Level 1 . DataFrame Basics

Data Frames

In [45]:
```
# import python package
import pandas as pd
import numpy as np

# read data from file
df = pd.read_csv("weather_data.csv")

# display data in data frame
df
```

Out[45]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain
5	1/6/2017	31	2	Sunny

In [2]:
```
#create a dictionary with shape 3 x 5
weather_data = {
    'day': ['1/1/2017','1/2/2017','1/3/2017','1/4/2017','1/5/2017','1/6/2017'
    'temperature': [32,35,28,24,32,31],
    'windspeed': [6,7,2,7,4,2],
    'event': ['Rain', 'Sunny', 'Snow','Snow','Rain', 'Sunny']
}



#load dictionary into data frame
df = pd.DataFrame(weather_data)


# display data frame
df
```

Out[2]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain
5	1/6/2017	31	2	Sunny

In [3]:
```
# print no of rows and columns of data frame
df.shape
```

Out[3]:
```
(6, 4)
```

Rows

In [4]:
```
# show first 5 records
df.head()
```

Out[4]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain

In [ ]:
```

```

In [5]:
```
# show first 3 records
df.head(3)
```

Out[5]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow

In [6]:
```
#show last 5 records
df.tail(5)
```

Out[6]:

	day	temperature	windspeed	event
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain
5	1/6/2017	31	2	Sunny

In [7]:
```
#show last 3 records
df.tail(3)
```

Out[7]:

	day	temperature	windspeed	event
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain
5	1/6/2017	31	2	Sunny

In [8]:
```
# show record no 3 to 8
df[3:8]
```

Out[8]:

	day	temperature	windspeed	event
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain
5	1/6/2017	31	2	Sunny

Columns

In [9]:
```
# print all column names
df.columns
```

Out[9]:
```
Index(['day', 'temperature', 'windspeed', 'event'], dtype='object')
```

In [10]:
```
# print entire data of one column
df['day']
```

Out[10]:
```
0    1/1/2017
1    1/2/2017
2    1/3/2017
3    1/4/2017
4    1/5/2017
5    1/6/2017
Name: day, dtype: object
```

In [11]:
```
#show type of one column
type(df['day'])
```

Out[11]:
```
pandas.core.series.Series
```

In [12]:
```
# print only two columns from a data frame
df[['day','temperature']]
```

Out[12]:

	day	temperature
0	1/1/2017	32
1	1/2/2017	35
2	1/3/2017	28
3	1/4/2017	24
4	1/5/2017	32
5	1/6/2017	31

Operations On DataFrame

In [13]:
```
# find maximum value in a column
df['temperature'].max()
```

Out[13]:
```
35
```

In [14]:
```
# print all data where gpa is greater than 3
#   or
# print all days when temperature was less than 35
df[df['temperature']>32]
```

Out[14]:

	day	temperature	windspeed	event
1	1/2/2017	35	7	Sunny

In [15]:
```
# find standard deviation of temperature
df['temperature'].std()
```

Out[15]:
```
3.8297084310253524
```

In [16]:
```
# find which event is maximum
df['event'].max()
```

Out[16]:
```
'Sunny'
```

In [17]:
```
# describe statistical report of data frame
df.describe()
```

Out[17]:

	temperature	windspeed
count	6.000000	6.000000
mean	30.333333	4.666667
std	3.829708	2.338090
min	24.000000	2.000000
25%	28.750000	2.500000
50%	31.500000	5.000000
75%	32.000000	6.750000
max	35.000000	7.000000

set_index

In [18]:
```
# set a new index of data frame
df.set_index('day')
```

Out[18]:

	temperature	windspeed	event
day			
1/1/2017	32	6	Rain
1/2/2017	35	7	Sunny
1/3/2017	28	2	Snow
1/4/2017	24	7	Snow
1/5/2017	32	4	Rain
1/6/2017	31	2	Sunny

In [19]:
```
# set day as its original index
df.set_index('day', inplace=True)
```

In [20]:
```
# print all indexis
df.index
```

Out[20]:
```
Index(['1/1/2017', '1/2/2017', '1/3/2017', '1/4/2017', '1/5/2017', '1/6/201
7'], dtype='object', name='day')
```

In [21]:
```
# print all record of one index
df.loc['1/2/2017']
```

Out[21]:
```
temperature       35
windspeed          7
event          Sunny
Name: 1/2/2017, dtype: object
```

In [22]:
```
# reset its original index
df.reset_index(inplace=True)
df.head()
```

Out[22]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain

In [23]:
```
# set event as original index by inplacing
df.set_index('event',inplace=True) # this is kind of building a hash map usir
df
```

Out[23]:

event	day	temperature	windspeed
Rain	1/1/2017	32	6
Sunny	1/2/2017	35	7
Snow	1/3/2017	28	2
Snow	1/4/2017	24	7
Rain	1/5/2017	32	4
Sunny	1/6/2017	31	2

In [24]:
```
# print information of all snowy days
df.loc['Snow']
```

Out[24]:

event	day	temperature	windspeed
Snow	1/3/2017	28	2
Snow	1/4/2017	24	7

Level 2 . DataFrame Construction

Using csv

In [ ]:
```
df = pd.read_csv("weather_data.csv")
df
```

Using excel

In [26]:
```
df=pd.read_excel("weather_data.xlsx","Sheet1")
df
```

Out[26]:

	day	temperature	windspeed	event
0	2017-01-01	32	6	Rain
1	2017-01-02	35	7	Sunny
2	2017-01-03	28	2	Snow

Using dictionary

In [27]:
```
import pandas as pd
weather_data = {
    'day': ['1/1/2017','1/2/2017','1/3/2017'],
    'temperature': [32,35,28],
    'windspeed': [6,7,2],
    'event': ['Rain', 'Sunny', 'Snow']
}
df = pd.DataFrame(weather_data)
df
```

Out[27]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow

Using tuples list

In [28]:
```
weather_data = [
    ('1/1/2017',32,6,'Rain'),
    ('1/2/2017',35,7,'Sunny'),
    ('1/3/2017',28,2,'Snow')
]
df = pd.DataFrame(data=weather_data, columns=['day','temperature','windspeed'
df
```

Out[28]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow

Using list of dictionaries

In [29]:
```
weather_data = [
    {'day': '1/1/2017', 'temperature': 32, 'windspeed': 6, 'event': 'Rain'},
    {'day': '1/2/2017', 'temperature': 35, 'windspeed': 7, 'event': 'Sunny'},
    {'day': '1/3/2017', 'temperature': 28, 'windspeed': 2, 'event': 'Snow'},

]
df = pd.DataFrame(data=weather_data, columns=['day','temperature','windspeed'
df
```

Out[29]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow

Level 3 . Handling Missing Data

In [30]:
```
# read weather data from csv file and convert date column into date datatype

# set day as index and print dataframe
import pandas as pd
df = pd.read_csv("weather_missing_data.csv",parse_dates=['day'])
type(df.day[0])
df
```

Out[30]:

	day	temperature	windspeed	event
0	2017-01-01	32.0	6.0	Rain
1	2017-01-04	NaN	9.0	Sunny
2	2017-01-05	28.0	NaN	Snow
3	2017-01-06	NaN	7.0	NaN
4	2017-01-07	32.0	NaN	Rain
5	2017-01-08	NaN	NaN	Sunny
6	2017-01-09	NaN	NaN	NaN
7	2017-01-10	34.0	8.0	Cloudy
8	2017-01-11	40.0	12.0	Sunny

fillna

In [31]:
```
# fill all NaN with 0 value
new_df = df.fillna(0)
new_df
```

Out[31]:

	day	temperature	windspeed	event
0	2017-01-01	32.0	6.0	Rain
1	2017-01-04	0.0	9.0	Sunny
2	2017-01-05	28.0	0.0	Snow
3	2017-01-06	0.0	7.0	0
4	2017-01-07	32.0	0.0	Rain
5	2017-01-08	0.0	0.0	Sunny
6	2017-01-09	0.0	0.0	0
7	2017-01-10	34.0	8.0	Cloudy
8	2017-01-11	40.0	12.0	Sunny

In [32]:
```
# fill temp and ws with 0 and event column with 'no event'  using dictionary
new_df = df.fillna({
        'temperature': 0,
        'windspeed': 0,
        'event': 'No Event'
    })
new_df
```

