In [1]:
```
# Import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
# Load the previous dataset
previous_application=pd.read_csv('previous.csv')
```

In [3]:
```
# Initial data exploration
# Check the information of the previous.csv dataset
print(previous_application.info())
```

```
<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
None
```

In [4]:
```
# Initial data exploration
# Print the top 5 rows to check the data records
print(previous_application.head())
```

```
   SK_ID_PREV  SK_ID_CURR NAME_CONTRACT_TYPE  AMT_ANNUITY  AMT_APPLICATION
\
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

   AMT_CREDIT  AMT_DOWN_PAYMENT  AMT_GOODS_PRICE WEEKDAY_APPR_PROCESS_START
\
0     17145.0               0.0          17145.0                  SATURDAY
1    679671.0               NaN         607500.0                  THURSDAY
2    136444.5               NaN         112500.0                   TUESDAY
3    470790.0               NaN         450000.0                    MONDAY
4    404055.0               NaN         337500.0                  THURSDAY

   HOUR_APPR_PROCESS_START  ... NAME_SELLER_INDUSTRY  CNT_PAYMENT  \
0                       15  ...         Connectivity         12.0
1                       11  ...                  XNA         36.0
2                       11  ...                  XNA         12.0
3                        7  ...                  XNA         12.0
4                        9  ...                  XNA         24.0

   NAME_YIELD_GROUP       PRODUCT_COMBINATION  DAYS_FIRST_DRAWING  \
0            middle  POS mobile with interest            365243.0
1       low_action          Cash X-Sell: low            365243.0
2              high         Cash X-Sell: high            365243.0
3            middle       Cash X-Sell: middle            365243.0
4              high         Cash Street: high                 NaN

  DAYS_FIRST_DUE DAYS_LAST_DUE_1ST_VERSION  DAYS_LAST_DUE DAYS_TERMINATION
\
0          -42.0                     300.0         -42.0            -37.0
1         -134.0                     916.0      365243.0         365243.0
2         -271.0                      59.0      365243.0         365243.0
3         -482.0                    -152.0        -182.0            -177.0
4            NaN                       NaN           NaN              NaN

  NFLAG_INSURED_ON_APPROVAL
0                       0.0
1                       1.0
2                       1.0
3                       1.0
4                       NaN

[5 rows x 37 columns]
```

In [5]:
```
# Initial data exploration
# Print the bottom 5 rows to check the data records
print(previous_application.tail())
```

```
         SK_ID_PREV  SK_ID_CURR NAME_CONTRACT_TYPE  AMT_ANNUITY  \
1670209     2300464      352015     Consumer loans    14704.290
1670210     2357031      334635     Consumer loans     6622.020
1670211     2659632      249544     Consumer loans    11520.855
1670212     2785582      400317         Cash loans    18821.520
1670213     2418762      261212         Cash loans    16431.300

         AMT_APPLICATION  AMT_CREDIT  AMT_DOWN_PAYMENT  AMT_GOODS_PRICE  \
1670209         267295.5    311400.0               0.0         267295.5
1670210          87750.0     64291.5           29250.0          87750.0
1670211         105237.0    102523.5           10525.5         105237.0
1670212         180000.0    191880.0               NaN         180000.0
1670213         360000.0    360000.0               NaN         360000.0

        WEEKDAY_APPR_PROCESS_START  HOUR_APPR_PROCESS_START  ...  \
1670209                  WEDNESDAY                       12  ...
1670210                    TUESDAY                       15  ...
1670211                     MONDAY                       12  ...
1670212                  WEDNESDAY                        9  ...
1670213                     SUNDAY                       10  ...

         NAME_SELLER_INDUSTRY  CNT_PAYMENT  NAME_YIELD_GROUP  \
1670209             Furniture         30.0        low_normal
1670210             Furniture         12.0            middle
1670211  Consumer electronics         10.0        low_normal
1670212                   XNA         12.0        low_normal
1670213                   XNA         48.0            middle

                PRODUCT_COMBINATION  DAYS_FIRST_DRAWING DAYS_FIRST_DUE  \
1670209   POS industry with interest            365243.0         -508.0
1670210   POS industry with interest            365243.0        -1604.0
1670211  POS household with interest            365243.0        -1457.0
1670212             Cash X-Sell: low            365243.0        -1155.0
1670213          Cash X-Sell: middle            365243.0        -1163.0

        DAYS_LAST_DUE_1ST_VERSION  DAYS_LAST_DUE DAYS_TERMINATION  \
1670209                     362.0         -358.0           -351.0
1670210                   -1274.0        -1304.0          -1297.0
1670211                   -1187.0        -1187.0          -1181.0
1670212                    -825.0         -825.0           -817.0
1670213                     247.0         -443.0           -423.0

        NFLAG_INSURED_ON_APPROVAL
1670209                       0.0
1670210                       0.0
1670211                       0.0
1670212                       1.0
1670213                       0.0

[5 rows x 37 columns]
```

In [6]:
```
# Initial data exploration
# Print the Statistical Summary of the dataset.
print(previous_application.describe())
```

```
        SK_ID_PREV    SK_ID_CURR   AMT_ANNUITY  AMT_APPLICATION  \
count  1.670214e+06  1.670214e+06  1.297979e+06     1.670214e+06
mean   1.923089e+06  2.783572e+05  1.595512e+04     1.752339e+05
std    5.325980e+05  1.028148e+05  1.478214e+04     2.927798e+05
min    1.000001e+06  1.000010e+05  0.000000e+00     0.000000e+00
25%    1.461857e+06  1.893290e+05  6.321780e+03     1.872000e+04
50%    1.923110e+06  2.787145e+05  1.125000e+04     7.104600e+04
75%    2.384280e+06  3.675140e+05  2.065842e+04     1.803600e+05
max    2.845382e+06  4.562550e+05  4.180581e+05     6.905160e+06

        AMT_CREDIT  AMT_DOWN_PAYMENT  AMT_GOODS_PRICE  \
count  1.670213e+06     7.743700e+05     1.284699e+06
mean   1.961140e+05     6.697402e+03     2.278473e+05
std    3.185746e+05     2.092150e+04     3.153966e+05
min    0.000000e+00    -9.000000e-01     0.000000e+00
25%    2.416050e+04     0.000000e+00     5.084100e+04
50%    8.054100e+04     1.638000e+03     1.123200e+05
75%    2.164185e+05     7.740000e+03     2.340000e+05
max    6.905160e+06     3.060045e+06     6.905160e+06

       HOUR_APPR_PROCESS_START  NFLAG_LAST_APPL_IN_DAY  RATE_DOWN_PAYMENT  \
count             1.670214e+06            1.670214e+06      774370.000000
mean              1.248418e+01            9.964675e-01           0.079637
std               3.334028e+00            5.932963e-02           0.107823
min               0.000000e+00            0.000000e+00          -0.000015
25%               1.000000e+01            1.000000e+00           0.000000
50%               1.200000e+01            1.000000e+00           0.051605
75%               1.500000e+01            1.000000e+00           0.108909
max               2.300000e+01            1.000000e+00           1.000000

       ...  RATE_INTEREST_PRIVILEGED  DAYS_DECISION  SELLERPLACE_AREA  \
count  ...                5951.000000   1.670214e+06      1.670214e+06
mean   ...                   0.773503  -8.806797e+02      3.139511e+02
std    ...                   0.100879   7.790997e+02      7.127443e+03
min    ...                   0.373150  -2.922000e+03     -1.000000e+00
25%    ...                   0.715645  -1.300000e+03     -1.000000e+00
50%    ...                   0.835095  -5.810000e+02      3.000000e+00
75%    ...                   0.852537  -2.800000e+02      8.200000e+01
max    ...                   1.000000  -1.000000e+00      4.000000e+06

        CNT_PAYMENT  DAYS_FIRST_DRAWING  DAYS_FIRST_DUE  \
count  1.297984e+06       997149.000000   997149.000000
mean   1.605408e+01       342209.855039    13826.269337
std    1.456729e+01        88916.115834    72444.869708
min    0.000000e+00        -2922.000000    -2892.000000
25%    6.000000e+00       365243.000000    -1628.000000
```

```
50%    1.200000e+01       365243.000000     -831.000000
75%    2.400000e+01       365243.000000     -411.000000
max    8.400000e+01       365243.000000   365243.000000

       DAYS_LAST_DUE_1ST_VERSION  DAYS_LAST_DUE  DAYS_TERMINATION  \
count              997149.000000  997149.000000     997149.000000
mean                33767.774054   76582.403064      81992.343838
std                106857.034789  149647.415123     153303.516729
min                 -2801.000000   -2889.000000      -2874.000000
25%                 -1242.000000   -1314.000000      -1270.000000
50%                  -361.000000    -537.000000       -499.000000
75%                   129.000000     -74.000000        -44.000000
max                365243.000000  365243.000000     365243.000000

       NFLAG_INSURED_ON_APPROVAL
count              997149.000000
mean                    0.332570
std                     0.471134
min                     0.000000
25%                     0.000000
50%                     0.000000
75%                     1.000000
max                     1.000000

[8 rows x 21 columns]
```

In [7]:
```
# Data cleaning
# Check for missing values
missing_values = previous_application.isnull().sum()
print(missing_values)
```

```
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 [9]:
```
# Data cleaning
# Identify the columns that have more than 80% of missing values and drop th
# List of columns to drop
columns_to_drop = [
    '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'
]
# Drop the identified columns
previous_application.drop(columns=columns_to_drop, inplace=True)
# Display the remaining columns
print("Remaining columns after dropping:")
print(previous_application.columns.tolist())
```

```
Remaining columns after dropping:
['SK_ID_PREV', 'SK_ID_CURR', 'NAME_CONTRACT_TYPE', 'AMT_ANNUITY', 'AMT_APPLI
CATION', 'AMT_CREDIT', 'AMT_GOODS_PRICE', 'WEEKDAY_APPR_PROCESS_START', 'HOU
R_APPR_PROCESS_START', 'FLAG_LAST_APPL_PER_CONTRACT', 'NFLAG_LAST_APPL_IN_DA
Y', 'NAME_CASH_LOAN_PURPOSE', 'NAME_CONTRACT_STATUS', 'DAYS_DECISION', 'NAME
_PAYMENT_TYPE', 'CODE_REJECT_REASON', 'NAME_CLIENT_TYPE', 'NAME_GOODS_CATEGO
RY', 'NAME_PORTFOLIO', 'NAME_PRODUCT_TYPE', 'CHANNEL_TYPE', 'SELLERPLACE_ARE
A', 'NAME_SELLER_INDUSTRY', 'CNT_PAYMENT', 'NAME_YIELD_GROUP', 'PRODUCT_COMB
INATION']
```

In [10]:
```
# Data cleaning
# Identify the columns that have missing values but less than 80% and fill t
# Columns with Missing Values (Less than 80%):
# AMT_ANNUITY: 372,235 missing values
# AMT_CREDIT: 1 missing value
# AMT_GOODS_PRICE: 385,515 missing values
# CNT_PAYMENT: 372,230 missing values
# PRODUCT_COMBINATION: 346 missing values

# Fill missing values with a constant (e.g., 0 for numerical columns)
previous_application['AMT_ANNUITY'] = previous_application['AMT_ANNUITY'].fi
previous_application['AMT_CREDIT'] = previous_application['AMT_CREDIT'].fill
previous_application['AMT_GOODS_PRICE'] = previous_application['AMT_GOODS_PR
previous_application['CNT_PAYMENT'] = previous_application['CNT_PAYMENT'].fi
previous_application['PRODUCT_COMBINATION'] = previous_application['PRODUCT_

# Verify that the missing values have been filled
print(previous_application.isnull().sum())
```

```
SK_ID_PREV                    0
SK_ID_CURR                    0
NAME_CONTRACT_TYPE            0
AMT_ANNUITY                   0
AMT_APPLICATION               0
AMT_CREDIT                    0
AMT_GOODS_PRICE               0
WEEKDAY_APPR_PROCESS_START    0
HOUR_APPR_PROCESS_START       0
FLAG_LAST_APPL_PER_CONTRACT   0
NFLAG_LAST_APPL_IN_DAY        0
NAME_CASH_LOAN_PURPOSE        0
NAME_CONTRACT_STATUS          0
DAYS_DECISION                 0
NAME_PAYMENT_TYPE             0
CODE_REJECT_REASON            0
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                   0
NAME_YIELD_GROUP              0
PRODUCT_COMBINATION           0
dtype: int64
```

In [11]:
```
# Confirm that there are no missing values left
print("Missing values after filling:", previous_application.isnull().sum())
```

```
Missing values after filling: SK_ID_PREV                     0
SK_ID_CURR                     0
NAME_CONTRACT_TYPE             0
AMT_ANNUITY                    0
AMT_APPLICATION                0
AMT_CREDIT                     0
AMT_GOODS_PRICE                0
WEEKDAY_APPR_PROCESS_START     0
HOUR_APPR_PROCESS_START        0
FLAG_LAST_APPL_PER_CONTRACT    0
NFLAG_LAST_APPL_IN_DAY         0
NAME_CASH_LOAN_PURPOSE         0
NAME_CONTRACT_STATUS           0
DAYS_DECISION                  0
NAME_PAYMENT_TYPE              0
CODE_REJECT_REASON             0
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                    0
NAME_YIELD_GROUP               0
PRODUCT_COMBINATION            0
dtype: int64
```

In [ ]:
```
# No missing values left
```

In [12]:
```
# Data cleaning
# Find duplicate rows
duplicates = previous_application[previous_application.duplicated()]

# Display the number of duplicate rows
print(f'Number of duplicate rows: {duplicates.shape[0]}')

# Display the duplicate rows
print(duplicates)
```

```
Number of duplicate rows: 0
Empty DataFrame
Columns: [SK_ID_PREV, SK_ID_CURR, NAME_CONTRACT_TYPE, AMT_ANNUITY, AMT_APPLI
CATION, AMT_CREDIT, AMT_GOODS_PRICE, WEEKDAY_APPR_PROCESS_START, HOUR_APPR_P
ROCESS_START, FLAG_LAST_APPL_PER_CONTRACT, NFLAG_LAST_APPL_IN_DAY, NAME_CASH
_LOAN_PURPOSE, NAME_CONTRACT_STATUS, DAYS_DECISION, NAME_PAYMENT_TYPE, CODE_
REJECT_REASON, NAME_CLIENT_TYPE, NAME_GOODS_CATEGORY, NAME_PORTFOLIO, NAME_P
RODUCT_TYPE, CHANNEL_TYPE, SELLERPLACE_AREA, NAME_SELLER_INDUSTRY, CNT_PAYME
NT, NAME_YIELD_GROUP, PRODUCT_COMBINATION]
Index: []

[0 rows x 26 columns]
```

In [13]:
```
# No duplicate rows have been found
```

In [14]:
```
# Data cleaning
# Convert columns: Numerical columns
previous_application['AMT_ANNUITY'] = pd.to_numeric(previous_application['AM
previous_application['AMT_APPLICATION'] = pd.to_numeric(previous_applicatior
previous_application['AMT_CREDIT'] = pd.to_numeric(previous_application['AMT
previous_application['AMT_GOODS_PRICE'] = pd.to_numeric(previous_applicatior
previous_application['CNT_PAYMENT'] = previous_application['CNT_PAYMENT'].as
previous_application['SK_ID_PREV'] = previous_application['SK_ID_PREV'].asty
previous_application['SK_ID_CURR'] = previous_application['SK_ID_CURR'].asty
```

In [15]:
```
# Data cleaning
# Convert columns: Categorical columns
categorical_cols = [
    'NAME_CONTRACT_TYPE',
    'NAME_CASH_LOAN_PURPOSE',
    'NAME_CONTRACT_STATUS',
    'NAME_PAYMENT_TYPE',
    'CODE_REJECT_REASON',
    'NAME_CLIENT_TYPE',
    'NAME_GOODS_CATEGORY',
    'NAME_PORTFOLIO',
    'NAME_PRODUCT_TYPE',
    'CHANNEL_TYPE',
    'NAME_SELLER_INDUSTRY',
    'NAME_YIELD_GROUP',
    'PRODUCT_COMBINATION'
]

for col in categorical_cols:
    previous_application[col] = previous_application[col].astype('category')
```

In [16]:
```
# Data cleaning
# Check that the columns have been converted
print(previous_application.dtypes)
```

```
SK_ID_PREV                      int64
SK_ID_CURR                      int64
NAME_CONTRACT_TYPE           category
AMT_ANNUITY                   float64
AMT_APPLICATION               float64
AMT_CREDIT                    float64
AMT_GOODS_PRICE               float64
WEEKDAY_APPR_PROCESS_START     object
HOUR_APPR_PROCESS_START         int64
FLAG_LAST_APPL_PER_CONTRACT    object
NFLAG_LAST_APPL_IN_DAY          int64
NAME_CASH_LOAN_PURPOSE       category
NAME_CONTRACT_STATUS         category
DAYS_DECISION                   int64
NAME_PAYMENT_TYPE            category
CODE_REJECT_REASON           category
NAME_CLIENT_TYPE             category
NAME_GOODS_CATEGORY          category
NAME_PORTFOLIO               category
NAME_PRODUCT_TYPE            category
CHANNEL_TYPE                 category
SELLERPLACE_AREA                int64
NAME_SELLER_INDUSTRY         category
CNT_PAYMENT                     int64
NAME_YIELD_GROUP             category
PRODUCT_COMBINATION          category
dtype: object
```

In [17]:
```
# Load the application dataset
application = pd.read_csv('application_data.csv')  # Load application data

# Segregate loan defaulters and non-defaulters based on the TARGET variable
had_difficulties = application[application['TARGET'] == 1]  # Default = had_
had_no_difficulties = application[application['TARGET'] == 0]  # Non-default
# Display the first few rows of each group
print("Defaulters:")
print(had_difficulties.head())
print("\nNon-defaulters:")
print(had_no_difficulties.head())
```

```
Defaulters:
    SK_ID_CURR  TARGET NAME_CONTRACT_TYPE CODE_GENDER FLAG_OWN_CAR  \
0       100002       1         Cash loans           M            N
26      100031       1         Cash loans           F            N
40      100047       1         Cash loans           M            N
42      100049       1         Cash loans           F            N
81      100096       1         Cash loans           F            N

   FLAG_OWN_REALTY  CNT_CHILDREN  AMT_INCOME_TOTAL  AMT_CREDIT  AMT_ANNUITY
\
```

```
0                Y             0          202500.0    406597.5      24700.5
26               Y             0          112500.0    979992.0      27076.5
40               Y             0          202500.0   1193580.0      35028.0
42               N             0          135000.0    288873.0      16258.5
81               Y             0           81000.0    252000.0      14593.5

    ...  FLAG_DOCUMENT_18 FLAG_DOCUMENT_19 FLAG_DOCUMENT_20 FLAG_DOCUMENT_21
\
0   ...                0                0                0                0
26  ...                0                0                0                0
40  ...                0                0                0                0
42  ...                0                0                0                0
81  ...                0                0                0                0

   AMT_REQ_CREDIT_BUREAU_HOUR AMT_REQ_CREDIT_BUREAU_DAY  \
0                         0.0                        0.0
26                        0.0                        0.0
40                        0.0                        0.0
42                        0.0                        0.0
81                        0.0                        0.0

    AMT_REQ_CREDIT_BUREAU_WEEK  AMT_REQ_CREDIT_BUREAU_MON  \
0                          0.0                        0.0
26                         0.0                        0.0
40                         0.0                        2.0
42                         0.0                        0.0
81                         0.0                        0.0

    AMT_REQ_CREDIT_BUREAU_QRT  AMT_REQ_CREDIT_BUREAU_YEAR
0                         0.0                        1.0
26                        2.0                        2.0
40                        0.0                        4.0
42                        0.0                        2.0
81                        0.0                        0.0

[5 rows x 122 columns]

Non-defaulters:
   SK_ID_CURR  TARGET NAME_CONTRACT_TYPE CODE_GENDER FLAG_OWN_CAR  \
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      100008       0         Cash loans           M            N

  FLAG_OWN_REALTY  CNT_CHILDREN  AMT_INCOME_TOTAL  AMT_CREDIT  AMT_ANNUITY
\
1               N             0          270000.0   1293502.5      35698.5
2               Y             0           67500.0    135000.0       6750.0
3               Y             0          135000.0    312682.5      29686.5
```

```
4               Y             0          121500.0    513000.0      21865.5
5               Y             0           99000.0    490495.5      27517.5

   ...  FLAG_DOCUMENT_18 FLAG_DOCUMENT_19 FLAG_DOCUMENT_20 FLAG_DOCUMENT_21
\
1  ...                 0                0                0                0
2  ...                 0                0                0                0
3  ...                 0                0                0                0
4  ...                 0                0                0                0
5  ...                 0                0                0                0

  AMT_REQ_CREDIT_BUREAU_HOUR AMT_REQ_CREDIT_BUREAU_DAY  \
1                        0.0                       0.0
2                        0.0                       0.0
3                        NaN                       NaN
4                        0.0                       0.0
5                        0.0                       0.0

   AMT_REQ_CREDIT_BUREAU_WEEK  AMT_REQ_CREDIT_BUREAU_MON  \
1                         0.0                        0.0
2                         0.0                        0.0
3                         NaN                        NaN
4                         0.0                        0.0
5                         0.0                        0.0

   AMT_REQ_CREDIT_BUREAU_QRT  AMT_REQ_CREDIT_BUREAU_YEAR
1                        0.0                       0.0
2                        0.0                       0.0
3                        NaN                       NaN
4                        0.0                       0.0
5                        1.0                       1.0

[5 rows x 122 columns]
```

In [18]:
```
# Descriptive statistics for both groups
print("\nDescriptive statistics for Defaulters:")
print(had_difficulties.describe())
print("\nDescriptive statistics for Non-defaulters:")
print(had_no_difficulties.describe())
```

```
Descriptive statistics for Defaulters:
          SK_ID_CURR   TARGET  CNT_CHILDREN  AMT_INCOME_TOTAL    AMT_CREDIT
\
count   24825.000000  24825.0  24825.000000      2.482500e+04  2.482500e+04
mean   277449.167936      1.0      0.463807      1.656118e+05  5.577785e+05
std    102383.123458      0.0      0.756903      7.466770e+05  3.464332e+05
min    100002.000000      1.0      0.000000      2.565000e+04  4.500000e+04
25%    189555.000000      1.0      0.000000      1.125000e+05  2.844000e+05
50%    276291.000000      1.0      0.000000      1.350000e+05  4.975200e+05
75%    366050.000000      1.0      1.000000      2.025000e+05  7.333155e+05
```

```
max    456254.000000      1.0     11.000000      1.170000e+08  4.027680e+06

         AMT_ANNUITY  AMT_GOODS_PRICE  REGION_POPULATION_RELATIVE  \
count   24825.000000     2.480400e+04                24825.000000
mean    26481.744290     4.889724e+05                    0.019131
std     12450.676999     3.116365e+05                    0.011905
min      2722.500000     4.500000e+04                    0.000533
25%     17361.000000     2.385000e+05                    0.009630
50%     25263.000000     4.500000e+05                    0.018634
75%     32976.000000     6.750000e+05                    0.025164
max    149211.000000     3.600000e+06                    0.072508

         DAYS_BIRTH  DAYS_EMPLOYED  ...  FLAG_DOCUMENT_18  FLAG_DOCUMENT_19
\
count  24825.000000   24825.000000  ...      24825.000000      24825.000000
mean  -14884.828077   42394.675448  ...          0.005720          0.000483
std     4192.844583  119484.634253  ...          0.075416          0.021981
min   -25168.000000  -16069.000000  ...          0.000000          0.000000
25%   -18037.000000   -2156.000000  ...          0.000000          0.000000
50%   -14282.000000   -1034.000000  ...          0.000000          0.000000
75%   -11396.000000    -379.000000  ...          0.000000          0.000000
max    -7678.000000  365243.000000  ...          1.000000          1.000000

       FLAG_DOCUMENT_20  FLAG_DOCUMENT_21  AMT_REQ_CREDIT_BUREAU_HOUR  \
count      24825.000000      24825.000000                20533.000000
mean           0.000524          0.000564                    0.006672
std            0.022878          0.023741                    0.084926
min            0.000000          0.000000                    0.000000
25%            0.000000          0.000000                    0.000000
50%            0.000000          0.000000                    0.000000
75%            0.000000          0.000000                    0.000000
max            1.000000          1.000000                    2.000000

       AMT_REQ_CREDIT_BUREAU_DAY  AMT_REQ_CREDIT_BUREAU_WEEK  \
count               20533.000000                20533.000000
mean                   0.008036                    0.034919
std                    0.106682                    0.203941
min                    0.000000                    0.000000
25%                    0.000000                    0.000000
50%                    0.000000                    0.000000
75%                    0.000000                    0.000000
max                    4.000000                    6.000000

       AMT_REQ_CREDIT_BUREAU_MON  AMT_REQ_CREDIT_BUREAU_QRT  \
count               20533.000000                20533.000000
mean                   0.227926                    0.259923
std                    0.745116                    0.643789
min                    0.000000                    0.000000
25%                    0.000000                    0.000000
50%                    0.000000                    0.000000
```

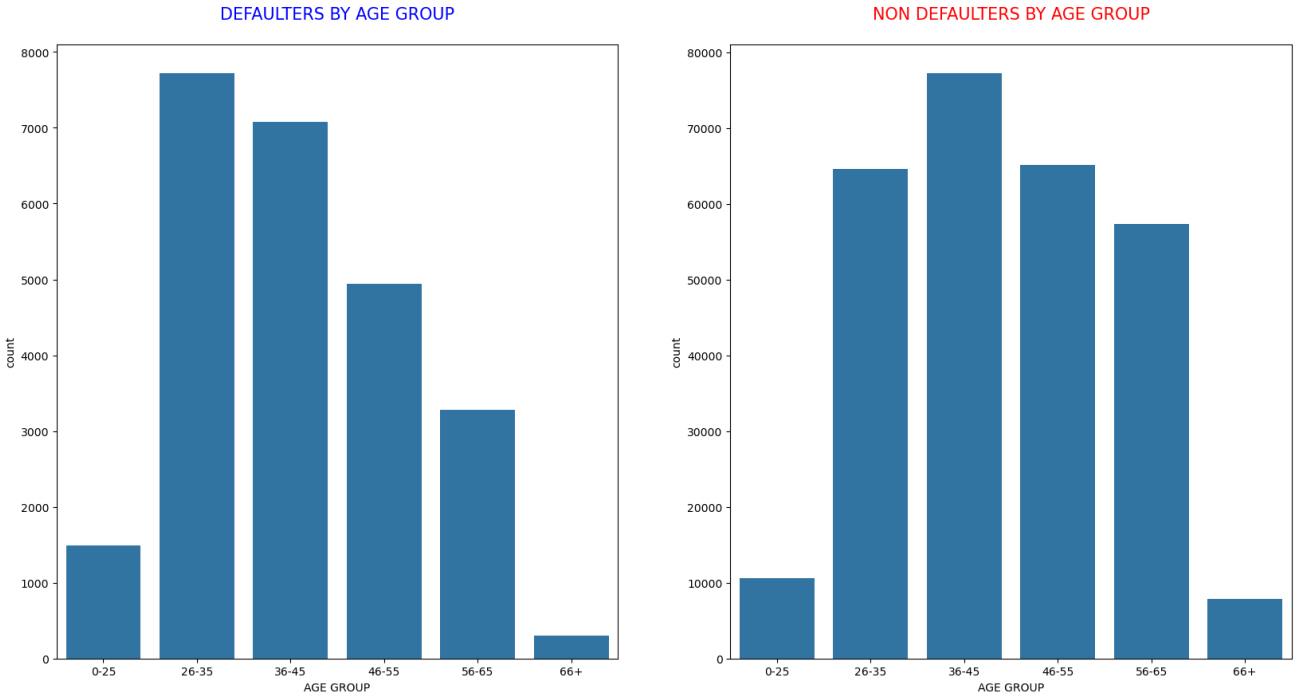
```
75%                     0.000000                    0.000000
max                    17.000000                   19.000000

       AMT_REQ_CREDIT_BUREAU_YEAR
count                20533.000000
mean                     2.028783
std                      1.934063
min                      0.000000
25%                      1.000000
50%                      2.000000
75%                      3.000000
max                     22.000000

[8 rows x 106 columns]

Descriptive statistics for Non-defaulters:
          SK_ID_CURR    TARGET   CNT_CHILDREN  AMT_INCOME_TOTAL    AMT_CREDI
T  \
count  282686.000000  282686.0  282686.000000      2.826860e+05  2.826860e+0
5
mean   278244.744536       0.0       0.412946      1.690777e+05  6.026483e+0
5
std    102825.776954       0.0       0.718843      1.104763e+05  4.068459e+0
5
min    100003.000000       0.0       0.000000      2.565000e+04  4.500000e+0
4
25%    189103.250000       0.0       0.000000      1.125000e+05  2.700000e+0
5
50%    278362.500000       0.0       0.000000      1.485000e+05  5.177880e+0
5
75%    367241.500000       0.0       1.000000      2.025000e+05  8.100000e+0
5
max    456255.000000       0.0      19.000000      1.800009e+07  4.050000e+0
6

         AMT_ANNUITY  AMT_GOODS_PRICE  REGION_POPULATION_RELATIVE  \
count  282674.000000     2.824290e+05                282686.000000
mean    27163.623349     5.427368e+05                     0.021021
std     14658.307178     3.737855e+05                     0.013978
min      1615.500000     4.050000e+04                     0.000290
25%     16456.500000     2.385000e+05                     0.010006
50%     24876.000000     4.500000e+05                     0.018850
75%     34749.000000     6.885000e+05                     0.028663
max    258025.500000     4.050000e+06                     0.072508

          DAYS_BIRTH  DAYS_EMPLOYED  ...  FLAG_DOCUMENT_18  FLAG_DOCUMENT_19
\
count  282686.000000  282686.000000  ...     282686.000000     282686.000000
mean   -16138.176397   65696.146123  ...          0.008341          0.000605
std      4364.200856  142877.810161  ...          0.090950          0.024588
```

```
min    -25229.000000  -17912.000000  ...          0.000000          0.000000
25%    -19793.750000   -2813.000000  ...          0.000000          0.000000
50%    -15877.000000   -1235.000000  ...          0.000000          0.000000
75%    -12536.000000    -278.000000  ...          0.000000          0.000000
max     -7489.000000  365243.000000  ...          1.000000          1.000000

       FLAG_DOCUMENT_20  FLAG_DOCUMENT_21  AMT_REQ_CREDIT_BUREAU_HOUR  \
count     282686.000000     282686.000000               245459.000000
mean           0.000506          0.000315                    0.006380
std            0.022486          0.017741                    0.083759
min            0.000000          0.000000                    0.000000
25%            0.000000          0.000000                    0.000000
50%            0.000000          0.000000                    0.000000
75%            0.000000          0.000000                    0.000000
max            1.000000          1.000000                    4.000000

       AMT_REQ_CREDIT_BUREAU_DAY  AMT_REQ_CREDIT_BUREAU_WEEK  \
count              245459.000000               245459.000000
mean                   0.006914                    0.034315
std                    0.111091                    0.204747
min                    0.000000                    0.000000
25%                    0.000000                    0.000000
50%                    0.000000                    0.000000
75%                    0.000000                    0.000000
max                    9.000000                    8.000000

       AMT_REQ_CREDIT_BUREAU_MON  AMT_REQ_CREDIT_BUREAU_QRT  \
count              245459.000000               245459.000000
mean                   0.270697                    0.265939
std                    0.928799                    0.805355
min                    0.000000                    0.000000
25%                    0.000000                    0.000000
50%                    0.000000                    0.000000
75%                    0.000000                    0.000000
max                   27.000000                  261.000000

       AMT_REQ_CREDIT_BUREAU_YEAR
count               245459.000000
mean                     1.889199
std                      1.863376
min                      0.000000
25%                      0.000000
50%                      1.000000
75%                      3.000000
max                     25.000000

[8 rows x 106 columns]
```

In [19]:
```
# Calculate age from DAYS_BIRTH
application['AGE'] = -application['DAYS_BIRTH'] // 365  # Convert days to ye
```

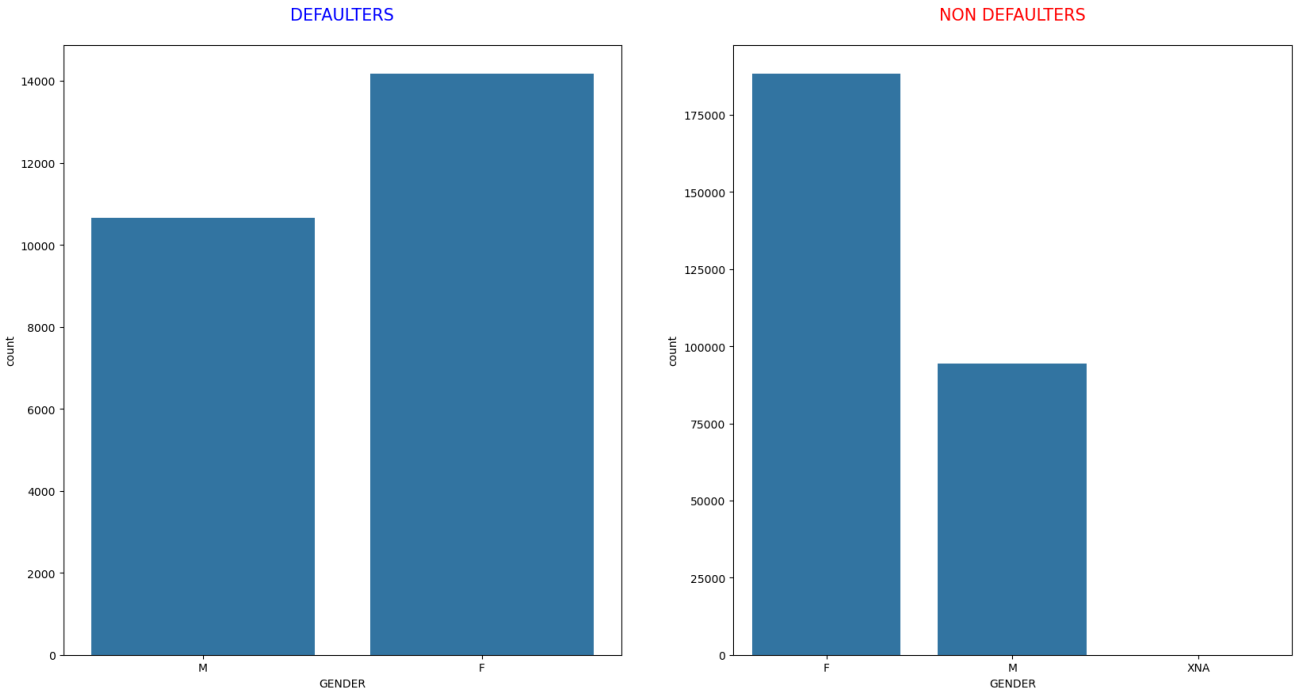
```
# Define bins and labels for age groups
bins = [0, 25, 35, 45, 55, 65, 100]
labels = ['0-25', '26-35', '36-45', '46-55', '56-65', '66+']

# Create age groups
application['AGE_GROUP'] = pd.cut(application['AGE'], bins=bins, labels=labe
```

In [20]:
```
# Analyze Loan defaults by Age Group
# Divide the data into two parts based on the TARGET variable
had_difficulties = application[application['TARGET'] == 1]  # Defaulters
had_no_difficulties = application[application['TARGET'] == 0]  # Non-default

# Plotting age group analysis
plt.figure(figsize=[20, 10])
plt.subplot(1, 2, 1)
sns.countplot(x='AGE_GROUP', data=had_difficulties)
plt.title('DEFAULTERS BY AGE GROUP\n', fontdict={'fontsize': 15, 'fontweight
plt.xlabel("AGE GROUP")

plt.subplot(1, 2, 2)
sns.countplot(x='AGE_GROUP', data=had_no_difficulties)
plt.title('NON DEFAULTERS BY AGE GROUP\n', fontdict={'fontsize': 15, 'fontwe
plt.xlabel("AGE GROUP")

plt.show()
```



In [21]:
```
# Analyze how loan defaults are affected by gender.
plt.figure(figsize=[20, 10])
plt.subplot(1, 2, 1)
sns.countplot(x='CODE_GENDER', data=had_difficulties)
```

```
plt.title('DEFAULTERS\n', fontdict={'fontsize': 15, 'fontweight': 5, 'color'
plt.xlabel("GENDER")

plt.subplot(1, 2, 2)
sns.countplot(x='CODE_GENDER', data=had_no_difficulties)
plt.title('NON DEFAULTERS\n', fontdict={'fontsize': 15, 'fontweight': 5, 'co
plt.xlabel("GENDER")
plt.show()
```



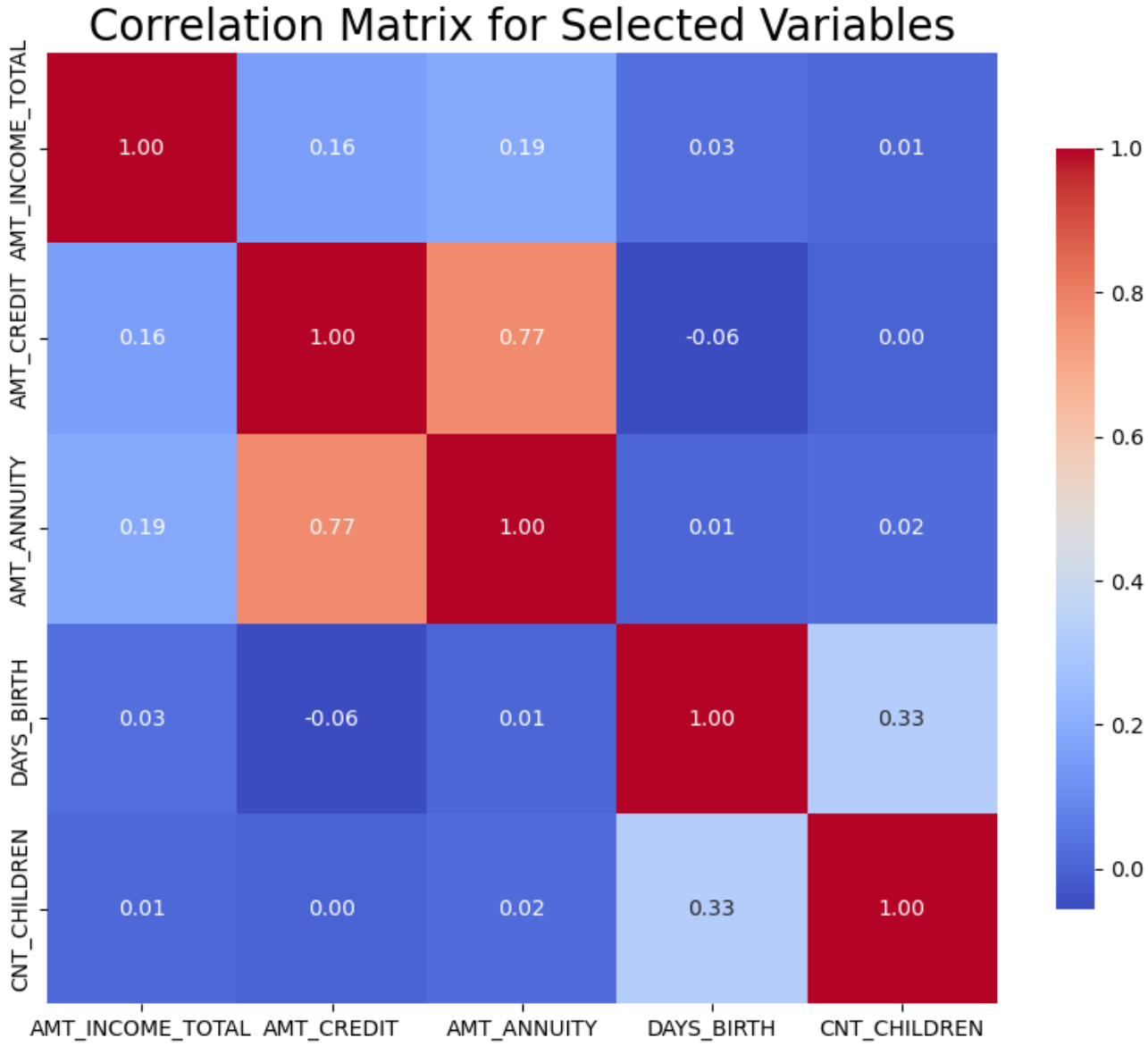
In [22]:
```
# Verify the structure of each data frame
print(had_difficulties.columns)
print(had_no_difficulties.columns)
```

```
Index(['SK_ID_CURR', 'TARGET', 'NAME_CONTRACT_TYPE', 'CODE_GENDER',
       'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'CNT_CHILDREN', 'AMT_INCOME_TOTA
L',
       'AMT_CREDIT', 'AMT_ANNUITY',
       ...
       'FLAG_DOCUMENT_20', 'FLAG_DOCUMENT_21', '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_CREDIT_BUREAU_YEAR', 'AGE', 'AGE_GROUP'],
      dtype='object', length=124)
Index(['SK_ID_CURR', 'TARGET', 'NAME_CONTRACT_TYPE', 'CODE_GENDER',
       'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'CNT_CHILDREN', 'AMT_INCOME_TOTA
L',
       'AMT_CREDIT', 'AMT_ANNUITY',
       ...
       'FLAG_DOCUMENT_20', 'FLAG_DOCUMENT_21', '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_CREDIT_BUREAU_YEAR', 'AGE', 'AGE_GROUP'],
      dtype='object', length=124)
```

In [23]:

```
# Load the application dataset
application_data = pd.read_csv('application_data.csv')
# Select only numeric columns
numeric_data = application_data.select_dtypes(include=['number'])
# Get the column names of the DataFrame
column_names = application_data.columns.tolist()
# Print the column names
print(column_names)
# Select only the specified numeric columns
subset_columns = ['AMT_INCOME_TOTAL', 'AMT_CREDIT', 'AMT_ANNUITY', 'DAYS_BIR
subset_data = application_data[subset_columns]

# Calculate the correlation matrix
corr_matrix_subset = subset_data.corr()

# Set up the matplotlib figure
plt.figure(figsize=[10, 8])

# Create a heatmap
sns.heatmap(corr_matrix_subset, annot=True, fmt=".2f", cmap='coolwarm', squa

# Add titles and labels
plt.title('Correlation Matrix for Selected Variables', fontsize=20)
plt.show()
```

```
['SK_ID_CURR', 'TARGET', 'NAME_CONTRACT_TYPE', 'CODE_GENDER', 'FLAG_OWN_CA
R', 'FLAG_OWN_REALTY', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL', 'AMT_CREDIT', 'AM
T_ANNUITY', 'AMT_GOODS_PRICE', 'NAME_TYPE_SUITE', 'NAME_INCOME_TYPE', 'NAME_
EDUCATION_TYPE', 'NAME_FAMILY_STATUS', 'NAME_HOUSING_TYPE', 'REGION_POPULATI
ON_RELATIVE', 'DAYS_BIRTH', 'DAYS_EMPLOYED', 'DAYS_REGISTRATION', 'DAYS_ID_P
UBLISH', 'OWN_CAR_AGE', 'FLAG_MOBIL', 'FLAG_EMP_PHONE', 'FLAG_WORK_PHONE', '
FLAG_CONT_MOBILE', 'FLAG_PHONE', 'FLAG_EMAIL', 'OCCUPATION_TYPE', 'CNT_FAM_M
EMBERS', 'REGION_RATING_CLIENT', 'REGION_RATING_CLIENT_W_CITY', 'WEEKDAY_APP
R_PROCESS_START', 'HOUR_APPR_PROCESS_START', 'REG_REGION_NOT_LIVE_REGION', '
REG_REGION_NOT_WORK_REGION', 'LIVE_REGION_NOT_WORK_REGION', 'REG_CITY_NOT_LI
VE_CITY', 'REG_CITY_NOT_WORK_CITY', 'LIVE_CITY_NOT_WORK_CITY', 'ORGANIZATION
_TYPE', 'EXT_SOURCE_1', 'EXT_SOURCE_2', 'EXT_SOURCE_3', 'APARTMENTS_AVG', 'B
ASEMENTAREA_AVG', 'YEARS_BEGINEXPLUATATION_AVG', 'YEARS_BUILD_AVG', 'COMMONA
REA_AVG', 'ELEVATORS_AVG', 'ENTRANCES_AVG', 'FLOORSMAX_AVG', 'FLOORSMIN_AV
G', 'LANDAREA_AVG', 'LIVINGAPARTMENTS_AVG', 'LIVINGAREA_AVG', 'NONLIVINGAPAR
TMENTS_AVG', 'NONLIVINGAREA_AVG', 'APARTMENTS_MODE', 'BASEMENTAREA_MODE', 'Y
EARS_BEGINEXPLUATATION_MODE', 'YEARS_BUILD_MODE', 'COMMONAREA_MODE', 'ELEVAT
ORS_MODE', 'ENTRANCES_MODE', 'FLOORSMAX_MODE', 'FLOORSMIN_MODE', 'LANDAREA_M
ODE', 'LIVINGAPARTMENTS_MODE', 'LIVINGAREA_MODE', 'NONLIVINGAPARTMENTS_MOD
E', 'NONLIVINGAREA_MODE', 'APARTMENTS_MEDI', 'BASEMENTAREA_MEDI', 'YEARS_BEG
INEXPLUATATION_MEDI', 'YEARS_BUILD_MEDI', 'COMMONAREA_MEDI', 'ELEVATORS_MED
I', 'ENTRANCES_MEDI', 'FLOORSMAX_MEDI', 'FLOORSMIN_MEDI', 'LANDAREA_MEDI', '
LIVINGAPARTMENTS_MEDI', 'LIVINGAREA_MEDI', 'NONLIVINGAPARTMENTS_MEDI', 'NONL
IVINGAREA_MEDI', 'FONDKAPREMONT_MODE', 'HOUSETYPE_MODE', 'TOTALAREA_MODE', '
WALLSMATERIAL_MODE', 'EMERGENCYSTATE_MODE', 'OBS_30_CNT_SOCIAL_CIRCLE', 'DEF
_30_CNT_SOCIAL_CIRCLE', 'OBS_60_CNT_SOCIAL_CIRCLE', 'DEF_60_CNT_SOCIAL_CIRCL
E', 'DAYS_LAST_PHONE_CHANGE', 'FLAG_DOCUMENT_2', 'FLAG_DOCUMENT_3', 'FLAG_DO
CUMENT_4', 'FLAG_DOCUMENT_5', 'FLAG_DOCUMENT_6', 'FLAG_DOCUMENT_7', 'FLAG_DO
CUMENT_8', 'FLAG_DOCUMENT_9', 'FLAG_DOCUMENT_10', 'FLAG_DOCUMENT_11', 'FLAG_
DOCUMENT_12', 'FLAG_DOCUMENT_13', 'FLAG_DOCUMENT_14', 'FLAG_DOCUMENT_15', 'F
LAG_DOCUMENT_16', 'FLAG_DOCUMENT_17', 'FLAG_DOCUMENT_18', 'FLAG_DOCUMENT_1
9', 'FLAG_DOCUMENT_20', 'FLAG_DOCUMENT_21', 'AMT_REQ_CREDIT_BUREAU_HOUR', 'A
MT_REQ_CREDIT_BUREAU_DAY', 'AMT_REQ_CREDIT_BUREAU_WEEK', 'AMT_REQ_CREDIT_BUR
EAU_MON', 'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_CREDIT_BUREAU_YEAR']
```



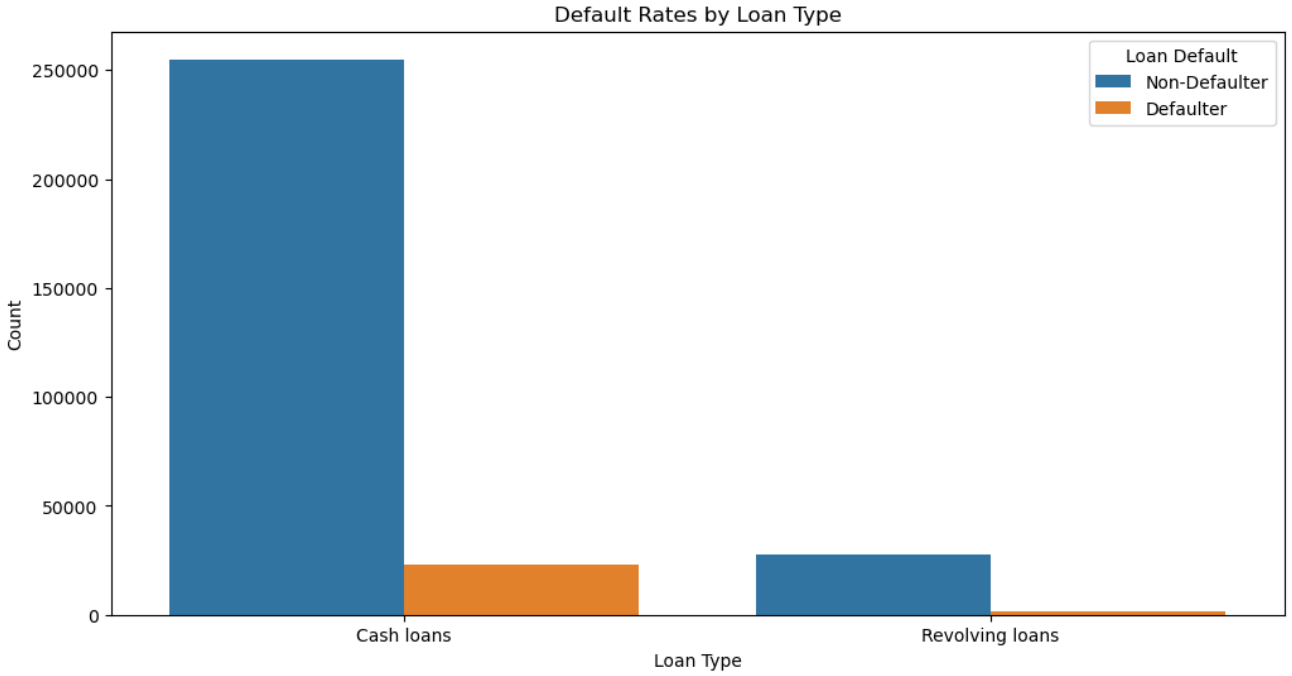
In [24]:
```
# Additional Analysis:
# Demographics of Defaulters
# Identify which age groups or income levels are associated with higher defa
age_groups = application['AGE_GROUP'].value_counts(normalize=True) * 100
defaulters_by_age = had_difficulties['AGE_GROUP'].value_counts(normalize=Tru

print("Defaulters by Age Group (%):")
print(defaulters_by_age)
```

```
Defaulters by Age Group (%):
AGE_GROUP
26-35    31.081571
36-45    28.531722
46-55    19.923464
56-65    13.220544
0-25      6.026183
66+       1.216516
Name: proportion, dtype: float64
```

In [25]:
```
# Analyze Income Levels:
# Calculate default rates by income group
# Define income bins
income_bins = [0, 20000, 40000, 60000, 80000, 100000, float('inf')]
income_labels = ['0-20k', '20k-40k', '40k-60k', '60k-80k', '80k-100k', '100k

# Create a new column for income groups
application_data['income_group'] = pd.cut(application_data['AMT_INCOME_TOTAL

# Calculate default rates by income group with observed=True
income_groups = application_data.groupby('income_group', observed=True)['TAR

# Print the default rates by income group
print("Default Rate by Income Group (%):")
print(income_groups)
```

```
Default Rate by Income Group (%):
income_group
20k-40k     8.307373
40k-60k     7.352747
60k-80k     8.138855
80k-100k    8.410623
100k+       8.038948
Name: TARGET, dtype: float64
```

In [26]:
```
# Loan Type
# Analyze if certain loan types have different default rates.
plt.figure(figsize=(12, 6))
sns.countplot(x='NAME_CONTRACT_TYPE', hue='TARGET', data=application)
plt.title('Default Rates by Loan Type')
plt.xlabel('Loan Type')
plt.ylabel('Count')
plt.legend(title='Loan Default', labels=['Non-Defaulter', 'Defaulter'])
plt.show()
```

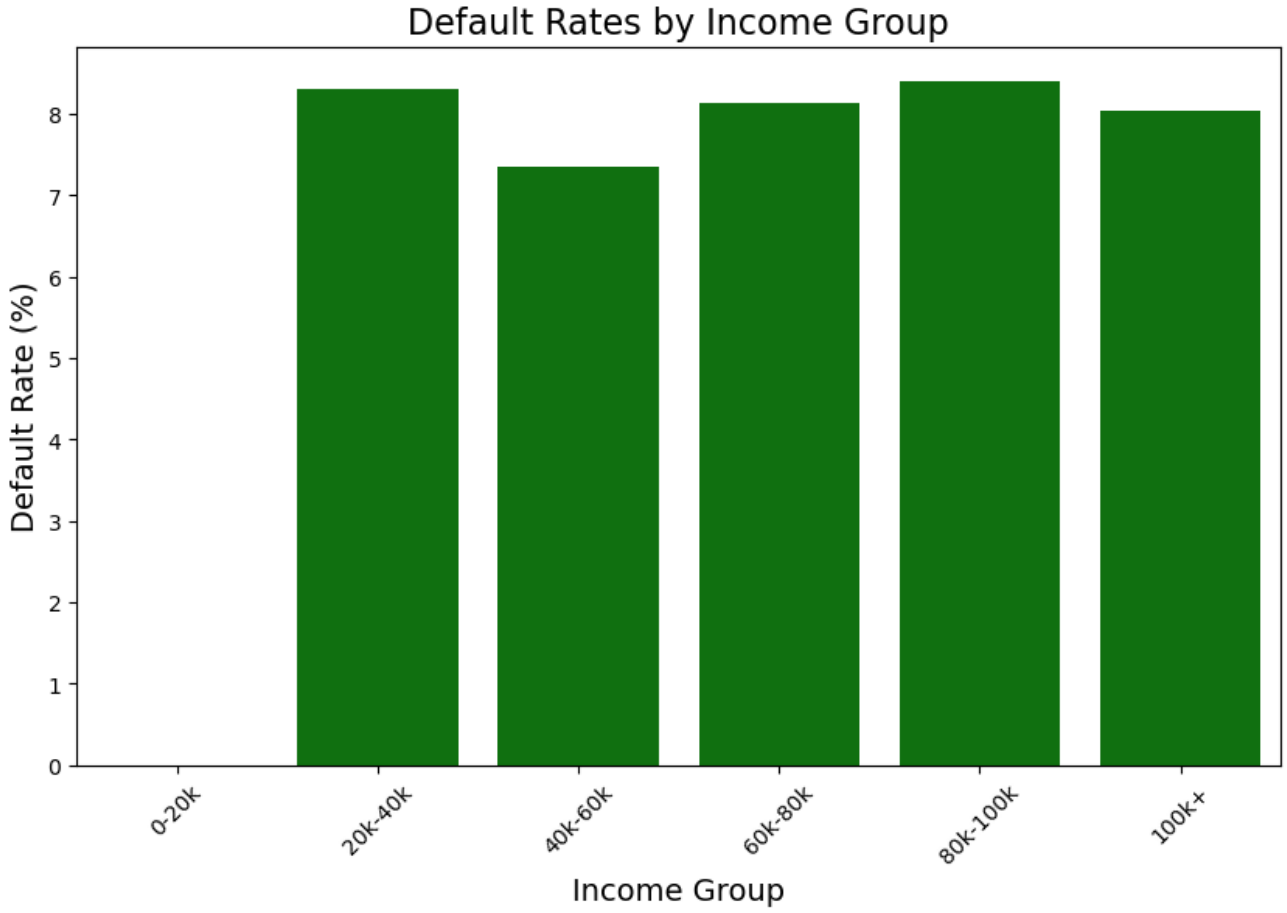


In [27]:
```
# Calculate Default Rates:
# Group by NAME_CONTRACT_TYPE and calculate the percentage of defaulters.
loan_default_rates = application.groupby('NAME_CONTRACT_TYPE')['TARGET'].mea
print("Default Rates by Loan Type (%):")
print(loan_default_rates)
```

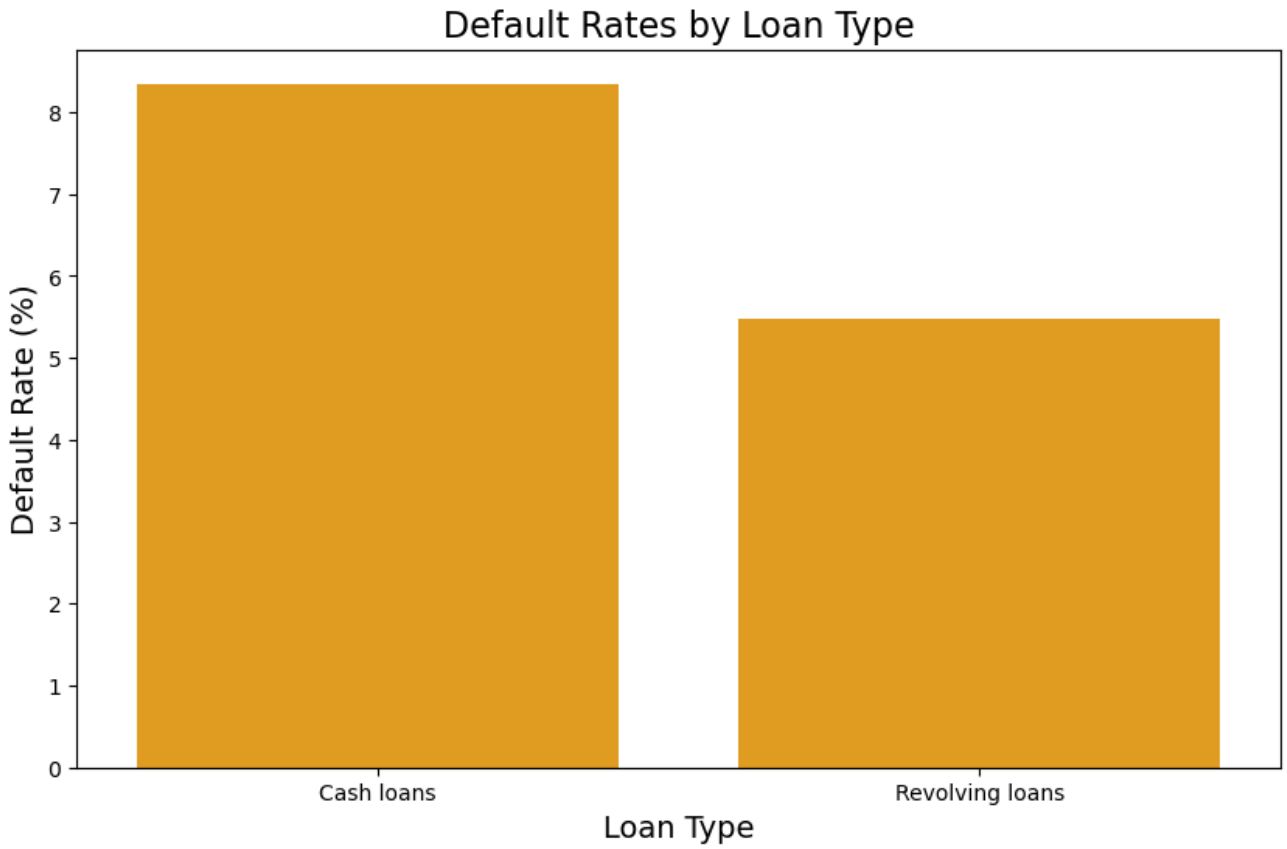
```
Default Rates by Loan Type (%):
NAME_CONTRACT_TYPE
Cash loans         8.345913
Revolving loans    5.478329
Name: TARGET, dtype: float64
```

In [28]:
```
# Bar Plot for "Default Rates by Income Group"
# Define income bins
income_bins = [0, 20000, 40000, 60000, 80000, 100000, float('inf')]
income_labels = ['0-20k', '20k-40k', '40k-60k', '60k-80k', '80k-100k', '100k

# Create a new column for income groups
application_data['income_group'] = pd.cut(application_data['AMT_INCOME_TOTAL

# Calculate default rates by income group
income_data = application_data.groupby('income_group', observed=True)['TARGE

# Reset index to prepare for plotting
income_data = income_data.reset_index()
income_data.columns = ['Income Group', 'Default Rate (%)']

# Drop NaN values
income_data = income_data.dropna()
```

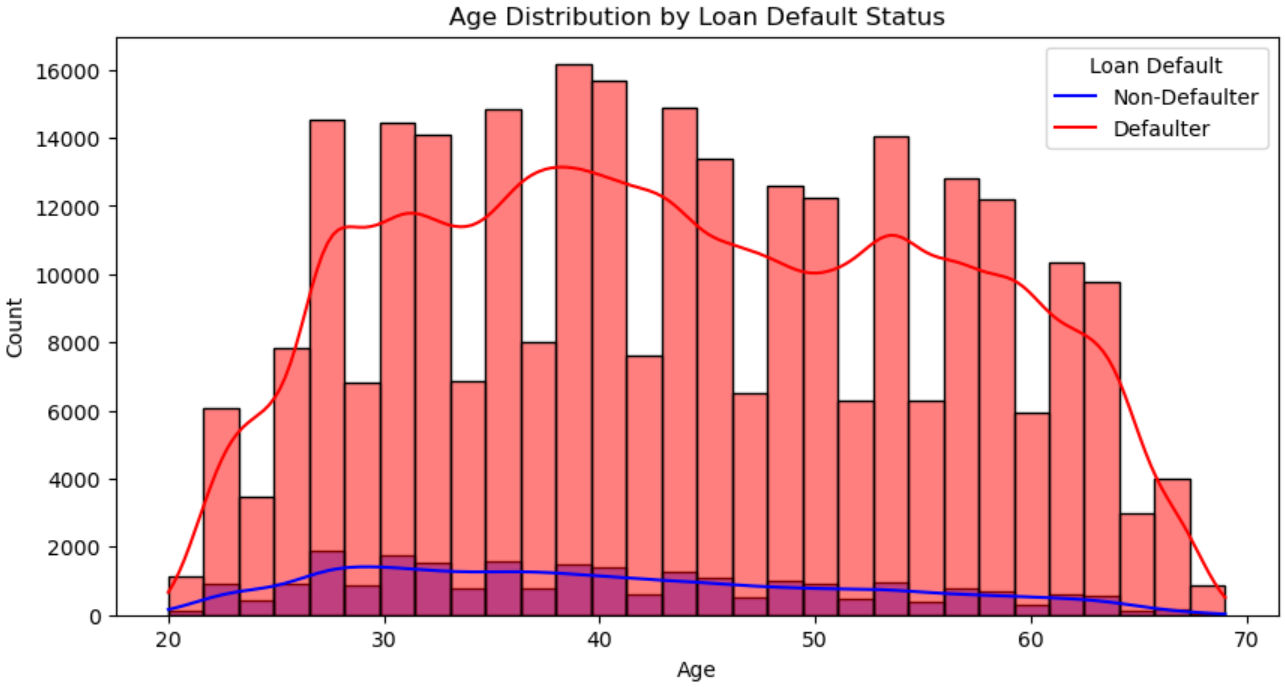
```
# Plot the bar chart
plt.figure(figsize=(10, 6))
sns.barplot(x='Income Group', y='Default Rate (%)', data=income_data, color=
plt.title('Default Rates by Income Group', fontsize=16)
plt.xlabel('Income Group', fontsize=14)
plt.ylabel('Default Rate (%)', fontsize=14)
plt.xticks(rotation=45)  # Rotate x labels for better visibility
plt.show()
```



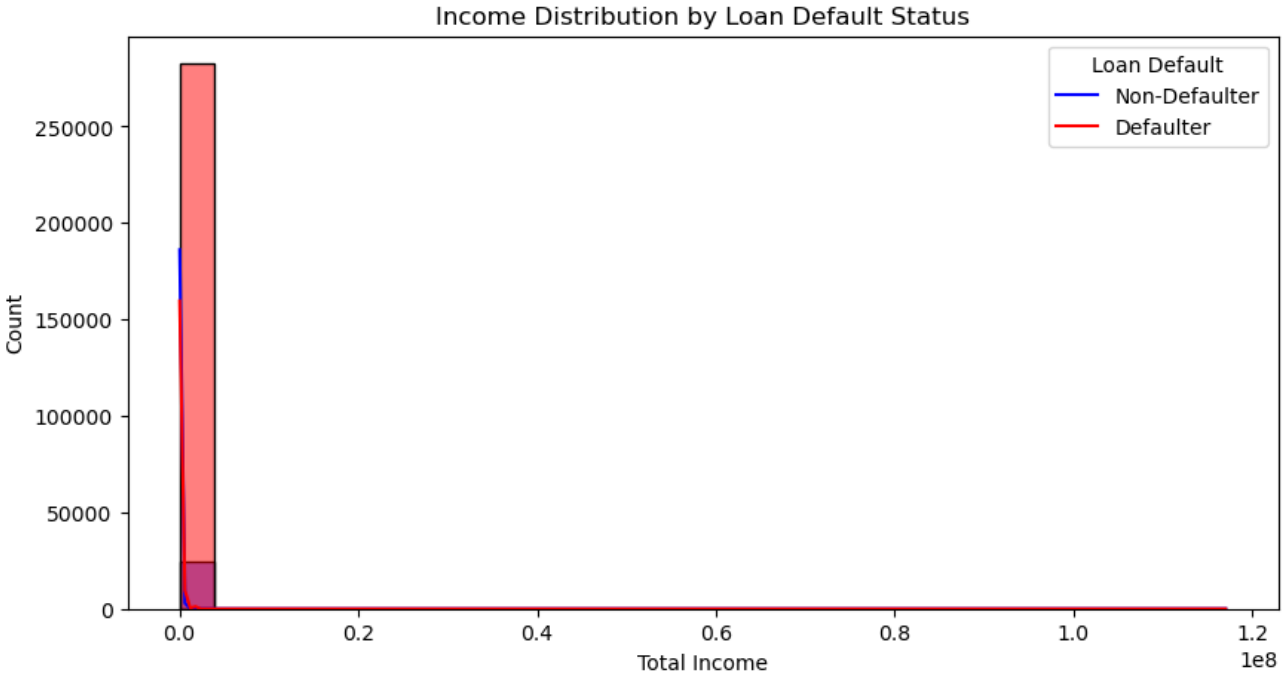
In [29]:
```
# Bar plot for "Default rates by Loan type"
loan_data = application.groupby('NAME_CONTRACT_TYPE')['TARGET'].mean() * 100
loan_data = loan_data.reset_index()  # Convert Series to DataFrame
loan_data.columns = ['Loan Type', 'Default Rate (%)']  # Rename columns
plt.figure(figsize=(10, 6))
sns.barplot(x='Loan Type', y='Default Rate (%)', data=loan_data, color='orar
plt.title('Default Rates by Loan Type', fontsize=16)
plt.xlabel('Loan Type', fontsize=14)
plt.ylabel('Default Rate (%)', fontsize=14)
plt.show()
```



In [30]:
```
# Histogram for "Age"
plt.figure(figsize=(10, 5))
sns.histplot(data=application, x='AGE', hue='TARGET', bins=30, palette=['red
plt.title('Age Distribution by Loan Default Status')
plt.xlabel('Age')
plt.ylabel('Count')
plt.legend(title='Loan Default', labels=['Non-Defaulter', 'Defaulter'])
plt.show()
```

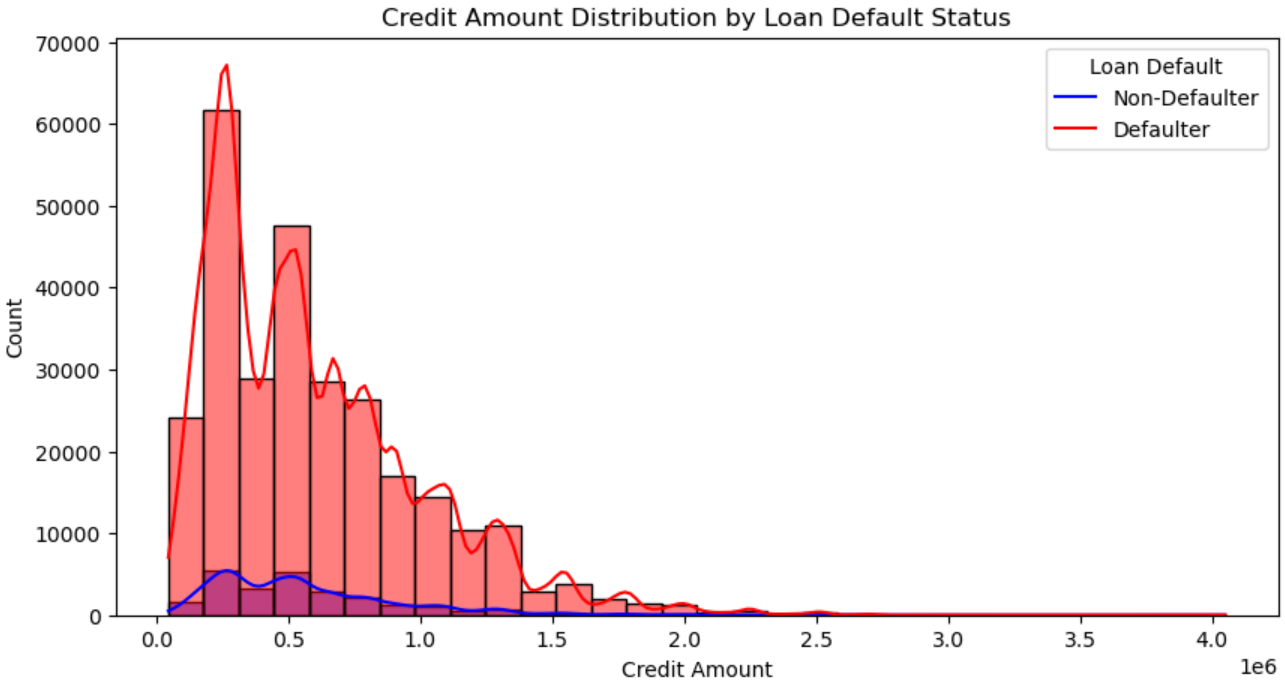


In [31]:
```
# Visualize the distribution of total income to see how it relates to loan d
plt.figure(figsize=(10, 5))
sns.histplot(data=application, x='AMT_INCOME_TOTAL', hue='TARGET', bins=30,
plt.title('Income Distribution by Loan Default Status')
plt.xlabel('Total Income')
plt.ylabel('Count')
plt.legend(title='Loan Default', labels=['Non-Defaulter', 'Defaulter'])
plt.show()
```

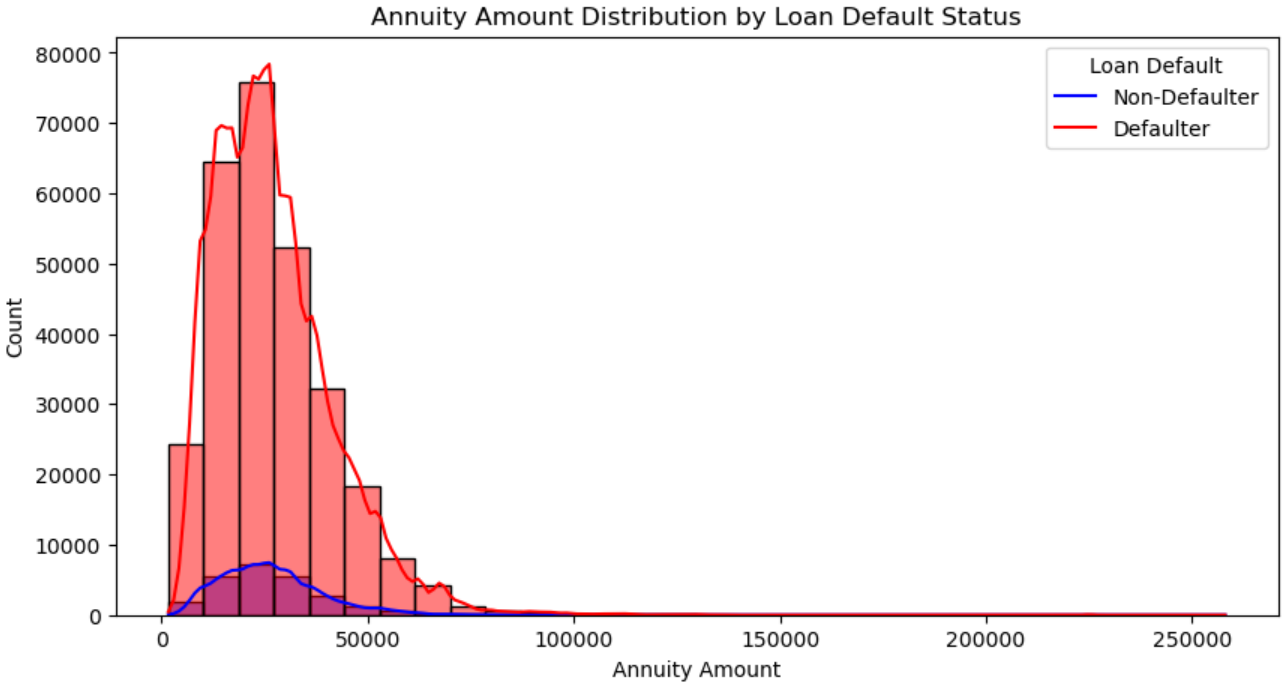


In [32]:
```
# Analyze how the credit amount is distributed among defaulters and non-defa
```

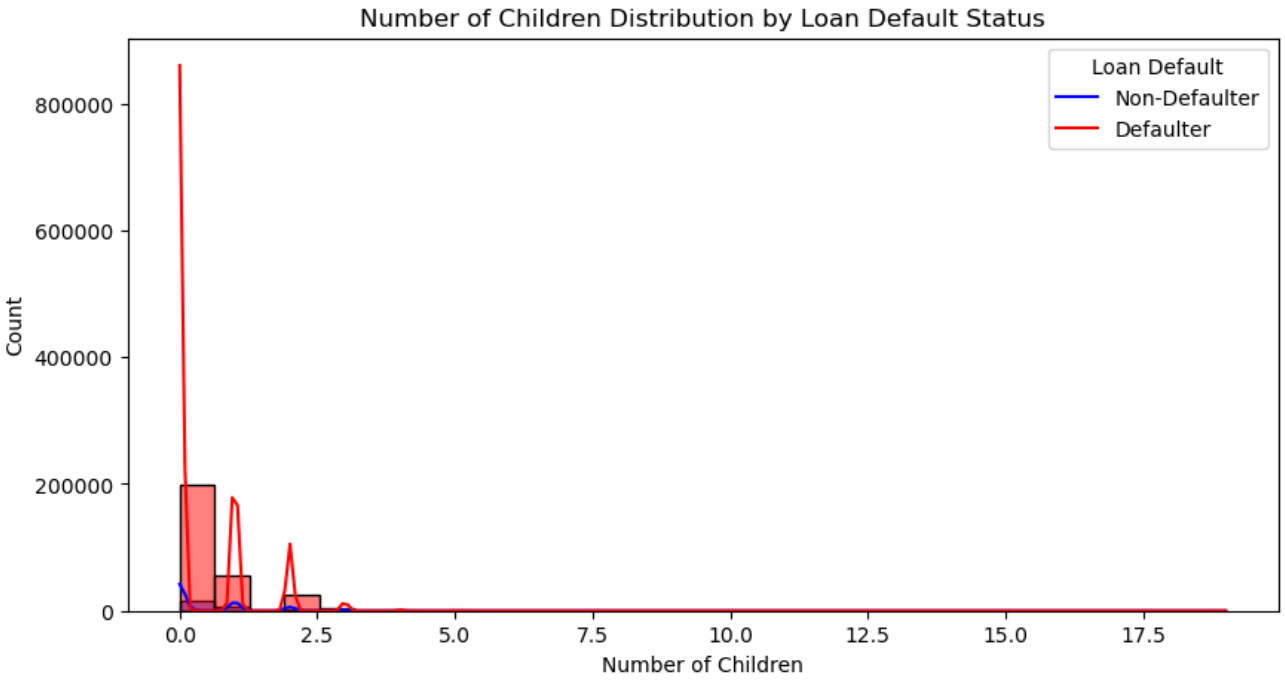
```
plt.figure(figsize=(10, 5))
sns.histplot(data=application, x='AMT_CREDIT', hue='TARGET', bins=30, palett
plt.title('Credit Amount Distribution by Loan Default Status')
plt.xlabel('Credit Amount')
plt.ylabel('Count')
plt.legend(title='Loan Default', labels=['Non-Defaulter', 'Defaulter'])
plt.show()
```



In [33]:
```
# Look at the distribution of annuity amounts.
plt.figure(figsize=(10, 5))
sns.histplot(data=application, x='AMT_ANNUITY', hue='TARGET', bins=30, palet
plt.title('Annuity Amount Distribution by Loan Default Status')
plt.xlabel('Annuity Amount')
plt.ylabel('Count')
plt.legend(title='Loan Default', labels=['Non-Defaulter', 'Defaulter'])
plt.show()
```



In [34]:
```
# See how the number of children affects loan defaults.
plt.figure(figsize=(10, 5))
sns.histplot(data=application, x='CNT_CHILDREN', hue='TARGET', bins=30, pale
plt.title('Number of Children Distribution by Loan Default Status')
plt.xlabel('Number of Children')
plt.ylabel('Count')
plt.legend(title='Loan Default', labels=['Non-Defaulter', 'Defaulter'])
plt.show()
```



In [35]:
```
# Univariate Analysis - Categorical Summary Statistics
```

```
# Print the column names in the application DataFrame
print(application.columns.tolist())
```

```
['SK_ID_CURR', 'TARGET', 'NAME_CONTRACT_TYPE', 'CODE_GENDER', 'FLAG_OWN_CA
R', 'FLAG_OWN_REALTY', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL', 'AMT_CREDIT', 'AM
T_ANNUITY', 'AMT_GOODS_PRICE', 'NAME_TYPE_SUITE', 'NAME_INCOME_TYPE', 'NAME_
EDUCATION_TYPE', 'NAME_FAMILY_STATUS', 'NAME_HOUSING_TYPE', 'REGION_POPULATI
ON_RELATIVE', 'DAYS_BIRTH', 'DAYS_EMPLOYED', 'DAYS_REGISTRATION', 'DAYS_ID_P
UBLISH', 'OWN_CAR_AGE', 'FLAG_MOBIL', 'FLAG_EMP_PHONE', 'FLAG_WORK_PHONE', '
FLAG_CONT_MOBILE', 'FLAG_PHONE', 'FLAG_EMAIL', 'OCCUPATION_TYPE', 'CNT_FAM_M
EMBERS', 'REGION_RATING_CLIENT', 'REGION_RATING_CLIENT_W_CITY', 'WEEKDAY_APP
R_PROCESS_START', 'HOUR_APPR_PROCESS_START', 'REG_REGION_NOT_LIVE_REGION', '
REG_REGION_NOT_WORK_REGION', 'LIVE_REGION_NOT_WORK_REGION', 'REG_CITY_NOT_LI
VE_CITY', 'REG_CITY_NOT_WORK_CITY', 'LIVE_CITY_NOT_WORK_CITY', 'ORGANIZATION
_TYPE', 'EXT_SOURCE_1', 'EXT_SOURCE_2', 'EXT_SOURCE_3', 'APARTMENTS_AVG', 'B
ASEMENTAREA_AVG', 'YEARS_BEGINEXPLUATATION_AVG', 'YEARS_BUILD_AVG', 'COMMONA
REA_AVG', 'ELEVATORS_AVG', 'ENTRANCES_AVG', 'FLOORSMAX_AVG', 'FLOORSMIN_AV
G', 'LANDAREA_AVG', 'LIVINGAPARTMENTS_AVG', 'LIVINGAREA_AVG', 'NONLIVINGAPAR
TMENTS_AVG', 'NONLIVINGAREA_AVG', 'APARTMENTS_MODE', 'BASEMENTAREA_MODE', 'Y
EARS_BEGINEXPLUATATION_MODE', 'YEARS_BUILD_MODE', 'COMMONAREA_MODE', 'ELEVAT
ORS_MODE', 'ENTRANCES_MODE', 'FLOORSMAX_MODE', 'FLOORSMIN_MODE', 'LANDAREA_M
ODE', 'LIVINGAPARTMENTS_MODE', 'LIVINGAREA_MODE', 'NONLIVINGAPARTMENTS_MOD
E', 'NONLIVINGAREA_MODE', 'APARTMENTS_MEDI', 'BASEMENTAREA_MEDI', 'YEARS_BEG
INEXPLUATATION_MEDI', 'YEARS_BUILD_MEDI', 'COMMONAREA_MEDI', 'ELEVATORS_MED
I', 'ENTRANCES_MEDI', 'FLOORSMAX_MEDI', 'FLOORSMIN_MEDI', 'LANDAREA_MEDI', '
LIVINGAPARTMENTS_MEDI', 'LIVINGAREA_MEDI', 'NONLIVINGAPARTMENTS_MEDI', 'NONL
IVINGAREA_MEDI', 'FONDKAPREMONT_MODE', 'HOUSETYPE_MODE', 'TOTALAREA_MODE', '
WALLSMATERIAL_MODE', 'EMERGENCYSTATE_MODE', 'OBS_30_CNT_SOCIAL_CIRCLE', 'DEF
_30_CNT_SOCIAL_CIRCLE', 'OBS_60_CNT_SOCIAL_CIRCLE', 'DEF_60_CNT_SOCIAL_CIRCL
E', 'DAYS_LAST_PHONE_CHANGE', 'FLAG_DOCUMENT_2', 'FLAG_DOCUMENT_3', 'FLAG_DO
CUMENT_4', 'FLAG_DOCUMENT_5', 'FLAG_DOCUMENT_6', 'FLAG_DOCUMENT_7', 'FLAG_DO
CUMENT_8', 'FLAG_DOCUMENT_9', 'FLAG_DOCUMENT_10', 'FLAG_DOCUMENT_11', 'FLAG_
DOCUMENT_12', 'FLAG_DOCUMENT_13', 'FLAG_DOCUMENT_14', 'FLAG_DOCUMENT_15', 'F
LAG_DOCUMENT_16', 'FLAG_DOCUMENT_17', 'FLAG_DOCUMENT_18', 'FLAG_DOCUMENT_1
9', 'FLAG_DOCUMENT_20', 'FLAG_DOCUMENT_21', 'AMT_REQ_CREDIT_BUREAU_HOUR', 'A
MT_REQ_CREDIT_BUREAU_DAY', 'AMT_REQ_CREDIT_BUREAU_WEEK', 'AMT_REQ_CREDIT_BUR
EAU_MON', 'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_CREDIT_BUREAU_YEAR', 'AGE',
'AGE_GROUP']
```

In [73]:
```
# Univariate Analysis - Categorical Summary Statistics
# Sample DataFrame for demonstration purposes
# Replace this with our actual data loading step
numeric_data = application.select_dtypes(include=['float64', 'int64'])
# Check that numeric_data is defined and has data
print(numeric_data.head())  # Display first few rows
print(numeric_data.info())  # Get info about DataFrame
```

```
   SK_ID_CURR  TARGET  CNT_CHILDREN  AMT_INCOME_TOTAL  AMT_CREDIT  \
0      100002       1             0          202500.0    406597.5
1      100003       0             0          270000.0   1293502.5
2      100004       0             0           67500.0    135000.0
3      100006       0             0          135000.0    312682.5
4      100007       0             0          121500.0    513000.0

   AMT_ANNUITY  AMT_GOODS_PRICE  REGION_POPULATION_RELATIVE  DAYS_BIRTH  \
0      24700.5         351000.0                    0.018801       -9461
1      35698.5        1129500.0                    0.003541      -16765
2       6750.0         135000.0                    0.010032      -19046
3      29686.5         297000.0                    0.008019      -19005
4      21865.5         513000.0                    0.028663      -19932

   DAYS_EMPLOYED  ...  FLAG_DOCUMENT_19  FLAG_DOCUMENT_20  FLAG_DOCUMENT_21
\
0           -637  ...                 0                 0                 0
1          -1188  ...                 0                 0                 0
2           -225  ...                 0                 0                 0
3          -3039  ...                 0                 0                 0
4          -3038  ...                 0                 0                 0

   AMT_REQ_CREDIT_BUREAU_HOUR  AMT_REQ_CREDIT_BUREAU_DAY  \
0                         0.0                        0.0
1                         0.0                        0.0
2                         0.0                        0.0
3                         NaN                        NaN
4                         0.0                        0.0

   AMT_REQ_CREDIT_BUREAU_WEEK  AMT_REQ_CREDIT_BUREAU_MON  \
0                         0.0                        0.0
1                         0.0                        0.0
2                         0.0                        0.0
3                         NaN                        NaN
4                         0.0                        0.0

   AMT_REQ_CREDIT_BUREAU_QRT  AMT_REQ_CREDIT_BUREAU_YEAR  AGE
0                        0.0                         1.0   25
1                        0.0                         0.0   45
2                        0.0                         0.0   52
3                        NaN                         NaN   52
4                        0.0                         0.0   54

[5 rows x 107 columns]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Columns: 107 entries, SK_ID_CURR to AGE
dtypes: float64(65), int64(42)
memory usage: 251.0 MB
None
```

In [76]:
```
# Initial data exploration
# Check the information of the previous.csv dataset
print(previous_application.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1670214 entries, 0 to 1670213
Data columns (total 26 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                  1670214 non-null  float64 
 4   AMT_APPLICATION              1670214 non-null  float64 
 5   AMT_CREDIT                   1670214 non-null  float64 
 6   AMT_GOODS_PRICE              1670214 non-null  float64 
 7   WEEKDAY_APPR_PROCESS_START   1670214 non-null  object  
 8   HOUR_APPR_PROCESS_START      1670214 non-null  int64   
 9   FLAG_LAST_APPL_PER_CONTRACT  1670214 non-null  object  
 10  NFLAG_LAST_APPL_IN_DAY       1670214 non-null  int64   
 11  NAME_CASH_LOAN_PURPOSE       1670214 non-null  category
 12  NAME_CONTRACT_STATUS         1670214 non-null  category
 13  DAYS_DECISION                1670214 non-null  int64   
 14  NAME_PAYMENT_TYPE            1670214 non-null  category
 15  CODE_REJECT_REASON           1670214 non-null  category
 16  NAME_CLIENT_TYPE             1670214 non-null  category
 17  NAME_GOODS_CATEGORY          1670214 non-null  category
 18  NAME_PORTFOLIO               1670214 non-null  category
 19  NAME_PRODUCT_TYPE            1670214 non-null  category
 20  CHANNEL_TYPE                 1670214 non-null  category
 21  SELLERPLACE_AREA             1670214 non-null  int64   
 22  NAME_SELLER_INDUSTRY         1670214 non-null  category
 23  CNT_PAYMENT                  1670214 non-null  int64   
 24  NAME_YIELD_GROUP             1670214 non-null  category
 25  PRODUCT_COMBINATION          1670214 non-null  category
dtypes: category(13), float64(4), int64(7), object(2)
memory usage: 186.4+ MB
None
```

In [91]:
```
# Univariate Analysis - Categorical Summary Statistics
# Segregate loan defaulters and non-defaulters based on the TARGET variable
had_difficulties = application[application['TARGET'] == 1]  # Default = had_
had_no_difficulties = application[application['TARGET'] == 0]  # Non-default

# Display the first few rows of each group
print("Defaulters:")
print(had_difficulties.head())
print("\nNon-defaulters:")
print(had_no_difficulties.head())
```

```
# Update this list based on the columns in our DataFrame
categorical_params = [
    'NAME_CONTRACT_TYPE',
    'CODE_GENDER',
    'FLAG_OWN_CAR',
    'FLAG_OWN_REALTY',
    'CNT_CHILDREN',
    'NAME_EDUCATION_TYPE',
    'NAME_HOUSING_TYPE',
    'NAME_FAMILY_STATUS',
    'NAME_INCOME_TYPE',
    'AMT_ANNUITY',
    'ORGANIZATION_TYPE',
    'EXPERIENCE_RANGE',
    'AGE_GROUP'
]

# Segregate loan defaulters and non-defaulters based on the TARGET variable
had_difficulties = application[application['TARGET'] == 1]  # Default = had_
had_no_difficulties = application[application['TARGET'] == 0]  # Non-default

# Display the first few rows of each group
print("Defaulters:")
print(had_difficulties.head())
print("\nNon-defaulters:")
print(had_no_difficulties.head())

# Update this list based on the actual columns in our DataFrame
categorical_params = [
    'NAME_CONTRACT_TYPE',
    'CODE_GENDER',
    'FLAG_OWN_CAR',
    'FLAG_OWN_REALTY',
    'CNT_CHILDREN',
    'NAME_EDUCATION_TYPE',
    'NAME_HOUSING_TYPE',
    'NAME_FAMILY_STATUS',
    'NAME_INCOME_TYPE',
    'AMT_ANNUITY',
    'ORGANIZATION_TYPE',
    'EXPERIENCE_RANGE',
    'AGE_GROUP'
]

# Function to plot count plots for categorical parameters
def plot_categorical_counts(data, categorical_param):
    if categorical_param in data.columns:
        plt.figure(figsize=(10, 5))
        sns.countplot(data=data, x=categorical_param, order=data[categorical
        plt.title(f'Distribution of {categorical_param.upper()}')
```

```
        plt.xlabel(categorical_param.upper())
        plt.ylabel('Count')
        plt.xticks(rotation=45)
        plt.show()
    else:
        print(f"Warning: {categorical_param} does not exist in the DataFrame

# Run the visualization for each categorical parameter for defaulters and no
for param in categorical_params:
    print(f"\nPlotting for {param} - Defaulters:")
    plot_categorical_counts(had_difficulties, param)

    print(f"\nPlotting for {param} - Non-defaulters:")
    plot_categorical_counts(had_no_difficulties, param)
```

```
Defaulters:
    SK_ID_CURR  TARGET NAME_CONTRACT_TYPE CODE_GENDER FLAG_OWN_CAR  \
0       100002       1         Cash loans           M            N
26      100031       1         Cash loans           F            N
40      100047       1         Cash loans           M            N
42      100049       1         Cash loans           F            N
81      100096       1         Cash loans           F            N

   FLAG_OWN_REALTY  CNT_CHILDREN  AMT_INCOME_TOTAL  AMT_CREDIT  AMT_ANNUITY
\
0                Y             0          202500.0    406597.5      24700.5
26               Y             0          112500.0    979992.0      27076.5
40               Y             0          202500.0   1193580.0      35028.0
42               N             0          135000.0    288873.0      16258.5
81               Y             0           81000.0    252000.0      14593.5

    ...  FLAG_DOCUMENT_18 FLAG_DOCUMENT_19 FLAG_DOCUMENT_20 FLAG_DOCUMENT_21
\
0   ...                 0                0                0                0
26  ...                 0                0                0                0
40  ...                 0                0                0                0
42  ...                 0                0                0                0
81  ...                 0                0                0                0

   AMT_REQ_CREDIT_BUREAU_HOUR AMT_REQ_CREDIT_BUREAU_DAY  \
0                         0.0                        0.0
26                        0.0                        0.0
40                        0.0                        0.0
42                        0.0                        0.0
81                        0.0                        0.0

    AMT_REQ_CREDIT_BUREAU_WEEK  AMT_REQ_CREDIT_BUREAU_MON  \
0                          0.0                        0.0
26                         0.0                        0.0
40                         0.0                        2.0
```

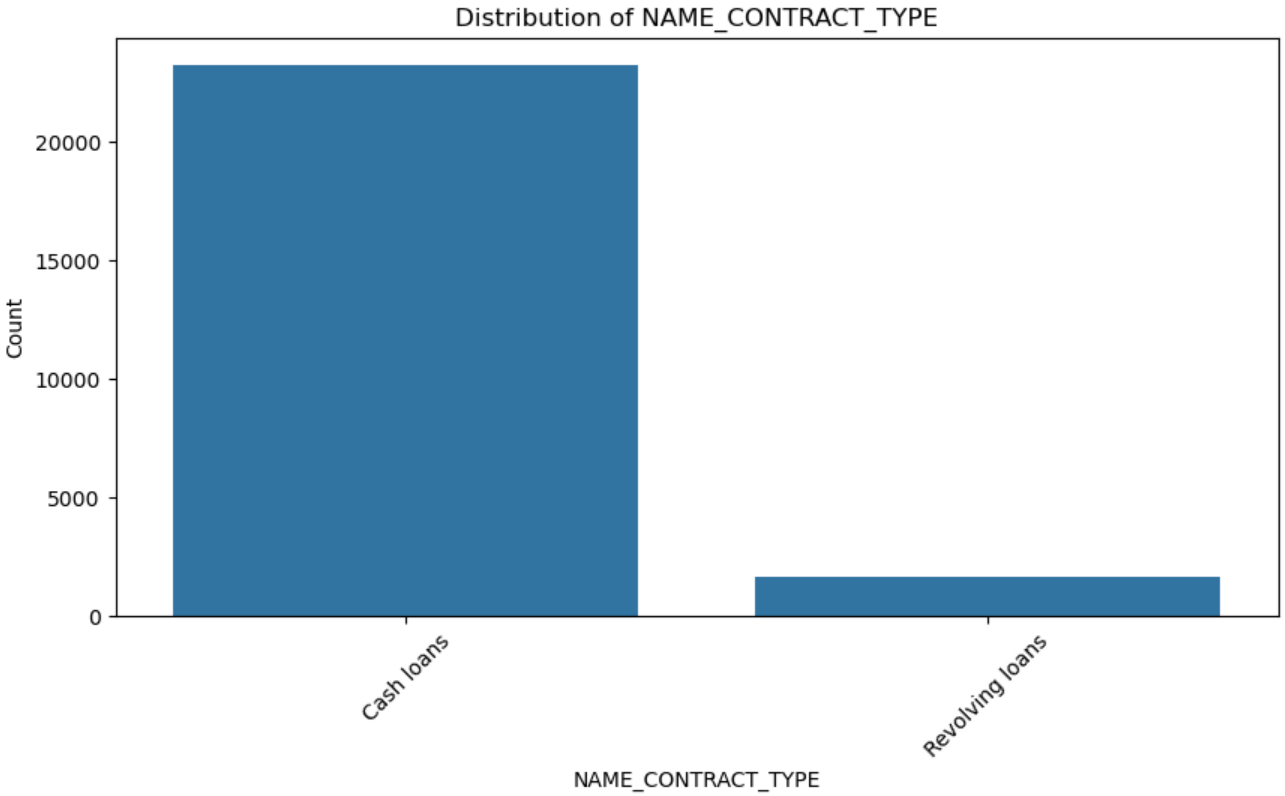
```
42                          0.0                          0.0
81                          0.0                          0.0

    AMT_REQ_CREDIT_BUREAU_QRT  AMT_REQ_CREDIT_BUREAU_YEAR
0                         0.0                         1.0
26                        2.0                         2.0
40                        0.0                         4.0
42                        0.0                         2.0
81                        0.0                         0.0

[5 rows x 122 columns]

Non-defaulters:
   SK_ID_CURR  TARGET NAME_CONTRACT_TYPE CODE_GENDER FLAG_OWN_CAR  \
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      100008       0         Cash loans           M            N

  FLAG_OWN_REALTY  CNT_CHILDREN  AMT_INCOME_TOTAL  AMT_CREDIT  AMT_ANNUITY
\
1               N             0          270000.0   1293502.5      35698.5
2               Y             0           67500.0    135000.0       6750.0
3               Y             0          135000.0    312682.5      29686.5
4               Y             0          121500.0    513000.0      21865.5
5               Y             0           99000.0    490495.5      27517.5

   ...  FLAG_DOCUMENT_18 FLAG_DOCUMENT_19 FLAG_DOCUMENT_20 FLAG_DOCUMENT_21
\
1  ...                 0                0                0                0
2  ...                 0                0                0                0
3  ...                 0                0                0                0
4  ...                 0                0                0                0
5  ...                 0                0                0                0

  AMT_REQ_CREDIT_BUREAU_HOUR AMT_REQ_CREDIT_BUREAU_DAY  \
1                        0.0                        0.0
2                        0.0                        0.0
3                        NaN                        NaN
4                        0.0                        0.0
5                        0.0                        0.0

   AMT_REQ_CREDIT_BUREAU_WEEK  AMT_REQ_CREDIT_BUREAU_MON  \
1                         0.0                        0.0
2                         0.0                        0.0
3                         NaN                        NaN
4                         0.0                        0.0
5                         0.0                        0.0
```

```
   AMT_REQ_CREDIT_BUREAU_QRT  AMT_REQ_CREDIT_BUREAU_YEAR
1                        0.0                         0.0
2                        0.0                         0.0
3                        NaN                         NaN
4                        0.0                         0.0
5                        1.0                         1.0

[5 rows x 122 columns]
Defaulters:
    SK_ID_CURR  TARGET NAME_CONTRACT_TYPE CODE_GENDER FLAG_OWN_CAR  \
0       100002       1         Cash loans           M            N
26      100031       1         Cash loans           F            N
40      100047       1         Cash loans           M            N
42      100049       1         Cash loans           F            N
81      100096       1         Cash loans           F            N

   FLAG_OWN_REALTY  CNT_CHILDREN  AMT_INCOME_TOTAL  AMT_CREDIT  AMT_ANNUITY
\
0                Y             0          202500.0    406597.5      24700.5
26               Y             0          112500.0    979992.0      27076.5
40               Y             0          202500.0   1193580.0      35028.0
42               N             0          135000.0    288873.0      16258.5
81               Y             0           81000.0    252000.0      14593.5

    ...  FLAG_DOCUMENT_18 FLAG_DOCUMENT_19 FLAG_DOCUMENT_20 FLAG_DOCUMENT_21
\
0   ...                 0                0                0                0
26  ...                 0                0                0                0
40  ...                 0                0                0                0
42  ...                 0                0                0                0
81  ...                 0                0                0                0

   AMT_REQ_CREDIT_BUREAU_HOUR AMT_REQ_CREDIT_BUREAU_DAY  \
0                         0.0                        0.0
26                        0.0                        0.0
40                        0.0                        0.0
42                        0.0                        0.0
81                        0.0                        0.0

    AMT_REQ_CREDIT_BUREAU_WEEK  AMT_REQ_CREDIT_BUREAU_MON  \
0                          0.0                        0.0
26                         0.0                        0.0
40                         0.0                        2.0
42                         0.0                        0.0
81                         0.0                        0.0

    AMT_REQ_CREDIT_BUREAU_QRT  AMT_REQ_CREDIT_BUREAU_YEAR
0                         0.0                        1.0
26                        2.0                        2.0
40                        0.0                        4.0
```

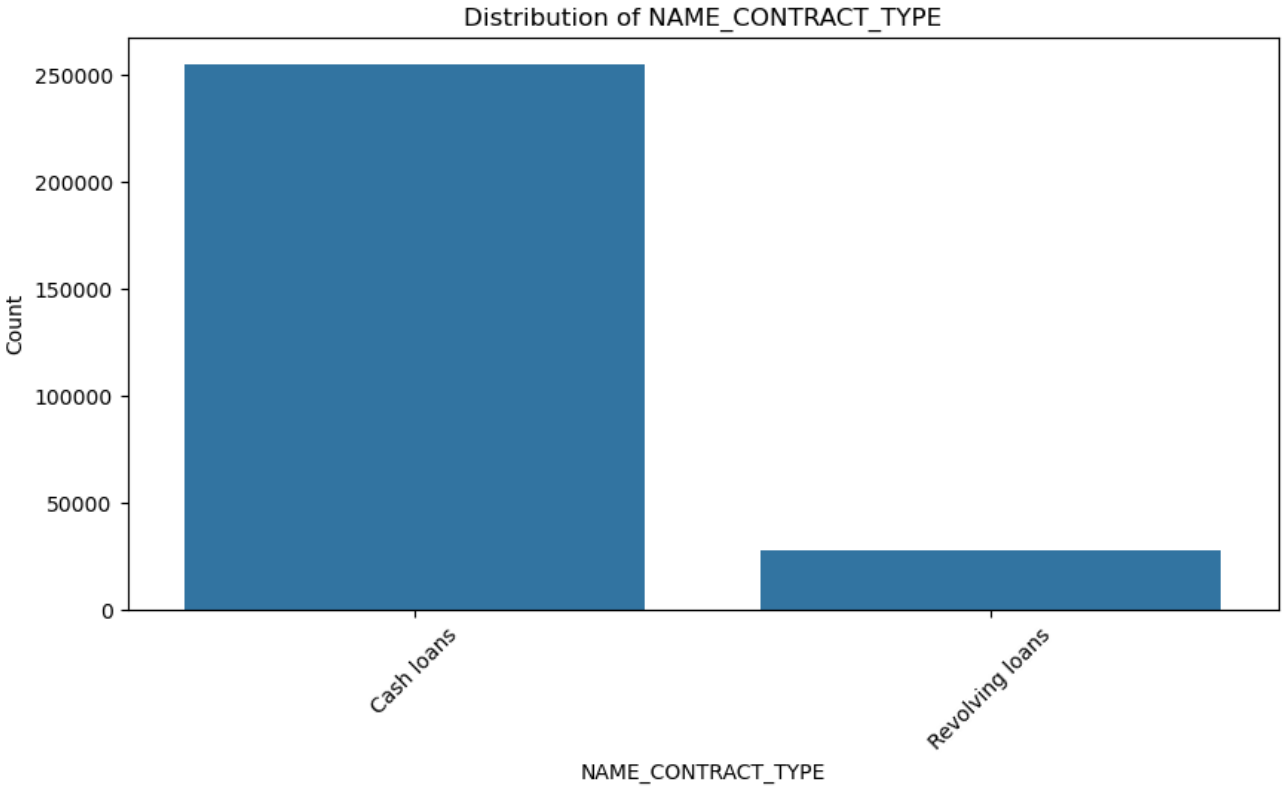
```
42                        0.0                          2.0
81                        0.0                          0.0

[5 rows x 122 columns]

Non-defaulters:
   SK_ID_CURR  TARGET NAME_CONTRACT_TYPE CODE_GENDER FLAG_OWN_CAR  \
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      100008       0         Cash loans           M            N

  FLAG_OWN_REALTY  CNT_CHILDREN  AMT_INCOME_TOTAL  AMT_CREDIT  AMT_ANNUITY
\
1               N             0          270000.0   1293502.5      35698.5
2               Y             0           67500.0    135000.0       6750.0
3               Y             0          135000.0    312682.5      29686.5
4               Y             0          121500.0    513000.0      21865.5
5               Y             0           99000.0    490495.5      27517.5

   ...  FLAG_DOCUMENT_18 FLAG_DOCUMENT_19 FLAG_DOCUMENT_20 FLAG_DOCUMENT_21
\
1  ...                 0                0                0                0
2  ...                 0                0                0                0
3  ...                 0                0                0                0
4  ...                 0                0                0                0
5  ...                 0                0                0                0

  AMT_REQ_CREDIT_BUREAU_HOUR AMT_REQ_CREDIT_BUREAU_DAY  \
1                        0.0                       0.0
2                        0.0                       0.0
3                        NaN                       NaN
4                        0.0                       0.0
5                        0.0                       0.0

   AMT_REQ_CREDIT_BUREAU_WEEK  AMT_REQ_CREDIT_BUREAU_MON  \
1                         0.0                        0.0
2                         0.0                        0.0
3                         NaN                        NaN
4                         0.0                        0.0
5                         0.0                        0.0

   AMT_REQ_CREDIT_BUREAU_QRT  AMT_REQ_CREDIT_BUREAU_YEAR
1                        0.0                        0.0
2                        0.0                        0.0
3                        NaN                        NaN
4                        0.0                        0.0
5                        1.0                        1.0
```

[5 rows x 122 columns]

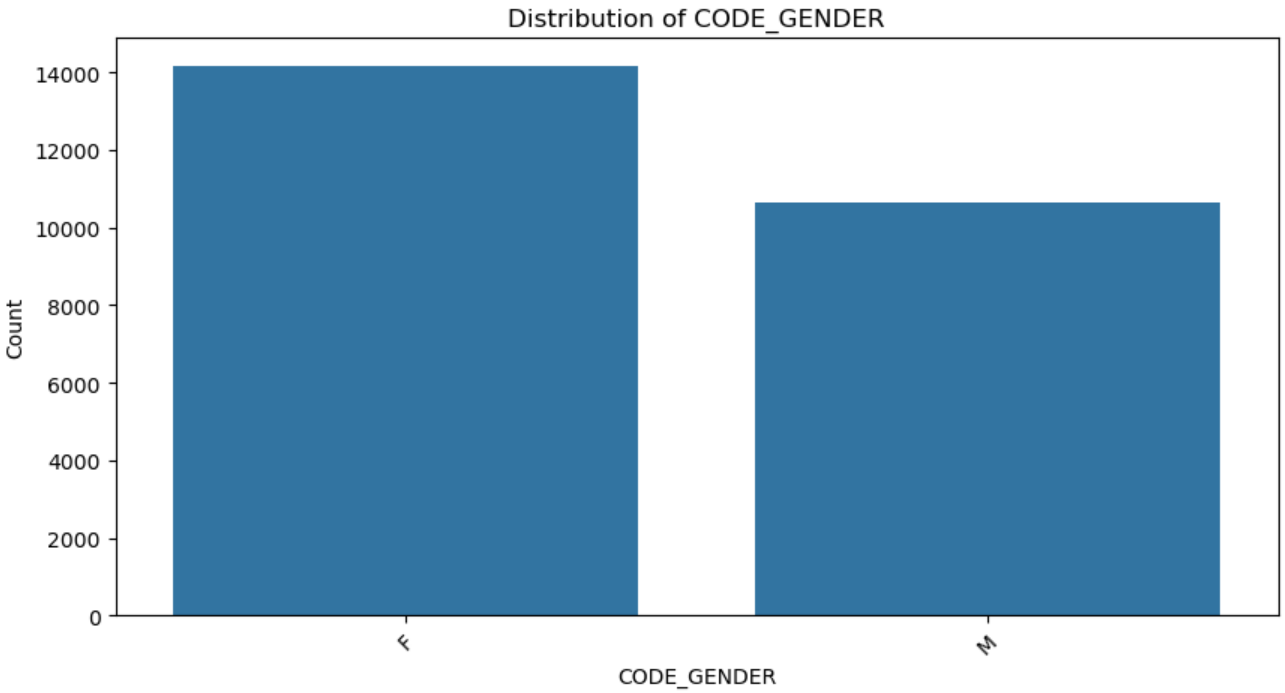
Plotting for NAME_CONTRACT_TYPE - Defaulters:



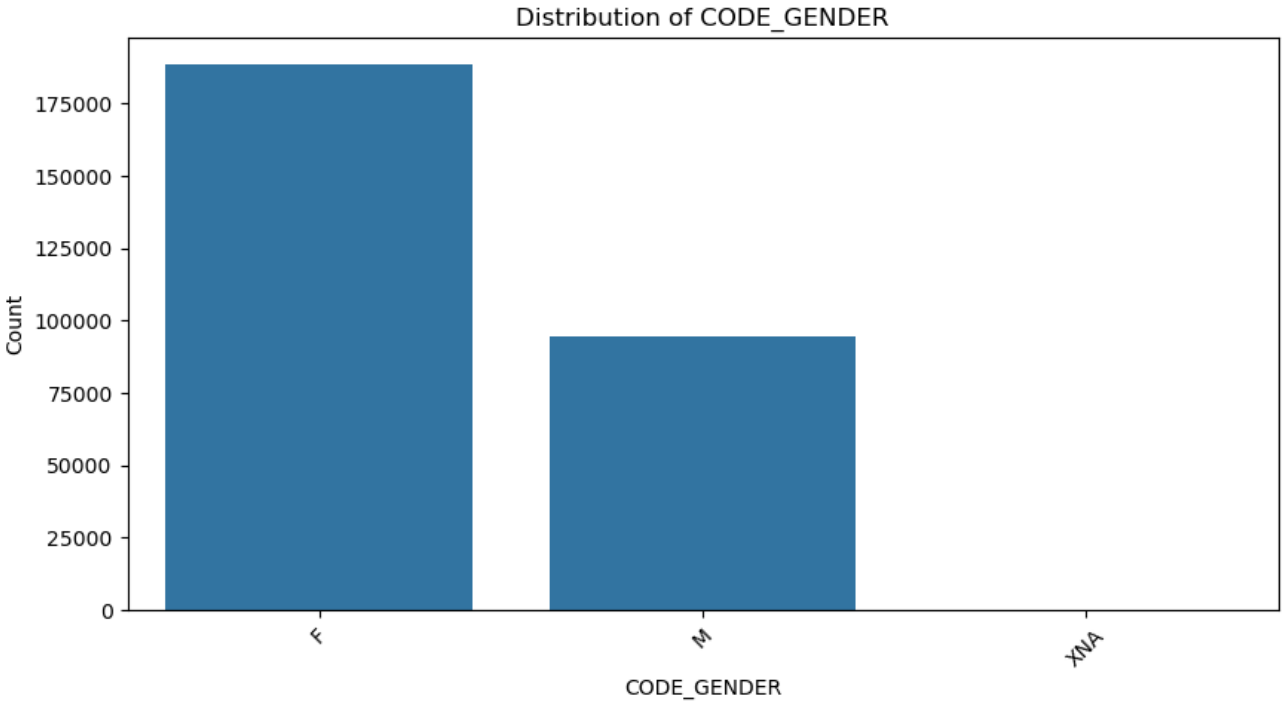
Plotting for NAME_CONTRACT_TYPE - Non-defaulters:



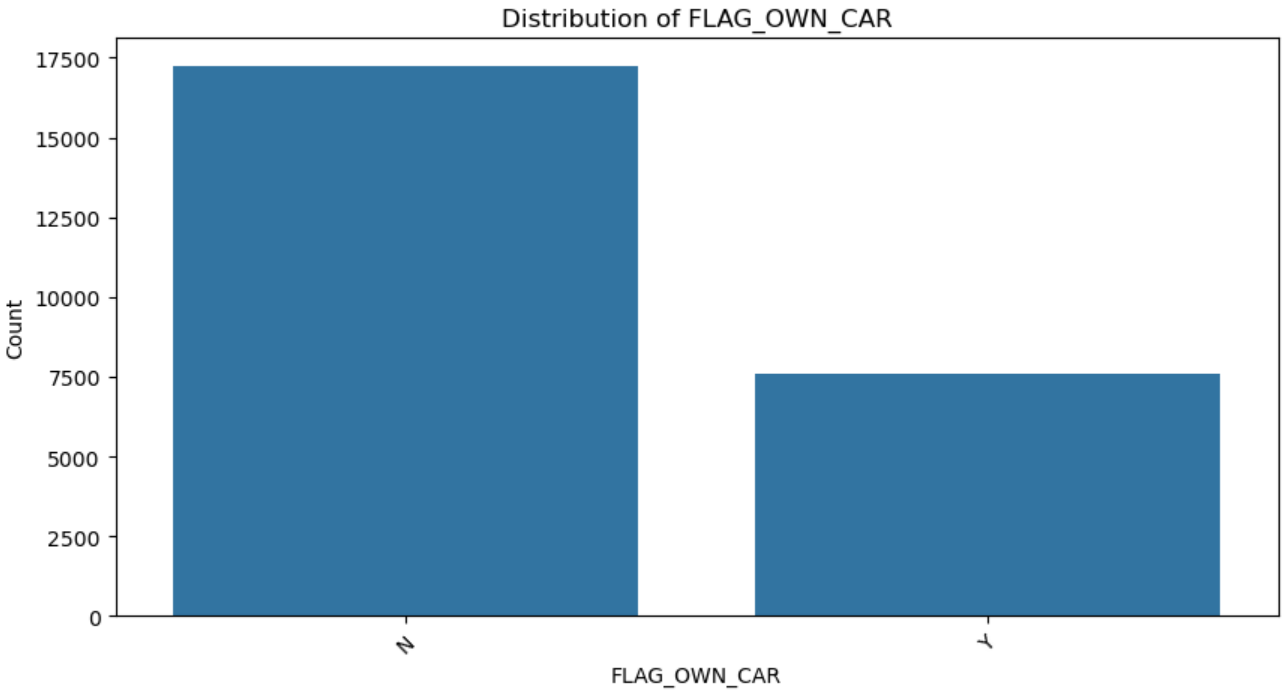
Plotting for CODE_GENDER - Defaulters:



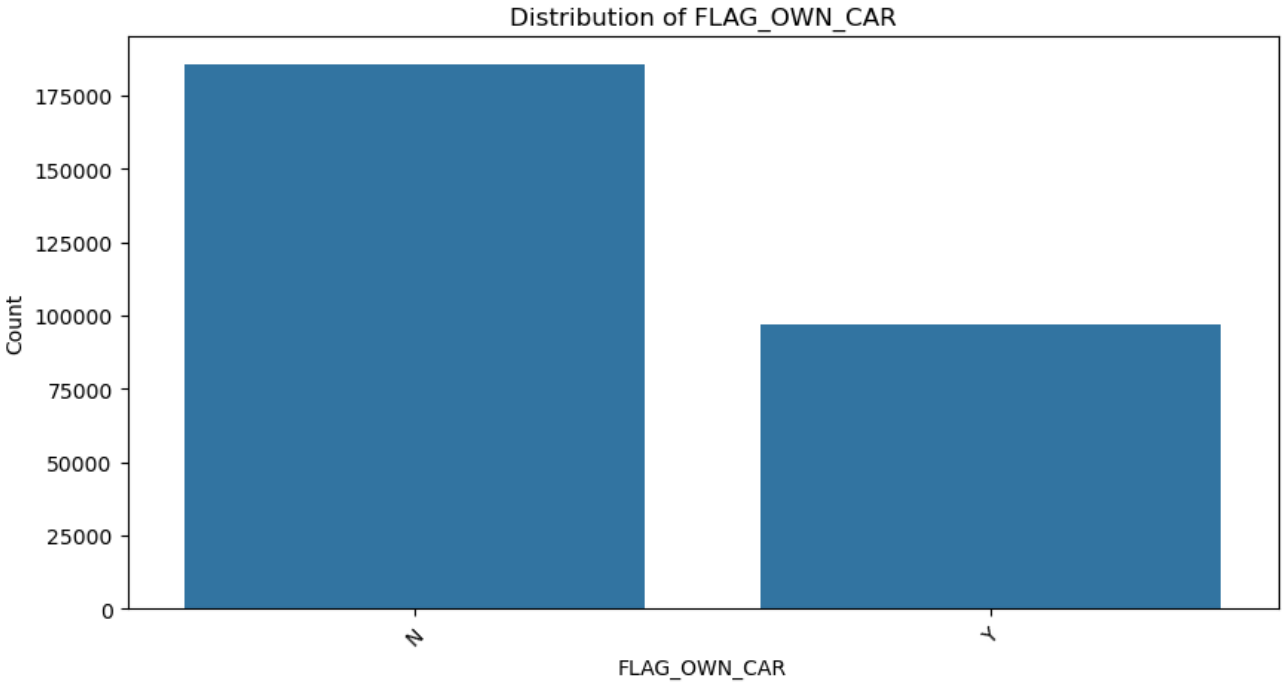
Plotting for CODE_GENDER - Non-defaulters:



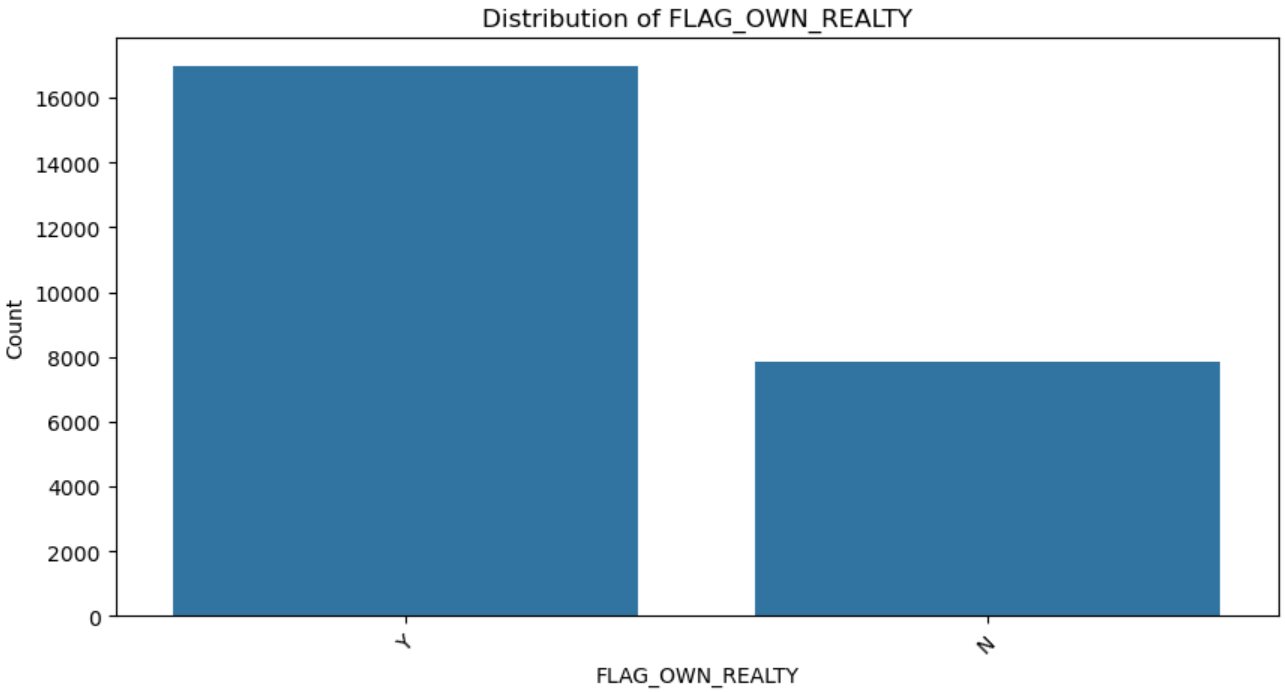
Plotting for FLAG_OWN_CAR - Defaulters:



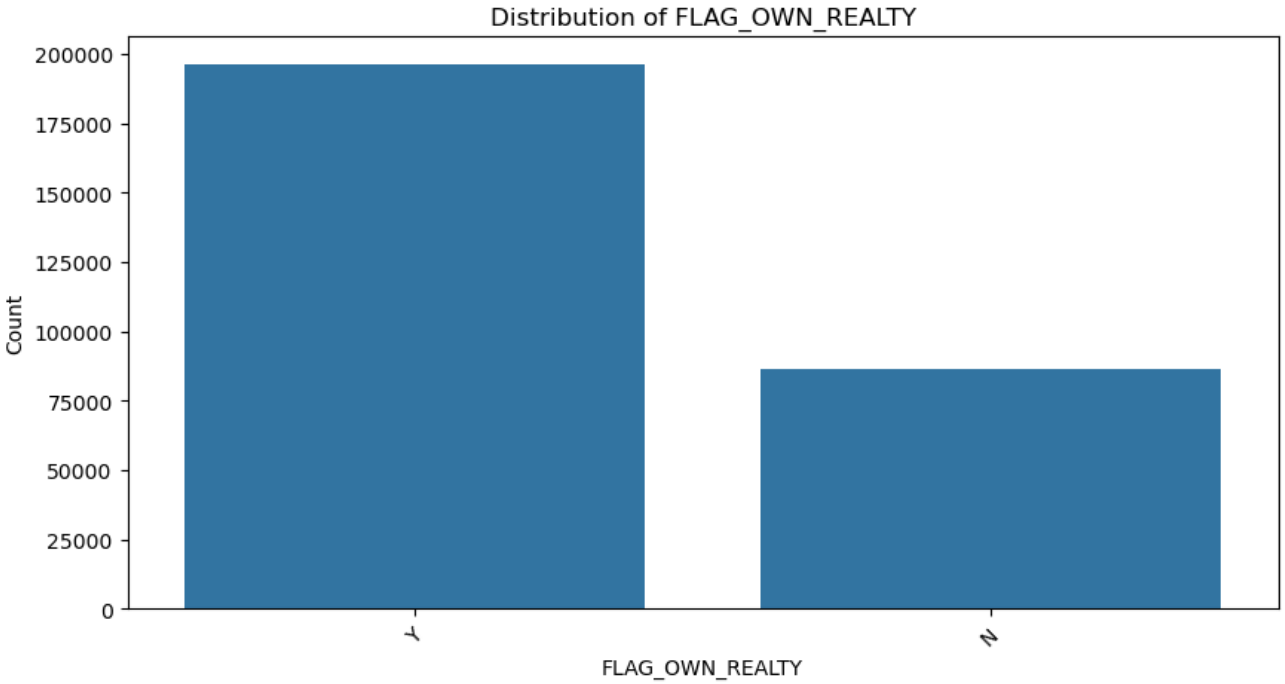
Plotting for FLAG_OWN_CAR - Non-defaulters:



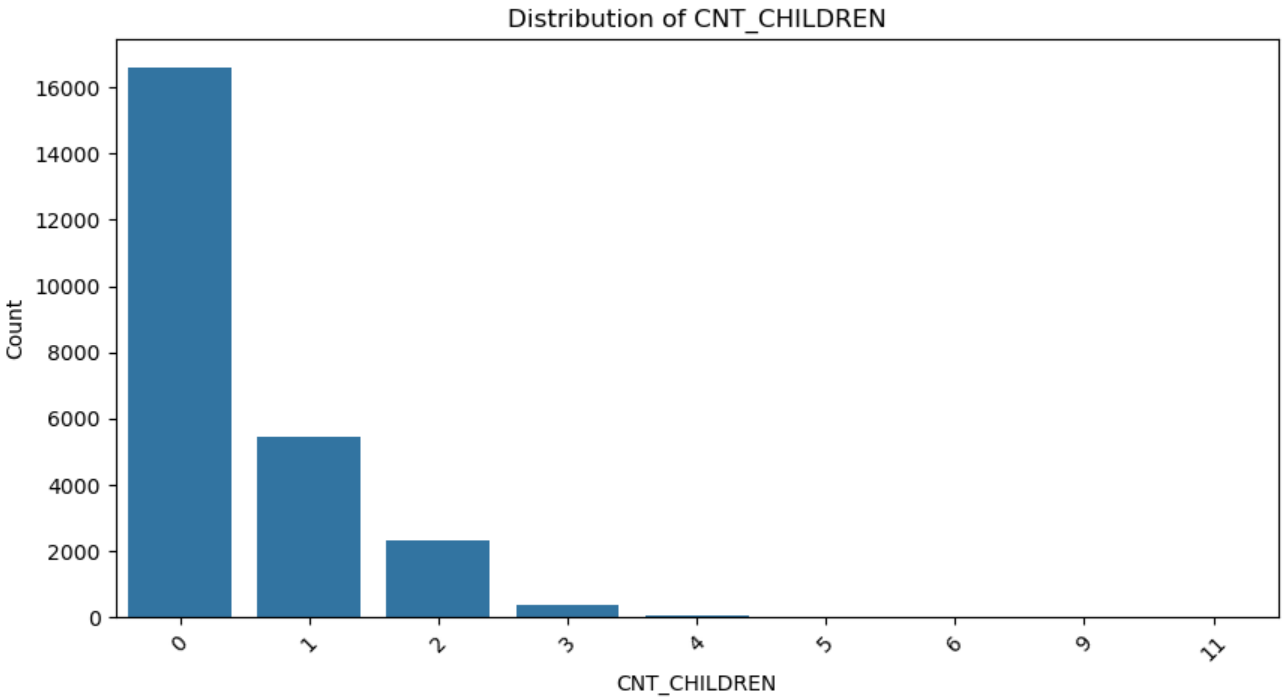
Plotting for FLAG_OWN_REALTY - Defaulters:



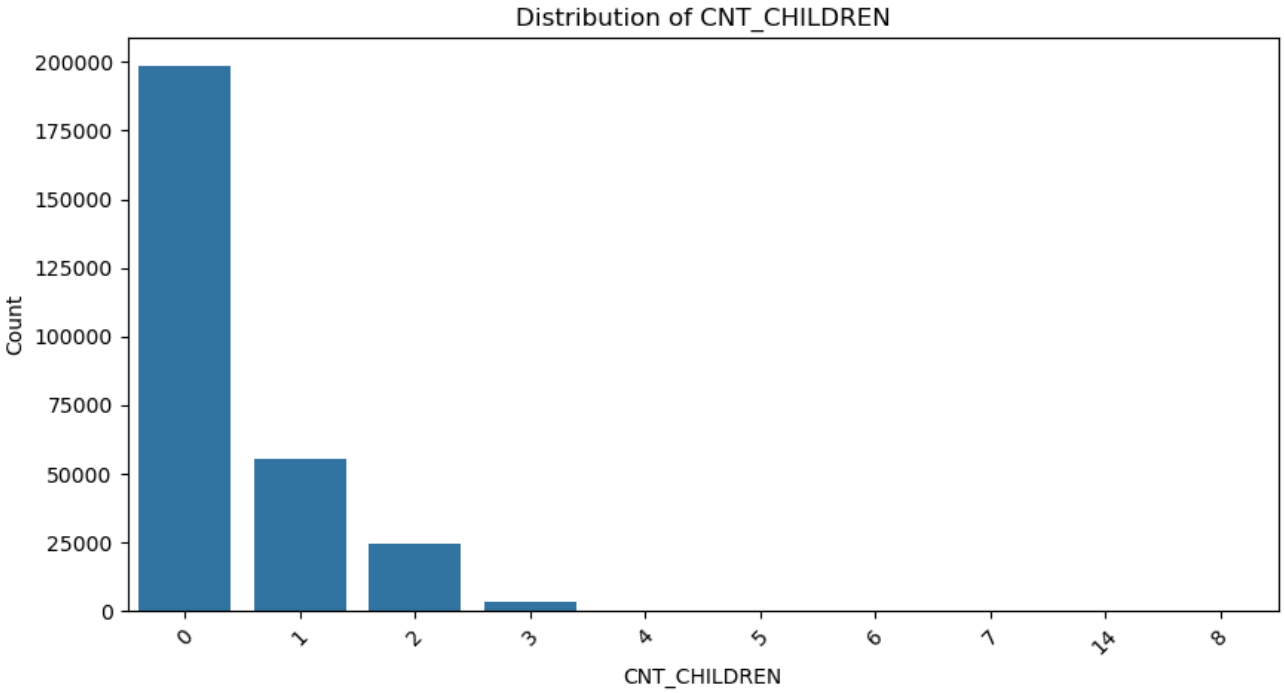
Plotting for FLAG_OWN_REALTY - Non-defaulters:



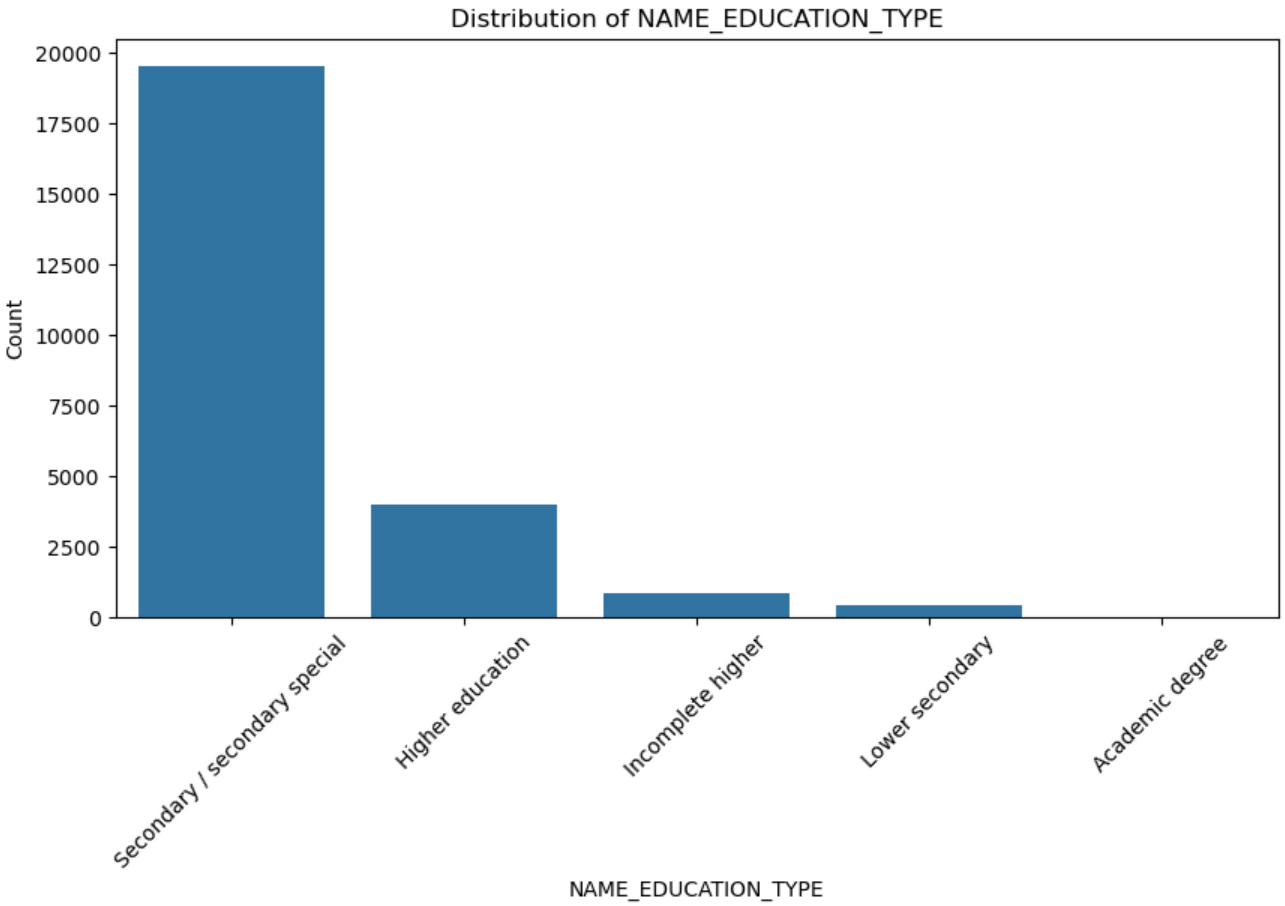
Plotting for CNT_CHILDREN - Defaulters:



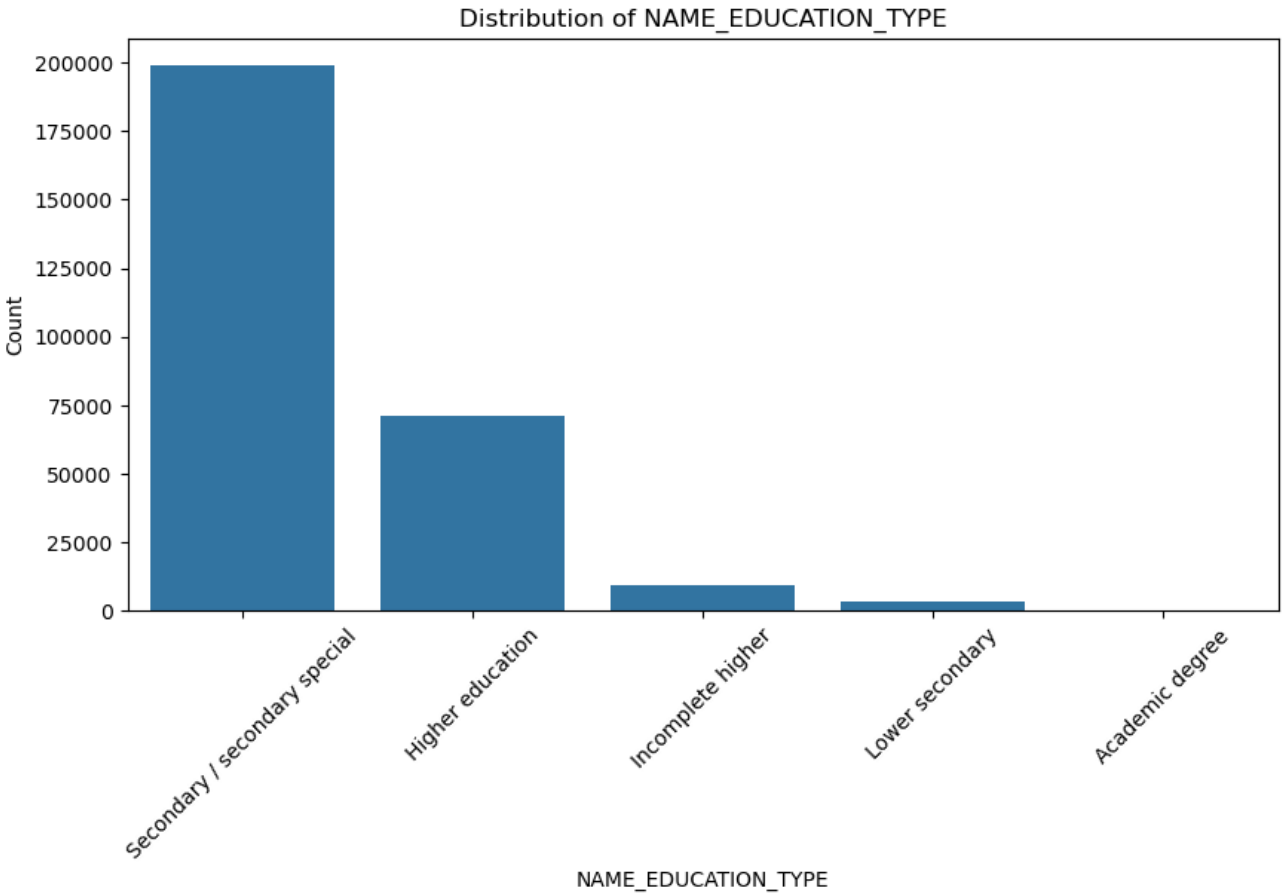
Plotting for CNT_CHILDREN - Non-defaulters:



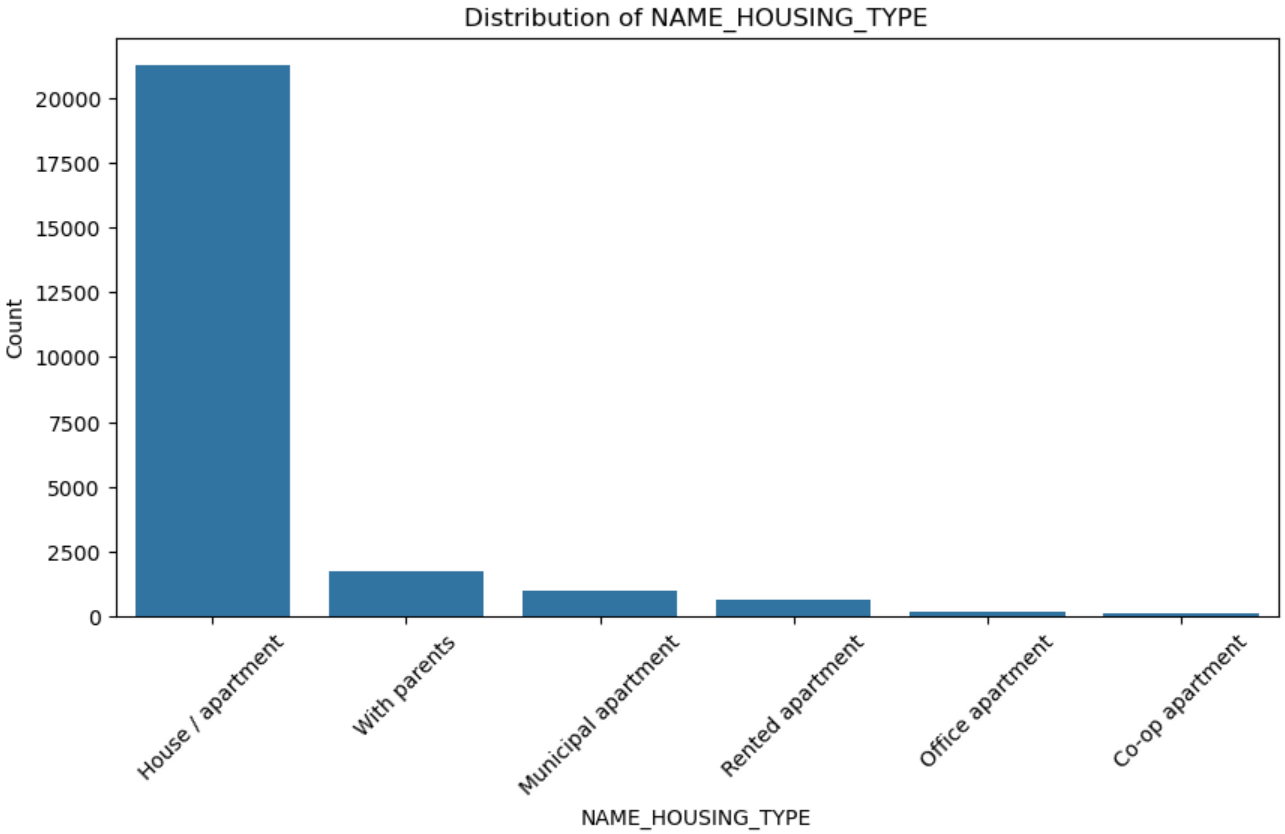
Plotting for NAME_EDUCATION_TYPE - Defaulters:



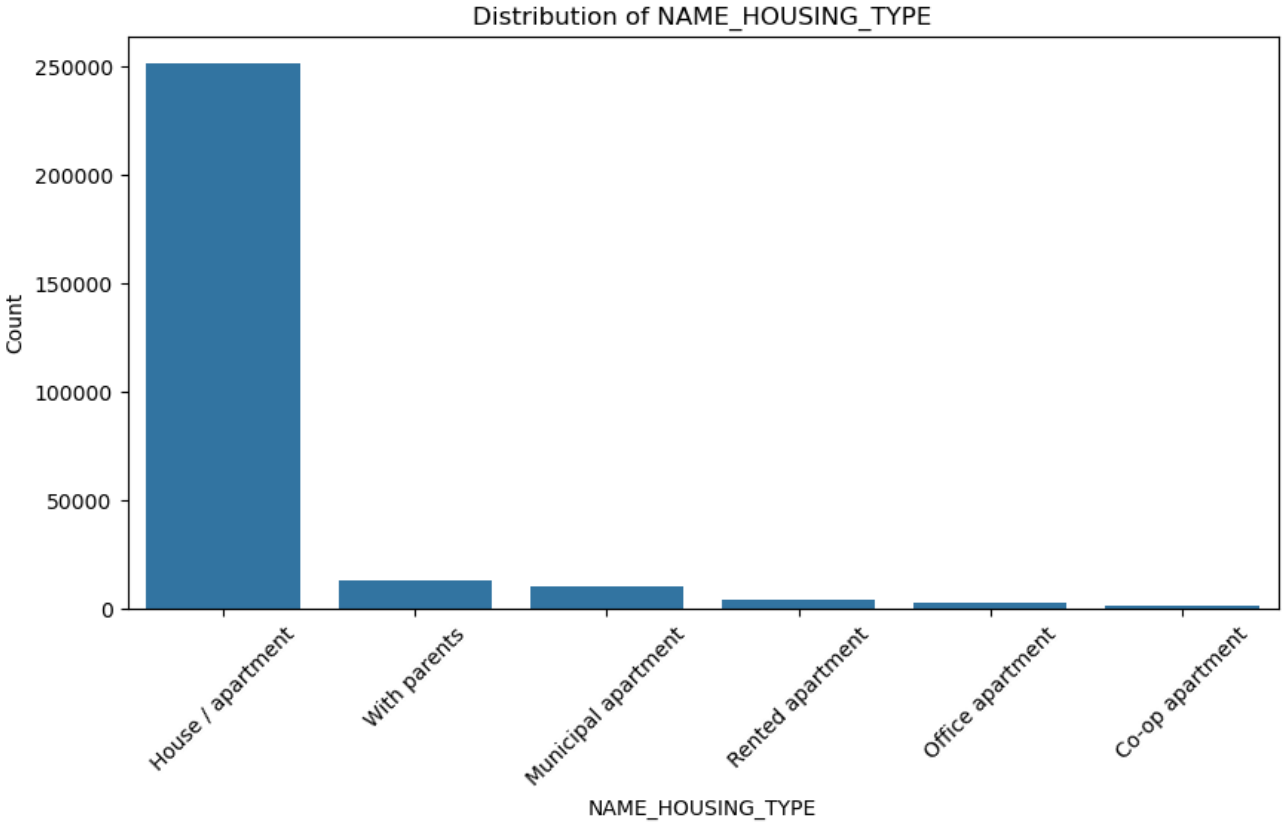
Plotting for NAME_EDUCATION_TYPE - Non-defaulters:



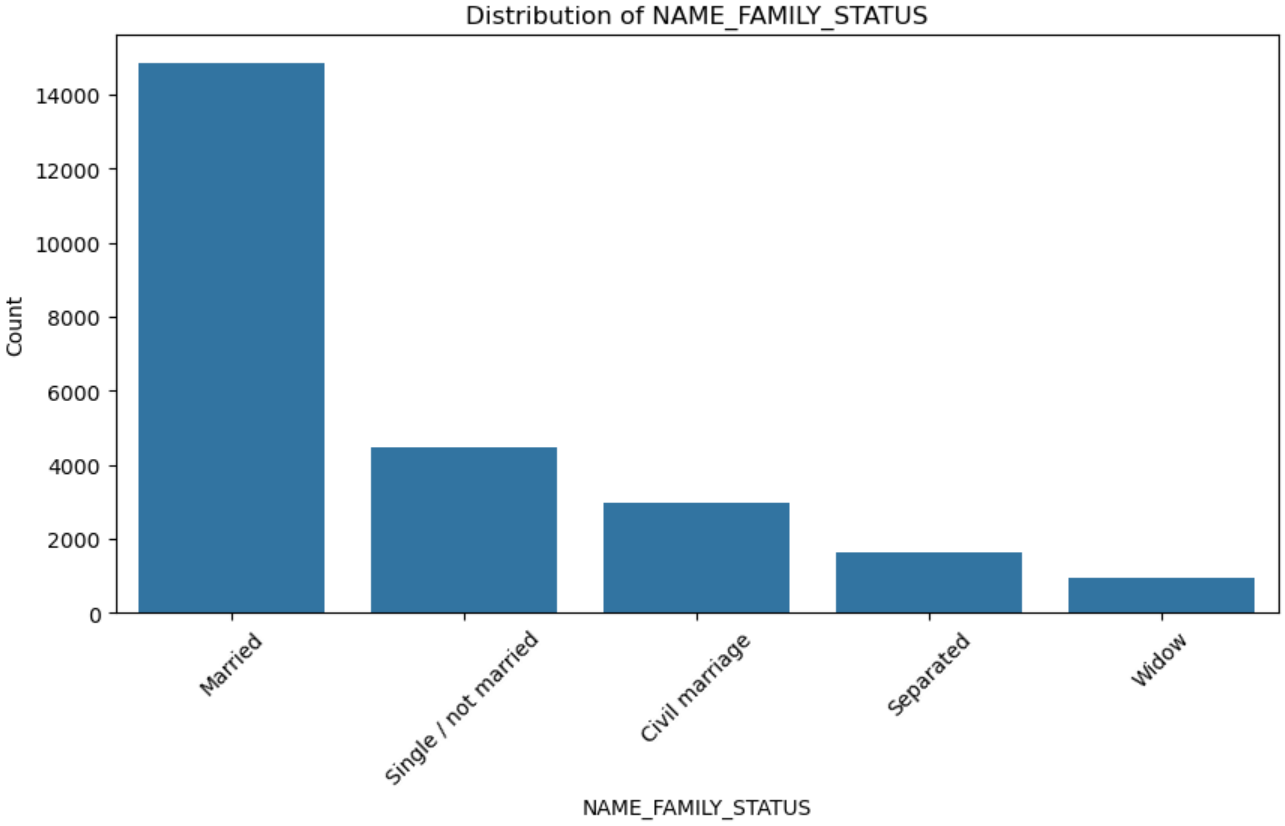
Plotting for NAME_HOUSING_TYPE - Defaulters:



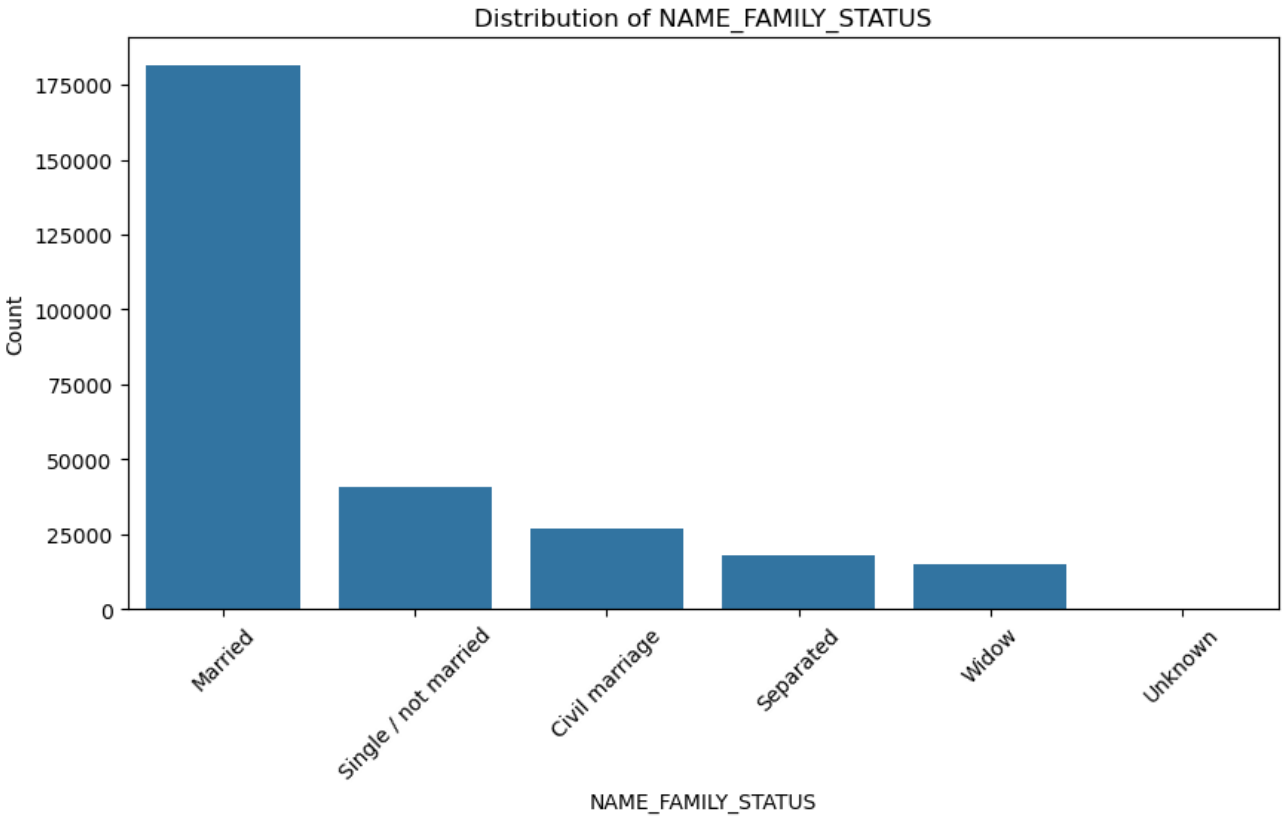
Plotting for NAME_HOUSING_TYPE - Non-defaulters:



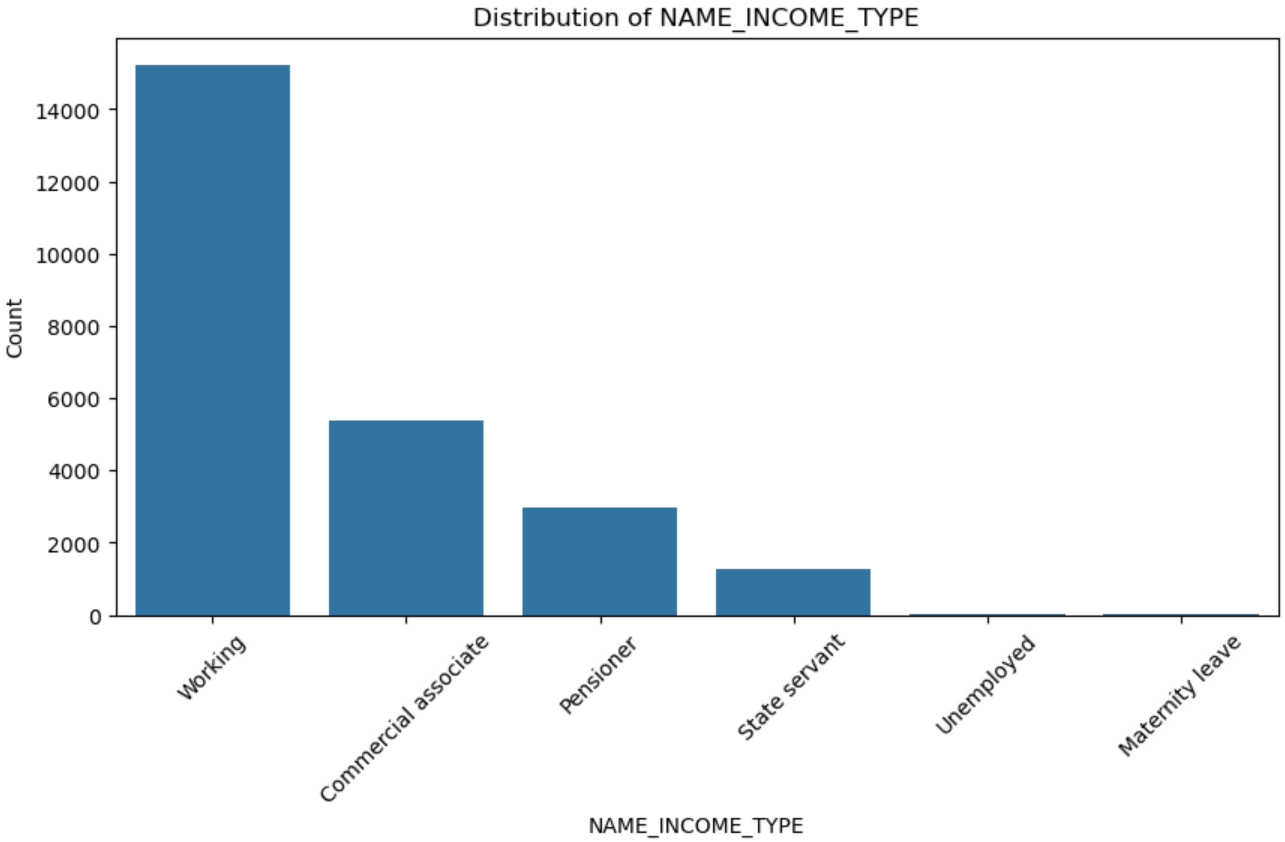
Plotting for NAME_FAMILY_STATUS - Defaulters:



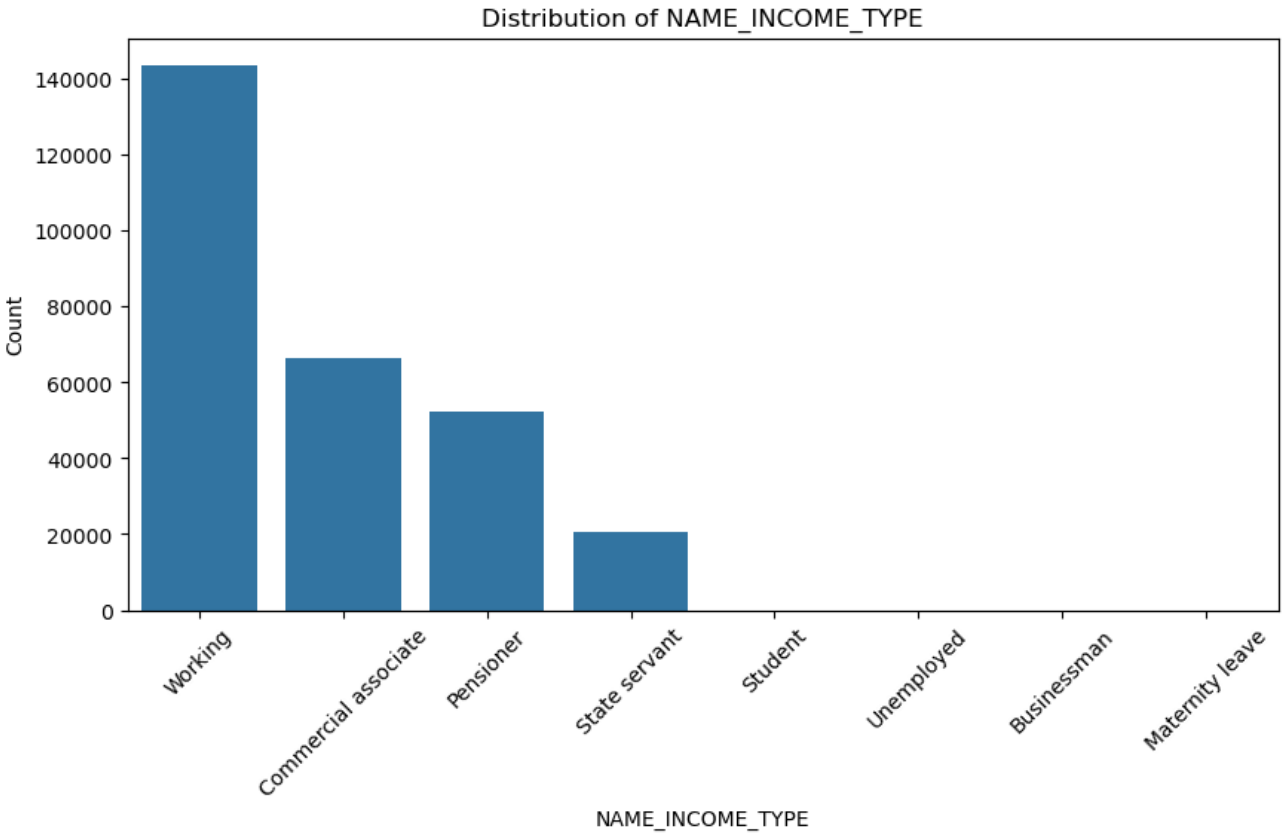
Plotting for NAME_FAMILY_STATUS - Non-defaulters:



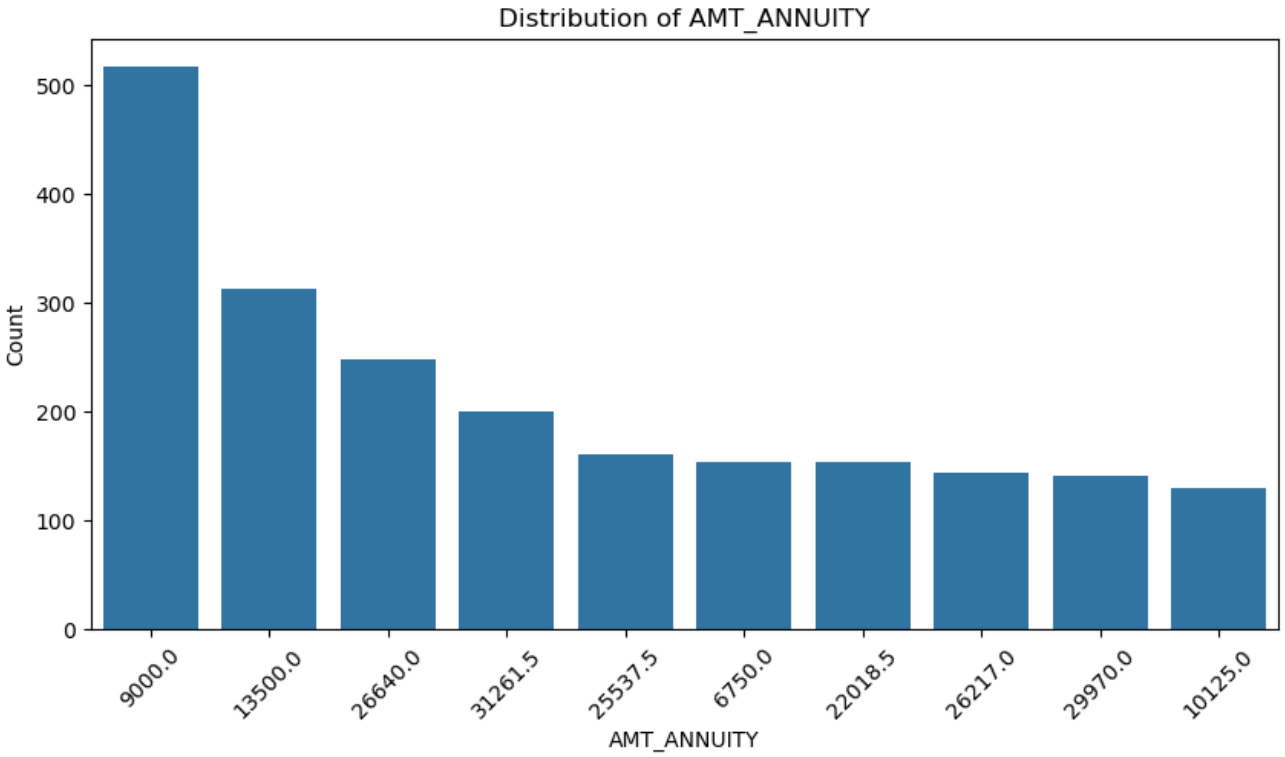
Plotting for NAME_INCOME_TYPE - Defaulters:



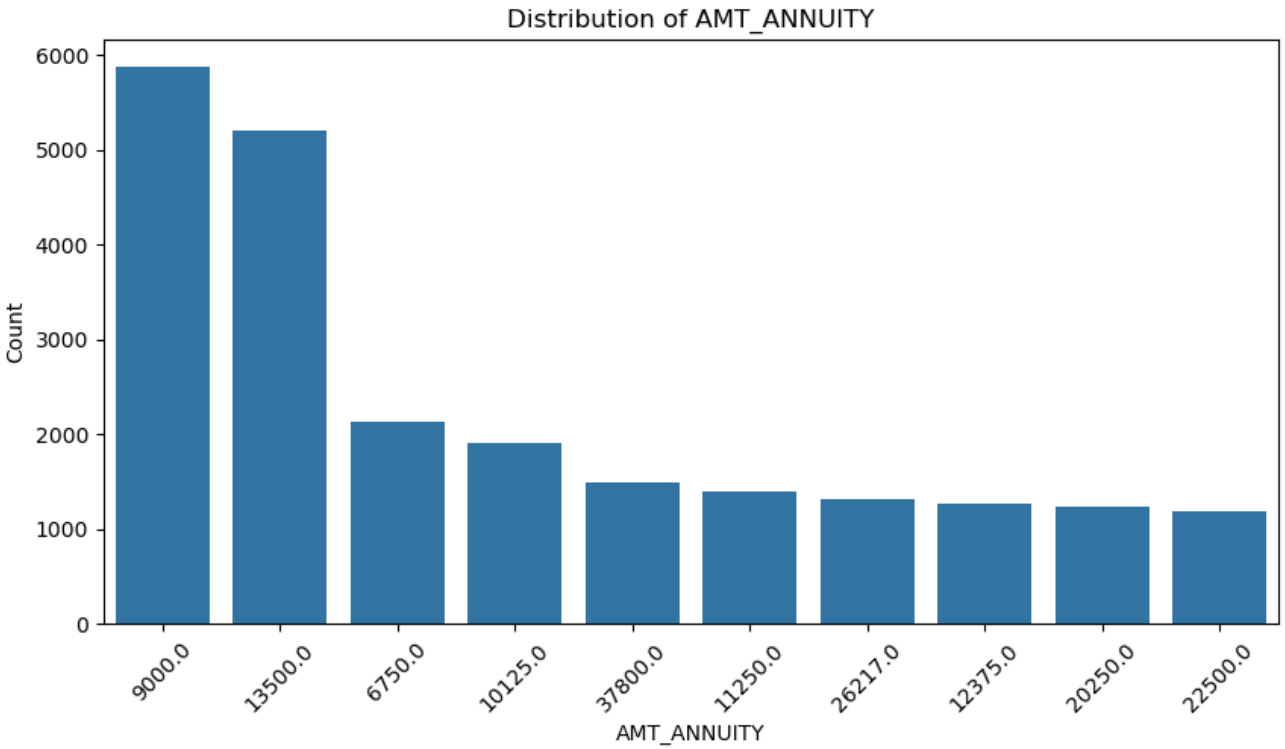
Plotting for NAME_INCOME_TYPE - Non-defaulters:



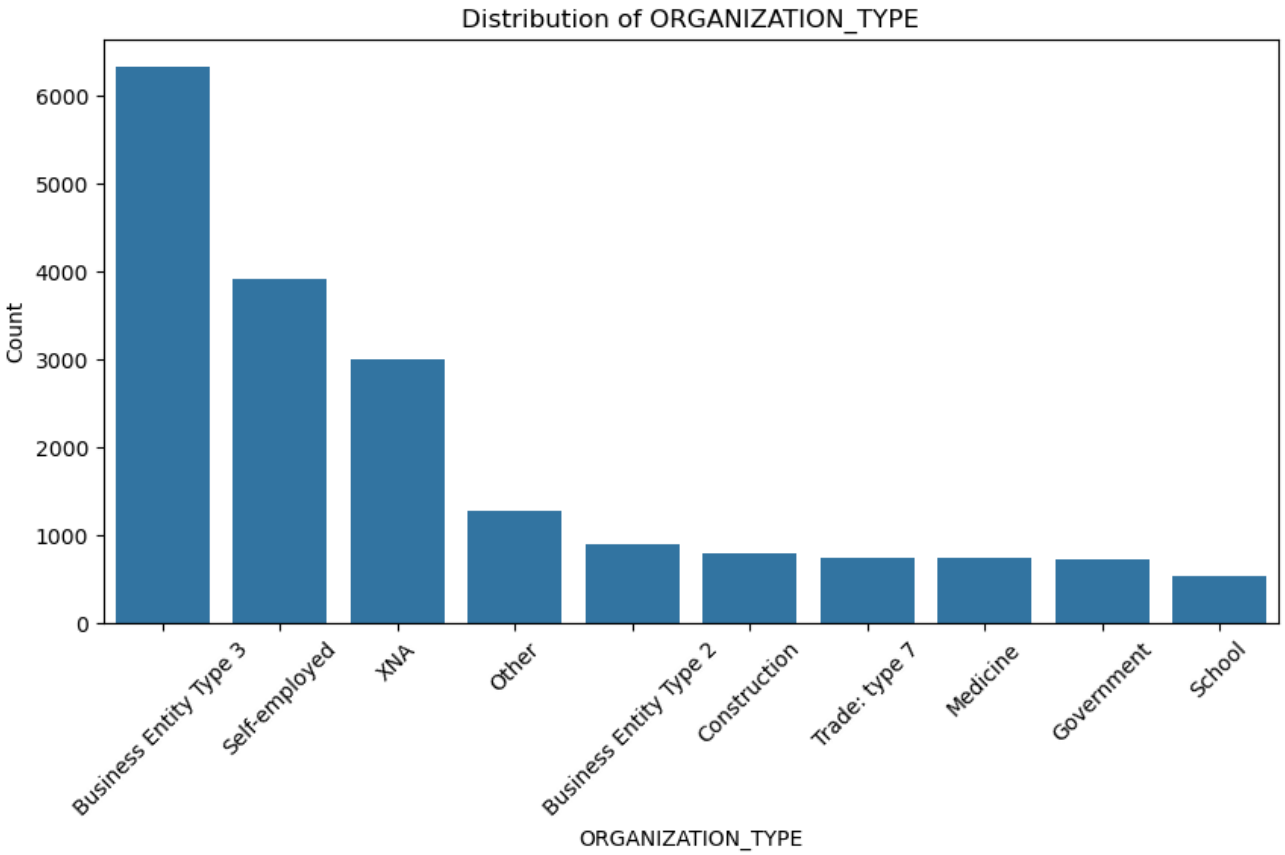
Plotting for AMT_ANNUITY - Defaulters:



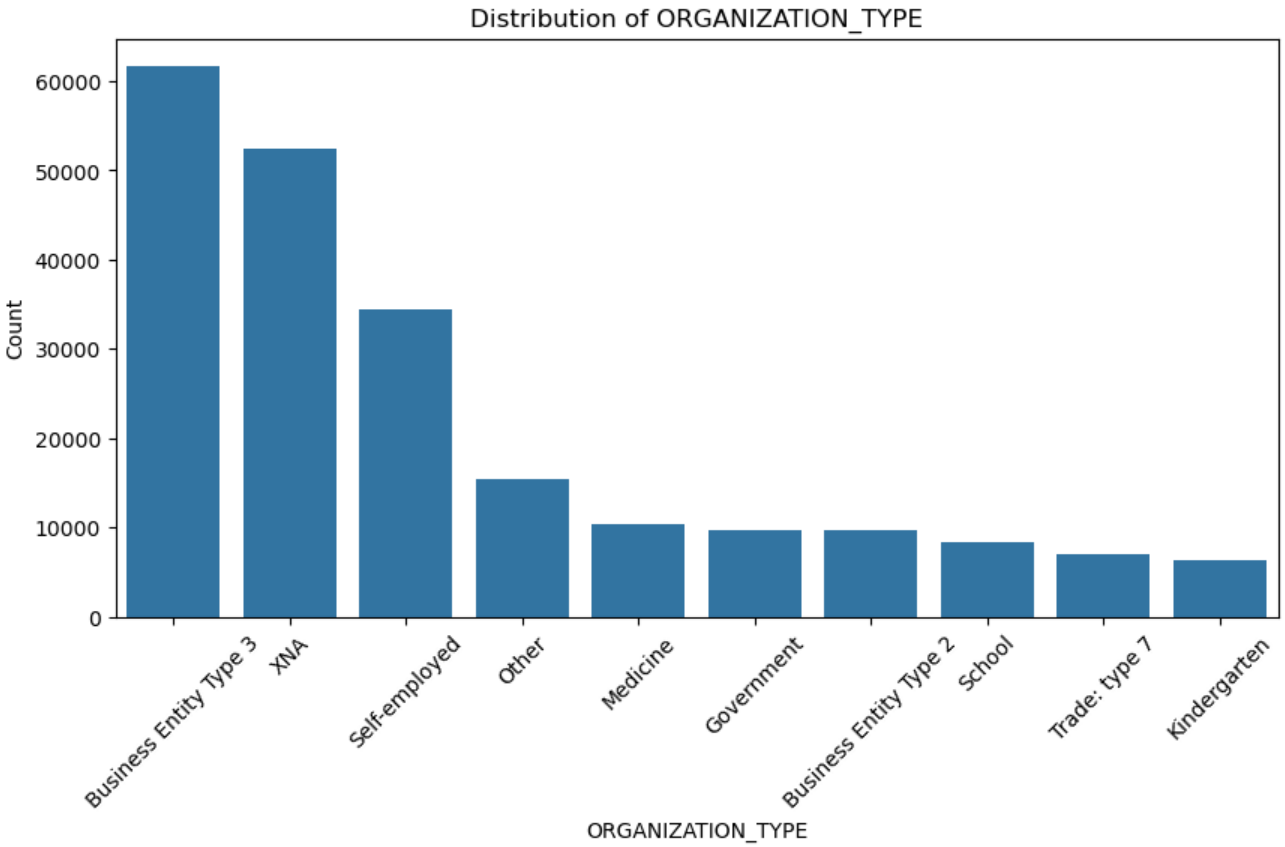
Plotting for AMT_ANNUITY - Non-defaulters:



Plotting for ORGANIZATION_TYPE - Defaulters:



Plotting for ORGANIZATION_TYPE - Non-defaulters:



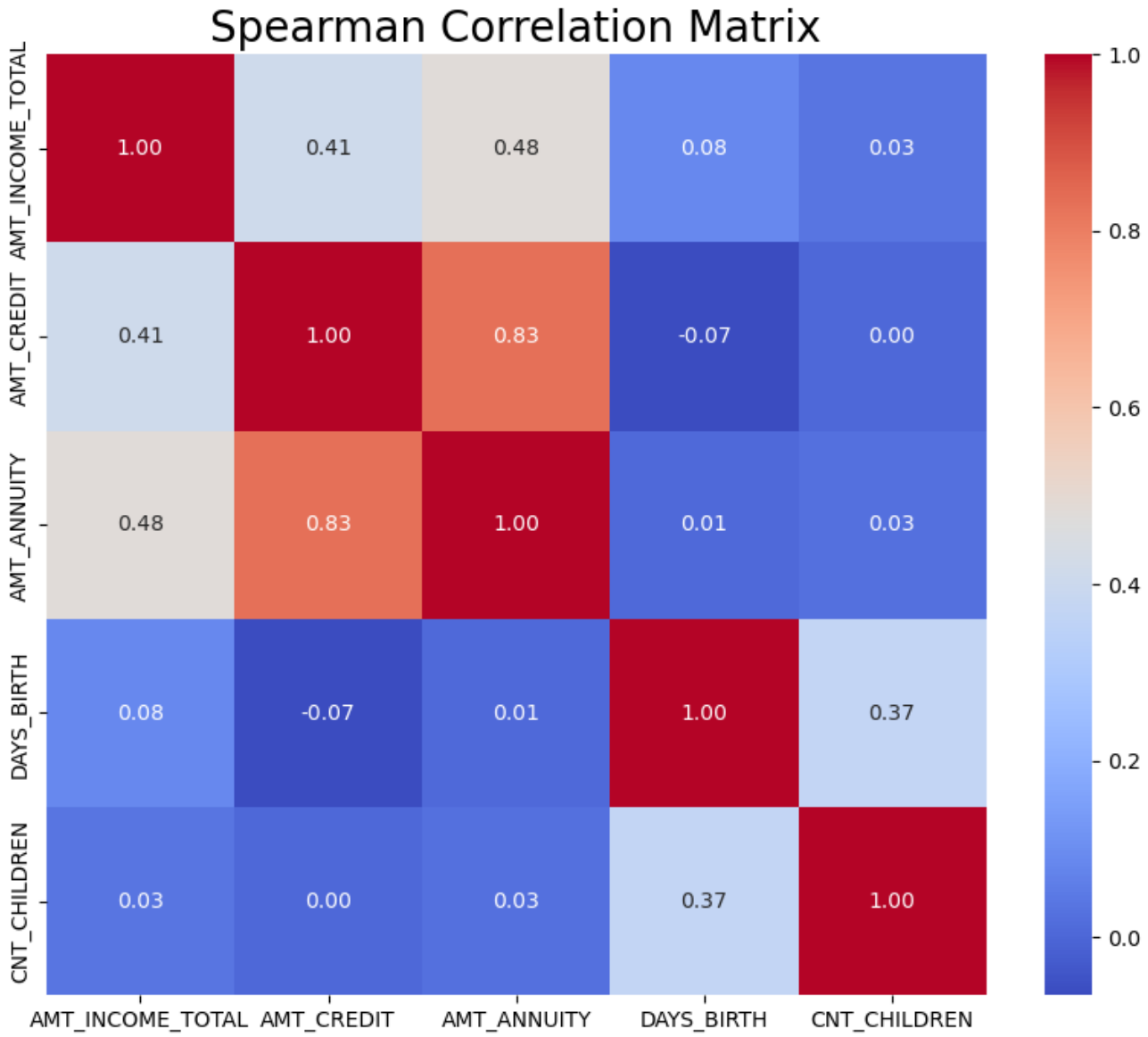

```
Plotting for EXPERIENCE_RANGE - Defaulters:
Warning: EXPERIENCE_RANGE does not exist in the DataFrame.

Plotting for EXPERIENCE_RANGE - Non-defaulters:
Warning: EXPERIENCE_RANGE does not exist in the DataFrame.

Plotting for AGE_GROUP - Defaulters:
Warning: AGE_GROUP does not exist in the DataFrame.

Plotting for AGE_GROUP - Non-defaulters:
Warning: AGE_GROUP does not exist in the DataFrame.
```

In [84]:
```
# Bivariate Analysis - Additional Correlation Measures

numeric_data = application.select_dtypes(include=['float64', 'int64'])

# Check that numeric_data is defined and has data
print(numeric_data.head())  # Display first few rows
print(numeric_data.info())  # Get info about DataFrame

# Select specific columns
selected_columns = ['AMT_INCOME_TOTAL', 'AMT_CREDIT', 'AMT_ANNUITY', 'DAYS_E
numeric_data_subset = numeric_data[selected_columns]

# Calculate Spearman correlation
spearman_corr = numeric_data_subset.corr(method='spearman')
```

```
# Print the correlation matrix
print(spearman_corr)

# Plot the Spearman correlation matrix
plt.figure(figsize=[10, 8])
sns.heatmap(spearman_corr, annot=True, fmt=".2f", cmap='coolwarm', square=Tr
plt.title('Spearman Correlation Matrix', fontsize=20)
plt.show()
```

```
   SK_ID_CURR  TARGET  CNT_CHILDREN  AMT_INCOME_TOTAL  AMT_CREDIT  \
0      100002       1             0          202500.0    406597.5
1      100003       0             0          270000.0   1293502.5
2      100004       0             0           67500.0    135000.0
3      100006       0             0          135000.0    312682.5
4      100007       0             0          121500.0    513000.0

   AMT_ANNUITY  AMT_GOODS_PRICE  REGION_POPULATION_RELATIVE  DAYS_BIRTH  \
0      24700.5         351000.0                    0.018801       -9461
1      35698.5        1129500.0                    0.003541      -16765
2       6750.0         135000.0                    0.010032      -19046
3      29686.5         297000.0                    0.008019      -19005
4      21865.5         513000.0                    0.028663      -19932

   DAYS_EMPLOYED  ...  FLAG_DOCUMENT_18  FLAG_DOCUMENT_19  FLAG_DOCUMENT_20
\
0           -637  ...                 0                 0                 0
1          -1188  ...                 0                 0                 0
2           -225  ...                 0                 0                 0
3          -3039  ...                 0                 0                 0
4          -3038  ...                 0                 0                 0

   FLAG_DOCUMENT_21  AMT_REQ_CREDIT_BUREAU_HOUR  AMT_REQ_CREDIT_BUREAU_DAY
\
0                 0                         0.0                        0.0
1                 0                         0.0                        0.0
2                 0                         0.0                        0.0
3                 0                         NaN                        NaN
4                 0                         0.0                        0.0

   AMT_REQ_CREDIT_BUREAU_WEEK  AMT_REQ_CREDIT_BUREAU_MON  \
0                         0.0                        0.0
1                         0.0                        0.0
2                         0.0                        0.0
3                         NaN                        NaN
4                         0.0                        0.0

   AMT_REQ_CREDIT_BUREAU_QRT  AMT_REQ_CREDIT_BUREAU_YEAR
0                        0.0                        1.0
1                        0.0                        0.0
```

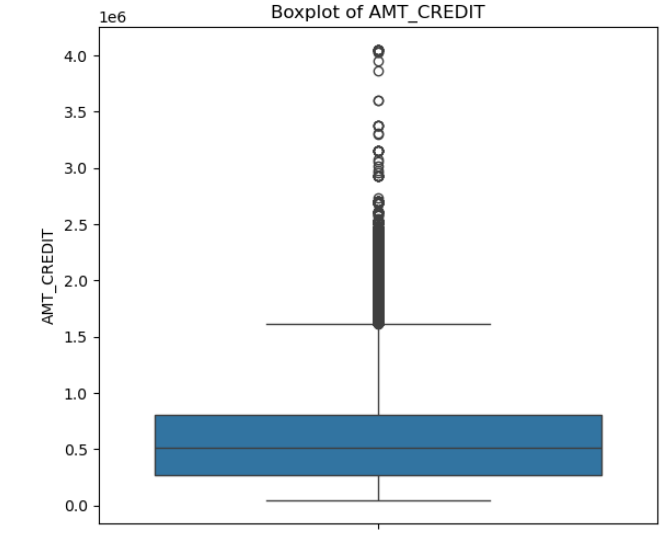
```
2                        0.0                          0.0
3                        NaN                          NaN
4                        0.0                          0.0

[5 rows x 106 columns]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Columns: 106 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
dtypes: float64(65), int64(41)
memory usage: 248.7 MB
None
                  AMT_INCOME_TOTAL  AMT_CREDIT  AMT_ANNUITY  DAYS_BIRTH  \
AMT_INCOME_TOTAL          1.000000    0.411876     0.481582    0.083973
AMT_CREDIT                0.411876    1.000000     0.830225   -0.066139
AMT_ANNUITY               0.481582    0.830225     1.000000    0.008338
DAYS_BIRTH                0.083973   -0.066139     0.008338    1.000000
CNT_CHILDREN              0.034464    0.001656     0.025454    0.367441

                  CNT_CHILDREN
AMT_INCOME_TOTAL      0.034464
AMT_CREDIT            0.001656
AMT_ANNUITY           0.025454
DAYS_BIRTH            0.367441
CNT_CHILDREN          1.000000
```

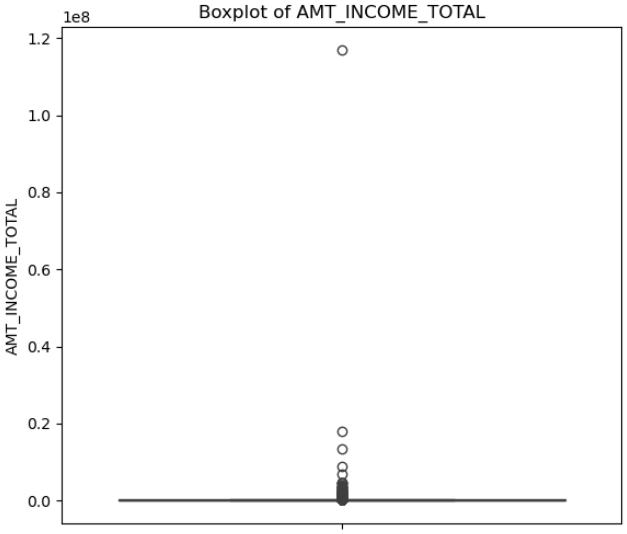


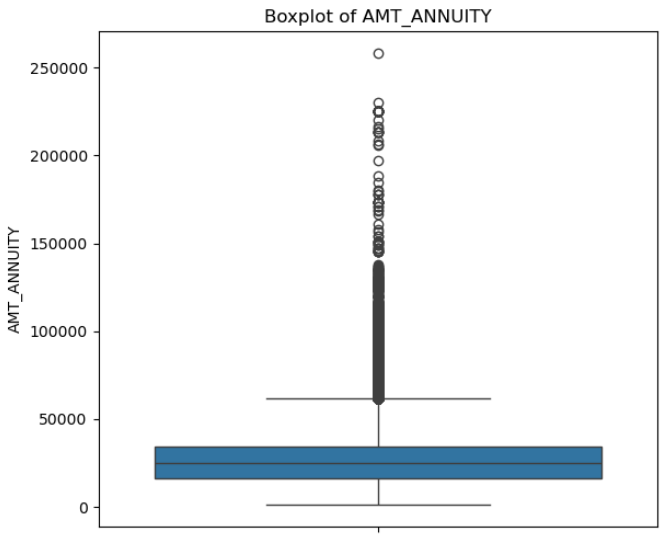
In [87]:
```python
# Outlier Detection

# Check available columns
print(application.columns)

# Ensure we have numeric data defined
numeric_data = application.select_dtypes(include=['float64', 'int64'])

# Variables to plot
variables_to_plot = ['AMT_CREDIT', 'AMT_INCOME_TOTAL', 'AMT_ANNUITY']

# Create boxplots for each variable
plt.figure(figsize=(12, 10))

for i, var in enumerate(variables_to_plot):
    plt.subplot(2, 2, i + 1)  # 2 rows, 2 columns
```

```
    sns.boxplot(y=numeric_data[var])  # Vertical boxplots
    plt.title(f'Boxplot of {var}')
    plt.ylabel(var)  # Y-axis label

plt.tight_layout()
plt.show()
```

```
Index(['SK_ID_CURR', 'TARGET', 'NAME_CONTRACT_TYPE', 'CODE_GENDER',
       'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'CNT_CHILDREN', 'AMT_INCOME_TOTA
L',
       'AMT_CREDIT', 'AMT_ANNUITY',
       ...
       'FLAG_DOCUMENT_18', 'FLAG_DOCUMENT_19', 'FLAG_DOCUMENT_20',
       'FLAG_DOCUMENT_21', '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_CREDIT_BUREAU_YEAR'],
      dtype='object', length=122)
```







In [ ]: