**Question 1 – Bash**

a) **Using "echo", print your student number to the terminal. Additionally, print out your username from the operating system. Take a screenshot and include it in your submission**

```
joshlegrice@Joshs-MacBook-Pro-2 ~ % echo "Student Number: 720017170"
echo "Username: $(whoami)"
Student Number: 720017170
Username: joshlegrice
```

b) **Move inside the directory DATE_FILES, which was provided along with unit 2 of the course, which you should store somewhere on your computer.**

```
joshlegrice@Joshs-MacBook-Pro-2 ~ % cd /Users/joshlegrice/Desktop/University/3rd\ Year/Data\ Science\ in\ Economics/DATE_FILES
joshlegrice@Joshs-MacBook-Pro-2 DATE_FILES %
```

# cd = changes the current working directory to the file path given

c) **Count the number of files in this directory.**

# ls -1 = Lists all files in the directory, one per line
# | allows the output of the command before to be used as the input to the next command
# wc -l = Counts the number of lines in the output of ls -1
# $ = allow the code inside the brackets to be executed and not just echoed, like Python f string

```
joshlegrice@Joshs-MacBook-Pro-2 DATE_FILES % echo "Number of files: $(ls -1 | wc -l)"

Number of files:     3289
```

d) **Print the names of the first 8 files in this directory, along with information about their ownership, date, and size.**

```
joshlegrice@Joshs-MacBook-Pro-2 DATE_FILES % ls -lh | head -n 8

total 26312
-rw-rw-r--@ 1 joshlegrice  staff   110B 25 Jan  2024 2015_01_01.txt
-rw-rw-r--@ 1 joshlegrice  staff   110B 25 Jan  2024 2015_01_02.txt
-rw-rw-r--@ 1 joshlegrice  staff   110B 25 Jan  2024 2015_01_03.txt
-rw-rw-r--@ 1 joshlegrice  staff   110B 25 Jan  2024 2015_01_04.txt
-rw-rw-r--@ 1 joshlegrice  staff   110B 25 Jan  2024 2015_01_05.txt
-rw-rw-r--@ 1 joshlegrice  staff   110B 25 Jan  2024 2015_01_06.txt
-rw-rw-r--@ 1 joshlegrice  staff   110B 25 Jan  2024 2015_01_07.txt
```

# ls -lh = lists the contents of the directory in long (l) format and human-readable (h).
# | allows the output of the command before to be used as the input to the next command
# head -n 8 = only displays the first 8 lines of the output

e) **Move to the parent directory of this folder**

```
joshlegrice@Joshs-MacBook-Pro-2 DATE_FILES % cd ..
```

# cd = change directory to the file path given
# .. = indicates the parent directory of the current file

f) **Create a new directory there, named second_10_days**

```
joshlegrice@Joshs-MacBook-Pro-2 Data Science in Economics % mkdir -p second_10_days
```

# mkdir = command to make a new directory
# -p = ensures parent directories exist
# second_10_days = the name of the new directory

g) **Copy from the DATE_FILES directory the files that are related to the days 10-19 of every month to the newly created directory.**

```
joshlegrice@Joshs-MacBook-Pro-2 Data Science in Economics % ls DATE_FILES | awk '/_[1][0-9]/ {print "DATE_FILES/"$0}' | xargs -I {} cp {} second_10_days/
```

# ls DATE_FILES = prints the contents of the directory DATE_FILES
# awk '/_[1] [0-9]/ {print "DATE_FILES/" $0}'
# /_[1] [0-9]/ = Regular expression to find dates that have _10 to _19
# {print "DATE_FILES/" $0} = prints out the entire file path of the selected files
# xargs -I {} cp {} second_10_days/
# xargs = processes each file one by one
# -I {} = allows {} to be replaced with filename
# cp {} second_10_days/ = copies selected file to second_10_days

**h) Move inside second_10_days directory, and append the line "This is the last Line" to the end of file 2015_04_10**

```
joshlegrice@Joshs-MacBook-Pro-2 second_10_days % cd /Users/joshlegrice/Desktop/University/3rd\ Year/Data\ Science\ in\ Economics/second_10_days
joshlegrice@Joshs-MacBook-Pro-2 second_10_days % echo "This is the last Line" >> 2015_04_10.txt
```

2015_04_10.txt

```
sePuv9QS1U
QRrXzs995A
HqoSmqiVmi
NrAC2vqf2S
GppjXcJIb5
M3Plik7eoY
mv8HtVGtjM
zK36mPwoQh
8nM93uPsty
QW0MTRfGOA
This is the last Line
```

# cd ...... = moves to the file path shown
# echo "This is the last Line" = outputs the text This is the last Line
# >> 2015_04_10.txt = appends the echoed text into the file 2015_04_10.txt

**i) Write a one-line command to append the line "This is the last Line of X", where X is the name of the file, to the end of every file in the directory second_10_days**

```
joshlegrice@Joshs-MacBook-Pro-2 second_10_days % find . -type f -exec bash -c 'echo "This is the last Line of $(basename "$1")" >> "$1"' _ {} \;
```

2015_03_15.txt

```
xqavaX3XPy
PAQyuuqjX5
gVOowRWr8r
kGBzvUzPfr
iJWoc8oUjt
1CiPXGpwRY
EH9gyFZqGH
GzwoPHxD6u
GPcGxn43Dm
AhccFYvQHV
This is the last Line of 2015_03_15.txt
```

# find . = recursively searches the current directory
# -type f = only searches files not directories
# -exec bash – c = runs a bash command for each file found
# echo "This is the last Line of $(basename "$1") = outputs 'This is the last Line of (only the name of the file)'
# >> "$1"' _ {} \; = appends the output of the echo into the file.

**j) Using Bash: create a bash file Q1.sh. Write your code from (i) to it. Run the file Q1.sh including a screenshot showing how this runs on your system. Please explain any steps needed to run this file.**

Had to place the .sh file into another directory as if placed in the second_10_days directory, it would write into the Q1.sh file with ' this is the last Line of Q1.sh'

First step = Open an .sh file

```
joshlegrice@Joshs-MacBook-Pro-2 ~ % nano Q1.sh
```

Second step = Write code in .sh file

```
joshlegrice — nano Q1.sh — 141x11
  UW PICO 5.09                                File: Q1.sh

#!/bin/bash
cd /Users/joshlegrice/Desktop/University/3rd\ Year/Data\ Science\ in\ Economics/second_10_days/
find . -type f -exec bash -c 'echo "This is the last line of $(basename "$1") " >> "$1"'  _ {} \;

^G Get Help    ^O WriteOut    ^R Read File    ^Y Prev Pg    ^K Cut Text    ^C Cur Pos
^X Exit        ^J Justify     ^W Where is     ^V Next Pg    ^U UnCut Text  ^T To Spell
```

Final Step = Run .sh file

```
Last login: Thu Feb 20 11:29:33 on console
joshlegrice@Joshs-MacBook-Pro-2 ~ % ./Q1.sh
joshlegrice@Joshs-MacBook-Pro-2 ~ %
```

Output in an example file – I ran it a lot to make sure it worked

```
2023_12_27.txt
TcyEMjZYKt
PArpAcEa9e
48QDGRKlMF
VZ5nV4MpOk
3mmblBfsSc
5Gkva3CkuS
9Am8wWncYK
AAVg6AHyCe
ijGMJJdgY9
cF3KQJQ11g
This is the last Line of 2023_12_27.txt
This is the last Line of 2023_12_27.txt
This is the last Line of 2023_12_27.txt
This is the last line of 2023_12_27.txt
This is the last line of 2023_12_27.txt
```

**Question 2 – SQL**

a) **Create a new database in SQLite named Q2.db**

```
joshlegrice@Joshs-MacBook-Pro-2 Assignment % sqlite3 Q2.db
SQLite version 3.43.2 2023-10-10 13:08:14
Enter ".help" for usage hints.
sqlite>
```

b) **Create two tables named US_Code and US_Pop with column headings that match these two data frames**

```
CREATE TABLE US_Code (
    CountryCode VARCHAR(5),
    ZipCode VARCHAR(10) PRIMARY KEY,
    City VARCHAR(100),
    StateFull VARCHAR(50),
    State2 VARCHAR(5),
    CountyFull VARCHAR(100),
    FIPSCountyCode VARCHAR(10),
    MunicipalityFull VARCHAR(100),
    MunicipalityCode VARCHAR(10),
    Latitude REAL,
    Longitude REAL,
    Accuracy INTEGER
);
```

```
CREATE TABLE US_Pop (
    ID INTEGER PRIMARY KEY AUTOINCREMENT,
    Geo_ID VARCHAR(20),
    Zip VARCHAR(10),
    Gender VARCHAR(10),
    AgeRange VARCHAR(20),
    Population INTEGER,
    FOREIGN KEY (Zip) REFERENCES US_Code(ZipCode)
);
```

c) **Insert the data from the two files into the two tables. Make sure you don't insert the column heading from the file US_population.csv. Explain how you did this.**

```
sqlite> .mode tabs
sqlite> .import US_codes.txt US_Code
US_codes.txt:41098: INSERT failed: UNIQUE constraint failed: US_Code.ZipCode
US_codes.txt:41439: INSERT failed: UNIQUE constraint failed: US_Code.ZipCode
US_codes.txt:41440: INSERT failed: UNIQUE constraint failed: US_Code.ZipCode
```

# Had to remove duplicates before loading in the US_Code data due to the above error

```
joshlegrice@Joshs-MacBook-Pro-2 Assignment % awk -F'\t' '{print $2}' US_codes.txt | sort | uniq -d

09464
96860
96863
joshlegrice@Joshs-MacBook-Pro-2 Assignment % awk -F'\t' '!seen[$2]++' US_codes.txt > US_codes_cleaned.txt
```

# awk -F'\t' '{print $2}' US_codes.txt = Extracts the second column from the .txt file which is ZipCode
# sort = sorts the values within the column
# uniq -d = identifies and prints only the duplicate values = Used to visualise all duplicate values

# awk -F'\t' '!seen[$2]++' US_codes.txt = collects all the non-duplicates in the column into an array
# > US_codes_cleaned.txt = saves the contents of the previous output into a new file

```
sqlite> .mode tabs
sqlite> .import US_codes_cleaned.txt US_Code
sqlite> select * from US_Code Limit 5;
```

```
sqlite> .mode box
sqlite> Select * from US_Code limit 5;
```

| CountryCode | ZipCode | City | StateFull | State2 | CountyFull | FIPSCountyCode | MunicipalityFull | MunicipalityCode | Latitude | Longitude | Accuracy |
|---|---|---|---|---|---|---|---|---|---|---|---|
| US | 99553 | Akutan | Alaska | AK | Aleutians East | 013 | | | 54.143 | -165.7854 | 1 |
| US | 99571 | Cold Bay | Alaska | AK | Aleutians East | 013 | | | 55.1858 | -162.7211 | 1 |
| US | 99583 | False Pass | Alaska | AK | Aleutians East | 013 | | | 54.841 | -163.4368 | 1 |
| US | 99612 | King Cove | Alaska | AK | Aleutians East | 013 | | | 55.0628 | -162.3056 | 1 |
| US | 99661 | Sand Point | Alaska | AK | Aleutians East | 013 | | | 55.3192 | -160.4914 | 1 |

# .mode tabs to set the delimiter to tabs to distinguish columns

# Remove headers from the US_populations.csv

```
joshlegrice@Joshs-MacBook-Pro-2 Assignment % tail -n +2 US_population.csv > US_population_cleaned.csv
joshlegrice@Joshs-MacBook-Pro-2 Assignment %
```

# tail -n +2 = starts at line 2 and collects all rows

# > US_population_cleaned.csv = moves the new data into the new file

# I had trouble with importing the data straight into the US_Pop table due to this error

```
sqlite> .mode csv
sqlite> .import US_Pop_Clean.csv US_Pop
```

```
US_Pop_Clean.csv:125718: expected 6 columns but found 5 - filling the rest with NULL
US_Pop_Clean.csv:125718: INSERT failed: datatype mismatch
US_Pop_Clean.csv:125719: expected 6 columns but found 5 - filling the rest with NULL
US_Pop_Clean.csv:125719: INSERT failed: datatype mismatch
```

# So, I imported the data into a temporary table and then copied the data into US_Pop

```
CREATE TABLE temp_US_Pop (
    Geo_ID VARCHAR(20),
    Zip VARCHAR(10),
    Gender VARCHAR(10),
    AgeRange VARCHAR(20),
    Population INTEGER
);
```

```
sqlite> .mode csv
sqlite> .import US_population_cleaned.csv temp_US_Pop
sqlite> .mode box
sqlite> Select * from temp_US_Pop limit 5;
```

| Geo_ID | Zip | Gender | AgeRange | Population |
|---|---|---|---|---|
| 8600000US61747 | 61747 | female | 30--34 | 50 |
| 8600000US64120 | 64120 | male | 85-- | 5 |
| 8600000US95117 | 95117 | male | 30--34 | 1389 |
| 8600000US74074 | 74074 | female | 60--61 | 231 |
| 8600000US58042 | 58042 | female | 0--4 | 56 |

# Inserting data from temp table to US_Pop

```
sqlite> INSERT INTO US_Pop (Geo_ID, Zip, Gender, AgeRange, Population)
   ...> SELECT Geo_ID, Zip, Gender, AgeRange, Population FROM temp_US_Pop;
sqlite> Select * from US_Pop Limit 5;
```

| ID | Geo_ID | Zip | Gender | AgeRange | Population |
|---|---|---|---|---|---|
| 1 | 8600000US61747 | 61747 | female | 30--34 | 50 |
| 2 | 8600000US64120 | 64120 | male | 85-- | 5 |
| 3 | 8600000US95117 | 95117 | male | 30--34 | 1389 |
| 4 | 8600000US74074 | 74074 | female | 60--61 | 231 |
| 5 | 8600000US58042 | 58042 | female | 0--4 | 56 |

**d) Write an SQL query to print the total population per gender (using the US_Pop table only)**

```
sqlite> SELECT Gender, SUM(Population) as Total_Population
   ...> FROM US_Pop
   ...> GROUP BY Gender;
```

| Gender | Total_Population |
|---|---|
| female | 378160746 |
| male | 365004493 |

e) **Write an SQL query to print the total population per gender but join the two tables. If you see any difference in your results between this question and part (d), explain why this occurs.**

```
sqlite> SELECT Gender, SUM(Population) AS Total_Population
   ...> FROM US_Pop
   ...> INNER JOIN US_Code ON US_Code.ZipCode = US_Pop.Zip
   ...> GROUP BY Gender;
```

| Gender | Total_Population |
|---|---|
| female | 345258967 |
| male | 333951977 |

The difference is because INNER JOIN only includes records where zip codes exist in both US_Pop and US_Code, excluding unmatched zip codes from US_Pop. This results in a lower total population in part (e) compared to part (d).

f) **Write an SQL query to print the total population per age group (use the US_Pop table only).**

```
sqlite> SELECT AgeRange, SUM(Population) AS Total_Population
   ...> FROM US_Pop
   ...> GROUP BY AgeRange;
```

| AgeRange | Total_Population |
|---|---|
| 0--4 | 48634771 |
| 10--14 | 49651810 |
| 15--17 | 31218634 |
| 18--19 | 21963734 |
| 20--20 | 10881126 |
| 21--21 | 10489720 |
| 22--24 | 30562377 |
| 25--29 | 50678918 |
| 30--34 | 47975723 |
| 35--39 | 48599887 |
| 40--44 | 50422779 |
| 45--49 | 54820056 |
| 5--9 | 48857962 |
| 50--54 | 53572750 |
| 55--59 | 47232435 |
| 60--61 | 17228473 |
| 62--64 | 23364196 |
| 65--66 | 12840592 |
| 67--69 | 17217450 |
| 70--74 | 22319761 |
| 75--79 | 17642946 |
| 80--84 | 13798224 |
| 85-- | 13190915 |

**g) Write an SQL query to print the Top 10 largest states (full name) in terms of population size**

```
sqlite> SELECT c.StateFull, SUM(p.Population) AS Total_Population
   ...> FROM US_Pop p
   ...> JOIN US_Code c ON p.Zip = c.ZipCode
   ...> GROUP BY c.StateFull
   ...> ORDER BY Total_Population DESC
   ...> LIMIT 10;
```

| StateFull | Total_Population |
|---|---|
| California | 88526840 |
| Texas | 59743982 |
| New York | 46247865 |
| Florida | 44945558 |
| Illinois | 30596527 |
| Pennsylvania | 30255696 |
| Ohio | 27462317 |
| Michigan | 23490804 |
| Georgia | 23123141 |
| North Carolina | 22670061 |

**h) Write an SQL query to print the number of existing counties (not countries) in the database**

```
sqlite> SELECT COUNT(DISTINCT CountyFull) AS Total_Counties
   ...> FROM US_Code;
```

| Total_Counties |
|---|
| 1853 |

**i) Write an SQL query to print the total population per gender and age group for any counties containing "Middlesex" in their name.**

```
sqlite> SELECT p.Gender, p.AgeRange, SUM(p.Population) AS Total_Population
   ...> FROM US_Pop p
   ...> JOIN US_Code c ON p.Zip = c.ZipCode
   ...> WHERE c.CountyFull LIKE '%Middlesex%'
   ...> GROUP BY p.Gender, p.AgeRange
   ...> ORDER BY p.Gender, p.AgeRange;
```

| Gender | AgeRange | Total_Population |
|---|---|---|
| female | 0--4 | 388 |
| female | 10--14 | 734 |
| female | 15--17 | 496 |
| female | 18--19 | 251 |
| female | 20--20 | 173 |
| female | 21--21 | 87 |
| female | 22--24 | 343 |
| female | 25--29 | 532 |
| female | 30--34 | 419 |
| female | 35--39 | 695 |
| female | 40--44 | 1021 |
| female | 45--49 | 1365 |
| female | 5--9 | 738 |
| female | 50--54 | 1610 |
| female | 55--59 | 1049 |
| female | 60--61 | 406 |
| female | 62--64 | 863 |
| female | 65--66 | 639 |
| female | 67--69 | 713 |
| female | 70--74 | 873 |
| female | 75--79 | 680 |
| female | 80--84 | 492 |
| female | 85-- | 684 |
| male | 0--4 | 532 |
| male | 10--14 | 712 |
| male | 15--17 | 518 |
| male | 18--19 | 275 |
| male | 20--20 | 184 |
| male | 21--21 | 147 |
| male | 22--24 | 477 |
| male | 25--29 | 405 |
| male | 30--34 | 501 |
| male | 35--39 | 763 |
| male | 40--44 | 976 |
| male | 45--49 | 1261 |
| male | 5--9 | 716 |
| male | 50--54 | 1541 |
| male | 55--59 | 1253 |
| male | 60--61 | 455 |
| male | 62--64 | 801 |
| male | 65--66 | 580 |
| male | 67--69 | 777 |
| male | 70--74 | 756 |
| male | 75--79 | 551 |
| male | 80--84 | 477 |
| male | 85-- | 244 |

# Question 3 + 4

720017170

**Question 3 - See QMD for Code**

**(a) Import each dataset into memory as a separate data frame, keeping all countries as your sample.**

```
plt.rcParams.update({'font.size': 14})

# Loading in the Data
Health_Data = pd.read_csv('Health.csv', index_col=None)
Infant_Data = pd.read_csv("Infant.csv",index_col=None)

# Replace .. with NA
Health_Data.replace("..", pd.NA, inplace=True)
Infant_Data.replace("..", pd.NA, inplace=True)

# Removing unnecessary columns
Health_Data = Health_Data.drop(columns=['Series Name', 'Series Code'])
Infant_Data = Infant_Data.drop(columns=['Series Name', 'Series Code'])

# Remove names in []
Health_Data.columns = Health_Data.columns.str.replace(r'\[.*\]', '', regex=True)
Infant_Data.columns = Infant_Data.columns.str.replace(r'\[.*\]', '', regex=True)

print(Infant_Data.head())
print(Health_Data.head())
```

```
     Country Name Country Code 2000  2001  2002  2003  2004  2005  2006   \
0     Afghanistan          AFG    92  89.3  86.6  83.7  80.9    78  75.1
1         Albania          ALB    24  22.9  21.6  20.4  19.1  17.8  16.5
2         Algeria          DZA  35.6  34.3    33  31.6  30.3    29  27.8
3  American Samoa          ASM  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
```

```
4         Andorra          AND   6.5   6.3     6   5.8   5.6   5.3   5.1

  2007   ... 2014  2015  2016  2017  2018  2019  2020  2021  2022  2023
0  72.3  ...  56.2  54.6    53  51.5  50.1  48.8  47.4  46.1  44.8  <NA>
1  15.3  ...   8.8   8.5   8.4   8.3   8.3   8.3   8.4   8.4   8.4  <NA>
2  26.6  ...    22  21.7  21.4    21  20.6  20.1  19.7  19.2  18.7  <NA>
3  <NA>  ...  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>  <NA>
4   4.9  ...   3.5   3.4   3.2   3.1     3   2.9   2.8   2.7   2.6  <NA>

[5 rows x 26 columns]
     Country Name Country Code          2000           2001         2002   \
0     Afghanistan          AFG          <NA>           <NA>  17.00758553
1         Albania          ALB    65.1501236    73.78884125  78.99478149
2         Algeria          DZA   62.11769485    67.33850098  66.94760132
3  American Samoa          ASM          <NA>           <NA>         <NA>
4         Andorra          AND  1287.00280762  1336.21142578  1486.171875

          2003         2004          2005          2006          2007   \
0   17.81492424  21.42946434   25.10707283   28.91982269   32.71720505
1  106.29218292 138.11340332  152.12762451  166.81382751  212.61096191
2   76.23547363  93.02433014  101.30373383  117.43313599  151.77920532
3          <NA>         <NA>          <NA>          <NA>          <NA>
4 1772.71337891 1990.0748291 2214.64697266 2139.27539063 2489.43115234

   ...          2014          2015          2016          2017   \
0  ...   60.18957901   60.05854034   61.48645782   66.90921783
1  ...  295.12359619  255.35635376  277.04321289   297.4619751
2  ...  361.15942383    292.275177  261.40023804  265.83843994
3  ...          <NA>          <NA>          <NA>          <NA>
4  ... 3089.84301758 2688.20629883 2755.44848633 2873.29614258

          2018          2019          2020          2021  2022  2023
0   71.33430481   74.23410797   80.28805542   81.31976318  <NA>  <NA>
1   351.3012085  367.75839233  396.88024902  464.74285889  <NA>  <NA>
2  266.46469116  235.99041748  206.03512573  204.56661987  <NA>  <NA>
3          <NA>          <NA>          <NA>          <NA>  <NA>  <NA>
4 3164.38842773 3026.59741211 3269.29736328 3505.99145508  <NA>  <NA>

[5 rows x 26 columns]
```

