Data Visualization using PowerBI Desktop

Exercise 4 –Visualizing Data using Microsoft PowerBI

Student Name: **Student Id:**

Date:

Please use the screenshots ONLY as a reference. The instructions have to be followed AS-IS.

Objective

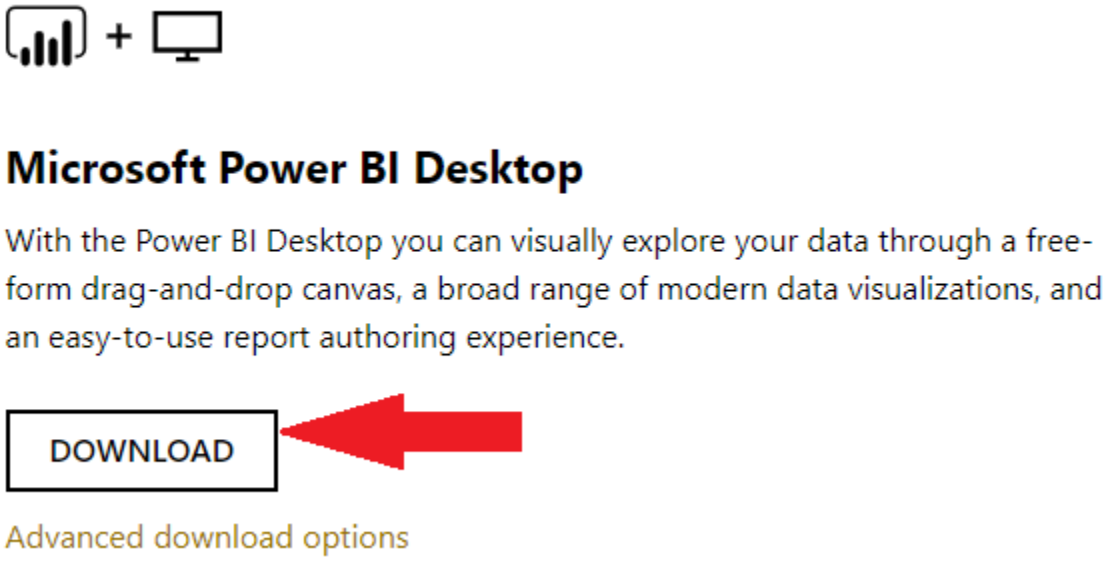
The objective of this exercise is to develop skills on how to visualize the data set using PowerBI tool.

Prerequisite: Install PowerBI Desktop

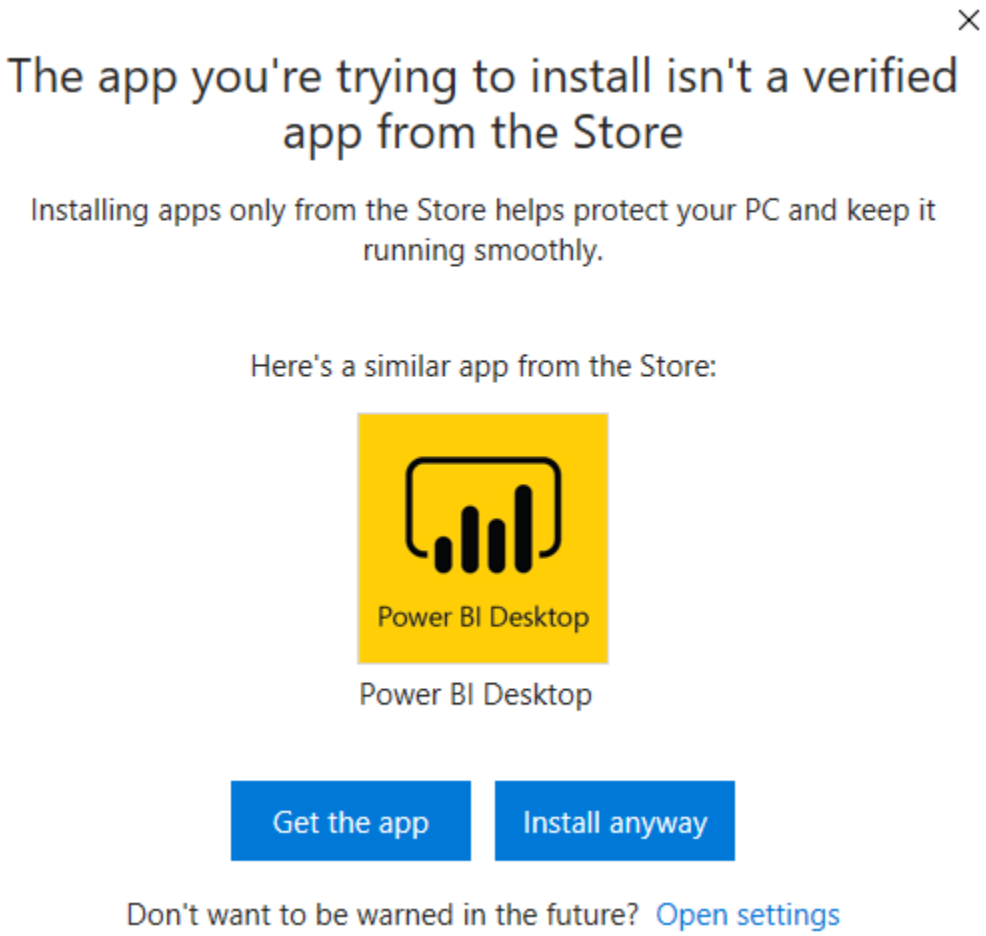
Step 1: Download the provided CSV file from eLearning

In order to download and install the PowerBI Desktop, please click on the below link.

https://powerbi.microsoft.com/en-us/downloads/



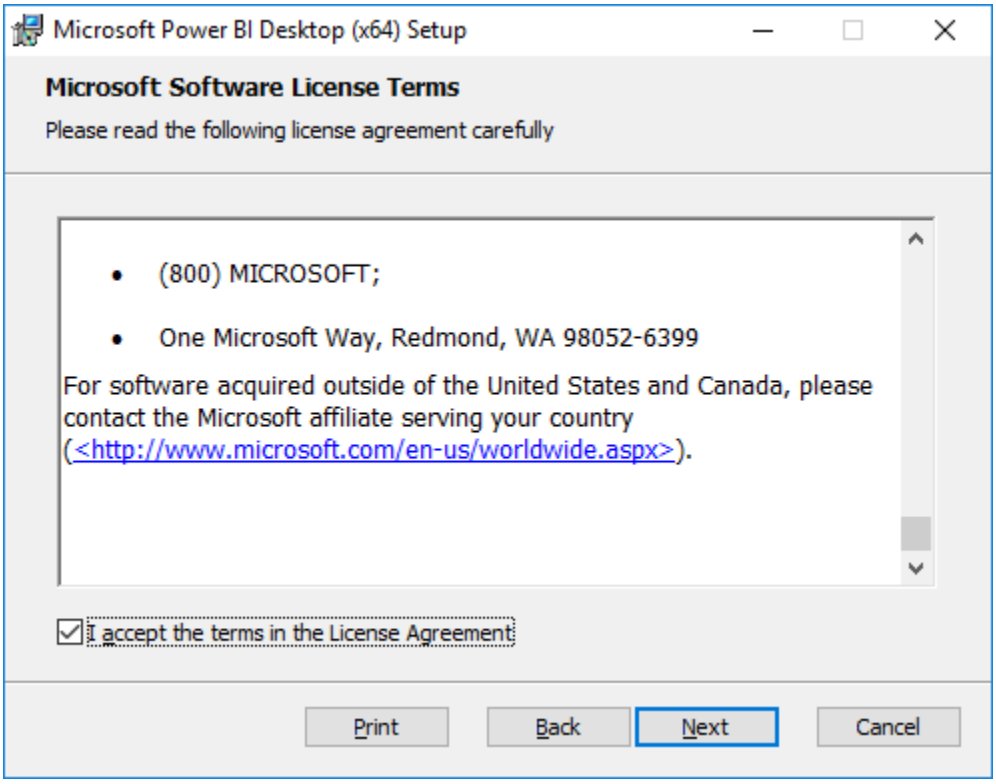
Click ‘Install anyway’



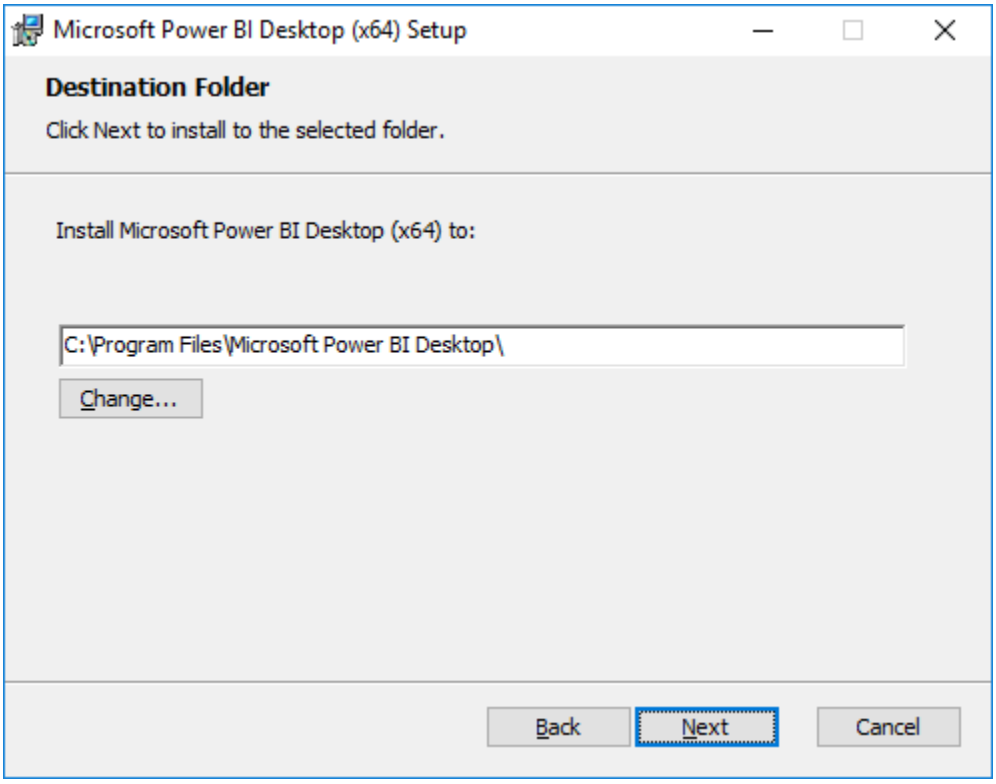
Note: You may or may not get different screen based on the version of your OS, or based on the app installation settings.

Click Next

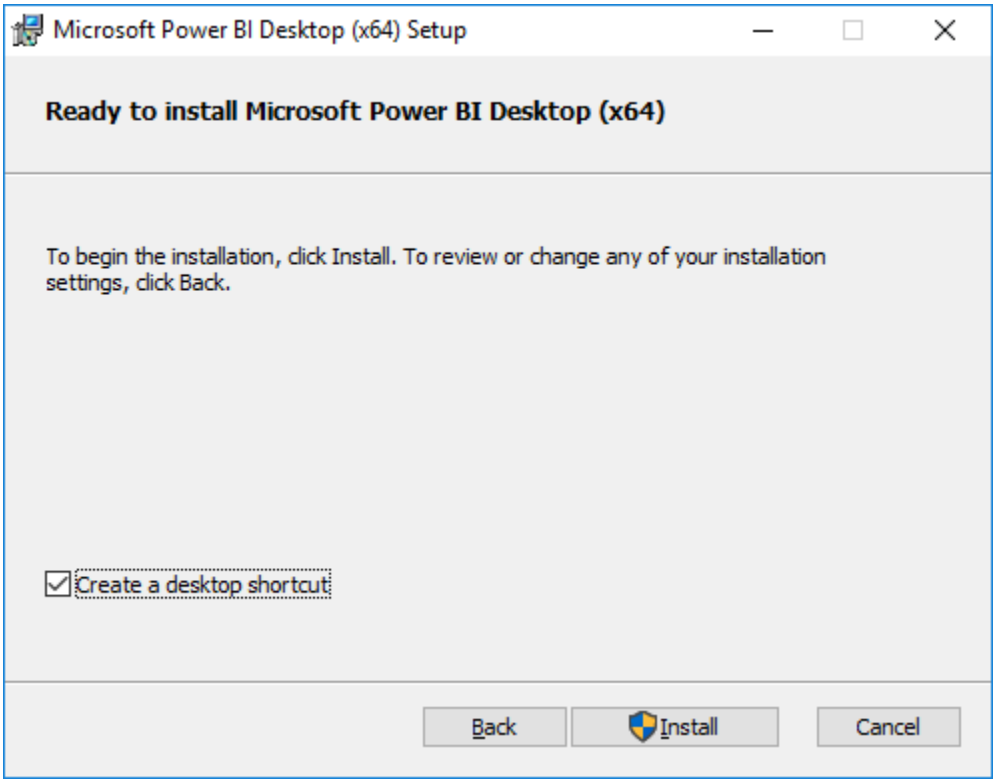
Read and accept the License Agreement



Provide the desired path of installation. You may keep the default.



Click install



Step 2: Write the following python program in notepad++ / sublime text

Purpose of this program is to format the data of the csv file and make it appropriate for PowerBI.

This python program will make following format changes in the downloaded csv file:

- It will merge the columns ‘Place’ and ‘State’ and make it one column with format ‘State-Place’
- It will transpose ‘Years and Population’ i.e. after running this python code the newly created csv file will have ‘Years’ and ‘Population’ as columns instead of rows
- A new column ‘Student Last Name - Rank’ will be created which will rank the ‘State-City’ based on population.

Note:
You need to make following changes in the code:

- In line number 12 of the coding, ‘Student Last name’ in column ‘Student Last Name - Rank’ should replace your last name
- In line number 7 of the coding, here you have to provide the location where you have saved the file in Step-1.
- In line number 71 of the code, you need to provide a location where you want to save the newly created CSV file.

Following is the code for the above program:

Step 1: At first we import the libraries that we need to reference for the third party functions in our code.

```
1  import csv
2  import operator
3  import copy
4  from token import EQUAL
5  from builtins import sorted
6
```

Step 2: We now read the CSV file that has been provided to you. Keep in mind the delimiter used in our CSV files is ','(comma). Don't stick to the notion that CSV's only use comma, they made use ' '(Tab), ';'(Semi-colon or maybe even a '^'(Caret)).

```
1  import csv
2  import operator
3  import copy
4  from token import EQUAL
5  from builtins import sorted
6
7  reader = csv.reader(open('C:\\Users\\AmanSharma\\Desktop\\Largest_Cities.csv'), delimiter = "," )
8
```

Step 3: This may be a little vague to you right now but we define a few variables which will determine the output file we generate from this unorganised CSV. The first 4 variables(u,x,y,z) represent the Columns our desired output file. 'rows_so_far' and 'c' are just 'counter' variables we'll be using in our loops later.

```
5  from builtins import sorted
6
7  reader = csv.reader(open('C:\\Users\
8
9  u = 'State - Place'
10 x = 'Year'
11 y = 'Population'
12 z = 'Sharma - Rank'
13 rows_so_far = 0
14 c= 0
15
```

Step 4: For those of you who have a programming background, just skip reading this paragraph but for others, we are defining a 2D Array here and allowing it to get appended. Remember this as a swimming pool, you will throw a ball (in our case- a data row), your dog will go fetch it and put it in his toy basket(final list).

```
13  rows_so_far = 0
14  c= 0
15
16  pool = []
17  pool.append([])
18
```

Step 5: Alright so now comes to complex part, there are 3 sub-Loops running underneath this loop, you have to be very careful with the indenting. Initially, we've got zero rows in our pool so let's put in the first row (Header) with the labels.

```
16  pool = []
17  pool.append([])
18
19
20  for row in reader:
21          if rows_so_far == 0:
22              rows_so_far +=1
23              header = row
24              for j in range (0,4):
25                  if j == 0:
26                      pool.append([])
27                      pool[0].append(u)
28                  if j == 1:
29                      pool[0].append(x)
30                  if j == 2:
31                      pool[0].append(y)
32                  if j == 3:
33                      pool[0].append(z)
```

Step 6: Now after we're done populating the header, we will get into the main data records that we need. We now append population of a city for each year. We create a new variable 'a' which is the length of our 'pool' array, extremely helpful in verification and not overplotting. In nested loops, there are a lot of conditional verification steps using 'if-else' need to be carried out. '.deepcopy()' is just a simple copy function we import from the 'copy' library. At the end of the loop, our 'rows_so_far' variable is increased by 1. AGAIN, BE VERY CAREFUL WITH THE INDENTATION.

```
32                  if j == 3:
33                      pool[0].append(z)
34          else:
35              for i in range(len(row)-2):
36                  a = len(pool)
37                  if not row == []:
38                      if i == 0 or i>=1:
39                          item = copy.deepcopy(row)
40                          r = copy.deepcopy(row)
41                          for j in range (0,4):
42                              if item[i+2] is not '':
43                                  if j == 0:
44                                      r[0] = item[j+1]+' - '+item[j]
45                                      pool.append([])
46                                      pool[a-1].append(r[0])
47                                  if j == 1:
48                                      pool[a-1].append(int(header[i+2]))
49                                  if j == 2:
50                                      if item [i+2] == '':
51                                          pool[a-1].append(int(0))
52                                      else:
53                                          pool[a-1].append(int(item[i+2]))
54                                  if j == 3:
55                                      pool[a-1].append(int(0))
56  rows_so_far += 1
57
```

Step 7: Length of the 'pool' is verified after populating the sheet.

```
54                                  if j == 3:
55                                      pool[a-1].append(int(0))
56      rows_so_far += 1
57
58  a = len(pool)
59
```

Step 8: We have to keep in mind that we don't populate the list including the header, hence 'a-1'. Also, sorting cannot be done on integer values when one record (header) has character value, hence we convert is to a 'list'.

```
58  a = len(pool)
59
60
61  list = pool[1:a-1]
62
```

Step 9: Here the list is sorted by year and population. The 'lambda' key is a simple one parameter anonymous callable function. In case of sorting, it can only do and return one thing. 'reverse' is an inbuilt python function which when true, sorts the list in DESCENDING order.

```
58  a = len(pool)
59
60
61  list = pool[1:a-1]
62
63  list.sort(key = lambda b: (b[1],b[2]),reverse = True)
64
```

Step 10: We simply add the header to the list using proxy variables.

```
63  list.sort(key = lambda b: (b[1],b[2]),reverse = True)
64
65
66  list1= []
67  list1.append([])
68  list1[0] = pool[0]
69  list1[1:a-1] = list[0:a-2]
70
```

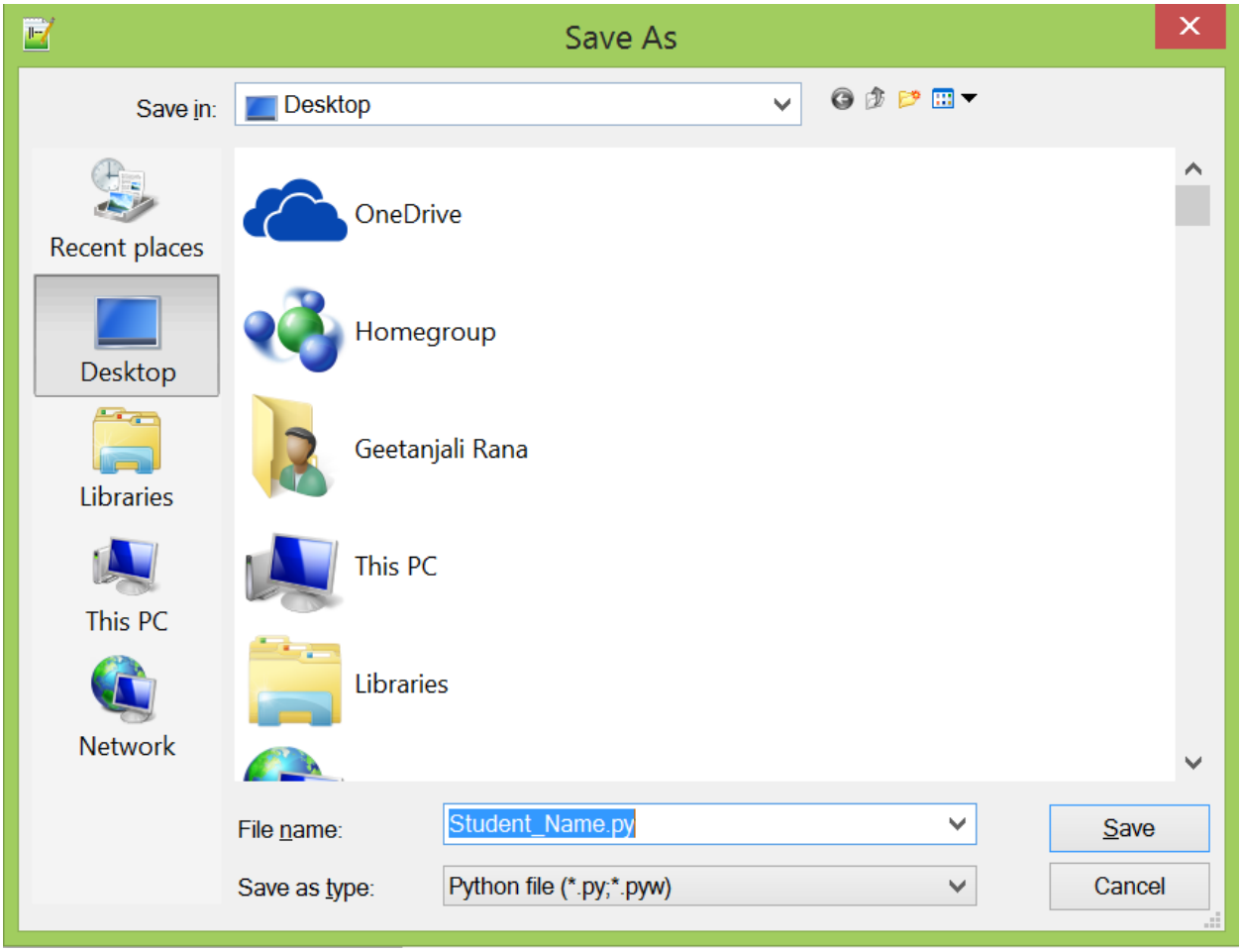
Step 11: 'csv.writer' is an output file function which creates a new CSV from the work done in the script. Here, we again use conditional verification using 'if-else' commands to make sure that our puzzle pieces(rows), fall into the right places. 'wb' is a FILE MODE which means 'Write and Binary'. If you were reading a file, you'd specifiy 'rb' (READ & Binary)

```
69  list1[1:a-1] = list[0:a-2]
70
71  mycsv = csv.writer(open('C:\\Users\\AmanSharma\\Desktop\\FinalList.csv','wb'))
72  for row in list1:
73      e = list1.index(row)
74      if row[1] != c and e != 0:
75          v=1
76          c=row[1]
77          row[3] = v
78      else:
79          if row[1] == c and e != 0:
80              v+=1
81              row[3] = v
82
83      mycsv.writerow(row)
84
```

Note: Make sure that the lines of code are indented highlighted in the above screenshot. Not having them indented might lead to errors or move lines of code out of the loop. Indentation should be a tab space and not a spacebar

Step 3: Save Program

After writing the program, save the program as Student_Name.py.



Step 4: Check Output

Run the program in command prompt using the following command.

```
C:\Users\Geetanjali\Desktop>python Largest_Cities.py
```

Note: The output file should be available at the path you have given in line 90 of the python program

```
89  # Convert the list into CSV file
90  mycsv = csv.writer(open('C:\Users\Geetanjali\Desktop\Student_Name.csv', 'wb'))
```

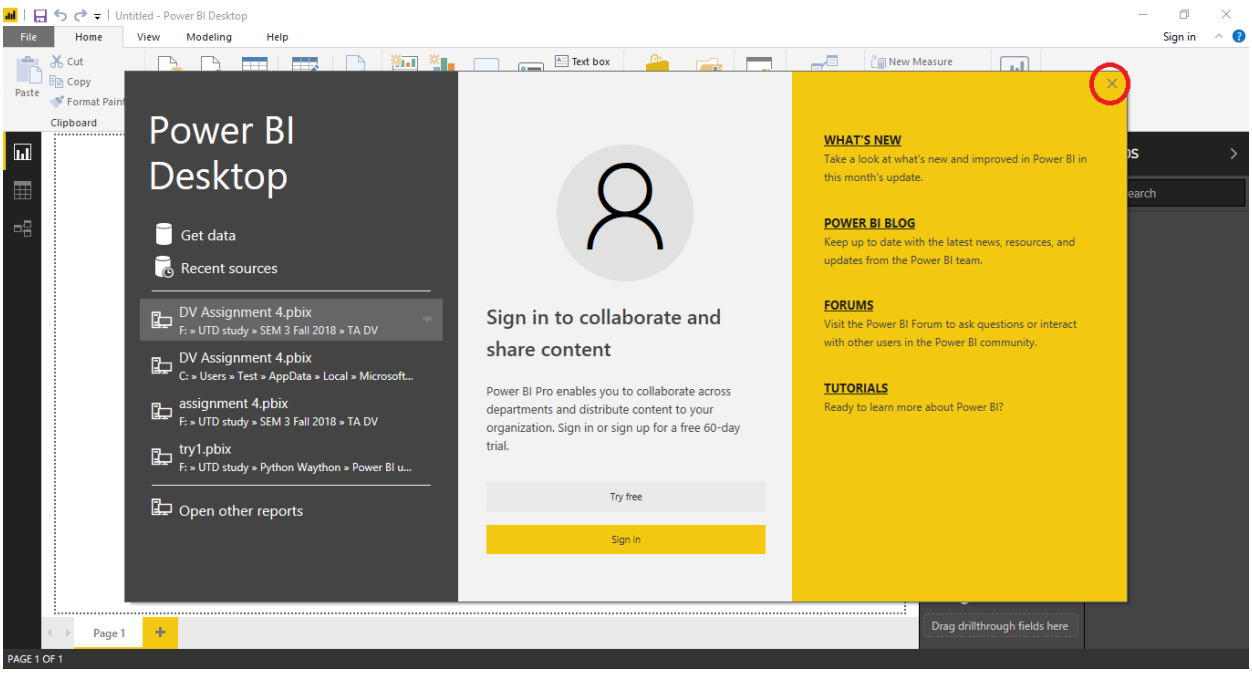
Output of the program should be saved as a csv file and should look as below:-

	A	B	C	D
1	State - Place	Year	Population	Student Last Name - Rank
2	NY - New York City	1990	7322564	1
3	CA - Los Angeles	1990	3485398	2
4	IL - Chicago	1990	2783726	3
5	TX - Houston	1990	1630553	4
6	PA - Philadelphia	1990	1585577	5

Question:
Paste the screenshot of your file

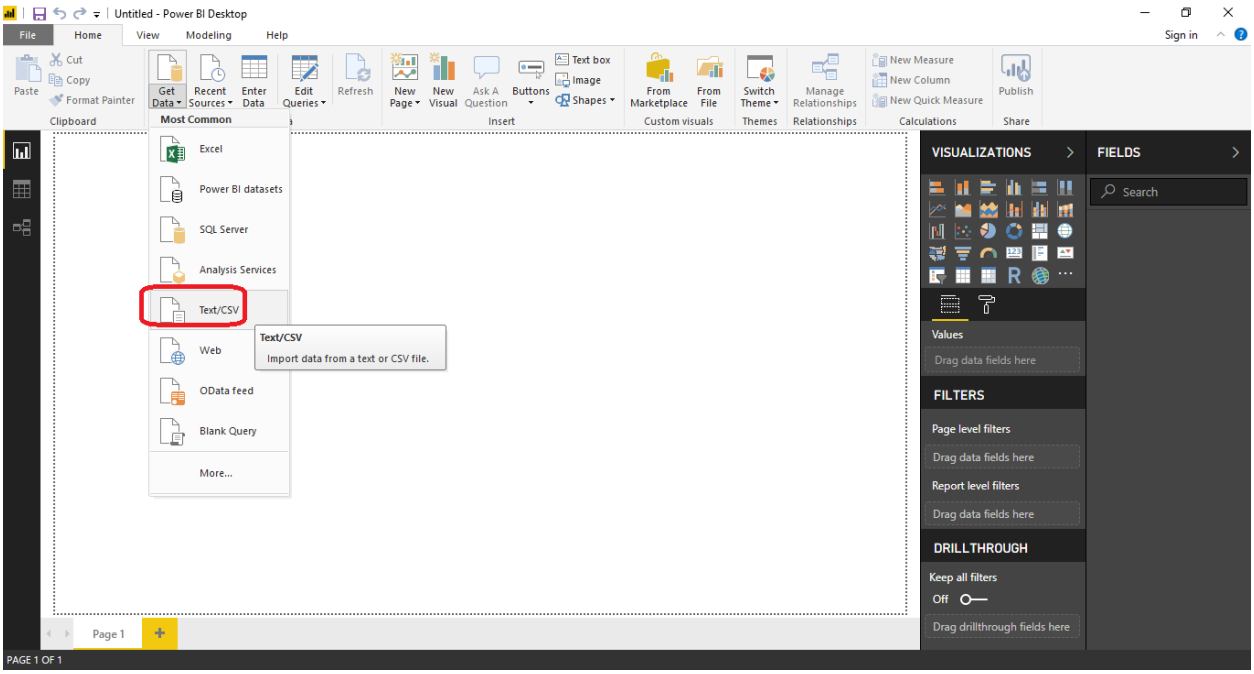
Step 5: Open PowerBI desktop application

