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

```
df = pd.read_csv("/content/credit_card.csv")
```

```
df.head(5)
```

	Client_Num	Card_Category	Annual_Fees	Activation_30_Days	Customer_Acq_Cost	Week_Start_Date	Week_Num	Qtr	current_year	C
0	708082083	Blue	200	0	87	01-01-2023	Week-1	Q1	2023	
1	708083283	Blue	445	1	108	01-01-2023	Week-1	Q1	2023	
2	708084558	Blue	140	0	106	01-01-2023	Week-1	Q1	2023	
3	708085458	Blue	250	1	150	01-01-2023	Week-1	Q1	2023	
4	708086958	Blue	320	1	106	01-01-2023	Week-1	Q1	2023	

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10108 entries, 0 to 10107
Data columns (total 18 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Client_Num             10108 non-null  int64
 1   Card_Category          10108 non-null  object
 2   Annual_Fees            10108 non-null  int64
 3   Activation_30_Days     10108 non-null  int64
 4   Customer_Acq_Cost      10108 non-null  int64
 5   Week_Start_Date        10108 non-null  object
 6   Week_Num               10108 non-null  object
 7   Qtr                    10108 non-null  object
 8   current_year           10108 non-null  int64
 9   Credit_Limit           10108 non-null  float64
 10  Total_Revolving_Bal    10108 non-null  int64
 11  Total_Trans_Amt        10108 non-null  int64
 12  Total_Trans_Vol        10108 non-null  int64
 13  Avg_Utilization_Ratio  10108 non-null  float64
 14  Use Chip               10108 non-null  object
 15  Exp Type               10108 non-null  object
 16  Interest_Earned        10108 non-null  float64
 17  Delinquent_Acc         10108 non-null  int64
dtypes: float64(3), int64(9), object(6)
memory usage: 1.4+ MB
```

```
df.head(5)
```

	Client_Num	Card_Category	Annual_Fees	Activation_30_Days	Customer_Acq_Cost	Week_Start_Date	Week_Num	Qtr	current_year	C
0	708082083	Blue	200	0	87	01-01-2023	Week-1	Q1	2023	
1	708083283	Blue	445	1	108	01-01-2023	Week-1	Q1	2023	
2	708084558	Blue	140	0	106	01-01-2023	Week-1	Q1	2023	
3	708085458	Blue	250	1	150	01-01-2023	Week-1	Q1	2023	
4	708086958	Blue	320	1	106	01-01-2023	Week-1	Q1	2023	

```
df.columns
```

```
Index(['Client_Num', 'Card_Category', 'Annual_Fees', 'Activation_30_Days',
       'Customer_Acq_Cost', 'Week_Start_Date', 'Week_Num', 'Qtr',
       'current_year', 'Credit_Limit', 'Total_Revolving_Bal',
       'Total_Trans_Amt', 'Total_Trans_Vol', 'Avg_Utilization_Ratio',
       'Use Chip', 'Exp Type', 'Interest_Earned', 'Delinquent_Acc'],
      dtype='object')
```

```
df1 = df[['Annual_Fees', 'Customer_Acq_Cost', 'Total_Trans_Amt']]
```

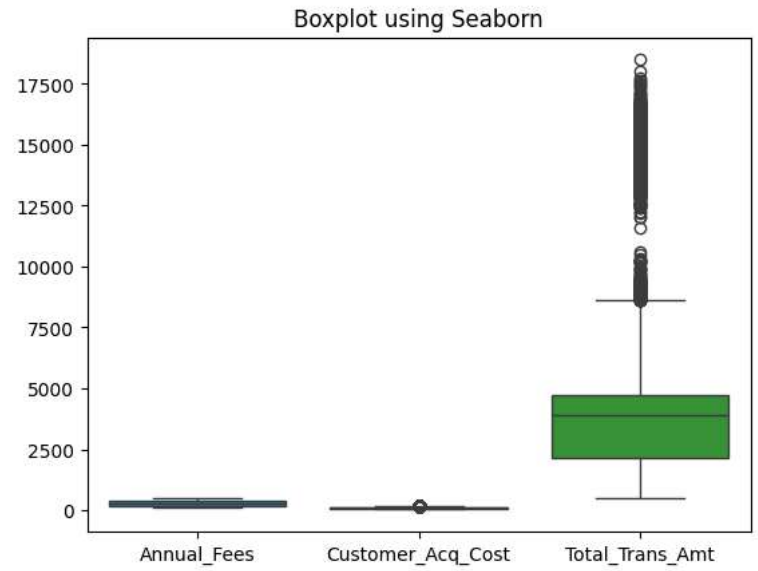
```
df1.head(6)
```

	Annual_Fees	Customer_Acq_Cost	Total_Trans_Amt
0	200	87	15149
1	445	108	992
2	140	106	1447
3	250	150	3940
4	320	106	4369
5	100	94	1448

```
df1.shape
```

```
(10108, 3)
```

```
sns.boxplot(data=df[['Annual_Fees', 'Customer_Acq_Cost', 'Total_Trans_Amt']])
plt.title("Boxplot using Seaborn")
plt.show()
```




```
#IQR

Q1 = df1.quantile(0.25)
Q3 = df1.quantile(0.75)
```

```
print(Q1)
```

```
Annual_Fees           195.00
Customer_Acq_Cost      79.00
Total_Trans_Amt      2155.75
Name: 0.25, dtype: float64
```

```
print(Q3)
```

```
Annual_Fees           395.0
Customer_Acq_Cost     112.0
Total_Trans_Amt      4741.0
Name: 0.75, dtype: float64
```

```
IQR = Q3-Q1
```

```
IQR
```

	0
Annual_Fees	200.00
Customer_Acq_Cost	33.00
Total_Trans_Amt	2585.25

dtype: float64

```
#IQR से Outlier Detection
```

```
Lower_Bound=Q1 - 1.5 * IQR
```

```
Upper_Bound = Q3 + 1.5 * IQR
```

```
Lower_Bound
```

	0
Annual_Fees	-105.000
Customer_Acq_Cost	29.500
Total_Trans_Amt	-1722.125

dtype: float64

```
Upper_Bound
```

	0
Annual_Fees	695.000
Customer_Acq_Cost	161.500
Total_Trans_Amt	8618.875

dtype: float64

```
df_cleaned = df1[
    (df1['Annual_Fees'] >= Lower_Bound['Annual_Fees']) & (df1['Annual_Fees'] <= Upper_Bound['Annual_Fees']) &
    (df1['Customer_Acq_Cost'] >= Lower_Bound['Customer_Acq_Cost']) & (df1['Customer_Acq_Cost'] <= Upper_Bound['Customer_Acq_Cos
    (df1['Total_Trans_Amt'] >= Lower_Bound['Total_Trans_Amt']) & (df1['Total_Trans_Amt'] <= Upper_Bound['Total_Trans_Amt'])
]

print(df_cleaned)
```

```
       Annual_Fees  Customer_Acq_Cost  Total_Trans_Amt
1              445                108              992
2              140                106             1447
3              250                150             3940
4              320                106             4369
5              100                 94             1448
...            ...                ...              ...
10103          340                106             3906
10104          395                104             4674
10105          125                107             4432
10106          410                 96             2089
10107          100                 43             3785

[9075 rows x 3 columns]
```

```
print(f"shape of df1 is {df1.shape}")
```

```
shape of df1 is (10108, 3)
```

```
print(f"shape of df_cleaned is {df_cleaned.shape}")
```

```
shape of df_cleaned is (9075, 3)
```

```
diff = 10108-9075
```

```
diff
```

```
1033
```

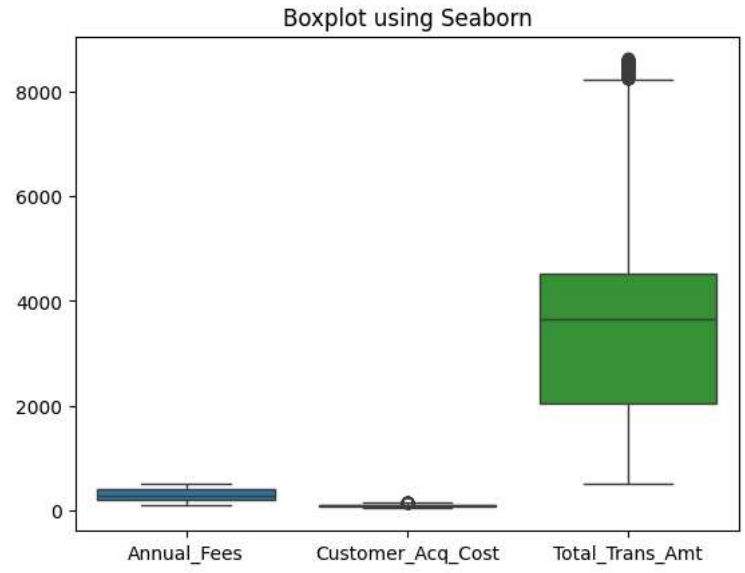
```
def remove_outliers_iqr(df, columns):
    df_no_outliers = df.copy()

    for col in columns:
        Q1 = df_no_outliers[col].quantile(0.25)
        Q3 = df_no_outliers[col].quantile(0.75)
        IQR = Q3 - Q1

        lower = Q1 - 1.5 * IQR
        upper = Q3 + 1.5 * IQR

        df_no_outliers = df_no_outliers[
            (df_no_outliers[col] >= lower) & (df_no_outliers[col] <= upper)
        ]

    return df_no_outliers
```

```
df_clean = remove_outliers_iqr(
    df1,
    ['Annual_Fees', 'Customer_Acq_Cost', 'Total_Trans_Amt']
)
```

```
df_clean.shape
```

```
(9077, 3)
```

```
sns.boxplot(data=df_clean[['Annual_Fees', 'Customer_Acq_Cost', 'Total_Trans_Amt']])
plt.title("Boxplot using Seaborn")
plt.show()
```




```
df.head()
```

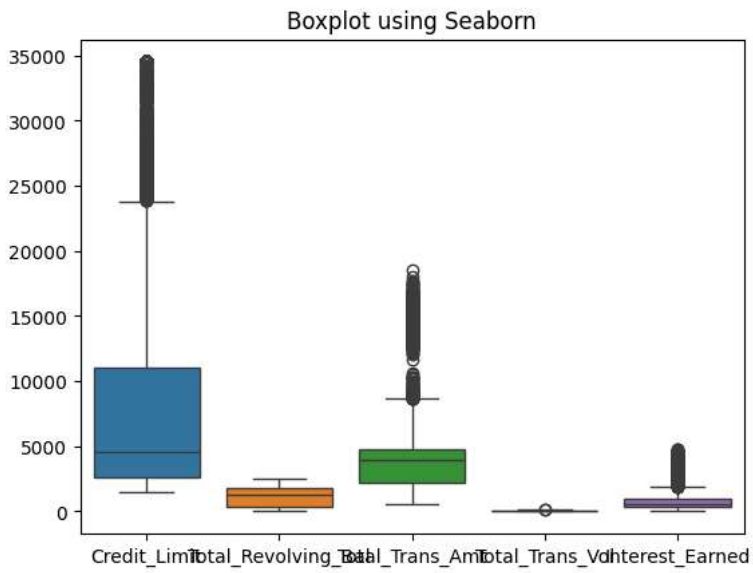
	Client_Num	Card_Category	Annual_Fees	Activation_30_Days	Customer_Acq_Cost	Week_Start_Date	Week_Num	Qtr	current_year	C
0	708082083	Blue	200	0	87	01-01-2023	Week-1	Q1	2023	
1	708083283	Blue	445	1	108	01-01-2023	Week-1	Q1	2023	
2	708084558	Blue	140	0	106	01-01-2023	Week-1	Q1	2023	
3	708085458	Blue	250	1	150	01-01-2023	Week-1	Q1	2023	
4	708086958	Blue	320	1	106	01-01-2023	Week-1	Q1	2023	

```
Df2 = df[["Credit_Limit","Total_Revolving_Bal","Total_Trans_Amt","Total_Trans_Vol","Interest_Earned"]]
```

```
Df2.head()
```

	Credit_Limit	Total_Revolving_Bal	Total_Trans_Amt	Total_Trans_Vol	Interest_Earned
0	3544.0	1661	15149	111	4393.21
1	3421.0	2517	992	21	69.44
2	8258.0	1771	1447	23	202.58
3	1438.3	0	3940	82	236.40
4	3128.0	749	4369	59	1004.87

```
sns.boxplot(data=Df2[["Credit_Limit","Total_Revolving_Bal","Total_Trans_Amt","Total_Trans_Vol","Interest_Earned"]])
plt.title("Boxplot using Seaborn")
plt.show()
```




```
mean = Df2.mean()
std = Df2.std()
```

```
print(mean)
print("________")
print(std)
```

```
Credit_Limit           8635.642808
Total_Revolving_Bal    1162.792145
Total_Trans_Amt        4404.631282
Total_Trans_Vol          64.864563
Interest_Earned         775.957878
dtype: float64
________
Credit_Limit           9093.136113
Total_Revolving_Bal     815.160709
Total_Trans_Amt        3397.910673
Total_Trans_Vol          23.475110
Interest_Earned         723.952320
```

```
dtype: float64
```

```
Df3 = (Df2 - mean) / std
```

```
Df3.head()
```

	Credit_Limit	Total_Revolving_Bal	Total_Trans_Amt	Total_Trans_Vol	Interest_Earned
0	-0.559944	0.611177	3.162052	1.965292	4.996534
1	-0.573470	1.661277	-1.004332	-1.868556	-0.975918
2	-0.041531	0.746120	-0.870426	-1.783360	-0.792011
3	-0.791514	-1.426458	-0.136740	0.729941	-0.745295
4	-0.605692	-0.507620	-0.010486	-0.249820	0.316198

```
# before describe funcation

Df2.describe()
```

	Credit_Limit	Total_Revolving_Bal	Total_Trans_Amt	Total_Trans_Vol	Interest_Earned
count	10108.000000	10108.000000	10108.000000	10108.000000	10108.000000
mean	8635.642808	1162.792145	4404.631282	64.864563	775.957878
std	9093.136113	815.160709	3397.910673	23.475110	723.952320
min	1438.300000	0.000000	510.000000	10.000000	42.140000
25%	2552.750000	355.500000	2155.750000	45.000000	326.150000
50%	4549.000000	1276.500000	3899.500000	67.000000	559.985000
75%	11070.250000	1784.000000	4741.000000	81.000000	962.685000
max	34516.000000	2517.000000	18484.000000	139.000000	4785.000000

```
# after detect outlier then using describe funcation

Df3.describe()
```

	Credit_Limit	Total_Revolving_Bal	Total_Trans_Amt	Total_Trans_Vol	Interest_Earned
count	1.010800e+04	1.010800e+04	1.010800e+04	1.010800e+04	1.010800e+04
mean	-8.997771e-17	7.556722e-17	1.096603e-16	2.732721e-17	1.068485e-16
std	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00
min	-7.915138e-01	-1.426458e+00	-1.146184e+00	-2.337138e+00	-1.013627e+00
25%	-6.689543e-01	-9.903472e-01	-6.618424e-01	-8.461968e-01	-6.213225e-01
50%	-4.494206e-01	1.394913e-01	-1.486594e-01	9.096602e-02	-2.983247e-01
75%	2.677412e-01	7.620680e-01	9.899281e-02	6.873423e-01	2.579274e-01
max	2.846142e+00	1.661277e+00	4.143537e+00	3.158044e+00	5.537716e+00

```
threshold = 3
Df2_cleaned = Df2[(Df3 < threshold).all(axis=1)]

print(Df2_cleaned)
```

```
       Credit_Limit  Total_Revolving_Bal  Total_Trans_Amt  Total_Trans_Vol  \
1            3421.0                 2517              992               21
2            8258.0                 1771             1447               23
3            1438.3                    0             3940               82
4            3128.0                  749             4369               59
5           33304.0                 1833             1448               29
...             ...                  ...              ...              ...
10103       34516.0                 1329             3906               77
10104       13426.0                    0             4674               66
10105        2346.0                 1373             4432               72
10106        6648.0                 2242             2089               60
10107        2062.0                 1302             3785               63
```

```
       Interest_Earned
1                69.44
2               202.58
3               236.40
4              1004.87
5               275.12
...                ...
10103           546.84
10104           607.62
10105           797.76
10106           146.23
10107          1059.80

[9591 rows x 5 columns]
```

```
# before shape when i do not use outlier technique then
Df2.shape
```

```
(10108, 5)
```

```
# after shape when i use outlier technique and remove outlier thenDf3
```

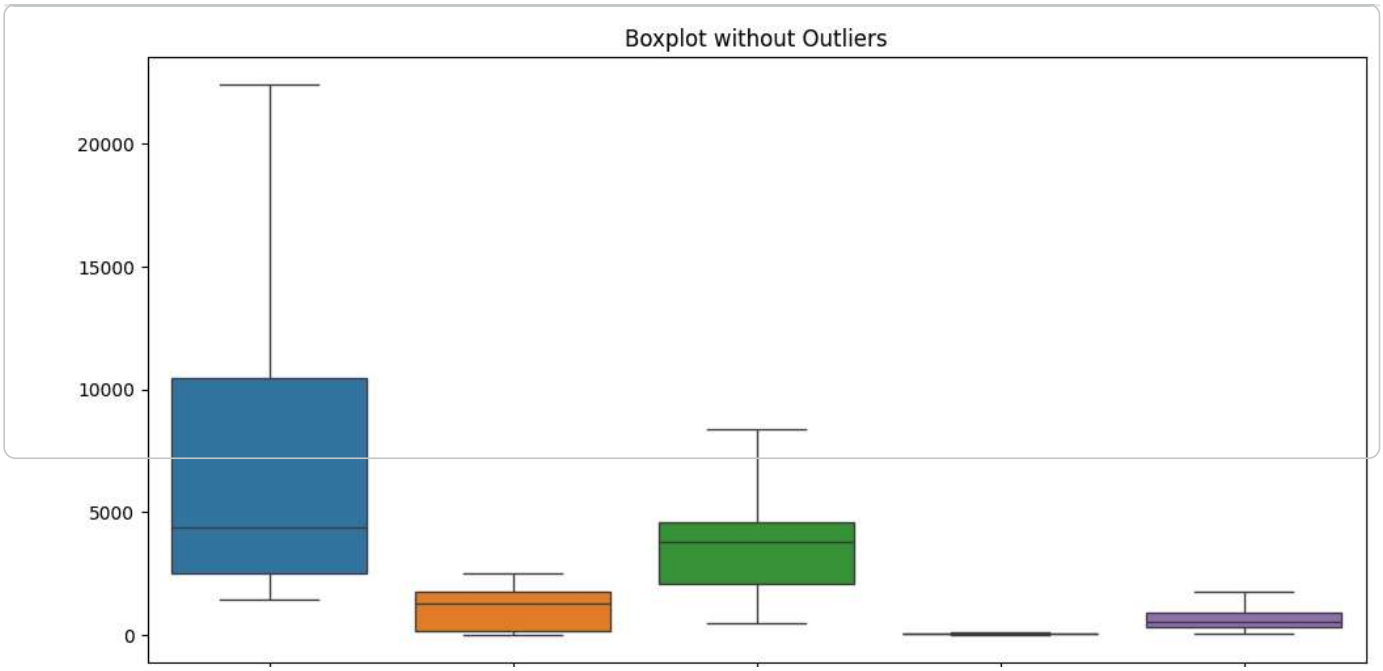
```
Df2_cleaned.shape
```

```
(9591, 5)
```

```
# This shows all rows in your data that have extreme values in at least one column.
outliers = Df2[(Df3 > threshold).any(axis=1)]
print(outliers)
```

```
       Credit_Limit  Total_Revolving_Bal  Total_Trans_Amt  Total_Trans_Vol  \
0            3544.0                 1661            15149              111
9           11898.0                 2517            15798              128
17          11463.0                    0            14511              105
37          34516.0                  638            13085              139
75          29937.0                    0            14863              109
...             ...                  ...              ...              ...
10037       19033.0                 1555            16033              116
10044       16453.0                 1660            14762              108
10079        1438.3                  726             5208               93
10085        9431.0                 1785            14261               99
10101        4107.0                 2517            16027              112

       Interest_Earned
0              4393.21
9              3791.52
17             3047.31
37             2355.30
75             1932.19
...                ...
10037          3687.59
10044           885.72
10079          3302.00
10085          3422.64
10101          4487.56

[517 rows x 5 columns]
```

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(12,6))
sns.boxplot(data=Df2_cleaned[["Credit_Limit","Total_Revolving_Bal","Total_Trans_Amt","Total_Trans_Vol","Interest_Earned"]],
            showfliers=False)
plt.title("Boxplot without Outliers")
plt.show()
```


Boxplot without Outliers
20000
15000
10000
5000
0