Python Project

Nischal Pant

2023-08-25

The libraries you will use are already loaded for you below

In [44]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from itertools import chain
```

First glace

Importing packages

In [45]:

```
level_salary = pd.read_csv('Levels_Fyi_Salary_Data.csv')
cost_of_living = pd.read_csv('cost_of_living.csv')
country_codes = pd.read_excel('country_codes.xlsx')
ds_salaries = pd.read_csv('ds_salaries.csv')

print(level_salary.columns)
print(cost_of_living.columns)
print(country_codes.columns)
print(ds_salaries.columns)
```

```
Index(['timestamp', 'company', 'level', 'title', 'totalyearlycompensation',
       'location', 'yearsofexperience', 'yearsatcompany', 'tag', 'basesalary',
       'stockgrantvalue', 'bonus', 'gender', 'otherdetails', 'cityid', 'dmaid',
       'rowNumber', 'Masters_Degree', 'Bachelors_Degree', 'Doctorate_Degree',
       'Highschool', 'Some_College', 'Race_Asian', 'Race_White',
       'Race_Two_Or_More', 'Race_Black', 'Race_Hispanic', 'Race', 'Education'],
      dtype='object')
Index(['Rank', 'City', 'Cost of Living Index', 'Rent Index',
       'Cost of Living Plus Rent Index', 'Groceries Index',
       'Restaurant Price Index', 'Local Purchasing Power Index'],
      dtype='object')
Index(['Country', 'Alpha-2 code', 'Alpha-3 code', 'Numeric'], dtype='object')
Index(['Unnamed: 0', 'work_year', 'experience_level', 'employment_type',
       'job_title', 'salary', 'salary_currency', 'salary_in_usd',
       'employee_residence', 'remote_ratio', 'company_location',
       'company_size'],
      dtype='object')
```

In [46]:

```
#here we can look into what the files look like and possibly think of ways we can comb
```

In [47]:

```
#first lets see what our orignal dataset looks like
print(cost_of_living.head(10))
print(cost_of_living.dtypes)

# there is alot of data so its best to group them together, and we should group them b
city_country = cost_of_living['City'].str.split(', ', expand=True)
cost_of_living['City'] = city_country[0]
cost_of_living['Country'] = city_country[1]
print(cost_of_living.head(10))

#here we can see some columns that we dont really need for our analysis are they are i
cost_of_living = cost_of_living.drop(columns=['City', 'Rank'])
column_order = ['Country', 'Cost of Living Index', 'Rent Index', 'Cost of Living Plus
                'Groceries Index', 'Restaurant Price Index', 'Local Purchasing Power I
cost_of_living = cost_of_living[column_order]

#we can now put them in order of whatever we like. Having the city names in the front
```

```
   Rank                 City  Cost of Living Index  Rent Index  \
0   NaN      Hamilton, Bermuda                149.02       96.10
1   NaN    Zurich, Switzerland                131.24       69.26
2   NaN     Basel, Switzerland                130.93       49.38
3   NaN       Zug, Switzerland                128.13       72.12
4   NaN    Lugano, Switzerland                123.99       44.99
5   NaN  Lausanne, Switzerland                122.03       59.55
6   NaN        Beirut, Lebanon                120.47       27.76
7   NaN      Bern, Switzerland                118.16       46.12
8   NaN    Geneva, Switzerland                114.05       75.05
9   NaN      Stavanger, Norway                104.61       35.38

   Cost of Living Plus Rent Index  Groceries Index  Restaurant Price Index  \
0                          124.22           157.89                  155.22
1                          102.19           136.14                  132.52
2                           92.70           137.07                  130.95
3                          101.87           132.61                  130.93
4                           86.96           129.17                  119.80
5                           92.74           122.56                  127.01
6                           77.01           141.33                  116.95
7                           84.39           118.37                  120.88
8                           95.77           112.70                  126.31
9                           72.16           102.46                  107.51

   Local Purchasing Power Index
0                         79.43
1                        129.79
2                        111.53
3                        143.40
4                        111.96
5                        127.01
6                         15.40
7                        112.46
8                        120.60
9                         85.90
Rank                              float64
City                               object
Cost of Living Index              float64
Rent Index                        float64
Cost of Living Plus Rent Index    float64
Groceries Index                   float64
Restaurant Price Index            float64
Local Purchasing Power Index      float64
dtype: object
   Rank       City  Cost of Living Index  Rent Index  \
0   NaN   Hamilton                149.02       96.10
1   NaN     Zurich                131.24       69.26
2   NaN      Basel                130.93       49.38
3   NaN        Zug                128.13       72.12
4   NaN     Lugano                123.99       44.99
5   NaN   Lausanne                122.03       59.55
6   NaN     Beirut                120.47       27.76
7   NaN       Bern                118.16       46.12
8   NaN     Geneva                114.05       75.05
9   NaN  Stavanger                104.61       35.38

   Cost of Living Plus Rent Index  Groceries Index  Restaurant Price Index  \
0                          124.22           157.89                  155.22
1                          102.19           136.14                  132.52
2                           92.70           137.07                  130.95
```

```
3                        101.87           132.61                   130.93
4                         86.96           129.17                   119.80
5                         92.74           122.56                   127.01
6                         77.01           141.33                   116.95
7                         84.39           118.37                   120.88
8                         95.77           112.70                   126.31
9                         72.16           102.46                   107.51

   Local Purchasing Power Index      Country
0                         79.43      Bermuda
1                        129.79  Switzerland
2                        111.53  Switzerland
3                        143.40  Switzerland
4                        111.96  Switzerland
5                        127.01  Switzerland
6                         15.40      Lebanon
7                        112.46  Switzerland
8                        120.60  Switzerland
9                         85.90       Norway
```

In [48]:

```python
level_salary.dropna(inplace=True)
cost_of_living.dropna(inplace=True)
country_codes.dropna(inplace=True)
ds_salaries.dropna(inplace=True)

print(cost_of_living.head(5))

#now that we have grouped together by country names and avraged our index columns. A r
#more organized.

avg_cost_of_living = cost_of_living.groupby('Country').mean().reset_index()
print(avg_cost_of_living.head())

#it seems that some countries are in code so we can now use the country code data and
```

```
       Country  Cost of Living Index  Rent Index  \
0      Bermuda                149.02       96.10   
1  Switzerland                131.24       69.26   
2  Switzerland                130.93       49.38   
3  Switzerland                128.13       72.12   
4  Switzerland                123.99       44.99   

   Cost of Living Plus Rent Index  Groceries Index  Restaurant Price Index  \
0                          124.22           157.89                  155.22   
1                          102.19           136.14                  132.52   
2                           92.70           137.07                  130.95   
3                          101.87           132.61                  130.93   
4                           86.96           129.17                  119.80   

   Local Purchasing Power Index  
0                         79.43  
1                        129.79  
2                        111.53  
3                        143.40  
4                        111.96  
       Country  Cost of Living Index  Rent Index  \
0           AK                 91.23   39.290000   
1           AL                 78.82   28.190000   
2           AR                 59.26   25.600000   
3           AZ                 65.79   34.833333   
4  Afghanistan                 21.35    3.170000   

   Cost of Living Plus Rent Index  Groceries Index  Restaurant Price Index  \
0                       66.880000        97.950000                   78.76   
1                       55.090000        84.300000                   75.48   
2                       43.480000        57.280000                   64.63   
3                       51.276667        63.963333                   69.79   
4                       12.830000        15.220000                   14.85   

   Local Purchasing Power Index  
0                    118.630000  
1                     84.930000  
2                    131.070000  
3                    107.683333  
4                     22.790000  
```

In [6]:

```python
print(country_codes.columns)
print(country_codes.head(10))

#there are 4 variables and now we can find a way to merge country codes and our cost c
```

```
Index(['Country', 'Alpha-2 code', 'Alpha-3 code', 'Numeric'], dtype='object')
               Country Alpha-2 code Alpha-3 code  Numeric
0          Afghanistan           AF          AFG        4
1              Albania           AL          ALB        8
2              Algeria           DZ          DZA       12
3       American Samoa           AS          ASM       16
4              Andorra           AD          AND       20
5               Angola           AO          AGO       24
6             Anguilla           AI          AIA      660
7           Antarctica           AQ          ATA       10
8  Antigua and Barbuda           AG          ATG       28
9            Argentina           AR          ARG       32
```

In [19]:
```
#the easist way we can combine the cost of living and country code is by creating a ma
#country codes to country names in our dataset
code_to_name_mapping = dict(zip(country_codes['Alpha-2 code'], country_codes['Country'

#now we can replace the country codes in avg_of_living with the corresponding country
avg_cost_of_living['Country'] = avg_cost_of_living['Country'].replace(code_to_name_map
merged_data = pd.merge(avg_cost_of_living, country_codes, on='Country', how='left')
column_order = ['Country', 'Cost of Living Index','Cost of Living Plus Rent Index', 'R
merged_data_cofl = merged_data[column_order]
print(merged_data_cofl.head())

#in the code above, I joined the two datasets together and now we have the indexs and
#there are still two letter country names which suggests that they are actually US sta
```

```
       Country  Cost of Living Index  Cost of Living Plus Rent Index  \
0           AK                 91.23                       66.880000   
1      Albania                 78.82                       55.090000   
2    Argentina                 59.26                       43.480000   
3   Azerbaijan                 65.79                       51.276667   
4  Afghanistan                 21.35                       12.830000   

   Rent Index  Groceries Index  Restaurant Price Index  \
0   39.290000        97.950000                   78.76   
1   28.190000        84.300000                   75.48   
2   25.600000        57.280000                   64.63   
3   34.833333        63.963333                   69.79   
4    3.170000        15.220000                   14.85   

   Local Purchasing Power Index  
0                    118.630000  
1                     84.930000  
2                    131.070000  
3                    107.683333  
4                     22.790000  
```

In [8]:
```
merged_data_cofl = merged_data_cofl[merged_data_cofl['Country'].str.len() != 2]
print(merged_data_cofl)


#here I have just removed the US states that were from the other dataset. Now it looks
```

```
            Country  Cost of Living Index  Cost of Living Plus Rent Index  \
1           Albania                 78.82                       55.090000   
2         Argentina                 59.26                       43.480000   
3        Azerbaijan                 65.79                       51.276667   
4       Afghanistan                 21.35                       12.830000   
5           Albania                 38.68                       25.860000   
..              ...                   ...                             ...   
162  Holy See (the)                 67.75                       53.580000   
163       Venezuela                 45.31                       29.730000   
164         Vietnam                 37.93                       27.195000   
167          Zambia                 33.57                       22.600000   
168        Zimbabwe                 45.69                       28.750000   

     Rent Index  Groceries Index  Restaurant Price Index  \
1     28.190000        84.300000                   75.48   
2     25.600000        57.280000                   64.63   
3     34.833333        63.963333                   69.79   
4      3.170000        15.220000                   14.85   
5     11.330000        30.990000                   29.86   
..          ...              ...                     ...   
162   37.520000        65.070000                   77.50   
163   12.080000        37.600000                   48.60   
164   15.030000        39.285000                   20.39   
167   10.180000        32.850000                   23.63   
168    9.560000        37.050000                   39.05   

     Local Purchasing Power Index  
1                       84.930000  
2                      131.070000  
3                      107.683333  
4                       22.790000  
5                       31.150000  
..                            ...  
162                    130.940000  
163                     15.870000  
164                     30.770000  
167                     37.480000  
168                     17.590000  

[150 rows x 7 columns]
```

In [9]:

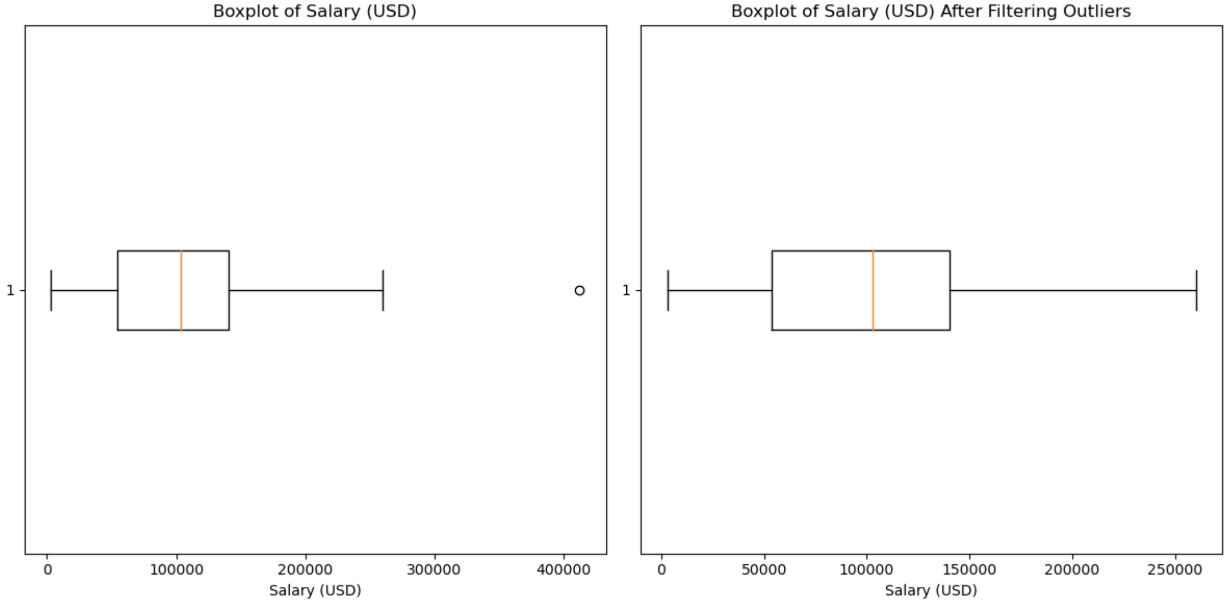
```
print(ds_salaries.head(5))
print(ds_salaries.columns)

#this is the next dataset that we will be filtering and merging with our previously me
#visual before proceeding. This is the same dataset which we analysed in our R project
```

```
   Unnamed: 0  work_year experience_level employment_type  \
0           0       2020               MI              FT   
1           1       2020               SE              FT   
2           2       2020               SE              FT   
3           3       2020               MI              FT   
4           4       2020               SE              FT   

                    job_title  salary salary_currency  salary_in_usd  \
0              Data Scientist   70000             EUR          79833   
1  Machine Learning Scientist  260000             USD         260000   
2          Big Data Engineer   85000             GBP         109024   
3        Product Data Analyst   20000             USD          20000   
4   Machine Learning Engineer  150000             USD         150000   

  employee_residence  remote_ratio company_location company_size  
0                 DE             0               DE            L  
1                 JP             0               JP            S  
2                 GB            50               GB            M  
3                 HN             0               HN            S  
4                 US            50               US            L  
Index(['Unnamed: 0', 'work_year', 'experience_level', 'employment_type',
       'job_title', 'salary', 'salary_currency', 'salary_in_usd',
       'employee_residence', 'remote_ratio', 'company_location',
       'company_size'],
      dtype='object')
```

In [49]:

```
#first lets only use the columns that apply to us.
# we want data scientist positions, salary_in_us for comparision, the location of the

ds_salaries_filtered = ds_salaries[['job_title', 'salary_in_usd', 'company_location',
ds_salaries_filtered = ds_salaries_filtered[ds_salaries_filtered['job_title'] == 'Data
ds_salaries_filtered.reset_index(drop=True, inplace=True)
print(ds_salaries_filtered.head(10))

#this is our first filtered salaries.


#lets create the box plot and first we can define the IQR values and then plug them in
Q1 = ds_salaries_filtered['salary_in_usd'].quantile(0.25)
Q3 = ds_salaries_filtered['salary_in_usd'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

#now lets see what our global salary looks like
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
plt.boxplot(ds_salaries_filtered['salary_in_usd'], vert=False)
plt.title('Boxplot of Salary (USD)')
plt.xlabel('Salary (USD)')



#here we can filter to not show anything past the lower and upper boundries
filtered_salaries = ds_salaries_filtered[
    (ds_salaries_filtered['salary_in_usd'] >= lower_bound) &
    (ds_salaries_filtered['salary_in_usd'] <= upper_bound)
]
plt.subplot(1, 2, 2)
plt.boxplot(filtered_salaries['salary_in_usd'], vert=False)
plt.title('Boxplot of Salary (USD) After Filtering Outliers')
plt.xlabel('Salary (USD)')
plt.tight_layout()
plt.show()

#it looks like it does make a difference. This was what I was lacking in my R project
#this step in this project to stop outliers from skewing the values too much. And alsc
#have a great discrepency.
```

```
        job_title  salary_in_usd company_location experience_level
0  Data Scientist          79833               DE               MI
1  Data Scientist          35735               HU               MI
2  Data Scientist          51321               FR               EN
3  Data Scientist          40481               IN               MI
4  Data Scientist          39916               FR               EN
5  Data Scientist          68428               US               SE
6  Data Scientist          45760               US               MI
7  Data Scientist          76958               GB               MI
8  Data Scientist         105000               US               MI
9  Data Scientist          38776               ES               MI
```



In [50]:

```
#there seems to me multile entries of the same country becuase they have a slight diff
#states entries
# we can group them together and avrage them out

merged_salaries = ds_salaries_filtered[['company_location', 'salary_in_usd']]
merged_salaries.rename(columns={'company_location': 'Country', 'salary_in_usd': 'Salar
print(merged_salaries.head(10))


merged_salaries = merged_salaries.groupby('Country')['Salary (USD)'].mean().reset_inde
(merged_salaries)


#Now lets create a dictinary to replace the two letter code with the alpha-2 code to t
country_code_dict = dict(zip(country_codes['Alpha-2 code'], country_codes['Country']))
merged_salaries['Country'] = merged_salaries['Country'].map(country_code_dict)
print(merged_salaries.head(10))
```

```
  Country  Salary (USD)
0      DE         79833
1      HU         35735
2      FR         51321
3      IN         40481
4      FR         39916
5      US         68428
6      US         45760
7      GB         76958
8      US        105000
9      ES         38776
       Country   Salary (USD)
0      Austria   76352.000000
1    Australia   86703.000000
2       Brazil   12901.000000
3       Canada   77787.000000
4  Switzerland  122346.000000
5        Chile   40038.000000
6      Germany   69640.142857
7      Algeria  100000.000000
8        Spain   41136.666667
9       France   50085.571429
```

```
C:\Users\Nischal\AppData\Local\Temp\ipykernel_15080\3062950383.py:6: SettingWithCopyW
arning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
er_guide/indexing.html#returning-a-view-versus-a-copy
  merged_salaries.rename(columns={'company_location': 'Country', 'salary_in_usd': 'Sa
lary (USD)'}, inplace=True)
```

In [ ]:

In [51]:

```
print(merged_data_cofl.columns)
print(merged_salaries.columns)


# now we have to group the country with multiple entries in our cofl file aswell as th
# I belive I did this before but my code seems to throw an error so i am putting this
grouped_cofl = merged_data_cofl.groupby('Country').mean()
grouped_salaries = merged_salaries.groupby('Country').mean()

#now lets merge them into a new one using the country name, so it will have the avg sa
grouped_data = pd.merge(grouped_salaries, grouped_cofl, on='Country', how='left')
grouped_data.dropna(inplace=True)

(grouped_data.head(10))
```

```
Index(['Country', 'Cost of Living Index', 'Cost of Living Plus Rent Index',
       'Rent Index', 'Groceries Index', 'Restaurant Price Index',
       'Local Purchasing Power Index'],
      dtype='object')
Index(['Country', 'Salary (USD)'], dtype='object')
```

Out[51]:

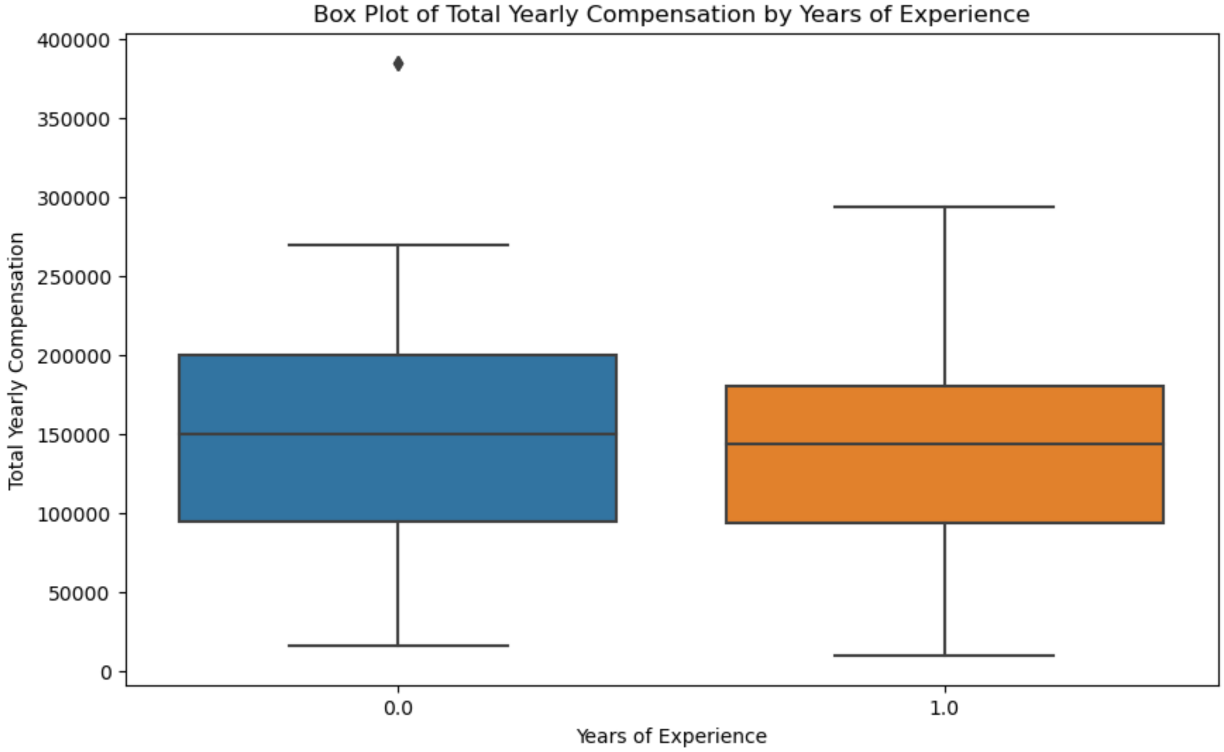
	Salary (USD)	Cost of Living Index	Cost of Living Plus Rent Index	Rent Index	Groceries Index	Restaurant Price Index	Local Purchasing Power Index
Country							
Algeria	100000.000000	29.840000	18.980000	6.670000	30.250000	20.790000	21.780000
Australia	86703.000000	77.601000	59.633000	39.267000	77.064000	75.104000	106.108000
Austria	76352.000000	72.870000	52.396000	29.194000	66.658000	70.666000	76.552000
Brazil	12901.000000	34.578571	23.137143	10.164286	29.082857	27.838571	29.360000
Canada	77787.000000	76.750160	66.927788	55.794968	77.501346	74.865417	108.086923
Chile	40038.000000	45.640000	31.420000	15.300000	40.120000	47.920000	32.080000
France	50085.571429	77.741667	55.858333	31.056667	77.201667	75.265000	82.336667
Germany	69640.142857	67.281154	49.305385	28.933846	52.963077	65.510769	100.196538
Hungary	35735.000000	41.767500	27.955000	12.300000	37.340000	34.515000	49.812500
India	26108.250000	46.258152	34.393696	20.947065	47.470000	43.115326	85.529783

In [52]:

```
print(level_salary.head(10))

level_salary = level_salary[(level_salary['title'] == 'Data Scientist') & (level_salar
grouped_stats = level_salary.groupby('yearsofexperience')['totalyearlycompensation'].a
grouped_stats.rename(columns={'<lambda>': 'IQR'}, inplace=True)
print(grouped_stats)



plt.figure(figsize=(10, 6))
sns.boxplot(x=level_salary['yearsofexperience'], y=level_salary['totalyearlycompensati
plt.xlabel('Years of Experience')
plt.ylabel('Total Yearly Compensation')
plt.title('Box Plot of Total Yearly Compensation by Years of Experience')
plt.show()
#I know this is a fyi file but I wanted to see the file in detail and include it in th
#its promising so I will keep putting in the work
```

```
                timestamp    company   level                           title  \
15710  1/27/2020 22:59:06     Google      L6               Software Engineer   
23532   7/3/2020 19:56:38  Microsoft      61               Software Engineer   
23533   7/3/2020 20:03:57     Google      L5               Software Engineer   
23534   7/3/2020 20:05:37  Microsoft      62               Software Engineer   
23535   7/3/2020 20:19:06      Blend     IC3               Software Engineer   
23537   7/3/2020 20:24:20     Amazon      L6               Software Engineer   
23538   7/3/2020 20:31:33    Chevron  PSG 20               Software Engineer   
23540   7/3/2020 22:35:58     Amazon      L7  Software Engineering Manager   
23541   7/3/2020 22:51:23    Shopify      L6               Software Engineer   
23543   7/3/2020 23:39:06      Apple    ICT3               Software Engineer   

       totalyearlycompensation               location  yearsofexperience  \
15710                   400000          Sunnyvale, CA                5.0   
23532                   136000            Redmond, WA                3.0   
23533                   337000          San Bruno, CA                6.0   
23534                   222000            Seattle, WA                4.0   
23535                   187000      San Francisco, CA                5.0   
23537                   310000            Seattle, WA               15.0   
23538                   113000            Houston, TX                3.0   
23540                   620000            Seattle, WA               19.0   
23541                    98000    Toronto, ON, Canada                9.0   
23543                   180000  Vancouver, BC, Canada                1.0   

       yearsatcompany                             tag  basesalary  ...  \
15710             5.0  Distributed Systems (Back-End)    210000.0  ...   
23532             2.0                          DevOps    124000.0  ...   
23533             6.0                      Full Stack    177000.0  ...   
23534             4.0      API Development (Back-End)    164000.0  ...   
23535             0.0                      Full Stack    165000.0  ...   
23537             3.0                         ML / AI    160000.0  ...   
23538             3.0                          DevOps    103000.0  ...   
23540             7.0                      Full Stack    160000.0  ...   
23541             4.0     Web Development (Front-End)     78000.0  ...   
23543             1.0                         ML / AI    130000.0  ...   

       Doctorate_Degree  Highschool Some_College Race_Asian  Race_White  \
15710                 1           0            0          1           0   
23532                 0           0            0          0           0   
23533                 0           0            0          1           0   
23534                 0           0            0          1           0   
23535                 0           0            0          0           1   
23537                 0           0            0          1           0   
23538                 0           0            0          0           0   
23540                 0           0            0          1           0   
23541                 0           0            0          1           0   
23543                 0           0            0          1           0   

       Race_Two_Or_More  Race_Black  Race_Hispanic         Race  \
15710                 0           0              0        Asian   
23532                 1           0              0  Two Or More   
23533                 0           0              0        Asian   
23534                 0           0              0        Asian   
23535                 0           0              0        White   
23537                 0           0              0        Asian   
23538                 0           0              1     Hispanic   
23540                 0           0              0        Asian   
23541                 0           0              0        Asian   
23543                 0           0              0        Asian   
```

```
             Education
15710              PhD
23532  Bachelor's Degree
23533  Bachelor's Degree
23534    Master's Degree
23535  Bachelor's Degree
23537  Bachelor's Degree
23538  Bachelor's Degree
23540  Bachelor's Degree
23541  Bachelor's Degree
23543  Bachelor's Degree

[10 rows x 29 columns]
                            mean    median
yearsofexperience
0.0                151000.000000  149500.0
1.0                140219.178082  144000.0
```



```
In [53]: print(grouped_data.head())
         print(grouped_data.columns)
         print(grouped_data.dtypes)
```

```
           Salary (USD)  Cost of Living Index  Cost of Living Plus Rent Index  \
Country
Algeria        100000.0             29.840000                       18.980000
Australia       86703.0             77.601000                       59.633000
Austria         76352.0             72.870000                       52.396000
Brazil          12901.0             34.578571                       23.137143
Canada          77787.0             76.750160                       66.927788

           Rent Index  Groceries Index  Restaurant Price Index  \
Country
Algeria      6.670000        30.250000               20.790000
Australia   39.267000        77.064000               75.104000
Austria     29.194000        66.658000               70.666000
Brazil      10.164286        29.082857               27.838571
Canada      55.794968        77.501346               74.865417

           Local Purchasing Power Index
Country
Algeria                       21.780000
Australia                    106.108000
Austria                       76.552000
Brazil                        29.360000
Canada                       108.086923
Index(['Salary (USD)', 'Cost of Living Index',
       'Cost of Living Plus Rent Index', 'Rent Index', 'Groceries Index',
       'Restaurant Price Index', 'Local Purchasing Power Index'],
      dtype='object')
Salary (USD)                      float64
Cost of Living Index              float64
Cost of Living Plus Rent Index    float64
Rent Index                        float64
Groceries Index                   float64
Restaurant Price Index            float64
Local Purchasing Power Index      float64
dtype: object
```

In [54]:

```
grouped_data.reset_index(inplace=True)
print(grouped_data.head())

grouped_data.nlargest(5, 'Salary (USD)')

#this is our final grouped data. I've tried to incorporate everything I've learned in
#missing a few things. I am somewhat satisfied with how it turned out so far but could

#the top countries with highest salaries.
```

```
     Country  Salary (USD)  Cost of Living Index  \
0    Algeria      100000.0             29.840000
1  Australia       86703.0             77.601000
2    Austria       76352.0             72.870000
3     Brazil       12901.0             34.578571
4     Canada       77787.0             76.750160

   Cost of Living Plus Rent Index  Rent Index  Groceries Index  \
0                       18.980000    6.670000        30.250000
1                       59.633000   39.267000        77.064000
2                       52.396000   29.194000        66.658000
3                       23.137143   10.164286        29.082857
4                       66.927788   55.794968        77.501346

   Restaurant Price Index  Local Purchasing Power Index
0               20.790000                     21.780000
1               75.104000                    106.108000
2               70.666000                     76.552000
3               27.838571                     29.360000
4               74.865417                    108.086923
```

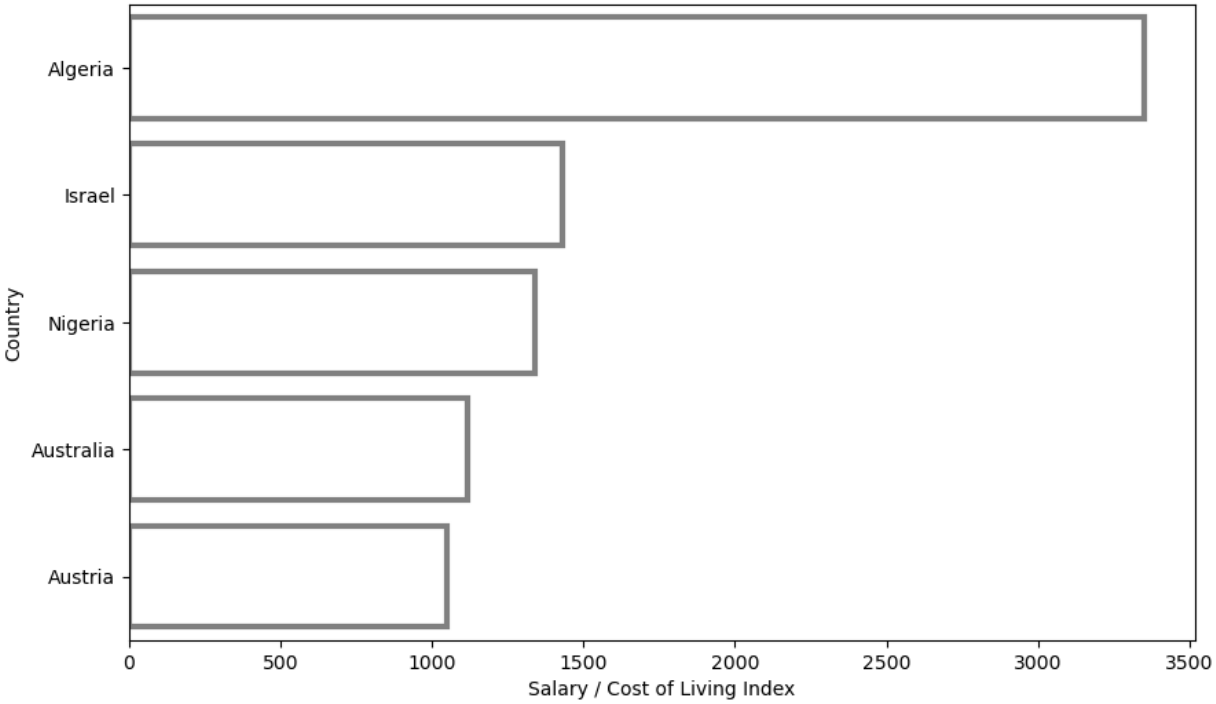
Out[54]:

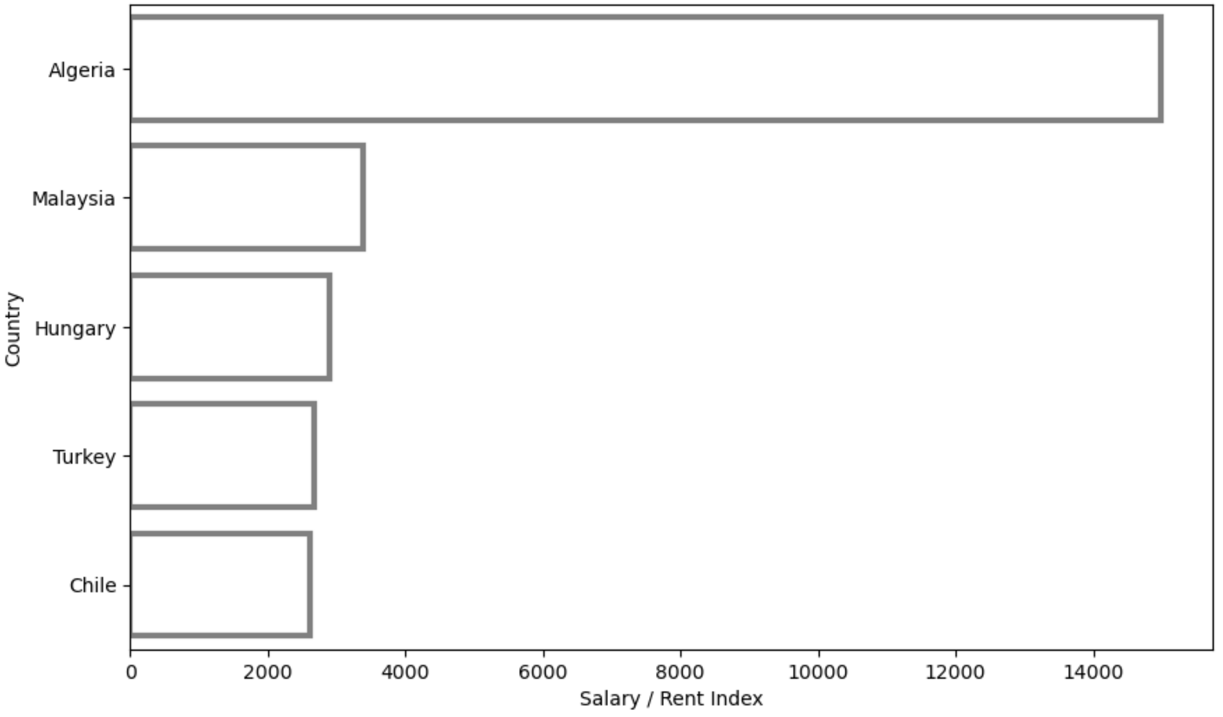
	Country	Salary (USD)	Cost of Living Index	Cost of Living Plus Rent Index	Rent Index	Groceries Index	Restaurant Price Index	Local Purchasing Power Index
18	Switzerland	122346.0	124.075714	93.802857	59.495714	126.945714	126.914286	122.392857
10	Israel	119059.0	83.118000	65.715000	45.995000	75.532000	87.640000	105.665000
0	Algeria	100000.0	29.840000	18.980000	6.670000	30.250000	20.790000	21.780000
1	Australia	86703.0	77.601000	59.633000	39.267000	77.064000	75.104000	106.108000
4	Canada	77787.0	76.750160	66.927788	55.794968	77.501346	74.865417	108.086923

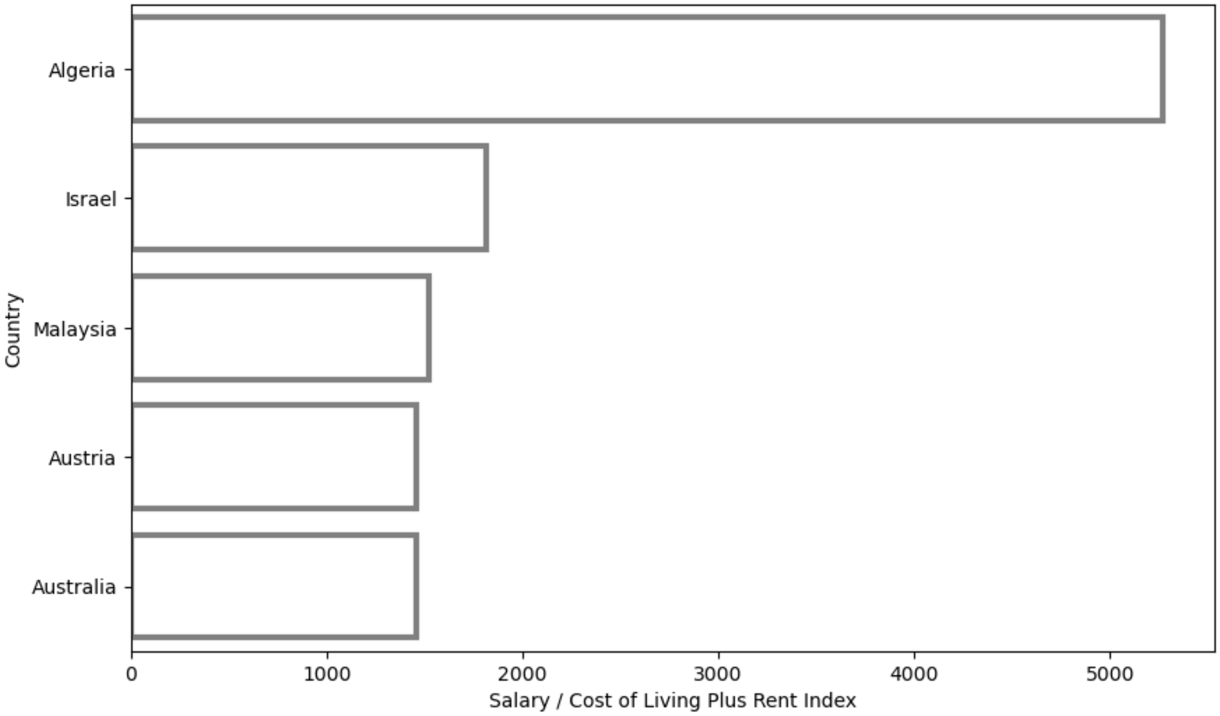
In [55]:

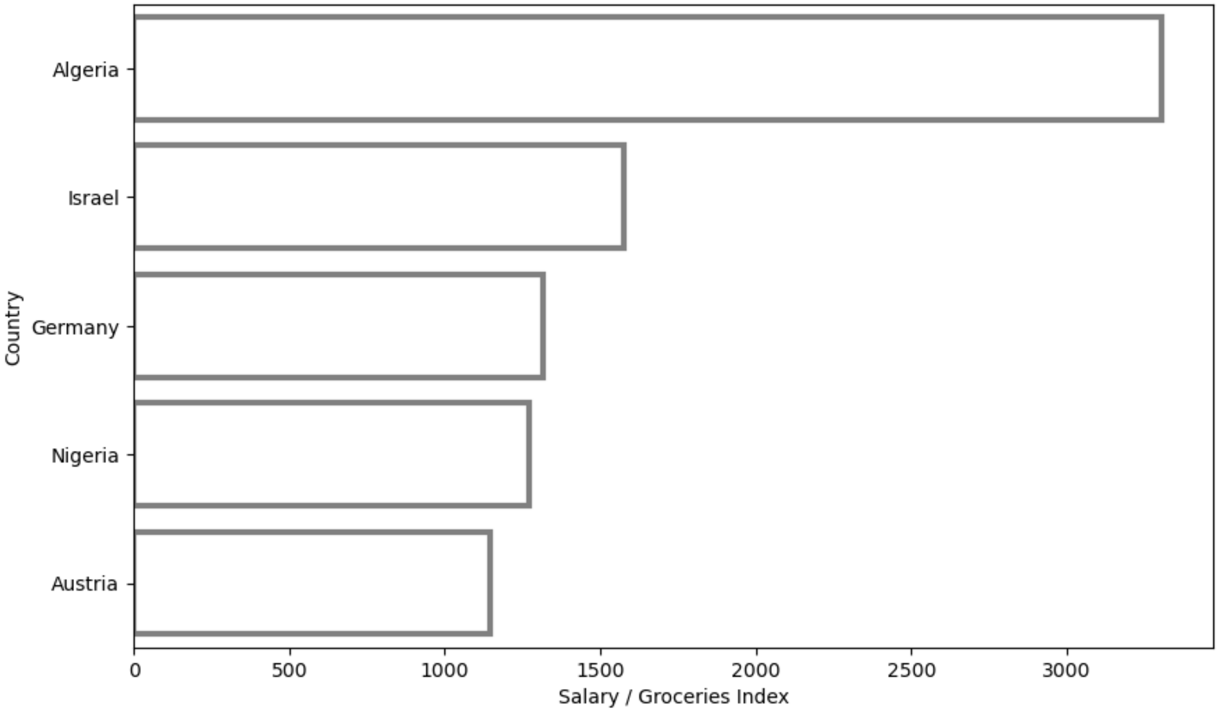
```python
#now lets define a function and then implement our parameters

def create_bar_plot(data, x_column, y_column, color_palette):
    plt.figure(figsize=(10, 6))
    sns.barplot(
        data=data,
        x=x_column,
        y=y_column,
        palette=color_palette,
        errorbar=("pi", 50),
        capsize=.4,
        errcolor=".5",
        linewidth=3,
        edgecolor=".5",
        facecolor=(0, 0, 0, 0)
    )
    plt.show()

#now lets create our indexes which we will use to build our boxplots
index_columns = ['Cost of Living Index', 'Rent Index', 'Cost of Living Plus Rent Index
color_palette = ["red", "green", "blue"]


#I dont know why the color palette is not showing. I look online and none of the solut
#it here until I figure it out.
```

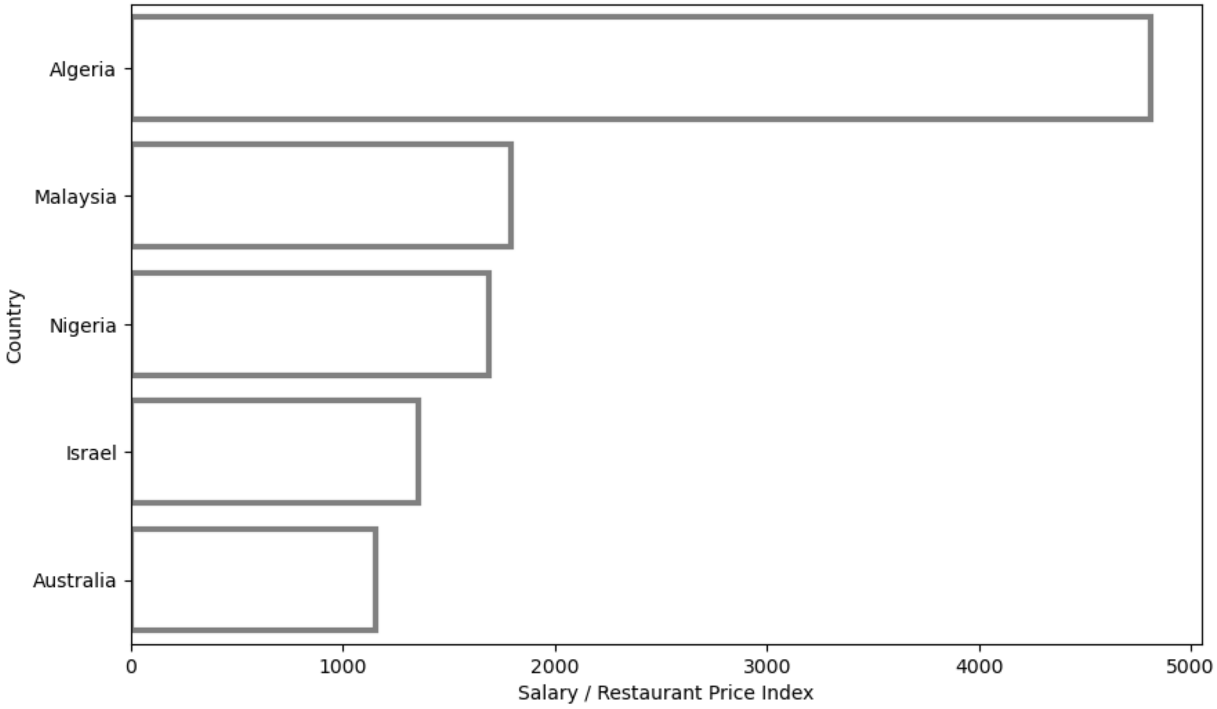
```
top_countries_by_index = {}

for column in index_columns:
    ratio_column = f'Salary / {column}'
    grouped_data[ratio_column] = grouped_data['Salary (USD)'] / grouped_data[column]
    top_countries_by_index[column] = grouped_data.nlargest(5, ratio_column)

for column, data in top_countries_by_index.items():
    create_bar_plot(data, f'Salary / {column}', 'Country', color_palette)
```

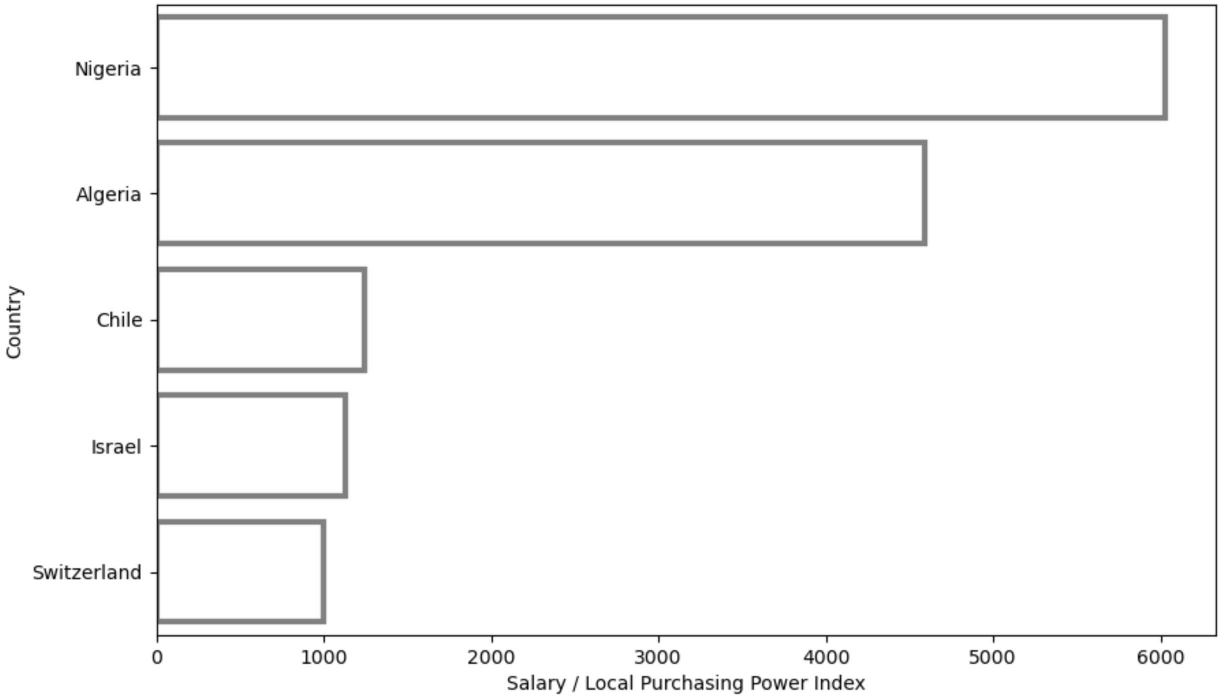





Algeria
Israel
Malaysia
Austria
Australia
Country
0
1000
2000
3000
4000
5000
Salary / Cost of Living Plus Rent Index


Algeria
Israel
Germany
Nigeria
Austria
Country
0
500
1000
1500
2000
2500
3000
Salary / Groceries Index





```
In [ ]: print("Algeria seems to be the place to be. There are remote jobs there ")
```

```
In [60]: print(ds_salaries[ds_salaries['company_location'] == 'DZ']) #algeria
         print(ds_salaries[ds_salaries['company_location'] == 'CH']) #switzerland
         print(ds_salaries[ds_salaries['company_location'] == 'NG']) #nigeria


         #here are just some examples of jobs from 3 countries.
```

```
     Unnamed: 0  work_year experience_level employment_type       job_title  \
487         487       2022               EN              PT  Data Scientist   

     salary salary_currency  salary_in_usd employee_residence  remote_ratio  \
487  100000             USD         100000                 DZ            50   

    company_location company_size  
487               DZ            M  
     Unnamed: 0  work_year experience_level employment_type  \
213         213       2021               EN              FT   
518         518       2022               MI              FT   

              job_title  salary salary_currency  salary_in_usd  \
213  Big Data Engineer  435000             INR           5882   
518     Data Scientist  115000             CHF         122346   

    employee_residence  remote_ratio company_location company_size  
213                 IN             0               CH            L  
518                 CH             0               CH            L  
     Unnamed: 0  work_year experience_level employment_type       job_title  \
38           38       2020               EN              FT    Data Analyst   
116         116       2021               MI              FT  Data Scientist   

     salary salary_currency  salary_in_usd employee_residence  remote_ratio  \
38    10000             USD          10000                 NG           100   
116   50000             USD          50000                 NG           100   

    company_location company_size  
38                NG            S  
116               NG            L  
```

Conclusion:

As you can see Data Scientist and Data related field's have many jobs available throughout the world. It is possible to work remote and make large salary from a company in Switzerland and live in Isreal. There are many combinations. I could not provide a definitive suggestion as my analysis lack many things.

Things I could have improved on:

- I could have included the united states two letter states entries but I filtred them out and did not include them in the average for United Stated merged data.

- I could have grouped the entries rather than writing code to mean as I could have seen how many entries made the average as I could also plot how many jobs were from a specific region

- I could have also doing a z score test to give an overall score to see which country's salary would go the further but I chose to show all the different countries for all the different indexes.

I apologize for the delayed submission and the lack of analysis for this project. I tried to highlight everything I have learned in the course but due to lack of time I could not dive in further. I would appreciate any feedback on

how to improve for my future projects and when I get the chance I will take your SQL course.