Close the sign-in option



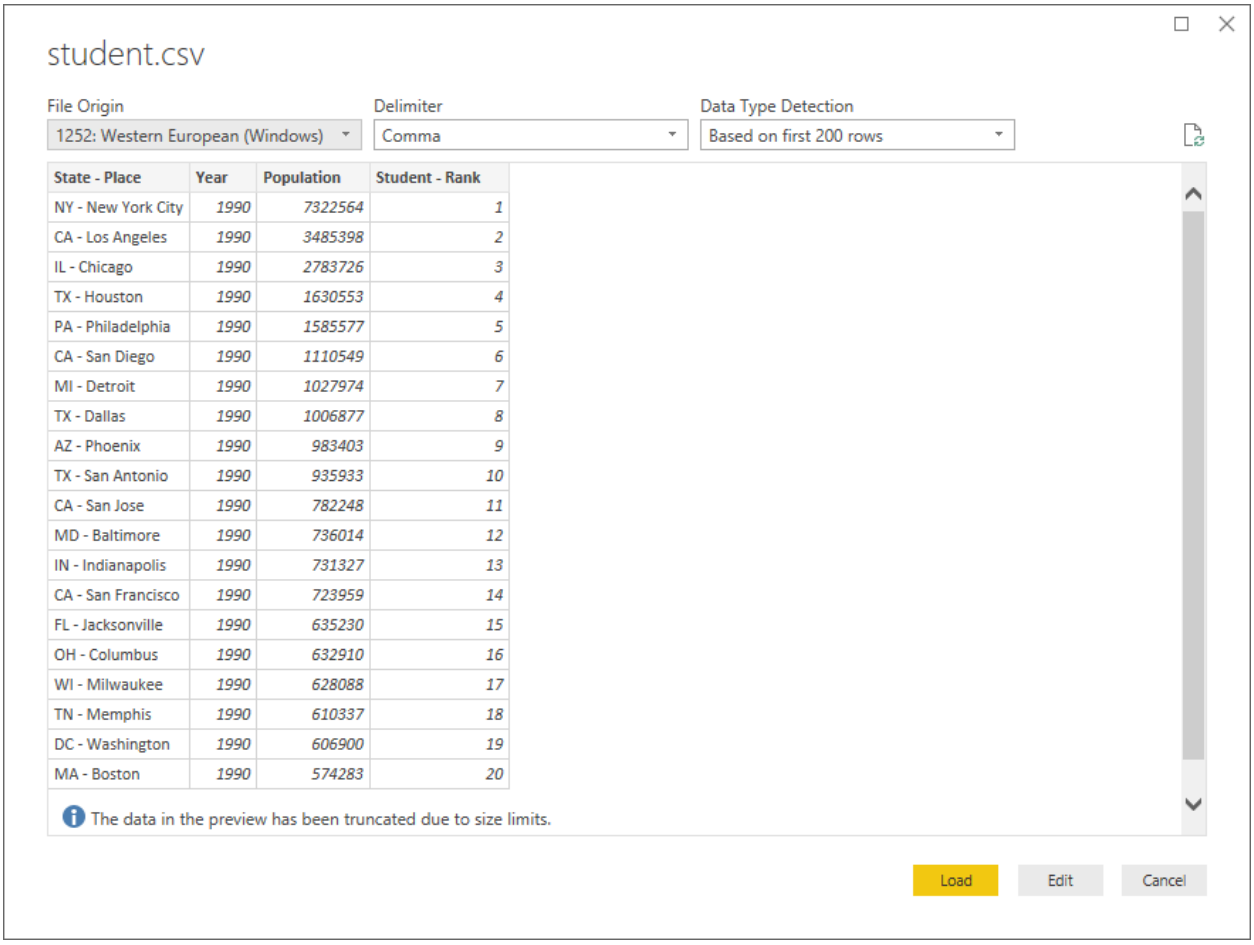
Step 6: Loading data

In Home ribbon click Get Data -> Text/CSV



Question: Paste the screen shot of the load data screen.

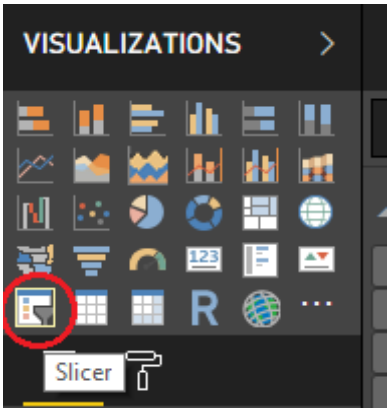
Click load



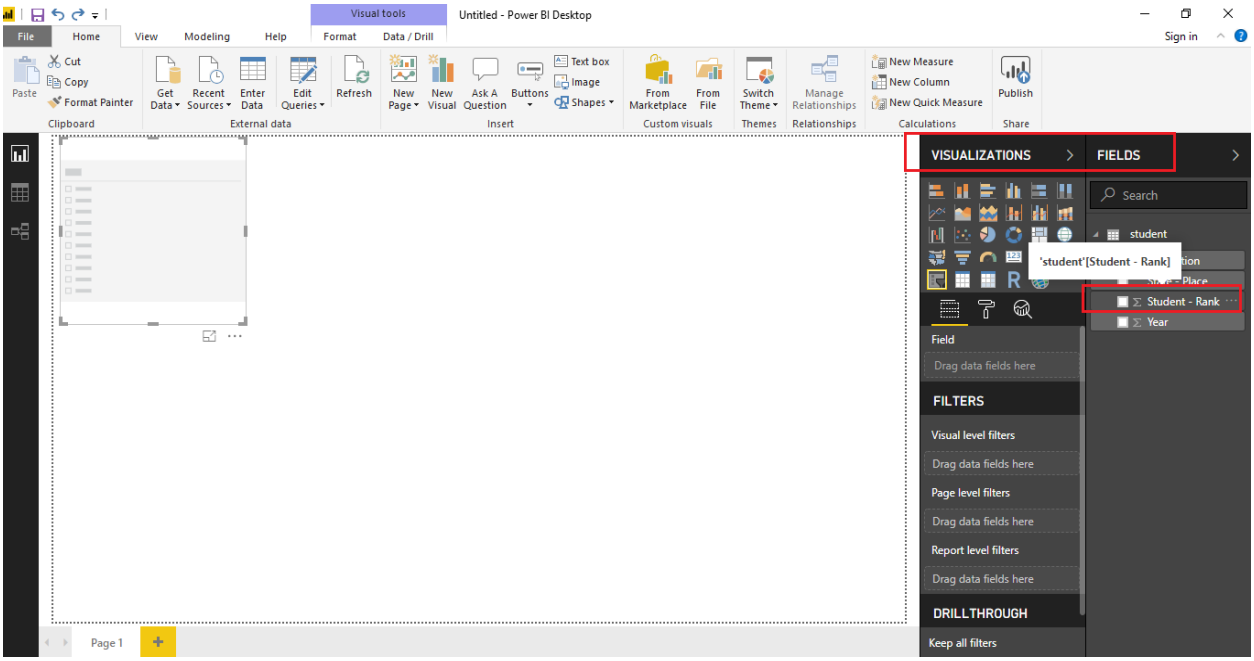
State - Place	Year	Population	Student - Rank
NY - New York City	1990	7322564	1
CA - Los Angeles	1990	3485398	2
IL - Chicago	1990	2783726	3
TX - Houston	1990	1630553	4
PA - Philadelphia	1990	1585577	5
CA - San Diego	1990	1110549	6
MI - Detroit	1990	1027974	7
TX - Dallas	1990	1006877	8
AZ - Phoenix	1990	983403	9
TX - San Antonio	1990	935933	10
CA - San Jose	1990	782248	11
MD - Baltimore	1990	736014	12
IN - Indianapolis	1990	731327	13
CA - San Francisco	1990	723959	14
FL - Jacksonville	1990	635230	15
OH - Columbus	1990	632910	16
WI - Milwaukee	1990	628088	17
TN - Memphis	1990	610337	18
DC - Washington	1990	606900	19
MA - Boston	1990	574283	20

Step 7: Creating slicer (filter) of Rank

Select slicer from Visulization pane



Select the Rank column you created as part of the previous code



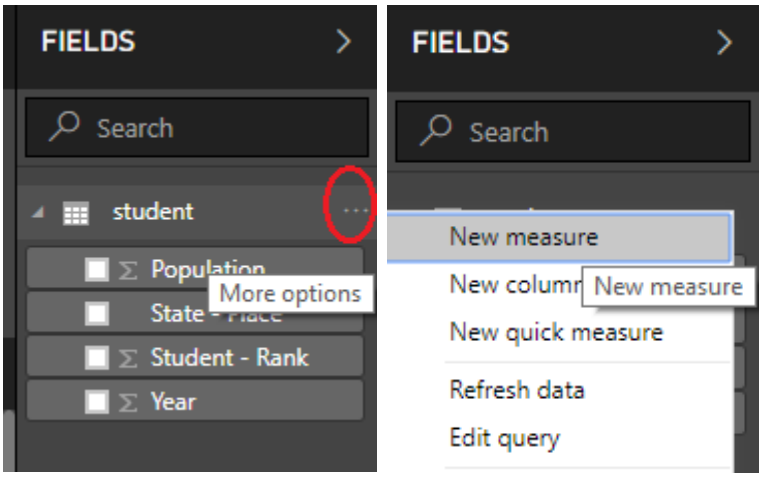
Position the rank slider to bottom right of the canvas

Question:

Paste the screen shot of the rank slider

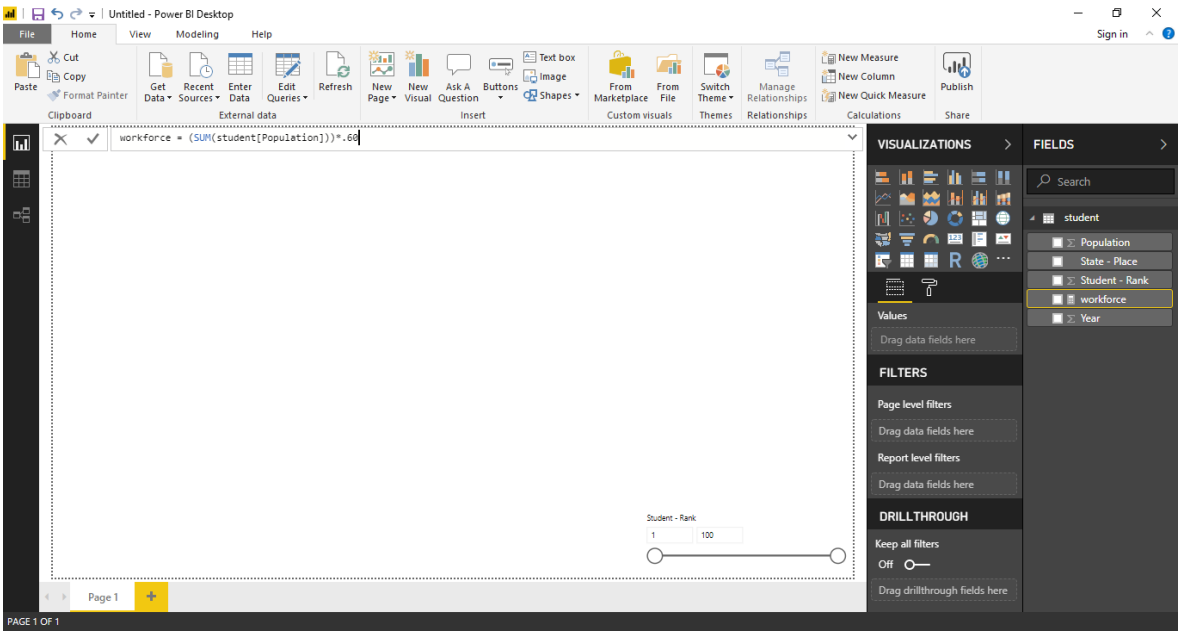
Step 8: creating new measure

click more options and select new measure

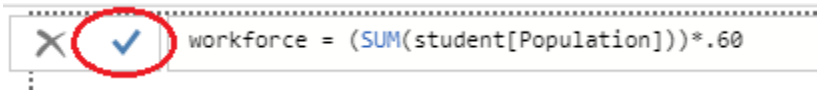


Configure the measure as: **workforce = (SUM(*student*[Population]))*.60**

Replace *student* with the name of your csv file



Click on green check mark after completion



Question: Paste the screen shot of the newly created measure

Step 9: Preparing the Scatter Chart

Place the Scatter chart from Visulizations pane on the canvas

Maintain the following fields under the details section of the visulization:

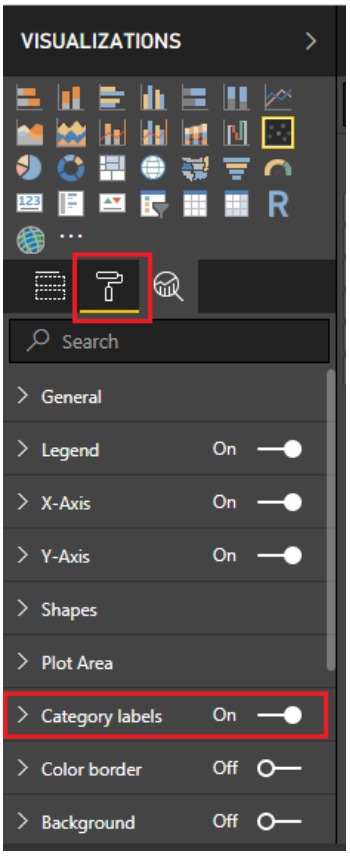
1. Drag **Student-Rank** from fields to **X axis**
2. Drag **Population** from fields to **Y axis**
3. Drag **Work Force** from fields to **Size section**
4. Drag **State-Place** from fields to **legend section**
5. Drag **Year** from fields to **play axis**

Expand the chart to fill the screen. Make sure the slider is visible too.

Question: Paste the screen shot of the chart

Change the slider to show top 5 ranks

Turn on the category labels under format



Question: Paste the screen shot of the chart

Step 10: Click on the play button to view how the population of cities change as we scan through the year.

Save the file as student_name.pbix.

Step 11: Download trial version of Camtasia software and record the video.

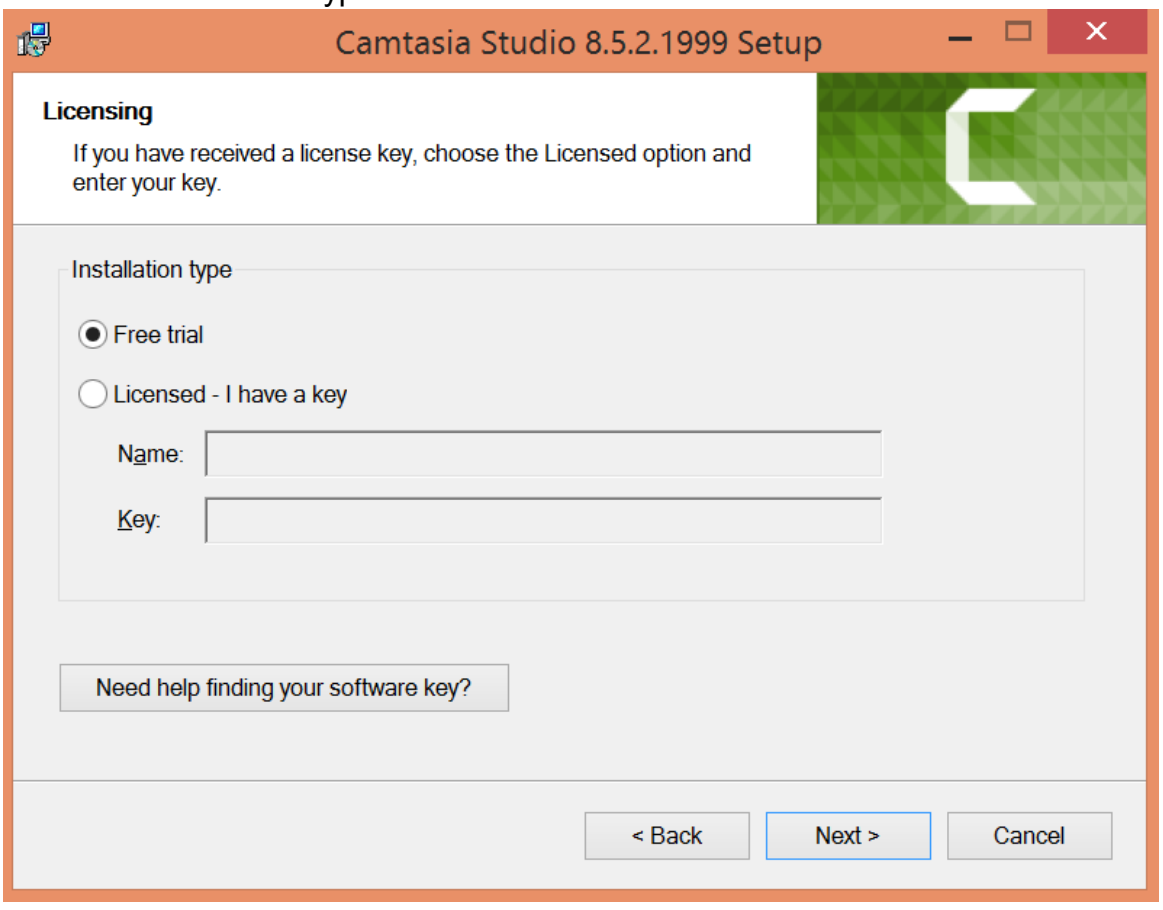
Go to the following link, click on the free trial option.

https://www.techsmith.com/camtasia.html

Sign in with your email and download the Camtasia software using the below link.

After downloading Camtasia, follow the below steps:-

1. Open the downloaded file, it will start the installation process
2. Accept the license agreement, in the setup
3. Select the installation type as 'Free Trial'

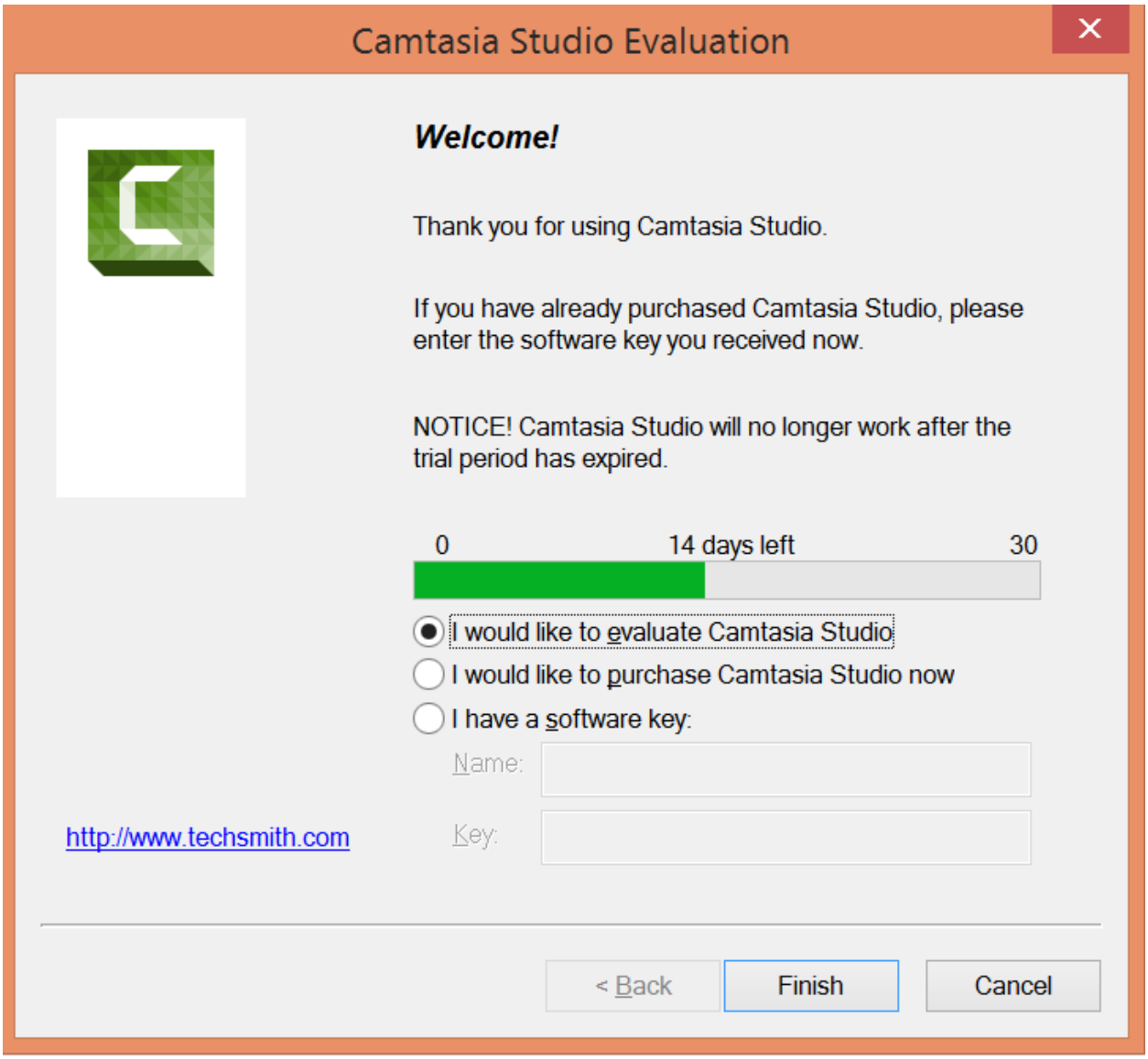


4. Click on next button, and then click finish to complete the setup.

Step 12: Record the video in Camtasia

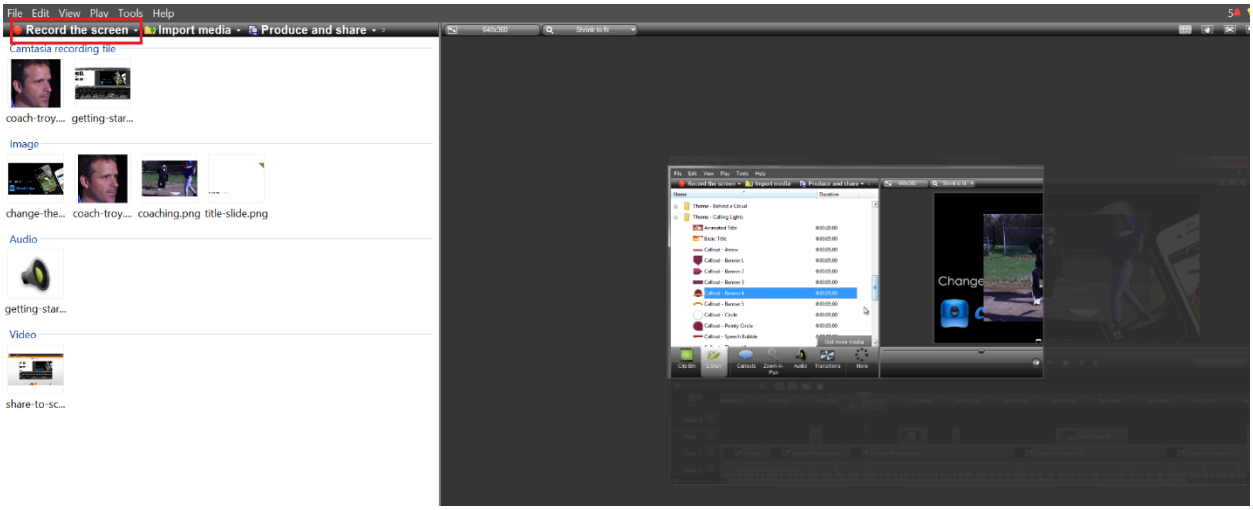
Open Camtasia, follow the below steps to record the video:

1. On attempting to open the Camtasia, following message would be prompted, select the option highlighted in the screen shot and then click on finish.

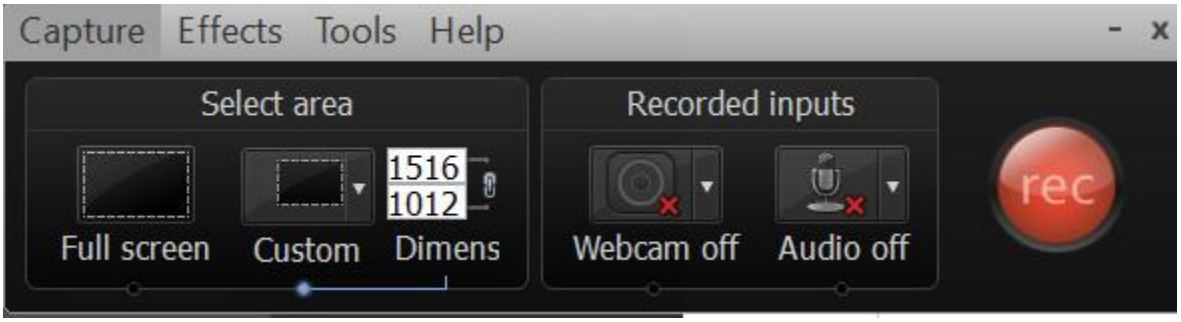


2. Open the PowerBI file created in step 10.

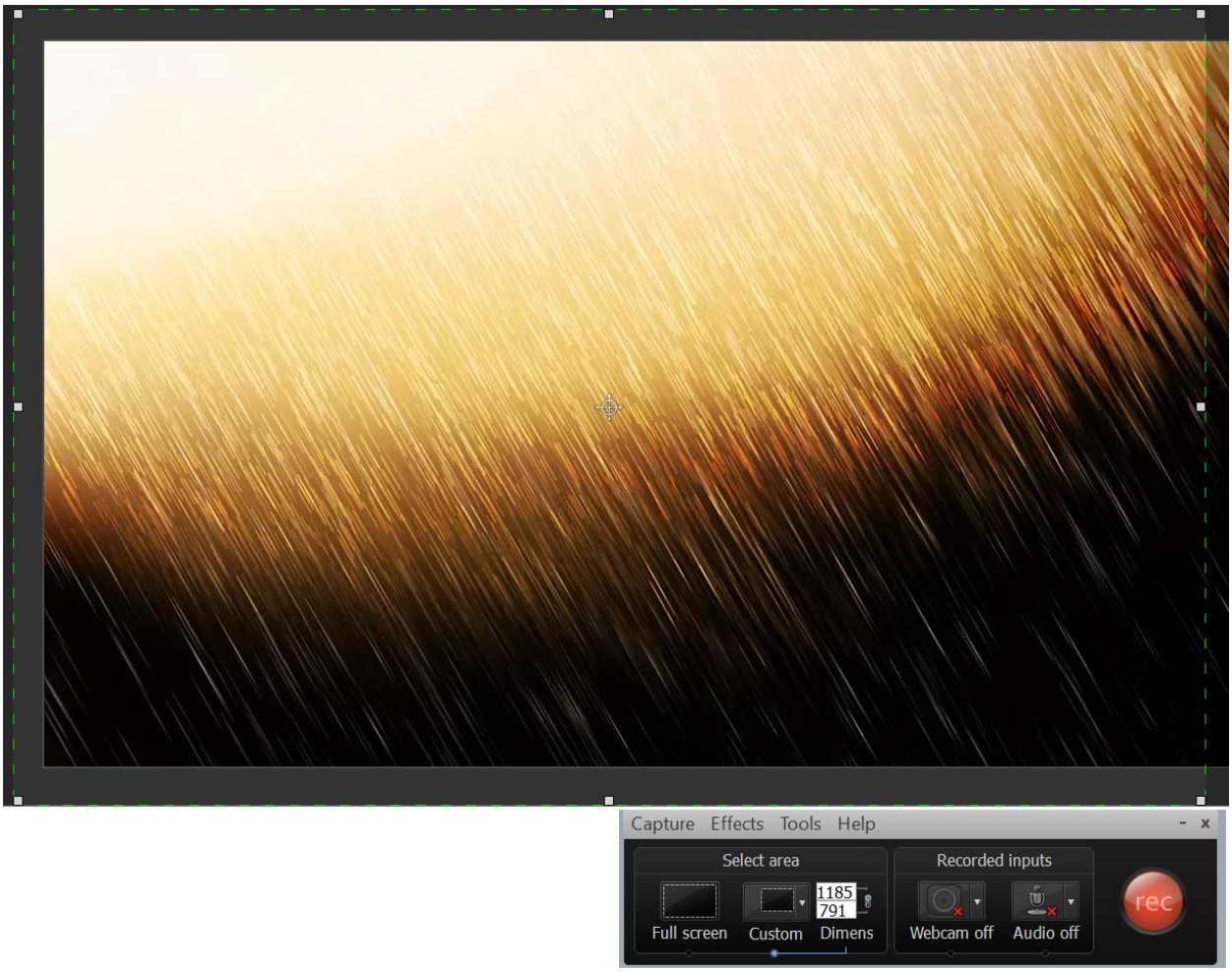
3. Click on record the screen button on the top left corner in Camtasia.



4. Select custom in the following window
   The green dotted line shows the area covered in the recording, increase or decrease the area to adjust to the size of the area to be recorded. By default the audio button is switched on, click on it to turn it off.



5. Select the area in the PowerBI canvas that needs to be recorded. (represented by green dotted line)



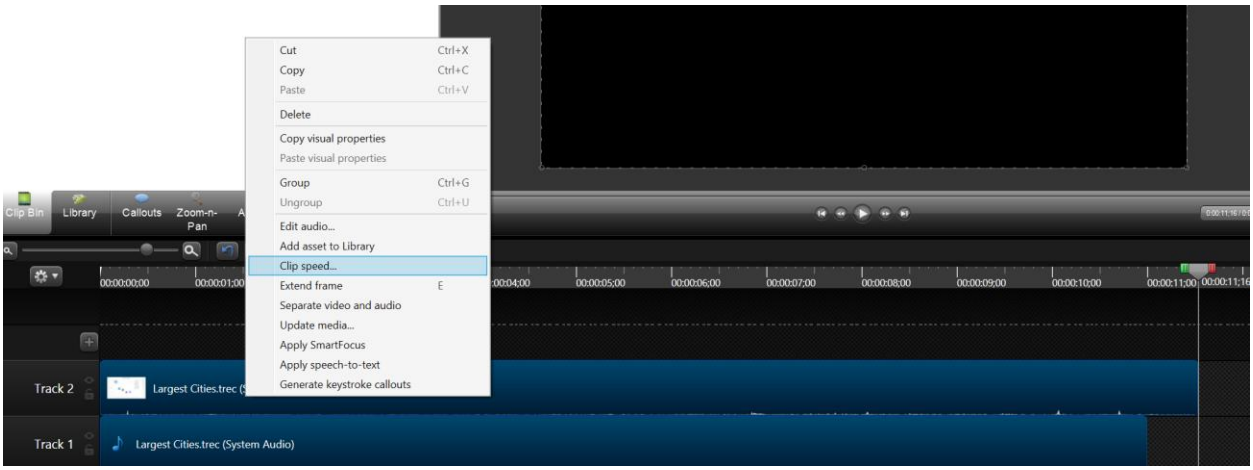
6. Click on “Rec” button in the Camtasia window and then click play in the PowerBI visulization.

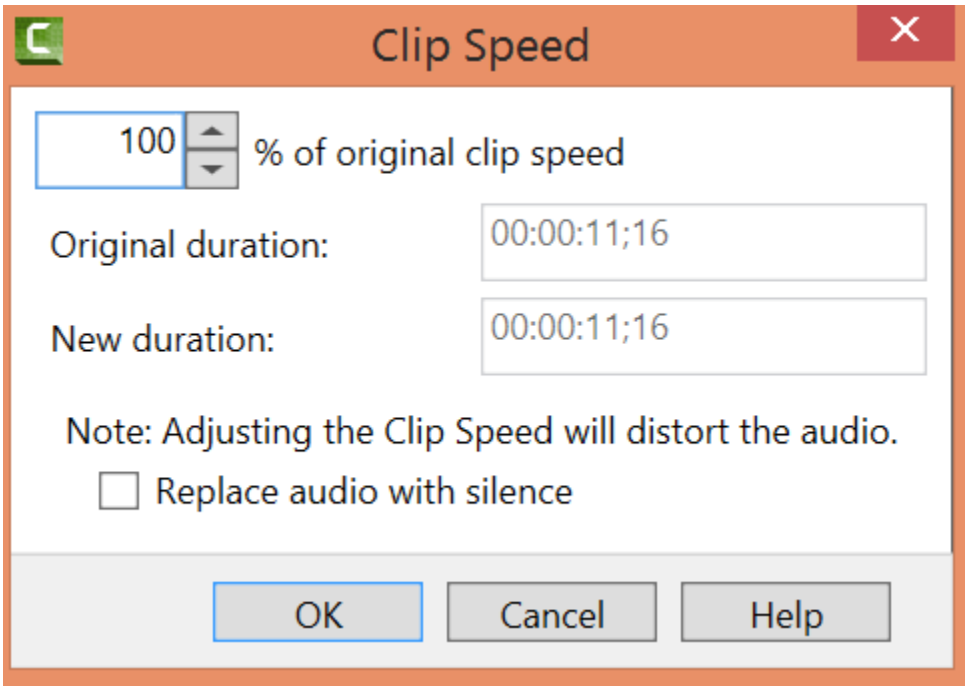
7. Press the stop recording button, once the play axis scrolls through the end.

8. Once the recording is complete, you would be able to view it in Camtasia. Click on **Save & Edit** button to save the recording.

9. Video should be visible in the right side of the Camtasia window.

10. In order to reduce the speed of the recording, right click on track 2 and select clip speed. By default the clip speed is 100, maintain it as 50 and then replay the video.





11. Finally, save the project using the below menu path
    File -> Save Project

Step 13: Attach assignments in elearning

1. Attach the **assignment document** in Microsoft Word
2. Attach the python **.py file** created in step 3
3. Attach the **.pbix file** created in Step 10
4. Attach the **Camtasia video** created in Step 18 as a **mp4** file type.