

Pandas dataframe

- when data in the form of tables is load in the pandas then pandas will recognise it as a dataframe
- whole table will be called as dataframe
- single row or single column will be called series

Column Label/ Header		0	1	2	3	4
Index Label		Name	Age	Marks	Grade	Hobby
0	S1	Joe	20	85.10	A	Swimming
1	S2	Nat	21	77.80	B	Reading
2	S3	Harry	19	91.54	A	Music
3	S4	Sam	20	88.78	A	Painting
4	S5	Monica	22	60.55	B	Dancing

Annotations:

- Column Index: Points to the column headers (Name, Age, Marks, Grade, Hobby).
- Row Index: Points to the index values (0, 1, 2, 3, 4).
- Row: Points to a specific row (e.g., Row 3: S4, Sam, 20, 88.78, A, Painting).
- Column: Points to a specific column (e.g., Marks).
- Element/ Value/ Entry: Points to a specific data point (e.g., 88.78).

In [1]:

```
1 import numpy as np
2 import pandas as pd
```

Creating a dataframe

using a list

In [2]:

```
1 student_data = [
2     [100,80,10],
3     [90,70,7],
4     [120,100,14],
5     [80,50,2]
6 ]
7 student_data
```

Out[2]:

```
[[100, 80, 10], [90, 70, 7], [120, 100, 14], [80, 50, 2]]
```

In [3]:

```
1 pd.DataFrame(student_data)
2 # index will be generated automatically
```

Out[3]:

	0	1	2
0	100	80	10
1	90	70	7
2	120	100	14
3	80	50	2

In [4]:

```
1 pd.DataFrame(student_data, columns=['iq', 'marks', 'package'])
```

Out[4]:

	iq	marks	package
0	100	80	10
1	90	70	7
2	120	100	14
3	80	50	2

using dictionary

In [5]:

```
1 student_dict = {
2     'name': ['nitish', 'ankit', 'rupesh', 'rishabh', 'amit', 'disha'],
3     'iq': [100, 90, 120, 80, 0, 0],
4     'marks': [80, 70, 100, 50, 0, 0],
5     'package': [10, 7, 14, 2, 0, 0]
6 }
7
8 students = pd.DataFrame(student_dict)
9 students
```

Out[5]:

	name	iq	marks	package
0	nitish	100	80	10
1	ankit	90	70	7
2	rupesh	120	100	14
3	rishabh	80	50	2
4	amit	0	0	0
5	disha	0	0	0

In [6]:

```
1 students.set_index('name', inplace=True)
2 students
```

Out[6]:

	iq	marks	package
name			
nitish	100	80	10
ankit	90	70	7
rupesh	120	100	14
rishabh	80	50	2
amit	0	0	0
disha	0	0	0

using read_csv()

In [7]:

```
1 movies = pd.read_csv('movies.csv')
2 movies
```

Out[7]:

	title_x	imdb_id	poster_path	
0	Uri: The Surgical Strike	tt8291224	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Uri:_The
1	Battalion 609	tt9472208	NaN	https://en.wikipedia.org/wiki/Ba
2	The Accidental Prime Minister (film)	tt6986710	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/The_Acc

In [8]:

```
1 ipl = pd.read_csv('ipl-matches.csv')
2 ipl
```

Out[8]:

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue	TossWinner
0	1312200	Ahmedabad	2022-05-29	2022	Final	Rajasthan Royals	Gujarat Titans	Narendra Modi Stadium, Ahmedabad	Rajasthan Royals
1	1312199	Ahmedabad	2022-05-27	2022	Qualifier 2	Royal Challengers Bangalore	Rajasthan Royals	Narendra Modi Stadium, Ahmedabad	Rajasthan Royals
2	1312198	Kolkata	2022-05-25	2022	Eliminator	Royal Challengers Bangalore	Lucknow Super Giants	Eden Gardens, Kolkata	Lucknow Super Giants
			2022-			Rajasthan	Gujarat	Eden	Gujarat

DataFrame Attributes and Methods

shape

In [9]:

```
1 movies.shape
2 # returns in tuple form
```

Out[9]:

(1629, 18)

In [10]:

```
1 ipl.shape
```

Out[10]:

(950, 20)

dtypes

In [11]:

```
1 movies.dtypes
2 # here you will get whole series for each attribute
```

Out[11]:

```
title_x          object
imdb_id          object
poster_path      object
wiki_link        object
title_y          object
original_title   object
is_adult         int64
year_of_release  int64
runtime          object
genres           object
imdb_rating      float64
imdb_votes       int64
story            object
summary          object
tagline          object
actors           object
wins_nominations object
release_date     object
dtype: object
```

In [12]:

```
1 ipl.dtypes
```

Out[12]:

```
ID          int64
City        object
Date        object
Season      object
MatchNumber object
Team1       object
Team2       object
Venue       object
TossWinner  object
TossDecision object
SuperOver   object
WinningTeam object
WonBy       object
Margin      float64
method      object
Player_of_Match object
Team1Players object
Team2Players object
```

index

In [13]:

```
1 movies.index
```

Out[13]:

```
RangeIndex(start=0, stop=1629, step=1)
```

In [14]:

```
1 ipl.index
```

Out[14]:

```
RangeIndex(start=0, stop=950, step=1)
```

columns

In [15]:

```
1 movies.columns
```

Out[15]:

```
Index(['title_x', 'imdb_id', 'poster_path', 'wiki_link', 'title_y',  
      'original_title', 'is_adult', 'year_of_release', 'runtime', 'genre  
s',  
      'imdb_rating', 'imdb_votes', 'story', 'summary', 'tagline', 'actor  
s',  
      'wins_nominations', 'release_date'],  
      dtype='object')
```

In [16]:

```
1 ipl.columns
```

Out[16]:

```
Index(['ID', 'City', 'Date', 'Season', 'MatchNumber', 'Team1', 'Team2',  
      'Venue', 'TossWinner', 'TossDecision', 'SuperOver', 'WinningTeam',  
      'WonBy', 'Margin', 'method', 'Player_of_Match', 'Team1Players',  
      'Team2Players', 'Umpire1', 'Umpire2'],  
      dtype='object')
```

values

In [17]:

```

1 movies.values
2 # we will get 2D numpy array

    '4 wins', '11 January 2019 (USA)'],
    ['Battalion 609', 'tt9472208', nan, ...,
     'Vicky Ahuja|Shoaib Ibrahim|Shrikant Kamat|Elena Kazan|Vishwas
Kini|Major Kishore|Jashn Kohli|Rammy C. Pandey|Manish Sharma|Sparsh Sha
rma|Farnaz Shetty|Vikas Shrivastav|Chandraprakash Thakur|Brajesh Tiwari
|',
     nan, '11 January 2019 (India)'],
    ['The Accidental Prime Minister (film)', 'tt6986710',
     'https://upload.wikimedia.org/wikipedia/en/thumb/a/a1/The_Accid
ental_Prime_Minister_film.jpg/220px-The_Accidental_Prime_Minister_film.
jpg',
     ...,
     'Anupam Kher|Akshaye Khanna|Aahana Kumra|Atul Sharma|Manoj Anan
d|Arjun Mathur|Suzanne Bernert|Abdul Quadir Amin|Bharat Mistri|Divya Se
th|Anil Rastogi|Ramesh Bhatkar|Parrgash Kaur|Jess Kaur|',
     nan, '11 January 2019 (USA)'],
    ...,
    ['Sabse Bada Sukh', 'tt0069204', nan, ...,
     'Vijay Arora|Asrani|Rajni Bala|Kumud Damle|Utpal Dutt|Meeta Fai
vvaz|Rahi Ghosh|Tarun Ghosh|Sanjeev Kumar|Keshu Mukherjee|Meena Rai|'.

```

In [18]:

```

1 student_dict = {
2     'name':['nitish','ankit','rupesh','rishabh','amit','disha'],
3     'iq':[100,90,120,80,0,0],
4     'marks':[80,70,100,50,0,0],
5     'package':[10,7,14,2,0,0]
6 }
7
8 students = pd.DataFrame(student_dict)
9 students
10
11 students.set_index('name', inplace=True)
12 students

```

Out[18]:

	iq	marks	package
name			
nitish	100	80	10
ankit	90	70	7
rupesh	120	100	14
rishabh	80	50	2
amit	0	0	0
disha	0	0	0

In [19]:

```
1 students.values
```

Out[19]:

```
array([[100, 80, 10],
       [ 90, 70,  7],
       [120, 100, 14],
       [ 80, 50,  2],
       [  0,  0,  0],
       [  0,  0,  0]], dtype=int64)
```

head and tail

In [20]:

```
1 movies.head()
2 # we will get top five rows by default
```

Out[20]:

	title_x	imdb_id	poster_path	wiki_
0	Uri: The Surgical Strike	tt8291224	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Uri:_The_Surg
1	Battalion 609	tt9472208	NaN	https://en.wikipedia.org/wiki/Battalion_
2	The Accidental Prime Minister (film)	tt6986710	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/The_Accidenta

In [21]:

```
1 movies.head(3)
```

Out[21]:

	title_x	imdb_id	poster_path
0	Uri: The Surgical Strike	tt8291224	https://upload.wikimedia.org/wikipedia/en/thum... https://en.wikipedia.org/wiki/
1	Battalion 609	tt9472208	NaN https://en.wikipedia.or
2	The Accidental Prime Minister (film)	tt6986710	https://upload.wikimedia.org/wikipedia/en/thum... https://en.wikipedia.org/wiki/

In [22]:

```
1 ipl.tail()
2 # we will get last five rows by default
```

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue	TossWinne
945	335986	Kolkata	2008-04-20	2007/08	4	Kolkata Knight Riders	Deccan Chargers	Eden Gardens	Deccan Chargers
946	335985	Mumbai	2008-04-20	2007/08	5	Mumbai Indians	Royal Challengers Bangalore	Wankhede Stadium	Mumbai Indians
947	335984	Delhi	2008-04-19	2007/08	3	Delhi Daredevils	Rajasthan Royals	Feroz Shah Kotla	Rajasthan Royals
								Punjab	

In [23]:

```
1 ipl.tail(4)
```

Out[23]:

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue	TossWinne
946	335985	Mumbai	2008-04-20	2007/08	5	Mumbai Indians	Royal Challengers Bangalore	Wankhede Stadium	Mumbai Indians
947	335984	Delhi	2008-04-19	2007/08	3	Delhi Daredevils	Rajasthan Royals	Feroz Shah Kotla	Rajasthan Royals
948	335983	Chandigarh	2008-04-19	2007/08	2	Kings XI Punjab	Chennai Super Kings	Punjab Cricket Association Stadium, Mohali	Chennai Super Kings

sample

- we can use this to fetch items from the data frame when there is a bias in the data

In [24]:

```
1 ipl.sample()  
2 # it randomly picks any single row
```

Out[24]:

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue	TossWinner	T
604	598020	Delhi	2013-04-18	2013	24	Delhi Daredevils	Chennai Super Kings	Feroz Shah Kotla	Chennai Super Kings	

In [27]:

```
1 ipl.sample(3)
```

Out[27]:

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue	T
376	981013	Bangalore	2016-05-24	2016	Qualifier 1	Gujarat Lions	Royal Challengers Bangalore	M Chinnaswamy Stadium	(
452	829761	Kolkata	2015-05-07	2015	42	Kolkata Knight Riders	Delhi Daredevils	Eden Gardens	
862	392212	Centurion	2009-05-06	2009	32	Deccan Chargers	Mumbai Indians	SuperSport Park	

In [28]:

```
1 movies.sample(5)
```

Out[28]:

	title_x	imdb_id	poster_path	w
65	Bypass Road (film)	tt9176260	https://upload.wikimedia.org/wikipedia/en/thum... https://en.wikipedia.org/wiki/Bypass_Road_(film)	
1534	Kyaa Dil Ne Kahaa	tt0327005	https://upload.wikimedia.org/wikipedia/en/thum... https://en.wikipedia.org/wiki/Kyaa_Dil_Ne_Kahaa	
1238	Chehrra	tt0449870	https://upload.wikimedia.org/wikipedia/en/thum... https://en.wikipedia.org/wiki/Chehrra_(2006_film)	

info

In [31]:

```
1 ipl.describe()
```

Out[31]:

	ID	Margin
count	9.500000e+02	932.000000
mean	8.304852e+05	17.056867
std	3.375678e+05	21.633109
min	3.359820e+05	1.000000
25%	5.012612e+05	6.000000
50%	8.297380e+05	8.000000
75%	1.175372e+06	19.000000
max	1.312200e+06	146.000000

In [32]:

```
1 movies.describe()
```

Out[32]:

	is_adult	year_of_release	imdb_rating	imdb_votes
count	1629.0	1629.000000	1629.000000	1629.000000
mean	0.0	2010.263966	5.557459	5384.263352
std	0.0	5.381542	1.567609	14552.103231
min	0.0	2001.000000	0.000000	0.000000
25%	0.0	2005.000000	4.400000	233.000000
50%	0.0	2011.000000	5.600000	1000.000000
75%	0.0	2015.000000	6.800000	4287.000000
max	0.0	2019.000000	9.400000	310481.000000

isnull

- it checks the presence of null values
- it gives boolean dataframe

In [33]:

```
1 ipl.isnull()
```

Out[33]:

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue	TossWinner	TossDecision	Super
0	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False
...
945	False	False	False	False	False	False	False	False	False	False	False
946	False	False	False	False	False	False	False	False	False	False	False
947	False	False	False	False	False	False	False	False	False	False	False
948	False	False	False	False	False	False	False	False	False	False	False

In [35]:

```
1 # to get the number of null values we will add .sum()  
2 ipl.isnull().sum()
```

Out[35]:

ID	0
City	51
Date	0
Season	0
MatchNumber	0
Team1	0
Team2	0
Venue	0
TossWinner	0
TossDecision	0
SuperOver	4
WinningTeam	4
WonBy	0
Margin	18
method	931
Player_of_Match	4
Team1Players	0
Team2Players	0

