In [66]:

```
#Assignment 4: Creating graphs for your data
# import libraries
%matplotlib inline
import pandas
import numpy as np
import seaborn as sn
import matplotlib.pyplot as plt

print("avoid run time error message")
pandas.set_option('display.float_format', lambda x:'%f'%x)
```

In [67]:

```
#Import dataset
data = pandas.read_csv("gapminder.csv", low_memory = False)

#Convert all variable names to lowercaes
data.columns = map(str.lower, data.columns)
```

In [103]:

```
# Set missing values to "nan"
data["incomeperperson"] = data["incomeperperson"].replace(0, np.nan)
data["suicideper100th"] = data["suicideper100th"].replace(0, np.nan)
data["employrate"] = data["employrate"].replace(0, np.nan)

#set avoid run time error message
data['incomeperperson'] = data['incomeperperson'].convert_objects(convert_numeric=True)
data['suicideper100th'] = data['suicideper100th'].convert_objects(convert_numeric=True)
```

```
C:\Users\Laptop\Anaconda3\lib\site-packages\ipykernel\__main__.py:7: FutureWarning: convert_objects is deprec
ated.  Use the data-type specific converters pd.to_datetime, pd.to_timedelta and pd.to_numeric.
C:\Users\Laptop\Anaconda3\lib\site-packages\ipykernel\__main__.py:8: FutureWarning: convert_objects is deprec
ated.  Use the data-type specific converters pd.to_datetime, pd.to_timedelta and pd.to_numeric.
```

In [104]:

```
#Create varible Income Categories (based on the worldbank information)
def INCOMECAT(row):
    if row['incomeperperson'] <= 1035:
        return 1
    elif 1035 < row['incomeperperson']  <= 4085:
        return 2
    elif 4085 < row['incomeperperson'] <= 12615:
        return 3
    else:
        return 4

data["INCOMECAT"] = data.apply(lambda row: INCOMECAT(row), axis=1)
data['INCOMECAT'] = data['INCOMECAT'].astype('category')
data['INCOMECAT'] = data['INCOMECAT'].cat.rename_categories(['low','lower middle', 'upper middle','high'])
```

In [ ]:

```
#Create varible Asia

def Asia(row):
    if row['country'] == "":
        return 1
    else:
        return 0
```

```
In [105]: #Create a subset of the dataset to include only variables of interest
          sub1 = data[["country", "incomeperperson","suicideper100th","employrate", "INCOMECAT"]]
          print('preview dataset')
          print(sub1.head(n=10))
```

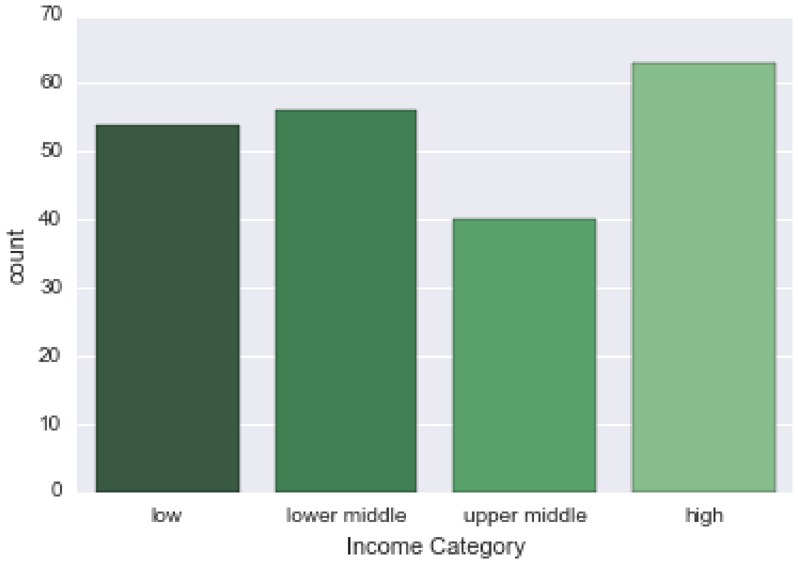
```
preview dataset
              country  incomeperperson  suicideper100th        employrate  \
0         Afghanistan              nan         6.684385  55.7000007629394
1             Albania      1914.996551         7.699330  51.4000015258789
2             Algeria      2231.993335         4.848770              50.5
3             Andorra     21943.339898         5.362179
4              Angola      1381.004268        14.554677  75.6999969482422
5  Antigua and Barbuda     11894.464075         2.161843
6           Argentina     10749.419238         7.765584  58.4000015258789
7             Armenia      1326.741757         3.741588  40.0999984741211
8               Aruba              nan              nan
9           Australia     25249.986061         8.470030              61.5

      INCOMECAT
0          high
1  lower middle
2  lower middle
3          high
4  lower middle
5  upper middle
6  upper middle
7  lower middle
8          high
9          high
```

In [106]:

```
# Make a categorical count plot for the different income groups.

sn.countplot(x='INCOMECAT', data = sub1, palette = 'Greens_d')
plt.xlabel("Income Category")
plt.ylabel("count")
plt.show(block=True)
```



In [107]:

```
#create DEVELOPED (boolean) row.

def DEVELOPED (row):
    if row["incomeperperson"] >= 12615.0:
        return 1
    else:
        return 0

data["DEVELOPED"] = data.apply(lambda row: DEVELOPED(row), axis =1)
```

In [112]:

```
#create sub2 dataset to include only developed countries ("incomeperperson" >= 12615.0)
sub2 = sub1[(data["DEVELOPED"] != 0)]
print('preview dataset')
print(sub2.head(n=100))
```

```
preview dataset
                  country  incomeperperson  suicideper100th        employrate  \
3                 Andorra     21943.339898         5.362179
9               Australia     25249.986061         8.470030              61.5
10                Austria     26692.984107        13.094370  57.0999984741211
12                Bahamas     19630.540547         3.374416  66.5999984741211
17                Belgium     24496.048264        15.953850  48.5999984741211
20                Bermuda     62682.147006              nan
26                 Brunei     17092.460004         1.370002  63.7999992370606
32                 Canada     25575.352623        10.100990              63.5
48                 Cyprus     15313.859347         2.206169  59.0999984741211
50                Denmark     30532.277044         8.973104  63.0999984741211
63                Finland     27110.731591        16.234370  57.2000007629394
64                 France     22878.466567        14.091530  51.2000007629394
69                Germany     25306.187193         9.211085              53.5
72                 Greece     13577.879885         2.816705  49.5999984741211
73              Greenland     20751.893424              nan
83       Hong Kong, China     35536.072471              nan                59
85                Iceland     33945.314422        11.426181  73.5999984741211
90                Ireland     27595.091347        10.365070  59.9000015258789
91                 Israel     22275.751661         5.931845  51.2999992370606
92                  Italy     18982.269285         4.930045  46.4000015258789
94                  Japan     39309.478859        18.946930  57.2999992370606
100           Korea, Rep.     16372.499781        22.404560  58.9000015258789
109         Liechtenstein     81647.100031              nan
111            Luxembourg     52301.587179        12.405918              53.5
112          Macao, China     33923.313868              nan  63.5999984741211
127                Monaco    105147.437697        11.151073
136           Netherlands     26551.844238         8.164005  61.2999992370606
139           New Zealand     14778.163929        12.179760                65
144                Norway     39972.352768        10.823000                65
155           Puerto Rico     15822.112141              nan  42.4000015258789
156                 Qatar     33931.832079         2.515721                76
165            San Marino     31993.200694         6.087671
173             Singapore     32535.832512         9.127511  62.4000015258789
175              Slovenia     12729.454400        19.422610  55.9000015258789
179                 Spain     15461.758372         5.888479              52.5
184                Sweden     32292.482984        11.115830  60.7000007629394
185           Switzerland     37662.751250        13.239810  64.3000030517578
```

```
201  United Arab Emirates     21087.394125         1.392951  75.1999969482422
202        United Kingdom     28033.489283         6.014659  59.2999992370606
203         United States     37491.179523         9.927033  62.2999992370606

    INCOMECAT
3        high
9        high
10       high
12       high
17       high
20       high
26       high
32       high
48       high
50       high
63       high
64       high
69       high
72       high
73       high
83       high
85       high
90       high
91       high
92       high
94       high
100      high
109      high
111      high
112      high
127      high
136      high
139      high
144      high
155      high
156      high
165      high
173      high
175      high
```

```
179      high
184      high
185      high
201      high
202      high
203      high
```

In [109]:

```
#Quantitative variables graphing study
#Describe each of the quantitative variables

desc1 = sub2['incomeperperson'].describe()
print(desc1)
```

```
count       40.000000
mean     30655.347961
std      18074.852378
min      12729.454400
25%      20471.555205
50%      26622.414172
75%      33935.202664
max     105147.437697
Name: incomeperperson, dtype: float64
```

In [110]:

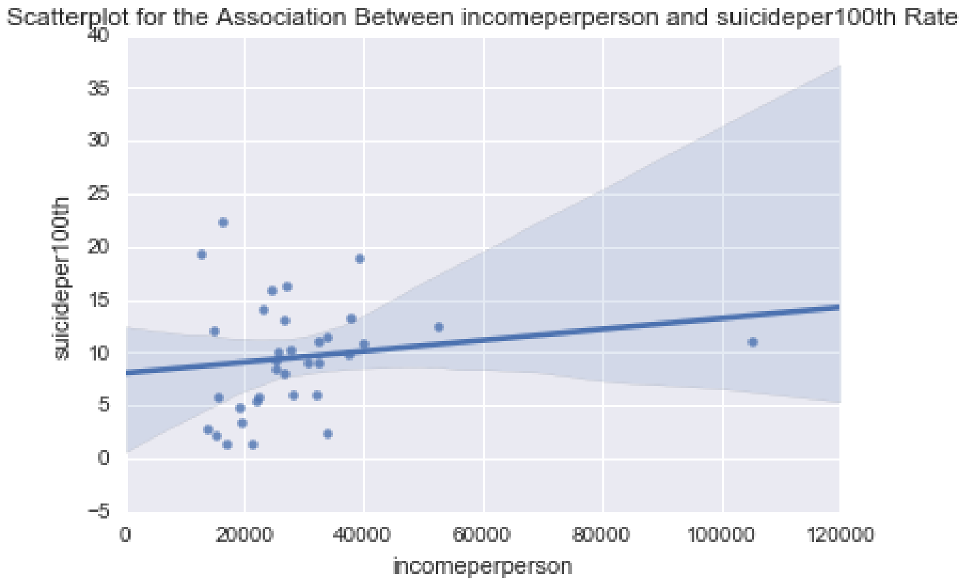
```
desc2 = sub2["suicideper100th"].describe()
print(desc2)
```

```
count   34.000000
mean     9.550572
std      5.290811
min      1.370002
25%      5.899320
50%      9.569059
75%     12.349379
max     22.404560
Name: suicideper100th, dtype: float64
```

In [111]:

```
#Quantitative plot study
# incomeperperson v.s suicideper100th rate (All Developed Countries)
#The plot indicates that the two variables have a low positive correlated relationship.


scat1 = sn.regplot(x='incomeperperson',y='suicideper100th', fit_reg=True, data=sub2)
plt.xlabel('incomeperperson')
plt.ylabel('suicideper100th')
plt.title("Scatterplot for the Association Between incomeperperson and suicideper100th Rate")
plt.show()
```



In [127]:

```
#create ASIA countries (boolean) row.

def ASIA (row):
    if row["country"] == "Brunei":
        return 1
    elif row["country"] == "Cyprus":
        return 1
    elif row["country"] == "Hong Kong, China":
        return 1
    elif row["country"] == "Israel":
        return 1
    elif row["country"] == "Japan":
        return 1
    elif row["country"] == "Korea, Rep.":
        return 1
    elif row["country"] == "Macao, China":
        return 1
    elif row["country"] == "Qatar":
        return 1
    elif row["country"] == "Singapore":
        return 1
    elif row["country"] == "United Arab Emirates":
        return 1
    else:
        return 0

data["ASIA"] = data.apply(lambda row: ASIA(row), axis =1)
```

In [128]:

```
#create sub3 dataset to include only asian developed countries ("incomeperperson" >= 12615.0)
sub3 = sub1[(data["ASIA"] == 1)]
print('preview dataset')
print(sub3.head(n=100))
```

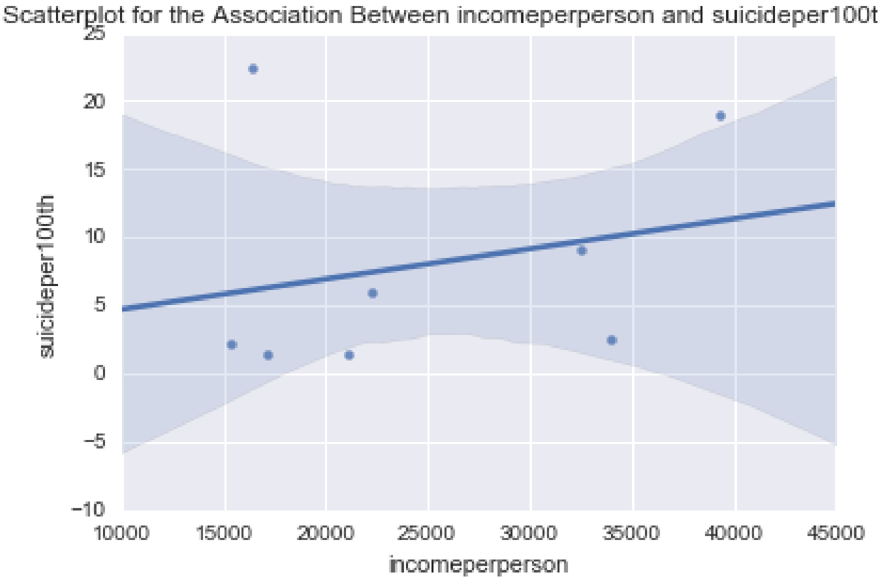
```
preview dataset
                  country  incomeperperson  suicideper100th        employrate  \
26                 Brunei     17092.460004         1.370002  63.7999992370606
48                 Cyprus     15313.859347         2.206169  59.0999984741211
83      Hong Kong, China     35536.072471              nan                59
91                 Israel     22275.751661         5.931845  51.2999992370606
94                  Japan     39309.478859        18.946930  57.2999992370606
100           Korea, Rep.     16372.499781        22.404560  58.9000015258789
112          Macao, China     33923.313868              nan  63.5999984741211
156                 Qatar     33931.832079         2.515721                76
173             Singapore     32535.832512         9.127511  62.4000015258789
201  United Arab Emirates     21087.394125         1.392951  75.1999969482422

    INCOMECAT
26       high
48       high
83       high
91       high
94       high
100      high
112      high
156      high
173      high
201      high
```

In [130]:

```
#Quantitative plot study
# incomeperperson v.s suicideper100th rate (All Asian Developed Countries)
#The plot indicates that the two variables have a positive correlated relationship. but also a lot of varibility.

scat2 = sn.regplot(x='incomeperperson',y='suicideper100th', fit_reg=True, data=sub3)
plt.xlabel('incomeperperson')
plt.ylabel('suicideper100th')
plt.title("Scatterplot for the Association Between incomeperperson and suicideper100th Rate")
plt.show()
```



In [ ]: