```
## Load and inspect the nls97 and covidtotals datasets

import pandas as pd

# Load the datasets (adjust the file paths as needed)
nls97 = pd.read_csv("nls97.csv")
covidtotals = pd.read_csv("covidtotals.csv")

# Display brief overview
print("nls97 Dataset Info:")
print(nls97.info())
print("\nCovidtotals Dataset Info:")
print(covidtotals.info())
```

```
nls97 Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5 entries, 0 to 4
Data columns (total 6 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   personid         5 non-null      int64
 1   gender           5 non-null      object
 2   maritalstatus    5 non-null      object
 3   govtsup          5 non-null      object
 4   nightlyhrssleep  5 non-null      int64
 5   childathome      5 non-null      int64
dtypes: int64(3), object(3)
memory usage: 372.0+ bytes
None

Covidtotals Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5 entries, 0 to 4
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   iso_code      5 non-null      object
 1   location      5 non-null      object
 2   total_cases   5 non-null      int64
 3   total_deaths  5 non-null      int64
dtypes: int64(2), object(2)
memory usage: 292.0+ bytes
None
```

```
## Set indices for nls97 (personid) and covidtotals (iso_code)
nls97.set_index('personid', inplace=True)
covidtotals.set_index('iso_code', inplace=True)
print(nls97.info())
print(covidtotals.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 5 entries, 101 to 105
Data columns (total 5 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   gender           5 non-null      object
 1   maritalstatus    5 non-null      object
 2   govtsup          5 non-null      object
 3   nightlyhrssleep  5 non-null      int64
 4   childathome      5 non-null      int64
dtypes: int64(2), object(3)
memory usage: 240.0+ bytes
None
<class 'pandas.core.frame.DataFrame'>
Index: 5 entries, USA to GBR
Data columns (total 3 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   location      5 non-null      object
 1   total_cases   5 non-null      int64
 2   total_deaths  5 non-null      int64
dtypes: int64(2), object(1)
memory usage: 160.0+ bytes
None
```

```
## View column names and data types for both datasets

print("nls97 Columns and Data Types:\n", nls97.dtypes)
print("\ncovidtotals Columns and Data Types:\n", covidtotals.dtypes)
```

```
nls97 Columns and Data Types:
 gender             object
maritalstatus      object
govtsup            object
nightlyhrssleep     int64
childathome         int64
dtype: object

covidtotals Columns and Data Types:
 location        object
total_cases      int64
total_deaths     int64
dtype: object
```

```
## Display the first few rows of both datasets

print("First 5 rows of nls97:\n", nls97.head())
print("\nFirst 5 rows of covidtotals:\n", covidtotals.head())
```

```
First 5 rows of nls97:
            gender maritalstatus govtsup  nightlyhrssleep  childathome
personid
101         Male        Single     Yes                6            0
102       Female       Married      No                5            2
103       Female        Single     Yes                4            3
104         Male      Divorced      No                3            4
105         Male       Married      No                7            1

First 5 rows of covidtotals:
                  location  total_cases  total_deaths
iso_code
USA        United States     33000000        588000
IND                India     29000000        350000
BRA               Brazil     17000000        470000
RUS               Russia      5000000        120000
GBR       United Kingdom      4500000        128000
```

```
## Check the shape and unique index values

print("nls97 Shape:", nls97.shape)
print("covidtotals Shape:", covidtotals.shape)

print("Unique nls97 Indices:", nls97.index.nunique())
print("Unique covidtotals Indices:", covidtotals.index.nunique())
```

```
nls97 Shape: (5, 5)
covidtotals Shape: (5, 3)
Unique nls97 Indices: 5
Unique covidtotals Indices: 5
```

```
## Select specific columns from

# Using []
print(nls97['gender'].head())

# Using loc
print(nls97.loc[:, 'gender'].head())

# Using iloc (assuming gender is the 2nd column)
print(nls97.iloc[:, 1].head())
```

```
personid
101      Male
102    Female
103    Female
104      Male
105      Male
Name: gender, dtype: object
personid
101      Male
102    Female
103    Female
104      Male
```

```
105      Male
Name: gender, dtype: object
personid
101      Single
102     Married
103      Single
104    Divorced
105     Married
Name: maritalstatus, dtype: object
```

```
### Select multiple columns based on a list

cols = ['gender', 'maritalstatus', 'nightlyhrssleep']
print(nls97[cols].head())
```

```
           gender maritalstatus  nightlyhrssleep
personid
101          Male        Single                6
102        Female       Married                5
103        Female        Single                4
104          Male      Divorced                3
105          Male       Married                7
```

```
## Select rows using slicing, loc, and iloc

# Slicing by position
print(nls97[10:15])

# Using loc with index labels
print(nls97.loc[[101, 102]])  # Assuming 101 and 102 are valid personids

# Using iloc
print(nls97.iloc[5:10])
```

```
Empty DataFrame
Columns: [gender, maritalstatus, govtsup, nightlyhrssleep, childathome]
Index: []
           gender maritalstatus govtsup  nightlyhrssleep  childathome
personid
101          Male        Single     Yes                6            0
102        Female       Married      No                5            2
Empty DataFrame
Columns: [gender, maritalstatus, govtsup, nightlyhrssleep, childathome]
Index: []
```

```
## Filter rows based on conditions

# Single condition
print(nls97[nls97['nightlyhrssleep'] <= 4])

# Multiple conditions
filtered = nls97[(nls97['nightlyhrssleep'] <= 4) & (nls97['childathome'] >= 3)]
print(filtered)
```

```
           gender maritalstatus govtsup  nightlyhrssleep  childathome
personid
103        Female        Single     Yes                4            3
104          Male      Divorced      No                3            4
           gender maritalstatus govtsup  nightlyhrssleep  childathome
personid
103        Female        Single     Yes                4            3
104          Male      Divorced      No                3            4
```

```
## Convert object columns to categorical

for col in nls97.select_dtypes(include='object').columns:
    nls97[col] = nls97[col].astype('category')

print(nls97.dtypes)
```

```
gender             category
maritalstatus      category
govtsup            category
nightlyhrssleep       int64
childathome           int64
dtype: object
```

```
## Generate frequency distributions for categorical variables

# Example for 'maritalstatus' and 'govtsup' columns
print(nls97['maritalstatus'].value_counts(dropna=False))
print(nls97['govtsup'].value_counts(dropna=False))
```

```
maritalstatus
Married     2
Single      2
Divorced    1
Name: count, dtype: int64
govtsup
No     3
Yes    2
Name: count, dtype: int64
```

```
## Save frequency distributions to a text file

with open("frequency_distributions.txt", "w") as f:
    for col in nls97.select_dtypes(include='category').columns:
        f.write(f"Column: {col}\n")
        f.write(str(nls97[col].value_counts(dropna=False)) + "\n\n")
```

```
## Descriptive statistics for continuous variables in covidtotals

print(covidtotals[['total_cases', 'total_deaths']].describe())
```

```
         total_cases   total_deaths
count  5.000000e+00       5.000000
mean   1.770000e+07  331200.000000
std    1.320795e+07  207039.126737
min    4.500000e+06  120000.000000
25%    5.000000e+06  128000.000000
50%    1.700000e+07  350000.000000
75%    2.900000e+07  470000.000000
max    3.300000e+07  588000.000000
```

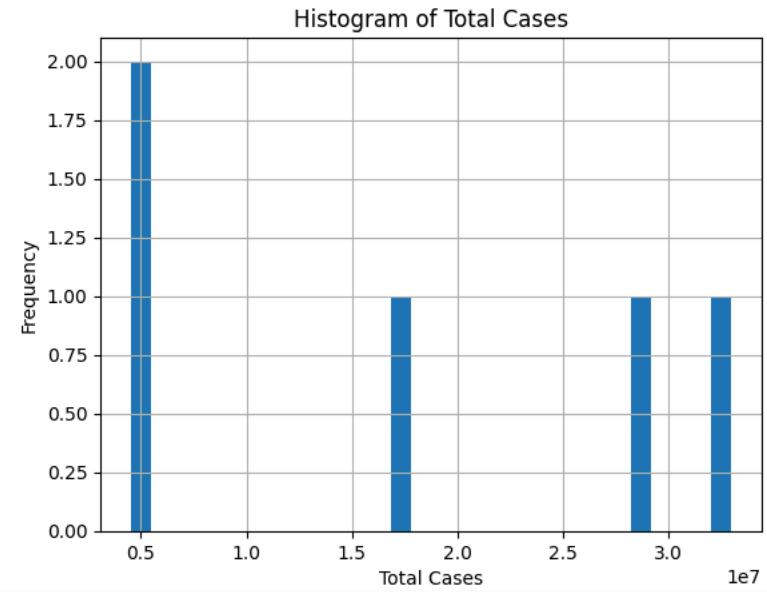
```
## Compute quantiles

print("Quantiles:\n", covidtotals[['total_cases', 'total_deaths']].quantile([0.25, 0.5, 0.75]))
```

```
Quantiles:
       total_cases  total_deaths
0.25    5000000.0      128000.0
0.50   17000000.0      350000.0
0.75   29000000.0      470000.0
```

```
# Create a histogram of total_cases
import matplotlib.pyplot as plt

plt.hist(covidtotals['total_cases'].dropna(), bins=30)
plt.title("Histogram of Total Cases")
plt.xlabel("Total Cases")
plt.ylabel("Frequency")
plt.grid(True)
plt.show()
```




```
pip install pandasai
```

```
        ^^^^^^^^^^
errupt
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```