INDEX

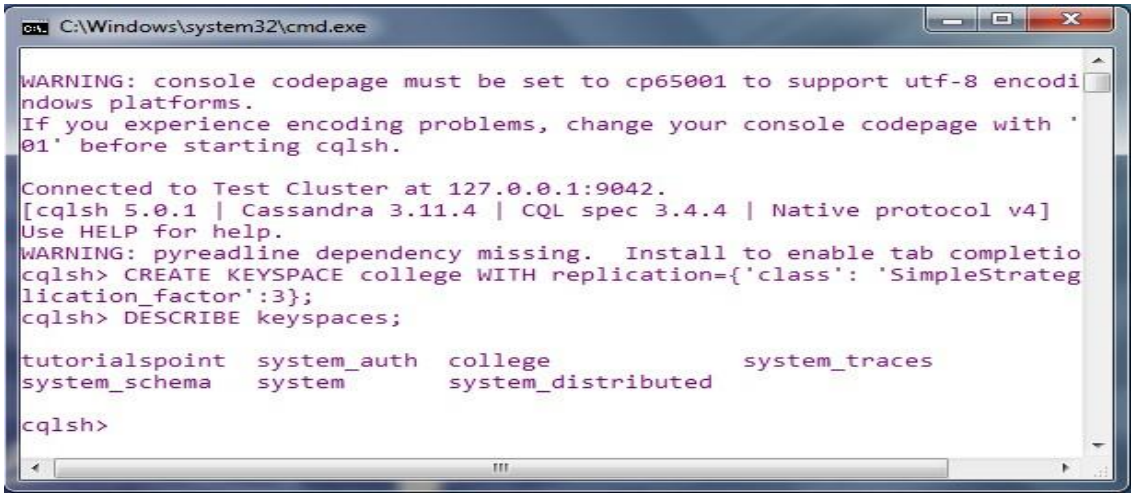
Practical No. 1

1. Create data model using Cassandra. Required Software:
Java v1.8, Python v2.7, Cassandra File

ANS:

Create Key space:

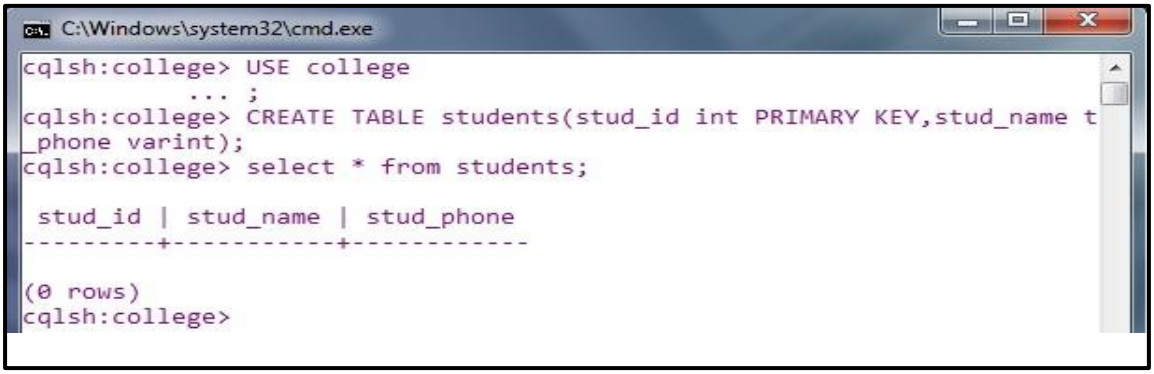
cqlsh> CREATE KEYSPACE college WITH replication = {'class':'SimpleStrategy', 'replication_factor' : 3};



Create Table

cqlsh:college> CREATE TABLE students(stud_id int PRIMARY KEY,stud_name text,stud_phone varint);

cqlsh:college> select * from students;



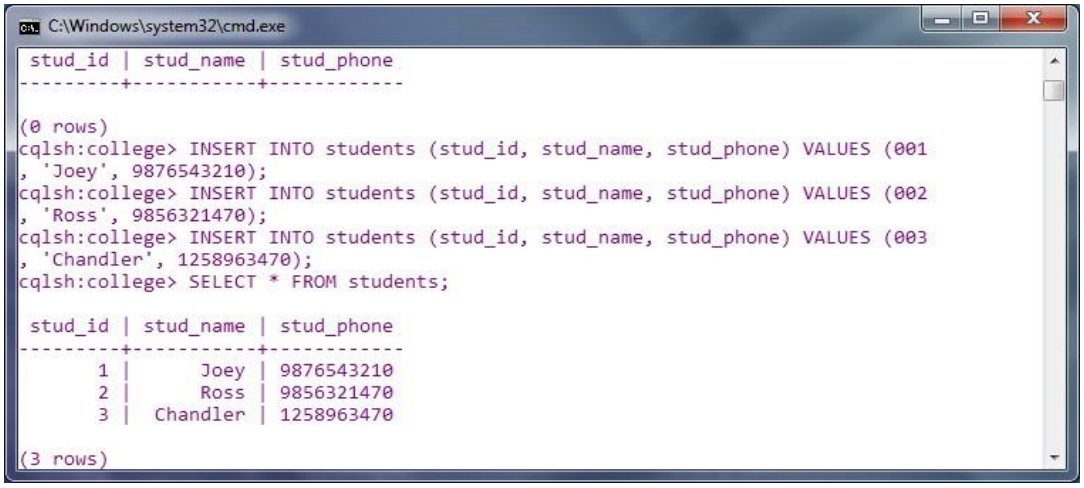
Creating Data in a Table

cqlsh:college> INSERT INTO students (stud_id, stud_name, stud_phone) VALUES (001, 'Joey', 9876543210);

cqlsh:college> INSERT INTO students (stud_id, stud_name, stud_phone) VALUES (002, 'Ross', 9856321470);

cqlsh:college> INSERT INTO students (stud_id, stud_name, stud_phone) VALUES (003, 'Chandler', 1258963470);

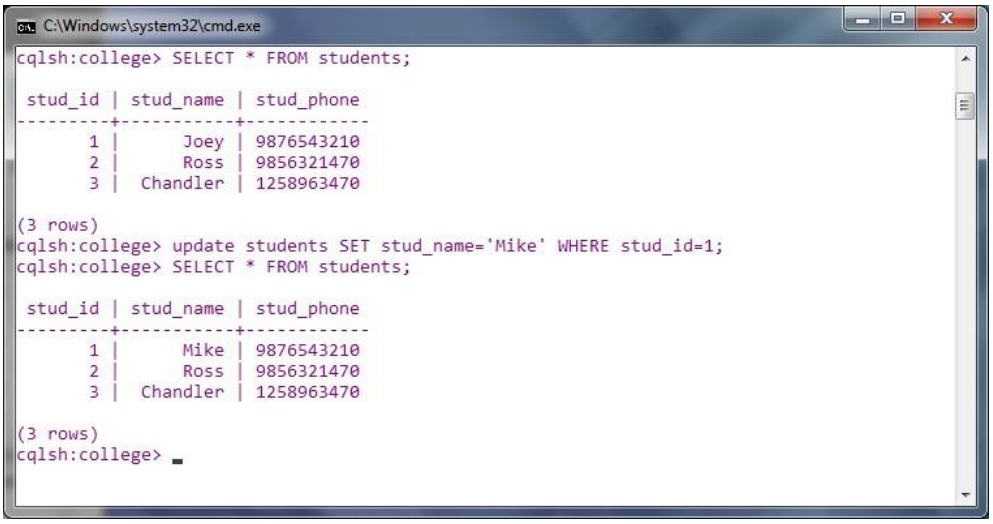
cqlsh:college> SELECT * FROM students;



Update data in table

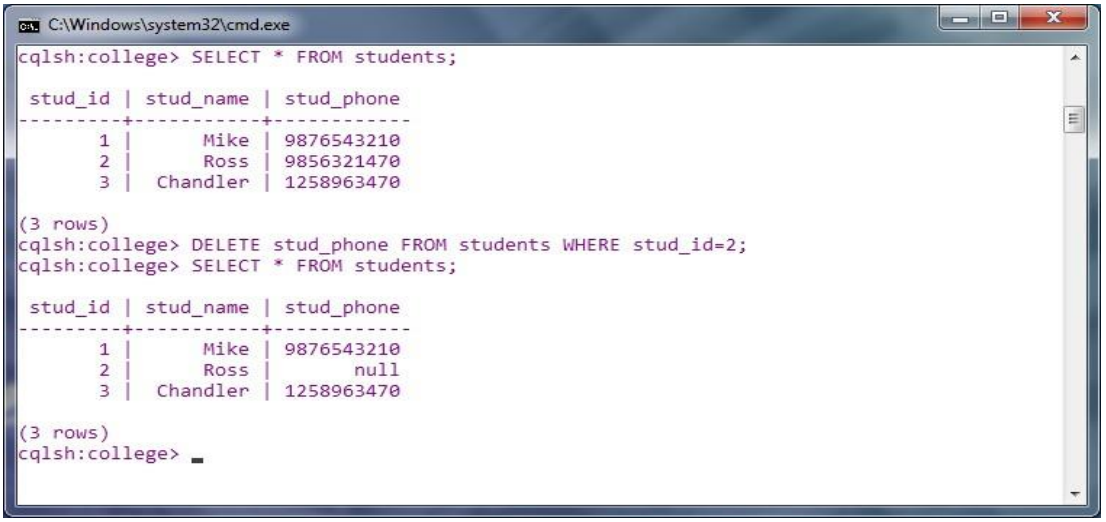
---

cqlsh:college> update students SET stud_name='Mike' WHERE stud_id=1; cqlsh:college> SELECT * FROM students;



Delete data from table

cqlsh:college> DELETE stud_phone FROM students WHERE stud_id=2; cqlsh:college> SELECT * FROM students;

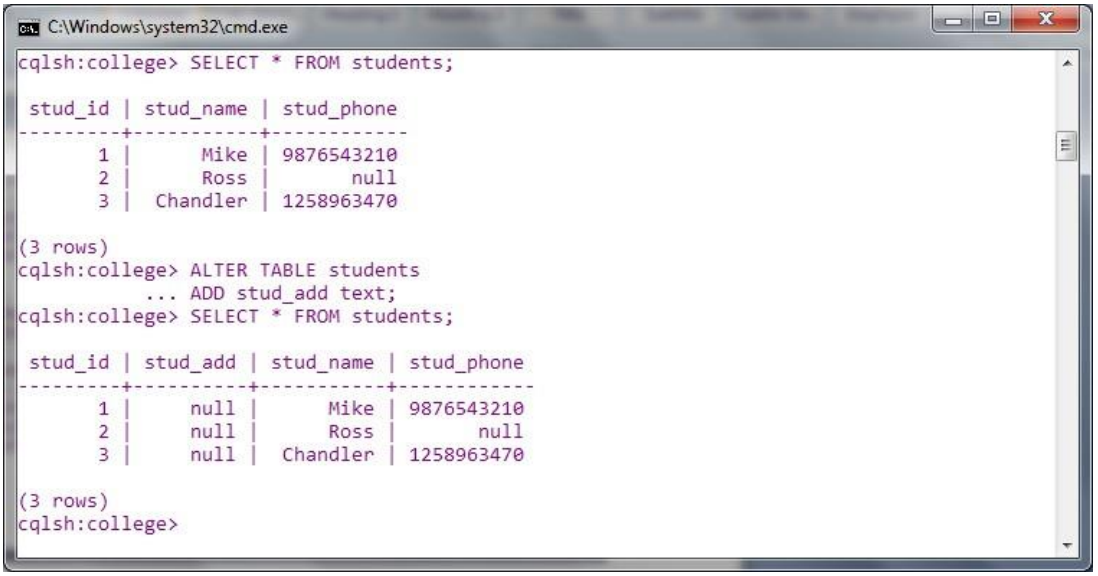


Alter Table

1. **Adding a column**

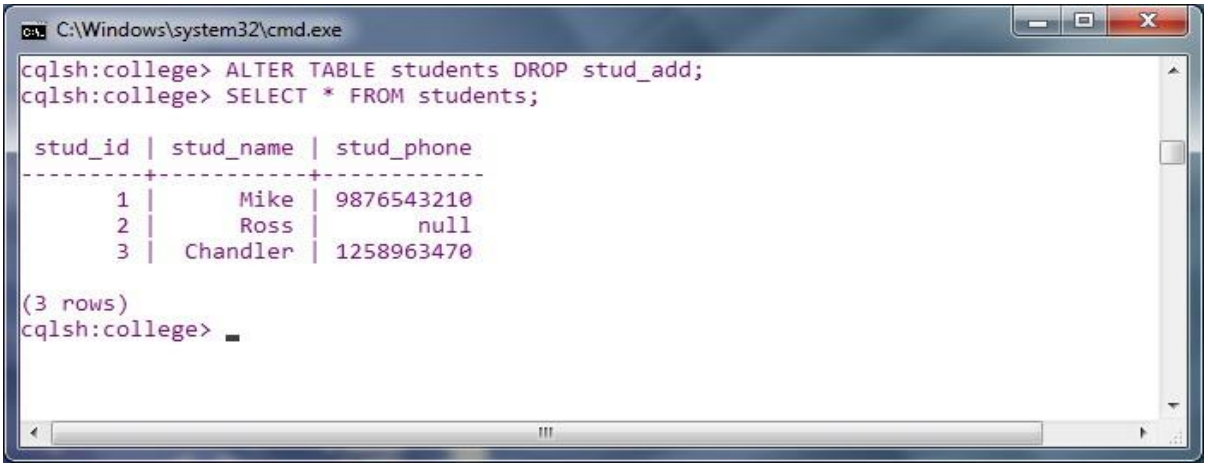
cqlsh:college> ALTER TABLE students

... ADD stud_add text; cqlsh:college> SELECT * FROM students



2. Dropping a column

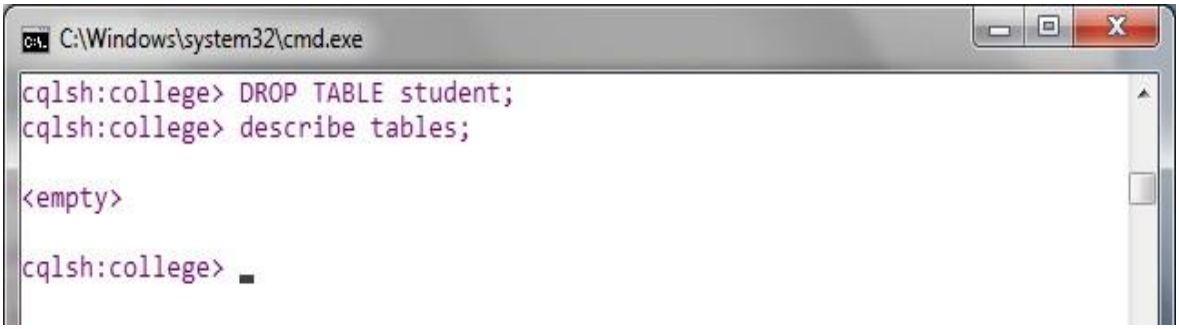
cqlsh:college> ALTER TABLE students DROP stud_add;
cqlsh:college> SELECT * FROM students;



Drop Table

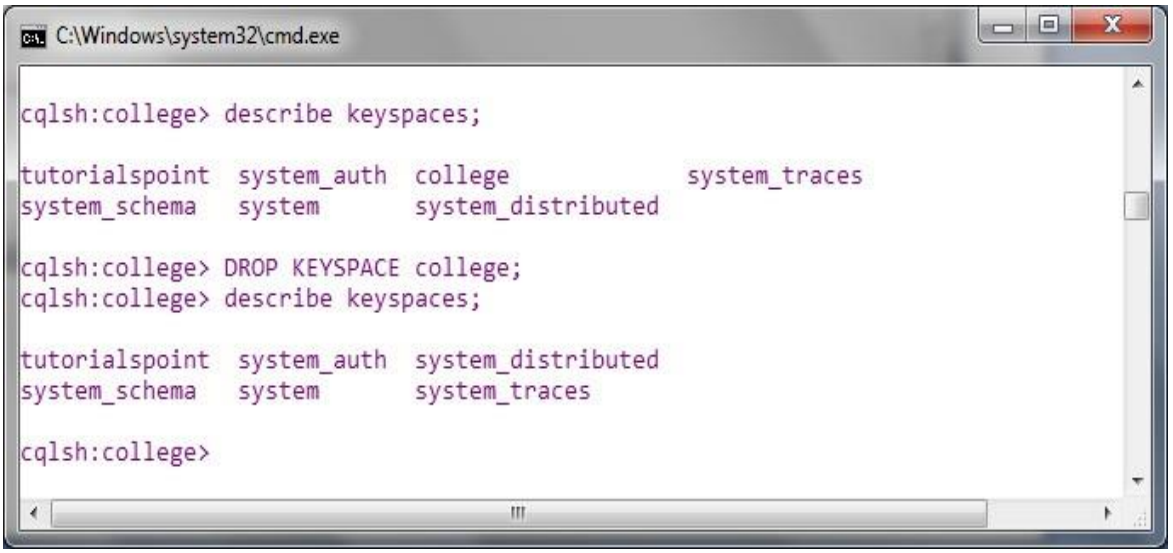
cqlsh:college> DROP TABLE student; cqlsh:college>

describe tables;



Drop Keyspace

cqlsh:college> DROP KEYSPACE college; cqlsh:college>
describe keyspaces;



Practical No. 2

Conversion from different formats to HORUS format.

CSV to HORUS

INPUT:

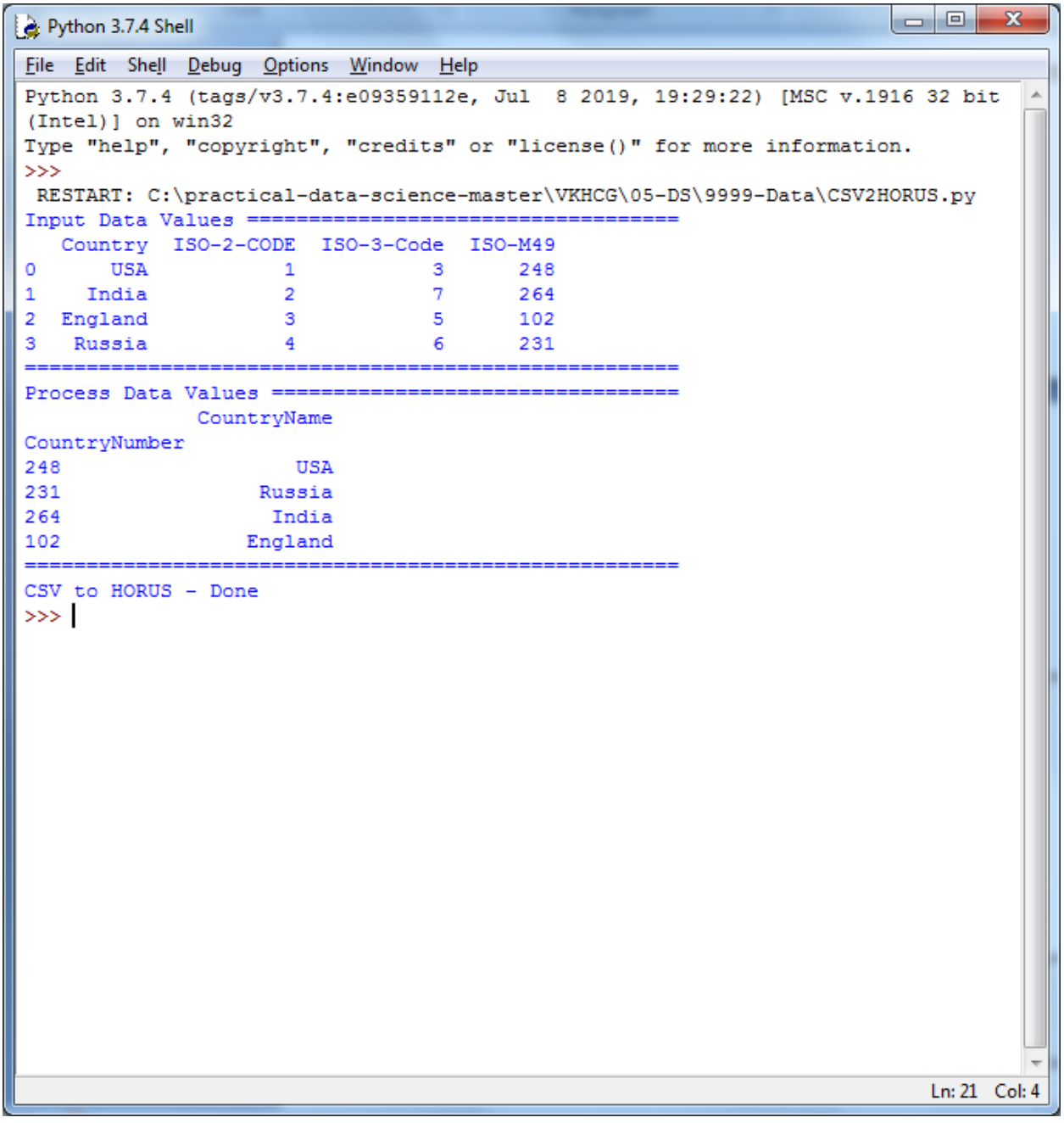
```
import pandas as pd
from datetime import datetime
sInputFileName='C:/practical-data-science-master/VKHCG/05-DS/9999-
Data/Country_Code.csv'
InputData=pd.read_csv(sInputFileName,encoding="latin-1")
print('Input Data Values ===================================')
print(InputData)
ProcessData=InputData

# Remove columns ISO-2-Code and ISO-3-CODE
ProcessData.drop('ISO-2-CODE', axis=1,inplace=True)
ProcessData.drop('ISO-3-Code', axis=1,inplace=True)

now=datetime.now() print("now
= ",now)
dt_string=now.strftime("%d/%m/%y %H:%M:%S")
print("Date and Time= ",dt_string)
f=open('C:/practical-data-science-master/VKHCG/05-DS/9999-
Data/Country_Code_Log.txt',"a")
f.write("Delete column activity recorded at ")
f.write(dt_string)
f.close()
# Rename Country and ISO-M49 ProcessData.rename(columns={'Country':
'CountryName'}, inplace=True)
ProcessData.rename(columns={'ISO-M49': 'CountryNumber'}, inplace=True) # Set
new Index
ProcessData.set_index('CountryNumber', inplace=True) #
Sort data by CurrencyNumber
ProcessData.sort_values('CountryName', axis=0, ascending=False,
inplace=True)
print('Process Data Values ===================================') print(ProcessData)
OutputData=ProcessData
```

```
sOutputFileName='C:/practical-data-science-master/VKHCG/05-DS/9999-
Data/HORUS-CSV-Country.csv'
OutputData.to_csv(sOutputFileName, index = False)
print('CSV to HORUS - Done')
```

OUTPUT:



CSV TO AUDIO

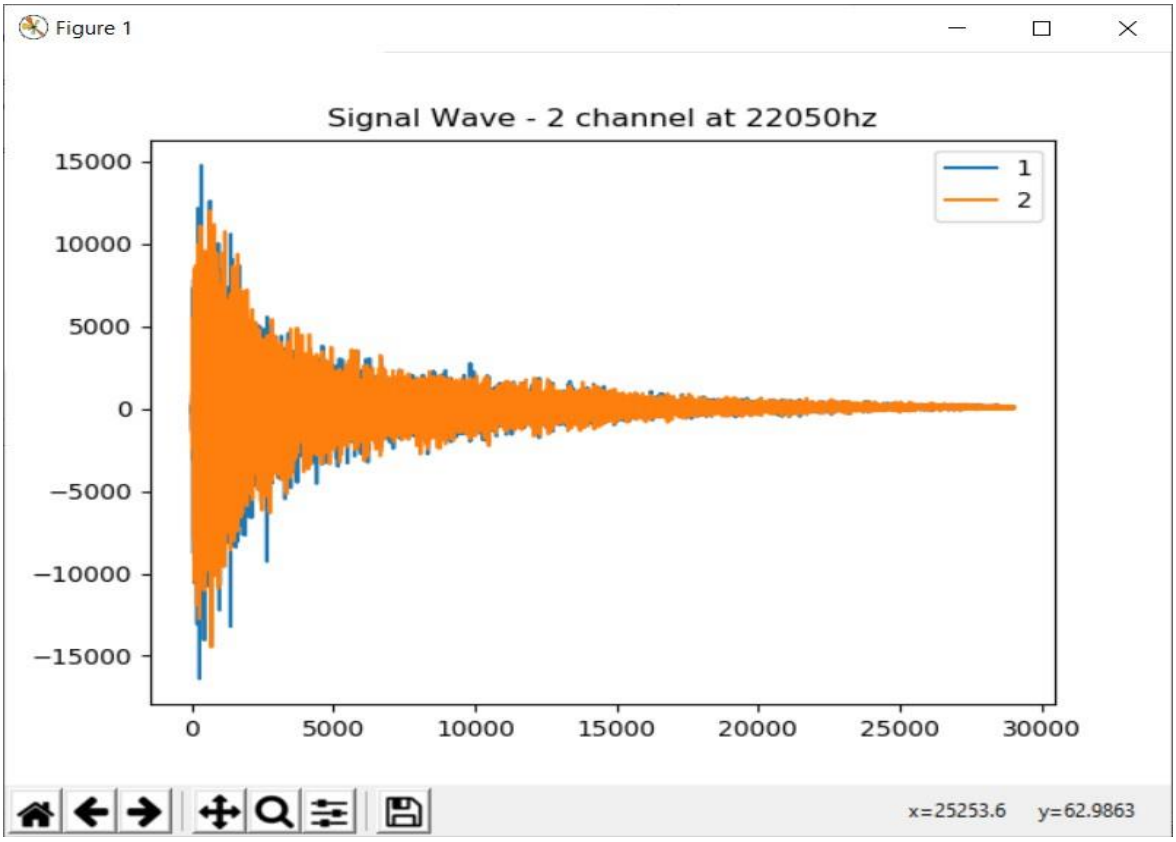
INPUT:

```
from scipy.io import wavfile
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
def show_info(aname, a,r):
   print ('----------')
   print ("Audio:", aname)
   print ('----------')
   print ("Rate:", r) print
   ('        ----------')
   print ("shape:", a.shape)
   print ("dtype:", a.dtype)
   print ("min, max:", a.min(), a.max())
   print ('----------')
   plot_info(aname, a,r)
def plot_info(aname, a,r):
   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:/Downloads/practical-data-science-master/VKHCG/05-
DS/9999-Data/2ch-sound.wav'
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:/Downloads/practical-data-science-master/VKHCG/05-
DS/9999-Data/HORUS-Audio-2ch.csv' OutputData.to_csv(sOutputFileName, index
= False) sInputFileName='D:/Downloads/practical-data-science-
master/VKHCG/05- DS/9999-Data/4ch-sound.wav'
```

```
 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:/Downloads/practical-data-science-master/VKHCG/05-
DS/9999-Data/HORUS-Audio-4ch.csv' OutputData.to_csv(sOutputFileName, index
= False) sInputFileName='D:/Downloads/practical-data-science-
master/VKHCG/05- DS/9999-Data/6ch-sound.wav'
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:/Downloads/practical-data-science-master/VKHCG/05-
DS/9999-Data/HORUS-Audio-6ch.csv' OutputData.to_csv(sOutputFileName, index
= False) sInputFileName='D:/Downloads/practical-data-science-
master/VKHCG/05- DS/9999-Data/8ch-sound.wav'
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:/Downloads/practical-data-science-master/VKHCG/05-
DS/9999-Data/HORUS-Audio-8ch.csv' OutputData.to_csv(sOutputFileName, index
= False)
print('Audio to HORUS - Done')
```

OUTPUT:

```
*Python 3.7.4 Shell*
File Edit Shell Debug Options Window Help
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
 RESTART: D:\Downloads\practical-data-science-master\VKHCG\05-DS\9999-Data\AUDIO
2HORUS.py
========================================================
Processing :  D:/Downloads/practical-data-science-master/VKHCG/05-DS/9999-Data/2
ch-sound.wav
========================================================
----------------
Audio: 2 channel
----------------
Rate: 22050
----------------
shape: (29016, 2)
dtype: int16
min, max: -16384 14767
----------------
```



Practical No 3

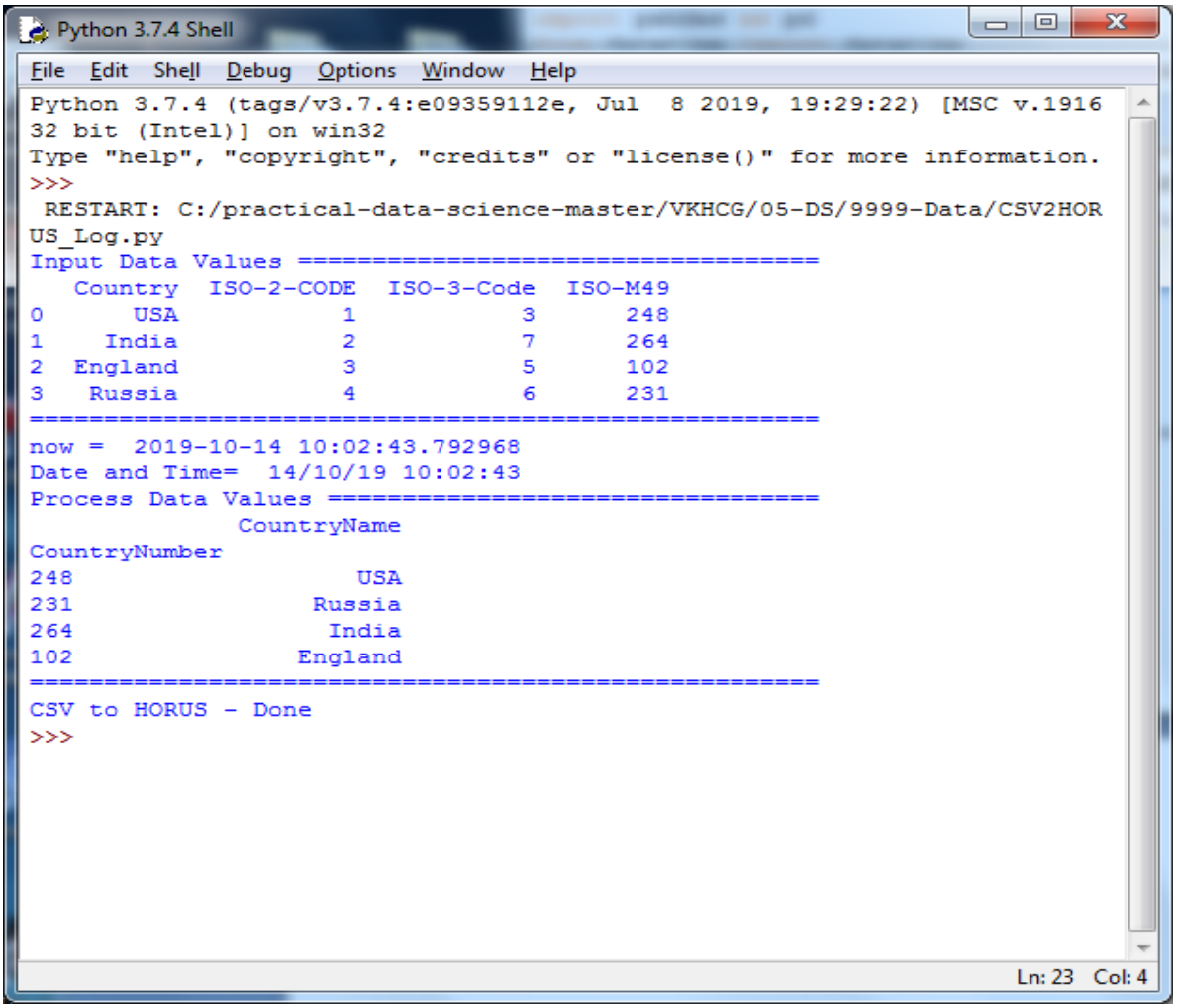
Auditing through Logging

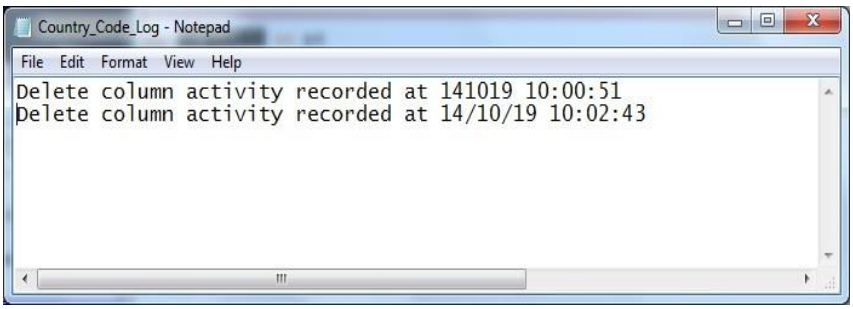
CSV to HORUS log file INPUT:

```
import pandas as pd
from datetime import datetime
sInputFileName='C:/practical-data-science-master/VKHCG/05-DS/9999-
Data/Country_Code.csv'
InputData=pd.read_csv(sInputFileName,encoding="latin-1")
print('Input Data Values ===================================')
print(InputData)
ProcessData=InputData
# Remove columns ISO-2-Code and ISO-3-CODE
ProcessData.drop('ISO-2-CODE', axis=1,inplace=True)
ProcessData.drop('ISO-3-Code', axis=1,inplace=True)
now=datetime.now()
print("now = ",now) dt_string=now.strftime("%d/%m/%y
%H:%M:%S") print("Date and Time= ",dt_string)
f=open('C:/practical-data-science-master/VKHCG/05-DS/9999-
Data/Country_Code_Log.txt',"a")
f.write("Delete column activity recorded at ")
f.write(dt_string)
f.close()
# Rename Country and ISO-M49 ProcessData.rename(columns={'Country':
'CountryName'}, inplace=True)
ProcessData.rename(columns={'ISO-M49': 'CountryNumber'}, inplace=True) # Set
new Index
ProcessData.set_index('CountryNumber', inplace=True)
# Sort data by CurrencyNumber
ProcessData.sort_values('CountryName', axis=0, ascending=False,
inplace=True)
print('Process Data Values ===================================') print(ProcessData)
OutputData=ProcessData
```

```
sOutputFileName='C:/practical-data-science-master/VKHCG/05-DS/9999-
Data/HORUS-CSV-Country.csv'
OutputData.to_csv(sOutputFileName, index = False)
print('CSV to HORUS - Done')
```

OUTPUT:





Practical No 04

Retrieve Superstep

Csv to Db

DATASET.PY

```
import sys
import os
import sqlite3 as sq
import pandas as pd
sDatabaseName='C:\Users\Acer\Desktop\sqlite-tools-win32-x86-
3340100\sqlite-tools-win32-x86-3340100\srk.db'
conn = sq.connect(sDatabaseName)
#################################################################
sFileName='C:\Users\Acer\Desktop\DataSet.csv'
print('Loading :',sFileName)
data=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin- 1")
data.index.names = ['RowIDCSV']
sTable='DataSet'
print('Storing :',sDatabaseName,' Table:',sTable)
data.to_sql(sTable, conn, if_exists="replace")
print('Loading :',sDatabaseName,' Table:',sTable)
TestData=pd.read_sql_query("select * from DataSet;", conn) print('################')
print('## Data Values')
print('################')
print(TestData)
print('################')
print('## Data Profile')
print('################')
print('Rows :',TestData.shape[0])
print('Columns :',TestData.shape[1])
print('################')
################################################################# print('###
Done!! ##############################################')
```

OUTPUT:

```
('Loading :', 'C:\\Users\\Acer\\Desktop\\sqlite-tools-win32-x86-3340100\\sq
tools-win32-x86-3340100\\srk.db', ' Table:', 'DataSet')
################
## Data Values
################
```

	RowIDCSV	Unnamed: 0	Unnamed: 0.1	...	tempo	valence	year
0	0	0	0	...	149.976	0.6340	1920.0
1	1	1	1	...	86.889	0.9500	1920.0
2	2	2	2	...	97.600	0.6890	1920.0
3	3	3	3	...	127.997	0.0422	1920.0
4	4	4	4	...	122.076	0.2990	1920.0
5	5	5	5	...	103.870	0.4770	1920.0
6	6	6	6	...	85.652	0.4870	1920.0
7	7	7	7	...	78.784	0.3660	1920.0
8	8	8	8	...	130.060	0.6210	1920.0
9	9	9	9	...	126.993	0.1190	1920.0
10	10	10	10	...	82.024	0.4140	1920.0
11	11	11	11	...	131.494	0.7030	1920.0
12	12	12	12	...	111.268	0.5660	1920.0
13	13	13	13	...	124.018	0.6390	1920.0
14	14	14	14	...	67.271	0.8940	1920.0
15	15	15	15	...	102.578	0.7920	1920.0
16	16	16	16	...	118.562	0.1460	1920.0
17	17	17	17	...	136.573	0.7280	1920.0
18	18	18	18	...	112.817	0.4930	1920.0
19	19	19	19	...	65.485	0.5150	1920.0
20	20	20	20	...	140.011	0.5780	1920.0

```
C:\Users\Acer\Desktop\sqlite-tools-win32-x86-3340100\sqlite-tools-win32-x86-3340100>sqlite3 srk.db
SQLite version 3.34.1 2021-01-20 14:10:07
Enter ".help" for usage hints.
sqlite> .schema
CREATE TABLE kkr(team_no integer,name string);
CREATE TABLE IF NOT EXISTS "prac" (
"RowIDCSV" INTEGER,
  "acousticness" REAL,
  "artists" TEXT,
  "danceability" REAL,
  "duration_ms" INTEGER,
  "energy" REAL,
  "explicit" INTEGER,
  "id" TEXT,
  "instrumentalness" REAL,
  "key" INTEGER,
  "liveness" REAL,
  "loudness" REAL,
  "mode" INTEGER,
  "name" TEXT,
  "popularity" INTEGER,
  "release_date" TEXT,
  "speechiness" REAL,
  "tempo" REAL,
  "valence" REAL,
  "year" INTEGER
```

4B. PERFORMING OPERATIONS ON DATASET

DATASETFIX.PY

```
import sys
import os
import pandas as pd
import sqlite3 as sq
#####################################################################
#####################################################################
sDatabaseName='C:/Users/Acer/Desktop/sqlite-tools-win32-x86-
3340100/sqlite-tools-win32-x86-3340100/srk.db'
conn = sq.connect(sDatabaseName)
print('Loading :',sDatabaseName)
DataSet=pd.read_sql_query("select * from DataSet;", conn)
#####################################################################
print('Rows:', DataSet.shape[0])
print('Columns:', DataSet.shape[1])
print('### Raw Data Set ##################################')
for i in range(0,len(DataSet.columns)):
  print(DataSet.columns[i],type(DataSet.columns[i]))
print('### Fixed Data Set ################################')
DataSet_FIX=DataSet
for i in range(0,len(DataSet.columns)):
    cNameOld=DataSet_FIX.columns[i] + '           '
    print("Old : ",cNameOld);
    cNameNew=cNameOld.strip().replace(" ", ",")
    print("New : ",cNameNew);
    DataSet_FIX.columns.values[i] = cNameNew
    print(DataSet.columns[i],type(DataSet.columns[i]))
#####################################################################
#print(DataSet_FIX.head())
#####################################################################
print('#################')
print('Fixed Data Set with ID')
print('#################')
DataSet_with_ID=DataSet_FIX
```

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

```
print(DataSet_with_ID.head())
print('################')
sTable2='Retrieve_IP_DATA'
DataSet_with_ID.to_sql(sTable2,conn,index_label="RowID", if_exists="replace")
#####################################################################
print('### Done!! ##############################################')
```

OUTPUT:

```
========== RESTART: C:\Users\Acer\Desktop\Datasetfix.py ==========
('Loading :', 'C:/Users/Acer/Desktop/sqlite-tools-win32-x86-3340100/sqlite-tools
-win32-x86-3340100/srk.db')
('Rows:', 4761)
('Columns:', 22)
### Raw Data Set ###################################
('RowIDCSV', <type 'str'>)
('Unnamed: 0', <type 'str'>)
('Unnamed: 0.1', <type 'str'>)
('acousticness', <type 'str'>)
('artists', <type 'str'>)
('danceability', <type 'str'>)
('duration_ms', <type 'str'>)
('energy', <type 'str'>)
('explicit', <type 'str'>)
('id', <type 'str'>)
('instrumentalness', <type 'str'>)
('key', <type 'str'>)
('liveness', <type 'str'>)
('loudness', <type 'str'>)
('mode', <type 'str'>)
('name', <type 'str'>)
('popularity', <type 'str'>)
('release_date', <type 'str'>)
('speechiness', <type 'str'>)
('tempo', <type 'str'>)
('valence', <type 'str'>)
('year', <type 'str'>)
### Fixed Data Set ##################################
('Old : ', 'RowIDCSV     ')
('New : ', 'RowIDCSV')
('RowIDCSV', <type 'str'>)
('Old : ', 'Unnamed: 0     ')
('New : ', 'Unnamed:,0')
('Unnamed:,0', <type 'str'>)
```

```
### Fixed Data Set ##################################
('Old : ', 'RowIDCSV     ')
('New : ', 'RowIDCSV')
('RowIDCSV', <type 'str'>)
('Old : ', 'Unnamed: 0     ')
('New : ', 'Unnamed:,0')
('Unnamed:,0', <type 'str'>)
('Old : ', 'Unnamed: 0.1     ')
('New : ', 'Unnamed:,0.1')
('Unnamed:,0.1', <type 'str'>)
('Old : ', 'acousticness     ')
('New : ', 'acousticness')
('acousticness', <type 'str'>)
('Old : ', 'artists     ')
('New : ', 'artists')
('artists', <type 'str'>)
('Old : ', 'danceability     ')
('New : ', 'danceability')
('danceability', <type 'str'>)
('Old : ', 'duration_ms     ')
('New : ', 'duration_ms')
('duration_ms', <type 'str'>)
('Old : ', 'energy     ')
('New : ', 'energy')
('energy', <type 'str'>)
('Old : ', 'explicit     ')
('New : ', 'explicit')
('explicit', <type 'str'>)
('Old : ', 'id     ')
('New : ', 'id')
('id', <type 'str'>)
('Old : ', 'instrumentalness     ')
('New : ', 'instrumentalness')
('instrumentalness', <type 'str'>)
('Old : ', 'key     ')
('New : ', 'key')
```

```
Fixed Data Set with ID
#################
#################
   RowIDCSV  Unnamed:,0  Unnamed:,0.1  ...    tempo valence    year
0         0           0             0  ...  149.976  0.6340  1920.0
1         1           1             1  ...   86.889  0.9500  1920.0
2         2           2             2  ...   97.600  0.6890  1920.0
3         3           3             3  ...  127.997  0.0422  1920.0
4         4           4             4  ...  122.076  0.2990  1920.0

[5 rows x 22 columns]
#################
```

```
CREATE TABLE IF NOT EXISTS "Retrieve_IP_DATA" (
"RowID" INTEGER,
  "RowIDCSV" INTEGER,
  "Unnamed:,0" INTEGER,
  "Unnamed:,0.1" INTEGER,
  "acousticness" REAL,
  "artists" TEXT,
  "danceability" REAL,
  "duration_ms" INTEGER,
  "energy" TEXT,
  "explicit" INTEGER,
  "id" TEXT,
  "instrumentalness" REAL,
  "key" INTEGER,
  "liveness" REAL,
  "loudness" REAL,
  "mode" INTEGER,
  "name" TEXT,
  "popularity" INTEGER,
  "release_date" TEXT,
  "speechiness" REAL,
  "tempo" REAL,
  "valence" REAL,
  "year" INTEGER
);
```

Practical No 5

Assess Superstep

1)Drop the Columns Where All Elements Are Missing Values

```
#-*-coding:utf-8-*-
######################################################################
import sys
import os
import pandas as pd
######################################################################
sFileName='C:\Users\Acer\Desktop\DataSet3.csv'
######################################################################
print('Loading:',sFileName)
RawData=pd.read_csv(sFileName,header=0)
print('#################################')
print('##RawDataValues')
print('#################################')
print(RawData)
print('#################################')
print('##DataProfile')
print('#################################')
print('Rows:',RawData.shape[0])
print('Columns:',RawData.shape[1])
print('#################################')
######################################################################
TestData=RawData.dropna(axis=1,how='all')
######################################################################
print('#################################')
print('##TestDataValues')
print('#################################')
print(TestData)
print('#################################')
print('##DataProfile')
print('#################################')
print('Rows:',TestData.shape[0])
print('Columns:',TestData.shape[1])
```

```
print('################################')
#####################################################################
sFileName='C:\Users\Acer\Desktop\DataSet3.csv'
TestData.to_csv(sFileName,index=False)
#####################################################################
print('################################') print('###Done!!#####################')
print('################################')
```

OUTPUT:

```
('Loading:', 'C:\\Users\\Acer\\Desktop\\DataSet3.csv')
################################
##RawDataValues
################################
                         artists  danceability  ...  Unnamed: 12  Unnamed: 13
0                  ['Mamie Smith']         0.598  ...          NaN          NaN
1          ["Screamin' Jay Hawkins"]         0.852  ...          NaN          NaN
2                  ['Mamie Smith']         0.647  ...          NaN          NaN
3               ['Oscar Velazquez']         0.730  ...          NaN          NaN
4                          ['Mixe']         0.704  ...          NaN          NaN
5   ['Mamie Smith & Her Jazz Hounds']         0.424  ...          NaN          NaN
6                  ['Mamie Smith']         0.782  ...          NaN          NaN
7   ['Mamie Smith & Her Jazz Hounds']         0.474  ...          NaN          NaN
8              ['Francisco Canaro']         0.469  ...          NaN          NaN
9                        ['Meetya']         0.571  ...          NaN          NaN
10                    ['Dorville']         0.688  ...          NaN          NaN
11             ['Francisco Canaro']         0.579  ...          NaN          NaN
12                    ['Ka Koula']         0.353  ...          NaN          NaN
13                    ['Justrock']         0.643  ...          NaN          NaN
14               ['Takis Nikolaou']         0.453  ...          NaN          NaN
15          ['Aggeliki Karagianni']         0.525  ...          NaN          NaN
16             ['Giorgos Katsaros']         0.528  ...          NaN          NaN
17             ['Francisco Canaro']         0.468  ...          NaN          NaN
18             ['Giorgos Katsaros']         0.528  ...          NaN          NaN

[19 rows x 14 columns]
################################
##DataProfile
################################
('Rows:', 19)
('Columns:', 14)
################################
################################
##TestDataValues
```

```
--------------------------------
##TestDataValues
################################
                         artists  ...  year
0                  ['Mamie Smith']  ...  1920
1          ["Screamin' Jay Hawkins"]  ...  1920
2                  ['Mamie Smith']  ...  1920
3               ['Oscar Velazquez']  ...  1920
4                          ['Mixe']  ...  1920
5   ['Mamie Smith & Her Jazz Hounds']  ...  1920
6                  ['Mamie Smith']  ...  1920
7   ['Mamie Smith & Her Jazz Hounds']  ...  1920
8              ['Francisco Canaro']  ...  1920
9                        ['Meetya']  ...  1920
10                    ['Dorville']  ...  1920
11             ['Francisco Canaro']  ...  1920
12                    ['Ka Koula']  ...  1920
13                    ['Justrock']  ...  1920
14               ['Takis Nikolaou']  ...  1920
15          ['Aggeliki Karagianni']  ...  1920
16             ['Giorgos Katsaros']  ...  1920
17             ['Francisco Canaro']  ...  1920
18             ['Giorgos Katsaros']  ...  1920

[19 rows x 10 columns]
################################
##DataProfile
################################
```

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

```
import sys
import os
import pandas as pd
sFileName='C:\Users\Acer\Desktop\DataSet2.csv'
print('Loading:',sFileName)
RawData=pd.read_csv(sFileName,header=0)
print('##RawDataValues')
print(RawData) print('##DataProfile')
print('Rows:',RawData.shape[0])
print('Columns:',RawData.shape[1])
print('################################')
TestData=RawData.dropna(axis=1,how='any')
print('##TestDataValues')
print(TestData) print('##DataProfile')
print('Rows:',TestData.shape[0])
print('Columns:',TestData.shape[1])
sFileName='C:\Users\Acer\Desktop\DataSet2.csv'
TestData.to_csv(sFileName,index=False)
print('###Done!!####################')
```

OUTPUT:

```
------------------------------
##RawDataValues
################################
                              artists  danceability  ...  Unnamed: 12  Unnamed: 13
0                       ['Mamie Smith']         0.598  ...          NaN          NaN
1             ["Screamin' Jay Hawkins"]           NaN  ...          NaN          NaN
2                       ['Mamie Smith']           NaN  ...          NaN          NaN
3                  ['Oscar Velazquez']           NaN  ...          NaN          NaN
4                             ['Mixe']           NaN  ...          NaN          NaN
5   ['Mamie Smith & Her Jazz Hounds']           NaN  ...          NaN          NaN
6                       ['Mamie Smith']           NaN  ...          NaN          NaN
7   ['Mamie Smith & Her Jazz Hounds']           NaN  ...          NaN          NaN
8                  ['Francisco Canaro']         0.469  ...          NaN          NaN
9                           ['Meetya']         0.571  ...          NaN          NaN
10                        ['Dorville']         0.688  ...          NaN          NaN
11                 ['Francisco Canaro']         0.579  ...          NaN          NaN
12                        ['Ka Koula']         0.353  ...          NaN          NaN
13                        ['Justrock']         0.643  ...          NaN          NaN
14                   ['Takis Nikolaou']         0.453  ...          NaN          NaN
15              ['Aggeliki Karagianni']         0.525  ...          NaN          NaN
16                 ['Giorgos Katsaros']         0.528  ...          NaN          NaN
17                 ['Francisco Canaro']         0.468  ...          NaN          NaN
18                 ['Giorgos Katsaros']         0.528  ...          NaN          NaN

[19 rows x 14 columns]
################################
##DataProfile
################################
('Rows:', 19)
('Columns:', 14)
################################
################################
##TestDataValues
################################
```

```
##TestDataValues
################################
                                artists  ...  year
0                       ['Mamie Smith']  ...  1920
1             ["Screamin' Jay Hawkins"]  ...  1920
2                       ['Mamie Smith']  ...  1920
3                  ['Oscar Velazquez']  ...  1920
4                              ['Mixe']  ...  1920
5   ['Mamie Smith & Her Jazz Hounds']  ...  1920
6                       ['Mamie Smith']  ...  1920
7   ['Mamie Smith & Her Jazz Hounds']  ...  1920
8                 ['Francisco Canaro']  ...  1920
9                           ['Meetya']  ...  1920
10                        ['Dorville']  ...  1920
11                ['Francisco Canaro']  ...  1920
12                        ['Ka Koula']  ...  1920
13                        ['Justrock']  ...  1920
14                  ['Takis Nikolaou']  ...  1920
15             ['Aggeliki Karagianni']  ...  1920
16                ['Giorgos Katsaros']  ...  1920
17                ['Francisco Canaro']  ...  1920
18                ['Giorgos Katsaros']  ...  1920

[19 rows x 7 columns]
################################
##DataProfile
################################
('Rows:', 19)
('Columns:', 7)
################################
```

3)Keep Only the Columns that missing only 2 values.

```
import sys
import os
import pandas as pd ###
Import Warehouse
sFileName='C:\Users\Acer\Desktop\DataSet2.csv'
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])
print('##################################')
TestData=RawData.dropna(thresh=2,axis=1) print('##
Test Data Values')
print(TestData) print('##
Data Profile')
print('Rows :',TestData.shape[0]) print('Columns
:',TestData.shape[1]) TestData.to_csv(sFileName,
index = False) print('### Done!!
######################')
```

OUTPUT:

```
('Loading :', 'C:\\Users\\Acer\\Desktop\\DataSet2.csv')
## Raw Data Values
    Unamed                              artists  ...  Unnamed: 15  Unnamed: 16
0        0                      ['Mamie Smith']  ...          NaN          NaN
1        1           ["Screamin' Jay Hawkins"]  ...          NaN          NaN
2        2                      ['Mamie Smith']  ...          NaN          NaN
3        3                  ['Oscar Velazquez']  ...          NaN          NaN
4        4                           ['Mixe']  ...          NaN          NaN
5        5  ['Mamie Smith & Her Jazz Hounds']  ...          NaN          NaN
6        6                      ['Mamie Smith']  ...          NaN          NaN
7        7  ['Mamie Smith & Her Jazz Hounds']  ...          NaN          NaN
8        8                ['Francisco Canaro']  ...          NaN          NaN
9        9                         ['Meetya']  ...          NaN          NaN
10      10                       ['Dorville']  ...          NaN          NaN
11      11                ['Francisco Canaro']  ...          NaN          NaN
12      12                       ['Ka Koula']  ...          NaN          NaN
13      13                       ['Justrock']  ...          NaN          NaN
14      14                               NaN  ...          NaN          NaN
15      15                               NaN  ...          NaN          NaN
16      16                               NaN  ...          NaN          NaN
17      17                               NaN  ...          NaN          NaN
18      18                ['Giorgos Katsaros']  ...          NaN          NaN

[19 rows x 17 columns]
## Data Profile
('Rows :', 19)
('Columns :', 17)
```

```
################################
## Test Data Values
    Unamed                              artists  danceability  duration_ms
0        0                       ['Mamie Smith']         0.598       168333
1        1             ["Screamin' Jay Hawkins"]         0.852       150200
2        2                       ['Mamie Smith']         0.647       163827
3        3                  ['Oscar Velazquez']         0.730       422087
4        4                              ['Mixe']         0.704       165224
5        5  ['Mamie Smith & Her Jazz Hounds']           NaN       198627
6        6                       ['Mamie Smith']           NaN       195200
7        7  ['Mamie Smith & Her Jazz Hounds']           NaN       186173
8        8                 ['Francisco Canaro']           NaN       146840
9        9                           ['Meetya']           NaN       476304
10      10                         ['Dorville']           NaN       150067
11      11                 ['Francisco Canaro']           NaN       167213
12      12                         ['Ka Koula']           NaN       285707
13      13                         ['Justrock']         0.643       304078
14      14                                  NaN         0.453       255520
15      15                                  NaN         0.525       258167
16      16                                  NaN         0.528       277720
17      17                                  NaN         0.468       177427
18      18                 ['Giorgos Katsaros']         0.528       278813
## Data Profile
('Rows :', 19)
('Columns :', 4)
```

Practical No 6

Process superstep

LOCATION.PY

```
import sys
import os
import pandas as pd
import sqlite3 as sq
from pandas.io import sql
import uuid
sDatabaseName='C:\Users\Acer\Desktop\sqlite-tools-win32-x86-
3340100\sqlite-tools-win32-x86-3340100\srk.db'
conn1 = sq.connect(sDatabaseName)
sDatabaseName='C:\Users\Acer\Desktop\sqlite-tools-win32-x86-
3340100\sqlite-tools-win32-x86-3340100\go.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, '+07f') +\ '-
                '+format(round(Latitude,3)*1000, '+07f')
      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_items(LocationLine) else:
         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 = 'HubLocation'
print('Storing :',sDatabaseName,' Table:',sTable) LocationHubIndex.to_sql(sTable,
conn2, if_exists="replace")
#################################################################
print('################')
print('Srk Databases')
sSQL="srk;"
sql.execute(sSQL,conn1)
sql.execute(sSQL,conn2)
print('################')
################################################################# print('###
Done!! ##############################################')
```

OUTPUT:

```
('Create:', 1, ' of ', 64800, ':', 'L-180000.000000--90000.000000')

('Create:', 2, ' of ', 64800, ':', 'L-180000.000000--80000.000000')

('Create:', 3, ' of ', 64800, ':', 'L-180000.000000--70000.000000')
('Create:', 4, ' of ', 64800, ':', 'L-180000.000000--60000.000000')
('Create:', 5, ' of ', 64800, ':', 'L-180000.000000--50000.000000')
('Create:', 6, ' of ', 64800, ':', 'L-180000.000000--40000.000000')
('Create:', 7, ' of ', 64800, ':', 'L-180000.000000--30000.000000')
('Create:', 8, ' of ', 64800, ':', 'L-180000.000000--20000.000000')
('Create:', 9, ' of ', 64800, ':', 'L-180000.000000--10000.000000')
('Create:', 10, ' of ', 64800, ':', 'L-180000.000000-+0.000000')
('Create:', 11, ' of ', 64800, ':', 'L-180000.000000-+10000.000000')
('Create:', 12, ' of ', 64800, ':', 'L-180000.000000-+20000.000000')
('Create:', 13, ' of ', 64800, ':', 'L-180000.000000-+30000.000000')
('Create:', 14, ' of ', 64800, ':', 'L-180000.000000-+40000.000000')
('Create:', 15, ' of ', 64800, ':', 'L-180000.000000-+50000.000000')
('Create:', 16, ' of ', 64800, ':', 'L-180000.000000-+60000.000000')
('Create:', 17, ' of ', 64800, ':', 'L-180000.000000-+70000.000000')
('Create:', 18, ' of ', 64800, ':', 'L-180000.000000-+80000.000000')
('Create:', 19, ' of ', 64800, ':', 'L-170000.000000--90000.000000')
('Create:', 20, ' of ', 64800, ':', 'L-170000.000000--80000.000000')
```

```
('Create:', 640, ' of ', 64800, ':', 'L+170000.000000-+0.000000')
('Create:', 641, ' of ', 64800, ':', 'L+170000.000000-+10000.000000')
('Create:', 642, ' of ', 64800, ':', 'L+170000.000000-+20000.000000')
('Create:', 643, ' of ', 64800, ':', 'L+170000.000000-+30000.000000')
('Create:', 644, ' of ', 64800, ':', 'L+170000.000000-+40000.000000')
('Create:', 645, ' of ', 64800, ':', 'L+170000.000000-+50000.000000')
('Create:', 646, ' of ', 64800, ':', 'L+170000.000000-+60000.000000')
('Create:', 647, ' of ', 64800, ':', 'L+170000.000000-+70000.000000')
('Create:', 648, ' of ', 64800, ':', 'L+170000.000000-+80000.000000')
('Storing :', 'C:\\Users\\Acer\\Desktop\\sqlite-tools-win32-x86-3340100\\sqlite-
tools-win32-x86-3340100\\go.db', ' Table:', 'ProcessLocation')
('Storing :', 'C:\\Users\\Acer\\Desktop\\sqlite-tools-win32-x86-3340100\\sqlite-
tools-win32-x86-3340100\\go.db', ' Table:', 'HubLocation')
################
Srk Databases
################
### Done!! ############################################
```

```
C:\Users\Acer\Desktop\sqlite-tools-win32-x86-3340100\sqlite-tools-win32-x86-3340100>sqlite3 go.db
SQLite version 3.34.1 2021-01-20 14:10:07
Enter ".help" for usage hints.
sqlite> .tables
Dim-BMI-Island    Dim_Person        Hub-Location      lol
Dim-BMI-Vertical  Dim_Time          HubEvent
DimBMIHorizontal  Fact_Person_Time  HubLocation
sqlite> select * from HubLocation;
6e953ce0-07f5-4237-b27f-3f5c1910ef7c|GPS|1|L-180000.000000--90000.000000|-180|-90
c9a0beeb-fe9b-408e-9eec-a13af392c17d|GPS|2|L-180000.000000--80000.000000|-180|-80
8fbd8f05-e83a-45e4-868f-0db331ed78b0|GPS|3|L-180000.000000--70000.000000|-180|-70
58ae5c7f-5753-4ea6-8027-55945989e14f|GPS|4|L-180000.000000--60000.000000|-180|-60
072a5aa6-5389-45fc-997d-0a26b44cab2b|GPS|5|L-180000.000000--50000.000000|-180|-50
f6873895-f56e-415b-9a2f-cbe4eeb4f726|GPS|6|L-180000.000000--40000.000000|-180|-40
d56d0ba4-ca7e-4c32-ae5d-fbcbd0d4815d|GPS|7|L-180000.000000--30000.000000|-180|-30
09901d9b-9243-47d2-a47d-f58880ca58c2|GPS|8|L-180000.000000--20000.000000|-180|-20
ba0707d4-5be7-4333-9890-474d0673367c|GPS|9|L-180000.000000--10000.000000|-180|-10
10825757-f82d-44f4-95be-45062fb60f5c|GPS|10|L-180000.000000-+0.000000|-180|0
87a46457-033b-41f9-a8c3-dda7616e629d|GPS|11|L-180000.000000-+10000.000000|-180|10
```

TIME.PY

```
from datetime import datetime
from pytz import timezone, all_timezones
now_date=datetime(2021,02,17,13,26,6,7);
now_utc=now_date.replace(tzinfo=timezone('GMT'))
print('Date:',str(now_utc.strftime("%Y-%m-%d %H:%M:%S (%Z)")))
print('Year:',str(now_utc.strftime("%Y")))
print('Month:',str(now_utc.strftime("%m")))
print('Month:',str(now_utc.strftime("%B")))
print('Day:',str(now_utc.strftime("%d")))
print('Hours:',str(now_utc.strftime("%H"))) print('Minutes:',str(now_utc.strftime("%M")))
print('Seconds:',str(now_utc.strftime("%S")))
print('Mill.Seconds:',str(now_utc.strftime("%f")))
```

OUTPUT:

```
========== RESTART: C:\Users\Acer\Desktop\ds\drop1.py ==========
('Date:', '2021-02-17 13:26:06 (GMT)')
('Year:', '2021')
('Month:', '02')
('Month:', 'February')
('Day:', '17')
('Hours:', '13')
('Minutes:', '26')
('Seconds:', '06')
('Mill.Seconds:', '000007')
>>>
```

EVENT.PY

```
import sys
import os
import pandas as pd
import sqlite3 as sq
from pandas.io import sql
#####################################################################
InputFileName='C:\Users\Acer\Desktop\DataSet.csv'
#####################################################################
#####################################################################
sDatabaseName='C:\Users\Acer\Desktop\sqlite-tools-win32-x86-
3340100\sqlite-tools-win32-x86-3340100\srk.db'
conn1 = sq.connect(sDatabaseName)
#####################################################################
#####################################################################
sDatabaseName='C:\Users\Acer\Desktop\sqlite-tools-win32-x86-
3340100\sqlite-tools-win32-x86-3340100\go.db'
conn2 = sq.connect(sDatabaseName)
#####################################################################
sFileName=InputFileName
print('Loading :',sFileName)
EventRawData=pd.read_csv(sFileName,header=0,low_memory=False,
encoding="latin-1")
EventRawData.index.names=['EventID']
EventHubIndex=EventRawData
#####################################################################
sTable = 'ProcessEvent'
print('Storing :',sDatabaseName,' Table:',sTable)
EventHubIndex.to_sql(sTable, conn1, if_exists="replace")
##################################################################### #
sTable = 'HubEvent'
print('Storing :',sDatabaseName,' Table:',sTable)
EventHubIndex.to_sql(sTable, conn2, if_exists="replace")
##################################################################### #
print('################')
print('Srk_Databases')
```

```
sSQL="Srk;"
sql.execute(sSQL,conn1)
sql.execute(sSQL,conn2)
print('################')
########################################################################## print('###
Done!! ##############################################')
```

OUTPUT:

```
Python 2.7.16 (v2.7.16:413a49145e, Mar  4 2019, 01:30:55) [MSC v.1500 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================== RESTART: C:\Users\Acer\Desktop\event.py ==================
('Loading :', 'C:\\Users\\Acer\\Desktop\\DataSet.csv')
('Storing :', 'C:\\Users\\Acer\\Desktop\\sqlite-tools-win32-x86-3340100\\sqlite-
tools-win32-x86-3340100\\go.db', ' Table:', 'ProcessEvent')

Warning (from warnings module):
  File "C:\Python27\lib\site-packages\pandas\core\generic.py", line 2531
    dtype=dtype, method=method)
UserWarning: The spaces in these column names will not be changed. In pandas ver
sions < 0.14, spaces were converted to underscores.
('Storing :', 'C:\\Users\\Acer\\Desktop\\sqlite-tools-win32-x86-3340100\\sqlite-
tools-win32-x86-3340100\\go.db', ' Table:', 'HubEvent')
################
Srk_Databases
################
### Done!! ############################################
>>>
```

```
sqlite> .tables
Dim-BMI-Island    Dim_Person        Hub-Location      lol
Dim-BMI-Vertical  Dim_Time          HubEvent
DimBMIHorizontal  Fact_Person_Time  HubLocation
sqlite> select * from HubEvent limit 5;
0|0|0|0.991|['Mamie Smith']|0.598|168333|0
1|1|1|0.643|["Screamin' Jay Hawkins"]|0.852|150200|0
2|2|2|0.993|['Mamie Smith']|0.647|163827|0
3|3|3|0.000173|['Oscar Velazquez']|0.73|422087|0
4|4|4|0.295|['Mixe']|0.704|165224|1
sqlite>
```

Practical No 7

Transform Superstep

LINEAR REGRESSION.PY

```
import sys
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import linear_model, datasets
sFileName='C:\Users\Acer\Desktop\DataSet.csv'
print('Loading :',sFileName)
DataRaw=pd.read_csv(sFileName,header=0,low_memory=False,
encoding="latin-1")

#####################################################################
   sFileName='C:\Users\Acer\Desktop\DataSet.csv'
print('Storing :',sFileName)
DataRaw.to_csv(sFileName, index = False, encoding="latin-1")
#####################################################################
# import data to ecosystem
iris = datasets.load_iris()
X = iris.data[:, :2] # we only take the first two features.
Y = iris.target
h = .02 # step size in the mesh
logreg = linear_model.LogisticRegression(C=1e5)
# we create an instance of Neighbours Classifier and fit the data.
logreg.fit(X, Y)
# Plot the decision boundary. For that, we will assign a color to each #
point in the mesh [x_min, x_max]x[y_min, y_max].
x_min, x_max = X[:, 0].min() - .5, X[:, 0].max() + .5
y_min, y_max = X[:, 1].min() - .5, X[:, 1].max() + .5
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = logreg.predict(np.c_[xx.ravel(), yy.ravel()])

# Put the result into a color plot
Z = Z.reshape(xx.shape)
plt.figure(1, figsize=(8, 6))
```

```
plt.pcolormesh(xx, yy, Z, cmap=plt.cm.Paired)

# Plot also the training points
plt.scatter(X[:, 0], X[:, 1], c=Y, edgecolors='k', cmap=plt.cm.Paired) plt.title('Shipping
Box Sizes')
plt.xlabel('Box Length')
plt.ylabel('Box Width')
plt.xlim(xx.min(), xx.max())
plt.ylim(yy.min(), yy.max())
plt.xticks(())
plt.yticks(())
plt.show()
```

OUTPUT:

CHISQUARE.PY

```
import numpy as np import
scipy.stats as st

np.random.seed(1)

# Create sample data.
nSet=3
if nSet==1:
   a = abs(np.random.randn(20))
   b = abs(50*np.random.randn(30)) if
nSet==2:
a=np.array([27.1,22.0,20.8,23.4,23.4,23.5,25.8,22.0,24.8,20.2,21.9,22.1,22.9,20.
5,24.4])
b=np.array([27.1,22.0,20.8,23.4,23.4,23.5,25.8,22.0,24.8,20.2,21.9,22.1,22.9,20
.5,24.41])
if nSet==3:
   a=np.array([17.2,20.9,22.6,18.1,21.7,21.4,23.5,24.2,14.7,21.8])
   b=np.array([21.5,22.8,21.0,23.0,21.6,23.6,22.5,20.7,23.4,21.8])
obs = np.array([a,b])
chi2, p, dof, expected = st.chi2_contingency(obs) msg
= "Test Statistic : {}\np-value: {}\ndof: {}\n" print(
msg.format( chi2, p , dof,expected) )

P=1-p
if P < 0.001:
   print('Statistically highly significant:',P)
else:
   if P < 0.05:
      print('Statistically significant:',P)
   else:
      print('No conclusion')
```

OUTPUT:

```
========== RESTART: C:\Users\Acer\Desktop\chisq.py ==========
Test Statistic : 0.00785295014029
p-value: 0.999999999675
dof: 7

('Statistically highly significant:', 3.2511682235281114e-10)
>>>
```

SUN MODEL.PY

```
import sys
import os
from datetime import datetime
from pytz import timezone
import pandas as pd
import sqlite3 as sq
import uuid
pd.options.mode.chained_assignment = None
#####################################################################

sDatabaseName='C:/Users/Acer/Desktop/sqlite-tools-win32-x86-
3340100/sqlite-tools-win32-x86-3340100/srk.db'
conn1 = sq.connect(sDatabaseName)

#####################################################################
sDatabaseName='C:/Users/Acer/Desktop/sqlite-tools-win32-x86-
3340100/sqlite-tools-win32-x86-3340100/go.db'
conn2 = sq.connect(sDatabaseName)
#####################################################################
print('\n#################################')
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) (%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_items(TimeLine)
```

```
#################################################################
DimTime=TimeFrame DimTimeIndex=DimTime.set_index(['TimeID'],inplace=False)
#################################################################
sTable = 'Dim_Time'
print('\n#################################')
print('Storing :',sDatabaseName,'\n Table:',sTable)
print('\n#################################')
DimTimeIndex.to_sql(sTable, conn1, if_exists="replace") DimTimeIndex.to_sql(sTable,
conn2, if_exists="replace")
#################################################################
print('\n#################################')
print('Dimension Person')
print('\n#################################')
FirstName = 'Shweta' LastName
= 'Kasbe'
#################################################################
IDPersonNumber=str(uuid.uuid4())
PersonLine=[('PersonID', [IDPersonNumber]),
            ('FirstName', [FirstName]),
            ('LastName', [LastName]),
            ('Zone', ['UTC']),
            ('DateTimeValue', [BirthDateZoneStr])] PersonFrame
= pd.DataFrame.from_items(PersonLine)
#################################################################
DimPerson=PersonFrame
DimPersonIndex=DimPerson.set_index(['PersonID'],inplace=False)
#################################################################
sTable = 'Dim_Person'
print('\n#################################')
print('Storing :',sDatabaseName,'\n Table:',sTable)
print('\n#################################')
DimPersonIndex.to_sql(sTable, conn1, if_exists="replace")
DimPersonIndex.to_sql(sTable, conn2, if_exists="replace")
#################################################################
print('\n#################################')
print('FactPersontime')
print('\n#################################')
IDFactNumber=str(uuid.uuid4())
PersonTimeLine=[('IDNumber', [IDFactNumber]),
              ('IDPersonNumber', [IDPersonNumber]),
              ('IDTimeNumber', [IDTimeNumber])]
```

```
PersonTimeFrame = pd.DataFrame.from_items(PersonTimeLine)
#####################################################################
FctPersonTime=PersonTimeFrame
FctPersonTimeIndex=FctPersonTime.set_index(['IDNumber'],inplace=False)
#####################################################################
sTable = 'Fact_Person_Time'
print('\n###################################')
print('Storing :',sDatabaseName,'\n Table:',sTable)
print('\n###################################')
FctPersonTimeIndex.to_sql(sTable,    conn1,    if_exists="replace")
FctPersonTimeIndex.to_sql(sTable,    conn2,    if_exists="replace")
print('Done')
```

OUTPUT:

```
================ RESTART: C:\Users\Acer\Desktop\SunModel.py ================

###################################
Time Dimension


###############################
('Storing :', 'C:/Users/Acer/Desktop/sqlite-tools-win32-x86-3340100/sqlite-tools-win32-x86-3340100/go.db', '\n Table:', 'Dim_Time')
###############################

###############################
Dimension Person
###############################


###############################
('Storing :', 'C:/Users/Acer/Desktop/sqlite-tools-win32-x86-3340100/sqlite-tools-win32-x86-3340100/go.db', '\n Table:', 'Dim_Person')
###############################

###############################
FactPersontime

###############################
('Storing :', 'C:/Users/Acer/Desktop/sqlite-tools-win32-x86-3340100/sqlite-tools-win32-x86-3340100/go.db', '\n Table:', 'Fact_Person_Time')

###############################
Done
```

```
sqlite> select * from Dim_Person;
f14d4ee8-858e-49be-8277-57ec257181f3|Shweta|Kasbe|UTC|1960-12-20 10:15:00
sqlite>
```

```
sqlite> select * from Dim_Time ;
b95da42b-2741-409c-9c4e-9818106c3e97|1960-12-20 10:15:00|1960-12-20 09:15:00|Atlantic/Reykjavik
sqlite>
```

Practical No 8

Organize Superstep

HORIZONTAL.PY

```
import sys
import os
import pandas as pd
import sqlite3 as sq
#####################################################################
sDatabaseName= 'C:\Users\Acer\Desktop\sqlite-tools-win32-x86-
3340100\sqlite-tools-win32-x86-3340100\srk.db'
conn1 = sq.connect(sDatabaseName)
#####################################################################
sDatabaseName1= 'C:\Users\Acer\Desktop\sqlite-tools-win32-x86-
3340100\sqlite-tools-win32-x86-3340100\go.db'
conn2 = sq.connect(sDatabaseName1)
#####################################################################
print('################')
sTable = 'DimBmi'
print('Loading :',sDatabaseName,' Table:',sTable) ##
id, key,tempo,year
sSQL="SELECT * FROM DimBmi;"
PersonFrame0=pd.read_sql_query(sSQL, conn1)
#####################################################################
print('################################')
sTable = 'DimBmi'
print('Loading :',sDatabaseName,' Table:',sTable, 'After apply Horizontal style')
print('################################')
sSQL="SELECT *\
  FROM [DimBmi]\ WHERE \
  key > 5 \
  and tempo= 1\
  ORDER BY \
      year;" PersonFrame1=pd.read_sql_query(sSQL,
conn1)
#####################################################################
DimPerson=PersonFrame1 DimPersonIndex=DimPerson.set_index(['key'],inplace=False)
#####################################################################
```

```
sTable = 'DimBMIHorizontal'
print('\n#################################')
print('Storing :',sDatabaseName1,'\n Table:',sTable)
print('\n#################################')
DimPersonIndex.to_sql(sTable, conn2, if_exists="replace")
sSQL="SELECT * FROM DimBMIHorizontal;"
PersonFrame2=pd.read_sql_query(sSQL, conn2)
print(PersonFrame2)
#####################################################################
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('#################################')
```

OUTPUT:

```
Columns: [key, liveness, mode, tempo, name, year]
Index: []
#################################
('Full Data Set (Rows):', 2)
('Full Data Set (Columns):', 6)
#################################
('Horizontal Data Set (Rows):', 0)
('Horizontal Data Set (Columns):', 6)
#################################
>>>
```

VERTICAL.PY

```
import sys
import os
import pandas as pd
import sqlite3 as sq
sDatabaseName='C:\Users\Acer\Desktop\sqlite-tools-win32-x86-
3340100\sqlite-tools-win32-x86-3340100\srk.db'
conn1=sq.connect(sDatabaseName)
##################################################
#############
sDatabaseName='C:\Users\Acer\Desktop\sqlite-tools-win32-x86-
3340100\sqlite-tools-win32-x86-3340100\go.db'
conn2=sq.connect(sDatabaseName)
print('################################')
sTable='DimBmi'
print('Loading:',sDatabaseName,'Table:',sTable)
sSQL="SELECT*FROM[DimBmi];"
PersonFrame0=pd.read_sql_query(sSQL,conn1)
##################################################
############ print('################################')
sTable='DimBmi'
print('Loading:',sDatabaseName,'Table:',sTable)
print('################################')
sSQL="SELECT name,key,tempo FROM [DimBmi];"
PersonFrame1=pd.read_sql_query(sSQL,conn1)
################################################## #############
DimPerson=PersonFrame1
DimPersonIndex=DimPerson.set_index(['key'],inplace=False)
################################################## #############
sTable='Dim-BMI-Vertical'
print('\n################################')
print('Storing:',sDatabaseName,'\nTable:',sTable)
print('\n################################')
DimPersonIndex.to_sql(sTable,conn2,if_exists="replace")
################################################## #############
print('################')
sTable='Dim-BMI-Vertical'
```

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

OUTPUT:

```
('Loading:', 'C:\\Users\\Acer\\Desktop\\sqlite-tools-win32-x86-3340100\\sqlite-t
ools-win32-x86-3340100\\go.db', 'Table:', 'Dim-BMI-Vertical')
################################
('FullDataSet(Rows):', 2)
('FullDataSet(Columns):', 6)
################################
('HorizontalDataSet(Rows):', 2)
('HorizontalDataSet(Columns):', 3)
################################
>>>
```

ISLAND.PY

```
import sys
import os
import pandas as pd
import sqlite3 as sq
################################################################ ##
sDatabaseName='C:\Users\Acer\Desktop\sqlite-tools-win32-x86- 3340100\sqlite-
tools-win32-x86-3340100\srk.db' conn1=sq.connect(sDatabaseName)
################################################################ ##
sDatabaseName='C:\Users\Acer\Desktop\sqlite-tools-win32-x86- 3340100\sqlite-
tools-win32-x86-3340100\go.db' conn2=sq.connect(sDatabaseName)
################################################################ ##
print('################')
sTable='DimBmi'
print('Loading:',sDatabaseName,'Table:',sTable)
sSQL="SELECT*FROM[DimBmi];"
PersonFrame0=pd.read_sql_query(sSQL,conn1)
################################################################ ##
print('################')
sTable='DimBmi'
print('Loading:',sDatabaseName,'Table:',sTable)
sSQL="SELECT name,key,tempo FROM [DimBmi] where key > 2 order by name;"
PersonFrame1=pd.read_sql_query(sSQL,conn1)
################################################################ ##
DimPerson=PersonFrame1
DimPersonIndex=DimPerson.set_index(['key'],inplace=False)
################################################################ ##
sTable='Dim-BMI-Island'
print('\n###################################')
print('Storing:',sDatabaseName,'\nTable:',sTable)
print('\n###################################')
DimPersonIndex.to_sql(sTable,conn2,if_exists="replace")
```

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

OUTPUT:

```
['Loading:', 'C:\\Users\\Acer\\Desktop\\sqlite-tools-win32-x86-3340100\\sqlite-
ools-win32-x86-3340100\\go.db', 'Table:', 'Dim-BMI-Island')
################################
################################
['FullDataSet(Rows):', 2)
['FullDataSet(Columns):', 6)
################################
['HorizontalDataSet(Rows):', 2)
['HorizontalDataSet(Columns):', 3)
################################
>>>
```

SECURE VAULT.PY

```
import sys
import os
import pandas as pd
import sqlite3 as sq
sDatabaseName='C:\Users\Acer\Desktop\sqlite-tools-win32-x86- 3340100\sqlite-
tools-win32-x86-3340100\srk.db' conn1=sq.connect(sDatabaseName)
#####################################################################
sDatabaseName='C:\Users\Acer\Desktop\sqlite-tools-win32-x86- 3340100\sqlite-
tools-win32-x86-3340100\srk.db' conn2=sq.connect(sDatabaseName)
#####################################################################
print('################')
sTable='DimBmi'
print('Loading:',sDatabaseName,'Table:',sTable)
sSQL="SELECT*FROM[DimBmi];"
PersonFrame0=pd.read_sql_query(sSQL,conn1)
#####################################################################
print('################')
sTable='DimBmi'
print('Loading:',sDatabaseName,'Table:',sTable)
sSQL="SELECT key,liveness from DimBmi where key > 1 order by name;"
PersonFrame1=pd.read_sql_query(sSQL,conn1)
#####################################################################
DimPerson=PersonFrame1 DimPersonIndex=DimPerson.set_index(['key'],inplace=False)
#####################################################################
sTable='Dim-BMI-Secure'
print('\n##################################')
print('Storing:',sDatabaseName,'\nTable:',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 key < 1;"
PersonFrame2=pd.read_sql_query(sSQL,conn2)
#####################################################################
```

```
print('################################')
print('FullDataSet(Rows):',PersonFrame0.shape[0])
print('FullDataSet(Columns):',PersonFrame0.shape[1])
print('################################')
print('HorizontalDataSet(Rows):',PersonFrame2.shape[0])
print('HorizontalDataSet(Columns):',PersonFrame2.shape[1])
print('OnlySamData')
print(PersonFrame2.head())
print('################################')
```

OUTPUT:

```
('Loading:', 'C:\\Users\\Acer\\Desktop\\sqlite-tools-win32-x86-3340100\\sqlite-t
ools-win32-x86-3340100\\srk.db', 'Table:', 'Dim-BMI-Secure')
################################
################################
('FullDataSet(Rows):', 2)
('FullDataSet(Columns):', 6)
################################
('HorizontalDataSet(Rows):', 0)
('HorizontalDataSet(Columns):', 2)
```

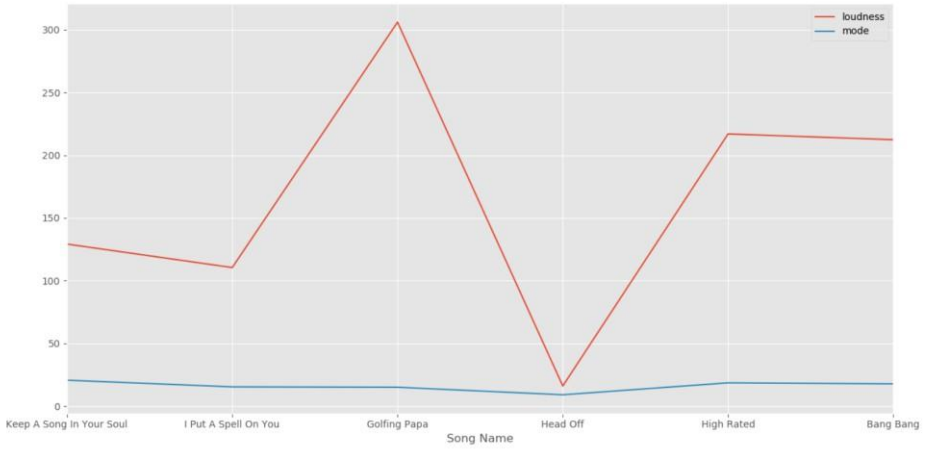
Practical No 9

Report Superstep

Line Graph :We can also visualize the data in the form of a Line graph.

```
import sys
import os
import pandas as pd
import matplotlib as ml
from matplotlib import pyplot as plt
ml.style.use('ggplot')
data=[['Keep A Song In Your Soul',129.2,20.8],['I Put A Spell On
You',110.5,15.5],['Golfing Papa',305.9,15.2],['Head Off',16.2,9.2],['High
Rated',216.9,18.7],['Bang Bang',212.3,17.9]]
new_data=pd.DataFrame(data) new_data.rename(columns={0:"Song
Name"},inplace=True)
new_data.rename(columns={1:"loudness"},inplace=True)
new_data.rename(columns={2:"mode"},inplace=True)
colors_name=['blue','red']
explode=(0,0,0,0,0,0)
label= new_data["Song Name"]
new_data.plot(figsize=(10,10),kind="line" ,y=["loudness","mode"],x='Song Name')
plt.savefig('C:\Users\Acer\Desktop\ds\Music.png',dpi=600)
plt.show()
```

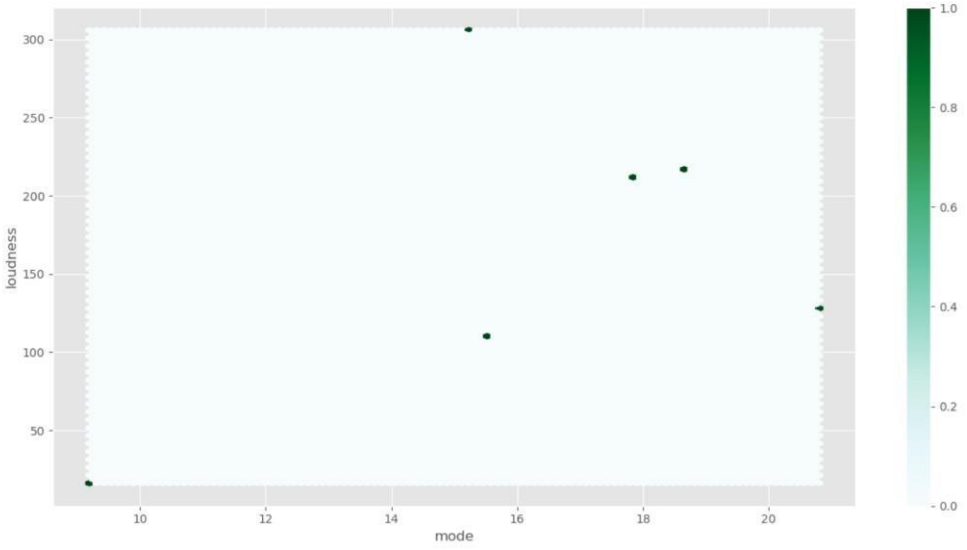
OUTPUT:



Hexbin Graph :We can also visualize the data in the form of a hexbingraph.

```
import sys
import os
import pandas as pd
import matplotlib as
ml
from matplotlib import pyplot as plt
ml.style.use('ggplot')
data=[['Keep A Song In Your Soul',129.2,20.8],['I Put A Spell On
You',110.5,15.5],['Golfing Papa',305.9,15.2],['Head Off',16.2,9.2],['High
Rated',216.9,18.7],['Bang Bang',212.3,17.9]]
new_data=pd.DataFrame(data)
new_data.rename(columns={0:"Song Name"},inplace=True)
new_data.rename(columns={1:"loudness"},inplace=True)
new_data.rename(columns={2:"mode"},inplace=True)
colors_name=['blue','red','green','gold','pink','yellow']
explode=(0,0,0,0,0,0)
label= new_data["Song Name"]
new_data.plot(figsize=(10,10),kind="hexbin" ,y="loudness",x="mode")
plt.savefig('C:\Users\Acer\Desktop\ds\Music3.png',dpi=600)
plt.show()
```

OUTPUT:



Scatter Graph: We could also visualize using scatter graphs

```
import sys
import os
import pandas as pd
import matplotlib as
ml
from matplotlib import pyplot as plt
ml.style.use('ggplot')
data=[['Keep A Song In Your Soul',129.2,20.8],['I Put A Spell On
You',110.5,15.5],['Golfing Papa',305.9,15.2],['Head Off',16.2,9.2],['High
Rated',216.9,18.7],['Bang Bang',212.3,17.9]]
new_data=pd.DataFrame(data)
new_data.rename(columns={0:"Song Name"},inplace=True)
new_data.rename(columns={1:"loudness"},inplace=True)
new_data.rename(columns={2:"mode"},inplace=True)
colors_name=['blue','red','green','golden','pink','yellow']
explode=(0,0,0,0,0,0)
label= new_data["Song Name"]
new_data.plot(figsize=(10,10),kind="scatter" ,y="loudness",x="mode")
plt.savefig('C:\Users\Acer\Desktop\ds\Music4.png',dpi=600)
plt.show()
```

OUTPUT: