```
import pandas as pd
```

Q1

creating a dataframe

```
data = {
    "Location": [
        "United States", "India", "Brazil", "Russia", "United Kingdom",
        "France", "Spain", "Italy", "Turkey", "Germany"
    ],
    "Cases": [
        28833039, 11079979, 10517232, 4246079, 4170519,
        3736016, 3188553, 2907825, 2693164, 2444169
    ],
    "Deaths": [
        517204, 156938, 254263, 86122, 122705,
        86332, 69142, 97507, 28503, 70601
    ],
    "Recovered": [
        None, 10763451, 9386440, 3811797, None,
        None, None, 2398352, 2565723, 2242767
    ]
}

df = pd.DataFrame(data)
df
```

	Location	Cases	Deaths	Recovered
0	United States	28833039	517204	NaN
1	India	11079979	156938	10763451.0
2	Brazil	10517232	254263	9386440.0
3	Russia	4246079	86122	3811797.0
4	United Kingdom	4170519	122705	NaN
5	France	3736016	86332	NaN
6	Spain	3188553	69142	NaN
7	Italy	2907825	97507	2398352.0
8	Turkey	2693164	28503	2565723.0
9	Germany	2444169	70601	2242767.0

creating a series

```
deaths_series = df["Deaths"]
deaths_series
```

	Deaths
0	517204
1	156938
2	254263
3	86122
4	122705
5	86332
6	69142
7	97507
8	28503
9	70601

dtype: int64

Q2

```
df["Status"] = pd.cut(df["Cases"],
                      bins=[0,3_000_000,4_000_000,float('inf')],
                            labels=["low","medium","high"])
df
```

	Location	Cases	Deaths	Recovered	Status
0	United States	28833039	517204	NaN	high
1	India	11079979	156938	10763451.0	high
2	Brazil	10517232	254263	9386440.0	high
3	Russia	4246079	86122	3811797.0	high
4	United Kingdom	4170519	122705	NaN	high
5	France	3736016	86332	NaN	medium
6	Spain	3188553	69142	NaN	medium
7	Italy	2907825	97507	2398352.0	low
8	Turkey	2693164	28503	2565723.0	low
9	Germany	2444169	70601	2242767.0	low

Q3

Death Series

```
deaths_series_ranked = deaths_series.rank(ascending=True)
deaths_series_ranked
```

	Deaths
0	10.0
1	8.0
2	9.0
3	4.0
4	7.0
5	5.0
6	2.0
7	6.0
8	1.0
9	3.0

dtype: float64

Dataframe rank

```
df_sorted = df.sort_values(by="Deaths", ascending=True)
df_sorted
```

	Location	Cases	Deaths	Recovered	Status
8	Turkey	2693164	28503	2565723.0	low
6	Spain	3188553	69142	NaN	medium
9	Germany	2444169	70601	2242767.0	low
3	Russia	4246079	86122	3811797.0	high
5	France	3736016	86332	NaN	medium
7	Italy	2907825	97507	2398352.0	low
4	United Kingdom	4170519	122705	NaN	high
1	India	11079979	156938	10763451.0	high
2	Brazil	10517232	254263	9386440.0	high
0	United States	28833039	517204	NaN	high

Q4

```
import pandas as pd
deaths_series = df["Deaths"]

# --- Statistics for Series ---
print("=== Deaths Series Statistics ===")
print("Count of non-NA values:", deaths_series.count())
print("Summary statistics:\n", deaths_series.describe())
print("Minimum:", deaths_series.min())
print("Maximum:", deaths_series.max())
print("Index position of min:", deaths_series.argmin())
print("Index position of max:", deaths_series.argmax())
print("Index label of min:", deaths_series.idxmin())
print("Index label of max:", deaths_series.idxmax())
print("Median (0.5 quantile):", deaths_series.quantile(0.5))
print("Sum:", deaths_series.sum())
print("Mean:", deaths_series.mean())
print("Median:", deaths_series.median())

# Mean absolute deviation (manual, version-safe)
mean_abs_dev = (deaths_series - deaths_series.mean()).abs().mean()
print("Mean absolute deviation:", mean_abs_dev)

print("Variance:", deaths_series.var())
print("Standard deviation:", deaths_series.std())
print("Skewness:", deaths_series.skew())
print("Kurtosis:", deaths_series.kurt())

print("Cumulative sum:\n", deaths_series.cumsum())
print("Cumulative min:\n", deaths_series.cummin())
print("Cumulative max:\n", deaths_series.cummax())
print("Cumulative product:\n", deaths_series.cumprod())
print("First difference:\n", deaths_series.diff())
print("Percentage change:\n", deaths_series.pct_change())
```

```python
# --- Statistics for DataFrame ---
print("\n=== DataFrame Statistics ===")
print("Count of non-NA values:\n", df.count())
print("Summary statistics:\n", df.describe())
print("Minimum:\n", df.min())
print("Maximum:\n", df.max())

# Row labels of min/max
print("Index label of min:\n", df.idxmin())
print("Index label of max:\n", df.idxmax())

# Integer positions of min/max for each column (numpy-based)
numeric_cols = df.select_dtypes(include="number")

# Integer positions of min/max for numeric columns
min_positions = numeric_cols.apply(lambda x: np.argmin(x.values))
max_positions = numeric_cols.apply(lambda x: np.argmax(x.values))

print("Index position of min (numeric columns):\n", min_positions)
print("Index position of max (numeric columns):\n", max_positions)

numeric_cols = df.select_dtypes(include="number")
print("Median (0.5 quantile):\n", numeric_cols.quantile(0.5))
df_numeric = df.select_dtypes(include="number")
print("Sum:\n", df_numeric.sum())
print("Mean:\n", df_numeric.mean())
print("Median:\n", df_numeric.median())

# Mean absolute deviation manually
mean_abs_dev_df = (df.select_dtypes(include="number") - df.select_dtypes(include="number").m
print("Mean absolute deviation (numeric columns):\n", mean_abs_dev_df)

print("Variance:\n", df_numeric.var())
print("Standard deviation:\n", df_numeric.std())
print("Skewness:\n", df_numeric.skew())
print("Kurtosis:\n", df_numeric.kurt())
```

```
Name: Deaths, dtype: int64
First difference:
 0        NaN
1   -360266.0
2     97325.0
3   -168141.0
4     36583.0
5    -36373.0
6    -17190.0
7     28365.0
8    -69004.0
9     42098.0
Name: Deaths, dtype: float64
Percentage change:
 0         NaN
1   -0.696565
2    0.620149
3   -0.661288
4    0.424781
5   -0.296426
6   -0.199115
7    0.410243
8   -0.707683
9    1.476967
Name: Deaths, dtype: float64

=== DataFrame Statistics ===
Count of non-NA values:
 Location     10
Cases         10
Deaths        10
Recovered      6
Status        10
dtype: int64
Summary statistics:
              Cases         Deaths     Recovered
count  1.000000e+01      10.000000  6.000000e+00
mean   7.381658e+06  148931.700000  5.194755e+06
std    8.172197e+06  143366.103826  3.845493e+06
min    2.444169e+06   28503.000000  2.242767e+06
25%    2.978007e+06   74481.250000  2.440195e+06
50%    3.953268e+06   91919.500000  3.188760e+06
75%    8.040444e+06  148379.750000  5.007770e+06
```

```
Minimum:
 Location        Brazil
Cases          2444169
Deaths           28503
Recovered      2242767.0
Status               low
dtype: object
Maximum:
 Location     United States
Cases              28833039
Deaths               517204
Recovered         10763451.0
Status                  high
dtype: object
Index label of min:
 Location     2
Cases        9
Deaths       8
Recovered    9
Status       7
dtype: int64
Index label of max:
 Location     0
Cases        0
Deaths       0
Recovered    1
Status       0
dtype: int64
Index position of min (numeric columns
 Cases        9
Deaths       8
Recovered    0
dtype: int64
Index position of max (numeric columns
 Cases        0
Deaths       0
Recovered    0
dtype: int64
Median (0.5 quantile):
 Cases        3953267.5
Deaths         91919.5
Recovered    3188760.0
Cumulative sum:
       Cases   Deaths   Recovered
0   28833039   517204         NaN
1   39913018   674142  10763451.0
2   50430250   928405  20149891.0
3   54676329  1014527  23961688.0
4   58846848  1137232         NaN
5   62582864  1223564         NaN
6   65771417  1292706         NaN
7   68679242  1390213  26360040.0
8   71372406  1418716  28925763.0
9   73816575  1489317  31168530.0
Cumulative min:
       Cases  Deaths  Recovered
0   28833039  517204        NaN
1   11079979  156938  10763451.0
2   10517232  156938   9386440.0
3    4246079   86122   3811797.0
4    4170519   86122         NaN
5    3736016   86122         NaN
6    3188553   69142         NaN
7    2907825   69142   2398352.0
8    2693164   28503   2398352.0
9    2444169   28503   2242767.0
Cumulative max:
       Cases  Deaths   Recovered
0   28833039  517204         NaN
1   28833039  517204  10763451.0
2   28833039  517204  10763451.0
3   28833039  517204  10763451.0
4   28833039  517204         NaN
5   28833039  517204         NaN
6   28833039  517204         NaN
7   28833039  517204  10763451.0
8   28833039  517204  10763451.0
9   28833039  517204  10763451.0
Cumulative product:
                Cases                Deaths
0            28833039                517204
1      319469466626181          81168961352
2  2627076008664456880  20638263620243576
3  7683677728832826704  6521108426500297136
```

```
Name: 0.5, dtype: float64
Sum:
 Cases        73816575.0
Deaths         1489317.0
Recovered     31168530.0
dtype: float64
Mean:
 Cases        7381657.5
Deaths        148931.7
Recovered     5194755.0
dtype: float64
Median:
 Cases        3953267.5
Deaths         91919.5
Recovered     3188760.0
dtype: float64
Mean absolute deviation (numeric columns):
 Cases        5.657056e+06
Deaths       9.632198e+04
Recovered    3.253460e+06
dtype: float64
Variance:
 Cases        6.678480e+13
Deaths       2.055384e+10
Recovered    1.478782e+13
dtype: float64
Standard deviation:
 Cases        8.172197e+06
Deaths       1.433661e+05
Recovered    3.845493e+06
dtype: float64
Skewness:
 Cases        2.405693
Deaths       2.249842
Recovered    0.938662
dtype: float64
Kurtosis:
 Cases        6.142260
Deaths       5.396014
Recovered   -1.580094
dtype: float64
Cumulative sum:
```

```
Cumulative product:
                Cases                Deaths     Recovered
0            28833039                517204           NaN
1      319469466626181          81168961352  1.076345e+07
2  2627076008664456880  20638263620243576  1.010305e+14
3  7683677728832826704  6521108426500297136  3.851077e+20
4 -3530367059382959568  8191788419739625648           NaN
5  6426695010879796992  2203555084571588928           NaN
6 -3145159625427226880  6546352681614863232           NaN
7  8329288016626499328  2525743648854594176  9.236238e+26
8 -6052080537096891392 -6370896385882158720  2.369763e+33
9  7311223601537315840 -4694990406290737792  5.314826e+39
First difference:
       Cases    Deaths  Recovered
0        NaN       NaN        NaN
1 -17753060.0 -360266.0        NaN
2   -562747.0   97325.0 -1377011.0
3  -6271153.0 -168141.0 -5574643.0
4    -75560.0   36583.0        NaN
5   -434503.0  -36373.0        NaN
6   -547463.0  -17190.0        NaN
7   -280728.0   28365.0        NaN
8   -214661.0  -69004.0   167371.0
9   -248995.0   42098.0 -322956.0
Percentage change:
       Cases    Deaths  Recovered
0        NaN       NaN        NaN
1 -0.615719 -0.696565        NaN
2 -0.050790  0.620149  -0.127934
3 -0.596274 -0.661288  -0.593004
4 -0.017795  0.424781   0.000000
5 -0.104184 -0.296426   0.000000
6 -0.146537 -0.199115   0.000000
7 -0.088042  0.410243  -0.370808
8 -0.073822 -0.707683   0.069786
9 -0.092454  1.476967  -0.125873
/tmp/ipython-input-1869990427.py:83: FutureWarning: The defa
  print("Percentage change:\n", df_numeric.pct_change())
```

```
=== Deaths Series Statistics ===
Count of non-NA values: 10
Summary statistics:
 count        10.000000
mean     148931.700000
std      143366.103826
min       28503.000000
25%       74481.250000
50%       91919.500000
75%      148379.750000
max      517204.000000
Name: Deaths, dtype: float64
Minimum: 28503
Maximum: 517204
Index position of min: 8
Index position of max: 0
Index label of min: 8
Index label of max: 0
Median (0.5 quantile): 91919.5
Sum: 1489317
Mean: 148931.7
Median: 91919.5
Mean absolute deviation: 96321.98000000001
Variance: 20553839726.23333
Standard deviation: 143366.10382595088
Skewness: 2.249841686549378
Kurtosis: 5.39601411977261
Cumulative sum:
 0     517204
1     674142
2     928405
3    1014527
4    1137232
5    1223564
6    1292706
7    1390213
8    1418716
9    1489317
Name: Deaths, dtype: int64
```

```
Cumulative min:
 0    517204
1    156938
2    156938
3     86122
4     86122
5     86122
6     69142
7     69142
8     28503
9     28503
Name: Deaths, dtype: int64
Cumulative max:
 0    517204
1    517204
2    517204
3    517204
4    517204
5    517204
6    517204
7    517204
8    517204
9    517204
Name: Deaths, dtype: int64
Cumulative product:
 0                  517204
1           81168961352
2       20638263620243576
3    6521108426500297136
4    8191788419739625648
5    2203555084571588928
6    6546352681614863232
7    2525743648854594176
8   -6370896385882158720
9   -4694990406290737792
Name: Deaths, dtype: int64
```

Q5

```python
# Create a copy to preserve original
df_mean_filled = df.copy()

# Fill missing values in 'Recovered' column using mean of Deaths (example)
df_mean_filled["Recovered"] = df_mean_filled["Recovered"].fillna(deaths_series.mean())

df_mean_filled
```

	Location	Cases	Deaths	Recovered	Status
0	United States	28833039	517204	148931.7	high
1	India	11079979	156938	10763451.0	high
2	Brazil	10517232	254263	9386440.0	high
3	Russia	4246079	86122	3811797.0	high
4	United Kingdom	4170519	122705	148931.7	high
5	France	3736016	86332	148931.7	medium
6	Spain	3188553	69142	148931.7	medium
7	Italy	2907825	97507	2398352.0	low
8	Turkey	2693164	28503	2565723.0	low
9	Germany	2444169	70601	2242767.0	low

Next steps: Generate code with df_mean_filled | New interactive sheet

```python
# Fill missing values in 'Recovered' column using median of Deaths (example)
df_median_filled["Recovered"] = df_median_filled["Recovered"].fillna(deaths_series.median()

df_median_filled
```