**(b) If data are not already stored in this way, please reshape data so that they consist of a single line of data for each country and year.**

```
# Pivoting the Data into a long format
Health_Data_long = pd.melt(Health_Data,
        id_vars=['Country Name', 'Country Code'],
        var_name='Year',
        value_name='Heathcare Expenditure (USD)')
Infant_Data_long = pd.melt(Infant_Data,
        id_vars=['Country Name', 'Country Code'],
        var_name='Year',
        value_name='Infant Mortality Rates (per 1,000 live births)')
```

| | Country Name | Country Code | Year | Heathcare Expenditure (USD) |
|---|---|---|---|---|
| 0 | Afghanistan | AFG | 2000 | <NA> |
| 1 | Albania | ALB | 2000 | 65.1501236 |
| 2 | Algeria | DZA | 2000 | 62.11769485 |
| 3 | American Samoa | ASM | 2000 | <NA> |
| 4 | Andorra | AND | 2000 | 1287.00280762 |

| | Country Name | Country Code | Year | Infant Mortality Rates (per 1,000 live births) |
|---|---|---|---|---|
| 0 | Afghanistan | AFG | 2000 | 92 |
| 1 | Albania | ALB | 2000 | 24 |
| 2 | Algeria | DZA | 2000 | 35.6 |
| 3 | American Samoa | ASM | 2000 | <NA> |
| 4 | Andorra | AND | 2000 | 6.5 |

**(c) Calculate the total number of countries observed in each data frame Calculate the total number of years observed in each data frame.**

```
# Counts the number of unique contries
num_countries_FDI = Health_Data_long["Country Name"].nunique()

# Outputs the number of unique countries using an f string
print(f"Total number of unique countries in Health_Data: {num_countries_FDI}")

num_years_FDI = Health_Data_long["Year"].nunique() # Counts the number of unique years
print(f"Total number of unique years observed in Health_Data: {num_years_FDI}")
```

```
num_countries_GDP = Infant_Data_long["Country Name"].nunique()
print(f"Total number of unique countries in Infant_Data: {num_countries_GDP}")

num_years_GDP = Infant_Data_long["Year"].nunique()
print(f"Total number of unique years observed in Infant_Data: {num_years_GDP}")
```

```
Total number of unique countries in Health_Data: 217
Total number of unique years observed in Health_Data: 24
Total number of unique countries in Infant_Data: 217
Total number of unique years observed in Infant_Data: 24
```

**(d) Calculate the number of observations for which data is missing**

```
# Sums the number of missing values in each dataset
missing_values_Health = Health_Data_long.isna().sum().sum()
print(f"Total missing observations in Health_Data: {missing_values_Health}")

missing_values_Infant = Infant_Data_long.isna().sum().sum()
print(f"Total missing observations in Infant_Data: {missing_values_Infant}")

# Create a dataframe showing the number of missing values for each dataset
missing_values_table = pd.DataFrame({
    'Dataset': ['Health_Data', 'Infant_Data'],
    'Total Missing Observations': [missing_values_Health, missing_values_Infant]
})
```

```
Total missing observations in Health_Data: 1099
Total missing observations in Infant_Data: 700
```

**(e) Join the two files by country and year so that you have single dataframe containing both variables. Explain clearly what type of join this is, and carefully check that the number of observations resulting from the join makes sense.**

```
# Merge the data on Country Name, Country code and Year
merged_data = pd.merge(Health_Data_long, Infant_Data_long, on=['Country Name',
'Country Code', 'Year'])
print(merged_data.head())

# Print the number of rows in the DataFrame
num_rows = merged_data.shape[0]
print(f"Number of rows in the DataFrame: {num_rows}")
```

```
     Country Name Country Code   Year Heathcare Expenditure (USD)  \
0     Afghanistan          AFG  2000                         <NA>
1         Albania          ALB  2000                   65.1501236
2         Algeria          DZA  2000                  62.11769485
3  American Samoa          ASM  2000                         <NA>
4         Andorra          AND  2000                1287.00280762

  Infant Mortality Rates (per 1,000 live births)
0                                             92
1                                             24
2                                           35.6
3                                           <NA>
4                                            6.5
Number of rows in the DataFrame: 5208
```

The join completed in the above code chunk is an inner join and only keeps rows that exist in both Health_Data_long and Infant_Data_long. If a country-year exists in one dataset but not the other, it will be dropped.

**Question 4 - Investigating the Relationship Between Current Healthcare Expenditure per capita and Infant Mortality Rates from 2000 - 2022**

**Missing Data - Table 1**

| | Dataset | Total Missing Observations |
|---|---|---|
| 0 | Health_Data | 1099 |
| 1 | Infant_Data | 700 |

Both datasets contained a considerable amount of missing data, illustrated in Table 1. Potentially due to countries not collecting the data or collecting the data at different year intervals. Missing data can have a large impact on data analysis if not handled properly and can lead to skewed or incorrect conclusions. The year 2023 contained no data; therefore, this column was dropped. To deal with the other missing data, I decided to drop all rows containing missing data, sometimes, this could result in a significant reduction of sample size; however, in this case, 1099 observations were removed (21.1% of the dataset) and only 27 countries were dropped, indicating this was an effective method to handling missing data as there were still 4109 observations. An alternative approach would've been mean, multiple or regression imputation if dropping rows with missing data caused a significant decrease in sample size.

**Summary Statistics - Table 2**

| Variable | N | Mean | Median | SD | Min | Max |
|---|---|---|---|---|---|---|
| Heathcare Expenditure (USD) | 4109.0 | 956.0 | 256.7 | 1685.7 | 4.0 | 12473.8 |
| Infant Mortality Rates (per 1,000 live births) | 4109.0 | 26.9 | 17.7 | 25.0 | 1.4 | 138.3 |

Table 2 displays the summary statistics for healthcare expenditure (USD) and infant mortality rates (per 1,000 live births) across 4109 observations, revealing significant differences between countries.

Healthcare expenditure per capita showed a mean of \$956.0 but a far lower median of \$256.7, indicating a negatively skewed distribution where few countries spend significantly more. The large standard deviation (\$1,685.7) and range (\$4.0–\$12,473.8) highlight large global and temporal differences in healthcare investment.

Infant mortality rates show similar variation, with a mean of 26.9 deaths per 1,000 live births and a median of 17.7. The high standard deviation (25.0) and range (1.4–138.3) suggest major differences in healthcare quality and access.