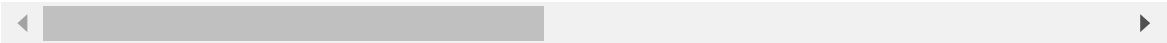
In [36]:

```
1 movies.isnull()
```

Out[36]:

	title_x	imdb_id	poster_path	wiki_link	title_y	original_title	is_adult	year_of_release
0	False	False	False	False	False	False	False	False
1	False	False	True	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	True	False	False	False	False	False
...
1624	False	False	False	False	False	False	False	False
1625	False	False	False	False	False	False	False	False
1626	False	False	True	False	False	False	False	False
1627	False	False	False	False	False	False	False	False
1628	False	False	False	False	False	False	False	False

1629 rows × 18 columns



In [38]:

```
1 movies.isnull().sum()
```

Out[38]:

```
title_x          0
imdb_id          0
poster_path     103
wiki_link        0
title_y          0
original_title   0
is_adult         0
year_of_release  0
runtime          0
genres           0
imdb_rating      0
imdb_votes       0
story           20
summary          0
tagline         1072
actors           5
wins_nominations 922
release_date     107
dtype: int64
```

deduplicated

In [39]:

```
1 ipl.duplicated()
```

Out[39]:

```
0      False
1      False
2      False
3      False
4      False
...
945    False
946    False
947    False
948    False
949    False
Length: 950, dtype: bool
```

In [40]:

```
1 ipl.duplicated().sum()
```

Out[40]:

```
0
```

In [41]:

```
1 movies.duplicated()
```

Out[41]:

```
0      False
1      False
2      False
3      False
4      False
...
1624   False
1625   False
1626   False
1627   False
1628   False
Length: 1629, dtype: bool
```

In [42]:

```
1 movies.duplicated().sum()
2 # to get the total number of duplicated rows
```

Out[42]:

```
0
```


In [43]:

```
1 student_dict = {  
2     'iq':[100,90,120,80,0,0],  
3     'marks':[80,70,100,50,0,0],  
4     'package':[10,7,14,2,0,0]  
5 }  
6  
7 students = pd.DataFrame(student_dict)  
8 students
```

Out[43]:

	iq	marks	package
0	100	80	10
1	90	70	7
2	120	100	14
3	80	50	2
4	0	0	0
5	0	0	0

In [44]:

```
1 students.duplicated().sum()
```

Out[44]:

1

rename

- help us to rename the column name of the dataframe

In [45]:

```
1 students  
2 # we want percentage instead of marks  
3 # and instead of package we want LPA
```

Out[45]:

	iq	marks	package
0	100	80	10
1	90	70	7
2	120	100	14
3	80	50	2
4	0	0	0
5	0	0	0

In [46]:

```
1 students.rename(columns={"marks":"percent","package":"LPA"})
2 # we are passing the dictionary using columns parameter inside rename
3 # this is not permanent change
4 # for permanent change use inplace=True parameter
```

Out[46]:

	iq	percent	LPA
0	100	80	10
1	90	70	7
2	120	100	14
3	80	50	2
4	0	0	0
5	0	0	0

In [47]:

```
1 students.rename(index={0:"zero",1:"first",2:"second",
2                        3:"third",4:"fourth",5:"fifth"})
```

Out[47]:

	iq	marks	package
zero	100	80	10
first	90	70	7
second	120	100	14
third	80	50	2
fourth	0	0	0
fifth	0	0	0

Math Methods

In [48]:

```
1 student_dict = {  
2     'iq':[100,90,120,80,0,0],  
3     'marks':[80,70,100,50,0,0],  
4     'package':[10,7,14,2,0,0]  
5 }  
6  
7 students = pd.DataFrame(student_dict)  
8 students
```

Out[48]:

	iq	marks	package
0	100	80	10
1	90	70	7
2	120	100	14
3	80	50	2
4	0	0	0
5	0	0	0

sum

In [49]:

```
1 '''this function will automatically apply sum on the  
2 columns of the dataframe'''  
3 students.sum()
```

Out[49]:

```
iq          390  
marks       300  
package      33  
dtype: int64
```

In [50]:

```
1 # we can apply sum on rows also by using axis parameter  
2 students.sum(axis=1)
```

Out[50]:

```
0    190  
1    167  
2    234  
3    132  
4      0  
5      0  
dtype: int64
```

In [51]:

```

1 '''even if string is present in the data, it will add them
2 and concatination will happen'''
3 movies.sum()

```

C:\Users\gadha\AppData\Local\Temp\ipykernel_2036\3281507241.py:3: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError. Select only valid columns before calling the reduction.

```
movies.sum()
```

Out[51]:

```

title_x      Uri: The Surgical StrikeBattalion 609The Accid...
imdb_id      tt8291224tt9472208tt6986710tt8108208tt6028796t...
wiki_link    https://en.wikipedia.org/wiki/Uri:_The_Surgica... (http
s://en.wikipedia.org/wiki/Uri:_The_Surgica...)
title_y      Uri: The Surgical StrikeBattalion 609The Accid...
original_title  Uri: The Surgical StrikeBattalion 609The Accid...
is_adult      0
year_of_release  3274720
runtime      1381311121211029710910414812013415314313014311...
genres      Action|Drama|WarWarBiography|DramaCrime|DramaD...
imdb_rating  9053.1
imdb_votes   8770965
summary      Indian army special forces execute a covert op...
dtype: object

```

In [48]:

```
1 ipl.sum()
```

C:\Users\gadha\AppData\Local\Temp\ipykernel_4404\599940423.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError. Select only valid columns before calling the reduction.

```
ipl.sum()
```

Out[48]:

```

ID      788960985
Date    2022-05-292022-05-272022-05-252022-05-242022-0...
Season  2022202220222022202220222022202220222022202220...
MatchNumber  FinalQualifier 2EliminatorQualifier 1706968676...
Team1    Rajasthan RoyalsRoyal Challengers BangaloreRoy...
Team2    Gujarat TitansRajasthan RoyalsLucknow Super Gi...
Venue    Narendra Modi Stadium, AhmedabadNarendra Modi ...
TossWinner  Rajasthan RoyalsRajasthan RoyalsLucknow Super ...
TossDecision  batfieldfieldfieldbatfieldbatbatbatfieldb...
WonBy     WicketsWicketsRunsWicketsWicketsWicketsWickets...
Margin    15897.0
Team1Players  ['YBK Jaiswal', 'JC Buttler', 'SV Samson', 'D ...

```

In [49]:

```
1 ipl.sum(axis=1)
2 # row wise
```

C:\Users\gadha\AppData\Local\Temp\ipykernel_4404\3452818570.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError. Select only valid columns before calling the reduction.

```
ipl.sum(axis=1)
```

Out[49]:

```
0      1312207.0
1      1312206.0
2      1312212.0
3      1312204.0
4      1304121.0
...
945     335991.0
946     335990.0
947     335993.0
948     336016.0
949     336122.0
Length: 950, dtype: float64
```

mean

In [50]:

```
1 students.mean()
```

Out[50]:

```
iq      65.0
marks   50.0
package  5.5
dtype: float64
```

In [51]:

```
1 students.mean(axis=1)
2 # for row wise mean
```

Out[51]:

```
0      63.333333
1      55.666667
2      78.000000
3      44.000000
4       0.000000
5       0.000000
dtype: float64
```

median

In [52]:

```
1 students.median()
```

Out[52]:

```
iq          85.0
marks       60.0
package     4.5
dtype: float64
```

var

In [55]:

```
1 students.var()
```

Out[55]:

```
iq          2710.0
marks       1760.0
package     33.5
dtype: float64
```

In [56]:

```
1 students.var(axis=1)
2 # row wise variance
```

Out[56]:

```
0    2233.333333
1    1876.333333
2    3172.000000
3    1548.000000
4         0.000000
5         0.000000
dtype: float64
```

std

In [57]:

```
1 students.std()
```

Out[57]:

```
iq          52.057660
marks       41.952354
package     5.787918
dtype: float64
```

min

In [58]:

```
1 students.min()
```

Out[58]:

```
iq          0
marks       0
package     0
dtype: int64
```

In [59]:

```
1 students.min(axis=1)
```

Out[59]:

```
0    10
1     7
2    14
3     2
4     0
5     0
dtype: int64
```

max

In [60]:

```
1 students.max()
```

Out[60]:

```
iq          120
marks       100
package     14
dtype: int64
```

In [61]:

```
1 students.max(axis=1)
```

Out[61]:

```
0    100
1     90
2    120
3     80
4     0
5     0
dtype: int64
```

Selecting cols from a DataFrame

single column

In [62]:

```
1 movies.head(2)
```

Out[62]:

	title_x	imdb_id	poster_path
0	Uri: The Surgical Strike	tt8291224	https://upload.wikimedia.org/wikipedia/en/thum... https://en.wikipedia.org/wiki/U
1	Battalion 609	tt9472208	NaN https://en.wikipedia.org/

In [63]:

```
1 movies['title_x']
```

Out[63]:

```
0          Uri: The Surgical Strike
1          Battalion 609
2    The Accidental Prime Minister (film)
3          Why Cheat India
4          Evening Shadows
...
1624      Tera Mera Saath Rahen
1625      Yeh Zindagi Ka Safar
1626      Sabse Bada Sukh
1627          Daaka
1628      Humsafar
Name: title_x, Length: 1629, dtype: object
```

In [64]:

```
1 type(movies['title_x'])
2 # fetched single column will be of series datatype
```

Out[64]:

pandas.core.series.Series

multiple columns

In [65]:

```
1 movies[['title_x','year_of_release','actors']]
2 # we will get dataframe because multiple columns are there
```

Out[65]:

	title_x	year_of_release	actors
0	Uri: The Surgical Strike	2019	Vicky Kaushal Paresh Rawal Mohit Raina Yami Ga...
1	Battalion 609	2019	Vicky Ahuja Shoaib Ibrahim Shrikant Kamat Elen...
2	The Accidental Prime Minister (film)	2019	Anupam Kher Akshaye Khanna Aahana Kumra Atul S...
3	Why Cheat India	2019	Emraan Hashmi Shreya Dhanwanthary Snighdadeep ...
4	Evening Shadows	2018	Mona Ambegaonkar Ananth Narayan Mahadevan Deva...
...
1624	Tera Mera Saath Rahen	2001	Ajay Devgn Sonali Bendre Namrata Shirodkar Pre...
1625	Yeh Zindagi Ka Safar	2001	Ameesha Patel Jimmy Sheirgill Nafisa Ali Gulsh...
1626	Sabse Bada Sukh	2018	Vijay Arora Asrani Rajni Bala Kumud Damle Utpa...
1627	Daaka	2019	Gippy Grewal Zareen Khan
1628	Humsafar	2011	Fawad Khan

1629 rows × 3 columns

In [66]:

```
1 # we can give the order for getting columns
2 movies[['year_of_release', 'actors', 'title_x']]
```

Out[66]:

	year_of_release	actors	title_x
0	2019	Vicky Kaushal Paresh Rawal Mohit Raina Yami Ga...	Uri: The Surgical Strike
1	2019	Vicky Ahuja Shoaib Ibrahim Shrikant Kamat Elen...	Battalion 609
2	2019	Anupam Kher Akshaye Khanna Aahana Kumra Atul S...	The Accidental Prime Minister (film)
3	2019	Emraan Hashmi Shreya Dhanwanthary Snighdadeep ...	Why Cheat India
4	2018	Mona Ambegaonkar Ananth Narayan Mahadevan Deva...	Evening Shadows
...
1624	2001	Ajay Devgn Sonali Bendre Namrata Shirodkar Pre...	Tera Mera Saath Rahen
1625	2001	Ameesha Patel Jimmy Sheirgill Nafisa Ali Gulsh...	Yeh Zindagi Ka Safar
1626	2018	Vijay Arora Asrani Rajni Bala Kumud Damle Utpa...	Sabse Bada Sukh
1627	2019	Gippy Grewal Zareen Khan	Daaka
1628	2011	Fawad Khan	Humsafar

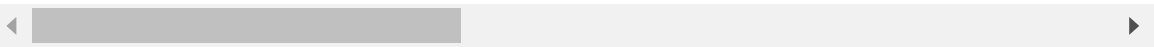
1629 rows × 3 columns

In [67]:

```
1 ipl.head(2)
```

Out[67]:

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue	Ti
0	1312200	Ahmedabad	2022-05-29	2022	Final	Rajasthan Royals	Gujarat Titans	Narendra Modi Stadium, Ahmedabad	
1	1312199	Ahmedabad	2022-05-27	2022	Qualifier 2	Royal Challengers Bangalore	Rajasthan Royals	Narendra Modi Stadium, Ahmedabad	



In [70]:

```
1 ipl[['Team1', 'Team2', 'WinningTeam']]
```

Out[70]:

	Team1	Team2	WinningTeam
0	Rajasthan Royals	Gujarat Titans	Gujarat Titans
1	Royal Challengers Bangalore	Rajasthan Royals	Rajasthan Royals
2	Royal Challengers Bangalore	Lucknow Super Giants	Royal Challengers Bangalore
3	Rajasthan Royals	Gujarat Titans	Gujarat Titans
4	Sunrisers Hyderabad	Punjab Kings	Punjab Kings
...
945	Kolkata Knight Riders	Deccan Chargers	Kolkata Knight Riders
946	Mumbai Indians	Royal Challengers Bangalore	Royal Challengers Bangalore
947	Delhi Daredevils	Rajasthan Royals	Delhi Daredevils
948	Kings XI Punjab	Chennai Super Kings	Chennai Super Kings
949	Royal Challengers Bangalore	Kolkata Knight Riders	Kolkata Knight Riders

950 rows × 3 columns

Selecting rows from a DataFrame

- **iloc** - searches using index positions
- **loc** - searches using index labels

