```
# Import various libraries

import pandas as pd
import numpy as np

# Load the dataset:

hist_stock_prices = pd.read_csv('historical_stock_prices.csv', on_bad_lines='skip')
hist_stocks = pd.read_csv('historical_stocks.csv')

print(hist_stock_prices.head())
print("---------------------------------------------------------------------------------")
print(hist_stocks.head())
```

```
  ticker   open  close  adj_close    low   high     volume        date
0    AHH  11.50  11.58   8.493155  11.25  11.68  4633900.0  2013-05-08
1    AHH  11.66  11.55   8.471151  11.50  11.66   275800.0  2013-05-09
2    AHH  11.55  11.60   8.507822  11.50  11.60   277100.0  2013-05-10
3    AHH  11.63  11.65   8.544494  11.55  11.65   147400.0  2013-05-13
4    AHH  11.60  11.53   8.456484  11.50  11.60   184100.0  2013-05-14
------------------------------------------------------------------------------
  ticker exchange                                  name            sector  \
0    PIH   NASDAQ  1347 PROPERTY INSURANCE HOLDINGS, INC.           FINANCE
1  PIHPP   NASDAQ  1347 PROPERTY INSURANCE HOLDINGS, INC.           FINANCE
2   TURN   NASDAQ                 180 DEGREE CAPITAL CORP.           FINANCE
3   FLWS   NASDAQ                  1-800 FLOWERS.COM, INC.  CONSUMER SERVICES
4   FCCY   NASDAQ            1ST CONSTITUTION BANCORP (NJ)           FINANCE

                    industry
0  PROPERTY-CASUALTY INSURERS
1  PROPERTY-CASUALTY INSURERS
2  FINANCE/INVESTORS SERVICES
3      OTHER SPECIALTY STORES
4        SAVINGS INSTITUTIONS
```

```
# Data Inspection:

print(hist_stock_prices.info())
print("-------------------------")
print(hist_stocks.info())

print("-------------------------")
print("Summary---Stats")
print(hist_stock_prices.describe(), hist_stocks.describe())


```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1747016 entries, 0 to 1747015
Data columns (total 8 columns):
 #   Column     Dtype  
---  ------     -----  
 0   ticker     object 
 1   open       float64
 2   close      float64
 3   adj_close  float64
 4   low        float64
 5   high       float64
 6   volume     float64
 7   date       object 
dtypes: float64(6), object(2)
memory usage: 106.6+ MB
None
-------------------------
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6460 entries, 0 to 6459
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   ticker    6460 non-null   object
 1   exchange  6460 non-null   object
 2   name      6460 non-null   object
 3   sector    5020 non-null   object
 4   industry  5020 non-null   object
dtypes: object(5)
memory usage: 252.5+ KB
None
-------------------------
Summary---Stats
```

```
               open         close     adj_close           low          high  \
count  1.747016e+06  1.747016e+06  1.747016e+06  1.747015e+06  1.747015e+06
mean   2.314312e+02  2.317348e+02  2.272249e+02  2.265874e+02  2.365962e+02
std    7.573427e+03  7.604517e+03  7.603762e+03  7.431390e+03  7.747283e+03
min    5.000000e-03  5.000000e-03  1.464524e-05  5.000000e-03  5.000000e-03
25%    6.810000e+00  6.810487e+00  4.378336e+00  6.680000e+00  6.938271e+00
50%    1.471857e+01  1.471857e+01  1.109731e+01  1.450000e+01  1.493750e+01
75%    2.975000e+01  2.975000e+01  2.509826e+01  2.933333e+01  3.013000e+01
max    8.028744e+05  8.002500e+05  8.002500e+05  7.290000e+05  8.190000e+05

             volume
count  1.747015e+06
mean   2.885017e+06
std    4.245034e+07
min    1.000000e+00
25%    2.190000e+04
50%    1.231000e+05
75%    5.387000e+05
max    4.483504e+09          ticker exchange                          name   sector  \
count    6460     6460                         6460     5020
unique   6460        2                         5462       13
top      ZYME   NASDAQ  BANK OF AMERICA CORPORATION  FINANCE
freq        1     3308                           16     1022

                     industry
```

Strategies I applied to the Data Cleaning for the Stock Prices File:

The raw historical stock price dataset contained multiple extreme and unrealistic values, including adjusted closing prices in the 10^19 range and stock prices exceeding two million dollars. These outliers were identified using summary statistics and removed by applying reasonable bounds for valid stock price ranges ($0.1–$10,000) and trading volumes (0–500 million shares). Missing or malformed dates were cleaned by converting the date column to datetime format and dropping invalid entries. Duplicate rows (ticker–date combinations) were removed to ensure accurate time-series analysis. After cleaning, all price and volume columns contained realistic values, and the dataset was safely prepared for analysis."

```
# Data Cleaning & Processing

# Convert Date to Date_Time:

hist_stock_prices['date'] = pd.to_datetime(hist_stock_prices['date'], errors='coe
hist_stock_prices = hist_stock_prices.dropna(subset=['date'])

# Drop the Duplicates for Ticker and Date: This ensures that the ticker and date 

hist_stock_prices = hist_stock_prices.drop_duplicates(subset=['ticker', 'date'])

# Fix Extreme Price Outliers in the dataset and remove unrealistic Prices (Outlie

valid_prices = (hist_stock_prices['close'] > 0.1) & (hist_stock_prices['close'] <
hist_stock_prices = hist_stock_prices[valid_prices]

for col in ['open', 'high', 'low', 'adj_close']:  # apply the same range to the o
  valid_range = (hist_stock_prices[col] >0.1) & (hist_stock_prices[col] < 1000)
  hist_stock_prices = hist_stock_prices[valid_range]

# Handle the volume col based on the reality:

valid_volume = (hist_stock_prices['volume'] > 0) & (hist_stock_prices['volume'] <
hist_stock_prices = hist_stock_prices[valid_volume]

# Remove Rows that have missing values to ensure that the price cols have usable 

price_cols=['open', 'high', 'low', 'close', 'adj_close'] # ensures missing rows a
hist_stock_prices = hist_stock_prices.dropna(subset=price_cols)

# Sort by Ticker (Unique Identifier) & the Date:

hist_stock_prices = hist_stock_prices.sort_values(by=['ticker', 'date'])

# Reset the Index - this syntax ensures that the index for the dataset is reset a
hist_stock_prices = hist_stock_prices.reset_index(drop=True)

hist_stock_prices = hist_stock_prices.set_index('date')
```

```

# Check the execution:

# print(hist_stock_prices.describe())
print(hist_stock_prices.info(), hist_stock_prices.head())

```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1714712 entries, 1987-09-23 to 2018-08-24
Data columns (total 7 columns):
 #   Column     Dtype  
---  ------     -----  
 0   ticker     object 
 1   open       float64
 2   close      float64
 3   adj_close  float64
 4   low        float64
 5   high       float64
 6   volume     float64
dtypes: float64(6), object(1)
memory usage: 104.7+ MB
None            ticker      open     close  adj_close       low      high  \
date                                                                     
1987-09-23   AAPL  1.933036  1.973214   0.101052  1.919643  2.000000   
1987-09-24   AAPL  1.973214  2.017857   0.103339  1.973214  2.066964   
1987-09-25   AAPL  2.026786  2.053571   0.105168  2.017857  2.071429   
1987-09-28   AAPL  2.053571  1.991071   0.101967  1.982143  2.098214   
1987-09-30   AAPL  1.937500  2.017857   0.103339  1.937500  2.035714   

                volume  
date                    
1987-09-23  63644000.0  
1987-09-24  45640000.0  
1987-09-25  26630800.0  
1987-09-28  50960000.0  
1987-09-30  30520000.0  
```

```
# Checking Cols in Historical Stock Dataset:

print(hist_stocks.info())
print(hist_stocks.describe(include='all'))
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6460 entries, 0 to 6459
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   ticker    6460 non-null   object
 1   exchange  6460 non-null   object
 2   name      6460 non-null   object
 3   sector    5020 non-null   object
 4   industry  5020 non-null   object
dtypes: object(5)
memory usage: 252.5+ KB
None
       ticker exchange                         name   sector  \
count    6460     6460                         6460     5020   
unique   6460        2                         5462       13   
top      ZYME   NASDAQ  BANK OF AMERICA CORPORATION  FINANCE   
freq        1     3308                           16     1022   

                     industry  
count                    5020  
unique                    136  
top     MAJOR PHARMACEUTICALS  
freq                      419  
```

```
# Create the Decade Cols ( Cols Segmentation)
#The best practice would be to create a decade col before merging to avoid comput

# Extract the year
hist_stock_prices['year'] = hist_stock_prices.index.year # extracting the year co

# Convert Year into Decades
hist_stock_prices['decade'] = (hist_stock_prices['year'] // 10) * 10 # this synta

hist_stock_prices['decade'] = hist_stock_prices['decade'].astype(str) + "s" # cha

# Segment Decade Cols
decades = sorted(hist_stock_prices['decade'].unique())
print(decades)
```

```

# Construct a Dataframe using the  Dictionary method:

decades_dfs ={decade: hist_stock_prices[hist_stock_prices['decade'] == decade]
              for decade in decades}

print(decades_dfs["1990s"].head())

```

```
['1970s', '1980s', '1990s', '2000s', '2010s']
           ticker      open     close  adj_close       low      high  \
date
1990-01-02   AAPL  1.258929  1.330357   0.122946  1.250000  1.339286
1990-01-03   AAPL  1.357143  1.339286   0.123771  1.339286  1.357143
1990-01-04   AAPL  1.366071  1.343750   0.124184  1.330357  1.383929
1990-01-05   AAPL  1.348214  1.348214   0.124597  1.321429  1.366071
1990-01-08   AAPL  1.339286  1.357143   0.125422  1.321429  1.357143

                volume  year decade
date
1990-01-02  45799600.0  1990  1990s
1990-01-03  51998800.0  1990  1990s
1990-01-04  55378400.0  1990  1990s
1990-01-05  30828000.0  1990  1990s
1990-01-08  25393200.0  1990  1990s
```

The metadata cleaning process successfully standardized all text-based columns, removed extra whitespace, handled missing values, and optimized categorical variables. The ‘ticker’, ‘exchange’, ‘sector’, and ‘industry’ columns were converted to categorical data types, improving memory efficiency and enabling faster analytical operations. Missing sector and industry values were filled with a placeholder category (“Unknown”), ensuring no null values remain in any categorical fields. After cleaning, the dataset retains all 6,460 unique companies, and the previews confirm that fields are well formatted and ready for merging with the stock price dataset.

```
# Data Cleaning Process

# Re-load hist_stocks to ensure it's a DataFrame
hist_stocks = pd.read_csv('historical_stocks.csv')

# REMOVE DUPLICATE ROWS BASED ON 'ticker' this is important before merging with t
hist_stocks = hist_stocks.drop_duplicates(subset=['ticker'])

# CLEAN TEXT COLUMNS (remove extra spaces, standardize case)

text_cols=['ticker','exchange','name','sector','industry']

for col in text_cols:
  hist_stocks[col] = hist_stocks[col].astype(str).str.strip() # this sytnax remov
  hist_stocks[col] = hist_stocks[col].replace('NaN', None) # this replaces NaN an

# FILL MISSING VALUES FOR SECTOR & INDUSTRY:

hist_stocks['sector'] = hist_stocks['sector'].fillna('Unknown')
hist_stocks['industry'] = hist_stocks['industry'].fillna('Unknown')

# Convert Cols to Category:

categorical_cols = ['exchange', 'sector', 'industry'] # this is a list that used

for cols in categorical_cols:
  hist_stocks[cols] = hist_stocks[cols].astype('category')

hist_stocks = hist_stocks.reset_index(drop=True)

# Check execution:
print("Inspect the Dataset & First 5 Heads")
print(hist_stocks.info())
print(hist_stocks.head())

```

```
Inspect the Dataset & First 5 Heads
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6460 entries, 0 to 6459
Data columns (total 5 columns):
```

```
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   ticker    6460 non-null   object
 1   exchange  6460 non-null   category
 2   name      6460 non-null   object
 3   sector    6460 non-null   category
 4   industry  6460 non-null   category
dtypes: category(3), object(2)
memory usage: 132.2+ KB
None
  ticker exchange                                    name             sector  \
0    PIH   NASDAQ  1347 PROPERTY INSURANCE HOLDINGS, INC.            FINANCE
1  PIHPP   NASDAQ  1347 PROPERTY INSURANCE HOLDINGS, INC.            FINANCE
2   TURN   NASDAQ                180 DEGREE CAPITAL CORP.            FINANCE
3   FLWS   NASDAQ                 1-800 FLOWERS.COM, INC.  CONSUMER SERVICES
4   FCCY   NASDAQ           1ST CONSTITUTION BANCORP (NJ)            FINANCE

                    industry
0  PROPERTY-CASUALTY INSURERS
1  PROPERTY-CASUALTY INSURERS
2  FINANCE/INVESTORS SERVICES
3      OTHER SPECIALTY STORES
4        SAVINGS INSTITUTIONS
```

```
# Merging the Dataset into  Dataframe ( using the most common value - ticker (lef

# Reset_Index so Date Col become an Index for merging:

hist_stock_prices = hist_stock_prices.reset_index()

merge_decade_df = hist_stock_prices.merge(hist_stocks, on='ticker', how='left')

merged_decade_df = merge_decade_df.set_index('date')

print(merged_decade_df.info())
print(merged_decade_df.head())
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 1714712 entries, 1987-09-23 to 2018-08-24
Data columns (total 13 columns):
 #   Column     Dtype
---  ------     -----
 0   ticker     object
 1   open       float64
 2   close      float64
 3   adj_close  float64
 4   low        float64
 5   high       float64
 6   volume     float64
 7   year       int32
 8   decade     object
 9   exchange   category
 10  name       object
 11  sector     category
 12  industry   category
dtypes: category(3), float64(6), int32(1), object(3)
memory usage: 143.9+ MB
None
           ticker      open     close  adj_close       low      high  \
date
1987-09-23   AAPL  1.933036  1.973214   0.101052  1.919643  2.000000
1987-09-24   AAPL  1.973214  2.017857   0.103339  1.973214  2.066964
1987-09-25   AAPL  2.026786  2.053571   0.105168  2.017857  2.071429
1987-09-28   AAPL  2.053571  1.991071   0.101967  1.982143  2.098214
1987-09-30   AAPL  1.937500  2.017857   0.103339  1.937500  2.035714

                volume  year decade exchange        name      sector  \
date
1987-09-23  63644000.0  1987  1980s   NASDAQ  APPLE INC.  TECHNOLOGY
1987-09-24  45640000.0  1987  1980s   NASDAQ  APPLE INC.  TECHNOLOGY
1987-09-25  26630800.0  1987  1980s   NASDAQ  APPLE INC.  TECHNOLOGY
1987-09-28  50960000.0  1987  1980s   NASDAQ  APPLE INC.  TECHNOLOGY
1987-09-30  30520000.0  1987  1980s   NASDAQ  APPLE INC.  TECHNOLOGY

                          industry
date
1987-09-23  COMPUTER MANUFACTURING
1987-09-24  COMPUTER MANUFACTURING
1987-09-25  COMPUTER MANUFACTURING
1987-09-28  COMPUTER MANUFACTURING
1987-09-30  COMPUTER MANUFACTURING
```

“The summary statistics reveal a steady increase in average stock prices across decades. Earlier decades such as the 1970s and 1980s show lower mean and median values, while the 1990s and 2000s present higher averages, indicating market growth. Standard deviation increases in the later decades, suggesting greater volatility and larger price movements during periods such as the dot-com boom in the late 1990s and the financial crisis of the late 2000s.”

```
# Summary Stats of the Dataset (finding the mean, median, max and std for each de

summary_stats = {}

for decade, df in decades_dfs.items():
    summary_stats[decade] = df[['open', 'high', 'low', 'close', 'volume']].agg(['

for decade, stats in summary_stats.items():
    print(f"Summary Statistics for {decade}:")
    print("--------------------------------------------------------------------------------
    print(stats)

```

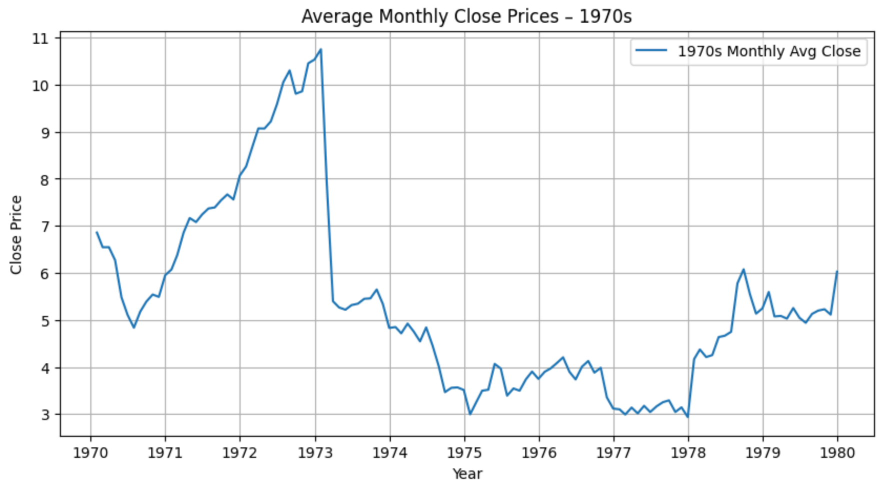
```
Summary Statistics for 1970s:
----------------------------------------------------------------------
             open       high        low      close        volume
mean     4.905704   4.958258   4.857528   4.906648  3.304594e+05
median   6.265625   6.312500   6.218750   6.265625  2.288000e+05
max     12.875000  13.750000  12.875000  13.750000  8.934400e+06
std      3.664308   3.694529   3.630428   3.666292  3.690889e+05
Summary Statistics for 1980s:
----------------------------------------------------------------------
              open        high         low       close        volume
mean      9.293568    9.422624    9.186185    9.293259  4.491279e+05
median    5.145833    5.250000    5.093750    5.140506  5.360000e+04
max     187.101242  188.786835  186.679840  187.733337  2.563540e+08
std      14.647354   14.837237   14.444184   14.650037  2.906768e+06
Summary Statistics for 1990s:
----------------------------------------------------------------------
              open        high         low       close        volume
mean     21.704312   22.075988   21.326369   21.702580  1.449757e+06
median    9.875000   10.000000    9.750000    9.875000  6.800000e+04
max     989.777771  997.777771  980.000000  996.888916  4.747008e+08
std      58.594048   59.746380   57.368122   58.531888  9.774339e+06
Summary Statistics for 2000s:
----------------------------------------------------------------------
              open        high         low       close        volume
mean     26.907055   27.406056   26.369693   26.890713  2.734352e+06
median   15.280000   15.531250   15.015000   15.280000  1.439000e+05
max     998.400024  999.799988  996.799988  998.500000  4.959490e+08
std      53.760724   54.969744   52.379907   53.632287  1.748728e+07
Summary Statistics for 2010s:
----------------------------------------------------------------------
              open        high         low       close        volume
mean     35.193171   35.623808   34.747779   35.191381  1.366161e+06
median   20.270000   20.549999   20.000000   20.274745  1.494000e+05
max     999.000000  999.000000  986.729980  992.039978  4.994696e+08
std      58.989201   59.717626   58.242893   58.972486  8.134466e+06
```

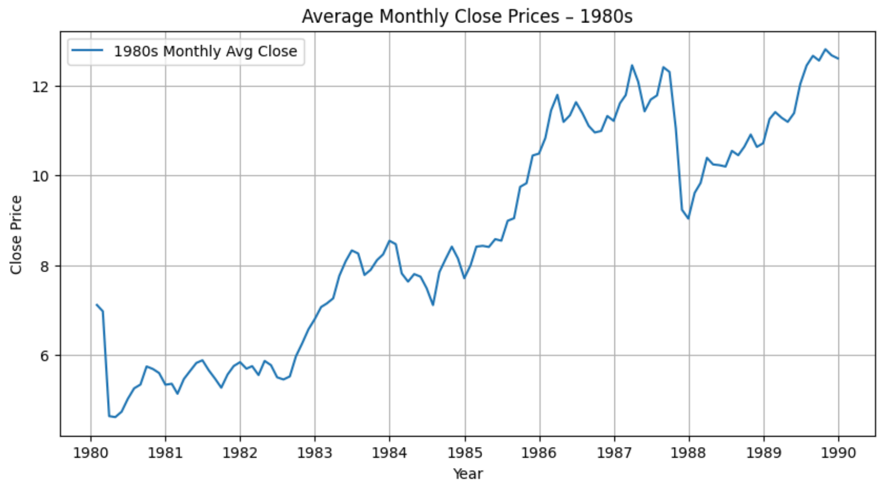
The monthly average close price plots display an upward long-term trend across all decades, reflecting overall market growth. The 1990s and early 2000s show pronounced fluctuations, consistent with the dot-com era and increased market speculation. The 2000s also exhibit a significant decline around 2008, corresponding to the global financial crisis. The smoother movement in earlier decades indicates lower volatility, while later decades show more dynamic price behavior due to increased market participation and technological changes.”

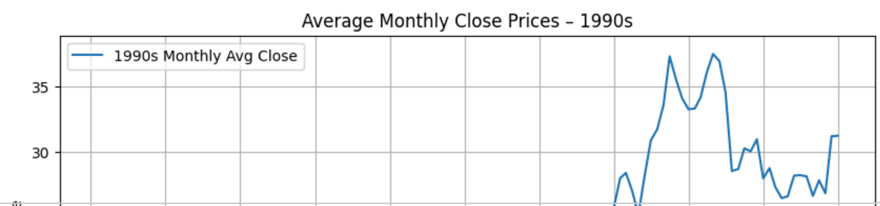
```
# Visualization

#Monthly average Close price plots for each decade

import matplotlib.pyplot as plt

for decade, df in decades_dfs.items():
    monthly_close = df['close'].resample('M').mean()

    plt.figure(figsize=(10, 5))
    plt.plot(monthly_close, label=f"{decade} Monthly Avg Close")
    plt.title(f"Average Monthly Close Prices – {decade}")
```

```
plt.xlabel("Year")
plt.ylabel("Close Price")
plt.grid(True)
plt.legend()
plt.show()

```

```
/tmp/ipython-input-1191909551.py:7: FutureWarning: 'M' is deprecated and will be remov
  monthly_close = df['close'].resample('M').mean()
```



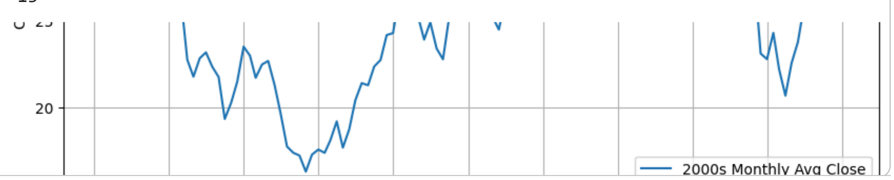


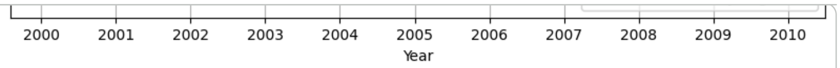


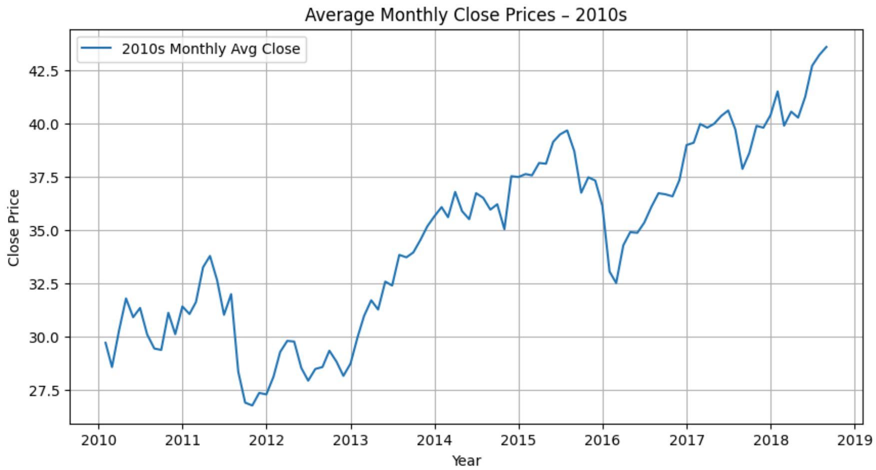
James Kiawu
12:47 PM Today
This represents the reference to my sources

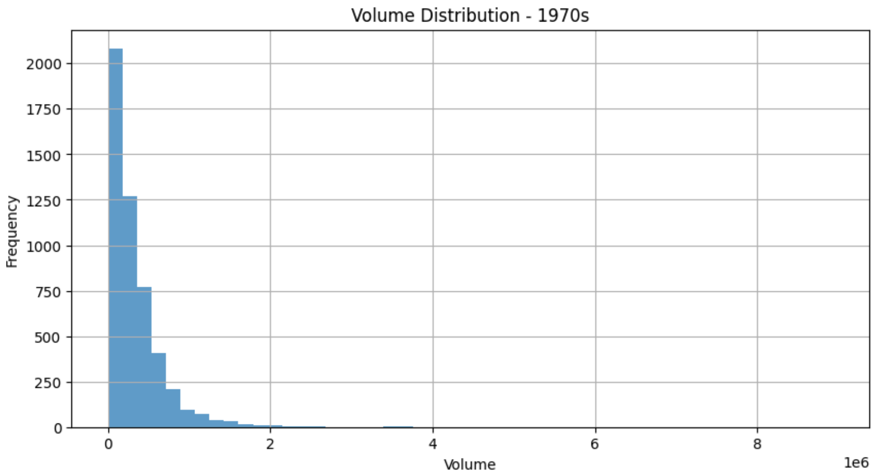
Volume histograms show a clear increase in trading activity across decades. The 1970s and 1980s demonstrate relatively low and tightly distributed volumes, whereas the 1990s onward show significantly higher and more widely spread trading activity. The broader distributions in the 2000s and 2010s indicate increased liquidity and more frequent high-volume trading days, likely due to institutional investors, technology-driven trading, and global market accessibility.

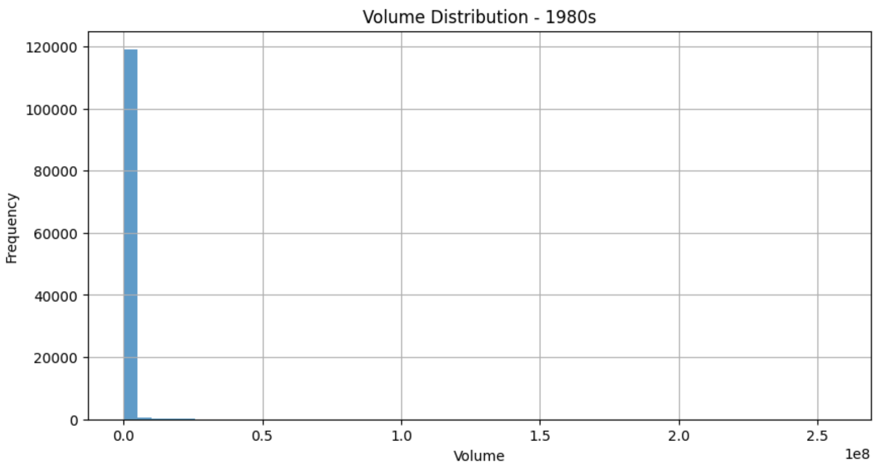
```
# Vizualization

# Volume Distribution Histogram for Each Decade

for decade, df in decades_dfs.items():
    plt.figure(figsize=(10, 5))
    plt.hist(df['volume'], bins=50, alpha=0.7)
    plt.title(f'Volume Distribution - {decade}')
    plt.xlabel('Volume')
    plt.ylabel('Frequency')
    plt.grid(True)
    plt.show()

```

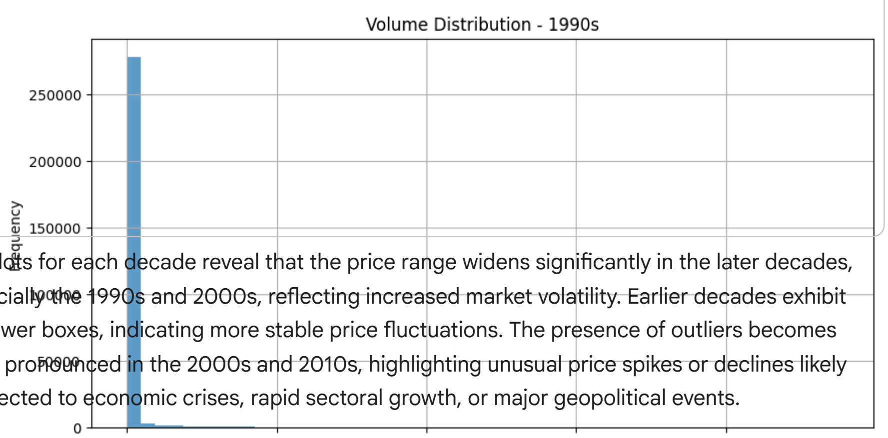



2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
Year


Average Monthly Close Prices – 2010s
2010s Monthly Avg Close
Close Price
42.5
40.0
37.5
35.0
32.5
30.0
27.5
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
Year







Boxplots for each decade reveal that the price range widens significantly in the later decades, especially the 1990s and 2000s, reflecting increased market volatility. Earlier decades exhibit narrower boxes, indicating more stable price fluctuations. The presence of outliers becomes more pronounced in the 2000s and 2010s, highlighting unusual price spikes or declines likely connected to economic crises, rapid sectoral growth, or major geopolitical events.

```
# Vizualization

#  Boxplots for High and Low prices per decade

for decade, df in decades_dfs.items():
    plt.figure(figsize=(10, 5))
    plt.boxplot([df['high'], df['low']], label=['High', 'Low'])
    plt.title(f'High & Low Price Distribution - {decade}')
    plt.ylabel('Price')
    plt.grid(True)
    plt.tight_layout()
    plt.show()
```