PRACTICAL JOURNAL IN

DATA SCIENCE

SOFT COMPUTING TECHNIQUES

SECURITY BREACHES AND COUNTERMEASURES

SUBMITTED BY

RACHANABEN RAJENDRAPRASAD JAISWAL

ROLL NO: 2414025

IN PARTIAL FULLFILMENT FOR THE DEGREE OF

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY PART-I SEMESTER I

ACADEMIC YEAR
2024-2025

PARLE TILAK VIDYALAYA ASSOCIATION'S
MULUND COLLEGE OF COMMERCE (AUTONOMOUS)
(AFFILIATED TO UNIVERSITY OF MUMBAI)
***NAAC RE-ACCREDITED A GRADE – III CYCLE* MULUND WEST, MUMBAI 400080 MAHARASHTRA, INDIA 2024-25**

Parle Tilak Vidyalaya Association's

MULUND COLLEGE OF COMMERCE (AUTONOMOUS)

(Affiliated to University of Mumbai)

MULUND WEST, MUMBAI 400080

MAHARASHTRA, INDIA

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that Ms **Rachanaben Rajendraprasad Jaiswal** of **M.Sc. I.T. Part I** Roll No 2414025 has successfully completed the practical work in Data Science in partial fulfilment of the requirements for the Semester I of **M.Sc. I.T. Part I** during the academic year **2024-25**.

Teacher In-charge

Examiner Coordinator

Date:

College Seal

Parle Tilak Vidyalaya Association's

MULUND COLLEGE OF COMMERCE (AUTONOMOUS)

(Affiliated to University of Mumbai)

MULUND WEST, MUMBAI 400080

MAHARASHTRA, INDIA

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that Ms **Rachanaben Rajendraprasad Jaiswal** of **M.Sc. I.T. Part I** Roll No 2414025 has successfully completed the practical work in Soft Computing Techniques in partial fulfilment of the requirements for the Semester I of **M.Sc. I.T. Part I** during the academic year **2024-25**.

Teacher In-charge

Examiner Coordinator

Date:

College Seal

Parle Tilak Vidyalaya Association's

MULUND COLLEGE OF COMMERCE (AUTONOMOUS)

(Affiliated to University of Mumbai)

MULUND WEST, MUMBAI 400080

MAHARASHTRA, INDIA

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that Ms **Rachanaben Rajendraprasad Jaiswal** of **M.Sc. I.T. Part I** Roll No 2414025 has successfully completed the practical work in Security Breaches and Countermeasures in partial fulfilment of the requirements for the Semester I of **M.Sc. I.T. Part I** during the academic year **2024-25**.

Teacher In-charge

Examiner Coordinator

Date:

College Seal

MAHARASHTRA, INDIA
2024-25
INDEX

Sr. No.	Title	Page No.	Sign
Pr 1	Write the programs for the following: 1 a) Text Delimited CSV to HORUS format 1 b) XML to HORUS format 1 c) JSON to HORUS format 1 d) MySql database to HORUS format 1 e) Picture(JPEG) to HORUS format 1 f) Video to HORUS format 1 g) Audio to HORUS format	**3**	
Pr 2	Write the programs for the following: 2 a) Fixers Utilities 2 b) Data Binning or Bucketing 2 c) Averaging of data 2 d) Outlier Detection 2 e) Logging	**12**	
Pr3	Write the programs for the following: 3 a) Perform following data processing using R 3 b) Program retrieve different attributes of data 3 c) Data pattern 3 d) Loading IP_DATA_ALL	**18**	

Pr4	Write the programs for the following: 4 a) Perform error management on the given data using pandas package 4 b) Write python/R program to create the network routing diagram from the given data on routers 4 c) Write a python/R program to build acyclic graph 4 d) Write python/R program to pick the content for BillBoards from the given data 4 e) Write a python/R program to generate GML file from given csv file 4 f) Write python/R program to plan location of warehouse from the given data 4 g) Write python/R program using data science via clustering to determine new warehouse using the data 4 h) Using the given data Write python/R program to plan the shipping routers from best-fit international logistics 4 i) Write python/R program to delete the best packing option to ship in container from the given data 4 j) Write python program to create delivery route using the given data 4 k) Write python program to crate simple forex trading planner from the given data 4 l) Write python program to process the balance sheet to ensure the only good data is processing 4 m) Write python program to generate payroll from the given data.	**26**	
Pr5	Write the programs for the following: Build the time hub, links and satellites.	**76**	
Pr6	Write the programs for the following: Transforming data	**83**	
Pr7	Write the programs for the following: Organizing data	**91**	
Pr8	Write the programs for the following: Generating data	**101**	

Pr9	Write the programs for the following: Data visualisation using power Bi	**116**	

A Aim:-Text Delimited CSV to HORUS format

```
import pandas as pd
sInputFileName='F:\\DS_Practicals_Msc-it\\Practical no 2\\sample_data\\Country_Code.txt'
InputData=pd.read_csv(sInputFileName,encoding="latin-1")
ProcessData=InputData
ProcessData.drop('ISO-2-CODE', axis=1,inplace=True)
ProcessData.drop('ISO-3-Code', axis=1,inplace=True)
ProcessData.rename(columns={'Country': 'CountryName'}, inplace=True)
ProcessData.rename(columns={'ISO-M49': 'CountryNumber'}, inplace=True)
ProcessData.set_index('CountryNumber', inplace=True)
ProcessData.sort_values('CountryName', axis=0, ascending=False,inplace=True)
print(ProcessData) OutputData=ProcessData sOutputFileName='HORUS-CSV-Country.csv' OutputData.to_csv(sOutputFileName, index = False) print('CSV to HORUS - Done')
```

Output:-

```
======= RESTART: F:\DS_Practicals_Msc-it\Pract
                              CountryName
CountryNumber
716                             Zimbabwe
894                               Zambia
887                                Yemen
732                        Western Sahara
876            Wallis and Futuna Islands
...                                  ...
16                         American Samoa
12                                Algeria
8                                 Albania
248                         Aland Islands
4                             Afghanistan

[247 rows x 1 columns]
CSV to HORUS - Done
```

Practical no 1

Code:-

B

Aim:-XML to HORUS format

```
import pandas as pd import
xml.etree.ElementTree as ET def
xml2df(xml_data): root = ET.XML(xml_data)
all_records = [] for i, child in enumerate(root):
record = {} for subchild in child:
record[subchild.tag] = subchild.text
            all_records.append(record)
      return pd.DataFrame(all_records)
sInputFileName='F:\\DS_Practicals_Msc-it\\Practical no
2\\sample_data\\Country_Code.xml'
InputData = open(sInputFileName).read()
ProcessDataXML=InputData
ProcessData=xml2df(ProcessDataXML)
ProcessData.drop('ISO-2-CODE', axis=1,inplace=True)
ProcessData.drop('ISO-3-Code', axis=1,inplace=True)
ProcessData.rename(columns={'Country': 'CountryName'}, inplace=True)
ProcessData.rename(columns={'ISO-M49': 'CountryNumber'}, inplace=True)
ProcessData.set_index('CountryNumber', inplace=True)
ProcessData.sort_values('CountryName', axis=0, ascending=False, inplace=True)
OutputData=ProcessData sOutputFileName='HORUS-XML-
Country.csv' OutputData.to_csv(sOutputFileName, index = False)
```

Practical no 1

Code:-

Output:-

	A
1	CountryName
2	Zimbabwe
3	Zambia
4	Yemen
5	Western Sahara
6	Wallis and Futuna Islands
7	Virgin Islands, US
8	Viet Nam
9	Venezuela每(Bolivarian Republic)
10	Vanuatu
11	Uzbekistan
12	Uruguay
13	United States of America

HORUS-XML-Country

C Aim:-

JSON to HORUS format

```
import pandas as pd
sInputFileName='F:\\DS_Practicals_Msc-it\\Practical no 2\\sample_data\\Country_Code.json'
InputData=pd.read_json(sInputFileName,orient='index',encoding="latin-1")
ProcessData=InputData
ProcessData.drop('ISO-2-CODE', axis=1,inplace=True)
ProcessData.drop('ISO-3-Code', axis=1,inplace=True)
ProcessData.rename(columns={'Country': 'CountryName'}, inplace=True)
ProcessData.rename(columns={'ISO-M49': 'CountryNumber'}, inplace=True)
ProcessData.set_index('CountryNumber', inplace=True)
ProcessData.sort_values('CountryName', axis=0, ascending=False, inplace=True)
OutputData=ProcessData sOutputFileName='HORUS-JSON-
Country.csv' OutputData.to_csv(sOutputFileName, index =
False) print('JSON to HORUS - Done')
```

Practical no 1

Code:-

Output:-

D Aim:-

MySql database to HORUS format

```
# MySql database to HORUS format
import pandas as pd
import sqlite3 as sq
sInputFileName='F:\\DS_Practicals_Msc-it\\Practical no 2\\sample_data\\utility.db'
sInputTable='Country_Code' conn = sq.connect(sInputFileName) sSQL='select *
FROM ' + sInputTable + ';' InputData=pd.read_sql_query(sSQL, conn)
ProcessData=InputData
ProcessData.drop('ISO-2-CODE', axis=1,inplace=True)
ProcessData.drop('ISO-3-Code', axis=1,inplace=True)
ProcessData.rename(columns={'Country': 'CountryName'}, inplace=True)
ProcessData.rename(columns={'ISO-M49': 'CountryNumber'}, inplace=True)
ProcessData.set_index('CountryNumber', inplace=True)
ProcessData.sort_values('CountryName', axis=0, ascending=False, inplace=True)
```

Practical no 1

Code:-

```
OutputData=ProcessData sOutputFileName='HORUS-DB-
Country.csv' OutputData.to_csv(sOutputFileName, index = False)
print('Database to HORUS - Done')
```

Output:-

```
======= RESTART: F:\DS_Practicals_
Database to HORUS - Done
```

	A	B
1	index	CountryName
2	246	Zimbabwe
3	245	Zambia
4	244	Yemen
5	243	Western Sahara
6	242	Wallis and Futuna Islands
7	241	Virgin Islands, US
8	240	Viet Nam
9	239	Venezuela每(Bolivarian Republic)
10	238	Vanuatu
11	237	Uzbekistan
12	236	Uruguay

E Aim:-

Picture(JPEG) to HORUS format

```
import cv2 import pandas as
pd import matplotlib.pyplot as
plt
import numpy as np
sInputFileName = 'F:\\DS_Practicals_Msc-it\\Practical no 2\\sample_data\\download.png'
InputData = cv2.imread(sInputFileName, cv2.IMREAD_UNCHANGED) if InputData is
None:
    print("Error: Image not found or unable to load!")
else:
    print('Input Data Values ==================================')
    print('X (Height): ', InputData.shape[0]) print('Y (Width): ', InputData.shape[1])
    print('Channels (RGBA): ', InputData.shape[2] if len(InputData.shape) > 2 else 1)
```

Practical no 1

Code:-

```
print('=====================================================')
) ProcessRawData = InputData.flatten()
y = InputData.shape[2] if len(InputData.shape) > 2 else 1
x = int(ProcessRawData.shape[0] / y)
ProcessData = pd.DataFrame(np.reshape(ProcessRawData, (x, y)), columns=['Red', 'Green',
'Blue', 'Alpha'][:y])
ProcessData['XAxis'] = np.tile(np.arange(InputData.shape[1]), InputData.shape[0])
ProcessData['YAxis'] = np.repeat(np.arange(InputData.shape[0]), InputData.shape[1])
ProcessData = ProcessData[['XAxis', 'YAxis', 'Red', 'Green', 'Blue', 'Alpha'][:y + 2]]
print('Rows: ', ProcessData.shape[0])
print('Columns: ', ProcessData.shape[1])
  print('Process Data Values ==================================')
print(ProcessData.head())
plt.imshow(cv2.cvtColor(InputData, cv2.COLOR_BGR2RGB))
plt.axis('off')
plt.show()
sOutputFileName = 'HORUS-Picture.csv'
ProcessData.to_csv(sOutputFileName, index=False)
print('Data saved to file:', sOutputFileName)
print('Picture to HORUS - Done')
```

```
======= RESTART: F:\DS_Practicals_Msc-it\P
Input Data Values ========================
X (Height):  823
Y (Width):  1089
Channels (RGBA):  4
==========================================
Rows:  896247
Columns:  6
==========================================
Process Data Values ======================
   XAxis  YAxis  Red  Green  Blue  Alpha
0      0      0  255    255   255    255
1      1      0  255    255   255    255
2      2      0  255    255   255    255
3      3      0  255    255   255    255
4      4      0  255    255   255    255
==========================================
```

F Aim:-

Video to HORUS format

```
import cv2 import pandas as
pd import matplotlib.pyplot as
plt import numpy as np
import os
# Input Agreement ==========================================
sDataBaseDir = 'F:\\DS_Practicals_Msc-it\\Practical no 2\\videotohorus\\temp'
f = 0
```

Practical no 1

Code:-

```
# Initialize an empty list to collect the data frames
frame_data_list = []
# Loop through all image files in the directory
for file in os.listdir(sDataBaseDir):
    if file.endswith(".jpg"):
        f += 1
        sInputFileName = os.path.join(sDataBaseDir, file)
        print('Processing:', sInputFileName)

       # Read the image
        InputData = cv2.imread(sInputFileName, cv2.IMREAD_UNCHANGED)
       # Convert all images to RGBA if
        len(InputData.shape) == 2: # Grayscale
            InputData = cv2.cvtColor(InputData, cv2.COLOR_GRAY2RGBA)
        elif InputData.shape[2] == 3:        # RGB
            InputData = cv2.cvtColor(InputData, cv2.COLOR_RGB2RGBA)
                # Print shape information print('Input
        Data Dimensions:') print('X (Height):',
        InputData.shape[0])          # Height print('Y
        (Width):', InputData.shape[1])       # Width
        print('Channels (RGBA):', InputData.shape[2])    # Channels (Always 4)
        print('==================================================')
        # Flatten the image and create a DataFrame
        ProcessRawData = InputData.flatten()
        x = int(ProcessRawData.shape[0] / 4)    # Divide by 4 channels (RGBA)
        ProcessFrameData = pd.DataFrame(np.reshape(ProcessRawData, (x, 4)))
        ProcessFrameData['FrameName'] = file    # Add the frame name
            # Append to the listframe_data_list.append(ProcessFrameData)
            # Display the image plt.imshow(cv2.cvtColor(InputData,
        cv2.COLOR_RGBA2RGB)) # Convert RGBA
to RGB for display
        plt.show()
# Combine all frames into a single DataFrame
if f > 0:
    # Concatenate all frames into one DataFrame
    ProcessData = pd.concat(frame_data_list, ignore_index=True)
        # Assign column names
    ProcessData.columns = ['Red', 'Green', 'Blue', 'Alpha', 'FrameName']
```

```
# Output Agreement
========================================== print('Processed
Frames:', f) print('Total Rows:', ProcessData.shape[0])
print('Total Columns:', ProcessData.shape[1])
# Save the DataFrame to CSV sOutputFileName = 'F:\\DS_Practicals_Msc-
it\\Practical no 2\\videotohorus\\HORUS-
Movie-Frame.csv'
ProcessData.to_csv(sOutputFileName, index=False)
print('Data saved to:', sOutputFileName)
print('=====================================================')
print('Movie to HORUS - Done')
print('=====================================================')
```

Output:-

1 G Aim:- Audio to HORUS format Code:-

```
from scipy.io import wavfile
import pandas as pd import
matplotlib.pyplot as plt import
numpy as np
def show_info(aname, a,r):
print ('----------------')
print ("Audio:", aname)
print ('----------------')
print ("Rate:", r)
print ('----------------') print
("shape:", a.shape) print ("dtype:",
a.dtype) print ("min, max:", a.min(),
a.max())
print ('----------------')
plot_info(aname, a,r)
def plot_info(aname, a,r):
```

Practical no

```
    sTitle= 'Signal Wave - '+ aname + ' at ' + str(r) +
    'hz' plt.title(sTitle) sLegend=[] for c in
    range(a.shape[1]):
        sLabel = 'Ch' + str(c+1)
        sLegend=sLegend+[str(c+1)]
        plt.plot(a[:,c], label=sLabel)
    plt.legend(sLegend,loc="best")
    plt.show()
  sInputFileName='D:\\MscIT\\DS\\DS_Practicals\\Practicals\\Practical no 2\\sample_data\\2ch-
sound.wav'
print('=====================================================')
print('Processing : ', sInputFileName)
print('=====================================================')
InputRate, InputData = wavfile.read(sInputFileName)
show_info("2 channel", InputData,InputRate)
ProcessData=pd.DataFrame(InputData) sColumns=
['Ch1','Ch2'] ProcessData.columns=sColumns
OutputData=ProcessData
sOutputFileName='Practical no 2/sample_output/HORUS-Audio-2ch.csv'
OutputData.to_csv(sOutputFileName, index = False)
   sInputFileName='D:\\MscIT\\DS\\DS_Practicals\\Practicals\\Practicalno 2\\sample_data\\4ch-
sound.wav'
print('=====================================================')
print('Processing : ', sInputFileName)
print('=====================================================')
InputRate, InputData = wavfile.read(sInputFileName)
show_info("4 channel", InputData,InputRate)
ProcessData=pd.DataFrame(InputData) sColumns=
['Ch1','Ch2','Ch3', 'Ch4']
ProcessData.columns=sColumns
OutputData=ProcessData
sOutputFileName='Practical no 2/sample_output/HORUS-Audio-4ch.csv'
OutputData.to_csv(sOutputFileName, index = False)
sInputFileName='D:\\MscIT\\DS\\DS_Practicals\\Practicals\\Practical no
2\\sample_data\\6chsound.wav'
print('=====================================================')
print('Processing : ', sInputFileName)
print('=====================================================')
InputRate, InputData = wavfile.read(sInputFileName)
show_info("6 channel", InputData,InputRate)
ProcessData=pd.DataFrame(InputData) sColumns=
['Ch1','Ch2','Ch3', 'Ch4', 'Ch5','Ch6']
ProcessData.columns=sColumns
OutputData=ProcessData
sOutputFileName='Practical no 2/sample_output/HORUS-Audio-6ch.csv'
OutputData.to_csv(sOutputFileName, index = False)
```

```
sInputFileName='D:\\MscIT\\DS\\DS_Practicals\\Practicals\\Practical no 2\\sample_data\\8chsound.wav'
print('=====================================================')
print('Processing : ', sInputFileName)
print('=====================================================')
InputRate, InputData = wavfile.read(sInputFileName)
show_info("8 channel", InputData,InputRate)
ProcessData=pd.DataFrame(InputData)
sColumns= ['Ch1','Ch2','Ch3', 'Ch4', 'Ch5','Ch6','Ch7','Ch8']
ProcessData.columns=sColumns
OutputData=ProcessData
sOutputFileName='Practical no 2/sample_output/HORUS-Audio-8ch.csv'
OutputData.to_csv(sOutputFileName, index = False)
print('=====================================================')
print('Audio to HORUS - Done')
```

Output:-

Practical no 2

A

Aim:- Fixers Utilities Code;-

```
# Fixers Utilities
import string
import datetime as dt
baddata = "        \nbad!!!            \nData Science with too many spaces is    \n"
print('Original with spaces:', '>' + baddata + '<') cleandata =
baddata.strip() print('Cleaned:', '>' + cleandata + '<') baddata =
"Data\x00Science with\x02 funny characters is        \x10bad!!!"
cleandata = ".join(filter(lambda x: x in string.printable, baddata))
print('Non-printable characters removed:', cleandata) baddate =
dt.date(2017, 1, 31) formatted_baddate = format(baddate, '%Y-%m-
%d') print('Original date format:', formatted_baddate) gooddate =
dt.datetime.strptime(formatted_baddate, '%Y-%m-%d')
formatted_gooddate = format(gooddate, '%d %B %Y')
print('Reformatted date:', formatted_gooddate)
```

Output:-

```
================================ RESTART: F:\DS_Practicals_Msc-it\Practical no
Original with spaces: >
bad!!!
Data Science with too many spaces is
<
Cleaned: >bad!!!
Data Science with too many spaces is<
Non-printable characters removed: DataScience with funny characters is  bad!!!
Original date format: 2017-01-31
Reformatted date: 31 January 2017
>
```

B Aim:-

Data Binning or Bucketing

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm
np.random.seed(0)
mu = 90
sigma = 25
x = mu + sigma * np.random.randn(5000) num_bins = 25
fig, ax = plt.subplots()
n, bins, patches = ax.hist(x, num_bins, density=True)
y = norm.pdf(bins, mu, sigma)
```

Practical no 2

Code:-

```
ax.plot(bins, y, '--')
ax.set_xlabel('Example Data')
ax.set_ylabel('Probability Density')
sTitle = f'Histogram {len(x)} entries into {num_bins} bins: μ={mu}, σ={sigma}'
ax.set_title(sTitle) fig.tight_layout() plt.show()
```

Output:-

C

Aim:- Averaging of data

Averaging of data

```
import pandas as pd
sFileName='F:\\DS_Practicals_Msc-it\\Practical no 3\\sample_data\\IP_DATA_CORE.csv'
print('Loading :',sFileName)
IP_DATA_ALL=pd.read_csv(sFileName,header=0,low_memory=False,
usecols=['Country','Place Name','Latitude','Longitude'], encoding="latin-
1")
IP_DATA_ALL.rename(columns={'Place Name': 'Place_Name'}, inplace=True)
AllData=IP_DATA_ALL[['Country', 'Place_Name','Latitude']]
print(AllData)
```

Practical no 2

Code:-

```
#
MeanData=AllData.groupby(['Country', 'Place_Name'])['Latitude'].mean()
print(MeanData)
```

OutPut:-

```
================================ RESTART: F:\DS_Practicals_Msc-it\Practical no
Loading : F:\DS_Practicals_Msc-it\Practical no 3\sample_data\IP_DATA_CORE.csv
     Country Place_Name  Latitude
0         US   New York   40.7528
1         US   New York   40.7528
2         US   New York   40.7528
3         US   New York   40.7528
4         US   New York   40.7528
...      ...        ...       ...
3557      DE     Munich   48.0915
3558      DE     Munich   48.1833
3559      DE     Munich   48.1000
3560      DE     Munich   48.1480
3561      DE     Munich   48.1480

[3562 rows x 3 columns]
Country  Place_Name
DE       Munich        48.143223
GB       London        51.509406
US       New York      40.747044
Name: Latitude, dtype: float64
```

Practical no 2

Code:-

#

D

Aim:- Outlier Detection

Outlier Detection

```
import pandas as pd
InputFileName = "Practical no 3\sample_data\IP_DATA_CORE.csv"
sFileName = InputFileName print('Loading:', sFileName)
IP_DATA_ALL = pd.read_csv(sFileName, header=0, low_memory=False,usecols=['Country',
'Place Name', 'Latitude', 'Longitude'], encoding="latin-1")
IP_DATA_ALL.rename(columns={'Place Name': 'Place_Name'}, inplace=True)
LondonData = IP_DATA_ALL.loc[IP_DATA_ALL['Place_Name'] == 'London']
AllData = LondonData[['Country', 'Place_Name', 'Latitude']]
print('All Data') print(AllData)
MeanData = AllData.groupby(['Country', 'Place_Name'])['Latitude'].mean()
StdData = AllData.groupby(['Country', 'Place_Name'])['Latitude'].std()
UpperBound = float((MeanData + StdData).iloc[0])
LowerBound = float((MeanData - StdData).iloc[0])
print('Outliers')
print('Higher than', UpperBound)
OutliersHigher = AllData[AllData.Latitude > UpperBound]
print(OutliersHigher)
print('Lower than', LowerBound)
OutliersLower = AllData[AllData.Latitude < LowerBound]
print(OutliersLower) print('Not Outliers')
OutliersNot = AllData[(AllData.Latitude >= LowerBound) & (AllData.Latitude <=
```

```
Loading: F:\DS_Practicals_Msc-it\Prac
All Data
     Country Place_Name  Latitude
1910      GB     London   51.5130
1911      GB     London   51.5508
1912      GB     London   51.5649
1913      GB     London   51.5895
1914      GB     London   51.5232
...      ...        ...       ...
3434      GB     London   51.5092
3435      GB     London   51.5092
3436      GB     London   51.5163
3437      GB     London   51.5085
3438      GB     London   51.5136

[1502 rows x 3 columns]
Outliers
Higher than 51.512635507867415
```

```
3438      GB     London   51.5136
Lower than 51.506176875621264
     Country Place_Name  Latitude
1915      GB     London   51.4739
Not Outliers
     Country Place_Name  Latitude
1917      GB     London   51.5085
1918      GB     London   51.5085
1922      GB     London   51.5085
1928      GB     London   51.5085
1929      GB     London   51.5085
...      ...        ...       ...
3432      GB     London   51.5092
3433      GB     London   51.5092
3434      GB     London   51.5092
3435      GB     London   51.5092
3437      GB     London   51.5085

[1485 rows x 3 columns]
```

Practical no 2

Code:-

```
#
UpperBound)]
print(OutliersNot)
```

Output:-

E Aim:-

Logging

```
  Logging import
os import logging
import uuid
import shutil
import time Base
= 'D:/MscIT'
sSubjects = ['CloudComputing', 'DS', 'SBC', 'SCT']
sSubfolders = {
    'DS': ['DS_Practicals/Practicals/Practical no 1', 'DS_Practicals/Practicals/Practical no 2',
'DS_Practicals/Practicals/Practical no 3'],
    'SBC': ['Practicals', 'presentation'],
    'SCT': ['Chapterwise notes', 'sct_practicals/practicals', 'webnotes']
} sLevels = ['debug', 'info', 'warning',
'error'] for subject in sSubjects:
    subject_dir = os.path.join(Base, subject)
    if not os.path.exists(subject_dir):
    os.makedirs(subject_dir)

    subdirs = sSubfolders.get(subject, [])
    for subdir in subdirs:
        log_dir = os.path.join(subject_dir, subdir, 'Logging')
        if os.path.exists(log_dir):
            shutil.rmtree(log_dir)
        time.sleep(2)
        os.makedirs(log_dir, exist_ok=True)
        skey = str(uuid.uuid4())
        log_file = os.path.join(log_dir,
        f'Logging_{skey}.log') print('Set up:', log_file)
        logging.basicConfig( level=logging.DEBUG,
            format='%(asctime)s %(name)-12s %(levelname)-8s %(message)s',
            datefmt='%m-%d %H:%M',
            filename=log_file,
            filemode='w'
        )
        console = logging.StreamHandler()
        console.setLevel(logging.INFO)
```

```
formatter = logging.Formatter('%(name)-12s: %(levelname)-8s
%(message)s') console.setFormatter(formatter)
logging.getLogger('').addHandler(console) for sLevel in sLevels: logger =
logging.getLogger(f"{subject}-{subdir}-{sLevel}") if sLevel == 'debug':
logger.debug('Logging a debugging message.')
    elif sLevel == 'info':
        logger.info('Logging an informational message.')
    elif sLevel == 'warning': logger.warning('Logging
        a warning message.')
    elif sLevel == 'error':
        logger.error('Logging an error message.')
```

Output:-

```
Set up: D:/MscIT\DS\DS_Practicals/Practicals/Practical no 1\Logging\Logging_36aeb726-dd5f-4c5f-903c-ed508dc9fa18.log
DS-DS_Practicals/Practicals/Practical no 1-info: INFO     Logging an informational message.
DS-DS_Practicals/Practicals/Practical no 1-warning: WARNING  Logging a warning message.
DS-DS_Practicals/Practicals/Practical no 1-error: ERROR    Logging an error message.
Set up: D:/MscIT\DS\DS_Practicals/Practicals/Practical no 2\Logging\Logging_96fbfc9f-234f-446d-9a43-96dbccda39c9.log
DS-DS_Practicals/Practicals/Practical no 2-info: INFO     Logging an informational message.
DS-DS_Practicals/Practicals/Practical no 2-info: INFO     Logging an informational message.
DS-DS_Practicals/Practicals/Practical no 2-warning: WARNING  Logging a warning message.
DS-DS_Practicals/Practicals/Practical no 2-warning: WARNING  Logging a warning message.
DS-DS_Practicals/Practicals/Practical no 2-error: ERROR    Logging an error message.
DS-DS_Practicals/Practicals/Practical no 2-error: ERROR    Logging an error message.
Set up: D:/MscIT\DS\DS_Practicals/Practicals/Practical no 3\Logging\Logging_35ecca51-6031-44eb-94e0-3efab6526776.log
DS-DS_Practicals/Practicals/Practical no 3-info: INFO     Logging an informational message.
DS-DS_Practicals/Practicals/Practical no 3-info: INFO     Logging an informational message.
DS-DS_Practicals/Practicals/Practical no 3-info: INFO     Logging an informational message.
DS-DS_Practicals/Practicals/Practical no 3-warning: WARNING  Logging a warning message.
DS-DS_Practicals/Practicals/Practical no 3-warning: WARNING  Logging a warning message.
DS-DS_Practicals/Practicals/Practical no 3-warning: WARNING  Logging a warning message.
DS-DS_Practicals/Practicals/Practical no 3-error: ERROR    Logging an error message.
DS-DS_Practicals/Practicals/Practical no 3-error: ERROR    Logging an error message.
DS-DS_Practicals/Practicals/Practical no 3-error: ERROR    Logging an error message.
Set up: D:/MscIT\SBC\Practicals\Logging\Logging_61a6c268-9297-4cb7-978f-8f4297468299.log
SBC-Practicals-info: INFO     Logging an informational message.
SBC-Practicals-info: INFO     Logging an informational message.
SBC-Practicals-info: INFO     Logging an informational message.
SBC-Practicals-info: INFO     Logging an informational message.
SBC-Practicals-warning: WARNING  Logging a warning message.
SBC-Practicals-warning: WARNING  Logging a warning message.
SBC-Practicals-warning: WARNING  Logging a warning message.
SBC-Practicals-warning: WARNING  Logging a warning message.
SBC-Practicals-error: ERROR    Logging an error message.
SBC-Practicals-error: ERROR    Logging an error message.
SBC-Practicals-error: ERROR    Logging an error message.
```

Practical no
3 A

Aim:- Perform following data processing using R

Code:-

```
library(readr)
library(tibble)
library(data.table)
library(e1071)
FileName=paste0('F:/DS_Practicals_Msc-it/Practical no 2/sample_data/IP_DATA_CORE.csv')

IP_DATA_ALL <- read_csv(FileName, col_types = cols(ID = col_integer(),Country = col_character(), 'First IP Number' = col_integer(), 'Last IP Number' = col_integer(),Latitude = col_double(), Longitude = col_double(),
                                                    'Place Name' = col_character(),
                                                    'Post Code' = col_character()
                                                  ))

IP_DATA_ALL
```

```
# A tibble: 3,562 × 8
      ID Country `Place Name` `Post Code` Latitude Longitude `First IP Number` `Last IP Number`
   <int> <chr>   <chr>        <chr>          <dbl>     <dbl>            <int>           <int>
 1     1 US      New York     10017           40.8     -74.0        204276480       204276735
 2     2 US      New York     10017           40.8     -74.0        301984864       301985791
 3     3 US      New York     10017           40.8     -74.0        404678736       404679039
 4     4 US      New York     10017           40.8     -74.0        411592704       411592959
 5     5 US      New York     10017           40.8     -74.0        416784384       416784639
 6     6 US      New York     10017           40.8     -74.0        643347456       643348479
 7     7 US      New York     10017           40.8     -74.0        643738112       643738623
 8     8 US      New York     10017           40.8     -74.0        644459264       644459519
 9     9 US      New York     10017           40.8     -74.0        644663808       644664319
10    10 US      New York     10017           40.8     -74.0        645591040       645591551
# i 3,552 more rows
```

```
IP_DATA_ALL_FIX=set_tidy_names(IP_DATA_ALL,syntactic = TRUE,quiet = TRUE)

IP_DATA_ALL_FIX
```

```
IP_DATA_ALL_FIX
A tibble: 3,562 × 8
      ID Country Place.Name Post.Code Latitude Longitude First.IP.Number Last.IP.Number
   <int> <chr>   <chr>      <chr>        <dbl>     <dbl>           <int>          <int>
1      1 US      New York   10017         40.8     -74.0       204276480      204276735
2      2 US      New York   10017         40.8     -74.0       301984864      301985791
3      3 US      New York   10017         40.8     -74.0       404678736      404679039
4      4 US      New York   10017         40.8     -74.0       411592704      411592959
```

```
NEWFILENAME="F:/DS_Practicals_Msc-it/Practical no 4/IP_DATA_ALL_FIX.csv"
write.csv(IP_DATA_ALL_FIX,NEWFILENAME)
IP_DATA_ALL_with_ID=rowid_to_column(IP_DATA_ALL_FIX,var="RowID")
View(IP_DATA_ALL_with_ID)
```

Practicalno4a.r* × | IP_DATA_ALL_with_ID ×

Filter

	RowID	ID	Country	Place.Name	Post.Code	Latitude	Longitude	First.IP.Number	Last.IP.Number
1	1	1	US	New York	10017	40.7528	-73.9725	204276480	204276735
2	2	2	US	New York	10017	40.7528	-73.9725	301984864	301985791

```
write.csv(IP_DATA_ALL_FIX,NEWFILENAME)
sapply(IP_DATA_ALL_with_ID,typeof)
      RowID          ID     Country  Place.Name   Post.Code    Latitude   Longitude First.IP.Number
  "integer"   "integer" "character" "character" "character"    "double"    "double"       "integer"
Last.IP.Number
  "integer"
hist_country=data.table(Country=unique(IP_DATA_ALL_with_ID[is.na(IP_DATA_ALL_with
_ID['Country'])==0,]$Country))
setorder(hist_country,'Country')
hist_country_with_id=rowid_to_column(hist_country,var="RowIDCountry")
View(hist_country_with_id)
```

	RowIDCountry	Country
1	1	DE
2	2	GB
3	3	US

```
IP_DATA_COUNTRY_FREQ=data.table(with(IP_DATA_ALL_with_ID,table(Country)))
View(IP_DATA_COUNTRY_FREQ)
```

Filter

	Country	N
1	DE	677
2	GB	1502
3	US	1383

```
#Longitude :

hist_Longitude=data.table(Longitude=unique(IP_DATA_ALL_with_ID[is.na(IP_DATA_ALL_with_ID['Longitude'])==0,
]$Longitude))
setkeyv(hist_Longitude,'Longitude')
setorder(hist_Longitude)
hist_Longitude_with_id=rowid_to_column(hist_Longitude,var="RowID")
View(hist_Longitude_with_id)
IP_DATA_Longitude_FREQ=data.table(with(IP_DATA_ALL_with_ID,table(Longitude)))
LongitudeCSVFname="F:/New folder/DSprac/LongitudeFreq.csv"
IP_DATA_Longitude_FREQ.to_csv(LongitudeCSVFname)
```

Practicalno4a.r* × hist_Longitude_with_id ×

Filter

	RowID	Longitude
1	1	-74.0203
2	2	-74.0166
3	3	-74.0150
4	4	-74.0132
5	5	-74.0131
6	6	-74.0104

```
> #minimum value
>
> min(hist_country$Country)
[1] "DE"
>
> sapply(hist_country[,'Country'],min,na.rm=TRUE)
Country
   "DE"
>
> sapply(hist_latitude_with_id[,'Latitude'],min,na.rm=TRUE)
Latitude
 40.6888
>
> sapply(hist_Longitude_with_id[,'Longitude'],min,na.rm=TRUE)
Longitude
 -74.0203
```

```
> #maximum value
>
> max(hist_country$Country)
[1] "US"
>
> sapply(hist_country[,'Country'],max,na.rm=TRUE)
Country
   "US"
>
> sapply(hist_latitude_with_id[,'Latitude'],max,na.rm=TRUE)
Latitude
 51.5895
>
> sapply(hist_Longitude_with_id[,'Longitude'],max,na.rm=TRUE)
Longitude
    11.75
>
> #mean
> sapply(hist_latitude_with_id[,'Latitude'],mean,na.rm=TRUE)
Latitude
45.23386
>
> sapply(hist_Longitude_with_id[,'Longitude'],mean,na.rm=TRUE)
Longitude
-26.43165
>
> #median :
> sapply(hist_latitude_with_id[,'Latitude'],median,na.rm=TRUE)
Latitude
 48.1098
```

```
#Mode :
IP_DATA_COUNTRY_FREQ=data.table(with(IP_DATA_ALL_with_ID,table(Country)))
setorder(IP_DATA_COUNTRY_FREQ,-N)
IP_DATA_COUNTRY_FREQ[1,'Country']

IP_DATA_Longitude_FREQ=data.table(with(IP_DATA_ALL_with_ID,table(Longitude)))
setorder(IP_DATA_Longitude_FREQ,-N) IP_DATA_Longitude_FREQ[1,'Longitude']
IP_DATA_Latitude_FREQ=data.table(with(IP_DATA_ALL_with_ID,table(Latitude)))
setorder(IP_DATA_Latitude_FREQ,-N)
IP_DATA_Latitude_FREQ[1,'Latitude']
```

```
> #Mode :
> IP_DATA_COUNTRY_FREQ=data.table(with(IP_DATA_ALL_with_ID,table(Country)))
> setorder(IP_DATA_COUNTRY_FREQ,-N)
> IP_DATA_COUNTRY_FREQ[1,'Country']
   Country
    <char>
1:      GB
>
> IP_DATA_Longitude_FREQ=data.table(with(IP_DATA_ALL_with_ID,table(Longitude)))
> setorder(IP_DATA_Longitude_FREQ,-N)
> IP_DATA_Longitude_FREQ[1,'Longitude']
   Longitude
      <char>
1:   -0.0955
>
> IP_DATA_Latitude_FREQ=data.table(with(IP_DATA_ALL_with_ID,table(Latitude)))
> setorder(IP_DATA_Latitude_FREQ,-N)
> IP_DATA_Latitude_FREQ[1,'Latitude']
   Latitude
     <char>
1:  51.5092
```

#Range :

```
sapply(hist_latitude_with_id[,'Latitude'],range,na.rm=TRUE)
sapply(hist_Longitude_with_id[,'Longitude'],range,na.rm=TRUE)
```

#Quartiles :

```
sapply(hist_latitude_with_id[,'Latitude'],quantile,na.rm=TRUE)
sapply(hist_Longitude_with_id[,'Longitude'],quantile,na.rm=TRUE)
```

#Standard Deviation :

```
sapply(hist_latitude_with_id[,'Latitude'],sd,na.rm=TRUE)
sapply(hist_Longitude_with_id[,'Longitude'],sd,na.rm=TRUE)
```

#Skewness :

```
skewness(hist_latitude_with_id$Latitude,na.rm=FALSE,type=2)
skewness(hist_Longitude_with_id$Longitude,na.rm=FALSE,type=2)
```

#Missing Values :

```
missing_country=data.table(Country=unique(IP_DATA_ALL_with_ID[is.na(IP_DATA_ALL_with_ID['Country'])==1,]))
View(missing_country)
```

```
> #Range :
>
> sapply(hist_latitude_with_id[,'Latitude'],range,na.rm=TRUE)
     Latitude
[1,]  40.6888
[2,]  51.5895
>
> sapply(hist_Longitude_with_id[,'Longitude'],range,na.rm=TRUE)
     Longitude
[1,]  -74.0203
[2,]   11.7500
>
> #Quartiles :
> sapply(hist_latitude_with_id[,'Latitude'],quantile,na.rm=TRUE)
     Latitude
0%    40.6888
25%   40.7586
50%   48.1098
75%   48.1526
100%  51.5895
> sapply(hist_Longitude_with_id[,'Longitude'],quantile,na.rm=TRUE)
     Longitude
0%    -74.0203
25%   -73.9778
50%    -0.0912
75%    11.5585
100%   11.7500
```

```
> #Standard Deviation :
>
> sapply(hist_latitude_with_id[,'Latitude'],sd,na.rm=TRUE)
Latitude
4.110526
> sapply(hist_Longitude_with_id[,'Longitude'],sd,na.rm=TRUE)
Longitude
 41.54072
>
> #Skewness :
> skewness(hist_latitude_with_id$Latitude,na.rm=FALSE,type=2)
[1] -0.03768831
> skewness(hist_Longitude_with_id$Longitude,na.rm=FALSE,type=2)
[1] -0.2677529
>
> #Missing Values :
>
> missing_country=data.table(Country=unique(IP_DATA_ALL_with_ID[is.na(IP_DATA_ALL_with_
> View(missing_country)
```

Practical no 3

B

Aim:- Program retrieve different attributes of data Code:-

```
# Program retrieve different attributes of data
import pandas as pd
sFileName='F:\DS_Practicals_Msc-it\DS_Practicals_Msc-it\\Practical no 4\\IP_DATA_ALL.csv'
print('Loading :',sFileName)
IP_DATA_ALL=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
print('Rows:', IP_DATA_ALL.shape[0]) print('Columns:', IP_DATA_ALL.shape[1])
print('Raw Data Set') for i in range(0,len(IP_DATA_ALL.columns)):
    print(IP_DATA_ALL.columns[i],type(IP_DATA_ALL.columns[i]))
print('Fixed Data Set')
IP_DATA_ALL_FIX=IP_DATA_ALL
for i in range(0,len(IP_DATA_ALL.columns)):
    cNameOld=IP_DATA_ALL_FIX.columns[i] + '        '
    cNameNew=cNameOld.strip().replace(" ", ".")
    IP_DATA_ALL_FIX.columns.values[i] = cNameNew
    print(IP_DATA_ALL.columns[i],type(IP_DATA_ALL.columns[i]))
print('Fixed Data Set with ID')
IP_DATA_ALL_with_ID=IP_DATA_ALL_FIX
IP_DATA_ALL_with_ID.index.names = ['RowID']
sFileName2='Practical no 4\\Retrieve_IP_DATA1.csv'
IP_DATA_ALL_with_ID.to_csv(sFileName2, index = True, encoding="latin-1")
```

print('Done') Output:-

```
======= RESTART: F:\DS_Practicals_Msc-it\Practical no 4\Practicalno4b.py =
Loading : F:\DS_Practicals_Msc-it\Practical no 4\IP_DATA_ALL.csv
Rows: 1247502
Columns: 9
Raw Data Set
Unnamed: 0 <class 'str'>
ID <class 'str'>
Country <class 'str'>
Place.Name <class 'str'>
Post.Code <class 'str'>
Latitude <class 'str'>
Longitude <class 'str'>
First.IP.Number <class 'str'>
Last.IP.Number <class 'str'>
Fixed Data Set
Unnamed:.0 <class 'str'>
ID <class 'str'>
Country <class 'str'>
Place.Name <class 'str'>
Post.Code <class 'str'>
Latitude <class 'str'>
Longitude <class 'str'>
First.IP.Number <class 'str'>
Last.IP.Number <class 'str'>
Fixed Data Set with ID
```

C

Practical no 3

Aim:-Data Pattern

Code:-

```
# Data pattern :
library(readr)
library(data.table)
FileName=paste0('F:\\DS_Practicals_Msc-it\\Practical no 4\\IP_DATA_ALL.csv')
IP_DATA_ALL <- read_csv(FileName)
hist_country=data.table(Country=unique(IP_DATA_ALL$Country))
pattern_country=data.table(Country=hist_country$Country,PatternCountry=hist_country$Coun
t ry)
oldchar=c(letters,LETTERS)
newchar=replicate(length(oldchar),"A")
for (r in seq(nrow(pattern_country))){
s=pattern_country[r,]$PatternCountry; for
(c in seq(length(oldchar))){
s=chartr(oldchar[c],newchar[c],s)
   };
   for (n in seq(0,9,1)){
      s=chartr(as.character(n),"N",s)
   }; s=chartr("
   ","b",s)
   s=chartr(".","u",s
   )
   pattern_country[r,]$PatternCountry=s;
};
View(pattern_country)
```

Output:-

```
> FileName=paste0('F:\\DS_Practicals_Msc-it\\Practical no 4\\IP_DATA_ALL.csv')
> IP_DATA_ALL <- read_csv(FileName)
New names:
• `` -> `...1`
Rows: 1247502 Columns: 9
── Column specification ────────────────────────────────────────────────────────
Delimiter: ","
chr (3): Country, Place.Name, Post.Code
dbl (6): ...1, ID, Latitude, Longitude, First.IP.Number, Last.IP.Number

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Practicalno4a.r | Practicalno4c.r* | pattern_country

Filter

	Country	PatternCountry
1	BW	AA
2	NE	AA
3	MZ	AA

D

Practical no 3

Aim:-Loading IP_DATA_ALL

Code:-

```
# Loading IP_DATA_ALL : import
pandas as pd
sFileName='F:\\DS_Practicals_Msc-it\\Practical no 4\\IP_DATA_ALL.csv'
print('Loading :',sFileName)
IP_DATA_ALL=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
print('Rows:', IP_DATA_ALL.shape[0]) print('Columns:', IP_DATA_ALL.shape[1])
print('Raw Data Set') for i in range(0,len(IP_DATA_ALL.columns)):
    print(IP_DATA_ALL.columns[i],type(IP_DATA_ALL.columns[i]))
print('Fixed Data Set')
IP_DATA_ALL_FIX=IP_DATA_ALL
for i in range(0,len(IP_DATA_ALL.columns)):
    cNameOld=IP_DATA_ALL_FIX.columns[i] + '        '
    cNameNew=cNameOld.strip().replace(" ", ".")
    IP_DATA_ALL_FIX.columns.values[i] = cNameNew
    print(IP_DATA_ALL.columns[i],type(IP_DATA_ALL.columns[i]))
print('Fixed Data Set with ID')
IP_DATA_ALL_with_ID=IP_DATA_ALL_FIX
IP_DATA_ALL_with_ID.index.names = ['RowID']
sFileName2='F:\\DS_Practicals_Msc-it\\Practical no 4\\Retrieve_IP_DATA.csv'
IP_DATA_ALL_with_ID.to_csv(sFileName2, index = True, encoding="latin-1")
print('Done') Output:-
```

```
======= RESTART: F:\DS_Practicals_Msc-it\Practical no 4\Practical
Loading : F:\DS_Practicals_Msc-it\Practical no 4\IP_DATA_ALL.csv
Rows: 1247502
Columns: 9
Raw Data Set
Unnamed: 0 <class 'str'>
ID <class 'str'>
Country <class 'str'>
Place.Name <class 'str'>
Post.Code <class 'str'>
Latitude <class 'str'>
Longitude <class 'str'>
First.IP.Number <class 'str'>
Last.IP.Number <class 'str'>
Fixed Data Set
Unnamed:.0 <class 'str'>
ID <class 'str'>
Country <class 'str'>
Place.Name <class 'str'>
Post.Code <class 'str'>
Latitude <class 'str'>
Longitude <class 'str'>
First.IP.Number <class 'str'>
Last.IP.Number <class 'str'>
Fixed Data Set with ID
Done
```

Practical no
4 A Aim:- Perform error management on the given data using pandas package

1)Drop the Columns Where All Elements Are Missing Values

Code:

```
import
pandas as pd
sInputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5A\\Good-or-Bad.csv'
sOutputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5A\\Good-or-Bad01.csv'
#################################################################
### Import Warehouse
#################################################################
sFileName=sInputFileName print('Loading :',sFileName)
RawData=pd.read_csv(sFileName,header=0)
print('## Raw Data Values')
print('#################################')
print(RawData)
print('#################################')
print('## Data Profile')
print('#################################')
print('Rows :',RawData.shape[0])
print('Columns :',RawData.shape[1])
print('#################################')
#################################################################
sFileName=sInputFileName
RawData.to_csv(sFileName, index = False)
#################################################################
TestData=RawData.dropna(axis=1, how='all')
#################################################################
print('#################################')
print('## Test Data Values')
print('#################################')
print(TestData)
print('#################################')
print('## Data Profile')
print('#################################')
print('Rows :',TestData.shape[0])
print('Columns :',TestData.shape[1])
print('#################################')
#################################################################
sFileName=sOutputFileName
TestData.to_csv(sFileName, index = False)
#################################################################
print('#################################')
print('### Done!! ####################')
print('#################################')
#################################################################
```

Output:-

```
= RESTART: F:\DS_Practicals_Msc-it\Practical no 5\practicalno5A\practicalno5a.py
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5A\Good-or-Bad.csv
################################
## Raw Data Values
################################
      ID FieldA  FieldB FieldC  FieldD  FieldE    FieldF  FieldG
0    1.0   Good  Better   Best  1024.0     NaN   10241.0       1
1    2.0   Good     NaN   Best   512.0     NaN    5121.0       2
2    3.0   Good  Better    NaN   256.0     NaN     256.0       3
3    4.0   Good  Better   Best     NaN     NaN     211.0       4
4    5.0   Good  Better    NaN    64.0     NaN    6411.0       5
5    6.0   Good     NaN   Best    32.0     NaN      32.0       6
6    7.0    NaN  Better   Best    16.0     NaN    1611.0       7
7    8.0    NaN     NaN   Best     8.0     NaN    8111.0       8
8    9.0    NaN     NaN    NaN     4.0     NaN      41.0       9
9   10.0      A       B      C     2.0     NaN   21111.0      10
10   NaN    NaN     NaN    NaN     NaN     NaN       NaN      11
11  10.0   Good  Better   Best  1024.0     NaN  102411.0      12
12  10.0   Good     NaN   Best   512.0     NaN     512.0      13
13  10.0   Good  Better    NaN   256.0     NaN    1256.0      14
14  10.0   Good  Better   Best     NaN     NaN       NaN      15
15  10.0   Good  Better    NaN    64.0     NaN     164.0      16
16  10.0   Good     NaN   Best    32.0     NaN     322.0      17
17  10.0    NaN  Better   Best    16.0     NaN     163.0      18
18  10.0    NaN     NaN   Best     8.0     NaN     844.0      19
19  10.0    NaN     NaN    NaN     4.0     NaN    4555.0      20
20  10.0      A       B      C     2.0     NaN     111.0      21
################################
## Data Profile
################################
Rows : 21
Columns : 8
################################
################################
## Test Data Values
################################
      ID FieldA  FieldB FieldC  FieldD    FieldF  FieldG
0    1.0   Good  Better   Best  1024.0   10241.0       1
1    2.0   Good     NaN   Best   512.0    5121.0       2
2    3.0   Good  Better    NaN   256.0     256.0       3
3    4.0   Good  Better   Best     NaN     211.0       4
4    5.0   Good  Better    NaN    64.0    6411.0       5
5    6.0   Good     NaN   Best    32.0      32.0       6
6    7.0    NaN  Better   Best    16.0    1611.0       7
7    8.0    NaN     NaN   Best     8.0    8111.0       8
8    9.0    NaN     NaN    NaN     4.0      41.0       9
9   10.0      A       B      C     2.0   21111.0      10
10   NaN    NaN     NaN    NaN     NaN       NaN      11
11  10.0   Good  Better   Best  1024.0  102411.0      12
12  10.0   Good     NaN   Best   512.0     512.0      13
13  10.0   Good  Better    NaN   256.0    1256.0      14
14  10.0   Good  Better   Best     NaN       NaN      15
15  10.0   Good  Better    NaN    64.0     164.0      16
16  10.0   Good     NaN   Best    32.0     322.0      17
17  10.0    NaN  Better   Best    16.0     163.0      18
18  10.0    NaN     NaN   Best     8.0     844.0      19
19  10.0    NaN     NaN    NaN     4.0    4555.0      20
20  10.0      A       B      C     2.0     111.0      21
################################
## Data Profile
################################
Rows : 21
Columns : 7
################################
################################
### Done!! ####################
################################
```

2)ii. Drop the Columns Where Any of the Elements Is Missing Values

Code:

```
import
pandas as pd
sInputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5A\\Good-or-Bad.csv'
sOutputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5A\\Good-or-Bad-
01.csv'
sFileName=sInputFileName
print('Loading :',sFileName)
RawData=pd.read_csv(sFileName,header=0)
print(RawData) print('Rows
:',RawData.shape[0]) print('Columns
:',RawData.shape[1])
sFileName=sInputFileName
RawData.to_csv(sFileName, index = False)
TestData=RawData.dropna(axis=1, how='any')
print('Test Data Values')
print(TestData) print('Data Profile')
print('Rows :',TestData.shape[0])
print('Columns :',TestData.shape[1])
sFileName=sOutputFileName

print('Done')
```

Output:-

```
==================
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5A\Good-or-Bad.cs
      ID FieldA  FieldB FieldC  FieldD  FieldE    FieldF  FieldG
0    1.0   Good  Better   Best  1024.0     NaN   10241.0       1
1    2.0   Good     NaN   Best   512.0     NaN    5121.0       2
2    3.0   Good  Better    NaN   256.0     NaN     256.0       3
3    4.0   Good  Better   Best     NaN     NaN     211.0       4
4    5.0   Good  Better    NaN    64.0     NaN    6411.0       5
5    6.0   Good     NaN   Best    32.0     NaN      32.0       6
6    7.0    NaN  Better   Best    16.0     NaN    1611.0       7
7    8.0    NaN     NaN   Best     8.0     NaN    8111.0       8
8    9.0    NaN     NaN    NaN     4.0     NaN      41.0       9
9   10.0      A       B      C     2.0     NaN   21111.0      10
10   NaN    NaN     NaN    NaN     NaN     NaN       NaN      11
11  10.0   Good  Better   Best  1024.0     NaN  102411.0      12
12  10.0   Good     NaN   Best   512.0     NaN     512.0      13
13  10.0   Good  Better    NaN   256.0     NaN    1256.0      14
14  10.0   Good  Better   Best     NaN     NaN       NaN      15
15  10.0   Good  Better    NaN    64.0     NaN     164.0      16
16  10.0   Good     NaN   Best    32.0     NaN     322.0      17
17  10.0    NaN  Better   Best    16.0     NaN     163.0      18
18  10.0    NaN     NaN   Best     8.0     NaN     844.0      19
19  10.0    NaN     NaN    NaN     4.0     NaN    4555.0      20
20  10.0      A       B      C     2.0     NaN     111.0      21
Rows : 21
Columns : 8
```

```
Columns : 8
Test Data Values
    FieldG
0        1
1        2
2        3
3        4
4        5
5        6
6        7
7        8
8        9
9       10
10      11
11      12
12      13
13      14
14      15
15      16
16      17
17      18
18      19
19      20
20      21
Data Profile
Rows : 21
Columns : 1
Done
```

TestData.to_csv(sFileName, index = False)

3)iii. Keep Only the Rows That Contain a Maximum of Two Missing Values

Code:import pandas as pd

```
sInputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5A\\Good-or-Bad.csv'
sOutputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5A\\Good-or-Bad-01.csv'
sFileName=sInputFileName
print('Loading :',sFileName)
RawData=pd.read_csv(sFileName,header=0)
print('Raw Data Values') print(RawData)
print('Data Profile') print('Rows :',RawData.shape[0]) print('Columns :',RawData.shape[1])
sFileName=sInputFileName
RawData.to_csv(sFileName, index = False)
TestData=RawData.dropna(thresh=2)
print('## Test Data Values') print(TestData)
print('## Data Profile') print('Rows :',TestData.shape[0]) print('Columns :',TestData.shape[1])
sFileName=sOutputFileName
TestData.to_csv(sFileName, index = False)
print('Done')
```

Output:-

```
Data Profile
Rows : 21
Columns : 8
## Test Data Values
      ID FieldA  FieldB FieldC  FieldD  FieldE    FieldF  FieldG
0    1.0   Good  Better   Best  1024.0     NaN   10241.0       1
1    2.0   Good     NaN   Best   512.0     NaN    5121.0       2
2    3.0   Good  Better    NaN   256.0     NaN     256.0       3
3    4.0   Good  Better   Best     NaN     NaN     211.0       4
4    5.0   Good  Better    NaN    64.0     NaN    6411.0       5
5    6.0   Good     NaN   Best    32.0     NaN      32.0       6
6    7.0    NaN  Better   Best    16.0     NaN    1611.0       7
7    8.0    NaN     NaN   Best     8.0     NaN    8111.0       8
8    9.0    NaN     NaN    NaN     4.0     NaN      41.0       9
9   10.0      A       B      C     2.0     NaN   21111.0      10
11  10.0   Good  Better   Best  1024.0     NaN  102411.0      12
12  10.0   Good     NaN   Best   512.0     NaN     512.0      13
13  10.0   Good  Better    NaN   256.0     NaN    1256.0      14
14  10.0   Good  Better   Best     NaN     NaN       NaN      15
15  10.0   Good  Better    NaN    64.0     NaN     164.0      16
16  10.0   Good     NaN   Best    32.0     NaN     322.0      17
17  10.0    NaN  Better   Best    16.0     NaN     163.0      18
18  10.0    NaN     NaN   Best     8.0     NaN     844.0      19
19  10.0    NaN     NaN    NaN     4.0     NaN    4555.0      20
20  10.0      A       B      C     2.0     NaN     111.0      21
## Data Profile
Rows : 20
Columns : 8
Done
```

Practical no 4 B Aim:-4 Write python/R program to create the network routing diagram from the given data on routers

1) Assess-Network-Routing-Customer

Code:import
pandas as pd
pd.options.mode.chained_assignment = None
sInputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5B\\Assess-NetworkRouting-Customer.csv'
sOutputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5B\\Assess-NetworkRouting-Customer.gml' sFileName=sInputFileName print('Loading :',sFileName)
CustomerData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")

print('Loaded Country:',CustomerData.columns.values) print(CustomerData.head())

print('Done') Output;-

```
================ RESTART: F:\DS_Practicals_Msc-it\Practical no 5\practicalno5B\Practicalno5b_ Assess-N
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5B\Assess-Network-Routing-Customer.csv
Loaded Country: ['Customer_Country_Code' 'Customer_Place_Name' 'Customer_Latitude'
 'Customer_Longitude' 'Customer_Country_Name']
  Customer_Country_Code  ... Customer_Country_Name
0                    BW  ...               Botswana
1                    BW  ...               Botswana
2                    BW  ...               Botswana
3                    BW  ...               Botswana
4                    NE  ...                  Niger

[5 rows x 5 columns]
Done
```

	A	B	C	D	E	F	G
1	Customer_	Customer_	Customer_	Customer_	Customer_Country_Name		
2	BW	Gaborone	-24.6464	25.9119	Botswana		
3	BW	Francistowı	-21.1667	27.5167	Botswana		
4	BW	Maun	-19.9833	23.4167	Botswana		
5	BW	Molepolole	-24.4167	25.5333	Botswana		
6	NE	Niamey	13.5167	2.1167	Niger		

2) Assess-Network-Routing-Company

Cdoe:import
pandas as pd
pd.options.mode.chained_assignment = None
sInputFileName1='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5B\\Retrieve_Country_Code.csv'
sInputFileName2='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5B\\Retrieve_Router_Location.csv'
sInputFileName3='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5B\\Retrieve_IP_DATA (1).csv'

sOutputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5B\\Assess-NetworkRouting-Company.csv' sFileName=sInputFileName1 print('Loading :',sFileName)

```
CountryData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
print('Loaded Country:',CountryData.columns.values) print('Changed
:',CountryData.columns.values)
CountryData.rename(columns={'Country': 'Country_Name'}, inplace=True)
CountryData.rename(columns={'ISO-2-CODE': 'Country_Code'}, inplace=True)
CountryData.drop('ISO-M49', axis=1, inplace=True)
CountryData.drop('ISO-3-Code', axis=1, inplace=True)
CountryData.drop('RowID', axis=1, inplace=True)
print('To :',CountryData.columns.values)
sFileName=sInputFileName2 print('Loading
:',sFileName)
  CompanyData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
print('Loaded Company :',CompanyData.columns.values) print('Changed
:',CompanyData.columns.values)
CompanyData.rename(columns={'Country': 'Country_Code'}, inplace=True)
print('To :',CompanyData.columns.values)
sFileName=sInputFileName3
print('Loading :',sFileName)
CustomerRawData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
print('Loaded Customer :',CustomerRawData.columns.values)
CustomerData=CustomerRawData.dropna(axis=0, how='any') print('Remove Blank Country
Code')
print('Reduce Rows from', CustomerRawData.shape[0],' to ', CustomerData.shape[0])
print('Changed :',CustomerData.columns.values)
CustomerData.rename(columns={'Country': 'Country_Code'}, inplace=True)
print('To :',CustomerData.columns.values) print('Merge Company and
Country Data')
  CompanyNetworkData=pd.merge(
            CompanyData,
            CountryData,
            how='inner',
            on='Country_Code' ) print('Change
',CompanyNetworkData.columns.values) for i in
CompanyNetworkData.columns.values:
j='Company_'+i
      CompanyNetworkData.rename(columns={i: j}, inplace=True)
print('To ', CompanyNetworkData.columns.values)
sFileName=sOutputFileName print('Storing :', sFileName)
CompanyNetworkData.to_csv(sFileName, index = False, encoding="latin-1")
print('Done!!')
```

Output:-

```
================= RESTART: F:\DS_Practicals_Msc-it\Practical no 5\practicalno5B\Practicalno5b_Assess
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5B\Retrieve_Country_Code.csv
Loaded Country: ['RowID' 'Country' 'ISO-2-CODE' 'ISO-3-Code' 'ISO-M49']
Changed : ['RowID' 'Country' 'ISO-2-CODE' 'ISO-3-Code' 'ISO-M49']
To : ['Country_Name' 'Country_Code']
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5B\Retrieve_Router_Location.csv
Loaded Company : ['Country' 'Place_Name' 'Latitude' 'Longitude']
Changed : ['Country' 'Place_Name' 'Latitude' 'Longitude']
To : ['Country_Code' 'Place_Name' 'Latitude' 'Longitude']
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5B\Retrieve_IP_DATA (1).csv
Loaded Customer : ['RowID' 'Country' 'Place.Name' 'Post.Code' 'Latitude' 'Longitude']
Remove Blank Country Code
Reduce Rows from 1247502  to  1150323
Changed : ['RowID' 'Country' 'Place.Name' 'Post.Code' 'Latitude' 'Longitude']
To : ['RowID' 'Country_Code' 'Place.Name' 'Post.Code' 'Latitude' 'Longitude']
Merge Company and Country Data
Change  ['Country_Code' 'Place_Name' 'Latitude' 'Longitude' 'Country_Name']
To  ['Company_Country_Code' 'Company_Place_Name' 'Company_Latitude'
 'Company_Longitude' 'Company_Country_Name']
Storing : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5B\Assess-Network-Routing-Company.csv
Done!!
```

	A	B	C	D	E	F
1	Company_	Company_	Company_	Company_	Company_Country_Name	
2	US	New York	40.7528	-73.9725	United States of America	
3	US	New York	40.7214	-74.0052	United States of America	
4	US	New York	40.7662	-73.9862	United States of America	
5	US	New York	40.7449	-73.9782	United States of America	
6	US	New York	40.7605	-73.9933	United States of America	
7	US	New York	40.7588	-73.968	United States of America	

3) Assess-Network-Routing-Node

Code:import
pandas as pd
pd.options.mode.chained_assignment = None
sInputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\practicalno5B\Retrieve_IP_DATA.csv'
sOutputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\practicalno5B\Assess-NetworkRouting-Node.csv' sFileName=sInputFileName print('Loading :',sFileName)
IPData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
print('Loaded IP :', IPData.columns.values) print('Changed :',IPData.columns.values) IPData.drop('RowID', axis=1, inplace=True)
IPData.drop('ID', axis=1, inplace=True)
IPData.rename(columns={'Country': 'Country_Code'}, inplace=True)
IPData.rename(columns={'Place.Name': 'Place_Name'}, inplace=True)
IPData.rename(columns={'Post.Code': 'Post_Code'}, inplace=True)
IPData.rename(columns={'First.IP.Number': 'First_IP_Number'}, inplace=True)
IPData.rename(columns={'Last.IP.Number': 'Last_IP_Number'}, inplace=True)
print('To :',IPData.columns.values) print('Change ',IPData.columns.values) for i in IPData.columns.values:
    j='Node_'+i

```
    IPData.rename(columns={i: j}, inplace=True)
print('To ', IPData.columns.values)
sFileName=sOutputFileName print('Storing :',
sFileName)
IPData.to_csv(sFileName, index = False, encoding="latin-1")
print('Done')
```

Output:-

```
================== RESTART: F:\DS_Practicals_Msc-it\Practical no 5\practicalno5B\Practicalno5b_As:
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5B\Retrieve_IP_DATA.csv
Loaded IP : ['RowID' 'ID' 'Country' 'Place.Name' 'Post.Code' 'Latitude' 'Longitude'
 'First.IP.Number' 'Last.IP.Number']
Changed : ['RowID' 'ID' 'Country' 'Place.Name' 'Post.Code' 'Latitude' 'Longitude'
 'First.IP.Number' 'Last.IP.Number']
To : ['Country_Code' 'Place_Name' 'Post_Code' 'Latitude' 'Longitude'
 'First_IP_Number' 'Last_IP_Number']
Change  ['Country_Code' 'Place_Name' 'Post_Code' 'Latitude' 'Longitude'
 'First_IP_Number' 'Last_IP_Number']
To  ['Node_Country_Code' 'Node_Place_Name' 'Node_Post_Code' 'Node_Latitude'
 'Node_Longitude' 'Node_First_IP_Number' 'Node_Last_IP_Number']
Storing : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5B\Assess-Network-Routing-Node.csv
Done
```

	A	B	C	D	E	F	G	H
1	Node_Cou	Node_Plac	Node_Post	Node_Latit	Node_Long	Node_First	Node_Last_IP_Number	
2	BW	Gaborone		-24.6464	25.9119	692781056	692781567	
3	BW	Gaborone		-24.6464	25.9119	692781824	692783103	
4	BW	Gaborone		-24.6464	25.9119	692909056	692909311	
5	BW	Gaborone		-24.6464	25.9119	692909568	692910079	

Practical no 4C

AIm:-Write a Python / R program to build directed acyclic graph.

1) Assess_DAG_GPS

```
Code:import networkx as nx
import matplotlib.pyplot as plt
import pandas as pd import os
sInputFileName = 'F:\\DS_Practicals_Msc-it\\Practical no 5\\Practicalno5C\\Retrieve_Router_Location.csv'
sOutputFileName = 'F:\\DS_Practicals_Msc-it\\Practical no 5\\Practicalno5C\\Assess-DAGCompany-GPS.png' if not os.path.exists(sInputFileName):
    print(f"Error: Input file '{sInputFileName}' not found!")
    exit()
print(f"Loading data from: {sInputFileName}")
CompanyData = pd.read_csv(sInputFileName, header=0, encoding="latin-1")
required_columns = ['Latitude', 'Longitude'] if not all(col in CompanyData.columns for col in required_columns):
    print(f"Error: Input file must contain the columns: {required_columns}")
    exit()
CompanyData['Latitude'] = CompanyData['Latitude'].round(1)
CompanyData['Longitude'] = CompanyData['Longitude'].round(1)

def format_coord(lat, lon):
    lat_dir = 'S' if lat < 0 else 'N'
    lon_dir = 'W' if lon < 0 else 'E'
```

```
        return f"{abs(lat)} {lat_dir}-{abs(lon)} {lon_dir}"
CompanyData['GPS'] =        CompanyData.apply(lambda  row:
format_coord(row['Latitude'], row['Longitude']), axis=1) G = nx.Graph()
nodes = CompanyData['GPS'].unique()
G.add_nodes_from(nodes) for i, node1
in enumerate(nodes):
        for node2 in nodes[i + 1:]:
                G.add_edge(node1, node2)
print(f"Nodes: {len(G.nodes())}")
print(f"Edges: {len(G.edges())}")
print(f"Saving graph to: {sOutputFileName}")
pos = nx.circular_layout(G) nx.draw(G, pos,
                node_color='red', edge_color='blue',
                with_labels=True,   node_size=3000,
font_size=10)
plt.savefig(sOutputFileName) plt.show()
```

Output:-

```
Loading data from: F:\DS_Prac
_Router_Location.csv
Nodes: 16
Edges: 120
Saving graph to: F:\DS_Practi
-Company-GPS.png
```

2) Practicalno5c_Assess_DAG_Location

Code:
```
import networkx as nx
import matplotlib.pyplot as plt
import pandas as pd
sInputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\\Practicalno5C\\Retrieve_Router_Location.csv'
sOutputFileName1='F:\\DS_Practicals_Msc-it\\Practical no 5\\Practicalno5C\\Assess-DAGCompany-Country.png'
sOutputFileName2='F:\\DS_Practicals_Msc-it\\Practical no 5\\Practicalno5C\\Assess-DAG-Company-Country-Place.png'
sFileName=sInputFileName
print('Loading :',sFileName)
CompanyData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
print('Loaded Company :',CompanyData.columns.values) print(CompanyData)
print('Rows : ',CompanyData.shape[0])
G1=nx.DiGraph() G2=nx.DiGraph() for i in
range(CompanyData.shape[0]):
G1.add_node(CompanyData['Country'][i])
    sPlaceName= CompanyData['Place_Name'][i] + '-' + CompanyData['Country'][i]
    G2.add_node(sPlaceName)
for n1 in G1.nodes():
    for n2 in G1.nodes():
        if n1 != n2:
            print('Link :',n1,' to ', n2)
            G1.add_edge(n1,n2)
print("Nodes of graph: ")
print(G1.nodes()) print("Edges
of graph: ") print(G1.edges())
sFileName=sOutputFileName1
  print('Storing :', sFileName)
  nx.draw(G1,pos=nx.spectral_layout(G1),
        node_color='r',edge_color='g',
        with_labels=True,node_size=8000,
        font_size=12)
plt.savefig(sFileName)
plt.show() for n1 in
G2.nodes():
    for n2 in G2.nodes():
        if n1 != n2:
            print('Link :',n1,' to ', n2)
            G2.add_edge(n1,n2)
print("Nodes of graph: ")
print(G2.nodes())
print("Edges of graph: ")
print(G2.edges())
  sFileName=sOutputFileName2
```

```
  print('Storing :', sFileName)
  nx.draw(G2,pos=nx.spectral_layout(G2),
  node_color='r',edge_color='b',
  with_labels=True,node_size=8000, font_size=12)
plt.savefig(sFileName)
plt.show()
```

OUtput:-

```
= RESTART: F:\DS_Practicals_Msc-it\Practical no 5\Practicalno5C\Practicalno5c_Assess_DAG_Location.py
Loading : F:\DS_Practicals_Msc-it\Practical no 5\Practicalno5C\Retrieve_Router_Location.csv
Loaded Company : ['Country' 'Place_Name' 'Latitude' 'Longitude']
    Country Place_Name  Latitude  Longitude
0        US   New York   40.7528   -73.9725
1        US   New York   40.7214   -74.0052
2        US   New York   40.7662   -73.9862
3        US   New York   40.7449   -73.9782
4        US   New York   40.7605   -73.9933
..      ...        ...       ...        ...
145      DE     Munich   48.1475    11.4635
146      DE     Munich   48.0915    11.5392
147      DE     Munich   48.1833    11.7500
148      DE     Munich   48.1000    11.4667
149      DE     Munich   48.1480    11.7434

[150 rows x 4 columns]
Rows :  150
Link : US  to  DE
Link : US  to  GB
Link : DE  to  US
Link : DE  to  GB
Link : GB  to  US
Link : GB  to  DE
Nodes of graph:
['US', 'DE', 'GB']
Edges of graph:
[('US', 'DE'), ('US', 'GB'), ('DE', 'US'), ('DE', 'GB'), ('GB', 'US'), ('GB', 'DE')]
Storing : F:\DS_Practicals_Msc-it\Practical no 5\Practicalno5C\Assess-DAG-Company-Country.png
```

Practical no 4 D
Aim:-Write a Python / R program to pick the content for Bill Boards from the given data.

Code:-

```
import sqlite3 as sq
import pandas as pd
sInputFileName1='F:\\DS_Practicals_Msc-it\\Practical no
5\\Practicalno5D\\Retrieve_DE_Billboard_Locations.csv'
sInputFileName2='F:\\DS_Practicals_Msc-it\\Practical no
5\\Practicalno5D\\Retrieve_Online_Visitor.csv'
sOutputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\\Practicalno5D\\Assess-
DEBillboard-Visitor.csv'
sDatabaseName='F:\\DS_Practicals_Msc-it\\Practical no 5\\Practicalno5D\\krennwallner.db'
conn = sq.connect(sDatabaseName) sFileName=sInputFileName1 print('Loading
:',sFileName)
BillboardRawData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
BillboardRawData.drop_duplicates(subset=None, keep='first', inplace=True)
BillboardData=BillboardRawData
print('Loaded Company :',BillboardData.columns.values)
sTable='Assess_BillboardData' print('Storing
:',sDatabaseName,' Table:',sTable)
BillboardData.to_sql(sTable, conn, if_exists="replace")
print(BillboardData.head()) print('Rows :
',BillboardData.shape[0]) sFileName=sInputFileName2
print('Loading :',sFileName)
VisitorRawData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
VisitorRawData.drop_duplicates(subset=None, keep='first', inplace=True)
VisitorData=VisitorRawData[VisitorRawData.Country=='DE']
print('Loaded Company :',VisitorData.columns.values)
sTable='Assess_VisitorData' print('Storing :',sDatabaseName,'
Table:',sTable) VisitorData.to_sql(sTable, conn,
if_exists="replace") print(VisitorData.head()) print('Rows :
',VisitorData.shape[0]) sTable='Assess_BillboardVisitorData'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="select distinct"
sSQL=sSQL+ " A.Country AS BillboardCountry,"
sSQL=sSQL+ " A.Place_Name AS BillboardPlaceName,"
sSQL=sSQL+ " A.Latitude AS BillboardLatitude, "
sSQL=sSQL+ " A.Longitude AS BillboardLongitude,"
sSQL=sSQL+ " B.Country AS VisitorCountry,"
sSQL=sSQL+ " B.Place_Name AS VisitorPlaceName,"
sSQL=sSQL+ " B.Latitude AS VisitorLatitude, "
sSQL=sSQL+ " B.Longitude AS VisitorLongitude,"
sSQL=sSQL+ "  (B.Last_IP_Number  -
B.First_IP_Number)  *  365.25 *  24  *
12  AS
```

```
VisitorYearRate"
sSQL=sSQL+ " from"
sSQL=sSQL+ " Assess_BillboardData as A"
sSQL=sSQL+ " JOIN "
sSQL=sSQL+ " Assess_VisitorData as B"
sSQL=sSQL+ " ON "
sSQL=sSQL+ " A.Country = B.Country"
sSQL=sSQL+ " AND "
sSQL=sSQL+ " A.Place_Name = B.Place_Name;"
BillboardVistorsData=pd.read_sql_query(sSQL, conn)
sTable='Assess_BillboardVistorsData' print('Storing
:',sDatabaseName,' Table:',sTable)
BillboardVistorsData.to_sql(sTable, conn, if_exists="replace")
print(BillboardVistorsData.head()) print('Rows :
',BillboardVistorsData.shape[0]) print('Storing :', sFileName)
sFileName=sOutputFileName
BillboardVistorsData.to_csv(sFileName, index = False)
print('Done')
```

Output:-

```
================= RESTART: F:\DS_Practicals_Msc-it\Practical no 5\Practicalno5D\Practicalno5d_Picking_Content_for_Bill
Loading : F:\DS_Practicals_Msc-it\Practical no 5\Practicalno5D\Retrieve_DE_Billboard_Locations.csv
Loaded Company : ['Country' 'Place_Name' 'Latitude' 'Longitude']
Storing : F:\DS_Practicals_Msc-it\Practical no 5\Practicalno5D\krennwallner.db  Table: Assess_BillboardData
  Country         Place_Name  Latitude  Longitude
0      DE               Lake   51.7833     8.5667
1      DE               Horb   48.4333     8.6833
2      DE         Hardenberg   51.1000     7.7333
3      DE  Horn-bad Meinberg   51.9833     8.9667
4      DE             Winkel   51.5500    13.3833
Rows :  8873
Loading : F:\DS_Practicals_Msc-it\Practical no 5\Practicalno5D\Retrieve_Online_Visitor.csv
Loaded Company : ['Country' 'Place_Name' 'Latitude' 'Longitude' 'First_IP_Number'
 'Last_IP_Number']
Storing : F:\DS_Practicals_Msc-it\Practical no 5\Practicalno5D\krennwallner.db  Table: Assess_VisitorData
       Country Place_Name  Latitude  Longitude  First_IP_Number  Last_IP_Number
368678      DE       Lake   51.7833     8.5667       1418011904      1418012159
368679      DE       Lake   51.7833     8.5667       1528171008      1528171263
368680      DE       Lake   51.7833     8.5667       1528177920      1528178175
408664      DE       Horb   48.4333     8.6833        777246720       777246975
408665      DE       Horb   48.4333     8.6833       1339562496      1339562751
Rows :  75999
Loading : F:\DS_Practicals_Msc-it\Practical no 5\Practicalno5D\krennwallner.db  Table: Assess_BillboardVisitorData
Storing : F:\DS_Practicals_Msc-it\Practical no 5\Practicalno5D\krennwallner.db  Table: Assess_BillboardVistorsData
  BillboardCountry BillboardPlaceName  ...  VisitorLongitude  VisitorYearRate
0               DE               Lake  ...            8.5667       26823960.0
1               DE               Horb  ...            8.6833       26823960.0
2               DE               Horb  ...            8.6833       53753112.0
3               DE               Horb  ...            8.6833      107611416.0
4               DE               Horb  ...            8.6833       13359384.0

[5 rows x 9 columns]
Rows :  181235
Storing : F:\DS_Practicals_Msc-it\Practical no 5\Practicalno5D\Retrieve_Online_Visitor.csv
Done
```

	A	B	C	D	E	F	G	H	I	J
1	BillboardCo	BillboardPl	BillboardLa	BillboardLo	VisitorCour	VisitorPlace	VisitorLatit	VisitorLong	VisitorYearRate	
2	DE	Lake	51.7833	8.5667	DE	Lake	51.7833	8.5667	26823960	
3	DE	Horb	48.4333	8.6833	DE	Horb	48.4333	8.6833	26823960	
4	DE	Horb	48.4333	8.6833	DE	Horb	48.4333	8.6833	53753112	
5	DE	Horb	48.4333	8.6833	DE	Horb	48.4333	8.6833	107611416	
6	DE	Horb	48.4333	8.6833	DE	Horb	48.4333	8.6833	13359384	
7	DE	Horb	48.4333	8.6833	DE	Horb	48.4889	8.6734	26823960	

Practical no 4E Aim:-Write a python/R program to generate GML file from given csv file

1)Understanding Your Online Visitor Data

Code:
```
import networkx as nx
import sqlite3 as sq import pandas as
pd from geopy.distance import
geodesic

sTable='Assess_BillboardVisitorData'
sOutputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\\Practicalno5E\\Assess-DEBillboard-Visitor.gml'

sDatabaseName='F:\\DS_Practicals_Msc-it\\Practical no 5\\Practicalno5E\\krennwallner.db'
conn = sq.connect(sDatabaseName)
#################################################################
print('################')
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="select " sSQL=sSQL+ " A.BillboardCountry," sSQL=sSQL+ "
A.BillboardPlaceName," sSQL=sSQL+ " ROUND(A.BillboardLatitude,3)
AS BillboardLatitude, " sSQL=sSQL+ " ROUND(A.BillboardLongitude,3)
AS BillboardLongitude,"

sSQL=sSQL+ " (CASE WHEN A.BillboardLatitude < 0 THEN "
sSQL=sSQL+ " 'S' || ROUND(ABS(A.BillboardLatitude),3)" sSQL=sSQL+
" ELSE "
sSQL=sSQL+ " 'N' || ROUND(ABS(A.BillboardLatitude),3)" sSQL=sSQL+
" END ) AS sBillboardLatitude,"

sSQL=sSQL+ " (CASE WHEN A.BillboardLongitude < 0 THEN "
sSQL=sSQL+  "  'W'  ||  ROUND(ABS(A.BillboardLongitude),3)"
sSQL=sSQL+ " ELSE "
sSQL=sSQL+ " 'E' || ROUND(ABS(A.BillboardLongitude),3)" sSQL=sSQL+
" END ) AS sBillboardLongitude,"

sSQL=sSQL+ " A.VisitorCountry," sSQL=sSQL+ "
A.VisitorPlaceName," sSQL=sSQL+ " ROUND(A.VisitorLatitude,3)
AS VisitorLatitude, " sSQL=sSQL+ " ROUND(A.VisitorLongitude,3)
AS VisitorLongitude,"

sSQL=sSQL+ " (CASE WHEN A.VisitorLatitude < 0 THEN "
sSQL=sSQL+ " 'S' || ROUND(ABS(A.VisitorLatitude),3)" sSQL=sSQL+
" ELSE "
sSQL=sSQL+  "  'N'  ||ROUND(ABS(A.VisitorLatitude),3)"

sSQL=sSQL+  "  END  )  AS  sVisitorLatitude,"  sSQL=sSQL+  "

(CASE WHEN A.VisitorLongitude < 0 THEN " sSQL=sSQL+ "

'W'  ||  ROUND(ABS(A.VisitorLongitude),3)"  sSQL=sSQL+  "

ELSE "
```

```
sSQL=sSQL+ " 'E' || ROUND(ABS(A.VisitorLongitude),3)" sSQL=sSQL+
" END ) AS sVisitorLongitude,"

sSQL=sSQL+ " A.VisitorYearRate" sSQL=sSQL+ "
from" sSQL=sSQL+ " Assess_BillboardVistorsData
AS A;"
BillboardVistorsData=pd.read_sql_query(sSQL, conn)
print('################')

#################################################################
BillboardVistorsData['Distance']=BillboardVistorsData.apply(lambda row:
round(
        geodesic((row['BillboardLatitude'],row['BillboardLongitude']),
                        (row['VisitorLatitude'],row['VisitorLongitude'])).miles
                ,4)
            ,axis=1)
#################################################################
G=nx.Graph()
#################################################################

for i in range(BillboardVistorsData.shape[0]):
        sNode0='MediaHub-' + BillboardVistorsData['BillboardCountry'][i]

        sNode1='B-'+ BillboardVistorsData['sBillboardLatitude'][i] + '-'
        sNode1=sNode1 + BillboardVistorsData['sBillboardLongitude'][i]
        G.add_node(sNode1,
                        Nodetype='Billboard',
                        Country=BillboardVistorsData['BillboardCountry'][i],
                        PlaceName=BillboardVistorsData['BillboardPlaceName'][i],
                        Latitude=round(BillboardVistorsData['BillboardLatitude'][i],3),
                        Longitude=round(BillboardVistorsData['BillboardLongitude'][i],3))

        sNode2='M-'+ BillboardVistorsData['sVisitorLatitude'][i] + '-'
        sNode2=sNode2 + BillboardVistorsData['sVisitorLongitude'][i]
        G.add_node(sNode2,
                        Nodetype='Mobile',
                        Country=BillboardVistorsData['VisitorCountry'][i],
                        PlaceName=BillboardVistorsData['VisitorPlaceName'][i],
                        Latitude=round(BillboardVistorsData['VisitorLatitude'][i],3),
                        Longitude=round(BillboardVistorsData['VisitorLongitude'][i],3))

        print('Link Media Hub :',sNode0,' to Billboard : ', sNode1)
        G.add_edge(sNode0,sNode1)

        print('Link Post Code :',sNode1,' to GPS : ', sNode2)
        G.add_edge(sNode1,sNode2,distance=round(BillboardVistorsData['Distance'][i]))
```

```
#################################################################
print('#################################') print("Nodes of graph:
",nx.number_of_nodes(G)) print("Edges of graph: ",nx.number_of_edges(G))
print('#################################') sFileName=sOutputFileName
print('#################################')
print('Storing :', sFileName)
print('#################################')
nx.write_gml(G,sFileName)
sFileName=sFileName +'.gz'
nx.write_gml(G,sFileName)
#################################################################
#################################################################
print('### Done!! ##############################################')
#################################################################
```

Output:-: This will produce a set of demonstrated values onscreen, plus a graph data file named Assess-DE-Billboard-Visitor.gml. (It takes a long time to complete the process, after completion the gml file can be viewed in text editor)

2)Planning an Event for Top-Ten Customers

Code:-

```
import sqlite3 as sq import
pandas as pd from pandas.io
import sql Company='02-
Krennwallner'
sInputFileName='F:\\DS_Practicals_Msc-it\\Practical no
5\\Practicalno5E\\Retrieve_Online_Visitor.csv'
sDatabaseName='F:\\DS_Practicals_Msc-it\\Practical no 5\\Practicalno5E\\krennwallner.db'
conn = sq.connect(sDatabaseName) sFileName=sInputFileName print('Loading
:',sFileName)
VisitorRawData=pd.read_csv(sFileName, header=0,
                           low_memory=False,
                           encoding="latin-1",
                           skip_blank_lines=True)
VisitorRawData.drop_duplicates(subset=None, keep='first', inplace=True)
VisitorData=VisitorRawData
print('Loaded Company :',VisitorData.columns.values)
sTable='Assess_Visitor' print('Storing
:',sDatabaseName,' Table:',sTable)
VisitorData.to_sql(sTable, conn, if_exists="replace")
print(VisitorData.head())

print('Rows : ',VisitorData.shape[0])
sView='Assess_Visitor_UseIt' print('Creating
:',sDatabaseName,' View:',sView)
sSQL="DROP VIEW IF EXISTS " + sView + ";"
sql.execute(sSQL,conn)
```

```
sSQL="CREATE VIEW " + sView + " AS"
sSQL=sSQL+ " SELECT" sSQL=sSQL+ "
A.Country," sSQL=sSQL+ "
A.Place_Name," sSQL=sSQL+ "
A.Latitude," sSQL=sSQL+ " A.Longitude,"
sSQL=sSQL+ " (A.Last_IP_Number - A.First_IP_Number) AS UsesIt"
sSQL=sSQL+ " FROM" sSQL=sSQL+ "
Assess_Visitor as A" sSQL=sSQL+ "
WHERE" sSQL=sSQL+ " Country is not
null" sSQL=sSQL+ " AND"
sSQL=sSQL+ " Place_Name is not null;"
sql.execute(sSQL,conn)
  #

sView='Assess_Total_Visitors_Location'
print('Creating :',sDatabaseName,' View:',sView)
sSQL="DROP VIEW IF EXISTS " + sView + ";"
sql.execute(sSQL,conn)

sSQL="CREATE VIEW " + sView + " AS"
sSQL=sSQL+ " SELECT" sSQL=sSQL+ "
Country," sSQL=sSQL+ " Place_Name,"
sSQL=sSQL+ " SUM(UsesIt) AS TotalUsesIt"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ " Assess_Visitor_UseIt"
sSQL=sSQL+ " GROUP BY"
sSQL=sSQL+ " Country,"
sSQL=sSQL+ " Place_Name"
sSQL=sSQL+ " ORDER BY"
sSQL=sSQL+ " TotalUsesIt DESC"
sSQL=sSQL+ " LIMIT 10;"
sql.execute(sSQL,conn)
  #

sView='Assess_Total_Visitors_GPS'
print('Creating :',sDatabaseName,' View:',sView)
sSQL="DROP VIEW IF EXISTS " + sView + ";"
sql.execute(sSQL,conn)

sSQL="CREATE VIEW " + sView + " AS"
sSQL=sSQL+ " SELECT" sSQL=sSQL+ "
Latitude," sSQL=sSQL+ " Longitude,"
sSQL=sSQL+ " SUM(UsesIt) AS TotalUsesIt"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ " Assess_Visitor_UseIt"
sSQL=sSQL+ " GROUP BY"
sSQL=sSQL+ " Latitude,"
sSQL=sSQL+ " Longitude"
```

```
sSQL=sSQL+ " ORDER BY"
sSQL=sSQL+ " TotalUsesIt DESC"
sSQL=sSQL+ " LIMIT 10;"
sql.execute(sSQL,conn)
  # sTables=['Assess_Total_Visitors_Location',
'Assess_Total_Visitors_GPS'] for sTable in sTables:

        print('Loading :',sDatabaseName,' Table:',sTable)
        sSQL=" SELECT "
        sSQL=sSQL+ " *"
        sSQL=sSQL+ " FROM"
        sSQL=sSQL+ " " + sTable + ";"
        TopData=pd.read_sql_query(sSQL, conn)

        print(TopData)


        print('Rows : ',TopData.shape[0])


print('Done')
```

OUtput:-

Practical no 4 F

Aim:-Write a Python / R program to plan the locations of the warehouses

```
Loading : F:\DS_Practicals_Msc-it\Practical no 5\Practicalno5E\krennwallner.db  Table: Assess_Tot
  Country     Place_Name  TotalUsesIt
0      CN        Beijing     53139475
1      US      Palo Alto     33682341
2      US  Fort Huachuca     33472427
3      JP          Tokyo     31404799
4      US      Cambridge     25598851
5      US      San Diego     17751367
6      CN      Guangzhou     17563744
7      US         Newark     17270604
8      US        Raleigh     17167484
9      US         Durham     16914033
Rows :  10
Loading : F:\DS_Practicals_Msc-it\Practical no 5\Practicalno5E\krennwallner.db  Table: Assess_Tot
   Latitude  Longitude  TotalUsesIt
0   39.9289   116.3883     53139732
1   37.3762  -122.1826     33551404
2   31.5273  -110.3607     33472427
3   35.6427   139.7677     31439772
4   23.1167   113.2500     17577053
5   42.3646   -71.1028     16890698
6   40.7355   -74.1741     16813373
7   42.3223   -83.1763     16777212
8   35.7977   -78.6253     16761084
9   32.8072  -117.1649     16747680
Rows :  10
Done
```

from the given data.

Code:-

```
import pandas as pd
from geopy.geocoders import Nominatim
geolocator = Nominatim(user_agent="my_custom_application")
InputFileName = 'F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5F\\Retrieve_GB_Postcode_Warehouse.csv'
OutputFileName = 'F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5F\\Assess_GB_Warehouse_Address.csv'
sFileName = InputFileName
print('Loading :', sFileName)
Warehouse = pd.read_csv(sFileName, header=0, low_memory=False)
Warehouse.sort_values(by='postcode', ascending=1)
WarehouseGoodHead = Warehouse[Warehouse.latitude != 0].head(5)
WarehouseGoodTail = Warehouse[Warehouse.latitude != 0].tail(5)
WarehouseGoodHead['Warehouse_Point'] = WarehouseGoodHead.apply(
    lambda row: (str(row['latitude']) + ',' + str(row['longitude'])), axis=1
)
WarehouseGoodHead['Warehouse_Address'] = WarehouseGoodHead.apply(
    lambda row: geolocator.reverse(row['Warehouse_Point']).address,
    axis=1
)
WarehouseGoodHead.drop('Warehouse_Point', axis=1, inplace=True)
WarehouseGoodHead.drop('id', axis=1, inplace=True)
WarehouseGoodHead.drop('postcode', axis=1, inplace=True)

WarehouseGoodTail['Warehouse_Point'] = WarehouseGoodTail.apply(
    lambda row: (str(row['latitude']) + ',' + str(row['longitude'])),
    axis=1
)
WarehouseGoodTail['Warehouse_Address'] = WarehouseGoodTail.apply(
    lambda row: geolocator.reverse(row['Warehouse_Point']).address,
    axis=1
)
WarehouseGoodTail.drop('Warehouse_Point', axis=1, inplace=True)
WarehouseGoodTail.drop('id', axis=1, inplace=True)
WarehouseGoodTail.drop('postcode', axis=1, inplace=True)
WarehouseGood = pd.concat([WarehouseGoodHead, WarehouseGoodTail],
ignore_index=True) print(WarehouseGood) sFileName = OutputFileName
WarehouseGood.to_csv(sFileName, index=False)
print('Done')
```

```
===================== RESTART: F:\DS_Practicals_Msc-it\Practical no 5\practic
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5F\Retrieve_GB_P
    latitude  longitude                                Warehouse_Address
0  57.135140  -2.117310  35, Broomhill Road, Broomhill, Ashley and Broo...
1  57.138750  -2.090890  South Esplanade West, Tullos, Torry, Aberdeen ...
2  57.101000  -2.110600  A92, Charlestown, Nigg, Aberdeen City, Alba / ...
3  57.108010  -2.237760  Colthill Circle, Cults, Bieldside and Milltimb...
4  57.100760  -2.270730  Johnston Gardens East, Culter, Peterculter, Ab...
5  53.837717  -1.780013  HM Revenue and Customs, Riverside Estate, Ship...
6  53.794470  -1.766539  Listerhills Road, Great Horton, Bradford, West...
7  51.518556  -0.714794  Howarth Road, Stafferton Way Industrial Area, ...
8  54.890923  -2.943847  Carlisle Delivery Office, Junction Street, Den...
9  57.481338  -4.223951  Railway Terrace, Longman, Crown and City Centr...
Done
```

Output:-

latitude	longitude	Warehouse_Address
57.13514	-2.11731	35, Broomhill Road, Broomhill, Ashley and Broomhill, Aberdeen City, Alba / Scotland, AB10 6JA, United Kingdom
57.13875	-2.09089	South Esplanade West, Tullos, Torry, Aberdeen City, Alba / Scotland, AB11 9AE, United Kingdom
57.101	-2.1106	A92, Charlestown, Nigg, Aberdeen City, Alba / Scotland, AB12 3ZT, United Kingdom
57.10801	-2.23776	Colthill Circle, Cults, Bieldside and Milltimber, Milltimber, Aberdeen City, Alba / Scotland, AB13 0EH, United Kingdom
57.10076	-2.27073	Johnston Gardens East, Culter, Peterculter, Aberdeen City, Alba / Scotland, AB14 0LA, United Kingdom
53.837717	-1.780013	HM Revenue and Customs, Riverside Estate, Shipley, Baildon, Bradford, West Yorkshire, England, BD98 8AA, United Kingdom
53.79447	-1.766539	Listerhills Road, Great Horton, Bradford, West Yorkshire, England, BD7 1HZ, United Kingdom
51.518556	-0.714794	Howarth Road, Stafferton Way Industrial Area, Fishery, Maidenhead, Windsor and Maidenhead, England, SL6 1WT, United Kingdom
54.890923	-2.943847	Carlisle Delivery Office, Junction Street, Denton Holme, Carlisle, Cumberland, England, CA1 1AA, United Kingdom
57.481338	-4.223951	Railway Terrace, Longman, Crown and City Centre, Inverness, Highland, Alba / Scotland, IV99 1AA, United Kingdom

Practical no 4 G

Aim:-Write a Python / R program using data science via clustering to determine new warehouses using the given data. .