1. iloc

single row

In [71]:

```
1 movies.head(3)
```

Out[71]:

	title_x	imdb_id	poster_path
0	Uri: The Surgical Strike	tt8291224	https://upload.wikimedia.org/wikipedia/en/thum... https://en.wikipedia.org/wiki/Uri:_The_Surgical_Strike
1	Battalion 609	tt9472208	NaN https://en.wikipedia.or
2	The Accidental Prime Minister (film)	tt6986710	https://upload.wikimedia.org/wikipedia/en/thum... https://en.wikipedia.org/wiki/The_Accidental_Prime_Minister_(film)

In [72]:

```
1 # suppose we want first row
2 movies.iloc[0]
3 # it will be a series because it's a single row
```

Out[72]:

```
title_x          Uri: The Surgical Strike
imdb_id          tt8291224
poster_path      https://upload.wikimedia.org/wikipedia/en/thum... (htt
ps://upload.wikimedia.org/wikipedia/en/thum...)
wiki_link        https://en.wikipedia.org/wiki/Uri:_The_Surgica... (htt
ps://en.wikipedia.org/wiki/Uri:_The_Surgica...)
title_y          Uri: The Surgical Strike
original_title    Uri: The Surgical Strike
is_adult          0
year_of_release  2019
runtime          138
genres            Action|Drama|War
imdb_rating       8.4
imdb_votes        35112
story            Divided over five chapters the film chronicle...
summary          Indian army special forces execute a covert op...
tagline           NaN
actors            Vicky Kaushal|Paresh Rawal|Mohit Raina|Yami Ga...
wins_nominations  4 wins
release_date      11 January 2019 (USA)
Name: 0, dtype: object
```

In [73]:

```
1 type(movies.iloc[0])
```

Out[73]:

pandas.core.series.Series

multiple row

In [74]:

```
1 movies.iloc[0:5]
2 # this will be dataframe data type
```

Out[74]:

	title_x	imdb_id	poster_path	wiki_
0	Uri: The Surgical Strike	tt8291224	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Uri:_The_Surg
1	Battalion 609	tt9472208	NaN	https://en.wikipedia.org/wiki/Battalion_
2	The Accidental Prime Minister (film)	tt6986710	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/The_Accidenta

In [75]:

```
1 # we want alternate movies between 5 to 15
2 movies.iloc[5:15:2]
```

Out[75]:

	title_x	imdb_id	poster_path	wiki_
5	Soni (film)	tt6078866	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Soni_(f
7	Bombairiya	tt4971258	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Bombai
9	Thackeray (film)	tt7777196	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Thackeray_(f

fancy indexing

In [76]:

```
1 movies.iloc[[0,4,7,8]]
2 # we are passing list of items we want
```

Out[76]:

	title_x	imdb_id	poster_path	wi
0	Uri: The Surgical Strike	tt8291224	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Uri:_The_Su
4	Evening Shadows	tt6028796	NaN	https://en.wikipedia.org/wiki/Evening_Sh
7	Bombairiya	tt4971258	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Bom

2. loc

- when we provide range for fetching multiple items then the last number of given range will be included where as in iloc the last number is not included

In [77]:

```
1 student_dict = {
2     'name': ['nitish', 'ankit', 'rupesh', 'rishabh', 'amit', 'himanshu'],
3     'iq': [100, 90, 120, 80, 0, 0],
4     'marks': [80, 70, 100, 50, 0, 0],
5     'package': [10, 7, 14, 2, 0, 0]
6 }
7
8 students = pd.DataFrame(student_dict)
9 students.set_index('name', inplace=True)
10 # we are setting name columns as our index
11 students
```

Out[77]:

	iq	marks	package
name			
nitish	100	80	10
ankit	90	70	7
rupesh	120	100	14
rishabh	80	50	2
amit	0	0	0
himanshu	0	0	0

single row

In [78]:

```
1 students.loc['rupesh']
```

Out[78]:

```
iq      120
marks   100
package   14
Name: rupesh, dtype: int64
```

multiple rows

In [79]:

```
1 students.loc['nitish':'rishabh']
```

Out[79]:

	iq	marks	package
name			
nitish	100	80	10
ankit	90	70	7
rupesh	120	100	14
rishabh	80	50	2

In [80]:

```
1 students.loc['nitish':'rishabh':2]  
2 # it will print alternare rows because step value is 2
```

Out[80]:

	iq	marks	package
name			
nitish	100	80	10
rupesh	120	100	14

fancy indexing

In [81]:

```
1 students.loc[['nitish','rupesh','himanshu']]
```

Out[81]:

	iq	marks	package
name			
nitish	100	80	10
rupesh	120	100	14
himanshu	0	0	0

- **Note** : we can also use `iloc` on the students data though we have name column as index, because there is default index by pandas so here 0 and nitish will point out on same row position

In [82]:

```
1 students.iloc[[1,3,5]]
```

Out[82]:

	iq	marks	package
name			
ankit	90	70	7
rishabh	80	50	2
himanshu	0	0	0

Selecting both rows and columns

In [83]:

```
1 movies.iloc[0:3,0:3]
```

Out[83]:

		title_x	imdb_id	poster_path
0	Uri: The Surgical Strike	tt8291224	https://upload.wikimedia.org/wikipedia/en/thum...	
1	Battalion 609	tt9472208		NaN
2	The Accidental Prime Minister (film)	tt6986710	https://upload.wikimedia.org/wikipedia/en/thum...	

In [84]:

```
1 movies.loc[0:2,'title_x':'poster_path']
```

Out[84]:

		title_x	imdb_id	poster_path
0	Uri: The Surgical Strike	tt8291224	https://upload.wikimedia.org/wikipedia/en/thum...	
1	Battalion 609	tt9472208		NaN
2	The Accidental Prime Minister (film)	tt6986710	https://upload.wikimedia.org/wikipedia/en/thum...	

Filtering a DataFrame

In [85]:

```
1 ipl.head()
```

Out[85]:

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue	TossWinner	To
0	1312200	Ahmedabad	2022-05-29	2022	Final	Rajasthan Royals	Gujarat Titans	Narendra Modi Stadium, Ahmedabad	Rajasthan Royals	To
1	1312199	Ahmedabad	2022-05-27	2022	Qualifier 2	Royal Challengers Bangalore	Rajasthan Royals	Narendra Modi Stadium, Ahmedabad	Rajasthan Royals	To
2	1312198	Kolkata	2022-05-25	2022	Eliminator	Royal Challengers Bangalore	Lucknow Super Giants	Eden Gardens, Kolkata	Lucknow Super Giants	To
			2022-			Rajasthan	Gujarat	Eden	Gujarat	

- Find all the final winners

In [86]:

```
1 '''we can see in the data that the Final match of each season
2 is labeled, so we will fetch that first'''
3
4 ipl["MatchNumber"] == "Final"
5 # with this code we will get the boolean series
```

Out[86]:

```
0      True
1     False
2     False
3     False
4     False
...
945    False
946    False
947    False
948    False
949    False
Name: MatchNumber, Length: 950, dtype: bool
```

