WORLD POPULATION DATA ANALYSIS USING PYTHON AND SQL

Presented by: Bruna Augustin

INTRODUCTION

Population analysis helps governments, organizations, and researchers understand demographic trends and their impact on the world.

Why Data Analysis?

- Identify population growth trends
- Understand country-wise and continent-wise distribution
- Help in future planning and policy making

PROJECT FOCUS

01 Data Cleaning & Preprocessing

Ensured data quality by handling missing values, correcting types, and formatting columns.

02 Insights Through Python Visualizations

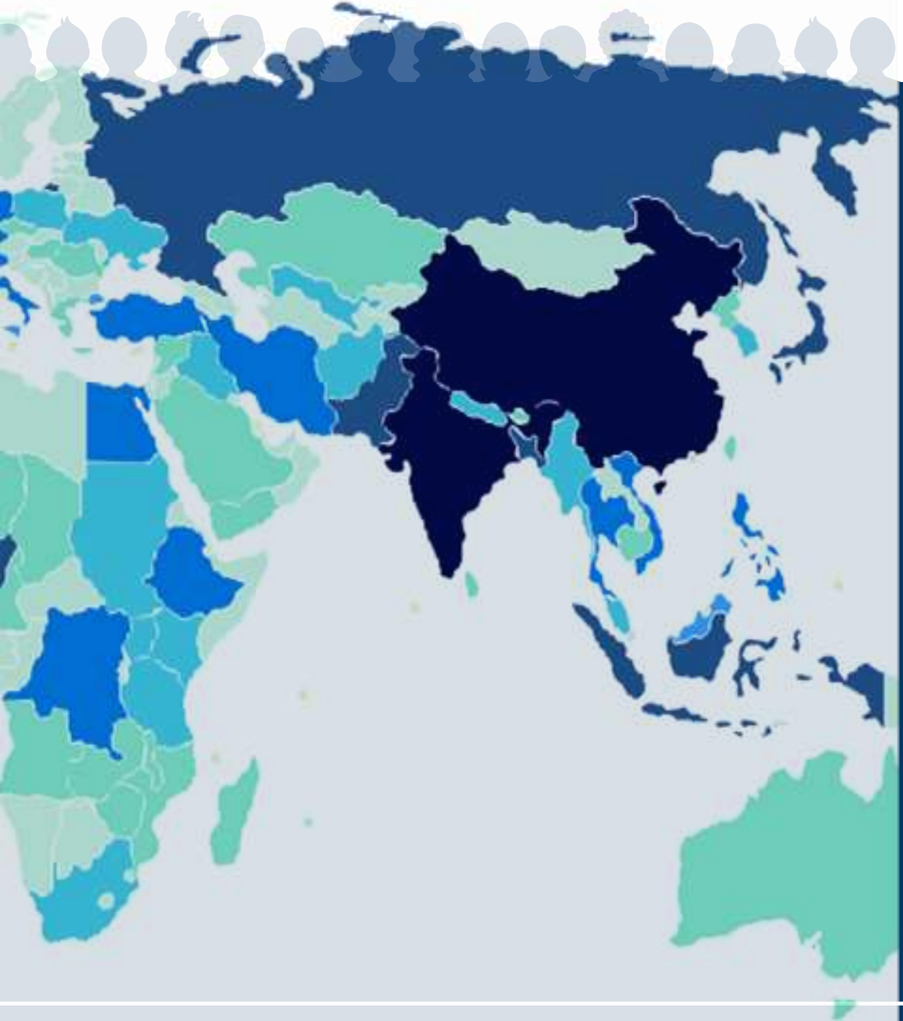
Created insightful graphs using Matplotlib and Seaborn to understand patterns in population data.

03 Data Analysis using SQL

Ran structured queries to extract key insights like year-wise, country-wise, and continent-wise population stats.

04 Challenges Faced

Studied global population growth over time to identify key trends and demographic shifts.



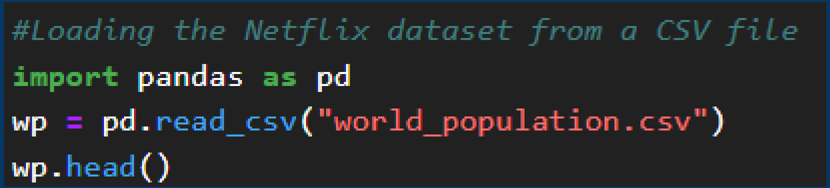
Understanding the Dataset

- Dataset Name: world_population
- Source: Unified Mentor
- Total Records: 200+ countries across multiple years
- Columns: Rank, CCA3, Country/Territory, Capital, Continent, 2022 Population, 2020 Population, 2015 Population, 2010 Population, 2000 Population, 1990 Population, 1980 Population, 1970 Population, Area (km²), Density (per km²), Growth Rate, World Population Percentage

01

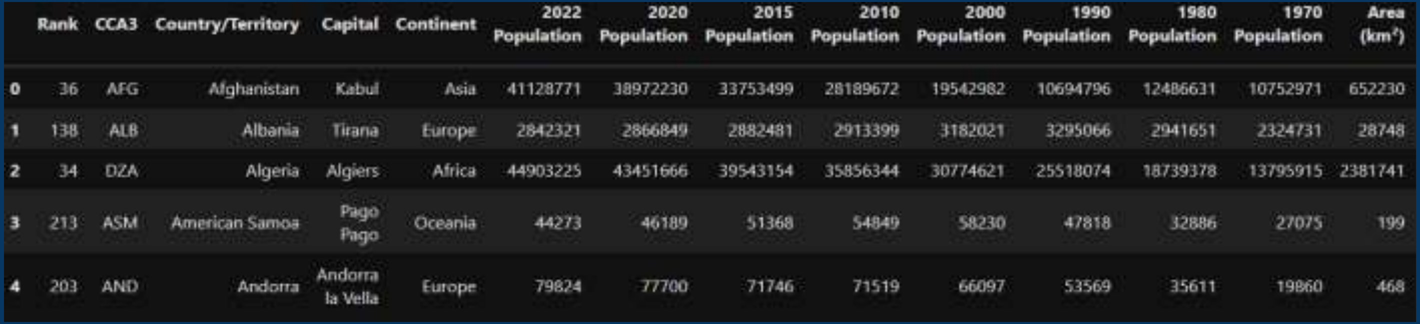
Data Cleaning & Preprocessing With Python

Raw Dataset Preview (Original Data)



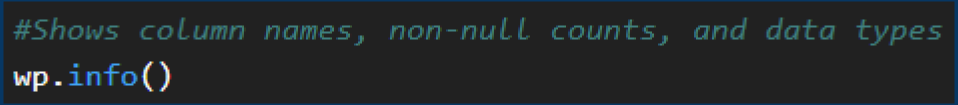
```
#Loading the Netflix dataset from a CSV file
import pandas as pd
wp = pd.read_csv("world_population.csv")
wp.head()
```

Output



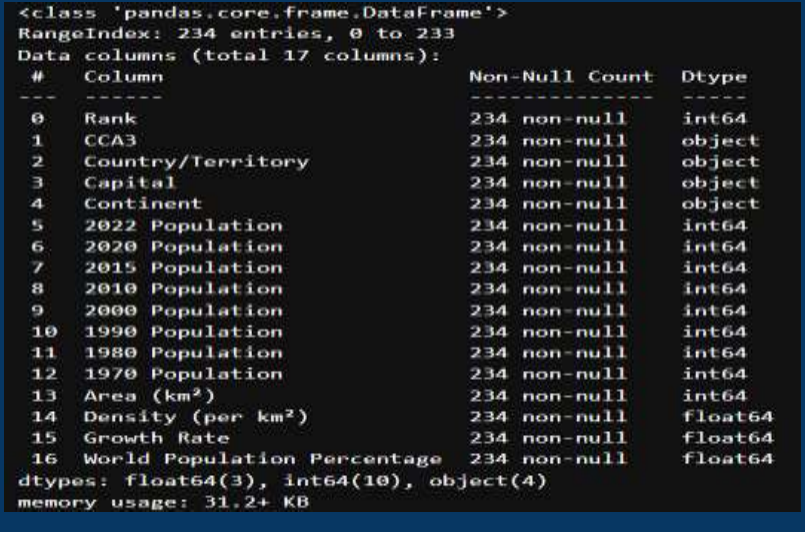
	Rank	CCA3	Country/Territory	Capital	Continent	2022 Population	2020 Population	2015 Population	2010 Population	2000 Population	1990 Population	1980 Population	1970 Population	Area (km²)
0	36	AFG	Afghanistan	Kabul	Asia	41128771	38972230	33753499	28189672	19542982	10694796	12486631	10752971	652230
1	138	ALB	Albania	Tirana	Europe	2842321	2866849	2882481	2913399	3182021	3295066	2941651	2324731	28748
2	34	DZA	Algeria	Algiers	Africa	44903225	43451666	39543154	35856344	30774621	25518074	18739378	13795915	2381741
3	213	ASM	American Samoa	Pago Pago	Oceania	44273	46189	51368	54849	58230	47818	32886	27075	199
4	203	AND	Andorra	Andorra la Vella	Europe	79824	77700	71746	71519	66097	53569	35611	19860	468

Checking Data Types & Structure



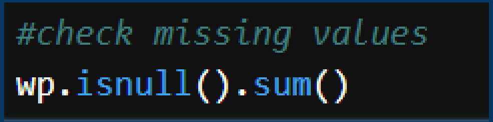
```
#Shows column names, non-null counts, and data types
wp.info()
```

Output



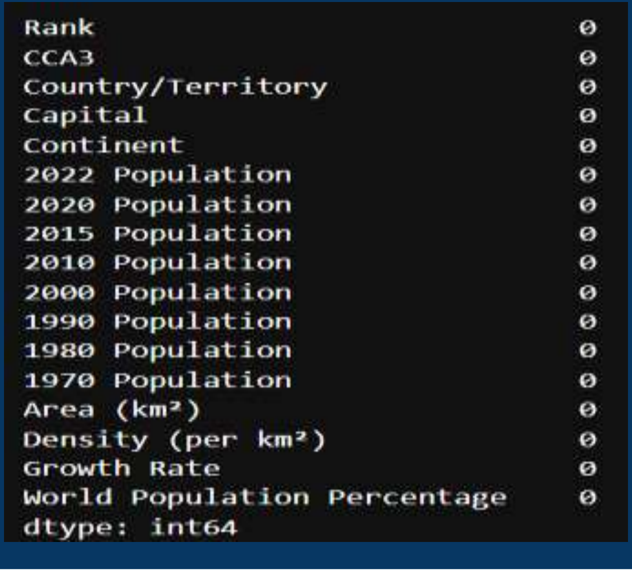
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 234 entries, 0 to 233
Data columns (total 17 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   Rank                         234 non-null    int64  
 1   CCA3                         234 non-null    object 
 2   Country/Territory            234 non-null    object 
 3   Capital                      234 non-null    object 
 4   Continent                    234 non-null    object 
 5   2022 Population              234 non-null    int64  
 6   2020 Population              234 non-null    int64  
 7   2015 Population              234 non-null    int64  
 8   2010 Population              234 non-null    int64  
 9   2000 Population              234 non-null    int64  
 10  1990 Population              234 non-null    int64  
 11  1980 Population              234 non-null    int64  
 12  1970 Population              234 non-null    int64  
 13  Area (km²)                   234 non-null    int64  
 14  Density (per km²)            234 non-null    float64
 15  Growth Rate                  234 non-null    float64
 16  World Population Percentage  234 non-null    float64
dtypes: float64(3), int64(10), object(4)
memory usage: 31.2+ KB
```

Checking Missing Values



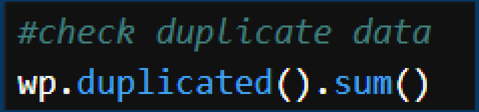
```
#check missing values
wp.isnull().sum()
```

Output



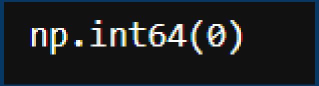
```
Rank                           0
CCA3                           0
Country/Territory              0
Capital                        0
Continent                      0
2022 Population                0
2020 Population                0
2015 Population                0
2010 Population                0
2000 Population                0
1990 Population                0
1980 Population                0
1970 Population                0
Area (km²)                     0
Density (per km²)              0
Growth Rate                    0
World Population Percentage    0
dtype: int64
```

Checking Duplicates Values



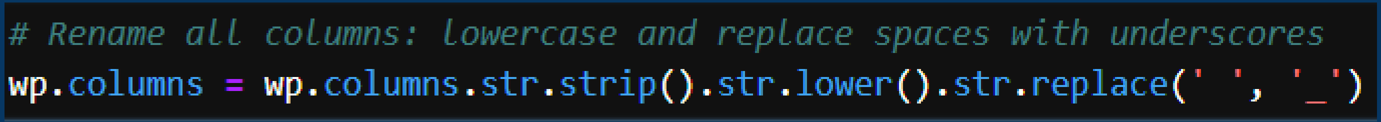
```
#check duplicate data
wp.duplicated().sum()
```

Output



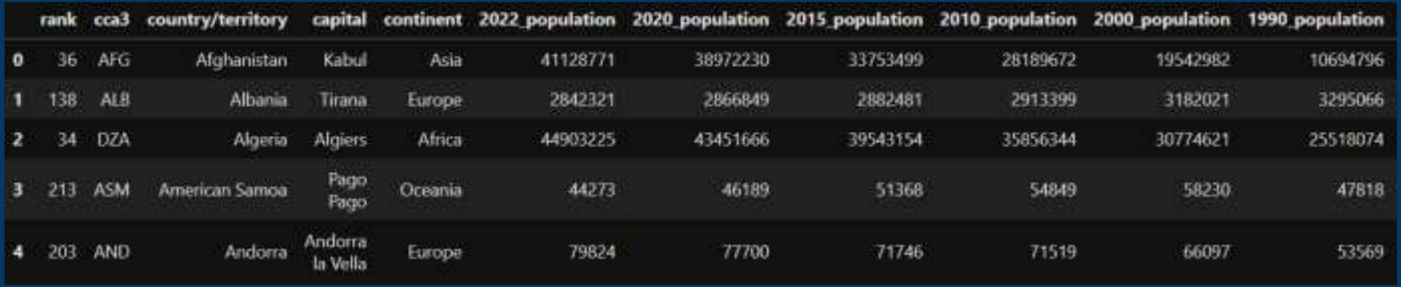
```
np.int64(0)
```

Rename All Columns



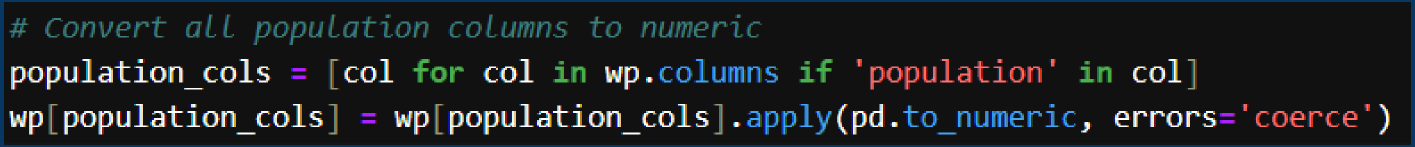
```
# Rename all columns: lowercase and replace spaces with underscores
wp.columns = wp.columns.str.strip().str.lower().str.replace(' ', '_')
```

Output



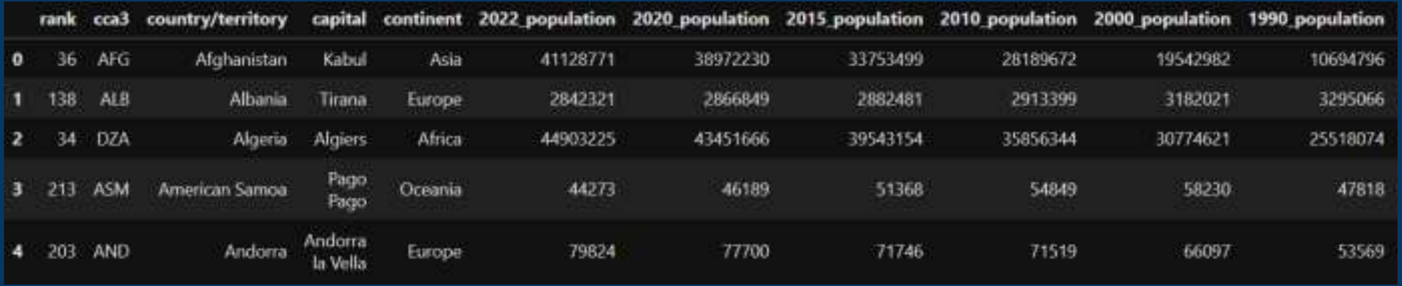
	rank	cca3	country/territory	capital	continent	2022_population	2020_population	2015_population	2010_population	2000_population	1990_population
0	36	AFG	Afghanistan	Kabul	Asia	41128771	38972230	33753499	28189672	19542982	10694796
1	138	ALB	Albania	Tirana	Europe	2842321	2866849	2882481	2913399	3182021	3295066
2	34	DZA	Algeria	Algiers	Africa	44903225	43451666	39543154	35856344	30774621	25518074
3	213	ASM	American Samoa	Pago Pago	Oceania	44273	46189	51368	54849	58230	47818
4	203	AND	Andorra	Andorra la Vella	Europe	79824	77700	71746	71519	66097	53569

Convert Columns to Numeric

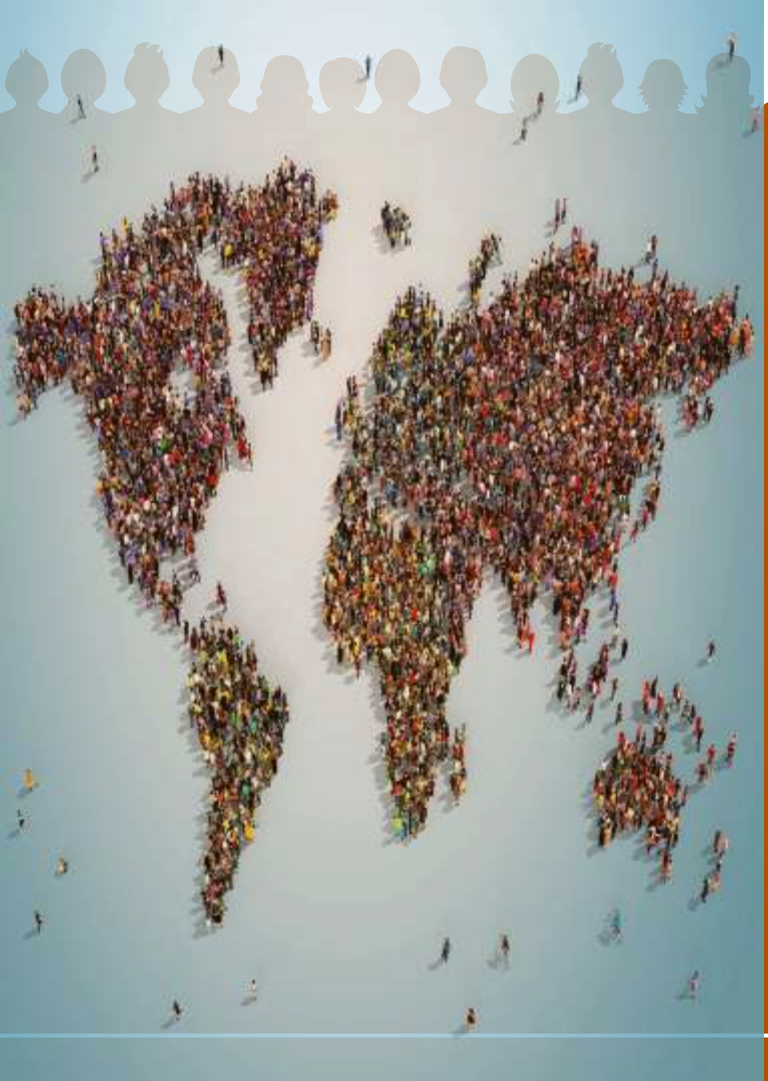


```
# Convert all population columns to numeric
population_cols = [col for col in wp.columns if 'population' in col]
wp[population_cols] = wp[population_cols].apply(pd.to_numeric, errors='coerce')
```

Output



	rank	cca3	country/territory	capital	continent	2022_population	2020_population	2015_population	2010_population	2000_population	1990_population
0	36	AFG	Afghanistan	Kabul	Asia	41128771	38972230	33753499	28189672	19542982	10694796
1	138	ALB	Albania	Tirana	Europe	2842321	2866849	2882481	2913399	3182021	3295066
2	34	DZA	Algeria	Algiers	Africa	44903225	43451666	39543154	35856344	30774621	25518074
3	213	ASM	American Samoa	Pago Pago	Oceania	44273	46189	51368	54849	58230	47818
4	203	AND	Andorra	Andorra la Vella	Europe	79824	77700	71746	71519	66097	53569

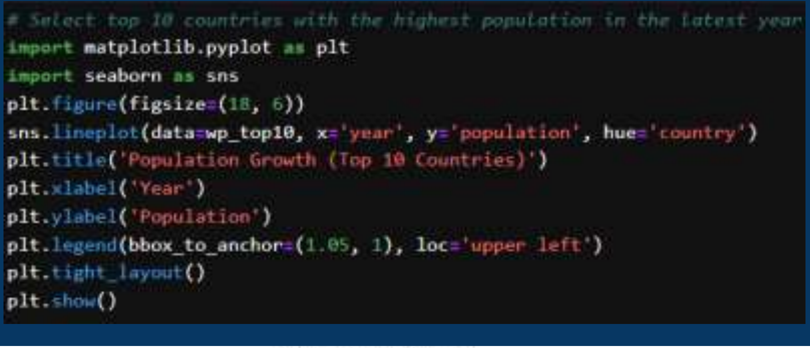


02

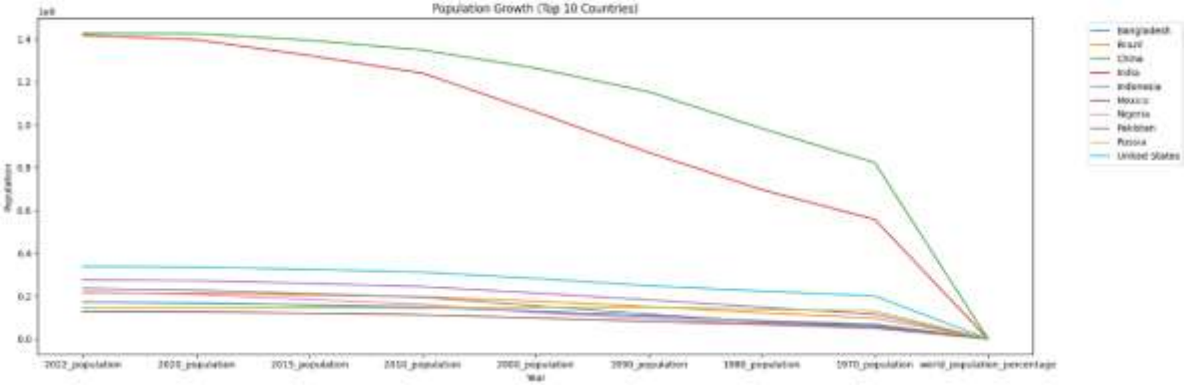
Insights Through Python Visualizations

Visualizations (Matplotlib & Seaborn)

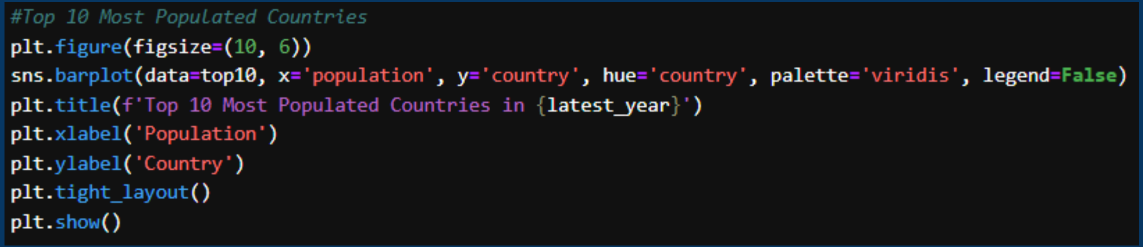
Population Growth Over Years



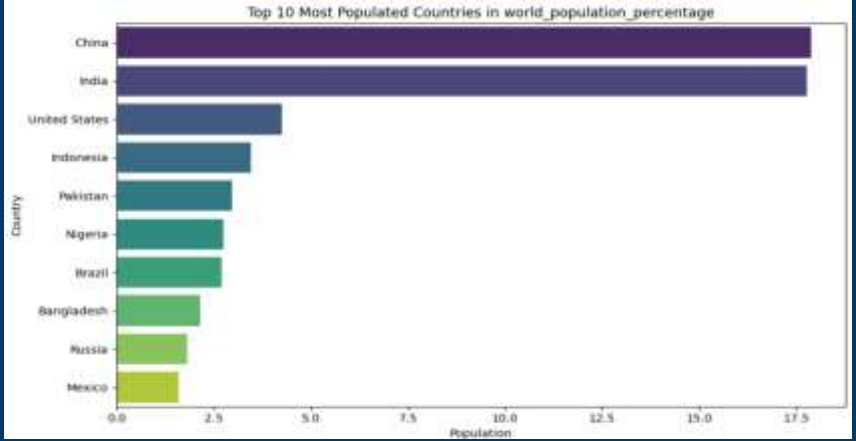
```
# Select top 10 countries with the highest population in the latest year
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(18, 6))
sns.lineplot(data=wp_top10, x='year', y='population', hue='country')
plt.title('Population Growth (Top 10 Countries)')
plt.xlabel('Year')
plt.ylabel('Population')
plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```



Top 10 Most Populated Country



```
#Top 10 Most Populated Countries
plt.figure(figsize=(10, 6))
sns.barplot(data=top10, x='population', y='country', hue='country', palette='viridis', legend=False)
plt.title(f'Top 10 Most Populated Countries in {latest_year}')
plt.xlabel('Population')
plt.ylabel('Country')
plt.tight_layout()
plt.show()
```

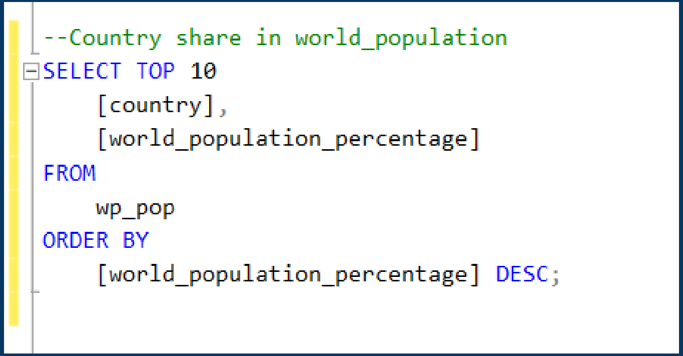


03

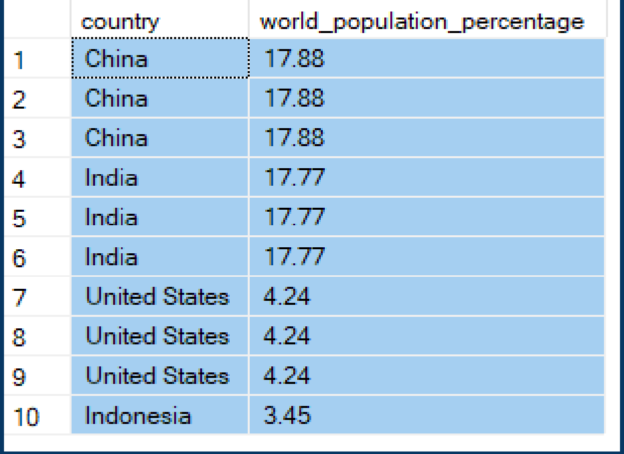
World Population Analysis With SQL

Country Share in World Population

1. Query



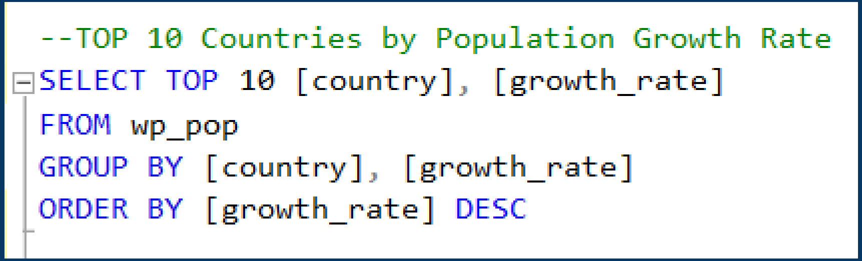
```
--Country share in world_population
SELECT TOP 10
    [country],
    [world_population_percentage]
FROM
    wp_pop
ORDER BY
    [world_population_percentage] DESC;
```



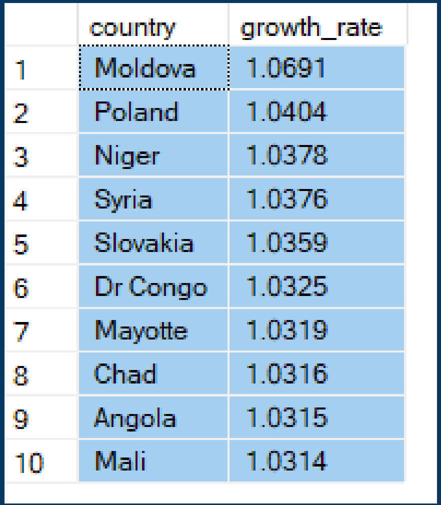
	country	world_population_percentage
1	China	17.88
2	China	17.88
3	China	17.88
4	India	17.77
5	India	17.77
6	India	17.77
7	United States	4.24
8	United States	4.24
9	United States	4.24
10	Indonesia	3.45

Top 10 Countries by Population Growth Rate

2. Query



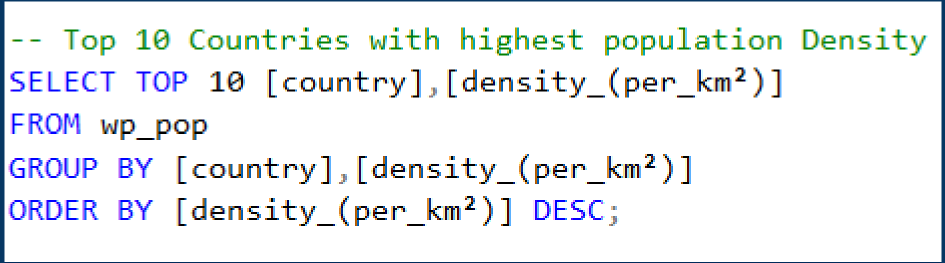
```
--TOP 10 Countries by Population Growth Rate
SELECT TOP 10 [country], [growth_rate]
FROM wp_pop
GROUP BY [country], [growth_rate]
ORDER BY [growth_rate] DESC
```



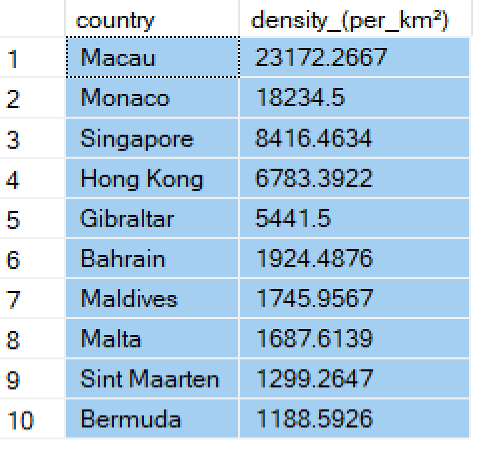
	country	growth_rate
1	Moldova	1.0691
2	Poland	1.0404
3	Niger	1.0378
4	Syria	1.0376
5	Slovakia	1.0359
6	Dr Congo	1.0325
7	Mayotte	1.0319
8	Chad	1.0316
9	Angola	1.0315
10	Mali	1.0314

Top 10 Countries with Highest Population Density

3. Query



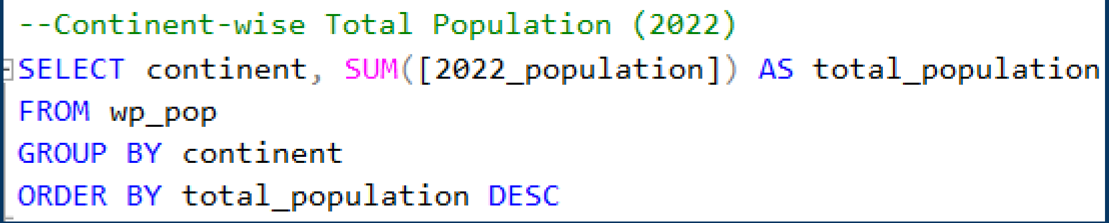
```
-- Top 10 Countries with highest population Density
SELECT TOP 10 [country],[density_(per_km²)]
FROM wp_pop
GROUP BY [country],[density_(per_km²)]
ORDER BY [density_(per_km²)] DESC;
```



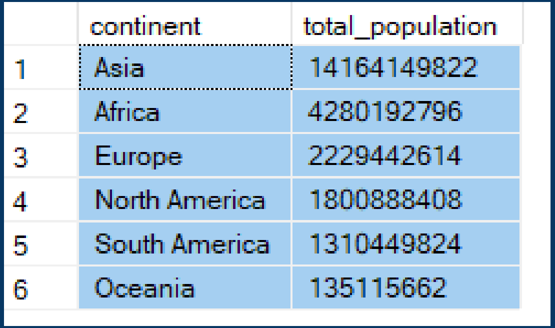
	country	density_(per_km²)
1	Macau	23172.2667
2	Monaco	18234.5
3	Singapore	8416.4634
4	Hong Kong	6783.3922
5	Gibraltar	5441.5
6	Bahrain	1924.4876
7	Maldives	1745.9567
8	Malta	1687.6139
9	Sint Maarten	1299.2647
10	Bermuda	1188.5926

Continent-wise Total Population (2022)

4. Query



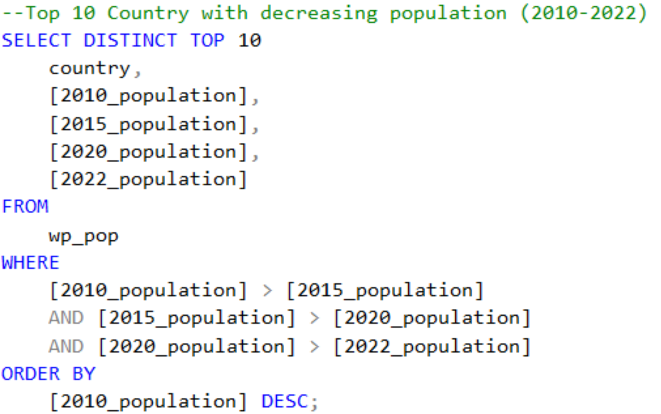
```
--Continent-wise Total Population (2022)
SELECT continent, SUM([2022_population]) AS total_population
FROM wp_pop
GROUP BY continent
ORDER BY total_population DESC
```



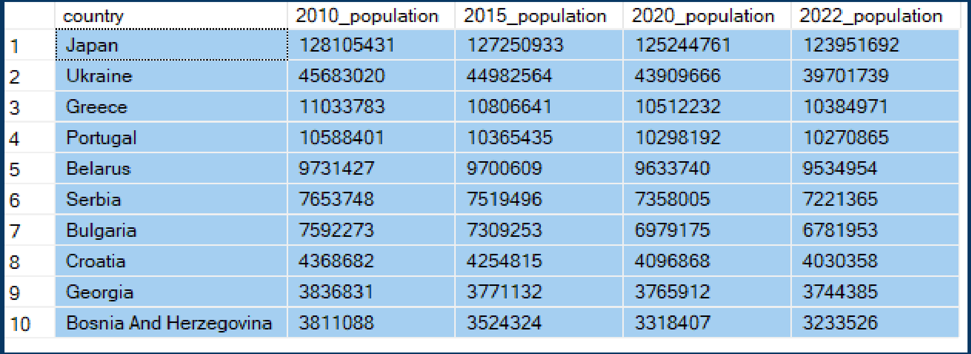
	continent	total_population
1	Asia	14164149822
2	Africa	4280192796
3	Europe	2229442614
4	North America	1800888408
5	South America	1310449824
6	Oceania	135115662

Top 10 Country with decreasing population (2010-2022)

5. Query

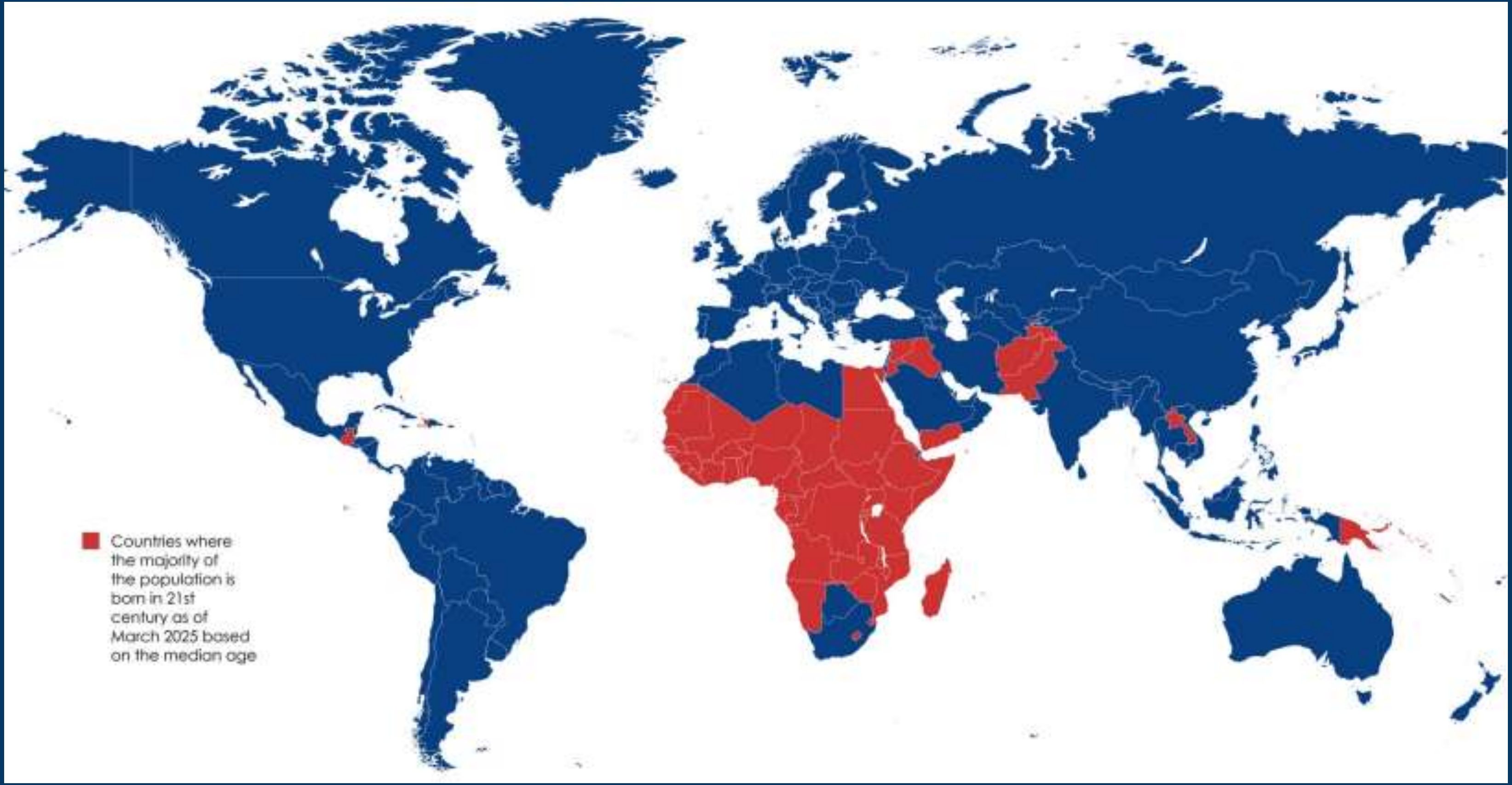


```
--Top 10 Country with decreasing population (2010-2022)
SELECT DISTINCT TOP 10
    country,
    [2010_population],
    [2015_population],
    [2020_population],
    [2022_population]
FROM
    wp_pop
WHERE
    [2010_population] > [2015_population]
    AND [2015_population] > [2020_population]
    AND [2020_population] > [2022_population]
ORDER BY
    [2010_population] DESC;
```



	country	2010_population	2015_population	2020_population	2022_population
1	Japan	128105431	127250933	125244761	123951692
2	Ukraine	45683020	44982564	43909666	39701739
3	Greece	11033783	10806641	10512232	10384971
4	Portugal	10588401	10365435	10298192	10270865
5	Belarus	9731427	9700609	9633740	9534954
6	Serbia	7653748	7519496	7358005	7221365
7	Bulgaria	7592273	7309253	6979175	6781953
8	Croatia	4368682	4254815	4096868	4030358
9	Georgia	3836831	3771132	3765912	3744385
10	Bosnia And Herzegovina	3811088	3524324	3318407	3233526

World Population Map (2025)



🟥 Countries where the majority of the population is born in 21st century as of March 2025 based on the median age

04

Challenges Faced

1. Data Loss after Cleaning and Saving

After cleaning the dataset and saving it as a new CSV file, only 3 columns were saved instead of the complete dataset. This caused the loss of crucial information needed for proper analysis.

2. Visualization Errors Due to Missing Columns

Because of the missing columns, several visualization attempts using Seaborn and Matplotlib failed. I faced errors like “ValueError: Could not interpret value” since the expected data wasn’t available.

3. Duplicate Country Entries in SQL Query Output

Some SQL queries resulted in repeated country names (appearing multiple times), which made the analysis confusing and cluttered. I had to debug and ensure distinct and clean outputs.

4. Difficulty in Writing Complex Conditional Queries

Writing SQL queries for trends like continuous population decline (2010 to 2022) required proper logical sequencing. Mistakes in syntax or condition order led to incorrect or incomplete results.

5. Data Type Mismatches During Analysis

Some numeric fields were read as strings, which caused problems during aggregation, sorting, and plotting. I had to convert data types to perform accurate calculations.

05

Conclusion & Key Takeaways

- Clean and structured data helped in understanding patterns more clearly.
- Asia has the highest share of the global population.
- African countries showed the fastest growth rates in population.
- Python and SQL made it easier to clean, analyze, and visualize data.
- Proper data handling turned raw numbers into meaningful insights.

THANK YOU

Connect with me: linkedin.com/in/bruna-augustin0927official