Overall, the data underscores global disparities in healthcare funding and outcomes, suggesting that higher healthcare expenditure may be linked to lower infant mortality.

**Distribution Analysis**

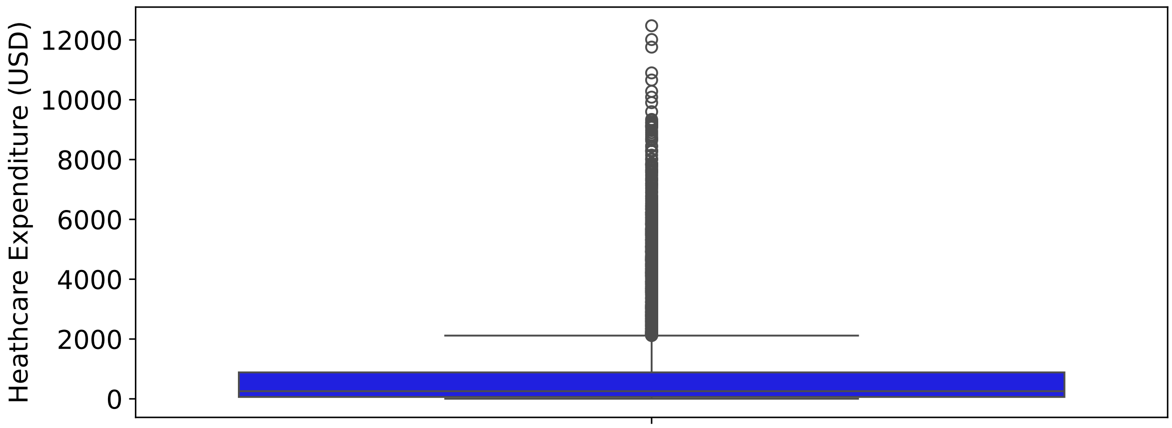


Figure 1: Box Plot of Healthcare Spend Per Capita (USD)

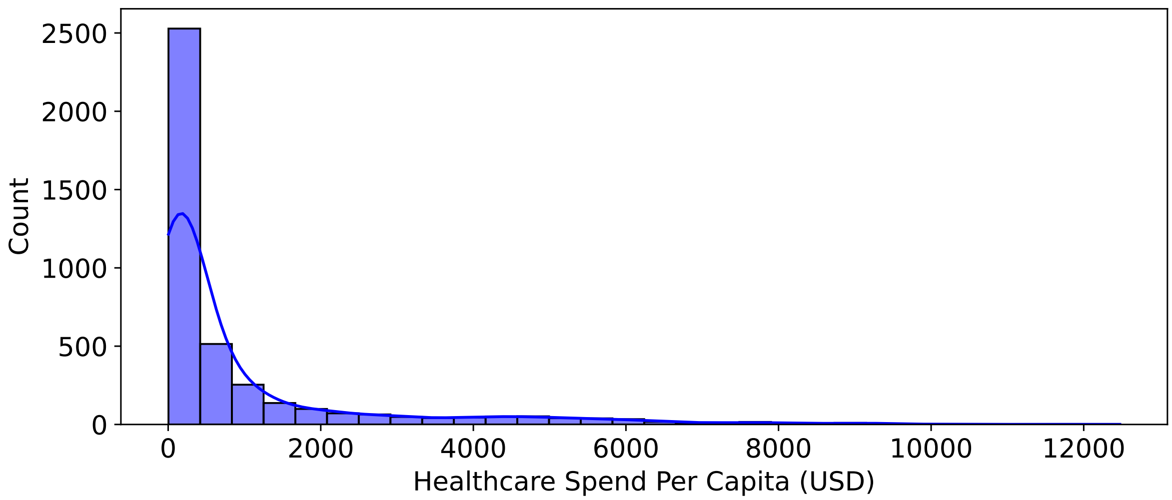


Figure 2: Histogram with Density of Healthcare Spend Per Capita (USD)

Figures 1 and 2 show the distribution of healthcare spending per capita (USD). Figure 1 shows that healthcare expenditure is highly negatively skewed, with many outliers at the upper end, supporting the analysis from the summary statistics. The median expenditure is positioned toward the lower end of the distribution, indicating that most countries spend relatively little, while a few spend significantly more. The whiskers of the box plot are short, suggesting that a large proportion of the data is concentrated within a lower range, while the numerous outliers highlight extreme spending levels in some countries.

Figure 2 reiterates the negative skew of the data. Most countries have low healthcare spending, grouped toward the left of the axis, with just a handful having exceptionally high expenditures. The density curve (smooth blue line) depicts the exponential drop in frequency as expenditure increases, emphasising that high-spending countries are exceptions rather than the rule

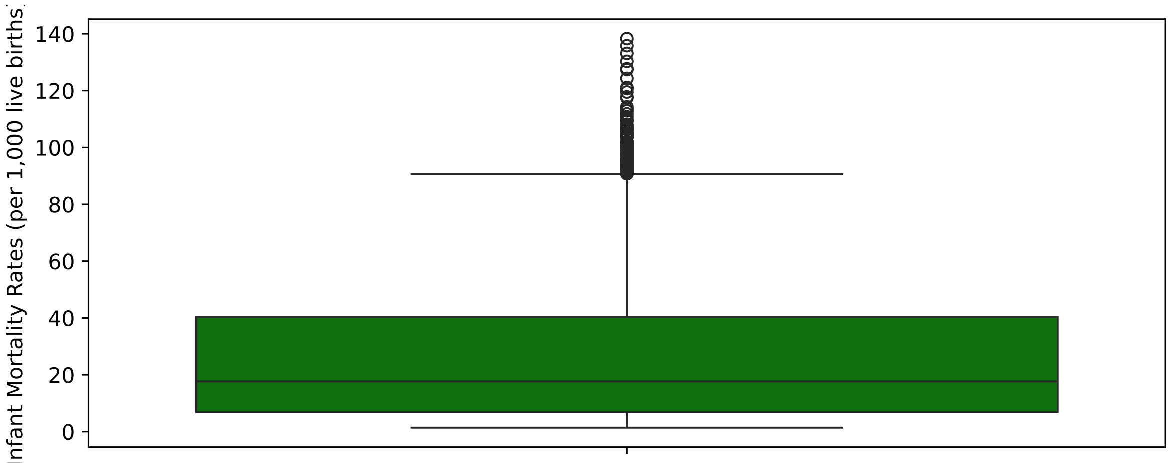


Figure 3: Box Plot of Infant Mortality Rate (per 1,000 live births)

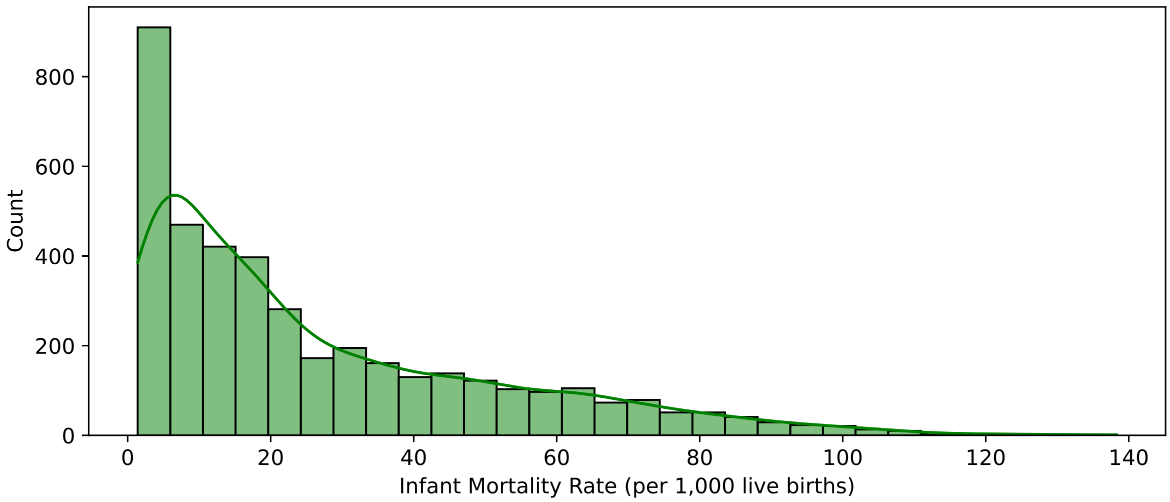


Figure 4: Histogram with Density of Infant Mortality Rate (per 1,000 live births)

The distribution of the infant mortality rate (per 1,000 live births) seen in Figures 3 and 4 is comparable to that of healthcare spend per capita (USD).

There are several outliers in Figure 3, with specific countries having abnormally high infant death rates. The data is still negatively skewed, as seen by the median being significantly lower than the upper quartile.

This is supported by Figure 4, which displays a dramatic fall as rates rise, with most values clustering below 40 deaths per 1,000 live births. While infant mortality is low in many nations, it is much higher in others, most likely because of infrastructural constraints, economic considerations, and healthcare discrepancies.

**Correlation Analysis**

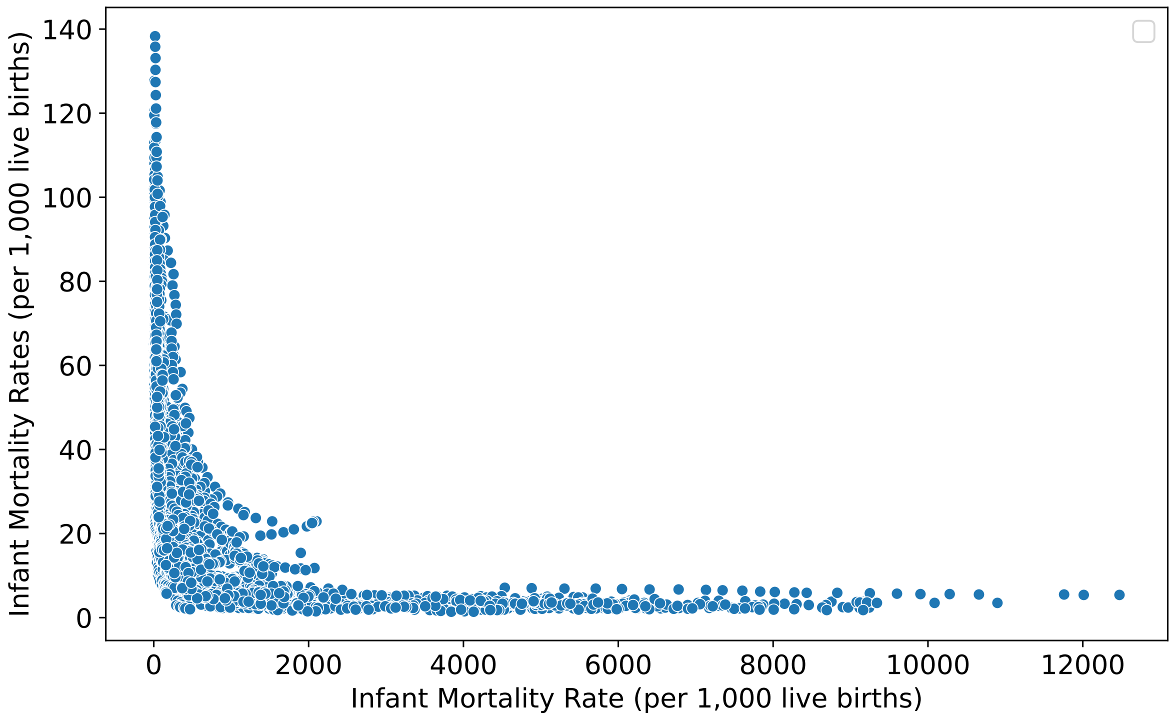

Figure 5: Scatter Plot: Healthcare Spend Per Capita vs Infant Mortality Rate

A scatter plot showing the correlation between infant mortality rate (per 1,000 live births) and healthcare spending per capita (USD) is shown in Figure 5. Higher healthcare spending is linked to decreased infant death rates, according to the trend, which shows a significant negative association.

However, the relationship is non-linear, with infant mortality declining sharply at lower healthcare spending levels and plateauing as spending rises. This points to diminishing returns, where early increases in healthcare spending have a major positive influence on infant mortality, with the effect decreasing as spending levels rise

