

Index

Sr. No	Practical Name	Date	Signature
1	Create Data Model using Cassandra.	09/10/2019	
2	Conversion from different formats to HORUS format.	14/10/2019	
3	Auditing through Logging.	14/10/2019	
4	Retrieving Data.	16/10/2019	
5	Assessing Data.	23/10/2019	
6	Processing Data.	04/11/2019	
7	Transforming Data.	06/11/2019	
8	Organizing Data.	13/11/2019	
9	Generating Reports.	13/11/2019	
10	Data Visualization with Power BI.	14/10/2019	

Practical No. 1

1. Create data model using Cassandra.

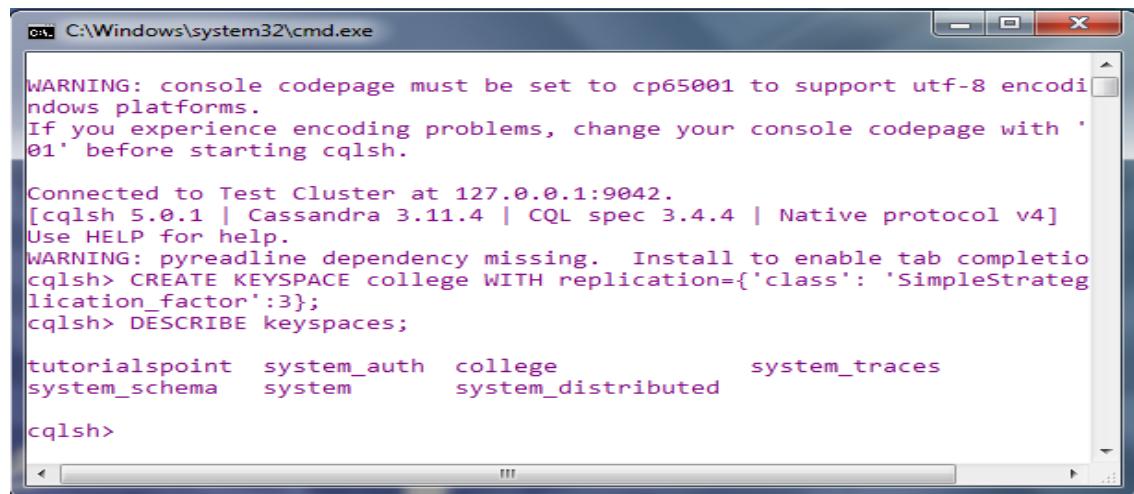
Required Software:

Java v1.8, Python v2.7, Cassandra File

ANS:

Create Key space:

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



```
C:\Windows\system32\cmd.exe
WARNING: console codepage must be set to cp65001 to support utf-8 encoding on windows platforms.
If you experience encoding problems, change your console codepage with 'chcp 65001' before starting cqlsh.

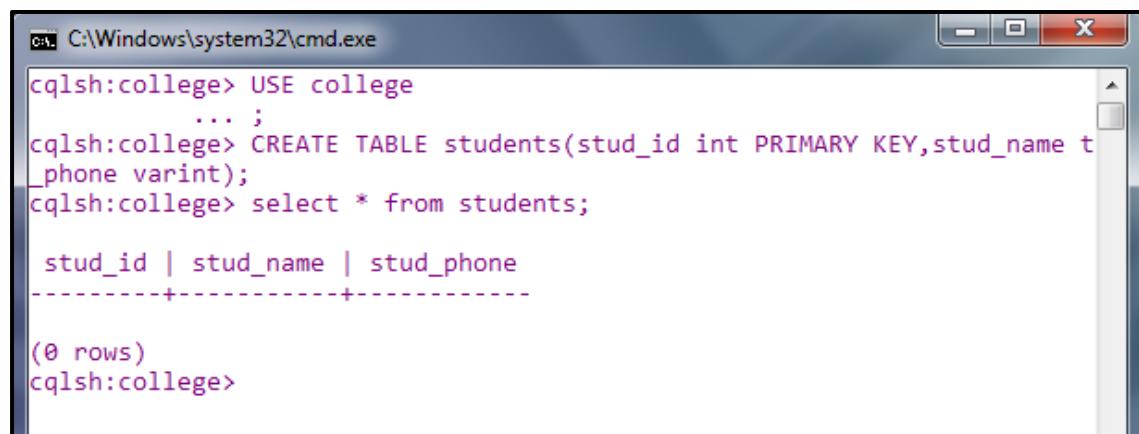
Connected to Test Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.11.4 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
WARNING: pyreadline dependency missing. Install to enable tab completion.
cqlsh> CREATE KEYSPACE college WITH replication={'class': 'SimpleStrategy', 'replication_factor':3};
cqlsh> DESCRIBE keyspaces;

tutorialspoint    system_auth    college          system_traces
system_schema     system        system_distributed
cqlsh>
```

Create Table

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

```
cqlsh:college> select * from students;
```



```
C:\Windows\system32\cmd.exe
cqlsh:college> USE college
...
cqlsh:college> CREATE TABLE students(stud_id int PRIMARY KEY,stud_name text,stud_phone varint);
cqlsh:college> select * from students;

stud_id | stud_name | stud_phone
-----+-----+-----
(0 rows)
cqlsh:college>
```

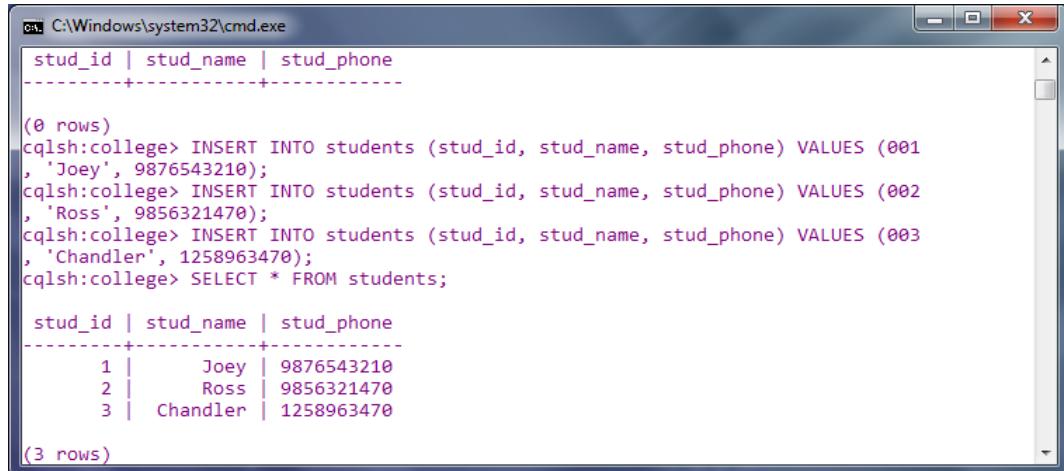
Creating Data in a Table

```
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;
```



The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The user has run several commands to insert data into a table named 'students' and then query it. The output shows three rows of data: (1, Joey, 9876543210), (2, Ross, 9856321470), and (3, Chandler, 1258963470).

```
stud_id | stud_name | stud_phone
-----+-----+
(0 rows)
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;

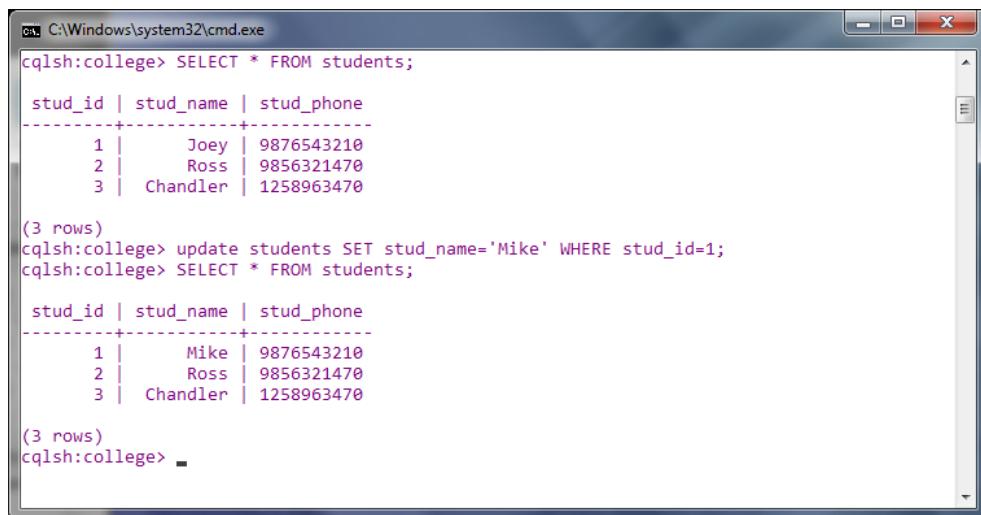
stud_id | stud_name | stud_phone
-----+-----+
    1 |      Joey | 9876543210
    2 |      Ross | 9856321470
    3 |   Chandler | 1258963470

(3 rows)
```

Update data in table

```
cqlsh:college> update students SET stud_name='Mike' WHERE stud_id=1;
```

```
cqlsh:college> SELECT * FROM students;
```



The screenshot shows a Windows command prompt window titled 'cmd C:\Windows\system32\cmd.exe'. The user has run commands to select all data from the 'students' table, update the name of the first row to 'Mike', and then select all data again to verify the change. The output shows the updated name 'Mike' for the first row.

```
cqlsh:college> SELECT * FROM students;
stud_id | stud_name | stud_phone
-----+-----+
    1 |      Joey | 9876543210
    2 |      Ross | 9856321470
    3 |   Chandler | 1258963470

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

stud_id | stud_name | stud_phone
-----+-----+
    1 |      Mike | 9876543210
    2 |      Ross | 9856321470
    3 |   Chandler | 1258963470

(3 rows)
cqlsh:college>
```

Delete data from table

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

The screenshot shows a command-line interface window titled 'cmd C:\Windows\system32\cmd.exe'. The session starts with a 'SELECT * FROM students;' query, which returns three rows of data:

stud_id	stud_name	stud_phone
1	Mike	9876543210
2	Ross	9856321470
3	Chandler	1258963470

Then, a 'DELETE stud_phone FROM students WHERE stud_id=2;' command is run, followed by another 'SELECT * FROM students;' query. The second query shows that the phone number for student ID 2 has been set to null.

stud_id	stud_name	stud_phone
1	Mike	9876543210
2	Ross	null
3	Chandler	1258963470

Alter Table

1. Adding a column

```
cqlsh:college> ALTER TABLE students
```

```
... ADD stud_add text;
```

```
cqlsh:college> SELECT * FROM students;
```

The screenshot shows a command-line interface window titled 'cmd C:\Windows\system32\cmd.exe'. The session starts with a 'SELECT * FROM students;' query, which returns three rows of data:

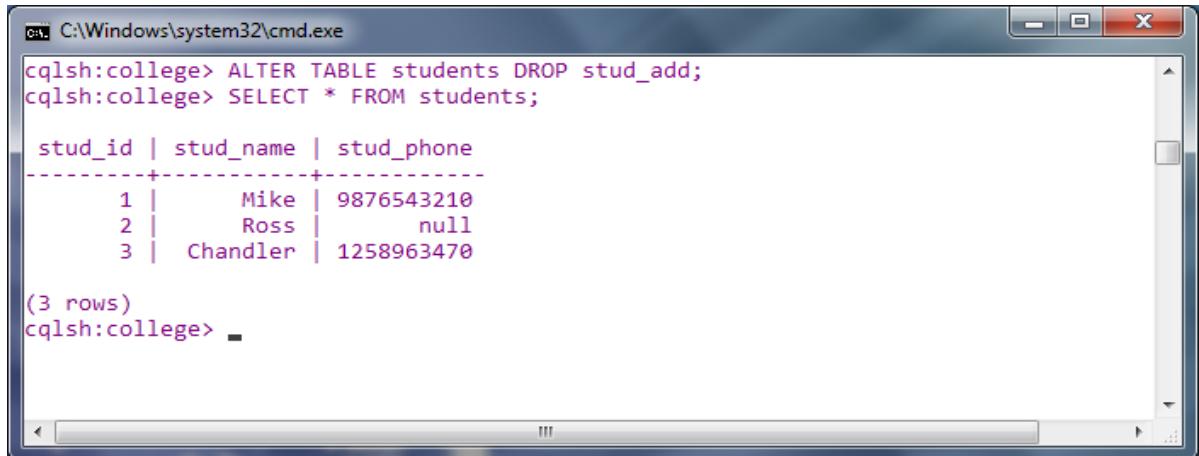
stud_id	stud_name	stud_phone
1	Mike	9876543210
2	Ross	null
3	Chandler	1258963470

Then, an 'ALTER TABLE students ... ADD stud_add text;' command is run, followed by another 'SELECT * FROM students;' query. The second query shows that a new column 'stud_add' has been added to the table, and it contains null values for all three rows.

stud_id	stud_add	stud_name	stud_phone
1	null	Mike	9876543210
2	null	Ross	null
3	null	Chandler	1258963470

2. Dropping a column

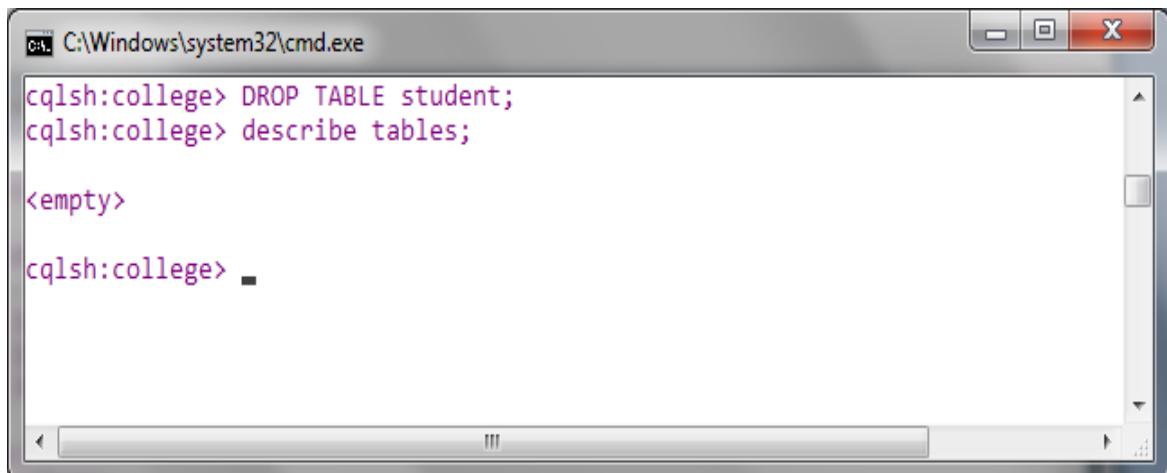
```
cqlsh:college> ALTER TABLE students DROP stud_add;  
cqlsh:college> SELECT * FROM students;
```



```
C:\ C:\Windows\system32\cmd.exe  
cqlsh:college> ALTER TABLE students DROP stud_add;  
cqlsh:college> SELECT * FROM students;  
  
stud_id | stud_name | stud_phone  
-----+-----+-----  
    1 |      Mike | 9876543210  
    2 |       Ross |      null  
    3 | Chandler | 1258963470  
  
(3 rows)  
cqlsh:college> -
```

Drop Table

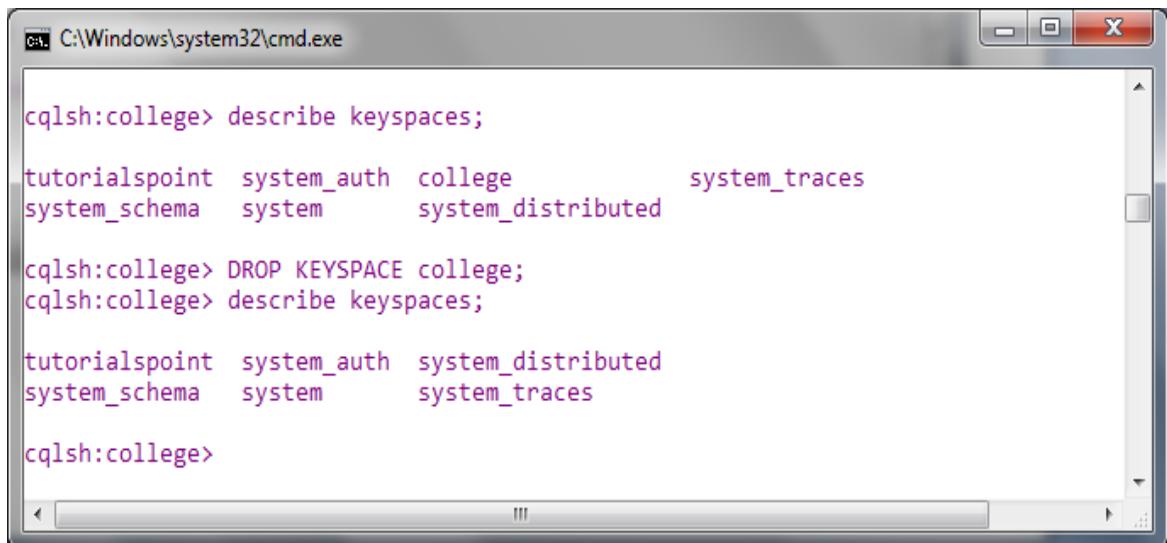
```
cqlsh:college> DROP TABLE student;  
cqlsh:college> describe tables;
```



```
C:\ C:\Windows\system32\cmd.exe  
cqlsh:college> DROP TABLE student;  
cqlsh:college> describe tables;  
  
<empty>  
  
cqlsh:college> -
```

Drop Keyspace

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



The screenshot shows a Windows command-line interface (cmd.exe) window titled 'C:\Windows\system32\cmd.exe'. Inside the window, a CQLSH session is running. The user has entered the following commands:

```
cqlsh:college> describe keyspaces;  
tutorialspoint  system_auth  college          system_traces  
system_schema   system       system_distributed  
  
cqlsh:college> DROP KEYSPACE college;  
cqlsh:college> describe keyspaces;  
  
tutorialspoint  system_auth  system_distributed  
system_schema   system       system_traces  
  
cqlsh:college>
```

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:

```
Python 3.7.4 Shell
File Edit Shell Debug Options Window Help
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC v.1916 32 bit
(Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
RESTART: C:\practical-data-science-master\VKHCG\05-DS\9999-Data\CSV2HORUS.py
Input Data Values =====
   Country ISO-2-CODE ISO-3-Code ISO-M49
0      USA           1          3     248
1    India           2          7     264
2  England           3          5     102
3   Russia           4          6     231
=====
Process Data Values =====
   CountryName
CountryNumber
 248          USA
 231         Russia
 264        India
102       England
=====
CSV to HORUS - Done
>>> |
Ln: 21 Col: 4
```

CSV TO AUDIO

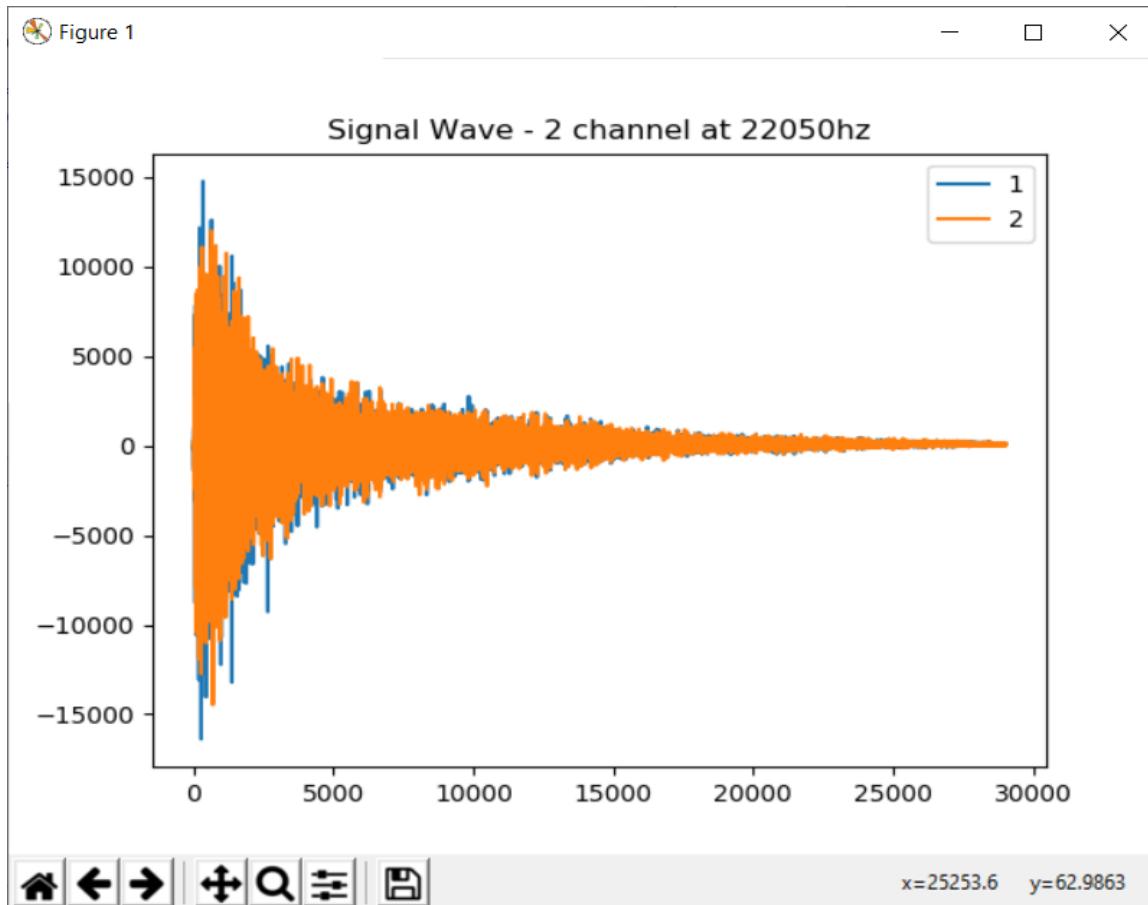
INPUT:

```
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:

```
Python 3.7.4 Shell
File Edit Shell Debug Options Window Help
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 19:29:22) [MSC v.1916
32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
RESTART: C:/practical-data-science-master/VKHCG/05-DS/9999-Data/CSV2HOR
US_Log.py
Input Data Values =====
Country ISO-2-CODE ISO-3-Code ISO-M49
0 USA 1 3 248
1 India 2 7 264
2 England 3 5 102
3 Russia 4 6 231
=====
now = 2019-10-14 10:02:43.792968
Date and Time= 14/10/19 10:02:43
Process Data Values =====
CountryName
CountryNumber
248 USA
231 Russia
264 India
102 England
=====
CSV to HORUS - Done
>>>
```

```
Country_Code_Log - Notepad
File Edit Format View Help
Delete column activity recorded at 141019 10:00:51
Delete column activity recorded at 14/10/19 10:02:43
```

Practical No. 4

Retrieving the data.

CSV to SQL

INPUT:

```
import sys
import os
import sqlite3 as sq
import pandas as pd

sDatabaseName="C:\\Users\\Administrator\\Desktop\\example.db"
conn = sq.connect(sDatabaseName)
sFileName='C:/practical-data-science-master/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")

IP_DATA_ALL.index.names = ['RowIDCSV']
sTable='IP_DATA_ALL'

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

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>>
RESTART: C:\practical-data-science-master\VKHCG\01-Vermeulen\01-Retrieve\Retrieve-
IP_DATA_ALL_2_SQLite.py
Loading : C:/practical-data-science-master/VKHCG/01-Vermeulen/00-RawData/IP_DATA_AL
L.csv
Storing : C:\Users\Administrator\Desktop\example.db Table: IP_DATA_ALL
Loading : C:\Users\Administrator\Desktop\example.db Table: IP_DATA_ALL
#####
## Data Values
#####
    RowIDCSV      Country          Capital
0            0       India     New Delhi
1            1        USA  Washington D.C.
2            2        UK         London
3            3   Srilanka      Colombo
4            4       Japan        Tokyo
5            5       UAE         Dubai
6            6       China      Bejieng
7            7       France        Paris
8            8       Italy       Venice
9            9      Canada       Toronto
#####
## Data Profile
#####
Rows : 10
Columns : 3
#####
### Done!! #####
>>>
Ln: 29 Col: 4
```

DB Browser for SQLite - C:\Users\Administrator\Desktop\example.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragmas Execute SQL

Table: IP_DATA_ALL New Record Delete Record

RowIDCSV	Country	Capital
1 0	India	New Delhi
2 1	USA	Washington D...
3 2	UK	London
4 3	Srilanka	Colombo
5 4	Japan	Tokyo
6 5	UAE	Dubai
7 6	China	Bejieng
8 7	France	Paris
9 8	Italy	Venice
10 9	Canada	Toronto

1 - 10 of 10 Go to: 1

Edit Database Cell Mode: Text Import Export Set as NULL

Type of data currently in cell: Text / Numeric 1 char(s) Apply

Remote Identity

Name Commit Last modified Size

SQL Log Plot DB Schema Remote

UTF-8

Practical No. 5

Assessing the Data.

1. Drop a column - parameter = "how"

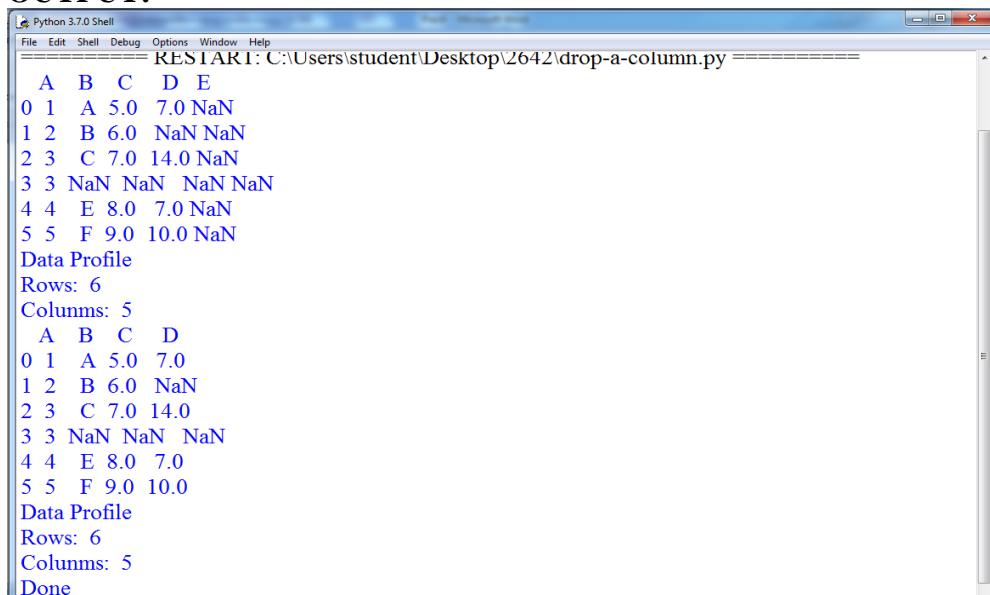
INPUT:

```
import sys
import os
import pandas as pd
sInputFile = "Good-or-Bad.csv"
sOutputFile="Good-orBaad-01.csv"
RawData=pd.read_csv("c:/Users/student/Desktop/2642/Good-or-Bad.csv",header=0)
print(RawData)
print("Data Profile")
print("Rows: ",RawData.shape[0])
print("Columns: ",RawData.shape[1])
TestData=RawData.dropna(axis=1, how="all")
print(TestData)
print("Data Profile")
print("Rows: ",RawData.shape[0])
print("Columns: ",RawData.shape[1])
TestData.to_csv("C:/Users/student/Desktop/2642/Good-or-Baad-01.csv",index=False)
print("Done")
```

Good-or-Bad.csv-

A	B	C	D	E
1	A		5	7
2	B		6	
3	C		7	14
3				
4	E		8	7
5	F		9	10

OUTPUT:



The screenshot shows the Python 3.7.0 Shell window with the following output:

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
===== RESTART: C:\Users\student\Desktop\2642\drop-a-column.py ======
A B C D E
0 1 A 5.0 7.0 NaN
1 2 B 6.0 NaN NaN
2 3 C 7.0 14.0 NaN
3 3 NaN NaN NaN NaN
4 4 E 8.0 7.0 NaN
5 5 F 9.0 10.0 NaN
Data Profile
Rows: 6
Columns: 5
A B C D
0 1 A 5.0 7.0
1 2 B 6.0 NaN
2 3 C 7.0 14.0
3 3 NaN NaN NaN
4 4 E 8.0 7.0
5 5 F 9.0 10.0
Data Profile
Rows: 6
Columns: 5
Done
```

2. Drop a column – parameter= “any”

INPUT:

```
import sys
import os
import pandas as pd
sInputFile = "Good-or-Bad.csv"
sOutputFile="Good-orBaad-01.csv"
RawData=pd.read_csv("c:/Users/student/Desktop/2642/Good-or-Bad.csv",header=0)
print(RawData)
print("Data Profile")
print("Rows: ",RawData.shape[0])
print("Columms: ",RawData.shape[1])
TestData=RawData.dropna(axis=1,how="any")
print(TestData)
print("Data Profile")
print("Rows: ",RawData.shape[0])
print("Columms: ",RawData.shape[1])
TestData.to_csv("C:/Users/student/Desktop/2642/Good-or-Baad-01.csv",index=False)
print("Done")
```

OUTPUT:

The screenshot shows the Python 3.7.0 Shell window. The title bar reads "Python 3.7.0 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main window displays the output of a script named "drop-a-column.py". The script starts by reading a CSV file ("Good-or-Bad.csv") with 6 rows and 5 columns. It then prints the "Data Profile" showing 6 rows and 5 columns. It drops the first column ("A") using the "dropna" method with "axis=1" and "how='any'". The resulting DataFrame has 6 rows and 4 columns. This modified DataFrame is then printed again, followed by its "Data Profile" which shows 6 rows and 4 columns. Finally, the script outputs "Done".

```
A   B   C   D   E
0 1  A  5.0  7.0  NaN
1 2  B  6.0  NaN  NaN
2 3  C  7.0  14.0 NaN
3 3  NaN  NaN  NaN  NaN
4 4  E  8.0  7.0  NaN
5 5  F  9.0  10.0 NaN
Data Profile
Rows: 6
Columns: 5
A
0 1
1 2
2 3
3 3
4 4
5 5
Data Profile
Rows: 6
Columns: 5
Done
```

3. Filing the missing using mean(), min(), max() method:

i. Using mean():

INPUT:

```
import sys
import os
import pandas as pd
sInputFile = "Good-or-Bad.csv"
sOutputFile="Good-orBaad-01.csv"
RawData=pd.read_csv("c:/Users/student/Desktop/2642/Good-or-Bad.csv",header=0)
print(RawData)
print("Data Profile")
print("Rows: ",RawData.shape[0])
print("Columms: ",RawData.shape[1])
TestData=RawData.fillna(RawData.mean())
print(TestData)
print("Data Profile")
print("Rows: ",RawData.shape[0])
print("Columms: ",RawData.shape[1])
TestData.to_csv("C:/Users/student/Desktop/2642/Good-or-Baad-01.csv",index=False)
print("Done")
```

OUTPUT:

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
===== RESTART: C:\Users\student\Desktop\2642\drop-a-column.py ======
      A   B   C   D   E
0 1  A  5.0  7.0  NaN
1 2  B  6.0  NaN  NaN
2 3  C  7.0  14.0  NaN
3 3  NaN  NaN  NaN  NaN
4 4  E  8.0  7.0  NaN
5 5  F  9.0  10.0  NaN
Data Profile
Rows: 6
Columns: 5
      A   B   C   D   E
0 1  A  5.0  7.0  NaN
1 2  B  6.0  9.5  NaN
2 3  C  7.0  14.0  NaN
3 3  NaN  7.0  9.5  NaN
4 4  E  8.0  7.0  NaN
5 5  F  9.0  10.0  NaN
Data Profile
Rows: 6
Columns: 5
Done
>>>
```

ii. Using min():

INPUT:

```
import sys
import os
import pandas as pd
sInputFile = "Good-or-Bad.csv"
sOutputFile="Good-orBaad-01.csv"
RawData=pd.read_csv("c:/Users/student/Desktop/2642/Good-or-Bad.csv",header=0)
print(RawData)
print("Data Profile")
print("Rows: ",RawData.shape[0])
print("Columns: ",RawData.shape[1])
TestData=RawData.fillna(RawData.min())
print(TestData)
print("Data Profile")
print("Rows: ",RawData.shape[0])
print("Columns: ",RawData.shape[1])
TestData.to_csv("C:/Users/student/Desktop/2642/Good-or-Baad-01.csv",index=False)
print("Done")
```

OUTPUT:

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
=====
RESTART: C:\Users\student\Desktop\2642\drop-a-column.py =====
      A   B   C   D   E
0 1   A  5.0  7.0  NaN
1 2   B  6.0  NaN  NaN
2 3   C  7.0  14.0 NaN
3 3   NaN  NaN  NaN  NaN
4 4   E  8.0  7.0  NaN
5 5   F  9.0  10.0 NaN
Data Profile
Rows: 6
Columns: 5
      A   B   C   D   E
0 1   A  5.0  7.0  NaN
1 2   B  6.0  7.0  NaN
2 3   C  7.0  14.0 NaN
3 3   NaN  5.0  7.0  NaN
4 4   E  8.0  7.0  NaN
5 5   F  9.0  10.0 NaN
Data Profile
Rows: 6
Columns: 5
Done
```

iii. Using max():

INPUT:

```
import sys
import os
import pandas as pd
sInputFile = "Good-or-Bad.csv"
sOutputFile="Good-orBaad-01.csv"
RawData=pd.read_csv("c:/Users/student/Desktop/2642/Good-or-Bad.csv",header=0)
print(RawData)
print("Data Profile")
print("Rows: ",RawData.shape[0])
print("Columns: ",RawData.shape[1])
TestData=RawData.fillna(RawData.max())
print(TestData)
print("Data Profile")
print("Rows: ",RawData.shape[0])
print("Columns: ",RawData.shape[1])
TestData.to_csv("C:/Users/student/Desktop/2642/Good-or-Baad-01.csv",index=False)
print("Done")
```

OUTPUT:

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
===== RESTART: C:\Users\student\Desktop\2642\drop-a-column.py =====
      A   B   C   D   E
0 1  A  5.0  7.0  NaN
1 2  B  6.0  NaN  NaN
2 3  C  7.0  14.0  NaN
3 3  NaN  NaN  NaN  NaN
4 4  E  8.0  7.0  NaN
5 5  F  9.0  10.0  NaN
Data Profile
Rows: 6
Columns: 5
      A   B   C   D   E
0 1  A  5.0  7.0  NaN
1 2  B  6.0  14.0  NaN
2 3  C  7.0  14.0  NaN
3 3  NaN  9.0  14.0  NaN
4 4  E  8.0  7.0  NaN
5 5  F  9.0  10.0  NaN
Data Profile
Rows: 6
Columns: 5
Done
>>> |
```

Practical No. 6

Processing Data

Process Location

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

```

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

```

OUTPUT:

The screenshot shows the DB Browser for SQLite interface. The main window displays the 'Process-Location' table with the following data:

IDNumber	ObjectBaseKey	LocationNumber	LocationName	Longitude	Latitude
1	4c010c6b-d6a...	GPS	1	L-180000--18...	-180
2	30ebcb4c-fe9...	GPS	2	L-180000--18...	-180
3	d15c43ed-599...	GPS	3	L-180000--18...	-180
4	45f53fa7-7bd...	GPS	4	L-180000--18...	-180
5	b6919bce-7d6...	GPS	5	L-180000--18...	-180
6	30b14bae-91f...	GPS	6	L-180000--18...	-180
7	df928294-16f...	GPS	7	L-180000--18...	-180
8	59cb711e-43d...	GPS	8	L-180000--18...	-180
9	dfdf0443-449...	GPS	9	L-180000--18...	-180
10	ef86406f-eb7...	GPS	10	L-180000--18...	-180
11	c9c5077e-a3b...	GPS	11	L-180000--18...	-180
12	1e61da0d-14...	GPS	12	L-180000--18...	-180
13	ffdbbf48-3672...	GPS	13	L-180000--18...	-180
14	4a5c211e-069...	GPS	14	L-180000--18...	-180
15	5f7410d2-33d...	GPS	15	L-180000--18...	-180
16	e9ee121b-a3...	GPS	16	L-180000--18...	-180
17	6f83b7b4-01a...	GPS	17	L-180000--18...	-180
18	08ffb892-98f0...	GPS	18	L-180000--18...	-180

The right side of the interface shows the 'Edit Database Cell' panel with the current cell set to 'Text' mode, containing the value '1'. Below it is the 'Remote' section and a file browser.

Process Event

```
import sys
import os
import pandas as pd
import sqlite3 as sq
from pandas.io import sql
Base='C:/VKHCG'
print('Working Base :',Base, ' using ', sys.platform)
Company='01-Vermeulen'
InputFileName='Action_Plan.csv'
sDataBaseDir=Base + '/' + Company + '/03-Process/SQLite'
if not os.path.exists(sDataBaseDir):
    os.makedirs(sDataBaseDir)
sDatabaseName=sDataBaseDir + '/Vermeulen.db'
conn1 = sq.connect(sDatabaseName)
sDataVaultDir=Base + '/88-DV'
if not os.path.exists(sDataVaultDir):
    os.makedirs(sDataVaultDir)
sDatabaseName=sDataVaultDir + '/datavault.db'
conn2 = sq.connect(sDatabaseName)
sFileName=Base + '/' + Company + '/00-RawData/' + InputFileName
print('Loading :',sFileName)
EventRawData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
EventRawData.index.names=['EventID']
EventHubIndex=EventRawData
sTable = 'Process-Event'
print('Storing :',sDatabaseName,' Table:',sTable)
EventHubIndex.to_sql(sTable, conn1, if_exists="replace")
sTable = 'Hub-Event'
print('Storing :',sDatabaseName,' Table:',sTable)
#EventHubIndex.to_sql(sTable, conn2, if_exists="replace")
print('#####')
print('Vacuum Databases')
sSQL="VACUUM;"
sql.execute(sSQL,conn1)
#sql.execute(sSQL,conn2)
print('## Done!! #####')
```

OUTPUT:

The screenshot shows the DB Browser for SQLite interface. The main window displays a table named "Process-Event" with three columns: EventID, Milestone, Year, and Action. The table contains 18 rows of data. The "Edit Database Cell" dialog is open over the first row (EventID 1), showing the value "1". The "Remote" panel is also visible on the right side of the interface.

EventID	Milestone	Year	Action
1	0		0,0,Born
2	1		1,15,First kiss
3	2		2,19,First full-time job
4	3		3,20,Pass driving test
5	4		4,21,First holiday with friends
6	5		5,22,Move out/rent with friends
7	6		6,22,Buy first car
8	7		7,23,First holiday with a partner
9	8		8,23,Be a bridesmaid/best man
10	9		9,24,Rent on your own
11	10		10,25,Get engaged
12	11		11,25,Rent with partner
13	12		12,27,Get married
14	13		13,27,Buy first flat
15	14		14,28,Have first child
16	15		15,29,First house
17	16		16,30,Start earning average wage
18	17		17,31,Second child

Practical No. 7

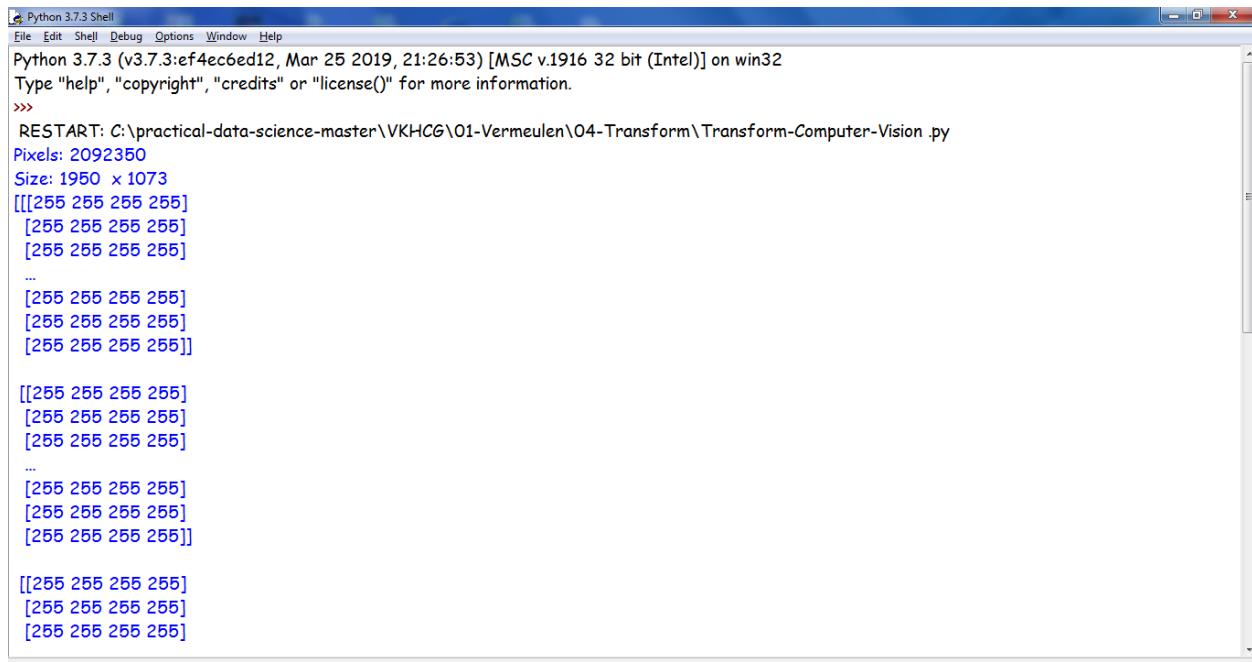
Transforming Data.

INPUT:

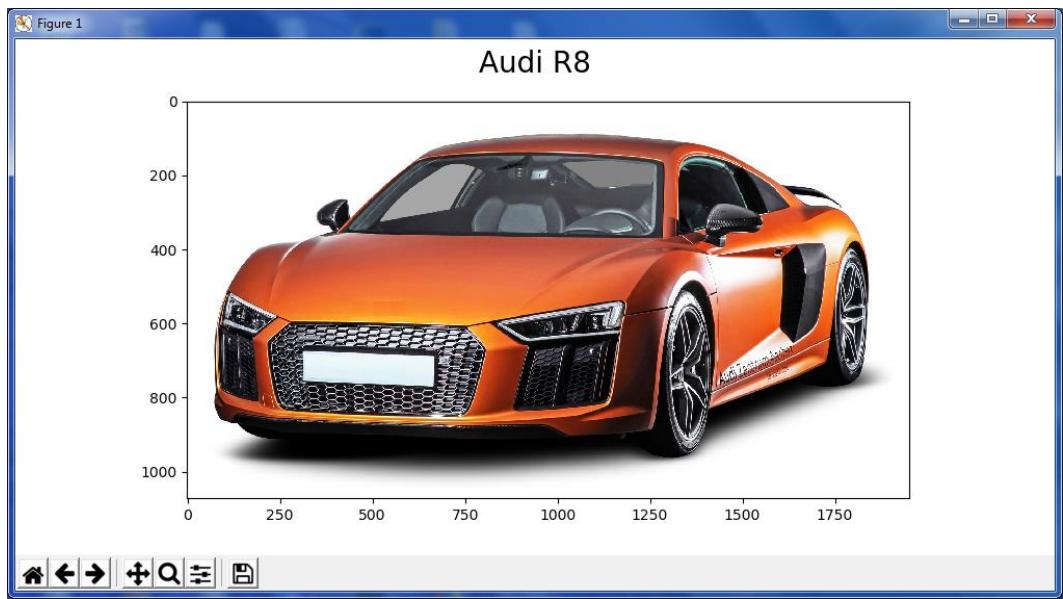
```
import matplotlib.pyplot as plt
from PIL import Image
import numpy as np
sPicNameIn='C:/practical-data-science-master/VKHCG/01-Vermeulen/00-
RawData/AudiR8.png'
imageIn = Image.open(sPicNameIn)
fig1=plt.figure(figsize=(10, 10))
fig1.suptitle('Audi R8', fontsize=20)
imgplot = plt.imshow(imageIn)
plt.show()

imagewidth, imageheight = imageIn.size
imageMatrix=np.asarray(imageIn)
pixelscnt = (imagewidth * imageheight)
print('Pixels:', pixelscnt)
print('Size:', imagewidth, 'x', imageheight,)
print(imageMatrix)
```

OUTPUT:



```
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
RESTART: C:\practical-data-science-master\VKHCG\01-Vermeulen\04-Transform\Transform-Computer-Vision.py
Pixels: 2092350
Size: 1950 x 1073
[[[255 255 255 255]
 [255 255 255 255]
 [255 255 255 255]
 ...
 [255 255 255 255]
 [255 255 255 255]
 [255 255 255 255]]
 [[255 255 255 255]
 [255 255 255 255]
 [255 255 255 255]
 ...
 [255 255 255 255]
 [255 255 255 255]
 [255 255 255 255]]
 [[255 255 255 255]
 [255 255 255 255]
 [255 255 255 255]
 ...
 [255 255 255 255]
 [255 255 255 255]
 [255 255 255 255]]]
```



Transform.py

INPUT:

```
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='D:/Data Science/Vermeulen.db'
conn1 = sq.connect(sDatabaseName)
sDatabaseName='D:/Data Science/datavault.db'
conn2 = sq.connect(sDatabaseName)
sDatabaseName='D:/Data Science/datawarehouse.db'
conn3 = sq.connect(sDatabaseName)
print('\n#####')
print('Time Category')
print('UTC Time')
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)")
print(BirthDateZoneUTCStr)
print('#####')

print('Birth Date in Reykjavik :')
BirthZone = 'Atlantic/Reykjavik'
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")
print(BirthDateStr)
print('#####')
#####
IDZoneNumber=str(uuid.uuid4())
sDateTimeKey=BirthDateZoneStr.replace(' ','-').replace(':','-')
TimeLine=[('ZoneBaseKey', ['UTC']),
          ('IDNumber', [IDZoneNumber]),
          ('DateTimeKey', [sDateTimeKey]),
          ('UTCDateTimeValue', [BirthDateZoneUTC]),
          ('Zone', [BirthZone]),
          ('DateTimeValue', [BirthDateStr])]

TimeFrame = pd.DataFrame.from_items(TimeLine)
#####
TimeHub=TimeFrame[['IDNumber','ZoneBaseKey','DateTimeKey','DateTimeValue']]
TimeHubIndex=TimeHub.set_index(['IDNumber'],inplace=False)
```

```

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

```

OUTPUT:

Transform Hub-Person-Gunnarsson Vermulan.db (1st)

The screenshot shows the DB Browser for SQLite interface with the database 'datavault.db' open. The 'Hub-Person-Gunnarsson' table is selected. The table has five columns: IDNumber, FirstName, LastName, Zone, and DateTimeValue. A single row is visible with values: 1, abc, pqr, UTC, and 1960-12-20 1... respectively. The 'Edit Database Cell' dialog is open over the last column, showing the value 'a3abe7af-07cf-483f-a005-ea2064475bc6'. The 'Mode' dropdown is set to 'Text'. The 'Remote' panel on the right shows a list of files with their names, commits, last modified dates, and sizes.

Transform Hub-Time-Gunnarsson

The screenshot shows the DB Browser for SQLite interface with the database 'datavault.db' open. The 'Hub-Time-Gunnarsson' table is selected. The table has four columns: IDNumber, ZoneBaseKey, DateTimeKey, and DateTimeValue. A single row is visible with values: 1, 1810da40-2ee0-45ac-b4ef-25fe8e4af696, UTC, and 1960-12-20-10-15... respectively. The 'Edit Database Cell' dialog is open over the fourth column, showing the value '1810da40-2ee0-45ac-b4ef-25fe8e4af696'. The 'Mode' dropdown is set to 'Text'. The 'Remote' panel on the right shows a list of files with their names, commits, last modified dates, and sizes.

Transform Hub-Person-Gunnarsson

The screenshot shows the DB Browser for SQLite interface. The main window displays a table named 'Satellite-Time-Atlantic-' with four columns: IDNumber, DateTimeKey, Zone, and DateTimeValue. A single row is selected, showing values: 1, 1810da40-2e..., 1960-12-20-1..., and Atlantic/Reykj.... To the right, an 'Edit Database Cell' dialog is open over the table view. The dialog has a text input field containing the value '1810da40-2ee0-45ac-b4ef-25fe8e4af696'. Below the input field, it says 'Type of data currently in cell: Text / Numeric 36 char(s)'. There are buttons for 'Mode: Text' and 'Apply'. In the bottom right corner of the dialog, there is a message: 'Activate Windows Go to Settings to activate Windows' and 'UTF-8'.

Sun_model.py (Data Warehouse)

INPUT:

```
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='D:/Data Science/Vermeulen.db'
conn1 = sq.connect(sDatabaseName)
sDatabaseName='D:/Data Science/datavault.db'
conn2 = sq.connect(sDatabaseName)
sDatabaseName='D:/Data Science/datawarehouse.db'
conn3 = sq.connect(sDatabaseName)
print("\n#####")
sSQL=" SELECT DateTimeValue FROM [Hub-Time-Gunnarsson];"
DateDataRaw=pd.read_sql_query(sSQL, conn2)
DateData=DateDataRaw.head(1000)
print(DateData)
```

```

print('Time Dimension')
print("\n#####")
t=0
mt=DateData.shape[0]
for i in range(mt):
    BirthZone = ('Atlantic/Reykjavik','Europe/London','UCT')
    for j in range(len(BirthZone)):
        t+=1
        print(t,mt*3)
        BirthDateZoneStr=DateData[DateTimeKey]
        BirthDateLocal=DateData[DateTimeValue]
        BirthZone='UCT'

        IDTimeNumber=str(uuid.uuid4())
        TimeLine=[('TimeID', [str(IDTimeNumber)]),
                  ('UTCDate', [str(BirthDateZoneStr)]),
                  ('LocalTime', [str(BirthDateLocal)]),
                  ('TimeZone', [str(BirthZone)])]

        if t==1:
            TimeFrame = pd.DataFrame.from_items(TimeLine)
        else:
            TimeRow = pd.DataFrame.from_items(TimeLine)
            TimeFrame=TimeFrame.append(TimeRow)

        DimTime=TimeFrame
        DimTimeIndex=DimTime.set_index(['TimeID'],inplace=False)
        sTable = 'Dim-Time'
        print("\n#####")
        print('Storing :',sDatabaseName,' Table:',sTable)
        print("\n#####")
        DimTimeIndex.to_sql(sTable, conn1, if_exists="replace")
        DimTimeIndex.to_sql(sTable, conn3, if_exists="replace")
        sSQL=" SELECT " + \
              " FirstName," + \
              " SecondName," + \
              " LastName," + \
              " BirthDateKey " + \
              " FROM [Hub-Person];"

        PersonDataRaw=pd.read_sql_query(sSQL, conn2)
        PersonData=PersonDataRaw.head(1000)
        print("\n#####")
        print('Dimension Person')
        print("\n#####")
        t=0
        mt=DateData.shape[0]
        for i in range(mt):
            t+=1
            print(t,mt)
            FirstName = str(PersonData["FirstName"])
            SecondName = str(PersonData["SecondName"])
            if len(SecondName) > 0:

```

```

SecondName=""
LastName = str(PersonData["LastName"])
BirthDateKey = str(PersonData["BirthDateKey"])
IDPersonNumber=str(uuid.uuid4())
PersonLine=[('PersonID', [str(IDPersonNumber)]),
            ('FirstName', [FirstName]),
            ('SecondName', [SecondName]),
            ('LastName', [LastName]),
            ('Zone', [str('UTC')]),
            ('BirthDate', [BirthDateKey])]

if t==1:
    PersonFrame = pd.DataFrame.from_items(PersonLine)
else:
    PersonRow = pd.DataFrame.from_items(PersonLine)
    PersonFrame = PersonFrame.append(PersonRow)

DimPerson=PersonFrame
print(DimPerson)
DimPersonIndex=DimPerson.set_index(['PersonID'],inplace=False)
sTable = 'Dim-Person'
print('\n#####')
print('Storing :',sDatabaseName,' Table:',sTable)
DimPersonIndex.to_sql(sTable, conn1, if_exists="replace")
DimPersonIndex.to_sql(sTable, conn3, if_exists="replace")

```

OUTPUT:

Dim-Hub-Person-Gunnarsson

The screenshot shows the DB Browser for SQLite interface with the following details:

- Database Structure:** Shows the table structure with columns: IDNumber, FirstName, LastName, Zone, and DateTimeValue.
- Data View:** Displays a single record with values: IDNumber=a3abe7af-07c..., FirstName=abc, LastName=pqr, Zone=UTC, and DateTimeValue=1960-12-20 1...
- Cell Editor:** An open editor for the first row, first column (IDNumber) showing the value '1'.
- Toolbar:** Includes buttons for New Database, Open Database, Write Changes, Revert Changes, Open Project, Save Project, Attach Database, and Close Database.
- Mode:** Set to Text.
- Buttons:** Import, Export, Set as NULL.
- Message Bar:** Shows "Type of data currently in cell: Text / Numeric" and "1 char(s)" with an Apply button.
- Remote Panel:** Shows a table with columns Name, Commit, Last modified, and Size.
- Bottom Navigation:** Includes Go to: input field with value '1', navigation icons (back, forward, search), and status bar with tabs SQL Log, Plot, DB Schema, Remote, and Activate Windows.

Dim- Atlantic-reyl

The screenshot shows the DB Browser for SQLite interface with the database file D:\Data Science\datawarehouse.db open. The main window displays the 'Dim-Time-Atlantic-Reyl' table with the following data:

IDNumber	DateTimeKey	Zone	DateTimeValue
1	1810da40-2e...	1960-12-20-1...	Atlantic/Reykj...

The 'Edit Database Cell' dialog is open over the first row, specifically for the 'IDNumber' column, which contains the value '1'. The 'Mode' dropdown is set to 'Text'. The 'Remote' pane on the right shows a single entry named 'Identity'.

Dim- Hub-Time-Gunnarsson

The screenshot shows the DB Browser for SQLite interface with the database file D:\Data Science\datawarehouse.db open. The main window displays the 'Dim-Time-Gunnarsson' table with the following data:

IDNumber	ZoneBaseKey	DateTimeKey	DateTimeValue
1	1810da40-2e...	UTC	1960-12-20-1...

The 'Edit Database Cell' dialog is open over the first row, specifically for the 'IDNumber' column, which contains the value '1'. The 'Mode' dropdown is set to 'Text'. The 'Remote' pane on the right shows a single entry named 'Identity'.

Practical No. 8

Organizing Data

ORGANISE

Organise-Network-Routing-Company

```
import sys
import os
import pandas as pd
import networkx as nx
import matplotlib.pyplot as plt
pd.options.mode.chained_assignment = None
sFileName='C:\\\\Users\\\\Administrator\\\\Downloads\\\\practical-data-science-master\\\\practical-
data-science-master\\\\VKHCG\\\\01-Vermeulen\\\\02-Assess\\\\01-EDS\\\\02-Python\\\\Assess-
Network-Routing-Company.csv'
print('Loading :',sFileName)
CompanyData=pd.read_csv(sFileName,header=0,low_memory=False, encoding="latin-1")
print(CompanyData.head())
print(CompanyData.shape)
G=nx.Graph()
for i in range(CompanyData.shape[0]):
    for j in range(CompanyData.shape[0]):
        Node0=CompanyData['Company_Country_Name'][i]
        Node1=CompanyData['Company_Country_Name'][j]
        if Node0 != Node1:
            G.add_edge(Node0,Node1)
for i in range(CompanyData.shape[0]):
    Node0=CompanyData['Company_Country_Name'][i]
    Node1=CompanyData['Company_Place_Name'][i] + '('+
CompanyData['Company_Country_Name'][i] + ')'
    if Node0 != Node1:
        G.add_edge(Node0,Node1)
print('Nodes:', G.number_of_nodes())
print('Edges:', G.number_of_edges())
sFileName='C:\\\\Users\\\\Administrator\\\\Downloads\\\\practical-data-science-master\\\\practical-
data-science-master\\\\VKHCG\\\\01-Vermeulen\\\\02-Assess\\\\01-EDS\\\\02-Python\\\\Assess-
Network-Routing-Company.csv'
print('Storing :',sFileName)
#nx.write_gml(G, sFileName)
sFileName='C:\\\\Users\\\\Administrator\\\\Downloads\\\\practical-data-science-master\\\\practical-
data-science-master\\\\VKHCG\\\\01-Vermeulen\\\\02-Assess\\\\01-EDS\\\\02-Python\\\\Assess-
Network-Routing-Company.csv'
print('Storing Graph Image:',sFileName)
plt.figure(figsize=(15, 15))
pos=nx.spectral_layout(G,dim=2)
nx.draw_networkx_nodes(G,pos, node_color='k', node_size=10, alpha=0.8)
nx.draw_networkx_edges(G, pos, edge_color='r', arrows=False, style='dashed')
nx.draw_networkx_labels(G, pos, font_size=12, font_family='sans-serif', font_color='b')
```

```

plt.axis('off')
#plt.savefig(sFileName,dpi=600)
plt.show()
print('### Done!! #####')

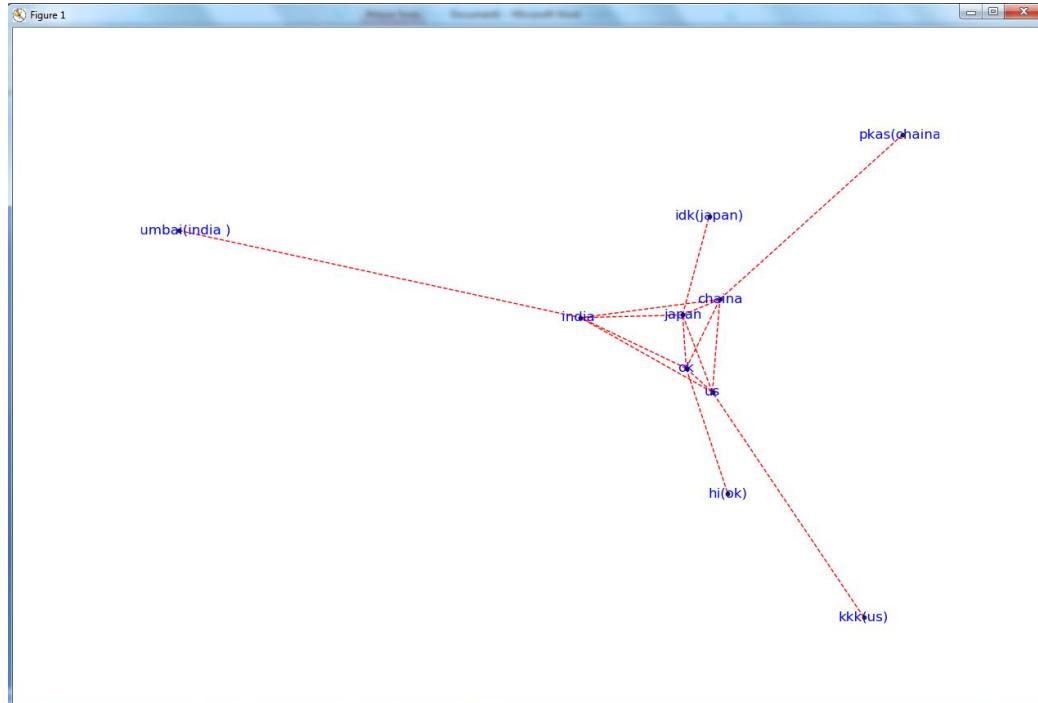
```

OUTPUT

```

#####
('Loading :', 'C:\\\\Users\\\\Administrator\\\\Download\\
ork-Routing-Company.csv')
#####
#####
    Company_Country_Name Company_Place_Name
0          india             mumbai
1          japan              idk
2            us                kkk
3        chaina              pkas
4           ok                hi
(5, 2)
('Nodes:', 10)
('Edges:', 15)
#####
('Storing :', 'C:\\\\Users\\\\Administrator\\\\Download\\
ork-Routing-Company.csv')
#####
#####
('Storing Graph Image:', 'C:\\\\Users\\\\Administrator\\\\
Assess-Network-Routing-Company.csv')
#####

```



Organize horizontal

```
import sys
import os
import pandas as pd
import sqlite3 as sq
sDatabaseName='C:/Users/student/Downloads/datawarehouse.db'
conn1 = sq.connect(sDatabaseName)
sDatabaseName= 'C:/Users/student/Downloads/datamart.db'
conn2 = sq.connect(sDatabaseName)
sTable = 'Dim-BMI'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="SELECT * FROM [Dim-BMI];"
PersonFrame0=pd.read_sql_query(sSQL, conn1)
sTable = 'Dim-BMI'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="SELECT PersonID,\n    Height,\n    Weight,\n    bmi,\n    Indicator\nFROM [Dim-BMI]\nWHERE \
Height > 1.5 \
and Indicator = 1\
ORDER BY \
    Height,\
    Weight;"
```

```
PersonFrame1=pd.read_sql_query(sSQL, conn1)
DimPerson=PersonFrame1
DimPersonIndex=DimPerson.set_index(['PersonID'],inplace=False)
sTable = 'Dim-BMI-Horizontal'
print('Storing :',sDatabaseName,'\\n Table:',sTable)
DimPersonIndex.to_sql(sTable, conn2, if_exists="replace")
sTable = 'Dim-BMI-Horizontal'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="SELECT * FROM [Dim-BMI-Horizontal];"
PersonFrame2=pd.read_sql_query(sSQL, conn2)
print('Full Data Set (Rows):', PersonFrame0.shape[0])
print('Full Data Set (Columns):', PersonFrame0.shape[1])
print('Horizontal Data Set (Rows):', PersonFrame2.shape[0])
print('Horizontal Data Set (Columns):', PersonFrame2.shape[1])
```

OUTPUT:

```
#####
('Loading :', 'C:/Users/student/Downloads/datamart.db', ' Table:', 'Dim-BMI')
#####
('Loading :', 'C:/Users/student/Downloads/datamart.db', ' Table:', 'Dim-BMI')
#####

#####
('Storing :', 'C:/Users/student/Downloads/datamart.db', '\n Table:', 'Dim-BMI-Horizontal')

#####
('Loading :', 'C:/Users/student/Downloads/datamart.db', ' Table:', 'Dim-BMI-Horizontal')
#####
('Full Data Set (Rows):', 1080)
('Full Data Set (Columns):', 5)
#####
('Horizontal Data Set (Rows):', 194)
('Horizontal Data Set (Columns):', 5)
#####
```

DB Browser for SQLite - C:\DataScience\99-DW\datamart.db

The screenshot shows the DB Browser for SQLite interface with the 'Dim-BMI' table selected. The table structure is as follows:

	PersonID	Height	Weight	bmi	Indicator
1	324	1.6	30	11.71875	1
2	325	1.6	35	13.671875	1
3	326	1.6	40	15.625	1
4	327	1.6	45	17.578125	1
5	378	1.7	30	10.380622837...	1
6	379	1.7	35	12.110726643...	1
7	380	1.7	40	13.840830449...	1
8	381	1.7	45	15.570934256...	1
9	382	1.7	50	17.301038062...	1
10	432	1.8	30	9.2592592592...	1
11	433	1.8	35	10.802469135...	1
12	434	1.8	40	12.345679012...	1
13	435	1.8	45	13.888888888...	1
14	436	1.8	50	15.432098765...	1

1 - 15 of 194 Go to: 1

DB Browser for SQLite - C:\DataScience\99-DW\datamart.db

The screenshot shows the DB Browser for SQLite interface with the 'Dim-BMI-Horizontal' table selected. The table structure is as follows:

	PersonID	Height	Weight	bmi	Indicator
1	324	1.6	30	11.71875	1
2	325	1.6	35	13.671875	1
3	326	1.6	40	15.625	1
4	327	1.6	45	17.578125	1
5	328	1.6	50	19.53125	2
6	329	1.6	55	21.484375	2
7	330	1.6	60	23.4375	2
8	331	1.6	65	25.390625	3
9	332	1.6	70	27.34375	3
10	333	1.6	75	29.296875	3
11	334	1.6	80	31.25	4
12	335	1.6	85	33.203125	4
13	336	1.6	90	35.15625	4
14	337	1.6	95	37.109375	4

1 - 15 of 756 Go to: 1

Organize vertical

```
import sys
import os
import pandas as pd
import sqlite3 as sq
sDatabaseName='C:/Users/student/Downloads/datawarehouse.db'
conn1 = sq.connect(sDatabaseName)
sDatabaseName='C:/Users/student/Downloads/datamart.db'
conn2 = sq.connect(sDatabaseName)
sTable = 'Dim-BMI'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="SELECT * FROM [Dim-BMI];"
PersonFrame0=pd.read_sql_query(sSQL, conn1)
sTable = 'Dim-BMI'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="SELECT \
    Height,\n    Weight,\n    Indicator\
FROM [Dim-BMI];"
PersonFrame1=pd.read_sql_query(sSQL, conn1)
DimPerson=PersonFrame1
DimPersonIndex=DimPerson.set_index(['Indicator'], inplace=False)
sTable = 'Dim-BMI-Vertical'

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

OUTPUT:

```
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\VKHCG\01-Verveulen\05-Organise\Organize-Vertical.py =====
('Loading :', 'C:/Users/student/Downloads/datamart.db', ' Table:', 'Dim-BMI')
#####
('Loading :', 'C:/Users/student/Downloads/datamart.db', ' Table:', 'Dim-BMI')
#####
#####
('Storing :', 'C:/Users/student/Downloads/datamart.db', '\n Table:', 'Dim-BMI-Vertical')

#####
#####
('Loading :', 'C:/Users/student/Downloads/datamart.db', ' Table:', 'Dim-BMI-Vertical')
#####
('Full Data Set (Rows):', 1080)
('Full Data Set (Columns):', 5)
#####
('Horizontal Data Set (Rows):', 1080)
('Horizontal Data Set (Columns):', 3)
#####
>>> |
```

DB Browser for SQLite - C:\DataScience\99-DW\datamart.db

Table: Dim-BMI

	PersonID	Height	Weight	bmi	Indicator
	Filter	Filter	Filter	Filter	Filter
1	324	1.6	30	11.71875	1
2	325	1.6	35	13.671875	1
3	326	1.6	40	15.625	1
4	327	1.6	45	17.578125	1
5	378	1.7	30	10.380622837...	1
6	379	1.7	35	12.110726643...	1
7	380	1.7	40	13.840830449...	1
8	381	1.7	45	15.570934256...	1
9	382	1.7	50	17.301038062...	1
10	432	1.8	30	9.259259259...	1
11	433	1.8	35	10.802469135...	1
12	434	1.8	40	12.345679012...	1
13	435	1.8	45	13.888888888...	1
14	436	1.8	50	15.432098765...	1
15					

1 - 15 of 194 Go to: 1

DB Browser for SQLite - C:\DataScience\99-DW\datamart.db

Table: Dim-BMI-Vertical

	PersonID	Height	Weight
	Filter	Filter	Filter
1	0	1.0	30
2	1	1.0	35
3	2	1.0	40
4	3	1.0	45
5	4	1.0	50
6	5	1.0	55
7	6	1.0	60
8	7	1.0	65
9	8	1.0	70
10	9	1.0	75
11	10	1.0	80
12	11	1.0	85
13	12	1.0	90
14	13	1.0	95
15	14	1.0	100

1 - 15 of 1080 Go to: 1

Organize Island

```
import sys
import os
import pandas as pd
import sqlite3 as sq
sDatabaseName='C:/Users/student/Downloads/datawarehouse.db'
conn1 = sq.connect(sDatabaseName)
#####
sDatabaseName='C:/Users/student/Downloads/datamart.db'
conn2 = sq.connect(sDatabaseName)
#####
print('#####')
sTable = 'Dim-BMI'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="SELECT * FROM [Dim-BMI];"
PersonFrame0=pd.read_sql_query(sSQL, conn1)
#####
print('#####')
sTable = 'Dim-BMI'
print('Loading :',sDatabaseName,' Table:',sTable)

sSQL="SELECT \
    Height,\n
    Weight,\n
    Indicator\
FROM [Dim-BMI]\n
WHERE Indicator > 2\n
ORDER BY \
    Height,\n
    Weight;"
```

PersonFrame1=pd.read_sql_query(sSQL, conn1)

```
#####
DimPerson=PersonFrame1
DimPersonIndex=DimPerson.set_index(['Indicator'],inplace=False)
#####
sTable = 'Dim-BMI-Vertical'
print('\n#####')
print('Storing :',sDatabaseName,'\n Table:',sTable)
print('\n#####')
DimPersonIndex.to_sql(sTable, conn2, if_exists="replace")
#####
print('#####')
sTable = 'Dim-BMI-Vertical'
print('Loading :',sDatabaseName,' Table:',sTable)
print('#####')
sSQL="SELECT * FROM [Dim-BMI-Vertical];"
PersonFrame2=pd.read_sql_query(sSQL, conn2)
#####
print('#####')
```

```

print('Full Data Set (Rows):', PersonFrame0.shape[0])
print('Full Data Set (Columns):', PersonFrame0.shape[1])
print('#####')
print('Horizontal Data Set (Rows):', PersonFrame2.shape[0])
print('Horizontal Data Set (Columns):', PersonFrame2.shape[1])

```

OUTPUT:

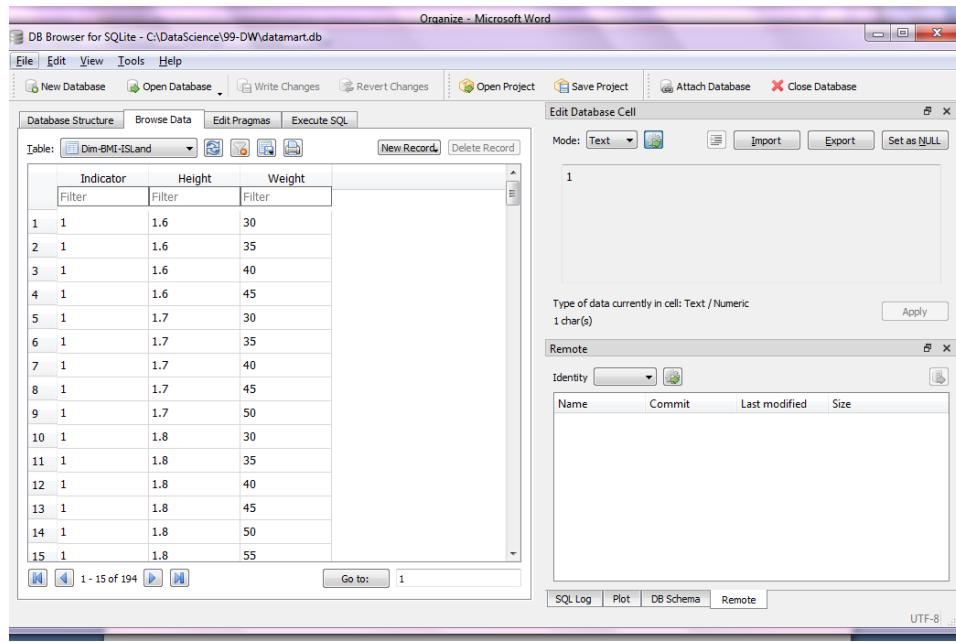
```

>>>
===== RESTART: C:\VKHCG\01-Vermeulen\05-Organise\Organize-Island.py ======
#####
('Loading : ', 'C:/Users/student/Downloads/datamart.db', ' Table:', 'Dim-BMI')
#####
('Loading : ', 'C:/Users/student/Downloads/datamart.db', ' Table:', 'Dim-BMI')
#####
('Storing : ', 'C:/Users/student/Downloads/datamart.db', '\n Table:', 'Dim-BMI-Vertical')
#####
('Loading : ', 'C:/Users/student/Downloads/datamart.db', ' Table:', 'Dim-BMI-Vertical')
#####
('Full Data Set (Rows):', 1080)
('Full Data Set (Columns):', 5)
#####
('Horizontal Data Set (Rows):', 771)
('Horizontal Data Set (Columns):', 3)
#####
>>> |

```

The screenshot shows the DB Browser for SQLite interface with the database 'C:\DataScience\99-DW\datamart.db' open. The 'Dim-BMI' table is selected. The table structure is as follows:

	PersonID	Height	Weight	bmi	Indicator
1	324	1.6	30	11.71875	1
2	325	1.6	35	13.671875	1
3	326	1.6	40	15.625	1
4	327	1.6	45	17.578125	1
5	378	1.7	30	10.380622837...	1
6	379	1.7	35	12.110726643...	1
7	380	1.7	40	13.840830449...	1
8	381	1.7	45	15.570934256...	1
9	382	1.7	50	17.301038062...	1
10	432	1.8	30	9.2592592592...	1
11	433	1.8	35	10.802469135...	1
12	434	1.8	40	12.345679012...	1
13	435	1.8	45	13.888888888...	1
14	436	1.8	50	15.432098765...	1



Secure-Vault:-

```
# -*- coding: utf-8 -*-
import sys
import os
import pandas as pd
import sqlite3 as sq
sDataWarehouseDir='C:\DataScience\99-DW'
if not os.path.exists(sDataWarehouseDir):
    os.makedirs(sDataWarehouseDir)
sDatabaseName=sDataWarehouseDir + '/datawarehouse.db'
conn1 = sq.connect(sDatabaseName)
sDatabaseName=sDataWarehouseDir + '/datamart.db'
conn2 = sq.connect(sDatabaseName)
sTable = 'Dim-BMI'
print('Loading :',sDatabaseName,' Table:',sTable)
sSQL="SELECT * FROM [Dim-BMI];"
PersonFrame0=pd.read_sql_query(sSQL, conn1)
sTable = 'Dim-BMI'
print('Loading :',sDatabaseName,' Table:',sTable)

sSQL="SELECT \
    Height,\n
    Weight,\n
    Indicator,\n
    CASE Indicator\n
    WHEN 1 THEN 'Pip'\n
    WHEN 2 THEN 'Norman'\n
    WHEN 3 THEN 'Grant'\n
    ELSE 'Sam'\n
    END AS Name\n
```

```

FROM [Dim-BMI]\
WHERE Indicator > 2\
ORDER BY \
    Height,\n
    Weight;"\n
PersonFrame1=pd.read_sql_query(sSQL, conn1)\n
DimPerson=PersonFrame1\n
DimPersonIndex=DimPerson.set_index(['Indicator'], inplace=False)\n
sTable = 'Dim-BMI-Secure'\n
print('Storing :',sDatabaseName,' Table:',sTable)\n
DimPersonIndex.to_sql(sTable, conn2, if_exists="replace")\n
sTable = 'Dim-BMI-Secure'\n
print('Loading :',sDatabaseName,' Table:',sTable)\n
sSQL="SELECT * FROM [Dim-BMI-Secure] WHERE Name = 'Sam';"\n
PersonFrame2=pd.read_sql_query(sSQL, conn2)\n
print('Full Data Set (Rows):', PersonFrame0.shape[0])\n
print('Full Data Set (Columns):', PersonFrame0.shape[1])\n
print('Horizontal Data Set (Rows):', PersonFrame2.shape[0])\n
print('Horizontal Data Set (Columns):', PersonFrame2.shape[1])\n
print('Only Sam Data')\n
print(PersonFrame2.head())

```

OUTPUT:

```

Python 3.7.4 Shell
File Edit Shell Debug Options Window Help
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\VKHCG\01-Vermeulen\05-Organise\Organize-Secure-Vault.py =====
#####
Loading : C:\DataScience/99-DW/datamart.db  Table: Dim-BMI
#####
Loading : C:\DataScience/99-DW/datamart.db  Table: Dim-BMI

#####
Storing : C:\DataScience/99-DW/datamart.db
Table: Dim-BMI-Secure

#####
Loading : C:\DataScience/99-DW/datamart.db  Table: Dim-BMI-Secure
#####
Full Data Set (Rows): 1080
Full Data Set (Columns): 5
#####
Horizontal Data Set (Rows): 692
Horizontal Data Set (Columns): 4
Only Sam Data
   Indicator  Height  Weight  Name
0          4      1.0     35  Sam
1          4      1.0     40  Sam
2          4      1.0     45  Sam
3          4      1.0     50  Sam
4          4      1.0     55  Sam
#####

```

DB Browser for SQLite - C:\DataScience\99-DW\datamart.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragmas Execute SQL

Table: Dim-BMI New Record Delete Record

PersonID Height Weight bmi Indicator

	PersonID	Height	Weight	bmi	Indicator
1	324	1.6	30	11.71875	1
2	325	1.6	35	13.671875	1
3	326	1.6	40	15.625	1
4	327	1.6	45	17.578125	1
5	378	1.7	30	10.380622837...	1
6	379	1.7	35	12.110726643...	1
7	380	1.7	40	13.840830449...	1
8	381	1.7	45	15.570934256...	1
9	382	1.7	50	17.301038062...	1
10	432	1.8	30	9.2592592592...	1
11	433	1.8	35	10.802469135...	1
12	434	1.8	40	12.345679012...	1
13	435	1.8	45	13.888888888...	1
14	436	1.8	50	15.432098765...	1

1 - 15 of 194 Go to: 1

Edit Database Cell Mode: Text Import Export Set as NULL

Type of data currently in cell: Text / Numeric 1 char(s) Apply

Remote Identity

Name	Commit	Last modified	Size

SQL Log Plot DB Schema Remote UTF-8

DB Browser for SQLite - C:\DataScience\99-DW\datamart.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragmas Execute SQL

Table: Dim-BMI-Secure New Record Delete Record

Indicator Height Weight Name

	Indicator	Height	Weight	Name
1	4	1.0	35	Sam
2	4	1.0	40	Sam
3	4	1.0	45	Sam
4	4	1.0	50	Sam
5	4	1.0	55	Sam
6	4	1.0	60	Sam
7	4	1.0	65	Sam
8	4	1.0	70	Sam
9	4	1.0	75	Sam
10	4	1.0	80	Sam
11	4	1.0	85	Sam
12	4	1.0	90	Sam
13	4	1.0	95	Sam
14	4	1.0	100	Sam
15	4	1.0	105	Sam

1 - 15 of 771 Go to: 1

Edit Database Cell Mode: Text Import Export Set as NULL

Type of data currently in cell: Text / Numeric 1 char(s) Apply

Remote Identity

Name	Commit	Last modified	Size

SQL Log Plot DB Schema Remote UTF-8

Practical No. 9

Generating Reports

INPUT:

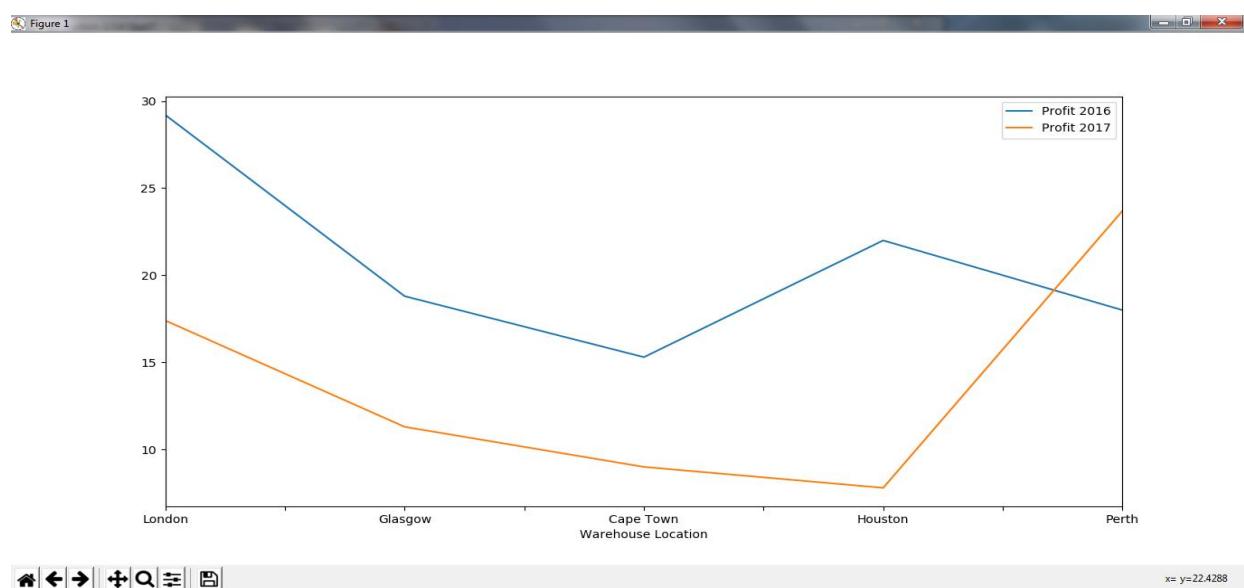
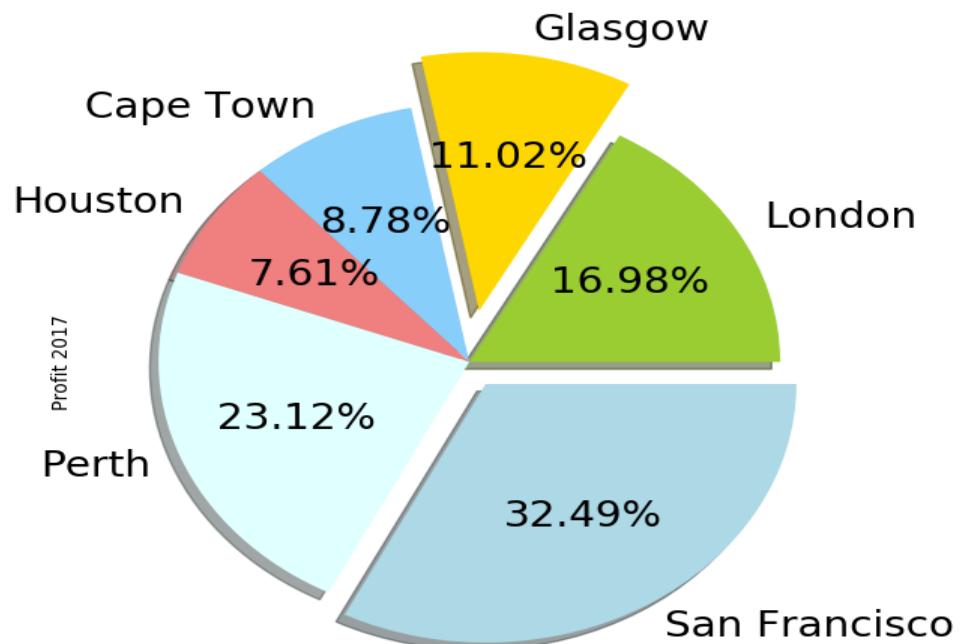
```
import sys
import os
import pandas as pd
import matplotlib as ml
from matplotlib import pyplot as plt
data=[['London',      29.2, 17.4],\n      ['Glasgow',    18.8, 11.3],\n      ['Cape Town',  15.3, 9.0],\n      ['Houston',     22.0, 7.8],\n      ['Perth',       18.0, 23.7],\n      ['San Francisco', 11.4, 33.3]]\n\nos_new=pd.DataFrame(data)\n\npd.Index(['Item', 'Value', 'Value Percent', 'Conversions', 'Conversion Percent',\n          'URL', 'Stats URL'],\n          dtype='object')\n\nos_new.rename(columns = {0 : "Warehouse Location"}, inplace=True)\n\nos_new.rename(columns = {1 : "Profit 2016"}, inplace=True)\n\nos_new.rename(columns = {2 : "Profit 2017"}, inplace=True)\n\n\nexplode = (0, 0.2, 0, 0, 0, 0.1)\nlabels=os_new['Warehouse Location']\ncolors_mine = ['yellowgreen', 'gold', 'lightskyblue', 'lightcoral', 'lightcyan','lightblue']\nos_new.plot(figsize=(10, 10),kind="pie", y="Profit 2017", autopct='%.2f%%', \\n      shadow=True, explode=explode, legend = False, colors = colors_mine,\\n      labels=labels, fontsize=20)\nsPicNameOut1='D:/DataScience/pie_explode.png'\nplt.savefig(sPicNameOut1,dpi=600)\nplt.show()\nos_new.iloc[:5].plot(figsize=(10, 10),kind='line',x='Warehouse Location',\\n      y=['Profit 2016','Profit 2017']);\nplt.show()\nos_new.iloc[:5].plot(figsize=(10, 10),kind='bar',x='Warehouse Location',\\n      y=['Profit 2016','Profit 2017']);\nplt.show()\nos_new.iloc[:5].plot(figsize=(10, 10),kind='barh',x='Warehouse Location',\\n      y=['Profit 2016','Profit 2017']);\nplt.show()\nos_new.iloc[:5].plot(figsize=(10, 10),kind='area',x='Warehouse Location',\\n      y=['Profit 2016','Profit 2017'],stacked=False);\nplt.show()\nos_new.iloc[:5].plot(figsize=(10, 10),kind='scatter',x='Profit 2016',\\n      y='Profit 2017',color='DarkBlue',marker='D');
```

```

plt.show()
os_new.iloc[:5].plot(figsize=(13, 10), kind='hexbin', x='Profit 2016', \
                     y='Profit 2017', gridsize=25);
plt.show()

```

OUTPUT:



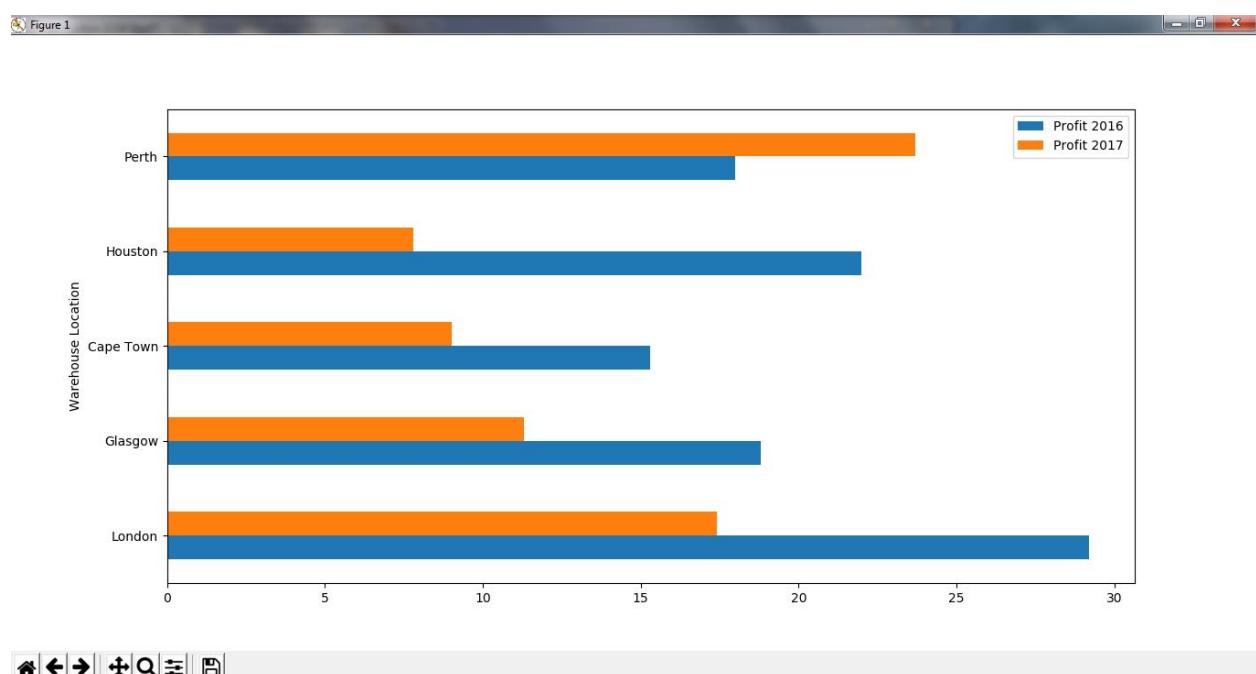
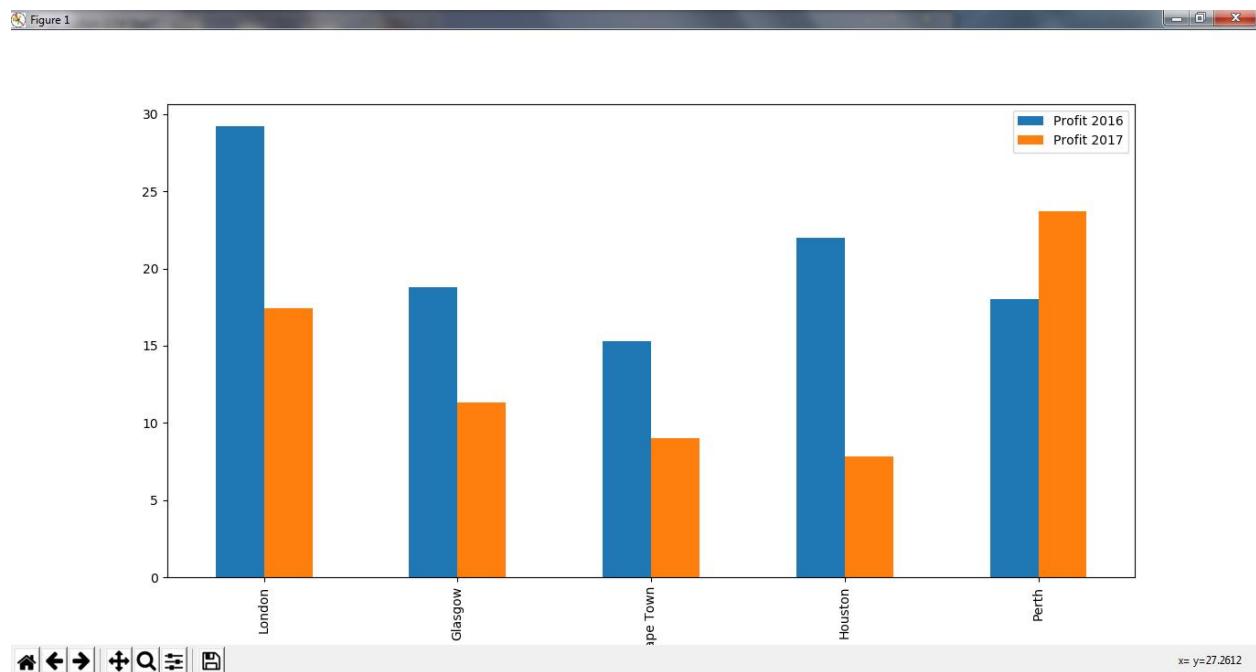


Figure 1

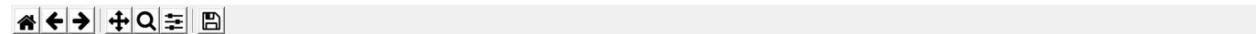
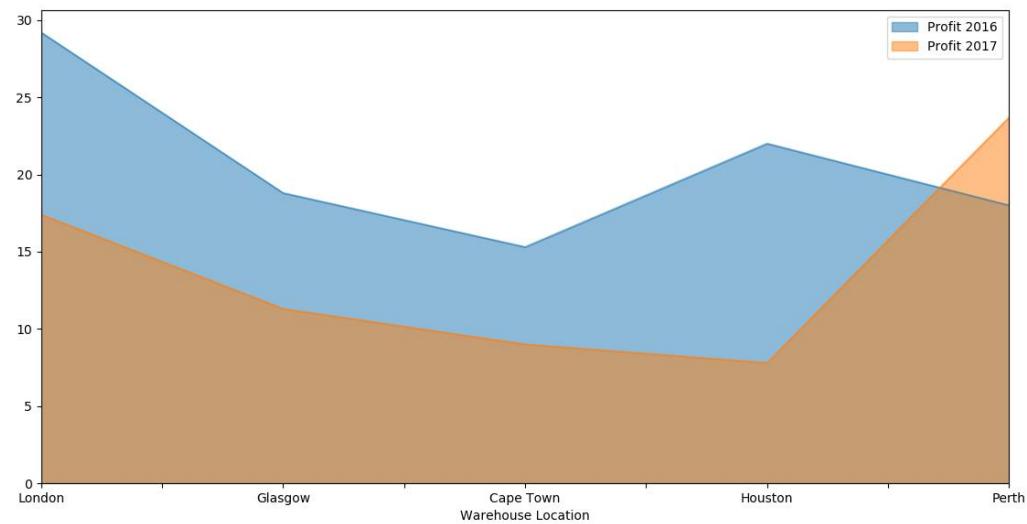
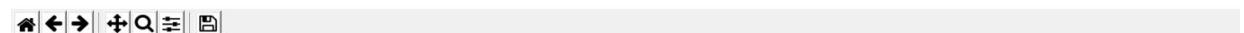
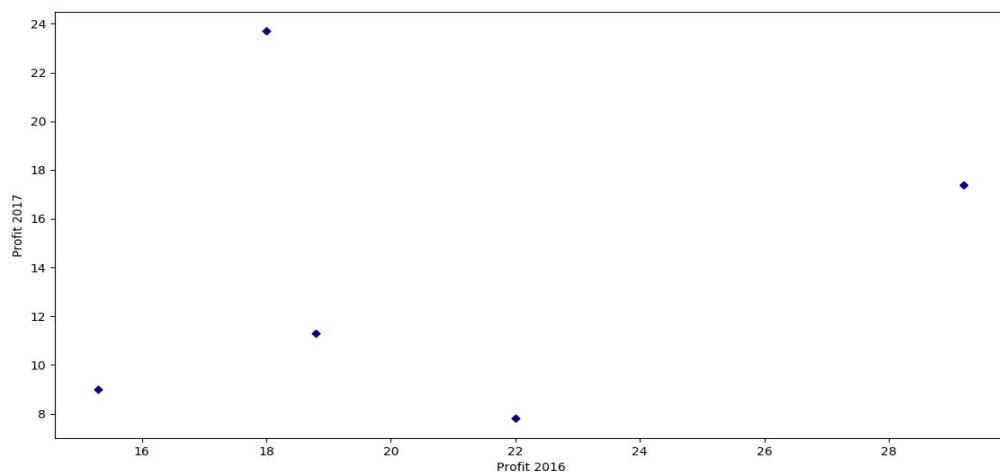
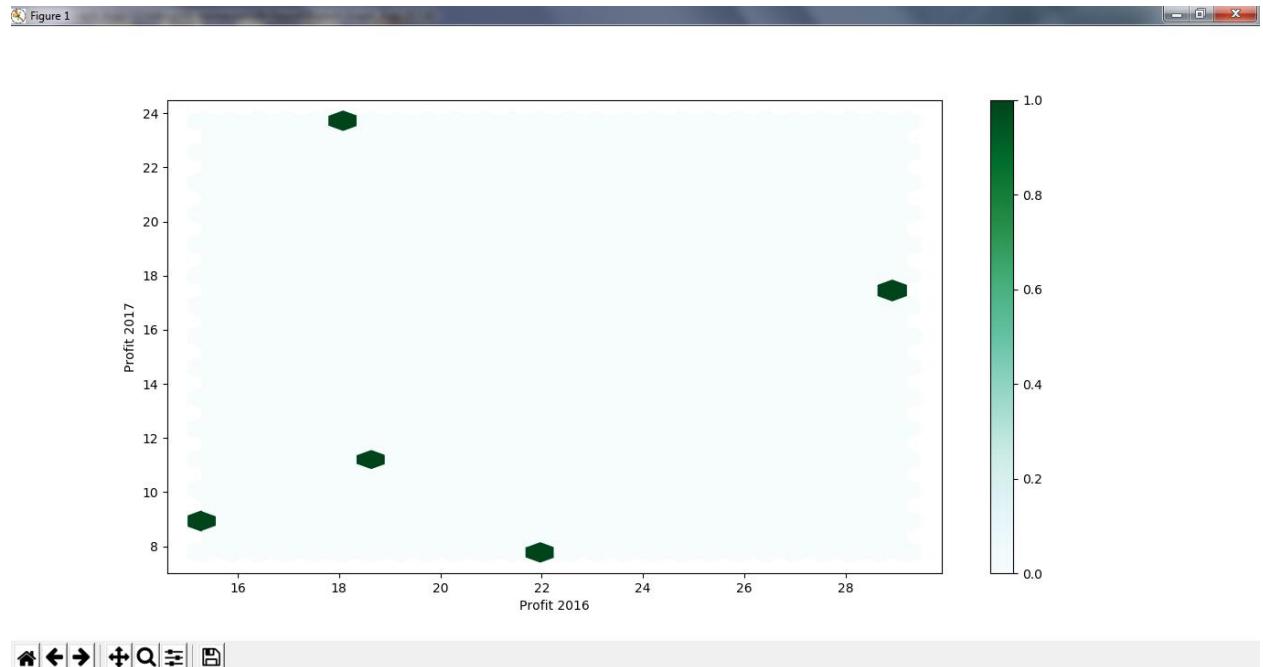


Figure 1





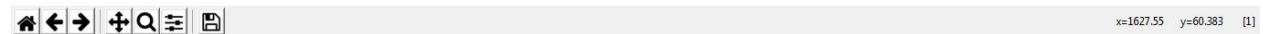
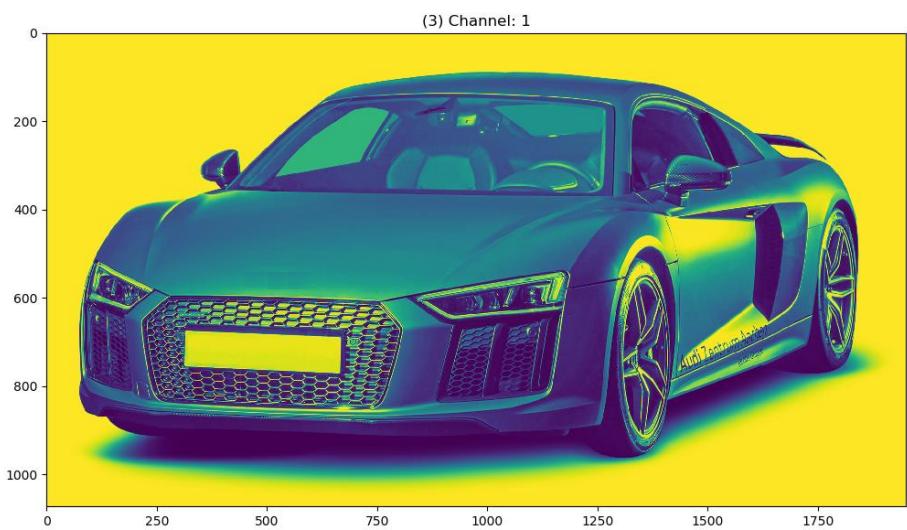
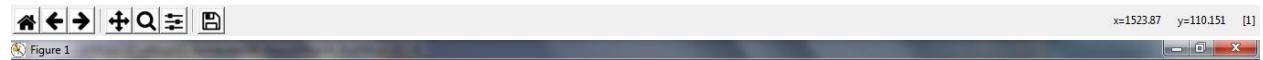
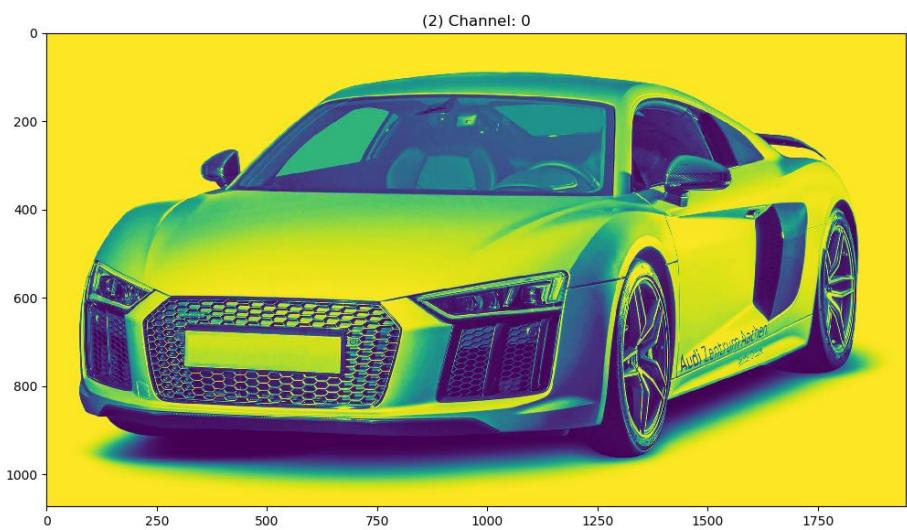
INPUT:

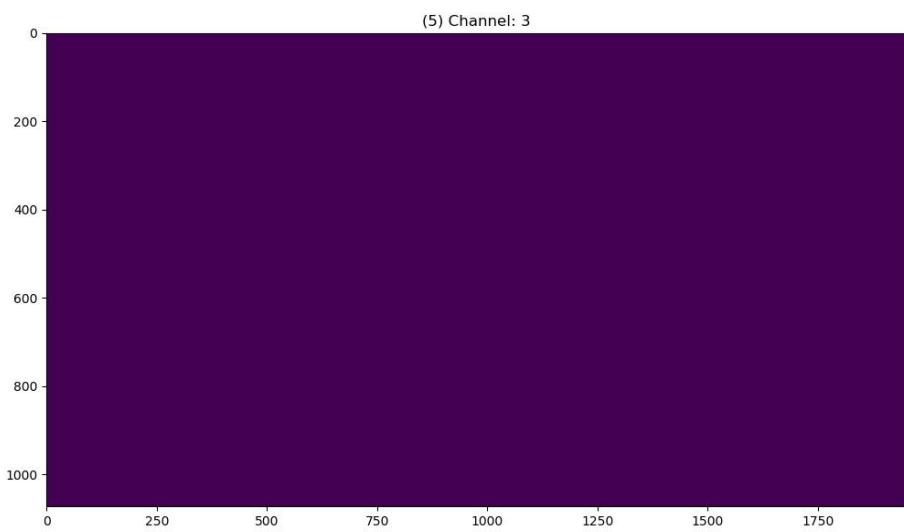
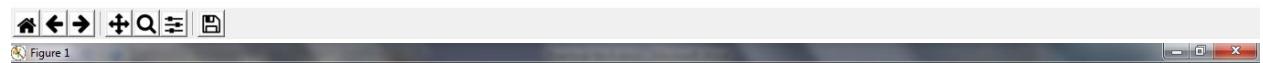
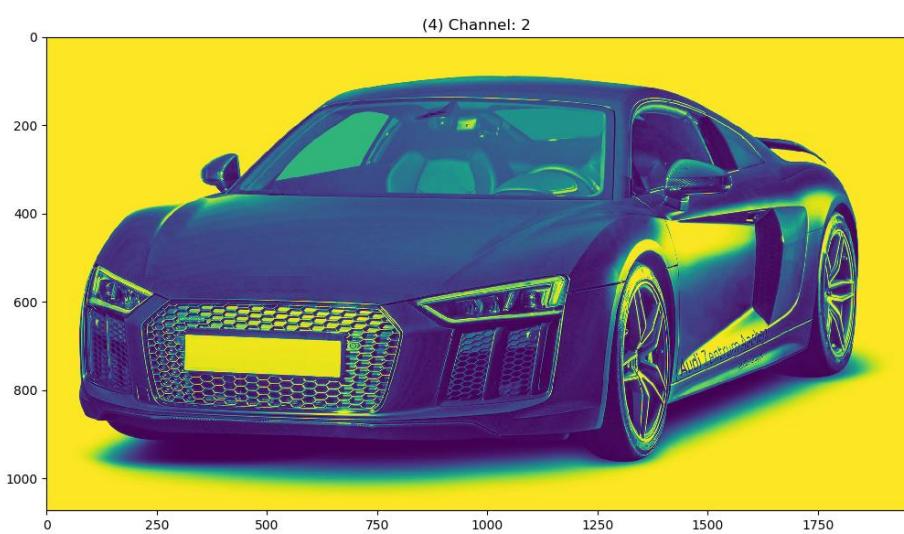
```
import sys
import os
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
sPicName='C:/vkhcg/01-Vermeulen/00-RawData/AudiR8.png'
t=0
img=mpimg.imread(sPicName)
print('Size:', img.shape)
plt.figure(figsize=(10, 10))
t+=1
sTitle= '(' + str(t) + ') Original'
plt.title(sTitle)
plt.imshow(img)
plt.show()
for c in range(img.shape[2]):
    t+=1
    plt.figure(figsize=(10, 10))
    sTitle= '(' + str(t) + ') Channel: ' + str(c)
    plt.title(sTitle)
    lum_img = img[:, :, c]
    plt.imshow(lum_img)
    plt.show()
```

OUTPUT:

Size: (1073, 1950, 4)





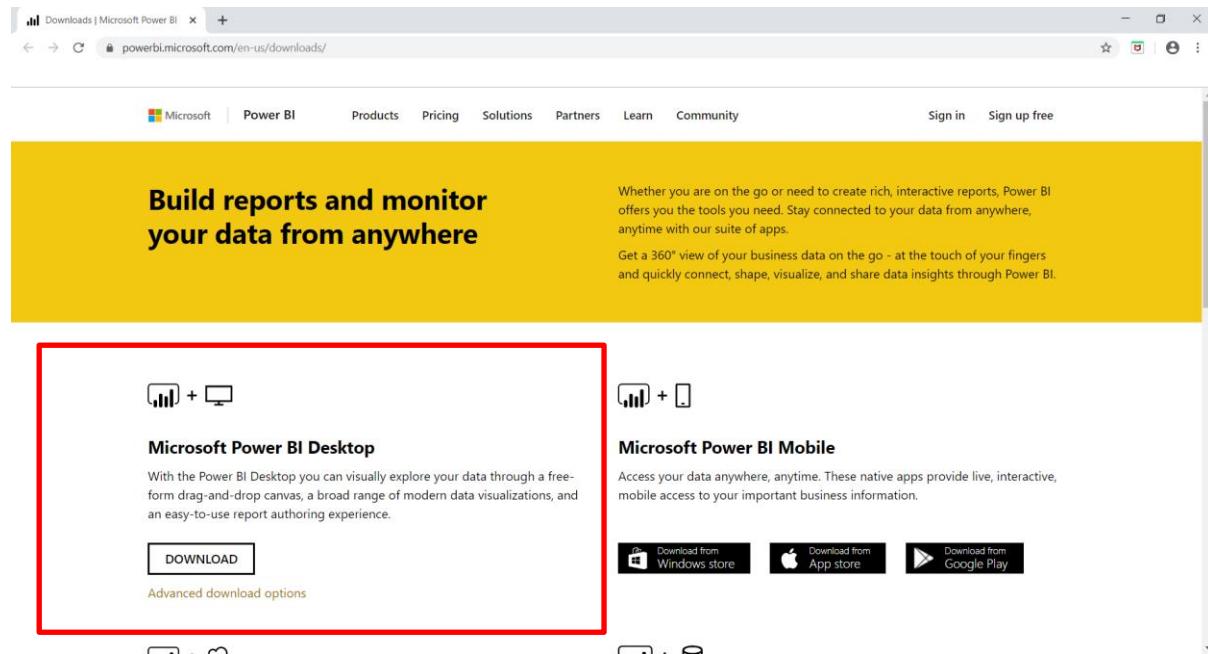


x=1679.39 y=20.9832 [1]

Practical No. 10

Data Visualization with Power BI

Step 1: Install and Open Power BI.

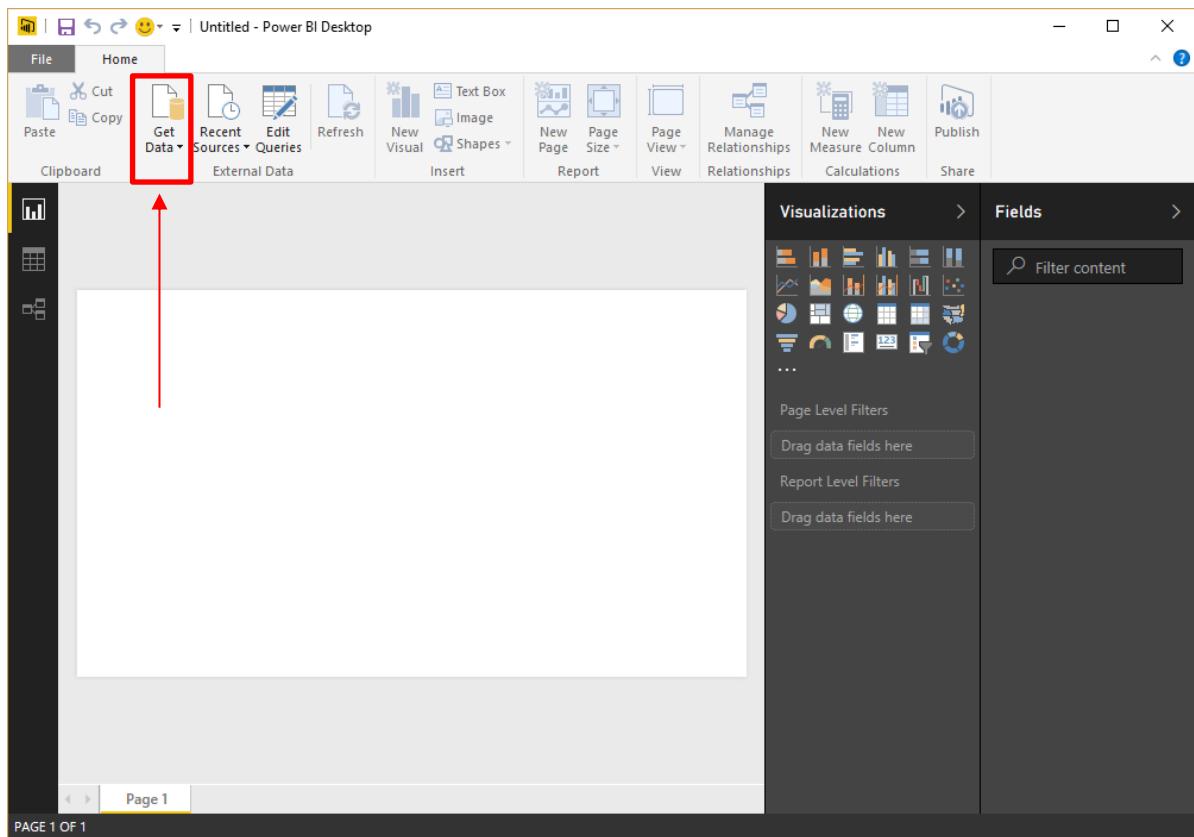


Step 2:Create .xlsx/.csv file like following

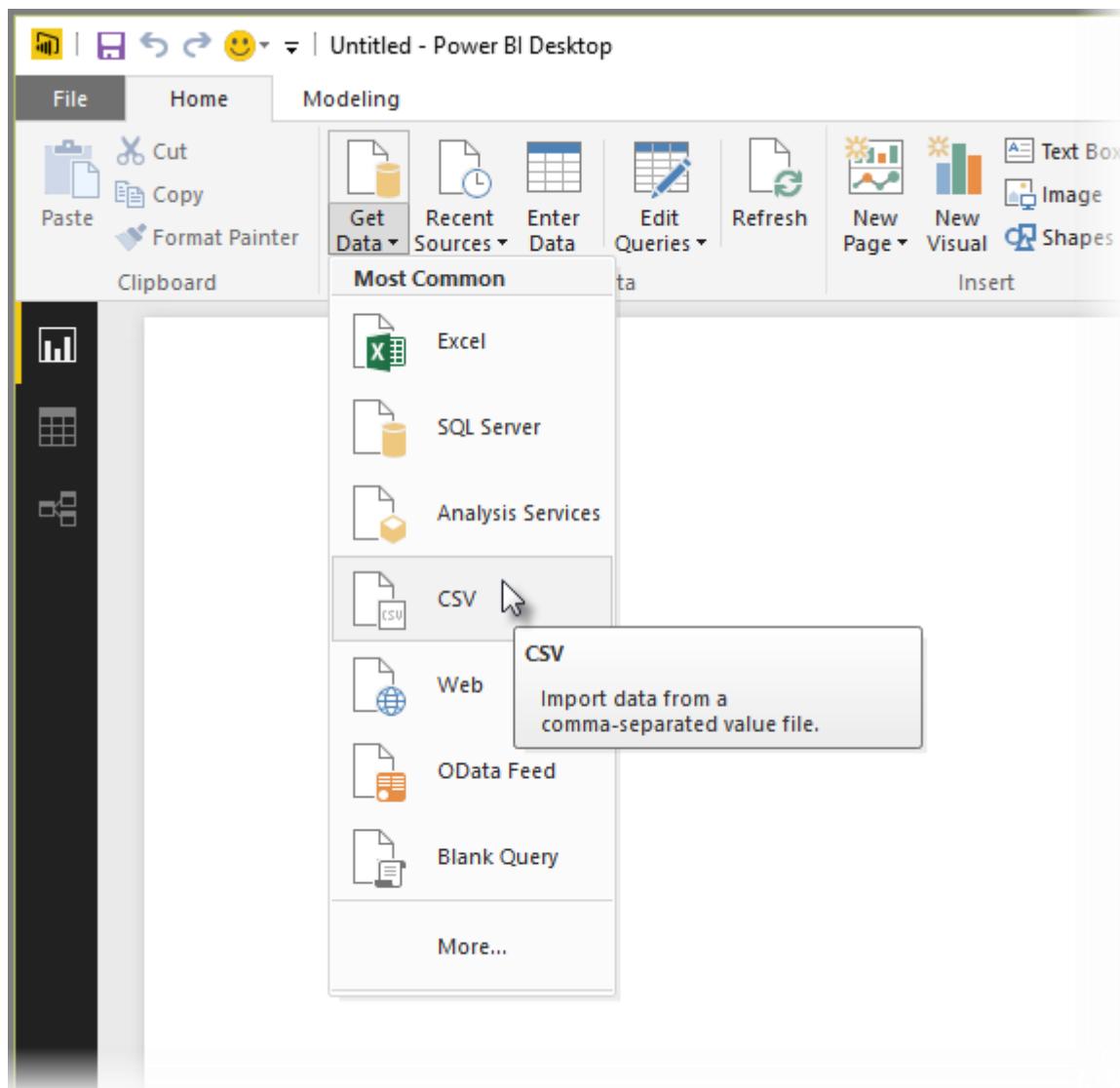
A screenshot of Microsoft Excel showing a dataset titled "Company Name". The data is organized into columns: "Company" (A), "Years" (B), and "Profit" (C). The rows are numbered from 1 to 16. The data shows the following values:

	Company	Years	Profit
1	A	2001	75000
2	B	2001	542000
3	C	2001	595000
4	D	2001	850020
5	E	2001	99900
6	A	2002	998700
7	B	2002	78500
8	C	2002	580040
9	D	2002	65900
10	E	2002	965400
11	A	2003	554400
12	B	2003	775540
13	C	2003	98000
14	D	2003	780000
15	E	2003	98500
16			
17			
18			
19			
20			
21			
22			
23			
24			
25			
26			
27			
28			
29			
30			
31			
32			

Step 3: Click on Get data in Power BI application



Step 4: And select the xlsx/csv file which was created then load it



Step5: On Axis select Year and Company name, On legend select company name, On value select profit.

OUTPUT:

