A Vida de Emile Zola    1937  1h 56m
40  Ziegfeld, o Criador de Estrelas    1936  2h 56m
41                    O Grande Motim    1935  2h 12m
42          Aconteceu Naquela Noite    1934  1h 45m
43                        Cavalgada    1933  1h 52m
44                     Grande Hotel    1932  1h 52m
45                          Cimarron    1931   2h 3m
46             Sem Novidade no Front    1930  2h 32m
47               Melodia da Broadway    1929  1h 40m
48                             Asas    1927  2h 24m
49                           Aurora    1927  1h 34m
```

Combinando os DataFrames

[ ]:
```
movies = pd.concat([df1, df2], ignore_index=True, sort=False)
movies
```

[ ]:
```
                               title release runtime
0                         Oppenheimer    2023      3h
1   Tudo em Todo o Lugar ao Mesmo Tempo    2022  2h 19m
2                   No Ritmo do Coração    2021  1h 51m
3                             Nomadland    2020  1h 47m
4                              Parasita    2019  2h 12m
..                                  …       …      …
95                          Cimarron    1931   2h 3m
96              Sem Novidade no Front    1930  2h 32m
97                Melodia da Broadway    1929  1h 40m
98                               Asas    1927  2h 24m
99                             Aurora    1927  1h 34m

[100 rows x 3 columns]
```

Removendo os títulos duplicados

[ ]:
```
movies = movies.drop_duplicates()
movies
```

[ ]:

	title	release	runtime
0	Oppenheimer	2023	3h
1	Tudo em Todo o Lugar ao Mesmo Tempo	2022	2h 19m
2	No Ritmo do Coração	2021	1h 51m
3	Nomadland	2020	1h 47m
4	Parasita	2019	2h 12m
..	…	…	…
95	Cimarron	1931	2h 3m
96	Sem Novidade no Front	1930	2h 32m
97	Melodia da Broadway	1929	1h 40m
98	Asas	1927	2h 24m
99	Aurora	1927	1h 34m

[97 rows x 3 columns]