In [87]:

```
1 ipl[ipl["MatchNumber"] == "Final"]
2 # we will get the data of final matches only
```

Out[87]:

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue	TossWin
0	1312200	Ahmedabad	2022-05-29	2022	Final	Rajasthan Royals	Gujarat Titans	Narendra Modi Stadium, Ahmedabad	Rajas Rc
74	1254117	Dubai	2021-10-15	2021	Final	Chennai Super Kings	Kolkata Knight Riders	Dubai International Cricket Stadium	Ko Ki Ri
134	1237181	NaN	2020-11-10	2020/21	Final	Delhi Capitals	Mumbai Indians	Dubai International Cricket Stadium	[Cap
194	1181768	Hyderabad	2019-	2019	Final	Mumbai	Chennai Super	Rajiv Gandhi International	Mu

In [88]:

```
1 # storing this result in new dataframe
2 new_df = ipl[ipl["MatchNumber"] == "Final"]
```

In [89]:

```
1 new_df[["Season", "WinningTeam"]]
2 # we are doing fancy indexing by passing a list
```

Out[89]:

	Season	WinningTeam
0	2022	Gujarat Titans
74	2021	Chennai Super Kings
134	2020/21	Mumbai Indians
194	2019	Mumbai Indians
254	2018	Chennai Super Kings
314	2017	Mumbai Indians
373	2016	Sunrisers Hyderabad
433	2015	Mumbai Indians
492	2014	Kolkata Knight Riders
552	2013	Mumbai Indians
628	2012	Kolkata Knight Riders

In [90]:

```
1 # we can do that in one line also
2 ipl[ipl["MatchNumber"] == "Final"][["Season", "WinningTeam"]]
```

Out[90]:

	Season	WinningTeam
0	2022	Gujarat Titans
74	2021	Chennai Super Kings
134	2020/21	Mumbai Indians
194	2019	Mumbai Indians
254	2018	Chennai Super Kings
314	2017	Mumbai Indians
373	2016	Sunrisers Hyderabad
433	2015	Mumbai Indians
492	2014	Kolkata Knight Riders
552	2013	Mumbai Indians
628	2012	Kolkata Knight Riders

- how many super over finishes have occurred

In [91]:

```
1 ipl["SuperOver"] == "Y"
```

Out[91]:

```
0    False
1    False
2    False
3    False
4    False
...
945  False
946  False
947  False
948  False
949  False
Name: SuperOver, Length: 950, dtype: bool
```

In [92]:

```
1 ipl[ipl["SuperOver"] == "Y"]
```

Out[92]:

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue	TossWinn
114	1254077	Chennai	2021-04-25	2021	20	Delhi Capitals	Sunrisers Hyderabad	MA Chidambaram Stadium, Chepauk, Chennai	De Capital
158	1216512	Abu Dhabi	2020-10-18	2020/21	35	Kolkata Knight Riders	Sunrisers Hyderabad	Sheikh Zayed Stadium	Sunrise Hyderabad
159	1216517	NaN	2020-10-18	2020/21	36	Mumbai Indians	Kings XI Punjab	Dubai International Cricket Stadium	Mumt India
184	1216547	NaN	2020-	2020/21	10	Royal Challengers	Mumbai	Dubai International	Mumt

In [93]:

```
1 ipl[ipl["SuperOver"] == "Y"].shape[0]
2 # so far 14 matches had a superover
```

Out[93]:

14

- how many matches has csk won in kolkata

In [94]:

```
1 ipl.head()
```

Out[94]:

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue	TossWinner	Toss
0	1312200	Ahmedabad	2022-05-29	2022	Final	Rajasthan Royals	Gujarat Titans	Narendra Modi Stadium, Ahmedabad	Rajasthan Royals	
1	1312199	Ahmedabad	2022-05-27	2022	Qualifier 2	Royal Challengers Bangalore	Rajasthan Royals	Narendra Modi Stadium, Ahmedabad	Rajasthan Royals	
2	1312198	Kolkata	2022-05-25	2022	Eliminator	Royal Challengers Bangalore	Lucknow Super Giants	Eden Gardens, Kolkata	Lucknow Super Giants	
3	1312197	Kolkata	2022-05-24	2022	Qualifier 1	Rajasthan Royals	Gujarat Titans	Eden Gardens, Kolkata	Gujarat Titans	
207	1178422	Kolkata	2019-04-28	2019	47	Kolkata Knight Riders	Mumbai Indians	Eden Gardens	Mumbai Indians	
211	1178418	Kolkata	2019-04-27	2019	43	Kolkata Knight Riders	Rajasthan Royals	Eden Gardens	Rajasthan Royals	

In [95]:

```
1 ipl[ipl["City"] == "Kolkata"]
2 # this are matches played in kolkata vanue
```

Out[95]:

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue	TossWinner	Toss
2	1312198	Kolkata	2022-05-25	2022	Eliminator	Royal Challengers Bangalore	Lucknow Super Giants	Eden Gardens, Kolkata	Lucknow Super Giants	
3	1312197	Kolkata	2022-05-24	2022	Qualifier 1	Rajasthan Royals	Gujarat Titans	Eden Gardens, Kolkata	Gujarat Titans	
207	1178422	Kolkata	2019-04-28	2019	47	Kolkata Knight Riders	Mumbai Indians	Eden Gardens	Mumbai Indians	
211	1178418	Kolkata	2019-04-27	2019	43	Kolkata Knight Riders	Rajasthan Royals	Eden Gardens	Rajasthan Royals	

In [100]:

```
1 (ipl['City'] == 'Kolkata') & (ipl['WinningTeam'] == 'Chennai Super Kings')
```

Out[100]:

```
0      False
1      False
2      False
3      False
4      False
...
945     False
946     False
947     False
948     False
949     False
Length: 950, dtype: bool
```

In [101]:

```
1 ipl[(ipl['City'] == 'Kolkata') & (ipl['WinningTeam'] == 'Chennai Super Kings')]
```

Out[101]:

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue	TossWinner	TossDecision
224	1178404	Kolkata	2019-04-14	2019	29	Kolkata Knight Riders	Chennai Super Kings	Eden Gardens	Chennai Super Kings	field
602	598022	Kolkata	2013-04-20	2013	26	Kolkata Knight Riders	Chennai Super Kings	Eden Gardens	Kolkata Knight Riders	ba
641	548368	Kolkata	2012-05-14	2012	63	Kolkata Knight Riders	Chennai Super Kings	Eden Gardens	Chennai Super Kings	field
						Kolkata	Chennai			

In [102]:

```
1 ipl[(ipl['City'] == 'Kolkata') &
2     (ipl['WinningTeam'] == 'Chennai Super Kings')].shape[0]
```

Out[102]:

5

- toss winner is match winner in percentage

In [103]:

```
1 ipl.head(1)
```

Out[103]:

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue	TossV
0	1312200	Ahmedabad	2022-05-29	2022	Final	Rajasthan Royals	Gujarat Titans	Narendra Modi Stadium, Ahmedabad	Raji

In [104]:

```
1 ipl['TossWinner'] == ipl['WinningTeam']
2 # we are doing compariosion for this two column
```

