In [1]:

```
#importing libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.style as style
import seaborn as sns
import itertools
%matplotlib inline

#setting up plot style
style.use('seaborn-poster')
style.use('fivethirtyeight')
```

In [2]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [3]:

```
# set_option( ) is used to adjust the jupiter view
pd.set_option('display.max_rows',500)
pd.set_option('display.max_rows',500)
pd.set_option('display.width',1000)
pd.set_option('display.expand_frame_repr',False)
```

In [4]:

```
application=pd.read_csv(r'C:\Users\abhir\Downloads\28th\28th\application_data.csv')
```

In [5]:

```
application.head()
```

Out[5]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_(
0	100002	1	Cash loans	M	N	
1	100003	0	Cash loans	F	N	
2	100004	0	Revolving loans	M	Y	
3	100006	0	Cash loans	F	N	
4	100007	0	Cash loans	M	N	

5 rows × 122 columns

In [6]:

```
#database dimensions
print('database dimensions',application.shape)
```

```
database dimensions (307511, 122)
```

In [7]:

```
#database size
print('database size',application.size)
```

```
database size 37516342
```

In [8]:

```
#for large dataframes use verbose
application.info(verbose=True)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Data columns (total 122 columns):
 #     Column                         Dtype
---    ------                         -----
 0     SK_ID_CURR                     int64
 1     TARGET                         int64
 2     NAME_CONTRACT_TYPE             object
 3     CODE_GENDER                    object
 4     FLAG_OWN_CAR                   object
 5     FLAG_OWN_REALTY                object
 6     CNT_CHILDREN                   int64
 7     AMT_INCOME_TOTAL               float64
 8     AMT_CREDIT                     float64
 9     AMT_ANNUITY                    float64
 10    AMT_GOODS_PRICE                float64
 11    NAME_TYPE_SUITE                object
 12    NAME_INCOME_TYPE               object
 13    NAME_EDUCATION_TYPE            object
 14    NAME_FAMILY_STATUS             object
```

In [9]:

```
application.describe()
```

Out[9]:

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AM
count	307511.000000	307511.000000	307511.000000	3.075110e+05	3.075110e+05	307
mean	278180.518577	0.080729	0.417052	1.687979e+05	5.990260e+05	27
std	102790.175348	0.272419	0.722121	2.371231e+05	4.024908e+05	14
min	100002.000000	0.000000	0.000000	2.565000e+04	4.500000e+04	1
25%	189145.500000	0.000000	0.000000	1.125000e+05	2.700000e+05	16
50%	278202.000000	0.000000	0.000000	1.471500e+05	5.135310e+05	24
75%	367142.500000	0.000000	1.000000	2.025000e+05	8.086500e+05	34
max	456255.000000	1.000000	19.000000	1.170000e+08	4.050000e+06	258

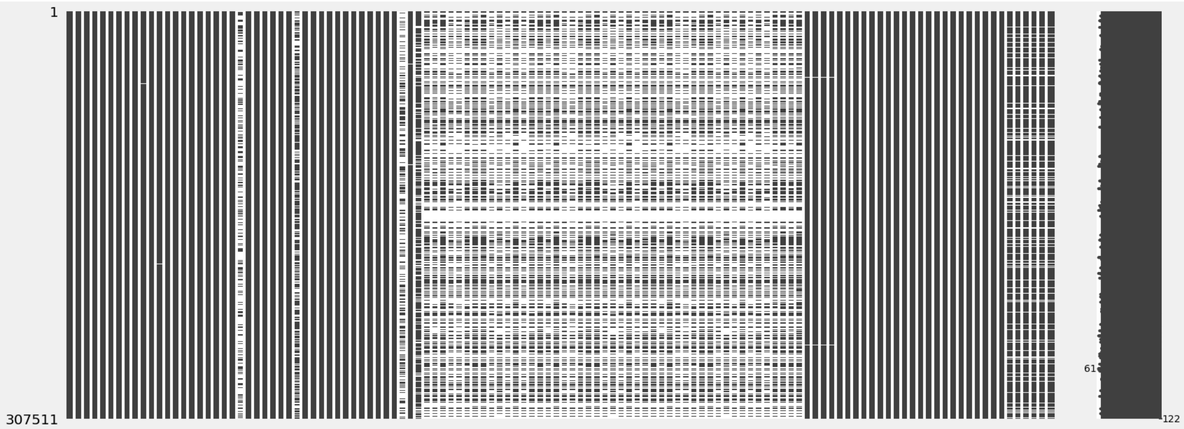
8 rows × 106 columns

In [10]:

```
#application missing vales
import missingno as mn
mn.matrix(application)
```

Out[10]:

<AxesSubplot:>



In [11]:

```
#sum of null value in each column
application.isnull().sum()
```

Out[11]:

```
SK_ID_CURR                      0
TARGET                          0
NAME_CONTRACT_TYPE              0
CODE_GENDER                     0
FLAG_OWN_CAR                    0
FLAG_OWN_REALTY                 0
CNT_CHILDREN                    0
AMT_INCOME_TOTAL                0
AMT_CREDIT                      0
AMT_ANNUITY                    12
AMT_GOODS_PRICE               278
NAME_TYPE_SUITE              1292
NAME_INCOME_TYPE                0
NAME_EDUCATION_TYPE             0
NAME_FAMILY_STATUS              0
NAME_HOUSING_TYPE               0
REGION_POPULATION_RELATIVE      0
DAYS_BIRTH                      0
```

In [12]:

```
application.shape[0]
```

Out[12]:

```
307511
```

In [13]:

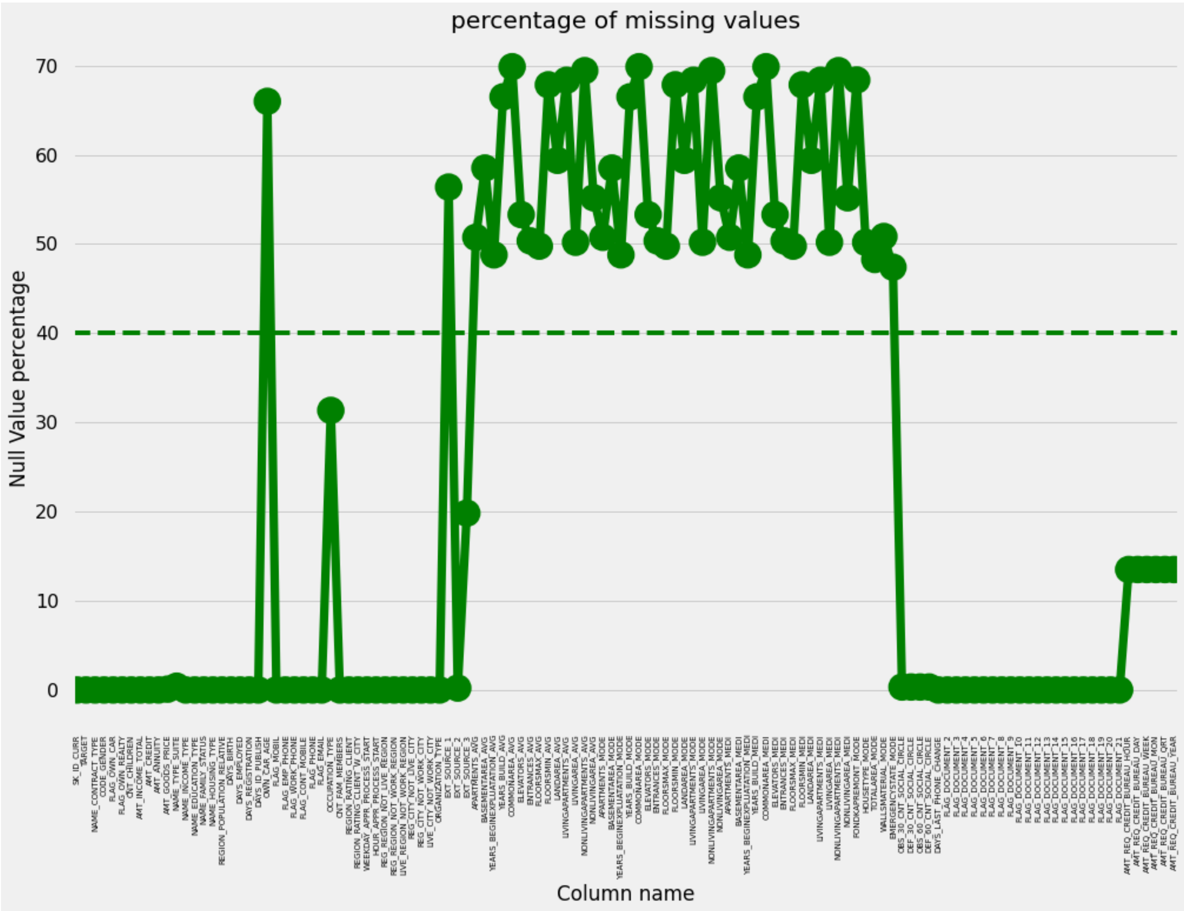
```
# % of null values
round(application.isnull().sum()/application.shape[0]*100.00,2)
```

Out[13]:

```
SK_ID_CURR                    0.00
TARGET                        0.00
NAME_CONTRACT_TYPE            0.00
CODE_GENDER                   0.00
FLAG_OWN_CAR                  0.00
FLAG_OWN_REALTY               0.00
CNT_CHILDREN                  0.00
AMT_INCOME_TOTAL              0.00
AMT_CREDIT                    0.00
AMT_ANNUITY                   0.00
AMT_GOODS_PRICE               0.09
NAME_TYPE_SUITE               0.42
NAME_INCOME_TYPE              0.00
NAME_EDUCATION_TYPE           0.00
NAME_FAMILY_STATUS            0.00
NAME_HOUSING_TYPE             0.00
REGION_POPULATION_RELATIVE    0.00
DAYS_BIRTH                    0.00
```

In [14]:

```
#to plot the columns Vs missing value % taking 40% as cutoff mark
null_application=pd.DataFrame(application.isnull().sum()/application.shape[0]*100).reset_in
null_application.columns=['Column name','Null Value percentage']
fig=plt.figure(figsize=(15,10))
ax=sns.pointplot(x='Column name',y='Null Value percentage',data=null_application,color='gre
plt.xticks(rotation=90,fontsize=7)
ax.axhline(40,ls='--',color='green')
plt.title('percentage of missing values')
plt.xlabel=('columns')
plt.ylabel=('Null values %')
plt.show()
```



In [15]:

```
#columns having more than 40% empty rows
nullColumns_40=null_application[null_application['Null Value percentage']>=40]
nullColumns_40
```

Out[15]:

	Column name	Null Value percentage
21	OWN_CAR_AGE	65.990810
41	EXT_SOURCE_1	56.381073
44	APARTMENTS_AVG	50.749729
45	BASEMENTAREA_AVG	58.515956
46	YEARS_BEGINEXPLUATATION_AVG	48.781019
47	YEARS_BUILD_AVG	66.497784
48	COMMONAREA_AVG	69.872297
49	ELEVATORS_AVG	53.295980
50	ENTRANCES_AVG	50.348768
51	FLOORSMAX_AVG	49.760822
52	FLOORSMIN_AVG	67.848630
53	LANDAREA_AVG	59.376738
54	LIVINGAPARTMENTS_AVG	68.354953
55	LIVINGAREA_AVG	50.193326
56	NONLIVINGAPARTMENTS_AVG	69.432963
57	NONLIVINGAREA_AVG	55.179164
58	APARTMENTS_MODE	50.749729
59	BASEMENTAREA_MODE	58.515956
60	YEARS_BEGINEXPLUATATION_MODE	48.781019
61	YEARS_BUILD_MODE	66.497784
62	COMMONAREA_MODE	69.872297
63	ELEVATORS_MODE	53.295980
64	ENTRANCES_MODE	50.348768
65	FLOORSMAX_MODE	49.760822
66	FLOORSMIN_MODE	67.848630
67	LANDAREA_MODE	59.376738
68	LIVINGAPARTMENTS_MODE	68.354953
69	LIVINGAREA_MODE	50.193326
70	NONLIVINGAPARTMENTS_MODE	69.432963
71	NONLIVINGAREA_MODE	55.179164
72	APARTMENTS_MEDI	50.749729
73	BASEMENTAREA_MEDI	58.515956
74	YEARS_BEGINEXPLUATATION_MEDI	48.781019

	Column name	Null Value percentage
75	YEARS_BUILD_MEDI	66.497784
76	COMMONAREA_MEDI	69.872297
77	ELEVATORS_MEDI	53.295980
78	ENTRANCES_MEDI	50.348768
79	FLOORSMAX_MEDI	49.760822
80	FLOORSMIN_MEDI	67.848630
81	LANDAREA_MEDI	59.376738
82	LIVINGAPARTMENTS_MEDI	68.354953
83	LIVINGAREA_MEDI	50.193326
84	NONLIVINGAPARTMENTS_MEDI	69.432963
85	NONLIVINGAREA_MEDI	55.179164
86	FONDKAPREMONT_MODE	68.386172
87	HOUSETYPE_MODE	50.176091
88	TOTALAREA_MODE	48.268517
89	WALLSMATERIAL_MODE	50.840783
90	EMERGENCYSTATE_MODE	47.398304

In [16]:

```
#no of coluns with missing values more than or equal to 40%
len(nullColumns_40)
```

Out[16]:

```
49
```

In [17]:

```
previous=pd.read_csv(r'C:\Users\abhir\Downloads\archive (1)\previous_application.csv')
```

In [18]:

```
previous.head()
```

Out[18]:

	SK_ID_PREV	SK_ID_CURR	NAME_CONTRACT_TYPE	AMT_ANNUITY	AMT_APPLICATION	AI
0	2030495	271877	Consumer loans	1730.430	17145.0	
1	2802425	108129	Cash loans	25188.615	607500.0	
2	2523466	122040	Cash loans	15060.735	112500.0	
3	2819243	176158	Cash loans	47041.335	450000.0	
4	1784265	202054	Cash loans	31924.395	337500.0	

5 rows × 37 columns

In [19]:

```
print('data dimensions',previous.shape)
```

```
data dimensions (1670214, 37)
```

In [20]:

```
print('data size',previous.size)
```

```
data size 61797918
```

In [21]:

```
previous.info(verbose=True)
```

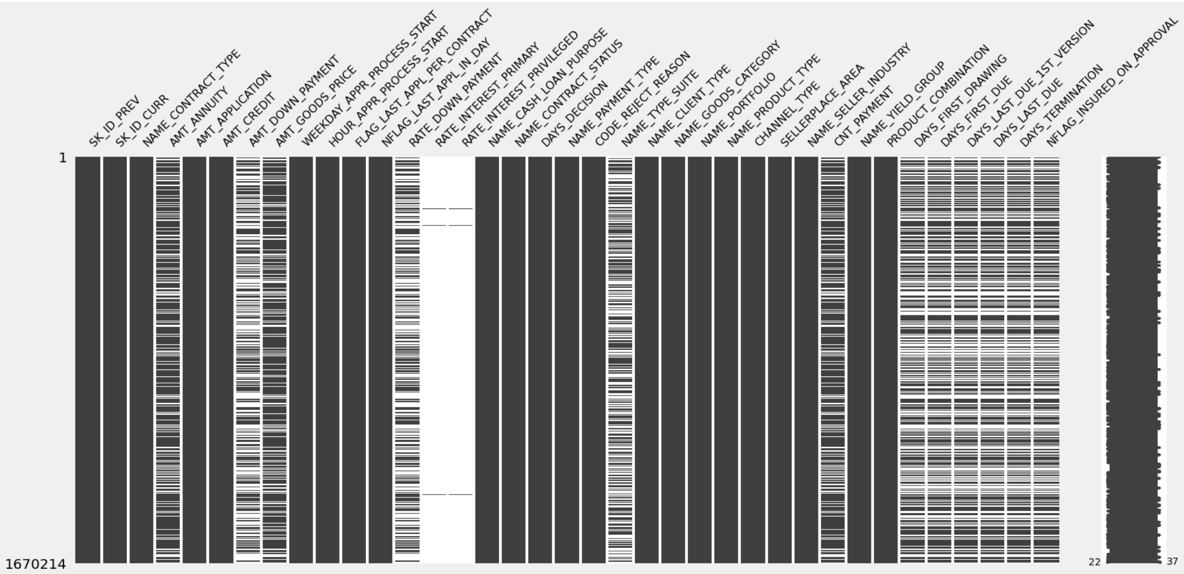
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1670214 entries, 0 to 1670213
Data columns (total 37 columns):
 #   Column                       Non-Null Count    Dtype  
---  ------                       --------------    -----  
 0   SK_ID_PREV                   1670214 non-null  int64  
 1   SK_ID_CURR                   1670214 non-null  int64  
 2   NAME_CONTRACT_TYPE           1670214 non-null  object 
 3   AMT_ANNUITY                  1297979 non-null  float64
 4   AMT_APPLICATION              1670214 non-null  float64
 5   AMT_CREDIT                   1670213 non-null  float64
 6   AMT_DOWN_PAYMENT             774370 non-null   float64
 7   AMT_GOODS_PRICE              1284699 non-null  float64
 8   WEEKDAY_APPR_PROCESS_START   1670214 non-null  object 
 9   HOUR_APPR_PROCESS_START      1670214 non-null  int64  
 10  FLAG_LAST_APPL_PER_CONTRACT  1670214 non-null  object 
 11  NFLAG_LAST_APPL_IN_DAY       1670214 non-null  int64  
 12  RATE_DOWN_PAYMENT            774370 non-null   float64
 13  RATE_INTEREST_PRIMARY        5951 non-null     float64
 14  RATE_INTEREST_PRIVILEGED     5951 non-null     float64
 15  NAME_CASH_LOAN_PURPOSE       1670214 non-null  object 
 16  NAME_CONTRACT_STATUS         1670214 non-null  object 
 17  DAYS_DECISION                1670214 non-null  int64  
 18  NAME_PAYMENT_TYPE            1670214 non-null  object 
 19  CODE_REJECT_REASON           1670214 non-null  object 
 20  NAME_TYPE_SUITE              849809 non-null   object 
 21  NAME_CLIENT_TYPE             1670214 non-null  object 
 22  NAME_GOODS_CATEGORY          1670214 non-null  object 
 23  NAME_PORTFOLIO               1670214 non-null  object 
 24  NAME_PRODUCT_TYPE            1670214 non-null  object 
 25  CHANNEL_TYPE                 1670214 non-null  object 
 26  SELLERPLACE_AREA             1670214 non-null  int64  
 27  NAME_SELLER_INDUSTRY         1670214 non-null  object 
 28  CNT_PAYMENT                  1297984 non-null  float64
 29  NAME_YIELD_GROUP             1670214 non-null  object 
 30  PRODUCT_COMBINATION          1669868 non-null  object 
 31  DAYS_FIRST_DRAWING           997149 non-null   float64
 32  DAYS_FIRST_DUE               997149 non-null   float64
 33  DAYS_LAST_DUE_1ST_VERSION    997149 non-null   float64
 34  DAYS_LAST_DUE                997149 non-null   float64
 35  DAYS_TERMINATION             997149 non-null   float64
 36  NFLAG_INSURED_ON_APPROVAL    997149 non-null   float64
dtypes: float64(15), int64(6), object(16)
memory usage: 471.5+ MB
```

In [22]:

```
mn.matrix(previous)
```

Out[22]:

<AxesSubplot:>



In [23]:

```
#number of null values in each column in previous dataset
previous.isnull().sum()
```

Out[23]:

```
SK_ID_PREV                           0
SK_ID_CURR                           0
NAME_CONTRACT_TYPE                   0
AMT_ANNUITY                     372235
AMT_APPLICATION                      0
AMT_CREDIT                           1
AMT_DOWN_PAYMENT                895844
AMT_GOODS_PRICE                 385515
WEEKDAY_APPR_PROCESS_START           0
HOUR_APPR_PROCESS_START              0
FLAG_LAST_APPL_PER_CONTRACT          0
NFLAG_LAST_APPL_IN_DAY               0
RATE_DOWN_PAYMENT               895844
RATE_INTEREST_PRIMARY          1664263
RATE_INTEREST_PRIVILEGED       1664263
NAME_CASH_LOAN_PURPOSE               0
NAME_CONTRACT_STATUS                 0
DAYS_DECISION                        0
NAME_PAYMENT_TYPE                    0
CODE_REJECT_REASON                   0
NAME_TYPE_SUITE                 820405
NAME_CLIENT_TYPE                     0
NAME_GOODS_CATEGORY                  0
NAME_PORTFOLIO                       0
NAME_PRODUCT_TYPE                    0
CHANNEL_TYPE                         0
SELLERPLACE_AREA                     0
NAME_SELLER_INDUSTRY                 0
CNT_PAYMENT                     372230
NAME_YIELD_GROUP                     0
PRODUCT_COMBINATION                346
DAYS_FIRST_DRAWING              673065
DAYS_FIRST_DUE                  673065
DAYS_LAST_DUE_1ST_VERSION       673065
DAYS_LAST_DUE                   673065
DAYS_TERMINATION                673065
NFLAG_INSURED_ON_APPROVAL       673065
dtype: int64
```

In [24]:

```
# % of null values in each columns
round(previous.isnull().sum()/previous.shape[0]*100.00,2)
```

Out[24]:

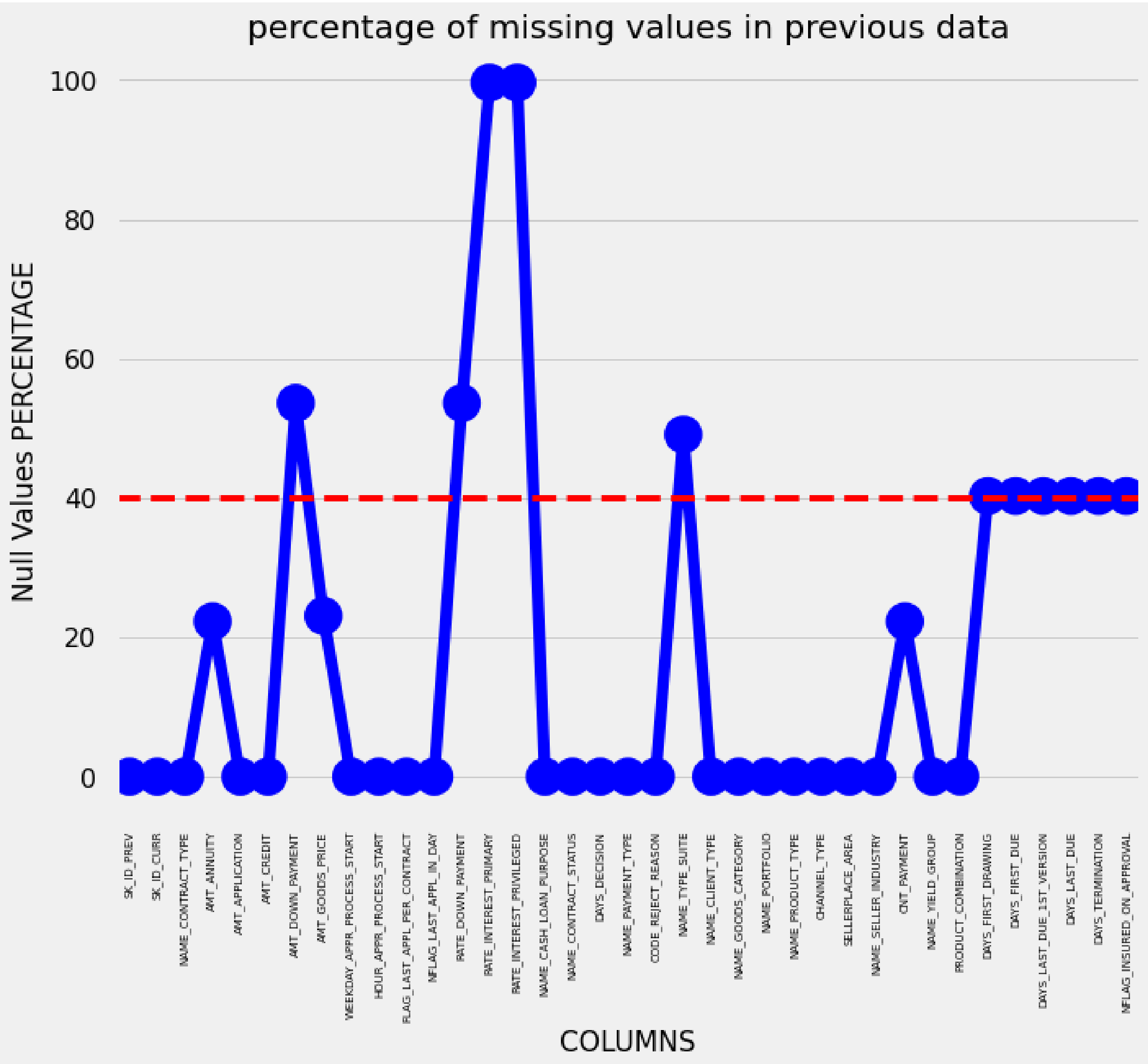
```
SK_ID_PREV                      0.00
SK_ID_CURR                      0.00
NAME_CONTRACT_TYPE              0.00
AMT_ANNUITY                    22.29
AMT_APPLICATION                 0.00
AMT_CREDIT                      0.00
AMT_DOWN_PAYMENT               53.64
AMT_GOODS_PRICE                23.08
WEEKDAY_APPR_PROCESS_START      0.00
HOUR_APPR_PROCESS_START         0.00
FLAG_LAST_APPL_PER_CONTRACT     0.00
NFLAG_LAST_APPL_IN_DAY          0.00
RATE_DOWN_PAYMENT              53.64
RATE_INTEREST_PRIMARY          99.64
RATE_INTEREST_PRIVILEGED       99.64
NAME_CASH_LOAN_PURPOSE          0.00
NAME_CONTRACT_STATUS            0.00
DAYS_DECISION                   0.00
NAME_PAYMENT_TYPE               0.00
CODE_REJECT_REASON              0.00
NAME_TYPE_SUITE                49.12
NAME_CLIENT_TYPE                0.00
NAME_GOODS_CATEGORY             0.00
NAME_PORTFOLIO                  0.00
NAME_PRODUCT_TYPE               0.00
CHANNEL_TYPE                    0.00
SELLERPLACE_AREA                0.00
NAME_SELLER_INDUSTRY            0.00
CNT_PAYMENT                    22.29
NAME_YIELD_GROUP                0.00
PRODUCT_COMBINATION             0.02
DAYS_FIRST_DRAWING             40.30
DAYS_FIRST_DUE                 40.30
DAYS_LAST_DUE_1ST_VERSION      40.30
DAYS_LAST_DUE                  40.30
DAYS_TERMINATION               40.30
NFLAG_INSURED_ON_APPROVAL      40.30
dtype: float64
```

In [25]:

```
#added this as I got error as 'str' object is not callable
import matplotlib.pyplot as plt
from importlib import reload
plt=reload(plt)
```

In [26]:

```
#to plot the columns Vs missing value % and taking 40% as cutoff mark
null_previous=pd.DataFrame(previous.isnull().sum()/previous.shape[0]*100).reset_index()
null_previous.columns = ['Column Name', 'Null Values Percentage']
fig=plt.figure(figsize=(10,8))
ax = sns.pointplot(x="Column Name",y="Null Values Percentage",data=null_previous,color ='bl
plt.xticks(rotation=90,fontsize=7)
ax.axhline(40,ls='--',color='red')
plt.title('percentage of missing values in previous data')
plt.ylabel("Null Values PERCENTAGE")
plt.xlabel("COLUMNS")
plt.show()
```



In [27]:

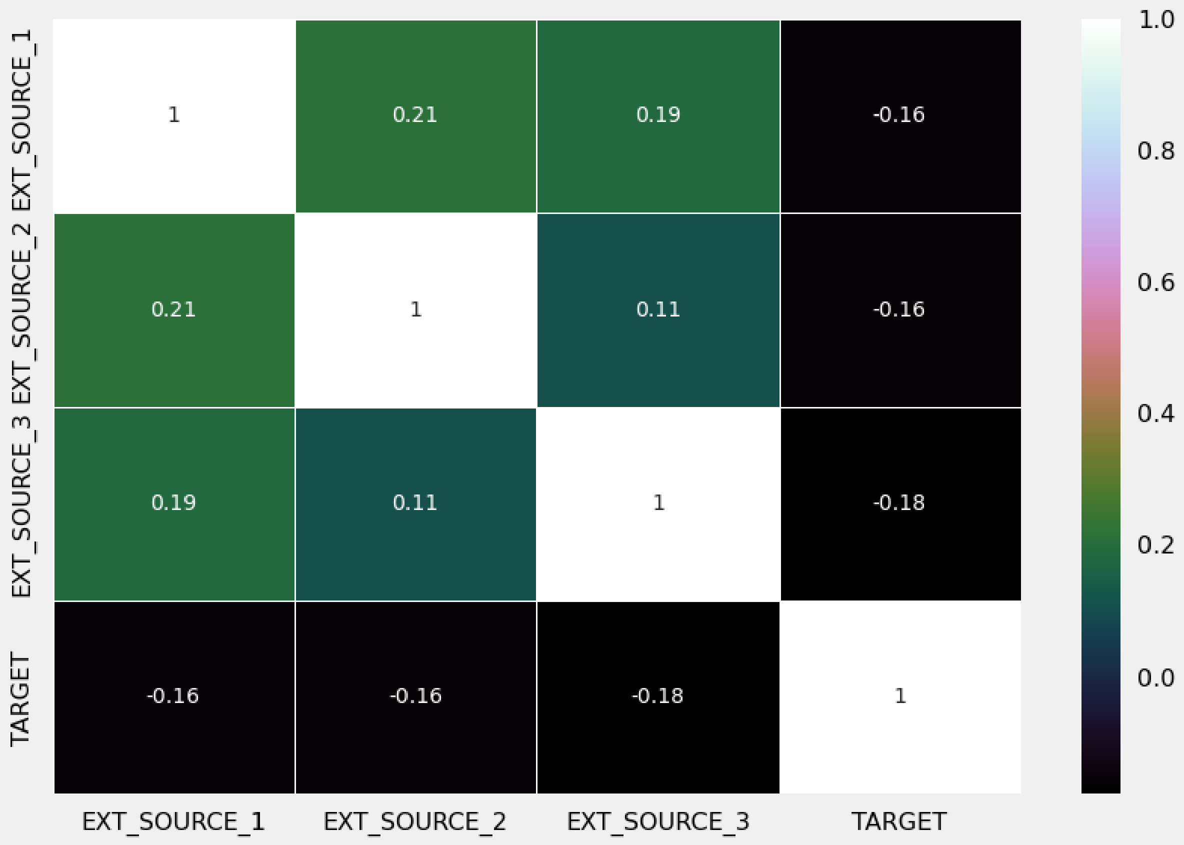
```
#columns having more than 40% empty rows
nullColumns_40prev=null_previous[null_previous['Null Values Percentage']>=40]
nullColumns_40prev
```

Out[27]:

	Column Name	Null Values Percentage
6	AMT_DOWN_PAYMENT	53.636480
12	RATE_DOWN_PAYMENT	53.636480
13	RATE_INTEREST_PRIMARY	99.643698
14	RATE_INTEREST_PRIVILEGED	99.643698
20	NAME_TYPE_SUITE	49.119754
31	DAYS_FIRST_DRAWING	40.298129
32	DAYS_FIRST_DUE	40.298129
33	DAYS_LAST_DUE_1ST_VERSION	40.298129
34	DAYS_LAST_DUE	40.298129
35	DAYS_TERMINATION	40.298129
36	NFLAG_INSURED_ON_APPROVAL	40.298129

In [28]:

```
#verfying whether EXT_SOURCE is correlated with target
source_corr=application[['EXT_SOURCE_1','EXT_SOURCE_2','EXT_SOURCE_3','TARGET']]
ax=sns.heatmap(source_corr.corr(),xticklabels=source_corr.columns, yticklabels=source_corr.
```



In [29]:

```
#creating a list with columns having >40% null values and adding the EXT_SOURCE
#.list() is used to convert a series to list
unwanted_columns=nullColumns_40['Column name'].tolist()+['EXT_SOURCE_2','EXT_SOURCE_3']
len(unwanted_columns)
```

Out[29]:

```
51
```

In [30]:

```
#added this as I got error 'str' object is not callable
import matplotlib.pyplot as plt
from importlib import reload
plt=reload(plt)
```

In [31]:

```
#checking wether the submitted flag_docments is related with Loan repayment status
col_Doc = [ 'FLAG_DOCUMENT_2', 'FLAG_DOCUMENT_3','FLAG_DOCUMENT_4', 'FLAG_DOCUMENT_5', 'FLA
           'FLAG_DOCUMENT_8', 'FLAG_DOCUMENT_9','FLAG_DOCUMENT_10', 'FLAG_DOCUMENT_11', 'FL
           'FLAG_DOCUMENT_14', 'FLAG_DOCUMENT_15','FLAG_DOCUMENT_16', 'FLAG_DOCUMENT_17', '
           'FLAG_DOCUMENT_19', 'FLAG_DOCUMENT_20', 'FLAG_DOCUMENT_21']
df_flag = application[col_Doc+["TARGET"]]
df_flag["TARGET"] = df_flag["TARGET"].replace({1:"Defaulter",0:"Repayer"})
fig = plt.figure(figsize=(25,20))
j=0
for i in col_Doc:
    plt.subplot(4,5,j+1)
    ax = sns.countplot(df_flag[i],hue=df_flag["TARGET"],palette=["r","g"])
    plt.yticks(fontsize=0.1)
    plt.title(i)
    plt.yticks(fontsize=8)
    plt.xlabel("")
    plt.ylabel("")
    j=j+1
```

In [32]:

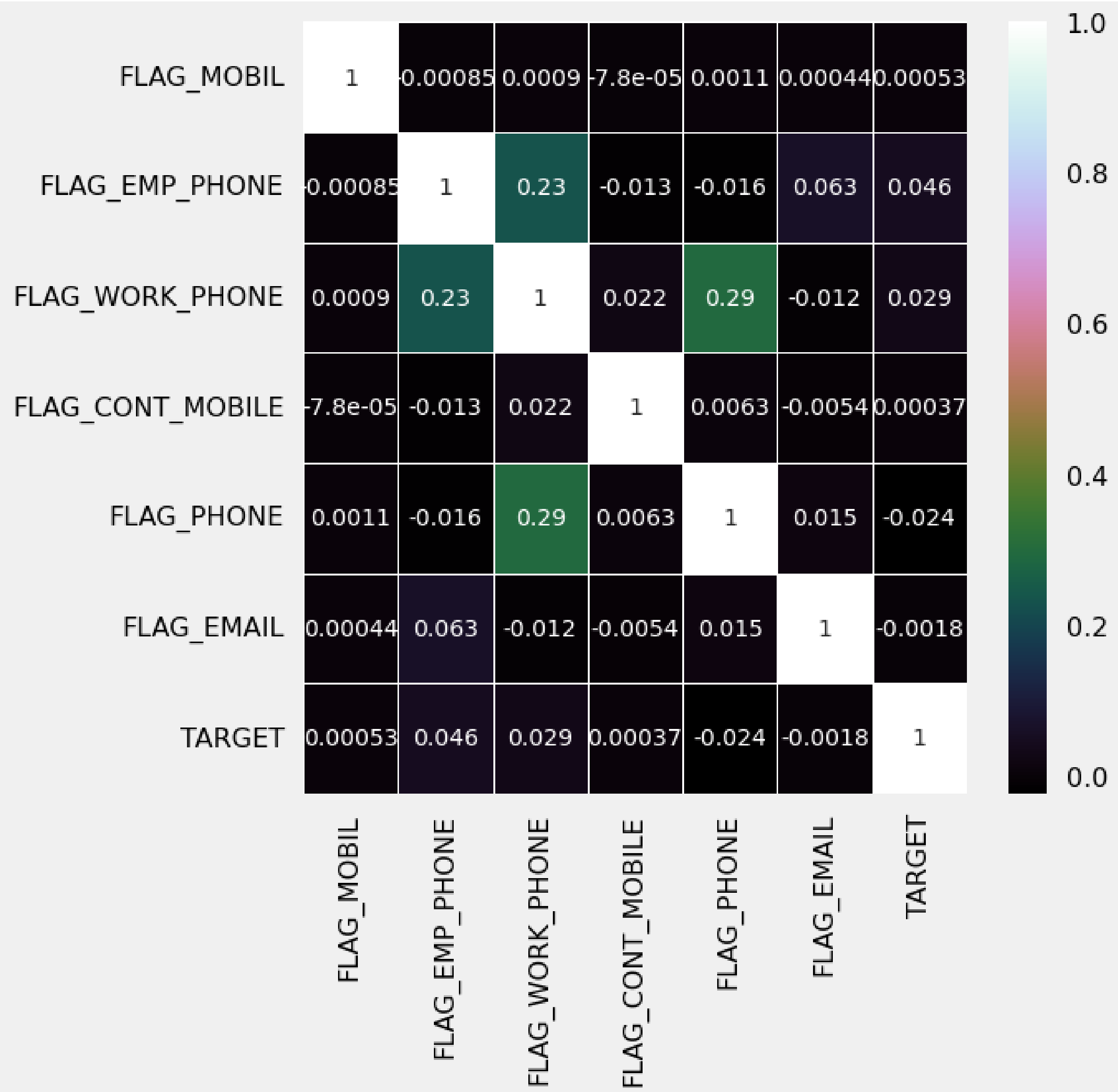
```
col_Doc.remove('FLAG_DOCUMENT_3')
unwanted_columns = unwanted_columns + col_Doc
len(unwanted_columns)
```

Out[32]:

```
70
```

In [33]:

```
#checking the correlation b/w contact details
contact_col = ['FLAG_MOBIL', 'FLAG_EMP_PHONE', 'FLAG_WORK_PHONE', 'FLAG_CONT_MOBILE', 'FLAG
Contact_corr = application[contact_col].corr()
fig = plt.figure(figsize=(8,8))
ax = sns.heatmap(Contact_corr, xticklabels=Contact_corr.columns,yticklabels=Contact_corr.co
```



In [34]:

```
#adding contact details to unwanted columns
contact_col.remove('TARGET')
unwanted_columns=unwanted_columns+contact_col
len(unwanted_columns)
```

Out[34]:

```
76
```

In [35]:

```
unwanted_columns
```

Out[35]:

```
['OWN_CAR_AGE',
 'EXT_SOURCE_1',
 'APARTMENTS_AVG',
 'BASEMENTAREA_AVG',
 'YEARS_BEGINEXPLUATATION_AVG',
 'YEARS_BUILD_AVG',
 'COMMONAREA_AVG',
 'ELEVATORS_AVG',
 'ENTRANCES_AVG',
 'FLOORSMAX_AVG',
 'FLOORSMIN_AVG',
 'LANDAREA_AVG',
 'LIVINGAPARTMENTS_AVG',
 'LIVINGAREA_AVG',
 'NONLIVINGAPARTMENTS_AVG',
 'NONLIVINGAREA_AVG',
 'APARTMENTS_MODE',
 'BASEMENTAREA_MODE',
 'YEARS_BEGINEXPLUATATION_MODE',
 'YEARS_BUILD_MODE',
 'COMMONAREA_MODE',
 'ELEVATORS_MODE',
 'ENTRANCES_MODE',
 'FLOORSMAX_MODE',
 'FLOORSMIN_MODE',
 'LANDAREA_MODE',
 'LIVINGAPARTMENTS_MODE',
 'LIVINGAREA_MODE',
 'NONLIVINGAPARTMENTS_MODE',
 'NONLIVINGAREA_MODE',
 'APARTMENTS_MEDI',
 'BASEMENTAREA_MEDI',
 'YEARS_BEGINEXPLUATATION_MEDI',
 'YEARS_BUILD_MEDI',
 'COMMONAREA_MEDI',
 'ELEVATORS_MEDI',
 'ENTRANCES_MEDI',
 'FLOORSMAX_MEDI',
 'FLOORSMIN_MEDI',
 'LANDAREA_MEDI',
 'LIVINGAPARTMENTS_MEDI',
 'LIVINGAREA_MEDI',
 'NONLIVINGAPARTMENTS_MEDI',
 'NONLIVINGAREA_MEDI',
 'FONDKAPREMONT_MODE',
 'HOUSETYPE_MODE',
 'TOTALAREA_MODE',
 'WALLSMATERIAL_MODE',
 'EMERGENCYSTATE_MODE',
 'EXT_SOURCE_2',
 'EXT_SOURCE_3',
 'FLAG_DOCUMENT_2',
 'FLAG_DOCUMENT_4',
 'FLAG_DOCUMENT_5',
 'FLAG_DOCUMENT_6',
```

```
 'FLAG_DOCUMENT_7',
 'FLAG_DOCUMENT_8',
 'FLAG_DOCUMENT_9',
 'FLAG_DOCUMENT_10',
 'FLAG_DOCUMENT_11',
 'FLAG_DOCUMENT_12',
 'FLAG_DOCUMENT_13',
 'FLAG_DOCUMENT_14',
 'FLAG_DOCUMENT_15',
 'FLAG_DOCUMENT_16',
 'FLAG_DOCUMENT_17',
 'FLAG_DOCUMENT_18',
 'FLAG_DOCUMENT_19',
 'FLAG_DOCUMENT_20',
 'FLAG_DOCUMENT_21',
 'FLAG_MOBIL',
 'FLAG_EMP_PHONE',
 'FLAG_WORK_PHONE',
 'FLAG_CONT_MOBILE',
 'FLAG_PHONE',
 'FLAG_EMAIL']
```

In [36]:

```
#removed all unwanted columns from the application set
application.drop(labels=unwanted_columns,axis=1,inplace=True)
```

In [37]:

```
application.columns
```

Out[37]:

```
Index(['SK_ID_CURR', 'TARGET', 'NAME_CONTRACT_TYPE', 'CODE_GENDER', 'FLAG_OW
N_CAR', 'FLAG_OWN_REALTY', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL', 'AMT_CREDIT',
'AMT_ANNUITY', 'AMT_GOODS_PRICE', 'NAME_TYPE_SUITE', 'NAME_INCOME_TYPE', 'NA
ME_EDUCATION_TYPE', 'NAME_FAMILY_STATUS', 'NAME_HOUSING_TYPE', 'REGION_POPUL
ATION_RELATIVE', 'DAYS_BIRTH', 'DAYS_EMPLOYED', 'DAYS_REGISTRATION', 'DAYS_I
D_PUBLISH', 'OCCUPATION_TYPE', 'CNT_FAM_MEMBERS', 'REGION_RATING_CLIENT', 'R
EGION_RATING_CLIENT_W_CITY', 'WEEKDAY_APPR_PROCESS_START', 'HOUR_APPR_PROCES
S_START', 'REG_REGION_NOT_LIVE_REGION', 'REG_REGION_NOT_WORK_REGION', 'LIVE_
REGION_NOT_WORK_REGION', 'REG_CITY_NOT_LIVE_CITY', 'REG_CITY_NOT_WORK_CITY',
'LIVE_CITY_NOT_WORK_CITY', 'ORGANIZATION_TYPE', 'OBS_30_CNT_SOCIAL_CIRCLE',
'DEF_30_CNT_SOCIAL_CIRCLE', 'OBS_60_CNT_SOCIAL_CIRCLE', 'DEF_60_CNT_SOCIAL_C
IRCLE', 'DAYS_LAST_PHONE_CHANGE', 'FLAG_DOCUMENT_3', 'AMT_REQ_CREDIT_BUREAU_
HOUR', 'AMT_REQ_CREDIT_BUREAU_DAY', 'AMT_REQ_CREDIT_BUREAU_WEEK',
       'AMT_REQ_CREDIT_BUREAU_MON', 'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_CR
EDIT_BUREAU_YEAR'],
      dtype='object')
```

In [38]:

```
application.shape
```

Out[38]:

```
(307511, 46)
```

In [39]:

```
application.info(verbose=True)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Data columns (total 46 columns):
 #   Column                       Non-Null Count   Dtype  
---  ------                       --------------   -----  
 0   SK_ID_CURR                   307511 non-null  int64  
 1   TARGET                       307511 non-null  int64  
 2   NAME_CONTRACT_TYPE           307511 non-null  object 
 3   CODE_GENDER                  307511 non-null  object 
 4   FLAG_OWN_CAR                 307511 non-null  object 
 5   FLAG_OWN_REALTY              307511 non-null  object 
 6   CNT_CHILDREN                 307511 non-null  int64  
 7   AMT_INCOME_TOTAL             307511 non-null  float64
 8   AMT_CREDIT                   307511 non-null  float64
 9   AMT_ANNUITY                  307499 non-null  float64
 10  AMT_GOODS_PRICE              307233 non-null  float64
 11  NAME_TYPE_SUITE              306219 non-null  object 
 12  NAME_INCOME_TYPE             307511 non-null  object 
 13  NAME_EDUCATION_TYPE          307511 non-null  object 
 14  NAME_FAMILY_STATUS           307511 non-null  object 
 15  NAME_HOUSING_TYPE            307511 non-null  object 
 16  REGION_POPULATION_RELATIVE   307511 non-null  float64
 17  DAYS_BIRTH                   307511 non-null  int64  
 18  DAYS_EMPLOYED                307511 non-null  int64  
 19  DAYS_REGISTRATION            307511 non-null  float64
 20  DAYS_ID_PUBLISH              307511 non-null  int64  
 21  OCCUPATION_TYPE              211120 non-null  object 
 22  CNT_FAM_MEMBERS              307509 non-null  float64
 23  REGION_RATING_CLIENT         307511 non-null  int64  
 24  REGION_RATING_CLIENT_W_CITY  307511 non-null  int64  
 25  WEEKDAY_APPR_PROCESS_START   307511 non-null  object 
 26  HOUR_APPR_PROCESS_START      307511 non-null  int64  
 27  REG_REGION_NOT_LIVE_REGION   307511 non-null  int64  
 28  REG_REGION_NOT_WORK_REGION   307511 non-null  int64  
 29  LIVE_REGION_NOT_WORK_REGION  307511 non-null  int64  
 30  REG_CITY_NOT_LIVE_CITY       307511 non-null  int64  
 31  REG_CITY_NOT_WORK_CITY       307511 non-null  int64  
 32  LIVE_CITY_NOT_WORK_CITY      307511 non-null  int64  
 33  ORGANIZATION_TYPE            307511 non-null  object 
 34  OBS_30_CNT_SOCIAL_CIRCLE     306490 non-null  float64
 35  DEF_30_CNT_SOCIAL_CIRCLE     306490 non-null  float64
 36  OBS_60_CNT_SOCIAL_CIRCLE     306490 non-null  float64
 37  DEF_60_CNT_SOCIAL_CIRCLE     306490 non-null  float64
 38  DAYS_LAST_PHONE_CHANGE       307510 non-null  float64
 39  FLAG_DOCUMENT_3              307511 non-null  int64  
 40  AMT_REQ_CREDIT_BUREAU_HOUR   265992 non-null  float64
 41  AMT_REQ_CREDIT_BUREAU_DAY    265992 non-null  float64
 42  AMT_REQ_CREDIT_BUREAU_WEEK   265992 non-null  float64
 43  AMT_REQ_CREDIT_BUREAU_MON    265992 non-null  float64
 44  AMT_REQ_CREDIT_BUREAU_QRT    265992 non-null  float64
 45  AMT_REQ_CREDIT_BUREAU_YEAR   265992 non-null  float64
dtypes: float64(18), int64(16), object(12)
memory usage: 107.9+ MB
```

In [40]:

```
#columns having more tha 40% null values converting to unwanted list
unwanted_previous=nullColumns_40prev['Column Name'].tolist()
unwanted_previous
```

Out[40]:

```
['AMT_DOWN_PAYMENT',
 'RATE_DOWN_PAYMENT',
 'RATE_INTEREST_PRIMARY',
 'RATE_INTEREST_PRIVILEGED',
 'NAME_TYPE_SUITE',
 'DAYS_FIRST_DRAWING',
 'DAYS_FIRST_DUE',
 'DAYS_LAST_DUE_1ST_VERSION',
 'DAYS_LAST_DUE',
 'DAYS_TERMINATION',
 'NFLAG_INSURED_ON_APPROVAL']
```

In [41]:

```
#removing some unnecessary columns
Unnecessary_previous = ['WEEKDAY_APPR_PROCESS_START','HOUR_APPR_PROCESS_START','FLAG_LAST_A
```

In [42]:

```
unwanted_previous=unwanted_previous+Unnecessary_previous
len(unwanted_previous)
```

Out[42]:

```
15
```

In [43]:

```
previous.drop(labels=unwanted_previous,axis=1,inplace=True)
```

In [44]:

```
previous.shape
```

Out[44]:

```
(1670214, 22)
```

In [45]:

```
previous.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1670214 entries, 0 to 1670213
Data columns (total 22 columns):
 #   Column                       Non-Null Count    Dtype
---  ------                       --------------    -----
 0   SK_ID_PREV                   1670214 non-null  int64
 1   SK_ID_CURR                   1670214 non-null  int64
 2   NAME_CONTRACT_TYPE           1670214 non-null  object
 3   AMT_ANNUITY                  1297979 non-null  float64
 4   AMT_APPLICATION              1670214 non-null  float64
 5   AMT_CREDIT                   1670213 non-null  float64
 6   AMT_GOODS_PRICE              1284699 non-null  float64
 7   NAME_CASH_LOAN_PURPOSE       1670214 non-null  object
 8   NAME_CONTRACT_STATUS         1670214 non-null  object
 9   DAYS_DECISION                1670214 non-null  int64
 10  NAME_PAYMENT_TYPE            1670214 non-null  object
 11  CODE_REJECT_REASON           1670214 non-null  object
 12  NAME_CLIENT_TYPE             1670214 non-null  object
 13  NAME_GOODS_CATEGORY          1670214 non-null  object
 14  NAME_PORTFOLIO               1670214 non-null  object
 15  NAME_PRODUCT_TYPE            1670214 non-null  object
 16  CHANNEL_TYPE                 1670214 non-null  object
 17  SELLERPLACE_AREA             1670214 non-null  int64
 18  NAME_SELLER_INDUSTRY         1670214 non-null  object
 19  CNT_PAYMENT                  1297984 non-null  float64
 20  NAME_YIELD_GROUP             1670214 non-null  object
 21  PRODUCT_COMBINATION          1669868 non-null  object
dtypes: float64(5), int64(4), object(13)
memory usage: 280.3+ MB
```

In [46]:

```
previous.describe()
```

Out[46]:

	SK_ID_PREV	SK_ID_CURR	AMT_ANNUITY	AMT_APPLICATION	AMT_CREDIT	AMT_GO
count	1.670214e+06	1.670214e+06	1.297979e+06	1.670214e+06	1.670213e+06	1.
mean	1.923089e+06	2.783572e+05	1.595512e+04	1.752339e+05	1.961140e+05	2.
std	5.325980e+05	1.028148e+05	1.478214e+04	2.927798e+05	3.185746e+05	3.
min	1.000001e+06	1.000010e+05	0.000000e+00	0.000000e+00	0.000000e+00	0.
25%	1.461857e+06	1.893290e+05	6.321780e+03	1.872000e+04	2.416050e+04	5.
50%	1.923110e+06	2.787145e+05	1.125000e+04	7.104600e+04	8.054100e+04	1.
75%	2.384280e+06	3.675140e+05	2.065842e+04	1.803600e+05	2.164185e+05	2.
max	2.845382e+06	4.562550e+05	4.180581e+05	6.905160e+06	6.905160e+06	6.

In [47]:

```
#converting negative to positive as days can't be negative
#abs() to return positive value
date_col=['DAYS_BIRTH','DAYS_EMPLOYED','DAYS_REGISTRATION','DAYS_ID_PUBLISH']
for col in date_col:
    application[col]=abs(application[col])
```

In [48]:

```
#Categorize the client income variable into bins
#pd.cut to seperate the array elements into different bins
application['AMT_INCOME_TOTAL']=application['AMT_INCOME_TOTAL']/100000
bins=[0,1,2,3,4,5,6,7,8,9,10,11]
slots=['0-100K','100K-200K','200K-300K','300K-400K','400K-500K','500K-600K','600K-700K','70
application['INCOME_RANGE']=pd.cut(application['AMT_INCOME_TOTAL'],bins,labels=slots)
```

In [49]:

```
#range of incomes
application['INCOME_RANGE']
```

Out[49]:

```
0         200K-300K
1         200K-300K
2            0-100K
3         100K-200K
4         100K-200K
            ...
307506    100K-200K
307507       0-100K
307508    100K-200K
307509    100K-200K
307510    100K-200K
Name: INCOME_RANGE, Length: 307511, dtype: category
Categories (11, object): ['0-100K' < '100K-200K' < '200K-300K' < '300K-400K'
... '700K-800K' < '800K-900K' < '900K-1M' < '1M Above']
```

In [50]:

```
#calculating the percentage in the range of salary
#IF normalize is true then the object returns the relative frequencies of the unique values
#More than 50% loan applicants have income amount in the range of 100K-200K. Almost 92% loa
application['INCOME_RANGE'].value_counts(normalize=True)*100
```

Out[50]:

```
100K-200K    50.735000
200K-300K    21.210691
0-100K       20.729695
300K-400K     4.776116
400K-500K     1.744669
500K-600K     0.356354
600K-700K     0.282805
800K-900K     0.096980
700K-800K     0.052721
900K-1M       0.009112
1M Above      0.005858
Name: INCOME_RANGE, dtype: float64
```

In [51]:

```
#Categorize the client credit amount into bins
application['AMT_CREDIT']=application['AMT_CREDIT']/100000
bins=[0,1,2,3,4,5,6,7,8,9,10,100]
slots = ['0 to 100K','100K to 200K', '200k to 300k','300k to 400k','400k to 500k','500k to
application['LOAN_AMT']=pd.cut(application['AMT_CREDIT'], bins,labels=slots)
```

In [52]:

```
#calculating the percentage of loan amount
application['LOAN_AMT'].value_counts(normalize=True)*100
```

Out[52]:

```
200k to 300k    17.824728
1M Above        16.254703
500k to 600k    11.131960
400k to 500k    10.418489
100K to 200K     9.801275
300k to 400k     8.564897
600k to 700k     7.820533
800k to 900k     7.086576
700k to 800k     6.241403
900k to 1M       2.902986
0 to 100K        1.952450
Name: LOAN_AMT, dtype: float64
```

In [53]:

```
#Categorize the client age into bins
application['AGE']=application['DAYS_BIRTH']//365
bins=(0,20,30,40,50,100)
slots=['0 to 20','20 to 30','30 to 40','40 to 50','50 above']
application['AGE_GROUP']=pd.cut(application['AGE'],bins,labels=slots)
```

In [54]:

```
#checking for which age group people are major
application['AGE_GROUP'].value_counts(normalize=True)*100
```

Out[54]:

```
50 above    31.604398
30 to 40    27.028952
40 to 50    24.194582
20 to 30    17.171743
0 to 20      0.000325
Name: AGE_GROUP, dtype: float64
```

In [55]:

```
#Categorize the clients working years into bins
application['YEARS_EMPLOYED'] = application['DAYS_EMPLOYED']//365
bins = [0,5,10,20,30,40,50,60,150]
slots = ['0-5','5-10','10-20','20-30','30-40','40-50','50-60','60 above']
application['WORKING_YEARS']=pd.cut(application['YEARS_EMPLOYED'],bins=bins,labels=slots)
```

In [56]:

```
application['WORKING_YEARS'].value_counts(normalize=True)*100
```

Out[56]:

```
0-5         55.582363
5-10        24.966441
10-20       14.564315
20-30        3.750117
30-40        1.058720
40-50        0.078044
50-60        0.000000
60 above     0.000000
Name: WORKING_YEARS, dtype: float64
```

In [57]:

```
#Checking the number of unique values each column possess to identify categorical columns
application.nunique().sort_values()
```

Out[57]:

```
LIVE_CITY_NOT_WORK_CITY            2
TARGET                             2
NAME_CONTRACT_TYPE                 2
REG_REGION_NOT_LIVE_REGION         2
FLAG_OWN_CAR                       2
FLAG_OWN_REALTY                    2
REG_REGION_NOT_WORK_REGION         2
LIVE_REGION_NOT_WORK_REGION        2
FLAG_DOCUMENT_3                    2
REG_CITY_NOT_LIVE_CITY             2
REG_CITY_NOT_WORK_CITY             2
REGION_RATING_CLIENT               3
CODE_GENDER                        3
REGION_RATING_CLIENT_W_CITY        3
AMT_REQ_CREDIT_BUREAU_HOUR         5
NAME_EDUCATION_TYPE                5
AGE_GROUP                          5
NAME_FAMILY_STATUS                 6
NAME_HOUSING_TYPE                  6
WORKING_YEARS                      6
WEEKDAY_APPR_PROCESS_START         7
NAME_TYPE_SUITE                    7
NAME_INCOME_TYPE                   8
AMT_REQ_CREDIT_BUREAU_WEEK         9
AMT_REQ_CREDIT_BUREAU_DAY          9
DEF_60_CNT_SOCIAL_CIRCLE           9
DEF_30_CNT_SOCIAL_CIRCLE          10
LOAN_AMT                          11
INCOME_RANGE                      11
AMT_REQ_CREDIT_BUREAU_QRT         11
CNT_CHILDREN                      15
CNT_FAM_MEMBERS                   17
OCCUPATION_TYPE                   18
HOUR_APPR_PROCESS_START           24
AMT_REQ_CREDIT_BUREAU_MON         24
AMT_REQ_CREDIT_BUREAU_YEAR        25
OBS_60_CNT_SOCIAL_CIRCLE          33
OBS_30_CNT_SOCIAL_CIRCLE          33
AGE                               50
YEARS_EMPLOYED                    51
ORGANIZATION_TYPE                 58
REGION_POPULATION_RELATIVE        81
AMT_GOODS_PRICE                 1002
AMT_INCOME_TOTAL                2548
DAYS_LAST_PHONE_CHANGE          3773
AMT_CREDIT                      5603
DAYS_ID_PUBLISH                 6168
DAYS_EMPLOYED                  12574
AMT_ANNUITY                    13672
DAYS_REGISTRATION              15688
DAYS_BIRTH                     17460
SK_ID_CURR                    307511
dtype: int64
```

In [58]:

```
#checking the data type of columns and correcting them
application.info(verbose=True)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Data columns (total 52 columns):
 #   Column                       Non-Null Count   Dtype
---  ------                       --------------   -----
 0   SK_ID_CURR                   307511 non-null  int64
 1   TARGET                       307511 non-null  int64
 2   NAME_CONTRACT_TYPE           307511 non-null  object
 3   CODE_GENDER                  307511 non-null  object
 4   FLAG_OWN_CAR                 307511 non-null  object
 5   FLAG_OWN_REALTY              307511 non-null  object
 6   CNT_CHILDREN                 307511 non-null  int64
 7   AMT_INCOME_TOTAL             307511 non-null  float64
 8   AMT_CREDIT                   307511 non-null  float64
 9   AMT_ANNUITY                  307499 non-null  float64
 10  AMT_GOODS_PRICE              307233 non-null  float64
 11  NAME_TYPE_SUITE              306219 non-null  object
 12  NAME_INCOME_TYPE             307511 non-null  object
 13  NAME_EDUCATION_TYPE          307511 non-null  object
 14  NAME_FAMILY_STATUS           307511 non-null  object
 15  NAME_HOUSING_TYPE            307511 non-null  object
 16  REGION_POPULATION_RELATIVE   307511 non-null  float64
 17  DAYS_BIRTH                   307511 non-null  int64
 18  DAYS_EMPLOYED                307511 non-null  int64
 19  DAYS_REGISTRATION            307511 non-null  float64
 20  DAYS_ID_PUBLISH              307511 non-null  int64
 21  OCCUPATION_TYPE              211120 non-null  object
 22  CNT_FAM_MEMBERS              307509 non-null  float64
 23  REGION_RATING_CLIENT         307511 non-null  int64
 24  REGION_RATING_CLIENT_W_CITY  307511 non-null  int64
 25  WEEKDAY_APPR_PROCESS_START   307511 non-null  object
 26  HOUR_APPR_PROCESS_START      307511 non-null  int64
 27  REG_REGION_NOT_LIVE_REGION   307511 non-null  int64
 28  REG_REGION_NOT_WORK_REGION   307511 non-null  int64
 29  LIVE_REGION_NOT_WORK_REGION  307511 non-null  int64
 30  REG_CITY_NOT_LIVE_CITY       307511 non-null  int64
 31  REG_CITY_NOT_WORK_CITY       307511 non-null  int64
 32  LIVE_CITY_NOT_WORK_CITY      307511 non-null  int64
 33  ORGANIZATION_TYPE            307511 non-null  object
 34  OBS_30_CNT_SOCIAL_CIRCLE     306490 non-null  float64
 35  DEF_30_CNT_SOCIAL_CIRCLE     306490 non-null  float64
 36  OBS_60_CNT_SOCIAL_CIRCLE     306490 non-null  float64
 37  DEF_60_CNT_SOCIAL_CIRCLE     306490 non-null  float64
 38  DAYS_LAST_PHONE_CHANGE       307510 non-null  float64
 39  FLAG_DOCUMENT_3              307511 non-null  int64
 40  AMT_REQ_CREDIT_BUREAU_HOUR   265992 non-null  float64
 41  AMT_REQ_CREDIT_BUREAU_DAY    265992 non-null  float64
 42  AMT_REQ_CREDIT_BUREAU_WEEK   265992 non-null  float64
 43  AMT_REQ_CREDIT_BUREAU_MON    265992 non-null  float64
 44  AMT_REQ_CREDIT_BUREAU_QRT    265992 non-null  float64
 45  AMT_REQ_CREDIT_BUREAU_YEAR   265992 non-null  float64
 46  INCOME_RANGE                 307279 non-null  category
 47  LOAN_AMT                     307511 non-null  category
 48  AGE                          307511 non-null  int64
 49  AGE_GROUP                    307511 non-null  category
```

```
 50  YEARS_EMPLOYED              307511 non-null  int64
 51  WORKING_YEARS               224233 non-null  category
dtypes: category(4), float64(18), int64(18), object(12)
memory usage: 113.8+ MB
```

In [59]:

```
#converting the datatype to categorical
categorical_columns=['NAME_CONTRACT_TYPE','CODE_GENDER','NAME_TYPE_SUITE','NAME_INCOME_TYPE
 'REG_CITY_NOT_LIVE_CITY','REG_CITY_NOT_WORK_CITY','REG_REGION_NOT_WORK_REGION','LIVE_REGIO
for col in categorical_columns:
    application[col]=pd.Categorical(application[col])
```

In [60]:

```
application.info(verbose=True)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Data columns (total 52 columns):
 #   Column                       Non-Null Count   Dtype
---  ------                       --------------   -----
 0   SK_ID_CURR                   307511 non-null  int64
 1   TARGET                       307511 non-null  int64
 2   NAME_CONTRACT_TYPE           307511 non-null  category
 3   CODE_GENDER                  307511 non-null  category
 4   FLAG_OWN_CAR                 307511 non-null  category
 5   FLAG_OWN_REALTY              307511 non-null  category
 6   CNT_CHILDREN                 307511 non-null  int64
 7   AMT_INCOME_TOTAL             307511 non-null  float64
 8   AMT_CREDIT                   307511 non-null  float64
 9   AMT_ANNUITY                  307499 non-null  float64
 10  AMT_GOODS_PRICE              307233 non-null  float64
 11  NAME_TYPE_SUITE              306219 non-null  category
 12  NAME_INCOME_TYPE             307511 non-null  category
 13  NAME_EDUCATION_TYPE          307511 non-null  category
 14  NAME_FAMILY_STATUS           307511 non-null  category
 15  NAME_HOUSING_TYPE            307511 non-null  category
 16  REGION_POPULATION_RELATIVE   307511 non-null  float64
 17  DAYS_BIRTH                   307511 non-null  int64
 18  DAYS_EMPLOYED                307511 non-null  int64
 19  DAYS_REGISTRATION            307511 non-null  float64
 20  DAYS_ID_PUBLISH              307511 non-null  int64
 21  OCCUPATION_TYPE              211120 non-null  category
 22  CNT_FAM_MEMBERS              307509 non-null  float64
 23  REGION_RATING_CLIENT         307511 non-null  category
 24  REGION_RATING_CLIENT_W_CITY  307511 non-null  category
 25  WEEKDAY_APPR_PROCESS_START   307511 non-null  category
 26  HOUR_APPR_PROCESS_START      307511 non-null  int64
 27  REG_REGION_NOT_LIVE_REGION   307511 non-null  int64
 28  REG_REGION_NOT_WORK_REGION   307511 non-null  category
 29  LIVE_REGION_NOT_WORK_REGION  307511 non-null  category
 30  REG_CITY_NOT_LIVE_CITY       307511 non-null  category
 31  REG_CITY_NOT_WORK_CITY       307511 non-null  category
 32  LIVE_CITY_NOT_WORK_CITY      307511 non-null  category
 33  ORGANIZATION_TYPE            307511 non-null  category
 34  OBS_30_CNT_SOCIAL_CIRCLE     306490 non-null  float64
 35  DEF_30_CNT_SOCIAL_CIRCLE     306490 non-null  float64
 36  OBS_60_CNT_SOCIAL_CIRCLE     306490 non-null  float64
 37  DEF_60_CNT_SOCIAL_CIRCLE     306490 non-null  float64
 38  DAYS_LAST_PHONE_CHANGE       307510 non-null  float64
 39  FLAG_DOCUMENT_3              307511 non-null  int64
 40  AMT_REQ_CREDIT_BUREAU_HOUR   265992 non-null  float64
 41  AMT_REQ_CREDIT_BUREAU_DAY    265992 non-null  float64
 42  AMT_REQ_CREDIT_BUREAU_WEEK   265992 non-null  float64
 43  AMT_REQ_CREDIT_BUREAU_MON    265992 non-null  float64
 44  AMT_REQ_CREDIT_BUREAU_QRT    265992 non-null  float64
 45  AMT_REQ_CREDIT_BUREAU_YEAR   265992 non-null  float64
 46  INCOME_RANGE                 307279 non-null  category
 47  LOAN_AMT                     307511 non-null  category
 48  AGE                          307511 non-null  int64
 49  AGE_GROUP                    307511 non-null  category
 50  YEARS_EMPLOYED               307511 non-null  int64
```

```
 51  WORKING_YEARS                224233 non-null  category
dtypes: category(23), float64(18), int64(11)
memory usage: 74.8 MB
```

In [61]:

```
#converting negative days to positive for previous dataset
#abs() to convert into whole number
previous['DAYS_DECISION']=abs(previous['DAYS_DECISION'])
```

In [62]:

```
previous.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1670214 entries, 0 to 1670213
Data columns (total 22 columns):
 #   Column                   Non-Null Count    Dtype
---  ------                   --------------    -----
 0   SK_ID_PREV               1670214 non-null  int64
 1   SK_ID_CURR               1670214 non-null  int64
 2   NAME_CONTRACT_TYPE       1670214 non-null  object
 3   AMT_ANNUITY              1297979 non-null  float64
 4   AMT_APPLICATION          1670214 non-null  float64
 5   AMT_CREDIT               1670213 non-null  float64
 6   AMT_GOODS_PRICE          1284699 non-null  float64
 7   NAME_CASH_LOAN_PURPOSE   1670214 non-null  object
 8   NAME_CONTRACT_STATUS     1670214 non-null  object
 9   DAYS_DECISION            1670214 non-null  int64
 10  NAME_PAYMENT_TYPE        1670214 non-null  object
 11  CODE_REJECT_REASON       1670214 non-null  object
 12  NAME_CLIENT_TYPE         1670214 non-null  object
 13  NAME_GOODS_CATEGORY      1670214 non-null  object
 14  NAME_PORTFOLIO           1670214 non-null  object
 15  NAME_PRODUCT_TYPE        1670214 non-null  object
 16  CHANNEL_TYPE             1670214 non-null  object
 17  SELLERPLACE_AREA         1670214 non-null  int64
 18  NAME_SELLER_INDUSTRY     1670214 non-null  object
 19  CNT_PAYMENT              1297984 non-null  float64
 20  NAME_YIELD_GROUP         1670214 non-null  object
 21  PRODUCT_COMBINATION      1669868 non-null  object
dtypes: float64(5), int64(4), object(13)
memory usage: 280.3+ MB
```

In [63]:

```
bins=[0,400,800,1200,1600,2000,2400,2800,3200]
previous['DAYS_DECISION_GROUP']=pd.cut(previous['DAYS_DECISION'],bins)
```

In [64]:

```
previous['DAYS_DECISION_GROUP'].value_counts(normalize=True)*100
```

Out[64]:

```
(0, 400]        37.574526
(400, 800]      22.900299
(800, 1200]     12.426012
(1200, 1600]     7.899646
(2400, 2800]     6.292188
(1600, 2000]     5.791174
(2000, 2400]     5.689750
(2800, 3200]     1.426404
Name: DAYS_DECISION_GROUP, dtype: float64
```

In [65]:

```
#Checking the number of unique values each column possess to identify categorical columns
previous.nunique().sort_values()
```

Out[65]:

```
NAME_PRODUCT_TYPE                3
NAME_CONTRACT_TYPE               4
NAME_CLIENT_TYPE                 4
NAME_PAYMENT_TYPE                4
NAME_CONTRACT_STATUS             4
NAME_YIELD_GROUP                 5
NAME_PORTFOLIO                   5
DAYS_DECISION_GROUP              8
CHANNEL_TYPE                     8
CODE_REJECT_REASON               9
NAME_SELLER_INDUSTRY            11
PRODUCT_COMBINATION             17
NAME_CASH_LOAN_PURPOSE          25
NAME_GOODS_CATEGORY             28
CNT_PAYMENT                     49
SELLERPLACE_AREA              2097
DAYS_DECISION                 2922
AMT_CREDIT                   86803
AMT_GOODS_PRICE              93885
AMT_APPLICATION              93885
SK_ID_CURR                  338857
AMT_ANNUITY                 357959
SK_ID_PREV                 1670214
dtype: int64
```

In [66]:

```
previous.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1670214 entries, 0 to 1670213
Data columns (total 23 columns):
 #   Column                       Non-Null Count    Dtype
---  ------                       --------------    -----
 0   SK_ID_PREV                   1670214 non-null  int64
 1   SK_ID_CURR                   1670214 non-null  int64
 2   NAME_CONTRACT_TYPE           1670214 non-null  object
 3   AMT_ANNUITY                  1297979 non-null  float64
 4   AMT_APPLICATION              1670214 non-null  float64
 5   AMT_CREDIT                   1670213 non-null  float64
 6   AMT_GOODS_PRICE              1284699 non-null  float64
 7   NAME_CASH_LOAN_PURPOSE       1670214 non-null  object
 8   NAME_CONTRACT_STATUS         1670214 non-null  object
 9   DAYS_DECISION                1670214 non-null  int64
 10  NAME_PAYMENT_TYPE            1670214 non-null  object
 11  CODE_REJECT_REASON           1670214 non-null  object
 12  NAME_CLIENT_TYPE             1670214 non-null  object
 13  NAME_GOODS_CATEGORY          1670214 non-null  object
 14  NAME_PORTFOLIO               1670214 non-null  object
 15  NAME_PRODUCT_TYPE            1670214 non-null  object
 16  CHANNEL_TYPE                 1670214 non-null  object
 17  SELLERPLACE_AREA             1670214 non-null  int64
 18  NAME_SELLER_INDUSTRY         1670214 non-null  object
 19  CNT_PAYMENT                  1297984 non-null  float64
 20  NAME_YIELD_GROUP             1670214 non-null  object
 21  PRODUCT_COMBINATION          1669868 non-null  object
 22  DAYS_DECISION_GROUP          1670214 non-null  category
dtypes: category(1), float64(5), int64(4), object(13)
memory usage: 281.9+ MB
```

In [67]:

```
previous_Cat=  ['NAME_CASH_LOAN_PURPOSE','NAME_CONTRACT_STATUS','NAME_PAYMENT_TYPE','CODE_R
                'NAME_GOODS_CATEGORY','NAME_PORTFOLIO','NAME_PRODUCT_TYPE','CHANNEL_TYPE','
                'PRODUCT_COMBINATION','NAME_CONTRACT_TYPE','DAYS_DECISION_GROUP']
for col in previous_Cat:
    previous[col]=pd.Categorical(previous[col])
```

In [68]:

```
previous.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1670214 entries, 0 to 1670213
Data columns (total 23 columns):
 #   Column                   Non-Null Count    Dtype   
---  ------                   --------------    -----   
 0   SK_ID_PREV               1670214 non-null  int64   
 1   SK_ID_CURR               1670214 non-null  int64   
 2   NAME_CONTRACT_TYPE       1670214 non-null  category
 3   AMT_ANNUITY              1297979 non-null  float64 
 4   AMT_APPLICATION          1670214 non-null  float64 
 5   AMT_CREDIT               1670213 non-null  float64 
 6   AMT_GOODS_PRICE          1284699 non-null  float64 
 7   NAME_CASH_LOAN_PURPOSE   1670214 non-null  category
 8   NAME_CONTRACT_STATUS     1670214 non-null  category
 9   DAYS_DECISION            1670214 non-null  int64   
 10  NAME_PAYMENT_TYPE        1670214 non-null  category
 11  CODE_REJECT_REASON       1670214 non-null  category
 12  NAME_CLIENT_TYPE         1670214 non-null  category
 13  NAME_GOODS_CATEGORY      1670214 non-null  category
 14  NAME_PORTFOLIO           1670214 non-null  category
 15  NAME_PRODUCT_TYPE        1670214 non-null  category
 16  CHANNEL_TYPE             1670214 non-null  category
 17  SELLERPLACE_AREA         1670214 non-null  int64   
 18  NAME_SELLER_INDUSTRY     1670214 non-null  category
 19  CNT_PAYMENT              1297984 non-null  float64 
 20  NAME_YIELD_GROUP         1670214 non-null  category
 21  PRODUCT_COMBINATION      1669868 non-null  category
 22  DAYS_DECISION_GROUP      1670214 non-null  category
dtypes: category(14), float64(5), int64(4)
memory usage: 137.0 MB
```

In [69]:

```
#checking null value percentage in each column
round(application.isnull().sum()/application.shape[0]*100.00,2)
```

Out[69]:

```
SK_ID_CURR                      0.00
TARGET                          0.00
NAME_CONTRACT_TYPE              0.00
CODE_GENDER                     0.00
FLAG_OWN_CAR                    0.00
FLAG_OWN_REALTY                 0.00
CNT_CHILDREN                    0.00
AMT_INCOME_TOTAL                0.00
AMT_CREDIT                      0.00
AMT_ANNUITY                     0.00
AMT_GOODS_PRICE                 0.09
NAME_TYPE_SUITE                 0.42
NAME_INCOME_TYPE                0.00
NAME_EDUCATION_TYPE             0.00
NAME_FAMILY_STATUS              0.00
NAME_HOUSING_TYPE               0.00
REGION_POPULATION_RELATIVE      0.00
DAYS_BIRTH                      0.00
DAYS_EMPLOYED                   0.00
DAYS_REGISTRATION               0.00
DAYS_ID_PUBLISH                 0.00
OCCUPATION_TYPE                31.35
CNT_FAM_MEMBERS                 0.00
REGION_RATING_CLIENT            0.00
REGION_RATING_CLIENT_W_CITY     0.00
WEEKDAY_APPR_PROCESS_START      0.00
HOUR_APPR_PROCESS_START         0.00
REG_REGION_NOT_LIVE_REGION      0.00
REG_REGION_NOT_WORK_REGION      0.00
LIVE_REGION_NOT_WORK_REGION     0.00
REG_CITY_NOT_LIVE_CITY          0.00
REG_CITY_NOT_WORK_CITY          0.00
LIVE_CITY_NOT_WORK_CITY         0.00
ORGANIZATION_TYPE               0.00
OBS_30_CNT_SOCIAL_CIRCLE        0.33
DEF_30_CNT_SOCIAL_CIRCLE        0.33
OBS_60_CNT_SOCIAL_CIRCLE        0.33
DEF_60_CNT_SOCIAL_CIRCLE        0.33
DAYS_LAST_PHONE_CHANGE          0.00
FLAG_DOCUMENT_3                 0.00
AMT_REQ_CREDIT_BUREAU_HOUR     13.50
AMT_REQ_CREDIT_BUREAU_DAY      13.50
AMT_REQ_CREDIT_BUREAU_WEEK     13.50
AMT_REQ_CREDIT_BUREAU_MON      13.50
AMT_REQ_CREDIT_BUREAU_QRT      13.50
AMT_REQ_CREDIT_BUREAU_YEAR     13.50
INCOME_RANGE                    0.08
LOAN_AMT                        0.00
AGE                             0.00
AGE_GROUP                       0.00
YEARS_EMPLOYED                  0.00
WORKING_YEARS                  27.08
dtype: float64
```

In [70]:

```
#NAME_TYPE_SUITE as only 0.42%null value
application['NAME_TYPE_SUITE'].describe()
```

Out[70]:

```
count              306219
unique                  7
top       Unaccompanied
freq               248526
Name: NAME_TYPE_SUITE, dtype: object
```

In [71]:

```
#imputing the null values with most occuring value by mode()
application['NAME_TYPE_SUITE'].fillna((application['NAME_TYPE_SUITE'].mode()[0]),inplace=Tr
```

In [72]:

```
#imputing the null values with new category
application['OCCUPATION_TYPE']=application['OCCUPATION_TYPE'].cat.add_categories('Unknown')
application['OCCUPATION_TYPE'].fillna('Unknown',inplace=True)
```

In [73]:

```
application[['AMT_REQ_CREDIT_BUREAU_HOUR','AMT_REQ_CREDIT_BUREAU_DAY','AMT_REQ_CREDIT_BUREA
               'AMT_REQ_CREDIT_BUREAU_QRT','AMT_REQ_CREDIT_BUREAU_YEAR']].describe()
```

Out[73]:

	AMT_REQ_CREDIT_BUREAU_HOUR	AMT_REQ_CREDIT_BUREAU_DAY	AMT_REQ_CREDIT_
count	265992.000000	265992.000000	
mean	0.006402	0.007000	
std	0.083849	0.110757	
min	0.000000	0.000000	
25%	0.000000	0.000000	
50%	0.000000	0.000000	
75%	0.000000	0.000000	
max	4.000000	9.000000	

In [74]:

```
#imputing with median as mean is in decimal
amount=['AMT_REQ_CREDIT_BUREAU_HOUR','AMT_REQ_CREDIT_BUREAU_DAY','AMT_REQ_CREDIT_BUREAU_WEE
               'AMT_REQ_CREDIT_BUREAU_QRT','AMT_REQ_CREDIT_BUREAU_YEAR']
for col in amount:
    application[col].fillna(application[col].median(),inplace=True)
```

In [75]:

```
round(application.isnull().sum()/application.shape[0]*100.0,2)
```

Out[75]:

```
SK_ID_CURR                     0.00
TARGET                         0.00
NAME_CONTRACT_TYPE             0.00
CODE_GENDER                    0.00
FLAG_OWN_CAR                   0.00
FLAG_OWN_REALTY                0.00
CNT_CHILDREN                   0.00
AMT_INCOME_TOTAL               0.00
AMT_CREDIT                     0.00
AMT_ANNUITY                    0.00
AMT_GOODS_PRICE                0.09
NAME_TYPE_SUITE                0.00
NAME_INCOME_TYPE               0.00
NAME_EDUCATION_TYPE            0.00
NAME_FAMILY_STATUS             0.00
NAME_HOUSING_TYPE              0.00
REGION_POPULATION_RELATIVE     0.00
DAYS_BIRTH                     0.00
DAYS_EMPLOYED                  0.00
DAYS_REGISTRATION              0.00
DAYS_ID_PUBLISH                0.00
OCCUPATION_TYPE                0.00
CNT_FAM_MEMBERS                0.00
REGION_RATING_CLIENT           0.00
REGION_RATING_CLIENT_W_CITY    0.00
WEEKDAY_APPR_PROCESS_START     0.00
HOUR_APPR_PROCESS_START        0.00
REG_REGION_NOT_LIVE_REGION     0.00
REG_REGION_NOT_WORK_REGION     0.00
LIVE_REGION_NOT_WORK_REGION    0.00
REG_CITY_NOT_LIVE_CITY         0.00
REG_CITY_NOT_WORK_CITY         0.00
LIVE_CITY_NOT_WORK_CITY        0.00
ORGANIZATION_TYPE              0.00
OBS_30_CNT_SOCIAL_CIRCLE       0.33
DEF_30_CNT_SOCIAL_CIRCLE       0.33
OBS_60_CNT_SOCIAL_CIRCLE       0.33
DEF_60_CNT_SOCIAL_CIRCLE       0.33
DAYS_LAST_PHONE_CHANGE         0.00
FLAG_DOCUMENT_3                0.00
AMT_REQ_CREDIT_BUREAU_HOUR     0.00
AMT_REQ_CREDIT_BUREAU_DAY      0.00
AMT_REQ_CREDIT_BUREAU_WEEK     0.00
AMT_REQ_CREDIT_BUREAU_MON      0.00
AMT_REQ_CREDIT_BUREAU_QRT      0.00
AMT_REQ_CREDIT_BUREAU_YEAR     0.00
INCOME_RANGE                   0.08
LOAN_AMT                       0.00
AGE                            0.00
AGE_GROUP                      0.00
YEARS_EMPLOYED                 0.00
WORKING_YEARS                 27.08
dtype: float64
```

In [76]:

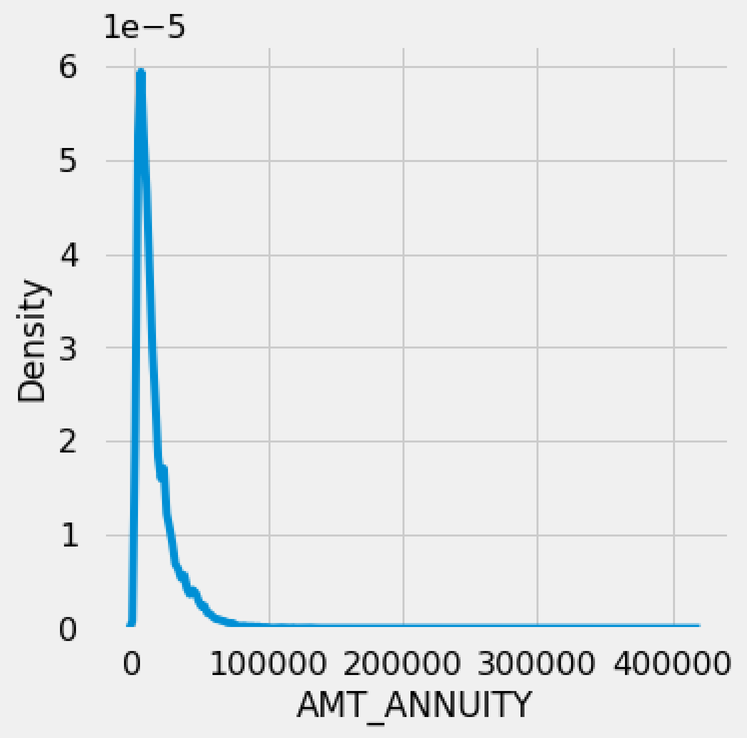
```
#percentage of null values in previous
round(previous.isnull().sum()/previous.shape[0]*100.00,3)
```

Out[76]:

```
SK_ID_PREV                 0.000
SK_ID_CURR                 0.000
NAME_CONTRACT_TYPE         0.000
AMT_ANNUITY               22.287
AMT_APPLICATION            0.000
AMT_CREDIT                 0.000
AMT_GOODS_PRICE           23.082
NAME_CASH_LOAN_PURPOSE     0.000
NAME_CONTRACT_STATUS       0.000
DAYS_DECISION              0.000
NAME_PAYMENT_TYPE          0.000
CODE_REJECT_REASON         0.000
NAME_CLIENT_TYPE           0.000
NAME_GOODS_CATEGORY        0.000
NAME_PORTFOLIO             0.000
NAME_PRODUCT_TYPE          0.000
CHANNEL_TYPE               0.000
SELLERPLACE_AREA           0.000
NAME_SELLER_INDUSTRY       0.000
CNT_PAYMENT               22.286
NAME_YIELD_GROUP           0.000
PRODUCT_COMBINATION        0.021
DAYS_DECISION_GROUP        0.000
dtype: float64
```

In [77]:

```
#plotting the distribution of columns
plt.figure(figsize=(5,5))
ax=sns.kdeplot(previous['AMT_ANNUITY'])
```

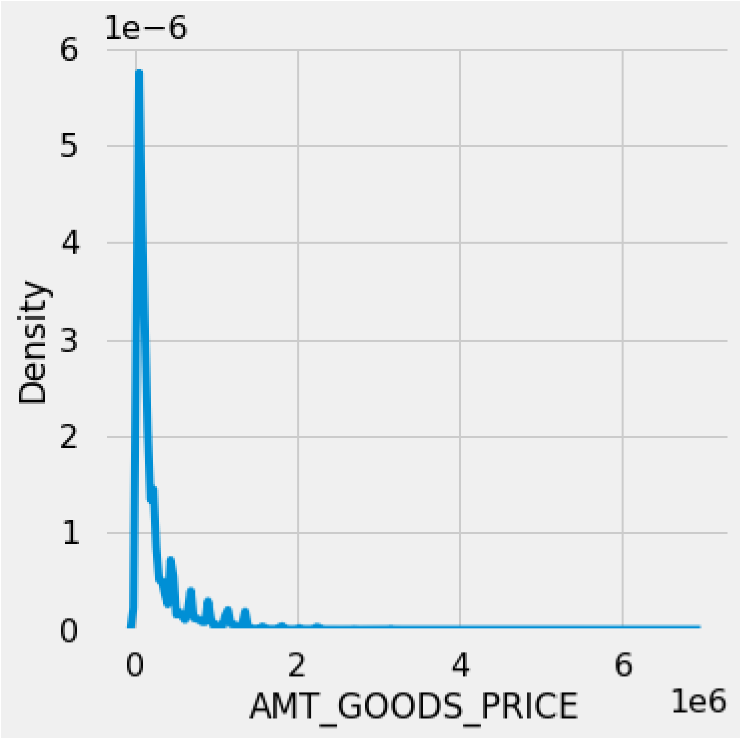


In [78]:

```
#imputing with median as there is an outlier and would impact the mean
previous['AMT_ANNUITY'].fillna(previous['AMT_ANNUITY'].median(),inplace=True)
```

In [79]:

```
#plotting the distribution of columns
plt.figure(figsize=(5,5))
bx=sns.kdeplot(previous['AMT_GOODS_PRICE'])
```

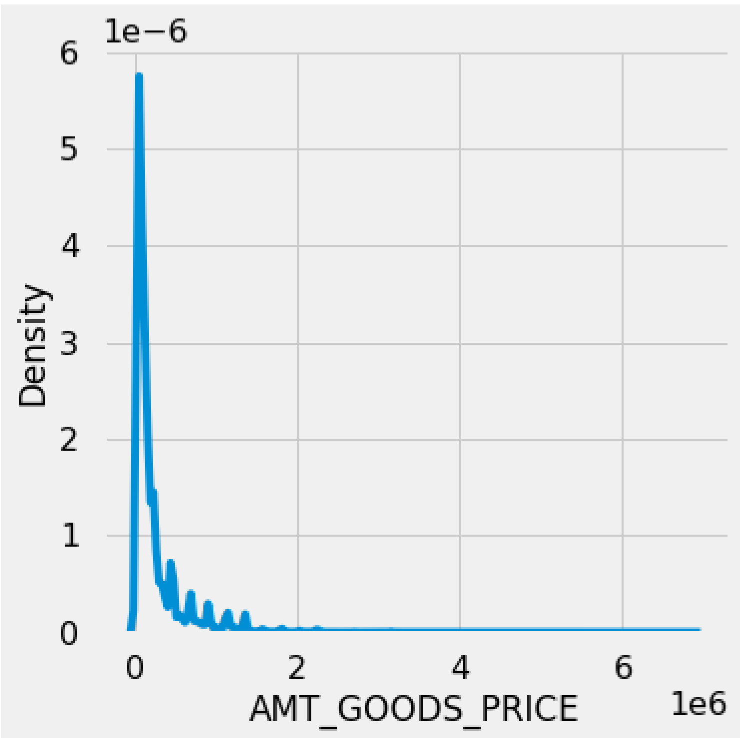


In [80]:

```
plt.figure(figsize=(5,5))
sns.kdeplot(previous['AMT_GOODS_PRICE'][pd.notnull(previous['AMT_GOODS_PRICE'])])
```

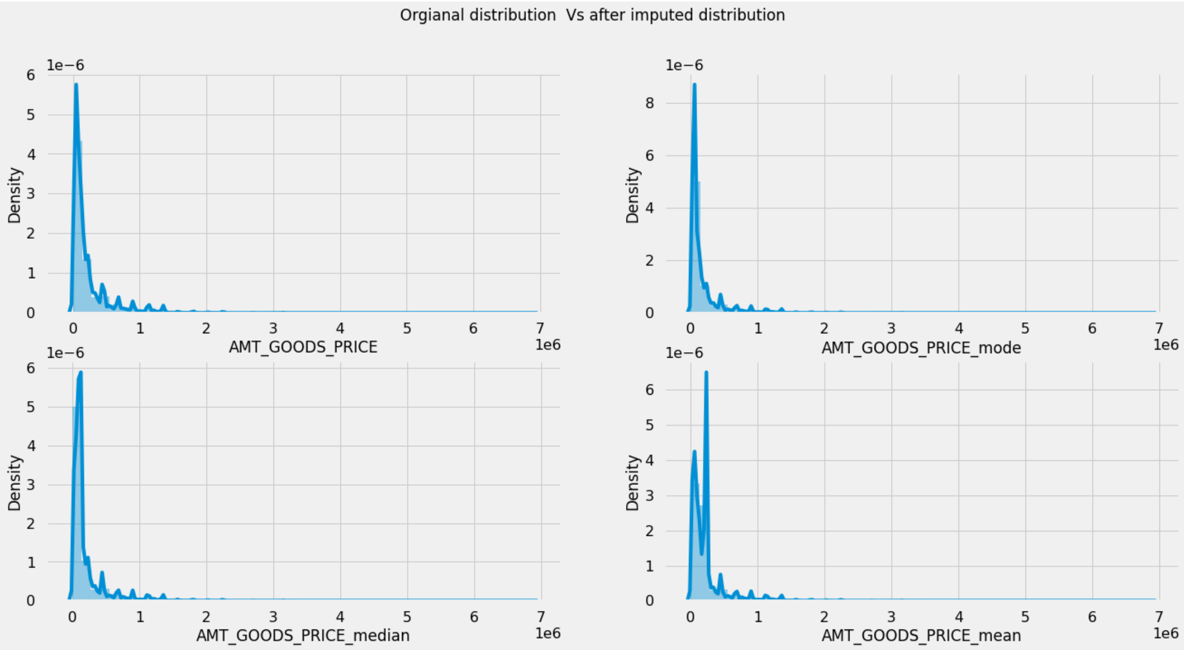
Out[80]:

<AxesSubplot:xlabel='AMT_GOODS_PRICE', ylabel='Density'>



In [81]:

```
#checking which method to impute as there are several peaks in the distribution
stats=pd.DataFrame()
stats['AMT_GOODS_PRICE_mode']=previous['AMT_GOODS_PRICE'].fillna(previous['AMT_GOODS_PRICE'
stats['AMT_GOODS_PRICE_median']=previous['AMT_GOODS_PRICE'].fillna(previous['AMT_GOODS_PRIC
stats['AMT_GOODS_PRICE_mean']=previous['AMT_GOODS_PRICE'].fillna(previous['AMT_GOODS_PRICE'
cols=['AMT_GOODS_PRICE_mode','AMT_GOODS_PRICE_median','AMT_GOODS_PRICE_mean']
plt.figure(figsize=(20,10))
plt.suptitle('Orgianal distribution  Vs after imputed distribution')
plt.subplot(221)
sns.distplot(previous['AMT_GOODS_PRICE'][pd.notnull(previous['AMT_GOODS_PRICE'])]);
for i in enumerate(cols):
    plt.subplot(2,2,i[0]+2)
    sns.distplot(stats[i[1]])
```



In [82]:

```
#imputing by mode as the orginal distribution is closer the imputed distribution
previous['AMT_GOODS_PRICE'].fillna(previous['AMT_GOODS_PRICE'].mode()[0],inplace=True)
```

In [83]:

```
#checking the relation for null values in CNT_PAYMENT with contract status
previous.loc[previous['CNT_PAYMENT'].isnull(),'NAME_CONTRACT_STATUS'].value_counts()
```

Out[83]:

```
Canceled        305805
Refused          40897
Unused offer     25524
Approved             4
Name: NAME_CONTRACT_STATUS, dtype: int64
```

In [84]:

```
#imputing by relevant information
previous['CNT_PAYMENT'].fillna(0,inplace=True)
```

In [85]:

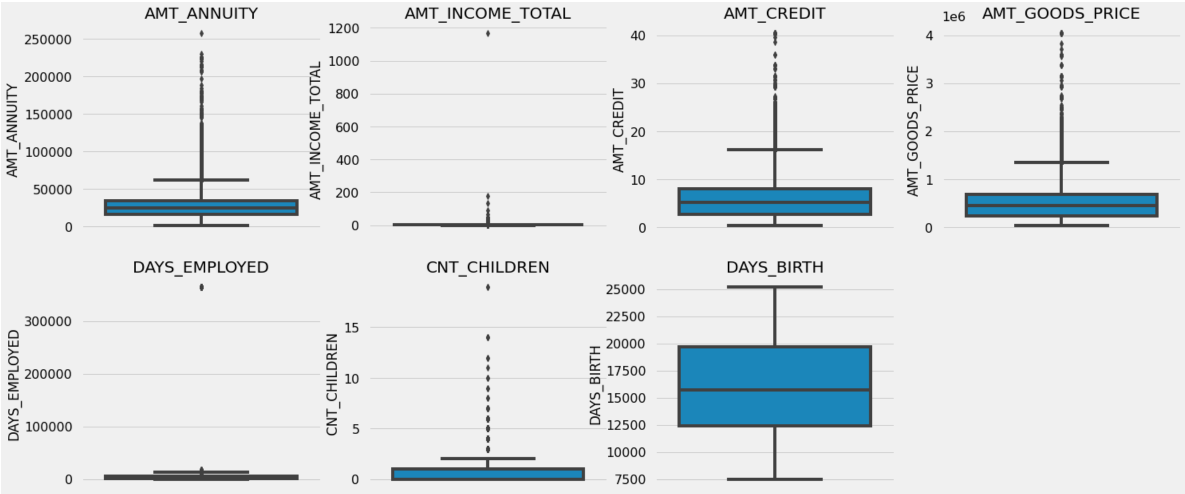
```
#checking the percentage of null values
round(previous.isnull().sum()/previous.shape[0]*100.0,3)
```

Out[85]:

```
SK_ID_PREV                 0.000
SK_ID_CURR                 0.000
NAME_CONTRACT_TYPE         0.000
AMT_ANNUITY                0.000
AMT_APPLICATION            0.000
AMT_CREDIT                 0.000
AMT_GOODS_PRICE            0.000
NAME_CASH_LOAN_PURPOSE     0.000
NAME_CONTRACT_STATUS       0.000
DAYS_DECISION              0.000
NAME_PAYMENT_TYPE          0.000
CODE_REJECT_REASON         0.000
NAME_CLIENT_TYPE           0.000
NAME_GOODS_CATEGORY        0.000
NAME_PORTFOLIO             0.000
NAME_PRODUCT_TYPE          0.000
CHANNEL_TYPE               0.000
SELLERPLACE_AREA           0.000
NAME_SELLER_INDUSTRY       0.000
CNT_PAYMENT                0.000
NAME_YIELD_GROUP           0.000
PRODUCT_COMBINATION        0.021
DAYS_DECISION_GROUP        0.000
dtype: float64
```

In [86]:

```
#checking for outliers
plt.figure(figsize=(22,10))
application_outliers=['AMT_ANNUITY','AMT_INCOME_TOTAL','AMT_CREDIT','AMT_GOODS_PRICE','DAYS
for i in enumerate(application_outliers):
    plt.subplot(2,4,i[0]+1)
    sns.boxplot(y=application[i[1]])
    plt.title(i[1])
```



In [87]:

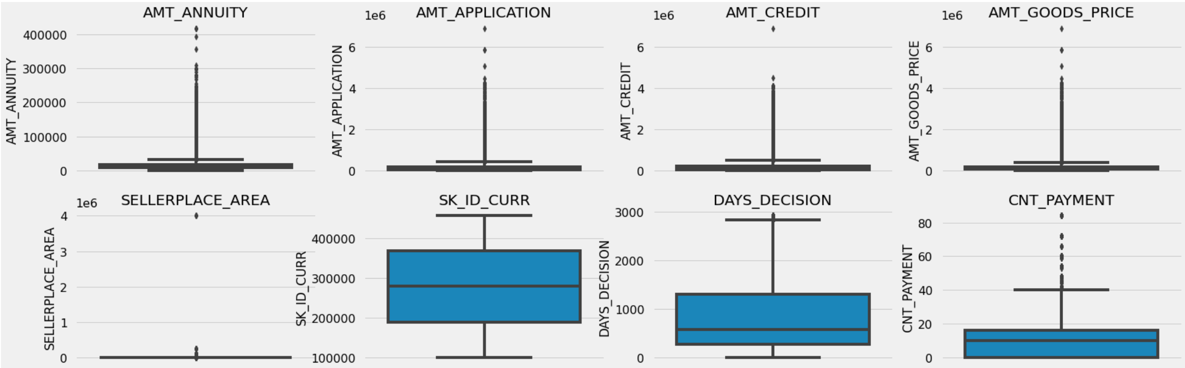
```
application[['AMT_ANNUITY', 'AMT_INCOME_TOTAL', 'AMT_CREDIT', 'AMT_GOODS_PRICE', 'DAYS_BIRT
```

Out[87]:

	AMT_ANNUITY	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_GOODS_PRICE	DAYS_BIRTH
count	307499.000000	307511.000000	307511.000000	3.072330e+05	307511.000000
mean	27108.573909	1.687979	5.990260	5.383962e+05	16036.995067
std	14493.737315	2.371231	4.024908	3.694465e+05	4363.988632
min	1615.500000	0.256500	0.450000	4.050000e+04	7489.000000
25%	16524.000000	1.125000	2.700000	2.385000e+05	12413.000000
50%	24903.000000	1.471500	5.135310	4.500000e+05	15750.000000
75%	34596.000000	2.025000	8.086500	6.795000e+05	19682.000000
max	258025.500000	1170.000000	40.500000	4.050000e+06	25229.000000

In [88]:

```
#to identify outliers in previous data set
plt.figure(figsize=(24,8))
from matplotlib import pyplot, pylab
import matplotlib.pyplot as plt
previous_outliers=['AMT_ANNUITY','AMT_APPLICATION','AMT_CREDIT','AMT_GOODS_PRICE','SELLERPL
for i in enumerate(previous_outliers):
    plt.subplot(2,4,i[0]+1)
    sns.boxplot(y=previous[i[1]])
    pylab.title(i[1])
```



In [89]:

```
previous[['AMT_ANNUITY', 'AMT_APPLICATION', 'AMT_CREDIT', 'AMT_GOODS_PRICE', 'SELLERPLACE_A
```

Out[89]:

	AMT_ANNUITY	AMT_APPLICATION	AMT_CREDIT	AMT_GOODS_PRICE	SELLERPLACE_A
count	1.670214e+06	1.670214e+06	1.670213e+06	1.670214e+06	1.670214
mean	1.490651e+04	1.752339e+05	1.961140e+05	1.856429e+05	3.139511
std	1.317751e+04	2.927798e+05	3.185746e+05	2.871413e+05	7.127443
min	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	-1.000000
25%	7.547096e+03	1.872000e+04	2.416050e+04	4.500000e+04	-1.000000
50%	1.125000e+04	7.104600e+04	8.054100e+04	7.105050e+04	3.000000
75%	1.682403e+04	1.803600e+05	2.164185e+05	1.804050e+05	8.200000
max	4.180581e+05	6.905160e+06	6.905160e+06	6.905160e+06	4.000000

In [90]:

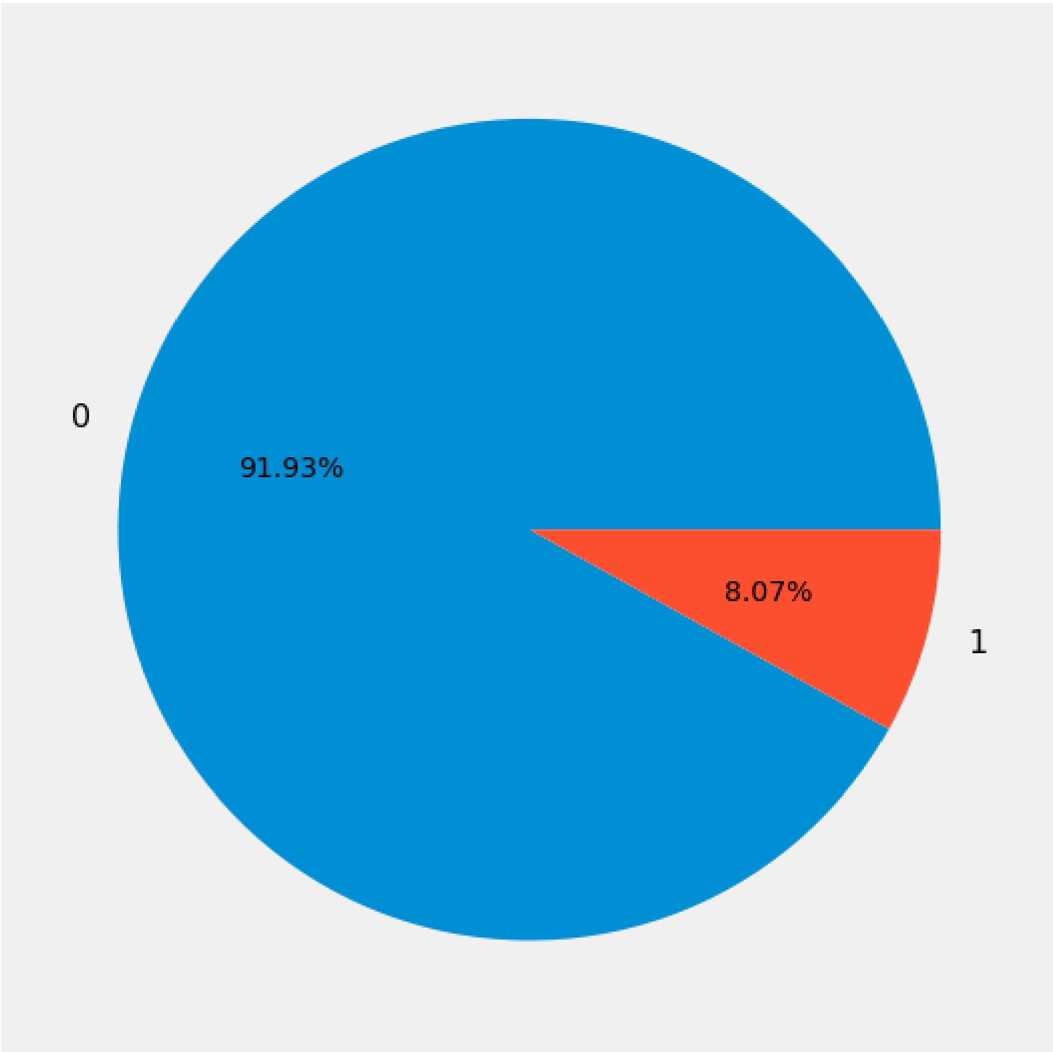
```
loan_repay_status=application["TARGET"].value_counts().reset_index()
```

In [91]:

```
freq_repay_status=loan_repay_status
freq_repay_status["index"] = freq_repay_status["index"].replace({1:"Defaulter",0:"Repayer"}

plt.pie(freq_repay_status.TARGET,labels=freq_repay_status.index,autopct='%.2f%%')
```

Out[91]:

```
([<matplotlib.patches.Wedge at 0x20826fae8b0>,
  <matplotlib.patches.Wedge at 0x20826fb0100>],
 [Text(-1.0648123216659293, 0.27599768047650985, '0'),
  Text(1.0648123152057372, -0.27599770540024077, '1')],
 [Text(-0.5808067209086887, 0.15054418935082356, '91.93%'),
  Text(0.5808067173849475, -0.15054420294558588, '8.07%')])
```

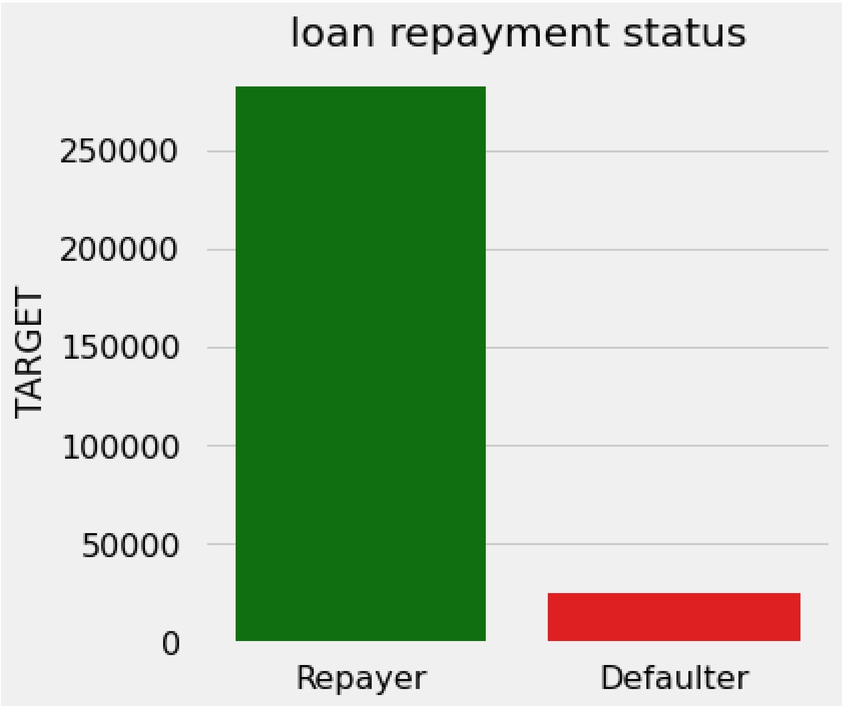


In [92]:

```
plt.figure(figsize=(5,5))
a=['Repayer','Defaulter']
sns.barplot(a,'TARGET',data=loan_repay_status,palette=['g','r'])
plt.title('loan repayment status')
```

Out[92]:

```
Text(0.5, 1.0, 'loan repayment status')
```



In [93]:

```
countof_0=loan_repay_status.iloc[0]['TARGET']
countof_1=loan_repay_status.iloc[1]['TARGET']
count_per_0=round(countof_0/(countof_0+countof_1)*100.00,2)
count_per_1=round(countof_1/(countof_0+countof_1)*100.00,2)
print('percentage of repayers and defaulters are: %.2f and %.2f'%(count_per_0,count_per_1))
print('ratios of repay stats with respective to repayer to defaulters is %.2f:1'%(countof_0
```

```
percentage of repayers and defaulters are: 91.93 and 8.07
ratios of repay stats with respective to repayer to defaulters is 11.39:1
```

In [94]:

```
# function for plotting countplots in univariate categorical analysis on application
# This function will create two subplots:
# 1. Count plot of categorical column w.r.t TARGET;
# 2. Percentage of defaulters within column
#feature- attribute
#ylog- to set the scale of y axis to log (Set the y-axis scale)
#label_rotation- to rotate the label of x axis to 90 degrees
#horizontal_layout- to get plots in two rows or columns
def univariate_categorical(feature,ylog=False,label_rotation=False,horizontal_layout=True):
    temp = application[feature].value_counts()
    df1 = pd.DataFrame({feature: temp.index,'Number of contracts': temp.values})

  # Calculate the percentage of target=1 per category value
    cat_perc = application[[feature, 'TARGET']].groupby([feature],as_index=False).mean()
    cat_perc["TARGET"] = cat_perc["TARGET"]*100
    cat_perc.sort_values(by='TARGET', ascending=False, inplace=True)

    if(horizontal_layout):
        fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(12,6))
    else:
        fig, (ax1, ax2) = plt.subplots(nrows=2, figsize=(20,24))

    # 1. Subplot 1: Count plot of categorical column
    # sns.set_palette("Set2")
    s = sns.countplot(ax=ax1, x = feature, data=application,hue ="TARGET",order=cat_perc[fe

    # Define common styling
    ax1.set_title(feature, fontdict={'fontsize' : 10, 'fontweight' : 3, 'color' : 'Blue'})
    ax1.legend(['Repayer','Defaulter'])

    # If the plot is not readable, use the log scale.
    if ylog:
        ax1.set_yscale('log')
        ax1.set_ylabel("Count (log)",fontdict={'fontsize' : 10, 'fontweight' : 3, 'color' :


    if(label_rotation):
        s.set_xticklabels(s.get_xticklabels(),rotation=90)

    # 2. Subplot 2: Percentage of defaulters within the categorical column
    s = sns.barplot(ax=ax2, x = feature, y='TARGET', order=cat_perc[feature], data=cat_perc

    if(label_rotation):
        s.set_xticklabels(s.get_xticklabels(),rotation=90)
    plt.ylabel('Percent of Defaulters [%]', fontsize=10)
    plt.tick_params(axis='both', which='major', labelsize=10)
    ax2.set_title(feature + " Defaulter %", fontdict={'fontsize' : 15, 'fontweight' : 5, 'c

    plt.show();
```

In [95]:

```
# function for plotting repetitive barplots in bivariate categorical analysis
def bivariate_bar(x,y,df,hue,figsize):
    plt.figure(figsize=figsize)
    sns.barplot(x=x,y=y,data=df, hue=hue, palette =['g','r'])
    plt.xlabel(x,fontdict={'fontsize' : 10, 'fontweight' : 3, 'color' : 'Blue'})
    plt.ylabel(y,fontdict={'fontsize' : 10, 'fontweight' : 3, 'color' : 'Blue'})
    plt.title(col, fontdict={'fontsize' : 15, 'fontweight' : 5, 'color' : 'Blue'})
    plt.xticks(rotation=90, ha='right')
    plt.legend(labels = ['Repayer','Defaulter'])
    plt.show()
```

In [96]:

```
# function for plotting repetitive rel plots in bivaritae numerical analysis on application
#.relplot() shows the relationships between two variables
def bivariate_rel(x,y,data, hue, kind, palette, legend,figsize):
    plt.figure(figsize=figsize)
    sns.relplot(x=x, y=y, data=application, hue="TARGET",kind=kind,palette = ['g','r'],lege
    plt.legend(['Repayer','Defaulter'])
    plt.xticks(rotation=90, ha='right')
    plt.show()
```

In [97]:

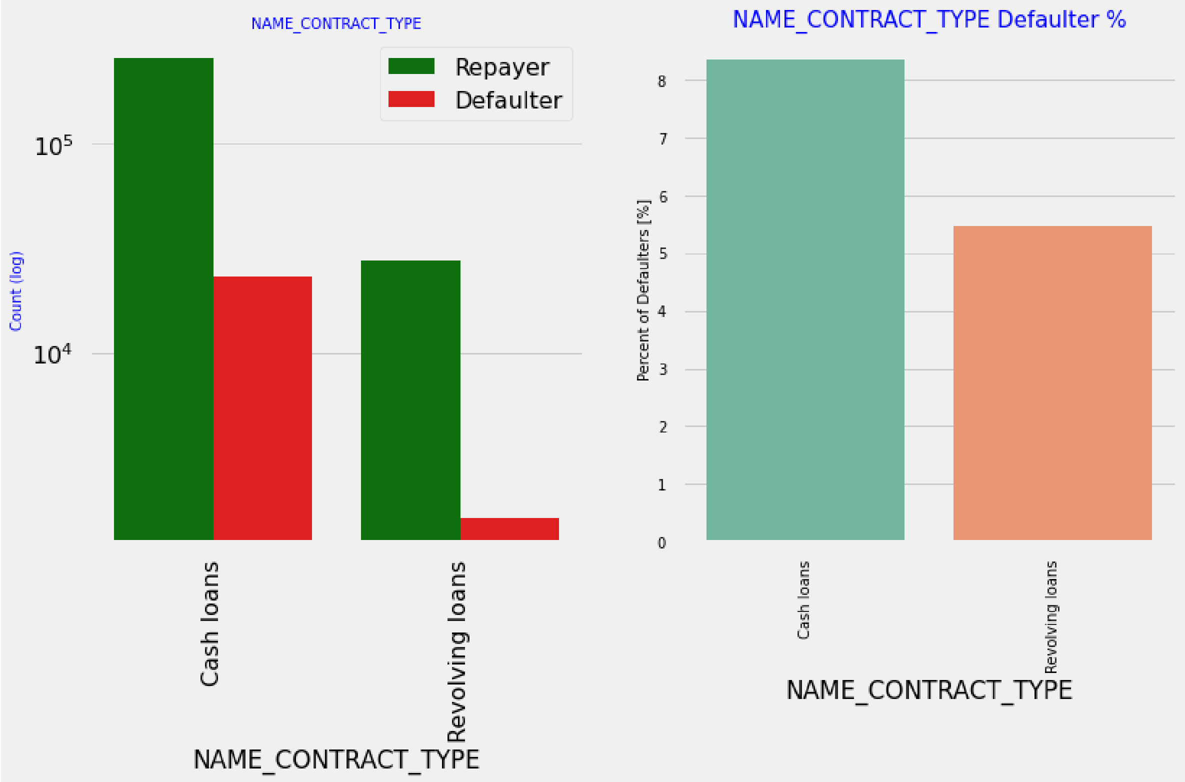
```
#function for plotting repetitive countplots in univariate categorical analysis on the merg
def univariate_merged(col,df,hue,palette,ylog,figsize):
    plt.figure(figsize=figsize)
    ax=sns.countplot(x=col, data=df,hue= hue,palette= palette,order=df[col].value_counts().
    if ylog:
        plt.yscale('log')
        plt.ylabel("Count (log)",fontdict={'fontsize' : 10, 'fontweight' : 3, 'color' : 'Bl
    else:
        plt.ylabel("Count",fontdict={'fontsize' : 10, 'fontweight' : 3, 'color' : 'Blue'})
    plt.title(col , fontdict={'fontsize' : 15, 'fontweight' : 5, 'color' : 'Blue'})
    plt.legend(loc = "upper right")
    plt.xticks(rotation=90, ha='right')

    plt.show()
```

In [98]:

```
# Function to plot point plots on merged dataframe
#Show point estimates and confidence intervals using scatter plot glyphs.
def merged_pointplot(x,y):
    plt.figure(figsize=(8,4))
    sns.pointplot(x=x,y=y, hue="TARGET", data=loan_application, palette =['g','r'])
```

In [99]:

```
# Checking the contract type based on Loan repayment status
univariate_categorical('NAME_CONTRACT_TYPE',True,True)
```



In [100]:

```
cat_perc = application[['NAME_CONTRACT_TYPE', 'TARGET']].groupby(['NAME_CONTRACT_TYPE'],as_
cat_perc
```

Out[100]:

	NAME_CONTRACT_TYPE	TARGET
0	Cash loans	0.083459
1	Revolving loans	0.054783

In [101]:

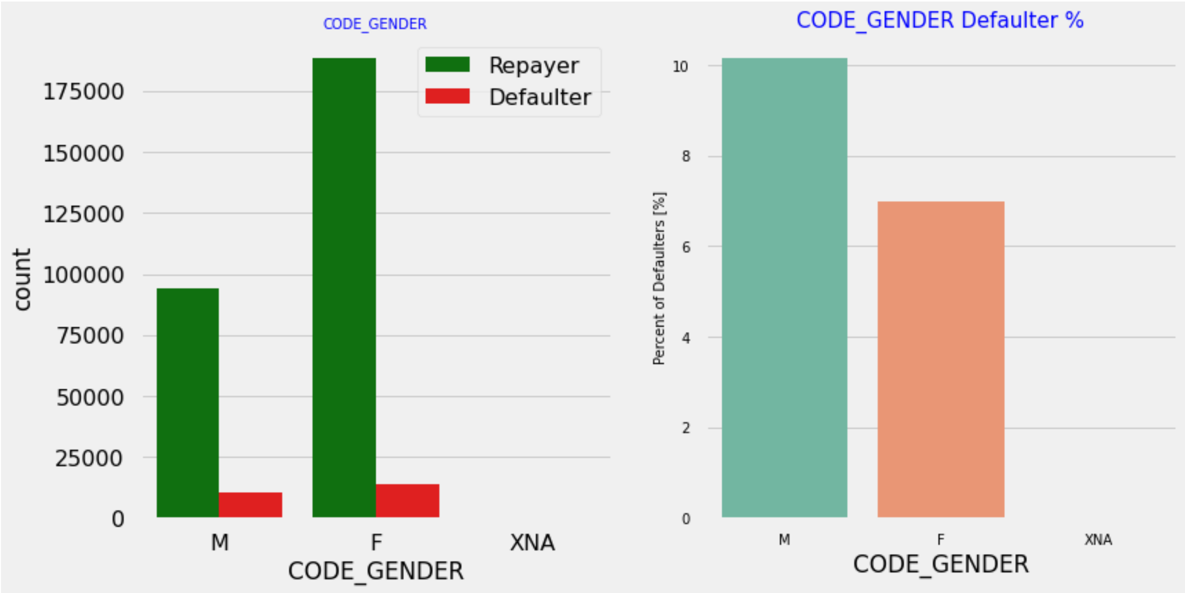
```
cat_perc = application[['NAME_CONTRACT_TYPE', 'TARGET']].groupby(['NAME_CONTRACT_TYPE'],as_
cat_perc
```

Out[101]:

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x0000020828C99490>
```

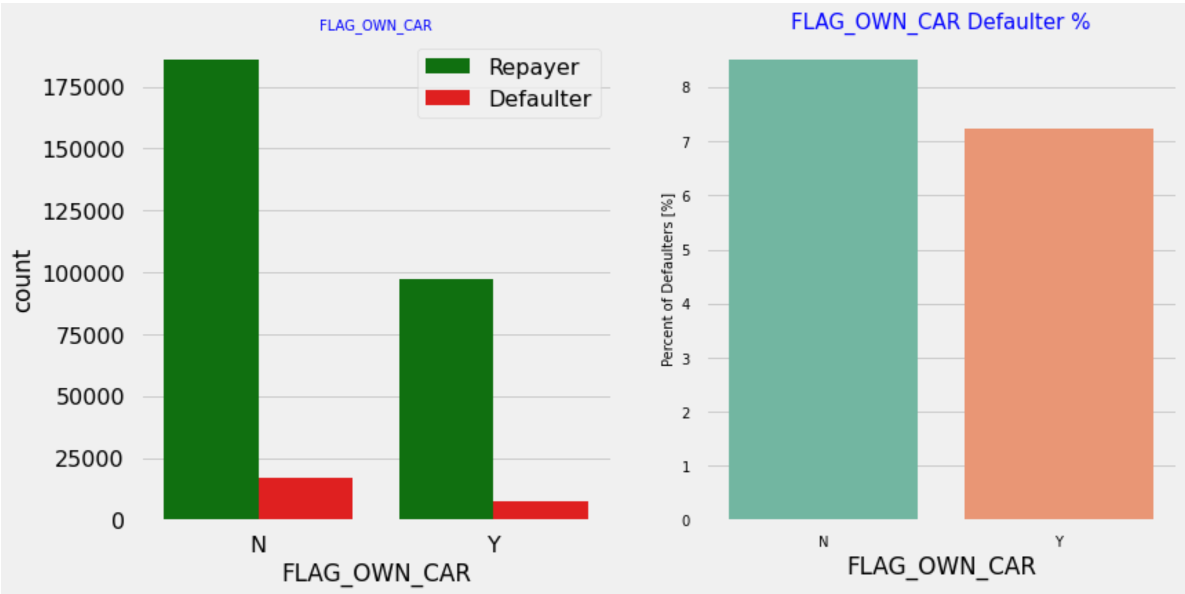
In [102]:

```
#repaying status based on gender
univariate_categorical('CODE_GENDER')
```



In [103]:

```
#checking whether car owned is related to loan repayment status or not
univariate_categorical('FLAG_OWN_CAR')
```

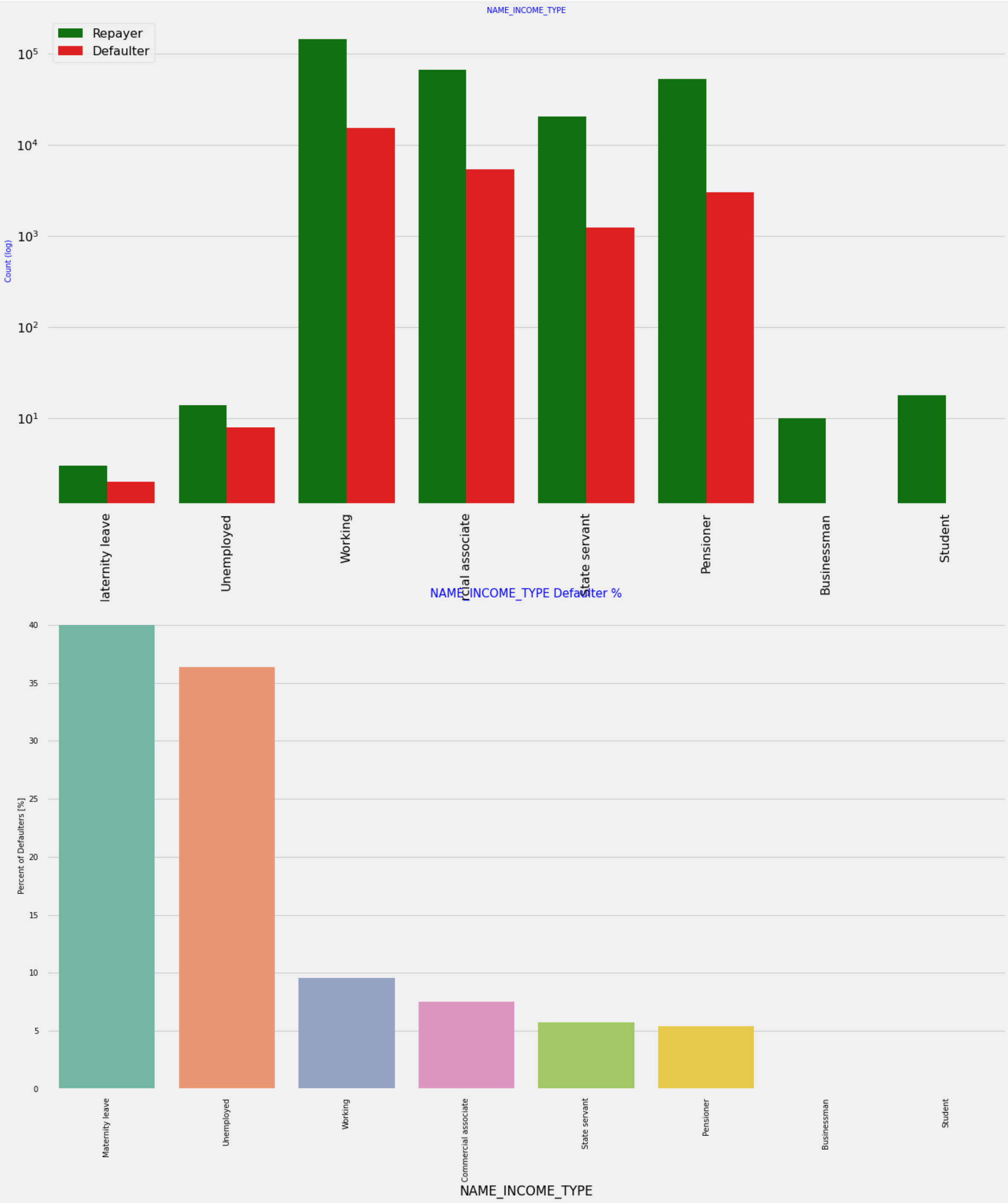


In [104]:

```
#checking whether a client owns a house or flat
univariate_categorical('FLAG_OWN_REALTY')
```

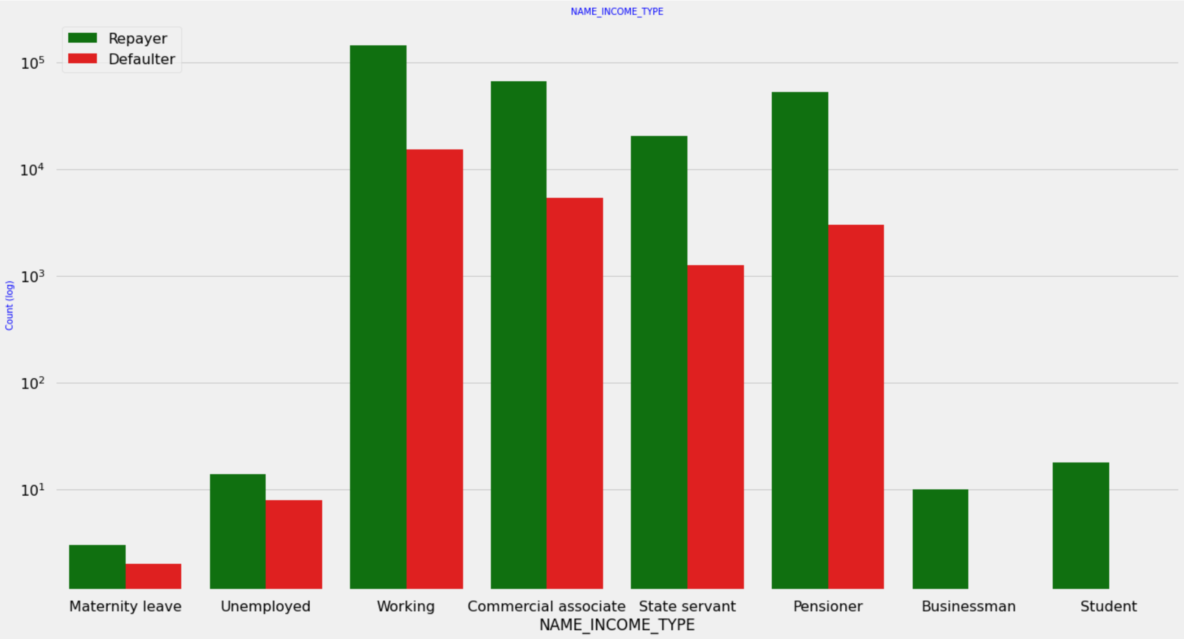
In [105]:

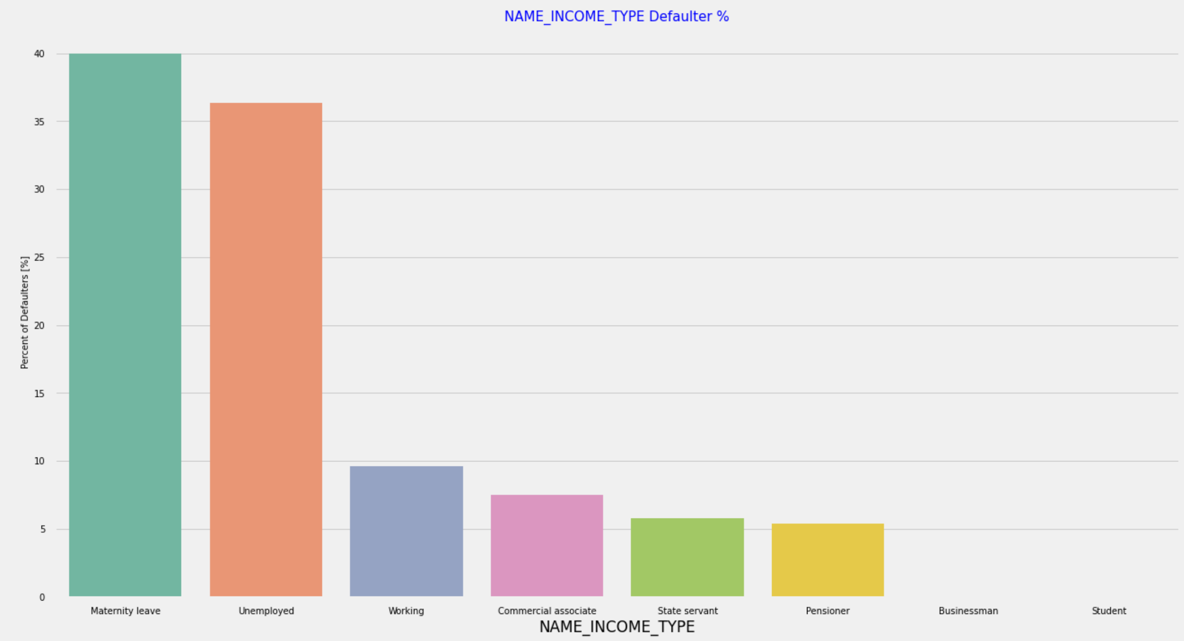
```
#analysing client income type with loan repayment status
univariate_categorical("NAME_INCOME_TYPE",True,True,False)
```



In [106]:

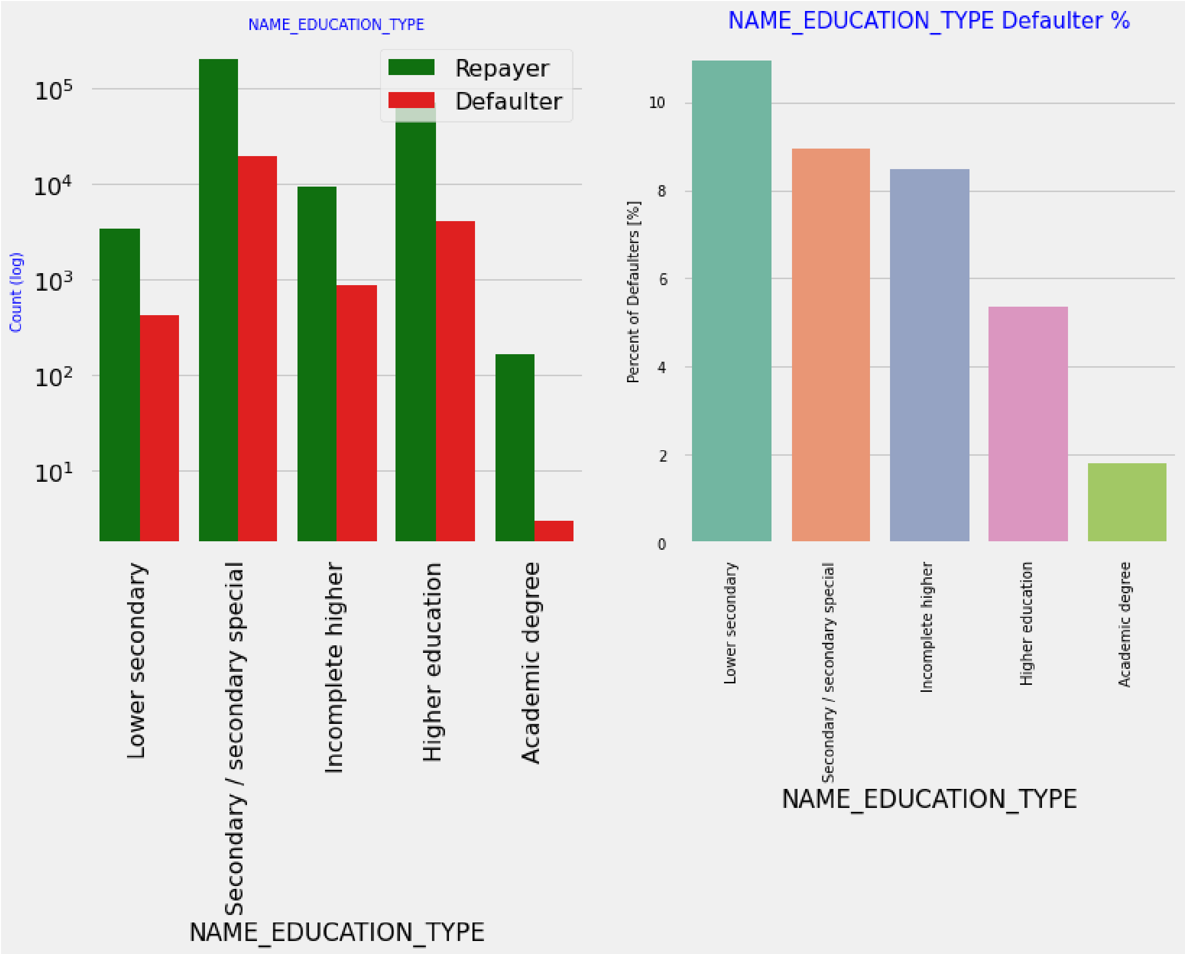
```
univariate_categorical("NAME_INCOME_TYPE",True,False,False)
```





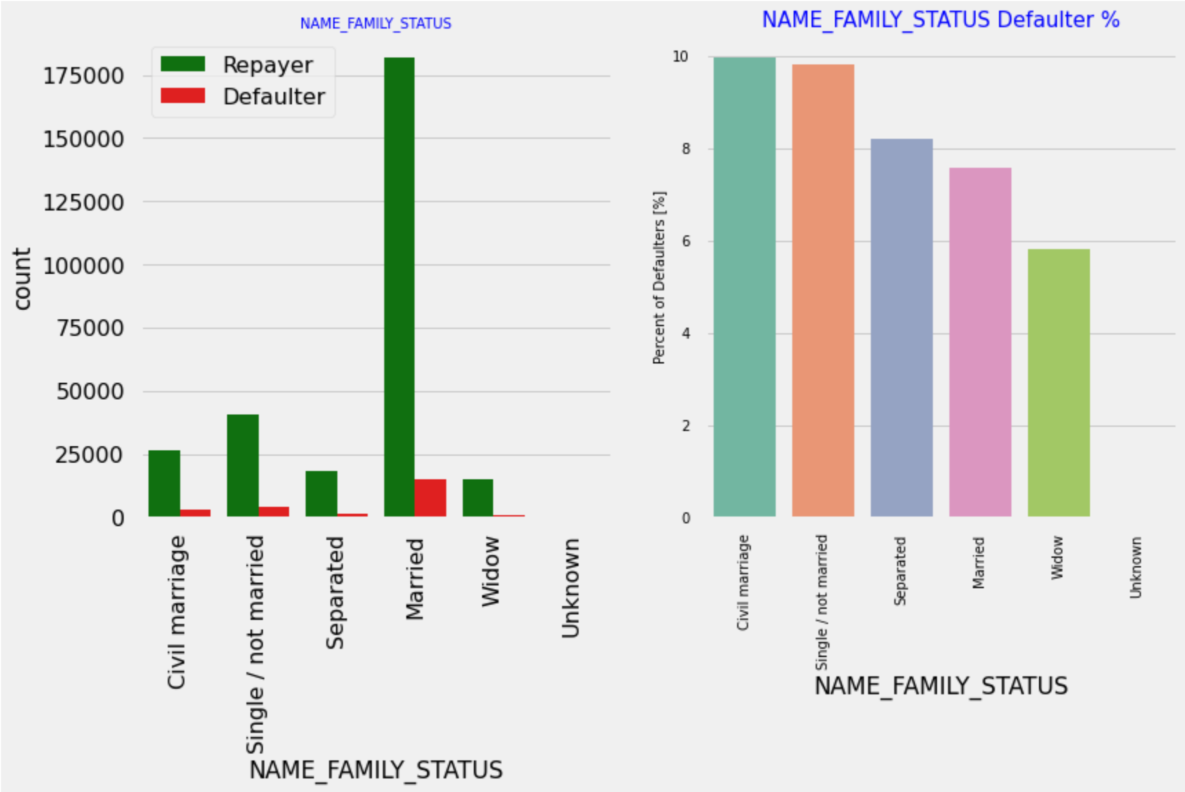
In [107]:

```
#analysing client education to loan repayment status
univariate_categorical("NAME_EDUCATION_TYPE",True,True,True)
```



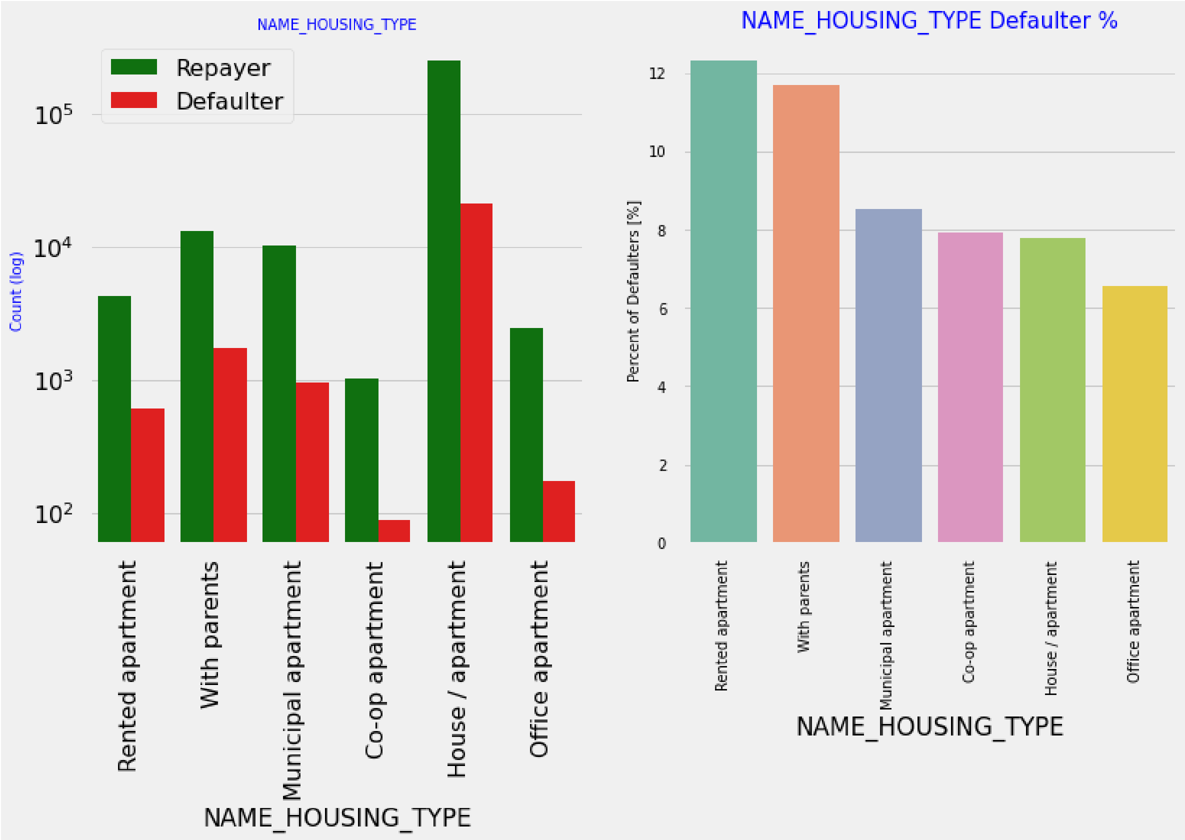
In [108]:

```
#analysing family status based on loan replyment status
univariate_categorical("NAME_FAMILY_STATUS",False,True,True)
```



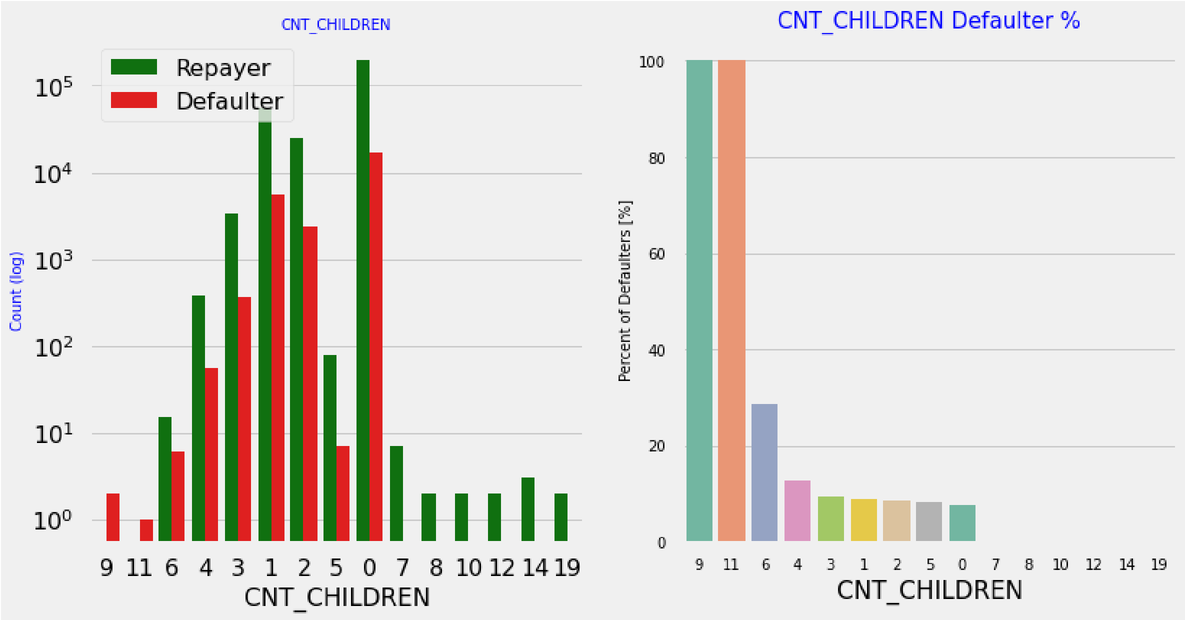
In [109]:

```
#analysing the housing situation of the client(renting,living with parents...) with the Loa
univariate_categorical("NAME_HOUSING_TYPE",True,True,True)
```



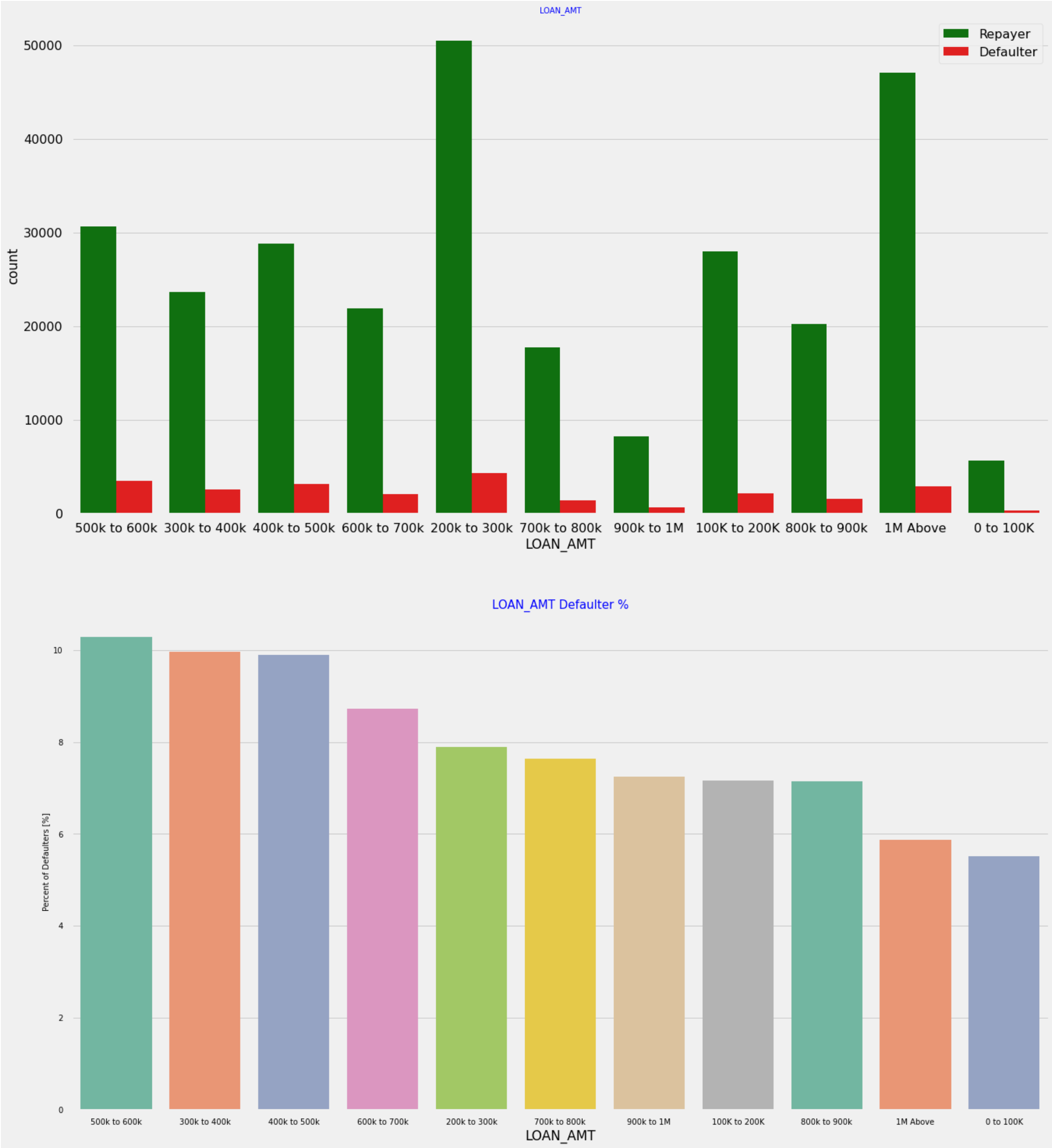
In [110]:

```
# Analyzing Number of children based on loan repayment status
univariate_categorical("CNT_CHILDREN",True)
```



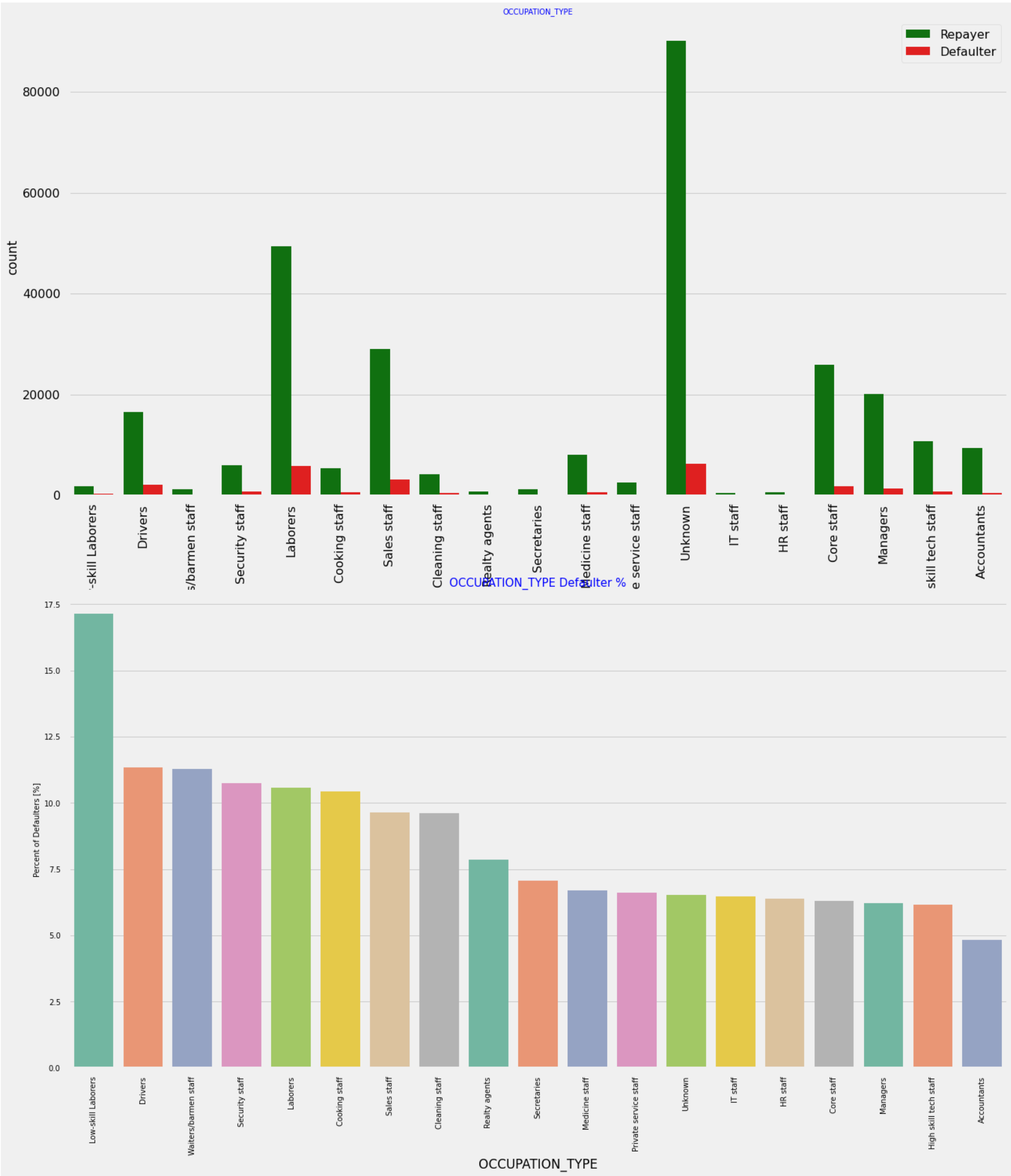
In [111]:

```
#analysing with the credit amount of loan to repayment status
univariate_categorical('LOAN_AMT',False,False,False)
```



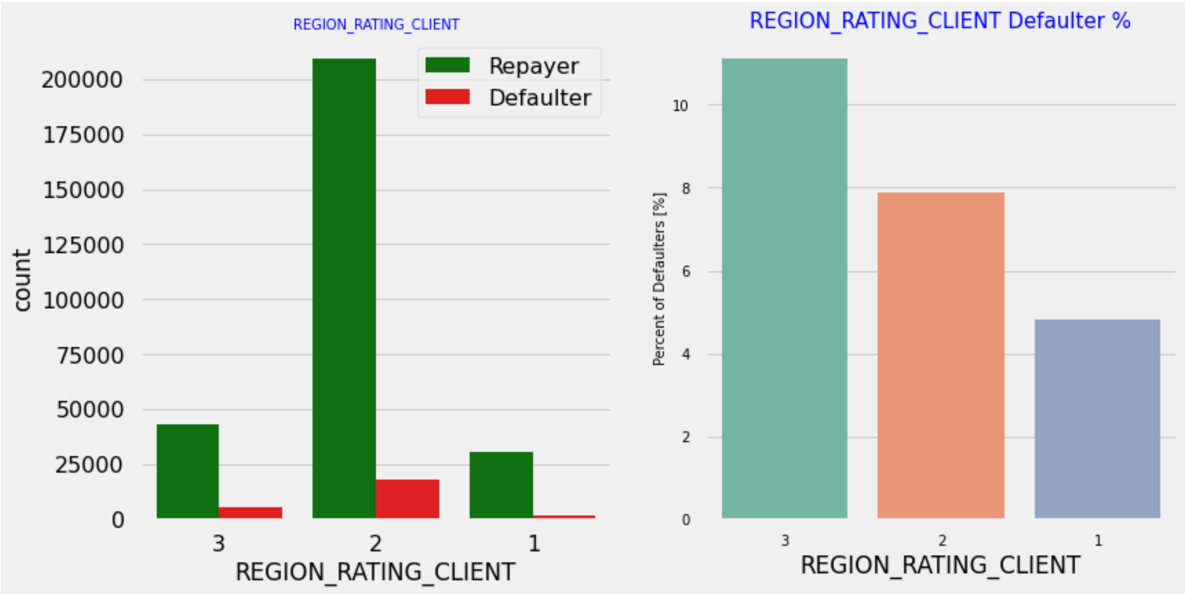
In [112]:

```
#analysing occupation type and loan repaymet status
univariate_categorical("OCCUPATION_TYPE",False,True,False)
```



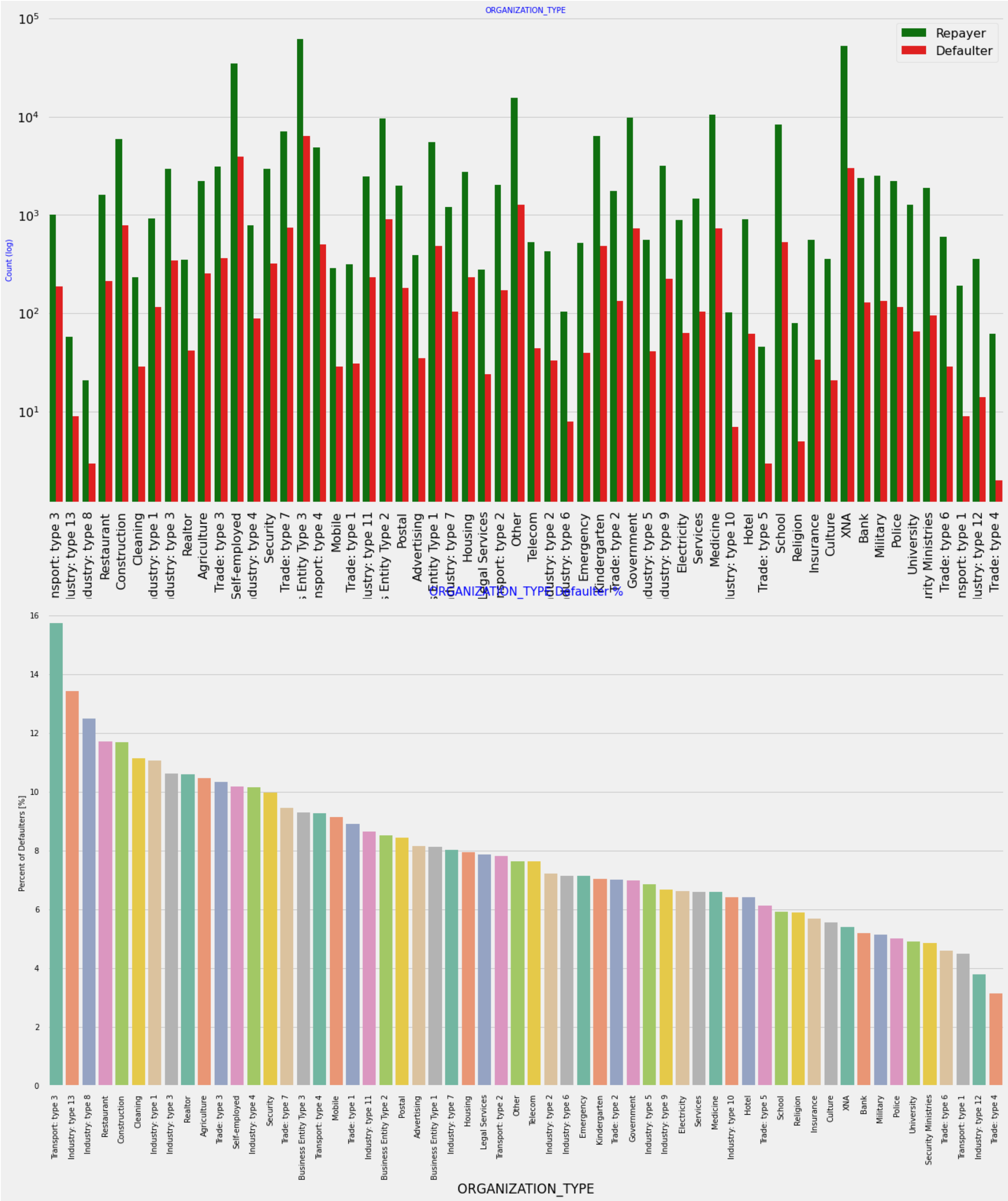
In [113]:

```
#analysing loan repayment status based on client living locality rating
univariate_categorical("REGION_RATING_CLIENT",False,False,True)
```



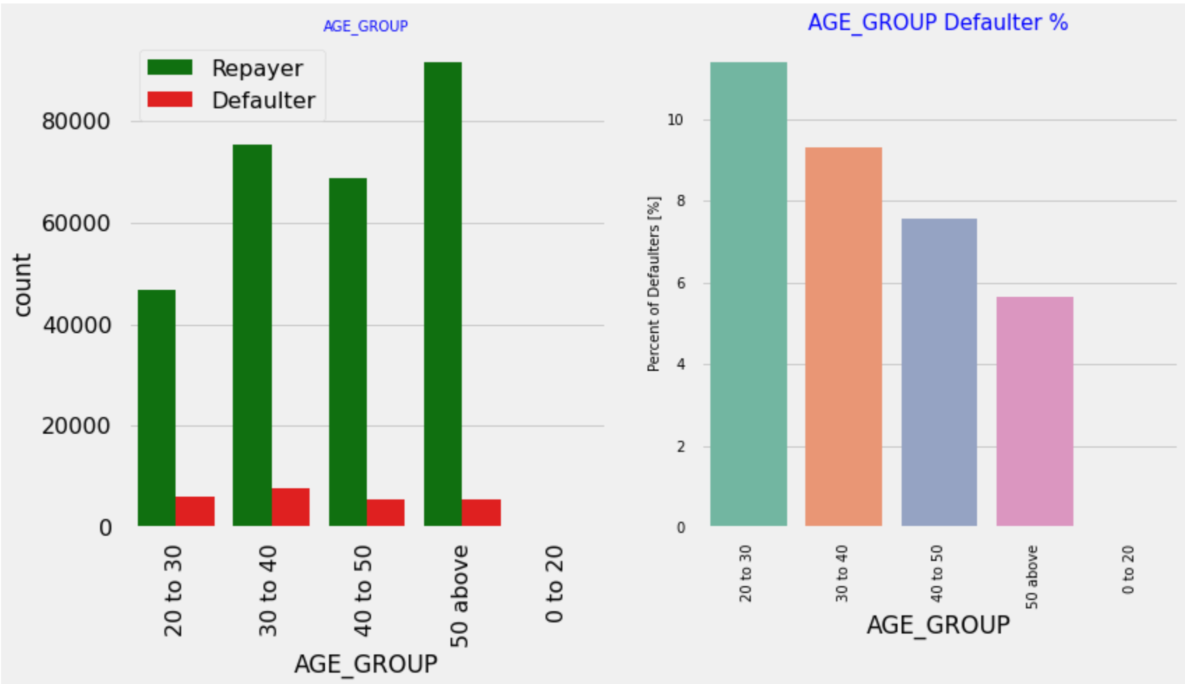
In [114]:

```
#analysing the loan repayment status bases on type of organisation the client works
univariate_categorical("ORGANIZATION_TYPE",True,True,False)
```



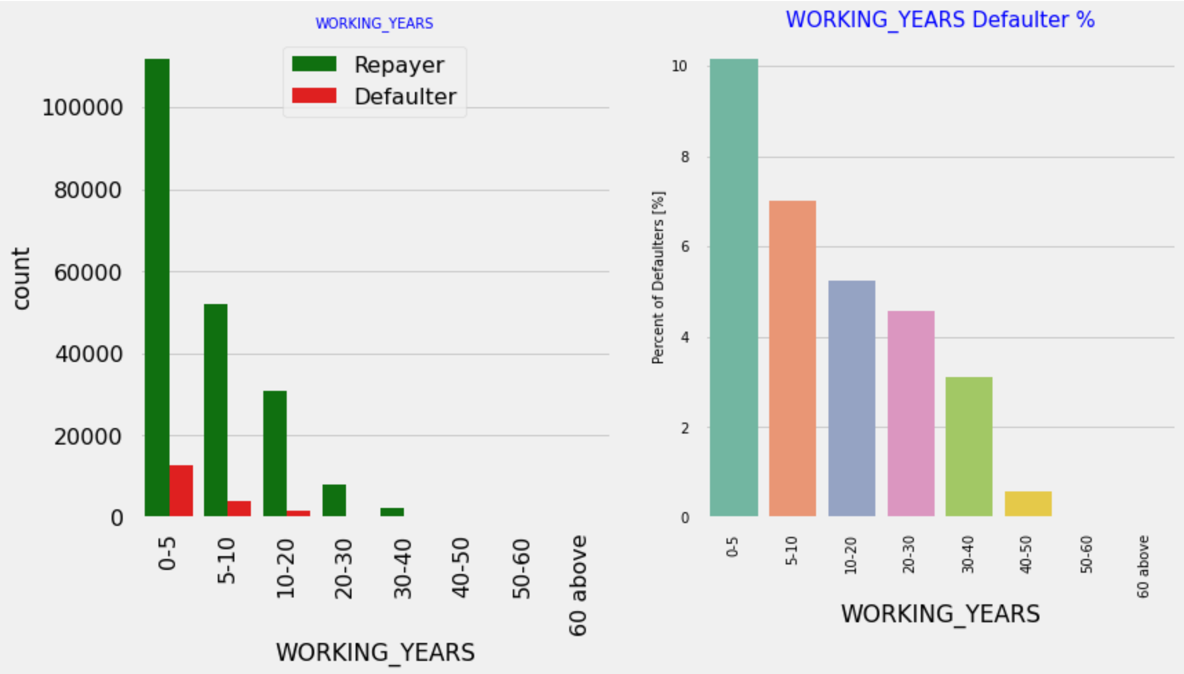
In [115]:

```
#analysing by age
univariate_categorical("AGE_GROUP",False,True,True)
```



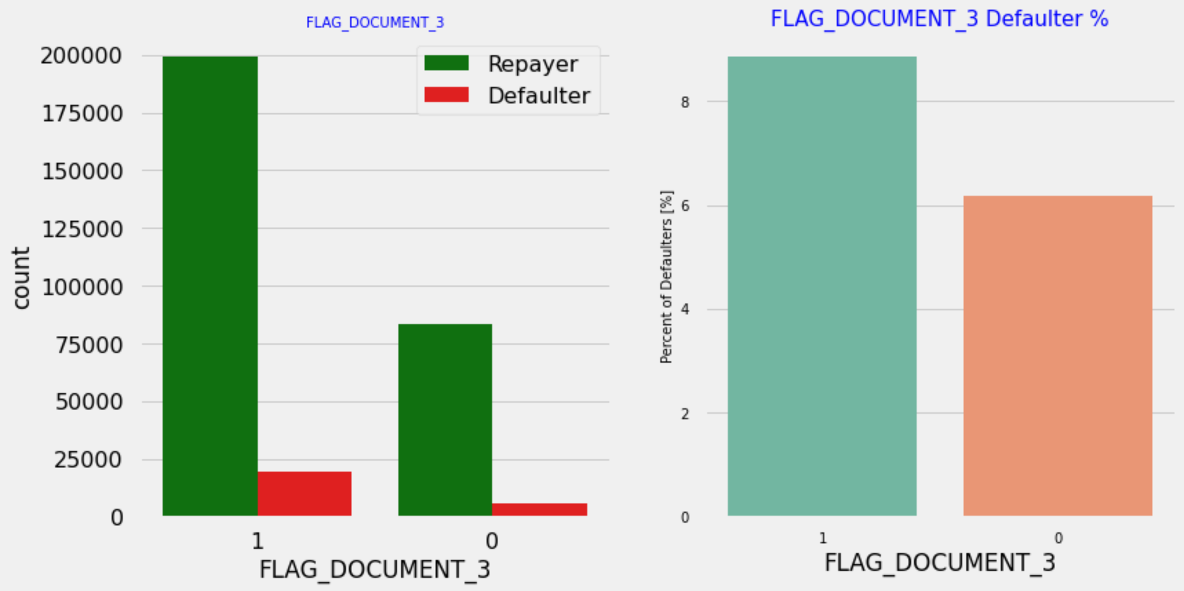
In [116]:

```
#analysing the employment years of the applicant to repayment status
univariate_categorical("WORKING_YEARS",False,True,True)
```



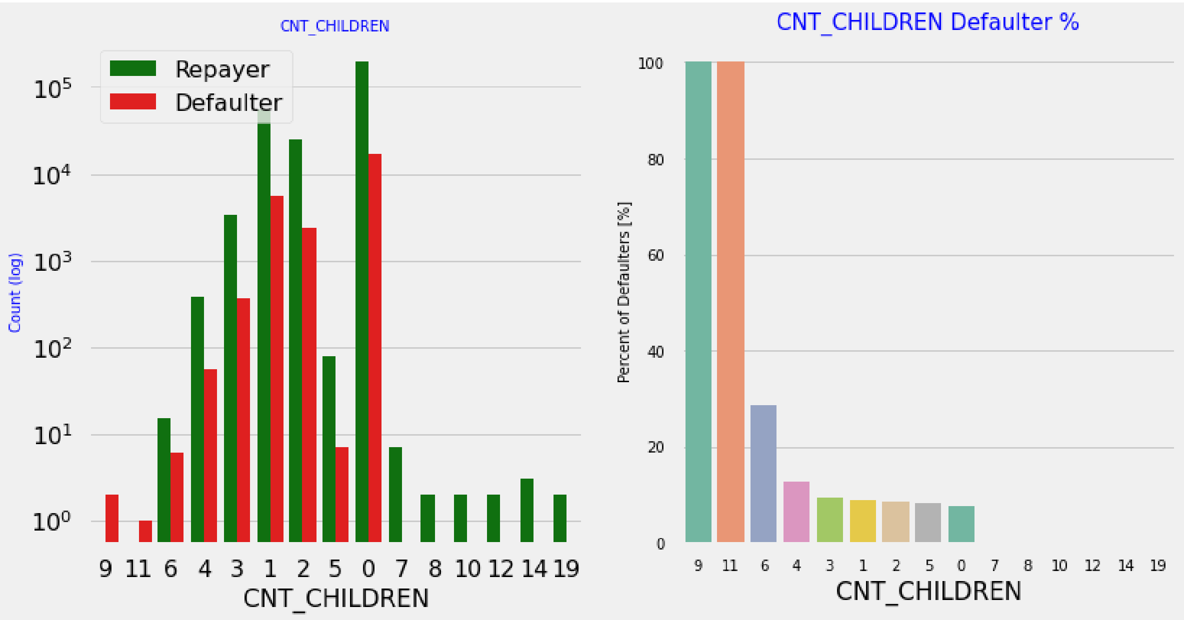
In [117]:

```
# Analyzing Flag_Doc_3 submission status based on loan repayment status
univariate_categorical("FLAG_DOCUMENT_3",False,False,True)
```



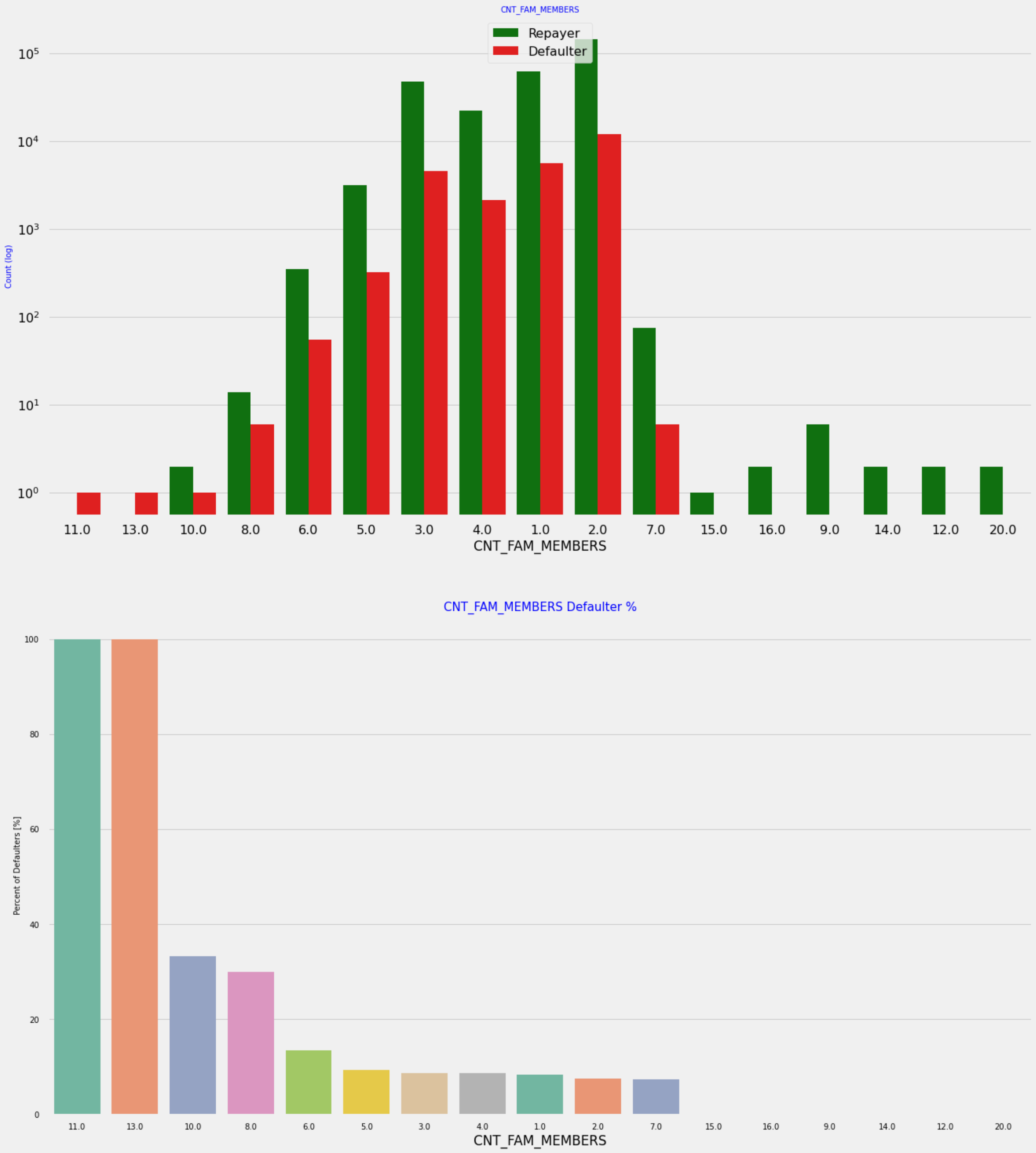
In [118]:

```
# Analyzing Number of children based on loan repayment status
univariate_categorical("CNT_CHILDREN",True)
```



In [119]:

```
# Analyzing Number of family members of client based on loan repayment status
univariate_categorical("CNT_FAM_MEMBERS",True, False, False)
```



In [120]:

```
application.groupby('NAME_INCOME_TYPE')['AMT_INCOME_TOTAL'].describe()
```

Out[120]:

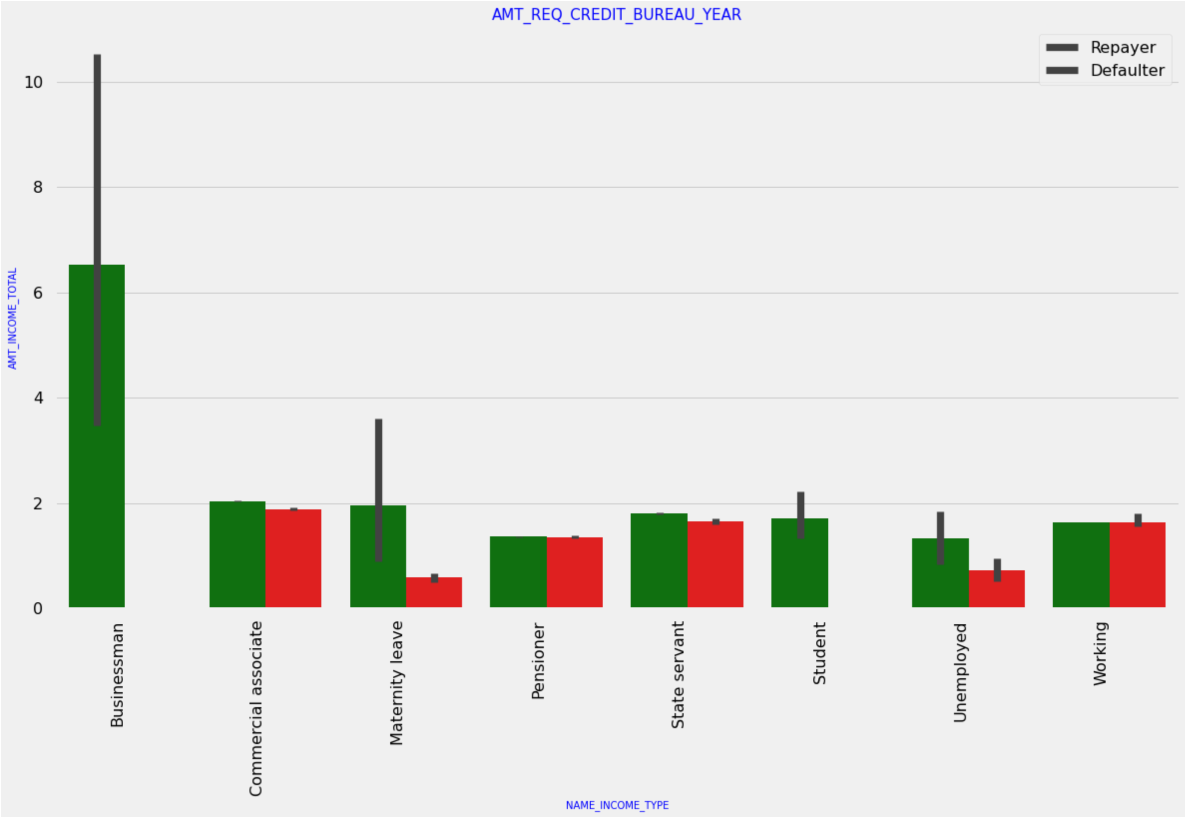
NAME_INCOME_TYPE	count	mean	std	min	25%	50%	75%	max
Businessman	10.0	6.525000	6.272260	1.8000	2.250	4.9500	8.43750	22.5000
Commercial associate	71617.0	2.029553	1.479742	0.2655	1.350	1.8000	2.25000	180.0009
Maternity leave	5.0	1.404000	1.268569	0.4950	0.675	0.9000	1.35000	3.6000
Pensioner	55362.0	1.364013	0.766503	0.2565	0.900	1.1700	1.66500	22.5000
State servant	21703.0	1.797380	1.008806	0.2700	1.125	1.5750	2.25000	31.5000
Student	18.0	1.705000	1.066447	0.8100	1.125	1.5750	1.78875	5.6250
Unemployed	22.0	1.105364	0.880551	0.2655	0.540	0.7875	1.35000	3.3750
Working	158774.0	1.631699	3.075777	0.2565	1.125	1.3500	2.02500	1170.0000

In [ ]:

```

```

In [121]:

```
#Income type vs Income Amount Range
bivariate_bar("NAME_INCOME_TYPE","AMT_INCOME_TOTAL",application,"TARGET",(18,10))
```



In [122]:

```
application.columns
```

Out[122]:

```
Index(['SK_ID_CURR', 'TARGET', 'NAME_CONTRACT_TYPE', 'CODE_GENDER', 'FLAG_OW
N_CAR', 'FLAG_OWN_REALTY', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL', 'AMT_CREDIT',
'AMT_ANNUITY', 'AMT_GOODS_PRICE', 'NAME_TYPE_SUITE', 'NAME_INCOME_TYPE', 'NA
ME_EDUCATION_TYPE', 'NAME_FAMILY_STATUS', 'NAME_HOUSING_TYPE', 'REGION_POPUL
ATION_RELATIVE', 'DAYS_BIRTH', 'DAYS_EMPLOYED', 'DAYS_REGISTRATION', 'DAYS_I
D_PUBLISH', 'OCCUPATION_TYPE', 'CNT_FAM_MEMBERS', 'REGION_RATING_CLIENT', 'R
EGION_RATING_CLIENT_W_CITY', 'WEEKDAY_APPR_PROCESS_START', 'HOUR_APPR_PROCES
S_START', 'REG_REGION_NOT_LIVE_REGION', 'REG_REGION_NOT_WORK_REGION', 'LIVE_
REGION_NOT_WORK_REGION', 'REG_CITY_NOT_LIVE_CITY', 'REG_CITY_NOT_WORK_CITY',
'LIVE_CITY_NOT_WORK_CITY', 'ORGANIZATION_TYPE', 'OBS_30_CNT_SOCIAL_CIRCLE',
'DEF_30_CNT_SOCIAL_CIRCLE', 'OBS_60_CNT_SOCIAL_CIRCLE', 'DEF_60_CNT_SOCIAL_C
IRCLE', 'DAYS_LAST_PHONE_CHANGE', 'FLAG_DOCUMENT_3', 'AMT_REQ_CREDIT_BUREAU_
HOUR', 'AMT_REQ_CREDIT_BUREAU_DAY', 'AMT_REQ_CREDIT_BUREAU_WEEK',
       'AMT_REQ_CREDIT_BUREAU_MON', 'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_CR
EDIT_BUREAU_YEAR', 'INCOME_RANGE', 'LOAN_AMT', 'AGE', 'AGE_GROUP', 'YEARS_EM
PLOYED', 'WORKING_YEARS'],
      dtype='object')
```

In [123]:

```
corr_of_colms =['NAME_CONTRACT_TYPE', 'CODE_GENDER', 'FLAG_OWN_CAR', 'FLAG_OWN_REALTY',
                        'CNT_CHILDREN', 'AMT_INCOME_TOTAL', 'AMT_CREDIT', 'AMT_ANNUITY', 'A
                        'NAME_TYPE_SUITE', 'NAME_INCOME_TYPE', 'NAME_EDUCATION_TYPE', 'NAME
                        'NAME_HOUSING_TYPE', 'REGION_POPULATION_RELATIVE', 'DAYS_BIRTH', 'D
                        'DAYS_REGISTRATION', 'DAYS_ID_PUBLISH', 'OCCUPATION_TYPE', 'CNT_FAM
                        'REGION_RATING_CLIENT_W_CITY', 'WEEKDAY_APPR_PROCESS_START', 'HOUR_
                        'REG_REGION_NOT_LIVE_REGION', 'REG_REGION_NOT_WORK_REGION', 'LIVE_R
                        'REG_CITY_NOT_LIVE_CITY', 'REG_CITY_NOT_WORK_CITY', 'LIVE_CITY_NOT_
                        'OBS_60_CNT_SOCIAL_CIRCLE', 'DEF_60_CNT_SOCIAL_CIRCLE', 'DAYS_LAST_
                        'AMT_REQ_CREDIT_BUREAU_HOUR', 'AMT_REQ_CREDIT_BUREAU_DAY', 'AMT_REQ
                        'AMT_REQ_CREDIT_BUREAU_MON', 'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_
repayer=application.loc[application['TARGET']==0, corr_of_colms]
defaulter=application.loc[application['TARGET']==1,corr_of_colms]
```

In [124]:

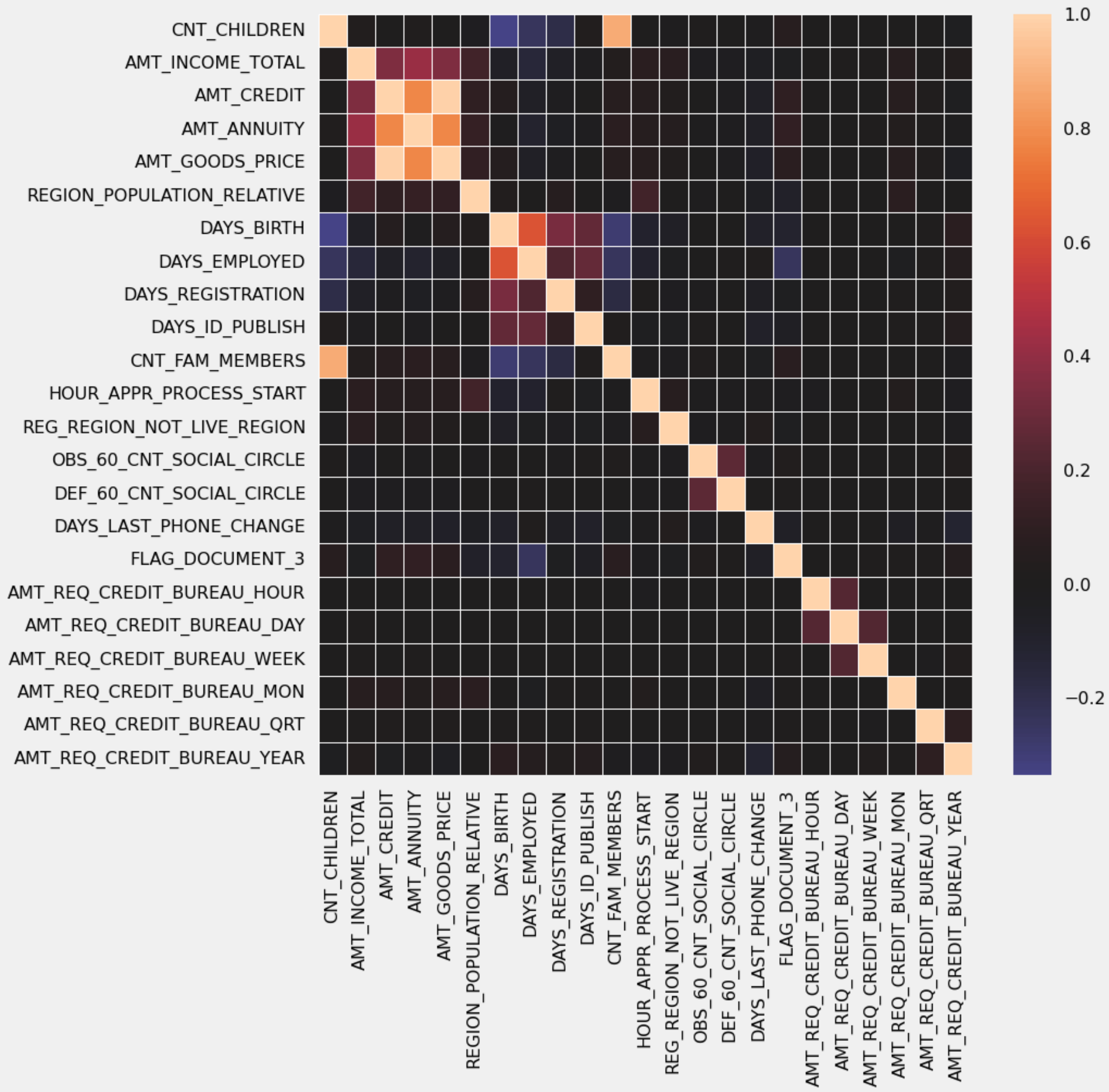
```
#top 10 correlation for the Repayers data
corr_repayer = repayer.corr()
corr_repayer = corr_repayer.where(np.triu(np.ones(corr_repayer.shape),k=1).astype(np.bool))
corr_df_repayer = corr_repayer.unstack().reset_index()
corr_df_repayer.columns =['atribute1','atribute2','Correlation']
corr_df_repayer.dropna(subset = ["Correlation"], inplace = True)
corr_df_repayer["Correlation"]=corr_df_repayer["Correlation"].abs()
corr_df_repayer.sort_values(by='Correlation', ascending=False, inplace=True)
corr_df_repayer.head(10)
```

Out[124]:

	atribute1	atribute2	Correlation
94	AMT_GOODS_PRICE	AMT_CREDIT	0.987250
230	CNT_FAM_MEMBERS	CNT_CHILDREN	0.878571
95	AMT_GOODS_PRICE	AMT_ANNUITY	0.776686
71	AMT_ANNUITY	AMT_CREDIT	0.771309
167	DAYS_EMPLOYED	DAYS_BIRTH	0.626114
70	AMT_ANNUITY	AMT_INCOME_TOTAL	0.418953
93	AMT_GOODS_PRICE	AMT_INCOME_TOTAL	0.349462
47	AMT_CREDIT	AMT_INCOME_TOTAL	0.342799
138	DAYS_BIRTH	CNT_CHILDREN	0.336966
190	DAYS_REGISTRATION	DAYS_BIRTH	0.333151

In [125]:

```
fig = plt.figure(figsize=(12,12))
ax = sns.heatmap(repayer.corr(), center=0,annot=False,linewidth =1)
```



In [126]:

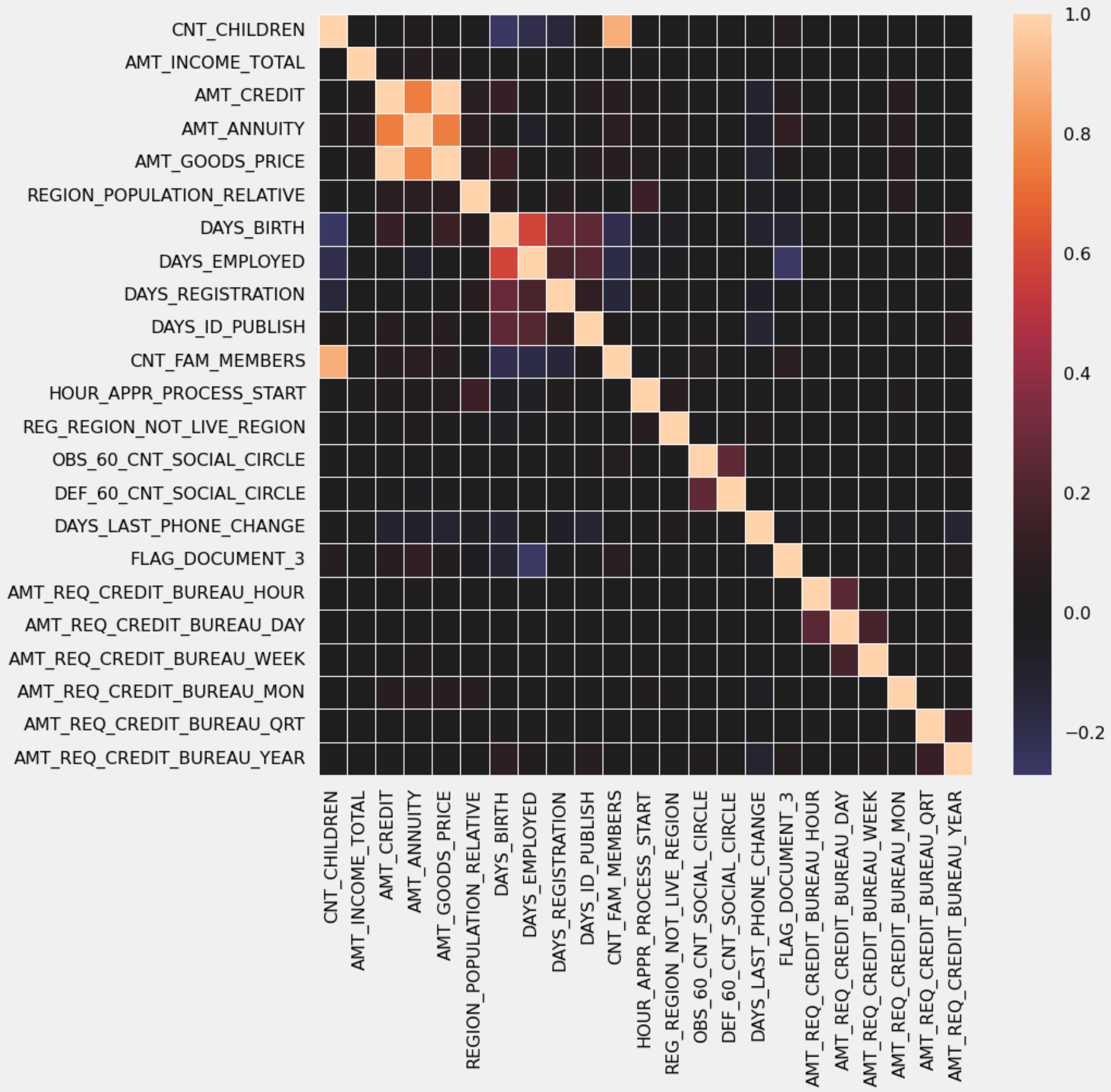
```
corr_Defaulter = defaulter.corr()
corr_Defaulter = corr_Defaulter.where(np.triu(np.ones(corr_Defaulter.shape),k=1).astype(np.
corr_df_Defaulter = corr_Defaulter.unstack().reset_index()
corr_df_Defaulter.columns =['atribute1','atribute2','Correlation']
corr_df_Defaulter.dropna(subset = ["Correlation"], inplace = True)
corr_df_Defaulter["Correlation"]=corr_df_Defaulter["Correlation"].abs()
corr_df_Defaulter.sort_values(by='Correlation', ascending=False, inplace=True)
corr_df_Defaulter.head(10)
```

Out[126]:

	atribute1	atribute2	Correlation
94	AMT_GOODS_PRICE	AMT_CREDIT	0.983103
230	CNT_FAM_MEMBERS	CNT_CHILDREN	0.885484
95	AMT_GOODS_PRICE	AMT_ANNUITY	0.752699
71	AMT_ANNUITY	AMT_CREDIT	0.752195
167	DAYS_EMPLOYED	DAYS_BIRTH	0.582185
190	DAYS_REGISTRATION	DAYS_BIRTH	0.289114
375	FLAG_DOCUMENT_3	DAYS_EMPLOYED	0.272169
335	DEF_60_CNT_SOCIAL_CIRCLE	OBS_60_CNT_SOCIAL_CIRCLE	0.264159
138	DAYS_BIRTH	CNT_CHILDREN	0.259109
213	DAYS_ID_PUBLISH	DAYS_BIRTH	0.252863

In [127]:

```
fig = plt.figure(figsize=(12,12))
ax = sns.heatmap(defaulter.corr(), center=0,annot=False,linewidth =1)
```

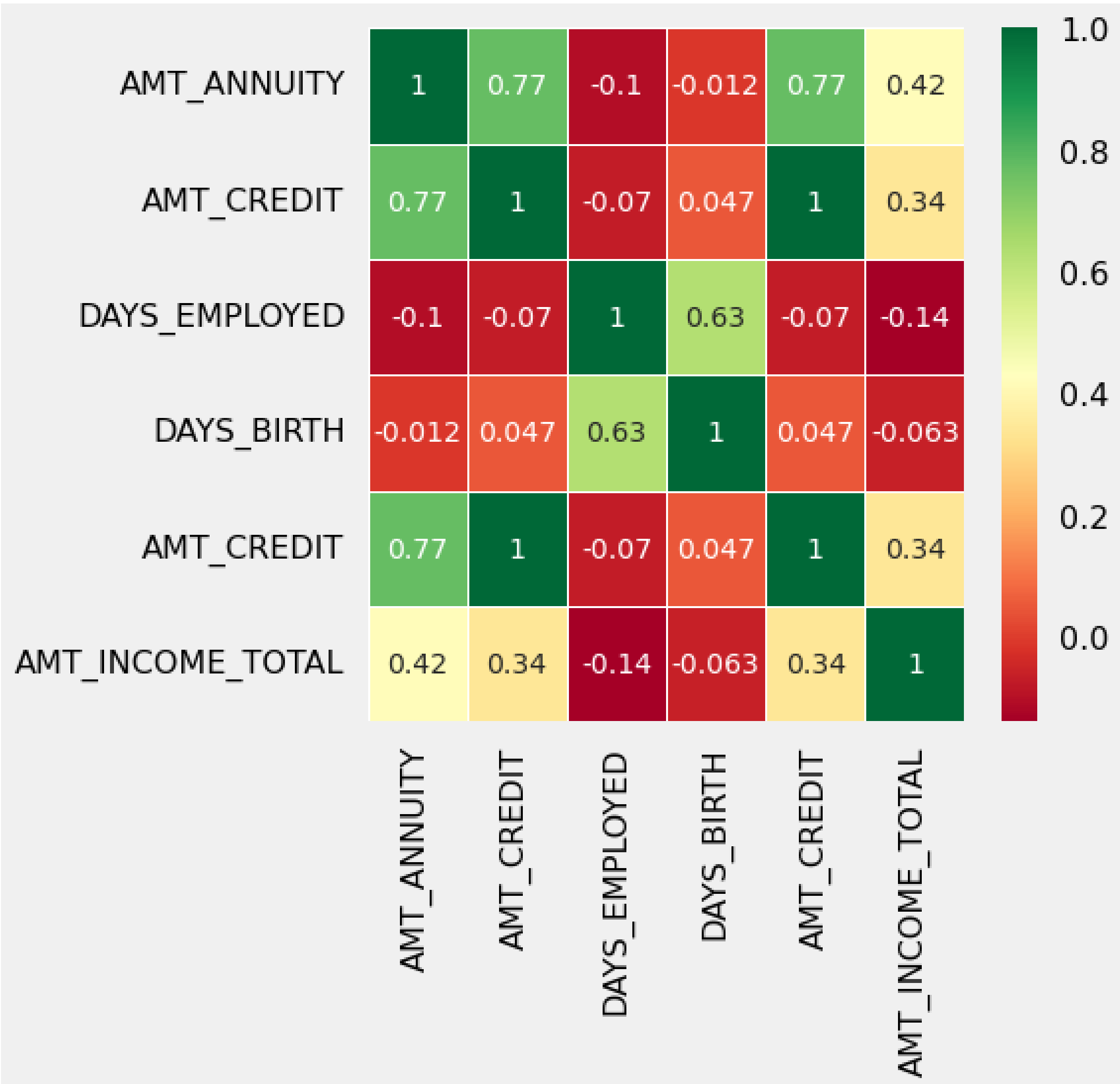


In [128]:

```
final_corr=['AMT_ANNUITY','AMT_CREDIT','DAYS_EMPLOYED','DAYS_BIRTH','AMT_CREDIT','AMT_INCOM
repayer_final=application.loc[application['TARGET']==0,final_corr]
defaulter_final=application.loc[application['TARGET']==1,final_corr]
```

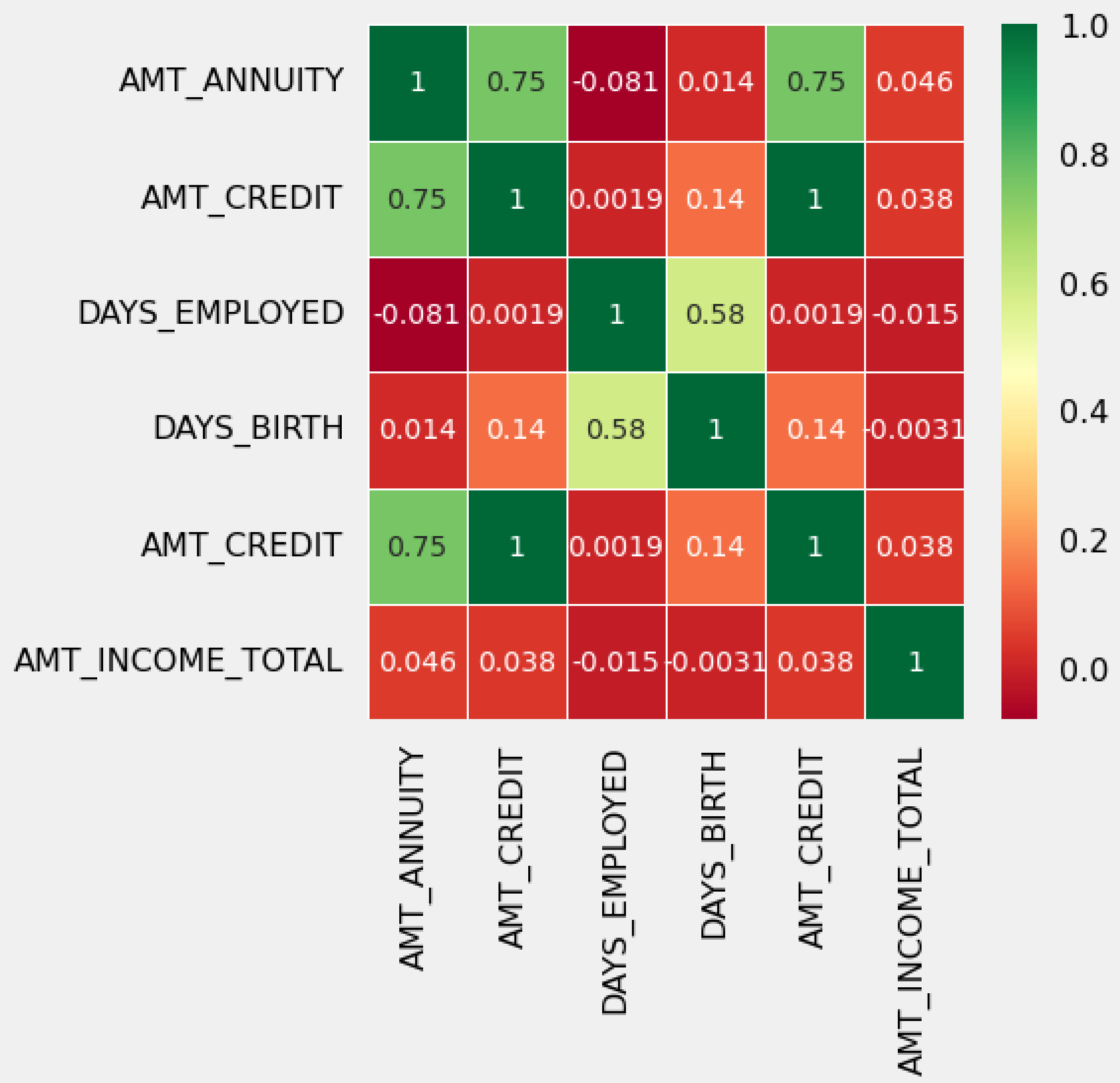
In [129]:

```
fig = plt.figure(figsize=(6,6))
ax = sns.heatmap(repayer_final.corr(), cmap="RdYlGn",annot=True,linewidth =1)
```



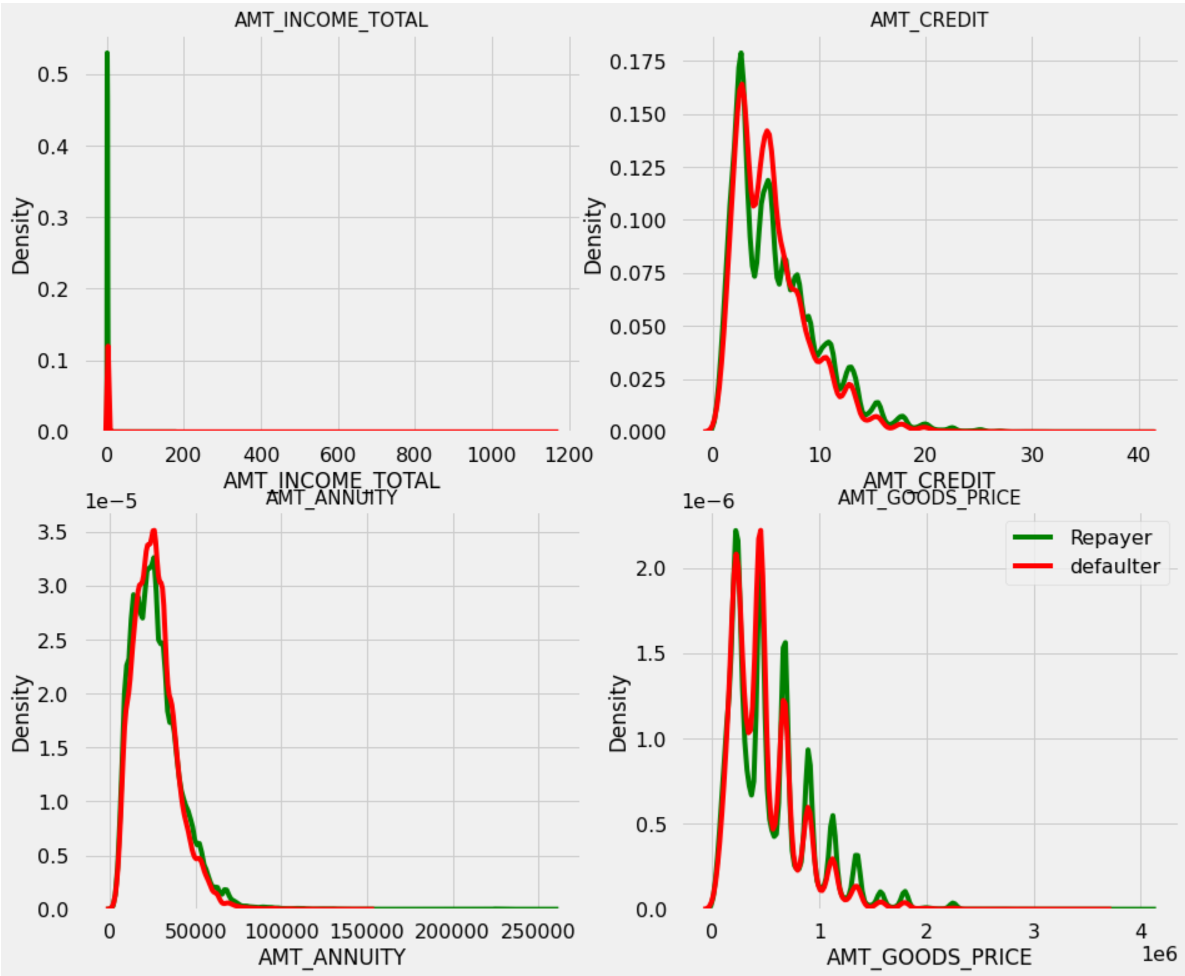
In [130]:

```
plt.figure(figsize=(6,6))
ax=sns.heatmap(defaulter_final.corr(),annot=True,cmap="RdYlGn",linewidth=1)
```



In [131]:

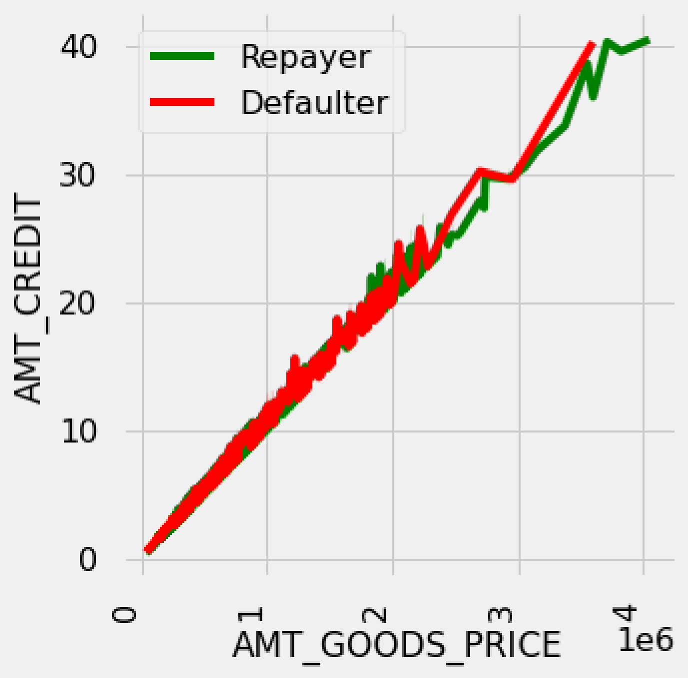
```
fig=plt.figure(figsize=(14,12))
amount=application[['AMT_INCOME_TOTAL','AMT_CREDIT','AMT_ANNUITY', 'AMT_GOODS_PRICE']]
for i in enumerate(amount):
    plt.subplot(2,2,i[0]+1)
    sns.distplot(repayer[i[1]],hist=False,color='g',label='Repayer')
    sns.distplot(defaulter[i[1]],hist=False,color='r',label='defaulter')
    plt.title(i[1],fontdict={'fontsize' : 15, 'fontweight' : 5})
plt.legend()
plt.show()
```



In [132]:

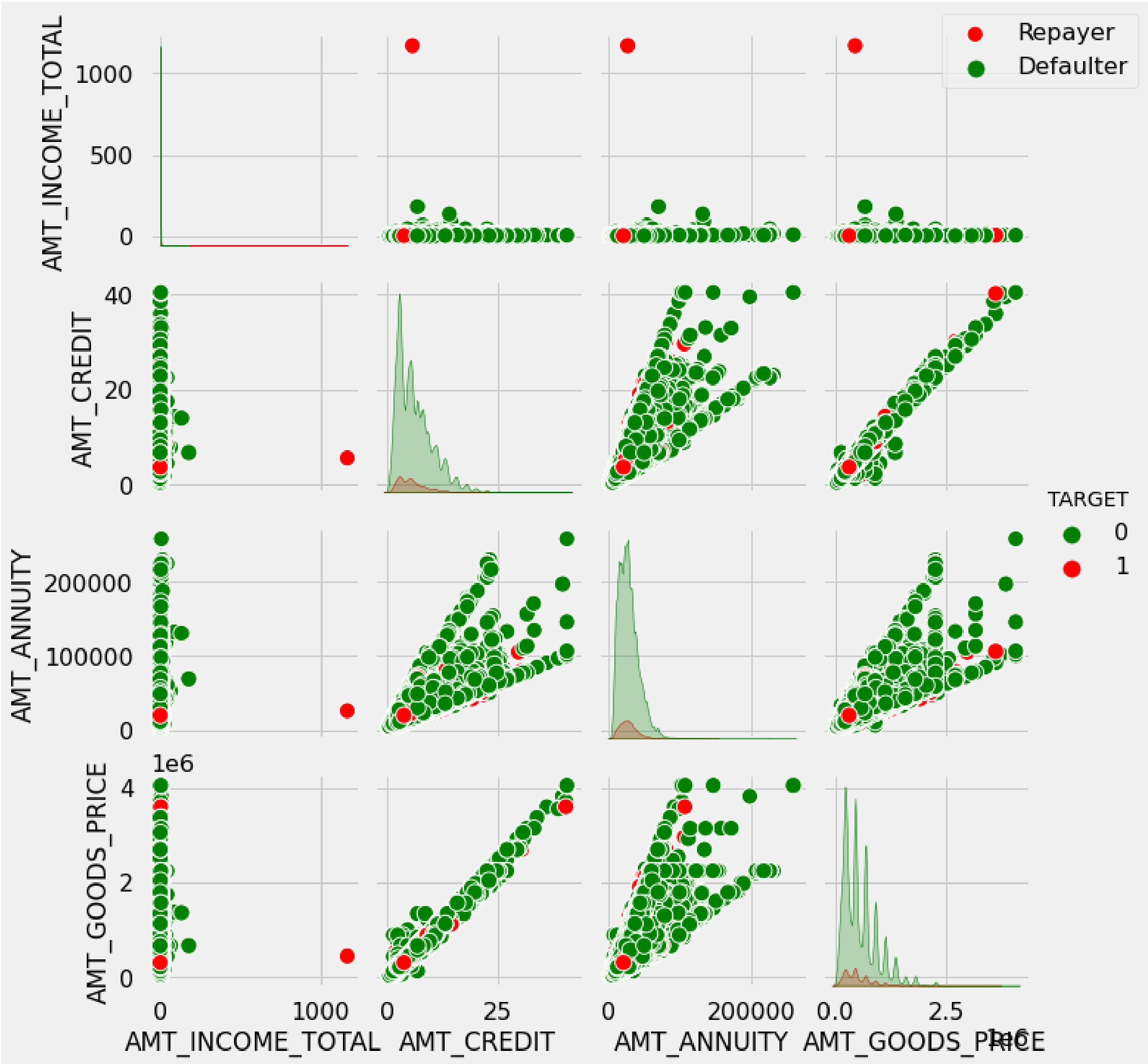
```
bivariate_rel('AMT_GOODS_PRICE','AMT_CREDIT',application,"TARGET", "line", ['g','r'], False
```

<Figure size 1080x432 with 0 Axes>



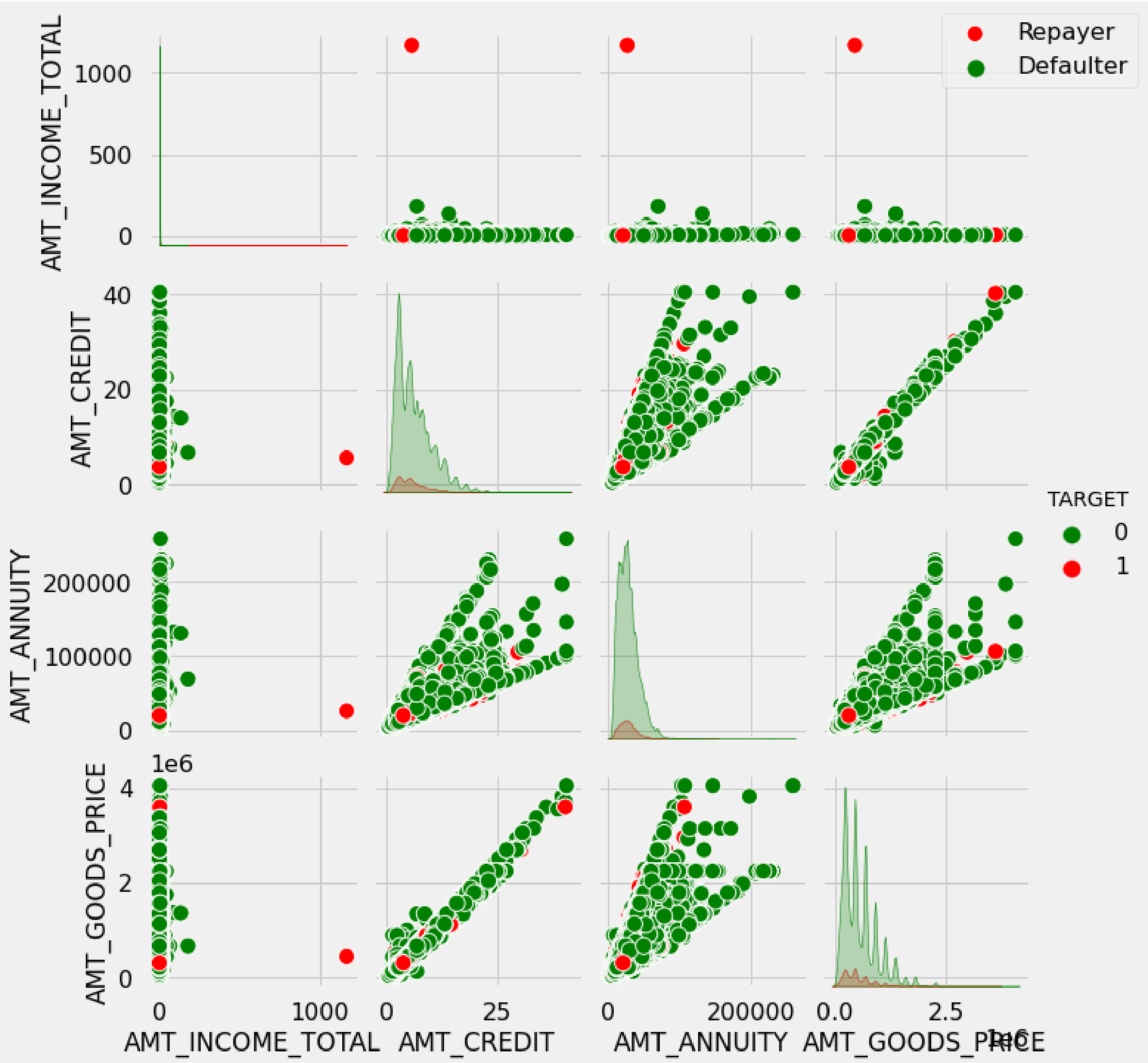
In [133]:

```
amount = application[[ 'AMT_INCOME_TOTAL','AMT_CREDIT',
                        'AMT_ANNUITY', 'AMT_GOODS_PRICE','TARGET']]
amount = amount[(amount["AMT_GOODS_PRICE"].notnull()) & (amount["AMT_ANNUITY"].notnull())]
ax= sns.pairplot(amount,hue="TARGET",palette=["g","r"])
ax.fig.legend(labels=['Repayer','Defaulter'])
plt.show()
```



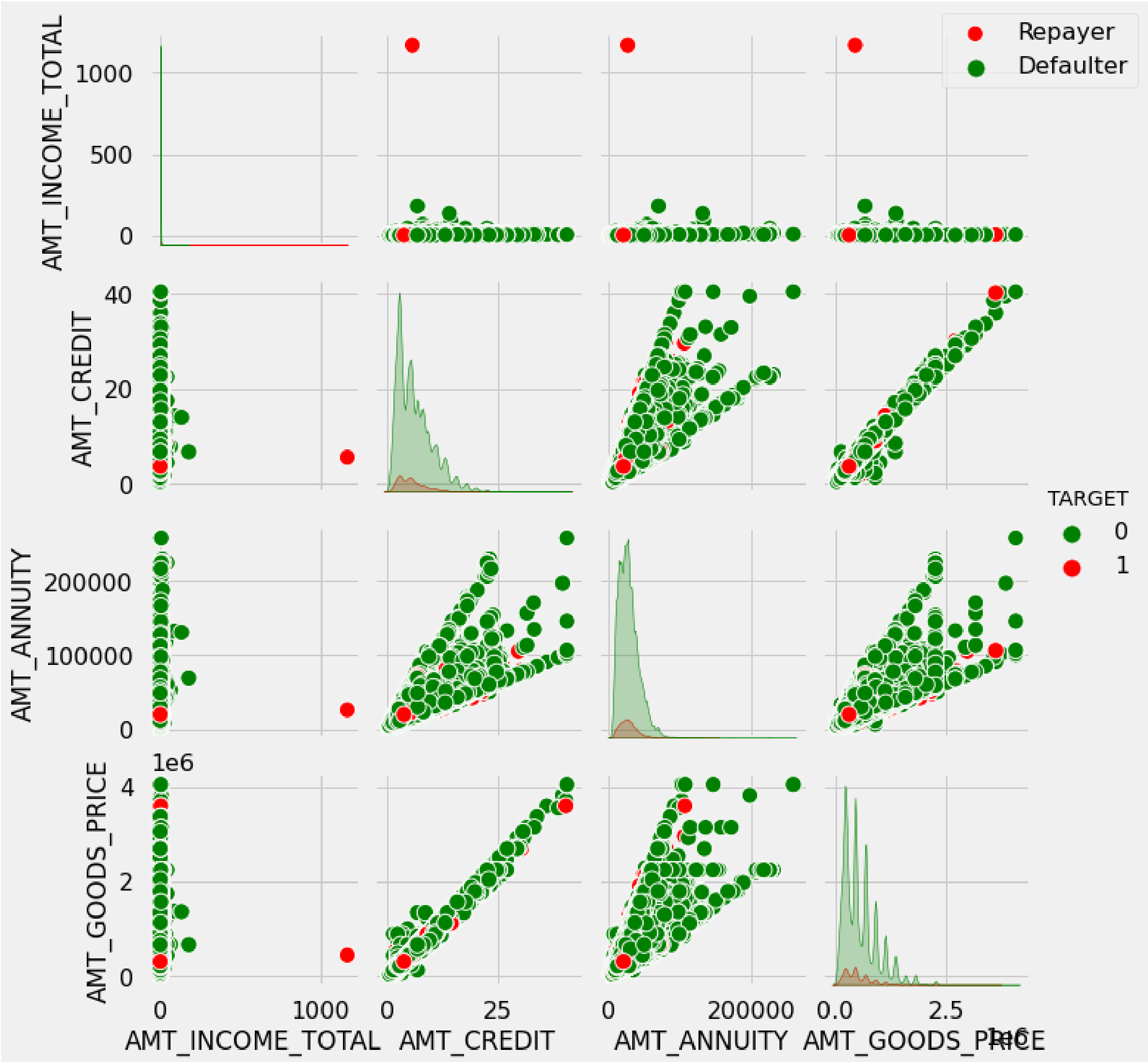
In [134]:

```
amount = application[[ 'AMT_INCOME_TOTAL','AMT_CREDIT',
                         'AMT_ANNUITY', 'AMT_GOODS_PRICE','TARGET']]
amount = amount[(amount["AMT_GOODS_PRICE"].notnull()) & (amount["AMT_ANNUITY"].notnull())]
ax= sns.pairplot(amount,hue="TARGET",palette=["g","r"])
ax.fig.legend(labels=['Repayer','Defaulter'])
plt.show()
```



In [135]:

```
amount = application[[ 'AMT_INCOME_TOTAL','AMT_CREDIT',
                        'AMT_ANNUITY', 'AMT_GOODS_PRICE','TARGET']]
ax= sns.pairplot(amount,hue="TARGET",palette=["g","r"])
ax.fig.legend(labels=['Repayer','Defaulter'])
plt.show()
```



In [136]:

```
loan_application = pd.merge(application, previous, how='inner', on='SK_ID_CURR')
loan_application.head()
```

Out[136]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE_x	CODE_GENDER	FLAG_OWN_CAR	FLAG
0	100002	1	Cash loans	M	N	
1	100003	0	Cash loans	F	N	
2	100003	0	Cash loans	F	N	
3	100003	0	Cash loans	F	N	
4	100004	0	Revolving loans	M	Y	

5 rows × 74 columns

In [137]:

```
loan_application.shape
```

Out[137]:

```
(1413701, 74)
```

In [138]:

```
loan_application.size
```

Out[138]:

```
104613874
```

In [139]:

```
loan_application.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1413701 entries, 0 to 1413700
Data columns (total 74 columns):
 #   Column                       Non-Null Count    Dtype   
---  ------                       --------------    -----   
 0   SK_ID_CURR                   1413701 non-null  int64   
 1   TARGET                       1413701 non-null  int64   
 2   NAME_CONTRACT_TYPE_x         1413701 non-null  category
 3   CODE_GENDER                  1413701 non-null  category
 4   FLAG_OWN_CAR                 1413701 non-null  category
 5   FLAG_OWN_REALTY              1413701 non-null  category
 6   CNT_CHILDREN                 1413701 non-null  int64   
 7   AMT_INCOME_TOTAL             1413701 non-null  float64 
 8   AMT_CREDIT_x                 1413701 non-null  float64 
 9   AMT_ANNUITY_x                1413608 non-null  float64 
 10  AMT_GOODS_PRICE_x            1412493 non-null  float64 
 11  NAME_TYPE_SUITE              1413701 non-null  category
 12  NAME_INCOME_TYPE             1413701 non-null  category
 13  NAME_EDUCATION_TYPE          1413701 non-null  category
 14  NAME_FAMILY_STATUS           1413701 non-null  category
 15  NAME_HOUSING_TYPE            1413701 non-null  category
 16  REGION_POPULATION_RELATIVE   1413701 non-null  float64 
 17  DAYS_BIRTH                   1413701 non-null  int64   
 18  DAYS_EMPLOYED                1413701 non-null  int64   
 19  DAYS_REGISTRATION            1413701 non-null  float64 
 20  DAYS_ID_PUBLISH              1413701 non-null  int64   
 21  OCCUPATION_TYPE              1413701 non-null  category
 22  CNT_FAM_MEMBERS              1413701 non-null  float64 
 23  REGION_RATING_CLIENT         1413701 non-null  category
 24  REGION_RATING_CLIENT_W_CITY  1413701 non-null  category
 25  WEEKDAY_APPR_PROCESS_START   1413701 non-null  category
 26  HOUR_APPR_PROCESS_START      1413701 non-null  int64   
 27  REG_REGION_NOT_LIVE_REGION   1413701 non-null  int64   
 28  REG_REGION_NOT_WORK_REGION   1413701 non-null  category
 29  LIVE_REGION_NOT_WORK_REGION  1413701 non-null  category
 30  REG_CITY_NOT_LIVE_CITY       1413701 non-null  category
 31  REG_CITY_NOT_WORK_CITY       1413701 non-null  category
 32  LIVE_CITY_NOT_WORK_CITY      1413701 non-null  category
 33  ORGANIZATION_TYPE            1413701 non-null  category
 34  OBS_30_CNT_SOCIAL_CIRCLE     1410555 non-null  float64 
 35  DEF_30_CNT_SOCIAL_CIRCLE     1410555 non-null  float64 
 36  OBS_60_CNT_SOCIAL_CIRCLE     1410555 non-null  float64 
 37  DEF_60_CNT_SOCIAL_CIRCLE     1410555 non-null  float64 
 38  DAYS_LAST_PHONE_CHANGE       1413701 non-null  float64 
 39  FLAG_DOCUMENT_3              1413701 non-null  int64   
 40  AMT_REQ_CREDIT_BUREAU_HOUR   1413701 non-null  float64 
 41  AMT_REQ_CREDIT_BUREAU_DAY    1413701 non-null  float64 
 42  AMT_REQ_CREDIT_BUREAU_WEEK   1413701 non-null  float64 
 43  AMT_REQ_CREDIT_BUREAU_MON    1413701 non-null  float64 
 44  AMT_REQ_CREDIT_BUREAU_QRT    1413701 non-null  float64 
 45  AMT_REQ_CREDIT_BUREAU_YEAR   1413701 non-null  float64 
 46  INCOME_RANGE                 1413024 non-null  category
 47  LOAN_AMT                     1413701 non-null  category
 48  AGE                          1413701 non-null  int64   
 49  AGE_GROUP                    1413701 non-null  category
 50  YEARS_EMPLOYED               1413701 non-null  int64   
```

```
 51  WORKING_YEARS               1032756 non-null  category
 52  SK_ID_PREV                  1413701 non-null  int64
 53  NAME_CONTRACT_TYPE_y        1413701 non-null  category
 54  AMT_ANNUITY_y               1413701 non-null  float64
 55  AMT_APPLICATION             1413701 non-null  float64
 56  AMT_CREDIT_y                1413700 non-null  float64
 57  AMT_GOODS_PRICE_y           1413701 non-null  float64
 58  NAME_CASH_LOAN_PURPOSE      1413701 non-null  category
 59  NAME_CONTRACT_STATUS        1413701 non-null  category
 60  DAYS_DECISION               1413701 non-null  int64
 61  NAME_PAYMENT_TYPE           1413701 non-null  category
 62  CODE_REJECT_REASON          1413701 non-null  category
 63  NAME_CLIENT_TYPE            1413701 non-null  category
 64  NAME_GOODS_CATEGORY         1413701 non-null  category
 65  NAME_PORTFOLIO              1413701 non-null  category
 66  NAME_PRODUCT_TYPE           1413701 non-null  category
 67  CHANNEL_TYPE                1413701 non-null  category
 68  SELLERPLACE_AREA            1413701 non-null  int64
 69  NAME_SELLER_INDUSTRY        1413701 non-null  category
 70  CNT_PAYMENT                 1413701 non-null  float64
 71  NAME_YIELD_GROUP            1413701 non-null  category
 72  PRODUCT_COMBINATION         1413388 non-null  category
 73  DAYS_DECISION_GROUP         1413701 non-null  category
dtypes: category(37), float64(23), int64(14)
memory usage: 459.8 MB
```

In [140]:

```
loan_application.describe()
```

Out[140]:

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT_x	AM
count	1.413701e+06	1.413701e+06	1.413701e+06	1.413701e+06	1.413701e+06	
mean	2.784813e+05	8.655296e-02	4.048933e-01	1.733160e+00	5.875537e+00	
std	1.028118e+05	2.811789e-01	7.173454e-01	1.985734e+00	3.849173e+00	
min	1.000020e+05	0.000000e+00	0.000000e+00	2.565000e-01	4.500000e-01	
25%	1.893640e+05	0.000000e+00	0.000000e+00	1.125000e+00	2.700000e+00	
50%	2.789920e+05	0.000000e+00	0.000000e+00	1.575000e+00	5.084955e+00	
75%	3.675560e+05	0.000000e+00	1.000000e+00	2.070000e+00	8.079840e+00	
max	4.562550e+05	1.000000e+00	1.900000e+01	1.170000e+03	4.050000e+01	

8 rows × 37 columns

In [141]:

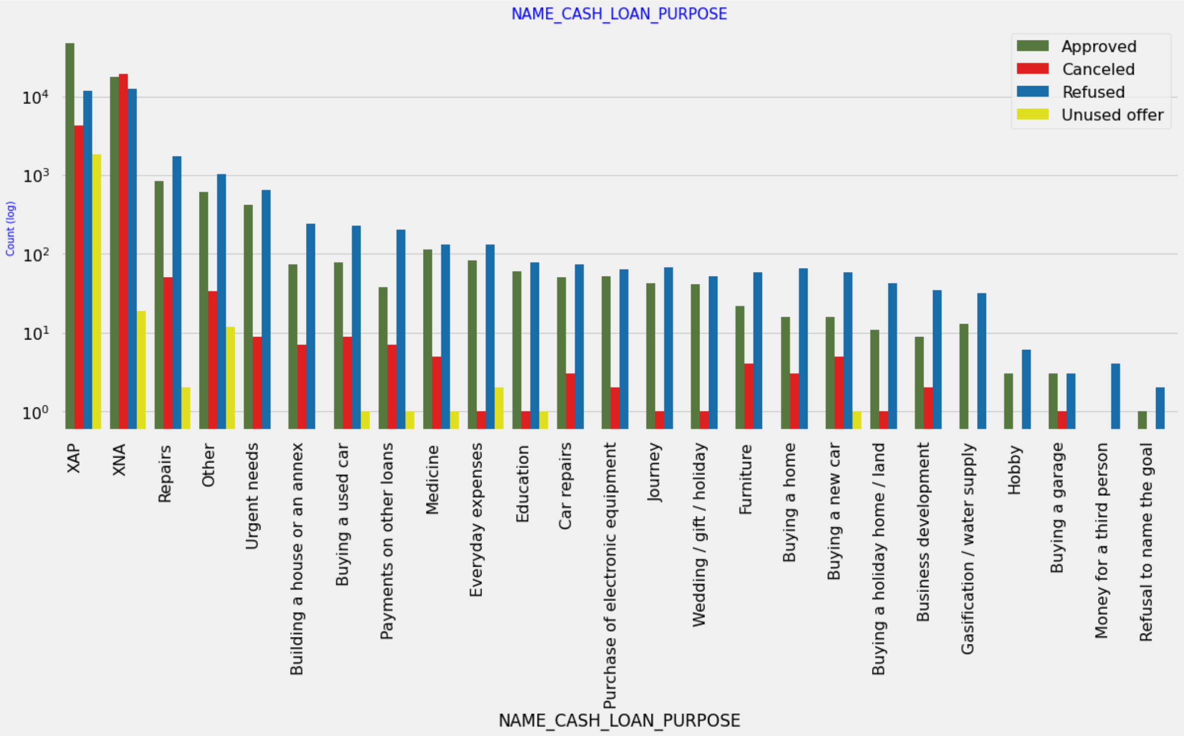
```
Target_0=loan_application[loan_application['TARGET']==0]
```

In [143]:

```
Target_1=loan_application[loan_application['TARGET']==1]
```

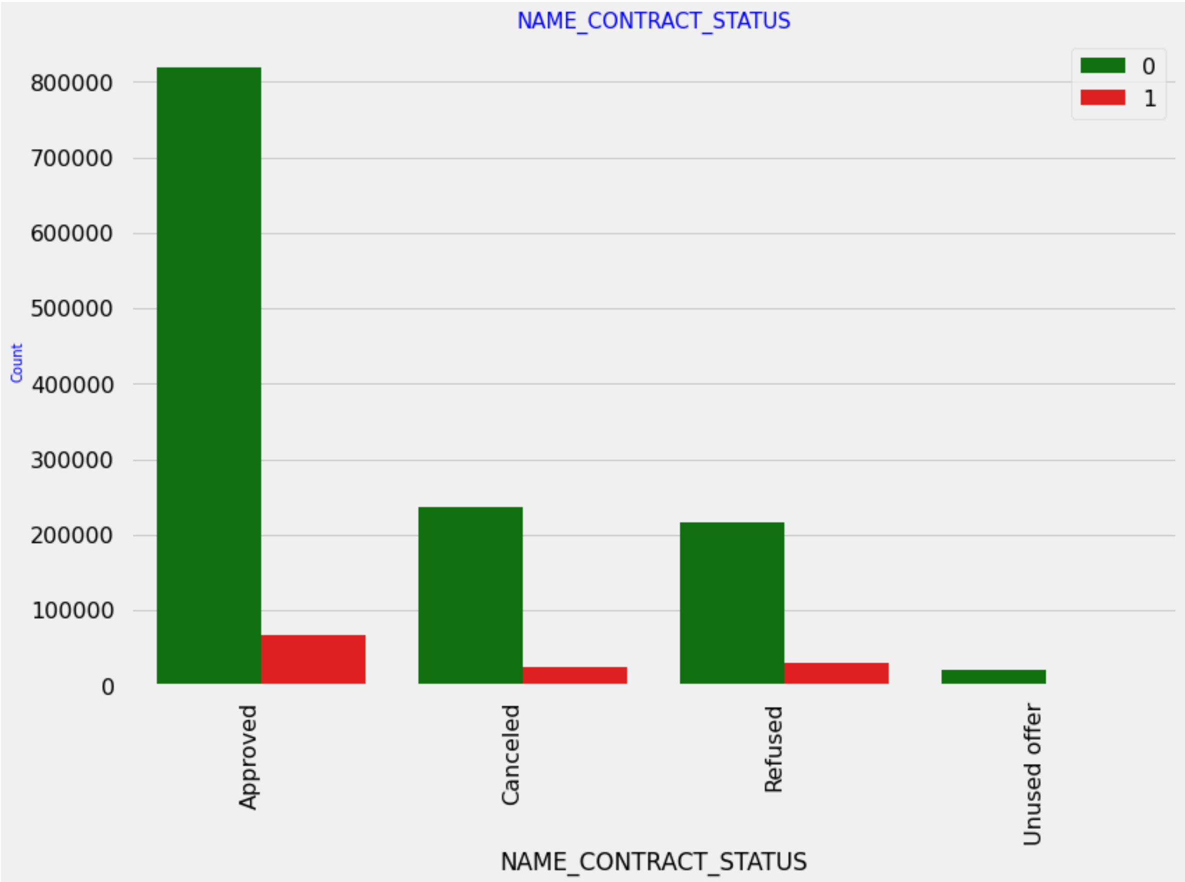
In [144]:

```
univariate_merged("NAME_CASH_LOAN_PURPOSE",Target_1,"NAME_CONTRACT_STATUS",["#548235","#FF0
```



In [145]:

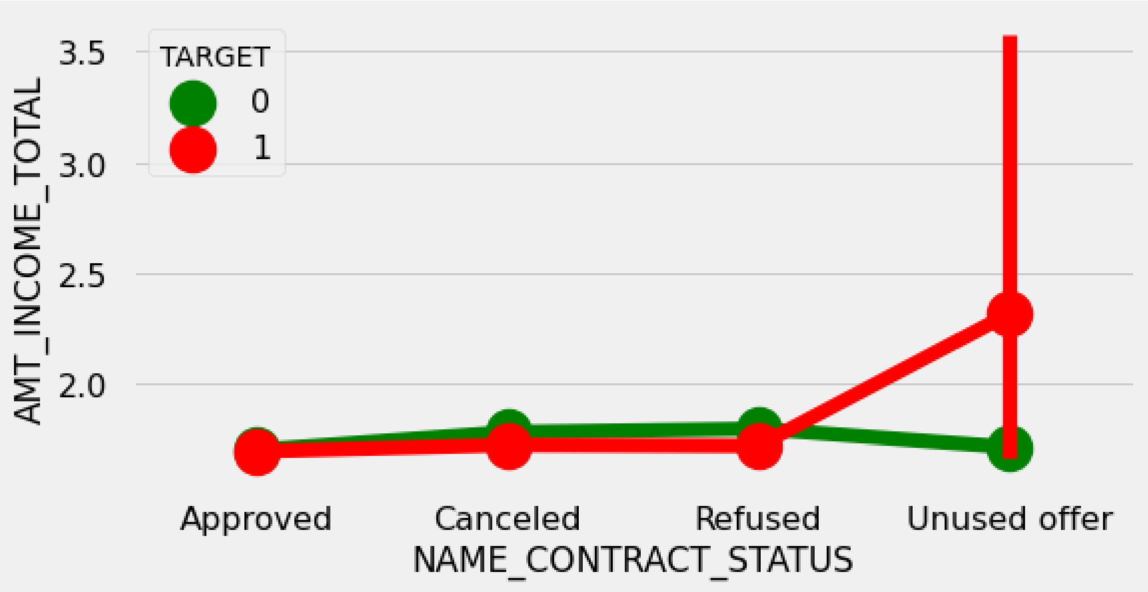
```
univariate_merged("NAME_CONTRACT_STATUS",loan_application,"TARGET",['g','r'],False,(12,8))
g =loan_application.groupby("NAME_CONTRACT_STATUS")["TARGET"]
df1 = pd.concat([g.value_counts(),round(g.value_counts(normalize=True).mul(100),2)],axis=1,
df1['Percentage'] = df1['Percentage'].astype(str) +"%" # adding percentage symbol in the re
print (df1)
```




```
                             Counts Percentage
NAME_CONTRACT_STATUS TARGET
Approved             0       818856     92.41%
                     1        67243      7.59%
Canceled             0       235641     90.83%
                     1        23800      9.17%
Refused              0       215952      88.0%
                     1        29438      12.0%
Unused offer         0        20892     91.75%
                     1         1879      8.25%
```

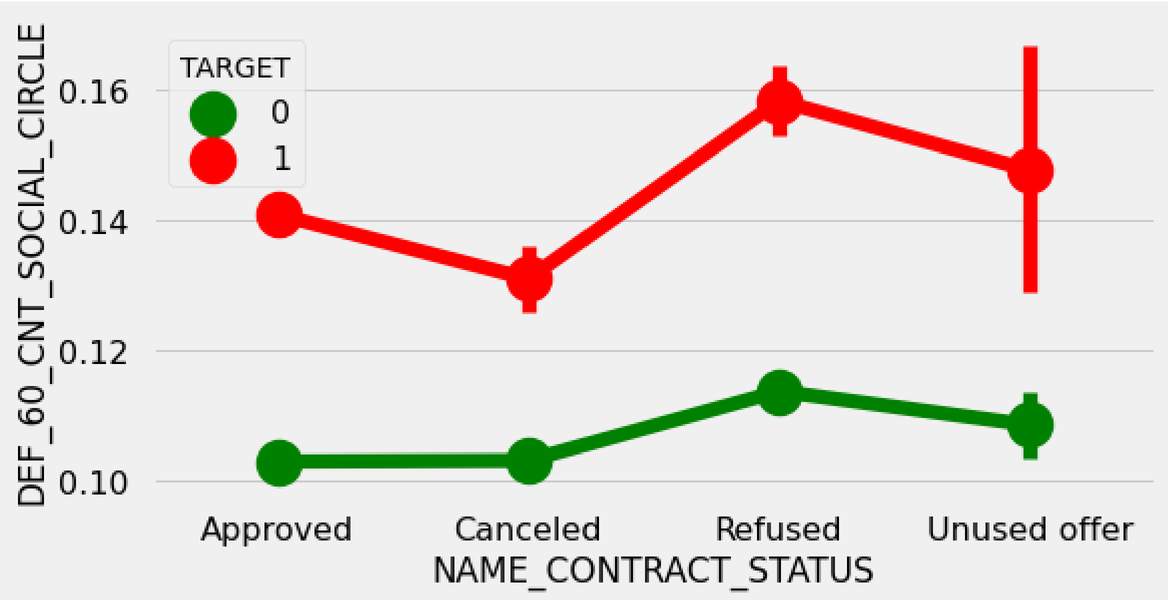
In [146]:

```
merged_pointplot("NAME_CONTRACT_STATUS",'AMT_INCOME_TOTAL')
```



In [147]:

```
merged_pointplot("NAME_CONTRACT_STATUS",'DEF_60_CNT_SOCIAL_CIRCLE')
```



In [ ]:

```

```