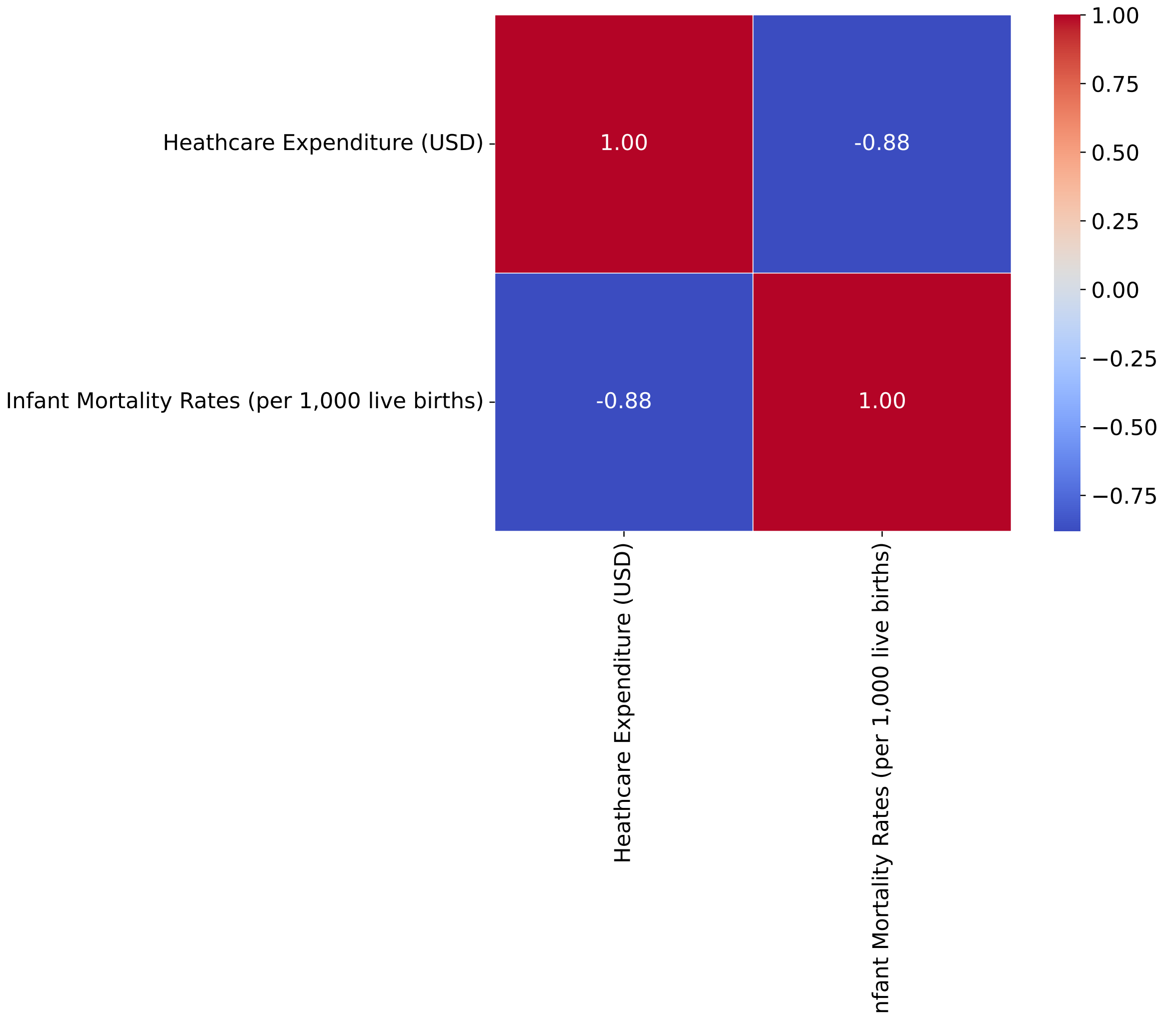


Figure 6: Correlation Matrix of Infant Mortality and Healthcare Spend per Capita

Due to this non-linearity, a correlation plot was created using Spearman's rank correlation illustrated in Figure 6. The correlation matrix confirms a strong negative correlation (-0.88) between healthcare spend per capita and infant mortality rates, indicating that as healthcare spending increases, infant mortality rates decrease, supporting the conclusions from Figure 5. The magnitude of this correlation suggests a substantial association, reinforcing the importance of healthcare investment in improving child survival outcomes. However, the correlation does not imply causation, and other socio-economic factors, such as healthcare infrastructure, accessibility, and policy effectiveness, likely influence this relationship.

**Regression Analysis**

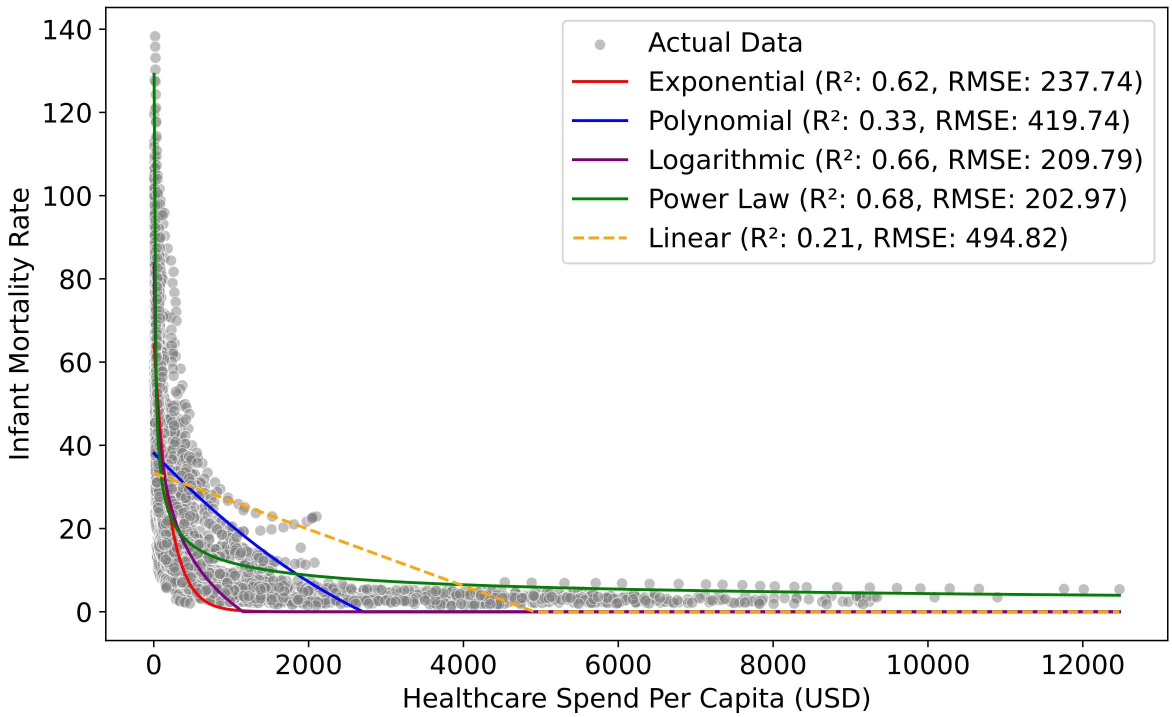


Figure 7: Comparison of Regression Models: Healthcare Spend vs Infant Mortality Rate

Figure 7 compares five regression models, exponential, polynomial, logarithmic, power law, and linear, in quantifying the relationship between healthcare spend per capita and infant mortality rates. The power law model achieves the best fit with an $R^2$ of 0.68, revealing that 68% of the variance in infant mortality rates is explained by healthcare spending per capita, and the lowest RMSE (14.25), supporting the strong inverse relationship shown in Figure 5. Exponential and logarithmic regression perform well ($R^2 = 0.62$ and 0.66, respectively), capturing the steep initial decline before stabilizing at lower mortality levels.

Polynomial regression initially follows the trend but ultimately performs poorly, indicated by its lower $R^2$ of 0.33. The linear model performs worst, failing to capture the non-linearity of the data. All models are constrained to prevent infant mortality predictions below 0, ensuring a realistic representation. This analysis highlights that non-linear models, particularly power law and exponential regression, provide the most accurate representation of the data, reiterating the non-linearity of the relationship between healthcare spend per capita and infant mortality rates.

### Time Series Analysis

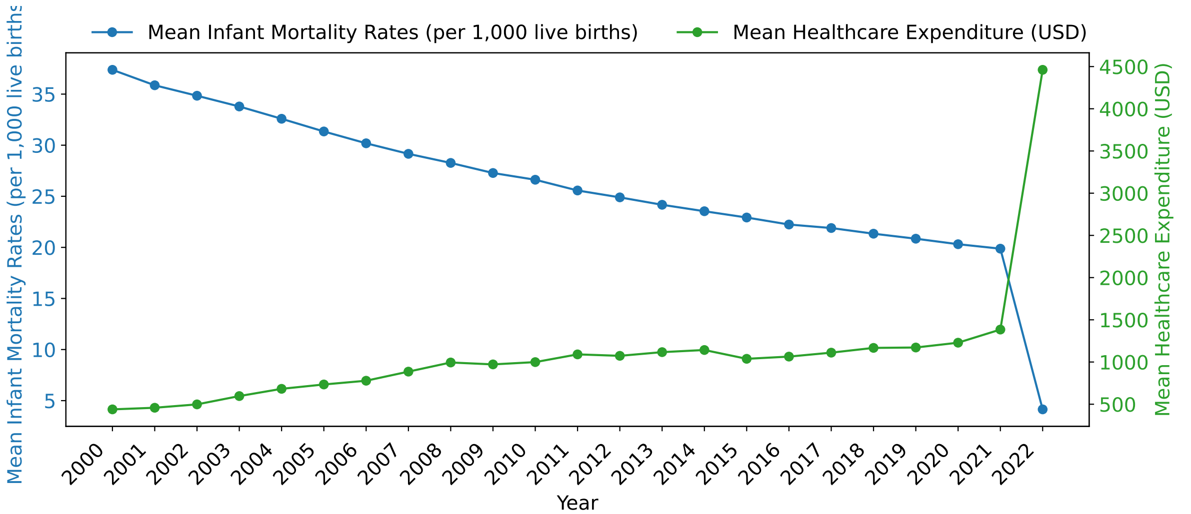


Figure 8: Time Series Analysis of Mean Infant Mortality Rates and Healthcare Expenditure

Figure 8 illustrates the negative association between mean infant mortality rates (IMR) and mean healthcare spending (USD) from 2000 to 2022. The trend indicates a continual drop in IMR, reflecting improvements in healthcare, economic development, and medical advances. Meanwhile, healthcare spending has steadily climbed, indicating greater investment in health infrastructure and services.

A noteworthy anomaly arises in 2022 when healthcare expenditures significantly increase and infant death rates significantly decrease. This large shift may be due to pandemic-related spending, emergency health interventions, or data errors. The high rise in expenditure may signal a significant policy shift, but further research is needed to assess the long-term consequences.

Overall, Figure 8 reinforces the strong negative correlation between healthcare investment and infant mortality reduction illustrated by Figures 6 and 7. However, the presence of diminishing returns suggests that beyond a certain threshold, additional spending alone may not yield proportional improvements, emphasizing the need for efficient resource allocation and targeted healthcare policies to maximize impact.

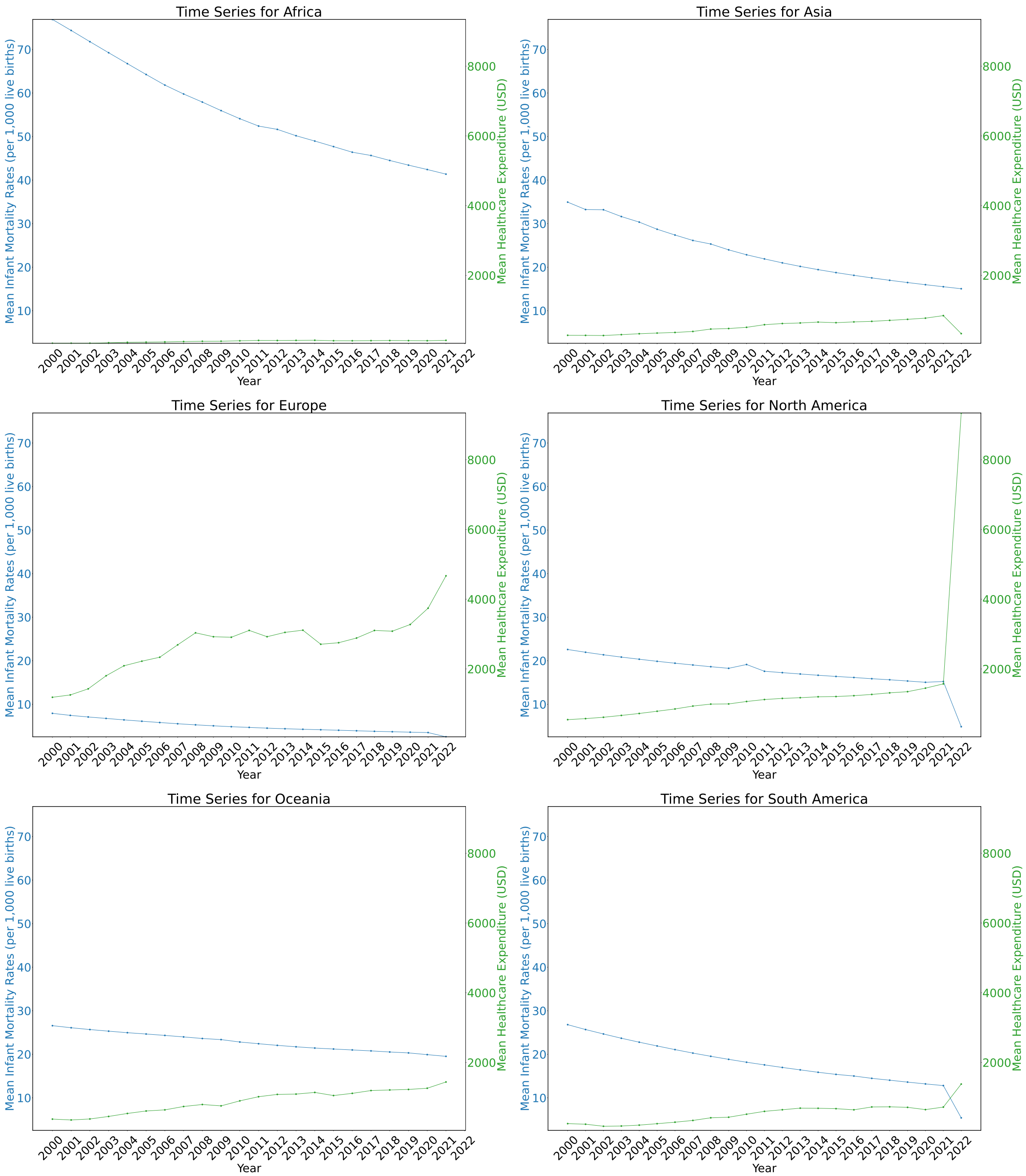


Figure 9: Time Series Analysis of Mean Infant Mortality Rates and Healthcare Expenditure By Continent

Figure 9 shows continent-specific trends in IMR and mean healthcare spending (USD) from 2000 to 2022. The constant negative relationship across each location suggests an association between increasing healthcare investment and reduced infant mortality.

Africa and Asia have had considerable drops in IMR, indicating significant improvements in healthcare despite relatively little investment. Europe and North America, with larger beginning expenditures, experienced lower IMR decreases, indicating declining returns on investment. South America and Oceania follow similar trends but have different purchasing preferences.

Healthcare expenditure rose significantly in 2022 in North and South America, most likely because of pandemic-related measures. This substantial rise in spending corresponds to a transient acceleration in IMR decreases, implying short-term healthcare gains.

Overall, Figure 9 highlights regional differences in healthcare availability and efficiency. While investment is critical to lowering infant mortality, the impact of expenditure varies by location.

**Conclusion**

This analysis confirms a strong negative relationship between healthcare expenditure and infant mortality rates across global regions. Countries with higher healthcare investment generally experience lower infant mortality, though the impact varies based on economic and healthcare infrastructure factors.

The non-linear nature of this relationship indicates diminishing returns, where initial increases in spending lead to substantial improvements, but further investments yield smaller reductions in infant mortality. Logarithmic and time-series analyses further reinforce this pattern, highlighting long-term declines in infant mortality alongside growing healthcare expenditure.

Regional disparities remain significant, with high-income regions investing more per capita while achieving lower mortality rates, whereas lower-income regions show greater relative improvements despite lower absolute spending. The 2022 anomaly suggests short-term shifts in healthcare spending and outcomes, likely due to pandemic-driven policies.

While healthcare expenditure is a crucial driver of infant survival, efficient allocation, accessibility, and policy effectiveness remain key determinants of long-term health improvements worldwide.