In [1]:

```
!pip install pandas
```

```
Requirement already satisfied: pandas in /Users/anantm/opt/anaconda3/l
ib/python3.8/site-packages (1.2.4)
Requirement already satisfied: python-dateutil>=2.7.3 in /Users/anant
m/opt/anaconda3/lib/python3.8/site-packages (from pandas) (2.8.1)
Requirement already satisfied: pytz>=2017.3 in /Users/anantm/opt/anaco
nda3/lib/python3.8/site-packages (from pandas) (2021.1)
Requirement already satisfied: numpy>=1.16.5 in /Users/anantm/opt/anac
onda3/lib/python3.8/site-packages (from pandas) (1.22.2)
Requirement already satisfied: six>=1.5 in /Users/anantm/opt/anaconda
3/lib/python3.8/site-packages (from python-dateutil>=2.7.3->pandas)
(1.15.0)
```

In [1]:

```
import pandas as pd
```

In [2]:

```
df = pd.read_csv("data/gapminder.csv")
```

In [4]:

```
df
```

Out[4]:

	country	continent	year	lifeExp	population	gdpPerCap
0	Afghanistan	Asia	1952	28.801	8425333	779.445314
1	Afghanistan	Asia	1957	30.332	9240934	820.853030
2	Afghanistan	Asia	1962	31.997	10267083	853.100710
3	Afghanistan	Asia	1967	34.020	11537966	836.197138
4	Afghanistan	Asia	1972	36.088	13079460	739.981106
...	...	...	...	...	...	...
1699	Zimbabwe	Africa	1987	62.351	9216418	706.157306
1700	Zimbabwe	Africa	1992	60.377	10704340	693.420786
1701	Zimbabwe	Africa	1997	46.809	11404948	792.449960
1702	Zimbabwe	Africa	2002	39.989	11926563	672.038623
1703	Zimbabwe	Africa	2007	43.487	12311143	469.709298

1704 rows × 6 columns

In [5]:

```
type(df)
```

Out[5]:

```
pandas.core.frame.DataFrame
```

In [6]:

```
# dictionary - specialised dictionary
```

In [7]:

```
df["country"]
```

Out[7]:

```
0       Afghanistan
1       Afghanistan
2       Afghanistan
3       Afghanistan
4       Afghanistan
           ...
1699       Zimbabwe
1700       Zimbabwe
1701       Zimbabwe
1702       Zimbabwe
1703       Zimbabwe
Name: country, Length: 1704, dtype: object
```

In [8]:

```
df[["country", "continent"]]
```

Out[8]:

	country	continent
0	Afghanistan	Asia
1	Afghanistan	Asia
2	Afghanistan	Asia
3	Afghanistan	Asia
4	Afghanistan	Asia
...	...	...
1699	Zimbabwe	Africa
1700	Zimbabwe	Africa
1701	Zimbabwe	Africa
1702	Zimbabwe	Africa
1703	Zimbabwe	Africa

1704 rows × 2 columns

In [10]:

```
type(df["country"])
```

Out[10]:

```
pandas.core.series.Series
```

In [12]:

```
1 df.head(9)
```

Out[12]:

	country	continent	year	lifeExp	population	gdpPerCap
0	Afghanistan	Asia	1952	28.801	8425333	779.445314
1	Afghanistan	Asia	1957	30.332	9240934	820.853030
2	Afghanistan	Asia	1962	31.997	10267083	853.100710
3	Afghanistan	Asia	1967	34.020	11537966	836.197138
4	Afghanistan	Asia	1972	36.088	13079460	739.981106
5	Afghanistan	Asia	1977	38.438	14880372	786.113360
6	Afghanistan	Asia	1982	39.854	12881816	978.011439
7	Afghanistan	Asia	1987	40.822	13867957	852.395945
8	Afghanistan	Asia	1992	41.674	16317921	649.341395

In [13]:

```
1 df.tail(20)
```

Out[13]:

	country	continent	year	lifeExp	population	gdpPerCap
1684	Zambia	Africa	1972	50.107	4506497	1773.498265
1685	Zambia	Africa	1977	51.386	5216550	1588.688299
1686	Zambia	Africa	1982	51.821	6100407	1408.678565
1687	Zambia	Africa	1987	50.821	7272406	1213.315116
1688	Zambia	Africa	1992	46.100	8381163	1210.884633
1689	Zambia	Africa	1997	40.238	9417789	1071.353818
1690	Zambia	Africa	2002	39.193	10595811	1071.613938
1691	Zambia	Africa	2007	42.384	11746035	1271.211593
1692	Zimbabwe	Africa	1952	48.451	3080907	406.884115
1693	Zimbabwe	Africa	1957	50.469	3646340	518.764268
1694	Zimbabwe	Africa	1962	52.358	4277736	527.272182
1695	Zimbabwe	Africa	1967	53.995	4995432	569.795071
1696	Zimbabwe	Africa	1972	55.635	5861135	799.362176
1697	Zimbabwe	Africa	1977	57.674	6642107	685.587682
1698	Zimbabwe	Africa	1982	60.363	7636524	788.855041
1699	Zimbabwe	Africa	1987	62.351	9216418	706.157306
1700	Zimbabwe	Africa	1992	60.377	10704340	693.420786
1701	Zimbabwe	Africa	1997	46.809	11404948	792.449960
1702	Zimbabwe	Africa	2002	39.989	11926563	672.038623
1703	Zimbabwe	Africa	2007	43.487	12311143	469.709298

In [14]:

```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1704 entries, 0 to 1703
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   country     1704 non-null   object 
 1   continent   1704 non-null   object 
 2   year        1704 non-null   int64  
 3   lifeExp     1704 non-null   float64
 4   population  1704 non-null   int64  
 5   gdpPerCap   1704 non-null   float64
dtypes: float64(2), int64(2), object(2)
memory usage: 80.0+ KB
```

In [15]:

```
1 df.describe()
```

Out[15]:

	year	lifeExp	population	gdpPerCap
count	1704.00000	1704.000000	1.704000e+03	1704.000000
mean	1979.50000	59.474439	2.960121e+07	7215.327081
std	17.26533	12.917107	1.061579e+08	9857.454543
min	1952.00000	23.599000	6.001100e+04	241.165876
25%	1965.75000	48.198000	2.793664e+06	1202.060309
50%	1979.50000	60.712500	7.023596e+06	3531.846988
75%	1993.25000	70.845500	1.958522e+07	9325.462346
max	2007.00000	82.603000	1.318683e+09	113523.132900

In [19]:

```
1 df.describe(include="object")
```

Out[19]:

	country	continent
count	1704	1704
unique	142	5
top	Israel	Africa
freq	12	624

In [20]:

```
1 df.index[4]
```

Out[20]:

4

In [21]:

```
1 df.index[3]
```

Out[21]:

3

In [22]:

```
df
```

Out[22]:

	country	continent	year	lifeExp	population	gdpPerCap
0	Afghanistan	Asia	1952	28.801	8425333	779.445314
1	Afghanistan	Asia	1957	30.332	9240934	820.853030
2	Afghanistan	Asia	1962	31.997	10267083	853.100710
3	Afghanistan	Asia	1967	34.020	11537966	836.197138
4	Afghanistan	Asia	1972	36.088	13079460	739.981106
...	...	...	...	...	...	...
1699	Zimbabwe	Africa	1987	62.351	9216418	706.157306
1700	Zimbabwe	Africa	1992	60.377	10704340	693.420786
1701	Zimbabwe	Africa	1997	46.809	11404948	792.449960
1702	Zimbabwe	Africa	2002	39.989	11926563	672.038623
1703	Zimbabwe	Africa	2007	43.487	12311143	469.709298

1704 rows × 6 columns

In [24]:

```
df["country"][2]
```

Out[24]:

```
'Afghanistan'
```

In [27]:

```
df_temp = df.set_index("continent")
```

In [28]:

```
df_temp["country"]["Asia"] # this is okay to use
```

Out[28]:

```
continent
Asia    Afghanistan
Asia    Afghanistan
Asia    Afghanistan
Asia    Afghanistan
Asia    Afghanistan
           ...
Asia    Yemen, Rep.
Asia    Yemen, Rep.
Asia    Yemen, Rep.
Asia    Yemen, Rep.
Asia    Yemen, Rep.
Name: country, Length: 396, dtype: object
```

In [31]:

```
df_temp["country"][2] # not to use it
```

Out[31]:

```
'Afghanistan'
```

In [34]:

```
df_temp["country"]
```

Out[34]:

```
continent
Asia      Afghanistan
Asia      Afghanistan
Asia      Afghanistan
Asia      Afghanistan
Asia      Afghanistan
             ...
Africa       Zimbabwe
Africa       Zimbabwe
Africa       Zimbabwe
Africa       Zimbabwe
Africa       Zimbabwe
Name: country, Length: 1704, dtype: object
```

In [35]:

```
# columns
```

In [36]:

```
df.columns
```

Out[36]:

```
Index(['country', 'continent', 'year', 'lifeExp', 'population', 'gdpPe
rCap'], dtype='object')
```

In [37]:

```
df.keys()
```

Out[37]:

```
Index(['country', 'continent', 'year', 'lifeExp', 'population', 'gdpPe
rCap'], dtype='object')
```

In [39]:

```
df[["country", "continent"]]
```

Out[39]:

	country	continent
0	Afghanistan	Asia
1	Afghanistan	Asia
2	Afghanistan	Asia
3	Afghanistan	Asia
4	Afghanistan	Asia
...	...	...
1699	Zimbabwe	Africa
1700	Zimbabwe	Africa
1701	Zimbabwe	Africa
1702	Zimbabwe	Africa
1703	Zimbabwe	Africa

1704 rows × 2 columns

In [40]:

```
df.country # attrubite style references
```

Out[40]:

```
0       Afghanistan
1       Afghanistan
2       Afghanistan
3       Afghanistan
4       Afghanistan
           ...
1699       Zimbabwe
1700       Zimbabwe
1701       Zimbabwe
1702       Zimbabwe
1703       Zimbabwe
Name: country, Length: 1704, dtype: object
```

In [41]:

```
1 df.continent
```

Out[41]:

```
0          Asia
1          Asia
2          Asia
3          Asia
4          Asia
          ...
1699     Africa
1700     Africa
1701     Africa
1702     Africa
1703     Africa
Name: continent, Length: 1704, dtype: object
```

In [42]:

```
1 df.country is df["country"]
```

Out[42]:

```
True
```

In [44]:

```
1 df.drop("continent", axis=1)
```

Out[44]:

	country	year	lifeExp	population	gdpPerCap
0	Afghanistan	1952	28.801	8425333	779.445314
1	Afghanistan	1957	30.332	9240934	820.853030
2	Afghanistan	1962	31.997	10267083	853.100710
3	Afghanistan	1967	34.020	11537966	836.197138
4	Afghanistan	1972	36.088	13079460	739.981106
...	...	...	...	...	...
1699	Zimbabwe	1987	62.351	9216418	706.157306
1700	Zimbabwe	1992	60.377	10704340	693.420786
1701	Zimbabwe	1997	46.809	11404948	792.449960
1702	Zimbabwe	2002	39.989	11926563	672.038623
1703	Zimbabwe	2007	43.487	12311143	469.709298

1704 rows × 5 columns

In [46]:

```
1 df.drop("continent", axis=1, inplace=True)
```

In [47]:

```
1 df
```

Out[47]:

	country	year	lifeExp	population	gdpPerCap
0	Afghanistan	1952	28.801	8425333	779.445314
1	Afghanistan	1957	30.332	9240934	820.853030
2	Afghanistan	1962	31.997	10267083	853.100710
3	Afghanistan	1967	34.020	11537966	836.197138
4	Afghanistan	1972	36.088	13079460	739.981106
...	...	...	...	...	...
1699	Zimbabwe	1987	62.351	9216418	706.157306
1700	Zimbabwe	1992	60.377	10704340	693.420786
1701	Zimbabwe	1997	46.809	11404948	792.449960
1702	Zimbabwe	2002	39.989	11926563	672.038623
1703	Zimbabwe	2007	43.487	12311143	469.709298

1704 rows × 5 columns

In [49]:

```
1 df = pd.read_csv("data/gapminder.csv")
2 df
```

Out[49]:

	country	continent	year	lifeExp	population	gdpPerCap
0	Afghanistan	Asia	1952	28.801	8425333	779.445314
1	Afghanistan	Asia	1957	30.332	9240934	820.853030
2	Afghanistan	Asia	1962	31.997	10267083	853.100710
3	Afghanistan	Asia	1967	34.020	11537966	836.197138
4	Afghanistan	Asia	1972	36.088	13079460	739.981106
...	...	...	...	...	...	...
1699	Zimbabwe	Africa	1987	62.351	9216418	706.157306
1700	Zimbabwe	Africa	1992	60.377	10704340	693.420786
1701	Zimbabwe	Africa	1997	46.809	11404948	792.449960
1702	Zimbabwe	Africa	2002	39.989	11926563	672.038623
1703	Zimbabwe	Africa	2007	43.487	12311143	469.709298

1704 rows × 6 columns

In [50]:

```
# iloc and loc
```

In [52]:

```
ser = df["country"]
ser
```

Out[52]:

```
0       Afghanistan
1       Afghanistan
2       Afghanistan
3       Afghanistan
4       Afghanistan
           ...
1699       Zimbabwe
1700       Zimbabwe
1701       Zimbabwe
1702       Zimbabwe
1703       Zimbabwe
Name: country, Length: 1704, dtype: object
```

In [55]:

```
ser[3] # indexing
```

Out[55]:

```
'Afghanistan'
```

In [56]:

```
ser[3:15]
```

Out[56]:

```
3     Afghanistan
4     Afghanistan
5     Afghanistan
6     Afghanistan
7     Afghanistan
8     Afghanistan
9     Afghanistan
10    Afghanistan
11    Afghanistan
12        Albania
13        Albania
14        Albania
Name: country, dtype: object
```

In [58]:

```
data = pd.Series(['a', 'b', 'c'], index=[1, 5, 3])
data
```

Out[58]:

```
1    a
5    b
3    c
dtype: object
```

In [59]:

```
data[1] # using explicit index
```

Out[59]:

```
'a'
```

In [60]:

```
data[1:3] # using implicity indices
```

Out[60]:

```
5    b
3    c
dtype: object
```

In [61]:

```
# indexing - explicit
# slicing - implcit
# iloc and loc
# loc - labels
# iloc - positional index, implicit indices
```

In [62]:

```
data.iloc[1]
```

Out[62]:

```
'b'
```

In [63]:

```
data.loc[1]
```

Out[63]:

```
'a'
```

In [65]:

```
1 data.iloc[1:3]
```

Out[65]:

```
5    b
3    c
dtype: object
```

In [66]:

```
1 data.loc[1:3]
```

Out[66]:

```
1    a
5    b
3    c
dtype: object
```

In [67]:

```
1 data.loc[2:3]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call
 last)
~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexes/base.p
y in get_loc(self, key, method, tolerance)
   3079             try:
-> 3080                 return self._engine.get_loc(casted_key)
   3081             except KeyError as err:

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.Int6
4HashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.Int6
4HashTable.get_item()

KeyError: 2

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

KeyError                                  Traceback (most recent call
 last)
<ipython-input-67-29f1ad944393> in <module>
----> 1 data.loc[2:3]

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py in
__getitem__(self, key)
    893
    894             maybe_callable = com.apply_if_callable(key, self.o
bj)
--> 895             return self._getitem_axis(maybe_callable, axis=axi
s)
    896
    897     def _is_scalar_access(self, key: Tuple):

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py in
_getitem_axis(self, key, axis)
   1100         if isinstance(key, slice):
   1101             self._validate_key(key, axis)
-> 1102             return self._get_slice_axis(key, axis=axis)
   1103         elif com.is_bool_indexer(key):
   1104             return self._getbool_axis(key, axis=axis)

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py in
_get_slice_axis(self, slice_obj, axis)
   1134
   1135         labels = obj._get_axis(axis)
-> 1136         indexer = labels.slice_indexer(
   1137             slice_obj.start, slice_obj.stop, slice_obj.step, k
ind="loc"
   1138         )

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexes/base.p
```

```
y in slice_indexer(self, start, end, step, kind)
   5275         slice(1, 3, None)
   5276         """
-> 5277         start_slice, end_slice = self.slice_locs(start, end, s
tep=step, kind=kind)
   5278
   5279         # return a slice

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexes/base.p
y in slice_locs(self, start, end, step, kind)
   5474         start_slice = None
   5475         if start is not None:
-> 5476             start_slice = self.get_slice_bound(start, "left",
 kind)
   5477         if start_slice is None:
   5478             start_slice = 0

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexes/base.p
y in get_slice_bound(self, label, side, kind)
   5394             except ValueError:
   5395                 # raise the original KeyError
-> 5396                 raise err
   5397
   5398         if isinstance(slc, np.ndarray):

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexes/base.p
y in get_slice_bound(self, label, side, kind)
   5388         # we need to look up the label
   5389         try:
-> 5390             slc = self.get_loc(label)
   5391         except KeyError as err:
   5392             try:

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexes/base.p
y in get_loc(self, key, method, tolerance)
   3080                 return self._engine.get_loc(casted_key)
   3081             except KeyError as err:
-> 3082                 raise KeyError(key) from err
   3083
   3084         if tolerance is not None:

KeyError: 2
```

In [70]:

```
1 df.values
```

Out[70]:

```
array([['Afghanistan', 'Asia', 1952, 28.801, 8425333, 779.4453145],
       ['Afghanistan', 'Asia', 1957, 30.332, 9240934, 820.8530296],
       ['Afghanistan', 'Asia', 1962, 31.997, 10267083, 853.10071],
       ...,
       ['Zimbabwe', 'Africa', 1997, 46.809, 11404948, 792.4499603],
       ['Zimbabwe', 'Africa', 2002, 39.989, 11926563, 672.0386227],
       ['Zimbabwe', 'Africa', 2007, 43.487, 12311143, 469.7092981]],
      dtype=object)
```

In [71]:

```
type(df.values[0, 1])
```

Out[71]:

```
str
```

In [72]:

```
type(df.values[0, 2])
```

Out[72]:

```
int
```

In [74]:

```
df
```

Out[74]:

	country	continent	year	lifeExp	population	gdpPerCap
0	Afghanistan	Asia	1952	28.801	8425333	779.445314
1	Afghanistan	Asia	1957	30.332	9240934	820.853030
2	Afghanistan	Asia	1962	31.997	10267083	853.100710
3	Afghanistan	Asia	1967	34.020	11537966	836.197138
4	Afghanistan	Asia	1972	36.088	13079460	739.981106
...	...	...	...	...	...	...
1699	Zimbabwe	Africa	1987	62.351	9216418	706.157306
1700	Zimbabwe	Africa	1992	60.377	10704340	693.420786
1701	Zimbabwe	Africa	1997	46.809	11404948	792.449960
1702	Zimbabwe	Africa	2002	39.989	11926563	672.038623
1703	Zimbabwe	Africa	2007	43.487	12311143	469.709298

1704 rows × 6 columns

In [73]:

```
df.loc[3]
```

Out[73]:

```
country       Afghanistan
continent            Asia
year                 1967
lifeExp             34.02
population       11537966
gdpPerCap      836.197138
Name: 3, dtype: object
```

In [75]:

```
df.iloc[3]
```

Out[75]:

```
country        Afghanistan
continent             Asia
year                  1967
lifeExp              34.02
population        11537966
gdpPerCap       836.197138
Name: 3, dtype: object
```

In [76]:

```
df.iloc[[1, 10, 100]]
```

Out[76]:

	country	continent	year	lifeExp	population	gdpPerCap
1	Afghanistan	Asia	1957	30.332	9240934	820.853030
10	Afghanistan	Asia	2002	42.129	25268405	726.734055
100	Bangladesh	Asia	1972	45.252	70759295	630.233627

In [77]:

```
df.loc[[1, 10, 100]]
```

Out[77]:

	country	continent	year	lifeExp	population	gdpPerCap
1	Afghanistan	Asia	1957	30.332	9240934	820.853030
10	Afghanistan	Asia	2002	42.129	25268405	726.734055
100	Bangladesh	Asia	1972	45.252	70759295	630.233627

In [78]:

```
df.iloc[-1]
```

Out[78]:

```
country          Zimbabwe
continent          Africa
year                 2007
lifeExp            43.487
population       12311143
gdpPerCap      469.709298
Name: 1703, dtype: object
```

In [ ]:

```
df.loc[-1]
```

In [85]:

```
1 df.iloc[1:6, 1:4]
```

Out[85]:

	continent	year	lifeExp
1	Asia	1957	30.332
2	Asia	1962	31.997
3	Asia	1967	34.020
4	Asia	1972	36.088
5	Asia	1977	38.438

In [82]:

```
1 df.loc[1:5, 1:4]
```

```
---------------------------------------------------------------------
-----
TypeError                                 Traceback (most recent call
 last)
<ipython-input-82-494208dc7680> in <module>
----> 1 df.loc[1:5, 1:4]

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py in
__getitem__(self, key)
    887                     # AttributeError for IntervalTree get_valu
e
    888                     return self.obj._get_value(*key, takeable=
self._takeable)
--> 889             return self._getitem_tuple(key)
    890         else:
    891             # we by definition only have the 0th axis

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py in
_getitem_tuple(self, tup)
   1067             return self._multi_take(tup)
   1068
-> 1069         return self._getitem_tuple_same_dim(tup)
   1070
   1071     def _get_label(self, label, axis: int):

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py in
_getitem_tuple_same_dim(self, tup)
    773                 continue
    774
--> 775             retval = getattr(retval, self.name)._getitem_axis(
key, axis=i)
    776             # We should never have retval.ndim < self.ndim, as
that should
    777             #  be handled by the _getitem_lowerdim call above.

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py in
_getitem_axis(self, key, axis)
   1100         if isinstance(key, slice):
   1101             self._validate_key(key, axis)
-> 1102             return self._get_slice_axis(key, axis=axis)
   1103         elif com.is_bool_indexer(key):
   1104             return self._getbool_axis(key, axis=axis)

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py in
_get_slice_axis(self, slice_obj, axis)
   1134
   1135         labels = obj._get_axis(axis)
-> 1136         indexer = labels.slice_indexer(
   1137             slice_obj.start, slice_obj.stop, slice_obj.step, k
ind="loc"
   1138         )

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexes/base.p
y in slice_indexer(self, start, end, step, kind)
   5275         slice(1, 3, None)
   5276         """
-> 5277         start_slice, end_slice = self.slice_locs(start, end, s
```

```
tep=step, kind=kind)
   5278
   5279         # return a slice

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexes/base.p
y in slice_locs(self, start, end, step, kind)
   5474         start_slice = None
   5475         if start is not None:
-> 5476             start_slice = self.get_slice_bound(start, "left",
 kind)
   5477         if start_slice is None:
   5478             start_slice = 0

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexes/base.p
y in get_slice_bound(self, label, side, kind)
   5384         # For datetime indices label may be a string that has
 to be converted
   5385         # to datetime boundary according to its resolution.
-> 5386         label = self._maybe_cast_slice_bound(label, side, kind
)
   5387
   5388         # we need to look up the label

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexes/base.p
y in _maybe_cast_slice_bound(self, label, side, kind)
   5336         # reject them, if index does not contain label
   5337         if (is_float(label) or is_integer(label)) and label no
t in self.values:
-> 5338             raise self._invalid_indexer("slice", label)
   5339
   5340         return label

TypeError: cannot do slice indexing on Index with these indexers [1] o
f type int
```

In [84]:

```
1 df.loc[1:5, ["country", "lifeExp"]]
```

Out[84]:

	country	lifeExp
1	Afghanistan	30.332
2	Afghanistan	31.997
3	Afghanistan	34.020
4	Afghanistan	36.088
5	Afghanistan	38.438

In [88]:

```
df.loc[1:5, "country":"population"]
```

Out[88]:

	country	continent	year	lifeExp	population
1	Afghanistan	Asia	1957	30.332	9240934
2	Afghanistan	Asia	1962	31.997	10267083
3	Afghanistan	Asia	1967	34.020	11537966
4	Afghanistan	Asia	1972	36.088	13079460
5	Afghanistan	Asia	1977	38.438	14880372

In [89]:

```
df.iloc[[0,10,100], [0,2,3]]
```

Out[89]:

	country	year	lifeExp
0	Afghanistan	1952	28.801
10	Afghanistan	2002	42.129
100	Bangladesh	1972	45.252

In [90]:

```
df.iloc[1:10:2]
```

Out[90]:

	country	continent	year	lifeExp	population	gdpPerCap
1	Afghanistan	Asia	1957	30.332	9240934	820.853030
3	Afghanistan	Asia	1967	34.020	11537966	836.197138
5	Afghanistan	Asia	1977	38.438	14880372	786.113360
7	Afghanistan	Asia	1987	40.822	13867957	852.395945
9	Afghanistan	Asia	1997	41.763	22227415	635.341351

In [92]:

```
df.drop(3, axis=0, inplace=True)
```

In [93]:

```
1 df
```

Out[93]:

	country	continent	year	lifeExp	population	gdpPerCap
0	Afghanistan	Asia	1952	28.801	8425333	779.445314
1	Afghanistan	Asia	1957	30.332	9240934	820.853030
2	Afghanistan	Asia	1962	31.997	10267083	853.100710
4	Afghanistan	Asia	1972	36.088	13079460	739.981106
5	Afghanistan	Asia	1977	38.438	14880372	786.113360
...	...	...	...	...	...	...
1699	Zimbabwe	Africa	1987	62.351	9216418	706.157306
1700	Zimbabwe	Africa	1992	60.377	10704340	693.420786
1701	Zimbabwe	Africa	1997	46.809	11404948	792.449960
1702	Zimbabwe	Africa	2002	39.989	11926563	672.038623
1703	Zimbabwe	Africa	2007	43.487	12311143	469.709298

1703 rows × 6 columns

In [94]:

```
1 df.iloc[3]
```

Out[94]:

```
country       Afghanistan
continent            Asia
year                 1972
lifeExp            36.088
population       13079460
gdpPerCap      739.981106
Name: 4, dtype: object
```

In [95]:

```
1 df.loc[3]
```

```
---------------------------------------------------------------------
-----
KeyError                                  Traceback (most recent call
 last)
~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexes/base.p
y in get_loc(self, key, method, tolerance)
   3079             try:
-> 3080                 return self._engine.get_loc(casted_key)
   3081             except KeyError as err:

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.Int6
4HashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.Int6
4HashTable.get_item()

KeyError: 3

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

KeyError                                  Traceback (most recent call
 last)
<ipython-input-95-2d3184acc116> in <module>
----> 1 df.loc[3]

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py in
__getitem__(self, key)
    893
    894             maybe_callable = com.apply_if_callable(key, self.o
bj)
--> 895             return self._getitem_axis(maybe_callable, axis=axi
s)
    896
    897     def _is_scalar_access(self, key: Tuple):

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py in
_getitem_axis(self, key, axis)
   1122         # fall thru to straight lookup
   1123         self._validate_key(key, axis)
-> 1124         return self._get_label(key, axis=axis)
   1125
   1126     def _get_slice_axis(self, slice_obj: slice, axis: int):

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py in
_get_label(self, label, axis)
   1071     def _get_label(self, label, axis: int):
   1072         # GH#5667 this will fail if the label is not present i
n the axis.
-> 1073         return self.obj.xs(label, axis=axis)
   1074
   1075     def _handle_lowerdim_multi_index_axis0(self, tup: Tuple):

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/generic.py in
```

```
xs(self, key, axis, level, drop_level)
   3737                 raise TypeError(f"Expected label or tuple of l
abels, got {key}") from e
   3738             else:
-> 3739                 loc = index.get_loc(key)
   3740
   3741                 if isinstance(loc, np.ndarray):

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexes/base.p
y in get_loc(self, key, method, tolerance)
   3080                 return self._engine.get_loc(casted_key)
   3081             except KeyError as err:
-> 3082                 raise KeyError(key) from err
   3083
   3084         if tolerance is not None:

KeyError: 3
```

In [96]:

```
1 df
```

Out[96]:

	country	continent	year	lifeExp	population	gdpPerCap
0	Afghanistan	Asia	1952	28.801	8425333	779.445314
1	Afghanistan	Asia	1957	30.332	9240934	820.853030
2	Afghanistan	Asia	1962	31.997	10267083	853.100710
4	Afghanistan	Asia	1972	36.088	13079460	739.981106
5	Afghanistan	Asia	1977	38.438	14880372	786.113360
...	...	...	...	...	...	...
1699	Zimbabwe	Africa	1987	62.351	9216418	706.157306
1700	Zimbabwe	Africa	1992	60.377	10704340	693.420786
1701	Zimbabwe	Africa	1997	46.809	11404948	792.449960
1702	Zimbabwe	Africa	2002	39.989	11926563	672.038623
1703	Zimbabwe	Africa	2007	43.487	12311143	469.709298

1703 rows × 6 columns

In [97]:

```
1 df["lifeExp"].mean()
```

Out[97]:

```
59.489386189078054
```

In [98]:

```
1 df["lifeExp"].sum()
```

Out[98]:

```
101310.42468000001
```

In [99]:

```
df["lifeExp"].count()
```

Out[99]:

```
1703
```

In [100]:

```
df["lifeExp"].sum()/df["lifeExp"].count()
```

Out[100]:

```
59.489386189078104
```

In [102]:

```
import numpy as np
np.sin(df["lifeExp"] * np.pi)
```

Out[102]:

```
0        0.585241
1        0.863923
2       -0.009425
4        0.272952
5        0.981091
           ...
1699     0.892428
1700     0.926266
1701     0.564679
1702    -0.034551
1703    -0.999166
Name: lifeExp, Length: 1703, dtype: float64
```

In [103]:

```
# filter thr rows where lifexp > 60
# SELECT * FROM df WHERE lifeExp > 60
```

In [104]:

```
df["lifeExp"] > 60 # masking
```

Out[104]:

```
0       False
1       False
2       False
4       False
5       False
        ...
1699     True
1700     True
1701    False
1702    False
1703    False
Name: lifeExp, Length: 1703, dtype: bool
```

In [107]:

```
# filtering
df.loc[df["lifeExp"] > 60]
```

Out[107]:

	country	continent	year	lifeExp	population	gdpPerCap
14	Albania	Europe	1962	64.820	1728137	2312.888958
15	Albania	Europe	1967	66.220	1984060	2760.196931
16	Albania	Europe	1972	67.690	2263554	3313.422188
17	Albania	Europe	1977	68.930	2509048	3533.003910
18	Albania	Europe	1982	70.420	2780097	3630.880722
...	...	...	...	...	...	...
1678	Yemen, Rep.	Asia	2002	60.308	18701257	2234.820827
1679	Yemen, Rep.	Asia	2007	62.698	22211743	2280.769906
1698	Zimbabwe	Africa	1982	60.363	7636524	788.855041
1699	Zimbabwe	Africa	1987	62.351	9216418	706.157306
1700	Zimbabwe	Africa	1992	60.377	10704340	693.420786

877 rows × 6 columns

In [108]:

```
df.loc[df.lifeExp > 60]
```

Out[108]:

	country	continent	year	lifeExp	population	gdpPerCap
14	Albania	Europe	1962	64.820	1728137	2312.888958
15	Albania	Europe	1967	66.220	1984060	2760.196931
16	Albania	Europe	1972	67.690	2263554	3313.422188
17	Albania	Europe	1977	68.930	2509048	3533.003910
18	Albania	Europe	1982	70.420	2780097	3630.880722
...	...	...	...	...	...	...
1678	Yemen, Rep.	Asia	2002	60.308	18701257	2234.820827
1679	Yemen, Rep.	Asia	2007	62.698	22211743	2280.769906
1698	Zimbabwe	Africa	1982	60.363	7636524	788.855041
1699	Zimbabwe	Africa	1987	62.351	9216418	706.157306
1700	Zimbabwe	Africa	1992	60.377	10704340	693.420786

877 rows × 6 columns

In [112]:

```
df.loc[df.lifeExp>60, ['country','lifeExp']]
```

Out[112]:

	country	lifeExp
14	Albania	64.820
15	Albania	66.220
16	Albania	67.690
17	Albania	68.930
18	Albania	70.420
...	...	...
1678	Yemen, Rep.	60.308
1679	Yemen, Rep.	62.698
1698	Zimbabwe	60.363
1699	Zimbabwe	62.351
1700	Zimbabwe	60.377

877 rows × 2 columns

In [123]:

```
df[df["lifeExp"] > 60]
```

Out[123]:

	country	continent	year	lifeExp	population	gdpPerCap
14	Albania	Europe	1962	64.820	1728137	2312.888958
15	Albania	Europe	1967	66.220	1984060	2760.196931
16	Albania	Europe	1972	67.690	2263554	3313.422188
17	Albania	Europe	1977	68.930	2509048	3533.003910
18	Albania	Europe	1982	70.420	2780097	3630.880722
...	...	...	...	...	...	...
1678	Yemen, Rep.	Asia	2002	60.308	18701257	2234.820827
1679	Yemen, Rep.	Asia	2007	62.698	22211743	2280.769906
1698	Zimbabwe	Africa	1982	60.363	7636524	788.855041
1699	Zimbabwe	Africa	1987	62.351	9216418	706.157306
1700	Zimbabwe	Africa	1992	60.377	10704340	693.420786

877 rows × 6 columns

In [129]:

```
1 df.loc[(df["lifeExp"] < 60)  &  (df["lifeExp"] > 30)]
```

Out[129]:

	country	continent	year	lifeExp	population	gdpPerCap
1	Afghanistan	Asia	1957	30.332	9240934	820.853030
2	Afghanistan	Asia	1962	31.997	10267083	853.100710
4	Afghanistan	Asia	1972	36.088	13079460	739.981106
5	Afghanistan	Asia	1977	38.438	14880372	786.113360
6	Afghanistan	Asia	1982	39.854	12881816	978.011439
...	...	...	...	...	...	...
1696	Zimbabwe	Africa	1972	55.635	5861135	799.362176
1697	Zimbabwe	Africa	1977	57.674	6642107	685.587682
1701	Zimbabwe	Africa	1997	46.809	11404948	792.449960
1702	Zimbabwe	Africa	2002	39.989	11926563	672.038623
1703	Zimbabwe	Africa	2007	43.487	12311143	469.709298

823 rows × 6 columns

In [130]:

```
# "Kenya"  or "Egypt
df.loc[(df["country"] =='Kenya') | (df["country"] =='Egypt')]
```

Out[130]:

	country	continent	year	lifeExp	population	gdpPerCap
456	Egypt	Africa	1952	41.893	22223309	1418.822445
457	Egypt	Africa	1957	44.444	25009741	1458.915272
458	Egypt	Africa	1962	46.992	28173309	1693.335853
459	Egypt	Africa	1967	49.293	31681188	1814.880728
460	Egypt	Africa	1972	51.137	34807417	2024.008147
461	Egypt	Africa	1977	53.319	38783863	2785.493582
462	Egypt	Africa	1982	56.006	45681811	3503.729636
463	Egypt	Africa	1987	59.797	52799062	3885.460710
464	Egypt	Africa	1992	63.674	59402198	3794.755195
465	Egypt	Africa	1997	67.217	66134291	4173.181797
466	Egypt	Africa	2002	69.806	73312559	4754.604414
467	Egypt	Africa	2007	71.338	80264543	5581.180998
816	Kenya	Africa	1952	42.270	6464046	853.540919
817	Kenya	Africa	1957	44.686	7454779	944.438315
818	Kenya	Africa	1962	47.949	8678557	896.966373
819	Kenya	Africa	1967	50.654	10191512	1056.736457
820	Kenya	Africa	1972	53.559	12044785	1222.359968
821	Kenya	Africa	1977	56.155	14500404	1267.613204
822	Kenya	Africa	1982	58.766	17661452	1348.225791
823	Kenya	Africa	1987	59.339	21198082	1361.936856
824	Kenya	Africa	1992	59.285	25020539	1341.921721
825	Kenya	Africa	1997	54.407	28263827	1360.485021
826	Kenya	Africa	2002	50.992	31386842	1287.514732
827	Kenya	Africa	2007	54.110	35610177	1463.249282

In [131]:

```
# "Nigeria"
df.loc[df["country"] < 'Nigeria']
```

Out[131]:

	country	continent	year	lifeExp	population	gdpPerCap
0	Afghanistan	Asia	1952	28.801	8425333	779.445314
1	Afghanistan	Asia	1957	30.332	9240934	820.853030
2	Afghanistan	Asia	1962	31.997	10267083	853.100710
4	Afghanistan	Asia	1972	36.088	13079460	739.981106
5	Afghanistan	Asia	1977	38.438	14880372	786.113360
...	...	...	...	...	...	...
1123	Niger	Africa	1987	44.555	7332638	668.300023
1124	Niger	Africa	1992	47.391	8392818	581.182725
1125	Niger	Africa	1997	51.313	9666252	580.305209
1126	Niger	Africa	2002	54.496	11140655	601.074501
1127	Niger	Africa	2007	56.867	12894865	619.676892

1127 rows × 6 columns

In [132]:

```
# Calculate the average life expectancy of Asia
df.loc[df["continent"] == "Asia", "lifeExp"].mean()
```

Out[132]:

```
60.13083969620247
```

In [4]:

```
# group level estimates - grouping
```

In [8]:

```
df.groupby("continent")[["lifeExp"]].mean()
```

Out[8]:

continent	lifeExp
Africa	48.865330
Americas	64.658737
Asia	60.064903
Europe	71.903686
Oceania	74.326208

In [9]:

```
df.groupby("continent")[["lifeExp", "gdpPerCap"]].mean()
```

Out[9]:

continent	lifeExp	gdpPerCap
Africa	48.865330	2193.754578
Americas	64.658737	7136.110356
Asia	60.064903	7902.150428
Europe	71.903686	14469.475533
Oceania	74.326208	18621.609223

In [12]:

```
df.groupby("continent")[["lifeExp"]].count()
```

Out[12]:

continent	lifeExp
Africa	624
Americas	300
Asia	396
Europe	360
Oceania	24

In [15]:

```
df_year_cont = df.groupby(["year", "continent"])[["population"]].mean()
```

In [21]:

```
df_year_cont.loc[1952]
```

Out[21]:

continent	population
Africa	4.570010e+06
Americas	1.380610e+07
Asia	4.228356e+07
Europe	1.393736e+07
Oceania	5.343003e+06

In [24]:

```
df_cont_year = df.groupby(["continent", "year"])[["population"]].mean()
```

In [26]:

```
df_cont_year.loc["Africa"]
```

Out[26]:

year	population
1952	4.570010e+06
1957	5.093033e+06
1962	5.702247e+06
1967	6.447875e+06
1972	7.305376e+06
1977	8.328097e+06
1982	9.602857e+06
1987	1.105450e+07
1992	1.267464e+07
1997	1.430448e+07
2002	1.603315e+07
2007	1.787576e+07

In [27]:

```
1 df_cont_year.iloc[1]
```

Out[27]:

```
population    5.093033e+06
Name: (Africa, 1957), dtype: float64
```

In [28]:

```
1 df_cont_year
```

Out[28]:

		population
continent	**year**	
Africa	**1952**	4.570010e+06
	1957	5.093033e+06
	1962	5.702247e+06
	1967	6.447875e+06
	1972	7.305376e+06
	1977	8.328097e+06
	1982	9.602857e+06
	1987	1.105450e+07
	1992	1.267464e+07
	1997	1.430448e+07
	2002	1.603315e+07
	2007	1.787576e+07
Americas	**1952**	1.380610e+07
	1957	1.547816e+07
	1962	1.733081e+07
	1967	1.922986e+07
	1972	2.117537e+07
	1977	2.312271e+07
	1982	2.521164e+07
	1987	2.731016e+07
	1992	2.957096e+07
	1997	3.187602e+07
	2002	3.399091e+07
	2007	3.595485e+07
Asia	**1952**	4.228356e+07
	1957	4.735699e+07
	1962	5.140476e+07
	1967	5.774736e+07
	1972	6.518098e+07
	1977	7.225799e+07
	1982	7.909502e+07
	1987	8.700669e+07

		population
continent	**year**	
	1992	9.494825e+07
	1997	1.025238e+08
	2002	1.091455e+08
	2007	1.155138e+08
Europe	**1952**	1.393736e+07
	1957	1.459635e+07
	1962	1.534517e+07
	1967	1.603930e+07
	1972	1.668784e+07
	1977	1.723882e+07
	1982	1.770890e+07
	1987	1.810314e+07
	1992	1.860476e+07
	1997	1.896480e+07
	2002	1.927413e+07
	2007	1.953662e+07
Oceania	**1952**	5.343003e+06
	1957	5.970988e+06
	1962	6.641759e+06
	1967	7.300207e+06
	1972	8.053050e+06
	1977	8.619500e+06
	1982	9.197425e+06
	1987	9.787208e+06
	1992	1.045983e+07
	1997	1.112072e+07
	2002	1.172741e+07
	2007	1.227497e+07

In [29]:

```
1 df_cont_year.reset_index()
```

Out[29]:

	continent	year	population
0	Africa	1952	4.570010e+06
1	Africa	1957	5.093033e+06
2	Africa	1962	5.702247e+06
3	Africa	1967	6.447875e+06
4	Africa	1972	7.305376e+06
5	Africa	1977	8.328097e+06
6	Africa	1982	9.602857e+06
7	Africa	1987	1.105450e+07
8	Africa	1992	1.267464e+07
9	Africa	1997	1.430448e+07
10	Africa	2002	1.603315e+07
11	Africa	2007	1.787576e+07
12	Americas	1952	1.380610e+07
13	Americas	1957	1.547816e+07
14	Americas	1962	1.733081e+07
15	Americas	1967	1.922986e+07
16	Americas	1972	2.117537e+07
17	Americas	1977	2.312271e+07
18	Americas	1982	2.521164e+07
19	Americas	1987	2.731016e+07
20	Americas	1992	2.957096e+07
21	Americas	1997	3.187602e+07
22	Americas	2002	3.399091e+07
23	Americas	2007	3.595485e+07
24	Asia	1952	4.228356e+07
25	Asia	1957	4.735699e+07
26	Asia	1962	5.140476e+07
27	Asia	1967	5.774736e+07
28	Asia	1972	6.518098e+07
29	Asia	1977	7.225799e+07
30	Asia	1982	7.909502e+07
31	Asia	1987	8.700669e+07
32	Asia	1992	9.494825e+07

	continent	year	population
33	Asia	1997	1.025238e+08
34	Asia	2002	1.091455e+08
35	Asia	2007	1.155138e+08
36	Europe	1952	1.393736e+07
37	Europe	1957	1.459635e+07
38	Europe	1962	1.534517e+07
39	Europe	1967	1.603930e+07
40	Europe	1972	1.668784e+07
41	Europe	1977	1.723882e+07
42	Europe	1982	1.770890e+07
43	Europe	1987	1.810314e+07
44	Europe	1992	1.860476e+07
45	Europe	1997	1.896480e+07
46	Europe	2002	1.927413e+07
47	Europe	2007	1.953662e+07
48	Oceania	1952	5.343003e+06
49	Oceania	1957	5.970988e+06
50	Oceania	1962	6.641759e+06
51	Oceania	1967	7.300207e+06
52	Oceania	1972	8.053050e+06
53	Oceania	1977	8.619500e+06
54	Oceania	1982	9.197425e+06
55	Oceania	1987	9.787208e+06
56	Oceania	1992	1.045983e+07
57	Oceania	1997	1.112072e+07
58	Oceania	2002	1.172741e+07
59	Oceania	2007	1.227497e+07

In [34]:

```
1 len(df["country"].unique())
```

Out[34]:

```
142
```

In [35]:

```
df["country"].nunique()
```

Out[35]:

```
142
```

In [36]:

```
# number of countries continent wise
```

In [38]:

```
df.groupby("continent")[['country']].nunique().reset_index()
```

Out[38]:

	continent	country
0	Africa	52
1	Americas	25
2	Asia	33
3	Europe	30
4	Oceania	2

In [40]:

```
df.groupby("continent")[["lifeExp", "gdpPerCap"]].aggregate(["min", "mean", "max"])
```

Out[40]:

	lifeExp			gdpPerCap		
	min	mean	max	min	mean	max
continent						
Africa	23.599	48.865330	76.442	241.165876	2193.754578	21951.21176
Americas	37.579	64.658737	80.653	1201.637154	7136.110356	42951.65309
Asia	28.801	60.064903	82.603	331.000000	7902.150428	113523.13290
Europe	43.585	71.903686	81.757	973.533195	14469.475533	49357.19017
Oceania	69.120	74.326208	81.235	10039.595640	18621.609223	34435.36744

In [43]:

```
df.groupby("continent")[["lifeExp", "gdpPerCap"]].aggregate(["min", "mean", "max"])
```

In [48]:

```
t.columns = [tup[0] + "_" + tup[1] for tup in t.columns]
```

In [49]:

```
1 t
```

Out[49]:

continent	lifeExp_min	lifeExp_mean	lifeExp_max	gdpPerCap_min	gdpPerCap_mean	gdpPerCap
Africa	23.599	48.865330	76.442	241.165876	2193.754578	21951.2
Americas	37.579	64.658737	80.653	1201.637154	7136.110356	42951.(
Asia	28.801	60.064903	82.603	331.000000	7902.150428	113523.
Europe	43.585	71.903686	81.757	973.533195	14469.475533	49357.
Oceania	69.120	74.326208	81.235	10039.595640	18621.609223	34435.3

In [53]:

```
1 df.groupby("continent")[['lifeExp', 'gdpPerCap']].aggregate(
2                                     {"lifeExp":'min',
3                                     "gdpPerCap": 'max'})
```

Out[53]:

continent	lifeExp	gdpPerCap
Africa	23.599	21951.21176
Americas	37.579	42951.65309
Asia	28.801	113523.13290
Europe	43.585	49357.19017
Oceania	69.120	34435.36744

In [ ]:

```
1 # group countries based on continent
```

In [61]:

```
1 df.groupby("continent")["country"].unique()
```

Out[61]:

```
continent
Africa      [Algeria, Angola, Benin, Botswana, Burkina Fas...
Americas    [Argentina, Bolivia, Brazil, Canada, Chile, Co...
Asia        [Afghanistan, Bahrain, Bangladesh, Cambodia, C...
Europe      [Albania, Austria, Belgium, Bosnia and Herzego...
Oceania                              [Australia, New Zealand]
Name: country, dtype: object
```

In [62]:

```
1 #og rows, filter the rows, condition is based on aggreation
```

In [ ]:

```
# filters all the rows where lifeExp < 50
```

In [63]:

```
df.loc[df["lifeExp"]<50]
```

Out[63]:

	country	continent	year	lifeExp	population	gdpPerCap
0	Afghanistan	Asia	1952	28.801	8425333	779.445314
1	Afghanistan	Asia	1957	30.332	9240934	820.853030
2	Afghanistan	Asia	1962	31.997	10267083	853.100710
3	Afghanistan	Asia	1967	34.020	11537966	836.197138
4	Afghanistan	Asia	1972	36.088	13079460	739.981106
...	...	...	...	...	...	...
1691	Zambia	Africa	2007	42.384	11746035	1271.211593
1692	Zimbabwe	Africa	1952	48.451	3080907	406.884115
1701	Zimbabwe	Africa	1997	46.809	11404948	792.449960
1702	Zimbabwe	Africa	2002	39.989	11926563	672.038623
1703	Zimbabwe	Africa	2007	43.487	12311143	469.709298

491 rows × 6 columns

In [ ]:

```
# filters all the rows where avg lifeExp (by continent) < 50
```

In [64]:

```
df.groupby("continent")["lifeExp"].mean()
```

Out[64]:

```
continent
Africa      48.865330
Americas    64.658737
Asia        60.064903
Europe      71.903686
Oceania     74.326208
Name: lifeExp, dtype: float64
```

In [65]:

```
df.loc[df["continent"]=="Africa"]
```

Out[65]:

	country	continent	year	lifeExp	population	gdpPerCap
24	Algeria	Africa	1952	43.077	9279525	2449.008185
25	Algeria	Africa	1957	45.685	10270856	3013.976023
26	Algeria	Africa	1962	48.303	11000948	2550.816880
27	Algeria	Africa	1967	51.407	12760499	3246.991771
28	Algeria	Africa	1972	54.518	14760787	4182.663766
...	...	...	...	...	...	...
1699	Zimbabwe	Africa	1987	62.351	9216418	706.157306
1700	Zimbabwe	Africa	1992	60.377	10704340	693.420786
1701	Zimbabwe	Africa	1997	46.809	11404948	792.449960
1702	Zimbabwe	Africa	2002	39.989	11926563	672.038623
1703	Zimbabwe	Africa	2007	43.487	12311143	469.709298

624 rows × 6 columns

In [67]:

```
def func(df):
    return df["lifeExp"].mean() < 50
```

In [69]:

```
df.groupby("continent").filter(lambda x: x["lifeExp"].mean() < 50)
```

Out[69]:

	country	continent	year	lifeExp	population	gdpPerCap
24	Algeria	Africa	1952	43.077	9279525	2449.008185
25	Algeria	Africa	1957	45.685	10270856	3013.976023
26	Algeria	Africa	1962	48.303	11000948	2550.816880
27	Algeria	Africa	1967	51.407	12760499	3246.991771
28	Algeria	Africa	1972	54.518	14760787	4182.663766
...	...	...	...	...	...	...
1699	Zimbabwe	Africa	1987	62.351	9216418	706.157306
1700	Zimbabwe	Africa	1992	60.377	10704340	693.420786
1701	Zimbabwe	Africa	1997	46.809	11404948	792.449960
1702	Zimbabwe	Africa	2002	39.989	11926563	672.038623
1703	Zimbabwe	Africa	2007	43.487	12311143	469.709298

624 rows × 6 columns

In [70]:

```
# transform some column
```

In [71]:

```
# center the lifeExp values around zero
```

In [72]:

```
def centering(x):
    x["lifeExp"] -= x["lifeExp"].mean()
```

In [73]:

```
df.groupby("continent").transform(centering)
```

```
-> 1410             slow_path = lambda group: group.apply(
   1411                 lambda x: func(x, *args, **kwargs), axis=self.
axis
   1412             )

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/frame.py in ap
ply(self, func, axis, raw, result_type, args, **kwds)
   7766             kwds=kwds,
   7767         )
-> 7768         return op.get_result()
   7769
   7770     def applymap(self, func, na_action: Optional[str] = None)
-> DataFrame:

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/apply.py in ge
t_result(self)
    183             return self.apply_raw()
    184
--> 185         return self.apply_standard()
    186
```

In [74]:

```
def inspect(x):
    print(type(x))
    raise
```

In [77]:

```
1 df.groupby("continent").transform(inspect)
```

```
<class 'pandas.core.series.Series'>

---------------------------------------------------------------------------
RuntimeError                              Traceback (most recent call
 last)
<ipython-input-77-580170fe7b50> in <module>
----> 1 df.groupby("continent").transform(inspect)

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/groupby/generi
c.py in transform(self, func, engine, engine_kwargs, *args, **kwargs)
   1357
   1358         if not isinstance(func, str):
-> 1359             return self._transform_general(func, *args, **kwar
gs)
   1360
   1361         elif func not in base.transform_kernel_allowlist:

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/groupby/generi
c.py in _transform_general(self, func, *args, **kwargs)
   1304             # Try slow path and fast path.
   1305             try:
-> 1306                 path, res = self._choose_path(fast_path, slow_
path, group)
   1307             except TypeError:
   1308                 return self._transform_item_by_item(obj, fast_
path)

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/groupby/generi
c.py in _choose_path(self, fast_path, slow_path, group)
   1415     def _choose_path(self, fast_path: Callable, slow_path: Cal
lable, group: DataFrame):
   1416         path = slow_path
-> 1417         res = slow_path(group)
   1418
   1419         # if we make it here, test if we can use the fast path

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/groupby/generi
c.py in <lambda>(group)
   1408         else:
   1409             fast_path = lambda group: func(group, *args, **kwa
rgs)
-> 1410             slow_path = lambda group: group.apply(
   1411                 lambda x: func(x, *args, **kwargs), axis=self.
axis
   1412             )

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/frame.py in ap
ply(self, func, axis, raw, result_type, args, **kwds)
   7766             kwds=kwds,
   7767         )
-> 7768         return op.get_result()
   7769
   7770     def applymap(self, func, na_action: Optional[str] = None)
-> DataFrame:

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/apply.py in ge
```

```
t_result(self)
    183             return self.apply_raw()
    184
--> 185         return self.apply_standard()
    186
    187     def apply_empty_result(self):

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/apply.py in ap
ply_standard(self)
    274
    275     def apply_standard(self):
--> 276         results, res_index = self.apply_series_generator()
    277
    278         # wrap results

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/apply.py in ap
ply_series_generator(self)
    288             for i, v in enumerate(series_gen):
    289                 # ignore SettingWithCopy here in case the user
mutates
--> 290                 results[i] = self.f(v)
    291                 if isinstance(results[i], ABCSeries):
    292                     # If we have a view on v, we need to make
 a copy because

~/opt/anaconda3/lib/python3.8/site-packages/pandas/core/groupby/generi
c.py in <lambda>(x)
   1409             fast_path = lambda group: func(group, *args, **kwa
rgs)
   1410             slow_path = lambda group: group.apply(
-> 1411                 lambda x: func(x, *args, **kwargs), axis=self.
axis
   1412             )
   1413         return fast_path, slow_path

<ipython-input-74-03de9c55a073> in inspect(x)
      1 def inspect(x):
      2     print(type(x))
----> 3     raise

RuntimeError: No active exception to reraise
```

In [79]:

```
def centering(x):
    x -= x.mean()
    return x

df.groupby("continent")[["lifeExp"]].transform(centering)
```

Out[79]:

	lifeExp
0	-31.263903
1	-29.732903
2	-28.067903
3	-26.044903
4	-23.976903
...	...
1699	13.485670
1700	11.511670
1701	-2.056330
1702	-8.876330
1703	-5.378330

1704 rows × 1 columns

In [84]:

```
def some_weird_centering(x):
    x["lifeExp"] -= x["gdpPerCap"].mean()/1000
    return x

df.groupby("continent").apply(some_weird_centering)
```

Out[84]:

	country	continent	year	lifeExp	population	gdpPerCap
0	Afghanistan	Asia	1952	20.898850	8425333	779.445314
1	Afghanistan	Asia	1957	22.429850	9240934	820.853030
2	Afghanistan	Asia	1962	24.094850	10267083	853.100710
3	Afghanistan	Asia	1967	26.117850	11537966	836.197138
4	Afghanistan	Asia	1972	28.185850	13079460	739.981106
...	...	...	...	...	...	...
1699	Zimbabwe	Africa	1987	60.157245	9216418	706.157306
1700	Zimbabwe	Africa	1992	58.183245	10704340	693.420786
1701	Zimbabwe	Africa	1997	44.615245	11404948	792.449960
1702	Zimbabwe	Africa	2002	37.795245	11926563	672.038623
1703	Zimbabwe	Africa	2007	41.293245	12311143	469.709298

1704 rows × 6 columns

In [83]:

```
def inspect(x):
    try:
        print(type(x))
    except:
        return None
df.groupby("continent").apply(inspect)
```

```
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.frame.DataFrame'>
```

Out[83]:

—

In [87]:

```
import numpy as np
df[["lifeExp", "gdpPerCap"]].apply(np.mean, axis=0)
```

Out[87]:

```
lifeExp        59.474439
gdpPerCap    7215.327081
dtype: float64
```

In [88]:

```
df[["lifeExp", "gdpPerCap"]].apply(np.mean, axis=1)
```

Out[88]:

```
0       404.123157
1       425.592515
2       442.548855
3       435.108569
4       388.034553
           ...
1699    384.254153
1700    376.898893
1701    419.629480
1702    356.013811
1703    256.598149
Length: 1704, dtype: float64
```

In [89]:

```
df[["lifeExp", "gdpPerCap"]].apply(np.mean)
```

Out[89]:

```
lifeExp        59.474439
gdpPerCap    7215.327081
dtype: float64
```

In [92]:

```
df.sort_values(["year"], ascending=False)
```

Out[92]:

	country	continent	year	lifeExp	population	gdpPerCap
1703	Zimbabwe	Africa	2007	43.487	12311143	469.709298
491	Equatorial Guinea	Africa	2007	51.579	551201	12154.089750
515	Ethiopia	Africa	2007	52.947	76511887	690.805576
527	Finland	Europe	2007	79.313	5238460	33207.084400
539	France	Europe	2007	80.657	61083916	30470.016700
...	...	...	...	...	...	...
1116	Niger	Africa	1952	37.444	3379468	761.879376
1128	Nigeria	Africa	1952	36.324	33119096	1077.281856
1140	Norway	Europe	1952	72.670	3327728	10095.421720
1152	Oman	Asia	1952	37.578	507833	1828.230307
852	Kuwait	Asia	1952	55.565	160000	108382.352900

1704 rows × 6 columns

In [94]:

```
df.sort_values(["year", "continent"])
```

Out[94]:

	country	continent	year	lifeExp	population	gdpPerCap
24	Algeria	Africa	1952	43.077	9279525	2449.008185
36	Angola	Africa	1952	30.015	4232095	3520.610273
120	Benin	Africa	1952	38.223	1738315	1062.752200
156	Botswana	Africa	1952	47.622	442308	851.241141
192	Burkina Faso	Africa	1952	31.975	4469979	543.255241
...	...	...	...	...	...	...
1487	Switzerland	Europe	2007	81.701	7554661	37506.419070
1583	Turkey	Europe	2007	71.777	71158647	8458.276384
1607	United Kingdom	Europe	2007	79.425	60776238	33203.261280
71	Australia	Oceania	2007	81.235	20434176	34435.367440
1103	New Zealand	Oceania	2007	80.204	4115771	25185.009110

1704 rows × 6 columns

In [95]:

```
df.sort_values(["year", "continent"], ascending=[True, False])
```

Out[95]:

	country	continent	year	lifeExp	population	gdpPerCap
60	Australia	Oceania	1952	69.120	8691212	10039.595640
1092	New Zealand	Oceania	1952	69.390	1994794	10556.575660
12	Albania	Europe	1952	55.230	1282697	1601.056136
72	Austria	Europe	1952	66.800	6927772	6137.076492
108	Belgium	Europe	1952	68.000	8730405	8343.105127
...	...	...	...	...	...	...
1547	Togo	Africa	2007	58.420	5701579	882.969944
1571	Tunisia	Africa	2007	73.923	10276158	7092.923025
1595	Uganda	Africa	2007	51.542	29170398	1056.380121
1691	Zambia	Africa	2007	42.384	11746035	1271.211593
1703	Zimbabwe	Africa	2007	43.487	12311143	469.709298

1704 rows × 6 columns

In [96]:

```
# creating df from scratch
```

In [98]:

```
pd.Series([10, 20, 30], index=[1, 2, 3])
```

Out[98]:

```
1    10
2    20
3    30
dtype: int64
```

In [99]:

```
# df - by rows, by columns
```

In [100]:

```
pd.DataFrame([[10, 20], [30, 40]], columns=["A", "B"])
```

Out[100]:

	A	B
0	10	20
1	30	40

In [102]:

```
pd.DataFrame([[10, 20]], columns=["A", "B"])
```

Out[102]:

	A	B
0	10	20

In [103]:

```
# column
```

In [104]:

```
pd.DataFrame({"A": [10, 30], "B": [20, 40]})
```

Out[104]:

	A	B
0	10	20
1	30	40

In [105]:

```
# Concatenate
```

In [106]:

```
a = pd.DataFrame({'A':[10,30], 'B':[20,40]})
b = pd.DataFrame({'A':[10,30], 'C':[20,40]})
```

In [107]:

```
a
```

Out[107]:

	A	B
0	10	20
1	30	40

In [108]:

```
1 b
```

Out[108]:

	A	C
0	10	20
1	30	40

In [109]:

```
1 pd.concat([a, b], axis=1)
```

Out[109]:

	A	B	A	C
0	10	20	10	20
1	30	40	30	40

In [115]:

```
1 pd.concat([a, b], axis=0).reset_index(drop=True)
```

Out[115]:

	A	B	C
0	10	20.0	NaN
1	30	40.0	NaN
2	10	NaN	20.0
3	30	NaN	40.0

In [116]:

```
1 pd.concat([a, b], axis=0, ignore_index=True)
```

Out[116]:

	A	B	C
0	10	20.0	NaN
1	30	40.0	NaN
2	10	NaN	20.0
3	30	NaN	40.0

In [120]:

```
pd.concat([a, b], axis=0, keys=["x", "y"])
```

Out[120]:

		A	B	C
x	0	10	20.0	NaN
	1	30	40.0	NaN
y	0	10	NaN	20.0
	1	30	NaN	40.0

In [121]:

```
pd.concat([a, b], join="outer")
```

Out[121]:

	A	B	C
0	10	20.0	NaN
1	30	40.0	NaN
0	10	NaN	20.0
1	30	NaN	40.0

In [122]:

```
pd.concat([a, b], join="inner")
```

Out[122]:

	A
0	10
1	30
0	10
1	30

In [124]:

```
users = pd.DataFrame({"userid": [1, 2, 3], "name": ["A", "B", "C"]})
users
```

Out[124]:

	userid	name
0	1	A
1	2	B
2	3	C

In [125]:

```
msgs = pd.DataFrame({"userid": [1, 1, 2], "msg":["hello", "world", "hi"]})
msgs
```

Out[125]:

	userid	msg
0	1	hello
1	1	world
2	2	hi

In [132]:

```
users.merge(msgs, on="userid")
```

Out[132]:

	userid	name	msg
0	1	A	hello
1	1	A	world
2	2	B	hi

In [135]:

```
users.rename(columns={"userid":"id"}, inplace=True)
```

In [136]:

```
users
```

Out[136]:

	id	name
0	1	A
1	2	B
2	3	C

In [137]:

```
msgs
```

Out[137]:

	userid	msg
0	1	hello
1	1	world
2	2	hi

In [138]:

```
users.merge(msgs, left_on="id", right_on="userid")
```

Out[138]:

	id	name	userid	msg
0	1	A	1	hello
1	1	A	1	world
2	2	B	2	hi

In [139]:

```
users.merge(msgs, left_on="id", right_on="userid", how="inner")
```

Out[139]:

	id	name	userid	msg
0	1	A	1	hello
1	1	A	1	world
2	2	B	2	hi

In [140]:

```
users.merge(msgs, left_on="id", right_on="userid", how="right")
```

Out[140]:

	id	name	userid	msg
0	1	A	1	hello
1	1	A	1	world
2	2	B	2	hi

In [141]:

```
users.merge(msgs, left_on="id", right_on="userid", how="left")
```

Out[141]:

	id	name	userid	msg
0	1	A	1.0	hello
1	1	A	1.0	world
2	2	B	2.0	hi
3	3	C	NaN	NaN

In [142]:

```
users.merge(msgs, left_on="id", right_on="userid", how="outer")
```

Out[142]:

	id	name	userid	msg
0	1	A	1.0	hello
1	1	A	1.0	world
2	2	B	2.0	hi
3	3	C	NaN	NaN

In [155]:

```
disease = pd.read_csv("data/disease.csv")
```

In [146]:

```
type(None)
```

Out[146]:

```
NoneType
```

In [147]:

```
type(np.NaN)
```

Out[147]:

```
float
```

In [148]:

```
pd.Series([1, np.nan, 2, None])
```

Out[148]:

```
0    1.0
1    NaN
2    2.0
3    NaN
dtype: float64
```

In [149]:

```
from numpy import nan, NAN, NaN
```

In [150]:

```
nan == NAN
```

Out[150]:

```
False
```

In [151]:

```
nan == nan
```

Out[151]:

```
False
```

In [152]:

```
nan is nan
```

Out[152]:

```
True
```

In [153]:

```
pd.isnull(nan)
```

Out[153]:

```
True
```

In [156]:

```
1 disease.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 122 entries, 0 to 121
Data columns (total 14 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Date                  122 non-null    object 
 1   Day                   122 non-null    int64  
 2   Cases_Guinea          93 non-null     float64
 3   Cases_Liberia         83 non-null     float64
 4   Cases_Nigeria         38 non-null     float64
 5   Cases_Senegal         25 non-null     float64
 6   Cases_UnitedKingdom   18 non-null     float64
 7   Cases_Mali            12 non-null     float64
 8   Deaths_Guinea         92 non-null     float64
 9   Deaths_Liberia        81 non-null     float64
 10  Deaths_Nigeria        38 non-null     float64
 11  Deaths_Senegal        22 non-null     float64
 12  Deaths_UnitedKingdom  18 non-null     float64
 13  Deaths_Mali           12 non-null     float64
dtypes: float64(12), int64(1), object(1)
memory usage: 13.5+ KB
```

In [158]:

```
1 disease.isna().sum()
```

Out[158]:

```
Date                    0
Day                     0
Cases_Guinea           29
Cases_Liberia          39
Cases_Nigeria          84
Cases_Senegal          97
Cases_UnitedKingdom   104
Cases_Mali            110
Deaths_Guinea          30
Deaths_Liberia         41
Deaths_Nigeria         84
Deaths_Senegal        100
Deaths_UnitedKingdom  104
Deaths_Mali           110
dtype: int64
```

In [161]:

```
disease["Cases_Guinea"].value_counts(dropna=False)
```

Out[161]:

```
NaN       29
86.0       3
112.0      2
495.0      2
390.0      2
          ..
344.0      1
1472.0     1
510.0      1
2416.0     1
2706.0     1
Name: Cases_Guinea, Length: 89, dtype: int64
```

In [166]:

```
len(disease['Cases_Guinea'].unique())
```

Out[166]:

```
89
```

In [168]:

```
disease['Cases_Guinea'].nunique(dropna=False)
```

Out[168]:

```
89
```

In [171]:

```
1 disease.isnull() # same as isna()
```

Out[171]:

	Date	Day	Cases_Guinea	Cases_Liberia	Cases_Nigeria	Cases_Senegal	Cases_UnitedKin
0	False	False	False	True	True	True	
1	False	False	False	True	True	True	
2	False	False	False	False	True	True	
3	False	False	True	False	True	True	
4	False	False	False	False	True	True	
...	...	...	...	...	...	...	
117	False	False	False	False	True	True	
118	False	False	False	True	True	True	
119	False	False	False	True	True	True	
120	False	False	False	True	True	True	
121	False	False	False	True	True	True	

122 rows × 14 columns

In [173]:

```
1 df_cont_year.index
```

Out[173]:

```
MultiIndex([(  'Africa', 1952),
            (  'Africa', 1957),
            (  'Africa', 1962),
            (  'Africa', 1967),
            (  'Africa', 1972),
            (  'Africa', 1977),
            (  'Africa', 1982),
            (  'Africa', 1987),
            (  'Africa', 1992),
            (  'Africa', 1997),
            (  'Africa', 2002),
            (  'Africa', 2007),
            ('Americas', 1952),
            ('Americas', 1957),
            ('Americas', 1962),
            ('Americas', 1967),
            ('Americas', 1972),
            ('Americas', 1977),
            ('Americas', 1982),
            ('Americas', 1987),
            ('Americas', 1992),
            ('Americas', 1997),
            ('Americas', 2002),
            ('Americas', 2007),
            (    'Asia', 1952),
            (    'Asia', 1957),
            (    'Asia', 1962),
            (    'Asia', 1967),
            (    'Asia', 1972),
            (    'Asia', 1977),
            (    'Asia', 1982),
            (    'Asia', 1987),
            (    'Asia', 1992),
            (    'Asia', 1997),
            (    'Asia', 2002),
            (    'Asia', 2007),
            (  'Europe', 1952),
            (  'Europe', 1957),
            (  'Europe', 1962),
            (  'Europe', 1967),
            (  'Europe', 1972),
            (  'Europe', 1977),
            (  'Europe', 1982),
            (  'Europe', 1987),
            (  'Europe', 1992),
            (  'Europe', 1997),
            (  'Europe', 2002),
            (  'Europe', 2007),
            ( 'Oceania', 1952),
            ( 'Oceania', 1957),
            ( 'Oceania', 1962),
            ( 'Oceania', 1967),
            ( 'Oceania', 1972),
            ( 'Oceania', 1977),
            ( 'Oceania', 1982),
```

```
 ( 'Oceania', 1987),
 ( 'Oceania', 1992),
 ( 'Oceania', 1997),
 ( 'Oceania', 2002),
 ( 'Oceania', 2007)],
names=['continent', 'year'])
```