QTM 350 Final Project Report

Caleb Sharkey (2514723) Katherine Martini (2549333)
Joyce Chen (2485906) Lucas Lobo (2555247)

Table of contents

1 Introduction

For this project, we will be investigating population dynamics from the Caribbean and Central and South America from 1975-2024. We are specifically interested in analyzing how indicators such as life expectancy, mortality rate, and adolescent fertility rate interact with one another. We will be utilizing data from the publicly availiable World Development Index (WDI) from the World Bank. We will use additional attributes like immunization (measles, DPT, HEPB3), HIV prevalence, urban and rural populations, unemployment rates, and surface area for a deeper look at our specified region's status and history.

1.0.1 Central Question:

How do various health-related, economic, and geographic factors contribute towards and interact with life expectancy at birth, mortality

rate, and adolescent fertility rate?

2 Data Description

We chose to focus on the 'population dynamics' research question within the WDI Dataset. Specifically, we wanted to explored how various health-related, economic, and geographic factors contributed towards and interacted with life expectancy at birth, mortality rate, and adolescent fertility rate. Some of these factors (or 'services,' as named by the WDI) include immunization rates for DPT, HepB3, measles, prevalence of HIV, TB case detection rates, proportion of expenditure spent on health-related measures, urban and rural populations, unemployment rates, surface area, and rule of law estimates. To narrow our focus, we only included data from Central and South American countries: Argentina, Belize, Costa Rica, El Salvador, Guatemala, Honduras, Mexico, Nicaragua, Panama, Bolivia, Brazil, Chile, Colombia, Ecuador, Guyana, Paraguay, Peru, Suriname, Uruguay, Venezuela, Cuba, Dominican Republic, Haiti, Puerto Rico (territory), St. Martin. Finally, we included data from the past 50 years (1975-2024).

3 Data Cleaning and Preparation

(Lucas)

3.1 Loading + Creating the Database.

```
# Relevant packages:
import sqlite3
import pandas as pd

# Loading the WDI data as a csv (using pandas):
# Change relative path to your downloaded WDI data
df = pd.read_csv("../data/wdi_rawdata.csv", encoding='latin1')

# Created a database by running the following two commands in my terminal:

## cd "C:\Users\lcsrl\Downloads"
## touch qtm350_project.db

import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

# Set up connection:
conn = sqlite3.connect("../data/qtm350_project.db")
cursor = conn.cursor()
```

```
# Write 'df' to SQL:
cursor.execute("DROP TABLE IF EXISTS wdi;")
df.to_sql('wdi', conn, if_exists='replace', index=False)
```

```
542
```

3.2 Data Cleaning + Transformation:

Off the bat, there are lots of oddly-named variables and warped data. We'll create a new cleaned table. We'll drop/modify N/A values (denoted as ..) according to the following procedures.

```
# new, cleaned dataframe
# 1: create a new 'year' list (to replace the current format)
year_columns = [
    f'"{year} [YR{year}]" AS "{year}"' for year in range(1975, 2025)
]

# 2: new table only with relevant columns: country_name, indicator, and years
cursor.execute("DROP TABLE IF EXISTS wdi_renamed;")
q_keep = f"""
CREATE TABLE wdi_renamed AS
SELECT
    "Country Name" AS country_name,
    "Series Name" AS indicator,
    {', '.join(year_columns)}
FROM wdi
"""
conn.execute(q_keep)
```

```
<sqlite3.Cursor at 0x1308f83c0>
```

```
pd.read_sql("SELECT * FROM wdi_renamed", conn).head()
```

	country_name	indicator	1975	1976	1977	1978
0	Argentina	Life expectancy at birth, total (years)	66.965	67.03	67.595	67.999
1	Argentina	Mortality rate, under-5 (per 1,000 live births)	63.3	59.8	55.6	51.2
2	Argentina	Adolescent fertility rate (births per 1,000 wo...	73.076	74.913	76.634	77.695
3	Argentina	Prevalence of HIV, total (% of population ages...	..	..	..	..
4	Argentina	Immunization, DPT (% of children ages 12-23 mo...	..	..	..	..

Sorting the countries into three regions as follows:

Central America (9 countries): - Belize, Costa Rica, El Salvador, Guatemala, Honduras, Nicaragua, Panama, Mexico, Bolivia, South America (10 countries): - Argentina, Brazil, Chile, Colombia, Guyana, Paraguay, Peru, Suriname, Uruguay, Venezuela, RB.

Caribbean (5 countries): - Dominican Republic, Haiti, Jamaica, St. Martin (French part), Puerto Rico.

```
conn.execute("DROP TABLE IF EXISTS wdi_region;")

# writes an implicit 'if else' function in sql to create a new column 'region' based on the
q_region = """
CREATE TABLE wdi_region AS
SELECT
    country_name,
    CASE
        WHEN country_name IN (
            'Belize', 'Costa Rica', 'El Salvador', 'Guatemala', 'Honduras', 'Nicaragua', 'Pa
        ) THEN 'Central America'
        WHEN country_name IN (
            'Argentina', 'Brazil', 'Chile', 'Colombia', 'Ecuador', 'Guyana', 'Paraguay', 'Pe
        ) THEN 'South America'
        WHEN country_name IN (
            'Cuba', 'Dominican Republic', 'Haiti', 'Jamaica', 'St. Martin (French part)', 'P
        ) THEN 'Caribbean'
        ELSE 'Other'
    END AS region,
    indicator,
    "1975", "1976", "1977", "1978", "1979", "1980", "1981", "1982", "1983", "1984", "1985",
    "1990", "1991", "1992", "1993", "1994", "1995", "1996", "1997", "1998", "1999", "2000",
    "2005", "2006", "2007", "2008", "2009", "2010", "2011", "2012", "2013", "2014", "2015",
    "2020", "2021", "2022", "2023", "2024"
FROM wdi_renamed
"""

conn.execute(q_region)
```

```
<sqlite3.Cursor at 0x1308f8d40>
```

We'll create a pivot table (long) in order to display each variable in each year by country.

```
# SQL code that mimics 'melt' in python: iterates over each year and stores a list of querie
year_range = range(1975, 2025)
union_queries = []

# creates a list of SQL commands
# column order: year, country_name
conn.execute("DROP TABLE IF EXISTS wdi_long;")
for year in year_range:
    union_queries.append(f"""
    SELECT
        '{year}' AS year,
        country_name,
        region,
        indicator,
        "{year}" AS value
    FROM wdi_region
    WHERE "{year}" IS NOT NULL
    """)

# joins and executes each individual year using UNION ALL (basically stacks on top of each
q_long = f"""
CREATE TABLE wdi_long AS
{'UNION ALL'.join(union_queries)}
"""

conn.execute(q_long)
```

```
<sqlite3.Cursor at 0x1308f9840>
```

Lots of information for various years is unavailable, perhaps because the country does not track certain info, they only started after a certain year, etc. We'll create a modified version of this pivot table to only include rows where the data is present

```
conn.execute("DROP TABLE IF EXISTS wdi_long_clean;")
q_filter_non_null = """
CREATE TABLE wdi_long_clean AS
SELECT *
FROM wdi_long
WHERE value IS NOT NULL AND value != '..'
"""
conn.execute(q_filter_non_null)

# Here's a sample of the cleaned data:
pd.read_sql("SELECT * FROM wdi_long_clean", conn).head(25)
```

	year	country_name	region	indicator	value
0	1975	Argentina	South America	Life expectancy at birth, total (years)	66.965
1	1975	Argentina	South America	Mortality rate, under-5 (per 1,000 live births)	63.3
2	1975	Argentina	South America	Adolescent fertility rate (births per 1,000 wo...	73.076
3	1975	Argentina	South America	Urban population	20950006
4	1975	Argentina	South America	Unemployment, total (% of total labor force) (...	2.3
5	1975	Argentina	South America	Trade (% of GDP)	11.802724
6	1975	Argentina	South America	Surface area (sq. km)	2780400

	year	country_name	region	indicator	value
7	1975	Argentina	South America	School enrollment, primary (% net)	96.32775
8	1975	Argentina	South America	Rural population	4924101
9	1975	Belize	Central America	Life expectancy at birth, total (years)	65.377
10	1975	Belize	Central America	Mortality rate, under-5 (per 1,000 live births)	80
11	1975	Belize	Central America	Adolescent fertility rate (births per 1,000 wo...	142.49
12	1975	Belize	Central America	Urban population	65416
13	1975	Belize	Central America	Surface area (sq. km)	22970
14	1975	Belize	Central America	Rural population	64971
15	1975	Costa Rica	Central America	Life expectancy at birth, total (years)	70.253
16	1975	Costa Rica	Central America	Mortality rate, under-5 (per 1,000 live births)	45.4
17	1975	Costa Rica	Central America	Adolescent fertility rate (births per 1,000 wo...	91.632
18	1975	Costa Rica	Central America	Urban population	865318
19	1975	Costa Rica	Central America	Trade (% of GDP)	68.609785
20	1975	Costa Rica	Central America	Surface area (sq. km)	51100
21	1975	Costa Rica	Central America	School enrollment, primary (% net)	90.45446
22	1975	Costa Rica	Central America	Rural population	1227401
23	1975	El Salvador	Central America	Life expectancy at birth, total (years)	53.422
24	1975	El Salvador	Central America	Mortality rate, under-5 (per 1,000 live births)	130.7

3.3 Descriptive Statistics (in SQL)

1. Top 10 Countries by Average Life Expectancy across various time periods:

- 1975-2024 (all years in dataset).
- 1980-1989 (80s).
- 2010-2019 (2010s).

```
q1_a = """
SELECT country_name, AVG(value) AS avg_life_exp
FROM wdi_long_clean
WHERE indicator = 'Life expectancy at birth, total (years)'
GROUP BY country_name
ORDER BY avg_life_exp DESC
LIMIT 10
"""
q1_a_table = pd.read_sql(q1_a, conn)
q1_a_table
```

	country_name	avg_life_exp
0	Costa Rica	76.953000
1	St. Martin (French part)	76.319592
2	Puerto Rico	75.974449

	country_name	avg_life_exp
3	Chile	75.545469
4	Cuba	75.323898
5	Uruguay	74.259551
6	Panama	73.476163
7	Argentina	73.002898
8	Venezuela, RB	71.322490
9	Colombia	71.205878

```
q1_b = """
SELECT country_name, AVG(value) AS avg_life_exp
FROM wdi_long_clean
WHERE indicator = 'Life expectancy at birth, total (years)'
  AND year IN ('1980', '1981', '1982', '1983', '1984', '1985', '1986', '1987', '1988', '1989
GROUP BY country_name
ORDER BY avg_life_exp DESC
LIMIT 10
"""
q1_b_table = pd.read_sql(q1_b, conn)
q1_b_table
```

	country_name	avg_life_exp
0	Costa Rica	74.5500
1	Cuba	73.4186
2	Puerto Rico	73.0772
3	St. Martin (French part)	73.0160
4	Uruguay	71.7568
5	Chile	71.3751
6	Panama	70.2783
7	Argentina	69.8508
8	Venezuela, RB	69.7803
9	Belize	68.2004

```
q1_c = """
SELECT country_name, AVG(value) AS avg_life_exp
FROM wdi_long_clean
WHERE indicator = 'Life expectancy at birth, total (years)'
  AND year IN ('2010', '2011', '2012', '2013', '2014', '2015', '2016', '2017', '2018', '2019
GROUP BY country_name
ORDER BY avg_life_exp DESC
LIMIT 10
"""
```

```
q1_c_table = pd.read_sql(q1_c, conn)
q1_c_table
```

	country_name	avg_life_exp
0	Costa Rica	80.0098
1	Puerto Rico	79.9724
2	Chile	79.8613
3	St. Martin (French part)	79.4908
4	Cuba	77.7397
5	Panama	77.4483
6	Uruguay	77.1427
7	Argentina	76.2544
8	Ecuador	76.0896
9	Colombia	75.9536

2. Highest rates of immunization for DPT, HepB3, and measles for each country:

In other words, the number in each cell represents the immunization rate that was highest for that country across the years in the dataset. We will also include a value 'agg_immunization_rate', which is computed by taking teh average of the three maximum immunization rates.

```
q2 = """
SELECT country_name,
       MAX(CASE WHEN indicator = 'Immunization, DPT (% of children ages 12-23 months)' THEN
       MAX(CASE WHEN indicator = 'Immunization, HepB3 (% of one-year-old children)' THEN val
       MAX(CASE WHEN indicator = 'Immunization, measles (% of children ages 12-23 months)' T
       (MAX(CASE WHEN indicator = 'Immunization, DPT (% of children ages 12-23 months)' THEN
        MAX(CASE WHEN indicator = 'Immunization, HepB3 (% of one-year-old children)' THEN va
        MAX(CASE WHEN indicator = 'Immunization, measles (% of children ages 12-23 months)'
FROM wdi_long_clean
WHERE indicator IN (
    'Immunization, DPT (% of children ages 12-23 months)',
    'Immunization, HepB3 (% of one-year-old children)',
    'Immunization, measles (% of children ages 12-23 months)'
)
GROUP BY country_name
ORDER BY agg_immunization_rate DESC
LIMIT 23;
"""
q2_table = pd.read_sql(q2, conn)
q2_table
```

	country_name	max_dpt	max_hepb3	max_measles	agg_immunization_rate
0	Mexico	99	99	99	99
1	Honduras	99	99	99	99
2	Guyana	99	99	99	99
3	El Salvador	99	99	99	99
4	Cuba	99	99	99	99
5	Brazil	99	99	99	99
6	Panama	99	98	99	98
7	Nicaragua	98	98	99	98
8	Costa Rica	99	98	99	98
9	Chile	99	97	99	98
10	Belize	98	98	99	98
11	Uruguay	97	96	99	97
12	Peru	99	95	98	97
13	Ecuador	96	96	99	97
14	Argentina	98	94	99	97
15	Guatemala	96	96	96	96
16	Paraguay	98	93	96	95
17	Bolivia	95	95	96	95
18	Colombia	94	95	95	94
19	Dominican Republic	91	90	96	92
20	Venezuela, RB	87	88	98	91
21	Suriname	94	87	91	90
22	Haiti	9	68	8	28

3. Yearly change in urban population for Mexico and Brazil (two countries with generally high urban populations) from 1975 to 2000:

We will use the LAG() function in SQL, which allows us to access data from a previous row in the same result set without the use of a self-join, in order to calculate the difference between years.

```
q3_a = """
SELECT year,
  value AS urban_population,
  value - LAG(value) OVER (ORDER BY year) AS yearly_growth
FROM wdi_long_clean
WHERE country_name = 'Mexico'
  AND indicator = 'Urban population'
  AND year BETWEEN '1975' AND '2000'
ORDER BY year;
"""
q3_a_table = pd.read_sql(q3_a, conn)
q3_a_table
```

	year	urban_population	yearly_growth
0	1975	37016764	NaN
1	1976	38504442	1487678.0
2	1977	40011399	1506957.0
3	1978	41544154	1532755.0
4	1979	43095854	1551700.0
5	1980	44646369	1550515.0
6	1981	46068153	1421784.0
7	1982	47469200	1401047.0
8	1983	48882146	1412946.0
9	1984	50305880	1423734.0
10	1985	51742434	1436554.0
11	1986	53195618	1453184.0
12	1987	54666745	1471127.0
13	1988	56155065	1488320.0
14	1989	57649750	1494685.0
15	1990	59149337	1499587.0
16	1991	60634660	1485323.0
17	1992	62132096	1497436.0
18	1993	63634911	1502815.0
19	1994	65145831	1510920.0
20	1995	66663966	1518135.0
21	1996	68109906	1445940.0
22	1997	69508938	1399032.0
23	1998	70903451	1394513.0
24	1999	72293903	1390452.0
25	2000	73694985	1401082.0

```
q3_b = """
SELECT year,
  value AS urban_population,
  value - LAG(value) OVER (ORDER BY year) AS yearly_growth
FROM wdi_long_clean
WHERE country_name = 'Brazil'
  AND indicator = 'Urban population'
  AND year BETWEEN '1975' AND '2000'
ORDER BY year;
"""
q3_b_table = pd.read_sql(q3_b, conn)
q3_b_table
```

	year	urban_population	yearly_growth
0	1975	65420857	NaN

	year	urban_population	yearly_growth
1	1976	68051232	2630375.0
2	1977	70760392	2709160.0
3	1978	73551099	2790707.0
4	1979	76416004	2864905.0
5	1980	79352101	2936097.0
6	1981	82340685	2988584.0
7	1982	85371053	3030368.0
8	1983	88441554	3070501.0
9	1984	91547882	3106328.0
10	1985	94673905	3126023.0
11	1986	97807964	3134059.0
12	1987	100930955	3122991.0
13	1988	104046863	3115908.0
14	1989	107150809	3103946.0
15	1990	110249653	3098844.0
16	1991	113322847	3073194.0
17	1992	116391291	3068444.0
18	1993	119447677	3056386.0
19	1994	122479568	3031891.0
20	1995	125522590	3043022.0
21	1996	128573880	3051290.0
22	1997	131742708	3168828.0
23	1998	134957264	3214556.0
24	1999	138164576	3207312.0
25	2000	141288924	3124348.0

4. Average trade amounts (as % of GDP) across 1975-2024, first by region, and second by country:

```
q4_a = """
SELECT region,
       AVG(value) AS avg_trade_gdp
FROM wdi_long_clean
WHERE indicator = 'Trade (% of GDP)'
  AND CAST(year AS INTEGER) BETWEEN 1975 AND 2024
GROUP BY region
ORDER BY avg_trade_gdp DESC;
"""
q4_a_table = pd.read_sql(q4_a, conn)
q4_a_table
```

	region	avg_trade_gdp
0	Central America	75.001154
1	Caribbean	65.423034
2	South America	53.059413
3	Other	0.000000

```
q4_b = """
SELECT country_name, region,
       AVG(value) AS avg_trade_gdp
FROM wdi_long_clean
WHERE indicator = 'Trade (% of GDP)'
  AND CAST(year AS INTEGER) BETWEEN 1975 AND 2024
GROUP BY country_name, region
ORDER BY avg_trade_gdp DESC
LIMIT 24;
"""
q4_b_table = pd.read_sql(q4_b, conn)
q4_b_table
```

	country_name	region	avg_trade_gdp
0	Guyana	South America	172.706599
1	Panama	Central America	123.698223
2	Puerto Rico	Caribbean	104.956422
3	Suriname	South America	97.518818
4	Honduras	Central America	95.767372
5	Belize	Central America	89.613579
6	Nicaragua	Central America	74.432156
7	Costa Rica	Central America	74.183460
8	Paraguay	South America	71.780297
9	El Salvador	Central America	65.621117
10	Dominican Republic	Caribbean	61.151605
11	Chile	South America	58.341319
12	Bolivia	Central America	56.918690
13	Venezuela, RB	South America	50.808681
14	Cuba	Caribbean	49.878871
15	Guatemala	Central America	49.082497
16	Ecuador	South America	48.241917
17	Mexico	Central America	47.184352
18	Uruguay	South America	45.534761
19	Peru	South America	41.024383
20	Haiti	Caribbean	38.153141
21	Colombia	South America	34.294808
22	Argentina	South America	24.314856

	country_name	region	avg_trade_gdp
23	Brazil	South America	22.548464

4 Exploratory Data Analysis

```python
import pandas as pd
import numpy as np
from scipy.stats import linregress
import matplotlib.pyplot as plt
import seaborn as sns
```

```python
region_mapping = {
    'Argentina': 'South America',
    'Belize': 'Central America',
    'Costa Rica': 'Central America',
    'El Salvador': 'Central America',
    'Guatemala': 'Central America',
    'Honduras': 'Central America',
    'Nicaragua': 'Central America',
    'Panama': 'Central America',
    'Mexico': 'Central America',
    'Bolivia': 'South America',
    'Brazil': 'South America',
    'Chile': 'South America',
    'Colombia': 'South America',
    'Ecuador': 'South America',
    'Guyana': 'South America',
    'Paraguay': 'South America',
    'Peru': 'South America',
    'Suriname': 'South America',
    'Uruguay': 'South America',
    'Venezuela, RB': 'South America',
    'Cuba': 'Caribbean',
    'Dominican Republic': 'Caribbean',
    'Haiti': 'Caribbean',
    'Jamaica': 'Caribbean',
    'St. Martin (French part)': 'Caribbean',
    'Puerto Rico': 'Caribbean',
}

df['Region'] = df['Country Name'].map(region_mapping)
```

```python
life_expectancy_df = df[df['Series Name'] == 'Life expectancy at birth, total (years)'].copy
year_columns = [col for col in df.columns if '[YR' in col]
```

```
life_expectancy_df.loc[:, year_columns] = life_expectancy_df[year_columns].apply(pd.to_numer
life_expectancy_df['Average Life Expectancy'] = life_expectancy_df[year_columns].mean(axis=1

life_result = life_expectancy_df[['Region', 'Average Life Expectancy']]
life_result = life_result.dropna(subset=['Average Life Expectancy'])
life_result = life_result.groupby('Region')['Average Life Expectancy'].mean().reset_index()
life_result = life_result.sort_values(by='Average Life Expectancy', ascending=False)
print(life_result)
```

```
            Region  Average Life Expectancy
0        Caribbean                70.697967
2    South America                69.356007
1  Central America                68.988212
```

```
mortality_df = df[df['Series Name'] == 'Mortality rate, under-5 (per 1,000 live births)'].co
mortality_df.loc[:, year_columns] = mortality_df[year_columns].apply(pd.to_numeric, errors='
mortality_df['Average Mortality Under 5 yo'] = mortality_df[year_columns].mean(axis=1)

mort_result = mortality_df[['Region', 'Average Mortality Under 5 yo']]
mort_result = mort_result.dropna(subset=['Average Mortality Under 5 yo'])
mort_result = mort_result.groupby('Region')['Average Mortality Under 5 yo'].mean().reset_ind
mort_result = mort_result.sort_values(by='Average Mortality Under 5 yo', ascending=False)
print(mort_result)
```

```
            Region  Average Mortality Under 5 yo
0        Caribbean                     61.526531
1  Central America                     41.316837
2    South America                     40.174876
```

```
fertility_df = df[df['Series Name'] == 'Adolescent fertility rate (births per 1,000 women ag
fertility_df.loc[:, year_columns] = fertility_df[year_columns].apply(pd.to_numeric, errors='
fertility_df['Average Fertility Rate'] = fertility_df[year_columns].mean(axis=1)

fertility_result = fertility_df[['Region', 'Average Fertility Rate']]
fertility_result = fertility_result.dropna(subset=['Average Fertility Rate'])
fertility_result = fertility_result.groupby('Region')['Average Fertility Rate'].mean().reset
fertility_result = fertility_result.sort_values(by='Average Fertility Rate', ascending=False
print(fertility_result)
```

```
            Region  Average Fertility Rate
1  Central America              100.462635
2    South America               77.902779
0        Caribbean               65.945947
```

```
immunization_group = [
    'Immunization, DPT (% of children ages 12-23 months)',
    'Immunization, HepB3 (% of one-year-old children)',
    'Immunization, measles (% of children ages 12-23 months)'
]

immunization_df = df[df['Series Name'].isin(immunization_group)].copy()
immunization_df.loc[:, year_columns] = immunization_df[year_columns].apply(pd.to_numeric, er
immunization_df['Grouped Immunization Average'] = immunization_df[year_columns].mean(axis=1)

immune_result = immunization_df[['Region', 'Grouped Immunization Average']]
immune_result = immune_result.dropna(subset=['Grouped Immunization Average'])
immune_result = immune_result.groupby('Region')['Grouped Immunization Average'].mean().reset
immune_result = immune_result.sort_values(by='Grouped Immunization Average', ascending=False
print(immune_result)
```

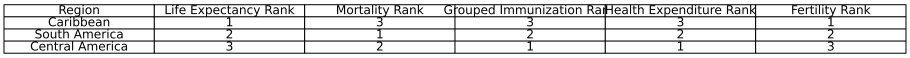
```
            Region Grouped Immunization Average
1  Central America                    84.240333
2    South America                    80.869707
0        Caribbean                     74.12531
```

```
health_df = df[df['Series Name'] == 'Current health expenditure (% of GDP)'].copy()
health_df.loc[:, year_columns] = health_df[year_columns].apply(pd.to_numeric, errors='coerce
health_df['Average Health Expenditure'] = health_df[year_columns].mean(axis=1)

health_result = health_df[['Region', 'Average Health Expenditure']]
health_result = health_result.dropna(subset=['Average Health Expenditure'])
health_result = health_result.groupby('Region')['Average Health Expenditure'].mean().reset_i
health_result = health_result.sort_values(by='Average Health Expenditure', ascending=False)
print(health_result)
```

```
            Region Average Health Expenditure
1  Central America                   6.785006
2    South America                   6.429862
0        Caribbean                   6.312227
```

```
ranking_merge = life_result.merge(mort_result, on='Region')
ranking_merge = ranking_merge.merge(fertility_result, on='Region')
ranking_merge = ranking_merge.merge(immune_result, on='Region')
ranking_merge = ranking_merge.merge(health_result, on='Region')

ranking_merge['Life Expectancy Rank'] = ranking_merge['Average Life Expectancy'].rank(ascend
ranking_merge['Mortality Rank'] = ranking_merge['Average Mortality Under 5 yo'].rank(ascendi
ranking_merge['Grouped Immunization Rank'] = ranking_merge['Grouped Immunization Average'].r
ranking_merge['Health Expenditure Rank'] = ranking_merge['Average Health Expenditure'].rank(
ranking_merge['Fertility Rank'] = ranking_merge['Average Fertility Rate'].rank(ascending=Tru

ranking_result = ranking_merge[['Region',
                            'Life Expectancy Rank',
                            'Mortality Rank',
                            'Grouped Immunization Rank',
                            'Health Expenditure Rank',
                            'Fertility Rank']]
```

```
fig, ax = plt.subplots(figsize=(16, 4))  #
ax.axis('off')

table = ax.table(
    cellText=ranking_result.values,
    colLabels=ranking_result.columns,
    cellLoc='center',
    loc='center'
)

table.auto_set_font_size(False)
table.set_fontsize(12)
plt.savefig('health_region_ranking.png', bbox_inches='tight', dpi=150)
```



Region	Life Expectancy Rank	Mortality Rank	Grouped Immunization Ra	Health Expenditure Rank	Fertility Rank
Caribbean	1	3	3	3	1
South America	2	1	2	2	2
Central America	3	2	1	1	3

```
# Define a function to find the highest and lowest values for a given metric
def analyze_metric(df, metric_name):
    metric_df = df[df['Series Name'] == metric_name].copy()
    year_columns = [col for col in df.columns if '[YR' in col]
    metric_df.loc[:, year_columns] = metric_df[year_columns].apply(pd.to_numeric, errors='co
    metric_df['Average Value'] = metric_df[year_columns].mean(axis=1)

    # Group by region and find the highest and lowest values
    grouped = metric_df.groupby('Region')
    highest = grouped.apply(lambda x: x.loc[x['Average Value'].idxmax(), ['Country Name', 'A
    lowest = grouped.apply(lambda x: x.loc[x['Average Value'].idxmin(), ['Country Name', 'Av

    return highest, lowest
```

```
# 1. Life expectancy at birth, total (years)
life_expectancy_highest, life_expectancy_lowest = analyze_metric(
    df, 'Life expectancy at birth, total (years)'
)
print("Life Expectancy (Highest):")
```

```python
print(life_expectancy_highest)
print("\nLife Expectancy (Lowest):")
print(life_expectancy_lowest)
```

```
Life Expectancy (Highest):
                             Country Name  Average Value
Region
Caribbean        St. Martin (French part)      76.319592
Central America                Costa Rica      76.953000
South America                       Chile      75.545469

Life Expectancy (Lowest):
                Country Name  Average Value
Region
Caribbean              Haiti      56.766245
Central America    Guatemala      64.522265
South America        Bolivia      60.270714
```

```python
# 2. Mortality rate, under-5 (per 1,000 live births)
mortality_highest, mortality_lowest = analyze_metric(
    df, 'Mortality rate, under-5 (per 1,000 live births)'
)
print("\nMortality Rate (Highest):")
print(mortality_highest)
print("\nMortality Rate (Lowest):")
print(mortality_lowest)
```

```

Mortality Rate (Highest):
                Country Name  Average Value
Region
Caribbean              Haiti     120.226531
Central America    Guatemala      66.069388
South America        Bolivia      90.569388

Mortality Rate (Lowest):
                Country Name  Average Value
Region
Caribbean               Cuba      11.965306
Central America   Costa Rica      16.157143
South America          Chile      17.112245
```

```python
# 3. Adolescent fertility rate (births per 1,000 women ages 15-19)
fertility_highest, fertility_lowest = analyze_metric(
    df, 'Adolescent fertility rate (births per 1,000 women ages 15-19)'
)
```

```
print("\nAdolescent Fertility Rate (Highest):")
print(fertility_highest)
print("\nAdolescent Fertility Rate (Lowest):")
print(fertility_lowest)
```

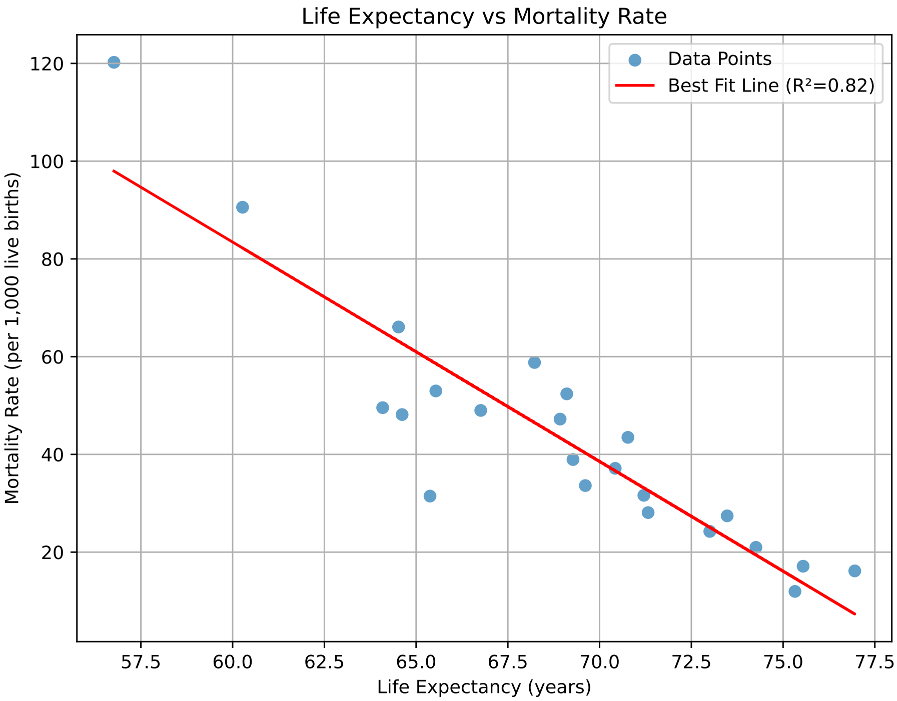
```
Adolescent Fertility Rate (Highest):
                       Country Name  Average Value
Region
Caribbean        Dominican Republic      99.143898
Central America           Nicaragua     126.473429
South America                Guyana      97.110224

Adolescent Fertility Rate (Lowest):
                             Country Name  Average Value
Region
Caribbean        St. Martin (French part)      35.587714
Central America                Costa Rica      75.293102
South America                       Chile      55.050367
```

```
# Filter the dataset for the three variables
life_expectancy = df[df['Series Name'] == 'Life expectancy at birth, total (years)'].copy()
mortality_rate = df[df['Series Name'] == 'Mortality rate, under-5 (per 1,000 live births)'].
fertility_rate = df[df['Series Name'] == 'Adolescent fertility rate (births per 1,000 women

# Extract year columns and calculate averages
year_columns = [col for col in map(str, df.columns) if '[YR' in col]

# Calculate the average values for each variable
life_expectancy['Average Value'] = life_expectancy[year_columns].apply(pd.to_numeric, errors
mortality_rate['Average Value'] = mortality_rate[year_columns].apply(pd.to_numeric, errors=
fertility_rate['Average Value'] = fertility_rate[year_columns].apply(pd.to_numeric, errors=

# Merge the datasets on 'Country Name' for scatterplots
merged_df = life_expectancy[['Country Name', 'Average Value']].merge(
    mortality_rate[['Country Name', 'Average Value']],
    on='Country Name',
    suffixes=('_LifeExpectancy', '_MortalityRate')
).merge(
    fertility_rate[['Country Name', 'Average Value']],
    on='Country Name'
)
merged_df.rename(columns={'Average Value': 'AdolescentFertilityRate'}, inplace=True)
```

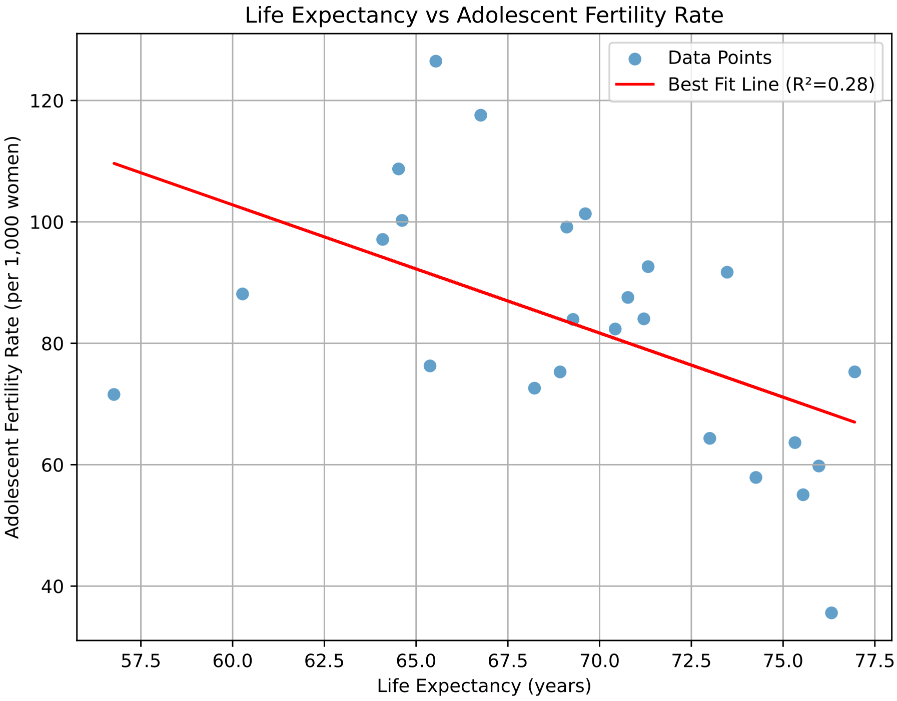
```
# Ensure no NaN values in the data for each scatterplot
valid_data1 = merged_df[['Average Value_LifeExpectancy', 'Average Value_MortalityRate']].dro
```

```
x1 = valid_data1['Average Value_LifeExpectancy']
y1 = valid_data1['Average Value_MortalityRate']

valid_data2 = merged_df[['Average Value_LifeExpectancy', 'AdolescentFertilityRate']].dropna(
x2 = valid_data2['Average Value_LifeExpectancy']
y2 = valid_data2['AdolescentFertilityRate']

valid_data3 = merged_df[['Average Value_MortalityRate', 'AdolescentFertilityRate']].dropna()
x3 = valid_data3['Average Value_MortalityRate']
y3 = valid_data3['AdolescentFertilityRate']
```

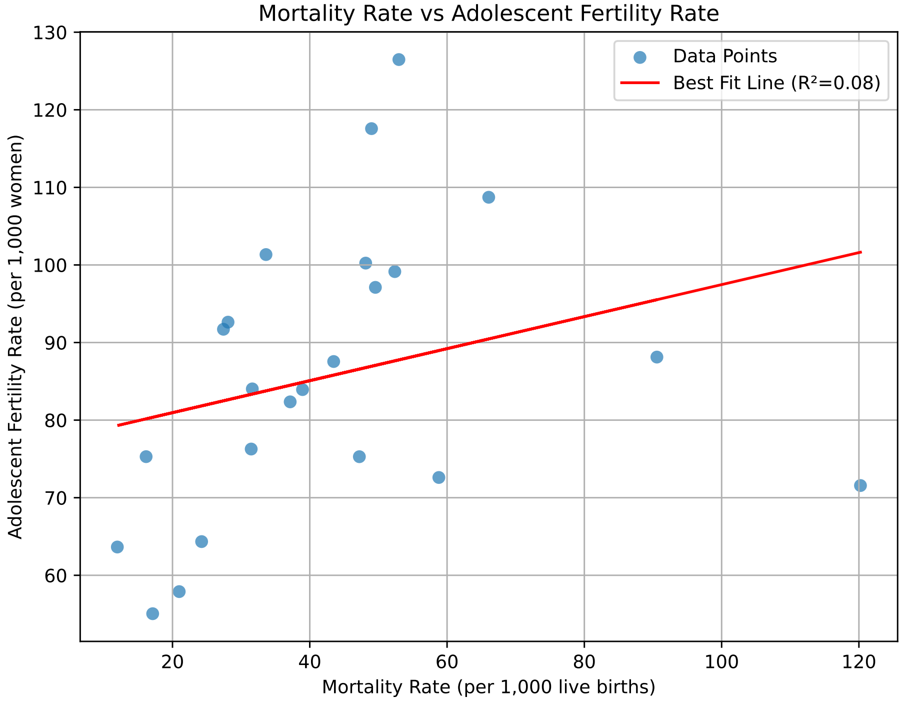
```
# Scatterplot 1: Life Expectancy vs Mortality Rate with regression line
slope1, intercept1, r_value1, p_value1, std_err1 = linregress(x1, y1)
plt.figure(figsize=(8, 6))
plt.scatter(x1, y1, alpha=0.7, label='Data Points')
plt.plot(x1, slope1 * x1 + intercept1, color='red', label=f'Best Fit Line (R²={r_value1**2:
plt.title('Life Expectancy vs Mortality Rate')
plt.xlabel('Life Expectancy (years)')
plt.ylabel('Mortality Rate (per 1,000 live births)')
plt.legend()
plt.grid(True)
plt.show()
```



```
# Scatterplot 2: Life Expectancy vs Adolescent Fertility Rate with regression line
x2 = merged_df['Average Value_LifeExpectancy']
y2 = merged_df['AdolescentFertilityRate']
slope2, intercept2, r_value2, p_value2, std_err2 = linregress(x2, y2)

plt.figure(figsize=(8, 6))
plt.scatter(x2, y2, alpha=0.7, label='Data Points')
plt.plot(x2, slope2 * x2 + intercept2, color='red', label=f'Best Fit Line (R²={r_value2**2:
plt.title('Life Expectancy vs Adolescent Fertility Rate')
plt.xlabel('Life Expectancy (years)')
plt.ylabel('Adolescent Fertility Rate (per 1,000 women)')
plt.legend()
plt.grid(True)
plt.show()
```



```
# Scatterplot 3: Mortality Rate vs Adolescent Fertility Rate with regression line
slope3, intercept3, r_value3, p_value3, std_err3 = linregress(x3, y3)
plt.figure(figsize=(8, 6))
plt.scatter(x3, y3, alpha=0.7, label='Data Points')
plt.plot(x3, slope3 * x3 + intercept3, color='red', label=f'Best Fit Line (R²={r_value3**2:
plt.title('Mortality Rate vs Adolescent Fertility Rate')
plt.xlabel('Mortality Rate (per 1,000 live births)')
plt.ylabel('Adolescent Fertility Rate (per 1,000 women)')
plt.legend()
plt.grid(True)
plt.show()
```



In order to do analysis on the data and comparing the countries within these regions to one another, need to modify the dataframes previously created. This will help in making visualizations easier.

```
#Life expectancy
life_expectancy_df = df[df['Series Name'] == 'Life expectancy at birth, total (years)'].copy
year_columns = [col for col in df.columns if '[YR' in col]
life_expectancy_df.loc[:, year_columns] = life_expectancy_df[year_columns].apply(pd.to_numer
life_result = life_expectancy_df.groupby(['Country Name', 'Region'])[year_columns].mean().re
life_result['Average Life Expectancy'] = life_result[year_columns].mean(axis=1)
life_result = life_result[['Region', 'Country Name', 'Average Life Expectancy']]

#Mortality Rate
mortality_df = df[df['Series Name'] == 'Mortality rate, under-5 (per 1,000 live births)'].co
mortality_df.loc[:, year_columns] = mortality_df[year_columns].apply(pd.to_numeric, errors='
mort_result = mortality_df.groupby(['Country Name', 'Region'])[year_columns].mean().reset_in
mort_result['Average Mortality Under 5 yo'] = mort_result[year_columns].mean(axis=1)
mort_result = mort_result[['Region', 'Country Name', 'Average Mortality Under 5 yo']]
```

```
#Fertility Rate
fertility_df = df[df['Series Name'] == 'Adolescent fertility rate (births per 1,000 women ag
fertility_df.loc[:, year_columns] = fertility_df[year_columns].apply(pd.to_numeric, errors=
fertility_result = fertility_df.groupby(['Country Name', 'Region'])[year_columns].mean().res
fertility_result['Average Fertility Rate'] = fertility_result[year_columns].mean(axis=1)
fertility_result = fertility_result[['Region', 'Country Name', 'Average Fertility Rate']]

#merge the dfs
combined = life_result.merge(mort_result, on=['Country Name', 'Region'])
combined = combined.merge(fertility_result, on=['Country Name', 'Region'])
```

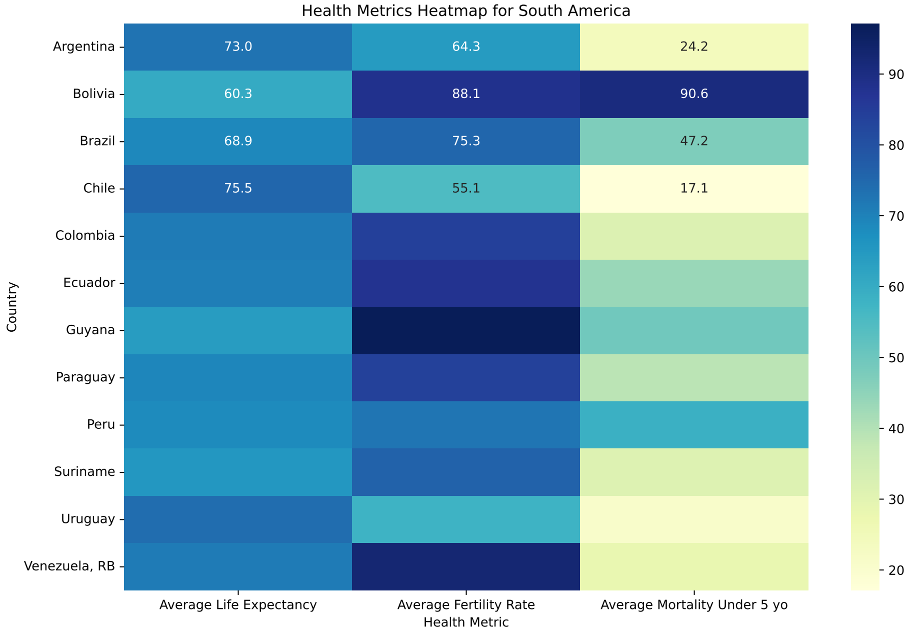
Now let's add two additional dataframes to work with to get better insight into health indicators:

```
immunization_group = [
    'Immunization, DPT (% of children ages 12-23 months)',
    'Immunization, HepB3 (% of one-year-old children)',
    'Immunization, measles (% of children ages 12-23 months)'
]
immunization_df = df[df['Series Name'].isin(immunization_group)].copy()
immunization_df.loc[:, year_columns] = immunization_df[year_columns].apply(pd.to_numeric, er
immune_result = immunization_df.groupby(['Country Name', 'Region'])[year_columns].mean().res
immune_result['Grouped Immunization Average'] = immune_result[year_columns].mean(axis=1)
immune_result = immune_result[['Region', 'Country Name', 'Grouped Immunization Average']]

health_df = df[df['Series Name'] == 'Current health expenditure (% of GDP)'].copy()
health_df.loc[:, year_columns] = health_df[year_columns].apply(pd.to_numeric, errors='coerce
health_result = health_df.groupby(['Country Name', 'Region'])[year_columns].mean().reset_inc
health_result['Average Health Expenditure'] = health_result[year_columns].mean(axis=1)
health_result = health_result[['Region', 'Country Name', 'Average Health Expenditure']]
```

```
south_america_df = combined[combined['Region'] == 'South America']

heatmap = south_america_df[['Country Name',
                            'Average Life Expectancy',
                            'Average Fertility Rate',
                            'Average Mortality Under 5 yo']]

heatmap = heatmap.set_index('Country Name')
heatmap = heatmap.apply(pd.to_numeric, errors='coerce')



plt.figure(figsize=(12, 8))
sns.heatmap(heatmap, annot=True, cmap="YlGnBu", fmt=".1f")
```

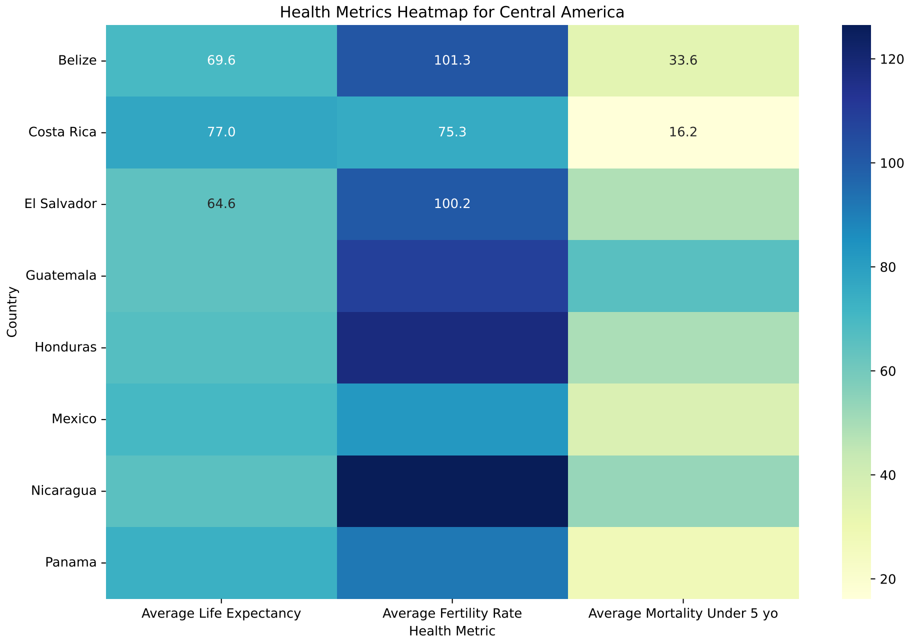
```
plt.title('Health Metrics Heatmap for South America')
plt.xlabel('Health Metric')
plt.ylabel('Country')
```

```
Text(364.16666666666663, 0.5, 'Country')
```



```
central_america_df = combined[combined['Region'] == 'Central America']

heatmap = central_america_df[['Country Name',
                              'Average Life Expectancy',
                              'Average Fertility Rate',
                              'Average Mortality Under 5 yo']]

heatmap = heatmap.set_index('Country Name')
heatmap = heatmap.apply(pd.to_numeric, errors='coerce')



plt.figure(figsize=(12, 8))
sns.heatmap(heatmap, annot=True, cmap="YlGnBu", fmt=".1f")
plt.title('Health Metrics Heatmap for Central America')
plt.xlabel('Health Metric')
plt.ylabel('Country')
```

```
Text(364.16666666666663, 0.5, 'Country')
```

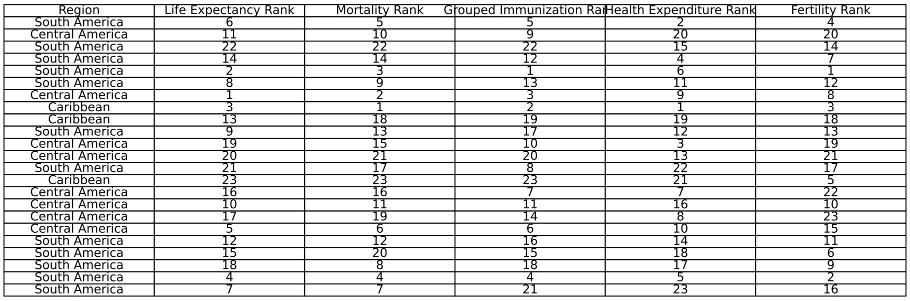


The heatmaps show the breakdown of each South and Central American country and their average for the following metrics. The darker the color indicates the higher the frequency for that certain attribute. This graph is able to display the countries in these regions that relate to mortality, life expectancy and fertility rates. Additionally, there is a table that ranks the Caribbean region along with the south and central American regions based on the five metrics listed in the table. By looking at this, we are able to see an overall ranking of which region has high mortality, low fertility, etc. This was further evaluated in the analysis by looking at each individual metric and comparing it with the regions.

```
# Dropping NA values in the dataframe
fertility_result = fertility_result.dropna(subset=['Average Fertility Rate'])
immune_result = immune_result.dropna(subset=['Grouped Immunization Average'])
life_result = life_result.dropna(subset=['Average Life Expectancy'])
mort_result = mort_result.dropna(subset=['Average Mortality Under 5 yo'])
health_result = health_result.dropna(subset=['Average Health Expenditure'])
```

```
ranking_merge = life_result.merge(mort_result, on=['Region','Country Name'])
ranking_merge = ranking_merge.merge(fertility_result, on=['Region','Country Name'])
ranking_merge = ranking_merge.merge(immune_result, on=['Region','Country Name'])
ranking_merge = ranking_merge.merge(health_result, on=['Region','Country Name'])

ranking_merge['Life Expectancy Rank'] = ranking_merge['Average Life Expectancy'].rank(ascend
ranking_merge['Mortality Rank'] = ranking_merge['Average Mortality Under 5 yo'].rank(ascendi
ranking_merge['Grouped Immunization Rank'] = ranking_merge['Grouped Immunization Average'].r
```

```
ranking_merge['Health Expenditure Rank'] = ranking_merge['Average Health Expenditure'].rank(
ranking_merge['Fertility Rank'] = ranking_merge['Average Fertility Rate'].rank(ascending=Tru

ranking_result = ranking_merge[['Region',
                               'Life Expectancy Rank',
                               'Mortality Rank',
                               'Grouped Immunization Rank',
                               'Health Expenditure Rank',
                               'Fertility Rank']]

fig, ax = plt.subplots(figsize=(16, 4))  #
ax.axis('off')

table = ax.table(
    cellText=ranking_result.values,
    colLabels=ranking_result.columns,
    cellLoc='center',
    loc='center'
)

table.auto_set_font_size(False)
table.set_fontsize(12)
plt.savefig('health_region_ranking.png', bbox_inches='tight', dpi=150)
```



Region	Life Expectancy Rank	Mortality Rank	Grouped Immunization Ra	Health Expenditure Rank	Fertility Rank
South America	6	5	5	2	4
Central America	11	10	9	20	20
South America	22	22	22	15	14
South America	14	14	12	4	7
South America	2	3	1	6	1
South America	8	9	13	11	12
Central America	1	2	3	9	8
Caribbean	3	1	2	1	3
Caribbean	13	18	19	19	18
South America	9	13	17	12	13
Central America	19	15	10	3	19
Central America	20	21	20	13	21
South America	21	17	8	22	17
Caribbean	23	23	23	21	5
Central America	16	16	7	7	22
Central America	10	11	11	16	10
Central America	17	19	14	8	23
Central America	5	6	6	10	15
South America	12	12	16	14	11
South America	15	20	15	18	6
South America	18	8	18	17	9
South America	4	4	4	5	2
South America	7	7	21	23	16

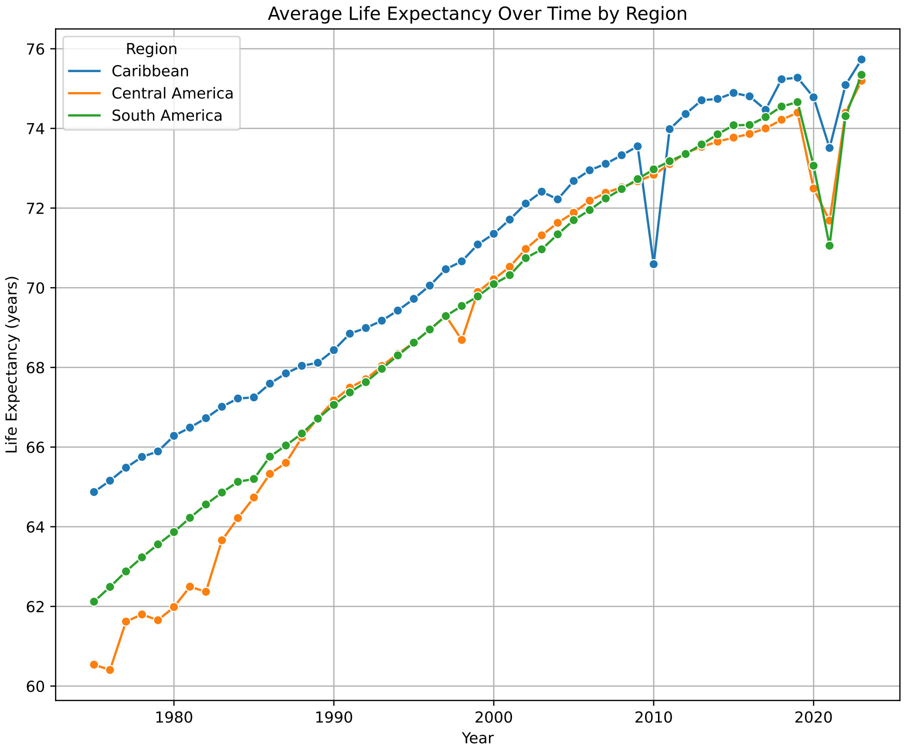
```
# Fix year columns
year_columns = [col for col in df.columns if '[YR' in col]

# Melt from wide to long format
df_long = df.melt(
    id_vars=['Country Name', 'Country Code', 'Region', 'Series Name', 'Series Code'],
    value_vars=year_columns,
    var_name='Year',
    value_name='Value'
)
```

```
# Clean Year (remove [YRxxxx] formatting)
df_long['Year'] = df_long['Year'].str.extract('(\d+)').astype(int)

# Clean Value: Turn '..' into NaN, ensure numeric
df_long['Value'] = pd.to_numeric(df_long['Value'], errors='coerce')

# Now your structure is:
# Country Name | Country Code | Region | Series Name | Series Code | Year | Value
print(df_long.head())
```

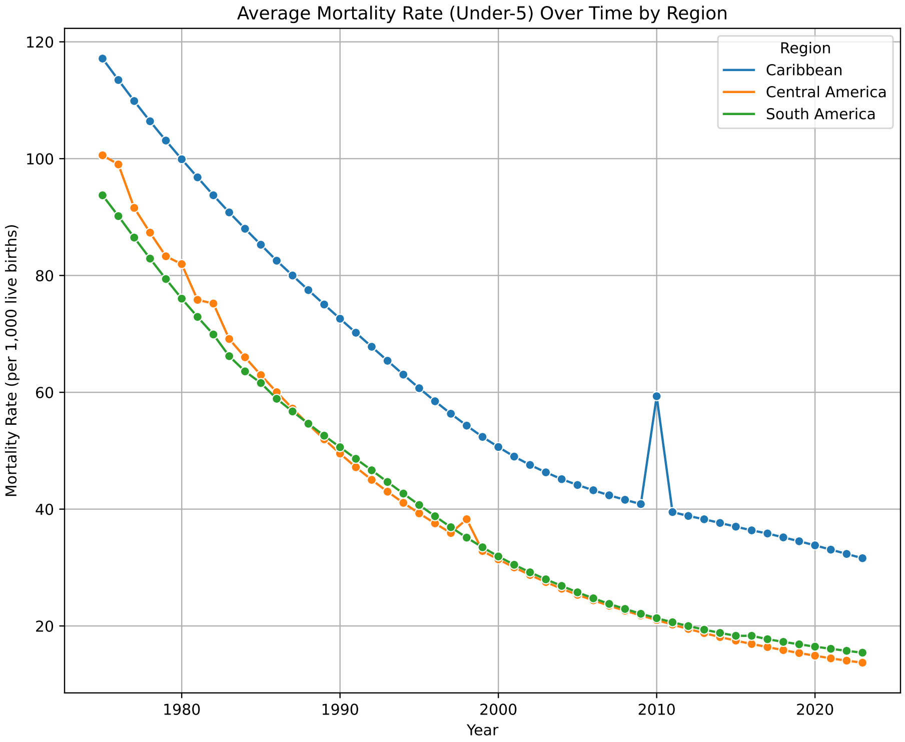
```
  Country Name Country Code         Region  \
0    Argentina          ARG  South America
1    Argentina          ARG  South America
2    Argentina          ARG  South America
3    Argentina          ARG  South America
4    Argentina          ARG  South America

                                         Series Name     Series Code  Year  \
0            Life expectancy at birth, total (years)  SP.DYN.LE00.IN  1975
1    Mortality rate, under-5 (per 1,000 live births)     SH.DYN.MORT  1975
2  Adolescent fertility rate (births per 1,000 wo...     SP.ADO.TFRT  1975
3  Prevalence of HIV, total (% of population ages...  SH.DYN.AIDS.ZS  1975
4  Immunization, DPT (% of children ages 12-23 mo...     SH.IMM.IDPT  1975

    Value
0  66.965
1  63.300
2  73.076
3     NaN
4     NaN
```

```
# Filter only Life Expectancy
life_exp = df_long[df_long['Series Name'] == 'Life expectancy at birth, total (years)'].copy

# Group by Region and Year, and take average across countries
region_life_exp = life_exp.groupby(['Region', 'Year'])['Value'].mean().reset_index()

# Plot
plt.figure(figsize=(10,8))
sns.lineplot(
    data=region_life_exp,
    x='Year',
    y='Value',
    hue='Region',
    marker='o'
)
```

```
plt.title('Average Life Expectancy Over Time by Region')
plt.xlabel('Year')
plt.ylabel('Life Expectancy (years)')
plt.grid(True)
plt.legend(title='Region')
plt.show()
```



```
# Filter only Mortality Rate
mortality = df_long[df_long['Series Name'] == 'Mortality rate, under-5 (per 1,000 live birth

# Group by Region and Year, and average across countries
region_mortality = mortality.groupby(['Region', 'Year'])['Value'].mean().reset_index()

# Plot
plt.figure(figsize=(10,8))
sns.lineplot(
    data=region_mortality,
    x='Year',
    y='Value',
    hue='Region',
    marker='o'
)

plt.title('Average Mortality Rate (Under-5) Over Time by Region')
plt.xlabel('Year')
plt.ylabel('Mortality Rate (per 1,000 live births)')
plt.grid(True)
plt.legend(title='Region')
plt.show()
```

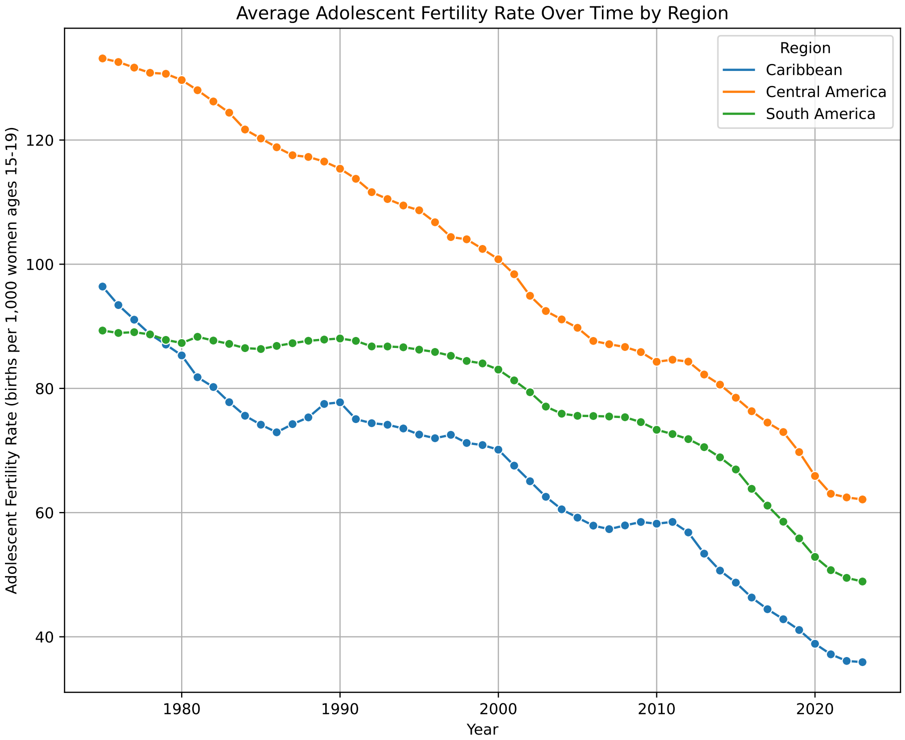


```
# Investigating sharp spikes for Caribbean region

# Step 1: Focus on Caribbean mortality
caribbean_mortality = mortality[mortality['Region'] == 'Caribbean'].copy()
caribbean_life_exp = life_exp[life_exp['Region'] == 'Caribbean'].copy()

# Step 2: Choose a suspicious year range (e.g., 2000-2024)
suspect_mort_years = caribbean_mortality[(caribbean_mortality['Year'] >= 2009) & (caribbean_
suspect_life_years = caribbean_life_exp[(caribbean_life_exp['Year'] >= 2009) & (caribbean_li
# Step 3: See country-level stats year by year
pivot_mort = suspect_mort_years.pivot_table(
    index='Year',
    columns='Country Name',
    values='Value'
)

pivot_life = suspect_life_years.pivot_table(
    index='Year',
    columns='Country Name',
    values='Value'
)

print(pivot_mort)
print(pivot_life)
```

```
Country Name  Cuba  Dominican Republic  Haiti
Year
2009           6.3                35.2   81.1
2010           6.2                35.1  136.7
2011           6.1                35.0   77.4
Country Name    Cuba  Dominican Republic   Haiti  Puerto Rico  \
Year
2009          77.584              71.587  61.694       78.377
2010          77.876              72.039  45.577       78.717
2011          78.051              72.496  61.581       78.766

Country Name  St. Martin (French part)
Year
2009                            78.513
2010                            78.752
2011                            79.016
```

It seems there was something that affected Haiti's life expectancy and mortality in 2010. After doing research, we found out that a catastrophic earthquake decimated Haiti, killing over 200,000 people.

```
# Filter only Adolescent Fertility Rate
adolescent_fertility = df_long[df_long['Series Name'] == 'Adolescent fertility rate (births

# Group by Region and Year, and average across countries
region_adolescent_fertility = adolescent_fertility.groupby(['Region', 'Year'])['Value'].mean

# Plot
plt.figure(figsize=(10,8))
sns.lineplot(
    data=region_adolescent_fertility,
    x='Year',
    y='Value',
    hue='Region',
    marker='o'
)

plt.title('Average Adolescent Fertility Rate Over Time by Region')
plt.xlabel('Year')
plt.ylabel('Adolescent Fertility Rate (births per 1,000 women ages 15-19)')
plt.grid(True)
plt.legend(title='Region')
plt.show()
```



5 Results and Discussion

For data analysis, we analyzed 5 certain metrics and compared the 3 regions to one another. The first was average life expectancy, where the Caribbean had the highest with ∼ 70 years and the lowest being Central America with ∼68 years old. This metric showed us that generally, these regions had fairly similar life expectancies. For Average mortality, that is where we begin to see a disparity between the regions. For example, the Caribbean exhibits a high mortality under 5 years old, around 61%, whereas Central and South America are both around 40%. For average fertility rates, Central America has the highest fertility rate, with the Caribbean being the lowest. Now, I looked at group immunization averages, which included DPT, measles, and HEPB3. It was mentioned that Central America had the highest, with Caribbean being the lowest. This could be due to the fact that Central America does have a higher fertility rate, and these immunizations were the % of children around the ages of 1-2. Finally, the last metric observed was average health expenditure. The region that utilizes on average the most for health expenditure is Central America, with again the Caribbean being the lowest.

6 Conclusion

Ultimately, we found

Summarize: - Main findings across indicators - Regional differences - Potential social or policy implications

7 References

World Bank's website: (https://databank.worldbank.org/source/world-development-indicators)

World Bank's API documnetation: (https://github.com/tgherzog/wbgapi)