Out[104]:

```
0      False
1       True
2      False
3       True
4      False
...
945     False
946     False
947     False
948      True
949     False
Length: 950, dtype: bool
```

In [105]:

```
1 ipl[ipl['TossWinner'] == ipl['WinningTeam']]
```

Out[105]:

	ID	City	Date	Season	MatchNumber	Team1	Team2	Venue	TossWinner
1	1312199	Ahmedabad	2022-05-27	2022	Qualifier 2	Royal Challengers Bangalore	Rajasthan Royals	Narendra Modi Stadium, Ahmedabad	Rajasthan Royals
3	1312197	Kolkata	2022-05-24	2022	Qualifier 1	Rajasthan Royals	Gujarat Titans	Eden Gardens, Kolkata	Gujarat Titans
5	1304115	Mumbai	2022-05-21	2022	69	Delhi Capitals	Mumbai Indians	Wankhede Stadium, Mumbai	Mumbai Indians
8	1304112	Navi	2022-	2022	66	Lucknow Super	Kolkata Knight	Dr DY Patil Sports	Lucknow Super

In [106]:

```
1 ipl[ipl['TossWinner'] == ipl['WinningTeam']].shape[0]
```

Out[106]:

489

In [107]:

```
1 ipl.shape[0]
```

Out[107]:

950

In [108]:

```
1 a = ipl[ipl['TossWinner'] == ipl['WinningTeam']].shape[0]
2 b = ipl.shape[0]
3
4 percentage = (a/b)*100
5
6 percentage
```

Out[108]:

51.473684210526315

- movies with rating higher than 8 and votes>10000

In [109]:

```
1 movies
```

Out[109]:

	title_x	imdb_id	poster_path	
0	Uri: The Surgical Strike	tt8291224	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Uri:_The
1	Battalion 609	tt9472208	NaN	https://en.wikipedia.org/wiki/Ba
2	The Accidental Prime Minister (film)	tt6986710	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/The_Acc

In [110]:

```
1 movies[(movies['imdb_rating'] > 8) & (movies['imdb_votes'] > 10000 )]
```

Out[110]:

	title_x	imdb_id	poster_path	
0	Uri: The Surgical Strike	tt8291224	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Ur
11	Gully Boy	tt2395469	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.o
37	Article 15 (film)	tt10324144	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wik
40	Super 30 (film)	tt7485048	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wik

In [111]:

```
1 (movies['imdb_rating'] > 8) & (movies['imdb_votes'] > 10000 )
```

Out[111]:

```
0      True
1     False
2     False
3     False
4     False
...
1624  False
1625  False
1626  False
1627  False
1628  False
Length: 1629, dtype: bool
```

In [113]:

```
1 movies[(movies['imdb_rating'] > 8) & (
2     movies['imdb_votes'] > 10000 )].shape[0]
```

Out[113]:

43

- Action movies with rating higher than 7.5

In [114]:

```
1 movies.head()
2 '''here in data if we look in to genres,
3 there is combination of genres in that column'''
```

Out[114]:

'here in data if we look in to genres,\nthere is combination of genres in that column'

In [115]:

```
1 movies['genres']
```

Out[115]:

```
0      Action|Drama|War
1                War
2      Biography|Drama
3        Crime|Drama
4          Drama
...
1624          Drama
1625          Drama
1626      Comedy|Drama
1627          Action
1628      Drama|Romance
Name: genres, Length: 1629, dtype: object
```

In [116]:

```
1 '''in the above column that data type is string,
2 now we want only action movies out of that,
3 if we directly apply split(|) on that Series then
4 it will throw an error,
5 so we need to convert it in string object method'''
6
7 movies['genres'].str.split('|')
```

Out[116]:

```
0      [Action, Drama, War]
1                [War]
2      [Biography, Drama]
3        [Crime, Drama]
4          [Drama]
...
1624          [Drama]
1625          [Drama]
1626      [Comedy, Drama]
1627          [Action]
1628      [Drama, Romance]
Name: genres, Length: 1629, dtype: object
```

In [117]:

```
1 movies['genres'].str.split('|').apply(lambda x: 'Action' in x)
2 # here we are checking if action genre is present or not
```

Out[117]:

```
0      True
1     False
2     False
3     False
4     False
...
1624    False
1625    False
1626    False
1627     True
1628    False
Name: genres, Length: 1629, dtype: bool
```

In [119]:

```
1 mask1 = movies['genres'].str.split('|').apply(lambda x: 'Action' in x)
2
3 # now we want movies with rating > 7.5
4
5 mask2 = movies['imdb_rating'] > 7.5
```

In [120]:

```
1 movies[mask1 & mask2].head()
```

Out[120]:

	title_x	imdb_id	poster_path	
0	Uri: The Surgical Strike	tt8291224	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Uri:_The_
41	Family of Thakurganj	tt8897986	https://upload.wikimedia.org/wikipedia/en/9/99...	https://en.wikipedia.org/wiki/Family_of
84	Mukkabaaz	tt7180544	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/M

In [121]:

```
1 # using containg function
2
3 mask1 = movies['genres'].str.contains('Action')
4
5 '''now we want movies with rating > 7.5'''
6
7 mask2 = movies['imdb_rating'] > 7.5
8
9 movies[mask1 & mask2].head()
```

Out[121]:

	title_x	imdb_id	poster_path	
0	Uri: The Surgical Strike	tt8291224	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Uri:_The_
41	Family of Thakurganj	tt8897986	https://upload.wikimedia.org/wikipedia/en/9/99...	https://en.wikipedia.org/wiki/Family_of
84	Mukkabaaz	tt7180544	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/M

- write a function that can return the track record of 2 teams against each other

In []:

```
1
```

Adding new cols

In [122]:

```
1 movies.head()
```

Out[122]:

	title_x	imdb_id	poster_path	wiki_
0	Uri: The Surgical Strike	tt8291224	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Uri:_The_Surg
1	Battalion 609	tt9472208	NaN	https://en.wikipedia.org/wiki/Battalion
2	The Accidental Prime Minister (film)	tt6986710	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/The_Accidenta

In [123]:

```
1 movies['Country'] = "India"
```

In [124]:

```
1 movies.head()
2 # new country column will be added in the dataframe
```

Out[124]:

	title_x	imdb_id	poster_path	wiki_
0	Uri: The Surgical Strike	tt8291224	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Uri:_The_Surg
1	Battalion 609	tt9472208	NaN	https://en.wikipedia.org/wiki/Battalion
2	The Accidental Prime Minister (film)	tt6986710	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/The_Accidenta

- we want to make new column lead actor and here value will come from Actors column. The first name in the Actors column will be the lead actor of that particular movies, so we want that.

In [125]:

```
1 movies['actors'].head()
```

Out[125]:

```
0 Vicky Kaushal|Paresh Rawal|Mohit Raina|Yami Ga...
1 Vicky Ahuja|Shoaib Ibrahim|Shrikant Kamat|Elen...
2 Anupam Kher|Akshaye Khanna|Aahana Kumra|Atul S...
3 Emraan Hashmi|Shreya Dhanwanthary|Snighdadeep ...
4 Mona Ambegaonkar|Ananth Narayan Mahadevan|Deva...
Name: actors, dtype: object
```

In [126]:

```
1 movies['actors'].str.split('|')
```

Out[126]:

```
0 [Vicky Kaushal, Paresh Rawal, Mohit Raina, Yam...
1 [Vicky Ahuja, Shoaib Ibrahim, Shrikant Kamat, ...
2 [Anupam Kher, Akshaye Khanna, Aahana Kumra, At...
3 [Emraan Hashmi, Shreya Dhanwanthary, Snighdade...
4 [Mona Ambegaonkar, Ananth Narayan Mahadevan, D...
...
1624 [Ajay Devgn, Sonali Bendre, Namrata Shirodkar,...
1625 [Ameesha Patel, Jimmy Sheirgill, Nafisa Ali, G...
1626 [Vijay Arora, Asrani, Rajni Bala, Kumud Damle,...
1627 [Gippy Grewal, Zareen Khan, ]
1628 [Fawad Khan, ]
Name: actors, Length: 1629, dtype: object
```

In [127]:

```

1 movies['actors'].str.split('|').apply(lambda x:x[0])
2 '''here this code will show an error because
3 there are some missing values in actors column
4
5 Pandas treat missing values as float values
6 so the logic we wrote in the code will throw an error
7
8 so here first we have to remove the missing value'''

```

```

-----
-----
TypeError                                Traceback (most recent call 1
ast)
~\AppData\Local\Temp\ipykernel_4404\3178307672.py in <module>
----> 1 movies['actors'].str.split('|').apply(lambda x:x[0])
      2 '''here this code will show an error because
      3 there are some missing values in actors column
      4
      5 Pandas treat missing values as float values

~\anaconda3\lib\site-packages\pandas\core\series.py in apply(self, fun
c, convert_dtype, args, **kwargs)
    4431         dtype: float64
    4432         """
-> 4433         return SeriesApply(self, func, convert_dtype, args, kwa
rgs).apply()
    4434
    4435         def _reduce(

```

In [128]:

```

1 movies.dropna(inplace=True)

```

In [129]:

```

1 movies.info()

```

```

<class 'pandas.core.frame.DataFrame'>
Int64Index: 298 entries, 11 to 1623
Data columns (total 19 columns):
 #   Column                Non-Null Count  Dtype
---  -
 0   title_x               298 non-null   object
 1   imdb_id               298 non-null   object
 2   poster_path           298 non-null   object
 3   wiki_link             298 non-null   object
 4   title_y               298 non-null   object
 5   original_title        298 non-null   object
 6   is_adult              298 non-null   int64
 7   year_of_release       298 non-null   int64
 8   runtime               298 non-null   object
 9   genres                298 non-null   object
10  imdb_rating           298 non-null   float64
11  imdb_votes            298 non-null   int64
12  story                 298 non-null   object
13  summary               298 non-null   object
..   ..                  ..          ..

```


In [130]:

```
1 movies['actors'].str.split('|').apply(lambda x:x[0])
```

Out[130]:

```
11          Ranveer Singh
34          Gavie Chahal
37      Ayushmann Khurrana
87          Sidharth Malhotra
96          Ajay Devgn
...
1600         Divya Dutta
1601         Anant Nag
1607         Anil Kapoor
1621     Priyanshu Chatterjee
1623         Karisma Kapoor
Name: actors, Length: 298, dtype: object
```

In [131]:

```
1 movies['lead actor'] = movies['actors'].str.split('|').apply(lambda x:x[0])
```

In [132]:

```
1 movies.head()
2 # lead actors column is added in the last in the data frame
```

Out[132]:

	title_x	imdb_id	poster_path	wiki_lin
11	Gully Boy	tt2395469	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Gully_Bc
34	Yeh Hai India	tt5525846	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Yeh_Hai_Ind
37	Article 15 (film)	tt10324144	https://upload.wikimedia.org/wikipedia/en/thum...	https://en.wikipedia.org/wiki/Article_15_(film)

Important DataFrame Functions

astype

- it changes the data type of the given column

In [133]:

```
1 ipl.info()
2 # here memory occupied by ipl dataset is 148.6 kb
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 950 entries, 0 to 949
Data columns (total 20 columns):
 #   Column                Non-Null Count  Dtype  
---  --
 0   ID                    950 non-null   int64  
 1   City                  899 non-null   object  
 2   Date                  950 non-null   object  
 3   Season                950 non-null   object  
 4   MatchNumber           950 non-null   object  
 5   Team1                 950 non-null   object  
 6   Team2                 950 non-null   object  
 7   Venue                 950 non-null   object  
 8   TossWinner            950 non-null   object  
 9   TossDecision          950 non-null   object  
10   SuperOver             946 non-null   object  
11   WinningTeam           946 non-null   object  
12   WonBy                 950 non-null   object  
13   Margin                932 non-null   float64  
..   ..
```

In [134]:

```
1 ipl['ID'].astype('int32')
2 # we changed the data type of ID column to int32
```

Out[134]:

```
0      1312200
1      1312199
2      1312198
3      1312197
4      1304116
...
945     335986
946     335985
947     335984
948     335983
949     335982
Name: ID, Length: 950, dtype: int32
```

In [135]:

```
1 ipl['ID'] = ipl['ID'].astype('int32')
```

In [136]:

```
1 ipl.info()
2 # now memory occupied by ipl dataset is 144.9 kb
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 950 entries, 0 to 949
Data columns (total 20 columns):
#   Column                Non-Null Count  Dtype
---  -
0    ID                    950 non-null   int32
1    City                  899 non-null   object
2    Date                  950 non-null   object
3    Season                950 non-null   object
4    MatchNumber           950 non-null   object
5    Team1                 950 non-null   object
6    Team2                 950 non-null   object
7    Venue                 950 non-null   object
8    TossWinner            950 non-null   object
9    TossDecision          950 non-null   object
10   SuperOver             946 non-null   object
11   WinningTeam           946 non-null   object
12   WonBy                 950 non-null   object
13   Margin                932 non-null   float64
```

In [137]:

```
1 # we will convert season datatype to category data type
2 # this will further reduce the occupied memory size
3
4 ipl['Season'] = ipl['Season'].astype('category')
```

In [139]:

```
1 ipl.info()
2 # here memory occupied is 139.0 KB
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 950 entries, 0 to 949
Data columns (total 20 columns):
#   Column                Non-Null Count  Dtype
---  -
0    ID                    950 non-null   int32
1    City                  899 non-null   object
2    Date                  950 non-null   object
3    Season                950 non-null   category
4    MatchNumber           950 non-null   object
5    Team1                 950 non-null   object
6    Team2                 950 non-null   object
7    Venue                 950 non-null   object
8    TossWinner            950 non-null   object
9    TossDecision          950 non-null   object
10   SuperOver             946 non-null   object
11   WinningTeam           946 non-null   object
12   WonBy                 950 non-null   object
13   Margin                932 non-null   float64
```

