BANA6620 Computing for BA

Analysis of Tuberculosis Affects on Different Countries

Code Documentation.

Rithik Rathinavel Ragupathi || Sashank Addanki Venkata Naga || Irfan Saleemudeen || Kalidindi Saketh Varma

Code Snippet along with output:

```
#importing required libraries
import yaml
import pandas as pd
import numpy as np
import sqlite3
import matplotlib.pyplot as plt
import seaborn as sns
import geopandas as gpd
import matplotlib.pyplot as plt
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from langchain.prompts import ChatPromptTemplate
from langchain_openai import ChatOpenAI
from sqlalchemy import create_engine
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from langchain.prompts import ChatPromptTemplate
from langchain_openai import ChatOpenAI
from statsmodels.tsa.arima.model import ARIMA
#Read the Excel file
```

```
df = pd.read_csv("TB_Burden_Country_original.csv")

#Create an engine to connect to the SQLite database
engine = create_engine('sqlite:///New_Database.db')

#establishing connection with sql database
conn = sqlite3.connect("TB_Burden_Country_original.db")
cursor = conn.cursor()
#Write the DataFrame to a SQL table
df.to_sql('TB_Burden_Country_one',con=engine, if_exists='replace', index=False)
```

```
... 5120
```

```
#Verify the table creation by reading the data back
df_from_sql = pd.read_sql('SELECT * FROM TB_Burden_Country_one', con=engine)
df_from_sql.head()
```

...

	Country or territory name	ISO 2-character country/territory code	ISO 3-character country/territory code	ISO numeric country/territory code	Region	Year	Estimated total population number	Estimated prevalence of TB (all forms) per 100 000 population	Estimated prevalence of TB (all forms) per 100 000 population, low bound	Estimated prevalence of TB (all forms) per 100 000 population, high bound	...	Estimated incidence of TB cases who are HIV-positive per 100 000 population	Estimated incidence of TB cases who are HIV-positive per 100 000 population, low bound
0	Afghanistan	AF	AFG	4	EMR	1990	11731193	306.0	156.0	506.0	...	0.11	0.08
1	Afghanistan	AF	AFG	4	EMR	1991	12612043	343.0	178.0	562.0	...	0.13	0.11
2	Afghanistan	AF	AFG	4	EMR	1992	13811876	371.0	189.0	614.0	...	0.16	0.14
3	Afghanistan	AF	AFG	4	EMR	1993	15175325	392.0	194.0	657.0	...	0.19	0.17
4	Afghanistan	AF	AFG	4	EMR	1994	16485018	410.0	198.0	697.0	...	0.21	0.18

5 rows × 47 columns

Estimated incidence of TB cases who are HIV-positive per 100 000 population, high bound	Estimated incidence of TB cases who are HIV-positive	Estimated incidence of TB cases who are HIV-positive, low bound	Estimated incidence of TB cases who are HIV-positive, high bound	Method to derive TBHIV estimates	Case detection rate (all forms), percent	Case detection rate (all forms), percent, low bound	Case detection rate (all forms), percent, high bound
0.14	12.0	9.4	16.0	None	20.0	15.0	24.0
0.16	17.0	14.0	20.0	None	96.0	80.0	110.0
0.18	22.0	19.0	24.0	None	NaN	NaN	NaN
0.21	28.0	25.0	31.0	None	NaN	NaN	NaN
0.24	35.0	30.0	39.0	None	NaN	NaN	NaN

```
#Select the desired columns (example: selecting columns 'A', 'B', and 'C')
selected_columns = df_from_sql[["Country or territory name",
"Year",
"Estimated total population number",
"Estimated prevalence of TB (all forms)",
"Method to derive prevalence estimates",
"Estimated number of deaths from TB (all forms, excluding HIV)",
"Estimated number of deaths from TB in people who are HIV-positive",
"Method to derive mortality estimates",
"Estimated number of incident cases (all forms)",
```

"Method to derive incidence estimates",

"Estimated HIV in incident TB (percent)",

"Estimated incidence of TB cases who are HIV-positive",

"Method to derive TBHIV estimates",

"Case detection rate (all forms), percent"]]

#Write the selected columns to a new SQL table

selected_columns.to_sql('NEW_TB_Burden_Country', con=engine, if_exists='replace', index=False)

```
... 5120
```

#Verify the new table creation by reading the data back

df_from_sql = pd.read_sql('SELECT * FROM NEW_TB_Burden_Country', con=engine)

df_from_sql.head()

...

	Country or territory name	Year	Estimated total population number	Estimated prevalence of TB (all forms)	Method to derive prevalence estimates	Estimated number of deaths from TB (all forms, excluding HIV)	Estimated number of deaths from TB in people who are HIV-positive	Method to derive mortality estimates	Estimated number of incident cases (all forms)	Method to derive incidence estimates	Estimated HIV in incident TB (percent)	Estimated incidence of TB cases who are HIV-positive	Method to derive TBHIV estimates	Case detection rate (all forms), percent
0	Afghanistan	1990	11731193	36000.0	predicted	4300.0	5.0	Indirect	22000.0	None	0.06	12.0	None	20.0
1	Afghanistan	1991	12612043	43000.0	predicted	5800.0	8.0	Indirect	24000.0	None	0.07	17.0	None	96.0
2	Afghanistan	1992	13811876	51000.0	predicted	7400.0	11.0	Indirect	26000.0	None	0.08	22.0	None	NaN
3	Afghanistan	1993	15175325	59000.0	predicted	9100.0	17.0	Indirect	29000.0	None	0.10	28.0	None	NaN
4	Afghanistan	1994	16485018	68000.0	predicted	11000.0	22.0	Indirect	31000.0	None	0.11	35.0	None	NaN

#saving the progress in sql

conn.commit()

conn.close()

#displaying the shape and the list of columns of the dataframe

print(df_from_sql.shape)

print(df_from_sql.columns)

```
... (5120, 14)
    Index(['Country or territory name', 'Year',
           'Estimated total population number',
           'Estimated prevalence of TB (all forms)',
           'Method to derive prevalence estimates',
           'Estimated number of deaths from TB (all forms, excluding HIV)',
           'Estimated number of deaths from TB in people who are HIV-positive',
           'Method to derive mortality estimates',
           'Estimated number of incident cases (all forms)',
           'Method to derive incidence estimates',
           'Estimated HIV in incident TB (percent)',
           'Estimated incidence of TB cases who are HIV-positive',
           'Method to derive TBHIV estimates',
           'Case detection rate (all forms), percent'],
          dtype='object')
```

#Handle the missing values

""""

Method to derive incidence estimates has 2133 missing values out of 5120,

so we are considering to fill the not available columns as "Other"

""""

df_from_sql['Method to derive incidence estimates'] = df_from_sql['Method to derive incidence estimates'].fillna('Other')

""""

Case detection rate (all forms), percent has only 449 missing values

so, we have handled it by updating with the mean of each of its country

""""

df_from_sql['Case detection rate (all forms), percent'] = df_from_sql.groupby('Country or territory name')['Case detection rate (all forms), percent'].transform(lambda x: x.fillna(x.mean()))

#saving the dataframe after dropping the unnecessary columns

Save the DataFrame to a new SQL table or overwrite the existing table

df_from_sql.to_sql('my_table', con=engine, if_exists='replace', index=False)

df_from_sql

	Country or territory name	Year	Estimated total population number	Estimated prevalence of TB (all forms)	Method to derive prevalence estimates	Estimated number of deaths from TB (all forms, excluding HIV)	Estimated number of deaths from TB in people who are HIV-positive	Method to derive mortality estimates	Estimated number of incident cases (all forms)	Method to derive incidence estimates	Estimated HIV in incident TB (percent)	Estimated incidence of TB cases who are HIV-positive	Method to derive TBHIV estimates	Case detection rate (all forms), percent
0	Afghanistan	1990	11731193	36000.0	predicted	4300.0	5.0	Indirect	22000.0	Other	0.06	12.0	None	20.000000
1	Afghanistan	1991	12612043	43000.0	predicted	5800.0	8.0	Indirect	24000.0	Other	0.07	17.0	None	96.000000
2	Afghanistan	1992	13811876	51000.0	predicted	7400.0	11.0	Indirect	26000.0	Other	0.08	22.0	None	39.768421
3	Afghanistan	1993	15175325	59000.0	predicted	9100.0	17.0	Indirect	29000.0	Other	0.10	28.0	None	39.768421
4	Afghanistan	1994	16485018	68000.0	predicted	11000.0	22.0	Indirect	31000.0	Other	0.11	35.0	None	39.768421
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
5115	Zimbabwe	2009	12888918	58000.0	predicted	5000.0	27000.0	Indirect	87000.0	Other	76.00	66000.0	None	50.000000
5116	Zimbabwe	2010	13076978	54000.0	predicted	4700.0	26000.0	Indirect	83000.0	Other	77.00	64000.0	None	53.000000
5117	Zimbabwe	2011	13358738	56000.0	predicted	5100.0	24000.0	Indirect	80000.0	Other	75.00	60000.0	None	48.000000
5118	Zimbabwe	2012	13724317	58000.0	predicted	5600.0	22000.0	Indirect	79000.0	Other	72.00	56000.0	None	45.000000
5119	Zimbabwe	2013	14149648	58000.0	predicted	5700.0	22000.0	Indirect	78000.0	Other	72.00	56000.0	None	42.000000

5120 rows × 14 columns

```
#Exploratory data analysis
#Storing all numerical columns in the list for Summary analysis
cols=['Estimated total population number',
 'Estimated prevalence of TB (all forms)',
 'Estimated number of deaths from TB (all forms, excluding HIV)',
 'Estimated number of deaths from TB in people who are HIV-positive',
 'Estimated number of incident cases (all forms)',
 'Case detection rate (all forms), percent']
stats = {
    'mean': {},
    'median': {},
    'std_dev': {},
    'variance': {},
    'coef_variation': {}
}

# Loop through each column in the DataFrame
for column in cols:
    mean_value = df[column].mean()
    median_value = df[column].median()
    std_deviation = df[column].std()
```

```
    variance_value = df[column].var()
    coef_variation = std_deviation / mean_value if mean_value != 0 else np.nan

    # Store the computed statistics in the dictionary
    stats['mean'][column] = mean_value
    stats['median'][column] = median_value
    stats['std_dev'][column] = std_deviation
    stats['variance'][column] = variance_value
    stats['coef_variation'][column] = coef_variation

# Print the results for each column
for stat, values in stats.items():
    print(f"\n{stat.capitalize()}:")
    for column, value in values.items():
        print(f"{column}: {value}")
```

```
Mean:
Estimated total population number: 29156711.61484375
Estimated prevalence of TB (all forms): 66543.31558007812
Estimated number of deaths from TB (all forms, excluding HIV): 6863.9859140625
Estimated number of deaths from TB in people who are HIV-positive: 1798.7302363281246
Estimated number of incident cases (all forms): 42188.352388671876
Case detection rate (all forms), percent: 68.21785056733034

Median:
Estimated total population number: 5172117.5
Estimated prevalence of TB (all forms): 4300.0
Estimated number of deaths from TB (all forms, excluding HIV): 280.0
Estimated number of deaths from TB in people who are HIV-positive: 6.5
Estimated number of incident cases (all forms): 3100.0
Case detection rate (all forms), percent: 75.0

Std_dev:
Estimated total population number: 118372539.24007975
Estimated prevalence of TB (all forms): 324948.7531330065
Estimated number of deaths from TB (all forms, excluding HIV): 30554.560699794867
Estimated number of deaths from TB in people who are HIV-positive: 7915.691846898022
Estimated number of incident cases (all forms): 186570.11907674014
Case detection rate (all forms), percent: 25.4653907977844

Variance:
Estimated total population number: 1.401205804614422e+16
Estimated prevalence of TB (all forms): 105591692162.6956
Estimated number of deaths from TB (all forms, excluding HIV): 933581179.557449
Estimated number of deaths from TB in people who are HIV-positive: 62658177.41504782
Estimated number of incident cases (all forms): 34808409332.309
Case detection rate (all forms), percent: 648.4861284838825

Coef_variation:
Estimated total population number: 4.0598727594443815
Estimated prevalence of TB (all forms): 4.883266640688555
Estimated number of deaths from TB (all forms, excluding HIV): 4.451431148364774
Estimated number of deaths from TB in people who are HIV-positive: 4.400710949884783
Estimated number of incident cases (all forms): 4.422313470739773
Case detection rate (all forms), percent: 0.37329512123297276
```

```
# for categorical columns
cat=['Year','Country or territory name','Method to derive prevalence estimates','Method to derive mortality estimates','Method to derive incidence estimates']
```

```
for i in cat:
   for j in cols:
      print(df.groupby(i)[j].mean())
      print("\n")
```

```
Year
1990    2.498926e+07
1991    2.540360e+07
1992    2.580802e+07
1993    2.620290e+07
1994    2.658960e+07
1995    2.696937e+07
1996    2.734196e+07
1997    2.770744e+07
1998    2.806797e+07
1999    2.842636e+07
2000    2.878490e+07
2001    2.914438e+07
2002    2.937069e+07
2003    2.973178e+07
2004    3.009563e+07
2005    3.032041e+07
2006    3.068943e+07
2007    3.106204e+07
2008    3.143785e+07
2009    3.181617e+07
2010    3.189820e+07
2011    3.212773e+07
2012    3.250546e+07
2013    3.288308e+07
Name: Estimated total population number, dtype: float64
```

```
Year
1990    67108.966604
1991    68309.523113
1992    69438.569434
1993    70581.419245
1994    71337.056698
1995    71966.547830
1996    73059.196934
1997    73632.117642
1998    73944.616557
1999    73609.096557
2000    73367.963679
2001    72116.774575
2002    71343.125681
2003    69893.519296
2004    68160.884319
2005    65725.159579
2006    64645.052757
2007    62717.217664
2008    60657.077664
2009    58551.681822
2010    56667.659120
2011    54828.002765
2012    54061.047097
2013    52550.909355
Name: Estimated prevalence of TB (all forms), dtype: float64
```

```
Year
1990    7202.815896
1991    7299.126604
1992    7419.511887
1993    7460.372406
1994    7624.699009
1995    7744.321462
1996    7747.908302
1997    7749.897123
1998    7757.389575
1999    7734.463774
2000    7694.364528
2001    7563.652311
2002    7417.863239
2003    7278.713099
2004    7044.303099
2005    6782.481215
2006    6512.759439
2007    6261.355047
2008    5997.769953
2009    5772.304579
2010    5494.362824
2011    5281.167650
2012    5122.023272
2013    4944.384700
Name: Estimated number of deaths from TB (all forms, excluding HIV), dtype: float64
```

```
Year
1990     639.806604
1991     740.429245
1992     868.457358
1993     994.306557
1994    1138.434009
1995    1302.573538
1996    1470.785849
1997    1665.788726
1998    1843.711415
1999    2039.766651
2000    2222.693868
2001    2369.765094
2002    2465.212207
2003    2535.134460
2004    2522.301127
2005    2445.369159
2006    2353.238785
2007    2234.778692
2008    2114.602523
2009    2014.407664
2010    1932.276111
2011    1833.834793
2012    1725.667972
2013    1667.171244
Name: Estimated number of deaths from TB in people who are HIV-positive, dtype: float64
```

```
Year
1990    37858.876887
1991    37862.996226
1992    38907.753774
1993    39502.190425
1994    39729.196226
1995    40840.767547
1996    41522.280566
1997    41573.958962
1998    42827.524528
1999    42976.741038
2000    44045.883019
2001    44378.293396
2002    44162.893897
2003    44398.045681
2004    44628.045352
2005    44613.892710
2006    44110.181776
2007    43650.150187
2008    43553.291402
2009    43575.784953
2010    42560.005417
2011    41987.084562
2012    41861.337143
2013    41307.489724
Name: Estimated number of incident cases (all forms), dtype: float64
```

```
Year
1990    62.356440
1991    63.367196
1992    62.705946
1993    63.597753
1994    65.247727
1995    64.015873
1996    64.545026
1997    65.995789
1998    67.191753
1999    67.126943
2000    65.524479
2001    65.396373
2002    66.034158
2003    70.456500
2004    68.556061
2005    69.357143
2006    70.190000
2007    71.974227
2008    71.929293
2009    73.134328
2010    74.000000
2011    72.933649
2012    73.946341
2013    74.680000
Name: Case detection rate (all forms), percent, dtype: float64
```

```
Country or territory name
Afghanistan                  2.187767e+07
Albania                      3.276170e+06
Algeria                      3.251161e+07
American Samoa               5.520650e+04
Andorra                      7.041712e+04
                                 ...
Wallis and Futuna Islands    1.404733e+04
West Bank and Gaza Strip     3.238497e+06
Yemen                        1.827772e+07
Zambia                       1.071155e+07
Zimbabwe                     1.234234e+07
Name: Estimated total population number, Length: 219, dtype: float64


Country or territory name
Afghanistan                  81166.666667
Albania                        967.083333
Algeria                      40833.333333
American Samoa                   7.950000
Andorra                         19.704167
                                 ...
Wallis and Futuna Islands       15.237500
West Bank and Gaza Strip       316.250000
Yemen                        27708.333333
Zambia                       51000.000000
Zimbabwe                     50666.666667
Name: Estimated prevalence of TB (all forms), Length: 219, dtype: float64
```

```
Country or territory name
Afghanistan                  11483.333333
Albania                         30.000000
Algeria                       4195.833333
American Samoa                   0.787500
Andorra                          1.492500
                                 ...
Wallis and Futuna Islands        1.366250
West Bank and Gaza Strip        14.595833
Yemen                         3099.583333
Zambia                        3841.666667
Zimbabwe                      4133.333333
Name: Estimated number of deaths from TB (all forms, excluding HIV), Length: 219, dtype: float64


Country or territory name
Afghanistan                     44.500000
Albania                          0.000000
Algeria                         20.291667
American Samoa                   0.000000
Andorra                          0.000000
                                 ...
Wallis and Futuna Islands        0.000000
West Bank and Gaza Strip         0.507500
Yemen                           51.166667
Zambia                       13925.000000
Zimbabwe                     24029.166667
Name: Estimated number of deaths from TB in people who are HIV-positive, Length: 219, dtype: float64
```

```
Country or territory name
Afghanistan                  41333.333333
Albania                        705.000000
Algeria                      26291.666667
American Samoa                   4.987500
Andorra                         13.962500
                                 ...
Wallis and Futuna Islands        8.966667
West Bank and Gaza Strip       250.416667
Yemen                        16958.333333
Zambia                       65625.000000
Zimbabwe                     76208.333333
Name: Estimated number of incident cases (all forms), Length: 219, dtype: float64


Country or territory name
Afghanistan                  39.768421
Albania                      81.000000
Algeria                      67.416667
American Samoa               74.619048
Andorra                      87.000000
                               ...
Wallis and Futuna Islands    92.533333
West Bank and Gaza Strip     21.038889
Yemen                        62.208333
Zambia                       64.000000
Zimbabwe                     49.583333
Name: Case detection rate (all forms), percent, Length: 219, dtype: float64
```

```
Method to derive prevalence estimates
NTP               4.371731e+07
pooled surveys    1.205625e+09
predicted         1.587531e+07
survey            3.057939e+08
survey imputed    2.223814e+08
Name: Estimated total population number, dtype: float64


Method to derive prevalence estimates
NTP               6.766667e+04
pooled surveys    3.200000e+06
predicted         2.625429e+04
survey            6.378867e+05
survey imputed    6.747131e+05
Name: Estimated prevalence of TB (all forms), dtype: float64


Method to derive prevalence estimates
NTP                11602.916667
pooled surveys    320000.000000
predicted           2847.514261
survey             51607.333333
survey imputed     67359.425676
Name: Estimated number of deaths from TB (all forms, excluding HIV), dtype: float64
```

```
Method to derive prevalence estimates
NTP                  15.250000
pooled surveys    45000.000000
predicted          1298.828532
survey             7422.066667
survey imputed     9695.986486
Name: Estimated number of deaths from TB in people who are HIV-positive, dtype: float64


Method to derive prevalence estimates
NTP               5.606250e+04
pooled surveys    2.200000e+06
predicted         1.772064e+04
survey            4.445867e+05
survey imputed    4.057240e+05
Name: Estimated number of incident cases (all forms), dtype: float64


Method to derive prevalence estimates
NTP               72.802564
pooled surveys    60.000000
predicted         69.560037
survey            53.466667
survey imputed    47.923239
Name: Case detection rate (all forms), percent, dtype: float64
```

```
Method to derive mortality estimates
Indirect      1.712175e+07
VR            4.510800e+07
VR imputed    1.764996e+07
Name: Estimated total population number, dtype: float64


Method to derive mortality estimates
Indirect      64661.150029
VR            83008.867713
VR imputed    28168.427593
Name: Estimated prevalence of TB (all forms), dtype: float64


Method to derive mortality estimates
Indirect      9403.212368
VR            6279.864684
VR imputed    2010.156790
Name: Estimated number of deaths from TB (all forms, excluding HIV), dtype: float64


Method to derive mortality estimates
Indirect      3724.878514
VR             554.634849
VR imputed     213.828778
Name: Estimated number of deaths from TB in people who are HIV-positive, dtype: float64
```

```
Method to derive mortality estimates
Indirect      44576.852820
VR            49131.132768
VR imputed    18008.256204
Name: Estimated number of incident cases (all forms), dtype: float64


Method to derive mortality estimates
Indirect      57.167989
VR            75.693363
VR imputed    74.856347
Name: Case detection rate (all forms), percent, dtype: float64


Method to derive incidence estimates
Capture-recapture    3.745389e+07
Expert opinion       1.510658e+06
High income          1.491218e+07
Mortality            1.410930e+07
Neighbour            4.082916e+06
Prevalence           2.922695e+07
Survey               6.021639e+07
Trends ARI           5.328776e+08
Name: Estimated total population number, dtype: float64
```

```
Method to derive incidence estimates
Capture-recapture    1.699917e+04
Expert opinion       6.779374e+02
High income          4.311252e+03
Mortality            3.165687e+04
Neighbour            3.540370e+03
Prevalence           1.534861e+05
Survey               1.024583e+04
Trends ARI           2.012581e+06
Name: Estimated prevalence of TB (all forms), dtype: float64


Method to derive incidence estimates
Capture-recapture      1115.650000
Expert opinion           52.359792
High income             221.395217
Mortality              3104.794048
Neighbour               438.900000
Prevalence            25334.722222
Survey                  622.083333
Trends ARI           181332.458333
Name: Estimated number of deaths from TB (all forms, excluding HIV), dtype: float64
```

```
Method to derive incidence estimates
Capture-recapture       15.716667
Expert opinion           2.937500
High income              7.436862
Mortality             1314.382143
Neighbour                0.451111
Prevalence            6782.180556
Survey                 138.916667
Trends ARI           18678.541667
Name: Estimated number of deaths from TB in people who are HIV-positive, dtype: float64


Method to derive incidence estimates
Capture-recapture    1.102917e+04
Expert opinion       5.265749e+02
High income          3.182580e+03
Mortality            1.973773e+04
Neighbour            2.000370e+03
Prevalence           1.079028e+05
Survey               7.904167e+03
Trends ARI           1.092825e+06
Name: Estimated number of incident cases (all forms), dtype: float64
```

```
Method to derive incidence estimates
Capture-recapture    83.666667
Expert opinion       86.611607
High income          86.911061
Mortality            53.919600
Neighbour            24.700000
Prevalence           33.033803
Survey               80.666667
Trends ARI           53.918750
Name: Case detection rate (all forms), percent, dtype: float64
```

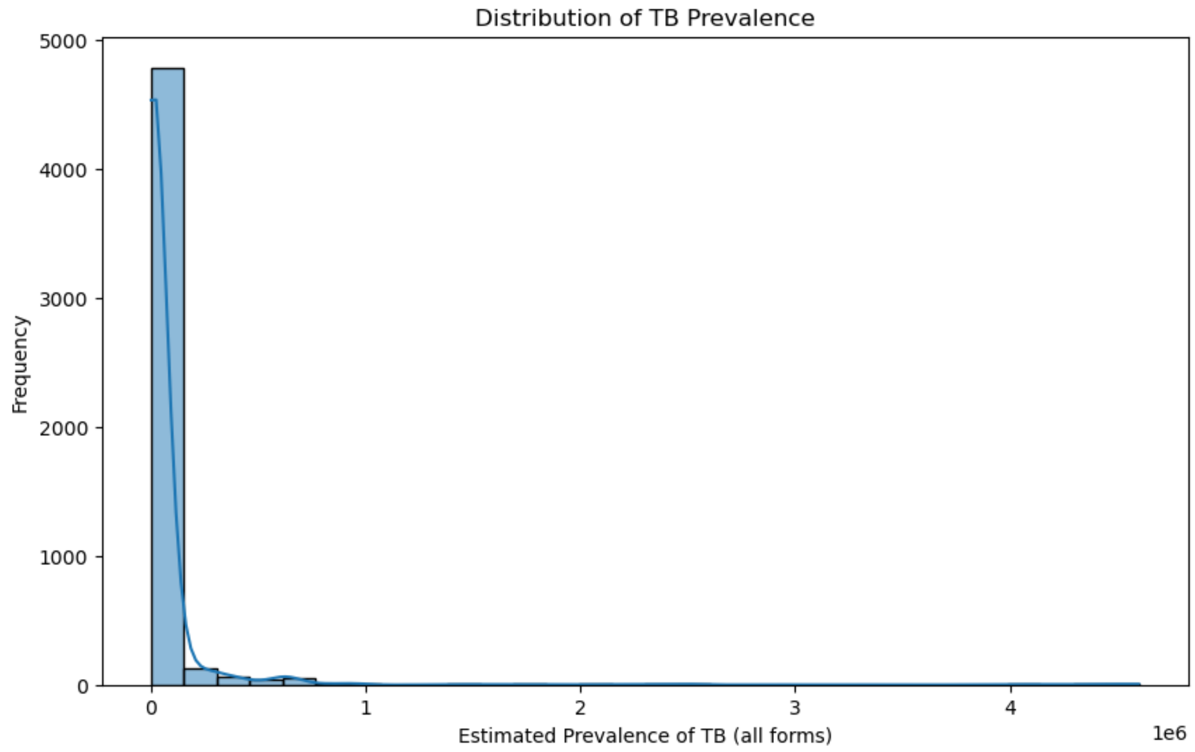
#Distribution of TB Prevalence:

plt.figure(figsize=(10, 6))

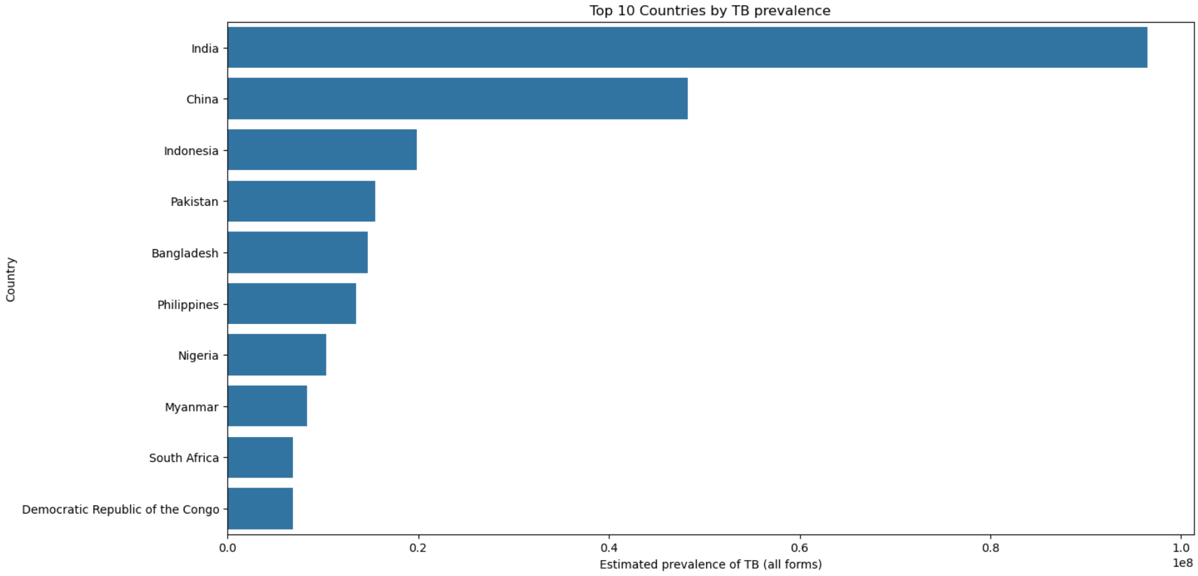
sns.histplot(df_from_sql['Estimated prevalence of TB (all forms)'], bins=30, kde=True)

plt.title('Distribution of TB Prevalence')

```
plt.xlabel('Estimated Prevalence of TB (all forms)')
plt.ylabel('Frequency')
plt.show()
```




```
#TB Prevalence:
plt.figure(figsize=(15, 8))
top_countries = df_from_sql.groupby('Country or territory name')['Estimated prevalence of TB (all forms)'].sum().nlargest(10)
sns.barplot(x=top_countries.values, y=top_countries.index)
plt.title('Top 10 Countries by TB prevalence')
plt.xlabel('Estimated prevalence of TB (all forms)')
plt.ylabel('Country')
plt.show()
```



#TB Mortality Rates by Country excluding HIV:

plt.figure(figsize=(15, 8))

top_countries = df_from_sql.groupby('Country or territory name')['Estimated number of deaths from TB (all forms, excluding HIV)'].sum().nlargest(10)

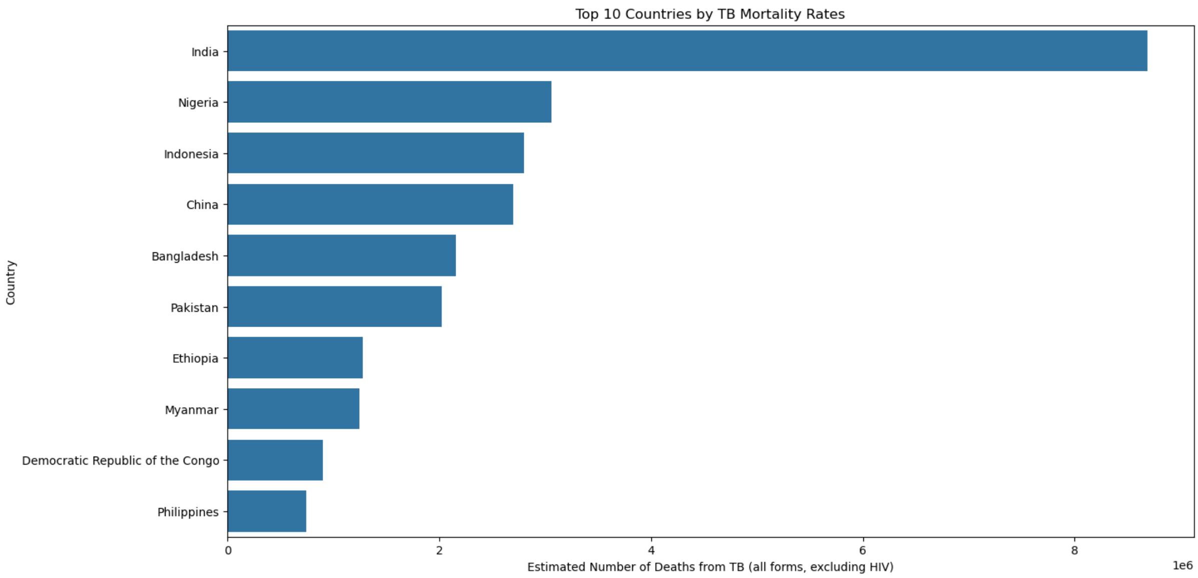
sns.barplot(x=top_countries.values, y=top_countries.index)

plt.title('Top 10 Countries by TB Mortality Rates')

plt.xlabel('Estimated Number of Deaths from TB (all forms, excluding HIV)')

plt.ylabel('Country')

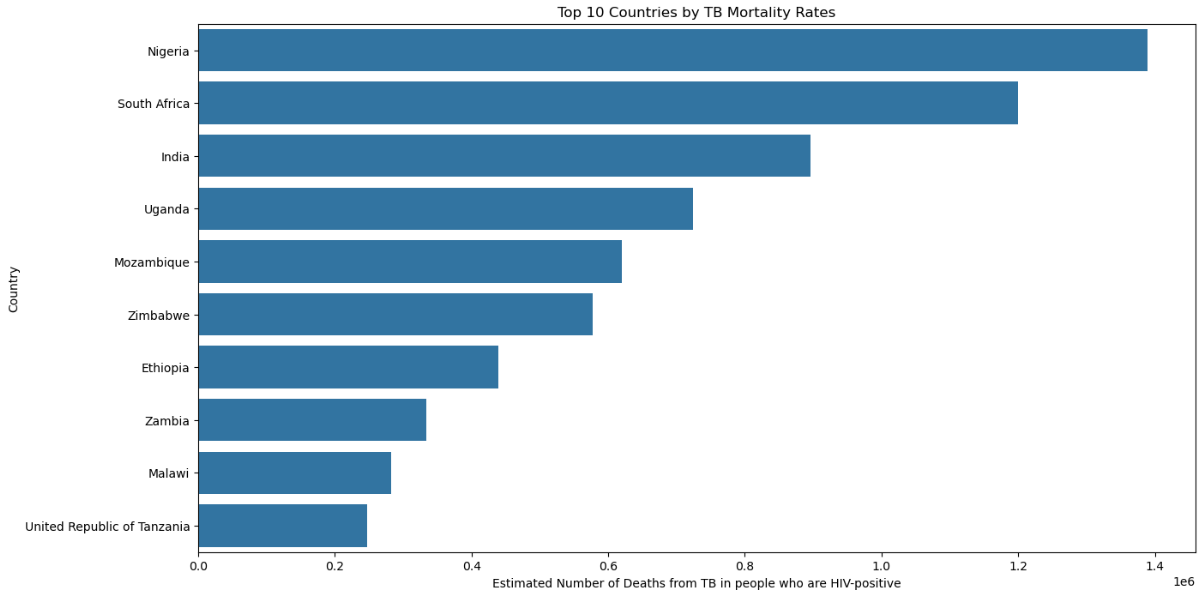
plt.show()



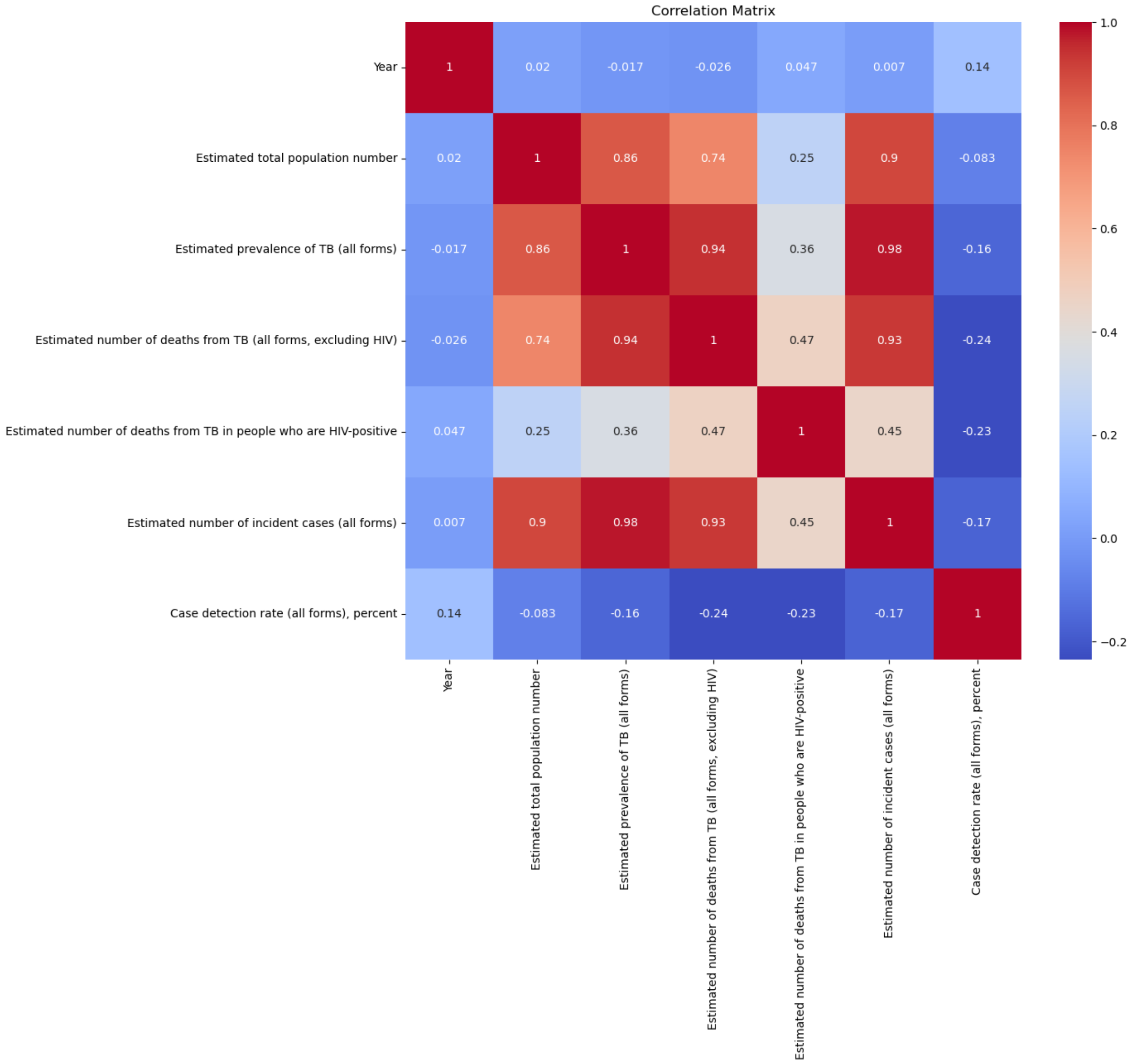
#TB Mortality Rates by Country including HIV positive:

plt.figure(figsize=(15, 8))

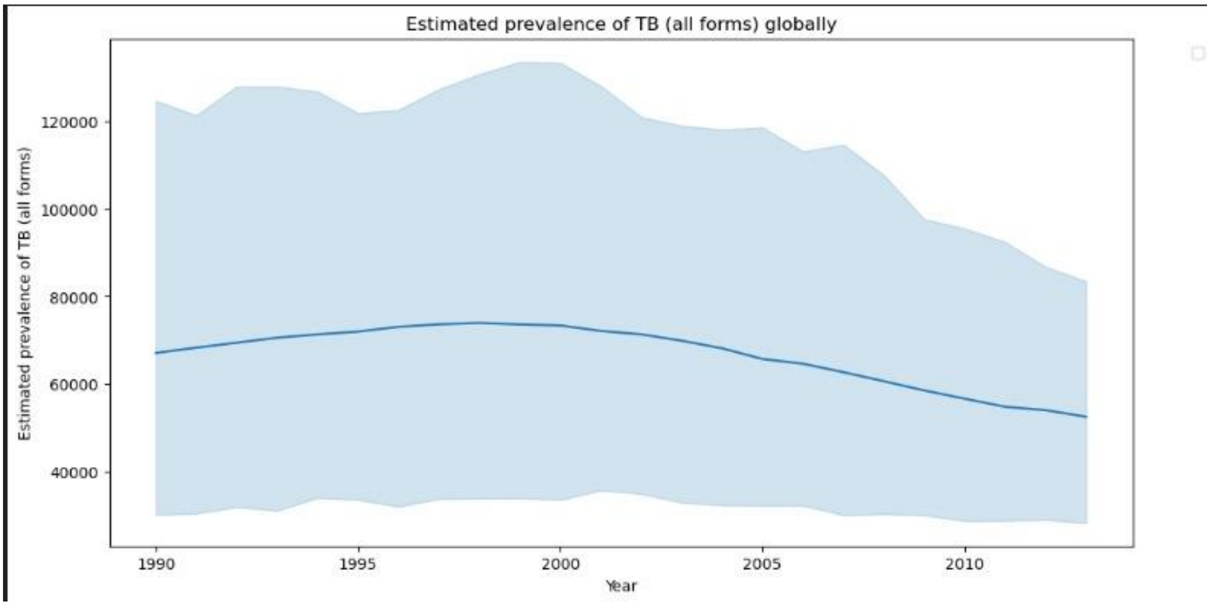
```
top_countries = df_from_sql.groupby('Country or territory name')['Estimated number of deaths from TB in people who are HIV-positive'].sum().nlargest(10)

sns.barplot(x=top_countries.values, y=top_countries.index)

plt.title('Top 10 Countries by TB Mortality Rates')

plt.xlabel('Estimated Number of Deaths from TB in people who are HIV-positive')

plt.ylabel('Country')

plt.show()
```




```
#Correlation Analysis:

# Select only numeric columns for correlation

numeric_df = df_from_sql.select_dtypes(include=['float64', 'int64'])


# Correlation Analysis

plt.figure(figsize=(12, 10))

correlation_matrix = numeric_df.corr()

sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')

plt.title('Correlation Matrix')

plt.show()
```

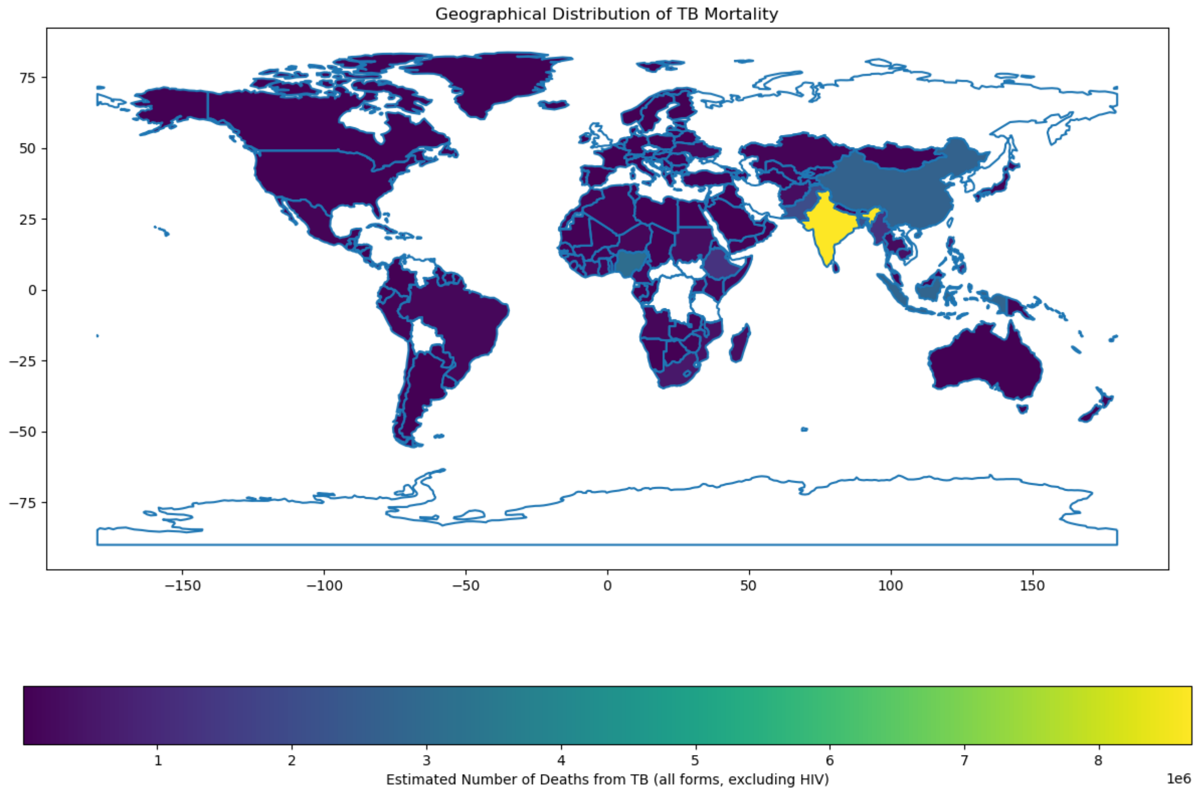



```
# Global Trend Analysis Over Time:

plt.figure(figsize=(12, 6))

sns.lineplot(data=df_from_sql, x='Year', y='Estimated prevalence of TB (all forms)')#,
hue='Country')

plt.title('Estimated Prevalence of TB (all forms) globally’)

plt.xlabel('Year')

plt.ylabel('Estimated prevalence of TB (all forms)')

plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

plt.show()
```



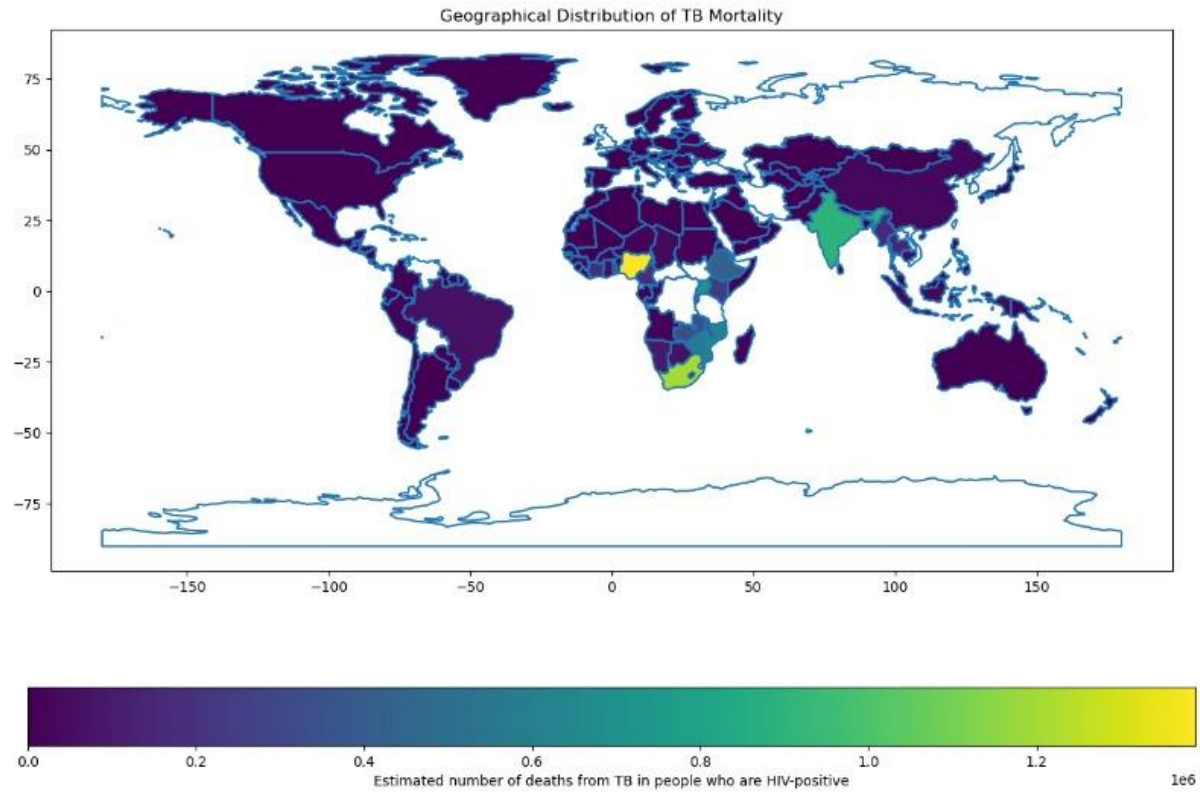

```
# Path to the downloaded shapefile
shapefile_path = "110m_cultural/ne_110m_admin_0_countries.shp"

# Load the shapefile
world = gpd.read_file(shapefile_path)

#Assuming df_geo is already prepared with the required columns
df_geo = df_from_sql.groupby('Country or territory name', as_index=False).sum()

# Merge GeoDataFrame with the data
world = world.merge(df_geo, how='left', left_on='NAME', right_on='Country or territory name')

# Plot the map
fig, ax = plt.subplots(1, 1, figsize=(15, 10))
world.boundary.plot(ax=ax)
world.plot(
    column='Estimated number of deaths from TB (all forms, excluding HIV)',
    ax=ax,
    legend=True,
    legend_kwds={
```

```
        'label': "Estimated Number of Deaths from TB (all forms, excluding HIV)",
        'orientation': "horizontal"
    }
)
plt.title('Geographical Distribution of TB Mortality')
plt.show()
```




```
#TB mortality
#Assuming df_geo is already prepared with the required columns
df_geo = df_from_sql.groupby('Country or territory name', as_index=False).sum()
# Merge GeoDataFrame with the data
world = world.merge(df_geo, how='left', left_on='NAME', right_on='Country or territory name')
# Plot the map
fig, ax = plt.subplots(1, 1, figsize=(15, 10))
world.boundary.plot(ax=ax)
world.plot(
    column='Estimated number of deaths from TB in people who are HIV-positive',
```

```
    ax=ax,
    legend=True,
    legend_kwds={
        'label':'Estimated number of deaths from TB in people who are HIV-positive',
        'orientation': "horizontal"
    }
)
plt.title('Geographical Distribution of TB Mortality')
plt.show()
```



#Summary Statistics:

df_from_sql.describe()

	Year	Estimated total population number	Estimated prevalence of TB (all forms)	Estimated number of deaths from TB (all forms, excluding HIV)	Estimated number of deaths from TB in people who are HIV-positive	Estimated number of incident cases (all forms)	Estimated HIV in incident TB (percent)	Estimated incidence of TB cases who are HIV-positive	Case detection rate (all forms), percent
count	5120.000000	5.120000e+03	5.120000e+03	5120.000000	5120.000000	5.120000e+03	3645.000000	3645.000000	5120.000000
mean	2001.549023	2.915671e+07	6.654332e+04	6863.985914	1798.730236	4.218835e+04	11.179119	6095.426979	68.897965
std	6.933272	1.183725e+08	3.249488e+05	30554.560700	7915.691847	1.865701e+05	17.133550	22807.804792	25.168609
min	1990.000000	1.129000e+03	0.000000e+00	0.000000	0.000000	0.000000e+00	0.000000	0.000000	0.000000
25%	1996.000000	5.601190e+05	3.800000e+02	18.000000	0.000000	2.600000e+02	0.640000	18.000000	53.000000
50%	2002.000000	5.172118e+06	4.300000e+03	280.000000	6.500000	3.100000e+03	3.400000	170.000000	77.000000
75%	2008.000000	1.752404e+07	2.700000e+04	2200.000000	270.000000	1.800000e+04	13.000000	1700.000000	87.000000
max	2013.000000	1.385567e+09	4.600000e+06	420000.000000	96000.000000	2.400000e+06	83.000000	320000.000000	320.000000

#Country-wise Analysis for TB incident cases:

highest_tb_countries = df_from_sql.groupby('Country or territory name')['Estimated number of incident cases (all forms)'].sum().nlargest(5)

lowest_tb_countries = df_from_sql.groupby('Country or territory name')['Estimated number of incident cases (all forms)'].sum().nsmallest(5)

print("Countries with highest TB incident cases:")

print(highest_tb_countries)

print("\nCountries with lowest TB incident cases:")

print(lowest_tb_countries)

```
...  Countries with highest TB incident cases:
     Country or territory name
     India        52400000.0
     China        32480000.0
     Indonesia    10200000.0
     Nigeria      10010000.0
     Pakistan      9740000.0
     Name: Estimated number of incident cases (all forms), dtype: float64

     Countries with lowest TB incident cases:
     Country or territory name
     Bonaire, Saint Eustatius and Saba     1.10
     Tokelau                               6.27
     Sint Maarten (Dutch part)             9.20
     Curaçao                              10.30
     Montserrat                           14.67
     Name: Estimated number of incident cases (all forms), dtype: float64
```

#Country wise Total population:

#Country-wise Analysis for TB incident cases:

highest_tb_countries = df_from_sql.groupby('Country or territory name')['Estimated total population number'].sum().nlargest(5)

lowest_tb_countries = df_from_sql.groupby('Country or territory name')['Estimated total population number'].sum().nsmallest(5)

print("Countries with highest population between 1990-2013: ")

print(highest_tb_countries)

print("\nCountries with lowest population between 1990-2013: ")

print(lowest_tb_countries)

```
...  Countries with highest population between 1990-2013:
     Country or territory name
     China                       30907331652
     India                       25563539547
     United States of America     6906473630
     Indonesia                    5136128634
     Brazil                       4244854830
     Name: Estimated total population number, dtype: int64

     Countries with lowest population between 1990-2013:
     Country or territory name
     Tokelau                             33083
     Niue                                44311
     Bonaire, Saint Eustatius and Saba   73634
     Montserrat                         161883
     Sint Maarten (Dutch part)          175558
     Name: Estimated total population number, dtype: int64
```

```
#Multiple Linear Regression

# One-Hot Encoding for the 'Country or territory name' column

data_encoded = pd.get_dummies(df_from_sql, columns=['Country or territory name'],
drop_first=True)


# Select independent variables (including country)

X = data_encoded[['Estimated total population number',
                  'Estimated prevalence of TB (all forms)',
                  'Estimated number of incident cases (all forms)',
                  'Case detection rate (all forms), percent'] +
                 [col for col in data_encoded.columns if 'Country or territory name' in col]]


# Select dependent variable (TB deaths, excluding HIV)

y = df_from_sql['Estimated number of deaths from TB (all forms, excluding HIV)']


# Split the data into training and testing sets (80/20 split)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


# Initialize the regression model

model = LinearRegression()


# Fit the model on the training data

model.fit(X_train, y_train)


# Make predictions on the test data
```

y_pred = model.predict(X_test)

Evaluate the model

mse = mean_squared_error(y_test, y_pred)

r2 = r2_score(y_test, y_pred)

Display the evaluation metrics and model coefficients

print("Mean Squared Error:", mse)

print("R-squared:", r2)

print("Regression Coefficients:", model.coef_)

print("Intercept:", model.intercept_)

```
...   Mean Squared Error: 8287670.066046737
      R-squared: 0.9807152325987263
      Regression Coefficients: [-1.78941698e-05  7.32458590e-02  7.42339953e-02 -5.31345696e+01
       -7.39475656e+02 -1.59863711e+03 -1.11644888e+03 -4.00354303e+02
       -2.95138284e+03 -4.74685960e+03 -3.99265823e+02 -1.86617310e+03
       -8.64520353e+02 -4.00130652e+02 -2.37344223e+02 -3.54424774e+02
       -1.04537087e+04 -4.87468878e+02 -4.07195697e+02  2.09086106e+04
       -1.03149187e+03 -1.21677460e+03 -3.36502689e+02 -3.59800279e+02
       -2.23559665e+03 -4.00098252e+02 -2.00509036e+03 -3.91324413e+03
       -4.00465381e+02 -1.77624339e+03 -2.57711468e+03 -7.82021857e+03
       -3.99752055e+02 -4.23134221e+02 -1.13693877e+03 -2.62718577e+03
       -2.13108101e+03 -2.73269444e+03 -2.57919433e+03 -2.24438241e+03
       -5.81009049e+01 -4.00384464e+02 -2.94792423e+03 -2.49570693e+03
       -3.71445820e+02 -1.12336358e+05 -1.15424430e+03 -4.51990420e+02
       -2.68227292e+03 -2.14329562e+03 -2.54664102e+03 -4.00474018e+02
       -1.34791820e+03 -4.30553137e+02 -1.53794022e+03 -3.98274637e+02
       -4.56231530e+02 -3.98561471e+02 -1.92558817e+03  5.30143289e+03
        2.80205434e+03 -3.82854292e+02 -1.27529517e+03 -3.98334132e+02
       -2.53002359e+03 -3.31007237e+03 -3.10268234e+03 -5.41927268e+02
       -4.20484785e+02 -3.36785735e+02 -3.94712684e+02  1.85216013e+04
       -3.00315749e+03 -3.12893468e+02 -3.41068396e+02 -4.02790685e+02
       -2.62196010e+03 -1.10425383e+03 -4.42200652e+03  7.41109148e+00
       -1.99395159e+03 -1.89477133e+02 -4.08645199e+02 -2.11234277e+03
       -4.03619142e+02 -3.82872621e+03 -2.12354236e+03 -2.86963362e+03
       -2.04064104e+03 -1.11199502e+03 -1.89431525e+03 -6.31612657e+02
      ...
       -4.22855338e+02 -2.32658674e+03 -1.06515821e+03 -1.51733256e+03
       -6.54844824e+03 -2.62781060e+02 -3.80567192e+03 -1.57311708e+03
       -6.23409239e+03 -7.12245251e+03]
      Intercept: 5023.467134163877
```

import matplotlib.pyplot as plt

import seaborn as sns

Create a scatter plot for predicted vs. actual values

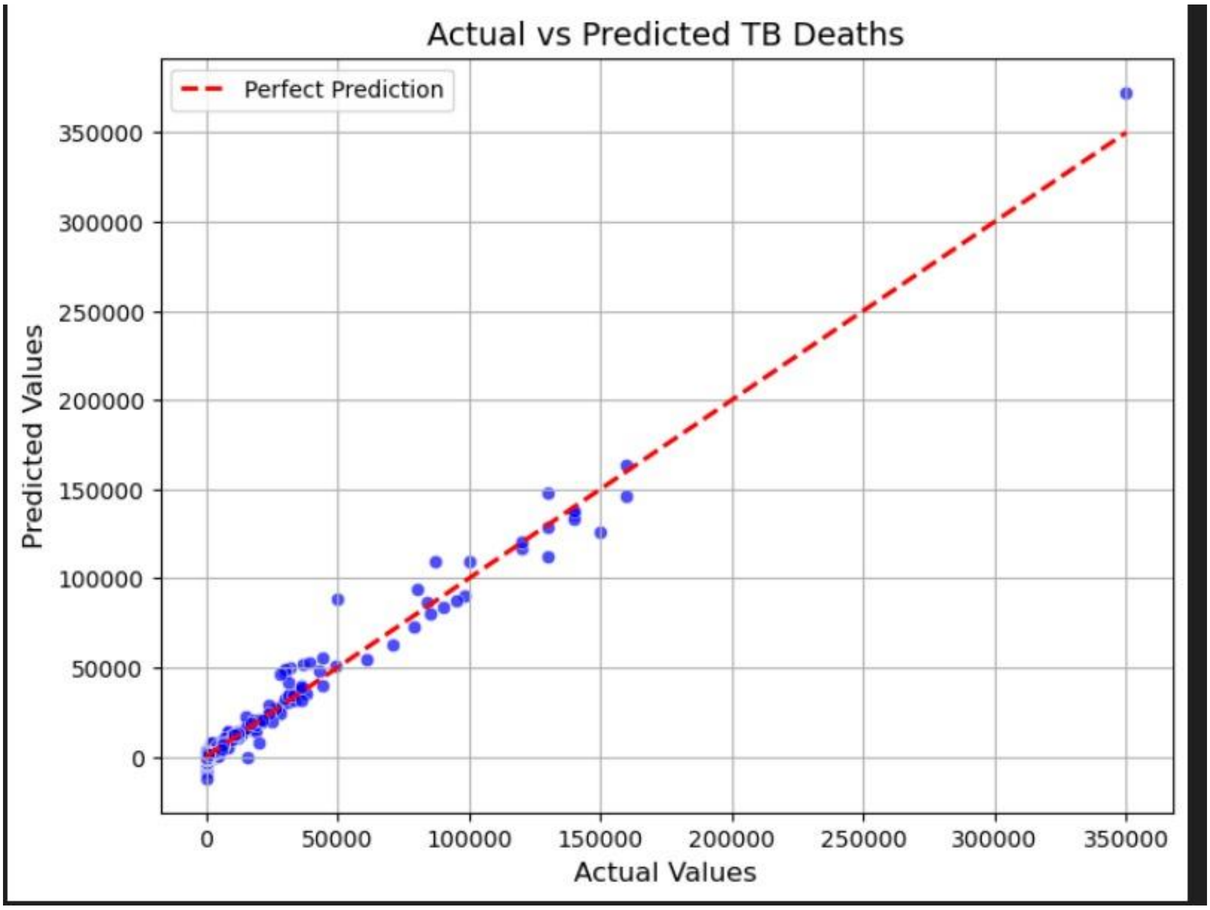
plt.figure(figsize=(8, 6))

sns.scatterplot(x=y_test, y=y_pred, color='blue', alpha=0.7)

Plot a diagonal line for perfect predictions

plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--', lw=2, label='Perfect Prediction')

```
# Labeling the axes and title
plt.xlabel('Actual Values', fontsize=12)
plt.ylabel('Predicted Values', fontsize=12)
plt.title('Actual vs Predicted TB Deaths', fontsize=14)
plt.legend()
plt.grid(True)

# Show the plot
plt.show()
```




```
#Document Summarization using LLM's and generating an overall global report
# Load your OpenAI key
OpenAI_Key = yaml.safe_load(open("credentials1.yml"))["openai"]

# Define the prompt template for a global report
```

```
global_prompt_template = """
write a business report based on the following TB dataset summary at a global level:
- Total countries: {total_countries}
- Total estimated deaths: {total_deaths}
- Global trends: {global_trends}
Use the following Markdown format:
# Global TB Mortality Report
## Summary
Provide an overview based on the data provided, including a summary of the total estimated deaths and trends.
## Important Financials
Discuss any relevant global financial impacts.
## Key Business Risks
Highlight key global risks related to TB mortality.
## Conclusions
Conclude with overarching global actions and implications.
"""


# Initialize the ChatOpenAI model
model = ChatOpenAI(model="gpt-4o-mini", temperature=0.7, api_key=OpenAI_Key)


def generate_global_report(data):
    # Calculate global statistics
    total_countries = len(data)
    total_deaths = data['Estimated number of deaths from TB (all forms, excluding HIV)'].sum()


    # Extract trends and other global insights from the data
    # (For example, a simple analysis of the top 5 countries with the highest deaths)
    global_trends = data[['Country or territory name', 'Estimated number of deaths from TB (all forms, excluding HIV)']] \
```

```
                    .sort_values(by='Estimated number of deaths from TB (all forms, excluding
HIV)', ascending=False) \
                    .head(5).to_dict(orient='records')

    # Format the prompt with global data
    prompt = global_prompt_template.format(
        total_countries=total_countries,
        total_deaths=total_deaths,
        global_trends=global_trends
    )

    # Generate the global report by calling the OpenAI model directly
    response = model.invoke(prompt)

    # Access the response content (text) from the AIMessage object
    report_text = response.content  # Extracting the 'content' field

    return report_text

# Generate the global report
global_report = generate_global_report(df_from_sql)

# Print the global report
print(global_report)
```

OUTPUT CELL:

```
··· # Global TB Mortality Report

    ## Summary
    This report provides a comprehensive overview of tuberculosis (TB) mortality on a global scale, derived from a dataset encompassing 5,120 countries. The total est

    ## Important Financials
    The global financial impact of TB mortality is profound, affecting healthcare systems, economic productivity, and national budgets. The burden of TB-related death

    ## Key Business Risks
    The persistence of high TB mortality rates presents several key risks for businesses and economies globally:

    1. **Healthcare System Strain**: Increased TB cases can overwhelm healthcare facilities, leading to a decline in the quality of care for all patients.
    2. **Workforce Impact**: High mortality rates can lead to a loss of skilled labor, negatively impacting business operations and economic growth.
    3. **Market Instability**: Regions severely affected by TB may experience economic instability, which can deter investment and affect market confidence.
    4. **Regulatory Changes**: Governments may implement stricter regulations and guidelines to control TB outbreaks, potentially impacting business operations, parti

    ## Conclusions
    The global TB mortality landscape presents a critical public health challenge that requires coordinated international efforts. Addressing the high mortality rates
```

Global TB Mortality Report

Summary

This report provides a comprehensive overview of tuberculosis (TB) mortality on a global scale, derived from a dataset encompassing 5,120 countries. The total estimated deaths attributed to TB stand at approximately 35,143,608, underscoring the significant impact of this disease worldwide. The majority of these deaths are concentrated in specific regions, with India featuring prominently in the data. Multiple estimates from India suggest that the number of deaths from TB (excluding HIV-related cases) is consistently reported around 400,000 to 420,000 annually. This trend highlights the ongoing struggle against TB, particularly in high-burden countries.

Important Financials

The global financial impact of TB mortality is profound, affecting healthcare systems, economic productivity, and national budgets. The burden of TB-related deaths not only strains public health resources but also leads to substantial economic losses due to decreased workforce productivity and increased healthcare expenditures. Countries heavily impacted by TB, such as India, face significant financial challenges in addressing the disease, including funding for healthcare infrastructure, treatments, and prevention programs. The economic implications extend beyond healthcare costs, as communities bear the brunt of lost income and productivity due to illness and mortality.

Key Business Risks

The persistence of high TB mortality rates presents several key risks for businesses and economies globally:

1. **Healthcare System Strain**: Increased TB cases can overwhelm healthcare facilities, leading to a decline in the quality of care for all patients.

2. **Workforce Impact**: High mortality rates can lead to a loss of skilled labor, negatively impacting business operations and economic growth.

3. **Market Instability**: Regions severely affected by TB may experience economic instability, which can deter investment and affect market confidence.

4. **Regulatory Changes**: Governments may implement stricter regulations and guidelines to control TB outbreaks, potentially impacting business operations, particularly in the healthcare and pharmaceutical sectors.

Conclusions

The global TB mortality landscape presents a critical public health challenge that requires coordinated international efforts. Addressing the high mortality rates associated with TB necessitates a multifaceted approach, including enhancing healthcare accessibility, investing in research for new treatments and vaccines, and implementing robust public health campaigns. Governments, NGOs, and private sectors must collaborate to mitigate the financial burdens and risks associated with TB. Failure to take decisive action could perpetuate the cycle of morbidity and mortality, ultimately hindering global health and economic progress. The urgency for targeted interventions and sustainable solutions is paramount to reversing the current trends and improving health outcomes worldwide.

Time series forecasting using ARIMA for India and Nigeria

#FINAL Code for ARIMA for INDIA because it is highest in no of deaths excluding HIV

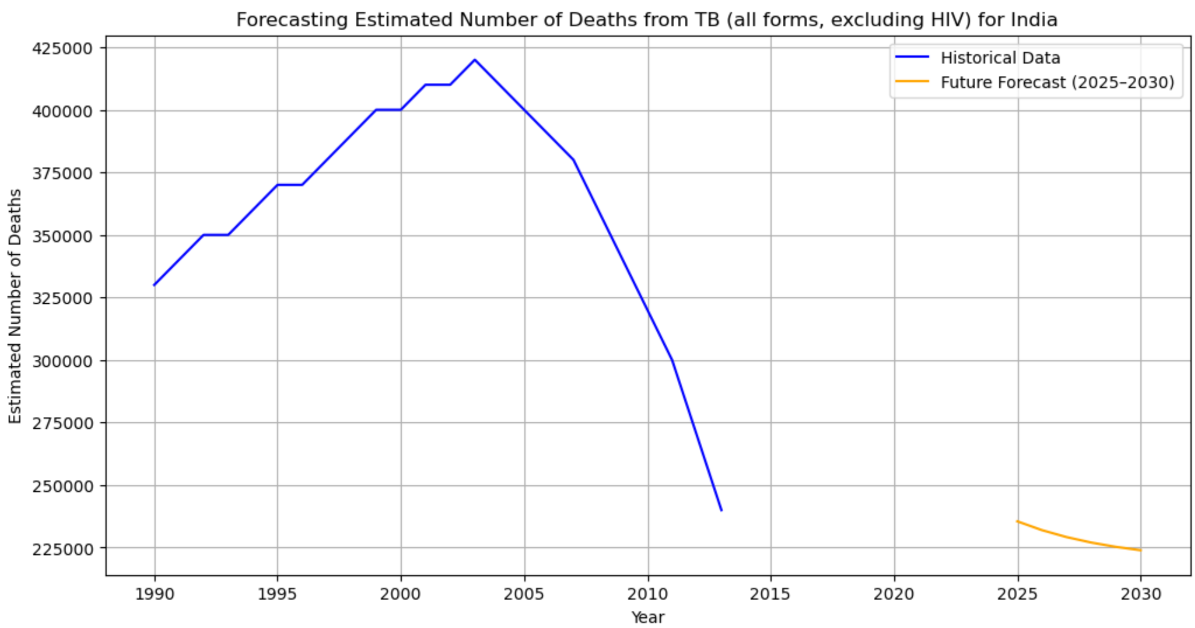
```
# Inspect the column names

print(df_from_sql.columns)
```

```
...   Index(['Country or territory name', 'Year',
             'Estimated total population number',
             'Estimated prevalence of TB (all forms)',
             'Method to derive prevalence estimates',
             'Estimated number of deaths from TB (all forms, excluding HIV)',
             'Estimated number of deaths from TB in people who are HIV-positive',
             'Method to derive mortality estimates',
             'Estimated number of incident cases (all forms)',
             'Method to derive incidence estimates',
             'Estimated HIV in incident TB (percent)',
             'Estimated incidence of TB cases who are HIV-positive',
             'Method to derive TBHIV estimates',
             'Case detection rate (all forms), percent'],
            dtype='object')
```

```
# Filter for India and select relevant columns

df_from_sql = df_from_sql[df_from_sql["Country or territory name"] == "India"]  # Use the correct column name for 'Country'

df_from_sql = df_from_sql[["Year", "Estimated number of deaths from TB (all forms, excluding HIV)"]]

df_from_sql.rename(columns={"Year": "Year", "Estimated number of deaths from TB (all forms, excluding HIV)": "deaths"}, inplace=True)


# Convert 'year' to integers and set it as the index
```

```
df_from_sql["Year"] = df_from_sql["Year"].astype(int)
df_from_sql.set_index("Year", inplace=True)

# Filter data for years 1990–2013
df_from_sql = df_from_sql.loc[1990:2013]

# Train the ARIMA model
model = ARIMA(df_from_sql["deaths"], order=(1, 1, 1))  # Adjust the order (p, d, q) as needed
model_fit = model.fit()

# Forecast for 2025–2030
forecast_years = range(2025, 2031)
forecast = model_fit.forecast(steps=len(forecast_years))

# Create a DataFrame for forecasted values
forecast_df = pd.DataFrame({
    "Year": forecast_years,
    "deaths": forecast
}).set_index("Year")

# Combine historical and forecasted data
combined_data = pd.concat([df_from_sql, forecast_df])

# Plot the results
plt.figure(figsize=(12, 6))
plt.plot(df_from_sql.index, df_from_sql["deaths"], label="Historical Data", color="blue")
plt.plot(forecast_df.index, forecast_df["deaths"], label="Future Forecast (2025–2030)", color="orange")
plt.title("Forecasting Estimated Number of Deaths from TB (all forms, excluding HIV) for India")
```

```python
plt.xlabel("Year")
plt.ylabel("Estimated Number of Deaths")
plt.legend()
plt.grid()
plt.show()

# Display forecasted values
print("\nForecasted values (2025–2030):")
print(forecast_df)
```



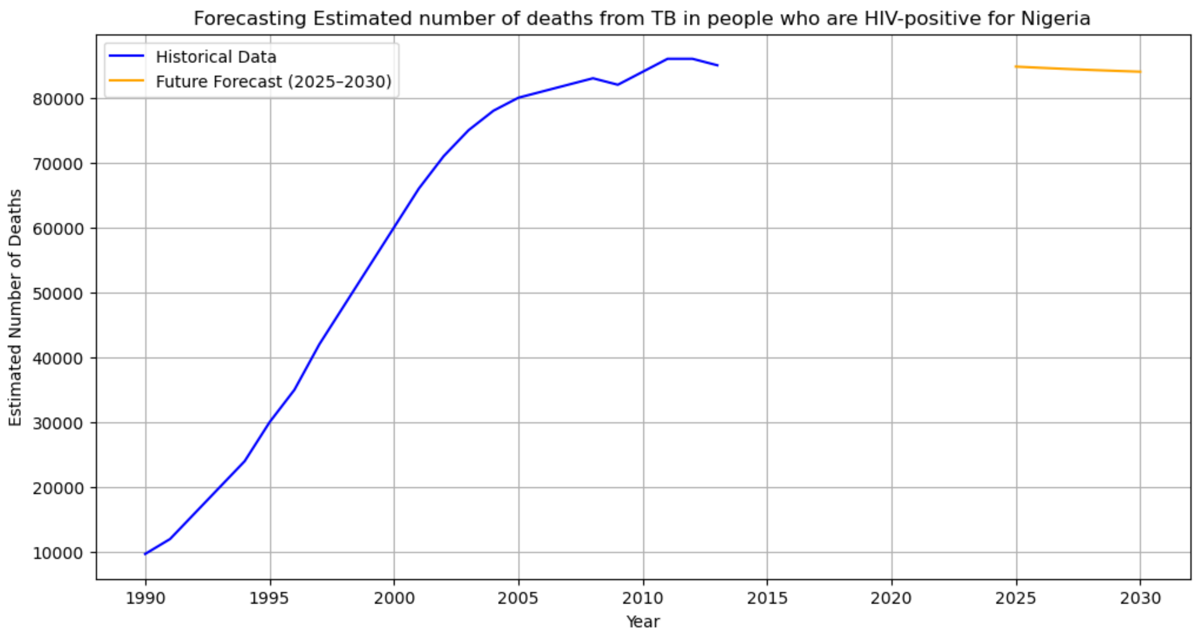

```
...
Forecasted values (2025–2030):
             deaths
Year
2025  235514.459527
2026  231985.407531
2027  229208.884151
2028  227024.421558
2029  225305.769788
2030  223953.600189
```

```python
#FINAL Code for ARIMA for NIGERIA because it is highest in no of deaths excluding HIV
# Inspect the column names
df_from_sql = pd.read_sql('NEW_TB_Burden_Country', con=engine)
print(df_from_sql.columns)

# Filter for Nigeria and select relevant columns
```

```
df_from_sql = df_from_sql[df_from_sql["Country or territory name"] == "Nigeria"]  # Use the correct column name for 'Country'

df_from_sql = df_from_sql[["Year",'Estimated number of deaths from TB in people who are HIV-positive']]

df_from_sql.rename(columns={"Year": "Year",'Estimated number of deaths from TB in people who are HIV-positive': "deaths"}, inplace=True)


# Convert 'year' to integers and set it as the index

df_from_sql["Year"] = df_from_sql["Year"].astype(int)

df_from_sql.set_index("Year", inplace=True)


# Filter data for years 1990–2013

df_from_sql = df_from_sql.loc[1990:2013]


# Train the ARIMA model

model = ARIMA(df_from_sql["deaths"], order=(1, 1, 1))  # Adjust the order (p, d, q) as needed

model_fit = model.fit()


# Forecast for 2025–2030

forecast_years = range(2025, 2031)

forecast = model_fit.forecast(steps=len(forecast_years))


# Create a DataFrame for forecasted values

forecast_df = pd.DataFrame({

    "Year": forecast_years,

    "deaths": forecast

}).set_index("Year")


# Combine historical and forecasted data

combined_data = pd.concat([df_from_sql, forecast_df])
```

```
# Plot the results
plt.figure(figsize=(12, 6))
plt.plot(df_from_sql.index, df_from_sql["deaths"], label="Historical Data", color="blue")
plt.plot(forecast_df.index, forecast_df["deaths"], label="Future Forecast (2025–2030)", color="orange")
plt.title("Forecasting Estimated number of deaths from TB in people who are HIV-positive for Nigeria")
plt.xlabel("Year")
plt.ylabel("Estimated Number of Deaths")
plt.legend()
plt.grid()
plt.show()


# Display forecasted values
print("\nForecasted values (2025–2030):")
print(forecast_df)
```

```
... Index(['Country or territory name', 'Year',
       'Estimated total population number',
       'Estimated prevalence of TB (all forms)',
       'Method to derive prevalence estimates',
       'Estimated number of deaths from TB (all forms, excluding HIV)',
       'Estimated number of deaths from TB in people who are HIV-positive',
       'Method to derive mortality estimates',
       'Estimated number of incident cases (all forms)',
       'Method to derive incidence estimates',
       'Estimated HIV in incident TB (percent)',
       'Estimated incidence of TB cases who are HIV-positive',
       'Method to derive TBHIV estimates',
       'Case detection rate (all forms), percent'],
      dtype='object')
```



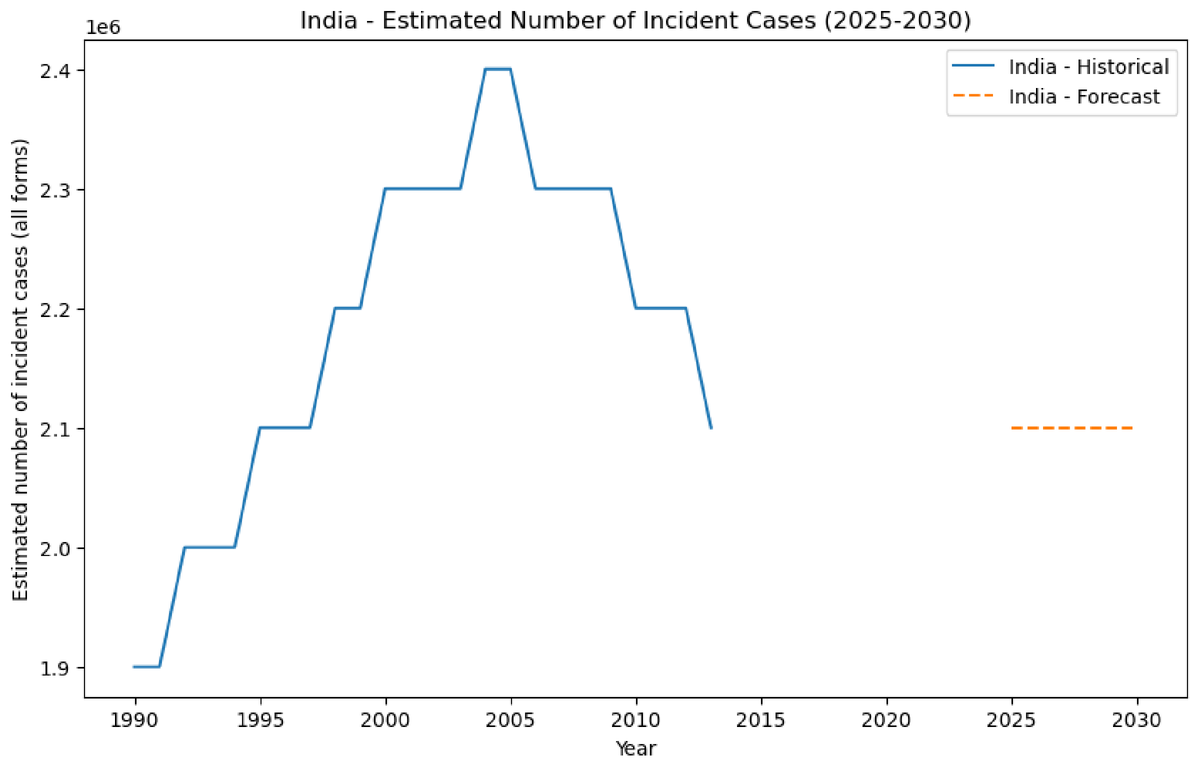
```
...
    Forecasted values (2025-2030):
                 deaths
    Year
    2025  84796.347251
    2026  84609.928643
    2027  84439.285735
    2028  84283.083505
    2029  84140.099911
    2030  84009.216322
```

```
df_from_sql = pd.read_sql('NEW_TB_Burden_Country', con=engine)

print(df_from_sql.columns)
```

```
...  Index(['Country or territory name', 'Year',
           'Estimated total population number',
           'Estimated prevalence of TB (all forms)',
           'Method to derive prevalence estimates',
           'Estimated number of deaths from TB (all forms, excluding HIV)',
           'Estimated number of deaths from TB in people who are HIV-positive',
           'Method to derive mortality estimates',
           'Estimated number of incident cases (all forms)',
           'Method to derive incidence estimates',
           'Estimated HIV in incident TB (percent)',
           'Estimated incidence of TB cases who are HIV-positive',
           'Method to derive TBHIV estimates',
           'Case detection rate (all forms), percent'],
          dtype='object')
```

```
#Forecast of incident cases for India and Nigeria

# Filter data for India and Nigeria

data_india = df_from_sql[df_from_sql["Country or territory name"] == "India"]

data_nigeria = df_from_sql[df_from_sql["Country or territory name"] == "Nigeria"]


# Ensure data is sorted by year and set year as index

data_india = data_india.sort_values(by="Year").set_index("Year")

data_nigeria = data_nigeria.sort_values(by="Year").set_index("Year")


# Train ARIMA model for India

model_india = ARIMA(data_india['Estimated number of incident cases (all forms)'],
order=(1, 1, 1))

fit_india = model_india.fit()


# Train ARIMA model for Nigeria

model_nigeria = ARIMA(data_nigeria['Estimated number of incident cases (all forms)'],
order=(1, 1, 1))

fit_nigeria = model_nigeria.fit()


# Forecast from 2025 to 2030

forecast_years = [2025, 2026, 2027, 2028, 2029, 2030]
```

```python
forecast_india = fit_india.forecast(steps=len(forecast_years))

forecast_nigeria = fit_nigeria.forecast(steps=len(forecast_years))


# Create forecast DataFrames

forecast_india_df = pd.DataFrame({"Year": forecast_years, "Forecast_India":
forecast_india})

forecast_nigeria_df = pd.DataFrame({"Year": forecast_years, "Forecast_Nigeria":
forecast_nigeria})


# Merge forecasts with historical data

historical_india = data_india.reset_index()

historical_nigeria = data_nigeria.reset_index()

# Plot India

plt.figure(figsize=(10, 6))

plt.plot(historical_india["Year"], historical_india['Estimated number of incident cases (all
forms)'], label="India - Historical")

plt.plot(forecast_india_df["Year"], forecast_india_df["Forecast_India"], label="India -
Forecast", linestyle="--")

plt.xlabel("Year")

plt.ylabel("Estimated number of incident cases (all forms)")

plt.title("India - Estimated Number of Incident Cases (2025-2030)")

plt.legend()

plt.show()
```



Plot Nigeria

plt.figure(figsize=(10, 6))

plt.plot(historical_nigeria["Year"], historical_nigeria['Estimated number of incident cases (all forms)'], label="Nigeria - Historical")

plt.plot(forecast_nigeria_df["Year"], forecast_nigeria_df["Forecast_Nigeria"], label="Nigeria - Forecast", linestyle="--")

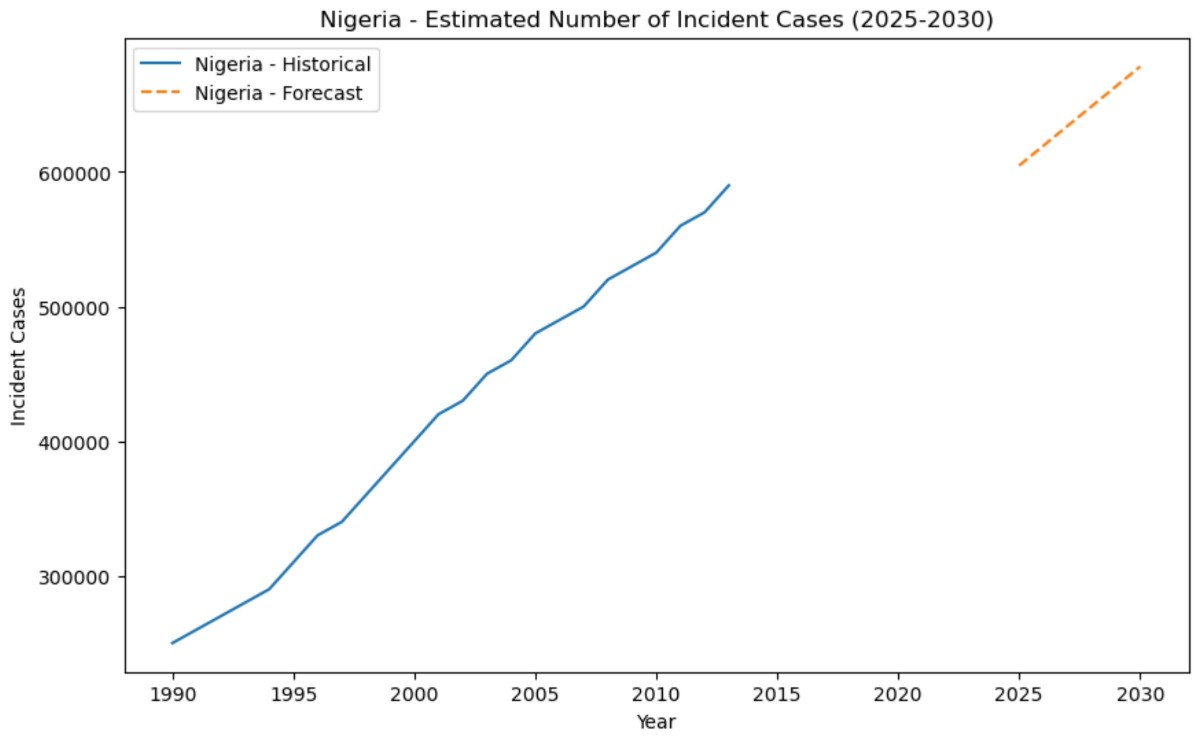
plt.xlabel("Year")

plt.ylabel("Incident Cases")

plt.title("Nigeria - Estimated Number of Incident Cases (2025-2030)")

plt.legend()

plt.show()




```
# Save results to CSV
forecast_india_df.to_csv("forecast_india_2025_2030.csv", index=False)
forecast_nigeria_df.to_csv("forecast_nigeria_2025_2030.csv", index=False)
```