Practical 4

A. Perform the following data processing using R.

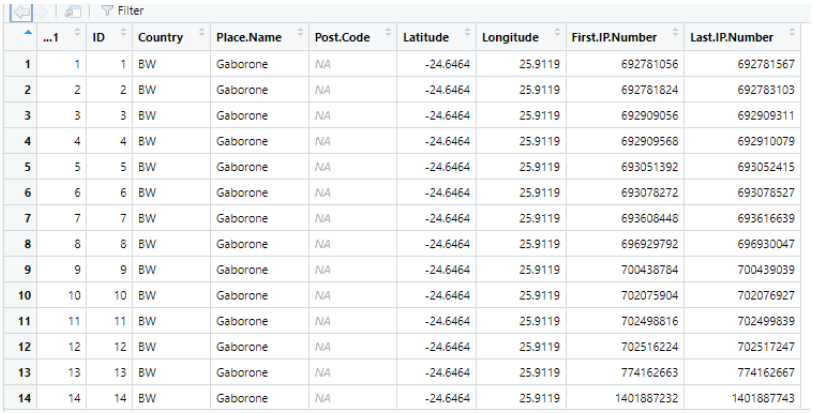
Code

```
library(readr)

IP_DATA_ALL <- read_csv("E:/NIKHILESH/VKHCG/01-Vermeulen/00-RawData/IP_DATA_ALL.csv")

View(IP_DATA_ALL)
```

Output



	...1	ID	Country	Place.Name	Post.Code	Latitude	Longitude	First.IP.Number	Last.IP.Number
1	1	1	BW	Gaborone	NA	-24.6464	25.9119	692781056	692781567
2	2	2	BW	Gaborone	NA	-24.6464	25.9119	692781824	692783103
3	3	3	BW	Gaborone	NA	-24.6464	25.9119	692909056	692909311
4	4	4	BW	Gaborone	NA	-24.6464	25.9119	692909568	692910079
5	5	5	BW	Gaborone	NA	-24.6464	25.9119	693051392	693052415
6	6	6	BW	Gaborone	NA	-24.6464	25.9119	693078272	693078527
7	7	7	BW	Gaborone	NA	-24.6464	25.9119	693608448	693616639
8	8	8	BW	Gaborone	NA	-24.6464	25.9119	696929792	696930047
9	9	9	BW	Gaborone	NA	-24.6464	25.9119	700438784	700439039
10	10	10	BW	Gaborone	NA	-24.6464	25.9119	702075904	702076927
11	11	11	BW	Gaborone	NA	-24.6464	25.9119	702498816	702499839
12	12	12	BW	Gaborone	NA	-24.6464	25.9119	702516224	702517247
13	13	13	BW	Gaborone	NA	-24.6464	25.9119	774162663	774162667
14	14	14	BW	Gaborone	NA	-24.6464	25.9119	1401887232	1401887743

spec(IP_DATA_ALL)

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

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

```
# A tibble: 1,247,502 x 9
    ...1    ID Country Place.Name Post.Code Latitude Longitude
   <dbl> <dbl> <chr>   <chr>      <chr>        <dbl>     <dbl>
 1     1     1 BW      Gaborone   NA           -24.6      25.9
 2     2     2 BW      Gaborone   NA           -24.6      25.9
 3     3     3 BW      Gaborone   NA           -24.6      25.9
 4     4     4 BW      Gaborone   NA           -24.6      25.9
 5     5     5 BW      Gaborone   NA           -24.6      25.9
 6     6     6 BW      Gaborone   NA           -24.6      25.9
 7     7     7 BW      Gaborone   NA           -24.6      25.9
 8     8     8 BW      Gaborone   NA           -24.6      25.9
 9     9     9 BW      Gaborone   NA           -24.6      25.9
10    10    10 BW      Gaborone   NA           -24.6      25.9
# i 1,247,492 more rows
# i 2 more variables: First.IP.Number <dbl>, Last.IP.Number <dbl>
# i Use `print(n = ...)` to see more rows
```

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)

```
sapply(IP_DATA_ALL_FIX, typeof)
          X          ID     Country  Place.Name   Post.Code    Latitude   Longitude First.IP.Number Last.IP.Number
  "integer"  "integer"  "character" "character" "character"  "double"   "double"    "integer"       "integer"
```

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)

	Country	N
1	AD	46
2	AE	1793
3	AF	15
4	AG	21
5	AI	9
6	AL	91

sapply(IP_DATA_ALL_FIX[,'Latitude'], min, na.rm=TRUE)

```
[1] -24.6464 -24.6464 -24.6464 -24.6464 -24.6464 -24.6464 -24.6464
[8] -24.6464 -24.6464 -24.6464 -24.6464 -24.6464 -24.6464 -24.6464
15] -24.6464  13.5167  13.5167  13.5167  13.5167  13.5167  13.5167
22]  13.5167 -25.9653 -25.9653 -25.9653 -25.9653 -25.9653 -25.9653
29] -25.9653 -25.9653 -25.9653 -25.9653 -25.9653 -25.9653 -25.9653
36] -25.9653 -25.9653 -25.9653   5.6167   5.6167   5.6167   5.6167
43]   5.6167   6.6833   6.6833   6.6833   6.6833   6.6833   6.6833
```

sapply(IP_DATA_ALL_FIX[,'Country'], min, na.rm=TRUE)

```
  BW    BW    BW    BW    BW    BW    BW    BW    BW    BW    BW    BW    BW    BW
'BW'  'BW'  'BW'  'BW'  'BW'  'BW'  'BW'  'BW'  'BW'  'BW'  'BW'  'BW'  'BW'  'BW'
  BW    NE    NE    NE    NE    NE    NE    NE    MZ    MZ    MZ    MZ    MZ    MZ
'BW'  'NE'  'NE'  'NE'  'NE'  'NE'  'NE'  'NE'  'MZ'  'MZ'  'MZ'  'MZ'  'MZ'  'MZ'
  MZ    MZ    MZ    MZ    MZ    MZ    MZ    MZ    MZ    MZ    GH    GH    GH    GH
'MZ'  'MZ'  'MZ'  'MZ'  'MZ'  'MZ'  'MZ'  'MZ'  'MZ'  'MZ'  'GH'  'GH'  'GH'  'GH'
```

sapply(IP_DATA_ALL_FIX[,'Latitude'], max, na.rm=TRUE)

```
 [1] -24.6464 -24.6464 -24.6464 -24.6464 -24.6464 -24.6464 -24.6464
 [8] -24.6464 -24.6464 -24.6464 -24.6464 -24.6464 -24.6464 -24.6464
[15] -24.6464  13.5167  13.5167  13.5167  13.5167  13.5167  13.5167
[22]  13.5167 -25.9653 -25.9653 -25.9653 -25.9653 -25.9653 -25.9653
[29] -25.9653 -25.9653 -25.9653 -25.9653 -25.9653 -25.9653 -25.9653
[36] -25.9653 -25.9653 -25.9653   5.6167   5.6167   5.6167   5.6167
```

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-

```
========== RESTART: E:/NIKHILESH/Notes/DS Pract/pract4b.py ==========
Loading : E:/NIKHILESH/VKHCG/01-Vermeulen/00-RawData/IP_DATA_ALL.csv
Rows: 1247502
Columns: 9
### Raw Data Set #################################
Unnamed: 0 <class 'str'>
ID <class 'str'>
Country <class 'str'>
Place.Name <class 'str'>
Post.Code <class 'str'>
Latitude <class 'str'>
Longitude <class 'str'>
First.IP.Number <class 'str'>
Last.IP.Number <class 'str'>
### Fixed Data Set ################################
Last.IP.Number <class 'str'>
Fixed Data Set with ID
### Done!! ########################################
>>>
```

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

	Country	PatternCountry
1	BW	AA
2	NE	AA
3	MZ	AA
4	GH	AA
5	DZ	AA

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-

	A	B	C	D	E	F	G	H	I	J
1	Country_from	Place_Name_from	Latitude_from	Longitude_from	Country_to	Place_Name_to	Latitude_to	Longitude_to	DistanceBetweenKilometers	DistanceBetweenMiles
2	US	New York	40.7528	-73.9725	US	New York	40.7528	-73.9725	0	0
3	US	New York	40.7528	-73.9725	US	New York	40.7214	-74.0052	4.448	2.762
4	US	New York	40.7528	-73.9725	US	New York	40.7662	-73.9862	1.885	1.17
5	US	New York	40.7528	-73.9725	US	New York	40.7449	-73.9782	1.001	0.622
6	US	New York	40.7528	-73.9725	US	New York	40.7605	-73.9933	1.95	1.211
7	US	New York	40.7528	-73.9725	US	New York	40.7588	-73.968	0.767	0.476
8	US	New York	40.7528	-73.9725	US	New York	40.7637	-73.9727	1.212	0.753
9	US	New York	40.7528	-73.9725	US	New York	40.7553	-73.9924	1.699	1.055
10	US	New York	40.7528	-73.9725	US	New York	40.7308	-73.9975	3.228	2.004

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

```
========== RESTART
Loading : E:/NIKHILESH/VKHCG/01-Vermeulen/00-RawData/IP_DATA_CORE.csv
Rows : 150
Columns : 4
### Done!! ############################################
```

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

```
################################
Working Base : E:/NIKHILESH/VKHCG/  using  win32
################################
Loading : E:/NIKHILESH/VKHCG//01-Vermeulen/00-RawData/IP_DATA_ALL.csv
Rows : 1247502
Columns : 6
Export CSV
Store All: E:/NIKHILESH/VKHCG//02-Krennwallner/01-Retrieve/01-EDS/02-Python/Retrieve_Online_Visitor.csv
Store 10: E:/NIKHILESH/VKHCG//02-Krennwallner/01-Retrieve/01-EDS/02-Python/Retrieve_Online_Visitor_10.csv
Store : E:/NIKHILESH/VKHCG//02-Krennwallner/01-Retrieve/01-EDS/02-Python/Retrieve_Online_Visitor.csv.xz
Export JSON
Store All: E:/NIKHILESH/VKHCG//02-Krennwallner/01-Retrieve/01-EDS/02-Python/Retrieve_Online_Visitor_table.json
Store 10: E:/NIKHILESH/VKHCG//02-Krennwallner/01-Retrieve/01-EDS/02-Python/Retrieve_Online_Visitor_10_table.json
Store GZIP All: E:/NIKHILESH/VKHCG//02-Krennwallner/01-Retrieve/01-EDS/02-Python/Retrieve_Online_Visitor_table.json.gz
Store UnGZIP All: E:/NIKHILESH/VKHCG//02-Krennwallner/01-Retrieve/01-EDS/02-Python/Retrieve_Online_Visitor_table_UnGZip.json
```

HCG\02-Krennwallner\01-Retrieve\01-EDS\02-Python.

	A	B	C	D	E	F
1	Country	Place_Name	Latitude	Longitude	First_IP_Number	Last_IP_Number
2	US	New York	40.6888	-74.0203	400887248	400887263
3	US	New York	40.6888	-74.0203	400904512	400904543
4	US	New York	40.6888	-74.0203	401402080	401402095
5	US	New York	40.6888	-74.0203	402261072	402261087
6	US	New York	40.6888	-74.0203	402288032	402288047
7	US	New York	40.6888	-74.0203	641892352	641900543
8	US	New York	40.6888	-74.0203	644464896	644465151
9	US	New York	40.6888	-74.0203	758770912	758770927
10	US	New York	40.6888	-74.0203	1075972352	1075975167

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

```
################################
Working Base : E:/NIKHILESH/VKHCG/  using  win32
################################
Loading : E:/NIKHILESH/VKHCG//01-Vermeulen/00-RawData/IP_DATA_ALL.csv
Original Subset Data Frame
Rows : 10000
Columns : 9
      Unnamed: 0     ID Country  ... Longitude First.IP.Number  Last.IP.Number
0              1      1      BW  ...   25.9119       692781056       692781567
1              2      2      BW  ...   25.9119       692781824       692783103
2              3      3      BW  ...   25.9119       692909056       692909311
3              4      4      BW  ...   25.9119       692909568       692910079
4              5      5      BW  ...   25.9119       693051392       693052415
...          ...    ...     ...  ...       ...             ...             ...
9995        9996   9996      US  ...  -83.3554      1144498560      1144498687
9996        9997   9997      US  ...  -83.3554      1144498816      1144499199
9997        9998   9998      US  ...  -83.3554      1144555136      1144555263
9998        9999   9999      US  ...  -83.3554      1144881664      1144881791
9999       10000  10000      US  ...  -83.3554      1171565568      1171566079

[10000 rows x 9 columns]
Export XML
Store XML: E:/NIKHILESH/VKHCG//02-Krennwallner/01-Retrieve/01-EDS/02-Python/Retrieve_Online_Visitor.xml
Raw XML Data Frame
Rows : 1
Columns : 1
  Last.IP.Number
0     1171566079
Deduplicated XML Data Frame
Rows : 1
Columns : 1
  Last.IP.Number
0     1171566079
```

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

```
################################
Working Base : E:/NIKHILESH/10th .pdfVKHCG  using  win32
################################
################
## Container
################
Rows : 900
Columns : 6
################
```

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

	A	B	C	D	E	F	G	H
1		X1	X2	X3	X4	X5	X10	X11
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 JuliÃ de LÃ²ria			42.4667	1.5
8	7	AD	AD700	Escaldes-Engordany			42.5	1.5667
9	8	AR	3636	POZO CERCADO (EL CHORRO (F), DPTO. RIVADAVIA (S))	Salta	A	-23.4933	-61.9267

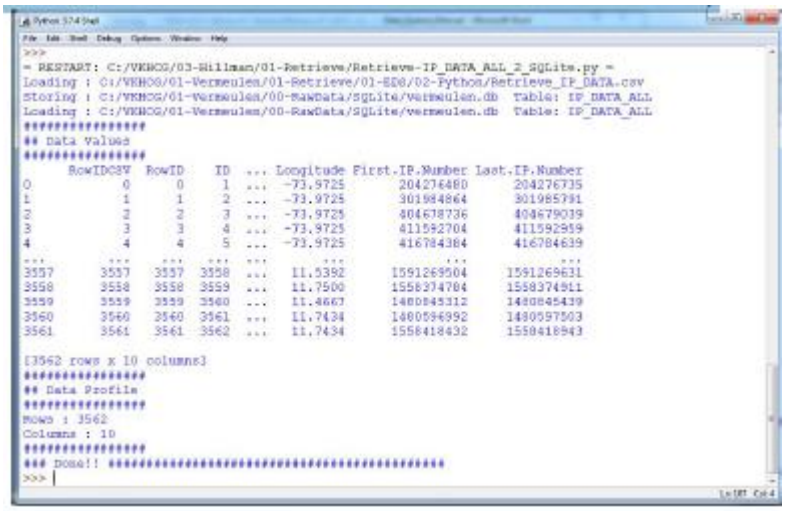
Program to connect to different data sources.

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

```
>>>
 RESTART: C:/Users/User/AppData/Local/Programs/Python/Python37-32/mysqlconnection.py
###### Connection With MySql Established Successfullly #####
>>>
```

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

```
Type "help", "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:/VKHCG/04-Clark/01-Retrieve/Retrieve-Country-Currency.py ====
                        Country              Currency CurrencyCode
1                   Afghanistan        Afghan afghani          AFN
2    Akrotiri and Dhekelia (UK)         European euro          EUR
3       Aland Islands (Finland)         European euro          EUR
4                       Albania          Albanian lek          ALL
5                       Algeria        Algerian dinar          DZD
..                          ...                   ...          ...
271           Wake Island (USA)  United States dollar          USD
272  Wallis and Futuna (France)             CFP franc          XPF
274                       Yemen          Yemeni rial          YER
276                      Zambia        Zambian kwacha          ZMW
277                    Zimbabwe  United States dollar          USD

[253 rows x 3 columns]
~~~~~~ Data from Excel Sheet Retrived Successfully ~~~~~~~
>>>
```