Practical 4

1. Perform the following data processing using R.

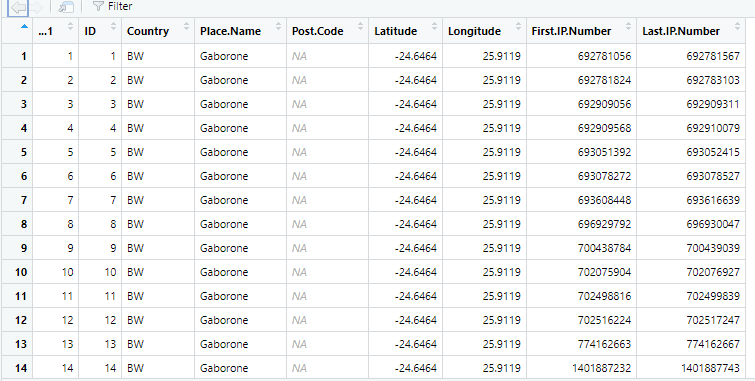
Code

library(readr)

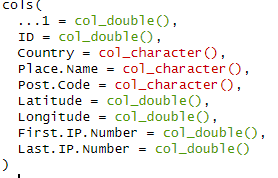
IP\_DATA\_ALL <- read\_csv("E:/NIKHILESH/VKHCG/01-Vermeulen/00-RawData/IP\_DATA\_ALL.csv")

View(IP\_DATA\_ALL)

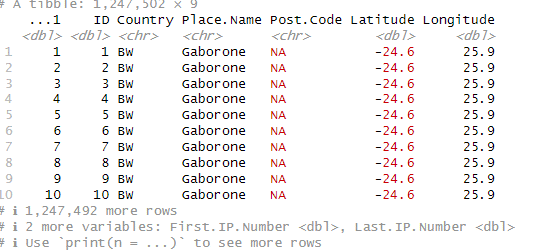
Output



spec(IP\_DATA\_ALL)

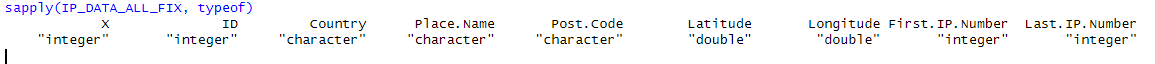


set\_tidy\_names(IP\_DATA\_ALL, syntactic = TRUE, quiet = FALSE)



IP\_DATA\_ALL\_FIX <- read.csv("E:/NIKHILESH/VKHCG/01-Vermeulen/01-Retrieve/01-EDS/01-R/IP\_DATA\_ALL\_FIX.csv")

sapply(IP\_DATA\_ALL\_FIX, typeof)



library(data.table)

hist\_country=data.table(Country=unique(IP\_DATA\_ALL\_FIX[is.na(IP\_DATA\_ALL\_FIX ['Country']) == 0, ]$Country))

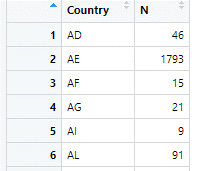
setorder(hist\_country,'Country')

hist\_country\_with\_id=rowid\_to\_column(hist\_country, var = "RowIDCountry")

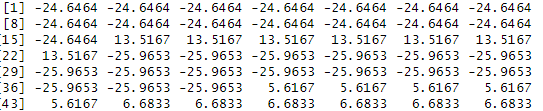
View(hist\_country\_fix)

IP\_DATA\_COUNTRY\_FREQ=data.table(with(IP\_DATA\_ALL\_FIX, table(Country)))

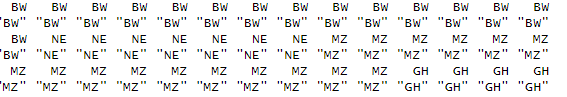
View(IP\_DATA\_COUNTRY\_FREQ)



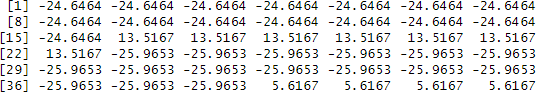
sapply(IP\_DATA\_ALL\_FIX[,'Latitude'], min, na.rm=TRUE)



sapply(IP\_DATA\_ALL\_FIX[,'Country'], min, na.rm=TRUE)



sapply(IP\_DATA\_ALL\_FIX[,'Latitude'], max, na.rm=TRUE)



Finding mean median range and quantile following are the commands are used-

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)

Finding the standard deviation of any column in table the commands will be –

sapply(IP\_DATA\_ALL\_FIX [,'Latitude'], sd, na.rm=TRUE)

**B.** **Program to retrieve different attributes of data.**

Code-

import sys

import os

import pandas as pd

sFileName='E:/NIKHILESH/VKHCG/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='E:/NIKHILESH/VKHCG/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

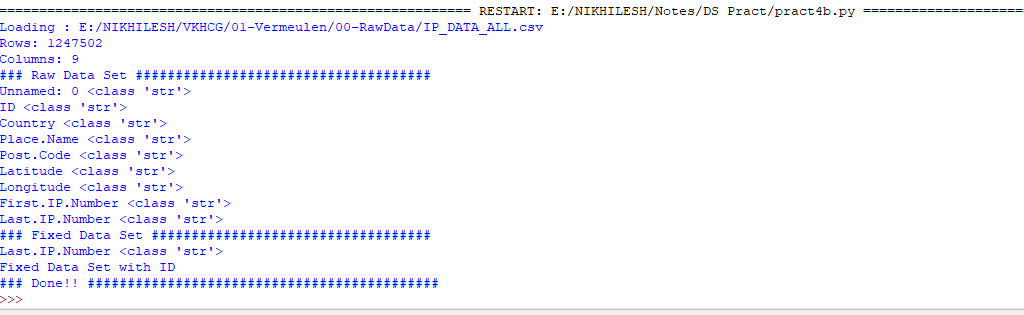
IP\_DATA\_ALL\_with\_ID.index.names = ['RowID']

sFileName2=sFileDir + '/Retrieve\_IP\_DATA.csv'

IP\_DATA\_ALL\_with\_ID.to\_csv(sFileName2, index = True, encoding="latin-1")

print('### Done!! ###')

**Output-**

****

**C.** Data Pattern

**Code**

**Write the program using r Studio**

library(readr)

library(data.table)

FileName=paste0('c:/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$Country)

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)

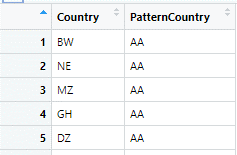
s=chartr(".","u",s)

pattern\_country[r,]$PatternCountry=s;

};

View(pattern\_country)

output



**D.** Loading IP\_DATA\_ALL:

Code

import sys

import os

import pandas as pd

Base='C:/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(IP\_DATA\_ALL\_FIX.head())

print('Fixed Data Set with ID')

IP\_DATA\_ALL\_with\_ID=IP\_DATA\_ALL\_FIX

IP\_DATA\_ALL\_with\_ID.index.names = ['RowID']

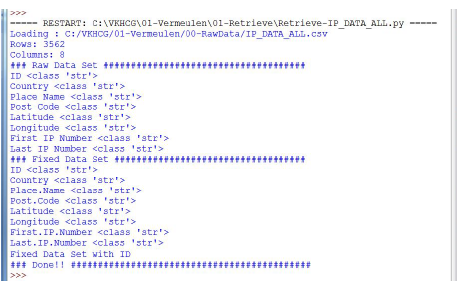
#print(IP\_DATA\_ALL\_with\_ID.head())

sFileName2=sFileDir + '/Retrieve\_IP\_DATA.csv'

IP\_DATA\_ALL\_with\_ID.to\_csv(sFileName2, index = True, encoding="latin-1")

print('### Done!! ############################################')

output



Vermeulen PLC

Code

import sys

import os

import pandas as pd

from math import radians, cos, sin, asin, sqrt

# Function to calculate haversine distance

def haversine(lon1, lat1, lon2, lat2, stype):

# Convert decimal degrees to radians

lon1, lat1, lon2, lat2 = map(radians, [lon1, lat1, lon2, lat2])

dlon = lon2 - lon1

dlat = lat2 - lat1

a = sin(dlat / 2)\*\*2 + cos(lat1) \* cos(lat2) \* sin(dlon / 2)\*\*2

c = 2 \* asin(sqrt(a))

# Determine the radius of Earth based on the unit type

if stype == 'km':

r = 6371 # Radius of Earth in kilometers

else:

r = 3956 # Radius of Earth in miles

# Calculate and return the distance

d = round(c \* r, 3)

return d

# File paths

sFileName = 'E:/NIKHILESH/VKHCG/01-Vermeulen/00-RawData/IP\_DATA\_CORE.csv'

sFileDir = 'E:/NIKHILESH/VKHCG/01-Vermeulen/01-Retrieve/01-EDS/02-Python'

# Check if output directory exists; create if not

if not os.path.exists(sFileDir):

os.makedirs(sFileDir)

# Load the CSV file

print('Loading:', sFileName)

IP\_DATA\_ALL = pd.read\_csv(

sFileName,

header=0,

low\_memory=False,

usecols=['Country', 'Place Name', 'Latitude', 'Longitude'],

encoding="latin-1"

)

# Process the data

IP\_DATA = IP\_DATA\_ALL.drop\_duplicates(subset=None, keep='first', inplace=False)

IP\_DATA.rename(columns={'Place Name': 'Place\_Name'}, inplace=True)

IP\_DATA1 = IP\_DATA.copy()

IP\_DATA1.insert(0, 'K', 1)

IP\_DATA2 = IP\_DATA1.copy()

# Cross-join to calculate pairwise distances

IP\_CROSS = pd.merge(right=IP\_DATA1, left=IP\_DATA2, on='K')

IP\_CROSS.drop('K', axis=1, inplace=True)

# Rename columns for clarity

IP\_CROSS.rename(columns={

'Longitude\_x': 'Longitude\_from', 'Longitude\_y': 'Longitude\_to',

'Latitude\_x': 'Latitude\_from', 'Latitude\_y': 'Latitude\_to',

'Place\_Name\_x': 'Place\_Name\_from', 'Place\_Name\_y': 'Place\_Name\_to',

'Country\_x': 'Country\_from', 'Country\_y': 'Country\_to'

}, inplace=True)

# Calculate distances in kilometers and miles

IP\_CROSS['DistanceBetweenKilometers'] = IP\_CROSS.apply(

lambda row: haversine(

row['Longitude\_from'],

row['Latitude\_from'],

row['Longitude\_to'],

row['Latitude\_to'],

'km'

),

axis=1

)

IP\_CROSS['DistanceBetweenMiles'] = IP\_CROSS.apply(

lambda row: haversine(

row['Longitude\_from'],

row['Latitude\_from'],

row['Longitude\_to'],

row['Latitude\_to'],

'miles'

),

axis=1

)

# Save the result to a CSV file

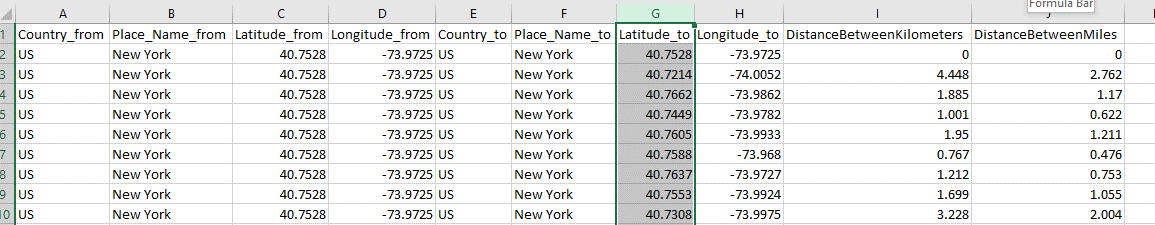
print('Saving results...')

sFileName2 = os.path.join(sFileDir, 'Retrieve\_IP\_Routing.csv')

IP\_CROSS.to\_csv(sFileName2, index=False, encoding="latin-1")

print('### Done!! ############################################')

output –

See the file named Retrieve\_IP\_Routing.csv in C:\VKHCG\01-Vermeulen\01-Retrieve\01-EDS\02-

**Total Records: 22501**

So, the distance between a router in New York (40.7528, -73.9725) to anoher router in New York

(40.7214, -74.0052) is 4.448 kilometers, or 2.762 miles.

Building a Diagram for the Scheduling of Jobs

Code

import sys

import os

import pandas as pd

InputFileName='IP\_DATA\_CORE.csv'

OutputFileName='Retrieve\_Router\_Location.csv'

sFileName='E:/NIKHILESH/VKHCG/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)

sFileDir='E:/NIKHILESH/VKHCG/01-Vermeulen/01-Retrieve/01-EDS/02-Python'

if not os.path.exists(sFileDir):

os.makedirs(sFileDir)

ROUTERLOC = IP\_DATA\_ALL.drop\_duplicates(subset=None, keep='first', inplace=False)

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

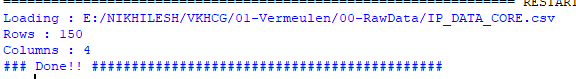
print('Columns :',ROUTERLOC.shape[1])

sFileName2=sFileDir + '/' + OutputFileName

ROUTERLOC.to\_csv(sFileName2, index = False, encoding="latin-1")

print('### Done!! ############################################')

output



Understanding Your Online Visitor Data

Code

import sys

import os

import pandas as pd

import gzip as gz

InputFileName='IP\_DATA\_ALL.csv'

OutputFileName='Retrieve\_Online\_Visitor'

CompanyIn= '01-Vermeulen'

CompanyOut= '02-Krennwallner'

Base='E:/NIKHILESH/VKHCG/'

print('################################')

print('Working Base :',Base, ' using ', sys.platform)

print('################################')

Base='E:/NIKHILESH/VKHCG/'

sFileName=Base + '/' + CompanyIn + '/00-RawData/' + InputFileName

print('Loading :',sFileName)

IP\_DATA\_ALL=pd.read\_csv(sFileName,header=0,low\_memory=False,usecols=['Country','Place.Name','Latitude','Longitude','First.IP.Number','Last.IP.Number'])

IP\_DATA\_ALL.rename(columns={'Place Name': 'Place\_Name'}, inplace=True)

IP\_DATA\_ALL.rename(columns={'First IP Number': 'First\_IP\_Number'}, inplace=True)

IP\_DATA\_ALL.rename(columns={'Last IP Number': 'Last\_IP\_Number'}, inplace=True)

sFileDir=Base + '/' + CompanyOut + '/01-Retrieve/01-EDS/02-Python'

if not os.path.exists(sFileDir):

os.makedirs(sFileDir)

visitordata = IP\_DATA\_ALL.drop\_duplicates(subset=None, keep='first', inplace=False)

visitordata10=visitordata.head(10)

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

print('Columns :',visitordata.shape[1])

print('Export CSV')

sFileName2=sFileDir + '/' + OutputFileName + '.csv'

visitordata.to\_csv(sFileName2, index = False)

print('Store All:',sFileName2)

sFileName3=sFileDir + '/' + OutputFileName + '\_10.csv'

visitordata10.to\_csv(sFileName3, index = False)

print('Store 10:',sFileName3)

for z in ['gzip', 'bz2', 'xz']:

if z == 'gzip':

sFileName4=sFileName2 + '.gz'

else:

sFileName4=sFileName2 + '.' + z

visitordata.to\_csv(sFileName4, index = False, compression=z)

print('Store :',sFileName4)

print('Export JSON')

for sOrient in ['split','records','index', 'columns','values','table']:

sFileName2=sFileDir + '/' + OutputFileName + '\_' + sOrient + '.json'

visitordata.to\_json(sFileName2,orient=sOrient,force\_ascii=True)

print('Store All:',sFileName2)

sFileName3=sFileDir + '/' + OutputFileName + '\_10\_' + sOrient + '.json'

visitordata10.to\_json(sFileName3,orient=sOrient,force\_ascii=True)

print('Store 10:',sFileName3)

sFileName4=sFileName2 + '.gz'

file\_in = open(sFileName2, 'rb')

file\_out = gz.open(sFileName4, 'wb')

file\_out.writelines(file\_in)

file\_in.close()

file\_out.close()

print('Store GZIP All:',sFileName4)

sFileName5=sFileDir + '/' + OutputFileName + '\_' + sOrient + '\_UnGZip.json'

file\_in = gz.open(sFileName4, 'rb')

file\_out = open(sFileName5, 'wb')

file\_out.writelines(file\_in)

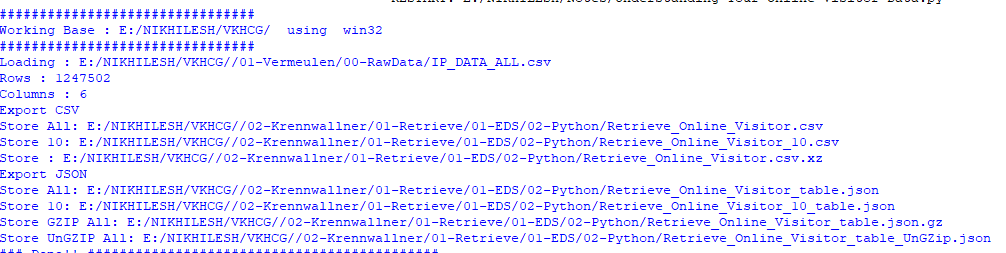
file\_in.close()

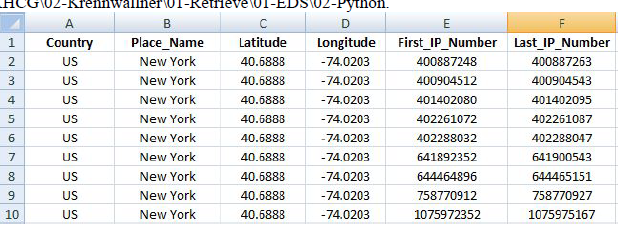
file\_out.close()

print('Store UnGZIP All:',sFileName5)

print('### Done!! ############################################')

output





XML processing

Code

import sys

import os

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)

InputFileName='IP\_DATA\_ALL.csv'

OutputFileName='Retrieve\_Online\_Visitor.xml'

CompanyIn= '01-Vermeulen'

CompanyOut= '02-Krennwallner'

if sys.platform == 'linux':

Base=os.path.expanduser('~') + '/VKHCG'

else:

Base='E:/NIKHILESH/VKHCG/'

print('Working Base :',Base, ' using ', sys.platform)

sFileName=Base + '/' + CompanyIn + '/00-RawData/' + InputFileName

print('Loading :',sFileName)

IP\_DATA\_ALL=pd.read\_csv(sFileName,header=0,low\_memory=False)

IP\_DATA\_ALL.rename(columns={'Place Name': 'Place\_Name'}, inplace=True)

IP\_DATA\_ALL.rename(columns={'First IP Number': 'First\_IP\_Number'}, inplace=True)

IP\_DATA\_ALL.rename(columns={'Last IP Number': 'Last\_IP\_Number'}, inplace=True)

IP\_DATA\_ALL.rename(columns={'Post Code': 'Post\_Code'}, inplace=True)

sFileDir=Base + '/' + CompanyOut + '/01-Retrieve/01-EDS/02-Python'

if not os.path.exists(sFileDir):

os.makedirs(sFileDir)

visitordata = IP\_DATA\_ALL.head(10000)

print('Original Subset Data Frame')

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

print('Columns :',visitordata.shape[1])

print(visitordata)

print('Export XML')

sXML=df2xml(visitordata)

sFileName=sFileDir + '/' + OutputFileName

file\_out = open(sFileName, 'wb')

file\_out.write(sXML)

file\_out.close()

print('Store XML:',sFileName)

xml\_data = open(sFileName).read()

unxmlrawdata=xml2df(xml\_data)

print('Raw XML Data Frame')

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

print('Columns :',unxmlrawdata.shape[1])

print(unxmlrawdata)

unxmldata = unxmlrawdata.drop\_duplicates(subset=None, keep='first', inplace=False)

print('Deduplicated XML Data Frame')

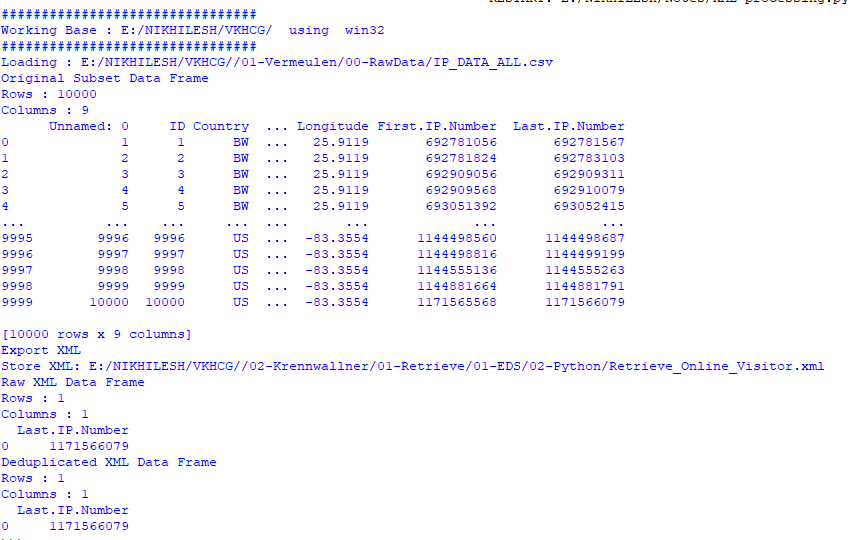
print('Rows :',unxmldata.shape[0])

print('Columns :',unxmldata.shape[1])

print(unxmldata)

#print('### Done!!#########')

Output



**Adopt New Shipping Containers**

Code

import sys

import os

import pandas as pd

ContainerFileName = 'Retrieve\_Container.csv'

BoxFileName = 'Retrieve\_Box.csv'

ProductFileName = 'Retrieve\_Product.csv'

Company = '03-Hillman'

Base = 'E:/NIKHILESH/10th .pdfVKHCG'

print('Working Base :', Base, ' using ', sys.platform)

sFileDir = Base + '/' + Company + '/01-Retrieve/01-EDS/02-Python'

if not os.path.exists(sFileDir):

os.makedirs(sFileDir)

containerLength = range(1, 21)

containerWidth = range(1, 10)

containerHeigth = range(1, 6)

containerStep = 1

c = 0

# Initialize an empty DataFrame for containers

ContainerFrame = pd.DataFrame()

for l in containerLength:

for w in containerWidth:

for h in containerHeigth:

containerVolume = (l / containerStep) \* (w / containerStep) \* (h / containerStep)

c += 1

ContainerLine = {

'ShipType': 'Container',

'UnitNumber': 'C' + format(c, "06d"),

'Length': round(l, 4),

'Width': round(w, 4),

'Height': round(h, 4),

'ContainerVolume': round(containerVolume, 6)

}

ContainerRow = pd.DataFrame([ContainerLine])

ContainerFrame = pd.concat([ContainerFrame, ContainerRow], ignore\_index=True)

ContainerFrame.index.name = 'IDNumber'

print('################')

print('## Container')

print('################')

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

print('Columns :', ContainerFrame.shape[1])

sFileContainerName = sFileDir + '/' + ContainerFileName

ContainerFrame.to\_csv(sFileContainerName, index=False)

boxLength = range(1, 21)

boxWidth = range(1, 21)

boxHeigth = range(1, 21)

packThick = range(0, 6)

boxStep = 10

b = 0

# Initialize an empty DataFrame for boxes

BoxFrame = pd.DataFrame()

for l in boxLength:

for w in boxWidth:

for h in boxHeigth:

for t in packThick:

boxVolume = round((l / boxStep) \* (w / boxStep) \* (h / boxStep), 6)

productVolume = round(((l - t) / boxStep) \* ((w - t) / boxStep) \* ((h - t) / boxStep), 6)

if productVolume > 0:

b += 1

BoxLine = {

'ShipType': 'Box',

'UnitNumber': 'B' + format(b, "06d"),

'Length': round(l / 10, 6),

'Width': round(w / 10, 6),

'Height': round(h / 10, 6),

'Thickness': round(t / 5, 6),

'BoxVolume': round(boxVolume, 9),

'ProductVolume': round(productVolume, 9)

}

BoxRow = pd.DataFrame([BoxLine])

BoxFrame = pd.concat([BoxFrame, BoxRow], ignore\_index=True)

BoxFrame.index.name = 'IDNumber'

print('## Box####')

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

print('Columns :', BoxFrame.shape[1])

sFileBoxName = sFileDir + '/' + BoxFileName

BoxFrame.to\_csv(sFileBoxName, index=False)

productLength = range(1, 21)

productWidth = range(1, 21)

productHeigth = range(1, 21)

productStep = 10

p = 0

# Initialize an empty DataFrame for products

ProductFrame = pd.DataFrame()

for l in productLength:

for w in productWidth:

for h in productHeigth:

productVolume = round((l / productStep) \* (w / productStep) \* (h / productStep), 6)

if productVolume > 0:

p += 1

ProductLine = {

'ShipType': 'Product',

'UnitNumber': 'P' + format(p, "06d"),

'Length': round(l / 10, 6),

'Width': round(w / 10, 6),

'Height': round(h / 10, 6),

'ProductVolume': round(productVolume, 9)

}

ProductRow = pd.DataFrame([ProductLine])

ProductFrame = pd.concat([ProductFrame, ProductRow], ignore\_index=True)

ProductFrame.index.name = 'IDNumber'

print('## Product')

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

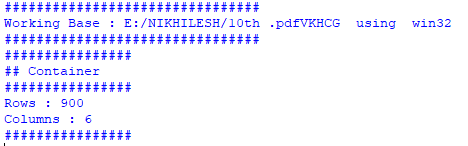
print('Columns :', ProductFrame.shape[1])

sFileProductName = sFileDir + '/' + ProductFileName

ProductFrame.to\_csv(sFileProductName, index=False)

print('### Done!! ##############')

output



Global Post Codes

Code in r studio

library(readr)

All\_Countries <- read\_delim("C:/VKHCG/03-Hillman/00-RawData/All\_Countries.txt",

"\t", col\_names = FALSE,

col\_types = cols(

X12 = col\_skip(),

X6 = col\_skip(),

X7 = col\_skip(),

X8 = col\_skip(),

X9 = col\_skip()),

na = "null", trim\_ws = TRUE)

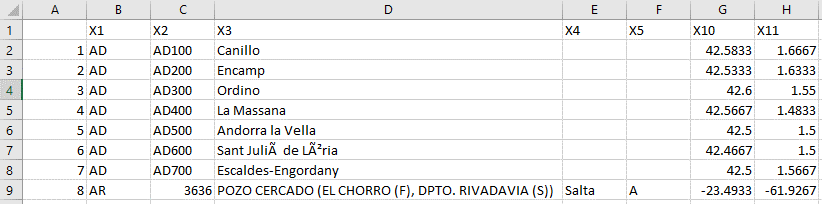
write.csv(All\_Countries,

file = "C:/VKHCG/03-Hillman/01-Retrieve/01-EDS/01-R/Retrieve\_All\_Countries.csv")

output

The program will successfully uploaded a new file named Retrieve\_All\_Countries.csv, after removing column

No. 6, 7, 8, 9 and 12 from All\_Countries.txt



**Program to connect to different data sources.**

**Code**

import sqlite3 as sq

import pandas as pd

Base='C:/VKHCG'

sDatabaseName=Base + '/01-Vermeulen/00-RawData/SQLite/vermeulen.db'

conn = sq.connect(sDatabaseName)

sFileName='C:/VKHCG/01-Vermeulen/01-Retrieve/01-EDS/02-Python/Retrieve\_IP\_DATA.csv'

print('Loading :',sFileName)

IP\_DATA\_ALL\_FIX=pd.read\_csv(sFileName,header=0,low\_memory=False)

IP\_DATA\_ALL\_FIX.index.names = ['RowIDCSV']

sTable='IP\_DATA\_ALL'

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

IP\_DATA\_ALL\_FIX.to\_sql(sTable, conn, if\_exists="replace")

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

TestData=pd.read\_sql\_query("select \* from IP\_DATA\_ALL;", conn)

print('## Data Values')

print(TestData)

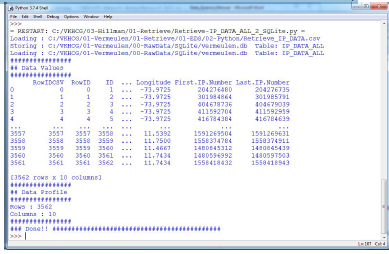
print('## Data Profile')

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

print('Columns :',TestData.shape[1])

print('### Done!! ###############')

output



**MySQL:**

Open MySql

Create a database “DataScience”

Create a python file and add the following code:

################ Connection With MySQL ######################

import mysql.connector

conn = mysql.connector.connect(host='localhost',

database='DataScience',

user='root',

password='root')

conn.connect

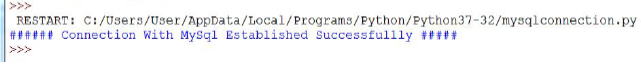
if(conn.is\_connected):

print('###### Connection With MySql Established Successfullly ##### ')

else:

print('Not Connected -- Check Connection Properites')

output



Microsoft Excel

Code

import os

import pandas as pd

################################################################

Base='E:/Nikhilesh/VKHCG'

################################################################

sFileDir=Base + '/01-Vermeulen/01-Retrieve/01-EDS/02-Python'

#if not os.path.exists(sFileDir):

#os.makedirs(sFileDir)

################################################################

CurrencyRawData = pd.read\_excel('E:/NIKHILESH/VKHCG/01-Vermeulen/00-RawData/Country\_Currency.xlsx')

sColumns = ['Country or territory', 'Currency', 'ISO-4217']

CurrencyData = CurrencyRawData[sColumns]

CurrencyData.rename(columns={'Country or territory': 'Country', 'ISO-4217':

'CurrencyCode'}, inplace=True)

CurrencyData.dropna(subset=['Currency'],inplace=True)

CurrencyData['Country'] = CurrencyData['Country'].map(lambda x: x.strip())

CurrencyData['Currency'] = CurrencyData['Currency'].map(lambda x:

x.strip())

CurrencyData['CurrencyCode'] = CurrencyData['CurrencyCode'].map(lambda x:

x.strip())

print(CurrencyData)

print('~~~~~~ Data from Excel Sheet Retrived Successfully ~~~~~~~ ')

sFileName=sFileDir + '/Retrieve-Country-Currency.csv'

CurrencyData.to\_csv(sFileName, index = False)

Output