Out[32]:

	day	temperature	windspeed	event
0	2017-01-01	32.0	6.0	Rain
1	2017-01-04	0.0	9.0	Sunny
2	2017-01-05	28.0	0.0	Snow
3	2017-01-06	0.0	7.0	No Event
4	2017-01-07	32.0	0.0	Rain
5	2017-01-08	0.0	0.0	Sunny
6	2017-01-09	0.0	0.0	No Event
7	2017-01-10	34.0	8.0	Cloudy
8	2017-01-11	40.0	12.0	Sunny

In [33]:
```
# fill temp and windspeed with previous entry
new_df = df.fillna(method="ffill")
new_df
```

Out[33]:

	day	temperature	windspeed	event
0	2017-01-01	32.0	6.0	Rain
1	2017-01-04	32.0	9.0	Sunny
2	2017-01-05	28.0	9.0	Snow
3	2017-01-06	28.0	7.0	Snow
4	2017-01-07	32.0	7.0	Rain
5	2017-01-08	32.0	7.0	Sunny
6	2017-01-09	32.0	7.0	Sunny
7	2017-01-10	34.0	8.0	Cloudy
8	2017-01-11	40.0	12.0	Sunny

In [34]:
```
# fill temp and windspeed with next value using back method
new_df = df.fillna(method="bfill")
new_df
```

Out[34]:

	day	temperature	windspeed	event
0	2017-01-01	32.0	6.0	Rain
1	2017-01-04	28.0	9.0	Sunny
2	2017-01-05	28.0	7.0	Snow
3	2017-01-06	32.0	7.0	Rain
4	2017-01-07	32.0	8.0	Rain
5	2017-01-08	34.0	8.0	Sunny
6	2017-01-09	34.0	8.0	Cloudy
7	2017-01-10	34.0	8.0	Cloudy
8	2017-01-11	40.0	12.0	Sunny

In [35]:
```
# fillna is filling all null value, set a constrain with 1, limit 1
new_df = df.fillna(method="bfill", axis="columns") # axis is either "index" c
new_df
```

Out[35]:

	day	temperature	windspeed	event
0	2017-01-01 00:00:00	32	6	Rain
1	2017-01-04 00:00:00	9	9	Sunny
2	2017-01-05 00:00:00	28	Snow	Snow
3	2017-01-06 00:00:00	7	7	NaN
4	2017-01-07 00:00:00	32	Rain	Rain
5	2017-01-08 00:00:00	Sunny	Sunny	Sunny
6	2017-01-09 00:00:00	NaN	NaN	NaN
7	2017-01-10 00:00:00	34	8	Cloudy
8	2017-01-11 00:00:00	40	12	Sunny

Interpolate

In [36]:
```
# interpolate all missing values
new_df = df.interpolate()
new_df
```

Out[36]:

	day	temperature	windspeed	event
0	2017-01-01	32.000000	6.00	Rain
1	2017-01-04	30.000000	9.00	Sunny
2	2017-01-05	28.000000	8.00	Snow
3	2017-01-06	30.000000	7.00	NaN
4	2017-01-07	32.000000	7.25	Rain
5	2017-01-08	32.666667	7.50	Sunny
6	2017-01-09	33.333333	7.75	NaN
7	2017-01-10	34.000000	8.00	Cloudy
8	2017-01-11	40.000000	12.00	Sunny

dropna

In [38]:
```
# drop all those indexis where even a single entry is NaN
new_df = df.dropna()
new_df
```

Out[38]:

	day	temperature	windspeed	event
0	2017-01-01	32.0	6.0	Rain
7	2017-01-10	34.0	8.0	Cloudy
8	2017-01-11	40.0	12.0	Sunny

In [39]:
```
# drop only those indexis where all entries all NaN
new_df = df.dropna(how='all')
new_df
```

Out[39]:

	day	temperature	windspeed	event
0	2017-01-01	32.0	6.0	Rain
1	2017-01-04	NaN	9.0	Sunny
2	2017-01-05	28.0	NaN	Snow
3	2017-01-06	NaN	7.0	NaN
4	2017-01-07	32.0	NaN	Rain
5	2017-01-08	NaN	NaN	Sunny
6	2017-01-09	NaN	NaN	NaN
7	2017-01-10	34.0	8.0	Cloudy
8	2017-01-11	40.0	12.0	Sunny

In [40]:
```
# set a threshold , as 1
new_df = df.dropna(thresh=1)
new_df
```

Out[40]:

	day	temperature	windspeed	event
0	2017-01-01	32.0	6.0	Rain
1	2017-01-04	NaN	9.0	Sunny
2	2017-01-05	28.0	NaN	Snow
3	2017-01-06	NaN	7.0	NaN
4	2017-01-07	32.0	NaN	Rain
5	2017-01-08	NaN	NaN	Sunny
6	2017-01-09	NaN	NaN	NaN
7	2017-01-10	34.0	8.0	Cloudy
8	2017-01-11	40.0	12.0	Sunny

Inserting Missing Dates

In [41]:
```
# insert all mising dates in index
dt = pd.date_range("01-01-2017","01-11-2017")
idx = pd.DatetimeIndex(dt)
df.reindex(idx)
```

Out[41]:

	day	temperature	windspeed	event
2017-01-01	NaT	NaN	NaN	NaN
2017-01-02	NaT	NaN	NaN	NaN
2017-01-03	NaT	NaN	NaN	NaN
2017-01-04	NaT	NaN	NaN	NaN
2017-01-05	NaT	NaN	NaN	NaN
2017-01-06	NaT	NaN	NaN	NaN
2017-01-07	NaT	NaN	NaN	NaN
2017-01-08	NaT	NaN	NaN	NaN
2017-01-09	NaT	NaN	NaN	NaN
2017-01-10	NaT	NaN	NaN	NaN
2017-01-11	NaT	NaN	NaN	NaN

Level 4 . Replace Missing Data

Replace (value, value)

In [42]:
```
import pandas as pd

#replace any single value with NaN -- e.g -99999 as NaN
data = pd.read_csv('replace_weather_data.csv')
data.replace(-99999, 0, inplace=True)
data
```

Out[42]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	0	7	Sunny
2	1/3/2017	28	0	Snow
3	1/4/2017	0	7	0
4	1/5/2017	32	0	Rain
5	1/6/2017	31	2	Sunny
6	1/6/2017	34	5	0

Replace ([value , value], value)

In [43]:
```
# replace two values with 0
data
new_df = df.replace(to_replace=[-99999,-88888], value=0)
new_df
```

Out[43]:

	day	temperature	windspeed	event
0	2017-01-01	32.0	6.0	Rain
1	2017-01-04	NaN	9.0	Sunny
2	2017-01-05	28.0	NaN	Snow
3	2017-01-06	NaN	7.0	NaN
4	2017-01-07	32.0	NaN	Rain
5	2017-01-08	NaN	NaN	Sunny
6	2017-01-09	NaN	NaN	NaN
7	2017-01-10	34.0	8.0	Cloudy
8	2017-01-11	40.0	12.0	Sunny

Replace (dic{lable:value, label:value} , Rep_value)

In [46]:
```
# replace value by giving column name -- hind dictionary
new_df = df.replace({
        -99999: np.nan,
        'no event': 'Sunny',
    })
new_df
```

Out[46]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain
5	1/6/2017	31	2	Sunny

Replace (dic{label:Rep_value, label:Rep_value)

In [47]:
```
# replace by using maping.
import numpy as np

data = data.replace({
    'NaN':'no event',
})

data
```

Out[47]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	0	7	Sunny
2	1/3/2017	28	0	Snow
3	1/4/2017	0	7	0
4	1/5/2017	32	0	Rain
5	1/6/2017	31	2	Sunny
6	1/6/2017	34	5	0

Replace (dic{value_with_regex, value_with_regex, }, Rep_value)

In [48]:
```
# when windspeed is 6 mph, 7 mph etc. & temperature is 32 F, 28 F etc.
# remove mph, F, C etc using Regex
new_df = df.replace({'temperature': '[A-Za-z]', 'windspeed': '[a-z]'},'', reg
new_df
```

Out[48]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain
5	1/6/2017	31	2	Sunny

Replace (list[value,value,value] , [Rep_value,Rep_value,Rep_value])

In [49]:
```
# replacing list with another list -- hind : students record
df = pd.DataFrame({
    'score': ['exceptional','average', 'good', 'poor', 'average', 'exceptiona
    'student': ['rob', 'maya', 'parthiv', 'tom', 'julian', 'erica']
})
df
```

Out[49]:

	score	student
0	exceptional	rob
1	average	maya
2	good	parthiv
3	poor	tom
4	average	julian
5	exceptional	erica

Level 5 . Concatination

Replace ([value , value], value)

In [51]:
```
import pandas as pd

Inside_weather = pd.DataFrame({
    "city": ["Lahore","Karachi","Islamabad"],
    "temperature": [32,45,30],
    "humidity": [80, 60, 78]
})
Inside_weather
```

Out[51]:

	city	temperature	humidity
0	Lahore	32	80
1	Karachi	45	60
2	Islamabad	30	78

create a dataframe

In [52]:
```
us_weather = pd.DataFrame({
    "city": ["new york","chicago","orlando"],
    "temperature": [21,14,35],
    "humidity": [68, 65, 75]
})
us_weather
```

Out[52]:

	city	temperature	humidity
0	new york	21	68
1	chicago	14	65
2	orlando	35	75

concate two data frames

In [53]:
```
df = pd.concat([Inside_weather, us_weather])
df
```

Out[53]:

	city	temperature	humidity
0	Lahore	32	80
1	Karachi	45	60
2	Islamabad	30	78
0	new york	21	68
1	chicago	14	65
2	orlando	35	75

re arrange indexis

In [54]:
```
df = pd.concat([Inside_weather, us_weather], ignore_index=True)
df
```

Out[54]:

	city	temperature	humidity
0	Lahore	32	80
1	Karachi	45	60
2	Islamabad	30	78
3	new york	21	68
4	chicago	14	65
5	orlando	35	75

concate and distint by using their keys

In [55]:
```
df = pd.concat([Inside_weather, us_weather], keys=["Pakistan", "us"])
df
```

Out[55]:

		city	temperature	humidity
	0	Lahore	32	80
Pakistan	**1**	Karachi	45	60
	2	Islamabad	30	78
	0	new york	21	68
us	**1**	chicago	14	65
	2	orlando	35	75

print only us data

In [56]:
```
df.loc["us"]
```

Out[56]:

	city	temperature	humidity
0	new york	21	68
1	chicago	14	65
2	orlando	35	75

print only india data

In [57]:
```
df.loc["Pakistan"]
```

Out[57]:

	city	temperature	humidity
0	Lahore	32	80
1	Karachi	45	60
2	Islamabad	30	78

re arange indexis of india

In [58]:
```
temperature_df = pd.DataFrame({
    "city": ["lahore","karachi","islamabad"],
    "temperature": [32,45,30],
}, index=[0,1,2])
temperature_df
```

Out[58]:

	city	temperature
0	lahore	32
1	karachi	45
2	islamabad	30

Replace ([value , value], value)

In [59]:
```
windspeed_df = pd.DataFrame({
    "city": ["lahore","karachi"],
    "windspeed": [7,12],
}, index=[1,0])
windspeed_df
```

Out[59]:

	city	windspeed
1	lahore	7
0	karachi	12

Replace ([value , value], value)

In [60]:
```
df = pd.concat([temperature_df,windspeed_df],axis=1)
df
```

Out[60]:

	city	temperature	city	windspeed
0	lahore	32	karachi	12.0
1	karachi	45	lahore	7.0
2	islamabad	30	NaN	NaN

Replace ([value , value], value)

In [61]:
```
s = pd.Series(["Humid","Dry","Rain"], name="event")
s
```

Out[61]:
```
0    Humid
1      Dry
2     Rain
Name: event, dtype: object
```

In [62]:
```
df = pd.concat([temperature_df,s],axis=1)
df
```

Out[62]:

	city	temperature	event
0	lahore	32	Humid
1	karachi	45	Dry
2	islamabad	30	Rain

Level 6 . Merging Dataframes

create a new data frame using dictionary

In [63]:
```
import pandas as pd
df1 = pd.DataFrame({
    "city": ["new york","chicago","orlando"],
    "temperature": [21,14,35],
})
df1
```

Out[63]:

	city	temperature
0	new york	21
1	chicago	14
2	orlando	35

create a new data frame using dictionary

In [64]:
```
df2 = pd.DataFrame({
    "city": ["chicago","new york","orlando"],
    "humidity": [65,68,75],
})
df2
```

Out[64]:

	city	humidity
0	chicago	65
1	new york	68
2	orlando	75

Merge two data frames

In [65]:
```
df3 = pd.merge(df1, df2, on="city")
df3
```

Out[65]:

	city	temperature	humidity
0	new york	21	68
1	chicago	14	65
2	orlando	35	75

In [ ]:
```

```

In [66]:
```
df1 = pd.DataFrame({
    "city": ["new york","chicago","orlando", "baltimore"],
    "temperature": [21,14,35, 38],
})
df1
```

Out[66]:

	city	temperature
0	new york	21
1	chicago	14
2	orlando	35
3	baltimore	38

In [67]:
```
df2 = pd.DataFrame({
    "city": ["chicago","new york","san diego"],
    "humidity": [65,68,71],
})
df2
```

Out[67]:

	city	humidity
0	chicago	65
1	new york	68
2	san diego	71

In [68]:
```
df3=pd.merge(df1,df2,on="city",how="inner")
df3
```

Out[68]:

	city	temperature	humidity
0	new york	21	68
1	chicago	14	65

In [69]:
```
df3=pd.merge(df1,df2,on="city",how="outer")
df3
```

Out[69]:

	city	temperature	humidity
0	new york	21.0	68.0
1	chicago	14.0	65.0
2	orlando	35.0	NaN
3	baltimore	38.0	NaN
4	san diego	NaN	71.0

In [70]:
```
df3=pd.merge(df1,df2,on="city",how="left")
df3
```

Out[70]:

	city	temperature	humidity
0	new york	21	68.0
1	chicago	14	65.0
2	orlando	35	NaN
3	baltimore	38	NaN

In [71]:
```
df3=pd.merge(df1,df2,on="city",how="right")
df3
```

Out[71]:

	city	temperature	humidity
0	new york	21.0	68
1	chicago	14.0	65
2	san diego	NaN	71

In [72]:
```
df3=pd.merge(df1,df2,on="city",how="outer",indicator=True)
df3
```

Out[72]:

	city	temperature	humidity	_merge
0	new york	21.0	68.0	both
1	chicago	14.0	65.0	both
2	orlando	35.0	NaN	left_only
3	baltimore	38.0	NaN	left_only
4	san diego	NaN	71.0	right_only

In [73]:
```
df1 = pd.DataFrame({
    "city": ["new york","chicago","orlando", "baltimore"],
    "temperature": [21,14,35,38],
    "humidity": [65,68,71, 75]
})
df1
```

Out[73]:

	city	temperature	humidity
0	new york	21	65
1	chicago	14	68
2	orlando	35	71
3	baltimore	38	75

In [74]:
```
df2 = pd.DataFrame({
    "city": ["chicago","new york","san diego"],
    "temperature": [21,14,35],
    "humidity": [65,68,71]
})
df2
```

Out[74]:

	city	temperature	humidity
0	chicago	21	65
1	new york	14	68
2	san diego	35	71

In [75]:
```
df3= pd.merge(df1,df2,on="city",how="outer", suffixes=('_first','_second'))
df3
```

Out[75]:

	city	temperature_first	humidity_first	temperature_second	humidity_second
0	new york	21.0	65.0	14.0	68.0
1	chicago	14.0	68.0	21.0	65.0
2	orlando	35.0	71.0	NaN	NaN
3	baltimore	38.0	75.0	NaN	NaN
4	san diego	NaN	NaN	35.0	71.0

In [76]:
```
df1 = pd.DataFrame({
    "city": ["new york","chicago","orlando"],
    "temperature": [21,14,35],
})
df1.set_index('city',inplace=True)
df1
```

Out[76]:

	temperature
city	
new york	21
chicago	14
orlando	35

In [77]:
```
df1 = pd.DataFrame({
    "city": ["new york","chicago","orlando"],
    "temperature": [21,14,35],
})
df1.set_index('city',inplace=True)
df1
```

Out[77]:

city	temperature
new york	21
chicago	14
orlando	35

In [78]:
```
df1.join(df2,lsuffix='_l', rsuffix='_r')
```

Out[78]:

city	temperature_l	city	temperature_r	humidity
new york	21	NaN	NaN	NaN
chicago	14	NaN	NaN	NaN
orlando	35	NaN	NaN	NaN

---------- Good Luck for Mids ----------