0.1 Bonus - (4 points)

Going back to the names data from Q3. Let's look at how the distribution of popular names has changed over time.

Create a small multiples plot (meaning each plot is the same type and showing similar information in a grid).

The small multiples plot will show how the distribution of names has changed from the 1960s to 2010s.

1960s data - https://www.ssa.gov/OACT/babynames/decades/names1960s.html
1970s data - https://www.ssa.gov/OACT/babynames/decades/names1970s.html
...
2010s data - https://www.ssa.gov/OACT/babynames/decades/names2010s.html

Let's focus on just how the distribution of girls names have changed. This can be displayed using a line plot on just the top 100 names per decade.

When getting the information, you should also scrape the total number of female births in that period to normalize the number of names as a percent in that decade.

Your final grid should display:

	Col 1	Col 2
Row 1	1960 plot	1970 plot
Row 2	1980 plot	1990 plot
Row 3	2000 plot	2010 plot

In [ ]:
```
# Create small multiples plot.
import requests
from bs4 import BeautifulSoup
import pandas as pd
import re

# Define a function to scrape names and counts from each decade page
def scrape_decade_data(url):
    response = requests.get(url)
    soup = BeautifulSoup(response.content, 'html.parser')

    # Locate the table of names
    table = soup.find('table', {'class': 't-stripe'})

    # Collect the names and counts into a list of dictionaries
    data = []
    for row in table.find_all('tr')[1:101]:  # Only top 100 rows
        cols = row.find_all('td')

        # Check if row has the expected number of columns
```

```
            if len(cols) < 3:
                continue  # Skip rows with insufficient data

            try:
                rank = int(cols[0].text.strip())
                name = cols[1].text.strip()
                count = int(cols[2].text.strip().replace(",", ""))
                data.append({'Rank': rank, 'Name': name, 'Count': count})
            except ValueError:
                continue  # Skip rows where parsing fails

    # Attempt to get the total female births by looking for a pattern
    total_female_births = None
    for p in soup.find_all('p'):
        paragraph_text = p.get_text()
        # Use regex to find total female births near phrases like "female births"
        match = re.search(r"Total female births:?\s*([\d,]+)", paragraph_text)
        if match:
            total_female_births = int(match.group(1).replace(",", ""))
            break

    # Debug print if total births are not found
    if total_female_births is None:
        print("Could not find total female births in paragraph text.")
        for p in soup.find_all('p'):
            print(p.get_text())  # Print each paragraph to help debug

    # Ensure we found the total female births; raise an error if not
    if total_female_births is None:
        raise ValueError("Total female births not found on page.")

    return pd.DataFrame(data), total_female_births

# URLs for each decade
urls = {
    '1960s': "https://www.ssa.gov/OACT/babynames/decades/names1960s.html",
    '1970s': "https://www.ssa.gov/OACT/babynames/decades/names1970s.html",
    '1980s': "https://www.ssa.gov/OACT/babynames/decades/names1980s.html",
    '1990s': "https://www.ssa.gov/OACT/babynames/decades/names1990s.html",
    '2000s': "https://www.ssa.gov/OACT/babynames/decades/names2000s.html",
    '2010s': "https://www.ssa.gov/OACT/babynames/decades/names2010s.html"
}

# Loop through each decade URL and store the data in a dictionary
decade_data = {}
for decade, url in urls.items():
    data, total_births = scrape_decade_data(url)
    data['Percentage'] = data['Count'] / total_births * 100
    decade_data[decade] = data



# Concatenate all decades into one DataFrame with a new 'Decade' column
all_data = pd.concat(
```

```
    [df.assign(Decade=decade) for decade, df in decade_data.items()],
    ignore_index=True
)

import matplotlib.pyplot as plt

# Set up the grid
fig, axes = plt.subplots(3, 2, figsize=(14, 12), sharey=True)
axes = axes.flatten()

# Plot each decade
for i, (decade, df) in enumerate(decade_data.items()):
    ax = axes[i]
    # Plot each name's rank as a line
    ax.plot(df['Rank'], df['Percentage'], marker='o', linestyle='-', color='b', alpha=0.6)
    ax.set_title(f"{decade} (Top 100 Names)")
    ax.set_xlabel("Rank")
    ax.set_ylabel("Percentage of Female Births")
    ax.grid(True)

# Adjust layout and show the plot
plt.tight_layout()
plt.show()
```