In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
data=pd.read_excel(r'C:\Users\Debu bhaiya\Downloads\sales_data.xlsx')
```

In [3]:
```
data
```

Out[3]:

	CustomerID	TOTAL_ORDERS	REVENUE	AVERAGE_ORDER_VALUE	CARRIAGE_REVENUE	AVERAGESHIPPING	FIR
0	22	124	11986.54	96.67	529.59	4.27	
1	29	82	11025.96	134.46	97.92	1.19	
2	83	43	7259.69	168.83	171.69	3.99	
3	95	44	6992.27	158.92	92.82	2.11	
4	124	55	6263.44	113.88	179.04	3.26	
...	...	...	...	...	...	...	
4995	173946	1	117.49	117.49	4.99	4.99	
4996	173987	1	117.49	117.49	4.99	4.99	
4997	174004	1	117.49	117.49	4.99	4.99	
4998	174038	1	117.49	117.49	4.99	4.99	
4999	200783	2	94.14	47.07	9.94	4.97	

5000 rows × 40 columns

In [4]:
```
data.shape
```

Out[4]:
```
(5000, 40)
```

In [5]:
```
data.columns
```

Out[5]:
```
Index(['CustomerID', 'TOTAL_ORDERS', 'REVENUE', 'AVERAGE_ORDER_VALUE',
       'CARRIAGE_REVENUE', 'AVERAGESHIPPING', 'FIRST_ORDER_DATE',
       'LATEST_ORDER_DATE', 'AVGDAYSBETWEENORDERS', 'DAYSSINCELASTORDER',
       'MONDAY_ORDERS', 'TUESDAY_ORDERS', 'WEDNESDAY_ORDERS',
       'THURSDAY_ORDERS', 'FRIDAY_ORDERS', 'SATURDAY_ORDERS', 'SUNDAY_ORDERS',
       'MONDAY_REVENUE', 'TUESDAY_REVENUE', 'WEDNESDAY_REVENUE',
       'THURSDAY_REVENUE', 'FRIDAY_REVENUE', 'SATURDAY_REVENUE',
       'SUNDAY_REVENUE', 'WEEK1_DAY01_DAY07_ORDERS',
       'WEEK2_DAY08_DAY15_ORDERS', 'WEEK3_DAY16_DAY23_ORDERS',
       'WEEK4_DAY24_DAY31_ORDERS', 'WEEK1_DAY01_DAY07_REVENUE',
       'WEEK2_DAY08_DAY15_REVENUE', 'WEEK3_DAY16_DAY23_REVENUE',
       'WEEK4_DAY24_DAY31_REVENUE', 'TIME_0000_0600_ORDERS',
       'TIME_0601_1200_ORDERS', 'TIME_1200_1800_ORDERS',
       'TIME_1801_2359_ORDERS', 'TIME_0000_0600_REVENUE',
       'TIME_0601_1200_REVENUE', 'TIME_1200_1800_REVENUE',
       'TIME_1801_2359_REVENUE'],
      dtype='object')
```

In [6]:
```
data=data.drop(['AVERAGE_ORDER_VALUE','CARRIAGE_REVENUE','AVERAGESHIPPING','FIRST_ORDER_DA
       'THURSDAY_ORDERS', 'FRIDAY_ORDERS', 'SATURDAY_ORDERS', 'SUNDAY_ORDERS',
       'MONDAY_REVENUE', 'TUESDAY_REVENUE', 'WEDNESDAY_REVENUE',
       'THURSDAY_REVENUE', 'FRIDAY_REVENUE', 'SATURDAY_REVENUE',
       'SUNDAY_REVENUE', 'WEEK1_DAY01_DAY07_ORDERS',
       'WEEK2_DAY08_DAY15_ORDERS', 'WEEK3_DAY16_DAY23_ORDERS',
       'WEEK4_DAY24_DAY31_ORDERS', 'WEEK1_DAY01_DAY07_REVENUE',
       'WEEK2_DAY08_DAY15_REVENUE', 'WEEK3_DAY16_DAY23_REVENUE',
       'WEEK4_DAY24_DAY31_REVENUE', 'TIME_0000_0600_ORDERS',
       'TIME_0601_1200_ORDERS', 'TIME_1200_1800_ORDERS',
       'TIME_1801_2359_ORDERS', 'TIME_0000_0600_REVENUE',
       'TIME_0601_1200_REVENUE', 'TIME_1200_1800_REVENUE',
       'TIME_1801_2359_REVENUE'], axis=1)
```

In [7]:

```
data
```

Out[7]:

	CustomerID	TOTAL_ORDERS	REVENUE	DAYSSINCELASTORDER
0	22	124	11986.54	1
1	29	82	11025.96	1
2	83	43	7259.69	1
3	95	44	6992.27	1
4	124	55	6263.44	1
...	...	...	...	...
4995	173946	1	117.49	207
4996	173987	1	117.49	207
4997	174004	1	117.49	207
4998	174038	1	117.49	207
4999	200783	2	94.14	207

5000 rows × 4 columns

In [8]:

```
data.shape
```

Out[8]:

```
(5000, 4)
```

In [9]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 4 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   CustomerID          5000 non-null   int64  
 1   TOTAL_ORDERS        5000 non-null   int64  
 2   REVENUE             5000 non-null   float64
 3   DAYSSINCELASTORDER  5000 non-null   int64  
dtypes: float64(1), int64(3)
memory usage: 156.4 KB
```

In [10]:

```
data.describe()
```

Out[10]:

	CustomerID	TOTAL_ORDERS	REVENUE	DAYSSINCELASTORDER

	CustomerID	TOTAL_ORDERS	REVENUE	DAYSSINCELASTORDER
count	5000.000000	5000.00000	5000.000000	5000.000000
mean	40709.227800	12.87040	1681.523840	87.420000
std	49949.848017	12.67988	1998.618678	80.156513
min	1.000000	1.00000	38.500000	1.000000
25%	1687.500000	3.00000	315.097500	7.000000
50%	13765.000000	8.00000	966.725000	68.000000
75%	71891.500000	20.00000	2493.072500	171.250000
max	277160.000000	156.00000	34847.400000	207.000000

In [11]:
```
data.isnull().sum()
```

Out[11]:
```
CustomerID            0
TOTAL_ORDERS          0
REVENUE               0
DAYSSINCELASTORDER    0
dtype: int64
```

In [12]:
```
data.rename(columns = {'TOTAL_ORDERS':'FREQUENCY', 'REVENUE':'MONETARY',
                       'DAYSSINCELASTORDER':'RECENCY'}, inplace = True)
```

In [13]:
```
data
```

Out[13]:

	CustomerID	FREQUENCY	MONETARY	RECENCY
0	22	124	11986.54	1
1	29	82	11025.96	1
2	83	43	7259.69	1
3	95	44	6992.27	1
4	124	55	6263.44	1
...	...	...	...	...
4995	173946	1	117.49	207
4996	173987	1	117.49	207
4997	174004	1	117.49	207
4998	174038	1	117.49	207
4999	200783	2	94.14	207

5000 rows × 4 columns

In [14]:
```
data.corr()
```

Out[14]:

	CustomerID	FREQUENCY	MONETARY	RECENCY
CustomerID	1.000000	-0.608092	-0.556489	0.375343

	CustomerID	FREQUENCY	MONETARY	RECENCY
FREQUENCY	-0.608092	1.000000	0.771996	-0.256272
MONETARY	-0.556489	0.771996	1.000000	-0.197782
RECENCY	0.375343	-0.256272	-0.197782	1.000000

In [15]:
```
MAX_REVENUE=data['MONETARY'].max()
MIN_REVENUE=data['MONETARY'].min()
print(MAX_REVENUE)
print(MIN_REVENUE)
```

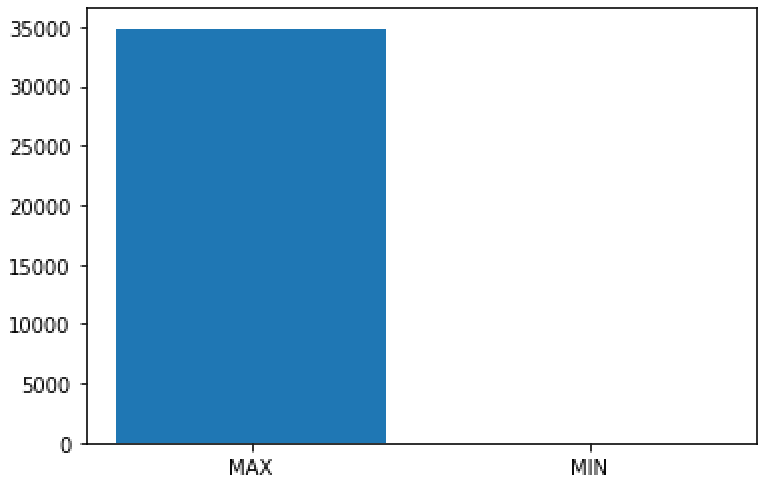
```
34847.4
38.5
```

In [35]:
```
MAX_FREQUENCY=data['FREQUENCY'].max()
MIN_FREQUENCY=data['FREQUENCY'].min()
print(MAX_FREQUENCY)
print(MIN_FREQUENCY)
```

```
156
1
```

In [16]:
```
import matplotlib.pyplot as plt
%matplotlib inline
noofvariables=[MAX_REVENUE,MIN_REVENUE]
datatypes=['MAX','MIN']
plt.bar(datatypes,noofvariables)
```

Out[16]:
```
<BarContainer object of 2 artists>
```



In [17]:
```
data['FREQUENCY'].value_counts()
```

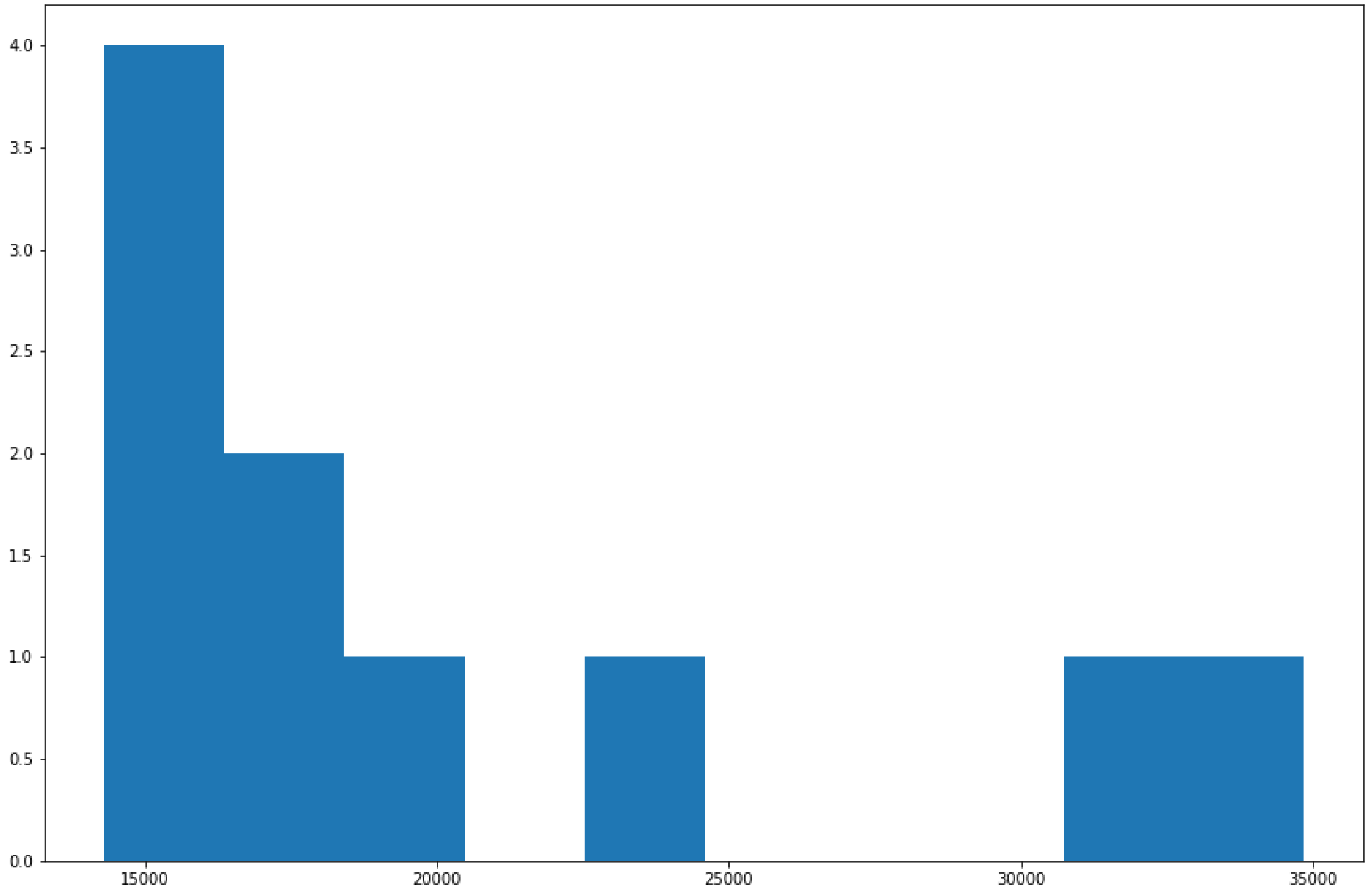
Out[17]:
```
2      560
3      509
1      366
4      360
5      238
      ...
88       1
84       1
76       1
56       1
```

```
111      1
Name: FREQUENCY, Length: 85, dtype: int64
```

In [18]:
```
top_10_revenues=data['MONETARY'].nlargest(n=10)
top_10_revenues
```

Out[18]:
```
2266    34847.40
2876    32486.98
2267    24178.97
1153    18554.49
3377    16884.99
1929    16693.78
3186    15999.94
2399    15840.36
3131    14526.72
1458    14309.92
Name: MONETARY, dtype: float64
```

In [19]:
```
plt.rcParams['figure.figsize']=(15,10)
plt.hist(top_10_revenues)
plt.show()
```



In [20]:
```
data['R_RANK'] = data['RECENCY'].rank(ascending=False)
data['F_RANK'] = data['FREQUENCY'].rank(ascending=True)
data['M_RANK'] = data['MONETARY'].rank(ascending=True)

# normalizing the rank of the customers
data['R_rank_norm'] = (data['R_RANK']/data['R_RANK'].max())*100
data['F_rank_norm'] = (data['F_RANK']/data['F_RANK'].max())*100
data['M_rank_norm'] = (data['F_RANK']/data['M_RANK'].max())*100

data.drop(columns=['R_RANK', 'F_RANK', 'M_RANK'], inplace=True)
data.head(10)
```

Out[20]:

	CustomerID	FREQUENCY	MONETARY	RECENCY	R_rank_norm	F_rank_norm	M_rank_norm
0	22	124	11986.54	1	100.0	99.96	99.96
1	29	82	11025.96	1	100.0	99.78	99.78
2	83	43	7259.69	1	100.0	97.28	97.28
3	95	44	6992.27	1	100.0	97.51	97.51
4	124	55	6263.44	1	100.0	99.00	99.00
5	153	49	5841.24	1	100.0	98.39	98.39
6	187	43	5470.27	1	100.0	97.28	97.28
7	219	54	5200.53	1	100.0	98.89	98.89
8	258	19	4967.06	1	100.0	73.32	73.32
9	308	21	4726.38	1	100.0	78.56	78.56

In [21]:

```
data['RFM_Score'] = 0.15*data['R_rank_norm']+0.28 * \
    data['F_rank_norm']+0.57*data['M_rank_norm']
data['RFM_Score'] *= 0.05
data = data.round(2)
RFM_SCORE=data[['CustomerID', 'RFM_Score']]
```

In [22]:

```
RFM_SCORE
```

Out[22]:

	CustomerID	RFM_Score
0	22	5.00
1	29	4.99
2	83	4.88
3	95	4.89
4	124	4.96
...	...	...
4995	173946	0.24
4996	173987	0.24
4997	174004	0.24
4998	174038	0.24
4999	200783	0.63

5000 rows × 2 columns

In [23]:

```
RFM_SCORE["Customer_segment"] = np.where(RFM_SCORE['RFM_Score'] >
                                      4.5, "CHAMPIONS",
                                      (np.where(
                                        RFM_SCORE['RFM_Score'] > 4,
                                        "High value Customer",
                                        (np.where(
    RFM_SCORE['RFM_Score'] > 3,
                             "Medium Value Customer",
                             np.where(RFM_SCORE['RFM_Score'] > 1.6,
                            'Low Value Customers', 'Lost Customers'))))))
RFM_SCORE[['CustomerID', 'RFM_Score', 'Customer_segment']].head(20)
```

```
C:\Users\DEBUBH~1\AppData\Local\Temp/ipykernel_12184/190580036.py:1: SettingWithCopyWarnin
g:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_gu
ide/indexing.html#returning-a-view-versus-a-copy
  RFM_SCORE["Customer_segment"] = np.where(RFM_SCORE['RFM_Score'] >
```

Out[23]:

	CustomerID	RFM_Score	Customer_segment
0	22	5.00	CHAMPIONS
1	29	4.99	CHAMPIONS
2	83	4.88	CHAMPIONS
3	95	4.89	CHAMPIONS
4	124	4.96	CHAMPIONS
5	153	4.93	CHAMPIONS
6	187	4.88	CHAMPIONS
7	219	4.95	CHAMPIONS
8	258	3.87	Medium Value Customer
9	308	4.09	High value Customer
10	491	4.82	CHAMPIONS
11	492	3.98	Medium Value Customer
12	572	4.09	High value Customer
13	595	4.76	CHAMPIONS
14	613	4.69	CHAMPIONS
15	669	4.57	CHAMPIONS
16	671	4.69	CHAMPIONS
17	740	3.09	Medium Value Customer
18	750	3.98	Medium Value Customer
19	785	3.40	Medium Value Customer

In [24]:

```
RFM_SCORE
```

Out[24]:

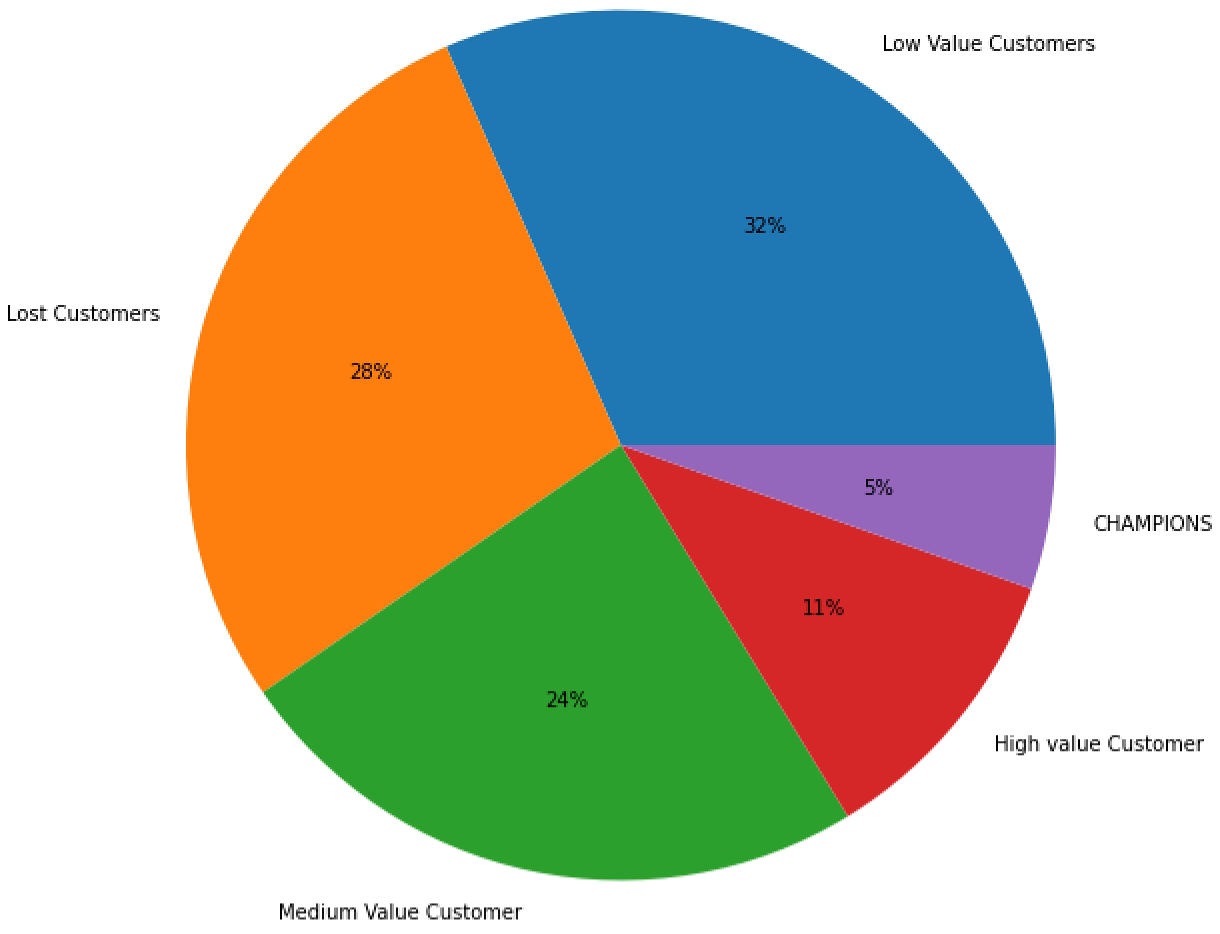
	CustomerID	RFM_Score	Customer_segment
0	22	5.00	CHAMPIONS
1	29	4.99	CHAMPIONS
2	83	4.88	CHAMPIONS
3	95	4.89	CHAMPIONS

	CustomerID	RFM_Score	Customer_segment
4	124	4.96	CHAMPIONS
...	...	...	...
4995	173946	0.24	Lost Customers
4996	173987	0.24	Lost Customers
4997	174004	0.24	Lost Customers
4998	174038	0.24	Lost Customers
4999	200783	0.63	Lost Customers

5000 rows × 3 columns

In [25]:

```
plt.pie(RFM_SCORE.Customer_segment.value_counts(),
        labels=RFM_SCORE.Customer_segment.value_counts().index,
        autopct='%.0f%%')
plt.show()
```



In [26]:

```
Segments=RFM_SCORE.Customer_segment.value_counts()
Segments
```

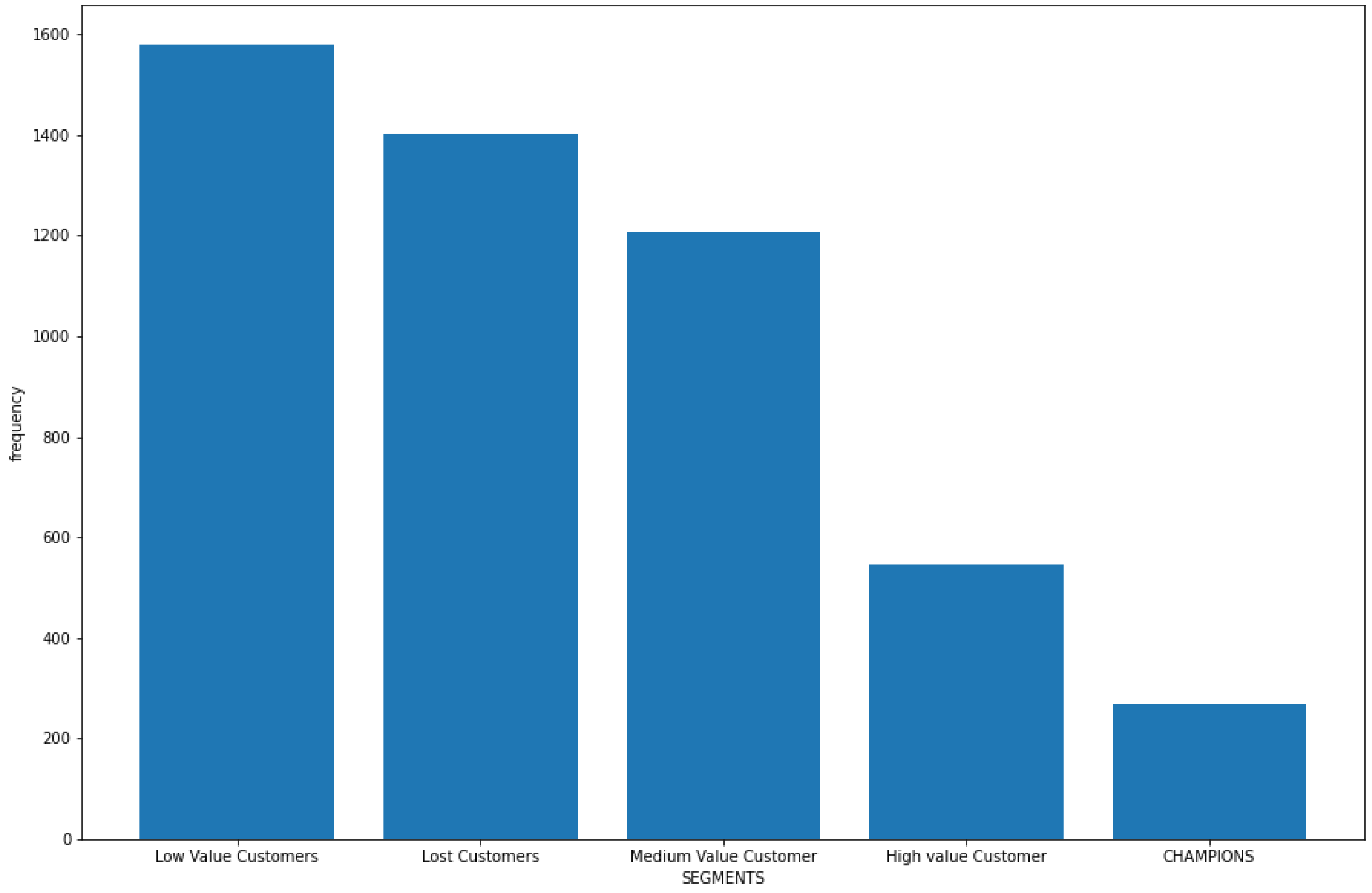
Out[26]:

```
Low Value Customers      1579
Lost Customers           1402
```

```
Medium Value Customer    1206
High value Customer       545
CHAMPIONS                 268
Name: Customer_segment, dtype: int64
```

In [27]:
```
plt.rcParams['figure.figsize']=(15,10)
noofvariables=[1579,1402,1206,545,268]
Segments=['Low Value Customers','Lost Customers','Medium Value Customer','High value Custo
plt.xlabel('SEGMENTS')
plt.ylabel('frequency')
plt.bar(Segments,noofvariables)
```

Out[27]:
```
<BarContainer object of 5 artists>
```



In [32]:
```
CHAMPIONS = RFM_SCORE.loc[RFM_SCORE['Customer_segment'] == 'CHAMPIONS']
```

In [33]:
```
CHAMPIONS
```

Out[33]:

	CustomerID	RFM_Score	Customer_segment
0	22.0	5.00	CHAMPIONS
1	29.0	4.99	CHAMPIONS
2	83.0	4.88	CHAMPIONS
3	95.0	4.89	CHAMPIONS
4	124.0	4.96	CHAMPIONS
...	...	...	...
3036	321.0	4.52	CHAMPIONS

	CustomerID	RFM_Score	Customer_segment
3096	315.0	4.51	CHAMPIONS
3131	9.0	4.55	CHAMPIONS
3186	7.0	4.54	CHAMPIONS
3219	25.0	4.52	CHAMPIONS

268 rows × 3 columns

```
In [ ]:
```