Import packages

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Read the data

In [2]:
```
visa_df=pd.read_csv(r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\N
visa_df.head(2)
```

Out[2]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_training
0	EZYV01	Asia	High School	N	N
1	EZYV02	Asia	Master's	Y	N

continent

In [3]:
```
visa_df[['continent']] # Data frame type
```

Out[3]:

	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 [7]:
```
visa_df['continent'] # Series
```

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 [4]:
```
visa_df.continent  # Series
```

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 [5]:
```
visa_df[['continent']]  # df
visa_df['continent']    # series
visa_df.continent       # series
```

Out[5]:
```
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 [14]:
```
dir(visa_df['continent'])
```

Out[14]:
```
['T',
 '_AXIS_LEN',
 '_AXIS_ORDERS',
 '_AXIS_TO_AXIS_NUMBER',
 '_HANDLED_TYPES',
 '__abs__',
 '__add__',
 '__and__',
 '__annotations__',
 '__array__',
 '__array_priority__',
 '__array_ufunc__',
 '__bool__',
 '__class__',
 '__column_consortium_standard__',
 '__contains__',
 '__copy__',
 '__deepcopy__',
 '__delattr__',
 '__delitem__',
 '__dict__',
 '__dir__',
 '__divmod__',
 '__doc__',
 '__eq__',
 '__finalize__',
 '__float__',
 '__floordiv__',
 '__format__',
 '__ge__',
 '__getattr__',
 '__getattribute__',
 '__getitem__',
 '__getstate__',
 '__gt__',
 '__hash__',
 '__iadd__',
 '__iand__',
 '__ifloordiv__',
 '__imod__',
 '__imul__',
 '__init__',
 '__init_subclass__',
 '__int__',
 '__invert__',
 '__ior__',
 '__ipow__',
 '__isub__',
 '__iter__',
 '__itruediv__',
 '__ixor__',
 '__le__',
 '__len__',
 '__lt__',
 '__matmul__',
 '__mod__',
 '__module__',
 '__mul__',
 '__ne__',
 '__neg__',
```

```
'__new__',
'__nonzero__',
'__or__',
'__pandas_priority__',
'__pos__',
'__pow__',
'__radd__',
'__rand__',
'__rdivmod__',
'__reduce__',
'__reduce_ex__',
'__repr__',
'__rfloordiv__',
'__rmatmul__',
'__rmod__',
'__rmul__',
'__ror__',
'__round__',
'__rpow__',
'__rsub__',
'__rtruediv__',
'__rxor__',
'__setattr__',
'__setitem__',
'__setstate__',
'__sizeof__',
'__str__',
'__sub__',
'__subclasshook__',
'__truediv__',
'__weakref__',
'__xor__',
'_accessors',
'_accum_func',
'_agg_examples_doc',
'_agg_see_also_doc',
'_align_for_op',
'_align_frame',
'_align_series',
'_append',
'_arith_method',
'_as_manager',
'_attrs',
'_binop',
'_cacher',
'_can_hold_na',
'_check_inplace_and_allows_duplicate_labels',
'_check_is_chained_assignment_possible',
'_check_label_or_level_ambiguity',
'_check_setitem_copy',
'_clear_item_cache',
'_clip_with_one_bound',
'_clip_with_scalar',
'_cmp_method',
'_consolidate',
'_consolidate_inplace',
'_construct_axes_dict',
'_construct_result',
'_constructor',
'_constructor_expanddim',
```

```
'_constructor_expanddim_from_mgr',
'_constructor_from_mgr',
'_convert_dtypes',
'_data',
'_deprecate_downcast',
'_dir_additions',
'_dir_deletions',
'_drop_axis',
'_drop_labels_or_levels',
'_duplicated',
'_expanddim_from_mgr',
'_find_valid_index',
'_flags',
'_flex_method',
'_from_mgr',
'_get_axis',
'_get_axis_name',
'_get_axis_number',
'_get_axis_resolvers',
'_get_block_manager_axis',
'_get_bool_data',
'_get_cacher',
'_get_cleaned_column_resolvers',
'_get_index_resolvers',
'_get_label_or_level_values',
'_get_numeric_data',
'_get_rows_with_mask',
'_get_value',
'_get_values_tuple',
'_get_with',
'_getitem_slice',
'_gotitem',
'_hidden_attrs',
'_indexed_same',
'_info_axis',
'_info_axis_name',
'_info_axis_number',
'_init_dict',
'_init_mgr',
'_inplace_method',
'_internal_names',
'_internal_names_set',
'_is_cached',
'_is_copy',
'_is_label_or_level_reference',
'_is_label_reference',
'_is_level_reference',
'_is_mixed_type',
'_is_view',
'_item_cache',
'_ixs',
'_logical_func',
'_logical_method',
'_map_values',
'_maybe_update_cacher',
'_memory_usage',
'_metadata',
'_mgr',
'_min_count_stat_function',
'_name',
```

```
'_needs_reindex_multi',
'_pad_or_backfill',
'_protect_consolidate',
'_reduce',
'_references',
'_reindex_axes',
'_reindex_indexer',
'_reindex_multi',
'_reindex_with_indexers',
'_rename',
'_replace_single',
'_repr_data_resource_',
'_repr_latex_',
'_reset_cache',
'_reset_cacher',
'_set_as_cached',
'_set_axis',
'_set_axis_name',
'_set_axis_nocheck',
'_set_is_copy',
'_set_labels',
'_set_name',
'_set_value',
'_set_values',
'_set_with',
'_set_with_engine',
'_shift_with_freq',
'_slice',
'_stat_function',
'_stat_function_ddof',
'_take_with_is_copy',
'_to_latex_via_styler',
'_typ',
'_update_inplace',
'_validate_dtype',
'_values',
'_where',
'abs',
'add',
'add_prefix',
'add_suffix',
'agg',
'aggregate',
'align',
'all',
'any',
'apply',
'argmax',
'argmin',
'argsort',
'array',
'asfreq',
'asof',
'astype',
'at',
'at_time',
'attrs',
'autocorr',
'axes',
'backfill',
```

```
'between',
'between_time',
'bfill',
'bool',
'clip',
'combine',
'combine_first',
'compare',
'convert_dtypes',
'copy',
'corr',
'count',
'cov',
'cummax',
'cummin',
'cumprod',
'cumsum',
'describe',
'diff',
'div',
'divide',
'divmod',
'dot',
'drop',
'drop_duplicates',
'droplevel',
'dropna',
'dtype',
'dtypes',
'duplicated',
'empty',
'eq',
'equals',
'ewm',
'expanding',
'explode',
'factorize',
'ffill',
'fillna',
'filter',
'first',
'first_valid_index',
'flags',
'floordiv',
'ge',
'get',
'groupby',
'gt',
'hasnans',
'head',
'hist',
'iat',
'idxmax',
'idxmin',
'iloc',
'index',
'infer_objects',
'info',
'interpolate',
'is_monotonic_decreasing',
```

```
'is_monotonic_increasing',
'is_unique',
'isin',
'isna',
'isnull',
'item',
'items',
'keys',
'kurt',
'kurtosis',
'last',
'last_valid_index',
'le',
'loc',
'lt',
'map',
'mask',
'max',
'mean',
'median',
'memory_usage',
'min',
'mod',
'mode',
'mul',
'multiply',
'name',
'nbytes',
'ndim',
'ne',
'nlargest',
'notna',
'notnull',
'nsmallest',
'nunique',
'pad',
'pct_change',
'pipe',
'plot',
'pop',
'pow',
'prod',
'product',
'quantile',
'radd',
'rank',
'ravel',
'rdiv',
'rdivmod',
'reindex',
'reindex_like',
'rename',
'rename_axis',
'reorder_levels',
'repeat',
'replace',
'resample',
'reset_index',
'rfloordiv',
'rmod',
```

```
 'rmul',
 'rolling',
 'round',
 'rpow',
 'rsub',
 'rtruediv',
 'sample',
 'searchsorted',
 'sem',
 'set_axis',
 'set_flags',
 'shape',
 'shift',
 'size',
 'skew',
 'sort_index',
 'sort_values',
 'squeeze',
 'std',
 'str',
 'sub',
 'subtract',
 'sum',
 'swapaxes',
 'swaplevel',
 'tail',
 'take',
 'to_clipboard',
 'to_csv',
 'to_dict',
 'to_excel',
 'to_frame',
 'to_hdf',
 'to_json',
 'to_latex',
 'to_list',
 'to_markdown',
 'to_numpy',
 'to_period',
 'to_pickle',
 'to_sql',
 'to_string',
 'to_timestamp',
 'to_xarray',
 'transform',
 'transpose',
 'truediv',
 'truncate',
 'tz_convert',
 'tz_localize',
 'unique',
 'unstack',
 'update',
 'value_counts',
 'values',
 'var',
 'view',
 'where',
 'xs']
```

unique

In [6]:
```
visa_df['continent'].unique()

# unique method available only on series
```

Out[6]:
```
array(['Asia', 'Africa', 'North America', 'Europe', 'South America',
       'Oceania'], dtype=object)
```

In [7]:
```
len(visa_df['continent'].unique())
# 6 unique values are there
```

Out[7]:
```
6
```

In [ ]:
```
# In your visadf total 25480 rows are there
# In that 100 applicants name sarasvathi
```

nunique

In [8]:
```
visa_df['continent'].nunique()
```

Out[8]:
```
6
```

task-1

I want to know how many members from asia

In the entire data we have 25480 rows avialable , in that how many are from asia

In [9]:
```
visa_df['continent']=='Asia'
```

Out[9]:
```
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 [21]:
```
con=visa_df['continent']=='Asia'
visa_df[con]
```

Out[21]:

	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 [10]:
```
con=visa_df['continent']=='Asia'
len(visa_df[con])
```

Out[10]: 16861

In [11]:
```
unique_cnt=visa_df['continent'].unique()
for i in unique_cnt:
    con=visa_df['continent']==i
    val=len(visa_df[con])
    print(f"{i}:{val}")
```

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

In [12]:
```
unique_cnt=visa_df['continent'].unique()
count_list=[]
for i in unique_cnt:
    con=visa_df['continent']==i
    val=len(visa_df[con])
    count_list.append(val)
```

In [23]:
```
continent_df=pd.DataFrame(zip(unique_cnt,count_list),
            columns=['Continent','Count'])
continent_df
```

Out[23]:

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

In [14]:
```
continent_df.to_csv('continent_df.csv',index=False)
```

Value counts

In [21]:
```
cdf=visa_df['continent'].value_counts()
cdf
```

Out[21]:
```
continent
Asia             16861
Europe            3732
North America     3292
South America      852
Africa             551
Oceania            192
Name: count, dtype: int64
```

In [19]:
```
cdf.keys()
```

Out[19]:
```
Index(['Asia', 'Europe', 'North America', 'South America', 'Africa',
       'Oceania'],
      dtype='object', name='continent')
```

In [20]:
```
cdf.index
```

Out[20]:
```
Index(['Asia', 'Europe', 'North America', 'South America', 'Africa',
       'Oceania'],
      dtype='object', name='continent')
```

In [22]:
```
cdf.values
```

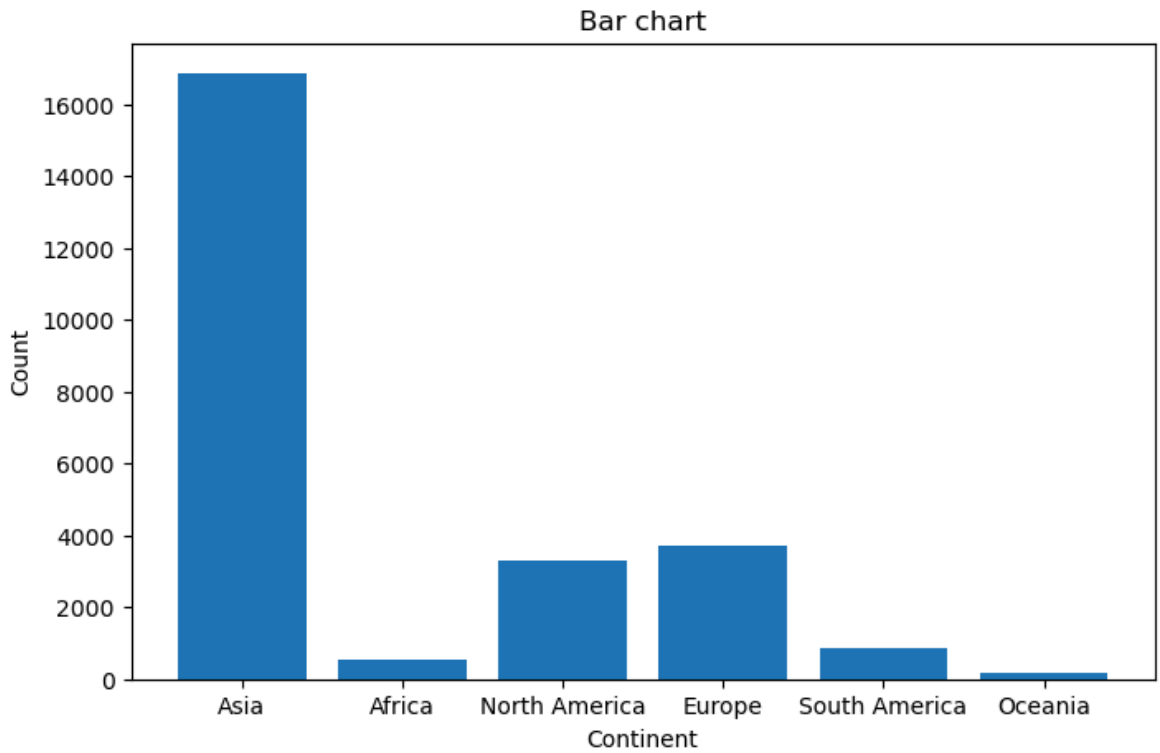
Out[22]:
```
array([16861,  3732,  3292,   852,   551,   192], dtype=int64)
```

Bar Chart

- Bar chart is representation of counts w.r.t classes
- If we want plot bar chart we required two columns
  - One column is categorical data column
  - another column is Numerical data column
- we already created a dataframe with continents **continent_df**

- It has two columns
  - Continent
  - Count
- Package : **matplotlib**

In [33]:
```
plt.figure(figsize=(8,5)) # Change figure lay out
plt.bar('Continent',
        'Count',
        data=continent_df)  # plot
plt.title("Bar chart")     # title of the plot
plt.xlabel("Continent")    # X-axis name
plt.ylabel("Count")        # y-axis name
plt.savefig("barchart.jpg") # Save the figure in jpg
plt.savefig("barchart.png") # Save the figure in png
plt.show()                  # show will always at last
```



Count plot

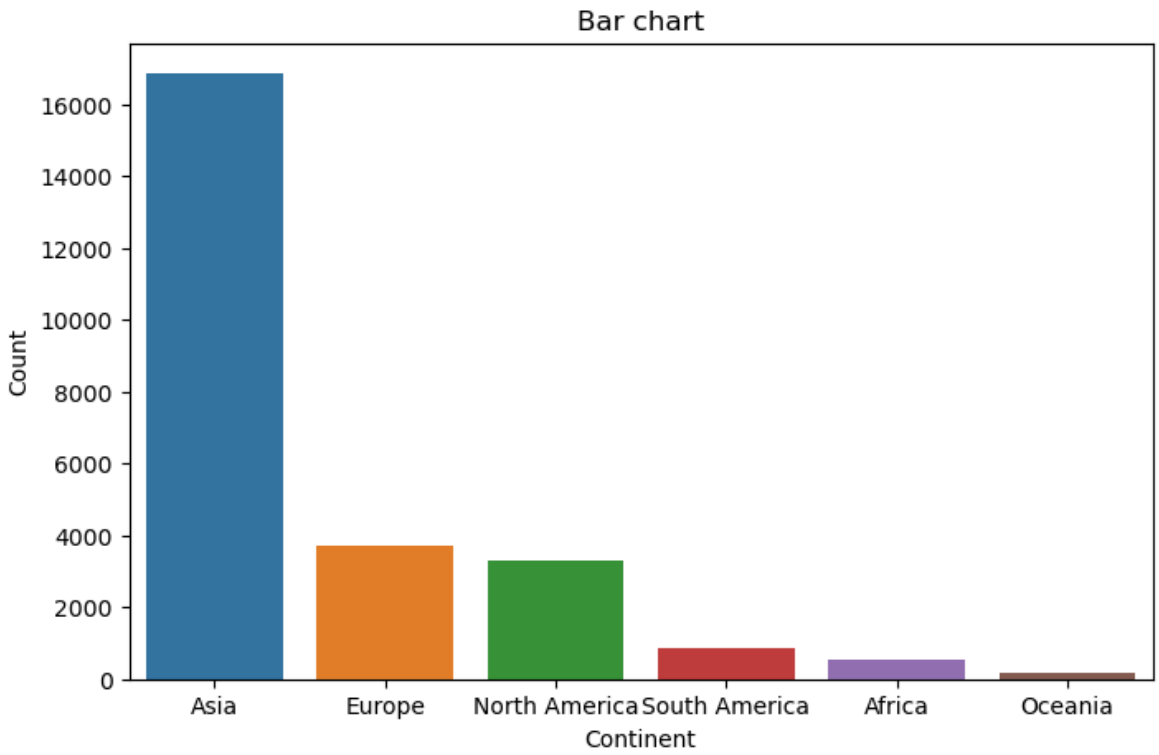
- Count plot from seaborn package
- It is also similar like bar chart only
- It is required only main data frame name
- And Column name
- Our main data frame name is : **visa_df**
- column name: **continent**

- Seaborn count plot is easy compare to matplotlib bar chart
- If you want plot bar chart with matplotlib we required two columns
- but Seaborn one column categorical column is enough

In [41]:
```
cdf=visa_df['continent'].value_counts()
cdf.keys()
```

Out[41]:
```
Index(['Asia', 'Europe', 'North America', 'South America', 'Africa',
       'Oceania'],
      dtype='object', name='continent')
```

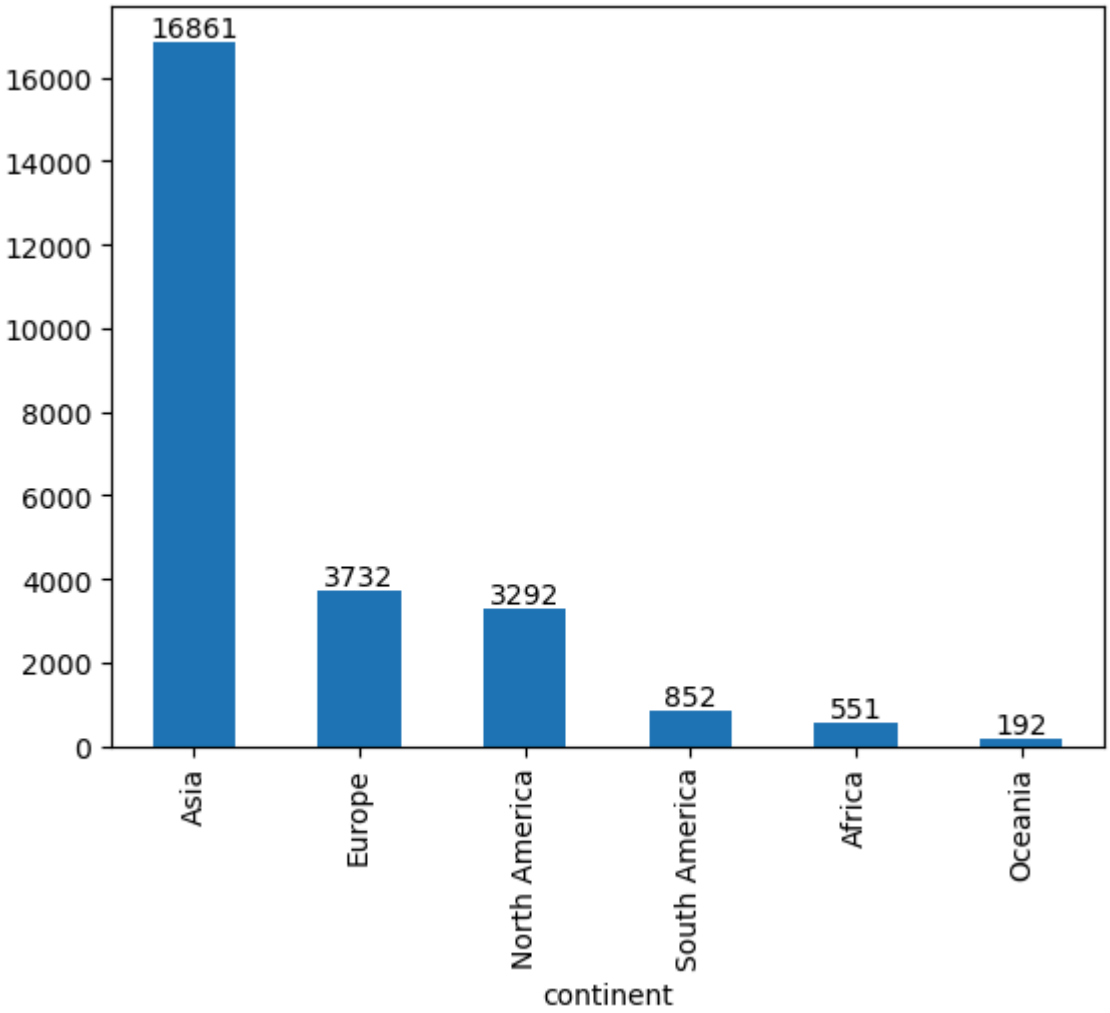
In [50]:
```
cdf=visa_df['continent'].value_counts()
keys=cdf.keys()
plt.figure(figsize=(8,5))
sns.countplot(data=visa_df,
              x='continent',
             order=keys)
plt.title("Bar chart")
plt.xlabel("Continent")
plt.ylabel("Count")
plt.show()
```



Method-3

- we can create a plot from value counts directly
- Always keep in mind , plotting is like a ocean
- Different people has different ideas
- And different methods are available
- Based on requirement we can choose the methods

In [55]:
```
cdf=visa_df['continent'].value_counts()
ax=cdf.plot(kind='bar')
ax.bar_label(ax.containers[0])
plt.show()
```



In [54]:
```
#####################=== Matplotlib=======##########################
plt.figure(figsize=(8,5)) # Change figure lay out
plt.bar('Continent',
        'Count',
        data=continent_df)  # plot
plt.title("Bar chart")     # title of the plot
plt.xlabel("Continent")    # X-axis name
plt.ylabel("Count")        # y-axis name
plt.savefig("barchart.jpg") # Save the figure in jpg
plt.savefig("barchart.png") # Save the figure in png
plt.show()

#################============Sear born=============#######################
cdf=visa_df['continent'].value_counts()
keys=cdf.keys()
plt.figure(figsize=(8,5))
sns.countplot(data=visa_df,
              x='continent',
             order=keys)
plt.title("Bar chart")
plt.xlabel("Continent")
plt.ylabel("Count")
```

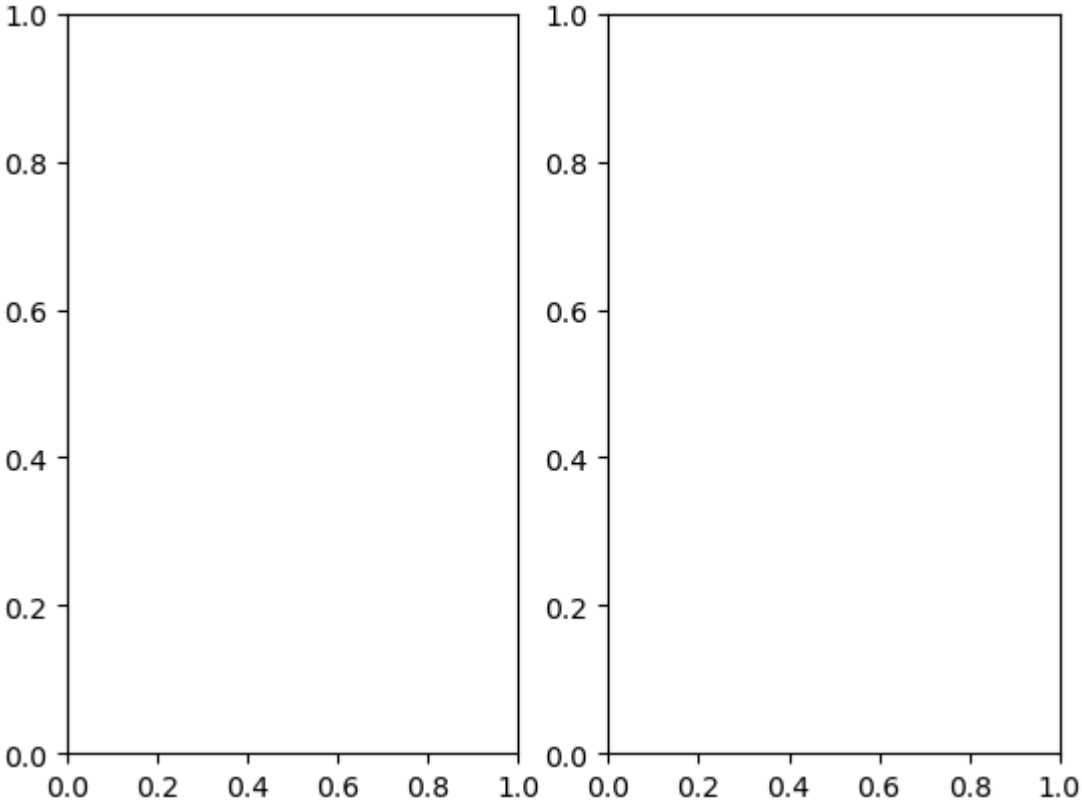
```
plt.show()

#######################================Value counts=================##########
cdf=visa_df['continent'].value_counts()
ax=cdf.plot(kind='bar')
ax.bar_label(ax.containers[0])
plt.show()
```

Out[54]: 0

Subplots

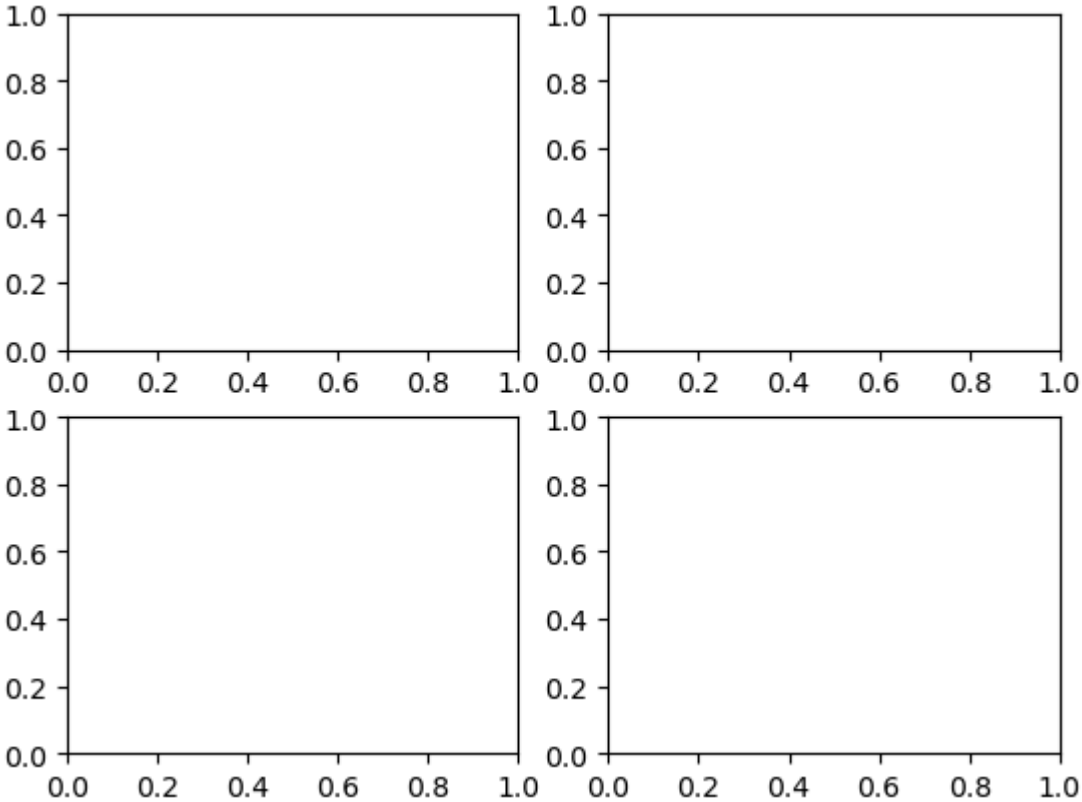
In [56]:
```
plt.subplot(1,2,1)
plt.subplot(1,2,2)

# (1,2)  1 row  and 2 columns
# How many plots 2 plots
```

Out[56]: <Axes: >

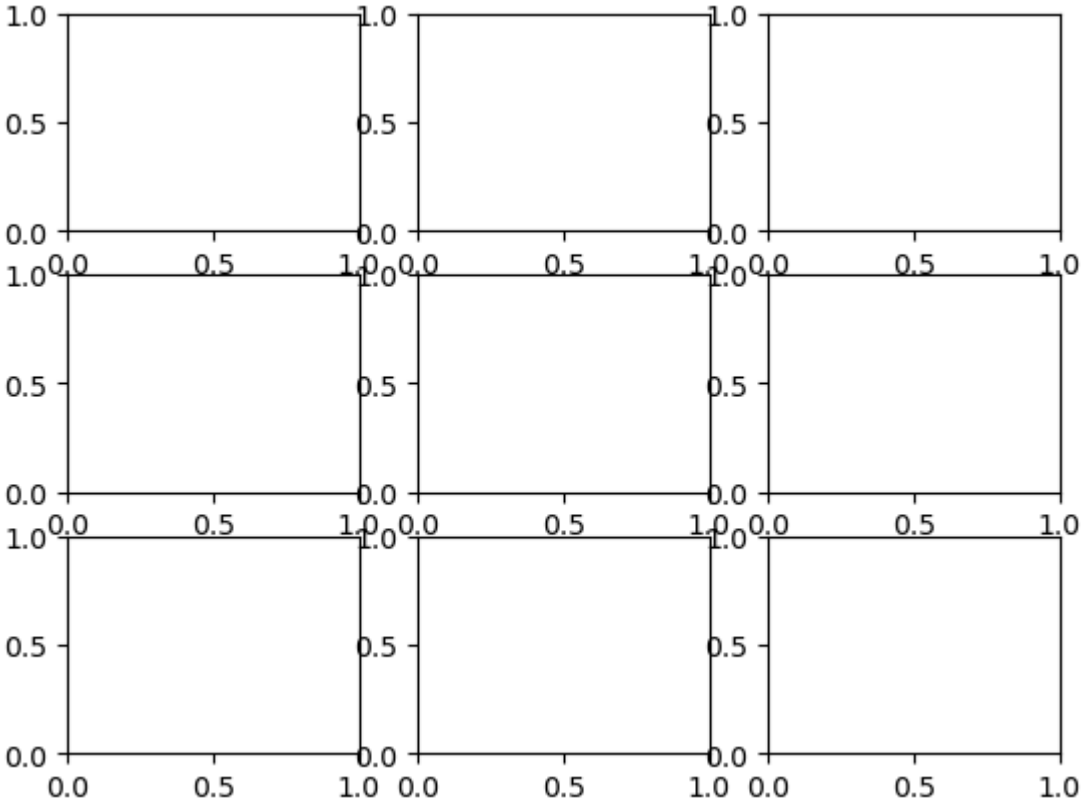


In [57]:
```
plt.subplot(2,2,1)
plt.subplot(2,2,2)
plt.subplot(2,2,3)
plt.subplot(2,2,4)
```

Out[57]: <Axes: >

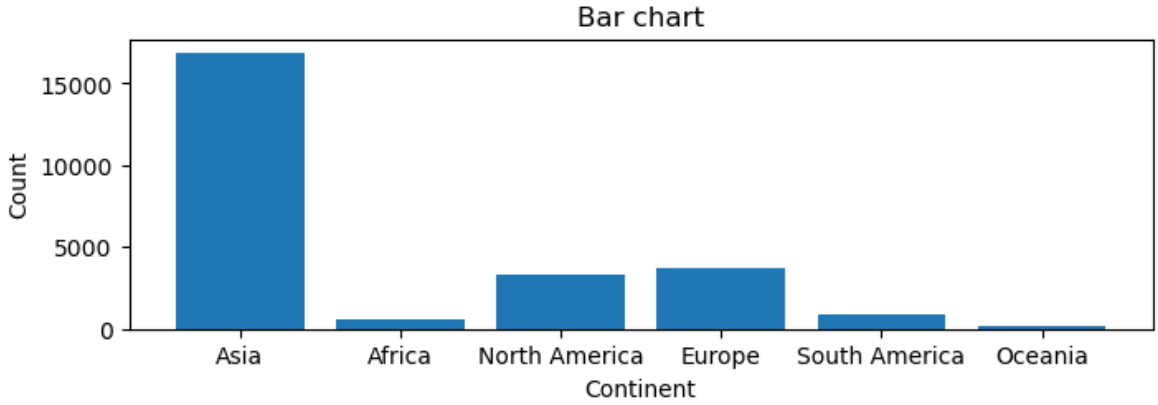


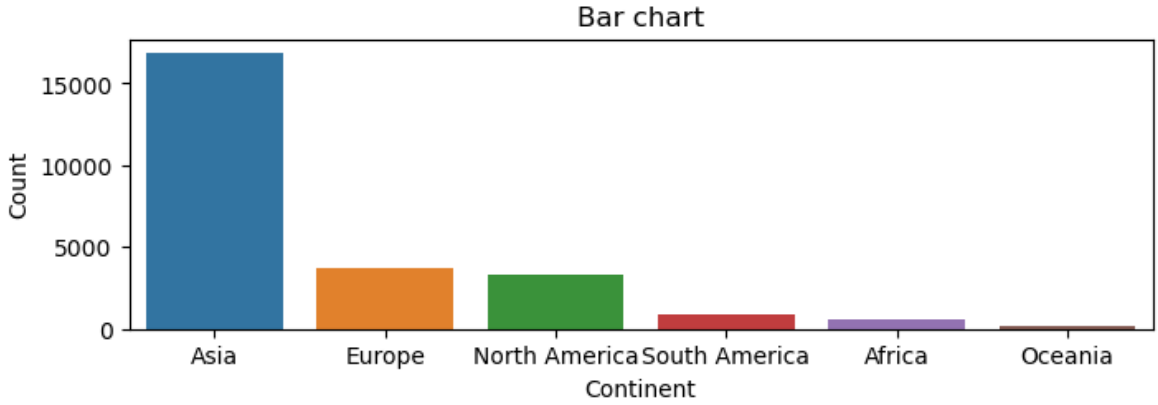
In [58]:
```
for i in range(1,10):
    plt.subplot(3,3,i)
```



In [62]:
```
#####################=== Matplotlib=======##########################
plt.figure(figsize=(8,5)) # Change figure lay out
plt.subplot(2,1,1)
plt.bar('Continent',
        'Count',
        data=continent_df)  # plot
```

```
plt.title("Bar chart")      # title of the plot
plt.xlabel("Continent")     # X-axis name
plt.ylabel("Count")         # y-axis name
plt.savefig("barchart.jpg") # Save the figure in jpg
plt.savefig("barchart.png") # Save the figure in png
plt.show()

################============Sear born============#######################
cdf=visa_df['continent'].value_counts()
keys=cdf.keys()
plt.figure(figsize=(8,5))
plt.subplot(2,1,2)
sns.countplot(data=visa_df,
              x='continent',
             order=keys)
plt.title("Bar chart")
plt.xlabel("Continent")
plt.ylabel("Count")
plt.show()
```



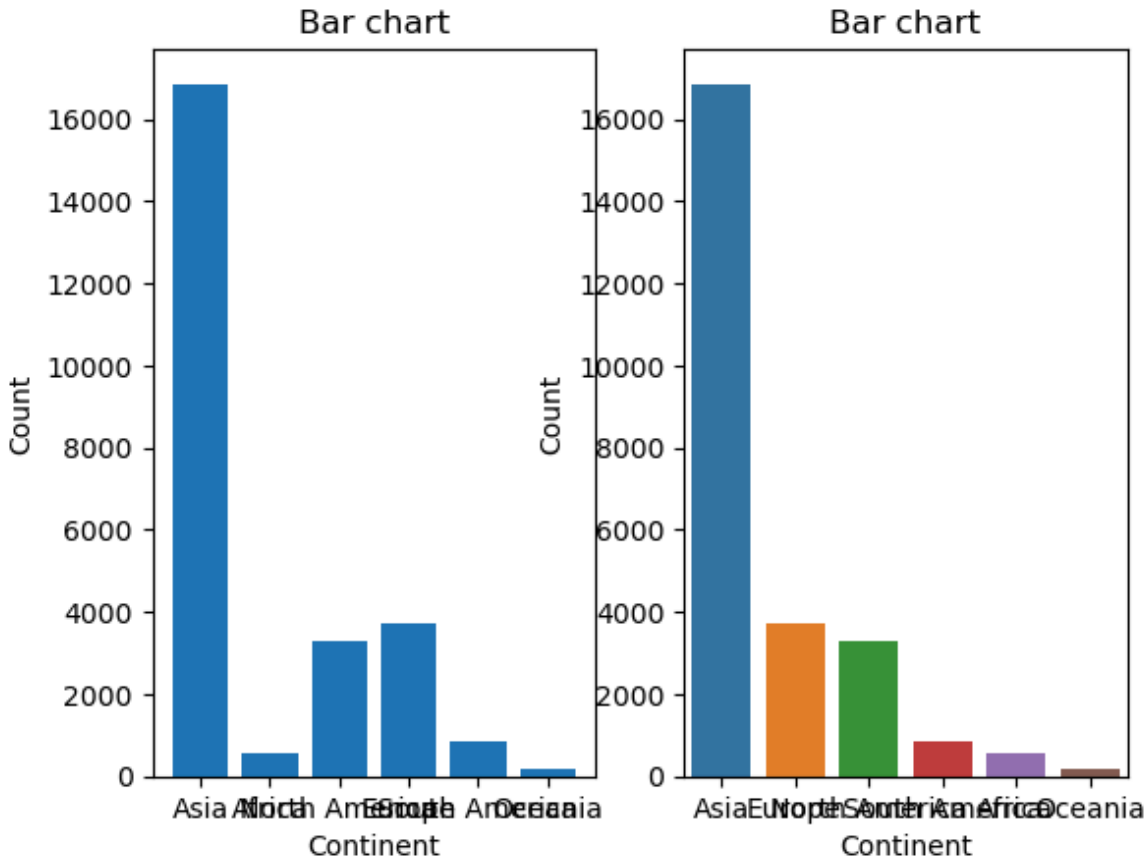


In [67]:

```
#####################=== Matplotlib=======##########################
 # Change figure lay out

plt.subplot(1,2,1).bar('Continent',
        'Count',
        data=continent_df)  # plot
plt.title("Bar chart")      # title of the plot
plt.xlabel("Continent")     # X-axis name
plt.ylabel("Count")         # y-axis name
plt.savefig("barchart.jpg") # Save the figure in jpg
plt.savefig("barchart.png") # Save the figure in png

################============Sear born============#######################
cdf=visa_df['continent'].value_counts()
```

```
keys=cdf.keys()
plt.subplot(1,2,2)
sns.countplot(data=visa_df,
              x='continent',
             order=keys)
plt.title("Bar chart")
plt.xlabel("Continent")
plt.ylabel("Count")
```

Out[67]: Text(0, 0.5, 'Count')



Relative frequency

- Frequency tabels values provides in percentages

In [69]:
```
visa_df['continent'].value_counts(normalize=True)
```

Out[69]:
```
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

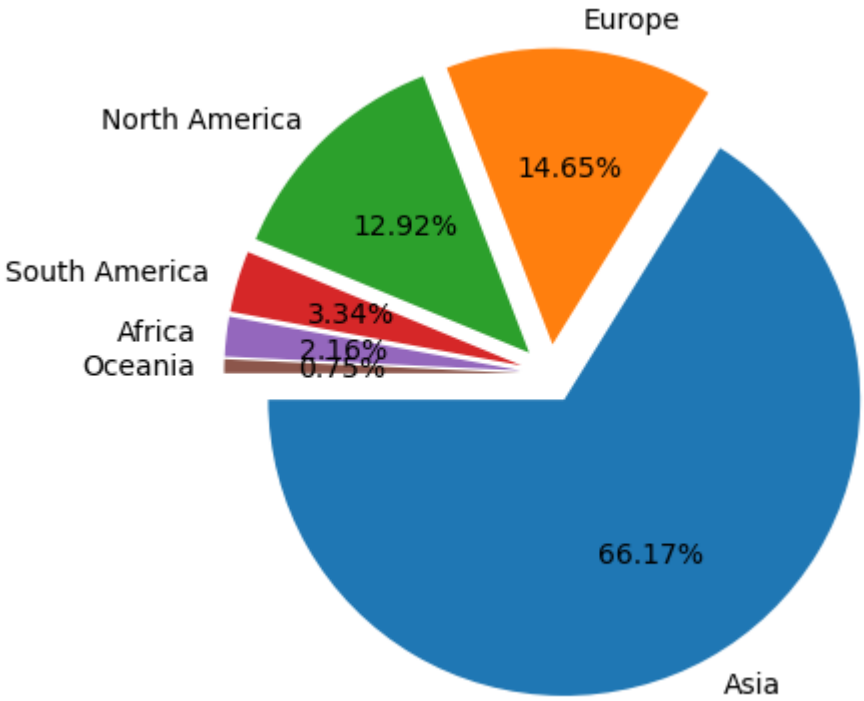
- Pie chart has 360 degrees view
- It provides percentage of vales

- Pie chart from **matplotlib**
- It requires keys and values , we can get from value counts

In [71]:
```
cdf=visa_df['continent'].value_counts()
keys=cdf.keys()
values=cdf.values
keys,values
```

Out[71]:
```
(Index(['Asia', 'Europe', 'North America', 'South America', 'Africa',
        'Oceania'],
       dtype='object', name='continent'),
 array([16861,  3732,  3292,   852,   551,   192], dtype=int64))
```

In [85]:
```
plt.pie(values,
        explode=[0.1,0.1,0.1,0.1,0.1,0.1],
        labels=keys,
       autopct="%0.2f%%",
       startangle=180,
       radius=1)
plt.show()
```



In [ ]: