```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns



df=pd.read_csv(r"C:\Users\User\Downloads\world_population.csv")
#reading csv file
df
```

```
     Rank CCA3            Country           Capital Continent  \
0      36  AFG        Afghanistan             Kabul      Asia
1     138  ALB            Albania            Tirana    Europe
2      34  DZA            Algeria           Algiers    Africa
3     213  ASM     American Samoa         Pago Pago   Oceania
4     203  AND            Andorra  Andorra la Vella    Europe
..    ...  ...                ...               ...       ...
229   226  WLF  Wallis and Futuna          Mata-Utu   Oceania
230   172  ESH     Western Sahara          El Aaiún    Africa
231    46  YEM              Yemen             Sanaa      Asia
232    63  ZMB             Zambia            Lusaka    Africa
233    74  ZWE           Zimbabwe            Harare    Africa

     2022 Population  2020 Population  2015 Population  2010
Population  \
0         41128771.0       38972230.0       33753499.0
28189672.0
1          2842321.0        2866849.0        2882481.0
2913399.0
2         44903225.0       43451666.0       39543154.0
35856344.0
3            44273.0          46189.0          51368.0
54849.0
4            79824.0          77700.0          71746.0
71519.0
..               ...              ...              ...               ..
.
229          11572.0          11655.0          12182.0
13142.0
230         575986.0         556048.0         491824.0
413296.0
231       33696614.0       32284046.0       28516545.0
24743946.0
232       20017675.0       18927715.0              NaN
13792086.0
233       16320537.0       15669666.0       14154937.0
12839771.0

     2000 Population  1990 Population  1980 Population  1970
Population  \
```

```
0         19542982.0       10694796.0       12486631.0
10752971.0
1          3182021.0        3295066.0        2941651.0
2324731.0
2         30774621.0       25518074.0       18739378.0
13795915.0
3            58230.0          47818.0          32886.0
27075.0
4            66097.0          53569.0          35611.0
19860.0
..               ...              ...              ...               ..
.
229          14723.0          13454.0          11315.0
9377.0
230         270375.0         178529.0         116775.0
76371.0
231       18628700.0       13375121.0        9204938.0
6843607.0
232        9891136.0        7686401.0        5720438.0
4281671.0
233       11834676.0       10113893.0        7049926.0
5202918.0

     Area (km²)  Density (per km²)  Growth Rate  World Population
Percentage
0      652230.0            63.0587       1.0257
0.52
1       28748.0            98.8702       0.9957
0.04
2     2381741.0            18.8531       1.0164
0.56
3         199.0           222.4774       0.9831
0.00
4         468.0           170.5641       1.0100
0.00
..          ...                ...          ...
...
229       142.0            81.4930       0.9953
0.00
230    266000.0             2.1654       1.0184
0.01
231    527968.0            63.8232       1.0217
0.42
232    752612.0            26.5976       1.0280
0.25
233    390757.0            41.7665       1.0204
0.20

[234 rows x 17 columns]
```

```
pd.set_option('display.float_format',lambda x:'%.2f' % x)

df.info() # it give infor abt data

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 234 entries, 0 to 233
Data columns (total 17 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Rank                            234 non-null    int64
 1   CCA3                            234 non-null    object
 2   Country                         234 non-null    object
 3   Capital                         234 non-null    object
 4   Continent                       234 non-null    object
 5   2022 Population                 230 non-null    float64
 6   2020 Population                 233 non-null    float64
 7   2015 Population                 230 non-null    float64
 8   2010 Population                 227 non-null    float64
 9   2000 Population                 227 non-null    float64
 10  1990 Population                 229 non-null    float64
 11  1980 Population                 229 non-null    float64
 12  1970 Population                 230 non-null    float64
 13  Area (km²)                      232 non-null    float64
 14  Density (per km²)               230 non-null    float64
 15  Growth Rate                     232 non-null    float64
 16  World Population Percentage     234 non-null    float64
dtypes: float64(12), int64(1), object(4)
memory usage: 31.2+ KB

df.describe() #gives statsictical data

        Rank  2022 Population  2020 Population  2015 Population  \
count 234.00           230.00           233.00           230.00
mean  117.50      34632250.88      33600710.95      32066004.16
std    67.69     137889172.44     135873196.61     131507146.34
min     1.00           510.00           520.00           564.00
25%    59.25        419738.50        406471.00        394295.00
50%   117.50       5762857.00       5456681.00       5244415.00
75%   175.75      22653719.00      21522626.00      19730853.75
max   234.00    1425887337.00    1424929781.00    1393715448.00

       2010 Population  2000 Population  1990 Population  1980
Population  \
count           227.00           227.00           229.00
229.00
mean       30270164.48      26840495.26      19330463.93
16282884.78
std       126074183.54     113352454.57      81309624.96
69345465.54
min             596.00           651.00           700.00
```

```
733.00
25%          382726.50        329470.00        261928.00
223752.00
50%         4889741.00       4491202.00       3785847.00
3135123.00
75%        16825852.50      15625467.00      11882762.00
9817257.00
max      1348191368.00    1264099069.00    1153704252.00
982372466.00

       1970 Population  Area (km²)  Density (per km²)  Growth Rate  \
count           230.00      232.00             230.00       232.00
mean       15866499.13   581663.75             456.81         1.01
std        68355859.75  1769133.06            2083.74         0.01
min            752.00        1.00               0.03         0.91
25%          145880.50     2567.25              36.60         1.00
50%         2511718.00    77141.00              95.35         1.01
75%         8817329.00   414643.25             236.88         1.02
max       822534450.00 17098242.00           23172.27         1.07

       World Population Percentage
count                       234.00
mean                          0.43
std                           1.71
min                           0.00
25%                           0.01
50%                           0.07
75%                           0.28
max                          17.88
```

```
df.isnull().sum() #to know what all values are none
```

```
Rank                            0
CCA3                            0
Country                         0
Capital                         0
Continent                       0
2022 Population                 4
2020 Population                 1
2015 Population                 4
2010 Population                 7
2000 Population                 7
1990 Population                 5
1980 Population                 5
1970 Population                 4
Area (km²)                      2
Density (per km²)               4
Growth Rate                     2
World Population Percentage     0
dtype: int64
```

```
df.nunique() #gives unique values
```

```
Rank                             234
CCA3                             234
Country                          234
Capital                          234
Continent                          6
2022 Population                  230
2020 Population                  233
2015 Population                  230
2010 Population                  227
2000 Population                  227
1990 Population                  229
1980 Population                  229
1970 Population                  230
Area (km²)                       231
Density (per km²)                230
Growth Rate                      178
World Population Percentage       70
dtype: int64
```

```
df.sort_values(by="2022 Population",ascending=False).head(10)
```

	Rank	CCA3	Country	Capital	Continent	2022 Population	2020 Population	2015 Population	2010 Population
41	1	CHN	China	Beijing	Asia	1425887337.00	1424929781.00	1393715448.00	1348191368.00
92	2	IND	India	New Delhi	Asia	1417173173.00	1396387127.00	1322866505.00	1240613620.00
221	3	USA	United States	Washington, D.C.	North America	338289857.00	335942003.00	324607776.00	311182845.00
93	4	IDN	Indonesia	Jakarta	Asia	275501339.00	271857970.00	259091970.00	244016173.00
156	5	PAK	Pakistan	Islamabad	Asia	235824862.00	227196741.00	210969298.00	194454498.00
149	6	NGA	Nigeria	Abuja	Africa	218541212.00	208327405.00	183995785.00	160952853.00
27	7	BRA	Brazil	Brasilia	South America	215313498.00	213196304.00	205188205.00	
16	8	BGD	Bangladesh	Dhaka	Asia				
171	9	RUS	Russia	Moscow	Europe				
131	10	MEX	Mexico	Mexico City	North America				

```
196353492.00
16      171186372.00     167420951.00     157830000.00
148391139.00
171     144713314.00     145617329.00     144668389.00
143242599.00
131     127504125.00     125998302.00     120149897.00
112532401.00

     2000 Population  1990 Population  1980 Population  1970
Population  \
41     1264099069.00    1153704252.00     982372466.00
822534450.00
92     1059633675.00              NaN              NaN
557501301.00
221     282398554.00     248083732.00     223140018.00
200328340.00
93      214072421.00     182159874.00     148177096.00
115228394.00
156     154369924.00     115414069.00      80624057.00
59290872.00
149     122851984.00      95214257.00      72951439.00
55569264.00
27      175873720.00     150706446.00     122288383.00
96369875.00
16      129193327.00     107147651.00      83929765.00
67541860.00
171     146844839.00     148005704.00     138257420.00
130093010.00
131      97873442.00      81720428.00      67705186.00
50289306.00

     Area (km²)  Density (per km²)  Growth Rate  World Population
Percentage
41   9706961.00             146.89         1.00
17.88
92   3287590.00             431.07         1.01
17.77
221  9372610.00              36.09         1.00
4.24
93   1904569.00             144.65         1.01
3.45
156   881912.00             267.40         1.02
2.96
149   923768.00             236.58         1.02
2.74
27   8515767.00              25.28         1.00
2.70
16    147570.00            1160.04         1.01
2.15
```

```
171 17098242.00                8.46          1.00
1.81
131  1964375.00               64.91          1.01
1.60

df.corr() # correlation between everything

---------------------------------------------------------------------------
-----
ValueError                                Traceback (most recent call
last)
Cell In[28], line 1
----> 1 df.corr()

File ~\anaconda3\Lib\site-packages\pandas\core\frame.py:11049, in
DataFrame.corr(self, method, min_periods, numeric_only)
  11047 cols = data.columns
  11048 idx = cols.copy()
> 11049 mat = data.to_numpy(dtype=float, na_value=np.nan, copy=False)
  11051 if method == "pearson":
  11052     correl = libalgos.nancorr(mat, minp=min_periods)

File ~\anaconda3\Lib\site-packages\pandas\core\frame.py:1993, in
DataFrame.to_numpy(self, dtype, copy, na_value)
   1991 if dtype is not None:
   1992     dtype = np.dtype(dtype)
-> 1993 result = self._mgr.as_array(dtype=dtype, copy=copy,
na_value=na_value)
   1994 if result.dtype is not dtype:
   1995     result = np.asarray(result, dtype=dtype)

File ~\anaconda3\Lib\site-packages\pandas\core\internals\
managers.py:1694, in BlockManager.as_array(self, dtype, copy,
na_value)
   1692         arr.flags.writeable = False
   1693 else:
-> 1694     arr = self._interleave(dtype=dtype, na_value=na_value)
   1695     # The underlying data was copied within _interleave, so no
need
   1696     # to further copy if copy=True or setting na_value
   1698 if na_value is lib.no_default:

File ~\anaconda3\Lib\site-packages\pandas\core\internals\
managers.py:1753, in BlockManager._interleave(self, dtype, na_value)
   1751     else:
   1752         arr = blk.get_values(dtype)
-> 1753     result[rl.indexer] = arr
   1754     itemmask[rl.indexer] = 1
   1756 if not itemmask.all():
```

```
ValueError: could not convert string to float: 'AFG'
```

```
sns.heatmap(df.corr(),annot=True)
plt.rcParams['figure.figsize']=(20,7)
plt.show() #shows correlation in heatmaps
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call
last)
Cell In[34], line 1
----> 1 sns.heatmap(df.corr(),annot=True)
      2 plt.rcParams['figure.figsize']=(20,7)
      3 plt.show()

File ~\anaconda3\Lib\site-packages\pandas\core\frame.py:11049, in
DataFrame.corr(self, method, min_periods, numeric_only)
  11047 cols = data.columns
  11048 idx = cols.copy()
> 11049 mat = data.to_numpy(dtype=float, na_value=np.nan, copy=False)
  11051 if method == "pearson":
  11052     correl = libalgos.nancorr(mat, minp=min_periods)

File ~\anaconda3\Lib\site-packages\pandas\core\frame.py:1993, in
DataFrame.to_numpy(self, dtype, copy, na_value)
   1991 if dtype is not None:
   1992     dtype = np.dtype(dtype)
-> 1993 result = self._mgr.as_array(dtype=dtype, copy=copy,
na_value=na_value)
   1994 if result.dtype is not dtype:
   1995     result = np.asarray(result, dtype=dtype)

File ~\anaconda3\Lib\site-packages\pandas\core\internals\
managers.py:1694, in BlockManager.as_array(self, dtype, copy,
na_value)
   1692         arr.flags.writeable = False
   1693 else:
-> 1694     arr = self._interleave(dtype=dtype, na_value=na_value)
   1695     # The underlying data was copied within _interleave, so no
need
   1696     # to further copy if copy=True or setting na_value
   1698 if na_value is lib.no_default:

File ~\anaconda3\Lib\site-packages\pandas\core\internals\
managers.py:1753, in BlockManager._interleave(self, dtype, na_value)
   1751     else:
   1752         arr = blk.get_values(dtype)
-> 1753     result[rl.indexer] = arr
   1754     itemmask[rl.indexer] = 1
```

```
   1756 if not itemmask.all():

ValueError: could not convert string to float: 'AFG'
```

df

```
     Rank CCA3            Country           Capital Continent  \
0      36  AFG        Afghanistan             Kabul      Asia
1     138  ALB            Albania            Tirana    Europe
2      34  DZA            Algeria           Algiers    Africa
3     213  ASM     American Samoa         Pago Pago   Oceania
4     203  AND            Andorra  Andorra la Vella    Europe
..    ...  ...                ...               ...       ...
229   226  WLF  Wallis and Futuna          Mata-Utu   Oceania
230   172  ESH     Western Sahara          El Aaiún    Africa
231    46  YEM              Yemen             Sanaa      Asia
232    63  ZMB             Zambia            Lusaka    Africa
233    74  ZWE           Zimbabwe            Harare    Africa

     2022 Population  2020 Population  2015 Population  2010
Population  \
0        41128771.00      38972230.00      33753499.00
28189672.00
1         2842321.00       2866849.00       2882481.00
2913399.00
2        44903225.00      43451666.00      39543154.00
35856344.00
3           44273.00         46189.00         51368.00
54849.00
4           79824.00         77700.00         71746.00
71519.00
..               ...              ...              ...               ..
.
229         11572.00         11655.00         12182.00
13142.00
230        575986.00        556048.00        491824.00
413296.00
231      33696614.00      32284046.00      28516545.00
24743946.00
232      20017675.00      18927715.00              NaN
13792086.00
233      16320537.00      15669666.00      14154937.00
12839771.00

     2000 Population  1990 Population  1980 Population  1970
Population  \
0        19542982.00      10694796.00      12486631.00
10752971.00
1         3182021.00       3295066.00       2941651.00
2324731.00
```

```
2        30774621.00      25518074.00      18739378.00
13795915.00
3           58230.00         47818.00         32886.00
27075.00
4           66097.00         53569.00         35611.00
19860.00
..               ...              ...              ...               ..
.
229         14723.00         13454.00         11315.00
9377.00
230        270375.00        178529.00        116775.00
76371.00
231      18628700.00      13375121.00       9204938.00
6843607.00
232       9891136.00       7686401.00       5720438.00
4281671.00
233      11834676.00      10113893.00       7049926.00
5202918.00

     Area (km²)  Density (per km²)  Growth Rate  World Population
Percentage
0     652230.00              63.06         1.03
0.52
1      28748.00              98.87         1.00
0.04
2    2381741.00              18.85         1.02
0.56
3        199.00             222.48         0.98
0.00
4        468.00             170.56         1.01
0.00
..          ...                ...          ...
...
229      142.00              81.49         1.00
0.00
230   266000.00               2.17         1.02
0.01
231   527968.00              63.82         1.02
0.42
232   752612.00              26.60         1.03
0.25
233   390757.00              41.77         1.02
0.20

[234 rows x 17 columns]

#df.groupby('Continent').mean() #gives all analysis
#df.groupby('Continent').mean().sort_values(by="2022
Population",ascending=False)
it gives mean of the popoulation in desc order
```

```
---------------------------------------------------------------------
-----
TypeError                                 Traceback (most recent call
last)
File ~\anaconda3\Lib\site-packages\pandas\core\groupby\
groupby.py:1942, in GroupBy._agg_py_fallback(self, how, values, ndim,
alt)
   1941 try:
-> 1942     res_values = self._grouper.agg_series(ser, alt,
preserve_dtype=True)
   1943 except Exception as err:

File ~\anaconda3\Lib\site-packages\pandas\core\groupby\ops.py:864, in
BaseGrouper.agg_series(self, obj, func, preserve_dtype)
    862     preserve_dtype = True
--> 864 result = self._aggregate_series_pure_python(obj, func)
    866 npvalues = lib.maybe_convert_objects(result, try_float=False)

File ~\anaconda3\Lib\site-packages\pandas\core\groupby\ops.py:885, in
BaseGrouper._aggregate_series_pure_python(self, obj, func)
    884 for i, group in enumerate(splitter):
--> 885     res = func(group)
    886     res = extract_result(res)

File ~\anaconda3\Lib\site-packages\pandas\core\groupby\
groupby.py:2454, in GroupBy.mean.<locals>.<lambda>(x)
   2451 else:
   2452     result = self._cython_agg_general(
   2453         "mean",
-> 2454         alt=lambda x: Series(x,
copy=False).mean(numeric_only=numeric_only),
   2455         numeric_only=numeric_only,
   2456     )
   2457     return result.__finalize__(self.obj, method="groupby")

File ~\anaconda3\Lib\site-packages\pandas\core\series.py:6549, in
Series.mean(self, axis, skipna, numeric_only, **kwargs)
   6541 @doc(make_doc("mean", ndim=1))
   6542 def mean(
   6543     self,
   (...)
   6547     **kwargs,
   6548 ):
-> 6549     return NDFrame.mean(self, axis, skipna, numeric_only,
**kwargs)

File ~\anaconda3\Lib\site-packages\pandas\core\generic.py:12420, in
NDFrame.mean(self, axis, skipna, numeric_only, **kwargs)
  12413 def mean(
  12414     self,
```

```
  12415     axis: Axis | None = 0,
   (...)
  12418     **kwargs,
  12419 ) -> Series | float:
> 12420     return self._stat_function(
  12421         "mean", nanops.nanmean, axis, skipna, numeric_only,
**kwargs
  12422     )

File ~\anaconda3\Lib\site-packages\pandas\core\generic.py:12377, in
NDFrame._stat_function(self, name, func, axis, skipna, numeric_only,
**kwargs)
  12375 validate_bool_kwarg(skipna, "skipna", none_allowed=False)
> 12377 return self._reduce(
  12378     func, name=name, axis=axis, skipna=skipna,
numeric_only=numeric_only
  12379 )

File ~\anaconda3\Lib\site-packages\pandas\core\series.py:6457, in
Series._reduce(self, op, name, axis, skipna, numeric_only,
filter_type, **kwds)
   6453     raise TypeError(
   6454         f"Series.{name} does not allow
{kwd_name}={numeric_only} "
   6455         "with non-numeric dtypes."
   6456     )
-> 6457 return op(delegate, skipna=skipna, **kwds)

File ~\anaconda3\Lib\site-packages\pandas\core\nanops.py:147, in
bottleneck_switch.__call__.<locals>.f(values, axis, skipna, **kwds)
    146 else:
--> 147     result = alt(values, axis=axis, skipna=skipna, **kwds)
    149 return result

File ~\anaconda3\Lib\site-packages\pandas\core\nanops.py:404, in
_datetimelike_compat.<locals>.new_func(values, axis, skipna, mask,
**kwargs)
    402     mask = isna(values)
--> 404 result = func(values, axis=axis, skipna=skipna, mask=mask,
**kwargs)
    406 if datetimelike:

File ~\anaconda3\Lib\site-packages\pandas\core\nanops.py:720, in
nanmean(values, axis, skipna, mask)
    719 the_sum = values.sum(axis, dtype=dtype_sum)
--> 720 the_sum = _ensure_numeric(the_sum)
    722 if axis is not None and getattr(the_sum, "ndim", False):

File ~\anaconda3\Lib\site-packages\pandas\core\nanops.py:1701, in
_ensure_numeric(x)
```

```
   1699 if isinstance(x, str):
   1700     # GH#44008, GH#36703 avoid casting e.g. strings to numeric
-> 1701     raise TypeError(f"Could not convert string '{x}' to
numeric")
   1702 try:

TypeError: Could not convert string
'DZAAGOBENBWABFABDICMRCPVCAFTCDCOMDJICODEGYGNQERISWZETHGABGMBGHAGINGNB
CIVKENLSOLBRLBYMDGMWIMLIMRTMUSMYTMARMOZNAMNERNGACOGREURWASTPSENSYCSLES
OMZAFSSDSDNTZATGOTUNUGAESHZMBZWE' to numeric

The above exception was the direct cause of the following exception:

TypeError                                 Traceback (most recent call
last)
Cell In[44], line 1
----> 1 df.groupby('Continent').mean() #gives all analysis
      2 df.groupby('Continent').mean().sort_values(by="2022
Population",ascending=False)

File ~\anaconda3\Lib\site-packages\pandas\core\groupby\
groupby.py:2452, in GroupBy.mean(self, numeric_only, engine,
engine_kwargs)
   2445     return self._numba_agg_general(
   2446         grouped_mean,
   2447         executor.float_dtype_mapping,
   2448         engine_kwargs,
   2449         min_periods=0,
   2450     )
   2451 else:
-> 2452     result = self._cython_agg_general(
   2453         "mean",
   2454         alt=lambda x: Series(x,
copy=False).mean(numeric_only=numeric_only),
   2455         numeric_only=numeric_only,
   2456     )
   2457     return result.__finalize__(self.obj, method="groupby")

File ~\anaconda3\Lib\site-packages\pandas\core\groupby\
groupby.py:1998, in GroupBy._cython_agg_general(self, how, alt,
numeric_only, min_count, **kwargs)
   1995     result = self._agg_py_fallback(how, values,
ndim=data.ndim, alt=alt)
   1996     return result
-> 1998 new_mgr = data.grouped_reduce(array_func)
   1999 res = self._wrap_agged_manager(new_mgr)
   2000 if how in ["idxmin", "idxmax"]:

File ~\anaconda3\Lib\site-packages\pandas\core\internals\
managers.py:1469, in BlockManager.grouped_reduce(self, func)
```

```
   1465 if blk.is_object:
   1466     # split on object-dtype blocks bc some columns may raise
   1467     #  while others do not.
   1468     for sb in blk._split():
-> 1469         applied = sb.apply(func)
   1470         result_blocks = extend_blocks(applied, result_blocks)
   1471 else:

File ~\anaconda3\Lib\site-packages\pandas\core\internals\
blocks.py:393, in Block.apply(self, func, **kwargs)
    387 @final
    388 def apply(self, func, **kwargs) -> list[Block]:
    389     """
    390     apply the function to my values; return a block if we are
not
    391     one
    392     """
--> 393     result = func(self.values, **kwargs)
    395     result = maybe_coerce_values(result)
    396     return self._split_op_result(result)

File ~\anaconda3\Lib\site-packages\pandas\core\groupby\
groupby.py:1995, in
GroupBy._cython_agg_general.<locals>.array_func(values)
   1992     return result
   1994 assert alt is not None
-> 1995 result = self._agg_py_fallback(how, values, ndim=data.ndim,
alt=alt)
   1996 return result

File ~\anaconda3\Lib\site-packages\pandas\core\groupby\
groupby.py:1946, in GroupBy._agg_py_fallback(self, how, values, ndim,
alt)
   1944     msg = f"agg function failed [how->{how},dtype-
>{ser.dtype}]"
   1945     # preserve the kind of exception that raised
-> 1946     raise type(err)(msg) from err
   1948 if ser.dtype == object:
   1949     res_values = res_values.astype(object, copy=False)

TypeError: agg function failed [how->mean,dtype->object]
```

```
df[df['Continent'].str.contains('Oceania')]
```

```
     Rank CCA3                  Country       Capital Continent  \
3     213  ASM           American Samoa     Pago Pago   Oceania
11     55  AUS                Australia      Canberra   Oceania
44    223  COK             Cook Islands        Avarua   Oceania
66    162  FJI                     Fiji          Suva   Oceania
70    183  PYF          French Polynesia       Papeete   Oceania
```

81	191	GUM	Guam	Hagåtña	Oceania
107	192	KIR	Kiribati	Tarawa	Oceania
126	215	MHL	Marshall Islands	Majuro	Oceania
132	194	FSM	Micronesia	Palikir	Oceania
142	225	NRU	Nauru	Yaren	Oceania
145	185	NCL	New Caledonia	Nouméa	Oceania
146	123	NZL	New Zealand	Wellington	Oceania
150	232	NIU	Niue	Alofi	Oceania
153	210	NFK	Northern Mariana Islands	Saipan	Oceania
157	222	PLW	Palau	Ngerulmud	Oceania
160	93	PNG	Papua New Guinea	Port Moresby	Oceania
179	188	WSM	Samoa	Apia	Oceania
191	166	SLB	Solomon Islands	Honiara	Oceania
209	233	TKL	Tokelau	Nukunonu	Oceania
210	197	TON	Tonga	Nuku‘alofa	Oceania
216	227	TUV	Tuvalu	Funafuti	Oceania
225	181	VUT	Vanuatu	Port-Vila	Oceania
229	226	WLF	Wallis and Futuna	Mata-Utu	Oceania

	2022 Population	2020 Population	2015 Population	2010 Population
3	44273.00	46189.00	51368.00	54849.00
11	26177413.00	25670051.00	23820236.00	22019168.00
44	17011.00	17029.00	17695.00	17212.00
66	929766.00	920422.00	917200.00	905169.00
70	306279.00	301920.00	291787.00	283788.00
81	171774.00	169231.00	167978.00	164905.00
107	131232.00	126463.00	116707.00	107995.00
126	41569.00	43413.00	49410.00	53416.00
132	114164.00	112106.00	109462.00	107588.00
142	12668.00	12315.00	11185.00	10241.00
145	289950.00	286403.00	283032.00	261426.00
146	5185288.00	5061133.00	4590590.00	4346338.00
150	1934.00	1942.00	1847.00	1812.00
153	49551.00	49587.00	51514.00	54087.00

```
157               NaN         17972.00         17794.00
18540.00
160      10142619.00       9749640.00       8682174.00
7583269.00
179        222382.00        214929.00        203571.00
194672.00
191        724273.00        691191.00        612660.00
540394.00
209          1871.00          1827.00          1454.00
1367.00
210        106858.00        105254.00        106122.00
107383.00
216         11312.00         11069.00         10877.00
10550.00
225        326740.00        311685.00        276438.00
245453.00
229         11572.00         11655.00         12182.00
13142.00

     2000 Population  1990 Population  1980 Population  1970
Population  \
3           58230.00         47818.00         32886.00
27075.00
11       19017963.00      17048003.00      14706322.00
12595034.00
44          15897.00         17123.00         17651.00
20470.00
66         832509.00        780430.00        644582.00
527634.00
70         250927.00        211089.00        163591.00
117891.00
81         160188.00        138263.00        110286.00
88300.00
107         88826.00         75124.00         60813.00
57437.00
126         54224.00         46047.00         31988.00
23969.00
132        111709.00         98603.00         76299.00
58989.00
142         10377.00          9598.00          7635.00
6663.00
145        221537.00        177264.00        148599.00
110982.00
146       3855266.00       3397389.00       3147168.00
2824061.00
150          2074.00          2533.00          3637.00
5185.00
153         80338.00         48002.00         17613.00
10143.00
```

```
157         19726.00         15293.00         12252.00
11366.00
160       5508297.00       3864972.00       3104788.00
2489059.00
179        184008.00        168186.00        164905.00
142771.00
191        429978.00        324171.00        233668.00
172833.00
209          1666.00          1669.00          1647.00
1714.00
210        102603.00         98727.00         96708.00
86484.00
216          9638.00          9182.00          7731.00
5814.00
225        192074.00        150882.00        118156.00
87019.00
229         14723.00         13454.00         11315.00
9377.00

     Area (km²)  Density (per km²)  Growth Rate  World Population
Percentage
3        199.00             222.48         0.98
0.00
11   7692024.00               3.40         1.01
0.33
44       236.00              72.08         1.00
0.00
66     18272.00              50.88         1.01
0.01
70      4167.00              73.50         1.01
0.00
81       549.00             312.89         1.01
0.00
107      811.00             161.81         1.02
0.00
126      181.00             229.66         0.99
0.00
132      702.00             162.63         1.01
0.00
142       21.00             603.24         1.01
0.00
145    18575.00              15.61         1.01
0.00
146   270467.00              19.17         1.01
0.07
150      260.00               7.44         1.00
0.00
153      464.00             106.79         1.00
0.00
```

```
157      459.00               39.34          1.00
0.00
160   462840.00               21.91          1.02
0.13
179     2842.00               78.25          1.02
0.00
191    28896.00               25.06          1.02
0.01
209       12.00              155.92          1.01
0.00
210      747.00              143.05          1.01
0.00
216       26.00              435.08          1.01
0.00
225    12189.00               26.81          1.02
0.00
229      142.00               81.49          1.00
0.00
```

```
df2=df.groupby('Continent').mean().sort_values(by="2022
Population",ascending=False)
df2.transpose#for visualization
df2.plot()#to visualize trends
```

```
---------------------------------------------------------------------------
-----
TypeError                                 Traceback (most recent call
last)
File ~\anaconda3\Lib\site-packages\pandas\core\groupby\
groupby.py:1942, in GroupBy._agg_py_fallback(self, how, values, ndim,
alt)
   1941 try:
-> 1942     res_values = self._grouper.agg_series(ser, alt,
preserve_dtype=True)
   1943 except Exception as err:

File ~\anaconda3\Lib\site-packages\pandas\core\groupby\ops.py:864, in
BaseGrouper.agg_series(self, obj, func, preserve_dtype)
    862     preserve_dtype = True
--> 864 result = self._aggregate_series_pure_python(obj, func)
    866 npvalues = lib.maybe_convert_objects(result, try_float=False)

File ~\anaconda3\Lib\site-packages\pandas\core\groupby\ops.py:885, in
BaseGrouper._aggregate_series_pure_python(self, obj, func)
    884 for i, group in enumerate(splitter):
--> 885     res = func(group)
    886     res = extract_result(res)

File ~\anaconda3\Lib\site-packages\pandas\core\groupby\
groupby.py:2454, in GroupBy.mean.<locals>.<lambda>(x)
```

```
   2451 else:
   2452     result = self._cython_agg_general(
   2453         "mean",
-> 2454         alt=lambda x: Series(x,
copy=False).mean(numeric_only=numeric_only),
   2455         numeric_only=numeric_only,
   2456     )
   2457     return result.__finalize__(self.obj, method="groupby")

File ~\anaconda3\Lib\site-packages\pandas\core\series.py:6549, in
Series.mean(self, axis, skipna, numeric_only, **kwargs)
   6541 @doc(make_doc("mean", ndim=1))
   6542 def mean(
   6543     self,
   (...)
   6547     **kwargs,
   6548 ):
-> 6549     return NDFrame.mean(self, axis, skipna, numeric_only,
**kwargs)

File ~\anaconda3\Lib\site-packages\pandas\core\generic.py:12420, in
NDFrame.mean(self, axis, skipna, numeric_only, **kwargs)
  12413 def mean(
  12414     self,
  12415     axis: Axis | None = 0,
   (...)
  12418     **kwargs,
  12419 ) -> Series | float:
> 12420     return self._stat_function(
  12421         "mean", nanops.nanmean, axis, skipna, numeric_only,
**kwargs
  12422     )

File ~\anaconda3\Lib\site-packages\pandas\core\generic.py:12377, in
NDFrame._stat_function(self, name, func, axis, skipna, numeric_only,
**kwargs)
  12375 validate_bool_kwarg(skipna, "skipna", none_allowed=False)
> 12377 return self._reduce(
  12378     func, name=name, axis=axis, skipna=skipna,
numeric_only=numeric_only
  12379 )

File ~\anaconda3\Lib\site-packages\pandas\core\series.py:6457, in
Series._reduce(self, op, name, axis, skipna, numeric_only,
filter_type, **kwds)
   6453     raise TypeError(
   6454         f"Series.{name} does not allow
{kwd_name}={numeric_only} "
   6455         "with non-numeric dtypes."
   6456     )
```

```
-> 6457 return op(delegate, skipna=skipna, **kwds)

File ~\anaconda3\Lib\site-packages\pandas\core\nanops.py:147, in
bottleneck_switch.__call__.<locals>.f(values, axis, skipna, **kwds)
    146 else:
--> 147     result = alt(values, axis=axis, skipna=skipna, **kwds)
    149 return result

File ~\anaconda3\Lib\site-packages\pandas\core\nanops.py:404, in
_datetimelike_compat.<locals>.new_func(values, axis, skipna, mask,
**kwargs)
    402     mask = isna(values)
--> 404 result = func(values, axis=axis, skipna=skipna, mask=mask,
**kwargs)
    406 if datetimelike:

File ~\anaconda3\Lib\site-packages\pandas\core\nanops.py:720, in
nanmean(values, axis, skipna, mask)
    719 the_sum = values.sum(axis, dtype=dtype_sum)
--> 720 the_sum = _ensure_numeric(the_sum)
    722 if axis is not None and getattr(the_sum, "ndim", False):

File ~\anaconda3\Lib\site-packages\pandas\core\nanops.py:1701, in
_ensure_numeric(x)
   1699 if isinstance(x, str):
   1700     # GH#44008, GH#36703 avoid casting e.g. strings to numeric
-> 1701     raise TypeError(f"Could not convert string '{x}' to
numeric")
   1702 try:

TypeError: Could not convert string
'DZAAGOBENBWABFABDICMRCPVCAFTCDCOMDJICODEGYGNQERISWZETHGABGMBGHAGINGNB
CIVKENLSOLBRLBYMDGMWIMLIMRTMUSMYTMARMOZNAMNERNGACOGREURWASTPSENSYCSLES
OMZAFSSDSDNTZATGOTUNUGAESHZMBZWE' to numeric

The above exception was the direct cause of the following exception:

TypeError                                 Traceback (most recent call
last)
Cell In[46], line 1
----> 1 df2=df.groupby('Continent').mean().sort_values(by="2022
Population",ascending=False)
      2 df2.transpose#for visualization
      3 df2.plot()

File ~\anaconda3\Lib\site-packages\pandas\core\groupby\
groupby.py:2452, in GroupBy.mean(self, numeric_only, engine,
engine_kwargs)
   2445     return self._numba_agg_general(
   2446         grouped_mean,
```

```
   2447         executor.float_dtype_mapping,
   2448         engine_kwargs,
   2449         min_periods=0,
   2450     )
   2451 else:
-> 2452     result = self._cython_agg_general(
   2453         "mean",
   2454         alt=lambda x: Series(x,
copy=False).mean(numeric_only=numeric_only),
   2455         numeric_only=numeric_only,
   2456     )
   2457     return result.__finalize__(self.obj, method="groupby")

File ~\anaconda3\Lib\site-packages\pandas\core\groupby\
groupby.py:1998, in GroupBy._cython_agg_general(self, how, alt,
numeric_only, min_count, **kwargs)
   1995     result = self._agg_py_fallback(how, values,
ndim=data.ndim, alt=alt)
   1996     return result
-> 1998 new_mgr = data.grouped_reduce(array_func)
   1999 res = self._wrap_agged_manager(new_mgr)
   2000 if how in ["idxmin", "idxmax"]:

File ~\anaconda3\Lib\site-packages\pandas\core\internals\
managers.py:1469, in BlockManager.grouped_reduce(self, func)
   1465 if blk.is_object:
   1466     # split on object-dtype blocks bc some columns may raise
   1467     #  while others do not.
   1468     for sb in blk._split():
-> 1469         applied = sb.apply(func)
   1470         result_blocks = extend_blocks(applied, result_blocks)
   1471 else:

File ~\anaconda3\Lib\site-packages\pandas\core\internals\
blocks.py:393, in Block.apply(self, func, **kwargs)
    387 @final
    388 def apply(self, func, **kwargs) -> list[Block]:
    389     """
    390     apply the function to my values; return a block if we are
not
    391     one
    392     """
--> 393     result = func(self.values, **kwargs)
    395     result = maybe_coerce_values(result)
    396     return self._split_op_result(result)

File ~\anaconda3\Lib\site-packages\pandas\core\groupby\
groupby.py:1995, in
GroupBy._cython_agg_general.<locals>.array_func(values)
   1992     return result
```

```
   1994 assert alt is not None
-> 1995 result = self._agg_py_fallback(how, values, ndim=data.ndim,
alt=alt)
   1996 return result

File ~\anaconda3\Lib\site-packages\pandas\core\groupby\
groupby.py:1946, in GroupBy._agg_py_fallback(self, how, values, ndim,
alt)
   1944     msg = f"agg function failed [how->{how},dtype-
>{ser.dtype}]"
   1945     # preserve the kind of exception that raised
-> 1946     raise type(err)(msg) from err
   1948 if ser.dtype == object:
   1949     res_values = res_values.astype(object, copy=False)

TypeError: agg function failed [how->mean,dtype->object]
```

```
df.columns
```

```
Index(['Rank', 'CCA3', 'Country', 'Capital', 'Continent', '2022
Population',
       '2020 Population', '2015 Population', '2010 Population',
       '2000 Population', '1990 Population', '1980 Population',
       '1970 Population', 'Area (km²)', 'Density (per km²)', 'Growth
Rate',
       'World Population Percentage'],
      dtype='object')
```

```
df2=df.groupby('Continent')
[df.columns[5:13]].mean().sort_values(by="2022
Population",ascending=False)
#because u want to analyze only population not all columns

df.boxplot() #use boxplot for outliers
```

```
<Axes: >
```

```
df
```

```
     Rank CCA3            Country           Capital Continent  \
0      36  AFG        Afghanistan             Kabul      Asia
1     138  ALB            Albania            Tirana    Europe
2      34  DZA            Algeria           Algiers    Africa
3     213  ASM     American Samoa         Pago Pago   Oceania
4     203  AND            Andorra  Andorra la Vella    Europe
..    ...  ...                ...               ...       ...
229   226  WLF  Wallis and Futuna          Mata-Utu   Oceania
230   172  ESH     Western Sahara          El Aaiún    Africa
231    46  YEM              Yemen             Sanaa      Asia
232    63  ZMB             Zambia            Lusaka    Africa
233    74  ZWE           Zimbabwe            Harare    Africa
```

```
     2022 Population  2020 Population  2015 Population  2010
Population  \
0        41128771.00      38972230.00      33753499.00
28189672.00
1         2842321.00       2866849.00       2882481.00
2913399.00
2        44903225.00      43451666.00      39543154.00
35856344.00
3           44273.00         46189.00         51368.00
54849.00
4           79824.00         77700.00         71746.00
71519.00
..               ...              ...              ...               ..
.
229         11572.00         11655.00         12182.00
13142.00
230        575986.00        556048.00        491824.00
413296.00
231      33696614.00      32284046.00      28516545.00
24743946.00
232      20017675.00      18927715.00              NaN
13792086.00
233      16320537.00      15669666.00      14154937.00
12839771.00

     2000 Population  1990 Population  1980 Population  1970
Population  \
0        19542982.00      10694796.00      12486631.00
10752971.00
1         3182021.00       3295066.00       2941651.00
2324731.00
2        30774621.00      25518074.00      18739378.00
13795915.00
3           58230.00         47818.00         32886.00
27075.00
4           66097.00         53569.00         35611.00
19860.00
..               ...              ...              ...               ..
.
229         14723.00         13454.00         11315.00
9377.00
230        270375.00        178529.00        116775.00
76371.00
231      18628700.00      13375121.00       9204938.00
6843607.00
232       9891136.00       7686401.00       5720438.00
4281671.00
233      11834676.00      10113893.00       7049926.00
```

```
5202918.00

     Area (km²)  Density (per km²)  Growth Rate  World Population
Percentage
0     652230.00              63.06         1.03
0.52
1      28748.00              98.87         1.00
0.04
2    2381741.00              18.85         1.02
0.56
3        199.00             222.48         0.98
0.00
4        468.00             170.56         1.01
0.00
..          ...                ...          ...
...
229      142.00              81.49         1.00
0.00
230   266000.00               2.17         1.02
0.01
231   527968.00              63.82         1.02
0.42
232   752612.00              26.60         1.03
0.25
233   390757.00              41.77         1.02
0.20

[234 rows x 17 columns]
```

```
df.dtypes
```

```
Rank                             int64
CCA3                            object
Country                         object
Capital                         object
Continent                       object
2022 Population                float64
2020 Population                float64
2015 Population                float64
2010 Population                float64
2000 Population                float64
1990 Population                float64
1980 Population                float64
1970 Population                float64
Area (km²)                     float64
Density (per km²)              float64
Growth Rate                    float64
World Population Percentage    float64
dtype: object
```

```
df.select_dtypes(include='number')
```

```
     Rank  2022 Population  2020 Population  2015 Population  2010
Population  \
0      36      41128771.00      38972230.00      33753499.00
28189672.00
1     138       2842321.00       2866849.00       2882481.00
2913399.00
2      34      44903225.00      43451666.00      39543154.00
35856344.00
3     213         44273.00         46189.00         51368.00
54849.00
4     203         79824.00         77700.00         71746.00
71519.00
..    ...              ...              ...              ...
...
229   226         11572.00         11655.00         12182.00
13142.00
230   172        575986.00        556048.00        491824.00
413296.00
231    46      33696614.00      32284046.00      28516545.00
24743946.00
232    63      20017675.00      18927715.00              NaN
13792086.00
233    74      16320537.00      15669666.00      14154937.00
12839771.00

     2000 Population  1990 Population  1980 Population  1970
Population  \
0        19542982.00      10694796.00      12486631.00
10752971.00
1         3182021.00       3295066.00       2941651.00
2324731.00
2        30774621.00      25518074.00      18739378.00
13795915.00
3           58230.00         47818.00         32886.00
27075.00
4           66097.00         53569.00         35611.00
19860.00
..               ...              ...              ...               ..
.
229         14723.00         13454.00         11315.00
9377.00
230        270375.00        178529.00        116775.00
76371.00
231      18628700.00      13375121.00       9204938.00
6843607.00
232       9891136.00       7686401.00       5720438.00
4281671.00
233      11834676.00      10113893.00       7049926.00
```

```
5202918.00

     Area (km²)  Density (per km²)  Growth Rate  World Population
Percentage
0     652230.00              63.06         1.03
0.52
1      28748.00              98.87         1.00
0.04
2    2381741.00              18.85         1.02
0.56
3        199.00             222.48         0.98
0.00
4        468.00             170.56         1.01
0.00
..          ...                ...          ...
...
229      142.00              81.49         1.00
0.00
230   266000.00               2.17         1.02
0.01
231   527968.00              63.82         1.02
0.42
232   752612.00              26.60         1.03
0.25
233   390757.00              41.77         1.02
0.20

[234 rows x 13 columns]
```

```
df.select_dtypes(include='object')
```

```
    CCA3            Country           Capital Continent
0    AFG        Afghanistan             Kabul      Asia
1    ALB            Albania            Tirana    Europe
2    DZA            Algeria           Algiers    Africa
3    ASM     American Samoa         Pago Pago   Oceania
4    AND            Andorra  Andorra la Vella    Europe
..   ...                ...               ...       ...
229  WLF  Wallis and Futuna          Mata-Utu   Oceania
230  ESH     Western Sahara          El Aaiún    Africa
231  YEM              Yemen             Sanaa      Asia
232  ZMB             Zambia            Lusaka    Africa
233  ZWE           Zimbabwe            Harare    Africa

[234 rows x 4 columns]
```

```
df.select_dtypes(include='float')
```

```
     2022 Population  2020 Population  2015 Population  2010
Population  \
```

```
0        41128771.00      38972230.00      33753499.00
28189672.00
1         2842321.00       2866849.00       2882481.00
2913399.00
2        44903225.00      43451666.00      39543154.00
35856344.00
3           44273.00         46189.00         51368.00
54849.00
4           79824.00         77700.00         71746.00
71519.00
..               ...              ...              ...               ..
.
229         11572.00         11655.00         12182.00
13142.00
230        575986.00        556048.00        491824.00
413296.00
231      33696614.00      32284046.00      28516545.00
24743946.00
232      20017675.00      18927715.00              NaN
13792086.00
233      16320537.00      15669666.00      14154937.00
12839771.00

     2000 Population  1990 Population  1980 Population  1970
Population  \
0        19542982.00      10694796.00      12486631.00
10752971.00
1         3182021.00       3295066.00       2941651.00
2324731.00
2        30774621.00      25518074.00      18739378.00
13795915.00
3           58230.00         47818.00         32886.00
27075.00
4           66097.00         53569.00         35611.00
19860.00
..               ...              ...              ...               ..
.
229         14723.00         13454.00         11315.00
9377.00
230        270375.00        178529.00        116775.00
76371.00
231      18628700.00      13375121.00       9204938.00
6843607.00
232       9891136.00       7686401.00       5720438.00
4281671.00
233      11834676.00      10113893.00       7049926.00
5202918.00

     Area (km²)  Density (per km²)  Growth Rate  World Population
```

```
Percentage
0     652230.00               63.06          1.03
0.52
1      28748.00               98.87          1.00
0.04
2    2381741.00               18.85          1.02
0.56
3        199.00              222.48          0.98
0.00
4        468.00              170.56          1.01
0.00
..          ...                 ...           ...
...
229      142.00               81.49          1.00
0.00
230   266000.00                2.17          1.02
0.01
231   527968.00               63.82          1.02
0.42
232   752612.00               26.60          1.03
0.25
233   390757.00               41.77          1.02
0.20

[234 rows x 12 columns]
```