Code:import sys
import os import
pandas as pd
InputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5G\\Retrieve_All_Countries.csv'
OutputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5G\\Assess_All_Warehouse.csv'
sFileName=InputFileName
print('###########')
print('Loading :',sFileName)
Warehouse=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")

```
sColumns={'X1' : 'Country',
          'X2' : 'PostCode',
          'X3' : 'PlaceName',
          'X4' : 'AreaName',
          'X5' : 'AreaCode',
          'X10' : 'Latitude',
          'X11' : 'Longitude'}
Warehouse.rename(columns=sColumns,inplace=True)
WarehouseGood=Warehouse

sFileName=OutputFileName
WarehouseGood.to_csv(sFileName, index = False)
print('Done') Output:-
```

Home | ds prs sanjeev.docx | Assess_All_Warehouse.csv

Menu | Home | Insert | Page Layout | Formulas | Dat

L7 | fx

	A	B	C	D	E	F	G	H	I
4	3	AD	AD300	Ordino			42.6	1.55	
5	4	AD	AD400	La Massana			42.5667	1.4833	
6	5	AD	AD500	Andorra la Vella			42.5	1.5	
7	6	AD	AD600	Sant Julià de Lòria			42.4667	1.5	
8	7	AD	AD700	Escaldes-Engordany			42.5	1.5667	
9	8	AR	3636	POZO CERC	Salta	A	-23.4933	-61.9267	
10	9	AR	4123	LAS SALADA	Salta	A	-25.7833	-64.5	
11	10	AR	4126	EL BRETE	Salta	A	-26.0667	-65.3667	
12	11	AR	4126	OVEJERO	Salta	A	-26.0833	-65.263	
13	12	AR	4126	MIRAFLORE	Salta	A	-26.0833	-65.263	
14	13	AR	4126	LA MARAVI	Salta	A	-26.0833	-65.263	
15	14	AR	4126	LA ASUNCI(	Salta	A	-26.0833	-65.263	
16	15	AR	4126	BARADERO	Salta	A	-26.0833	-65.263	
17	16	AR	4126	EL CUIBAL	Salta	A	-26.0833	-65.263	
18	17	AR	4126	CEIBAL	Salta	A	-26.1	-65.0167	
19	18	AR	4126	ALEM	Salta	A	-26.0833	-65.263	
20	19	AR	4126	EL SUNCHA	Salta	A	-26.0833	-65.263	
21	20	AR	4126	TALA	Salta	A	-26.1167	-65.2833	

Practical no 4 H

AIm:-Using the given data, write a Python / R program to plan the shipping routes for best-fit international logistics.

Code:-

```
import pandas as pd
import networkx as nx
from geopy.distance import
geodesic import sqlite3 as sq from
pandas.io import sql
InputFileName='F:\\DS_Practicals_Msc-it\\Practical no
5\\practicalno5H\\Retrieve_All_Countries.csv'
OutputFileName='F:\\DS_Practicals_Msc-it\\Practical no
5\\practicalno5H\\Assess_Best_Logistics.gml'
sDatabaseName='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5H\\Hillman.db'
conn = sq.connect(sDatabaseName)
##################################################################
sFileName=InputFileName print('###########') print('Loading :',sFileName)
Warehouse=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
##################################################################
sColumns={'X1' : 'Country',
          'X2' : 'PostCode',
          'X3' : 'PlaceName',
          'X4' : 'AreaName',
          'X5' : 'AreaCode',
          'X10' : 'Latitude',
          'X11' : 'Longitude'}
Warehouse.rename(columns=sColumns,inplace=True)
WarehouseGood=Warehouse
#print(WarehouseGood.head())
##################################################################
RoutePointsCountry=pd.DataFrame(WarehouseGood.groupby(['Country'])[['Latitude','Longitud
e']].mean())
#print(RoutePointsCountry.head())
print('################')
sTable='Assess_RoutePointsCountry'
print('Storing :',sDatabaseName,' Table:',sTable)
RoutePointsCountry.to_sql(sTable, conn, if_exists="replace")
print('################')
##################################################################
RoutePointsPostCode=pd.DataFrame(WarehouseGood.groupby(['Country',
'PostCode'])[['Latitude','Longitude']].mean())
#print(RoutePointsPostCode.head())
print('################')
sTable='Assess_RoutePointsPostCode'
print('Storing :',sDatabaseName,' Table:',sTable)
```

```
RoutePointsPostCode.to_sql(sTable, conn, if_exists="replace")
print('################')
#################################################################
RoutePointsPlaceName=pd.DataFrame(WarehouseGood.groupby(['Country',
'PostCode','PlaceName'])[['Latitude','Longitude']].mean())
#print(RoutePointsPlaceName.head())
print('################')
sTable='Assess_RoutePointsPlaceName'
print('Storing :',sDatabaseName,' Table:',sTable)
RoutePointsPlaceName.to_sql(sTable, conn, if_exists="replace")
print('################')
#################################################################
### Fit Country to Country
#################################################################
print('################') sView='Assess_RouteCountries' print('Creating
:',sDatabaseName,' View:',sView)
sSQL="DROP VIEW IF EXISTS " + sView + ";"
sql.execute(sSQL,conn)

sSQL="CREATE VIEW " + sView + " AS"
sSQL=sSQL+ " SELECT DISTINCT"
sSQL=sSQL+ "      S.Country AS SourceCountry,"
sSQL=sSQL+ "      S.Latitude AS SourceLatitude,"
sSQL=sSQL+ "      S.Longitude AS SourceLongitude,"
sSQL=sSQL+ "      T.Country AS TargetCountry,"
sSQL=sSQL+ "      T.Latitude AS TargetLatitude,"
sSQL=sSQL+ "      T.Longitude AS TargetLongitude"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ "          Assess_RoutePointsCountry AS S"
sSQL=sSQL+ "          ,"
sSQL=sSQL+  "  Assess_RoutePointsCountry  AS  T"
sSQL=sSQL+  "  WHERE  S.Country  <>  T.Country"
sSQL=sSQL+ " AND"
sSQL=sSQL+ " S.Country in ('GB','DE','BE','AU','US','IN')"
sSQL=sSQL+ " AND"
sSQL=sSQL+ " T.Country in ('GB','DE','BE','AU','US','IN');"
sql.execute(sSQL,conn)

print('################')
print('Loading :',sDatabaseName,' Table:',sView)
sSQL=" SELECT "
sSQL=sSQL+ " *"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ " " + sView + ";"
RouteCountries=pd.read_sql_query(sSQL, conn)

RouteCountries['Distance']=RouteCountries.apply(lambda row:
      round(
```

```
        geodesic((row['SourceLatitude'],row['SourceLongitude']),
                        (row['TargetLatitude'],row['TargetLongitude'])).miles
                ,4)
            ,axis=1)

print(RouteCountries.head(5))
#################################################################
### Fit Country to Post Code
#################################################################
print('################') sView='Assess_RoutePostCode' print('Creating
:',sDatabaseName,' View:',sView)
sSQL="DROP VIEW IF EXISTS " + sView + ";"
sql.execute(sSQL,conn)

sSQL="CREATE VIEW " + sView + " AS"
sSQL=sSQL+ " SELECT DISTINCT"
sSQL=sSQL+ "      S.Country AS SourceCountry,"
sSQL=sSQL+ "      S.Latitude AS SourceLatitude,"
sSQL=sSQL+ "      S.Longitude AS SourceLongitude,"
sSQL=sSQL+ "      T.Country AS TargetCountry,"
sSQL=sSQL+ "      T.PostCode AS TargetPostCode,"
sSQL=sSQL+ "      T.Latitude AS TargetLatitude,"
sSQL=sSQL+ "      T.Longitude AS TargetLongitude"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ "          Assess_RoutePointsCountry AS S"
sSQL=sSQL+ "          ,"
sSQL=sSQL+ " Assess_RoutePointsPostCode AS T" sSQL=sSQL+ "
WHERE S.Country = T.Country"
sSQL=sSQL+ " AND"
sSQL=sSQL+ " S.Country in ('GB','DE','BE','AU','US','IN')"
sSQL=sSQL+ " AND"
sSQL=sSQL+ " T.Country in ('GB','DE','BE','AU','US','IN');"
sql.execute(sSQL,conn)

print('################')
print('Loading :',sDatabaseName,' Table:',sView)
sSQL=" SELECT "
sSQL=sSQL+ " *"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ " " + sView + ";"
RoutePostCode=pd.read_sql_query(sSQL, conn)

RoutePostCode['Distance']=RoutePostCode.apply(lambda row:
        round(
        geodesic((row['SourceLatitude'],row['SourceLongitude']),
                        (row['TargetLatitude'],row['TargetLongitude'])).mile
            s ,4) ,axis=1)
```

```
print(RoutePostCode.head(5))
#################################################################
### Fit Post Code to Place Name
#################################################################
print('################') sView='Assess_RoutePlaceName' print('Creating :',sDatabaseName,'
View:',sView)
sSQL="DROP VIEW IF EXISTS " + sView + ";"
sql.execute(sSQL,conn)

sSQL="CREATE VIEW " + sView + " AS"
sSQL=sSQL+ " SELECT DISTINCT"
sSQL=sSQL+ "      S.Country AS SourceCountry,"
sSQL=sSQL+ "      S.PostCode AS SourcePostCode,"
sSQL=sSQL+ "      S.Latitude AS SourceLatitude,"
sSQL=sSQL+ "      S.Longitude AS SourceLongitude,"
sSQL=sSQL+ "      T.Country AS TargetCountry,"
sSQL=sSQL+ "      T.PostCode AS TargetPostCode,"
sSQL=sSQL+ "      T.PlaceName AS TargetPlaceName,"
sSQL=sSQL+ "      T.Latitude AS TargetLatitude,"
sSQL=sSQL+ "      T.Longitude AS TargetLongitude"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ "          Assess_RoutePointsPostCode AS S"
sSQL=sSQL+ "          ,"
sSQL=sSQL+ "      Assess_RoutePointsPLaceName AS T"
sSQL=sSQL+ " WHERE" sSQL=sSQL+ "
S.Country = T.Country" sSQL=sSQL+ "
AND" sSQL=sSQL+ " S.PostCode =
T.PostCode"
sSQL=sSQL+ " AND"
sSQL=sSQL+ " S.Country in ('GB','DE','BE','AU','US','IN')"
sSQL=sSQL+ " AND"
sSQL=sSQL+ " T.Country in ('GB','DE','BE','AU','US','IN');"
sql.execute(sSQL,conn)

print('################')
print('Loading :',sDatabaseName,' Table:',sView)
sSQL=" SELECT "
sSQL=sSQL+ " *"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ " " + sView + ";"
RoutePlaceName=pd.read_sql_query(sSQL, conn)

RoutePlaceName['Distance']=RoutePlaceName.apply(lambda row:
        round(
        geodesic((row['SourceLatitude'],row['SourceLongitude']),
                        (row['TargetLatitude'],row['TargetLongitude'])).mile
            s ,4) ,axis=1)
```

```
print(RoutePlaceName.head(5))
#################################################################
G=nx.Graph()
#################################################################
print('Countries:',RouteCountries.shape) for i in range(RouteCountries.shape[0]):
	sNode0='C-' + RouteCountries['SourceCountry'][i]
	G.add_node(sNode0,
			Nodetype='Country',
			Country=RouteCountries['SourceCountry'][i],
			Latitude=round(RouteCountries['SourceLatitude'][i],4),
			Longitude=round(RouteCountries['SourceLongitude'][i],4))

	sNode1='C-' + RouteCountries['TargetCountry'][i]
	G.add_node(sNode1,
			Nodetype='Country',
			Country=RouteCountries['TargetCountry'][i],
			Latitude=round(RouteCountries['TargetLatitude'][i],4),
			Longitude=round(RouteCountries['TargetLongitude'][i],4))
	G.add_edge(sNode0,sNode1,distance=round(RouteCountries['Distance'][i],3))
	#print(sNode0,sNode1)
#################################################################
print('Post Code:',RoutePostCode.shape) for i in
range(RoutePostCode.shape[0]):
	sNode0='C-' + RoutePostCode['SourceCountry'][i]
	G.add_node(sNode0,
			Nodetype='Country',
			Country=RoutePostCode['SourceCountry'][i],
			Latitude=round(RoutePostCode['SourceLatitude'][i],4),
			Longitude=round(RoutePostCode['SourceLongitude'][i],4))

	sNode1='P-'	+	RoutePostCode['TargetPostCode'][i]		+	'-'	+
RoutePostCode['TargetCountry'][i]
	G.add_node(sNode1,
			Nodetype='PostCode',
			Country=RoutePostCode['TargetCountry'][i],
			PostCode=RoutePostCode['TargetPostCode'][i],
			Latitude=round(RoutePostCode['TargetLatitude'][i],4),
			Longitude=round(RoutePostCode['TargetLongitude'][i],4))
	G.add_edge(sNode0,sNode1,distance=round(RoutePostCode['Distance'][i],3))
	#print(sNode0,sNode1)
#################################################################
print('Place Name:',RoutePlaceName.shape) for i in
range(RoutePlaceName.shape[0]):
	sNode0='P-' + RoutePlaceName['TargetPostCode'][i]	+ '-' sNode0=sNode0
	+ RoutePlaceName['TargetCountry'][i]
	G.add_node(sNode0,
			Nodetype='PostCode',
			Country=RoutePlaceName['SourceCountry'][i],
```

```
                    PostCode=RoutePlaceName['TargetPostCode'][i],
                    Latitude=round(RoutePlaceName['SourceLatitude'][i],4),
                    Longitude=round(RoutePlaceName['SourceLongitude'][i],4))
    sNode1='L-' + RoutePlaceName['TargetPlaceName'][i]          + '-'
    sNode1=sNode1 + RoutePlaceName['TargetPostCode'][i]       + '-'
    sNode1=sNode1 + RoutePlaceName['TargetCountry'][i]
    G.add_node(sNode1,
                    Nodetype='PlaceName',
                    Country=RoutePlaceName['TargetCountry'][i],
                    PostCode=RoutePlaceName['TargetPostCode'][i],
                    PlaceName=RoutePlaceName['TargetPlaceName'][i],
                    Latitude=round(RoutePlaceName['TargetLatitude'][i],4),
                    Longitude=round(RoutePlaceName['TargetLongitude'][i],4))
    G.add_edge(sNode0,sNode1,distance=round(RoutePlaceName['Distance'][i],3))
    #print(sNode0,sNode1)
##################################################################
sFileName=OutputFileName
print('################################')
print('Storing :', sFileName)
print('################################')
nx.write_gml(G,sFileName)
sFileName=sFileName +'.gz'
nx.write_gml(G,sFileName)
##################################################################
print('################################') print('Path:', nx.shortest_path(G,source='P-SW1-
GB',target='P-01001-US',weight='distance')) print('Path      length:',
nx.shortest_path_length(G,source='P-SW1-GB',target='P-01001US',weight='distance'))
print('Path      length  (1):',    nx.shortest_path_length(G,source='P-SW1-
GB',target='CGB',weight='distance'))
print('Path          length          (2):',          nx.shortest_path_length(G,source='C-GB',target='C-
US',weight='distance'))
print('Path      length  (3):',    nx.shortest_path_length(G,source='C-US',target='P-
01001US',weight='distance'))
print('################################')
print('Routes    from    P-SW1-GB    <        2:        ',
nx.single_source_shortest_path(G,source='P-SW1GB' ,cutoff=1))
print('Routes from P-01001-US < 2: ', nx.single_source_shortest_path(G,source='P-01001US'
,cutoff=1))
print('################################')
##################################################################
print('################') print('Vacuum Database') sSQL="VACUUM;"
sql.execute(sSQL,conn)
print('################')
##################################################################
print('### Done!! ##########################################')
################################################################## Output:-
```

```
  SourceCountry SourcePostcode  ...  TargetLongitude  Distance
0            AU           0200  ...         149.1189       0.0
1            AU           0221  ...         149.1412       0.0
2            AU           0800  ...         130.8418       0.0
3            AU           0801  ...         130.8418       0.0
4            AU           0803  ...         130.7443       0.0

[5 rows x 10 columns]
Countries: (30, 7)
Post Code: (76316, 8)
Place Name: (260360, 10)
################################
Storing : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5H\Assess_Best_Logistics.gml
################################
################################
Path: ['P-SW1-GB', 'C-GB', 'C-US', 'P-01001-US']
Path length: 5137.452
Path length (1): 155.802
Path length (2): 4055.869
Path length (3): 925.781
################################
Routes from P-SW1-GB < 2:  {'P-SW1-GB': ['P-SW1-GB'], 'C-GB': ['P-SW1-GB', 'C-GB'], 'L-Belgr
-Buckingham Palace-SW1-GB': ['P-SW1-GB', 'L-Buckingham Palace-SW1-GB'], 'L-Green Park-SW1-GB
y Road-SW1-GB': ['P-SW1-GB', 'L-Horseferry Road-SW1-GB'], 'L-Hyde Park Corner-SW1-GB': ['P-S
ge-SW1-GB': ['P-SW1-GB', 'L-Knightsbridge-SW1-GB'], 'L-Pimlico-SW1-GB': ['P-SW1-GB', 'L-Piml
'L-Sloane Square-SW1-GB'], "L-St James's Park-SW1-GB": ['P-SW1-GB', "L-St James's Park-SW1-G
xhall Bridge-SW1-GB'], 'L-Victoria-SW1-GB': ['P-SW1-GB', 'L-Victoria-SW1-GB'], 'L-Westminste
Routes from P-01001-US < 2:  {'P-01001-US': ['P-01001-US'], 'C-US': ['P-01001-US', 'C-US'],
US']}
################################
################
Vacuum Database

Warning (from warnings module):
  File "F:\DS_Practicals_Msc-it\Practical no 5\practicalno5H\Practicalno5h.py", line 275
    sql.execute(sSQL,conn)
FutureWarning: `pandas.io.sql.execute` is deprecated and will be removed in the future versi
################
### Done!! ##########################################
```

file:///F:/DS_Practicals_Msc-it/Practical%20no%205/practicalno5H/Assess_Best_Logistics.gml

```
graph [
  node [
    id 0
    label "C-AU"
    Nodetype "Country"
    Country "AU"
    Latitude NP.FLOAT64(-32.1091)
    Longitude NP.FLOAT64(143.7176)
  ]
  node [
    id 1
    label "C-BE"
    Nodetype "Country"
    Country "BE"
    Latitude NP.FLOAT64(50.6403)
    Longitude NP.FLOAT64(4.5578)
  ]
  node [
    id 2
    label "C-DE"
    Nodetype "Country"
    Country "DE"
    Latitude NP.FLOAT64(51.2126)
    Longitude NP.FLOAT64(10.2662)
  ]
  node [
    id 3
```

Practical no 4 I

Aim:-Write a Python / R program to decide the best packing option to ship in container from the given data.

```
Code:import pandas
as pd import sqlite3

# File paths (update these paths as needed)
InputFileName1 = "F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5I\\Retrieve_Product.csv"
InputFileName2 = "F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5I\\Retrieve_Box.csv"
InputFileName3 = "F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5I\\Retrieve_Container.csv"
DatabaseFileName = "F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5I\\hillman.db"

# SQLite connection
conn = sqlite3.connect(DatabaseFileName)

# Function to process and store data def process_data(filename, table_name, id_field_name,
additional_filters=None, max_rows=None):
    print(f"Loading: {filename}")
    data_raw = pd.read_csv(filename, header=0, low_memory=False, encoding="latin-1")
    data_raw.drop_duplicates(inplace=True)

    # Add IDNumber dynamically
    data_raw.reset_index(drop=True, inplace=True)
    data_raw[id_field_name] = data_raw.index
    data_raw.set_index(id_field_name, inplace=True)

    # Apply filters if provided
    if additional_filters:
        for col, condition in additional_filters.items():
            data_raw = data_raw.query(f"{col} {condition}")

    # Limit the number of rows if needed
    if max_rows:
        data_raw = data_raw.head(max_rows)

    print(f"Rows in {table_name}: {data_raw.shape[0]}")
    data_raw.to_sql(table_name, conn, if_exists="replace",
    index=True) print(f"Stored: {DatabaseFileName}Table:
    {table_name}") return data_raw
# Process Product Data
ProductData = process_data(
    filename=InputFileName1,
    table_name="Assess_Product",
    id_field_name="IDNumber"
)

# Process Box Data with filtering and max row limit
BoxData = process_data(
    filename=InputFileName2,
    table_name="Assess_Box",
    id_field_name="IDNumber",
    additional_filters={"Length": "<= 1"},    # Example filter for Length
    max_rows=1000
```

```
)
# Process Container Data
ContainerData = process_data(
    filename=InputFileName3,
    table_name="Assess_Container",
    id_field_name="IDNumber"
)

# Create view for Product in Box
print("################") print("Creating
View: Assess_Product_in_Box") sSQL = """
CREATE VIEW IF NOT EXISTS Assess_Product_in_Box AS
SELECT P.IDNumber AS ProductID, P.UnitNumber AS ProductUnit,
        B.IDNumber AS BoxID, B.UnitNumber AS BoxUnit
FROM Assess_Product AS P, Assess_Box AS B
WHERE P.ProductVolume <= B.BoxVolume;
""" with conn:
conn.execute(sSQL)
print("View: Assess_Product_in_Box created successfully.")

# Create PalletFrame by combining data dynamically
print("################")
print("Creating PalletFrame...")
PalletFrame = pd.DataFrame(columns=["PalletID", "BoxID"])
PalletID = 0

for index, box_row in BoxData.iterrows():
    if PalletFrame.empty or len(PalletFrame[PalletFrame["PalletID"] == PalletID]) >= 10:
        PalletID += 1
    PalletFrame = pd.concat([
        PalletFrame,
        pd.DataFrame({"PalletID": [PalletID], "BoxID": [index]})
    ], ignore_index=True)

# Save PalletFrame to database
PalletFrame.to_sql("Assess_Pallet", conn, if_exists="replace", index=False)
print("PalletFrame stored in database successfully.")
print("################")

# Close the database connection
conn.close()
print("Done")
```

Output;-

```
================================ RESTART: F:\DS_Practicals_Msc-it\Practical no 5\practicalno5I\Prac
Loading: F:\DS_Practicals_Msc-it\Practical no 5\practicalno5I\Retrieve_Product.csv
Rows in Assess_Product: 8000
Stored: F:\DS_Practicals_Msc-it\Practical no 5\practicalno5I\hillman.db  Table: Assess_Product
Loading: F:\DS_Practicals_Msc-it\Practical no 5\practicalno5I\Retrieve_Box.csv
Rows in Assess_Box: 1000
Stored: F:\DS_Practicals_Msc-it\Practical no 5\practicalno5I\hillman.db  Table: Assess_Box
Loading: F:\DS_Practicals_Msc-it\Practical no 5\practicalno5I\Retrieve_Container.csv
Rows in Assess_Container: 900
Stored: F:\DS_Practicals_Msc-it\Practical no 5\practicalno5I\hillman.db  Table: Assess_Container
################
Creating View: Assess_Product_in_Box
View: Assess_Product_in_Box created successfully.
################
Creating PalletFrame...
PalletFrame stored in database successfully.
################
Done
```

Practical no 4 J

Aim:-Write a Python program to create a delivery route using the given data.

Code:

```
import pandas as pd import
sqlite3 as sq from pandas.io import
sql import networkx as nx from
geopy.distance import geodesic
nMax=3
nMaxPath=10
nSet=False
nVSet=False
InputFileName1='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5J\\Retrieve_GB_Postcode_Warehouse.csv'
InputFileName2='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5J\\Retrieve_GB_Postcodes_Shops.csv'
OutputFileName1='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5J\\Assess_Shipping_Routes.gml'
OutputFileName2='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5J\\Assess_Shipping_Routes.txt'
sDatabaseName='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5J\\hillman.db'
conn = sq.connect(sDatabaseName) sFileName=InputFileName1 print('Loading :',sFileName)
WarehouseRawData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
WarehouseRawData.drop_duplicates(subset=None, keep='first', inplace=True)
WarehouseRawData.index.name = 'IDNumber'
# WarehouseData=WarehouseRawData.head(nMax)
# WarehouseData=WarehouseData.append(WarehouseRawData.tail(nMax))
# WarehouseData=WarehouseData.append(WarehouseRawData[WarehouseRawData.postcode=='KA13'])
# if nSet==True:
# WarehouseData=WarehouseData.append(WarehouseRawData[WarehouseRawData.postcode=='SW1W'])

# Replace these lines
# WarehouseData = WarehouseRawData.head(nMax)
```

```
# WarehouseData = WarehouseData.append(WarehouseRawData.tail(nMax))
# WarehouseData = WarehouseData.append(WarehouseRawData[WarehouseRawData.postcode
== 'KA13'])
# if nSet == True:
#                                                    WarehouseData          =
WarehouseData.append(WarehouseRawData[WarehouseRawData.postcode == 'SW1W'])

# With this:
frames = [WarehouseRawData.head(nMax),
          WarehouseRawData.tail(nMax),
          WarehouseRawData[WarehouseRawData.postcode == 'KA13']]
if nSet:
    frames.append(WarehouseRawData[WarehouseRawData.postcode == 'SW1W'])

WarehouseData = pd.concat(frames).drop_duplicates(keep='first')

WarehouseData.drop_duplicates(subset=None, keep='first', inplace=True)
print('Loaded Warehouses :',WarehouseData.columns.values)
sTable='Assess_Warehouse_UK'
print('Storing :',sDatabaseName,' Table:',sTable)
WarehouseData.to_sql(sTable, conn, if_exists="replace")
print(WarehouseData.head()) print('Rows :
',WarehouseData.shape[0]) sFileName=InputFileName2
print('###########') print('Loading :',sFileName)
ShopRawData=pd.read_csv(sFileName,header=0,low_memory=False,encoding="latin-1")
ShopRawData.drop_duplicates(subset=None, keep='first', inplace=True)
ShopRawData.index.name = 'IDNumber'
ShopData=ShopRawData
print('Loaded Shops :',ShopData.columns.values)
sTable='Assess_Shop_UK'
print('Storing :',sDatabaseName,' Table:',sTable)
ShopData.to_sql(sTable, conn, if_exists="replace")
print(ShopData.head())
print('#################################')
print('Rows : ',ShopData.shape[0])
print('#################################')
sView='Assess_HQ'
print('Creating :',sDatabaseName,' View:',sView)
sSQL="DROP VIEW IF EXISTS " + sView + ";"
sql.execute(sSQL,conn)
sSQL="CREATE VIEW " + sView + " AS"
sSQL=sSQL+ " SELECT" sSQL=sSQL+ "
W.postcode AS HQ_PostCode," sSQL=sSQL+ " 'HQ-'
|| W.postcode AS HQ_Name," sSQL=sSQL+ "
round(W.latitude,6) AS HQ_Latitude,"
sSQL=sSQL+ " round(W.longitude,6) AS HQ_Longitude"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ " Assess_Warehouse_UK as W"
```

```
sSQL=sSQL+ " WHERE"
sSQL=sSQL+ " TRIM(W.postcode) in ('KA13','SW1W');"
sql.execute(sSQL,conn)
sView='Assess_Warehouse' print('Creating
:',sDatabaseName,' View:',sView)
sSQL="DROP VIEW IF EXISTS " + sView + ";"
sql.execute(sSQL,conn)
sSQL="CREATE VIEW " + sView + " AS"
sSQL=sSQL+ " SELECT"
sSQL=sSQL+ " W.postcode AS Warehouse_PostCode,"
sSQL=sSQL+ " 'WH-' || W.postcode AS Warehouse_Name,"
sSQL=sSQL+ " round(W.latitude,6) AS Warehouse_Latitude,"
sSQL=sSQL+ " round(W.longitude,6) AS Warehouse_Longitude"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ " Assess_Warehouse_UK as W;"
sql.execute(sSQL,conn) sView='Assess_Shop'
print('Creating :',sDatabaseName,' View:',sView)
sSQL="DROP VIEW IF EXISTS " + sView + ";"
sql.execute(sSQL,conn)
sSQL="CREATE VIEW " + sView + " AS"
sSQL=sSQL+ " SELECT"
sSQL=sSQL+ " TRIM(S.postcode) AS Shop_PostCode,"
sSQL=sSQL+ " 'SP-' || TRIM(S.FirstCode) || '-' || TRIM(S.SecondCode) AS Shop_Name,"
sSQL=sSQL+ " TRIM(S.FirstCode) AS Warehouse_PostCode," sSQL=sSQL+ "
round(S.latitude,6) AS Shop_Latitude,"
sSQL=sSQL+ " round(S.longitude,6) AS Shop_Longitude"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ " Assess_Warehouse_UK as W"
sSQL=sSQL+    "    JOIN"    sSQL=sSQL+    "
Assess_Shop_UK as S"
sSQL=sSQL+ " ON"
sSQL=sSQL+ " TRIM(W.postcode) = TRIM(S.FirstCode);"
sql.execute(sSQL,conn) G=nx.Graph() sTable =
'Assess_HQ'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL=" SELECT DISTINCT"
sSQL=sSQL+ " *"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ " " + sTable + ";"
RouteData=pd.read_sql_query(sSQL, conn)
print(RouteData.head())
print('##################################')
print('HQ Rows : ',RouteData.shape[0]) for i in
range(RouteData.shape[0]):
    sNode0=RouteData['HQ_Name'][i]
    G.add_node(sNode0,
               Nodetype='HQ',
               PostCode=RouteData['HQ_PostCode'][i],
```

```
                    Latitude=round(RouteData['HQ_Latitude'][i],6),
                    Longitude=round(RouteData['HQ_Longitude'][i],6))
sTable = 'Assess_Warehouse'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL=" SELECT DISTINCT"
sSQL=sSQL+ " *"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ " " + sTable + ";"
RouteData=pd.read_sql_query(sSQL, conn)
print(RouteData.head())
print('################################') print('Warehouse
Rows : ',RouteData.shape[0]) for i in range(RouteData.shape[0]):
    sNode0=RouteData['Warehouse_Name'][i]
    G.add_node(sNode0,
                    Nodetype='Warehouse',
                    PostCode=RouteData['Warehouse_PostCode'][i],
                    Latitude=round(RouteData['Warehouse_Latitude'][i],6),
                    Longitude=round(RouteData['Warehouse_Longitude'][i],6))
#################################################################
print('################') sTable = 'Assess_Shop'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL=" SELECT DISTINCT"
sSQL=sSQL+ " *"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ " " + sTable + ";"
RouteData=pd.read_sql_query(sSQL, conn)
print('################')
#################################################################
print(RouteData.head()) print('################################')
print('Shop Rows : ',RouteData.shape[0])
print('################################')
#################################################################
for i in range(RouteData.shape[0]):
    sNode0=RouteData['Shop_Name'][i]
    G.add_node(sNode0,
                    Nodetype='Shop',
                    PostCode=RouteData['Shop_PostCode'][i],
                    WarehousePostCode=RouteData['Warehouse_PostCode'][i],
                    Latitude=round(RouteData['Shop_Latitude'][i],6),
                    Longitude=round(RouteData['Shop_Longitude'][i],6))
for sNode0 in G.nodes:
    for sNode1 in G.nodes:
        if G.nodes[sNode0]['Nodetype']=='HQ' and \
            G.nodes[sNode1]['Nodetype']=='HQ' and \
            sNode0 != sNode1:
                distancemeters=round(\
                                geodesic(\
                                        (\
```

```
                                                G.nodes[sNode0]['Latitude'],\
                                                G.nodes[sNode0]['Longitude']\
                                                ),\
                                                (\
                                                  G.nodes[sNode1]['Latitude']\
                                                  ,\
                                                  G.nodes[sNode1]['Longitude']\
                                                  )\
                                                  ).meters\
                                    ,0)
                    distancemiles=round(\
                                      geodesic(\
                                                (\
                                                G.nodes[sNode0]['Latitude'],\
                                                G.nodes[sNode0]['Longitude']\
                                                ),\
                                                (\
                                                  G.nodes[sNode1]['Latitude']\
                                                  ,\
                                                  G.nodes[sNode1]['Longitude']\
                                                  )\
                                                  ).miles\
                                    ,3)

                    if distancemiles >= 0.05: cost =
                          round(150+(distancemiles * 2.5),6)
                          vehicle='V001'
                    else:
                          cost = round(2+(distancemiles * 0.10),6)
                          vehicle='ForkLift'

                    G.add_edge(sNode0,sNode1,DistanceMeters=distancemeters, \
                                DistanceMiles=distancemiles, \
                                Cost=cost,Vehicle=vehicle)
                    if nVSet==True:
                          print('Edge-H-H:',sNode0,'    to    ',    sNode1,    \    '
                             Distance:',distancemeters,'meters',\
                             distancemiles,'miles','Cost', cost,'Vehicle',vehicle)

          if G.nodes[sNode0]['Nodetype']=='HQ' and \
                G.nodes[sNode1]['Nodetype']=='Warehouse' and \
                sNode0 != sNode1:
                    distancemeters=round(\
                                      geodesic(\
                                                (\
                                                G.nodes[sNode0]['Latitude'],\
                                                G.nodes[sNode0]['Longitude']\
```

```
                                                    ),\
                                                    (\
                                                      G.nodes[sNode1]['Latitude']\
                                                      ,\
                                                      G.nodes[sNode1]['Longitude']\
                                                      )\
                                                      ).meters\
                                        ,0)
                distancemiles=round(\
                                        geodesic(\
                                                    (\
                                                    G.nodes[sNode0]['Latitude'],\
                                                    G.nodes[sNode0]['Longitude']\
                                                    ),\
                                                    (\
                                                      G.nodes[sNode1]['Latitude']\
                                                      ,\
                                                      G.nodes[sNode1]['Longitude']\
                                                      )\
                                                      ).miles\
                                        ,3)
                if distancemiles >= 10: cost =
                        round(50+(distancemiles * 2),6)
                        vehicle='V002'
                else:
                        cost = round(5+(distancemiles * 1.5),6)
                        vehicle='V003'
                if distancemiles <= 50:
                        G.add_edge(sNode0,sNode1,DistanceMeters=distancemeters, \
                                        DistanceMiles=distancemiles, \
                                        Cost=cost,Vehicle=vehicle)
                        if nVSet==True:
                                print('Edge-H-W:',sNode0,'  to  ',  sNode1,  \  '
                                    Distance:',distancemeters,'meters',\
                                    distancemiles,'miles','Cost', cost,'Vehicle',vehicle)

        if nSet==True and \
                G.nodes[sNode0]['Nodetype']=='Warehouse' and \
                G.nodes[sNode1]['Nodetype']=='Warehouse' and \
                sNode0 != sNode1:
                distancemeters=round(\
                                        geodesic(\
                                                    (\
                                                    G.nodes[sNode0]['Latitude'],\
                                                    G.nodes[sNode0]['Longitude']\
                                                    ),\
                                                    (\
```

```
                                                G.nodes[sNode1]['Latitude']\
                                                ,\
                                                G.nodes[sNode1]['Longitude']\
                                                )\
                                                ).meters\
                                        ,0)
                distancemiles=round(\
                                        geodesic(\
                                                (\
                                                G.nodes[sNode0]['Latitude'],\
                                                G.nodes[sNode0]['Longitude']\
                                                ),\
                                                (\
                                                G.nodes[sNode1]['Latitude']\
                                                ,\
                                                G.nodes[sNode1]['Longitude']\
                                                )\
                                                ).miles\
                                        ,3)
                if distancemiles >= 10: cost =
                        round(50+(distancemiles * 1.10),6)
                        vehicle='V004'
                else:
                        cost = round(5+(distancemiles * 1.05),6)
                        vehicle='V005'

                if distancemiles <= 20:
                        G.add_edge(sNode0,sNode1,DistanceMeters=distancemeters, \
                                        DistanceMiles=distancemiles, \
                                        Cost=cost,Vehicle=vehicle)
                        if nVSet==True:
                                print('Edge-W-W:',sNode0,'    to    ',    sNode1,    \    '
                                        Distance:',distancemeters,'meters',\
                                        distancemiles,'miles','Cost', cost,'Vehicle',vehicle)

        if G.nodes[sNode0]['Nodetype']=='Warehouse' and \
                G.nodes[sNode1]['Nodetype']=='Shop' and \
                G.nodes[sNode0]['PostCode']==G.nodes[sNode1]['WarehousePostCode'] and \
                sNode0 != sNode1:

                distancemeters=round(\
                                        geodesic(\
                                                (\
                                                G.nodes[sNode0]['Latitude'],\
                                                G.nodes[sNode0]['Longitude']\
                                                ),\
                                                (\
```

```
                                                    G.nodes[sNode1]['Latitude']\
                                                    ,\
                                                    G.nodes[sNode1]['Longitude']\
                                                    )\
                                                    ).meters\
                                        ,0)
                    distancemiles=round(\
                                        geodesic(\
                                                (\
                                                  G.nodes[sNode0]['Latitude'],\
                                                  G.nodes[sNode0]['Longitude']\
                                                  ),\
                                                  (\
                                                    G.nodes[sNode1]['Latitude']\
                                                    ,\
                                                    G.nodes[sNode1]['Longitude']\
                                                    )\
                                                    ).miles\
                                        ,3)
                    if distancemiles >= 10: cost =
                          round(50+(distancemiles * 1.50),6)
                          vehicle='V006'
                    else:
                          cost = round(5+(distancemiles * 0.75),6)
                          vehicle='V007'
                    if distancemiles <= 10:
                          G.add_edge(sNode0,sNode1,DistanceMeters=distancemeters, \
                                     DistanceMiles=distancemiles, \
                                     Cost=cost,Vehicle=vehicle)
                          if nVSet==True:
                                print('Edge-W-S:',sNode0,'    to    ',    sNode1,    \    '
                                   Distance:',distancemeters,'meters',\
                                   distancemiles,'miles','Cost', cost,'Vehicle',vehicle)

          if nSet==True and \
                G.nodes[sNode0]['Nodetype']=='Shop' and \
                G.nodes[sNode1]['Nodetype']=='Shop' and \

G.nodes[sNode0]['WarehousePostCode']==G.nodes[sNode1]['WarehousePostCode'] and \
                sNode0 != sNode1:

                    distancemeters=round(\
                                        geodesic(\
                                                (\
                                                  G.nodes[sNode0]['Latitude'],\
                                                  G.nodes[sNode0]['Longitude']\
                                                  ),\
```

```
                                                    (\
                                                     G.nodes[sNode1]['Latitude']\
                                                     ,\
                                                     G.nodes[sNode1]['Longitude']\
                                                     )\
                                                     ).meters\
                                        ,0)
                    distancemiles=round(\
                                          geodesic(\
                                                    (\
                                                     G.nodes[sNode0]['Latitude'],\
                                                     G.nodes[sNode0]['Longitude']\
                                                     ),\
                                                     (\
                                                     G.nodes[sNode1]['Latitude']\
                                                     ,\
                                                     G.nodes[sNode1]['Longitude']\
                                                     )\
                                                     ).miles\
                                        ,3)
                    if distancemiles >= 0.05: cost =
                          round(5+(distancemiles * 0.5),6)
                          vehicle='V008'
                    else:
                          cost = round(1+(distancemiles * 0.1),6)
                          vehicle='V009'

                    if distancemiles <= 0.075:
                          G.add_edge(sNode0,sNode1,DistanceMeters=distancemeters, \
                                     DistanceMiles=distancemiles, \
                                     Cost=cost,Vehicle=vehicle)
                          if nVSet==True:
                                print('Edge-S-S:',sNode0,'    to    ',    sNode1,    \    '
                                   Distance:',distancemeters,'meters',\
                                   distancemiles,'miles','Cost', cost,'Vehicle',vehicle)

          if nSet==True and \
                G.nodes[sNode0]['Nodetype']=='Shop' and \
                G.nodes[sNode1]['Nodetype']=='Shop' and \

G.nodes[sNode0]['WarehousePostCode']!=G.nodes[sNode1]['WarehousePostCode'] and \
                sNode0 != sNode1:

                    distancemeters=round(\
                                          geodesic(\
                                                    (\
                                                     G.nodes[sNode0]['Latitude'],\
```

```
                                                G.nodes[sNode0]['Longitude']\
                                                ),\
                                                (\
                                                  G.nodes[sNode1]['Latitude']\
                                                  ,\
                                                  G.nodes[sNode1]['Longitude']\
                                                  )\
                                                  ).meters\
                                        ,0)
                    distancemiles=round(\
                                          geodesic(\
                                                  (\
                                                G.nodes[sNode0]['Latitude'],\
                                                G.nodes[sNode0]['Longitude']\
                                                ),\
                                                (\
                                                  G.nodes[sNode1]['Latitude']\
                                                  ,\
                                                  G.nodes[sNode1]['Longitude']\
                                                  )\
                                                  ).miles\
                                        ,3)
                    cost = round(1+(distancemiles * 0.1),6)
                    vehicle='V010'

                    if distancemiles <= 0.025:
                          G.add_edge(sNode0,sNode1,DistanceMeters=distancemeters, \
                                      DistanceMiles=distancemiles, \
                                      Cost=cost,Vehicle=vehicle)
                          if nVSet==True:
                                print('Edge-S-S:',sNode0,'    to    ',    sNode1,    \    '
                                   Distance:',distancemeters,'meters',\
                                   distancemiles,'miles','Cost', cost,'Vehicle',vehicle)

sFileName=OutputFileName1
print('#################################')
print('Storing :', sFileName)
print('#################################')
nx.write_gml(G,sFileName)
sFileName=sFileName +'.gz'
nx.write_gml(G,sFileName)
print('Nodes:',nx.number_of_nodes(G))
print('Edges:',nx.number_of_edges(G))
sFileName=OutputFileName2
print('#################################')
print('Storing :', sFileName)
```

```
print('##################################')
print('Loading Paths')
print('##################################')
f = open(sFileName,'w') l=0
sline = 'ID|Cost|StartAt|EndAt|Path|Measure'
if nVSet==True: print ('0', sline)
f.write(sline+ '\n') for sNode0 in G.edges:
        for sNode1 in G.edges:
                if sNode0 != sNode1 and \ nx.has_path(G, sNode0,
                        sNode1)==True and \
                        nx.shortest_path_length(G, \
                            source=sNode0, \ target=sNode1, \
                            weight='DistanceMiles') <
                            nMaxPath: l+=1
                                sID='{:.0f}'.format(l) spath =
                                ','.join(nx.shortest_path(G, \
                                    source=sNode0, \
                                    target=sNode1, \
                                    weight='DistanceMiles'))
                                slength= '{:.6f}'.format(\
                                    nx.shortest_path_length(G, \
                                    source=sNode0, \
                                    target=sNode1, \
                                    weight='DistanceMiles'))
                                sline = sID + '|"DistanceMiles"|"' + sNode0 + '"|"'
                                \ + sNode1 + '"|"' + spath + '"|' + slength if
                                nVSet==True: print (sline) f.write(sline + '\n')
                                l+=1
                                sID='{:.0f}'.format(l) spath =
                                ','.join(nx.shortest_path(G, \
                                    source=sNode0, \
                                    target=sNode1, \
                                    weight='DistanceMeters'))
                                slength= '{:.6f}'.format(\
                                    nx.shortest_path_length(G, \
                                    source=sNode0, \
                                    target=sNode1, \
                                    weight='DistanceMeters'))
                                sline = sID + '|"DistanceMeters"|"' + sNode0 + '"|"'
                                \ + sNode1 + '"|"' + spath + '"|' + slength if
                                nVSet==True: print (sline) f.write(sline + '\n') l+=1
                                sID='{:.0f}'.format(l) spath =
                                ','.join(nx.shortest_path(G, \
                                    source=sNode0, \
                                    target=sNode1, \
                                    weight='Cost'))
                                slength= '{:.6f}'.format(\
```

```
                    nx.shortest_path_length(G, \
                    source=sNode0, \
                    target=sNode1, \
                    weight='Cost'))
                sline = sID + '|"Cost"|"' + sNode0 + '"|"'
                \ + sNode1 + '"|"' + spath + '"|' + slength
                if nVSet==True: print (sline)
                f.write(sline + '\n')
f.close()
print('Nodes:',nx.number_of_nodes(G))
print('Edges:',nx.number_of_edges(G))
print('Paths:',sID)
sSQL="VACUUM;"
sql.execute(sSQL,conn)
```

Output:-

```
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5J\hillman.db  Table: Assess_HQ
  HQ_PostCode  HQ_Name  HQ_Latitude  HQ_Longitude
0        KA13  HQ-KA13      55.6553      -4.69921
################################
HQ Rows :  1
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5J\hillman.db  Table: Assess_Warehouse
  Warehouse_PostCode Warehouse_Name  Warehouse_Latitude  Warehouse_Longitude
0               AB10        WH-AB10            57.13514             -2.11731
1               AB11        WH-AB11            57.13875             -2.09089
2               AB12        WH-AB12            57.10100             -2.11060
3               PA80        WH-PA80             0.00000              0.00000
4                L80         WH-L80             0.00000              0.00000
################################
Warehouse Rows :  7
################
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5J\hillman.db  Table: Assess_Shop
################
  Shop_PostCode    Shop_Name Warehouse_PostCode  Shop_Latitude  Shop_Longitude
0      AB10 1XG  SP-AB10-1XG               AB10      57.144165       -2.114848
1      AB10 6RN  SP-AB10-6RN               AB10      57.137880       -2.121487
2      AB10 7JB  SP-AB10-7JB               AB10      57.124274       -2.127190
3      AB11 5QN  SP-AB11-5QN               AB11      57.142701       -2.093295
4      AB11 6UL  SP-AB11-6UL               AB11      57.137547       -2.112233
################################
Shop Rows :  451
################################
################################
Storing : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5J\Assess_Shipping_Routes.gml
################################
Nodes: 459
Edges: 452
################################
Storing : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5J\Assess_Shipping_Routes.txt
################################
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5J\hillman.db  Table: Assess_HQ
  HQ_PostCode  HQ_Name  HQ_Latitude  HQ_Longitude
0        KA13  HQ-KA13      55.6553      -4.69921
################################
HQ Rows :  1
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5J\hillman.db  Table: Assess_Warehouse
  Warehouse_PostCode Warehouse_Name  Warehouse_Latitude  Warehouse_Longitude
0               AB10        WH-AB10            57.13514             -2.11731
1               AB11        WH-AB11            57.13875             -2.09089
2               AB12        WH-AB12            57.10100             -2.11060
3               PA80        WH-PA80             0.00000              0.00000
4                L80         WH-L80             0.00000              0.00000
################################
Warehouse Rows :  7
################
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5J\hillman.db  Table: Assess_Shop
################
  Shop_PostCode    Shop_Name Warehouse_PostCode  Shop_Latitude  Shop_Longitude
0      AB10 1XG  SP-AB10-1XG               AB10      57.144165       -2.114848
1      AB10 6RN  SP-AB10-6RN               AB10      57.137880       -2.121487
2      AB10 7JB  SP-AB10-7JB               AB10      57.124274       -2.127190
3      AB11 5QN  SP-AB11-5QN               AB11      57.142701       -2.093295
4      AB11 6UL  SP-AB11-6UL               AB11      57.137547       -2.112233
################################
Shop Rows :  451
################################
################################
Storing : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5J\Assess_Shipping_Routes.gml
################################
Nodes: 459
Edges: 452
################################
Storing : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5J\Assess_Shipping_Routes.txt
################################
Loading Paths
################################
```

Pratical no 4 K

Aim:-Write a Python program to create Simple forex trading planner from the given data.

Code:-

```
import sqlite3 as sq
import pandas as pd
sInputFileName1='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5K\\Retrieve-
CountryCurrency.csv'
sInputFileName2='F:\\DS_Practicals_Msc-it\\Practical no
5\\practicalno5K\\Retrieve_Euro_EchangeRates.csv'
sDatabaseName='F:\\DS_Practicals_Msc-it\\Practical no 5\\practicalno5K\\clark.db'
conn = sq.connect(sDatabaseName) sFileName1=sInputFileName1 print('Loading
:',sFileName1)
CountryRawData=pd.read_csv(sFileName1,header=0,low_memory=False, encoding="latin-1")
CountryRawData.drop_duplicates(subset=None, keep='first', inplace=True)
CountryData=CountryRawData
print('Loaded Company :',CountryData.columns.values)
sTable='Assess_Country' print('Storing
:',sDatabaseName,' Table:',sTable)
CountryData.to_sql(sTable, conn, if_exists="replace")
print(CountryData.head()) print('Rows :
',CountryData.shape[0]) sFileName2=sInputFileName2
print('Loading :',sFileName2)
ForexRawData=pd.read_csv(sFileName2,header=0,low_memory=False, encoding="latin-1")
ForexRawData.drop_duplicates(subset=None, keep='first', inplace=True)
ForexData=ForexRawData.head(5)
prin('Loaded Company :',ForexData.columns.values)
sTable='Assess_Forex' print('Storing
:',sDatabaseName,' Table:',sTable)
ForexData.to_sql(sTable, conn, if_exists="replace")
print(ForexData.head()) print('Rows :
',ForexData.shape[0])
sTable='Assess_Forex'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="select distinct" sSQL=sSQL+ "
A.CodeIn" sSQL=sSQL+ " from" sSQL=sSQL+
" Assess_Forex as A;"
odeData=pd.read_sql_query(sSQL, conn) for c
in range(CodeData.shape[0]):

    sTable='Assess_Forex & 2x Country > ' + CodeData['CodeIn'][c]
    print('Loading :',sDatabaseName,'
    Table:',sTable) sSQL="select distinct"
    sSQL=sSQL+ " A.Date," sSQL=sSQL+ "
    A.CodeIn," sSQL=sSQL+ " B.Country as
    CountryIn," sSQL=sSQL+ " B.Currency as
    CurrencyNameIn," sSQL=sSQL+ " A.CodeOut,"
    sSQL=sSQL+ " C.Country as CountryOut,"
    sSQL=sSQL+ " C.Currency as
    CurrencyNameOut," sSQL=sSQL+ " A.Rate"
    sSQL=sSQL+ " from" sSQL=sSQL+ "
```

```
Assess_Forex as A" sSQL=sSQL+ " JOIN"
sSQL=sSQL+ " Assess_Country as B"
sSQL=sSQL+ " ON A.CodeIn =
B.CurrencyCode" sSQL=sSQL+ " JOIN"
sSQL=sSQL+ " Assess_Country as C"
sSQL=sSQL+ " ON A.CodeOut =
C.CurrencyCode" sSQL=sSQL+ " WHERE"
sSQL=sSQL+ " A.CodeIn ='" + CodeData['CodeIn'][c] + "';"
ForexData=pd.read_sql_query(sSQL, conn).head(1000)
print(ForexData) sTable='Assess_Forex_' +
CodeData['CodeIn'][c] print('Storing :',sDatabaseName,'
Table:',sTable) ForexData.to_sql(sTable, conn,
if_exists="replace") print('Rows : ',ForexData.shape[0])
print('Done') Output:-
```

```
================================ RESTART: F:\DS_Practicals_Msc-it\Practical no 5\practicalno5K\Practicalno5k.py =
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5K\Retrieve-Country-Currency.csv
Loaded Company : ['Country' 'Currency' 'CurrencyCode']
Storing : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5K\clark.db  Table: Assess_Country
                     Country        Currency CurrencyCode
0                Afghanistan  Afghan afghani          AFN
1  Akrotiri and Dhekelia (UK)   European euro          EUR
2     Aland Islands (Finland)   European euro          EUR
3                     Albania    Albanian lek          ALL
4                     Algeria  Algerian dinar          DZD
Rows :  253
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5K\Retrieve_Euro_EchangeRates.csv
Loaded Company : ['Date' 'CodeIn' 'CodeOut' 'Rate']
Storing : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5K\clark.db  Table: Assess_Forex
         Date CodeIn CodeOut      Rate
0  2017-05-05    AUD     AUD  1.000000
1  2017-05-05    AUD     BGN  1.318635
2  2017-05-05    AUD     BRL  2.354167
3  2017-05-05    AUD     CAD  1.017665
4  2017-05-05    AUD     CHF  0.730785
Rows :  5
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5K\clark.db  Table: Assess_Forex
Loading : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5K\clark.db  Table: Assess_Forex & 2x Country > AUD
          Date CodeIn  ...    CurrencyNameOut      Rate
0   2017-05-05    AUD  ...  Australian dollar  1.000000
1   2017-05-05    AUD  ...  Australian dollar  1.000000
2   2017-05-05    AUD  ...  Australian dollar  1.000000
3   2017-05-05    AUD  ...  Australian dollar  1.000000
4   2017-05-05    AUD  ...  Australian dollar  1.000000
..         ...    ...  ...               ...       ...
79  2017-05-05    AUD  ...        Swiss franc  0.730785
80  2017-05-05    AUD  ...        Swiss franc  0.730785
81  2017-05-05    AUD  ...        Swiss franc  0.730785
82  2017-05-05    AUD  ...        Swiss franc  0.730785
83  2017-05-05    AUD  ...        Swiss franc  0.730785

[84 rows x 8 columns]
Storing : F:\DS_Practicals_Msc-it\Practical no 5\practicalno5K\clark.db  Table: Assess_Forex_AUD
Rows :  84
Done
```

Practical no 4 L Aim;-Write a Python program to process the balance sheet to ensure that only good data is processing

1) Financial

Code;import sqlite3
as sq import pandas
as pd

```
sInputFileName='F:\\DS_Practicals_Msc-it\\Practical no 5\\Practicalno5L\\Retrieve_Profit_And_Loss.csv'
sDatabaseName='F:\\DS_Practicals_Msc-it\\Practical no 5\\Practicalno5L\\clark.db'
conn = sq.connect(sDatabaseName) sFileName=sInputFileName print('Loading :',sFileName)
FinancialRawData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
FinancialData=FinancialRawData
print('Loaded Company :',FinancialData.columns.values)
sTable='Assess-Financials' print('Storing :',sDatabaseName,' Table:',sTable)
FinancialData.to_sql(sTable, conn, if_exists="replace")
print(FinancialData.head()) print('Rows : ',FinancialData.shape[0]) print('Done')
```

Output:-

```
================================================ RESTART: F:\DS_Practicals_Msc-it\Practical no 5\Pract
Loading : F:\DS_Practicals_Msc-it\Practical no 5\Practicalno5L\Retrieve_Profit_And_Loss.csv
Loaded Company : ['QTR' 'TypeOfEntry' 'ProductClass1' 'ProductClass2' 'ProductClass3'
 'Amount' 'QTY']
Storing : F:\DS_Practicals_Msc-it\Practical no 5\Practicalno5L\clark.db  Table: Assess-Financials
       QTR    TypeOfEntry ProductClass1  ... ProductClass3    Amount       QTY
0  2017Q02  Cost of Sales   Hot Blanket  ...           NaN   2989.20   12000.0
1  2017Q02  Cost of Sales     Kitty Box  ...           NaN  19928.00   30000.0
2  2017Q02  Cost of Sales      Maxi Dog  ...           NaN  34874.00   15000.0
3  2017Q02  Cost of Sales     Muis Huis  ...           NaN  29892.00    4000.0
4  2017Q02  Cost of Sales     Water Jug  ...           NaN    199.28  180000.0

[5 rows x 7 columns]
Rows :  2442
Done
```

2) financial_calendar Code:-

```
import sqlite3 as sq
import pandas as pd
sDatabaseNameIn='F:\\DS_Practicals_Msc-it\\Practical no 5\\Practicalno5L\\clark.db'
connIn = sq.connect(sDatabaseNameIn)
sDatabaseNameOut='F:\\DS_Practicals_Msc-it\\Practical no 5\\Practicalno5L\\clark.db'
connOut = sq.connect(sDatabaseNameOut) sTableIn='Retrieve_Date' sSQL='select * FROM ' + sTableIn + ';'
sTableOut='Assess_Time'
print('Loading :',sDatabaseNameIn,' Table:',sTableIn)
dateRawData=pd.read_sql_query(sSQL, connIn)
dateData=dateRawData
print('Load Rows : ',dateRawData.shape[0], ' records')
dateData.drop_duplicates(subset='FinDate', keep='first', inplace=True)
sTableOut='Assess_Date' print('Storing :',sDatabaseNameOut,' Table:',sTableOut) dateData.to_sql(sTableOut, connOut, if_exists="replace") print('Store Rows : ',dateData.shape[0], ' records')
sTableIn='Retrieve_Time' sSQL='select * FROM ' + sTableIn + ';'
sTableOut='Assess_Time' print('Loading :',sDatabaseNameIn,' Table:',sTableIn) timeRawData=pd.read_sql_query(sSQL, connIn)
timeData=timeRawData
```

print('Load Rows : ',timeData.shape[0], ' records')
timeData.drop_duplicates(subset=None, keep='first', inplace=True)
sTableOut='Assess_Time' print('Storing :',sDatabaseNameOut,'
Table:',sTableOut) timeData.to_sql(sTableOut, connOut,
if_exists="replace") print('Store Rows : ',timeData.shape[0], '
records') print('Done') Output:-

```
========== RESTART: C:\VKHCG\04-Clark\02-Assess\Assess-Calendar.py =======
################################
Working Base : C:/VKHCG  using  win32
################################
################
Loading : C:/VKHCG/04-Clark/01-Retrieve/SQLite/clark.db  Table: Retrieve_I
################################
Load Rows :  50406  records
################################
################
Storing : C:/VKHCG/04-Clark/01-Retrieve/SQLite/clark.db  Table: Assess_Dat
################
################################
Store Rows :  43830  records
################################
################
Loading : C:/VKHCG/04-Clark/01-Retrieve/SQLite/clark.db  Table: Retrieve_T
################################
Load Rows :  7196  records
################################
################
Storing : C:/VKHCG/04-Clark/01-Retrieve/SQLite/clark.db  Table: Assess_Tim
################
################################
Store Rows :  1440  records
################################
### Done!! ##############################################
```

Practical no 4 M Aim;-Write a Python program to generate payroll from the given data. Code:-

```
import sqlite3 as sq
import pandas as pd
Company = '04-Clark'
sInputFileName1 = 'F:\\DS_Practicals_Msc-it\\Practical no 5\\Retrieve-Data_female-names.csv'
sInputFileName2 = 'F:\\DS_Practicals_Msc-it\\Practical no 5\\Retrieve-Data_male-names.csv'
sInputFileName3 = 'F:\\DS_Practicals_Msc-it\\Practical no 5\\Retrieve-Data_last-names.csv'
sOutputFileName1 = 'F:\\DS_Practicals_Msc-it\\Practical no 5\\Assess-Staff.csv'
sOutputFileName2 = 'F:\\DS_Practicals_Msc-it\\Practical no 5\\Assess-Customers.csv'
sDatabaseName ='F:\\DS_Practicals_Msc-it\\Practical no 5\\clark.db'
conn = sq.connect(sDatabaseName)
sFileName =sInputFileName1
print('Loading :', sFileName)
FemaleRawData = pd.read_csv(sFileName, header=0, low_memory=False, encoding="latin-1")
FemaleRawData.rename(columns={'NameValues': 'FirstName'}, inplace=True)
FemaleRawData.drop_duplicates(subset=None, keep='first', inplace=True)
FemaleData = FemaleRawData.sample(100)
sTable = 'Assess_FemaleName'
print('Storing :', sDatabaseName, ' Table:', sTable)
FemaleData.to_sql(sTable, conn, if_exists="replace")
```

```
print('Rows:', FemaleData.shape[0], 'records')
sFileName =sInputFileName2
print('Loading :', sFileName)
MaleRawData = pd.read_csv(sFileName, header=0, low_memory=False, encoding="latin-1")
MaleRawData.rename(columns={'NameValues': 'FirstName'}, inplace=True)
MaleRawData.drop_duplicates(subset=None, keep='first', inplace=True)
MaleData = MaleRawData.sample(100)
sTable = 'Assess_MaleName'
print('Storing :', sDatabaseName, ' Table:', sTable)
MaleData.to_sql(sTable, conn, if_exists="replace")
print('Rows:', MaleData.shape[0], 'records')
sFileName =sInputFileName3
print('Loading :', sFileName)
SurnameRawData = pd.read_csv(sFileName, header=0, low_memory=False, encoding="latin-1")
SurnameRawData.rename(columns={'NameValues': 'LastName'}, inplace=True)
SurnameRawData.drop_duplicates(subset=None, keep='first', inplace=True)
SurnameData = SurnameRawData.sample(200) sTable
= 'Assess_Surname' print('Storing :', sDatabaseName, '
Table:', sTable) SurnameData.to_sql(sTable, conn,
if_exists="replace") print('Rows:',
SurnameData.shape[0], 'records') sTable =
'Assess_FemaleName & Assess_MaleName'
print('Loading :', sDatabaseName, ' Table:', sTable)
sSQL = """
	SELECT DISTINCT
		A.FirstName,
		'Female' AS Gender
	FROM
		Assess_FemaleName AS A
	UNION
	SELECT DISTINCT
		A.FirstName,
		'Male' AS Gender
	FROM
		Assess_MaleName AS A;
"""
FirstNameData = pd.read_sql_query(sSQL, conn)


sTable = 'Assess_FirstName'
print('Storing :', sDatabaseName, ' Table:', sTable)
FirstNameData.to_sql(sTable, conn, if_exists="replace")



sTable = 'Assess_FirstName x2 & Assess_Surname'
print('Loading :', sDatabaseName, ' Table:', sTable)
sSQL = """
```

```
        SELECT DISTINCT
            A.FirstName,
            B.FirstName AS SecondName,
            C.LastName,
            A.Gender
        FROM
            Assess_FirstName AS A,
            Assess_FirstName AS B,
            Assess_Surname AS C
        WHERE
            A.Gender = B.Gender
        AND
            A.FirstName <> B.FirstName;
"""
PeopleRawData = pd.read_sql_query(sSQL, conn)
People1Data = PeopleRawData.sample(10000)


sTable = 'Assess_FirstName & Assess_Surname'
print('Loading :', sDatabaseName, ' Table:', sTable)
sSQL = """
        SELECT DISTINCT
            A.FirstName,
            '' AS SecondName,
            B.LastName,
            A.Gender
        FROM
            Assess_FirstName AS A,
            Assess_Surname AS B;
"""PeopleRawData = pd.read_sql_query(sSQL, conn)
People2Data = PeopleRawData.sample(10000)
PeopleData = pd.concat([People1Data, People2Data], ignore_index=True)
print(PeopleData)
    sTable = 'Assess_People'
print('Storing :', sDatabaseName, ' Table:', sTable)
PeopleData.to_sql(sTable, conn, if_exists="replace")
    sOutputFileName = "Practical no 5\\" +sTable + '.csv'
sFileName = sOutputFileName
print('Storing :', sFileName)
PeopleData.to_csv(sFileName, index=False)

print('Done') Output:-
```

```
================================================== RESTART: F:\DS_Practicals_Msc-it\Practical no 5\Prac
Loading : F:\DS_Practicals_Msc-it\Practical no 5\Retrieve-Data_female-names.csv
Storing : F:\DS_Practicals_Msc-it\Practical no 5\clark.db  Table: Assess_FemaleName
Rows: 100 records
Loading : F:\DS_Practicals_Msc-it\Practical no 5\Retrieve-Data_male-names.csv
Storing : F:\DS_Practicals_Msc-it\Practical no 5\clark.db  Table: Assess_MaleName
Rows: 100 records
Loading : F:\DS_Practicals_Msc-it\Practical no 5\Retrieve-Data_last-names.csv
Storing : F:\DS_Practicals_Msc-it\Practical no 5\clark.db  Table: Assess_Surname
Rows: 200 records
Loading : F:\DS_Practicals_Msc-it\Practical no 5\clark.db  Table: Assess_FemaleName & Assess_MaleName
Storing : F:\DS_Practicals_Msc-it\Practical no 5\clark.db  Table: Assess_FirstName
Loading : F:\DS_Practicals_Msc-it\Practical no 5\clark.db  Table: Assess_FirstName x2 & Assess_Surname
Loading : F:\DS_Practicals_Msc-it\Practical no 5\clark.db  Table: Assess_FirstName & Assess_Surname
        FirstName SecondName  LastName  Gender
0      Florentina     Sherie   Eastman  Female
1           Angel         Fe   Belcher  Female
2           Janel     Jesusa     Bryan  Female
3           Angel    Chelsey    Thorne  Female
4        Benedict      Merle  Aldridge    Male
...           ...        ...       ...     ...
19995     Sheldon                Toro    Male
19996      Jesusa              Wallis  Female
19997       Twyla               Gates  Female
19998       Tresa              Cooley  Female
19999        Sook              Thorne  Female

[20000 rows x 4 columns]
Storing : F:\DS_Practicals_Msc-it\Practical no 5\clark.db  Table: Assess_People
Storing : Practical no 5\Assess_People.csv
```

	A	B	C	D
1	FirstName	SecondNan	LastName	Gender
2	Casey	Harland	Finnegan	Male
3	Alleen	Shirley	Almond	Female
4	Diego	Hershel	Etheridge	Male
5	Lesha	Iona	Otis	Female
6	Alexander	Cruz	Pike	Male
7	Gary	Britta	Negrete	Female
8	Nettie	Hye	Pauley	Female
9	Kerry	Alexander	Cash	Male
10	Arielle	Robbie	Gorman	Female
11	Nestor	Casey	Fogle	Male
12	Concepcior	Melodie	Harry	Female
13	Tamesha	Robbie	Sears	Female
14	See	Shirley	Mcinnis	Female

Practival no 5

Aim:-Build the time hub, links and satellites.

Code:import
os
from datetime import datetime, timedelta
import pandas as pd
import sqlite3 as sq
import uuid
from pytz import timezone, all_timezones
Base = 'F://MCSIT add documentsDS_Practicals_Msc-it//Practical no 6//Build_time_hub_links_and_satellites'
Company = '01-Vermeulen'

```
sDataBaseDir = os.path.join(Base, Company, '03-Process', 'SQLite')
sDataVaultDir = os.path.join(Base, '88-DV')
os.makedirs(sDataBaseDir, exist_ok=True)
os.makedirs(sDataVaultDir, exist_ok=True) conn1 =
sq.connect(os.path.join(sDataBaseDir, 'Hillman.db')) conn2 =
sq.connect(os.path.join(sDataVaultDir, 'datavault.db'))
base = datetime(2018, 1, 1, 0, 0, 0)
numUnits = 7 * 24
date_list = [base - timedelta(hours=x) for x in range(0, numUnits)]
batch_size = 100 time_data = [] for t, i in enumerate(date_list,
start=1):
    now_utc = i.replace(tzinfo=timezone('UTC')) sDateTime =
    now_utc.strftime("%Y-%m-%d %H:%M:%S")
    sDateTimeKey = sDateTime.replace(' ', '-').replace(':', '-')
    IDNumber = str(uuid.uuid4())

    TimeLine = {
        'ZoneBaseKey': 'UTC',
        'IDNumber': IDNumber,
        'nDateTimeValue': now_utc,
        'DateTimeValue': sDateTime,
        'DateTimeKey': sDateTimeKey
    }
    time_data.append(TimeLine)    print(f"Processed    DateTime:
    {sDateTime} (ID: {IDNumber})") if t % batch_size == 0 or t ==
    len(date_list):
        TimeFrame = pd.DataFrame(time_data)
        TimeFrame.set_index(['IDNumber'], inplace=True)
        TimeFrame.to_sql('Process-Time', conn1, if_exists="append", index=True)
        time_data.clear()

if 'IDNumber' in TimeFrame.columns:
    TimeHub = TimeFrame[['IDNumber', 'ZoneBaseKey', 'DateTimeKey', 'DateTimeValue']]
    TimeHubIndex = TimeHub.set_index(['IDNumber'], inplace=False)
    TimeHubIndex.to_sql('Hub-Time', conn2, if_exists="replace", index=True)
else:
    print("Error: 'IDNumber' not found in the columns of TimFrame")
for zone in all_timezones:
    time_zone_data = [] for t, row in TimeFrame.iterrows(): now_date =
    row['nDateTimeValue'] DateTimeKey = row['DateTimeKey'] now_utc
    = now_date.replace(tzinfo=timezone('UTC')) sDateTime =
    now_utc.strftime("%Y-%m-%d %H:%M:%S") now_zone =
    now_utc.astimezone(timezone(zone)) sZoneDateTime =
    now_zone.strftime("%Y-%m-%d %H:%M:%S")

        IDZoneNumber = str(uuid.uuid4())
        TimeZoneLine = {
            'ZoneBaseKey': 'UTC',
```

```
                'IDZoneNumber': IDZoneNumber,
                'DateTimeKey': DateTimeKey,
                'UTCDateTimeValue': sDateTime,
                'Zone': zone,
                'DateTimeValue': sZoneDateTime
            }
            time_zone_data.append(TimeZoneLine)
        TimeZoneFrame = pd.DataFrame(time_zone_data)
        TimeZoneFrame.set_index(['IDZoneNumber'], inplace=True)
        TimeZoneFrame.to_sql(f'Process-Time-{zone.replace("/", "-").replace(" ", "")}', conn1,
if_exists="replace", index=True)
        TimeZoneFrame.to_sql(f'Satellite-Time-{zone.replace("/", "-").replace(" ", "")}', conn2,
if_exists="replace", index=True) sq.execute("VACUUM;", conn1) sq.execute("VACUUM;",
conn2)

print('Done')
```

Output:-

```
========================= RESTART: F:\DS_Practicals_Msc-it\Practical no 6\Build_time_hub_links_and_s
Processed DateTime: 2018-01-01 00:00:00 (ID: 21ba70bc-f951-46cb-beda-2fff02344402)
Processed DateTime: 2017-12-31 23:00:00 (ID: b869527e-e0ba-4027-9dbb-05dfeb655bcb)
Processed DateTime: 2017-12-31 22:00:00 (ID: 540d856c-b2cd-4d26-b6b5-7d203a5254ea)
Processed DateTime: 2017-12-31 21:00:00 (ID: dec4fd50-118c-4f56-be5e-de196e0c057d)
Processed DateTime: 2017-12-31 20:00:00 (ID: 80a16798-19c2-4b3a-a0f9-18463581fb46)
Processed DateTime: 2017-12-31 19:00:00 (ID: 7c183e5c-48b2-4d22-b83b-7695fcc27b08)
Processed DateTime: 2017-12-31 18:00:00 (ID: 5d0976c5-21b9-48f3-a80d-482fa9fd97b8)
Processed DateTime: 2017-12-31 17:00:00 (ID: fd732a34-55e3-4f22-90a0-4516d0943a18)
Processed DateTime: 2017-12-31 16:00:00 (ID: 21858a51-768e-4236-be91-f433569e1513)
Processed DateTime: 2017-12-31 15:00:00 (ID: 5b40554f-6030-4a16-8509-9cc8e725aa3b)
Processed DateTime: 2017-12-31 14:00:00 (ID: e213e170-53b5-4d5f-8bdd-14d9f33ee9c3)
Processed DateTime: 2017-12-31 13:00:00 (ID: 19392b01-71ff-4a90-af76-c161d518ace4)
Processed DateTime: 2017-12-31 12:00:00 (ID: 4f723398-2c7c-4c26-8246-bb32dbf81c56)
Processed DateTime: 2017-12-31 11:00:00 (ID: a5032ce8-6817-4848-b99e-4a0fafc74250)
Processed DateTime: 2017-12-31 10:00:00 (ID: f16f0b0e-a719-4a1c-adfa-e618de9e8335)
Processed DateTime: 2017-12-31 09:00:00 (ID: 8c97e59a-0939-4dc1-8c34-1348e32a81fa)
Processed DateTime: 2017-12-31 08:00:00 (ID: 755c0662-c261-4a67-b66a-0fd5d9b54114)
Processed DateTime: 2017-12-31 07:00:00 (ID: 278328a4-56ef-4324-95da-6a42678f81ef)
Processed DateTime: 2017-12-31 06:00:00 (ID: 829d2d0a-61f4-465c-b9d7-3cc80cea4efe)
Processed DateTime: 2017-12-31 05:00:00 (ID: 77c84e7b-57ce-40d8-9bff-967acb5dd737)
Processed DateTime: 2017-12-31 04:00:00 (ID: 2d66343a-2e36-4e42-acc1-c56f161b93b9)
```

1)Vehicle

Code:

```
import os import pandas as pd import
sqlite3 as sq import uuid from pandas.io import
sql pd.options.mode.chained_assignment =
None Base="F:\\DS_Practicals_Msc-it\\Practical
no 6"
sDataBaseDir = os.path.join(Base,'vehicle', '03-Process/SQLite')
os.makedirs(sDataBaseDir, exist_ok=True)
sDatabaseName = os.path.join(sDataBaseDir, 'Hillman.db')
conn1 = sq.connect(sDatabaseName)
sDataVaultDir = os.path.join(Base,'vehicle', '88-DV')
os.makedirs(sDataVaultDir, exist_ok=True)
sDatabaseName = os.path.join(sDataVaultDir, 'datavault.db')
conn2 = sq.connect(sDatabaseName)
```

```
sFileName = "F:\\DS_Practicals_Msc-it\\Practical no 6\\VehicleData.csv"
print('Loading:', sFileName)
VehicleRaw = pd.read_csv(sFileName, header=0, low_memory=False, encoding="latin-1")


sTable = 'Process_Vehicles' print('Storing:',
sDatabaseName, 'Table:', sTable)
VehicleRaw.to_sql(sTable, conn1, if_exists="replace")


VehicleRawKey = VehicleRaw[['Make', 'Model']].copy()
VehicleKey = VehicleRawKey.drop_duplicates()
VehicleKey['ObjectKey'] = VehicleKey.apply(lambda row: f"({row['Make'].strip().replace(' ',
'').replace('/', '-').lower()})-({row['Model'].strip().replace(' ', '-').replace('/', '-').lower()})", axis=1)
VehicleKey['ObjectType'] = 'vehicle'
VehicleKey['ObjectUUID'] = VehicleKey.apply(lambda row: str(uuid.uuid4()), axis=1)


VehicleHub = VehicleKey[['ObjectType', 'ObjectKey', 'ObjectUUID']].copy()
VehicleHub.index.name = 'ObjectHubID' sTable =
'Hub-Object-Vehicle' print('Storing:', sDatabaseName,
'Table:', sTable) VehicleHub.to_sql(sTable, conn2,
if_exists="replace")


VehicleSatellite = VehicleKey[['ObjectType', 'ObjectKey', 'ObjectUUID', 'Make',
'Model']].copy()
VehicleSatellite.index.name = 'ObjectSatelliteID' sTable =
'Satellite-Object-Make-Model' print('Storing:',
sDatabaseName, 'Table:', sTable)
VehicleSatellite.to_sql(sTable, conn2, if_exists="replace")
sView = 'Dim-Object'
print('Storing:', sDatabaseName, 'View:', sView)
sSQL = f"""
    CREATE VIEW IF NOT EXISTS [{sView}] AS
    SELECT DISTINCT
        H.ObjectType,
        H.ObjectKey AS VehicleKey,
        TRIM(S.Make) AS VehicleMake,
        TRIM(S.Model) AS VehicleModel
    FROM
        [Hub-Object-Vehicle] AS H
    JOIN
        [Satellite-Object-Make-Model] AS S
    ON
        H.ObjectType = S.ObjectType
    AND
        H.ObjectUUID = S.ObjectUUID;
```

```
"""
sql.execute(sSQL, conn2)
print('Loading:', sDatabaseName, 'Table:', sView)
sSQL = f"""
    SELECT DISTINCT
        VehicleMake,
        VehicleModel
    FROM
        [{sView}]
    ORDER BY
        VehicleMake, VehicleModel;
"""
DimObjectData = pd.read_sql_query(sSQL, conn2)
DimObjectData.index.name = 'ObjectDimID'
DimObjectData.sort_values(['VehicleMake', 'VehicleModel'], inplace=True, ascending=True)

print(DimObjectData)
print('Vacuum Databases')
sql.execute("VACUUM;", conn1)
sql.execute("VACUUM;", conn2)
conn1.close() conn2.close() print('Done') Output:
```

```
Loading: F:\DS_Practicals_Msc-it\Practical no 6\vehicle\88-DV\datavault.db Table: Dim-Object
                VehicleMake                      VehicleModel
ObjectDimID
0                AM General                DJ Po Vehicle 2WD
1                AM General                 FJ8c Post Office
2                AM General              Post Office DJ5 2WD
3                AM General              Post Office DJ8 2WD
4           ASC Incorporated                             GNX
...                      ...                             ...
3880                   smart              fortwo convertible
3881                   smart                    fortwo coupe
3882                   smart   fortwo electric drive cabriolet
3883                   smart fortwo electric drive convertible
3884                   smart       fortwo electric drive coupe

[3885 rows x 2 columns]
Vacuum Databases
```

3)

HumanEnvironmentInteraction

```
Code:import sys import os import pandas as pd
import sqlite3 as sq import uuid conn1 =
sq.connect('Practical no 6\\Vermeulen.db')
conn2 = sq.connect('Practical no 6\\datavault.db')
t = 0 tMax = 360
* 180
LocationFrame = pd.DataFrame() for
Longitude in range(-180, 180, 10):
    for Latitude in range(-90, 90, 10):
        t += 1
        IDNumber = str(uuid.uuid4())
        LocationName = f'L{round(Longitude, 3) * 1000:07d}-{round(Latitude, 3) *
1000:07d}' print(f'Create: {t} of {tMax}:
        {LocationName}')
```

```
        LocationLine = {
            'ObjectBaseKey': ['GPS'],
            'IDNumber': [IDNumber],
            'LocationNumber': [str(t)],
            'LocationName': [LocationName],
            'Longitude': [Longitude],
            'Latitude': [Latitude]          }
        LocationRow = pd.DataFrame(LocationLine)
        LocationFrame = pd.concat([LocationFrame, LocationRow], ignore_index=True)
LocationHubIndex = LocationFrame.set_index(['IDNumber'], inplace=False)
sTable = 'Process-Location'
print(f'Storing: Datavault.db Table: {sTable}')
LocationHubIndex.to_sql(sTable, conn1, if_exists="replace")
sTable = 'Hub-Location'
print(f'Storing: Datavault.db Table: {sTable}')
LocationHubIndex.to_sql(sTable, conn2, if_exists="replace")
print('Vacuum Databases') conn1.execute("VACUUM;")
conn2.execute("VACUUM;")
print('Done')
```

```
Create: 643 of 64800: L0170000-0030000
Create: 644 of 64800: L0170000-0040000
Create: 645 of 64800: L0170000-0050000
Create: 646 of 64800: L0170000-0060000
Create: 647 of 64800: L0170000-0070000
Create: 648 of 64800: L0170000-0080000
Storing: Datavault.db Table: Process-Location
Storing: Datavault.db Table: Hub-Location
Vacuum Databases
Done
```

4)Forecasting

Code:

```
import sys
import os import
sqlite3 as sq import
quandl import pandas
as pd # Base directory
setup
BASE_DIR = 'PRACTICALS'
COMPANY = 'Practical no 6'
print('##################################')
print('Working Base:', BASE_DIR, 'using', sys.platform)
print('##################################')
# File paths and directories
INPUT_FILE = 'VKHCG_Shares.csv'
OUTPUT_FILE = 'Shares.csv'
# Directory paths for storing processed data
PROCESS_DIR = f"{BASE_DIR}/{COMPANY}/03-Process/SQLite"
RETRIEVE_DIR = f"{BASE_DIR}/{COMPANY}/01-Retrieve/01-EDS/02-Python"
ASSESS_DIR = f"{BASE_DIR}/{COMPANY}/02-Assess/01-EDS/02-Python"
PROCESS_PYTHON_DIR = f"{BASE_DIR}/{COMPANY}/03-Process/01-EDS/02-Python"
```

```
# Ensure directories exist
os.makedirs(PROCESS_DIR, exist_ok=True)
os.makedirs(RETRIEVE_DIR, exist_ok=True)
os.makedirs(ASSESS_DIR, exist_ok=True)
os.makedirs(PROCESS_PYTHON_DIR, exist_ok=True)
# Database connection
DB_NAME = f"{PROCESS_DIR}/clark.db"
conn = sq.connect(DB_NAME) # Import
Share Names Data
input_file_path = f"F:\\DS_Practicals_Msc-it\\Practical no 6\\VKHCG_Shares.csv"
print('################################') print('Loading:', input_file_path)
print('################################')
# Read CSV data
raw_data = pd.read_csv(input_file_path, header=0, low_memory=False, encoding="latin-1")
raw_data.drop_duplicates(inplace=True) print('Rows          :', raw_data.shape[0])
print('Columns:', raw_data.shape[1]) print('################')
# Save raw data to different directories
def save_data(data, dir_path, prefix):
    output_file_path = f"{dir_path}/{prefix}{OUTPUT_FILE}"
    print('################################')
    print('Storing:', output_file_path)
    print('################################')
    data.to_csv(output_file_path, index=False)
    print('################################')
save_data(raw_data, RETRIEVE_DIR, 'Retrieve_')
save_data(raw_data, ASSESS_DIR, 'Assess_') save_data(raw_data,
PROCESS_PYTHON_DIR, 'Process_')
# Import Shares Data Details
n_shares = raw_data.shape[0]
# Fetch and process share data from Quandl
for s_share in range(n_shares):
    s_share_name = str(raw_data['Shares'][s_share])
    share_data = quandl.get(s_share_name)

    # Add 'UnitsOwn' and 'ShareCode' columns to share data
    units_own = raw_data['Units'][s_share] share_data['UnitsOwn'] = units_own
share_data['ShareCode'] = s_share_name print('################') print('Share :',
s_share_name) print('Rows :', share_data.shape[0]) print('Columns:', share_data.shape[1])
print('################') # Store share data in SQLite database s_table =
str(raw_data['sTable'][s_share]) print('Storing:', DB_NAME, 'Table:', s_table)
share_data.to_sql(s_table, conn, if_exists="replace") # Save share data to CSV in different
directories save_data(share_data, RETRIEVE_DIR, f'Retrieve_{s_table.replace("/", "-")}_')
save_data(share_data, ASSESS_DIR, f'Assess_{s_table.replace("/", "-")}_')
save_data(share_data, PROCESS_PYTHON_DIR, f'Process_{s_table.replace("/", "-")}_')
print('### Done!! ############################################')
```

```
.................................
Working Base: PRACTICALS using win32
################################
################################
Loading: F:\DS_Practicals_Msc-it\Practical no 6\VKHCG_Shares.csv
################################
Rows    : 10
Columns: 3
################
################################
Storing: PRACTICALS/Practical no 6/01-Retrieve/01-EDS/02-Python/Retrieve_Shares.csv
################################
################################
################################
Storing: PRACTICALS/Practical no 6/02-Assess/01-EDS/02-Python/Assess_Shares.csv
################################
################################
################################
Storing: PRACTICALS/Practical no 6/03-Process/01-EDS/02-Python/Process_Shares.csv
.................................
```

Practical no 6 Aim:-Transforming Data

1) Transform Superstep_A Code:-

```
from datetime import datetime
from pytz import timezone
import pandas as pd import
sqlite3 as sq import uuid
pd.options.mode.chained_assignment = None conn1 =
sq.connect("F:\\DS_Practicals_Msc-it\\Practical no 7\\Vermeulen.db") conn2 =
sq.connect("F:\\DS_Practicals_Msc-it\\Practical no 7\\datavault.db") conn3 =
sq.connect("F:\\DS_Practicals_Msc-it\\Practical no 7\\datawarehouse.db")
print('Time Category') print('UTC Time') birth_date_utc = datetime(1960, 12, 20, 10,
15, 0).replace(tzinfo=timezone('UTC')) birth_date_utc_str =
birth_date_utc.strftime("%Y-%m-%d %H:%M:%S (%Z) (%z)")
print(birth_date_utc_str)

print('Birth Date in Reykjavik:')
birth_zone = 'Atlantic/Reykjavik'
birth_date_local = birth_date_utc.astimezone(timezone(birth_zone))
birth_date_local_str = birth_date_local.strftime("%Y-%m-%d %H:%M:%S (%Z) (%z)")
print(birth_date_local_str)

time_zone_id = str(uuid.uuid4())
date_time_key = birth_date_utc.strftime("%Y-%m-%d-%H-%M-%S")
time_data = {
    'ZoneBaseKey': ['UTC'],
    'IDNumber': [time_zone_id],
    'DateTimeKey': [date_time_key],
    'UTCDateTimeValue': [birth_date_utc],
    'Zone': [birth_zone],
    'DateTimeValue': [birth_date_local_str]
} time_frame =
pd.DataFrame.from_dict(time_data)
```

```
time_hub = time_frame[['IDNumber', 'ZoneBaseKey', 'DateTimeKey', 'DateTimeValue']].set_index('IDNumber') time_hub.to_sql('Hub-Time-Gunnarsson', conn2, if_exists='replace') time_hub.to_sql('Dim-Time-Gunnarsson', conn3, if_exists='replace')

time_satellite = time_frame[['IDNumber', 'DateTimeKey', 'Zone', 'DateTimeValue']].set_index('IDNumber') zone_fix = birth_zone.replace(' ', '-').replace('/', '-') satellite_table_name = f'Satellite-Time-{zone_fix}-Gunnarsson'
time_satellite.to_sql(satellite_table_name, conn2, if_exists='replace')
time_satellite.to_sql(f'Dim-Time-{zone_fix}-Gunnarsson', conn3, if_exists='replace')
print()
print('Person Category') first_name = 'Guðmundur' last_name = 'Gunnarsson'
print(f'Name: {first_name} {last_name}')
print(f'Birth Date: {birth_date_local.strftime("%Y-%m-%d %H:%M:%S")}')
print(f'Birth Zone: {birth_zone}') print(f'UTC Birth Date: {birth_date_utc.strftime("%Y-%m-%d %H:%M:%S")}')

person_id = str(uuid.uuid4())
person_data = {
    'IDNumber': [person_id],
    'FirstName': [first_name],
    'LastName': [last_name],
    'Zone': ['UTC'],
    'DateTimeValue': [birth_date_utc_str]
} person_frame = pd.DataFrame.from_dict(person_data)
person_hub = person_frame.set_index('IDNumber')
person_hub.to_sql('Hub-Person', conn2, if_exists='replace')
person_hub.to_sql('Dim-Person', conn3, if_exists='replace') print('All Data Stored Successfully!')
```

Op:-

```
>
  ==================================================
  Time Category
  UTC Time
  1960-12-20 10:15:00 (UTC) (+0000)
  Birth Date in Reykjavik:
  1960-12-20 10:15:00 (GMT) (+0000)

  Person Category
  Name: Guðmundur Gunnarsson
  Birth Date: 1960-12-20 10:15:00
  Birth Zone: Atlantic/Reykjavik
  UTC Birth Date: 1960-12-20 10:15:00
  All Data Stored Successfully!
>
```

2) Transform Superstep_B Code:-

```
from datetime import datetime
from pytz import timezone
```

```
import pandas as pd import
sqlite3 as sq import uuid
pd.options.mode.chained_assignment = None conn1 =
sq.connect("F:\\DS_Practicals_Msc-it\\Practical no 7\\Vermeulen.db") conn2 =
sq.connect("F:\\DS_Practicals_Msc-it\\Practical no 7\\datawarehouse.db") print(")
print('Time Dimension')
BIRTH_ZONE = 'Atlantic/Reykjavik'
BIRTH_DATE_UTC = datetime(1960, 12, 20, 10, 15, 0).replace(tzinfo=timezone('UTC'))
BIRTH_DATE_UTC_STR = BIRTH_DATE_UTC.strftime("%Y-%m-%d %H:%M:%S (%Z)
(%z)") birth_date_local = BIRTH_DATE_UTC.astimezone(timezone(BIRTH_ZONE))
birth_date_local_str = birth_date_local.strftime("%Y-%m-%d %H:%M:%S (%Z) (%z)")
time_id = str(uuid.uuid4()) time_data = {
        'TimeID': [time_id],
        'UTCDate': [BIRTH_DATE_UTC.strftime("%Y-%m-%d %H:%M:%S")],
        'LocalTime': [birth_date_local.strftime("%Y-%m-%d %H:%M:%S")],
        'TimeZone': [BIRTH_ZONE]
}
time_frame = pd.DataFrame(time_data)
time_frame.set_index('TimeID').to_sql('Dim-Time', conn1, if_exists='replace')
time_frame.set_index('TimeID').to_sql('Dim-Time', conn2, if_exists='replace')
print(time_frame)
print(")
print('Dimension Person')
first_name = 'Guðmundur'
last_name = 'Gunnarsson'
person_id = str(uuid.uuid4())


person_data = {
        'PersonID': [person_id],
        'FirstName': [first_name],
        'LastName': [last_name],
        'Zone': ['UTC'],
        'DateTimeValue': [BIRTH_DATE_UTC_STR]
}
person_frame = pd.DataFrame(person_data)
person_frame.set_index('PersonID').to_sql('Dim-Person', conn1, if_exists='replace')

person_frame.set_index('PersonID').to_sql('Dim-Person', conn2, if_exists='replace')

print(person_frame)

print(")
print('Fact - Person - Time')

fact_id = str(uuid.uuid4())
fact_data = {
        'IDNumber': [fact_id],
        'IDPersonNumber': [person_id],
```

```
    'IDTimeNumber': [time_id]
}
fact_frame = pd.DataFrame(fact_data)


fact_frame.set_index('IDNumber').to_sql('Fact-Person-Time', conn1, if_exists='replace')

fact_frame.set_index('IDNumber').to_sql('Fact-Person-Time', conn2, if_exists='replace')

print(fact_frame) print('All Data Stored Successfully!')
```

Output:-

```
Time Dimension
                                TimeID  ...            TimeZone
0  75f98227-d9b1-48f5-8052-f35b47f5968a  ...  Atlantic/Reykjavik

[1 rows x 4 columns]

Dimension Person
                              PersonID  ...                      DateTimeValue
0  8958525d-da17-4f6e-81a4-77789db2c5c3  ...  1960-12-20 10:15:00 (UTC) (+0000)

[1 rows x 5 columns]

Fact - Person - Time
                              IDNumber  ...                          IDTimeNumber
0  73de456b-5b9a-4cd1-8794-8f33c93ef0cc  ...  75f98227-d9b1-48f5-8052-f35b47f5968a

[1 rows x 3 columns]
All Data Stored Successfully!
```

3) SimpleRegression

Code:

```
import pandas as pd
import sqlite3 as sq import
matplotlib.pyplot as plt import
numpy as np
from sklearn import datasets, linear_model
from sklearn.metrics import mean_squared_error, r2_score conn1 =
sq.connect("F:\\DS_Practicals_Msc-it\\Practical no 7\\Vermeulen.db") conn2 =
sq.connect("F:\\DS_Practicals_Msc-it\\Practical no 7\\datavault.db") conn3 =
sq.connect("F:\\DS_Practicals_Msc-it\\Practical no 7\\datawarehouse.db") t = 0
bmi_data = []

for height_select in range(100, 300, 10):
    for weight_select in range(30, 300, 5):
        height = round(height_select / 100,
        3) weight = int(weight_select) bmi =
        weight / (height * height)

        if bmi <= 18.5:
            bmi_result = 1
        elif 18.5 < bmi < 25:
            bmi_result = 2
```

```
            elif 25 <= bmi < 30:
                bmi_result = 3
            else:
                bmi_result = 4


            bmi_data.append({
                'PersonID': t,
                'Height': height,
                'Weight': weight,
                'BMI': bmi,
                'Indicator': bmi_result
            })
            t += 1 bmi_df = pd.DataFrame(bmi_data)
bmi_df.set_index('PersonID', inplace=True)
bmi_df.to_sql('Transform-BMI', conn1, if_exists='replace')
bmi_df.to_sql('Person-Satellite-BMI', conn2, if_exists='replace')
bmi_df.to_sql('Dim-BMI', conn3, if_exists='replace') print('BMI
Data Saved Successfully!')
fig = plt.figure()
for indicator, marker, label in zip([1, 2, 3, 4], [".", "o", "+", "^"],
                                    ["Underweight",        "Normal",
"Overweight", "Obese"]):      subset = bmi_df[bmi_df['Indicator'] == indicator]
plt.plot(subset['Height'], subset['Weight'], marker, label=label)
plt.title("BMI Curve")
plt.xlabel("Height (meters)")
plt.ylabel("Weight (kg)")
plt.legend()
plt.tight_layout()
plt.show()
diabetes = datasets.load_diabetes() diabetes_X
= diabetes.data[:, np.newaxis, 2]
diabetes_X_train = diabetes_X[:-30]
diabetes_X_test = diabetes_X[-30:]
diabetes_y_train = diabetes.target[:-30]
diabetes_y_test = diabetes.target[-30:] regr =
linear_model.LinearRegression()
regr.fit(diabetes_X_train, diabetes_y_train)
diabetes_y_pred = regr.predict(diabetes_X_test)
print('Coefficients:', regr.coef_)
print("Mean squared error: %.2f" % mean_squared_error(diabetes_y_test, diabetes_y_pred))
print('Variance score: %.2f' % r2_score(diabetes_y_test, diabetes_y_pred))
plt.scatter(diabetes_X_test, diabetes_y_test, color='black', label="Actual Data")
plt.plot(diabetes_X_test, diabetes_y_pred, color='blue', linewidth=2, label="Predicted Data")
plt.title("Diabetes - BMI vs Disease Progression")
plt.xlabel("BMI")
plt.ylabel("Disease Progression")
```

```
plt.legend()
plt.tight_layout()
plt.show()
```

Output:-

4)

4) Building_a_DataWarehouse Code:-

```
import sys import os from
datetime import datetime from
pytz import timezone import
pandas as pd import sqlite3 as
sq import uuid
pd.options.mode.chained_assignment = None conn1 =
sq.connect("F:\\DS_Practicals_Msc-it\\Practical no 7\\Vermeulen.db") conn2 =
sq.connect("F:\\DS_Practicals_Msc-it\\Practical no 7\\datavault.db")
conn3 = sq.connect("F:\\DS_Practicals_Msc-it\\Practical no 7\\datawarehouse.db")
date_query = "SELECT DateTimeValue FROM [Hub-Time];"
date_data_raw = pd.read_sql_query(date_query, conn2)
date_data = date_data_raw.head(1000)

print(date_data) print('\n
Time Dimension \n ')

time_frames = [] birth_zones = ['Atlantic/Reykjavik',
'Europe/London', 'UTC']

for i, row in date_data.iterrows():
```

```
    birth_date_utc = datetime.strptime(row['DateTimeValue'], "%Y-%m-%d %H:%M:%S").replace(tzinfo=timezone('UTC'))
    for zone in birth_zones:
        birth_date_local =
        birth_date_utc.astimezone(timezone(zone)) time_id =
        str(uuid.uuid4()) time_entry = {
            'TimeID': time_id,
            'UTCDate': birth_date_utc.strftime("%Y-%m-%d %H:%M:%S"),
            'LocalTime': birth_date_local.strftime("%Y-%m-%d %H:%M:%S"),
            'TimeZone': zone
        }
        time_frames.append(time_entry)
time_frame = pd.DataFrame(time_frames)
time_frame.set_index('TimeID').to_sql('Dim-Time', conn1, if_exists='replace')
time_frame.set_index('TimeID').to_sql('Dim-Time', conn3, if_exists='replace')
person_query = """
SELECT
    FirstName,
    LastName
FROM [Hub-Person];
"""
person_data_raw = pd.read_sql_query(person_query, conn2)
person_data = person_data_raw.head(1000) print('\n
Dimension Person \n ')
person_frames = [] for i, row in
person_data.iterrows():
    first_name = row['FirstName'] second_name
    = row.get('SecondName', '') or '' last_name =
    row['LastName'] person_id =
    str(uuid.uuid4()) person_entry = {
        'PersonID': person_id, 'FirstName': first_name, 'LastName': last_name, 'Zone':
        'UTC', } person_frames.append(person_entry) person_frame =
        pd.DataFrame(person_frames)
person_frame.set_index('PersonID').to_sql('Dim-Person', conn1, if_exists='replace')
person_frame.set_index('PersonID').to_sql('Dim-Person', conn3, if_exists='replace')
print('All Data Stored Successfully!')
```

Code:-

```
==================================
           DateTimeValue
0    2018-01-01 00:00:00
1    2017-12-31 23:00:00
2    2017-12-31 22:00:00
3    2017-12-31 21:00:00
4    2017-12-31 20:00:00
..                   ...
995  2017-11-20 13:00:00
996  2017-11-20 12:00:00
997  2017-11-20 11:00:00
998  2017-11-20 10:00:00
999  2017-11-20 09:00:00

[1000 rows x 1 columns]

 Time Dimension


 Dimension Person

All Data Stored Successfully!
```

Practical no 7 Aim:-Organizing Data

1) Horizontal Style

Code:

```
import
pandas as pd import
sqlite3 as sq
datawarehouse_db = "F:\\DS_Practicals_Msc-it\\Practical no 8\\datawarehouse.db"
datamart_db = "F:\\DS_Practicals_Msc-it\\Practical no 8\\datamart.db"
conn1 = sq.connect(datawarehouse_db)
conn2 = sq.connect(datamart_db)
sTable = "Dim-BMI"
print(f"Loading table: {sTable} from {datawarehouse_db}")
PersonFrame0 = pd.read_sql_query("SELECT * FROM [Dim-BMI];", conn1)
print(f"Filtering data from table: {sTable}")
filtered_sql = """
    SELECT PersonID, Height, Weight, bmi, Indicator
    FROM [Dim-BMI]
    WHERE Height > 1.5 AND Indicator = 1
    ORDER BY Height, Weight;
"""
PersonFrame1 = pd.read_sql_query(filtered_sql, conn1)
DimPerson = PersonFrame1.set_index(["PersonID"], inplace=False)
sTableHorizontal = "Dim-BMI-Horizontal" print(f"Storing table:
{sTableHorizontal} into {datamart_db}")
DimPerson.to_sql(sTableHorizontal, conn2, if_exists="replace")
print(f"Loading table: {sTableHorizontal} from {datamart_db}")
PersonFrame2 = pd.read_sql_query("SELECT * FROM [Dim-BMI-Horizontal];", conn2)
```

```
print(f"Full Data Set (Rows, Columns): {PersonFrame0.shape}")
print(f"Horizontal Data Set (Rows, Columns): {PersonFrame2.shape}")
conn1.close() conn2.close()
```

Output:-

```
========================================================= RESTART: F:\DS_Practicals_Msc-it\Practi
Loading table: Dim-BMI from F:\DS_Practicals_Msc-it\Practical no 8\datawarehouse.db
Filtering data from table: Dim-BMI
Storing table: Dim-BMI-Horizontal into F:\DS_Practicals_Msc-it\Practical no 8\datamart.db
Loading table: Dim-BMI-Horizontal from F:\DS_Practicals_Msc-it\Practical no 8\datamart.db
Full Data Set (Rows, Columns): (1080, 5)
Horizontal Data Set (Rows, Columns): (194, 5)
```

2)Vertical Style

Code:-

```
import pandas as pd
import sqlite3 as sq
datawarehouse_db = "F:\\DS_Practicals_Msc-it\\Practical no 8\\datawarehouse.db"
datamart_db = "F:\\DS_Practicals_Msc-it\\Practical no 8\\datamart.db"
conn1 = sq.connect(datawarehouse_db)
conn2 = sq.connect(datamart_db)
PersonFrame0 = pd.read_sql_query("SELECT * FROM [Dim-BMI];", conn1)
PersonFrame1 = pd.read_sql_query( """
        SELECT Height, Weight, Indicator
        FROM [Dim-BMI];
        """,
      conn1,
)DimPerson = PersonFrame1.set_index(["Indicator"], inplace=False)
vertical_table = "Dim-BMI-Vertical"
DimPerson.to_sql(vertical_table, conn2, if_exists="replace")
PersonFrame2 = pd.read_sql_query("SELECT * FROM [Dim-BMI-Vertical];", conn2)
print(f"Full Data Set (Rows, Columns): {PersonFrame0.shape}")
print(f"Vertical Data Set (Rows, Columns): {PersonFrame2.shape}")
conn1.close() conn2.close()
```

Output:-

```
>>
   ========================================================
   Full Data Set (Rows, Columns): (1080, 5)
   Vertical Data Set (Rows, Columns): (1080, 3)
>> D
```

3)Island Style

Code:

```
import
pandas as pd import
sqlite3 as sq
datawarehouse_db = "F:\\DS_Practicals_Msc-it\\Practical no 8\\datawarehouse.db"
datamart_db = "F:\\DS_Practicals_Msc-it\\Practical no 8\\datamart.db"
conn1 = sq.connect(datawarehouse_db)
conn2 = sq.connect(datamart_db)
PersonFrame0 = pd.read_sql_query("SELECT * FROM [Dim-BMI];", conn1)
query = """
```

```
    SELECT Height, Weight, Indicator
    FROM [Dim-BMI]
    WHERE Indicator > 2
    ORDER BY Height, Weight;
"""
PersonFrame1 = pd.read_sql_query(query, conn1)
DimPerson = PersonFrame1.set_index("Indicator", inplace=False)
vertical_table = "Dim-BMI-Vertical"
DimPerson.to_sql(vertical_table, conn2, if_exists="replace")
PersonFrame2 = pd.read_sql_query("SELECT * FROM [Dim-BMI-Vertical];", conn2) print(f"Original Data Set (Rows, Columns): {PersonFrame0.shape}") print(f"Vertical Data Set (Rows, Columns): {PersonFrame2.shape}") conn1.close() conn2.close()
```

OUtpUt:-

```
==================================================
Original Data Set (Rows, Columns): (1080, 5)
Vertical Data Set (Rows, Columns): (771, 3)
>>
```

4)Secure Vault Style

Code:

```
import
pandas as pd import
sqlite3 as sq
datawarehouse_db = "F:\\DS_Practicals_Msc-it\\Practical no 8\\datawarehouse.db"
datamart_db = "F:\\DS_Practicals_Msc-it\\Practical no 8\\datamart.db"
conn1 = sq.connect(datawarehouse_db)
conn2 = sq.connect(datamart_db)
PersonFrame0 = pd.read_sql_query("SELECT * FROM [Dim-BMI];", conn1)
query = """
    SELECT
        Height,
        Weight,
        Indicator,
        CASE Indicator
            WHEN 1 THEN 'Pip'
            WHEN 2 THEN 'Norman'
            WHEN 3 THEN 'Grant'
            ELSE 'Sam'
        END AS Name
    FROM [Dim-BMI]
    WHERE Indicator > 2
    ORDER BY Height, Weight;
"""
PersonFrame1 = pd.read_sql_query(query, conn1)
DimPerson = PersonFrame1.set_index("Indicator", inplace=False)
secure_table = "Dim-BMI-Secure"
DimPerson.to_sql(secure_table, conn2, if_exists="replace")
```

```
filtered_query = "SELECT * FROM [Dim-BMI-Secure] WHERE Name = 'Sam';"
PersonFrame2 = pd.read_sql_query(filtered_query, conn2) print(f"Original Data
Set (Rows, Columns): {PersonFrame0.shape}") print(f"Filtered Data Set (Rows,
Columns): {PersonFrame2.shape}") print("Only Sam Data:\n",
PersonFrame2.head())
conn1.close() conn2.close()
```

Output:-

```
================================================
Original Data Set (Rows, Columns): (1080, 5)
Filtered Data Set (Rows, Columns): (692, 4)
Only Sam Data:
   Indicator  Height  Weight Name
0          4     1.0      35  Sam
1          4     1.0      40  Sam
2          4     1.0      45  Sam
3          4     1.0      50  Sam
4          4     1.0      55  Sam
```

5) Association Rule Mining

Code:

```
import
pandas as pd
from mlxtend.frequent_patterns import apriori,
association_rules
InputFilePath = 'F:\\DS_Practicals_Msc-it\\Practical no 8\\Online-Retail-Billboard.xlsx'
df = pd.read_excel(InputFilePath) print(f'Data Shape: {df.shape}') df['Description'] =
df['Description'].str.strip() df.dropna(subset=['InvoiceNo'], inplace=True)
df['InvoiceNo'] = df['InvoiceNo'].astype(str) df = df[~df['InvoiceNo'].str.contains('C')]
def encode_units(x):
    return True if x > 0 else False
basket_france = (df[df['Country'] == "France"]
                .groupby(['InvoiceNo', 'Description'])['Quantity']
                .sum().unstack().reset_index().fillna(0).set_index('InvoiceNo'))
basket_france = basket_france.apply(lambda x: x > 0)
if 'POSTAGE' in basket_france.columns:
    basket_france.drop('POSTAGE', axis=1, inplace=True)
frequent_itemsets_fr = apriori(basket_france, min_support=0.07, use_colnames=True) rules_fr
=     association_rules(frequent_itemsets_fr,     metric="lift",   min_threshold=1,
num_itemsets=2)
print(rules_fr[(rules_fr['lift'] >= 6) & (rules_fr['confidence'] >= 0.8)])
product_1 = 'ALARM CLOCK BAKELIKE GREEN'
product_2 = 'ALARM CLOCK BAKELIKE RED'
print(f"{product_1}: {basket_france[product_1].sum()}")
print(f"{product_2}: {basket_france[product_2].sum()}")
basket_germany = (df[df['Country'] == "Germany"]
                 .groupby(['InvoiceNo', 'Description'])['Quantity']
                 .sum().unstack().reset_index().fillna(0).set_index('InvoiceNo'))
basket_germany = basket_germany.apply(lambda x: x > 0)
if 'POSTAGE' in basket_germany.columns:
    basket_germany.drop('POSTAGE', axis=1, inplace=True)
frequent_itemsets_de = apriori(basket_germany, min_support=0.05, use_colnames=True)
rules_de     =     association_rules(frequent_itemsets_de,     metric="lift",
min_threshold=1, num_itemsets=2)
```

print(rules_de[(rules_de['lift'] >= 4) & (rules_de['confidence'] >= 0.5)])
print("Done") **OUtput:-**

```
================================================ RESTART: F:\DS_Practica
Data Shape: (541909, 8)
                                    antecedents  ... kulczynski
2                     (ALARM CLOCK BAKELIKE GREEN)  ...   0.826814
3                       (ALARM CLOCK BAKELIKE RED)  ...   0.826814
16                 (SET/6 RED SPOTTY PAPER PLATES)  ...   0.784615
18                 (SET/6 RED SPOTTY PAPER PLATES)  ...   0.924444
19                   (SET/6 RED SPOTTY PAPER CUPS)  ...   0.924444
20  (SET/6 RED SPOTTY PAPER PLATES, SET/6 RED SPOT...  ...   0.781250
21  (SET/6 RED SPOTTY PAPER PLATES, SET/20 RED RET...  ...   0.848611
22  (SET/6 RED SPOTTY PAPER CUPS, SET/20 RED RETRO...  ...   0.877500

[8 rows x 14 columns]
ALARM CLOCK BAKELIKE GREEN: 38
ALARM CLOCK BAKELIKE RED: 37
                        antecedents  ... kulczynski
0   (PLASTERS IN TIN CIRCUS PARADE)  ...   0.538485
6        (PLASTERS IN TIN SPACEBOY)  ...   0.507937
11    (RED RETROSPOT CHARLOTTE BAG)  ...   0.654634

[3 rows x 14 columns]
Done
```

6) Create a Network Routing Diagram

Code:import pandas as pd
import networkx as nx import
matplotlib.pyplot as plt
sInputFileName = 'F:\\DS_Practicals_Msc-it\\Practical no 8\\Assess-Network-RoutingCompany.csv'
sOutputFileName1 = 'F:\\DS_Practicals_Msc-it\\Practical no 8\\Organise-Network-RoutingCompany.gml'
sOutputFileName2 = 'F:\\DS_Practicals_Msc-it\\Practical no 8\\Organise-Network-RoutingCompany.png'
sFileName ='F:\\DS_Practicals_Msc-it\\Practical no 8\\Assess-Network-Routing-Company.csv'
print(f"Loading: {sFileName}")
CompanyData = pd.read_csv(sFileName, header=0, low_memory=False, encoding="latin-1")
print(CompanyData.head())
print(f"Data Shape: {CompanyData.shape}")
G = nx.Graph() for i in
range(CompanyData.shape[0]):
    for j in range(CompanyData.shape[0]):
        Node0 = CompanyData['Company_Country_Name'][i]
        Node1 = CompanyData['Company_Country_Name'][j]
        if Node0 != Node1:
            G.add_edge(Node0, Node1)
for i in range(CompanyData.shape[0]):
    Node0 = CompanyData['Company_Country_Name'][i]
    Node1 = f"{CompanyData['Company_Place_Name'][i]} ({CompanyData['Company_Country_Name'][i]})"
    if Node0 != Node1:
        G.add_edge(Node0, Node1)
print(f"Nodes: {G.number_of_nodes()}")
print(f"Edges: {G.number_of_edges()}")
sFileName = f"{sOutputFileName1}"
print(f"Storing GML file: {sFileName}")
nx.write_gml(G, sFileName) sFileName =

```
f"{sOutputFileName2}" print(f"Storing
Graph Image: {sFileName}")
plt.figure(figsize=(15, 15)) pos = nx.spectral_layout(G, dim=2)
nx.draw_networkx_nodes(G, pos, node_color='k', node_size=10, alpha=0.8)
nx.draw_networkx_edges(G, pos, edge_color='r', arrows=False, style='dashed')
nx.draw_networkx_labels(G, pos, font_size=12, font_family='sans-serif', font_color='b')
plt.axis('off')
plt.savefig(sFileName, dpi=600)
plt.show()
print("Done")
```

Output:-

```
Loading: F:\DS_Practicals_Msc-it\Practical no 8\Assess-Network-Routing-Company.csv
  Company_Country_Code  ...      Company_Country_Name
0                   US  ...  United States of America
1                   US  ...  United States of America
2                   US  ...  United States of America
3                   US  ...  United States of America
4                   US  ...  United States of America

[5 rows x 5 columns]
Data Shape: (150, 5)
Nodes: 6
Edges: 6
Storing GML file: F:\DS_Practicals_Msc-it\Practical no 8\Organise-Network-Routing-Company.gml
Storing Graph Image: F:\DS_Practicals_Msc-it\Practical no 8\Organise-Network-Routing-Company.png
```

7) Picking Content for Billboards

Code:

```
import pandas as pd
import networkx as nx import
matplotlib.pyplot as plt import
numpy as np
sInputFileName = 'F:\\DS_Practicals_Msc-it\\Practical no 8\\Assess-DE-Billboard-Visitor.csv'
sOutputFileName1 = 'F:\\DS_Practicals_Msc-it\\Practical no 8\\Organise-Billboards.gml'
sOutputFileName2 = 'F:\\DS_Practicals_Msc-it\\Practical no 8\\Organise-Billboards.png'
sFileName = 'F:\\DS_Practicals_Msc-it\\Practical no 8\\Assess-DE-Billboard-Visitor.csv'
print(f"Loading: {sFileName}")
BillboardDataRaw = pd.read_csv(sFileName, header=0, low_memory=False,
encoding="latin1")
print(BillboardDataRaw.head())
print(f"Data Shape: {BillboardDataRaw.shape}")
BillboardData = BillboardDataRaw
sSample = list(np.random.choice(BillboardData.shape[0],
20)) G = nx.Graph() for i in sSample: for j in sSample:


        Node0 =
f"{BillboardData['BillboardPlaceName'][i]}({BillboardData['BillboardCountry'][i]})"
        Node1 =
f"{BillboardData['BillboardPlaceName'][j]}({BillboardData['BillboardCountry'][i]})"
        if Node0 != Node1:
            G.add_edge(Node0, Node1)
for i in sSample:
    Node0 =
f"{BillboardData['BillboardPlaceName'][i]}({BillboardData['VisitorPlaceName'][i]})"
```

```
    Node1 =
f"{BillboardData['BillboardPlaceName'][i]}({BillboardData['VisitorCountry'][i]})"
    if Node0 != Node1:
        G.add_edge(Node0, Node1)
print(f"Nodes: {G.number_of_nodes()}")
print(f"Edges: {G.number_of_edges()}")
sFileName = f"{sOutputFileName1}"
print(f"Storing GML file: {sFileName}")
nx.write_gml(G, sFileName) sFileName =
f"{sOutputFileName2}" print(f"Storing
Graph Image: {sFileName}")
plt.figure(figsize=(15, 15)) pos = nx.circular_layout(G, dim=2)
nx.draw_networkx_nodes(G, pos, node_color='k', node_size=150, alpha=0.8)
nx.draw_networkx_edges(G, pos, edge_color='r', arrows=False, style='solid')
nx.draw_networkx_labels(G, pos, font_size=12, font_family='sans-serif',
font_color='b') plt.axis('off')
plt.savefig(sFileName, dpi=600)
plt.show()
print("Done")
```

Output:-

```
================================================ RESTART: F:\DS_Practicals_Msc-it\Practic
Loading: F:\DS_Practicals_Msc-it\Practical no 8\Assess-DE-Billboard-Visitor.csv
  BillboardCountry BillboardPlaceName  ...  VisitorLongitude  VisitorYearRate
0               DE               Lake  ...            8.5667       26823960.0
1               DE               Horb  ...            8.6833       26823960.0
2               DE               Horb  ...            8.6833       53753112.0
3               DE               Horb  ...            8.6833      107611416.0
4               DE               Horb  ...            8.6833       13359384.0

[5 rows x 9 columns]
Data Shape: (181235, 9)
Nodes: 28
Edges: 105
Storing GML file: F:\DS_Practicals_Msc-it\Practical no 8\Organise-Billboards.gml
Storing Graph Image: F:\DS_Practicals_Msc-it\Practical no 8\Organise-Billboards.png
```

8) Create a Delivery Route

Code:

```
import pandas as pd
sOutputFileName = 'F:\\DS_Practicals_Msc-it\\Practical no 8\\Organise-Routes.csv'
sFileName = 'F:\\DS_Practicals_Msc-it\\Practical no 8\\Assess_Shipping_Routes.txt'
print(f"Loading: {sFileName}")
RouteDataRaw = pd.read_csv(sFileName, header=0, low_memory=False, sep='|', encoding="latin-1")
RouteStart = RouteDataRaw[RouteDataRaw['StartAt'] == 'WH-KA13']
RouteDistance = RouteStart[RouteStart['Cost'] == 'DistanceMiles']
RouteDistance = RouteDistance.sort_values(by=['Measure'], ascending=False)
RouteMax = RouteStart["Measure"].max()
RouteMaxCost = round(((RouteMax / 1000) * 1.5 * 2), 2)
print("Maximum (£) per day:")
print(RouteMaxCost)
RouteMean = RouteStart["Measure"].mean()
RouteMeanMonth = round(((RouteMean / 1000) * 2 * 30), 6)
print("Mean per Month (Miles):")
print(RouteMeanMonth)
```

Output:-

```
========================================================== RESTART: F:\DS_Practical
Loading: F:\DS_Practicals_Msc-it\Practical no 8\Assess_Shipping_Routes.txt
Maximum (£) per day:
21.82
Mean per Month (Miles):
21.56191
```

```
ID|Cost|StartAt|EndAt|Path|Measure
1|"DistanceMiles"|"HQ-KA13"|"WH-KA13"|"HQ-KA13,WH-KA13"|0.000000
2|"DistanceMeters"|"HQ-KA13"|"WH-KA13"|"HQ-KA13,WH-KA13"|0.000000
3|"Cost"|"HQ-KA13"|"WH-KA13"|"HQ-KA13,WH-KA13"|5.000000
4|"DistanceMiles"|"HQ-KA13"|"SP-KA13-7QJ"|"HQ-KA13,WH-KA13,SP-KA13-7QJ"|0.658000
5|"DistanceMeters"|"HQ-KA13"|"SP-KA13-7QJ"|"HQ-KA13,WH-KA13,SP-KA13-7QJ"|1059.000000
6|"Cost"|"HQ-KA13"|"SP-KA13-7QJ"|"HQ-KA13,WH-KA13,SP-KA13-7QJ"|10.493500
7|"DistanceMiles"|"HQ-KA13"|"SP-KA13-7QL"|"HQ-KA13,WH-KA13,SP-KA13-7QL"|1.371000
8|"DistanceMeters"|"HQ-KA13"|"SP-KA13-7QL"|"HQ-KA13,WH-KA13,SP-KA13-7QL"|2206.000000
```

9)Simple Forex Trading Planner

Code:import pandas
as pd import sqlite3
as sq import re

```
sOutputFileName = 'F:\\DS_Practicals_Msc-it\\Practical no 8\\Organise-Forex.csv'
sDatabaseName = "F:\\DS_Practicals_Msc-it\\Practical no 8\\clark.db" sFileName
= 'F:\\DS_Practicals_Msc-it\\Practical no 8\\Process_ExchangeRate.csv' conn =
sq.connect(sDatabaseName) print(f"Loading: {sFileName}")
ForexDataRaw = pd.read_csv(sFileName, sep=",", header=0, encoding="latin-1")
print("Forex Data Sample:") print(ForexDataRaw.head()) if
ForexDataRaw.shape[1] == 4:
    ForexDataRaw.columns = ["Date", "CodeIn", "CodeOut", "Rate"]else: raise
    ValueError(f"Unexpected number of columns in {sFileName}: {ForexDataRaw.shape[1]}")
ForexDataRaw.index.names = ['RowID']
sTable = 'Forex_All'
print(f"Storing in {sDatabaseName} Table: {sTable}")
ForexDataRaw.to_sql(sTable, conn, if_exists="replace", index=True)
sSQL = """ SELECT 1 as Bag,
        CAST(min(Date) AS VARCHAR(10)) as Date,
        CAST(1000000.0000000 as NUMERIC(12,4)) as Money,
        'USD' as Currency FROM Forex_All;""" sSQL =
re.sub(r"\s\s+", " ", sSQL) nMoney = pd.read_sql_query(sSQL,
conn) nMoney.index.names = ['RowID'] sTable = 'MoneyData'
print(f"Storing in {sDatabaseName} Table: {sTable}")
nMoney.to_sql(sTable, conn, if_exists="replace", index=True)
ForexDay = pd.read_sql_query("SELECT Date FROM Forex_All GROUP BY Date;", conn)
t = 0 for i in range(ForexDay.shape[0]):
    sDay = str(ForexDay['Date'][i])
    sSQL = f""" SELECT M.Bag as Bag, F.Date as
    Date, round(M.Money * F.Rate, 6) AS Money,
            F.CodeIn AS PCurrency,
            F.CodeOut AS Currency
    FROM MoneyData AS M
    JOIN (
        SELECT CodeIn, CodeOut, Date, Rate
        FROM Forex_All
        WHERE CodeIn = "USD" AND CodeOut = "GBP"
        UNION
        SELECT CodeOut AS CodeIn, CodeIn AS CodeOut, Date, (1/Rate) AS Rate
        FROM Forex_All
        WHERE CodeIn = "USD" AND CodeOut = "GBP"
    ) AS F
```

```
    ON M.Currency = F.CodeIn AND F.Date = "{sDay}";
  """ sSQL = re.sub(r"\s\s+", " ",
    sSQL)
    ForexDayRate = pd.read_sql_query(sSQL, conn)

    for j in range(ForexDayRate.shape[0]):
        sBag = str(ForexDayRate['Bag'][j]) nMoney =
        str(round(ForexDayRate['Money'][j], 2)) sCodeIn =
        ForexDayRate['PCurrency'][j] sCodeOut =
        ForexDayRate['Currency'][j] sSQL = f"""
        UPDATE MoneyData
        SET Date = "{sDay}", Money = {nMoney}, Currency = "{sCodeOut}"
        WHERE Bag = {sBag} AND Currency = "{sCodeIn}";
        """
        sSQL = re.sub(r"\s\s+", " ",
        sSQL) cur = conn.cursor()
        cur.execute(sSQL)
        conn.commit() sSQL = f"""
        INSERT INTO TransactionData (RowID, Bag, Date, Money, Currency)
        SELECT {t} AS RowID, Bag, Date, Money, Currency
        FROM MoneyData;
        """
        sSQL = re.sub(r"\s\s+", " ", sSQL)
        cur.execute(sSQL)
        conn.commit()
        t += 1
        print(f"Trade: {t}, {sDay}, {sCodeOut}, {nMoney}")
sSQL = "SELECT RowID, Bag, Date, Money, Currency FROM TransactionData ORDER BY RowID;"
sSQL = re.sub(r"\s\s+", " ", sSQL)
TransactionData = pd.read_sql_query(sSQL, conn)
OutputFile = f"{sOutputFileName}"
TransactionData.to_csv(OutputFile, index=False)
```

print("Done") **Output:-**

```
Loading: F:\DS_Practicals_Msc-it\Practical no 8\Process_ExchangeRate.csv
Forex Data Sample:
    ï»¿Date CodeIn CodeOut        Rate
0  20170505    USD     USD    1.000000
1  20170505    USD     JPY  112.480613
2  20170505    USD     BGN    1.784326
3  20170505    USD     CZK   24.432990
4  20170505    USD     DKK    6.784874
Storing in F:\DS_Practicals_Msc-it\Practical no 8\clark.db Table: Forex_All
Storing in F:\DS_Practicals_Msc-it\Practical no 8\clark.db Table: MoneyData
Trade: 1, 20170505, GBP, 772830.95
Done
```

Pratical no 8 Aim:-Generating Data

1) Vermeulen PLC

Code:import os import pandas as pd import networkx as nx import matplotlib.pyplot as

```
plt pd.options.mode.chained_assignment =
None
sInputFileName = 'F:\\DS_Practicals_Msc-it\\Practical no
9\\VermulenPLC_DATA\\AssessNetwork-Routing-Customer.csv'
sOutputFileName1 = 'F:\\DS_Practicals_Msc-it\\Practical no
9\\VermulenPLC_DATA\\ReportNetwork-Routing-Customer.gml'
sOutputFileName2 = 'F:\\DS_Practicals_Msc-it\\Practical no 9\\VermulenPLC_DATA\\Report-
Network-Routing-Customer.png'
Company = '01-Vermeulen'
sFileName = f"{sInputFileName}"
print(f"Loading: {sFileName}")
CustomerData = pd.read_csv(sFileName, header=0, low_memory=False, encoding="latin-
1").head(100)
print(f"Loaded columns: {CustomerData.columns.values}")
G = nx.Graph() for i in range(CustomerData.shape[0]):
    NodeCountry = CustomerData['Customer_Country_Name'][i]
    NodePlace = f"{CustomerData['Customer_Place_Name'][i]} ({NodeCountry})"
    NodeCoords = f"({CustomerData['Customer_Latitude'][i]:.9f},
{CustomerData['Customer_Longitude'][i]:.9f})"
    G.add_edge(NodeCountry, NodePlace)
    G.add_edge(NodePlace, NodeCoords) print(f"Nodes:
{G.number_of_nodes()}") print(f"Edges:
{G.number_of_edges()}") sFileNameGML =
f"{sOutputFileName1}" print(f"Storing graph in GML
format: {sFileNameGML}") nx.write_gml(G,
sFileNameGML) sFileNameImg =
f"{sOutputFileName2}" print(f"Storing graph image:
{sFileNameImg}")
plt.figure(figsize=(25, 25)) pos = nx.spectral_layout(G,dim=2)
nx.draw_networkx_nodes(G, pos, node_color='k', node_size=10, alpha=0.8)
nx.draw_networkx_edges(G, pos, edge_color='r',arrows=False, style='dashed')
nx.draw_networkx_labels(G, pos, font_size=12,font_family='sans-serif', font_color='b')
plt.axis('off')
plt.savefig(sFileNameImg, dpi=600)
plt.show()
print("Done")
```

Output:-

```
Loading: F:\DS_Practicals_Msc-it\Practical no 9\VermulenPLC_DATA\Assess-Network-Routing-Customer.csv
Loaded columns: ['Customer_Country_Code' 'Customer_Place_Name' 'Customer_Latitude'
 'Customer_Longitude' 'Customer_Country_Name']
Nodes: 205
Edges: 200
Storing graph in GML format: F:\DS_Practicals_Msc-it\Practical no 9\VermulenPLC_DATA\Report-Network-Routing-Customer.gml
Storing graph image: F:\DS_Practicals_Msc-it\Practical no 9\VermulenPLC_DATA\Report-Network-Routing-Customer.png
Done
>
```

2) Krennwallner AG

Code:

```
import os
import pandas as pd
from folium.plugins import FastMarkerCluster, HeatMap
from folium import Marker, Map
import webbrowser
sFileName = f"F:\\DS_Practicals_Msc-it\\Practical no 9\\KrennwallnerAG_DATA\\Retrieve_DE_Billboard_Locations.csv" df = pd.read_csv(sFileName, header=0, low_memory=False, encoding="latin-1")
df.fillna(value=0, inplace=True)
print(df.shape)
DataCluster = []
DataPoint = []

for i in range(df.shape[0]):
    try:
        sLongitude = float(df["Longitude"][i])
    except Exception:
        sLongitude = 0.0

    try:
        sLatitude = float(df["Latitude"][i])
    except Exception:
        sLatitude = 0.0

    try:
```

```
            sDescription = f"{df['Place_Name'][i]} ({df['Country'][i]})"
        except Exception:
            sDescription = 'VKHCG'
        if sLongitude != 0.0 and sLatitude != 0.0:
            DataCluster.append([sLatitude, sLongitude])
            DataPoint.append([sLatitude, sLongitude, sDescription]) pins =
pd.DataFrame(DataPoint, columns=['Latitude', 'Longitude', 'Description'])
stops_map1 = Map(location=[48.1459806, 11.4985484], zoom_start=5)
FastMarkerCluster(DataCluster).add_to(stops_map1)
sFileNameHtml1 = f"F:\\DS_Practicals_Msc-it\\Practical no
9\\KrennwallnerAG_DATA\\Billboard1.html"
stops_map1.save(sFileNameHtml1)
webbrowser.open('file://' + os.path.realpath(sFileNameHtml1))
stops_map2 = Map(location=[48.1459806, 11.4985484], zoom_start=5)
for _, row in pins.iloc[:100].iterrows():
        Marker([row["Latitude"], row["Longitude"]],
popup=row["Description"]).add_to(stops_map2)
sFileNameHtml2 = f"F:\\DS_Practicals_Msc-it\\Practical no
9\\KrennwallnerAG_DATA\\Billboard2.html"
stops_map2.save(sFileNameHtml2)
webbrowser.open('file://' + os.path.realpath(sFileNameHtml2))
stops_heatmap = Map(location=[48.1459806, 11.4985484], zoom_start=5)
stops_heatmap.add_child(HeatMap([[row["Latitude"], row["Longitude"]] for _,
row in pins.iloc[:100].iterrows()]))
sFileNameHtml3 = f"F:\\DS_Practicals_Msc-it\\Practical no
9\\KrennwallnerAG_DATA\\Billboard_heatmap.html"
stops_heatmap.save(sFileNameHtml3)
webbrowser.open('file://' + os.path.realpath(sFileNameHtml3))

print("Done")
```

Output:-

File
F:/DS_Practicals_Msc-it/Practical%20no%209/KrennwallnerAG_DATA/Billboard1.html
Amsterdam
Nederland
Utrecht
Hannover
Braunschweig
Hildesheim
Bielefeld
Münster
Hamm
Essen
Düsseldorf
Köln
Bonn
Koblenz
Frankfurt am Main
Kassel
Halle (Saale)
Chemnitz
Praha
Nürnberg
Erlangen
Würzburg
Mannheim
Heidelberg
Saarbrücken
Trier
Luxembourg
Letzebuerg
Antwerpen
Bruxelles - Brussel
Maastricht
Aachen
Szczecin
Gorzów Wielkopolski
Zielona Góra
Liberec
Jelenia Góra
Ústí nad Labem
Karlovy Vary
Plzeň
Česko
Jihlava
Brandenburg
Sachsen
Thüringen
Hessen
Niedersachsen
Nordrhein-Westfalen
Rheinland-Pfalz
Saarland
Noord-Holland
Zuid-Holland
Noord-Brabant
Gelderland
Overijssel
Drenthe
Groningen
Fryslân
Limburg
Wallonie
Severozápad
Jihozápad
województwo lubuskie
85
6
73
102
18
106
197
105
191
67
83
22
142
117
288
8
128
111
8
126
15
119
48
281
281
151
24
214
263
166
424
366
149
27
299
203
222
410
294

file:///F:/DS_Practicals_Msc-it/Practical%20no%209/KrennwallnerAG_DATA/Billboard_heatmap.html
Would you like to make Opera your everyday browser? How do I do that?
Great Britain
Leeds
Sheffield
Liverpool
Birmingham
Nottingham
Peterborough
Norwich
Cambridge
Colchester
London
Southend-on-Sea
Brighton
Portsmouth
Southampton
Exeter
Cardiff
Bath
Gloucester
Oxford
Worcester
Milton Keynes
Lincoln
Hull
Preston
Groningen
Bremerhaven
Oldenburg
Bremen
Assen
Zwolle
Haarlem
Nederland
Den Haag
Arnhem
Osnabrück
Hannover
Wolfsburg
Münster
Bielefeld
Eindhoven
Middelburg
Brugge
Dunkerque
Lille
Mons
Belgie / Belgique / Belgien
Düsseldorf
Aachen
Bonn
Siegen
Koblenz
Kassel
Deutschland
Leipzig
Berlin
Cottbus
Chóśebuż
Zielona Góra
Gorzów Wielkopolski
Słupsk
Koszalin
Chemnitz
Liberec
Praha
Pardubice
Česko
Plzeň
Frankfurt am Main
Würzburg
Nürnberg
Mannheim
Saarbrücken
Karlsruhe
Ingolstadt
Augsburg
Ulm
München
Freiburg im Breisgau
Basel
Zürich
Besançon
Letzebuerg
Guernsey
Jersey
Le Havre
Rouen
Paris
Reims
Rennes
Le Mans
Angers
Wien
Salzburg
Osterreich
Szombathely

file:///F:/DS_Practicals_Msc-it/Practical%20no%209/KrennwallnerAG_DATA/Billboard2.html
ou like to make Opera your everyday browser? How do I do that?
København
Odense
Region Syddanmark
Slupsk
Gdynia
Koszalin
Grudziądz
Gorzów Wielkopolski
Groningen
Bremerhaven
Bremen
Assen
Zwolle
Haarlem
Nederland
Den Haag
Arnhem
Wolfsburg
Middelburg
Eindhoven
Brugge
Dunkerque
Lille
Belgie / Belgique / Belgien
Mons
Koblenz
Deutschland
Leipzig
Zielona Góra
Chóśebuż
Kalisz
Chemnitz
Liberec
Praha
Pardubice
Česko
Ostrava
Plzeň
Frankfurt am Main
Letzebuerg
Nürnberg
Saarbrücken
Reims
Paris
Ingolstadt
Augsburg
Freiburg im Breisgau
München
České Budějovice
Wien
Bratislava
Győr
Zlín
Žilina

3) Hillman Ltd

Code:

```
from time
import time import
numpy as np
import matplotlib.pyplot as plt
from sklearn import manifold, datasets, decomposition,
discriminant_analysis, random_projection
digits = datasets.load_digits(n_class=6)
X, y = digits.data, digits.target def
plot_embedding(X, title):
    X = (X - X.min(0)) / (X.max(0) - X.min(0))
    plt.figure(figsize=(8, 8))
    for i in range(X.shape[0]):
        plt.text(X[i, 0], X[i, 1], str(y[i]), color=plt.cm.Set1(y[i] / 10.0), fontsize=9)
    plt.xticks([]), plt.yticks([]), plt.title(title)
plt.figure(figsize=(8, 8))
plt.imshow( np.concatenate([X[i].reshape(8, 8) for i in range(100)],
    axis=1), cmap=plt.cm.binary,
)
plt.title("Sample digits from dataset"), plt.axis("off"), plt.show()
methods = {
    "Random Projection": random_projection.SparseRandomProjection(n_components=2,
random_state=42),
    "PCA": decomposition.TruncatedSVD(n_components=2),
    "LDA": discriminant_analysis.LinearDiscriminantAnalysis(n_components=2),
    "Isomap": manifold.Isomap(n_neighbors=30, n_components=2),
    "LLE": manifold.LocallyLinearEmbedding(n_neighbors=30, n_components=2), "t-
    SNE": manifold.TSNE(n_components=2, init="pca", random_state=0),
} for name, model in
methods.items():
    print(f"Computing {name} embedding...")
    t0 = time()
    X_embedded = model.fit_transform(X, y) if name == "LDA" else model.fit_transform(X)
    plot_embedding(X_embedded, f"{name} (time {time() - t0:.2f}s)")
```

plt.show() **Output;-**

Random Projection (time 0.01s)

PCA (time 0.03s)

LDA (time 0.03s)

Isomap (time 3.38s)

LLE (time 0.18s)

t-SNE (time 2.44s)

4) Clark Ltd Code:-

```
import os import
pandas as pd import
sqlite3 as sq
from openpyxl import load_workbook
import re
DATABASE_PATH = "F:\\DS_Practicals_Msc-it\\Practical no 9\\ClarkLTD_DATA\\clark.db"
TEMPLATE_PATH = "F:\\DS_Practicals_Msc-it\\Practical no
9\\ClarkLTD_DATA\\BalanceSheet-Template.xlsx"
OUTPUT_PATH = "F:\\DS_Practicals_Msc-it\\Practical no 9\\ClarkLTD_DATA"
conn = sq.connect(DATABASE_PATH) for month in range(1, 13):
    file_path = f"F:\\DS_Practicals_Msc-it\\Practical no
9\\Clarktltdbalancesheet\\BalanceSheets{str(month).zfill(2)}.csv"
    print(f"Loading: {file_path}")
    raw_data = pd.read_csv(file_path, header=0, encoding="latin-1",
    low_memory=False) raw_data.index.names = ['RowID'] table_name =
    'BalanceSheets' if month == 1:
        raw_data.to_sql(table_name, conn, if_exists="replace")
    else: raw_data.to_sql(table_name, conn,
if_exists="append") sql_query = """ SELECT
        Year,
        Quarter,
        Country,
        Company,
        CAST(Year AS INT) || 'Q' || CAST(Quarter AS INT) AS sDate,
        Company || ' (' || Country || ')' AS sCompanyName,
        CAST(Year AS INT) || 'Q' || CAST(Quarter AS INT) || '-' || Company || '-' || Country
AS sCompanyFile FROM
    BalanceSheets
    WHERE Year IS NOT NULL
    GROUP BY Year, Quarter, Country, Company;
""" dates_data = pd.read_sql_query(sql_query,
conn) print("Fetched report data:",
dates_data.shape)
fields = pd.DataFrame([
    ['Cash', 'D16', 1],
    ['Accounts_Receivable', 'D17', 1],
    ['Doubtful_Accounts', 'D18', 1],
    ['Inventory', 'D19', 1],
    ['Temporary_Investment', 'D20', 1],
    ['Prepaid_Expenses', 'D21', 1],
    ['Long_Term_Investments', 'D24', 1],
    ['Land', 'D25', 1],
    ['Buildings', 'D26', 1],
    ['Depreciation_Buildings', 'D27', -1],
```

```
        ['Plant_Equipment', 'D28', 1],
        ['Depreciation_Plant_Equipment', 'D29', -1],
        ['Furniture_Fixtures', 'D30', 1],
        ['Depreciation_Furniture_Fixtures', 'D31', -1],
        ['Accounts_Payable', 'H16', 1],
        ['Short_Term_Notes', 'H17', 1],
        ['Current_Long_Term_Notes', 'H18', 1],
        ['Interest_Payable', 'H19', 1],
        ['Taxes_Payable', 'H20', 1],
        ['Accrued_Payroll', 'H21', 1],
        ['Mortgage', 'H24', 1],
        ['Other_Long_Term_Liabilities', 'H25', 1],
        ['Capital_Stock', 'H30', 1]
])

for i in range(dates_data.shape[0]):
        year = str(int(dates_data['Year'][i]))
        quarter = str(int(dates_data['Quarter'][i]))
        country = str(dates_data['Country'][i])
        company = str(dates_data['Company'][i])
        company_file = re.sub(r"\s+", "", dates_data['sCompanyFile'][i]) output_file =
        f"{OUTPUT_PATH}/Report-Balance-Sheet-{company_file}.xlsx"
        workbook = load_workbook(TEMPLATE_PATH)
        worksheet = workbook["Balance-Sheet"]
        worksheet['D3'] = dates_data['sDate'][i]
        worksheet['D5'] = dates_data['sCompanyName'][i]


        for field_name, cell, sign in fields.itertuples(index=False):
                field_query = f"""
                        SELECT SUM({field_name}) AS nSumTotal
                        FROM BalanceSheets
                        WHERE Year={year} AND Quarter={quarter}
                        AND Country='{country}' AND Company='{company}';
                """
                field_data = pd.read_sql_query(re.sub(r"\s+", " ", field_query),
                conn) worksheet[cell] = field_data["nSumTotal"][0] * sign
                print(f"Set {cell} to {field_name} total.")
        workbook.save(output_file)
        print(f"Report saved: {output_file}")
```

OUTPUT:-

Pie Graph

Double Pie

C:\VKHCG\01-Vermeulen\06-Report\Report_Graph_A.py

Line Graph

C:/VKHCG/01-Vermeulen/06-Report/Report_Graph_A.py

Bar Graph / Horizontal Bar Graph

C:/VKHCG/01-Vermeulen/06-Report/Report_Graph_A.py

Area Graph

C:/VKHCG/01-Vermeulen/06-Report/Report_Graph_A.py

Scatter Graph : VKHCG/03-Hillman/06-Report/Report-Scatterplot-With-Encircling.r

Hexbin:

Program : C:\VKHCG\01-Vermeulen\06-Report\Report_Graph_A.py

Kernel Density Estimation (KDE) Graph

C:\VKHCG\01-Vermeulen\06-Report\Report_Graph_B.py

Scatter Matrix Graph

C:\VKHCG\01-Vermeulen\06-Report\Report_Graph_B.py

Andrews' Curves

C:\VKHCG\01-Vermeulen\06-Report\Report_Graph_C.py

Parallel Coordinates

C:\VKHCG\01-Vermeulen\06-Report\Report_Graph_C.py

RADVIZ Method

C:\VKHCG\01-Vermeulen\06-Report\Report_Graph_C.py

Lag Plot

C:\VKHCG\01-Vermeulen\06-Report\Report_Graph_D.py

Autocorrelation Plot

C:\VKHCG\01-Vermeulen\06-Report\Report_Graph_D.py

Bootstrap Plot

C:\VKHCG\01-Vermeulen\06-Report\Report_Graph_D.py

Contour Graphs

C:\VKHCG\01-Vermeulen\06-Report\Report_Graph_G.py

3D Graphs

C:\VKHCG\01-Vermeulen\06-Report\Report_PCA_IRIS.py

(add →import matplotlib.cm as cm & Replace : plt.cm.spectral →cm.get_cmap("Spectral") at Line 44)

Practical no 9 Aim:-Data visualisation using power Bi

Case Study : Sales Data

Step 1: Connect to an Excel workbook

1. Launch Power BI Desktop.
2. From the Home ribbon, select **Get Data**. Excel is one of the **Most Common** data connections, so you can select it directly from the **Get Data** menu.

3. If you select the Get Data button directly, you can also select **File > Excel** and select **Connect**.
4. In the **Open File** dialog box, select the **Products.xlsx** file.

In this step you remove all columns except **ProductID**, **ProductName**, **UnitsInStock**, and **QuantityPerUnit**.

You can also open the Query Editor by selecting Edit Queries from the Home ribbon in Power BI Desktop. The following steps are performed in Query Editor.

1. Query Editor, select the ProductID, ProductName, QuantityPerUnit, and UnitsInStock columns (use Ctrl+Click to select more than one column, or Shift+Click to select columns that are beside each other)
2. Select Remove Columns Remove Other Columns from the ribbon, or right-click on a column header and click Remove Other Columns.

3. Change the data type of the UnitsInStock column

For the Excel workbook, products in stock will always be a whole number, so in this step you confirm the UnitsInStock column's datatype is Whole Number.

1. Select the UnitsInStock column.
2. Select the Data Type drop-down button in the Home ribbon.
3. If not already a Whole Number, select Whole Number for data type from the drop down (the Data Type: button also displays the data type for the current selection).

Task 2: Import order data from an OData feed

You import data into Power BI Desktop from the sample Northwind OData feed at the following URL, which you can copy (and then paste) in the steps below:
http://services.odata.org/V3/Northwind/Northwind.svc/

Step 1: Connect to an OData feed

1. From the Home ribbon tab in Query Editor, select Get Data.
2. Browse to the OData Feed data source.
3. In the OData Feed dialog box, paste the URL for the Northwind OData feed.
4. Select OK.

Step 2: Expand the Order_Details table

Expand the Order_Details table that is related to the Orders table, to combine the ProductID, UnitPrice, and Quantity columns from Order_Details into the Orders table.

The Expand operation combines columns from a related table into a subject table. When the query runs, rows from the related table (Order_Details) are combined into rows from the subject table (Orders).

After you expand the Order_Details table, three new columns and additional rows are added to the

Orders table, one for each row in the nested or related table.

1. In the Query View, scroll to the Order_Details column.
2. In the Order_Details column, select the expand icon ().
3. In the Expand drop-down: a. Select (Select All Columns) to clear all columns. Select ProductID, UnitPrice, and Quantity. click OK.

Step 3: Remove other columns to only display columns of interest

In this step you remove all columns except OrderDate, ShipCity, ShipCountry, Order_Details.ProductID, Order_Details.UnitPrice, and Order_Details.Quantity columns. In the previous task, you used Remove Other Columns. For this task, you remove selected columns.

In the Query View, select all columns by completing

a) Click the first column (OrderID).
b) Shift+Click the last column (Shipper).
c) Now that all columns are selected, use Ctrl+Click to unselect the following columns: OrderDate, ShipCity, ShipCountry, Order_Details.ProductID, Order_Details.UnitPrice, and Order_Details.Quantity.

Now that only the columns we want to remove are selected, right-click on any selected column header and click Remove Columns.

Step 4: Calculate the line total for each Order_Details row

Power BI Desktop lets you to create calculations based on the columns you are importing, so you can enrich the data that you connect to. In this step, you create a Custom Column to calculate the line total for each Order_Details row.

Calculate the line total for each Order_Details row:

1. In the Add Column ribbon tab, click Add Custom Column.
2. In the Add Custom Column dialog box, in the Custom Column Formula textbox, enter [Order_Details.UnitPrice] * [Order_Details.Quantity].
3. In the New column name textbox, enter LineTotal.

Step 5: Set the datatype of the LineTotal field

Right click the **LineTotal** column.

2. Select Change Type and choose Decimal Number.

Step 6: Rename and reorder columns in the query

1. In Query Editor, drag the LineTotal column to the left, after ShipCountry.

2. Remove the Order_Details. prefix from the Order_Details.ProductID, Order_Details.UnitPrice and Order_Details.Quantity columns, by double-clicking on each column header, and then deleting that text from the column name.

Task 3: Combine the Products and Total Sales queries

Power BI Desktop loads the data from the two queries

1. Once the data is loaded, select the Manage Relationships button Home ribbon
2. Select the New… button
3. When we attempt to create the relationship, we see that one already exists! As shown in the Create Relationship dialog (by the shaded columns), the ProductsID fields in each query already have an established relationship.

4. Select Cancel, and then select Relationship view in Power BI Desktop.

Task 4: Build visuals using your data

Step 1: Create charts showing Units in Stock by Product and Total Sales by Year

2. Next, drag ShipCountry to a space on the canvas in the top right. Because you selected a geographic field, a map was created automatically. Now drag LineTotal to the Values field; the circles on the map for each country are now relative in size to the LineTotal for orders shipped to that country.

Step 2: Interact with your report visuals to analyze further

INDEX

Sr. No.	Title	Page No.	Sign
Pr 1A	Design a simple linear neural network model.	2	
Pr 1B	Calculate the output of neural net using both binary and bipolar sigmoidal function.	3	
Pr 2A	Generate AND/NOT function using McCulloch-Pitts neural net.	4	
Pr 2B	Generate XOR function using McCulloch-Pitts neural net.	5	
Pr 3A	Write a program to implement Hebb's rule.	7	
Pr 3B	Write a program to implement of delta rule.	8	
Pr 4A	Write a program for Back Propagation Algorithm	9	
Pr 4B	Write a program for error Backpropagation algorithm.	12	
Pr 5A	Write a program for Hopfield Network.	13	
Pr 5B	Write a program for Radial Basis function	14	
Pr 6A	Kohonen Self organizing map	15	
Pr 6B	Adaptive resonance theory	16	
Pr 7A	Write a program for Linear separation.	17	
Pr 7B	Write a program for Hopfield network model for associative memory	18	
Pr 8A	Membership and Identity Operators | in, not in,	20	
Pr 8B	Membership and Identity Operators is, is not	20	
Pr 9A	Find ratios using fuzzy logic	21	
Pr 9B	Solve Tipping problem using fuzzy logic	22	
Pr 10A	Implementation of Simple genetic algorithm	23	

Practical

Pr 10B	Create two classes: City and Fitness using Genetic algorithm	25	

1 A

Aim:-Design a simple linear neural network model

```
Code:n = int(input("Enter number of
elements : "))
print("Enter the inputs")
inputs = [] for i in
range(0, n): ele =
float(input())
inputs.append(ele)
print(inputs) print("Enter
the weights") weights =
[] for i in range(0, n): ele
= float(input())
weights.append(ele)
print(weights)
print("The net input can be calculated as Yin = x1w1 + x2w2 + x3w3")
Yin = [] for i in range(0, n):
    Yin.append(inputs[i]*weights[i])
print(round(sum(Yin),3))
```

Output:-

Practical

```
==== RESTART: F:\practicals - Copy\practicalno1a.py ===
Enter number of elements : 3
Enter the inputs
0.3
0.5
0.6
[0.3, 0.5, 0.6]
Enter the weights
0.2
0.1
-0.3
[0.2, 0.1, -0.3]
The net input can be calculated as Yin = x1w1 + x2w2 + x3w3
-0.07
```

1 B

Aim:- Calculate the output of neural net using both binary and bipolar sigmoidal function.

Code:-

```
n = int(input("Enter number of elements : "))
print("Enter the inputs")
inputs = [] for i in
range(0, n): ele =
float(input())
inputs.append(ele)
print(inputs) print("Enter
the weights")
```

```
========================== RESTART: F:\practicals - Copy\prac
Enter number of elements : 2
Enter the inputs
0.2
0.6
[0.2, 0.6]
Enter the weights
0.3
0.7
[0.3, 0.7]
Enter Bias Value :0.45
The net input can be calculated as   Yin = x1w1 + x2w2 + x3w3
0.93
```

Practical

```
weights = [] for i in
range(0, n): ele =
float(input())
weights.append(ele)
print(weights)
bias=float(input("Enter Bias Value :"))
print("The net input can be calculated as Yin = x1w1 + x2w2 + x3w3")
Yin = [] for i in range(0, n):
   Yin.append(inputs[i]*weights[i])
print(round((sum(Yin)+bias),3))
```

Output:-

2 A

Aim :-Generate AND/NOT function using McCulloch-Pitts neural net.

Code:-

```
num=int(input("Enter no.of input :"))
w1=1 w2=1
x1=[] x2=[] for i in
range(num):
    ele1=int(input("x1= "))
    ele2=int(input("x2= "))
    x1.append(ele1)
    x2.append(ele2)
n=x1*w1
m=x2*w2
yin=[] for i in
range(num):
    yin.append(n[i]+m[i])
print("Y-IN :",yin)
yin=[] for i in
range(num):
    yin.append(n[i]-m[i])
print("After assuming one weight as excitatory and the other as inhibitory YIN
:",yin) Y=[] for i in range(num):
    if(yin[i]>=1):
        ele=1
        Y.append(ele)
    if(yin[i]<1):
    ele=0
```

Practical

```
    Y.append(ele)
print("Y :",Y)
```

Output:-

```
================ RESTART: F:\practicals - Co
py\practicalno2a.py ================
Enter no.of input :4
x1= 0
x2= 0
x1= 0
x2= 1
x1= 1
x2= 0
x1= 1
x2= 1
Y-IN : [0, 1, 1, 2]
After assuming one weight as excitatory and
the other as inhibitory Y-IN : [0, -1, 1, 0]
Y : [0, 0, 1, 0]
```

x_1	x_2	y
0	0	0
0	1	0
1	0	1
1	1	0

Practical 2 B

Aim:-Generate XOR function using McCulloch-Pitts neural net

Code:import
numpy as np
print('Enter weights:') w11 =
int(input('Weight w11=')) w12 =
int(input('Weight w12=')) w21 =
int(input('Weight w21=')) w22 =
int(input('Weight w22=')) v1 =
int(input('Weight v1=')) v2 =
int(input('Weight v2='))
print('Enter Threshold Value')
theta = int(input('Theta=')) x1 =
np.array([0, 0, 1, 1]) x2 =
np.array([0, 1, 0, 1]) z =
np.array([0, 1, 1, 0]) con = 1
y1 = np.zeros(4)
y2 = np.zeros(4)
y = np.zeros(4)
while con == 1:
zin1 = x1 * w11 + x2 * w21
zin2 = x1 * w12 + x2 * w22
print("z1:", zin1) print("z2:", zin2)
for i in range(4): y1[i] = 1 if zin1[i]
>= theta else 0 y2[i] = 1 if zin2[i]
>= theta else 0 yin = y1 * v1 + y2 *
v2 for i in range(4):
y[i] = 1 if yin[i] >= theta else 0
print("yin:", yin) print('Output of
Net:') y = y.astype(int) print("y:",
y) print("z:", z) if
np.array_equal(y, z): con = 0
else:
print("Net is not learning; enter another set of weights and Threshold value.")
w11 = int(input("Weight w11="))
w12 = int(input("Weight w12="))
w21 = int(input("Weight w21="))
w22 = int(input("Weight w22="))
v1 = int(input("Weight v1="))

```
    v2 = int(input("Weight v2="))
    theta = int(input("Theta="))
print("McCulloch-Pitts Net for XOR function")
print("Weights of Neuron Z1:", w11, w21)
print("Weights of Neuron Z2:", w12, w22)
print("Weights of Neuron Y:", v1, v2)
print("Threshold value:", theta)
```

Output:-

Input1	Input2	XOR Output
0	0	0
0	1	1
1	0	1
1	1	0

```
========================== RESTART: F:
Enter weights:
Weight w11=1
Weight w12=-1
Weight w21=-1
Weight w22=1
Weight v1=1
Weight v2=1
Enter Threshold Value
Theta=1
z1: [ 0 -1  1  0]
z2: [ 0  1 -1  0]
yin: [0. 1. 1. 0.]
Output of Net:
y: [0 1 1 0]
z: [0 1 1 0]
McCulloch-Pitts Net for XOR function
Weights of Neuron Z1: 1 -1
Weights of Neuron Z2: -1 1
Weights of Neuron Y: 1 1
Threshold value: 1
```

Practical 3 A

Aim:-: Write a program to implement Hebb's rule.

Code:import numpy as np x1 =
np.array([1, 1, 1, -1, 1, -1, 1, 1, 1]) x2 =

```
np.array([1, 1, 1, 1, -1, 1, 1, 1, 1]) y =
np.array([1, -1]) wtold = np.zeros(9)
wtnew = np.zeros(9)
b = 0
print("First input with target = 1")
for i in range(9):
    wtold[i] = wtold[i] + x1[i] * y[0]
wtnew = wtold
b = b + y[0]
print("Updated weights:", wtnew)
print("Bias value:", b) print("\nSecond
input with target = -1") for i in
range(9):
    wtnew[i] = wtold[i] + x2[i] * y[1]
b = b + y[1]
print("Updated weights:", wtnew)
print("Bias value:", b)
```

Output:-

Practical 3 B

Aim: Write a program to implement of delta rule

Code:import
numpy as np
x = np.array([float(input("Initial input: ")) for _ in range(3)]) weights
= np.array([float(input("Initial weight: ")) for _ in range(3)]) desired

```
================= RESTART: F:\practicals - Copy\practic
First input with target = 1
Updated weights: [ 1.  1.  1. -1.  1. -1.  1.  1.  1.]
Bias value: 1

Second input with target = -1
Updated weights: [ 0.  0.  0. -2.  2. -2.  0.  0.  0.]
Bias value: 0
```

```
= np.array([float(input("Desired output: ")) for _ in range(3)]) a =
float(input("Enter learning rate: ")) actual = x * weights
print("Initial actual output:", actual)
while not np.array_equal(desired, actual):
    weights += a * (desired - actual)
    actual = x * weights print("Updated
    weights:", weights) print("Updated
    actual output:", actual) print("*" * 30)
print("Final output")
print("Corrected weights:", weights)
print("Final actual output:", actual)
```

Output:-

```
================ RESTART: F:\practicals - Copy
\practicalno3b.py ================
Initial input: 1
Initial input: 1
Initial input: 1
Initial weight: 1
Initial weight: 1
Initial weight: 1
Desired output: 2
Desired output: 3
Desired output: 4
Enter learning rate: 1
Initial actual output: [1. 1. 1.]
Updated weights: [2. 3. 4.]
Updated actual output: [2. 3. 4.]
******************************
Final output
Corrected weights: [2. 3. 4.]
Final actual output: [2. 3. 4.]
```

Practical 4 A

Aim: Write a program for Back Propagation Algorithm

Code:import
numpy as np
import math

```
np.set_printoptions(precision=2)
v1 = np.array([0.6, 0.3]) v2 =
np.array([-0.1, 0.4])
w = np.array([-0.2, 0.4, 0.1])
b1 = 0.3
b2 = 0.5
x1 = 0
x2 = 1
alpha = 0.25 zin1 = round(b1 + x1 * v1[0] +
x2 * v2[0], 4)
print("Net input to z1 layer (zin1):", round(zin1, 3))
zin2 = round(b2 + x1 * v1[1] + x2 * v2[1], 4)
print("Net input to z2 layer (zin2):", round(zin2, 4))
z1 = 1 / (1 + math.exp(-zin1))
z1 = round(z1, 4) z2 = 1 / (1
+ math.exp(-zin2)) z2 =
round(z2, 4) print("Output
z1:", z1) print("Output z2:",
z2)
yin = w[0] + z1 * w[1] + z2 * w[2]
print("Net input to output layer (yin):", yin)
y = 1 / (1 + math.exp(-yin))
print("Output y:", y)
fyin = y * (1 - y) dk
= (1 - y) * fyin
print("dk:", dk)
dw1 = alpha * dk * z1
dw2 = alpha * dk * z2
dw0 = alpha * dk
print("Weight changes for output layer:")
print("dw1:", dw1) print("dw2:", dw2)
print("dw0:", dw0) din1 = dk * w[1]
din2 = dk * w[2]
print("Error in delta for hidden layer:")
print("din1:", din1)
print("din2:", din2)
fzin1 = z1 * (1 - z1)
fzin2 = z2 * (1 - z2)
```

```
d1 = din1 * fzin1
d2 = din2 * fzin2
print("d1:", d1)
print("d2:", d2)
dv11 = alpha * d1 * x1
dv21 = alpha * d1 * x2
dv01 = alpha * d1
dv12 = alpha * d2 * x1
dv22 = alpha * d2 * x2
dv02 = alpha * d2
print("Weight changes for hidden layer:")
print("dv11:", dv11) print("dv21:", dv21)
print("dv01:", dv01) print("dv12:", dv12)
print("dv22:", dv22) print("dv02:", dv02)
v1[0] = v1[0] + dv11 v1[1] = v1[1] +
dv12 v2[0] = v2[0] + dv21 v2[1] = v2[1]
+ dv22 w[1] = w[1] + dw1 w[2] = w[2] +
dw2 w[0] = w[0] + dw0 b1 = b1 + dv01
b2 = b2 + dv02
```

```
print("Final weights and biases:")
print("Updated weights v1:", v1)
print("Updated weights v2:", v2)
```

```
= RESTART: F:\practicals - Copy\practicalno4a.py
Net input to z1 layer (zin1): 0.2
Net input to z2 layer (zin2): 0.9
Output z1: 0.5498
Output z2: 0.7109
Net input to output layer (yin): 0.09101
Output y: 0.5227368084248941
dk: 0.11906907074145694
Weight changes for output layer:
dw1: 0.016366043773413254
dw2: 0.021161550597525433
dw0: 0.029767267685364234
Error in delta for hidden layer:
din1: 0.04762762829658278
din2: 0.011906907074145694
d1: 0.011788788650865037
d2: 0.0024471217110978417
Weight changes for hidden layer:
dv11: 0.0
dv21: 0.0029471971627162592
dv01: 0.0029471971627162592
dv12: 0.0
dv22: 0.0006117804277744604
dv02: 0.0006117804277744604
Final weights and biases:
Updated weights v1: [0.6 0.3]
Updated weights v2: [-0.1  0.4]
Updated weights w: [-0.17  0.42  0.12]
Updated biases b1: 0.30294719716271623  b2: 0.5006117804277744
```

```
print("Updated weights w:", w)
print("Updated biases b1:", b1, " b2:", b2)
```

Output:-

Practical

4 B

Aim: Write a Program For Error Back Propagation Algorithm (Ebpa) Learning

```
Code:impor
t math a0 =
-1 t = -1
w10 = float(input("Enter weight for the first network: "))
b10 = float(input("Enter bias for the first network: ")) w20
= float(input("Enter weight for the second network: ")) b20
= float(input("Enter bias for the second network: "))
c = float(input("Enter learning coefficient: "))
n1 = w10 * c + b10
a1 = math.tanh(n1)
n2 = w20 * a1 + b20
a2 = math.tanh(n2) e = t - a2
s2 = -2 * (1 - a2 * a2) * e s1
= (1 - a1 * a1) * w20 * s2
w21 = w20 - (c * s2 * a1)
w11 = w10 - (c * s1 * a0)
b21 = b20 - (c * s2) b11 =
b10 - (c * s1)
print("The updated weight of the first network (w11) =", w11)
print("The updated weight of the second network (w21) =", w21)
print("The updated bias of the first network (b11) =", b11)
print("The updated bias of the second network (b21) =", b21)
```

Output:-

```
==================================== RESTART: F:\practica
Enter weight for the first network: 56
Enter bias for the first network: 45
Enter weight for the second network: 12
Enter bias for the second network: 12
Enter learning coefficient: 11
The updated weight of the first network (w11) = 56.0
The updated weight of the second network (w21) = 12.0
The updated bias of the first network (b11) = 45.0
The updated bias of the second network (b21) = 12.0
```

5

Aim: Write a program for Hopfield Network

Practical

Code:-

```
import numpy as np
class HopfieldNetwork:
    def __init__(self, num_neurons):
        self.num_neurons = num_neurons
        self.weights = np.zeros((num_neurons, num_neurons))
    def train(self, patterns):
        for pattern in patterns: pattern =
            pattern.reshape(self.num_neurons, 1)
            self.weights += np.dot(pattern, pattern.T)
        np.fill_diagonal(self.weights, 0)
    def recall(self, pattern, max_iterations=10):
        pattern = pattern.copy() for _ in
        range(max_iterations):
            for i in range(self.num_neurons): net_input
                = np.dot(self.weights[i], pattern) pattern[i]
                = 1 if net_input >= 0 else -1
        return pattern
patterns = np.array([
    [1, -1, 1, -1],
    [-1, 1, -1, 1],
])
hopfield_net = HopfieldNetwork(num_neurons=patterns.shape[1])
hopfield_net.train(patterns) test_pattern = np.array([1, -1, -1, -1])
recalled_pattern = hopfield_net.recall(test_pattern)

print("Original pattern:", test_pattern)
print("Recalled pattern:", recalled_pattern)
```

Output:-

```
==== RESTART: F:\practicals - Copy\practicalno5a.py =
Original pattern: [ 1 -1 -1 -1]
Recalled pattern: [ 1 -1  1 -1]
```

Activate Windows
Go to Settings to activate Windows.

5 B

Aim:Write a program for Radial Basis function

Practical A

Code:-

```
import numpy as np from sklearn.cluster import
KMeans from sklearn.metrics import
mean_squared_error
np.random.seed(0) X =
np.random.rand(100, 2) y =
np.sin(X[:, 0]) + np.cos(X[:, 1])

def rbf(x, center, sigma=0.5): return np.exp(-np.linalg.norm(x -
    center) ** 2 / (2 * sigma ** 2))

kmeans = KMeans(n_clusters=10).fit(X)
centers = kmeans.cluster_centers_
R = np.array([[rbf(x, center) for center in centers] for x in X])
W = np.dot(np.linalg.pinv(R), y) def rbf_network(X, centers,
W, sigma=0.5):
    R = np.array([[rbf(x, center, sigma) for center in centers] for x in X])
    return np.dot(R, W)
y_pred = rbf_network(X, centers, W)
mse = mean_squared_error(y, y_pred)
print(f"Mean Squared Error: {mse}")
```

Output:-

```
================ RESTART: F:\practicals -
Mean Squared Error: 0.0003426937510061592

================ RESTART: F:\practicals -
Mean Squared Error: 0.0003426937510061592
```

6

Aim:Self-Organizing Maps

```
from minisom import MiniSom
import matplotlib.pyplot as plt
data = [
    [0.80, 0.30, 0.22, 0.03],
    [0.82, 0.50, 0.23, 0.03],
    [0.80, 0.54, 0.22, 0.03],
```

Practical

Code:-

```
    [0.80, 0.53, 0.26, 0.03],
    [0.79, 0.56, 0.22, 0.03],
    [0.75, 0.60, 0.25, 0.03],
    [0.77, 0.59, 0.22, 0.03]
]
som = MiniSom(6, 6, 4, sigma=0.3, learning_rate=0.5)
som.train_random(data, 100)
plt.imshow(som.distance_map())
plt.show() Output:-
```

A

Practical 6 B

Aim: Adaptive resonance theory

Code:import
numpy as np class
ART1:

```
    def __init__(self, num_features, num_categories, vigilance=0.75):
        self.vigilance = vigilance
        self.weights = np.ones((num_categories, num_features))
    def train(self, data):
        for sample in data:
            activations = np.dot(self.weights, sample) category =
            np.argmax(activations) if self._vigilance_test(sample, category):
            self.weights[category] = np.minimum(self.weights[category], sample)
    def _vigilance_test(self, sample, category):
        return np.dot(self.weights[category], sample) / np.sum(sample) >=
self.vigilance def predict(self, sample):
    activations = np.dot(self.weights, sample)
    category = np.argmax(activations)
        return category if self._vigilance_test(sample, category) else None
data = np.array([[1, 1, 0, 0], [1, 0, 1, 0], [0, 0, 1, 1]])
art1 = ART1(num_features=data.shape[1], num_categories=3, vigilance=0.75)
art1.train(data)
test_sample = np.array([1, 0, 1, 0]) print("Data Belongs
to cat :", art1.predict(test_sample))
```

Output:-

```
================ RESTART: F:\practicals - Copy\practicalno6b.py
Data Belongs to cat : 1
```

Practical 7

Aim: Line Separation

```
import numpy as np
import matplotlib.pyplot as plt def
create_distance_function(a, b, c):
    def distance(x, y):
        nom = a * x + b * y + c
        if nom == 0:
```

```
            pos = 0
        elif (nom < 0 and b < 0) or (nom > 0 and b > 0):
        pos = -1 else:
            pos = 1
        return (np.abs(nom) / np.sqrt(a ** 2 + b ** 2), pos)
    return distance
points = [(3.5, 1.8), (1.1, 3.9)] fig, ax
= plt.subplots()
ax.set_xlabel("Sweetness")
ax.set_ylabel("Sourness")
ax.set_xlim([-1, 6]) ax.set_ylim([-1,
8]) for index, (x, y) in
enumerate(points):
    color = "orange" if index == 0 else "yellow"
    ax.plot(x, y, "o", color=color, markersize=10)
  X = np.arange(-0.5, 5, 0.1) step = 0.05 for x in
      np.arange(0, 1 + step, step): slope =
      np.tan(np.arccos(x))
    dist4line1 = create_distance_function(slope, -1, 0)
  Y = slope * X
    results = [dist4line1(*point) for point in points]
    if results[0][1] != results[1][1]:
        ax.plot(X, Y, "g-", label="Decision Boundary")
    else:
        ax.plot(X, Y, "r-", label="Decision Boundary")
plt.show()
```

OUTPUT:-

A

Practical 7 B

Aim: Hopfield Network model of associative memory

Code:import
numpy as np
import matplotlib.pyplot as plt
class HopfieldNetwork:
    def __init__(self, n):
        self.n = n self.weights =
        np.zeros((n, n)) self.neurons
        = np.zeros(n)
    def train(self, patterns):
        num_patterns = len(patterns)
        for p in patterns: p = 2 * p -
        1
            self.weights += np.outer(p, p)
        np.fill_diagonal(self.weights, 0)
    def recall(self, input_pattern, steps=5):
        input_pattern = 2 * input_pattern -
        1 self.neurons = input_pattern for _
        in range(steps):
            for i in range(self.n): summation =
                np.dot(self.weights[i], self.neurons)
                self.neurons[i] = 1 if summation > 0 else -1
        return (self.neurons + 1) // 2 pattern1 =
np.array([1, 1, 1]) pattern2 = np.array([1, 0, 1])
hopfield_net = HopfieldNetwork(3)
hopfield_net.train([pattern1, pattern2])
noisy_input = np.array([1, 0, 1]) recalled_output =
hopfield_net.recall(noisy_input) print("Noisy
Input Pattern: ", noisy_input) print("Recalled
Pattern: ", recalled_output)
plt.figure(figsize=(8, 4))
plt.subplot(1, 2, 1)
plt.title("Original Pattern 2")
plt.imshow(pattern2.reshape(1, 3), cmap='binary', aspect='auto')
plt.subplot(1, 2, 2) plt.title("Recalled Pattern")

```
plt.imshow(recalled_output.reshape(1, 3), cmap='binary', aspect='auto')
plt.show()
```

Output:-

```
================ RESTART: F:\practicals - Copy\pr
Noisy Input Pattern:  [1 0 1]
Recalled Pattern:  [1 0 1]
```

Practical 8 A

Aim: Membership and Identity operators in, not in.

Code:
```
def overlapping(list1,
list2): c = 0 d = 0 for i in
list1: c += 1 for i in list2: d
+= 1 for i in range(0, c):
        for j in range(0, d):
            if list1[i] == list2[j]:
                return 1
    return 0
list1 = [1, 2, 3, 4, 5] list2
= [6, 7, 8, 9] if
overlapping(list1, list2):
    print("overlapping")
else: print("not
    overlapping")
```

Output:-

Practical 8B

Aim:- Membership and Identity Operators is, is not

Code:-

```
================= RESTART: F:\practicals - Copy\practicalno8a.p
not overlapping
```

```
x = 5 if (type(x)
is int):
    print("true")
else:
    print("false")
x = 5.2 if (type(x) is
not int):
    print("true")
else:
    print("false")
```

Output:-

```
================ RESTART: F:\practicals - Copy\practicalno8b.py
true
true
```

9 A

Aim:-Find the ratios using fuzzy logic

Practical

Code:from fuzzywuzzy import fuzz from fuzzywuzzy import process
s1 = "I love fuzzysforfuzzys" s2 = "I am loving fuzzysforfuzzys"
print("FuzzyWuzzy Ratio:", fuzz.ratio(s1, s2)) print("FuzzyWuzzy Partial Ratio: ", fuzz.partial_ratio(s1, s2)) print("FuzzyWuzzy Token Sort Ratio: ", fuzz.token_sort_ratio(s1, s2)) print("FuzzyWuzzy Token Set Ratio: ", fuzz.token_set_ratio(s1, s2)) print("FuzzyWuzzy WRatio: ", fuzz.WRatio(s1, s2), '\n\n') query = 'fuzzys for fuzzys'
choices = ['fuzzy for fuzzy', 'fuzzy fuzzy', 'g. for fuzzys']
print("List of ratios: ")
print(process.extract(query, choices), '\n') print("Best among the above list: ", process.extractOne(query, choices))

Output:-

```
================ RESTART: F:\practicals - Copy\practicalno9a.py ======
FuzzyWuzzy Ratio: 86
FuzzyWuzzy Partial Ratio:  86
FuzzyWuzzy Token Sort Ratio:  86
FuzzyWuzzy Token Set Ratio:  87
FuzzyWuzzy WRatio:  86


List of ratios:
[('g. for fuzzys', 95), ('fuzzy for fuzzy', 94), ('fuzzy fuzzy', 86)]

Best among the above list:  ('g. for fuzzys', 95)
```

Practical 9 B

Aim: Solve Tipping Problem using fuzzy logic

Code:-

```
#pip install scikit-fuzzy from fuzzywuzzy import fuzz from
fuzzywuzzy import process s1 = "I love fuzzysforfuzzys" s2 = "I am
loving fuzzysforfuzzys" print("FuzzyWuzzy Ratio:", fuzz.ratio(s1,
s2)) print("FuzzyWuzzy Partial Ratio: ", fuzz.partial_ratio(s1, s2))
print("FuzzyWuzzy Token Sort Ratio: ", fuzz.token_sort_ratio(s1, s2))
print("FuzzyWuzzy Token Set Ratio: ", fuzz.token_set_ratio(s1, s2))
print("FuzzyWuzzy WRatio: ", fuzz.WRatio(s1, s2), '\n\n') query =
'fuzzys for fuzzys'
choices = ['fuzzy for fuzzy', 'fuzzy fuzzy', 'g. for fuzzys']
print("List of ratios: ")
print(process.extract(query, choices), '\n') print("Best among the above
list: ", process.extractOne(query, choices))
```

Output:-

10 A

Aim: Implementation of simple genetic algorithm

Practical

Code:import
random

```
POPULATION_SIZE = 100
GENES = '''abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOP
QRSTUVWXYZ 1234567890, .-;:_!"#%&/()=?@${[]}'''
TARGET = "SANJEEV" class Individual:
    def __init__(self, chromosome):
        self.chromosome = chromosome
        self.fitness = self.cal_fitness()
    @classmethod def
    mutated_genes(cls):
        return random.choice(GENES)
    @classmethod def create_gnome(cls): return
    [cls.mutated_genes() for _ in range(len(TARGET))]
    def mate(self, partner):
        child_chromosome = [gp1 if random.random() < 0.45 else gp2 if
random.random() < 0.90 else self.mutated_genes() for gp1, gp2 in
                    zip(self.chromosome, partner.chromosome)]
        return Individual(child_chromosome)
    def cal_fitness(self):
        return sum(1 for gs, gt in zip(self.chromosome, TARGET) if gs != gt)
def main():
    generation = 1
    population = [Individual(Individual.create_gnome()) for _ in
range(POPULATION_SIZE)]
    while True: population = sorted(population, key=lambda x:
        x.fitness) if population[0].fitness == 0: break
        next_gen = population[:10] # Elitism: keep top 10%
        next_gen += [random.choice(population[:50]).mate(random.choice(population[:50])) for _ in
range(90)] population = next_gen print(f"Generation: {generation}\tString:
{".join(population[0].chromosome)}\tFitness: {population[0].fitness}")
        generation += 1
    print(f"Generation: {generation}\tString:
{".join(population[0].chromosome)}\tFitness: {population[0].fitness}")
if __name__ == '__main__': main()
```

Output:-

```
================ RESTART: F:\practicals - Copy\practi
Generation: 1   String: SQXctGN Fitness: 6
Generation: 2   String: SQXctGN Fitness: 6
Generation: 3   String: S)XctEb Fitness: 5
Generation: 4   String: SM#fEaV Fitness: 4
Generation: 5   String: SFN?hEV Fitness: 3
Generation: 6   String: SFN?hEV Fitness: 3
Generation: 7   String: S7CJEEV Fitness: 2
Generation: 8   String: SArJEEV Fitness: 1
Generation: 9   String: SArJEEV Fitness: 1
Generation: 10  String: SANJEEV Fitness: 0
>>
```

Practical

10 B

Aim:Create two classes: City and Fitness using Genetic algorithm

Code:

```
import numpy as np import random, operator, pandas as pd,
matplotlib.pyplot as plt class City:
    def __init__(self, x, y):
        self.x, self.y = x, y
    def distance(self, city): return np.sqrt((self.x - city.x) ** 2 +
        (self.y - city.y) ** 2)
    def __repr__(self): return
        f"({self.x},{self.y})"
class Fitness:
    def __init__(self, route): self.route, self.distance,
        self.fitness = route, 0, 0.0
    def routeDistance(self):
        if self.distance == 0:
            self.distance = sum(self.route[i].distance(self.route[i+1 if i+1 <
len(self.route) else 0]) for i in range(len(self.route)))
        return self.distance
    def routeFitness(self):
        if self.fitness == 0:
            self.fitness = 1 / float(self.routeDistance())
        return self.fitness
def createRoute(cityList): return
    random.sample(cityList, len(cityList))
def initialPopulation(popSize, cityList): return
    [createRoute(cityList) for _ in range(popSize)]
def rankRoutes(population):
    return sorted({i: Fitness(route).routeFitness() for i, route in
enumerate(population)}.items(), key=operator.itemgetter(1), reverse=True)
def selection(popRanked, eliteSize):
    df = pd.DataFrame(np.array(popRanked), columns=["Index","Fitness"])
    df['cum_sum'], df['cum_perc'] = df.Fitness.cumsum(),
100*df.Fitness.cumsum()/df.Fitness.sum() selectionResults =
    [popRanked[i][0] for i in range(eliteSize)] for _ in
    range(len(popRanked) - eliteSize):
        pick = 100*random.random()
        for i in range(len(popRanked)):
```

```
            if pick <= df.iat[i,3]:
                selectionResults.append(popRanked[i][0])
                break
        return selectionResults
    def matingPool(population, selectionResults):
        return [population[i] for i in selectionResults]
    def breed(parent1, parent2):
        geneA, geneB = sorted([int(random.random() * len(parent1)) for _ in
range(2)])
        return parent1[geneA:geneB] + [gene for gene in parent2 if gene not in
parent1[geneA:geneB]] def
    breedPopulation(matingpool, eliteSize):
        pool, children = random.sample(matingpool, len(matingpool)),
matingpool[:eliteSize]
        children += [breed(pool[i], pool[len(matingpool)-i-1]) for i in
range(len(matingpool) - eliteSize)] return children
    def mutate(individual, mutationRate):
        for i in range(len(individual)):
            if random.random() < mutationRate: swapWith =
                int(random.random() * len(individual))
                individual[i], individual[swapWith] = individual[swapWith],
individual[i] return
        individual
    def mutatePopulation(population, mutationRate): return
        [mutate(ind, mutationRate) for ind in population]
    def nextGeneration(currentGen, eliteSize, mutationRate):
        ranked = rankRoutes(currentGen) selectionResults =
        selection(ranked, eliteSize) matingpool =
        matingPool(currentGen, selectionResults) children =
        breedPopulation(matingpool, eliteSize) return
        mutatePopulation(children, mutationRate)
    def geneticAlgorithm(population, popSize, eliteSize, mutationRate, generations):
        pop = initialPopulation(popSize, population) print("Initial distance:", 1 /
        rankRoutes(pop)[0][1]) for _ in range(generations): pop =
        nextGeneration(pop, eliteSize, mutationRate)
        print("Final distance:", 1 / rankRoutes(pop)[0][1])
        return pop[rankRoutes(pop)[0][0]]
    def geneticAlgorithmPlot(population, popSize, eliteSize, mutationRate,
generations):
```

Practical

```
pop = initialPopulation(popSize, population) progress
= [1 / rankRoutes(pop)[0][1]] for _ in
range(generations): pop = nextGeneration(pop,
eliteSize, mutationRate)
```

```
        progress.append(1 / rankRoutes(pop)[0][1])
    plt.plot(progress)
    plt.ylabel('Distance')
    plt.xlabel('Generation')
    plt.show()
if __name__ == "__main__": cityList = [City(x=int(random.random() * 200),
    y=int(random.random() *
200)) for _ in range(25)] geneticAlgorithmPlot(population=cityList, popSize=100,
    eliteSize=20,
mutationRate=0.01, generations=500)
```

Output:-

INDEX

Sr. No.	Title	Page No.	Sign
1	a. Use the following tools to perform footprinting and reconnaissance i. Recon-ng (Using Kali Linux) ii. FOCA Tool iii. Windows Command Line Utilities • Ping • Tracert using Ping • Tracert • NSLookup iv. Website Copier Tool – HTTrack v. Metasploit (for information gathering) vi. Whois Lookup Tools for Mobile – DNS Tools, Whois, Ultra Tools Mobile vii. Smart Whois viii. eMailTracker Pro ix. Tools for Mobile – Network Scanner, Fing – Network Tool, Network Discovery Tool, Port Droid Tool b. Scan the network using the following tools: i. Hping2 / Hping3 ii. Advanced IP Scanner iii. Angry IP Scanner iv. Masscan v. NEET vi. CurrPorts vii. Colasoft Packet Builder viii. The Dude	4	
`2	a. Use Proxy Workbench to see the data passing through it and save the data to file. b. Perform Network Discovery using the following tools: i. Solar Wind Network Topology Mapper ii. OpManager iii. Network View	49	

	iv. LANState Pro c. Use the following censorship circumvention tools: i. Alkasir ii. Tails OS d. Use Scanning Tools for Mobile – Network Scanner, Fing – Network Tool, Network Discovery Tool, Port Droid Tool		
3	a. Perform Enumeration using the following tools: i. Nmap ii. NetBIOS Enumeration Tool iii. SuperScan Software iv. Hyena v. SoftPerfect Network Scanner Tool vi. OpUtils vii. SolarWinds Engineer's Toolset viii. Wireshark b. Perform the vulnerability analysis using the following tools: i. Nessus ii. OpenVas	55	
4	a. Perform mobile network scanning using NESSUS b. Perform the System Hacking using the following tools: i. Winrtgen ii. PWDump iii. Ophcrack iv. Flexispy v. NTFS Stream Manipulation vi. ADS Spy vii. Snow viii. Quickstego ix. Clearing Audit Policies x. Clearing Logs	62	

5	a. Use wireshark to sniff the network. b. Use SMAC for MAC Spoofing. c. Use Caspa Network Analyser. d. Use Omnipeek Network Analyzer.	81	
6	a. Use Social Engineering Toolkit on Kali Linux to perform Social Engineering using Kali Linux. b. Perform the DDOS attack using the following tools: i. HOIC ii. LOIC iii. HULK iv. Metasploit c. Using Burp Suite to inspect and modify traffic between the browser and target application.	93	
7	a. Perform Web App Scanning using OWASP Zed Proxy. b. Use droidsheep on mobile for session hijacking c. Demonstrate the use of the following firewalls: i. Zonealarm and analyse using Firewall Analyzer. ii. Comodo Firewall d. Use HoneyBOT to capture malicious network traffic. e. Use the following tools to protect attacks on the web servers: i. ID Server ii. Microsoft Baseline Security Analyzer iii. Syhunt Hybrid	105	
8	a. Protect the Web Application using dotDefender. b. Demonstrate the following tools to perform SQL Injection: i. Tyrant SQL ii. Havij iii. BBQSQL	118	
9	Use Aircrack-ng suite for wireless hacking and countermeasures.	122	

10	Use the following tools for cryptography i. HashCalc ii. Advanced Encryption Package iii. TrueCrypt iv. CrypTool	127	

Practical No. 1

A. Tools to perform footprinting and reconnaissance

Footprinting and reconnaissance are used to collect basic information about the target systems in order to exploit them. The target information is IP location information, routing information, business information, address, phone number and DNS records. **i. Recon-ng (Using Kali Linux)**

Recong0-ng is a full feature Web Reconnaissance framework used for information gathering purpose as well as network detection. This tool is written in python, having independent modules, database interaction and other features. You can download the software from www.bitbucket.org. This Open Source Web Reconnaissance tool requires kali Linux Operating system.

1- Run the Application Recon-ng or open the terminal of Kali-Linux and type recon-ng and hit enter.

2- Enter the command "***show modules***" to show all independent modules available.

3- You can search for any entity within a module. For example, in above figure, the command "***Search Netcraft***" is used.

4- To use the Netcraft module, use the command syntax "***use recon/domain-hosts/Netcraft***" and hit enter.

5- Set the source by the command "***set source*** *[domain]*." Press enter to continue. Type ***Run*** to execute and press enter.

ii. FOCA Tool

FOCA stands for Fingerprinting Organizations with Collected Archives. FOCA tool finds Metadata, and other hidden information within a document may locate on web pages. Scanned searches can be downloaded and Analyzed. FOCA is a powerful tool which can support various types of documents including Open Office, Microsoft Office, Adobe InDesign, PDF, SVG, and others. Search uses three search engines, Google, Bing, and DuckDuckGo.

1- Download the software ***FOCA*** from https://www.elevenpaths.com. Now,Go to ***Project > New Project.***

2- Now, Enter the Project Name, Domain Website, Alternate Website (if required), Directory to save the results, Project Date. Click Create to proceed.

3- Select the Search Engines, Extensions, and other parameters as required. Click on Search All Button.

4 -Once Search completes, the search box shows multiple files. You can select the file, download it, Extract Metadata, and gather other information like username, File creation date, and Modification.

iii. Windows Command Line Utilities

Consider a network where you have access to a Windows PC connected to the Internet. Using Windows-based tools, let's gather some information about the target. You can assume any target domain or IP address, in our case, we are using **example.com** as a target.

Topology Diagram:

• Ping

1- Open Windows Command Line (cmd) from Windows PC

2 -Enter the command " **Ping example.com** " to ping.

From the output, you can observe and extract the following information:

- Example.com is live
- IP address of example.com.
- Round Trip Time
- TTL value
- Packet loss statistics

3- Now, Enter the command " Ping example.com –f –l 1500 " to check the value of fragmentation.

```
Command Prompt

C:\Users\IPSpecialist>ping example.com -f -l 1500

Pinging example.com [93.184.216.34] with 1500 bytes of data:
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.

Ping statistics for 93.184.216.34:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\Users\IPSpecialist>
```

The output shows " **Packet needs to be fragmented but DF set** " which means 150o bits will require being fragmented. Let's try again with smaller value:

```
Command Prompt

C:\Users\IPSpecialist>ping example.com -f -l 1400

Pinging example.com [93.184.216.34] with 1400 bytes of data:
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.

Ping statistics for 93.184.216.34:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\Users\IPSpecialist>
```

Output again shows " **Packet needs to be fragmented but DF set** " which means 140o bits will require being fragmented. Let's try again with smaller value:

```
Command Prompt

C:\Users\IPSpecialist>ping example.com -f -l 1300

Pinging example.com [93.184.216.34] with 1300 bytes of data:
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.
Packet needs to be fragmented but DF set.

Ping statistics for 93.184.216.34:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\Users\IPSpecialist>
```

Output again shows " **Packet needs to be fragmented but DF set** " which means 130o bits will require being fragmented. Let's try again with smaller value:

```
Command Prompt

Pinging example.com [93.184.216.34] with 1200 bytes of data:
Reply from 93.184.216.34: bytes=1200 time=215ms TTL=52
Reply from 93.184.216.34: bytes=1200 time=213ms TTL=52
Reply from 93.184.216.34: bytes=1200 time=214ms TTL=52
Reply from 93.184.216.34: bytes=1200 time=216ms TTL=52

Ping statistics for 93.184.216.34:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 213ms, Maximum = 216ms, Average = 214ms

C:\Users\IPSpecialist>_
```

The output shows the reply now, which means 1200 bits will not require being fragmented. You can try again to get the more appropriate fragment value. **• Tracert using Ping**

Enter the command " **Tracert example.com** " to trace the target.

```
Command Prompt

C:\Users\IPSpecialist>tracert example.com

Tracing route to example.com [93.184.216.34]
over a maximum of 30 hops:

  1      1 ms      1 ms      2 ms  192.168.0.1
  2      *         *         *     Request timed out.
  3      3 ms      2 ms      2 ms  110.37.216.157
  4      9 ms      3 ms      2 ms  58.27.182.149
  5      3 ms      2 ms      2 ms  58.27.209.54
  6      3 ms      5 ms      4 ms  58.27.183.230
  7     28 ms      8 ms      9 ms  tw31-static109.tw1.com [117.20.31.109]
  8      5 ms      4 ms      4 ms  110.93.253.117
  9    102 ms    103 ms    104 ms  be4932.ccr22.mrs01.atlas.cogentco.com [149.14.125.
89]
 10    191 ms    127 ms    118 ms  be3093.ccr42.par01.atlas.cogentco.com [130.117.50.
165]
 11    114 ms    140 ms    123 ms  prs-b2-link.telia.net [213.248.86.169]
 12    278 ms    201 ms    232 ms  prs-bb3-link.telia.net [62.115.122.4]
 13    204 ms    202 ms    202 ms  ash-bb3-link.telia.net [80.91.251.243]
 14    202 ms    202 ms    202 ms  ash-b1-link.telia.net [80.91.248.157]
 15    273 ms    221 ms    240 ms  verizon-ic-315152-ash-b1.c.telia.net [213.248.83.1
19]
 16    218 ms    215 ms    213 ms  152.195.65.133
 17    211 ms    211 ms    322 ms  93.184.216.34

Trace complete.

C:\Users\IPSpecialist>_
```

• Tracert

Tracert options are available in all operating system as a command line feature. Visual traceroute, graphical and other GUI based traceroute applications are also available. Traceroute or Tracert command results in the path information from source to destination in the hop by hop manner. The result includes all hops in between source to destination. The result also includes latency between these hops.

Consider an example, in which an attacker is trying to get network information by using tracert. After observing the following result, you can identify the network map.

10.0.0.1 is the first hop, which means it is the gateway. Tracert result of 200.100.50.3 shows, 200.100.50.3 is another interface of first hop device whereas connected IP includes 200.100.50.2 & 200.100.50.1.

192.168.0.254 is next to last hop 10.0.0.1. It can either connected to 200.100.50.1 or 200.100.50.2. To verify, trace next route.

192.168.0.254 is another interface of the network device, i.e. 200.100.50.1 connected next to 10.0.0.1. 192.168.0.1, 192.168.0.2 & 192.168.0.3 are connected directly to 192.168.0.254.

192.168.10.254 is another interface of the network device i.e. 200.100.50.2 connected next to 10.0.0.1. 192.168.10.1, 192.168.10.2 & 192.168.10.3 are connected directly to 192.168.10.254.

Traceroute Tools

Traceroute tools are listed below: -

-

Traceroute Tools	Website
Path Analyzer Pro	www.pathanalyzer.com
Visual Route	www.visualroute.com
Troute	www.mcafee.com
3D Traceroute	www.d3tr.de

The following figure shows graphical view and other trace information usingVisual Route tool.

• DNS Zone Transfer Enumeration Using NSLookup

Nslookup (stands for "Name Server Lookup") is a useful command for getting information from the DNS server. It is a network administration tool for querying the Domain Name System (DNS) to obtain domain name or IP address mapping or any other specific DNS record. It is also used to troubleshoot DNS-related problems.

In the enumeration process through DNS Zone transfer, attacker find the target's TCP port 53, as TCP port 53 is used by DNS and Zone transfer uses this port by default. Using port scanning techniques, you can find if the port is open.

DNS Zone transfer is the process that is performed by DNS. In the process of Zone transfer, DNS passes a copy containing database records to another DNS server. DNS Zone transfer process provides support for resolving queries, as more than one DNS server can respond to the queries.

Consider a scenario in which both primary and secondary DNS Servers are responding to the queries. Secondary DNS server gets the DNS records copy to update the information in its database.

1. Go to Windows command line (CMD) and enter Nslookup and press Enter.

2. Command prompt will proceed to " > " symbol.
3. Enter " server <DNS Server Name> " or " server <DNS Server Address> ".
4. Enter set type=any and press Enter. It will retrieve all records from a DNS server.
5. Enter ls -d <Domain> this will display the information from the target domain (if allowed).

6. If not allowed, it will show the request failed.

7. Linux support dig command, At a command prompt enter dig <domain.com> axfr. **iv.**

Website Copier tool (HTTrack)

1- Download and Install the WinHTTrack Website Copier Tool from the website **http://www.httrack.com.** You can check the compatibility of HTTrack Website copier tool on different platforms such as Windows, Linux, and Android from the website.

2- HTTrack Website Copier tool installation.

3- Click Next

4- Enter a Project name, as in our case, **Testing_Project.**

5- Click on **Set Options** button.

6- Go to **Scan Rules** Tab and Select options as required.

7- Enter the Web Address in the field and Click Next.

8- Click Next.

9- Click **Browse Mirrored Website**.

10- Select your favorite web browser.

Observed the above output. Example.com website is copied into a local directory and browsed from there. Now you can explore the website in an offline environment for the structure of the website and other parameters.

To make sure, compare the website to the original example.com website. Open a new tab and go to URL example.com.

v. Metasploit (for information gathering)

In this lab, we are using Metasploit Framework, default application in Kali Linux for gathering more information about the host in a network. A Metasploit Framework is a powerful tool, popularly used for scanning & gathering information in the hacking environment. Metasploit Pro enables you to automate the process of discovery and exploitation and provides you with the necessary tools to perform the manual testing phase of a penetration test. You can use Metasploit Pro to scan for open ports and services, exploit vulnerabilities, pivot further into a network, collect evidence, and create a report of the test results.

Topology Information: In this lab, we are running Metasploit Framework on a private network 10.10.50.0/24 where different hosts are live including Windows 7, Kali Linux, Windows Server 2016 and others.

1- Open Kali Linux and Run Metasploit Framework.
2- Metasploit Framework initialization as shown below in the figure.

msf > **db_status**
[*] postgresql connected to msf

// If your database is not connected, it means your database is not initiated. You will need to exit msfconsole & restart the postgresql service.

// Performing nmap Scan for ping sweep on the subnet 10.10.50.0/24

msf > **nmap -Pn -sS -A -oX Test 10.10.50.0/24**
[*] exec: nmap -Pn -sS -A -oX Test 10.10.50.0/24
Starting Nmap 7.60 (https://nmap.org) at 2018-04-26 01:49 EDT
Stats: 0:04:31 elapsed; 247 hosts completed (8 up), 8 undergoing Script Scan
NSE Timing: About 99.77% done; ETC: 01:53 (0:00:00 remaining)
Stats: 0:05:04 elapsed; 247 hosts completed (8 up), 8 undergoing Script Scan
NSE Timing: About 99.79% done; ETC: 01:54 (0:00:00 remaining)
Stats: 0:06:21 elapsed; 247 hosts completed (8 up), 8 undergoing Script Scan
NSE Timing: About 99.93% done; ETC: 01:55 (0:00:00 remaining)
Nmap scan report for 10.10.50.1
Host is up (0.0012s latency).
Not shown: 996 closed ports
PORT STATE SERVICE VERSION

```
22/tcp   open  ssh       Cisco SSH 1.25 (protocol 1.5)
| ssh-hostkey:
|_  512 ca:9c:c7:d2:d4:b0:78:82:3e:34:8f:cf:00:9d:75:db (RSA1)
|_sshv1: Server supports SSHv1
23/tcp   open  telnet    Cisco router telnetd
5060/tcp open  sip-proxy  Cisco SIP Gateway (IOS 15.2.4.M4)
|_sip-methods: INVITE, OPTIONS, BYE, CANCEL, ACK, PRACK, UPDATE, REFER,
SUBSCRIBE, NOTIFY, INFO, REGISTER
5061/tcp open  tcpwrapped
MAC Address: C0:67:AF:C7:D9:80 (Cisco Systems)
OS details: Cisco 836, 890, 1751, 1841, 2800, or 2900 router (IOS 12.4 - 15.1), Cisco Aironet 1141N
(IOS 12.4) or 3602I (IOS 15.3) WAP, Cisco Aironet 2600-series WAP (IOS 15.2(2))
Network Distance: 1 hop
Service Info: OS: IOS; Device: router; CPE: cpe:/o:cisco:ios

TRACEROUTE
HOP RTT     ADDRESS
1   1.15 ms 10.10.50.1

Nmap scan report for 10.10.50.10
Host is up (0.00030s latency).
Not shown: 990 filtered ports
PORT     STATE  SERVICE        VERSION
22/tcp   open   ssh            OpenSSH 5.6 (protocol 2.0)
| ssh-hostkey:
|   1024 e3:93:64:12:9c:c0:70:72:35:e1:ac:61:af:cc:49:ec (DSA)
|_  2048 2a:0b:42:38:f4:ca:d6:07:95:aa:87:ed:52:de:d1:14 (RSA)
80/tcp   open   http           VMware ESXi Server httpd
|_http-title: Did not follow redirect to https://10.10.50.10/
427/tcp  open   svrloc?
443/tcp  open   ssl/http       VMware ESXi Server httpd
|_http-title: " + ID_EESX_Welcome + "
| ssl-cert: Subject: commonName=localhost.localdomain/organizationName=VMware,
Inc/stateOrProvinceName=California/countryName=US
| Subject Alternative Name: DNS:localhost.localdomain
| Not valid before: 2014-01-15T03:42:31
|_Not valid after:  2025-07-16T03:42:31
|_ssl-date: 2018-04-25T19:58:24+00:00; -9h53m36s from scanner time.
| vmware-version:
|   Server version: VMware ESXi 5.1.0
|   Build: 1065491
|   Locale version: INTL 000
|   OS type: vmnix-x86
|_  Product Line ID: embeddedEsx
902/tcp  open   ssl/vmware-auth VMware Authentication Daemon 1.10 (Uses VNC, SOAP)
5988/tcp closed wbem-http
5989/tcp open   ssl/wbem       SBLIM Small Footprint CIM Broker
| ssl-cert: Subject: commonName=localhost.localdomain/organizationName=VMware,
Inc/stateOrProvinceName=California/countryName=US
| Subject Alternative Name: DNS:localhost.localdomain
```

```
| Not valid before: 2014-01-15T03:42:31
|_Not valid after:  2025-07-16T03:42:31
|_ssl-date: 2018-04-25T19:58:23+00:00; -9h53m36s from scanner time.
8000/tcp open   http-alt?
8100/tcp open   tcpwrapped
8300/tcp closed tmi
MAC Address: F8:72:EA:A4:A1:CC (Cisco Systems)
Aggressive OS guesses: VMware ESXi 5.0 - 5.5 (96%), VMware ESXi 5.5 (96%), VMware ESXi
4.1 (95%), VMware ESXi 6.0.0 (93%), FreeBSD 7.0-RELEASE-p1 - 10.0-CURRENT (93%),
VMware ESXi 4.1.0 (93%), VMware ESX Server 4.0.1 (91%), FreeBSD 5.2.1-RELEASE (91%),
FreeBSD 8.0-BETA2 - 10.1-RELEASE (90%), FreeBSD 5.3 - 5.5 (90%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
Service Info: Host: localhost.localdomain; CPE: cpe:/o:vmware:esxi, cpe:/o:vmware:ESXi:5.1.0

Host script results:
|_clock-skew: mean: -9h53m36s, deviation: 0s, median: -9h53m36s

TRACEROUTE
HOP RTT     ADDRESS
1   0.30 ms 10.10.50.10

Nmap scan report for 10.10.50.11
Host is up (0.00058s latency).
Not shown: 990 filtered ports
PORT     STATE  SERVICE        VERSION
22/tcp   open   ssh            OpenSSH 5.6 (protocol 2.0)
| ssh-hostkey:
|   1024 6f:d3:3d:cb:54:0b:83:3e:bd:25:1c:da:67:b6:92:fb (DSA)
|_  2048 f9:bc:20:c5:6e:db:6a:86:ea:f5:24:06:57:c6:d9:6f (RSA)
80/tcp   open   http           VMware ESXi Server httpd
|_http-title: Did not follow redirect to https://10.10.50.11/
427/tcp  open   svrloc?
443/tcp  open   ssl/http       VMware ESXi Server httpd
|_http-title: " + ID_EESX_Welcome + "
| ssl-cert: Subject: commonName=localhost.localdomain/organizationName=VMware,
Inc/stateOrProvinceName=California/countryName=US
| Subject Alternative Name: DNS:localhost.localdomain
| Not valid before: 2014-01-18T05:33:03
|_Not valid after:  2025-07-19T05:33:03
|_ssl-date: 2018-04-25T19:50:12+00:00; -10h01m33s from scanner time.
| vmware-version:
|   Server version: VMware ESXi 5.1.0
|   Build: 1065491
|   Locale version: INTL 000
|   OS type: vmnix-x86
|_  Product Line ID: embeddedEsx
902/tcp  open   ssl/vmware-auth VMware Authentication Daemon 1.10 (Uses VNC, SOAP)
5988/tcp closed wbem-http
5989/tcp open   ssl/wbem       SBLIM Small Footprint CIM Broker
```

```
| ssl-cert: Subject: commonName=localhost.localdomain/organizationName=VMware,
Inc/stateOrProvinceName=California/countryName=US
| Subject Alternative Name: DNS:localhost.localdomain
| Not valid before: 2014-01-18T05:33:03
|_Not valid after:  2025-07-19T05:33:03
|_ssl-date: 2018-04-25T19:50:25+00:00; -10h01m35s from scanner time.
8000/tcp open   http-alt?
8100/tcp open   tcpwrapped
8300/tcp closed tmi
MAC Address: F8:72:EA:A4:A1:2C (Cisco Systems)
Device type: specialized
Running: VMware ESXi 5.X
OS CPE: cpe:/o:vmware:esxi:5
OS details: VMware ESXi 5.0 - 5.5
Network Distance: 1 hop
Service Info: Host: localhost.localdomain; CPE: cpe:/o:vmware:esxi, cpe:/o:vmware:ESXi:5.1.0

Host script results:
|_clock-skew: mean: -10h01m34s, deviation: 1s, median: -10h01m35s

TRACEROUTE
HOP RTT     ADDRESS
1   0.58 ms 10.10.50.11

Nmap scan report for vc.ooredoocloud.qa (10.10.50.20)
Host is up (0.00065s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 8d:b4:b0:01:63:84:eb:c7:bf:cf:f7:b0:c3:12:0e:13 (RSA)
|   256 02:31:3e:d3:75:97:f2:10:88:30:6a:c1:ca:a4:82:bf (ECDSA)
|_  256 c5:21:3a:a7:81:f5:a6:00:ee:5e:76:94:88:68:03:1d (EdDSA)
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
MAC Address: 00:0C:29:72:4A:C1 (VMware)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.8
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.65 ms 10.10.50.20

Nmap scan report for 10.10.50.100
Host is up (0.00078s latency).
```

```
Not shown: 983 closed ports
PORT     STATE SERVICE        VERSION
135/tcp   open  msrpc          Microsoft Windows RPC
139/tcp   open  netbios-ssn     Microsoft Windows netbios-ssn
443/tcp   open  ssl/http       VMware VirtualCenter Web service
|_http-title: Site doesn't have a title (text; charset=plain).
| ssl-cert: Subject: commonName=VMware/countryName=US
| Not valid before: 2017-12-19T17:36:01
|_Not valid after:  2018-12-19T17:36:01
|_ssl-date: TLS randomness does not represent time
| vmware-version:
|   Server version: VMware Workstation 12.5.6
|   Build: 5528349
|   Locale version: INTL
|   OS type: win32-x86
|_  Product Line ID: ws
445/tcp   open  microsoft-ds    Windows 7 Professional 7601 Service Pack 1 microsoft-ds
(workgroup: WORKGROUP)
554/tcp   open  rtsp?
902/tcp   open  ssl/vmware-auth VMware Authentication Daemon 1.10 (Uses VNC, SOAP)
912/tcp   open  vmware-auth     VMware Authentication Daemon 1.0 (Uses VNC, SOAP)
1025/tcp  open  msrpc          Microsoft Windows RPC
1026/tcp  open  msrpc          Microsoft Windows RPC
1027/tcp  open  msrpc          Microsoft Windows RPC
1028/tcp  open  msrpc          Microsoft Windows RPC
1030/tcp  open  msrpc          Microsoft Windows RPC
1031/tcp  open  msrpc          Microsoft Windows RPC
2869/tcp  open  http           Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
3389/tcp  open  ms-wbt-server   Microsoft Terminal Service
| ssl-cert: Subject: commonName=Win7-PC
| Not valid before: 2017-12-12T19:55:25
|_Not valid after:  2018-06-13T19:55:25
|_ssl-date: 2018-04-26T05:47:49+00:00; -3m54s from scanner time.
5357/tcp  open  http           Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Service Unavailable
10243/tcp open  http           Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Not Found
MAC Address: 00:0C:29:95:04:33 (VMware)
Device type: general purpose
Running: Microsoft Windows 7|2008|8.1
OS CPE: cpe:/o:microsoft:windows_7::- cpe:/o:microsoft:windows_7::sp1
cpe:/o:microsoft:windows_server_2008::sp1 cpe:/o:microsoft:windows_server_2008:r2
cpe:/o:microsoft:windows_8 cpe:/o:microsoft:windows_8.1
OS details: Microsoft Windows 7 SP0 - SP1, Windows Server 2008 SP1, Windows Server 2008 R2,
Windows 8, or Windows 8.1 Update 1
Network Distance: 1 hop
Service Info: Host: WIN7-PC; OS: Windows; CPE: cpe:/o:microsoft:windows,
cpe:/o:vmware:Workstation:12.5.6
```

```
Host script results:
|_clock-skew: mean: -3m54s, deviation: 0s, median: -3m54s
|_nbstat: NetBIOS name: WIN7-PC, NetBIOS user: <unknown>, NetBIOS MAC: 00:0c:29:95:04:33
(VMware)
| smb-os-discovery:
|   OS: Windows 7 Professional 7601 Service Pack 1 (Windows 7 Professional 6.1)
|   OS CPE: cpe:/o:microsoft:windows_7::sp1:professional
|   Computer name: Win7-PC
|   NetBIOS computer name: WIN7-PC\x00
|   Workgroup: WORKGROUP\x00
|_  System time: 2018-04-26T10:47:56+05:00
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
| smb2-security-mode:
|   2.02:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2018-04-26 01:48:04
|_  start_date: 2018-03-27 07:26:43

TRACEROUTE
HOP RTT     ADDRESS
1   0.78 ms 10.10.50.100

Nmap scan report for 10.10.50.202
Host is up (0.00096s latency).
Not shown: 986 closed ports
PORT      STATE SERVICE       VERSION
135/tcp   open  msrpc         Microsoft Windows RPC
139/tcp   open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds  Windows 7 Professional 7601 Service Pack 1 microsoft-ds (workgroup:
WORKGROUP)
554/tcp   open  rtsp?
2869/tcp  open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
3389/tcp  open  ms-wbt-server Microsoft Terminal Service
| ssl-cert: Subject: commonName=Win7-1-PC
| Not valid before: 2018-03-05T06:10:47
|_Not valid after:  2018-09-04T06:10:47
|_ssl-date: 2018-04-26T05:51:38+00:00; -28s from scanner time.
5357/tcp  open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Service Unavailable
10243/tcp open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Not Found
49152/tcp open  msrpc         Microsoft Windows RPC
```

```
49153/tcp open  msrpc       Microsoft Windows RPC
49154/tcp open  msrpc       Microsoft Windows RPC
49156/tcp open  msrpc       Microsoft Windows RPC
49157/tcp open  msrpc       Microsoft Windows RPC
49160/tcp open  msrpc       Microsoft Windows RPC
MAC Address: 00:0C:29:20:C4:A9 (VMware)
Device type: general purpose
Running: Microsoft Windows 7|2008|8.1
OS CPE: cpe:/o:microsoft:windows_7::- cpe:/o:microsoft:windows_7::sp1
cpe:/o:microsoft:windows_server_2008::sp1 cpe:/o:microsoft:windows_server_2008:r2
cpe:/o:microsoft:windows_8 cpe:/o:microsoft:windows_8.1
OS details: Microsoft Windows 7 SP0 - SP1, Windows Server 2008 SP1, Windows Server 2008 R2,
Windows 8, or Windows 8.1 Update 1
Network Distance: 1 hop
Service Info: Host: WIN7-1-PC; OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_clock-skew: mean: -28s, deviation: 0s, median: -28s
|_nbstat: NetBIOS name: WIN7-1-PC, NetBIOS user: <unknown>, NetBIOS MAC:
00:0c:29:20:c4:a9 (VMware)
| smb-os-discovery:
|   OS: Windows 7 Professional 7601 Service Pack 1 (Windows 7 Professional 6.1)
|   OS CPE: cpe:/o:microsoft:windows_7::sp1:professional
|   Computer name: Win7-1-PC
|   NetBIOS computer name: WIN7-1-PC\x00
|   Workgroup: WORKGROUP\x00
|_  System time: 2018-04-25T22:51:33-07:00
| smb-security-mode:
|   account_used: <blank>
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
| smb2-security-mode:
|   2.02:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2018-04-26 01:51:33
|_  start_date: 2018-03-29 05:57:42

TRACEROUTE
HOP RTT     ADDRESS
1   0.96 ms 10.10.50.202

Nmap scan report for 10.10.50.210
Host is up (0.00065s latency).
Not shown: 998 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 3c:9c:fb:cb:58:35:f9:d7:d7:32:6f:ad:6a:f8:c7:9b (RSA)
```

```
|   256 70:e7:d9:a2:6a:54:92:e6:07:c9:89:58:b5:99:7d:0d (ECDSA)
|_  256 b1:be:a6:62:96:69:76:64:aa:23:bb:ad:54:cc:c0:db (EdDSA)
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
MAC Address: 00:0C:29:EA:BD:DF (VMware)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.8
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.65 ms 10.10.50.210

Nmap scan report for 10.10.50.211
Host is up (0.00037s latency).
Not shown: 999 filtered ports
PORT     STATE SERVICE       VERSION
3389/tcp open  ms-wbt-server Microsoft Terminal Services
| ssl-cert: Subject: commonName=WIN-2HMGPM3UAD7
| Not valid before: 2018-03-28T12:23:16
|_Not valid after:  2018-09-27T12:23:16
|_ssl-date: 2018-04-26T05:51:41+00:00; -5s from scanner time.
MAC Address: 00:0C:29:BA:AC:AA (VMware)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed
port
Device type: general purpose
Running (JUST GUESSING): FreeBSD 6.X (85%)
OS CPE: cpe:/o:FreeBSD:FreeBSD:6.2
Aggressive OS guesses: FreeBSD 6.2-RELEASE (85%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
Service Info: OS: Windows; CPE: cpe:/o:Microsoft:windows

Host script results:
|_clock-skew: mean: -5s, deviation: 0s, median: -5s

TRACEROUTE
HOP RTT     ADDRESS
1   0.37 ms 10.10.50.211

Nmap scan report for 10.10.50.200
Host is up (0.000042s latency).
All 1000 scanned ports on 10.10.50.200 are closed
Too many fingerprints match this host to give specific OS details
Network Distance: 0 hops
```

```
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 256 IP addresses (9 hosts up) scanned in 384.48 seconds
```

//Importing Nmap XML file

```
msf > db_import Test
[*] Importing 'Nmap XML' data
[*] Import: Parsing with 'Nokogiri v1.8.1'
[*] Importing host 10.10.50.1
[*] Importing host 10.10.50.10
[*] Importing host 10.10.50.11
[*] Importing host 10.10.50.20
[*] Importing host 10.10.50.100
[*] Importing host 10.10.50.202
[*] Importing host 10.10.50.210
[*] Importing host 10.10.50.211
[*] Importing host 10.10.50.200
[*] Successfully imported /root/Test
```

```
Hosts
=====

Address         mac             name        os_name   os_flavor os_sp purpose info comments
-------     ---           ----        -------   --------- ----- ------- ----
10.10.50.1          c0:67:af:c7:d9:80           IOS           12.X  device
10.10.50.10         f8:72:ea:a4:a1:cc           ESXi          5.X   device
10.10.50.11         f8:72:ea:a4:a1:2c           ESXi          5.X   device
10.10.50.20         00:0c:29:72:4a:c1           Linux         3.X   server
10.10.50.100        00:0c:29:95:04:33           Windows 7           client
10.10.50.200        Unknown             device
10.10.50.202        00:0c:29:20:c4:a9           Windows 7           client
10.10.50.210        00:0c:29:ea:bd:df           Linux         3.X   server
10.10.50.211        00:0c:29:ba:ac:aa           FreeBSD       6.X   device
```

//Performing Services scan

msf > **db_nmap -sS -A 10.10.50.211**

```
Applications  Places  Terminal  Thu 02:02
Terminal
File  Edit  View  Search  Terminal  Help
msf > db_nmap -sS -A 10.10.50.211
[*] Nmap: Starting Nmap 7.60 ( https://nmap.org ) at 2018-04-26 02:01 EDT
[*] Nmap: Nmap scan report for 10.10.50.211
[*] Nmap: Host is up (0.00032s latency).
[*] Nmap: Not shown: 999 filtered ports
[*] Nmap: PORT     STATE SERVICE       VERSION
[*] Nmap: 3389/tcp open  ms-wbt-server Microsoft Terminal Services
[*] Nmap: | ssl-cert: Subject: commonName=WIN-2HMGPM3UAD7
[*] Nmap: | Not valid before: 2018-03-28T12:23:16
[*] Nmap: |_Not valid after:  2018-09-27T12:23:16
[*] Nmap: |_ssl-date: 2018-04-26T06:01:58+00:00; -4s from scanner time.
[*] Nmap: MAC Address: 00:0C:29:BA:AC:AA (VMware)
[*] Nmap: Warning: OSScan results may be unreliable because we could not find at least
1 open and 1 closed port
[*] Nmap: Device type: general purpose
[*] Nmap: Running (JUST GUESSING): FreeBSD 6.X (85%)
[*] Nmap: OS CPE: cpe:/o:freebsd:freebsd:6.2
[*] Nmap: Aggressive OS guesses: FreeBSD 6.2-RELEASE (85%)
[*] Nmap: No exact OS matches for host (test conditions non-ideal).
[*] Nmap: Network Distance: 1 hop
[*] Nmap: Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
[*] Nmap: Host script results:
[*] Nmap: |_clock-skew: mean: -4s, deviation: 0s, median: -4s
[*] Nmap: TRACEROUTE
[*] Nmap: HOP RTT     ADDRESS
[*] Nmap: 1   0.31 ms 10.10.50.211
[*] Nmap: OS and Service detection performed. Please report any incorrect results at ht
```

Observe the scan result showing different services, open and closed port information of live hosts.

msf > **services**

```
msf > services

Services
========

host          port   proto  name            state   info
----          ----   -----  ----            -----   ----
10.10.50.1    22     tcp    ssh             open    Cisco SSH 1.25 protocol 1.5
10.10.50.1    23     tcp    telnet          open    Cisco router telnetd
10.10.50.1    5060   tcp    sip-proxy       open    Cisco SIP Gateway IOS 15.2.4.M4
10.10.50.1    5061   tcp    tcpwrapped      open
10.10.50.10   22     tcp    ssh             open    OpenSSH 5.6 protocol 2.0
10.10.50.10   80     tcp    http            open    VMware ESXi Server httpd
10.10.50.10   427    tcp    svrloc          open
10.10.50.10   443    tcp    ssl/http        open    VMware ESXi Server httpd
10.10.50.10   902    tcp    ssl/vmware-auth open    VMware Authentication Daemon 1.10
Uses VNC, SOAP
10.10.50.10   5988   tcp    wbem-http       closed
10.10.50.10   5989   tcp    ssl/wbem        open    SBLIM Small Footprint CIM Broker
10.10.50.10   8000   tcp    http-alt        open
10.10.50.10   8100   tcp    tcpwrapped      open
10.10.50.10   8300   tcp    tmi             closed
10.10.50.11   22     tcp    ssh             open    OpenSSH 5.6 protocol 2.0
10.10.50.11   80     tcp    http            open    VMware ESXi Server httpd
10.10.50.11   427    tcp    svrloc          open
10.10.50.11   443    tcp    ssl/http        open    VMware ESXi Server httpd
10.10.50.11   902    tcp    ssl/vmware-auth open    VMware Authentication Daemon 1.10
Uses VNC, SOAP
10.10.50.11   5988   tcp    wbem-http       closed
```

msf > **use scanner/smb/smb_version**

msf auxiliary(scanner/smb/smb_version) > **show options**

Module options (auxiliary/scanner/smb/smb_version):

Name	Current Setting	Required	Description
RHOSTS		yes	The target address range or CIDR identifier
SMBDomain.		no	The Windows domain to use for authentication
SMBPass		no	The password for the specified username
SMBUser		no	The username to authenticate as
THREADS	1	yes	The number of concurrent threads

msf auxiliary(scanner/smb/smb_version) > **set RHOSTS 10.10.50.100-211**
RHOSTS => 10.10.50.100-211

msf auxiliary(scanner/smb/smb_version) > **set THREADS 100**
THREADS => 100

msf auxiliary(scanner/smb/smb_version) > **show options**

Module options (auxiliary/scanner/smb/smb_version):

Name	Current Setting	Required	Description
RHOSTS	10.10.50.100-211	yes	The target address range or CIDR identifier
SMBDomain.		no	The Windows domain to use for authentication
SMBPass		no	The password for the specified username
SMBUser		no	The username to authenticate as
THREADS	100	yes	The number of concurrent threads

msf auxiliary(scanner/smb/smb_version) > **run**

```
Terminal
File  Edit  View  Search  Terminal  Help
msf auxiliary(scanner/smb/smb_version) > run

[+] 10.10.50.100:445      - Host is running Windows 7 Professional SP1 (build:7601)
(name:WIN7-PC) (workgroup:WORKGROUP )
[+] 10.10.50.202:445      - Host is running Windows 7 Professional SP1 (build:7601)
(name:WIN7-1-PC) (workgroup:WORKGROUP )
[*] Scanned  24 of 112 hosts (21% complete)
[*] Scanned  28 of 112 hosts (25% complete)
[*] Scanned  76 of 112 hosts (67% complete)
[*] Scanned  79 of 112 hosts (70% complete)
[*] Scanned  81 of 112 hosts (72% complete)
[*] Scanned 103 of 112 hosts (91% complete)
[*] Scanned 110 of 112 hosts (98% complete)
[*] Scanned 111 of 112 hosts (99% complete)
[*] Scanned 112 of 112 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(scanner/smb/smb_version) >
```

msf auxiliary(scanner/smb/smb_version) > **hosts**

```
Terminal
File  Edit  View  Search  Terminal  Help
msf auxiliary(scanner/smb/smb_version) > hosts

Hosts
=====

address       mac               name                os_name    os_flavor     os_sp  purpose  info  comments
-------       ---               ----                -------    ---------     -----  -------  ----  --------
10.10.50.1    c0:67:af:c7:d9:80                     IOS                      12.X   device
10.10.50.10   f8:72:ea:a4:a1:cc                     ESXi                     5.X    device
10.10.50.11   f8:72:ea:a4:a1:2c                     ESXi                     5.X    device
10.10.50.20   00:0c:29:72:4a:c1  vc.ooredoocloud.qa Linux                    3.X    server
10.10.50.100  00:0c:29:95:04:33  WIN7-PC             Windows 7  Professional SP1    client
10.10.50.200                                        Unknown                         device
10.10.50.202  00:0c:29:20:c4:a9  WIN7-1-PC           Windows 7  Professional SP1    client
10.10.50.210  00:0c:29:ea:bd:df                     Linux                    3.X    server
10.10.50.211  00:0C:29:BA:AC:AA                     FreeBSD                  6.X    device

msf auxiliary(scanner/smb/smb_version) >
```

Observe the OS_Flavor field. SMB scanning scans for Operating System Flavor for the RHOST range configured.

vi. Whois Lookup Tools for Mobile – DNS Tools, Whois, Ultra Tools Mobile

"WHOIS" helps to gain information regarding domain name, ownership information. IP Address,

Netblock data, Domain Name Servers and other information's. Regional Internet Registries (RIR) maintain WHOIS database. WHOIS lookup helps to find out who is behind the target domain name.

1. Go to the URL **https://www.whois.com/**

2. A search of Target Domain

WHOIS Lookup Result Analysis

Lookup Result shows complete domain profile, including

- Registrant information
- Registrant Organization
- Registrant Country
- Domain name server information
- IP Address
- IP location
- ASN
- Domain Status
- WHOIS history
- IP history,
- Registrar history,
- Hosting history

It also includes other information such as Email and postal address of registrar & admin along with contact details. You can go to ***https://whois.domaintools.com*** can enter the targeted URL for whois lookup information

vii. Smart Whois

You can download software "***SmartWhois***" from ***www.tamos.com*** for Whois lookup as shown in the figure below: -

viii. eMailTracker Pro

eMailTrackerPro is **a Windows based email tracker that can be used to monitor employees, senders and recipients**. This powerful tool can be used in conjunction with other programs such as Windows Nuke (also known as Spamwasher) to quickly identify where a computer has been and how it has been used.

Click on Trace Headers/Trace email address and enter the Message Header and click Okay. The Status of the Trace will be shown inside Trace Reports

ix. Tools for Mobile – Network Scanner, Fing – Network Tool, Network Discovery Tool, Port Droid Tool

Scanning Tool for Mobile

Fing- Network Tool

Network Discovery Tool

b. Scan the network using the following tools:

i. Hping2 / Hping3

Hping is a command-line TCP/IP packet assembler and analyzer tool that is used to send customized TCP/IP packets and display the target reply as ping command display the ICMP Echo Reply packet from targeted host. Hping can also handle fragmentation, arbitrary packets body, and size and file transfer. It supports TCP, UDP, ICMP and RAW-IP protocols. Using Hping, the following parameters can be performed: -

- Test firewall rules.
- Advanced port scanning.
- Testing net performance.
- Path MTU discovery.
- Transferring files between even fascist firewall rules.
- Traceroute-like under different protocols.
- Remote OS fingerprinting & others

Using Hping commands on Kali Linux, we are pinging a Window 7 host with different customized packets in this lab.

- To create an ACK packet:

root@kali:~# **hping3 –A 192.168.0.1**

- To create SYN scan against different ports: root@kali:~# **hping3 -8 1-600 –S 10.10.50.202**

- To create a packet with FIN, URG, and PSH flags sets root@kali:~# **hping3 –F –P -U 10.10.50.202**

ii. Advanced IP Scanner

Advanced IP Scanner is **a fast and powerful network scanner with a user-friendly interface**. In seconds, Advanced IP Scanner can locate all computers on your wired or wireless local network and scan their ports. The program provides easy access to various network resources such as HTTP, HTTPS, FTP, and shared folders.

iii. Angry IP Scanner

Angry IP Scanner (or simply ipscan) is an open-source and cross-platform network scanner designed to be fast and simple to use. It scans IP addresses and ports as well as has many other features.

It is widely used by network administrators and just curious users around the world, including large and small enterprises, banks, and government agencies.

It runs on Linux, Windows, and Mac OS X, possibly supporting other platforms as well.

iv. Masscan

MASSCAN is **TCP port scanner which transmits SYN packets asynchronously and produces results similar to nmap, the most famous port scanner**. Internally, it operates more like scanrand, unicornscan, and ZMap, using asynchronous transmission. It's a flexible utility that allows arbitrary address and port ranges.

Scan for a selection of ports (-p22,80,445) across a given subnet (192.168.1.0/24):
root@kali:~# masscan -p22,80,445 192.168.1.0/24

```
Starting masscan 1.0.3 (http://bit.ly/14GZzcT) at 2014-05-13 21:35:12 GMT
-- forced options: -sS -Pn -n --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 256 hosts [3 ports/host]
Discovered open port 22/tcp on 192.168.1.217
Discovered open port 445/tcp on 192.168.1.220 Discovered open port 80/tcp on 192.168.1.230
```

v. NEET

Neet is a flexible, multi-threaded tool for network penetration testing. It runs on Linux and coordinates the use of numerous other open-source network tools, with the aim of gathering as much network information as possible in clear, easy-to-use formats. The core scanning engine finds and

identifies network services, the modules test or enumerate those services, and the Neet Shell provides an integrated environment for processing the results and exploiting known vulnerabilities. As such, it sits somewhere between manually running your own port scans and subsequent tests, and running a fully automated vulnerability assessment (VA) tool. It has many options which allow the user to tune the test parameters for network scanning in the most efficient and practical way.

vi. CurrPorts

Case Study: Using the Previous lab, we are going to re-execute HTTP Remote Access Trojan (RAT) on Windows 12 machine (10.10.50.211) and observed the TCP/IP connections to detect and kill the connection.

Topology:

Configuration:

1. Run the application **Currports** on Windows Server 2016 and observe theprocesses.

2.Run the HTTP Trojan created in the previous lab

The new process is added to the list.

You can observe the process name, Protocol, Local and remote port and IP address information.

3. For more detail, right click on httpserver.exe and go to properties

Properties are showing more details about tcp connection.

4. Go to Windows 7 machine and initiate the connection as mentioned in the previous lab using a web browser.

Connection successfully established.

5. Back to Windows Server 2016, Kill the connection.

6. To verify, retry to establish the connection from windows 7.

vii. Colasoft Packet Builder

Colasoft Packet Builder software enables to create the customized network packets. These Customized Network packets can penetrate the network for attacks. Customization can also use to create fragmented packets. You can download the software from www.colasoft.com.

Colasoft packet builder offers Import and Export options for a set of packets. You can also add a new packet by clicking **Add**/button. Select the Packet type from the drop-down option. Available options are: -

- ARP Packet
- IP Packet
- TCP Packet
- UDP Packet

After Selecting the Packet Type, now you can customize the packet, Select the Network Adapter and Send it towards the destination.

viii. The Dude

The Dude network monitor is a new application by MikroTik which can dramatically improve the way you manage your network environment. It will automatically scan all devices within specified subnets, draw and layout a map of your networks, monitor services of your devices and alert you in case some service has problems.

Main Features:

Auto network discovery and layout

- Discovers any type or brand of device
- Device, Link monitoring, and notifications
- Includes SVG icons for devices, and supports custom icons and backgrounds
- Easy installation and usage
- Allows you to draw your own maps and add custom devices
- Supports SNMP, ICMP, DNS and TCP monitoring for devices that support it
- Individual Link usage monitoring and graphs
- Direct access to remote control tools for device management
- Supports remote Dude server and local

Local
Settings
Discover
Tools
Layer: links
Zoom: 100%
Add item -
Remove item -
Copy -
Paste -
Lock -
Drag -
Select -
Map settings -
Find -
Item alignment -
0.0.0.0/0
10.10.2.2
cpu: 5%
Demo (5660) -35
Rx: 11.3 kbps (54 Mbps)
Tx: 33.9 kbps (54 Mbps)
Demo (5660) -62
Rx: 72.8 kbps (54 Mbps)
Tx: 2.34 Mbps (54 Mbps)
- wireless link with traffic data
66.228.113.24
cpu: 18% disk: 47%
Demo2 (5680) -57
Rx: 64 bps (6 Mbps)
Tx: 64 bps (54 Mbps)
66.228.113.25
cpu: 7% disk: 34%
- device
Rx: 0 bps
Tx: 0 bps
Rx: 2.92 Mbps
Tx: 85.5 kbps
192.168.212.0/24
192.168.222.0/24
192.168.213.0/24
159.148.147.209
cpu: 21% disk: 76%
192.168.223.0/24
- network
- link
192.168.232.0/24
192.168.233.0/24

10.5.104.0/24 - Network Map
General
Polling
Outages
Appearance
Background
Export
Ok
Cancel
Apply
Notes
Name: 10.5.104.0/24
Default Zoom: 100%
Status: partially down
Devices:
Partially Down - 1
Up - 16
Report Scan Status
Network
Progress
Next S...
192.168.88.0/24
24
00:59:32
Auto Scan:

Practical No. 2

a. **Use Proxy Workbench to see the data passing through it and save the data to file.** Proxy Workbench is a unique proxy server ideal for developers, trainers and security experts that displays its data in real-time. You can actually see the data flowing between your e-mail client and the e-mail server, web browser and web server or even analyse FTP in both Passive and Active modes. In addition, the 'pass through' protocol handler enables analysis of protocols where the server does not readily change.

The best feature is the animated connection diagram that graphically represents the history of each socket connection and allows you to drill into the finest of detail. This animation can even be exported to HTML and saved to the web!

b. Perform Network Discovery using the following tools:

i. Solar Wind Network Topology Mapper

SolarWinds Network Topology Mapper (NTM) **shows nodes on your network, indicates and updates status both for the nodes and the network connections between them in interrelated, scalable maps with customizable icons**.

ii. OpManager

OpManager is an advanced network monitoring tool which offers fault management, supporting over WAN links, Router, Switch, VoIP & servers. It can also perform performance management.

iii. Network View

NetworkView is a network visualization tool that aims to provide a simple interface for the complex function involved in the discovery and monitoring of multi-vendor IP networks. With NetworkView you can get a quick overview of your network, whether it is a small office or a corporate network. Version 3 adds functionalities oriented to network management tasks. NetworkView uses multiple methods such as ICMP, MDNS, SSDP, DNS, NetBIOS, SNMP MIB2, Bridge MIB, LLDP, CPD and propietary MIB's to discover devices and generates a graphical representation of your network. NetworkView generates views of both logical and physical network structure. Virtual structure representation is also displayed for wireless systems (Cisco, Aruba/Alcatel-Lucent and Fortinet).

iv. LANState Pro

LANState is **a simple network topology mapping, host monitoring, and management program**. Monitor the service availability. Manage servers, computers, switches, and other devices easier using the graphic map. Access devices' properties, RDP, web UI faster.

c. Use the following censorship circumvention tools:

i. Alkasir

Alkasir was created to bypass restrictions imposed by ISPs, "*to allow users to access information about their countries and regions that are concealed by the states* *mainly because of political reasons.*

ii. Tails OS

Tails OS is used by journalists, activists, and others **to keep their digital activity safe and anonymous**. Learn about the operating system and how to source it safely. Tails, which stands for The Amnesic Incognito Live System, is an open-source, security and privacy-focused operating system.

d. Use Scanning Tools for Mobile – Network Scanner, Fing – Network Tool, Network Discovery Tool, Port Droid Tool

There are several basic and advanced network tools available for the Mobile device on application stores. The following are some effective tools for network Scanning.

Network Scanner
-Detailed hosts information
-Very fast searching
-Easy to use
Network Scanner
First Row
UNINSTALL
OPEN
Contains ads • In-app purchases
100
4.5
Downloads
Tools
Similar
IP discover, Port scanner
WHAT'S NEW
Network Scann...
IP: 192.168.0.100/24
SSID: "IPSpecialist"
MODE: WIFI (39 Mbps)
IP calculator
DNS lookup
Whois
Traceroute
Port scanner
192.168.0.100 (This device)
192.168.0.1 (Gateway)
192.168.0.106
192.168.0.104

Practical No. 3

a. Perform Enumeration using the following tools:

i. Nmap

NMAP, as we know, is a powerful networking tool which supports many features and commands. Operating System detection capability allows to send TCP and UDP packet and observe the response from the targeted host. A detailed assessment of this response bring some clues regarding nature of an operating system disclosing the type an OS. To perform OS detection with nmap perform the following: nmap -O<ip address>

ii. NetBIOS Enumeration Tool

NetBIOS stands for Network Basic Input Output System. It **Allows computer communication over a LAN and allows them to share files and printers**. NetBIOS names are used to identify network devices over TCP/IP (Windows).

```
┌──(ritik㉿ritik)-[~]
└─$ netstat -a
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 ritik:45204             del12s05-in-f4.1e:https ESTABLISHED
tcp        0      0 ritik:49222             server-13-224-20-:https ESTABLISHED
tcp        0      0 ritik:34744             ec2-35-167-149-24:https ESTABLISHED
tcp        0      0 ritik:58126             ec2-35-161-6-128.:https ESTABLISHED
tcp        0      0 ritik:55236             104.18.32.68:http       TIME_WAIT
tcp        0      0 ritik:60936             98.203.120.34.bc.:https ESTABLISHED
tcp        0      0 ritik:43858             104.22.24.131:https     ESTABLISHED
tcp        0      0 ritik:37840             20.120.65.166:https     ESTABLISHED
tcp        0      0 ritik:46330             104.16.122.175:https    ESTABLISHED
udp        0      0 ritik:bootpc            WS-GFGDC01.ad.ge:bootps ESTABLISHED
raw6       0      0 [::]:ipv6-icmp          [::]:*                  7
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ACC ]     STREAM     LISTENING     197448   /run/user/1000/speech-dispatcher/speechd.sock
unix  2      [ ACC ]     STREAM     LISTENING     17408    /tmp/.X11-unix/X1
unix  2      [ ACC ]     STREAM     LISTENING     19999    @/tmp/.ICE-unix/1182
unix  3      [ ]         DGRAM      CONNECTED     14870    /run/systemd/notify
```

iii. SuperScan

SuperScan is a multi-functional tool that will help you manage your network and make sure your connections and TCP ports are working as well as they should be. One of the best features or advantages of this tool is just how quickly it works. The scans are made very rapidly and faster than with most other scanning tools out there.

iv. Hyena

Hyena is GUI based, NetBIOS Enumeration tool that shows Shares, User login information and other related information

v. SoftPerfect Network Scanner Tool

SoftPerfect Network Scanner can ping computers, scan ports, discover shared folders and retrieve practically any information about network devices via WMI, SNMP, HTTP, SSH and PowerShell.

vi. OpUtils

OpUtils is **a IP address and Switch port management software that is geared towards helping engineers efficiently monitor, diagnose and troubleshoot IT resources**. OpUtils complements existing management tools by providing trouble shooting and real-time monitoring capabilities.

vii. SolarWinds Engineer's Toolset

Engineer's Toolset **provides the tools you need as a network engineer or consultant to get your job done**. Toolset includes solutions that provide diagnostic, performance, and bandwidth measurements.

viii. Wireshark

Wireshark is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education. Originally named Ethereal, the project was renamed Wireshark in May 2006 due to trademark issues

b. Perform the vulnerability analysis using the following tools:

i. Nessus

Nessus is a proprietary vulnerability scanner developed by Tenable, Inc. Tenable.io is a subscription-based service. Tenable also contains what was previously known as Nessus Cloud, which used to be Tenable's Software-as-a-Service solution. Nessus is an open-source network vulnerability scanner that uses the Common Vulnerabilities and Exposures architecture for easy cross-linking between compliant security tools. In fact, Nessus is one of the many vulnerability scanners used during vulnerability assessments and penetration testing engagements, including malicious attacks. Nessus is a tool that checks computers to find vulnerabilities that hackers COULD exploit.

ii. OpenVas

OpenVAS is a full-featured vulnerability scanner. Its capabilities include unauthenticated and authenticated testing, various high-level and low-level internet and industrial protocols, performance tuning for large-scale scans and a powerful internal programming language to implement any type of vulnerability test.
The scanner obtains the tests for detecting vulnerabilities from a feed that has a long history and daily updates.

OpenVAS has been developed and driven forward by the company Greenbone Networks since 2006. As part of the commercial vulnerability management product family Greenbone Enterprise Appliance, the scanner forms the Greenbone Community Edition together with other open-source modules.

Practical No. 4

a. Perform mobile network scanning using NESSUS

Nessus has implemented new features to help users combat mobile threats. Network-based scanning is not the right approach to identify vulnerabilities on mobile devices, due in large part to the fact that most devices are in "sleep" mode and/or using a 3G/4G network. However, MDM (Mobile Device Management) technologies maintain information about the devices, including information about security vulnerabilities.

With Nessus Manager, the Nessus Mobile Devices plugin family allows you to obtain information from devices registered in a Mobile Device Manager (MDM) and from Active Directory servers that contain information from Microsoft Exchange Servers.

- To query for information, the Nessus scanner must be able to reach the Mobile Device Management servers. Ensure no screening devices block traffic to these systems from the Nessus scanner. In addition, you must give Nessus administrative credentials (for example, domain administrator) to the Active Directory servers.

- To scan for mobile devices, you must configure Nessus with authentication information for the management server and the mobile plugins. Since Nessus authenticates directly to the management servers, you do not need to configure a scan policy to scan specific hosts.

- For ActiveSync scans that access data from Microsoft Exchange servers, Nessus retrieves information from phones that have been updated in the last 365 days.

b. Perform the System Hacking using the following tools:

i. Winrtgen

In this article, we will go through the process of generating rainbow tables using WinRTGen.

To generate rainbow tables first we will have to modify the properties of WinRTGen according to our need, and to do so Click on "**Add Table**". After this, a new box will appear named "**Rainbow Table Properties**"

In the "**Rainbow Table Properties**" window we have the option to modify settings in order to generate rainbow tables according to our needs. The following properties can be modified:

- **Hash**: The type of encryption we want the rainbow table to be generated. For example MD5, MD4, SHA1, etc.

After assigning the values to the properties according to our needs click on “Benchmarks”. This will show the estimated time, Hash speed, Step speed, Table Pre-computing time, etc. that will be required to generate the Rainbow Table according to assigned properties.

After "Benchmark" click on "Ok". This will add the Rainbow Table to the queue in the main window of WinRTGen .

After this click on "Rainbow Table" You want to start processing and click "OK" .

After clicking on 'OK' the WinRTGen" will start generating a rainbow table.

After completion, the window will appear as follows.

This table will be saved to your WinRTGen Directory.

ii. PWDump

The Security Account Manager, or SAM for short, controls all user accounts and passwords. Every password is hashed before being saved in SAM. Passwords that are hashed and saved in SAM can be retrieved in the registry; simply open the Registry Editor and navigate to HKEY LOCAL MACHINESAM. SAM is located in C:\Windows\System32\config.

This utility was created by Tarasco. This utility dumps the system's SAM file's credentials after extracting it.

This utility was created by Tarasco. This utility dumps the system's SAM file's credentials after extracting it. Simply enter the following line on the command prompt after downloading to use this tool:

PwDump7.exe

As a result, it will spill all the hashes kept in the SAM file. The next step is to use the commands below to save the registry values for the SAM file and system file in a system file:

reg save hklm\sam c:\sam reg

save hklm\system c:\system **iii.**

Ophcrack

When it comes to free Windows password crackers, users usually opt for Ophcrack as it is free and easily available.

Step 1: Since we are assuming that your Windows PC is locked and you do not know the password, the first step needs to be carried out on a different PC with internet access and administrator privileges.

Step 2 : Download the correct version of Ophcrack Live CD from the official website to the second PC.

Step 3 : Burn the ISO file to a USB or CD. To do this, you will need an ISO burning application. Now proceed to the next step of the password reset process.

Step 4 : Remove the bootable media from the second PC and insert it into your locked Windows machine. Let the computer boot up from this media instead of the native Windows installation. This is made possible by the fact that Ophcrack itself contains a small operating system that can run independently of your Windows OS. In a few moments, you will see the Ophcrack interface on your computer.

Step 5 : You will now see a menu with 4 options. Leave it on the default option, which is automatic. After a few seconds, you will see the Ophcrack Live CD loading and then the disk partition information being displayed as Ophcrack identifies the one with the SAM file. **Step 6** : Once the process has been complete, you will see a window with several user accounts and their passwords displayed in column format. Against the previously locked username, look for an entry in the NT Pwd column.

Step 7: This will be your recovered password, so note it down. You can now remove the Live CD from the drive and restart your computer. You will be able to login to your user account using the password that was recovered by Ophcrack.

iv. Flexispy

FlexiSPY is a phone application which comes with an android keylogger for the phone as a feature. It will always appear in the list whenever one is speaking about the world's best spy phone applications. This app comes with everything you expect when looking for a monitoring system for your phone.

It will help you record phone calls, capture SMS, WhatsApp messages, even capture keystrokes, allow you to read emails, read Facebook messages.

The app will as well track the device and you know what, from where you are you can turn on its recorder and record conversations without the owner noticing.

v. NTFS Stream Manipulation

NTFS is a filesystem that stores files utilizing two data streams known as NTFS data streams, as well as file attributes. The first data stream contains the security descriptor for the file to be stored, such as permissions, while the second contains the data contained within a file. Another form of the data stream that can be found within each file is an alternate data stream (ADS).

ADS is a file attribute available solely in NTFS, and it refers to any type of data associated with a file but not in the file itself on an NTFS system. NTFS ADS is a Windows hidden stream that stores file metadata such as properties, word count, access and author name, and modification timings.

ADSs can fork data into existing files without changing or altering their functionality, size, or display to file-browsing utilities. They enable an attacker to inject malicious code into files on a vulnerable system and execute them without the user knowing. Attackers use ADS to hide rootkits or hacker tools on a breached system and allow users to execute them while hiding from the system administrator.

Once the ADS is attached to a file, the size of the original file will not change. One can only identify the changes in files through modification of timestamps, which can be innocuous.

Creation of NTFS streams:

When the user reads or writes a file, their only manipulation in the main data stream by default. The following is the syntax of ADSs filename.extension:alternativeNmae

Open the terminal and type the following command to create a file named file_1.txt. echo "this is file no 1" > file_1.txt

Now, type the following command to write to the stream named secret.txt. echo "this is a hidden file inside the file_1.txt" > file_1.txt:secret.txt

We've just created a stream named secret.txt that is associated with file_1.txt and when you look at the file_1.txt you will only find the data present in file_1.txt. And also stream will not be shown in the directory as well.

The following command can be used to view or modify the stream hidden in file_1.txt notepad file_1.txt:secret.txt

Note: Notepad is a stream-compliant application. Never use alternative streams to store sensitive information.

Hiding Trojan.exe in note.txt file stream:

The following command has used the copy the trojan.exe into a note.txt(stream)

C:\test>type Trojan.exe > note.txt:Trojan.exe

Here type command is used to hide trojan in the ADS inside an existing file.

After hiding trojan.exe behind note.txt, we need to create a link to launch the trojan.exe file from the stream. The following command is used to create a shortcut in the stream.

C:\test>mklink game.exe note.txt:Trojan.exe

Type game.exe to run the trojan that is hidden behind the note.txt. Here, game.exe is the shortcut created to launch trojan.exe.

vi. ADS Spy

AdSpy offers the most search options of any Ad Intelligence Tool, so you can find the data you want, how you want. Search in the usual way: ad text, URL, page name. Search true data from user reactions in advert comments. Be as rigorous as you need to: search or filter by affiliate network, affiliate ID, Offer ID, landing page technologies - whatever helps you find the information you can work with. Open ADS Spy application and select the option if you want to:

- Quick Scan
- Full Scan
- Scan Specific Folder

As we store the file in the Document folder, Selecting Document folder to scan particular folder only.

Select an Option, if you want to scan for ADS, click "**Scan the system for ADS**"/ or click **removes** button to remove the file

As shown in the figure below, ADS Spy has detected the **Testfile.txt:hidden.txt** file from the directory.

Figure 6-50 ADS Detection

vii. Snow

Create a text file with some data in the same directory where Snow Tool is installed.

Go to Command Prompt

Change the directory to run Snow tool

Type the command

Snow –C –m “text to be hide” –p “password” <Sourcefile> <Destinationfile>

The source file is a Hello.txt file as shown above. Destination file will be the exact copy of source file containing hidden information.

Go to the directory; you will a new file **HelloWorld.txt**. Open the File

New File has the same text as an original file without any hidden information. This file can be sent to the target.

Recovering Hidden Information

On destination, Receiver can reveal information by using the command

Snow –C –p "password123" HelloWorld.txt

As shown in the above figure, File decrypted, showing hidden information encrypted in the previous section.

viii. Quickstego

Image Steganography using QuickStego

1. Open QuickStego Application

2. Upload an Image. This Image is term as **Cover**, as it will hide the text.

3. Enter the Text or Upload Text File

4. Click Hide Text Button

5. Save Image

This Saved Image containing Hidden information is termed as Stego Object.

Recovering Data from Image Steganography using QuickStego

1. Open QuickStego
2. Click Get Text

3. Open and Compare Both Images

Left Image is without Hidden Text; Right Image is with hidden text

ix. Clearing Audit Policies

Enabling and Clearing Audit Policies

To check command's available option Enter
C:\Windows\system32> **auditpol /?**

```
Administrator: Command Prompt
C:\Windows\system32>auditpol /?
Usage: AuditPol command [<sub-command><options>]


Commands (only one command permitted per execution)
  /?              Help (context-sensitive)
  /get            Displays the current audit policy.
  /set            Sets the audit policy.
  /list           Displays selectable policy elements.
  /backup         Saves the audit policy to a file.
  /restore        Restores the audit policy from a file.
  /clear          Clears the audit policy.
  /remove         Removes the per-user audit policy for a user account.
  /resourceSACL   Configure global resource SACLs


Use AuditPol <command> /? for details on each command

C:\Windows\system32>_
```

Enter the following command to enable auditing for System and Account logon:
C:\Windows\system32>**auditpol /set /category:"System","Account logon" /success:enable /failure:enable**

```
Administrator: Command Prompt

Commands (only one command permitted per execution)
  /?              Help (context-sensitive)
  /get            Displays the current audit policy.
  /set            Sets the audit policy.
  /list           Displays selectable policy elements.
  /backup         Saves the audit policy to a file.
  /restore        Restores the audit policy from a file.
  /clear          Clears the audit policy.
  /remove         Removes the per-user audit policy for a user account.
  /resourceSACL   Configure global resource SACLs


Use AuditPol <command> /? for details on each command

C:\Windows\system32>auditpol /set /category:"System","Account logon" /success:enable /failure:enable
The command was successfully executed.

C:\Windows\system32>_
```

To check Auditing is enabled, enter the command

C:\Windows\system32>**auditpol logon","System"/get /category:"Account**

To clear Audit Policies, Enter the following command
C:\Windows\system32>**auditpol /clear**
Are you sure (Press N to cancel or any other key to continue)?**Y**

To check Auditing, enter the command
C:\Windows\system32>**auditpol /get /category:"Account logon","System"**

x. Clearing Logs 1. Go to

Kali Linux Machine

2. Open the /**var** directory:
3. Go to **Logs** folder:

4. Select any log file:

5. Open any log file; you can delete

Practical No. 5

a. Use wireshark to sniff the network.

Wireshark is a GUI-based packet capture program. As noted, it comes with some command-line programs. There are a lot of advantages to using Wireshark. First, it gives us a way to view the packets easily, moving around the complete capture. Unlike with tcpdump and tshark, we see the entire network stack in Wireshark, which technically makes what we have captured frames rather than packets.

- Start Wireshark. Under the "Capture" header, select the "Interface List" option; or click on the "Interfaces" button on the toolbar:
  This will bring up a list of network interfaces that Wireshark is able to capture packets from:

List of available capture interfaces

Select the network adapter (wired or wireless) that you are currently using to connect to the Internet, and hit the "Start" button. This will take you to the main window:

Wireshark is now capturing live network activity on your network interface. Notice that the list of packets is color-coded to highlight different types of network traffic.

- Open your web browser and navigate to a few random web pages - observe that the network packets corresponding to your web browsing activity are captured and show up in Wireshark as well.
- By default, the list of captured packets will keep scrolling automatically during a live capture. You can toggle this on/off using the AutoScroll toggle button in the toolbar.
- After letting the capture run for a couple of minutes, press the stop capture button. Do not close this capture session.

Filtering the Packet List

Capturing network traffic for a couple minutes could include traffic on many different protocols such as ARP, TCP, UDP, DNS, HTTP, etc.

We may not be interested in all of these, depending on what we are trying to achieve. Fortunately, Wireshark allows us to filter the list based on different criteria using the "Filter" toolbar:

Filter toolbar

Let us take a look at the HTTP traffic that occurs when we browse the web.

In the filter toolbar, type “http” and then click on “Apply”. The window will now list only captured packets related to HTTP traffic:

Examining HTTP Traffic

The HTTP traffic that occurs during web browsing.

- Stop and close any capture that you may have open, and start a new capture.
- Set the filter to show only HTTP traffic.

Start with the HTTP request sent from your web browser.

- In your web browser, navigate to some webpage like http://wwwscf.usc.edu/~csci571/Special/HTTP/proxy.html.
- In the top frame of the Wireshark main window, look for the packet that corresponds to your request. This contains the URL in the “Info” section. Select this packet.
- In the middle frame of the Wireshark window, expand the “Hypertext Transfer Protocol” section. Notice the details given for the: ○ GET request
    - ○ Host
    - ○ User-Agent ○ Accepts ○

cookie ○
etc

Take a look at the HTTP response to the above request.

In the top frame of the Wireshark main window, find and select the "HTTP/1.1 200 OK" packet immediately below the request for proxy.html. This is the response containing the requested web page.

Again, expand the "Hypertext Transfer Protocol" section. Notice the details given for ○ Cache-Control ○ Content-Type

○ Server ○
Etc

Details of incoming HTTP response corresponding to proxy.html

b. Use SMAC for MAC Spoofing.

SMAC is a MAC address changer that has a simple-to-use graphical interface that enables the less experienced user all the way up to the guru to change a piece of hardware's MAC address. The less experienced user will appreciate the random generator whereas the guru will appreciate the ability to hand enter a new MAC address.

Once it is installed, you will find the application launcher in a Start Menu subdirectory called KLC. Click on that folder and you will see SMAC 2.0. Click on that launcher and the SMAC main window (**Figure A**) will open.

Using SMAC can be very simple, depending on how you want to use it. The simplest way to use SMAC is to assign a random MAC address to a piece of hardware. Before we actually assign a new address, let's take a look at the other hardware on the machine. In the main window there is a check box that tells SMAC to show only active hardware. This checkbox is checked by default. Uncheck that box and your listing will grow, depending on the hardware on your machine. Take a look at **Figure B** to see how much the listing grows on my laptop that includes wireless, wired, and dial-up connections.

Figure A

Figure B

When you click on a different listing, the information about that hardware will be displayed below. Let's change the MAC address of the Wired Marvell Yukon PCI-E Faster Ethernet Controller. To do this, select that entry from the list and click the Random button. As you can see in **Figure C**, the new, random MAC address is displayed in the New Spoofed MAC Address section.

Figure C

The address listed will correspond to a manufacturer list that you can choose from.
If you know you want to spoof your MAC address to that of a specific manufacturer you can select a different manufacturer from the drop-down list. When you make this selection, the address listed will change. You can keep hitting Random until you get an address you like (or you can just take the first random address you get).

Once you have your address, select the Options menu and make sure Automatically Restart Adapter is checked. Once that is checked, hit the Update MAC Address button and the new MAC address will be applied.

c. Use Caspa Network Analyser.

When we correctly deployed Capsa, we cannot wait to start our first capture right away. Capsa 7's new Start Page guides us start an accurate capture mission step by step:

1. Double-click icon on the desktop.
2. In the Start Page, select your NICs (multiple selections available) in the Capture panel first.

Name	IP	Packets/s
☑ Local Area Connection	192.168.6.12	22
☐ Local Area Connection 2		0
☐ TAP-Win32 Adapter V9		0
☐ VMware Network Adapter VMnet1	192.168.58.1	0
☐ VMware Network Adapter VMnet8	192.168.168.1	0

3. Select any Network Profile in the Network Profile panel.

4. Select Full Analysis in the Analysis Profile panel.

5. Click the big Run button to start a capture right away.

This is the common procedure to start a capture, which helps us get accurate and useful analysis data: Select NIC -> Select Network Profile -> Select Analysis Profile -> Run.

d. Use Omnipeek Network Analyzer.

Omnipeek is a high-performance network protocol analyzer, capable of decoding thousands of protocols for fast network troubleshooting and diagnostics, anywhere network issues happen.

Real-Time Network Protocol Analyzer

Omnipeek provides real-time analysis for every type of network segment – 1/10/40/100 Gigabit, 802.11, and voice and video over IP – and for every level of network traffic.

Intuitive Graphic Displays and Visualization

Omnipeek delivers intuitive visualization and effective forensics for faster resolution of network and application performance issues and security investigations.

Best-In-Class Network Analysis Workflow

Widely recognized as the best network analysis workflow in the industry, we make it easy to drill down to a single packet – all from a single pane of glass.

WiFi Troubleshooting

The Omnipeek WiFi adaptor is a USB-connected WLAN device designed for wireless packet capture. The 802.11ac adapter supports 802.11ac capture up to 2 transmit/receive streams (866Mbps wireless traffic) and supports 20MHz, 40MHz, and 80MHz channel operation.

Monitor Distributed Networks Remotely
Integrating with LiveCapture, Omnipeek extends network monitoring and visibility for troubleshooting application-level issues at remote sites and branches, WAN links, and data centers.

Voice and Video Monitoring and Troubleshooting
Monitor and troubleshoot voice and video over IP traffic in real-time with high-level multi-media summary statistics, call playback, and comprehensive signaling and media analyses.

Simplify Troubleshooting Remote Devices

Easily troubleshoot end-user devices remotely and securely with encrypted files, avoiding the need to travel to a user's location.

Practical No. 6

a. Use Social Engineering Toolkit on Kali Linux to perform Social Engineering using Kali Linux.

We are using Kali Linux Social Engineering Toolkit to clone a website and send clone link to victim. Once Victim attempt to login to the website using the link, his credentials will be extracted from Linux terminal.

Procedure:

1. Open Kali Linux
2. Go to Application
3. Click Social Engineering Tools 4. Click Social Engineering Toolkit

5. Enter "Y" to proceed.

6. Type "1" for Social Engineering Attacks

7. Type “2” for website attack vector

8. Type “2” for Site Cloner

9. Type IP address of Kali Linux machine (10.10.50.200 in our case).

11. Type target URL

12. Now, http://10.10.50.200 will be used. We can use this address directly, but it is not an effective way in real scenarios. This address is hidden in a fake URL and forwarded to the victim. Due to cloning, the user could not identify the fake website unless he observes the URL. If he accidentally clicks and attempts to log in, credentials will be fetched to Linux terminal. In the figure below, we are using http://10.10.50.200 to proceed.
13. Login using username and Password

Username: admin

Password: Admin@123

14. Go back to Linux terminal and observe.

Username admin and password is extracted. If the user types it correctly, exact spelling can be used. However, you will get the closest guess of user ID and password. The victim will observe a page redirect, and he will be redirected to a legitimate site where he can re-attempt to log in and browse the site.

b. Perform the DDOS attack using the following tools:

i. HOIC

High Orbit Ion Cannon (HOIC) is a free, open-source network stress application developed by Anonymous, a hacktivist collective, to replace the Low Orbit Ion Cannon (LOIC). Used for denial of service (DoS) and distributed denial of service (DDoS) attacks, it functions by flooding target systems with junk HTTP GET and POST requests.

Widespread HOIC availability means that users having limited knowledge and experience can execute potentially significant DDoS attacks. The application can open up to 256 simultaneous attack sessions at once, bringing down a target system by sending a continuous stream of junk traffic until legitimate requests are no longer able to be processed.

ii. LOIC

The LOIC was originally developed by Praetox Technologies as a stress testing application before becoming available within the public domain. The tool is able to perform a simple dos attack by sending a large sequence of UDP, TCP or HTTP requests to the target server. It's a very easy tool to use, even by those lacking any basic knowledge of hacking. The only thing a user needs to know for using the tool is the URL of the target. A would-be hacker need only then select some easy options (address of target system and method of attack) and click a button to start the attack.

The tool takes the URL of the target server on which you want to perform the attack. You can also enter the IP address of the target system. The IP address of the target is used in place of an internal local network where DNS is not being used. The tool has three chief methods of attack: TCP, UDP and HTTP. You can select the method of attack on the target server. Some other options include timeout, TCP/UDP message, Port and threads. See the basic screen of the tool in the snapshot above in Figure.

- **Step 1:** Run the tool.
- **Step 2:** Enter the URL of the website in The URL field and click on Lock O. Then, select attack method (TCP, UDP or HTTP). I will recommend TCP to start. These 2 options are necessary to start the attack.

Figure3: LOIC in action (I painted the URL and IP white to hide the identity of the victim in snap)

- **Step 3:** Change other parameters per your choice or leave it to the default. Now click on the Big Button labeled as "IMMA CHARGIN MAH LAZER." You have just mounted an attack on the target.

After starting the attack you will see some numbers in the Attack status fields. When the requested number stops increasing, restart the LOIC or change the IP. You can also give the UDP attack a try. Users can also set the speed of the attack by the slider. It is set to faster as default but you can slow down it with the slider. I don't think anyone is going to slow down the attack.

iii. HULK

HULK is an abbreviation for **H**TTP **U**nbearable **L**oad **K**ing, which is a web server Distributed Denial of Service tool. It is mainly designed for research purpose, and helps pen testers check the efficiency of a server. With its help, security specialists can find loopholes in their security implementation against DDoS, and correct them before an actual threat actor exploits it. Hulk begins the HTTP flooding attack with a typical TCP handshake. So, the SYN request is sent first, SYN ACK comes the next, and ACK thereafter.

Once the first request bypasses the hurdles, the user agent starts sending diverse HTTP GET requests to the target URL. For this, it makes use of a randomized suffix.

Observation 4

The host sends out various HTTP GET requests with different/randomized suffices and receives the response as 200 (OK).

No.	Time	Source	Destination	Protocol	Length	Info
3	0.000109	10.128.0.2	10.0.0.2	HTTP	439	GET /?OONNNV=LPVQ HTTP/1.1
21	0.123457	10.0.0.2	10.128.0.2	HTTP	1198	HTTP/1.1 200 OK (text/html)
27	0.244025	10.128.0.2	10.0.0.2	HTTP	439	GET /?OONNNV=LPVQ HTTP/1.1
45	0.366065	10.0.0.2	10.128.0.2	HTTP	1198	HTTP/1.1 200 OK (text/html)
51	0.487810	10.128.0.2	10.0.0.2	HTTP	405	GET /?PLQZYRZW=GXVXTSW HTTP/1.1
69	0.610763	10.0.0.2	10.128.0.2	HTTP	1198	HTTP/1.1 200 OK (text/html)
75	0.732811	10.128.0.2	10.0.0.2	HTTP	405	GET /?PLQZYRZW=GXVXTSW HTTP/1.1
92	0.855685	10.0.0.2	10.128.0.2	HTTP	1198	HTTP/1.1 200 OK (text/html)
99	0.978240	10.128.0.2	10.0.0.2	HTTP	434	GET /?MAHW=MFKR HTTP/1.1
117	1.101789	10.0.0.2	10.128.0.2	HTTP	1198	HTTP/1.1 200 OK (text/html)
123	1.224351	10.128.0.2	10.0.0.2	HTTP	434	GET /?MAHW=MFKR HTTP/1.1
139	1.348224	10.0.0.2	10.128.0.2	HTTP	1198	HTTP/1.1 200 OK (text/html)
145	1.470494	10.128.0.2	10.0.0.2	HTTP	394	GET /?TRYNWWNH=LTXGXW HTTP/1.1
162	1.594277	10.0.0.2	10.128.0.2	HTTP	1198	HTTP/1.1 200 OK (text/html)
169	1.716297	10.128.0.2	10.0.0.2	HTTP	394	GET /?TRYNWWNH=LTXGXW HTTP/1.1
186	1.839118	10.0.0.2	10.128.0.2	HTTP	1198	HTTP/1.1 200 OK (text/html)
193	1.961653	10.128.0.2	10.0.0.2	HTTP	389	GET /?RHF=NJOMVW HTTP/1.1
211	2.085068	10.0.0.2	10.128.0.2	HTTP	1198	HTTP/1.1 200 OK (text/html)

iv. Metasploit

First, select your target's IP address. I am taking **testphp.vulnweb.com** as a victim. So you know how to get an IP address from a domain name. Simple doping and that will give to domain IP address.

```
┌──(kali㉿kali)-[~]
└─$ ping testphp.vulnweb.com
PING testphp.vulnweb.com (18.192.172.30) 56(84) bytes of data.
64 bytes from ec2-18-192-172-30.eu-central-1.compute.amazonaws.com (18.192.
172.30): icmp_seq=1 ttl=39 time=206 ms
64 bytes from ec2-18-192-172-30.eu-central-1.compute.amazonaws.com (18.192.
172.30): icmp_seq=2 ttl=39 time=228 ms
^C
--- testphp.vulnweb.com ping statistics ---
3 packets transmitted, 2 received, 33.3333% packet loss, time 2004ms
rtt min/avg/max/mdev = 205.509/216.576/227.643/11.067 ms
```

So now I know the victim's IP Address **18.192.182.30.**

Launching Metasploit by typing **msfconsole** in your kali terminal

```
      dBBBBBBb  dBBBP dBBBBBBP dBBBBBBb  .                    o
         '    dB'                          BBP
    dB'dB'dB' dBBP     dBP     dBP BB
   dB'dB'dB' dBP      dBP     dBP  BB
  dB'dB'dB' dBBBBP   dBP     dBBBBBBB

                                   dBBBBBP  dBBBBBb  dBP    dBBBBP dBP dBBBBBBP
                                  .                 dB' dBP    dB'.BP
                                  |       dBP    dBBBB' dBP    dB'.BP dBP    dBP
                                 --o--    dBP    dBP    dBP    dB'.BP dBP    dBP
                                  |     dBBBBP dBP    dBBBBP dBBBBP dBP    dBP


        o                 To boldly go where no
                           shell has gone before


       =[ metasploit v6.0.15-dev                          ]
+ -- --=[ 2071 exploits - 1123 auxiliary - 352 post       ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 7 evasion                                       ]

Metasploit tip: Metasploit can be configured at startup, see msfconsole --help to learn more

msf6 >
```

Then use the select the auxiliary “auxiliary/dos/TCP/synflood” by typing the following command.

Msf6 > use auxiliary/dos/tcp/synflood

Msf6> show options

```
       =[ metasploit v6.0.15-dev                          ]
+ -- --=[ 2071 exploits - 1123 auxiliary - 352 post       ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 7 evasion                                       ]

Metasploit tip: Metasploit can be configured at startup, see msfconsole --help to learn more

msf6 > use auxiliary/dos/tcp/synflood
msf6 auxiliary(dos/tcp/synflood) > show options

Module options (auxiliary/dos/tcp/synflood):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   INTERFACE                   no        The name of the interface
   NUM                         no        Number of SYNs to send (else unlimited)
   RHOSTS                      yes       The target host(s), range CIDR identifier, or hosts fi
le with syntax 'file:<path>'
   RPORT      80               yes       The target port
   SHOST                       no        The spoofable source address (else randomizes)
   SNAPLEN    65535            yes       The number of bytes to capture
   SPORT                       no        The source port (else randomizes)
   TIMEOUT    500              yes       The number of seconds to wait for new data

msf6 auxiliary(dos/tcp/synflood) >
```

Now you can see you have all the available options that you can set.

To set an option just you have to type**set** and the **option name** and option.

You have to set two main option

RHOST= target IP Address

RPORT=target PORT Address

Set RPORT 18.192.182.30

Set RPORT 80

```
       =[ metasploit v6.0.15-dev                          ]
+ -- --=[ 2071 exploits - 1123 auxiliary - 352 post       ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 7 evasion                                       ]

Metasploit tip: Metasploit can be configured at startup, see msfconsole --help to learn more

msf6 > use auxiliary/dos/tcp/synflood
msf6 auxiliary(dos/tcp/synflood) > show options

Module options (auxiliary/dos/tcp/synflood):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   INTERFACE                   no        The name of the interface
   NUM                         no        Number of SYNs to send (else unlimited)
   RHOSTS                      yes       The target host(s), range CIDR identifier, or hosts fi
le with syntax 'file:<path>'
   RPORT      80               yes       The target port
   SHOST                       no        The spoofable source address (else randomizes)
   SNAPLEN    65535            yes       The number of bytes to capture
   SPORT                       no        The source port (else randomizes)
   TIMEOUT    500              yes       The number of seconds to wait for new data

msf6 auxiliary(dos/tcp/synflood) >
```

To launch the attack just type. **exploit**

```
msf6 auxiliary(dos/tcp/synflood) > options

Module options (auxiliary/dos/tcp/synflood):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   INTERFACE                   no        The name of the interface
   NUM                         no        Number of SYNs to send (else unlimited)
   RHOSTS                      yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
   RPORT      80               yes       The target port
   SHOST                       no        The spoofable source address (else randomizes)
   SNAPLEN    65535            yes       The number of bytes to capture
   SPORT                       no        The source port (else randomizes)
   TIMEOUT    500              yes       The number of seconds to wait for new data

msf6 auxiliary(dos/tcp/synflood) > set RHOSTS 18.192.182.30
RHOSTS => 18.192.182.30
msf6 auxiliary(dos/tcp/synflood) > exploit
[*] Running module against 18.192.182.30

[*] SYN flooding 18.192.182.30:80 ...
```

to see the packets you can open Wireshark.

So that's how you can perform a DOS attack.

c. Using Burp Suite to inspect and modify traffic between the browser and target application. Burp Suite is a fully featured web application attack tool: it does almost anything that you could ever want to do when penetration testing a web application.

One of Burp Suite's main features is its ability to intercept HTTP requests. Normally HTTP requests go from your browser straight to a web server and then the web server response is sent back to your browser. With Burp Suite, however, HTTP requests go from your browser straight to Burp Suite, which intercepts the traffic.

burp intruder repeater window help
target proxy spider scanner intruder repeater sequencer decoder comparer options alerts
site map scope
Filter: hiding not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders
http://syngress.com
host method URL params status length MIME type title
http://syngres... GET / 200 15928 HTML Syngress.com - Syngress is a pr...
http://syngres... GET /?cur=eur 200 15925 HTML Syngress.com - Syngress is a pr...
http://syngres... GET /?cur=gbp 200 15923 HTML Syngress.com - Syngress is a pr...
http://syngres... GET /?cur=usd 200 15943 HTML Syngress.com - Syngress is a pr...
http://syngres... GET /about-us 200 8795 HTML About Us
http://syngres... GET /certification/ 200 26630 HTML Certification
http://syngres... GET /certification/Cisco-CCNA-CCE... 200 13104 HTML Cisco CCNA/CCENT Exam 640-...
http://syngres... GET /certification/CISSP-Study-Guide/ 200 12349 HTML CISSP Study Guide
http://syngres... GET /certification/CompTIA-A-Certif... 200 13095 HTML CompTIA A+ Certification Study...
http://syngres... GET /certification/CompTIA-Linux-C... 200 12977 HTML CompTIA Linux+ Certification St...
response request
raw headers hex html render
HTTP/1.0 200 OK
Date: Sun, 20 Feb 2011 16:11:40 GMT
Server: Apache
X-Powered-By: Phusion Passenger (mod_rails/mod_rack) 2.2.5
X-Rack-Cache: miss
X-Runtime: 1474
Cache-Control: no-cache, private, max-age=
Set-Cookie:
_syngress_session=
Status: 200
Vary: Accept-Encoding
Content-Type: text/html; charset=utf-8
X-Cache: MISS from smoothwall
Via: 1.1 smoothwall:800 (squid/2.7.STABLE6)
Connection: keep-alive
Content-Length: 15244
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>
Syngress.com - Syngress is a premier publisher of content in the Information Security field. We cover Digital Forensics, Hacking and Penetration Testing, Certification, IT Security and Administration, and more.
</title>
<meta name="description" content="" />
0 matches

Practical No. 7

a. Perform Web App Scanning using OWASP Zed Proxy.

Zed Attack Proxy (ZAP) is a free, open-source penetration testing tool being maintained under the umbrella of the Open Web Application Security Project (OWASP). ZAP is designed specifically for testing web applications and is both flexible and extensible.

To run a Quick Start Automated Scan :

1. Start ZAP and click the **Quick Start** tab of the Workspace Window.
2. Click the large Automated Scan button.
3. In the **URL to attack** text box, enter the full URL of the web application you want to attack.
4. Click the **Attack**

ZAP will proceed to crawl the web application with its spider and passively scan each page it finds. Then ZAP will use the active scanner to attack all of the discovered pages, functionality, and parameters.

ZAP provides 2 spiders for crawling web applications, you can use either or both of them from this screen.

The traditional ZAP spider which discovers links by examining the HTML in responses from the web application. This spider is fast, but it is not always effective when exploring an AJAX web application that generates links using JavaScript.

b. Use droidsheep on mobile for session hijacking

DroidSheep is a simple Android tool for web session hijacking (sidejacking). It listens for HTTP packets sent via a wireless (802.11) network connection and extracts the session id from these packets in order to reuse them.

DroidSheep can capture sessions using the libpcap library and supports: OPEN Networks WEP encrypted networks WPA and WPA2 encrypted networks (PSK only). This software uses libpcap and arpspoof. DroidSheep has been developed with support of the information security team of the University of Trier.

c. Demonstrate the use of the following firewalls:

i. Zonealarm and analyse using Firewall Analyzer.

To open the ZoneAlarm client interface, do one of these:

- Double-click on the ZoneAlarm Security desktop icon.
- Go to MS Windows Start Menu > Check Point > ZoneAlarm > ZoneAlarm Security.
- Use the ZoneAlarm icon in the MS Windows system notification area ("MS Windows System Notification Area Icons and Menus" on page 14).
  The startup page of the ZoneAlarm software client interface consists of these components:
- The main status bar - shows you if YOUR COMPUTER IS SECURE or YOUR COMPUTER IS AT RISK. If the computer is at risk, you can click Fix Now to quickly fix the security problem.
- The three panels:
  1. ANTIVIRUS & FIREWALL - lets you configure the Antivirus and Anti-spyware ("Protecting Your Computer With Antivirus/Anti-Spyware" on page 15) settings, the Firewall "Protecting Your Computer with ZoneAlarm Firewall" on page 30 settings, the Application Control ("Using Application Control for Application Security" on page 44) settings, and the Threat Emulation ("Using Threat Emulation Against ZeroDay Attacks" on page 28) settings
  2. WEB & PRIVACY - lets you configure enable or disable Anti-Keylogger ("Using Anti-Keylogger" on page 65)

3. MOBILITY & DATA - lets you configure Identity Protection ("Identity Protection Service (USA Only)" on page 68) settings.

ii. Comodo Firewall

Comodo Internet Security offers 360° protection against internal and external threats by combining a powerful antivirus, an enterprise class packet filtering firewall, and a threat containment system which automatically runs unrecognized files in a secure, virtual environment.

Click 'Tasks' > 'Firewall Tasks'

- The firewall offers the following main benefits
- Monitor all network traffic to protect your computer against inbound and outbound threats
- Hides your computer's ports from hackers
- Blocks malicious software from transmitting your confidential data over the internet
- The firewall tasks area lets you configure internet access rights per-application, stealth your computer ports, view active connections, and even block all traffic in and out of your computer
- In addition to this tasks screen, you can also configure advanced firewall settings at 'Settings' > 'Firewall'.

d. Use HoneyBOT to capture malicious network traffic.

HoneyBot is a set of scripts and libraries for capturing and analyzing packet captures with PacketTotal.com. Currently, this library provides three scripts:

- capture-and-analyze.py - Capture on an interface for some period of time, and upload capture for analysis.
- upload-and-analyze.py - Upload and analyze multiple packets captures to PacketTotal.com.
- trigger-and-analyze.py - Listen for unknown connections, and begin capturing when one is made. Captures are automatically uploaded and analyzed. **capture-and-analyze.py**

```
usage: capture-and-analyze.py [-h] [--seconds SECONDS] [--interface INTERFACE]
                              [--analyze] [--list-interfaces] [--list-pcaps]
                              [--export-pcaps]

Capture, upload and analyze network traffic; powered by PacketTotal.com.

optional arguments:
-h, --help            show this help message and exit
--seconds SECONDS       The number of seconds to capture traffic for.
  --interface INTERFACE
                      The name of the interface (--list-interfaces to show
                      available)
  --analyze       If included, capture will be uploaded for analysis to PacketTotal.com.
--list-interfaces       Lists the available interfaces.
--list-pcaps            Lists pcaps submitted to PacketTotal.com for analysis.
  --export-pcaps          Writes pcaps submitted to PacketTotal.com for analysis
                      to a csv file.
```

e. Use the following tools to protect attacks on the web servers:

i. ID Server

Download and install ID Server tool.

1. Enter URL or IP address of the target server

2. Enter the **Query The Server**/button.

3. Copy the Extracted information.

Information such as Domain name, open ports, Server type and other information are extracted.

ii. Microsoft Baseline Security Analyzer

The Microsoft Baseline Security Analyzer is a Windows-based Patch management tool powered by Microsoft. MBSA identify the missing security updates and common security misconfigurations.

MBSA is capable of scanning Local system, remote system, and range of the computer.

Select the scanning options as required

MBSA will first get updates from Microsoft, Scan, and then download the security updates.

Microsoft Baseline Security Analyzer 2.3
Microsoft
Baseline Security Analyzer
Microsoft
Currently scanning WORKGROUP\WIN7-1-PC
Done downloading security update information.
Cancel

Microsoft Baseline Security Analyzer 2.3
Microsoft
Baseline Security Analyzer
Microsoft
Report Details for WORKGROUP - WIN7-1-PC (2018-03-07 22:33:31)
Security assessment:
Severe Risk (One or more critical checks failed.)
Computer name: WORKGROUP\WIN7-1-PC
IP address: 10.10.50.202
Security report name: WORKGROUP - WIN7-1-PC (3-7-2018 10-33 PM)
Scan date: 3/7/2018 10:33 PM
Scanned with MBSA version: 2.3.2211.0
Catalog synchronization date:
Security update catalog: Microsoft Update
Sort Order: Score (worst first)
Security Update Scan Results
Score Issue Result
Developer Tools, Runtimes, and Redistributables Security Updates
1 security updates are missing.
What was scanned Result details How to correct this
Windows Security Updates
117 security updates are missing. 3 service packs or update rollups are missing.
What was scanned Result details How to correct this
SQL Server Security Updates
No security updates are missing.
What was scanned Result details
Windows Scan Results
Administrative Vulnerabilities
Print this report
Copy to clipboard
Previous security report
Next security report
OK

In the above figure, MBSA Scanning result showing **Security Update Scan Results**. Security Update scan results are categorized by issue and results showing a number of missing updates.

In the figure above, MBSA Scanning result showing **Administrative Vulnerabilities**. Vulnerabilities such as Password expiry, updates, firewalls issues, accounts and other vulnerabilities are mentioned.

In the above figure, MBSA Scanning result showing **System information, IIS scan results, SQL Server Result and Desktop application results**.

iii. Syhunt Hybrid

Using **Syhunt Hybrid,** go to Dynamic Scanning. This package also supports Code Scanning and Log Scanning.

Enter the URL or IP address

Showing Scanning Results, you click on the vulnerability to check the issue and its solution.

Showing Description of vulnerability detected by the tool. Solution tool will provide a recommendation to resolve the issue.

Practical No. 8

a. Protect the Web Application using dotDefender.

dotDefender allows businesses to protect external websites and internal applications in an affordable, effective and simple manner without involving costly security experts. dotDefender is

a multi-platform solution running on Apache and IIS web servers. Central management ensures a single point of control and reporting for all servers.

You can modify the Default Security Profile or any of the Website Security Profiles.

b. Demonstrate the following tools to perform SQL Injection:

i. Tyrant SQL

Tyrant SQL is a Havij based cross-platform. It's Sqlmap's gui version.

ii. Havij

Havij is an automated SQL Injection tool that helps penetration testers to find and exploit SQL Injection vulnerabilities on a web page.

iii. BBQSQL

BBQSQL is a blind SQL injection framework written in Python. It is extremely useful when attacking tricky SQL injection vulnerabilities. BBQSQL is also a semi-automatic tool, allowing

quite a bit of customization for those hard to trigger SQL injection findings. The tool is built to be database agnostic and is extremely versatile. It also has an intuitive UI to make setting up attacks much easier. Python gevent is also implemented, making BBQSQL extremely fast.

```
root@darknet:~# bbqsql
    _______   _______    ______    ______    ______   __
   |       \ |       \  /      \  /      \  /      \ |  \
   | $$$$$$$\| $$$$$$$\|  $$$$$$\|  $$$$$$\|  $$$$$$\| $$
   | $$__/ $$| $$__/ $$| $$  | $$| $$___\$$| $$  | $$| $$
   | $$    $$| $$    $$| $$  | $$ \$$    \ | $$  | $$| $$
   | $$$$$$$\| $$$$$$$\| $$ _| $$ _\$$$$$$\| $$ _| $$| $$
   | $$__/ $$| $$__/ $$| $$/ \ $$|  \__| $$| $$/ \ $$| $$_____
   | $$    $$| $$    $$ \$$ $$ $$ \$$    $$ \$$ $$ $$| $$     \
    \$$$$$$$  \$$$$$$$   \$$$$$$\  \$$$$$$   \$$$$$$\ \$$$$$$$$
                      \$$$                \$$$

                  _.(-)._
                .'         '.
               / 'or '1'='1  \
               |'-...___...-'|
                \    '='    /
                 ''._____.''
                  /   |   \
                 /.--'|'--.\
               □/'-.__|__.-'\□
                       |
                       □

    BBQSQL injection toolkit (bbqsql)
    Lead Development: Ben Toews(mastahyeti)
    Development: Scott Behrens(arbit)
    Menu modified from code for Social Engineering Toolkit (SET) by: David Kennedy (ReL1K)
    SET is located at: http://www.secmaniac.com(SET)
    Version: 1.0

    The 5 S's of BBQ:
    Sauce, Spice, Smoke, Sizzle, and SQLi



 Select from the menu:

   1) Setup HTTP Parameters
   2) Setup BBQSQL Options
   3) Export Config
   4) Import Config
   5) Run Exploit
   6) Help, Credits, and About

  99) Exit the bbqsql injection toolkit

bbqsql>
```

Practical No. 9

Use Aircrack-ng suite for wireless hacking and countermeasures.

In this case, we have captured some 802.11 (Wireless Network) packets and save the file. Using this file with "**Cupp**" and "**Aircrack-ng.**",we will create a password file and crack the password.

Procedure:

1. Capture some wlan packets using filter "**eth.add==aa:bb:cc:dd:ee"** and save the file.
2. Go to Kali Linux terminal.
3. Change the directory to the desktop.

   root@kali:~# **cd Desktop**
4. Download the "**Cupp"** utility to create wordlist root@kali:~# git clone https://github.com/chetan31295/cupp.git

5. Change the directory to /Desktop/Cupp root@kali:~/Desktop# **cd cupp**
6. List the folders in the current directory.

root@kali:~/Desktop/cupp# **ls** 7. Run the utility **cupp.py**

root@kali:~/Desktop/cupp# **./cuppy.py**

8.Use Interactive Question for user password profiling

root@kali:~/Desktop/cupp# **./cupp.py -i**

9. Provide the closest information about the target. It will increase the chances of successful cracking.
10. You can add keywords.

11. You can add special characters.
12. You can add random numbers.
13. You can enable leet mode.

14.After successful completion, you find a new text file named as the first name you type in interactive option. This file will contain a lot of possible combinations. As shown in the figure below, Albert.txt file has been created in the current directory

15. You can check the file by opening it.

16. Now crack the password using Aircrack-ng with the help of password file created. root@kali:~ # cd root@kali:~ # **aircrack-ng –a2 –b <*BSSID of WLAN Router*> -w /root/Desktop/cupp/Albert.txt '/root/Desktop/WPA.cap'**

WPA.cap is captured packet file.

17. This will start the process, and all keys will be checked

18. The result will either show you the key or refuse to crack from the dictionary

Practical No. 10

Use the following tools for cryptography

i. HashCalc

Calculating MD5 value using HashCalc

1. Open HashCalc tool.

2. Create a new file with some content in it as shown below.

3. Select Data Format as “File” and upload your file

4. Select Hashing Algorithm and Click Calculate
5. Now Select the Data Format to “**Text String**” and Type “**IPSpecialist…**” into Data filed and calculated MD5.

MD5 Calculated for the text string "**IPSpecialist…**" is "**a535590bec93526944bd4b94822a7625**"

6. Now, let's see how MD5 value is changed from minor change.

Just lowering the case of single alphabet changes entire hashing value. MD5 Calculated for the text string "**IPspecialist…**" is "**997bd71ad0158de71f6e97a57261b9a7"**

ii. Advanced Encryption Package

1. Download and Install Advance Encryption Package Latest Version. In this Lab, we are using Advanced Encryption Package 2014 and 2017 to ensure compatibilities on Windows 7 and Windows 10.

2. Select the File you want to Encrypt.
3. Set password
4. Select Algorithm

5. Click Encrypt

6. Compare both Files

7. Now, After forwarding it to another PC, in our case, in Windows 10 PC, decrypting it using Advanced Encryption package 2017.
8. Enter password

9. File Successfully decrypted.

iii. TrueCrypt

TrueCrypt is a leading disk encryption software program that lets you secure disk partitions on your Windows computer. There are times when your hard drive is accessible by other people, such as in an office setting, while travelling, or at home. The data you have on the PC may be vulnerable to attack and compromise your privacy. However, in these moments of risk, **TrueCrypt may just be the tool to protect your data.**

Click Next two times on the following screens to create an encrypted file container with a standard TrueCrypt volume (those are the default options). Click Select File and browse to a location where you want to create the new container. **Make sure it is not in the Dropbox folder if Dropbox is running.** You can name the container anyway you want, e.g. holiday2010.avi.

Click Next on the encryption options page unless you want to change the encryption algorithm or hash algorithm. Select the volume size on the next screen. I suggest you keep it at a few hundred Megabytes tops.

You need to enter a secure password on the next screen. It is suggested to use as many characters as possible (24+) with upper and lower letters, numbers and special characters. The maximum length of a True Crypt password is 64 characters.

Now it is time to select the volume format on the next screen. If you only use Windows computers you may want to select NTFS as the file system. If you use others you may be better of with FAT. Juggle the mouse around a bit and click on format once you are done with that.

Congratulations, the new True Crypt volume has been created.

iv. CrypTool

Cryptool is a free e-learning tool to illustrate the concepts of cryptography. Try Various Encryption/Decryption algorithms.