**6.Pandas Joining and merging DataFrame:**

**i)Write a Pandas program to join the two given dataframes along rows and assign all data**.

**Program:**

import pandas as pd

student\_data1 = pd.DataFrame({

'student\_id': ['S1', 'S2', 'S3', 'S4', 'S5'],

'name': ['Danniella Fenton', 'Ryder Storey', 'Bryce Jensen', 'Ed Bernal', 'Kwame Morin'],

'marks': [200, 210, 190, 222, 199]})

student\_data2 = pd.DataFrame({

'student\_id': ['S4', 'S5', 'S6', 'S7', 'S8'],

'name': ['Scarlette Fisher', 'Carla Williamson', 'Dante Morse', 'Kaiser William', 'Madeeha Preston'],

'marks': [201, 200, 198, 219, 201]})

print("Original DataFrames:")

print(student\_data1)

print("-------------------------------------")

print(student\_data2)

print("\nJoin the said two dataframes along rows:")

result\_data = pd.concat([student\_data1, student\_data2])

print(result\_data)

**Output:**

Original DataFrames:

student\_id name marks

0 S1 Danniella Fenton 200

1 S2 Ryder Storey 210

2 S3 Bryce Jensen 190

3 S4 Ed Bernal 222

4 S5 Kwame Morin 199

-------------------------------------

student\_id name marks

0 S4 Scarlette Fisher 201

1 S5 Carla Williamson 200

2 S6 Dante Morse 198

3 S7 Kaiser William 219

4 S8 Madeeha Preston 201

Join the said two dataframes along rows:

student\_id name marks

0 S1 Danniella Fenton 200

1 S2 Ryder Storey 210

2 S3 Bryce Jensen 190

3 S4 Ed Bernal 222

4 S5 Kwame Morin 199

0 S4 Scarlette Fisher 201

1 S5 Carla Williamson 200

2 S6 Dante Morse 198

3 S7 Kaiser William 219

4 S8 Madeeha Preston 201

**ii)Write a Pandas program to append a list of dictioneries or series to a existing DataFrame and display the combined data**

**Pogram:**

import pandas as pd

student\_data1 = pd.DataFrame({

'student\_id': ['S1', 'S2', 'S3', 'S4', 'S5'],

'name': ['Danniella Fenton', 'Ryder Storey', 'Bryce Jensen', 'Ed Bernal', 'Kwame Morin'],

'marks': [200, 210, 190, 222, 199]})

s6 = pd.Series(['S6', 'Scarlette Fisher', 205], index=['student\_id', 'name', 'marks'])

dicts = [{'student\_id': 'S6', 'name': 'Scarlette Fisher', 'marks': 203},

{'student\_id': 'S7', 'name': 'Bryce Jensen', 'marks': 207}]

print("Original DataFrames:")

print(student\_data1)

print("\nDictionary:")

print(s6)

combined\_data = student\_data1.append(dicts, ignore\_index=True, sort=False)

print("\nCombined Data:")

print(combined\_data)

**Output:**

Original DataFrames:

student\_id name marks

0 S1 Danniella Fenton 200

1 S2 Ryder Storey 210

2 S3 Bryce Jensen 190

3 S4 Ed Bernal 222

4 S5 Kwame Morin 199

Dictionary:

student\_id S6

name Scarlette Fisher

marks 205

dtype: object

Combined Data:

student\_id name marks

0 S1 Danniella Fenton 200

1 S2 Ryder Storey 210

2 S3 Bryce Jensen 190

3 S4 Ed Bernal 222

4 S5 Kwame Morin 199

5 S6 Scarlette Fisher 203

6 S7 Bryce Jensen 207

**iii)Write a Pandas program to join the two dataframes with matching records from both sides where available.**

**Program:**

import pandas as pd

student\_data1 = pd.DataFrame({

'student\_id': ['S1', 'S2', 'S3', 'S4', 'S5'],

'name': ['Danniella Fenton', 'Ryder Storey', 'Bryce Jensen', 'Ed Bernal', 'Kwame Morin'],

'marks': [200, 210, 190, 222, 199]})

student\_data2 = pd.DataFrame({

'student\_id': ['S4', 'S5', 'S6', 'S7', 'S8'],

'name': ['Scarlette Fisher', 'Carla Williamson', 'Dante Morse', 'Kaiser William', 'Madeeha Preston'],

'marks': [201, 200, 198, 219, 201]})

print("Original DataFrames:")

print(student\_data1)

print(student\_data2)

merged\_data = pd.merge(student\_data1, student\_data2, on='student\_id', how='outer')

print("Merged data (outer join):")

print(merged\_data)

**Output:**

Original DataFrames:

student\_id name marks

0 S1 Danniella Fenton 200

1 S2 Ryder Storey 210

2 S3 Bryce Jensen 190

3 S4 Ed Bernal 222

4 S5 Kwame Morin 199

student\_id name marks

0 S4 Scarlette Fisher 201

1 S5 Carla Williamson 200

2 S6 Dante Morse 198

3 S7 Kaiser William 219

4 S8 Madeeha Preston 201

Merged data (outer join):

student\_id name\_x marks\_x name\_y marks\_y

0 S1 Danniella Fenton 200.0 NaN NaN

1 S2 Ryder Storey 210.0 NaN NaN

2 S3 Bryce Jensen 190.0 NaN NaN

3 S4 Ed Bernal 222.0 Scarlette Fisher 201.0

4 S5 Kwame Morin 199.0 Carla Williamson 200.0

5 S6 NaN NaN Dante Morse 198.0

6 S7 NaN NaN Kaiser William 219.0

7 S8 NaN NaN Madeeha Preston 201.0

**7. Write a NumPy program to find the number of elements of an array, length of one array element in bytes and total bytes consumed by the elements.**

**Program:**

import numpy as np

x = np.array([1,2,3], dtype=np.float64)

print("Size of the array: ", x.size)

print("Length of one array element in bytes: ", x.itemsize)

print("Total bytes consumed by the elements of the array: ", x.nbytes)

**Output:**

Size of the array: 3

Length of one array element in bytes: 8

Total bytes consumed by the elements of the array: 24

**8.Write a Pandas program to create  
a) Datetime object for Jan 15 2012.  
b) Specific date and time of 9:20 pm.  
c) Local date and time.  
d) A date without time.  
e) Current date.  
f) Time from a datetime.  
g) Current local time.**

**Program:**

import datetime

from datetime import datetime

print("Datetime object for Jan 11 2012:")

print(datetime(2012, 1, 11))

print("\nSpecific date and time of 9:20 pm")

print(datetime(2011, 1, 11, 21, 20))

print("\nLocal date and time:")

print(datetime.now())

print("\nA date without time: ")

print(datetime.date(datetime(2012, 5, 22)))

print("\nCurrent date:")

print(datetime.now().date())

print("\nTime from a datetime:")

print(datetime.time(datetime(2012, 12, 15, 18, 12)))

print("\nCurrent local time:")

print(datetime.now().time())

**Output:**

Datetime object for Jan 11 2012:

2012-01-11 00:00:00

Specific date and time of 9:20 pm

2011-01-11 21:20:00

Local date and time:

2020-08-17 09:56:17.459790

A date without time:

2012-05-22

Current date:

2020-08-17

Time from a datetime:

18:12:00

Current local time:

09:56:17.461250

**EXPERIMENT-9:**

**i)Write a Pandas program to create a date from a given year, month, day and another date from a given string formats.**

**Program:**

from datetime import datetime

date1 = datetime(year=2020, month=12, day=25)

print("Date from a given year, month, day:")

print(date1)

from dateutil import parser

date2 = parser.parse("1st of January, 2021")

print("\nDate from a given string formats:")

print(date2)

**Output:**

Date from a given year, month, day:

2020-12-25 00:00:00

Date from a given string formats:

2021-01-01 00:00:00

**ii)Write a Pandas program to create a time-series with two index labels and random values. Also print the type of the index.**

**Program:**

import pandas as pd

import numpy as np

import datetime

from datetime import datetime, date

dates = [datetime(2011, 9, 1), datetime(2011, 9, 2)]

print("Time-series with two index labels:")

time\_series = pd.Series(np.random.randn(2), dates)

print(time\_series)

print("\nType of the index:")

print(type(time\_series.index))

**Output:**

Time-series with two index labels:

2011-09-01 -0.257567

2011-09-02 0.947341

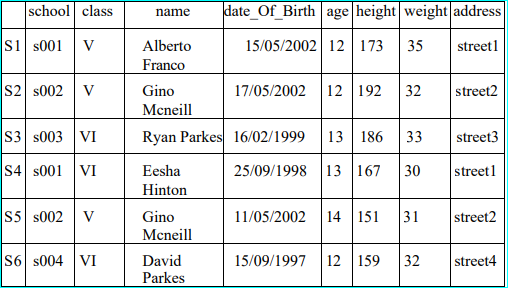
dtype: float64

Type of the index:

<class 'pandas.core.indexes.datetimes.DatetimeIndex'>

**10. Pandas Grouping Aggregate:**

**Consider dataset:**



**i) Write a Pandas program to split the following dataframe into groups based on school code. Also check the type of GroupBy object.**

**Program:**

import pandas as pd

pd.set\_option('display.max\_rows', None)

#pd.set\_option('display.max\_columns', None)

student\_data = pd.DataFrame({

'school\_code': ['s001','s002','s003','s001','s002','s004'],

'class': ['V', 'V', 'VI', 'VI', 'V', 'VI'],

'name': ['Alberto Franco','Gino Mcneill','Ryan Parkes', 'Eesha Hinton', 'Gino Mcneill', 'David Parkes'],

'date\_Of\_Birth ': ['15/05/2002','17/05/2002','16/02/1999','25/09/1998','11/05/2002','15/09/1997'],

'age': [12, 12, 13, 13, 14, 12],

'height': [173, 192, 186, 167, 151, 159],

'weight': [35, 32, 33, 30, 31, 32],

'address': ['street1', 'street2', 'street3', 'street1', 'street2', 'street4']},

index=['S1', 'S2', 'S3', 'S4', 'S5', 'S6'])

print("Original DataFrame:")

print(student\_data)

print('\nSplit the said data on school\_code wise:')

result = student\_data.groupby(['school\_code'])

for name,group in result:

print("\nGroup:")

print(name)

print(group)

print("\nType of the object:")

print(type(result))

**Output:**

address

S1 s001 V Alberto Franco ... 173 35 street1

S4 s001 VI Eesha Hinton ... 167 30 street1

[2 rows x 8 columns]

Group:

s002

school\_code class name ... height weight address

S2 s002 V Gino Mcneill ... 192 32 street2

S5 s002 V Gino Mcneill ... 151 31 street2

[2 rows x 8 columns]

Group:

s003

school\_code class name ... height weight address

S3 s003 VI Ryan Parkes ... 186 33 street3

[1 rows x 8 columns]

Group:

s004

school\_code class name ... height weight address

S6 s004 VI David Parkes ... 159 32 street4

[1 rows x 8 columns]

Type of the object:

<class 'pandas.core.groupby.groupby.DataFrameGroupBy'>

**ii)Write a Pandas program to split the following dataframe by school code and get mean, min, and max value of age for each school.**

**Program:**

import pandas as pd

pd.set\_option('display.max\_rows', None)

#pd.set\_option('display.max\_columns', None)

student\_data = pd.DataFrame({

'school\_code': ['s001','s002','s003','s001','s002','s004'],

'class': ['V', 'V', 'VI', 'VI', 'V', 'VI'],

'name': ['Alberto Franco','Gino Mcneill','Ryan Parkes', 'Eesha Hinton', 'Gino Mcneill', 'David Parkes'],

'date\_Of\_Birth ': ['15/05/2002','17/05/2002','16/02/1999','25/09/1998','11/05/2002','15/09/1997'],

'age': [12, 12, 13, 13, 14, 12],

'height': [173, 192, 186, 167, 151, 159],

'weight': [35, 32, 33, 30, 31, 32],

'address': ['street1', 'street2', 'street3', 'street1', 'street2', 'street4']},

index=['S1', 'S2', 'S3', 'S4', 'S5', 'S6'])

print("Original DataFrame:")

print(student\_data)

print('\nMean, min, and max value of age for each value of the school:')

grouped\_single = student\_data.groupby('school\_code').agg({'age': ['mean', 'min', 'max']})

print(grouped\_single)

**Output:**

**Original DataFrame:**

school\_code class name ... height weight address

S1 s001 V Alberto Franco ... 173 35 street1

S2 s002 V Gino Mcneill ... 192 32 street2

S3 s003 VI Ryan Parkes ... 186 33 street3

S4 s001 VI Eesha Hinton ... 167 30 street1

S5 s002 V Gino Mcneill ... 151 31 street2

S6 s004 VI David Parkes ... 159 32 street4

[6 rows x 8 columns]

Mean, min, and max value of age for each value of the school:

age

mean min max

school\_code

s001 12.5 12 13

s002 13.0 12 14

s003 13.0 13 13

s004 12.0 12 12