In [1]:

```python
# Task:Perform clustering (Both hierarchical and K means clustering) for the airl
```

1. Import Libraries

In [2]:

```python
import pandas as pd
from scipy.cluster.hierarchy import linkage
import scipy.cluster.hierarchy as sch
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

2. Import Data set

In [3]:

```python
xlc=pd.ExcelFile('EastWestAirlines.xlsx')
data=pd.read_excel(xlc,'data')
data.head()
```

Out[3]:

	ID#	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_r
0	1	28143	0	1	1	1	174	1	
1	2	19244	0	1	1	1	215	2	
2	3	41354	0	1	1	1	4123	4	
3	4	14776	0	1	1	1	500	1	
4	5	97752	0	4	1	1	43300	26	

3. Data Understanding

In [4]:

```python
data.shape #How many columns and rows in data
```

Out[4]:

```
(3999, 12)
```

In [5]:

```python
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3999 entries, 0 to 3998
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   ID#                3999 non-null   int64
 1   Balance            3999 non-null   int64
 2   Qual_miles         3999 non-null   int64
 3   cc1_miles          3999 non-null   int64
 4   cc2_miles          3999 non-null   int64
 5   cc3_miles          3999 non-null   int64
 6   Bonus_miles        3999 non-null   int64
 7   Bonus_trans        3999 non-null   int64
 8   Flight_miles_12mo  3999 non-null   int64
 9   Flight_trans_12    3999 non-null   int64
 10  Days_since_enroll  3999 non-null   int64
 11  Award?             3999 non-null   int64
dtypes: int64(12)
memory usage: 375.0 KB
```

In [6]:

```
data.isnull().sum()   # is any null is available in data
```

Out[6]:

```
ID#                  0
Balance              0
Qual_miles           0
cc1_miles            0
cc2_miles            0
cc3_miles            0
Bonus_miles          0
Bonus_trans          0
Flight_miles_12mo    0
Flight_trans_12      0
Days_since_enroll    0
Award?               0
dtype: int64
```

In [7]:

```
data.describe()
```

Out[7]:

	ID#	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_n
count	3999.000000	3.999000e+03	3999.000000	3999.000000	3999.000000	3999.000000	3999.00
mean	2014.819455	7.360133e+04	144.114529	2.059515	1.014504	1.012253	17144.84
std	1160.764358	1.007757e+05	773.663804	1.376919	0.147650	0.195241	24150.96
min	1.000000	0.000000e+00	0.000000	1.000000	1.000000	1.000000	0.00
25%	1010.500000	1.852750e+04	0.000000	1.000000	1.000000	1.000000	1250.00
50%	2016.000000	4.309700e+04	0.000000	1.000000	1.000000	1.000000	7171.00
75%	3020.500000	9.240400e+04	0.000000	3.000000	1.000000	1.000000	23800.50
max	4021.000000	1.704838e+06	11148.000000	5.000000	3.000000	5.000000	263685.00

4.Data Preparing

In [8]:

```
# Normalization function

def norm_fun(i):
    x=(i-i.min())/(i.max()-i.min())
    return x
```

In [9]:

```
data_norm=norm_fun(data.iloc[:,1:])
```

In [10]:

```
data_norm.head()
```

Out[10]:

	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_miles_
0	0.016508	0.0	0.00	0.0	0.0	0.000660	0.011628	0.0
1	0.011288	0.0	0.00	0.0	0.0	0.000815	0.023256	0.0
2	0.024257	0.0	0.00	0.0	0.0	0.015636	0.046512	0.0
3	0.008667	0.0	0.00	0.0	0.0	0.001896	0.011628	0.0
4	0.057338	0.0	0.75	0.0	0.0	0.164211	0.302326	0.0

5. Dendogram Representation

In [11]:

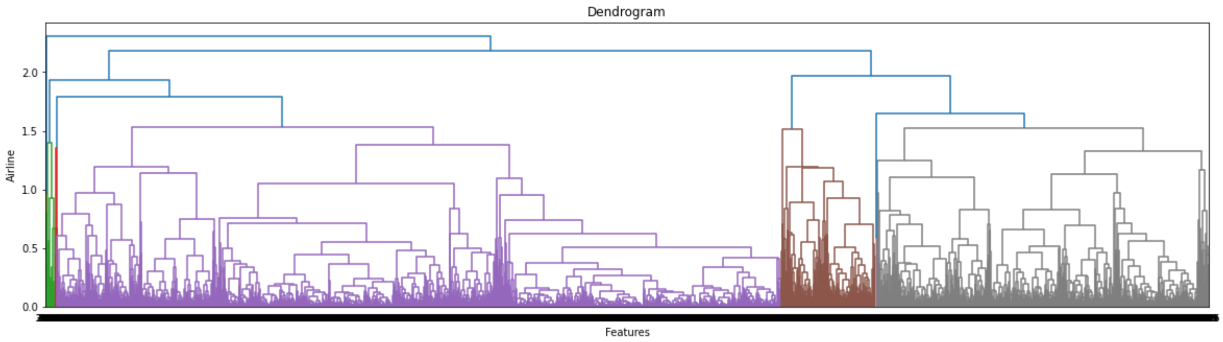
```
z=linkage(data_norm,method='complete', metric='euclidean')
```

In [12]:

```python
plt.figure(figsize=(20, 5))

plt.title('Dendrogram')
plt.xlabel('Features')
plt.ylabel('Airline')

sch.dendrogram(z,leaf_font_size=8.,leaf_rotation=0)
plt.show()  #creating dendrogram
```



6. elbow curve

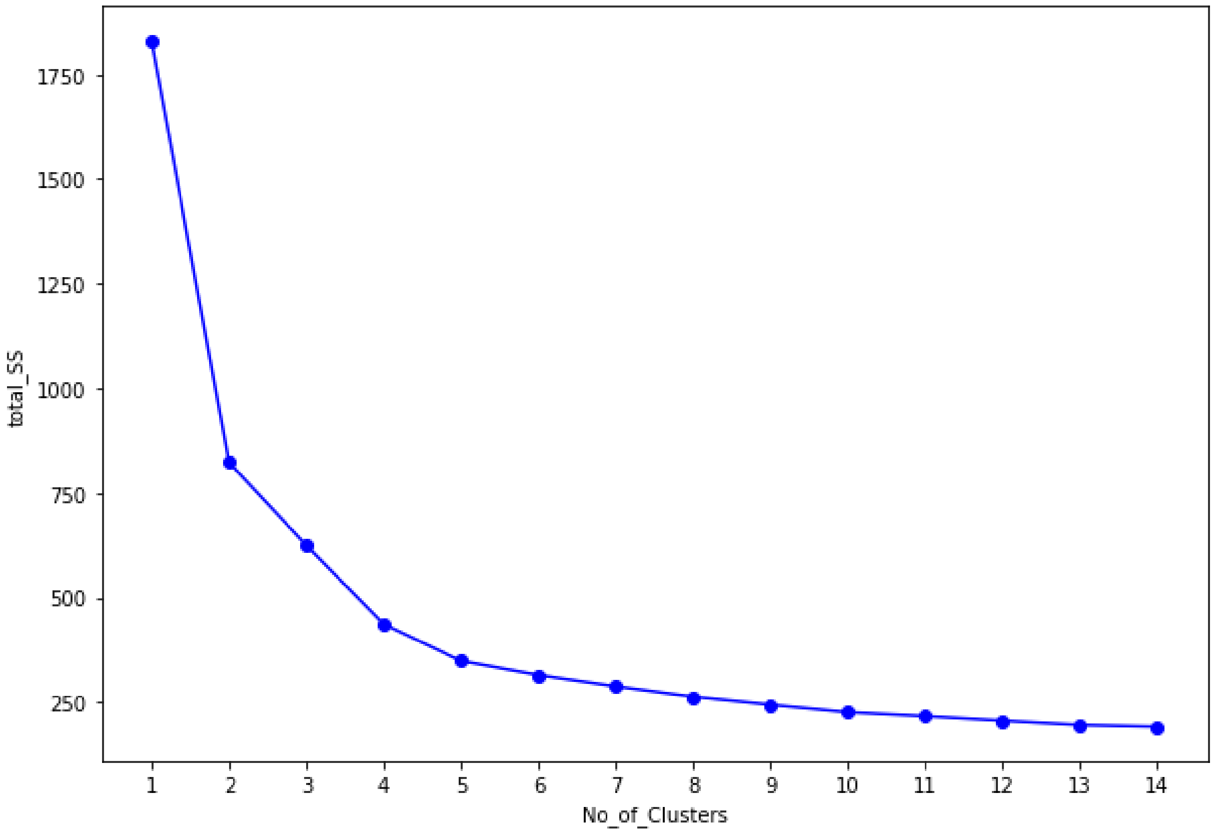
In [13]:

```python
######  elbow curve ###########
wcss=[]
for i in range(1,15):
    knn=KMeans(n_clusters=i)
    knn.fit(data_norm)
    wcss.append(knn.inertia_) #variable for storing inertia value of each kmeans
print(wcss)
```

```
[1830.7932128584155, 823.6756984125224, 625.1684881570746, 436.7088576193263, 3
48.9433217254146, 315.1564691705778, 287.9418214739785, 263.2045343320302, 243.
91697150847332, 226.10985005073985, 216.67873043910373, 205.88213083111705, 19
5.72766812710302, 191.74210176410708]
```

In [14]:

```python
plt.figure(figsize=(10,7))
plt.plot(range(1,15),wcss,'bo-')
plt.xlabel("No_of_Clusters")
plt.ylabel("total_SS")
plt.xticks(range(1,15))
plt.show()
```



7. build Model

In [15]:
```
#taking Cluster =4
```

In [16]:
```
X1 = data[['Balance','Qual_miles','cc1_miles','cc2_miles','cc3_miles','Bonus_mile
cluster=KMeans(4)
cluster.fit(data_norm)
cluster.labels_
data['cluster']=cluster.labels_
data.head()
```

Out[16]:

	ID#	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Flight_r
0	1	28143	0	1	1	1	174	1	
1	2	19244	0	1	1	1	215	2	
2	3	41354	0	1	1	1	4123	4	
3	4	14776	0	1	1	1	500	1	
4	5	97752	0	4	1	1	43300	26	

In [17]:
```
data.sort_values(by='cluster',ascending=True)
data.sort_values(by='Days_since_enroll',ascending=True)
```

Out[17]:

	ID#	Balance	Qual_miles	cc1_miles	cc2_miles	cc3_miles	Bonus_miles	Bonus_trans	Fli
3697	3720	972	972	1	1	1	0	0	
3696	3719	862	0	1	1	1	0	0	
3722	3745	3230	0	1	1	1	0	0	
3725	3748	2627	0	1	1	1	0	0	
3747	3770	6015	4929	1	1	1	0	0	
...	...	...	...	...	...	...	...	...	
410	416	620498	0	5	1	1	25395	53	
409	415	10732	0	1	1	1	1296	6	
408	414	5581	0	1	1	1	0	0	
415	421	109087	0	2	1	1	10462	16	
393	399	16999	0	1	1	1	140	1	

3999 rows × 13 columns

8. Find Out Cluster & Centers

In [18]:
```
# Selecting 4 clusters from the above  plot which is the optimum number of cluste

model=KMeans(n_clusters=4)
model.fit(data_norm)
```

Out[18]:
```
KMeans(n_clusters=4)
```

In [19]:
```
model.labels_ #indicate from which group data is belong to
```

Out[19]:
```
array([1, 1, 1, ..., 2, 1, 1])
```

```
In [20]: model.cluster_centers_ # indicate center point of 4 cluster for all dimension
```

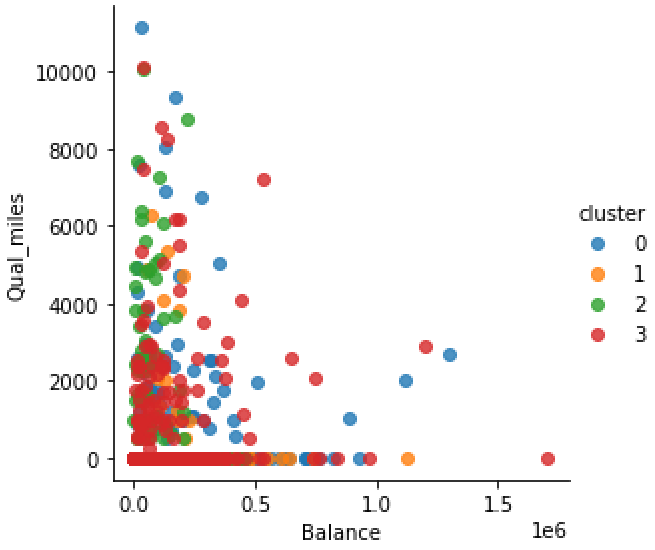
```
Out[20]: array([[ 4.89953609e-02,  2.60542873e-02,  3.90044577e-02,
                  1.63447251e-02,  2.22882615e-03,  3.35642727e-02,
                  1.21825219e-01,  3.34267751e-02,  5.94073285e-02,
                  5.22892182e-01,  1.00000000e+00],
                [ 2.39011667e-02,  8.28362120e-03,  2.31945177e-02,
                  8.96151819e-03,  1.05429626e-03,  1.26482465e-02,
                  7.54496083e-02,  7.35308092e-03,  1.24327389e-02,
                  4.36111859e-01, -4.99600361e-16],
                [ 6.35352962e-02,  1.77912301e-02,  7.28960396e-01,
                  6.18811881e-04,  6.49752475e-03,  1.72970238e-01,
                  2.34903868e-01,  2.31602349e-02,  4.04212591e-02,
                  5.86139300e-01,  1.00000000e+00],
                [ 6.92335936e-02,  6.55837114e-03,  6.44122383e-01,
                  8.05152979e-04,  5.63607085e-03,  1.18636504e-01,
                  2.00595439e-01,  7.31260853e-03,  1.19405706e-02,
                  5.34640411e-01, -3.33066907e-16]])
```

9. Plot the Data

Plot :1

```
In [21]: # Plot between pairs Balance~Qual_miles
         sns.lmplot('Balance','Qual_miles',hue='cluster',data=data,fit_reg=False,size=4)
```

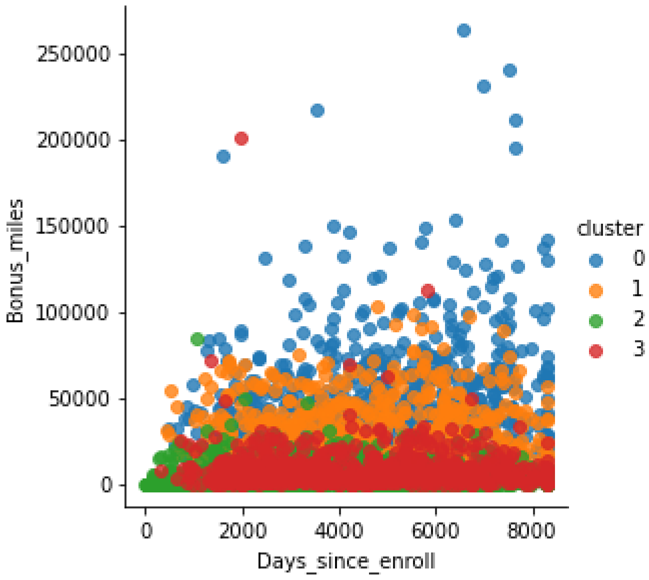
```
Out[21]: <seaborn.axisgrid.FacetGrid at 0x2449f0117c0>
```



plot:2

```
In [22]: # Plot between pairs Days_since_enroll~Bonus_miles
         sns.lmplot('Days_since_enroll','Bonus_miles',hue='cluster',data=data,fit_reg=Fals
```

```
Out[22]: <seaborn.axisgrid.FacetGrid at 0x244a6412c70>
```


Bonus_miles
Days_since_enroll
cluster
● 0
● 1
● 2
● 3