Categorical data analysis

In [22]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Read the data

In [2]:
```
visadf=pd.read_csv('C:/Users/Anuja_PC/OneDrive/Documents/dataFiles/Visadataset -
visadf
```

Out[2]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_t
0	EZYV01	Asia	High School	N	
1	EZYV02	Asia	Master's	Y	
2	EZYV03	Asia	Bachelor's	N	
3	EZYV04	Asia	Bachelor's	N	
4	EZYV05	Africa	Master's	Y	
...	...	...	...	...	
25475	EZYV25476	Asia	Bachelor's	Y	
25476	EZYV25477	Asia	High School	Y	
25477	EZYV25478	Asia	Master's	Y	
25478	EZYV25479	Asia	Master's	Y	
25479	EZYV25480	Asia	Bachelor's	Y	

25480 rows × 12 columns

In [4]:
```
visadf['continent']
```

Out[4]:
```
0          Asia
1          Asia
2          Asia
3          Asia
4        Africa
          ...
25475      Asia
25476      Asia
25477      Asia
25478      Asia
25479      Asia
Name: continent, Length: 25480, dtype: object
```

In [4]:
```
visadf[['continent']]
```

Out[4]:

	continent
0	Asia
1	Asia
2	Asia
3	Asia
4	Africa
...	...
25475	Asia
25476	Asia
25477	Asia
25478	Asia
25479	Asia

25480 rows × 1 columns

In [5]:
```
visadf.dtypes
```

Out[5]:
```
case_id                 object
continent               object
education_of_employee   object
has_job_experience      object
requires_job_training   object
no_of_employees          int64
yr_of_estab              int64
region_of_employment    object
prevailing_wage        float64
unit_of_wage            object
full_time_position      object
case_status             object
dtype: object
```

In [6]:
```
visadf.continent
```

Out[6]:
```
0          Asia
1          Asia
2          Asia
3          Asia
4        Africa
          ...
25475      Asia
25476      Asia
25477      Asia
25478      Asia
25479      Asia
Name: continent, Length: 25480, dtype: object
```

In [7]:
```
visadf['continent']
```

Out[7]:
```
0          Asia
1          Asia
2          Asia
3          Asia
4        Africa
          ...
25475      Asia
25476      Asia
25477      Asia
25478      Asia
25479      Asia
Name: continent, Length: 25480, dtype: object
```

In [8]:
```
visadf[['continent']]
```

Out[8]:

	continent
0	Asia
1	Asia
2	Asia
3	Asia
4	Africa
...	...
25475	Asia
25476	Asia
25477	Asia
25478	Asia
25479	Asia

25480 rows × 1 columns

unique

In [5]:
```
visadf['continent'].unique()  # unique available on , only series
```

Out[5]:
```
array(['Asia', 'Africa', 'North America', 'Europe', 'South America',
       'Oceania'], dtype=object)
```

nunique

In [12]:
```
visadf['continent'].nunique()
```

Out[12]:
```
6
```

task-1

1. how many members from asia

In [14]:
```
visadf['continent']=='Asia'
```

Out[14]:
```
0         True
1         True
2         True
3         True
4        False
         ...
25475     True
25476     True
25477     True
25478     True
25479     True
Name: continent, Length: 25480, dtype: bool
```

In [6]:
```
condition = visadf['continent']=='Asia'   # fetching the records of continent As

visadf[condition]
```

Out[6]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_t
0	EZYV01	Asia	High School	N	
1	EZYV02	Asia	Master's	Y	
2	EZYV03	Asia	Bachelor's	N	
3	EZYV04	Asia	Bachelor's	N	
5	EZYV06	Asia	Master's	Y	
...	...	...	...	...	
25475	EZYV25476	Asia	Bachelor's	Y	
25476	EZYV25477	Asia	High School	Y	
25477	EZYV25478	Asia	Master's	Y	
25478	EZYV25479	Asia	Master's	Y	
25479	EZYV25480	Asia	Bachelor's	Y	

16861 rows × 12 columns

In [18]:
```
len(visadf[condition])
```

Out[18]:
```
16861
```

In [7]:
```
unique_cnt= visadf['continent'].unique()
listOfContinentCont=[]
for i in unique_cnt:
    con = visadf['continent']==i
    val = len(visadf[con])
    listOfContinentCont.append(val)
    print(f"{i}:{val}")
```

```
Asia:16861
Africa:551
North America:3292
Europe:3732
South America:852
Oceania:192
```

In [35]:
```
import pandas as pd
```

In [8]:
```
continentDataframe=pd.DataFrame(zip(unique_cnt,listOfContinentCont),columns=['co
```

In [46]:
```
continentDataframe.to_csv('continentdf.csv',index=False)
```

$value - counts$

In [12]:
```
cdf=visadf['continent'].value_counts()
```

In [48]:
```
visadf
```

Out[48]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_t
0	EZYV01	Asia	High School	N	
1	EZYV02	Asia	Master's	Y	
2	EZYV03	Asia	Bachelor's	N	
3	EZYV04	Asia	Bachelor's	N	
4	EZYV05	Africa	Master's	Y	
...	...	...	...	...	
25475	EZYV25476	Asia	Bachelor's	Y	
25476	EZYV25477	Asia	High School	Y	
25477	EZYV25478	Asia	Master's	Y	
25478	EZYV25479	Asia	Master's	Y	
25479	EZYV25480	Asia	Bachelor's	Y	

25480 rows × 12 columns

In [10]:
```
visadf['education_of_employee'].value_counts()
```

Out[10]:
```
education_of_employee
Bachelor's     10234
Master's        9634
High School     3420
Doctorate       2192
Name: count, dtype: int64
```

In [ ]:
```
education_of_employee
```

In [13]:
```
cdf.keys()
```

Out[13]:
```
Index(['Asia', 'Europe', 'North America', 'South America', 'Africa',
       'Oceania'],
      dtype='object', name='continent')
```

In [15]:
```
cdf.index
```

Out[15]:
```
Index(['Asia', 'Europe', 'North America', 'South America', 'Africa',
       'Oceania'],
      dtype='object', name='continent')
```

In [16]:
```
cdf.values
```

Out[16]:
```
array([16861,  3732,  3292,   852,   551,   192], dtype=int64)
```

In [17]:
```
listKeys = cdf.keys()
listKeys
```

Out[17]:
```
Index(['Asia', 'Europe', 'North America', 'South America', 'Africa',
       'Oceania'],
      dtype='object', name='continent')
```

In [19]:
```
cdfvalues= cdf.values
cdfvalues
```

Out[19]:
```
array([16861,  3732,  3292,   852,   551,   192], dtype=int64)
```

In [20]:
```
dictCdf=[[listKeys,cdfvalues],columns='continet','count']
dictCdf
```

```
  Cell In[20], line 1
    dictCdf=[[listKeys,cdfvalues],columns='continet','count']
                                  ^
SyntaxError: invalid syntax. Maybe you meant '==' or ':=' instead of '='?
```

Bar Chart

- is the represenatation of counts WRT classes
- if we want to plot bar chart we require 2 columns
  - one column iscategorical data columns
  - another column is numerical
- we can use continentDataframe which we created
- package matplotlib

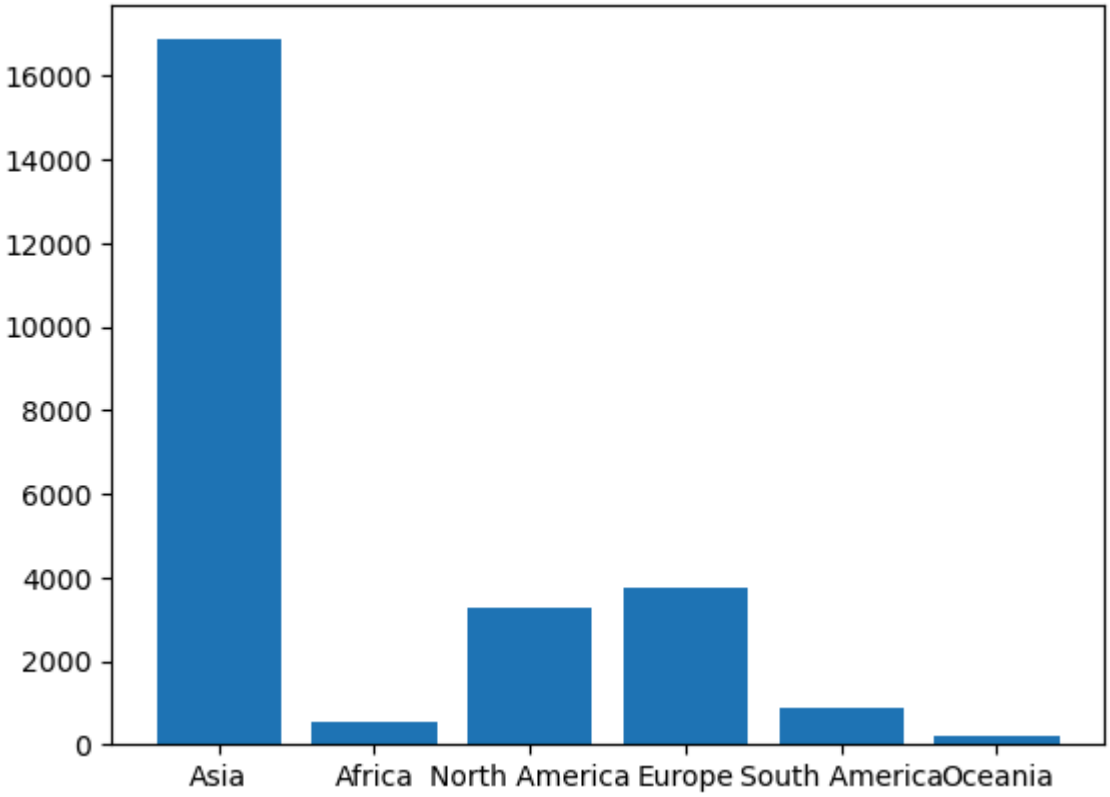
In [21]:
```
continentDataframe
```

Out[21]:

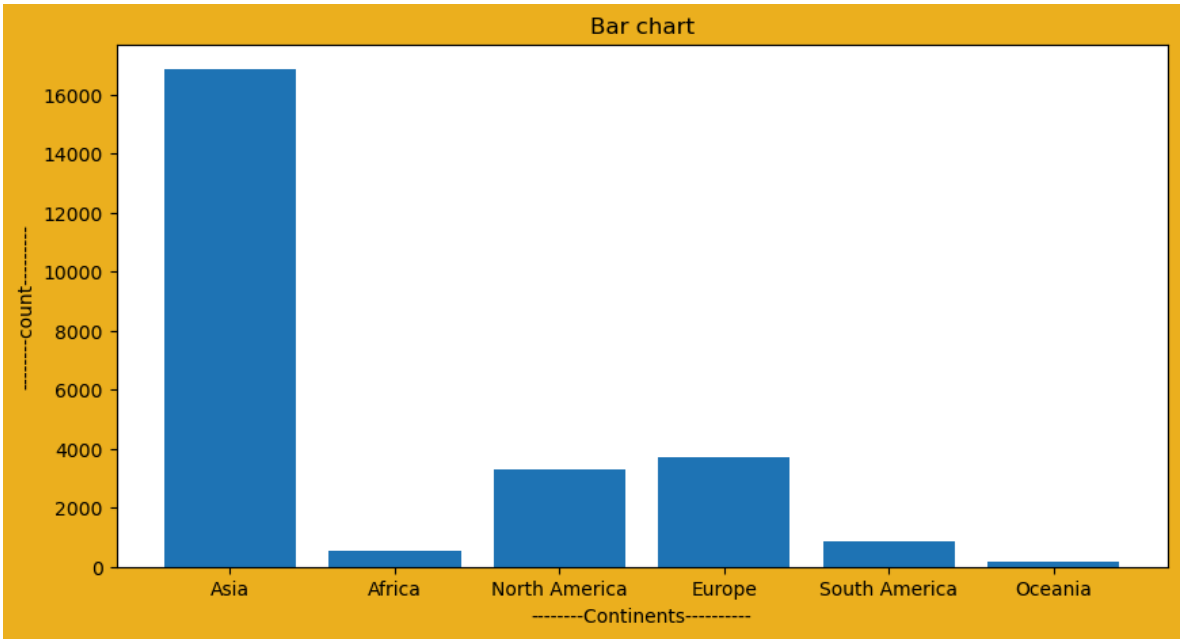
	continent	count
0	Asia	16861
1	Africa	551
2	North America	3292
3	Europe	3732
4	South America	852
5	Oceania	192

In [24]:
```
plt.bar('continent','count',data=continentDataframe)
```

Out[24]:
```
<BarContainer object of 6 artists>
```



In [44]:
```
plt.figure(figsize=[10,5],facecolor='#EDB120')
plt.bar('continent','count',data=continentDataframe)
plt.title("Bar chart")
plt.xlabel("--------Continents----------")
plt.ylabel("--------count---------")
plt.savefig("barchart.png")
plt.show()
```

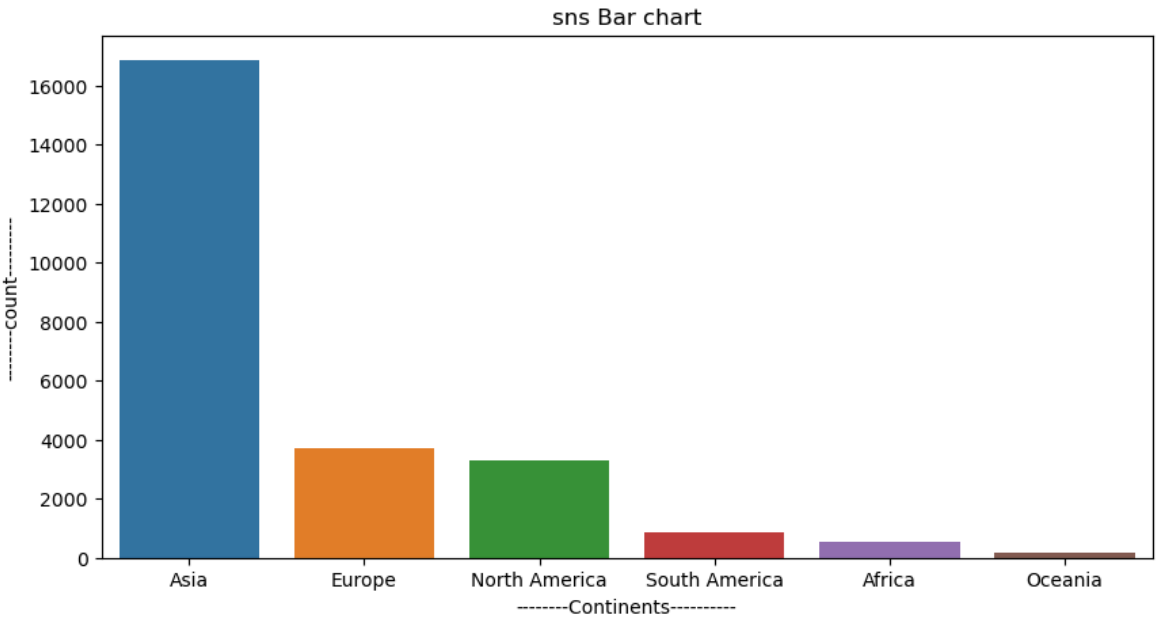


Cloud Plot

- Count plot from seaborn package
- It is also similar like bar chart only
- It is required only main data frame name and column name
- our main data frame name is :**visadf**
- column name : **continent**
- Seaborn count plot is easy compare to matlotlib bar chart
- If u want to plot bar chart with matplotlib we require 2 columns
- in case of seaborn 1 column categorical column is enough

In [54]:
```
cdf=visadf['continent'].value_counts()
keysInOrder=cdf.keys()   # keys are in order
```

In [55]:
```
cdf=visadf['continent'].value_counts()
keysInOrder=cdf.keys()   # keys are in order
plt.figure(figsize=[10,5])
sns.countplot(data=visadf,x='continent', order=keysInOrder)
plt.title("sns Bar chart")
plt.xlabel("--------Continents----------")
plt.ylabel("--------count---------")
plt.savefig("snsbarchart.png")
plt.show()
```



$Method - 3$

- we can create a plot from value counts directly
- Always keep in mind plotting is like a ocean
- Diffrent ppl has different ideas and methods
- based on requirement we can choose the methods

In [57]:
```
visadf['continent'].value_counts()
```

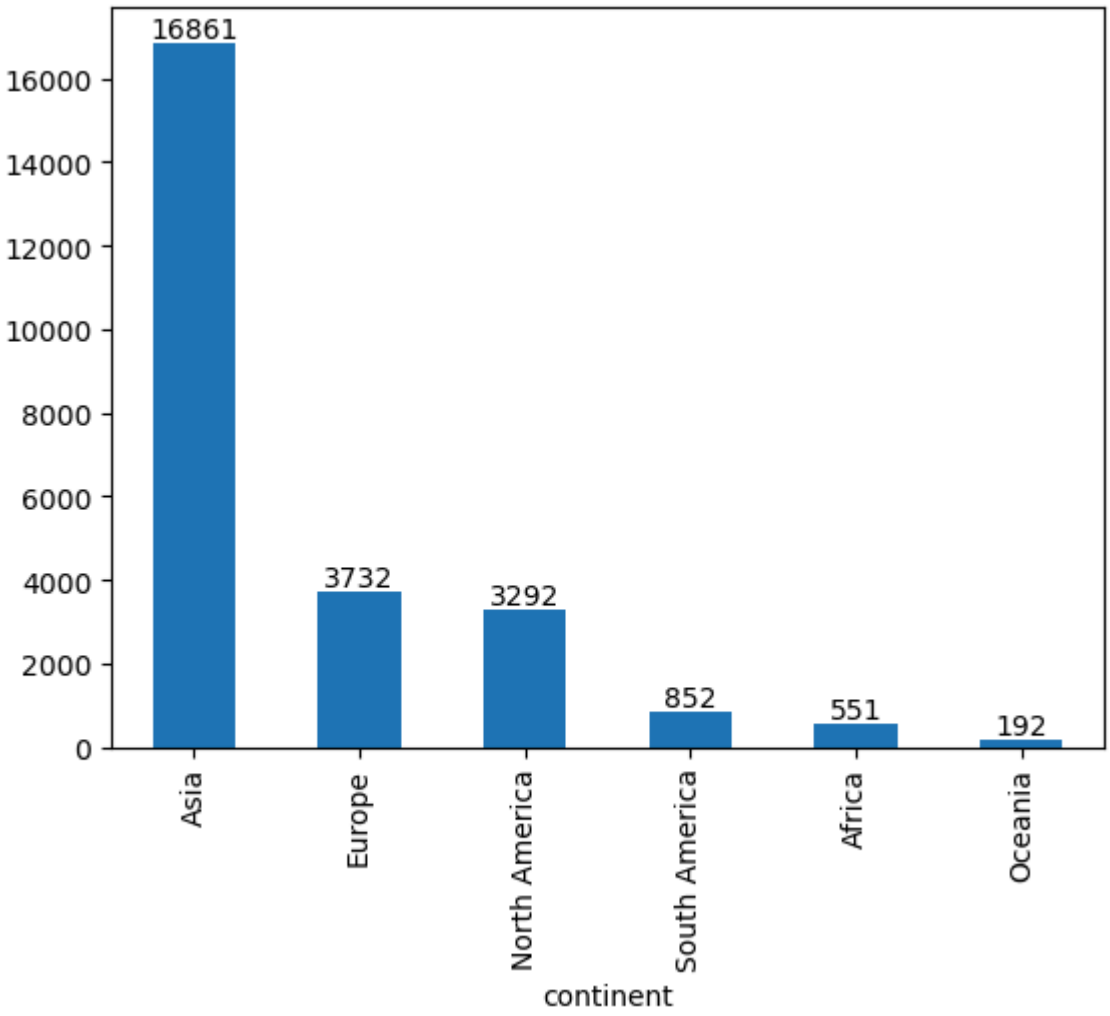
Out[57]:
```
continent
Asia             16861
Europe            3732
North America     3292
South America      852
Africa             551
Oceania            192
Name: count, dtype: int64
```

In [58]:
```
cdf = visadf['continent'].value_counts()
cdf.values
```

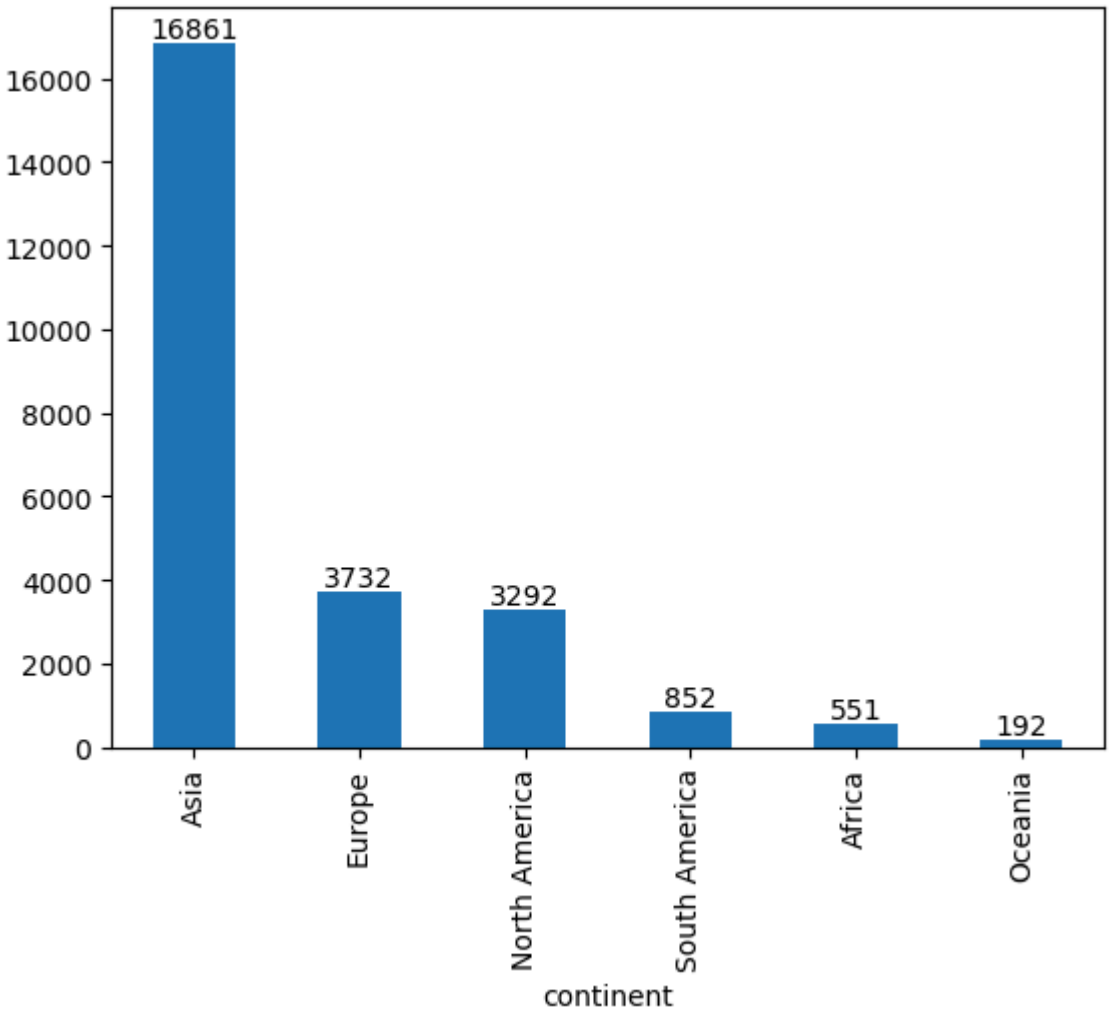
Out[58]:
```
array([16861,  3732,  3292,   852,   551,   192], dtype=int64)
```

In [72]:
```
cdf = visadf['continent'].value_counts()
ax=cdf.plot(kind='bar')
ax.bar_label(ax.containers[0])
```

Out[72]:
```
[Text(0, 0, '16861'),
 Text(0, 0, '3732'),
 Text(0, 0, '3292'),
 Text(0, 0, '852'),
 Text(0, 0, '551'),
 Text(0, 0, '192')]
```

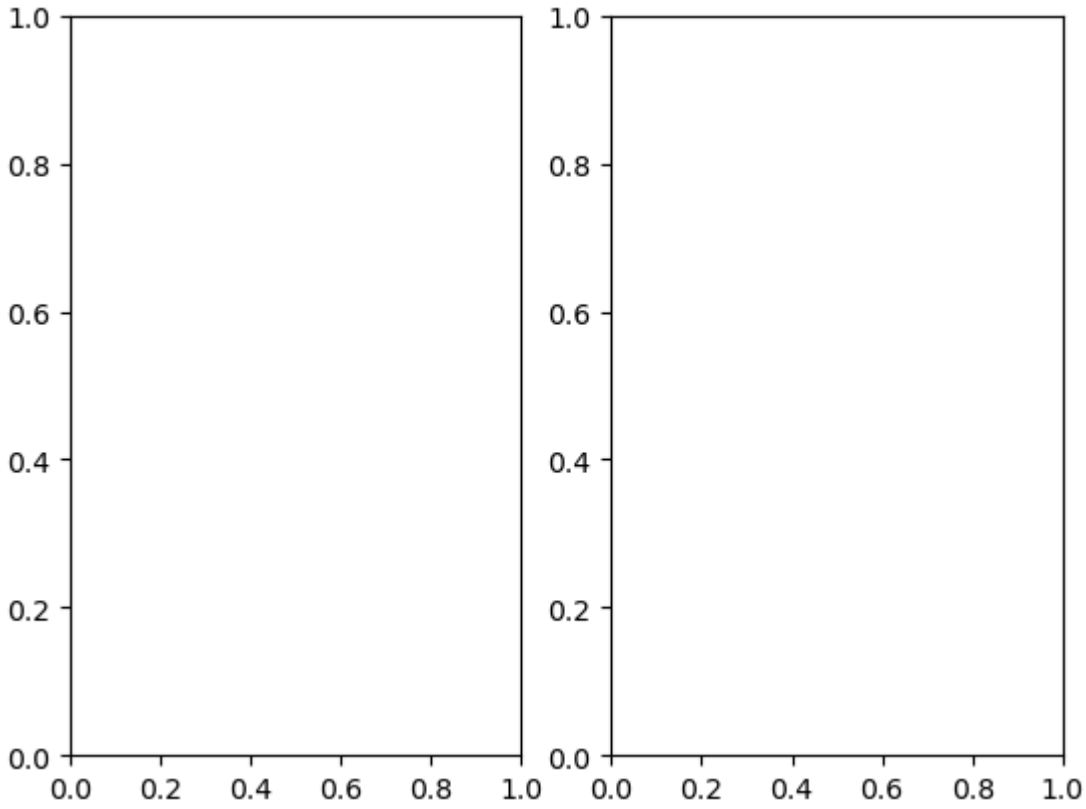


In [73]:
```
cdf = visadf['continent'].value_counts()
ax=cdf.plot(kind='bar')
ax.bar_label(ax.containers[0])
plt.show()
```



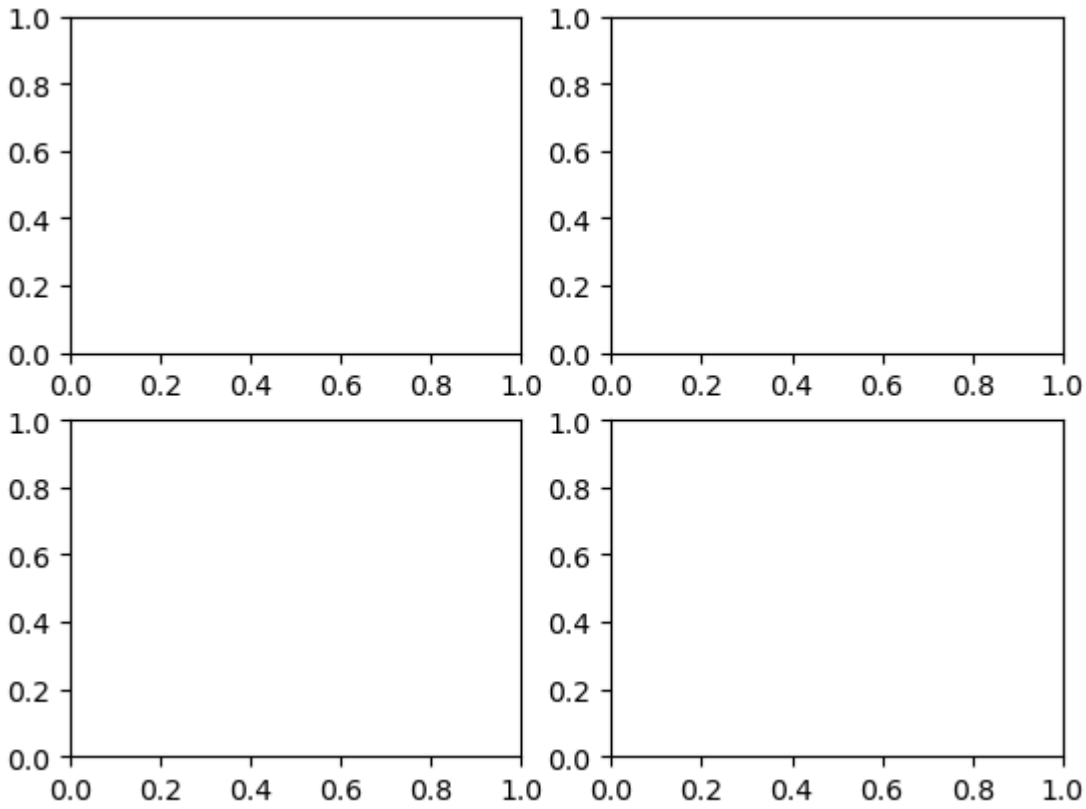
In [75]:
```
plt.subplot(1,2,1)     # 1 row, 2 columns
plt.subplot(1,2,2)     # 2 plots
```

Out[75]: <Axes: >

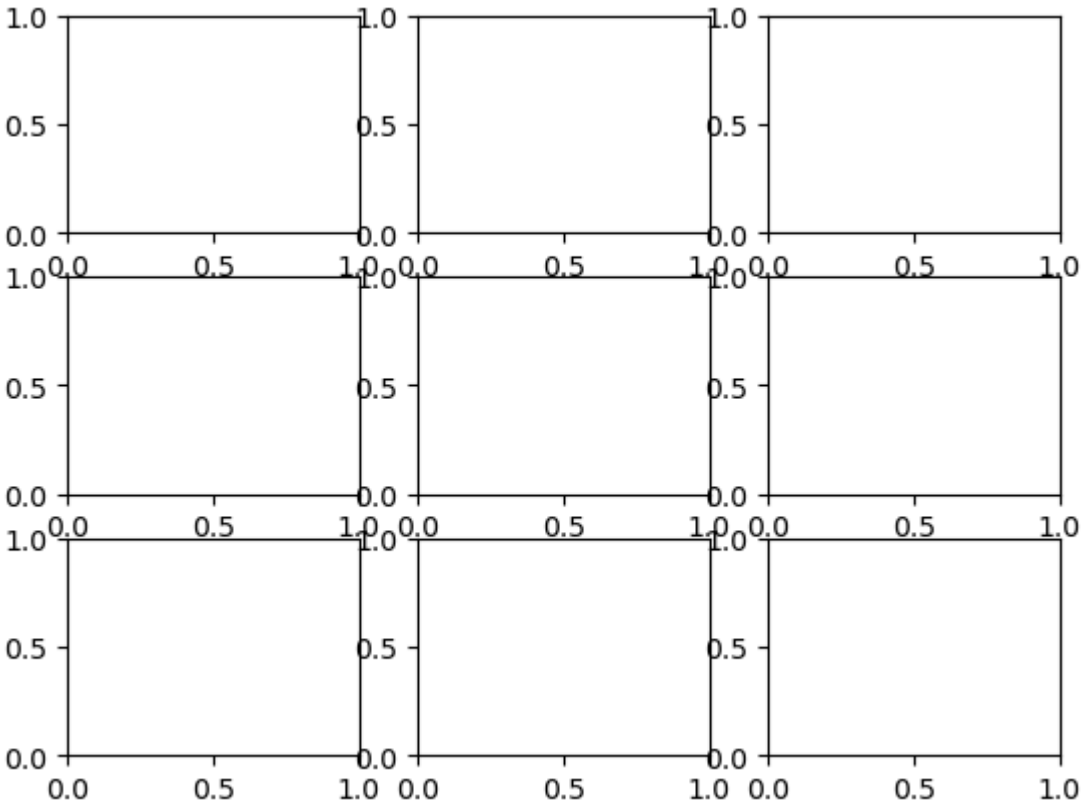


In [77]:
```
plt.subplot(2,2,1)
plt.subplot(2,2,2)
plt.subplot(2,2,3)
plt.subplot(2,2,4)
```

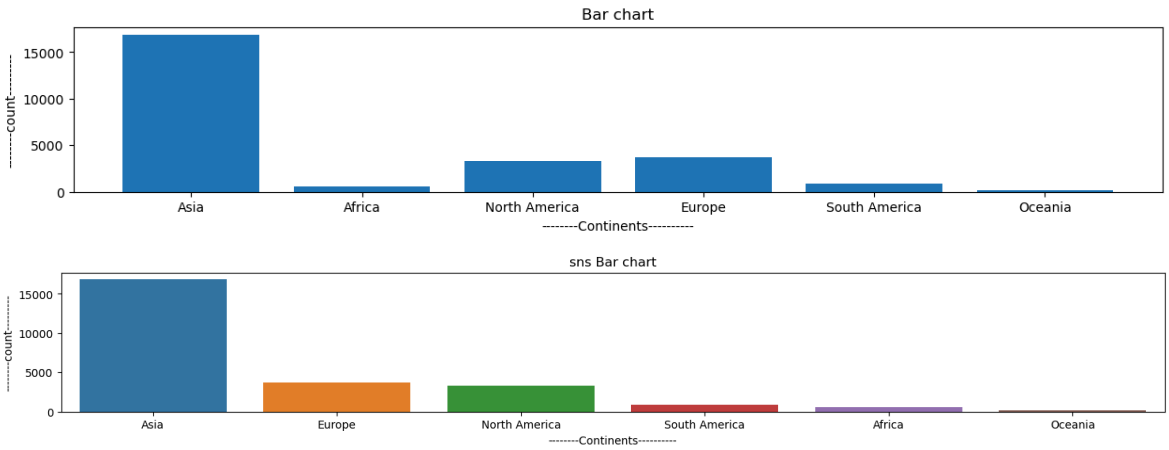
Out[77]: <Axes: >



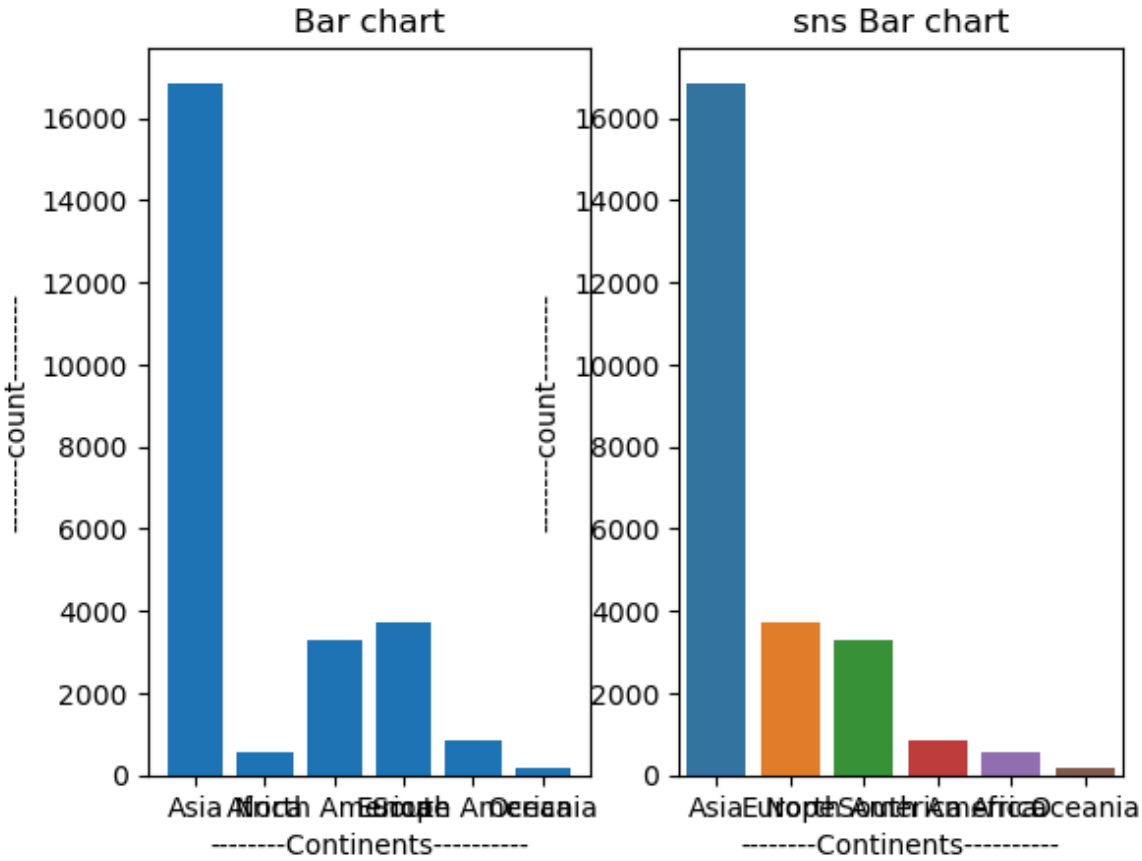
In [79]:
```
for i in range (1,10):
    plt.subplot(3,3,i)
```



In [86]:
```
plt.figure(figsize=[15,5])
plt.subplot(2,1,1)
plt.bar('continent','count',data=continentDataframe)
plt.title("Bar chart")
plt.xlabel("--------Continents----------")
plt.ylabel("--------count---------")
plt.savefig("barchart.png")
plt.show()

cdf=visadf['continent'].value_counts()
keysInOrder=cdf.keys()   # keys are in order
plt.figure(figsize=[18,5])
plt.subplot(2,1,2)
sns.countplot(data=visadf,x='continent', order=keysInOrder)
plt.title("sns Bar chart")
plt.xlabel("--------Continents----------")
plt.ylabel("--------count---------")
plt.savefig("snsbarchart.png")
plt.show()
```



In [105…
```
plt.subplot(1,2,1)
plt.bar('continent','count',data=continentDataframe)
plt.title("Bar chart")
plt.xlabel("--------Continents----------")
plt.ylabel("--------count---------")
plt.savefig("barchart.png")


cdf=visadf['continent'].value_counts()
keysInOrder=cdf.keys()   # keys are in order

plt.subplot(1,2,2)
sns.countplot(data=visadf,x='continent', order=keysInOrder)
plt.title("sns Bar chart")
plt.xlabel("--------Continents----------")
plt.ylabel("--------count---------")
plt.savefig("snsbarchart.png")
```



Relative Frequency

In [101…
```
relFre = visadf['continent'].value_counts(normalize=True)
```

In [104…
```
relFre
```

Out[104…
```
continent
Asia             0.661735
Europe           0.146468
North America    0.129199
South America    0.033438
Africa           0.021625
Oceania          0.007535
Name: proportion, dtype: float64
```

Pie Chart

- it has 360 degree view
- it proides % of values
- pie chart from **matplotlib**
- it requires keys and values, we can get from value counts

In [106…
```
visadf['continent'].value_counts()
```

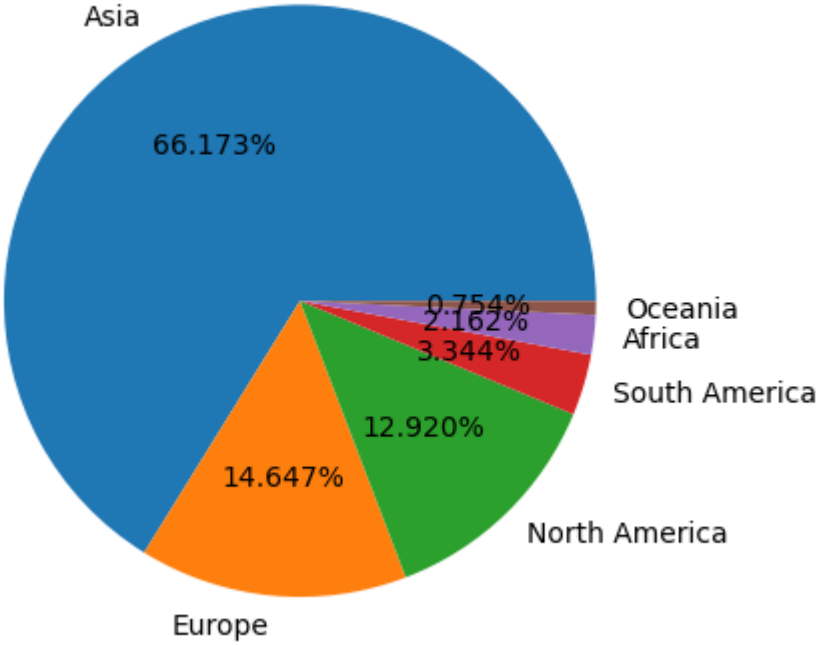
Out[106…
```
continent
Asia             16861
Europe            3732
North America     3292
South America      852
Africa             551
Oceania            192
Name: count, dtype: int64
```

In [110…
```
cdf = visadf['continent'].value_counts()
keys=cdf.keys()
```

In [111…
```
values=cdf.values
```

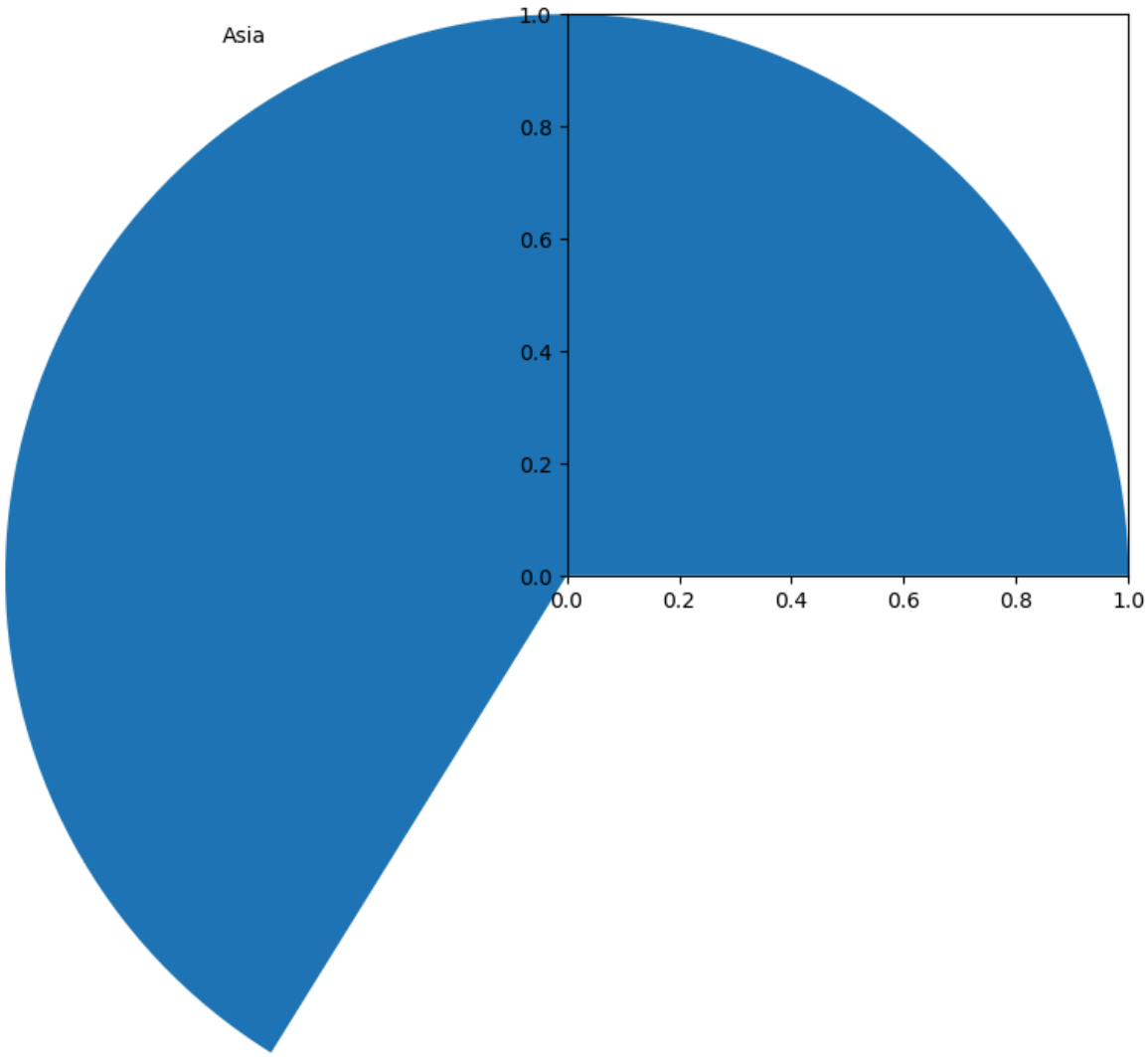
In [119…
```
plt.pie(values, labels=keys, autopct='%0.3f%%')
```

Out[119…
```
([<matplotlib.patches.Wedge at 0x1bbd7e92f50>,
  <matplotlib.patches.Wedge at 0x1bbd7e93350>,
  <matplotlib.patches.Wedge at 0x1bbd7e6ba90>,
  <matplotlib.patches.Wedge at 0x1bbd7e68bd0>,
  <matplotlib.patches.Wedge at 0x1bbd7ab6050>,
  <matplotlib.patches.Wedge at 0x1bbd829d690>],
 [Text(-0.5351743362316361, 0.9610350825224997, 'Asia'),
  Text(-0.10373513115748138, -1.0950977228374372, 'Europe'),
  Text(0.7670026411947619, -0.7884839557024984, 'North America'),
  Text(1.0546117976794491, -0.3127202522947962, 'South America'),
  Text(1.0926986108246142, -0.12652962460213996, 'Africa'),
  Text(1.0996917916121562, -0.026037731484255974, 'Oceania')],
 [Text(-0.2919132743081651, 0.5242009541031816, '66.173%'),
  Text(-0.05658279881317166, -0.597326030638602, '14.647%'),
  Text(0.4183650770153246, -0.4300821576559082, '12.920%'),
  Text(0.5752427987342449, -0.17057468306988885, '3.344%'),
  Text(0.5960174240861531, -0.0690161588738945, '2.162%'),
  Text(0.5998318863339034, -0.014202398991412348, '0.754%')])
```



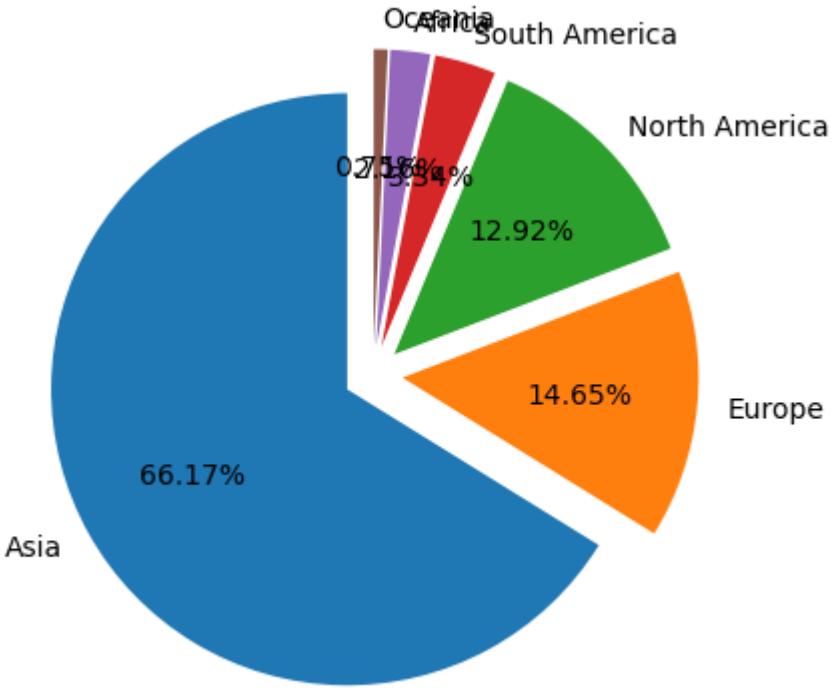
In [116…
```
plt.pie(values, labels=keys, autopct='%0.3f%')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[116], line 1
----> 1 plt.pie(values, labels=keys, autopct='%0.3f%')

File ~\anaconda3\Lib\site-packages\matplotlib\pyplot.py:3546, in pie(x, explode,
labels, colors, autopct, pctdistance, shadow, labeldistance, startangle, radius,
counterclock, wedgeprops, textprops, center, frame, rotatelabels, normalize, hatc
h, data)
   3523 @_copy_docstring_and_deprecators(Axes.pie)
   3524 def pie(
   3525     x: ArrayLike,
   (...)
   3544     data=None,
   3545 ) -> tuple[list[Wedge], list[Text]] | tuple[list[Wedge], list[Text], list
[Text]]:
-> 3546     return gca().pie(
   3547         x,
   3548         explode=explode,
   3549         labels=labels,
   3550         colors=colors,
   3551         autopct=autopct,
   3552         pctdistance=pctdistance,
   3553         shadow=shadow,
   3554         labeldistance=labeldistance,
   3555         startangle=startangle,
   3556         radius=radius,
   3557         counterclock=counterclock,
   3558         wedgeprops=wedgeprops,
   3559         textprops=textprops,
   3560         center=center,
   3561         frame=frame,
   3562         rotatelabels=rotatelabels,
   3563         normalize=normalize,
   3564         hatch=hatch,
   3565         **({"data": data} if data is not None else {}),
   3566     )

File ~\anaconda3\Lib\site-packages\matplotlib\__init__.py:1465, in _preprocess_da
ta.<locals>.inner(ax, data, *args, **kwargs)
   1462 @functools.wraps(func)
   1463 def inner(ax, *args, data=None, **kwargs):
   1464     if data is None:
-> 1465         return func(ax, *map(sanitize_sequence, args), **kwargs)
   1467     bound = new_sig.bind(ax, *args, **kwargs)
   1468     auto_label = (bound.arguments.get(label_namer)
   1469                   or bound.kwargs.get(label_namer))

File ~\anaconda3\Lib\site-packages\matplotlib\axes\_axes.py:3314, in Axes.pie(sel
f, x, explode, labels, colors, autopct, pctdistance, shadow, labeldistance, start
angle, radius, counterclock, wedgeprops, textprops, center, frame, rotatelabels,
normalize, hatch)
   3312 yt = y + pctdistance * radius * math.sin(thetam)
   3313 if isinstance(autopct, str):
-> 3314     s = autopct % (100. * frac)
   3315 elif callable(autopct):
   3316     s = autopct(100. * frac)

ValueError: incomplete format
```

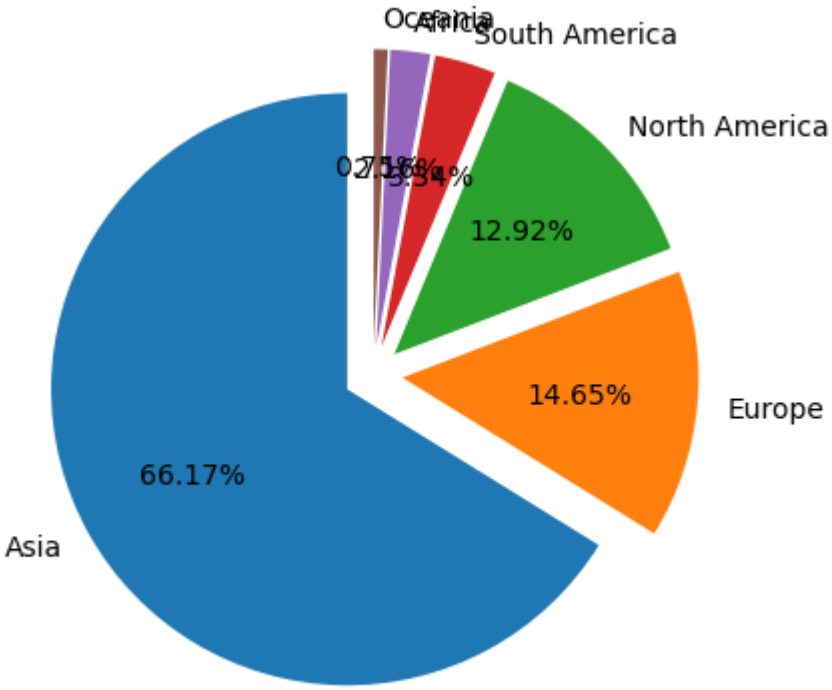


In [123…

```
plt.pie(values,explode=[0.1,0.1,0.1,0.1,0.1,0.1], labels=keys, autopct='%0.2f%%'
plt.show()
```



In [125…
```
plt.pie(values,explode=[0.1,0.1,0.1,0.1,0.1,0.1], labels=keys, autopct='%0.2f%%'
plt.show()
```



In [ ]: