In [1]:

```
import pandas as pd
```

In [2]:

```
loan=pd.read_excel('loan_sanction.xlsx')
```

In [3]:

```
loan.head()
```

Out[3]:

	Unnamed: 0	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncor
0	0	LP001015	Male	Yes	0	Graduate	No	57
1	1	LP001022	Male	Yes	1	Graduate	No	30
2	2	LP001031	Male	Yes	2	Graduate	No	50
3	3	LP001035	Male	Yes	2	Graduate	No	23
4	4	LP001051	Male	No	0	Not Graduate	No	32

In [4]:

```
loan.tail()
```

Out[4]:

	Unnamed: 0	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantInc
362	362	LP002971	Male	Yes	3+	Not Graduate	Yes	
363	363	LP002975	Male	Yes	0	Graduate	No	
364	364	LP002980	Male	No	0	Graduate	No	
365	365	LP002986	Male	Yes	0	Graduate	No	
366	366	LP002989	Male	No	0	Graduate	Yes	

In [5]:

```
loan.columns
```

Out[5]:

```
Index(['Unnamed: 0', 'Loan_ID', 'Gender', 'Married', 'Dependents', 'Ed
ucation',
       'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanA
mount',
       'Loan_Amount_Term', 'Credit_History', 'Property_Area'],
      dtype='object')
```

In [6]:

```
loan.shape
```

Out[6]:

```
(367, 13)
```

In [7]:

```
loan=loan.drop(columns='Unnamed: 0')#removing the unnammed column as it was not neec
loan
```

Out[7]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coap
0	LP001015	Male	Yes	0	Graduate	No	5720	
1	LP001022	Male	Yes	1	Graduate	No	3076	
2	LP001031	Male	Yes	2	Graduate	No	5000	
3	LP001035	Male	Yes	2	Graduate	No	2340	
4	LP001051	Male	No	0	Not Graduate	No	3276	
...	...	...	...	...	...	...	...	
362	LP002971	Male	Yes	3+	Not Graduate	Yes	4009	
363	LP002975	Male	Yes	0	Graduate	No	4158	
364	LP002980	Male	No	0	Graduate	No	3250	
365	LP002986	Male	Yes	0	Graduate	No	5000	
366	LP002989	Male	No	0	Graduate	Yes	9200	

367 rows × 12 columns

In [8]:

```
#checking the duplicates
duplicates = loan[loan.duplicated]
duplicates #seems no duplicate rows
```

Out[8]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coapplica

In [9]:

```
#checking the columns having null values
check_nulls= [col for col in loan.columns if loan[col].isnull().sum()>0]
check_nulls
```

Out[9]:

```
['Dependents']
```

In [10]:

```
loan = loan.dropna() #dropping the rows that have null values
loan
```

Out[10]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coap
0	LP001015	Male	Yes	0	Graduate	No	5720	
1	LP001022	Male	Yes	1	Graduate	No	3076	
2	LP001031	Male	Yes	2	Graduate	No	5000	
3	LP001035	Male	Yes	2	Graduate	No	2340	
4	LP001051	Male	No	0	Not Graduate	No	3276	
...	...	...	...	...	...	...	...	
362	LP002971	Male	Yes	3+	Not Graduate	Yes	4009	
363	LP002975	Male	Yes	0	Graduate	No	4158	
364	LP002980	Male	No	0	Graduate	No	3250	
365	LP002986	Male	Yes	0	Graduate	No	5000	
366	LP002989	Male	No	0	Graduate	Yes	9200	

357 rows × 12 columns

In [11]:

```
#replacing null values with the mode in the gender column
mode_values_gender = loan.mode().iloc[0]
loan = loan.fillna({'Gender' : mode_values_gender['Gender']})
loan
```

Out[11]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coap
0	LP001015	Male	Yes	0	Graduate	No	5720	
1	LP001022	Male	Yes	1	Graduate	No	3076	
2	LP001031	Male	Yes	2	Graduate	No	5000	
3	LP001035	Male	Yes	2	Graduate	No	2340	
4	LP001051	Male	No	0	Not Graduate	No	3276	
...	...	...	...	...	...	...	...	
362	LP002971	Male	Yes	3+	Not Graduate	Yes	4009	
363	LP002975	Male	Yes	0	Graduate	No	4158	
364	LP002980	Male	No	0	Graduate	No	3250	
365	LP002986	Male	Yes	0	Graduate	No	5000	
366	LP002989	Male	No	0	Graduate	Yes	9200	

357 rows × 12 columns

In [12]:

```
females = loan[loan['Gender']=='Female']
females
```

Out[12]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coap
6	LP001055	Female	No	1	Not Graduate	No	2226	
14	LP001096	Female	No	0	Graduate	No	4666	
21	LP001124	Female	No	3+	Not Graduate	No	2083	
23	LP001135	Female	No	0	Not Graduate	No	3765	
30	LP001177	Female	No	0	Not Graduate	No	2478	
...	...	...	...	...	...	...	...	
332	LP002826	Female	Yes	1	Not Graduate	No	3621	
333	LP002843	Female	Yes	0	Graduate	No	4709	
336	LP002853	Female	No	0	Not Graduate	No	3015	
339	LP002858	Female	No	0	Graduate	No	4333	
360	LP002965	Female	Yes	0	Graduate	No	8550	

68 rows × 12 columns

In [13]:

```
loan.mode().iloc[0]
```

Out[13]:

```
Loan_ID               LP001015
Gender                    Male
Married                    Yes
Dependents                   0
Education             Graduate
Self_Employed               No
ApplicantIncome         3500.0
CoapplicantIncome          0.0
LoanAmount               125.0
Loan_Amount_Term         360.0
Credit_History             1.0
Property_Area            Urban
Name: 0, dtype: object
```

In [14]:

```
#replacing null values with the mode in the gender column
loan = loan.fillna({'Self_Employed' : mode_values_gender['Self_Employed']})
loan
```

Out[14]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coap
0	LP001015	Male	Yes	0	Graduate	No	5720	
1	LP001022	Male	Yes	1	Graduate	No	3076	
2	LP001031	Male	Yes	2	Graduate	No	5000	
3	LP001035	Male	Yes	2	Graduate	No	2340	
4	LP001051	Male	No	0	Not Graduate	No	3276	
...	...	...	...	...	...	...	...	
362	LP002971	Male	Yes	3+	Not Graduate	Yes	4009	
363	LP002975	Male	Yes	0	Graduate	No	4158	
364	LP002980	Male	No	0	Graduate	No	3250	
365	LP002986	Male	Yes	0	Graduate	No	5000	
366	LP002989	Male	No	0	Graduate	Yes	9200	

357 rows × 12 columns

In [15]:

```
#replacing null values with the mode in the gender column
median_loan_amnt = loan['LoanAmount'].median()
median_loan_amnt
loan = loan.fillna({'LoanAmount':median_loan_amnt})
loan
#loan = loan.fillna({''})
```

Out[15]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coap
0	LP001015	Male	Yes	0	Graduate	No	5720	
1	LP001022	Male	Yes	1	Graduate	No	3076	
2	LP001031	Male	Yes	2	Graduate	No	5000	
3	LP001035	Male	Yes	2	Graduate	No	2340	
4	LP001051	Male	No	0	Not Graduate	No	3276	
...	...	...	...	...	...	...	...	
362	LP002971	Male	Yes	3+	Not Graduate	Yes	4009	
363	LP002975	Male	Yes	0	Graduate	No	4158	
364	LP002980	Male	No	0	Graduate	No	3250	
365	LP002986	Male	Yes	0	Graduate	No	5000	
366	LP002989	Male	No	0	Graduate	Yes	9200	

357 rows × 12 columns

In [16]:

```
#top 5 rows
loan.head()
```

Out[16]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coappli
0	LP001015	Male	Yes	0	Graduate	No	5720	
1	LP001022	Male	Yes	1	Graduate	No	3076	
2	LP001031	Male	Yes	2	Graduate	No	5000	
3	LP001035	Male	Yes	2	Graduate	No	2340	
4	LP001051	Male	No	0	Not Graduate	No	3276	

In [17]:

```
#exporting dataframe
loan.to_excel("new loan min exel.xlsx", index=False)
```

In [18]:

```
#number of rows and columns
loan.shape
```

Out[18]:

```
(357, 12)
```

In [19]:

```
#printi all the columns
loan.columns
```

Out[19]:

```
Index(['Loan_ID', 'Gender', 'Married', 'Dependents', 'Education',
       'Self_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanA
mount',
       'Loan_Amount_Term', 'Credit_History', 'Property_Area'],
      dtype='object')
```

In [20]:

```
#descriptive statistics
des_stat = loan.describe()
des_stat
```

Out[20]:

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	357.000000	357.000000	357.000000	357.000000	357.0
mean	4793.602241	1587.098039	136.501401	343.282913	1.0
std	4936.879103	2357.321156	61.503523	64.015457	0.0
min	0.000000	0.000000	28.000000	6.000000	1.0
25%	2858.000000	0.000000	102.000000	360.000000	1.0
50%	3786.000000	1025.000000	125.000000	360.000000	1.0
75%	5062.000000	2458.000000	158.000000	360.000000	1.0
max	72529.000000	24000.000000	550.000000	480.000000	1.0

In [33]:

```
#Display numeric data type columns in a DF
numeric_cols =[col for col in loan.columns if loan[col].dtype == 'int64']
numeric_cols
```

Out[33]:

```
['ApplicantIncome',
 'CoapplicantIncome',
 'LoanAmount',
 'Loan_Amount_Term',
 'Credit_History']
```

In [26]:

```
loan.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 357 entries, 0 to 366
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Loan_ID            357 non-null    object 
 1   Gender             357 non-null    object 
 2   Married            357 non-null    object 
 3   Dependents         357 non-null    object 
 4   Education          357 non-null    object 
 5   Self_Employed      357 non-null    object 
 6   ApplicantIncome    357 non-null    int64  
 7   CoapplicantIncome  357 non-null    int64  
 8   LoanAmount         357 non-null    int64  
 9   Loan_Amount_Term   357 non-null    int64  
 10  Credit_History     357 non-null    int64  
 11  Property_Area      357 non-null    object 
dtypes: int64(5), object(7)
memory usage: 36.3+ KB
```

In [34]:

```
# Select all data types of columns in a DF except object data type.
col_other_than_object = [col for col in loan.columns if loan[col].dtype != 'object']
col_other_than_object
```

Out[34]:

```
['ApplicantIncome',
 'CoapplicantIncome',
 'LoanAmount',
 'Loan_Amount_Term',
 'Credit_History']
```

In [35]:

```
#Extract all the Records where Self_Employed is equal to Yes.
self_employed_yes = loan[loan['Self_Employed']=='Yes']
self_employed_yes
```

Out[35]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	
	Male	Yes	0	Not Graduate	Yes	2165	3422	
	Male	Yes	0	Graduate	Yes	2267	2792	
	Male	No	0	Graduate	Yes	5833	0	
	Male	Yes	3+	Not Graduate	Yes	8000	250	
	Male	Yes	0	Not Graduate	Yes	5293	0	
	Male	Yes	0	Graduate	Yes	7016	292	
	Male	Yes	0	Graduate	Yes	3900	2094	
	Male	No	0	Graduate	Yes	6356	0	
	Male	Yes	0	Graduate	Yes	3188	2286	
	Female	No	0	Graduate	Yes	4742	0	
	Male	Yes	1	Graduate	Yes	3343	1517	
	Male	Yes	0	Graduate	Yes	32000	0	
	Male	Yes	2	Graduate	Yes	10890	0	
	Male	No	0	Not Graduate	Yes	8703	0	
	Male	No	0	Not Graduate	Yes	1599	2474	
	Male	Yes	2	Graduate	Yes	4246	4246	
	Male	No	0	Graduate	Yes	5833	0	
	Male	Yes	0	Graduate	Yes	1900	1442	
	Male	Yes	0	Graduate	Yes	8706	0	
	Male	Yes	3+	Graduate	Yes	5384	0	
	Male	Yes	1	Graduate	Yes	3507	3148	
	Female	No	2	Graduate	Yes	5184	0	
	Male	No	0	Graduate	Yes	2860	2988	
	Male	Yes	2	Graduate	Yes	5000	2166	
	Male	Yes	0	Not Graduate	Yes	3943	0	
	Female	No	0	Graduate	Yes	3333	3916	
	Male	Yes	1	Graduate	Yes	7500	0	
	Male	Yes	1	Not Graduate	Yes	570	2125	
	Male	Yes	2	Graduate	Yes	7500	0	
	Male	Yes	3+	Graduate	Yes	6958	1411	
	Male	Yes	1	Graduate	Yes	2360	3355	
	Male	Yes	0	Graduate	Yes	2623	4831	

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	I
	Male	No	0	Graduate	Yes	3972	4275	
	Male	Yes	0	Graduate	Yes	2500	296	
	Male	Yes	3+	Not Graduate	Yes	4009	1777	
In [35]:	Male	No	0	Graduate	Yes	9200	0	

```
#Extract all the Records where Property_area is equal to urban in a DF.

priority_area_urban = loan[loan['Property_Area'] == 'Urban']
priority_area_urban
```

Out[36]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	Coap
0	LP001015	Male	Yes	0	Graduate	No	5720	
1	LP001022	Male	Yes	1	Graduate	No	3076	
2	LP001031	Male	Yes	2	Graduate	No	5000	
3	LP001035	Male	Yes	2	Graduate	No	2340	
4	LP001051	Male	No	0	Not Graduate	No	3276	
...	...	...	...	...	...	...	...	
356	LP002935	Male	Yes	1	Graduate	No	3791	
357	LP002952	Male	No	0	Graduate	No	2500	
360	LP002965	Female	Yes	0	Graduate	No	8550	
362	LP002971	Male	Yes	3+	Not Graduate	Yes	4009	
363	LP002975	Male	Yes	0	Graduate	No	4158	

136 rows × 12 columns

In [40]:

```
#Print number of unique vaues in Gender column
gender_unique_val = loan['Gender'].unique()
print(len(gender_unique_val))
```

2

In [ ]: