In [135]:

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

Load the Dataset

In [136]:

```
df = pd.read_csv('/kaggle/input/ccdata/CC GENERAL.csv')
```

In [137]:

```
df.head()
```

Out[137]:

	CUST_ID	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES
0	C10001	40.900749	0.818182	95.40	0.00
1	C10002	3202.467416	0.909091	0.00	0.00
2	C10003	2495.148862	1.000000	773.17	773.17
3	C10004	1666.670542	0.636364	1499.00	1499.00
4	C10005	817.714335	1.000000	16.00	16.00

In [138]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8950 entries, 0 to 8949
Data columns (total 18 columns):
 #   Column                            Non-Null Count  Dtype  
---  ------                            --------------  -----  
 0   CUST_ID                           8950 non-null   object 
 1   BALANCE                           8950 non-null   float64
 2   BALANCE_FREQUENCY                 8950 non-null   float64
 3   PURCHASES                         8950 non-null   float64
 4   ONEOFF_PURCHASES                  8950 non-null   float64
 5   INSTALLMENTS_PURCHASES            8950 non-null   float64
 6   CASH_ADVANCE                      8950 non-null   float64
 7   PURCHASES_FREQUENCY               8950 non-null   float64
 8   ONEOFF_PURCHASES_FREQUENCY        8950 non-null   float64
 9   PURCHASES_INSTALLMENTS_FREQUENCY  8950 non-null   float64
 10  CASH_ADVANCE_FREQUENCY            8950 non-null   float64
 11  CASH_ADVANCE_TRX                  8950 non-null   int64  
 12  PURCHASES_TRX                     8950 non-null   int64  
 13  CREDIT_LIMIT                      8949 non-null   float64
 14  PAYMENTS                          8950 non-null   float64
 15  MINIMUM_PAYMENTS                  8637 non-null   float64
 16  PRC_FULL_PAYMENT                  8950 non-null   float64
 17  TENURE                            8950 non-null   int64  
dtypes: float64(14), int64(3), object(1)
memory usage: 1.2+ MB
```

```
In [139]: df.drop('CUST_ID', axis=1, inplace=True)
```

```
In [140]: df.describe().T
```

```
Out[140]:
```

	count	mean	std	min
BALANCE	8950.0	1564.474828	2081.531879	0.000000
BALANCE_FREQUENCY	8950.0	0.877271	0.236904	0.000000
PURCHASES	8950.0	1003.204834	2136.634782	0.000000
ONEOFF_PURCHASES	8950.0	592.437371	1659.887917	0.000000
INSTALLMENTS_PURCHASES	8950.0	411.067645	904.338115	0.000000
CASH_ADVANCE	8950.0	978.871112	2097.163877	0.000000
PURCHASES_FREQUENCY	8950.0	0.490351	0.401371	0.000000
ONEOFF_PURCHASES_FREQUENCY	8950.0	0.202458	0.298336	0.000000
PURCHASES_INSTALLMENTS_FREQUENCY	8950.0	0.364437	0.397448	0.000000
CASH_ADVANCE_FREQUENCY	8950.0	0.135144	0.200121	0.000000
CASH_ADVANCE_TRX	8950.0	3.248827	6.824647	0.000000
PURCHASES_TRX	8950.0	14.709832	24.857649	0.000000
CREDIT_LIMIT	8949.0	4494.449450	3638.815725	50.000000
PAYMENTS	8950.0	1733.143852	2895.063757	0.000000
MINIMUM_PAYMENTS	8637.0	864.206542	2372.446607	0.019163
PRC_FULL_PAYMENT	8950.0	0.153715	0.292499	0.000000
TENURE	8950.0	11.517318	1.338331	6.000000

Data Pre-Processing

In [141]:

```
# Check for NULL Values

df.isna().sum()
```

Out[141]:

```
BALANCE                               0
BALANCE_FREQUENCY                     0
PURCHASES                             0
ONEOFF_PURCHASES                      0
INSTALLMENTS_PURCHASES                0
CASH_ADVANCE                          0
PURCHASES_FREQUENCY                   0
ONEOFF_PURCHASES_FREQUENCY            0
PURCHASES_INSTALLMENTS_FREQUENCY      0
CASH_ADVANCE_FREQUENCY                0
CASH_ADVANCE_TRX                      0
PURCHASES_TRX                         0
CREDIT_LIMIT                          1
PAYMENTS                              0
MINIMUM_PAYMENTS                    313
PRC_FULL_PAYMENT                      0
TENURE                                0
dtype: int64
```

In [142]:

```
# Handle NULL Values : Replace with Median Values

df['MINIMUM_PAYMENTS'].fillna(df['MINIMUM_PAYMENTS'].median(), inplace
=True)
df['CREDIT_LIMIT'].fillna(df['CREDIT_LIMIT'].median(), inplace= True)

df.isna().sum().sum()
```

Out[142]:

```
0
```

Build Model and Predict Output

In [143]:

```
from sklearn.cluster import KMeans
```

In [144]:

```
kmeans = KMeans(n_clusters=8,init = 'k-means++',random_state=0)
```

In [145]:

```
y_kmeans = kmeans.fit_predict(df)
```

```
/opt/conda/lib/python3.10/site-packages/sklearn/cluster/_kmeans.py:87
0: FutureWarning: The default value of `n_init` will change from 10 to
'auto' in 1.4. Set the value of `n_init` explicitly to suppress the wa
rning
  warnings.warn(
```

In [146]:

```
y_kmeans = y_kmeans.reshape(len(y_kmeans),1)

y_kmeans.shape
```

Out[146]:

```
(8950, 1)
```

In [147]:
```
b = np.concatenate((y_kmeans,df),axis = 1)
```

In [149]:
```
df.columns
```

Out[149]:
```
Index(['BALANCE', 'BALANCE_FREQUENCY', 'PURCHASES', 'ONEOFF_PURCHASE
S',
       'INSTALLMENTS_PURCHASES', 'CASH_ADVANCE', 'PURCHASES_FREQUENC
Y',
       'ONEOFF_PURCHASES_FREQUENCY', 'PURCHASES_INSTALLMENTS_FREQUENC
Y',
       'CASH_ADVANCE_FREQUENCY', 'CASH_ADVANCE_TRX', 'PURCHASES_TRX',
       'CREDIT_LIMIT', 'PAYMENTS', 'MINIMUM_PAYMENTS', 'PRC_FULL_PAYME
NT',
       'TENURE'],
      dtype='object')
```

In [150]:
```
df_final =pd.DataFrame(data = b ,columns =['Cluster_no','BALANCE', 'BA
LANCE_FREQUENCY', 'PURCHASES',
                                          'ONEOFF_PURCHASES', 'INSTAL
LMENTS_PURCHASES', 'CASH_ADVANCE',
                                          'PURCHASES_FREQUENCY','ONEO
FF_PURCHASES_FREQUENCY',
                                          'PURCHASES_INSTALLMENTS_FRE
QUENCY','CASH_ADVANCE_FREQUENCY',
                                          'CASH_ADVANCE_TRX', 'PURCHA
SES_TRX', 'CREDIT_LIMIT',
                                          'PAYMENTS', 'MINIMUM_PAYMEN
TS', 'PRC_FULL_PAYMENT','TENURE'])
```

In [151]:
```
df_final.head()
```

Out[151]:

	Cluster_no	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASE
0	1.0	40.900749	0.818182	95.40	0.0
1	3.0	3202.467416	0.909091	0.00	0.0
2	2.0	2495.148862	1.000000	773.17	773.1
3	2.0	1666.670542	0.636364	1499.00	1499.0
4	1.0	817.714335	1.000000	16.00	16.0

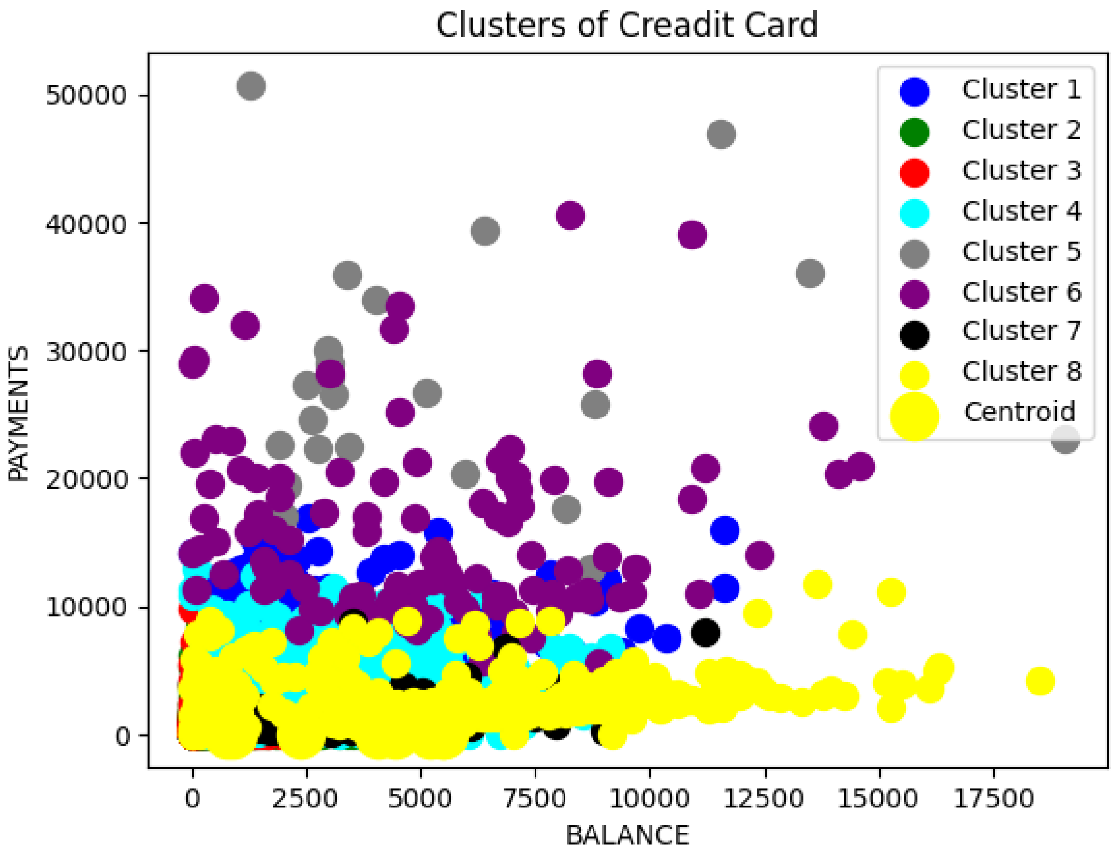
Visualize the Results

In [153]:

```
#visulaizing the clusters
plt.scatter(df_final['BALANCE'][df_final['Cluster_no'] == 0],
            df_final['PAYMENTS'][df_final['Cluster_no'] == 0], s = 10
0,
            c = 'blue', label = 'Cluster 1') #for first cluster
plt.scatter(df_final['BALANCE'][df_final['Cluster_no'] == 1],
            df_final['PAYMENTS'][df_final['Cluster_no'] == 1], s = 10
0,
            c = 'green', label = 'Cluster 2') #for second cluster
plt.scatter(df_final['BALANCE'][df_final['Cluster_no'] == 2],
            df_final['PAYMENTS'][df_final['Cluster_no'] == 2], s = 10
0,
            c = 'red', label = 'Cluster 3') #for third cluster
plt.scatter(df_final['BALANCE'][df_final['Cluster_no'] == 3],
            df_final['PAYMENTS'][df_final['Cluster_no'] == 3], s = 10
0,
            c = 'cyan', label = 'Cluster 4') #for fourth cluster
plt.scatter(df_final['BALANCE'][df_final['Cluster_no'] == 4],
            df_final['PAYMENTS'][df_final['Cluster_no'] == 4], s = 10
0,
            c = 'grey', label = 'Cluster 5') #for fifth cluster
plt.scatter(df_final['BALANCE'][df_final['Cluster_no'] == 5],
            df_final['PAYMENTS'][df_final['Cluster_no'] == 5], s = 10
0,
            c = 'purple', label = 'Cluster 6') #for sixth cluster
plt.scatter(df_final['BALANCE'][df_final['Cluster_no'] == 6],
            df_final['PAYMENTS'][df_final['Cluster_no'] == 6], s = 10
0,
            c = 'black', label = 'Cluster 7') #for seventh cluster
plt.scatter(df_final['BALANCE'][df_final['Cluster_no'] == 7],
            df_final['PAYMENTS'][df_final['Cluster_no'] == 7], s = 10
0,
            c = 'yellow', label = 'Cluster 8') #for seventh cluster

plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:,
1],
            s = 300, c = 'yellow', label = 'Centroid')

plt.title('Clusters of Creadit Card')
plt.xlabel('BALANCE')
plt.ylabel('PAYMENTS')
plt.legend()
plt.show()
```


Clusters of Creadit Card
PAYMENTS
50000
40000
30000
20000
10000
0
0
2500
5000
7500
10000
12500
15000
17500
BALANCE
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5
Cluster 6
Cluster 7
Cluster 8
Centroid

In [ ]: