untitled

October 7, 2023

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

[2]:
```
app_data = pd.read_csv('application_data.csv')
```

[3]:
```
pre_app_data = pd.read_csv('previous_application.csv')
```

[4]:
```
app_data.head()
```

[4]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR
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

	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY
0	Y	0	202500.0	406597.5	24700.5
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

	…	FLAG_DOCUMENT_18	FLAG_DOCUMENT_19	FLAG_DOCUMENT_20	FLAG_DOCUMENT_21
0	…	0	0	0	0
1	…	0	0	0	0
2	…	0	0	0	0
3	…	0	0	0	0
4	…	0	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
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 122 columns]
```

[5]:
```
app_data.columns
```

[5]:
```
Index(['SK_ID_CURR', 'TARGET', 'NAME_CONTRACT_TYPE', 'CODE_GENDER',
       'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL',
       '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)
```

[6]:
```
app_data.shape
```

[6]:
```
(49999, 122)
```

[7]:
```
pre_app_data.shape
```

[7]:
```
(49999, 37)
```

[8]:
```
object_columns = app_data.select_dtypes(include=['object'])
```

[9]:
```
object_columns
```

[9]:
```
      NAME_CONTRACT_TYPE CODE_GENDER FLAG_OWN_CAR FLAG_OWN_REALTY  \
0             Cash loans           M            N               Y
1             Cash loans           F            N               N
2        Revolving loans           M            Y               Y
```

```
3            Cash loans           F            N               Y
4            Cash loans           M            N               Y
…                 …         …          …            …
49994        Cash loans           F            N               N
49995        Cash loans           M            N               N
49996        Cash loans           M            N               N
49997        Cash loans           F            N               Y
49998        Cash loans           F            N               Y

      NAME_TYPE_SUITE      NAME_INCOME_TYPE            NAME_EDUCATION_TYPE  \
0       Unaccompanied               Working  Secondary / secondary special
1              Family         State servant               Higher education
2       Unaccompanied               Working  Secondary / secondary special
3       Unaccompanied               Working  Secondary / secondary special
4       Unaccompanied               Working  Secondary / secondary special
…                 …                     …                           …
49994             NaN               Working               Higher education
49995   Unaccompanied  Commercial associate  Secondary / secondary special
49996   Unaccompanied               Working  Secondary / secondary special
49997          Family               Working  Secondary / secondary special
49998   Unaccompanied             Pensioner               Higher education

         NAME_FAMILY_STATUS  NAME_HOUSING_TYPE       OCCUPATION_TYPE  \
0      Single / not married  House / apartment             Laborers
1                   Married  House / apartment           Core staff
2      Single / not married  House / apartment             Laborers
3            Civil marriage  House / apartment             Laborers
4      Single / not married  House / apartment           Core staff
…                      …                 …                     …
49994  Single / not married  House / apartment  Waiters/barmen staff
49995               Married  House / apartment             Laborers
49996               Married       With parents                   NaN
49997               Married  House / apartment        Cleaning staff
49998               Married  House / apartment                   NaN

      WEEKDAY_APPR_PROCESS_START       ORGANIZATION_TYPE FONDKAPREMONT_MODE  \
0                      WEDNESDAY  Business Entity Type 3   reg oper account
1                         MONDAY                  School   reg oper account
2                         MONDAY              Government                NaN
3                      WEDNESDAY  Business Entity Type 3                NaN
4                       THURSDAY                Religion                NaN
…                            …                       …                …
49994                  WEDNESDAY              Restaurant                NaN
49995                    TUESDAY            Construction   reg oper account
49996                     MONDAY  Business Entity Type 1                NaN
49997                   THURSDAY                   Other                NaN
49998                     MONDAY                     XNA                NaN
```

```
       HOUSETYPE_MODE WALLSMATERIAL_MODE EMERGENCYSTATE_MODE
0      block of flats       Stone, brick                  No
1      block of flats              Block                  No
2                 NaN                NaN                 NaN
3                 NaN                NaN                 NaN
4                 NaN                NaN                 NaN
…                 …                  …                   …
49994             NaN                NaN                 NaN
49995  block of flats              Block                  No
49996             NaN                NaN                 NaN
49997             NaN                NaN                 NaN
49998             NaN       Stone, brick                  No

[49999 rows x 16 columns]
```

[10]:
```
numerical_columns = app_data.select_dtypes(include=['int64', 'float64'])
```

[11]:
```
numerical_columns
```

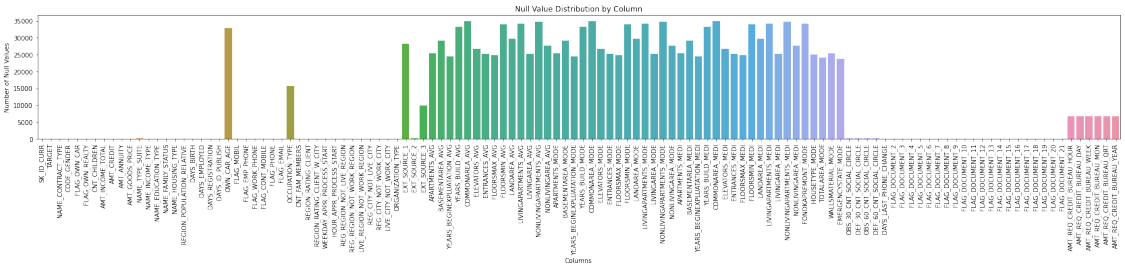
[11]:
```
       SK_ID_CURR  TARGET  CNT_CHILDREN  AMT_INCOME_TOTAL  AMT_CREDIT  \
0          100002       1             0          202500.0    406597.5
1          100003       0             0          270000.0   1293502.5
2          100004       0             0           67500.0    135000.0
3          100006       0             0          135000.0    312682.5
4          100007       0             0          121500.0    513000.0
…             …     …            …                …          …
49994      157871       0             0          180000.0   1206000.0
49995      157872       0             0          126000.0   1125000.0
49996      157873       0             1          112500.0    900000.0
49997      157874       0             0          270000.0    820638.0
49998      157875       0             0          117000.0    254700.0

       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
…            …              …                          …          …
49994      45936.0        1206000.0                    0.035792      -10667
49995      47794.5        1125000.0                    0.015221      -20211
49996      26316.0         900000.0                    0.025164      -10280
49997      34897.5         733500.0                    0.022625      -23485
49998      14751.0         225000.0                    0.005084      -19251

       DAYS_EMPLOYED  …  FLAG_DOCUMENT_18  FLAG_DOCUMENT_19  \
```

```
0               -637  …                0                0
1              -1188  …                0                0
2               -225  …                0                0
3              -3039  …                0                0
4              -3038  …                0                0
…                …  …                …                …
49994           -285  …                0                0
49995          -4651  …                0                0
49996          -1158  …                0                0
49997          -2181  …                0                0
49998         365243  …                0                0

       FLAG_DOCUMENT_20  FLAG_DOCUMENT_21  AMT_REQ_CREDIT_BUREAU_HOUR  \
0                     0                 0                         0.0
1                     0                 0                         0.0
2                     0                 0                         0.0
3                     0                 0                         NaN
4                     0                 0                         0.0
…                   …                 …                         …
49994                 0                 0                         0.0
49995                 0                 0                         0.0
49996                 0                 0                         0.0
49997                 0                 0                         0.0
49998                 0                 0                         0.0

       AMT_REQ_CREDIT_BUREAU_DAY  AMT_REQ_CREDIT_BUREAU_WEEK  \
0                            0.0                         0.0
1                            0.0                         0.0
2                            0.0                         0.0
3                            NaN                         NaN
4                            0.0                         0.0
…                          …                           …
49994                        0.0                         0.0
49995                        0.0                         0.0
49996                        0.0                         0.0
49997                        0.0                         0.0
49998                        0.0                         0.0

       AMT_REQ_CREDIT_BUREAU_MON  AMT_REQ_CREDIT_BUREAU_QRT  \
0                            0.0                         0.0
1                            0.0                         0.0
2                            0.0                         0.0
3                            NaN                         NaN
4                            0.0                         0.0
…                          …                           …
49994                        0.0                         0.0
49995                        0.0                         0.0
```

```
49996                         0.0                          0.0
49997                         0.0                          2.0
49998                         0.0                          0.0

       AMT_REQ_CREDIT_BUREAU_YEAR
0                             1.0
1                             0.0
2                             0.0
3                             NaN
4                             0.0
…                             …
49994                         0.0
49995                         0.0
49996                         2.0
49997                         4.0
49998                         0.0

[49999 rows x 106 columns]
```

[12]:
```
null_counts = app_data.isnull().sum()
```

[13]:
```
plt.figure(figsize=(25, 6))
sns.barplot(x=null_counts.index, y=null_counts.values)
plt.xticks(rotation=90)
plt.xlabel('Columns')
plt.ylabel('Number of Null Values')
plt.title('Null Value Distribution by Column')
plt.tight_layout()  # Ensures labels are not cut off
plt.show()
```



[14]:
```
print(null_counts)
```

```
SK_ID_CURR                    0
TARGET                        0
NAME_CONTRACT_TYPE            0
CODE_GENDER                   0
FLAG_OWN_CAR                  0
```

```
                                ...
AMT_REQ_CREDIT_BUREAU_DAY       6734
AMT_REQ_CREDIT_BUREAU_WEEK      6734
AMT_REQ_CREDIT_BUREAU_MON       6734
AMT_REQ_CREDIT_BUREAU_QRT       6734
AMT_REQ_CREDIT_BUREAU_YEAR      6734
Length: 122, dtype: int64
```

[15]:
```
app_data = app_data.dropna(axis=1)
```

[16]:
```
app_data.shape
```

[16]:
```
(49999, 55)
```

[17]:
```
app_data.head()
```

[17]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	\
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	

	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	\
0	Y	0	202500.0	406597.5	
1	N	0	270000.0	1293502.5	
2	Y	0	67500.0	135000.0	
3	Y	0	135000.0	312682.5	
4	Y	0	121500.0	513000.0	

	NAME_INCOME_TYPE	…	FLAG_DOCUMENT_12	FLAG_DOCUMENT_13	FLAG_DOCUMENT_14	\
0	Working	…	0	0	0	
1	State servant	…	0	0	0	
2	Working	…	0	0	0	
3	Working	…	0	0	0	
4	Working	…	0	0	0	

	FLAG_DOCUMENT_15	FLAG_DOCUMENT_16	FLAG_DOCUMENT_17	FLAG_DOCUMENT_18	\
0	0	0	0	0	
1	0	0	0	0	
2	0	0	0	0	
3	0	0	0	0	
4	0	0	0	0	

	FLAG_DOCUMENT_19	FLAG_DOCUMENT_20	FLAG_DOCUMENT_21
0	0	0	0
1	0	0	0

```
2                0                0                0
3                0                0                0
4                0                0                0

[5 rows x 55 columns]
```

[18]:
```
app_data.columns
```

[18]:
```
Index(['SK_ID_CURR', 'TARGET', 'NAME_CONTRACT_TYPE', 'CODE_GENDER',
       'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL',
       'AMT_CREDIT', 'NAME_INCOME_TYPE', 'NAME_EDUCATION_TYPE',
       'NAME_FAMILY_STATUS', 'NAME_HOUSING_TYPE', 'REGION_POPULATION_RELATIVE',
       'DAYS_BIRTH', 'DAYS_EMPLOYED', 'DAYS_REGISTRATION', 'DAYS_ID_PUBLISH',
       'FLAG_MOBIL', 'FLAG_EMP_PHONE', 'FLAG_WORK_PHONE', 'FLAG_CONT_MOBILE',
       'FLAG_PHONE', 'FLAG_EMAIL', 'REGION_RATING_CLIENT',
       'REGION_RATING_CLIENT_W_CITY', 'WEEKDAY_APPR_PROCESS_START',
       'HOUR_APPR_PROCESS_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', 'FLAG_DOCUMENT_2',
       'FLAG_DOCUMENT_3', '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'],
      dtype='object')
```

[19]:
```
columns_drop=['FLAG_MOBIL', 'FLAG_EMP_PHONE', 'FLAG_WORK_PHONE',␣
↪'FLAG_CONT_MOBILE', 'FLAG_PHONE',
                  'FLAG_EMAIL', 'FLAG_DOCUMENT_2', 'FLAG_DOCUMENT_3',␣
↪'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']
```

[20]:
```
app_data = app_data.drop(columns=columns_drop)
```

[21]:
```
app_data.shape
```

[21]:
```
(49999, 29)
```

[22]:
```
app_data.head()
```

[22]:

```
   SK_ID_CURR  TARGET NAME_CONTRACT_TYPE CODE_GENDER FLAG_OWN_CAR  \
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

  FLAG_OWN_REALTY  CNT_CHILDREN  AMT_INCOME_TOTAL  AMT_CREDIT  \
0               Y             0          202500.0    406597.5
1               N             0          270000.0   1293502.5
2               Y             0           67500.0    135000.0
3               Y             0          135000.0    312682.5
4               Y             0          121500.0    513000.0

  NAME_INCOME_TYPE  … REGION_RATING_CLIENT_W_CITY  \
0          Working  …                           2
1    State servant  …                           1
2          Working  …                           2
3          Working  …                           2
4          Working  …                           2

  WEEKDAY_APPR_PROCESS_START HOUR_APPR_PROCESS_START  \
0                  WEDNESDAY                      10
1                     MONDAY                      11
2                     MONDAY                       9
3                  WEDNESDAY                      17
4                   THURSDAY                      11

   REG_REGION_NOT_LIVE_REGION  REG_REGION_NOT_WORK_REGION  \
0                           0                           0
1                           0                           0
2                           0                           0
3                           0                           0
4                           0                           0

   LIVE_REGION_NOT_WORK_REGION  REG_CITY_NOT_LIVE_CITY  \
0                            0                       0
1                            0                       0
2                            0                       0
3                            0                       0
4                            0                       0

   REG_CITY_NOT_WORK_CITY  LIVE_CITY_NOT_WORK_CITY       ORGANIZATION_TYPE
0                       0                       0  Business Entity Type 3
1                       0                       0                  School
```

```
2                       0                       0              Government
3                       0                       0  Business Entity Type 3
4                       1                       1                Religion

[5 rows x 29 columns]
```

[23]:
```
columns_convert = ['DAYS_BIRTH', 'DAYS_EMPLOYED', 'DAYS_REGISTRATION',␣
↪'DAYS_ID_PUBLISH']

app_data[columns_convert] = app_data[columns_convert].abs()
```

[24]:
```
app_data[columns_convert].describe()
```

[24]:

	DAYS_BIRTH	DAYS_EMPLOYED	DAYS_REGISTRATION	DAYS_ID_PUBLISH
count	49999.000000	49999.000000	49999.000000	49999.000000
mean	16022.042081	67160.324386	4977.282666	2996.797176
std	4361.400270	138957.897110	3525.548305	1509.235410
min	7680.000000	0.000000	0.000000	0.000000
25%	12378.500000	933.000000	1998.000000	1722.000000
50%	15731.000000	2216.000000	4490.000000	3261.000000
75%	19644.000000	5718.000000	7463.500000	4297.000000
max	25184.000000	365243.000000	22392.000000	6232.000000

[25]:
```
app_data.isnull().sum()
```

[25]:
```
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
NAME_INCOME_TYPE              0
NAME_EDUCATION_TYPE           0
NAME_FAMILY_STATUS            0
NAME_HOUSING_TYPE             0
REGION_POPULATION_RELATIVE    0
DAYS_BIRTH                    0
DAYS_EMPLOYED                 0
DAYS_REGISTRATION             0
DAYS_ID_PUBLISH               0
REGION_RATING_CLIENT          0
REGION_RATING_CLIENT_W_CITY   0
WEEKDAY_APPR_PROCESS_START    0
HOUR_APPR_PROCESS_START       0
```

```
REG_REGION_NOT_LIVE_REGION     0
REG_REGION_NOT_WORK_REGION     0
LIVE_REGION_NOT_WORK_REGION    0
REG_CITY_NOT_LIVE_CITY         0
REG_CITY_NOT_WORK_CITY         0
LIVE_CITY_NOT_WORK_CITY        0
ORGANIZATION_TYPE              0
dtype: int64
```

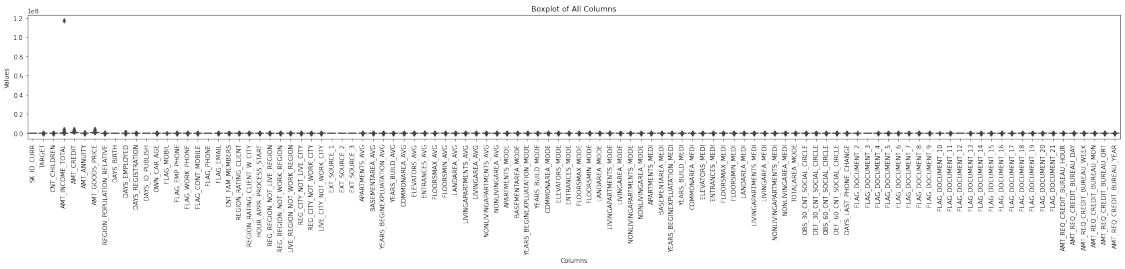
[26]:
```
app_data.dtypes
```

[26]:
```
SK_ID_CURR                     int64
TARGET                         int64
NAME_CONTRACT_TYPE            object
CODE_GENDER                   object
FLAG_OWN_CAR                  object
FLAG_OWN_REALTY               object
CNT_CHILDREN                   int64
AMT_INCOME_TOTAL             float64
AMT_CREDIT                   float64
NAME_INCOME_TYPE              object
NAME_EDUCATION_TYPE           object
NAME_FAMILY_STATUS            object
NAME_HOUSING_TYPE             object
REGION_POPULATION_RELATIVE   float64
DAYS_BIRTH                     int64
DAYS_EMPLOYED                  int64
DAYS_REGISTRATION              int64
DAYS_ID_PUBLISH                int64
REGION_RATING_CLIENT           int64
REGION_RATING_CLIENT_W_CITY    int64
WEEKDAY_APPR_PROCESS_START    object
HOUR_APPR_PROCESS_START        int64
REG_REGION_NOT_LIVE_REGION     int64
REG_REGION_NOT_WORK_REGION     int64
LIVE_REGION_NOT_WORK_REGION    int64
REG_CITY_NOT_LIVE_CITY         int64
REG_CITY_NOT_WORK_CITY         int64
LIVE_CITY_NOT_WORK_CITY        int64
ORGANIZATION_TYPE             object
dtype: object
```

0.1 Outliers detection

0.1.1 For numerical columns

[27]:
```
plt.figure(figsize=(25, 6))  # Adjust the figure size as needed

# Use seaborn to create a boxplot for each column
sns.boxplot(data=numerical_columns)

# Rotate x-axis labels for better readability
plt.xticks(rotation=90)

# Set labels and title
plt.xlabel('Columns')
plt.ylabel('Values')
plt.title('Boxplot of All Columns')

# Show the plot
plt.tight_layout()  # Ensures labels are not cut off
plt.show()
```

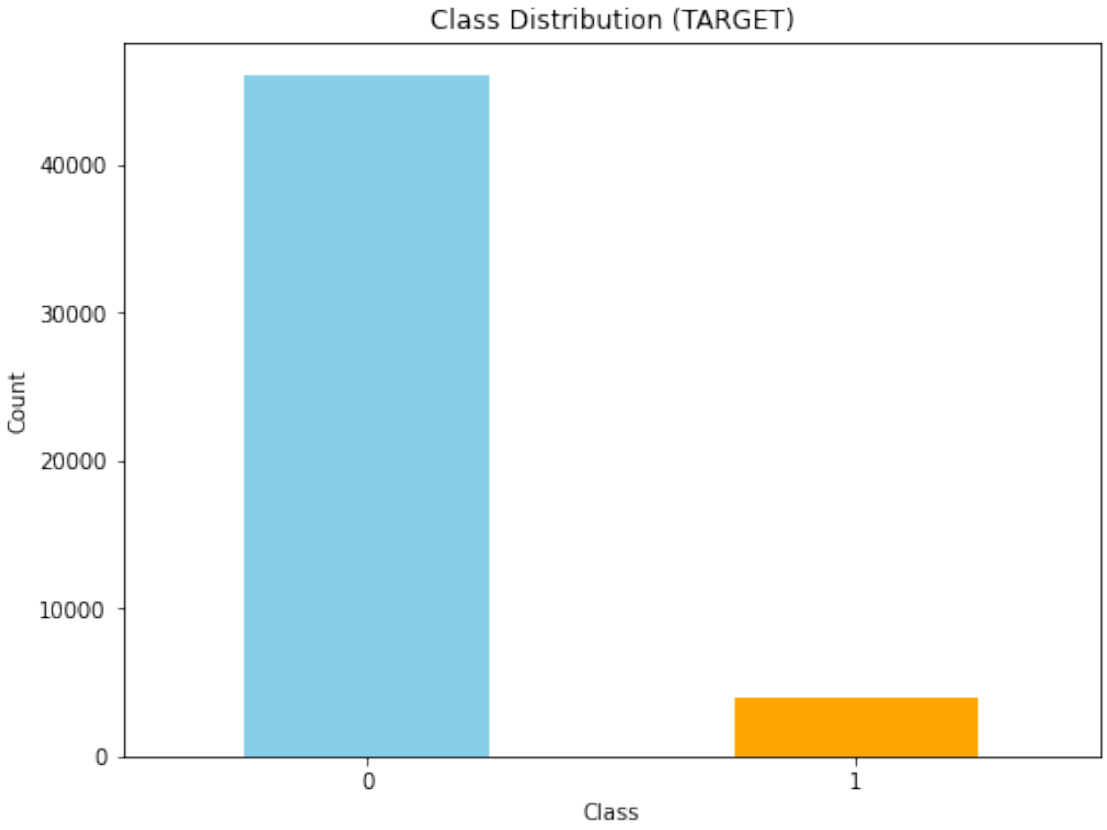


0.1.2 Data imbalance analysis

[28]:
```
class_distribution = app_data['TARGET'].value_counts()

# Print the class distribution
print("Class Distribution:")
print(class_distribution)

# Step 2: Visualize class distribution
plt.figure(figsize=(8, 6))
app_data['TARGET'].value_counts().plot(kind='bar', color=['skyblue', 'orange'])
plt.title('Class Distribution (TARGET)')
plt.xlabel('Class')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.show()
```

```
# Step 3: Calculate class imbalance ratio
majority_class_count = app_data['TARGET'].value_counts().max()
minority_class_count = app_data['TARGET'].value_counts().min()
imbalance_ratio = majority_class_count / minority_class_count

print(f'Imbalance Ratio: {imbalance_ratio:.2f}')
```

```
Class Distribution:
0    45973
1     4026
Name: TARGET, dtype: int64
```

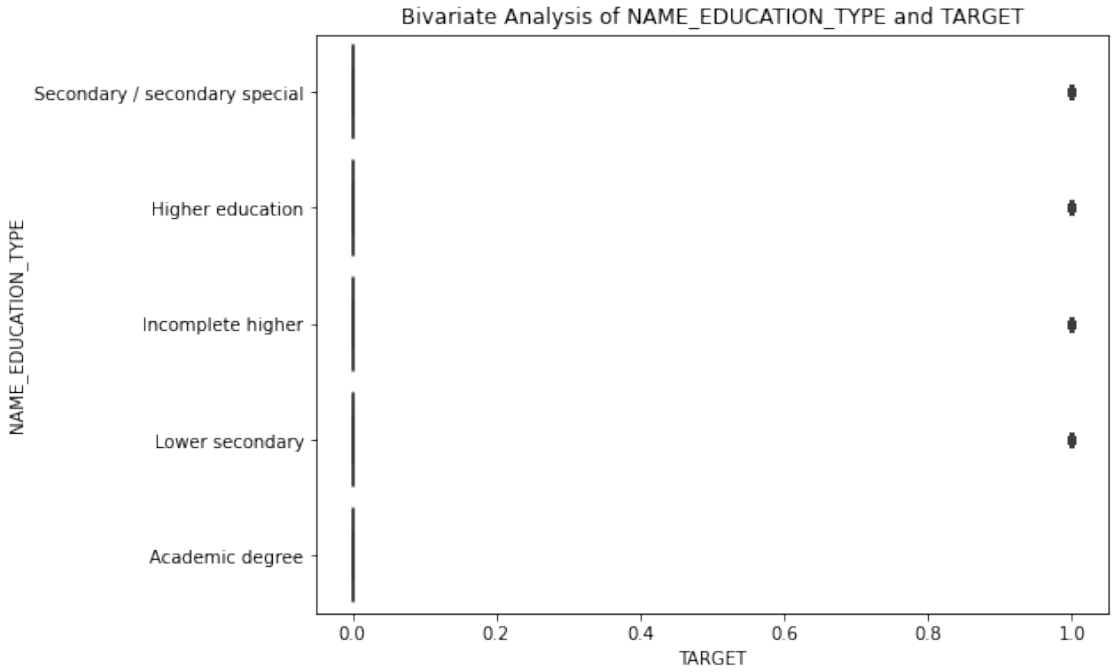


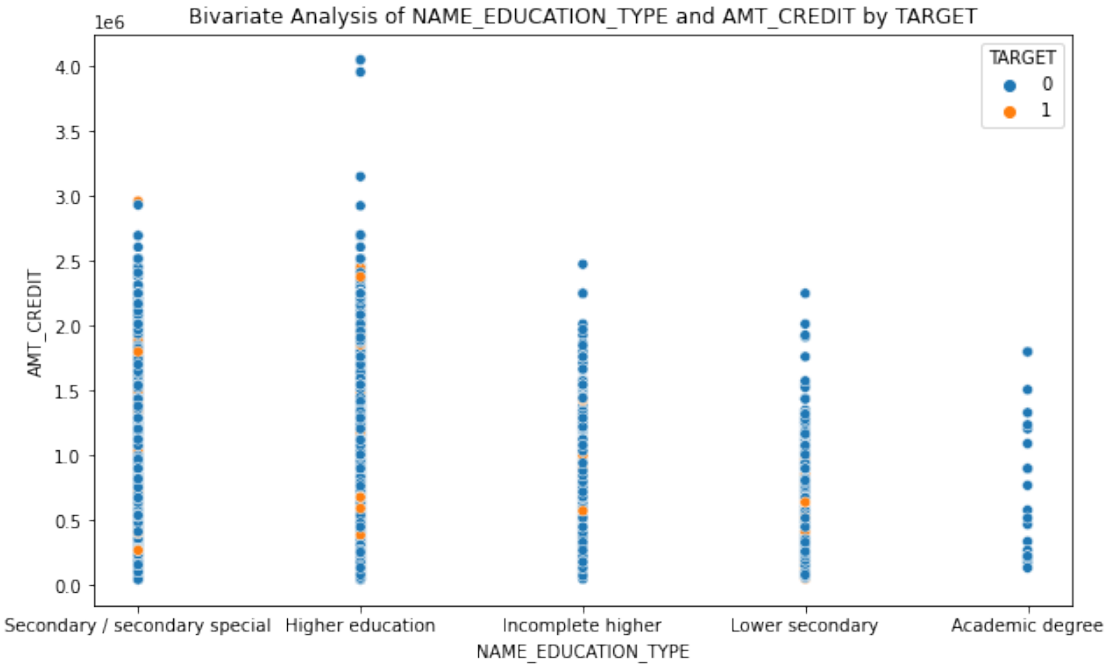

```
Imbalance Ratio: 11.42
```

0.1.3 Bivariate

[29]:
```
plt.figure(figsize=(8, 6))
sns.boxplot(data=app_data, x='TARGET', y='NAME_EDUCATION_TYPE')
plt.title('Bivariate Analysis of NAME_EDUCATION_TYPE and TARGET')
plt.xlabel('TARGET')
```

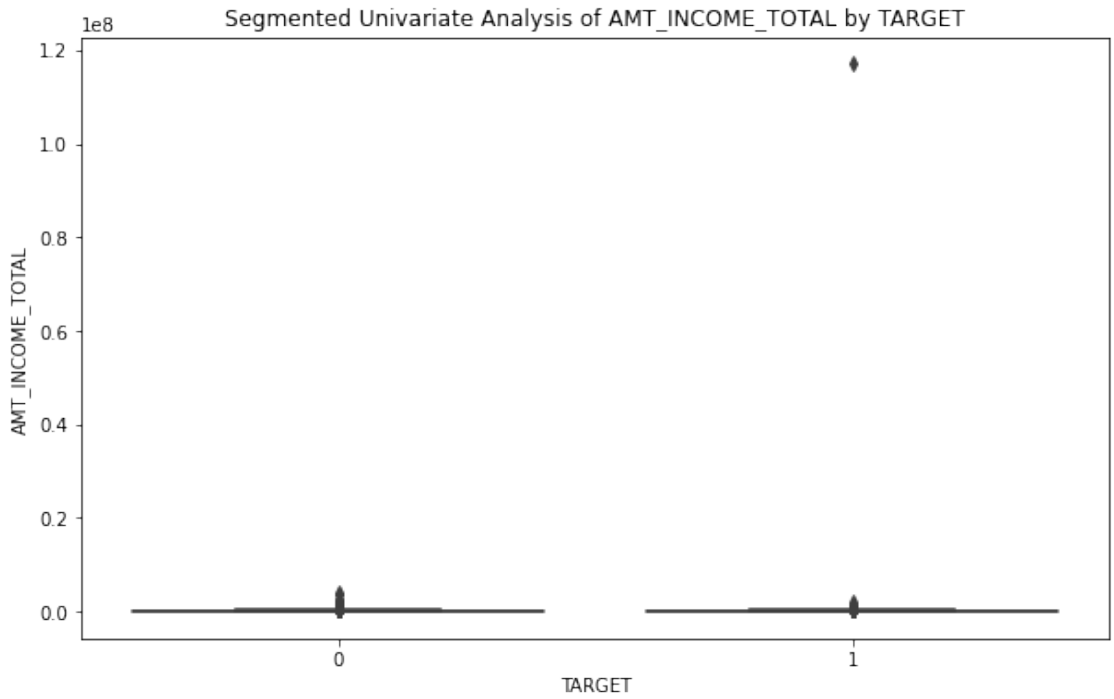
```
plt.ylabel('NAME_EDUCATION_TYPE')
plt.show()

# Bivariate analysis between two numeric variables (e.g., AMT_INCOME_TOTAL and␣
 ↪AMT_CREDIT)
plt.figure(figsize=(10, 6))
sns.scatterplot(data=app_data, x='NAME_EDUCATION_TYPE', y='AMT_CREDIT',␣
 ↪hue='TARGET')
plt.title('Bivariate Analysis of NAME_EDUCATION_TYPE and AMT_CREDIT by TARGET')
plt.xlabel('NAME_EDUCATION_TYPE')
plt.ylabel('AMT_CREDIT')
plt.show()
```





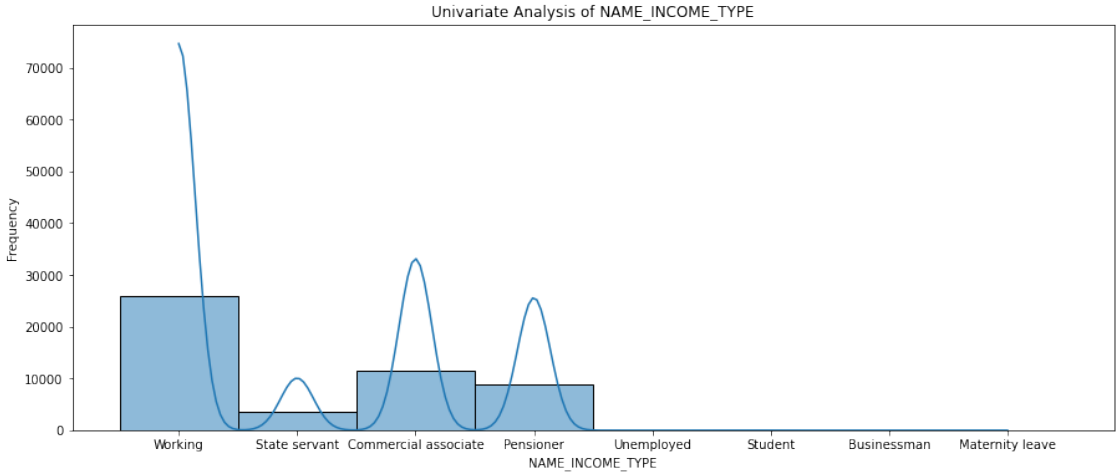
0.1.4 Segmented Univariate Analysis:

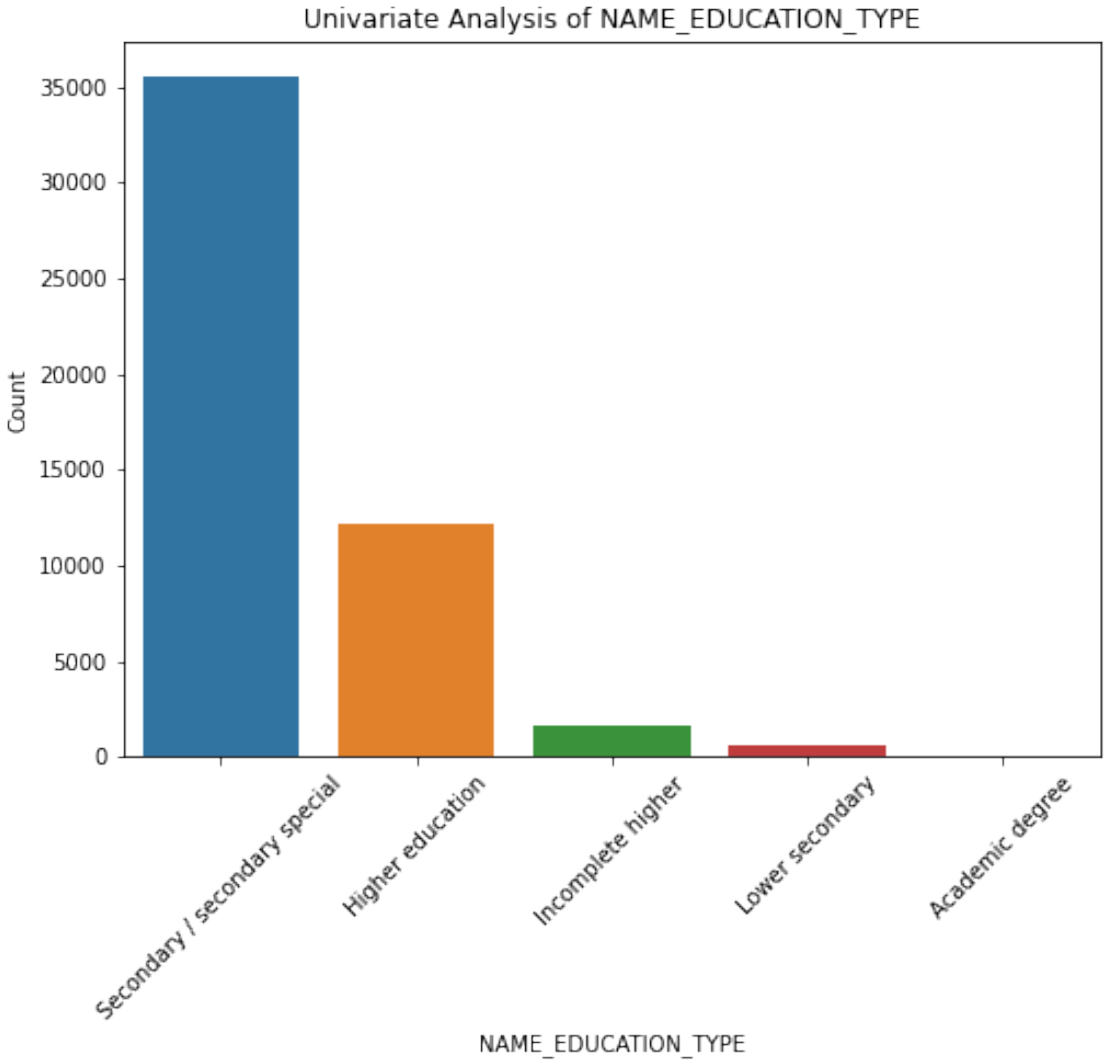
[30]:
```
# Segmented univariate analysis for a numeric variable by a categorical variable
plt.figure(figsize=(10, 6))
sns.boxplot(data=app_data, x='TARGET', y='AMT_INCOME_TOTAL')
plt.title('Segmented Univariate Analysis of AMT_INCOME_TOTAL by TARGET')
plt.xlabel('TARGET')
plt.ylabel('AMT_INCOME_TOTAL')
plt.show()
```



0.2 Univariate Analysis:

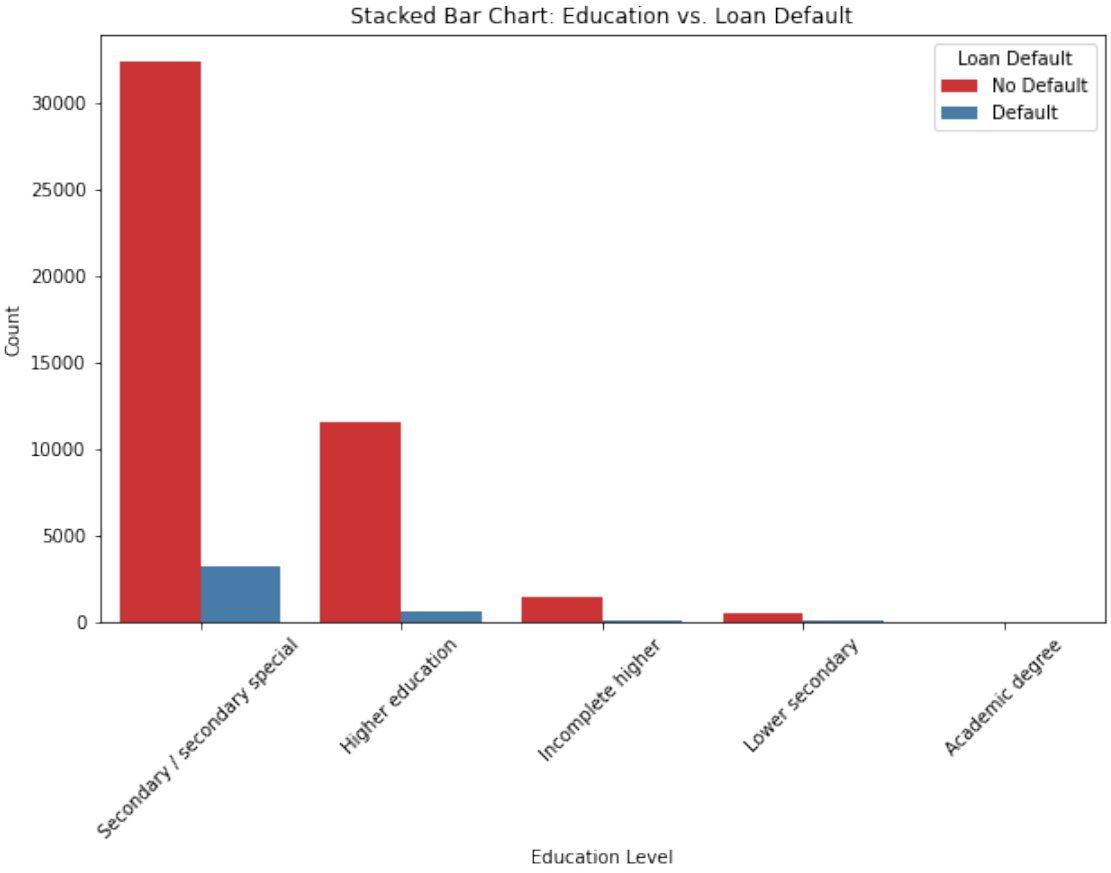
[31]:
```
numeric_variable = 'NAME_INCOME_TYPE'
plt.figure(figsize=(15, 6))
sns.histplot(data=app_data, x=numeric_variable, kde=True)
plt.title(f'Univariate Analysis of {numeric_variable}')
plt.xlabel(numeric_variable)
plt.ylabel('Frequency')
plt.show()

# Univariate analysis for a categorical variable (e.g., NAME_EDUCATION_TYPE)
categorical_variable = 'NAME_EDUCATION_TYPE'
plt.figure(figsize=(8, 6))
sns.countplot(data=app_data, x=categorical_variable)
plt.title(f'Univariate Analysis of {categorical_variable}')
plt.xlabel(categorical_variable)
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```

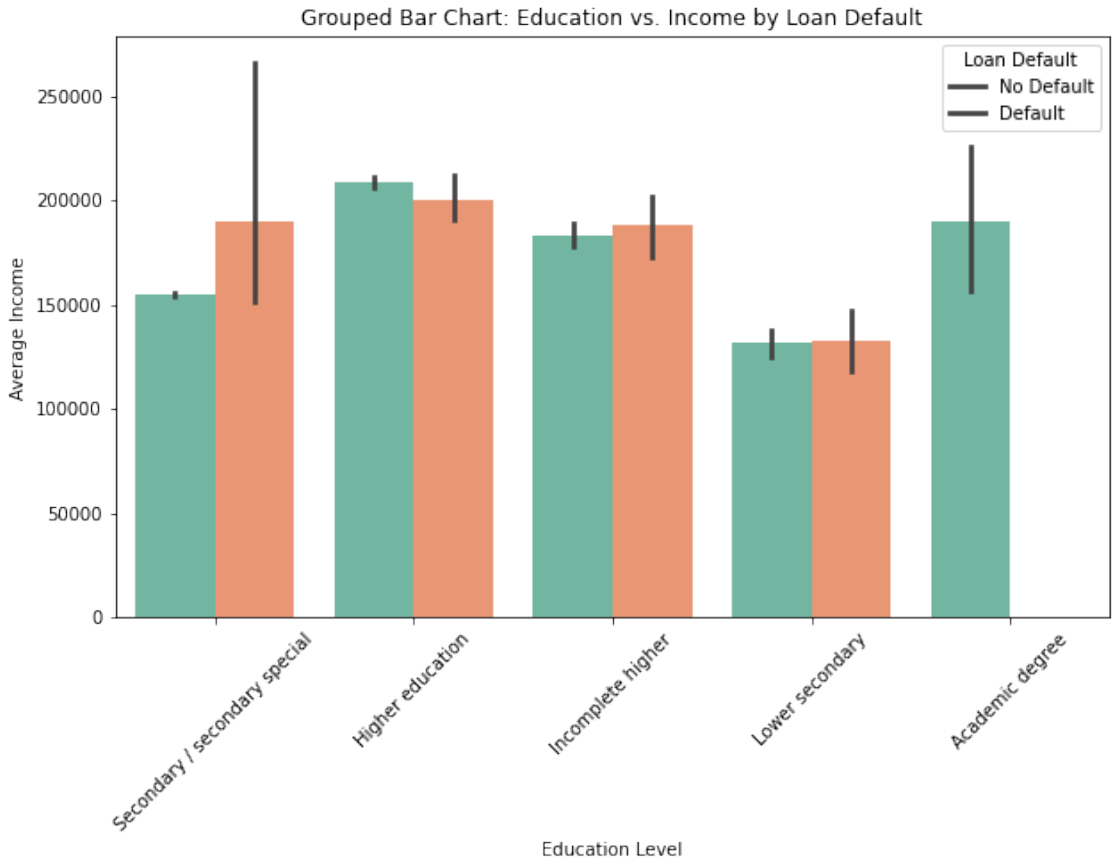

Univariate Analysis of NAME_INCOME_TYPE
Frequency
70000
60000
50000
40000
30000
20000
10000
0
Working
State servant
Commercial associate
Pensioner
Unemployed
Student
Businessman
Maternity leave
NAME_INCOME_TYPE


Univariate Analysis of NAME_EDUCATION_TYPE
Count
35000
30000
25000
20000
15000
10000
5000
0
Secondary / secondary special
Higher education
Incomplete higher
Lower secondary
Academic degree
NAME_EDUCATION_TYPE

0.2.1 Stacked Bar chart

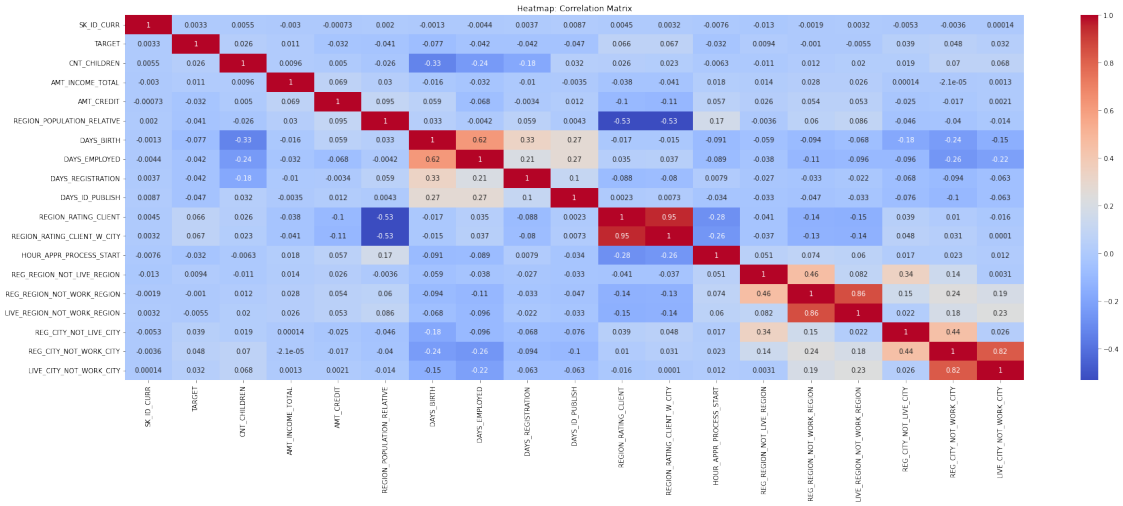
[32]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=app_data, x='NAME_EDUCATION_TYPE', hue='TARGET',␣
 ↪palette='Set1')
plt.title('Stacked Bar Chart: Education vs. Loan Default')
plt.xlabel('Education Level')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.legend(title='Loan Default', loc='upper right', labels=['No Default',␣
 ↪'Default'])
plt.show()

# Create a grouped bar chart for comparing variable distributions across␣
 ↪scenarios
plt.figure(figsize=(10, 6))
sns.barplot(data=app_data, x='NAME_EDUCATION_TYPE', y='AMT_INCOME_TOTAL',␣
 ↪hue='TARGET', palette='Set2')
plt.title('Grouped Bar Chart: Education vs. Income by Loan Default')
plt.xlabel('Education Level')
plt.ylabel('Average Income')
plt.xticks(rotation=45)
plt.legend(title='Loan Default', loc='upper right', labels=['No Default',␣
 ↪'Default'])
plt.show()
```


Stacked Bar Chart: Education vs. Loan Default
Loan Default
No Default
Default
Count
30000
25000
20000
15000
10000
5000
0
Secondary / secondary special
Higher education
Incomplete higher
Lower secondary
Academic degree
Education Level

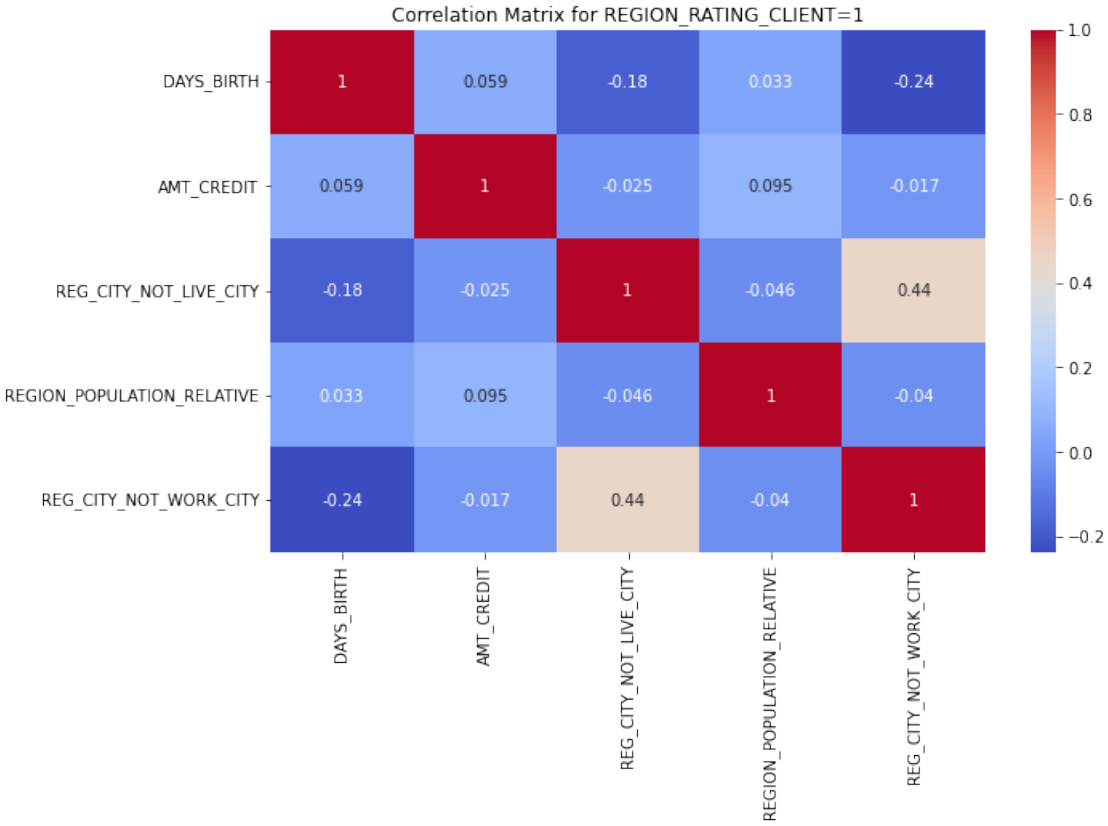


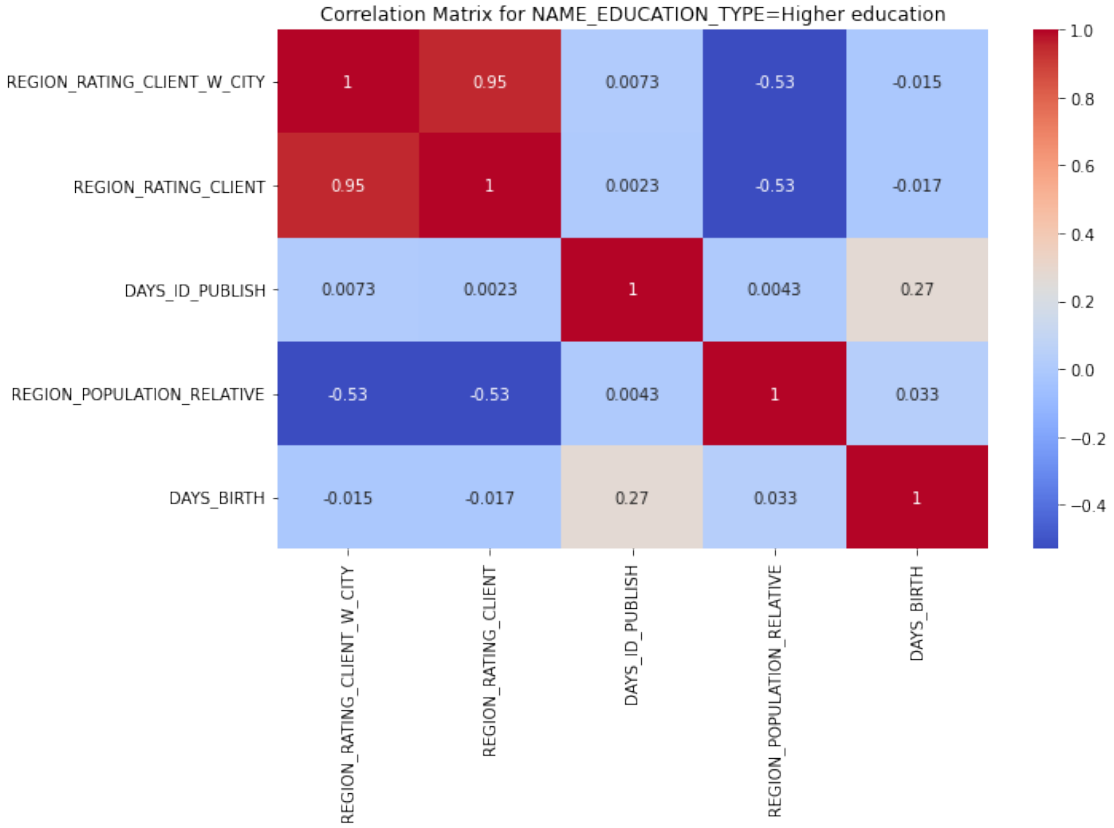
0.2.2 Heatmap

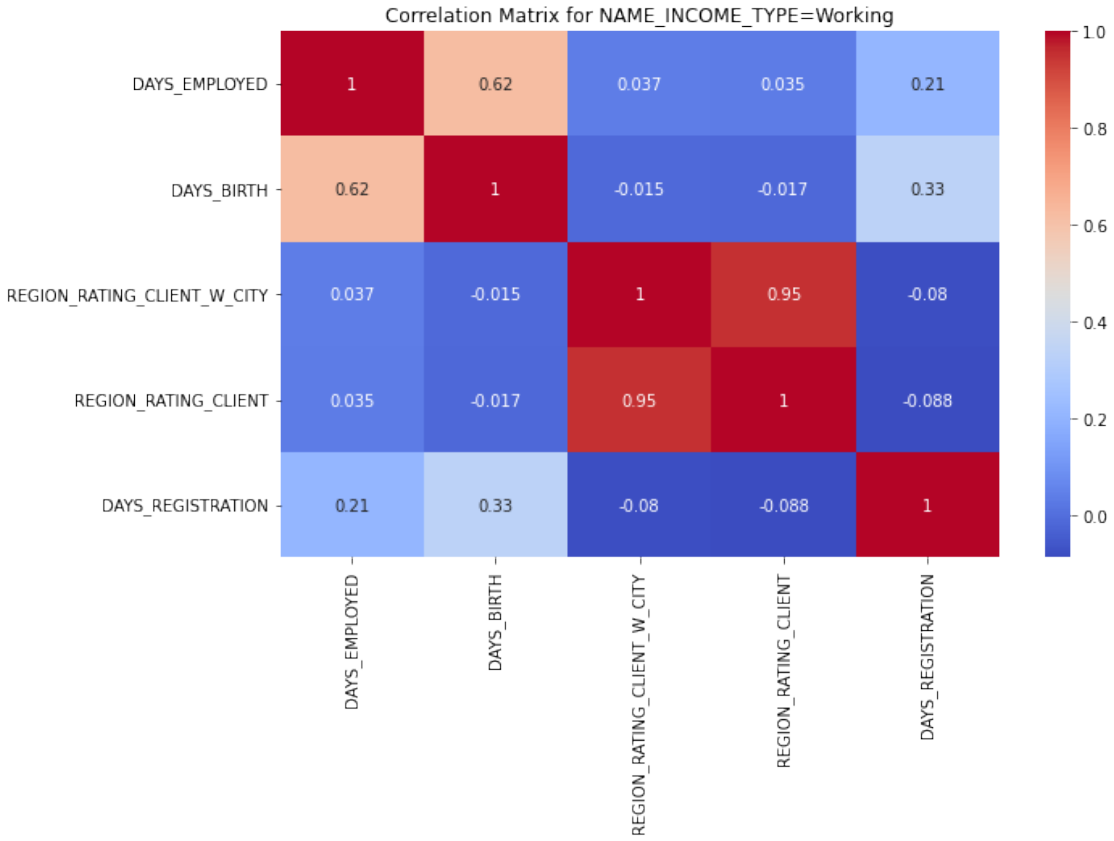
[33]:
```
correlation_matrix = app_data.corr()
plt.figure(figsize=(30, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Heatmap: Correlation Matrix')
plt.show()
```



[34]:
```
def visualize_top_correlations(app_data, scenario_variable, scenario_value,␣
↪target_variable, top_n=5):
    # Filter the DataFrame for the specific scenario
    scenario_df = app_data[app_data[scenario_variable] == scenario_value]

    # Calculate correlations with the target variable
    correlations = scenario_df.corr()[target_variable].drop(target_variable)

    # Get the top correlated variables
    top_correlations = correlations.abs().nlargest(top_n)

    # Create a heatmap for the top correlated variables
    plt.figure(figsize=(10, 6))
    sns.heatmap(app_data[top_correlations.index].corr(), annot=True,␣
↪cmap='coolwarm')
    plt.title(f'Correlation Matrix for {scenario_variable}={scenario_value}')
    plt.show()

# Example usage for different scenarios
visualize_top_correlations(app_data, 'REGION_RATING_CLIENT', 1, 'TARGET')
visualize_top_correlations(app_data, 'NAME_EDUCATION_TYPE', 'Higher education',␣
↪'TARGET')
visualize_top_correlations(app_data, 'NAME_INCOME_TYPE', 'Working', 'TARGET')
```


Correlation Matrix for REGION_RATING_CLIENT=1
DAYS_BIRTH
AMT_CREDIT
REG_CITY_NOT_LIVE_CITY
REGION_POPULATION_RELATIVE
REG_CITY_NOT_WORK_CITY
1
0.059
-0.18
0.033
-0.24
0.059
1
-0.025
0.095
-0.017
-0.18
-0.025
1
-0.046
0.44
0.033
0.095
-0.046
1
-0.04
-0.24
-0.017
0.44
-0.04
1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2


Correlation Matrix for NAME_EDUCATION_TYPE=Higher education
REGION_RATING_CLIENT_W_CITY
REGION_RATING_CLIENT
DAYS_ID_PUBLISH
REGION_POPULATION_RELATIVE
DAYS_BIRTH
1
0.95
0.0073
-0.53
-0.015
0.95
1
0.0023
-0.53
-0.017
0.0073
0.0023
1
0.0043
0.27
-0.53
-0.53
0.0043
1
0.033
-0.015
-0.017
0.27
0.033
1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4



[35]:
```
pre_app_data.isnull().sum()
```

[35]:
```
SK_ID_PREV                        0
SK_ID_CURR                        0
NAME_CONTRACT_TYPE                0
AMT_ANNUITY                   10592
AMT_APPLICATION                   0
AMT_CREDIT                        0
AMT_DOWN_PAYMENT              25198
AMT_GOODS_PRICE               10744
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             25198
RATE_INTEREST_PRIMARY         49834
RATE_INTEREST_PRIVILEGED      49834
NAME_CASH_LOAN_PURPOSE            0
NAME_CONTRACT_STATUS              0
DAYS_DECISION                     0
```

```
NAME_PAYMENT_TYPE                 0
CODE_REJECT_REASON                0
NAME_TYPE_SUITE               24243
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                   10592
NAME_YIELD_GROUP                  0
PRODUCT_COMBINATION               8
DAYS_FIRST_DRAWING            19160
DAYS_FIRST_DUE                19160
DAYS_LAST_DUE_1ST_VERSION     19160
DAYS_LAST_DUE                 19160
DAYS_TERMINATION              19160
NFLAG_INSURED_ON_APPROVAL     19160
dtype: int64
```

[36]:
```
pre_app_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 49999 entries, 0 to 49998
Data columns (total 37 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   SK_ID_PREV                   49999 non-null  int64
 1   SK_ID_CURR                   49999 non-null  int64
 2   NAME_CONTRACT_TYPE           49999 non-null  object
 3   AMT_ANNUITY                  39407 non-null  float64
 4   AMT_APPLICATION              49999 non-null  float64
 5   AMT_CREDIT                   49999 non-null  float64
 6   AMT_DOWN_PAYMENT             24801 non-null  float64
 7   AMT_GOODS_PRICE              39255 non-null  float64
 8   WEEKDAY_APPR_PROCESS_START   49999 non-null  object
 9   HOUR_APPR_PROCESS_START      49999 non-null  int64
 10  FLAG_LAST_APPL_PER_CONTRACT  49999 non-null  object
 11  NFLAG_LAST_APPL_IN_DAY       49999 non-null  int64
 12  RATE_DOWN_PAYMENT            24801 non-null  float64
 13  RATE_INTEREST_PRIMARY        165 non-null    float64
 14  RATE_INTEREST_PRIVILEGED     165 non-null    float64
 15  NAME_CASH_LOAN_PURPOSE       49999 non-null  object
 16  NAME_CONTRACT_STATUS         49999 non-null  object
 17  DAYS_DECISION                49999 non-null  int64
 18  NAME_PAYMENT_TYPE            49999 non-null  object
```

```
 19  CODE_REJECT_REASON           49999 non-null  object
 20  NAME_TYPE_SUITE              25756 non-null  object
 21  NAME_CLIENT_TYPE             49999 non-null  object
 22  NAME_GOODS_CATEGORY          49999 non-null  object
 23  NAME_PORTFOLIO               49999 non-null  object
 24  NAME_PRODUCT_TYPE            49999 non-null  object
 25  CHANNEL_TYPE                 49999 non-null  object
 26  SELLERPLACE_AREA             49999 non-null  int64
 27  NAME_SELLER_INDUSTRY         49999 non-null  object
 28  CNT_PAYMENT                  39407 non-null  float64
 29  NAME_YIELD_GROUP             49999 non-null  object
 30  PRODUCT_COMBINATION          49991 non-null  object
 31  DAYS_FIRST_DRAWING           30839 non-null  float64
 32  DAYS_FIRST_DUE               30839 non-null  float64
 33  DAYS_LAST_DUE_1ST_VERSION    30839 non-null  float64
 34  DAYS_LAST_DUE                30839 non-null  float64
 35  DAYS_TERMINATION             30839 non-null  float64
 36  NFLAG_INSURED_ON_APPROVAL    30839 non-null  float64
dtypes: float64(15), int64(6), object(16)
memory usage: 14.1+ MB
```

[37]:
```
numerical_cols = pre_app_data.select_dtypes(include=['int64', 'float64'])
```

[38]:
```
numerical_cols
```

[38]:

	SK_ID_PREV	SK_ID_CURR	AMT_ANNUITY	AMT_APPLICATION	AMT_CREDIT	AMT_DOWN_PAYMENT	AMT_GOODS_PRICE	HOUR_APPR_PROCESS_START
0	2030495	271877	1730.430	17145.0	17145.0	0.0	17145.0	15
1	2802425	108129	25188.615	607500.0	679671.0	NaN	607500.0	11
2	2523466	122040	15060.735	112500.0	136444.5	NaN	112500.0	11
3	2819243	176158	47041.335	450000.0	470790.0	NaN	450000.0	7
4	1784265	202054	31924.395	337500.0	404055.0	NaN	337500.0	9
…	…	…	…	…	…	…	…	…
49994	1171956	339569	NaN	0.0	0.0	NaN	NaN	11
49995	1904808	363980	NaN	0.0	0.0	NaN	NaN	10
49996	2331005	231295	22176.405	180000.0	216418.5			
49997	1960897	346691	NaN	0.0	0.0			
49998	1979352	363244	24909.390	360000.0	409896.0			

```
49996               NaN         180000.0                        12
49997               NaN              NaN                        16
49998               NaN         360000.0                        18

       NFLAG_LAST_APPL_IN_DAY  RATE_DOWN_PAYMENT  …  \
0                           1                0.0  …
1                           1                NaN  …
2                           1                NaN  …
3                           1                NaN  …
4                           1                NaN  …
…                         …                  …  …
49994                       1                NaN  …
49995                       1                NaN  …
49996                       1                NaN  …
49997                       1                NaN  …
49998                       1                NaN  …

       RATE_INTEREST_PRIVILEGED  DAYS_DECISION  SELLERPLACE_AREA  CNT_PAYMENT  \
0                      0.867336            -73                35         12.0
1                           NaN           -164                -1         36.0
2                           NaN           -301                -1         12.0
3                           NaN           -512                -1         12.0
4                           NaN           -781                -1         24.0
…                         …               …                 …           …
49994                       NaN            -24                -1          NaN
49995                       NaN           -425                -1          NaN
49996                       NaN           -700                -1         12.0
49997                       NaN            -13                -1          NaN
49998                       NaN           -441                -1         36.0

       DAYS_FIRST_DRAWING  DAYS_FIRST_DUE  DAYS_LAST_DUE_1ST_VERSION  \
0                365243.0           -42.0                      300.0
1                365243.0          -134.0                      916.0
2                365243.0          -271.0                       59.0
3                365243.0          -482.0                     -152.0
4                     NaN             NaN                        NaN
…                   …               …                          …
49994                 NaN             NaN                        NaN
49995                 NaN             NaN                        NaN
49996            365243.0          -670.0                     -340.0
49997                 NaN             NaN                        NaN
49998                 NaN             NaN                        NaN

       DAYS_LAST_DUE  DAYS_TERMINATION  NFLAG_INSURED_ON_APPROVAL
0              -42.0             -37.0                        0.0
1           365243.0          365243.0                        1.0
2           365243.0          365243.0                        1.0
```

```
3             -182.0            -177.0                        1.0
4                NaN               NaN                        NaN
…                …                 …                          …
49994            NaN               NaN                        NaN
49995            NaN               NaN                        NaN
49996         -340.0            -338.0                        1.0
49997            NaN               NaN                        NaN
49998            NaN               NaN                        NaN

[49999 rows x 21 columns]
```

[39]:
```
summary_stats = numerical_cols.describe()
print(summary_stats)
```

```
         SK_ID_PREV     SK_ID_CURR    AMT_ANNUITY  AMT_APPLICATION  \
count  4.999900e+04   49999.000000   39407.000000     4.999900e+04
mean   1.922254e+06  278983.187604   15482.596847     1.688925e+05
std    5.351980e+05  102780.124434   14530.971854     2.822035e+05
min    1.000001e+06  100007.000000       0.000000     0.000000e+00
25%    1.457920e+06  189919.500000    6122.835000     2.204550e+04
50%    1.920889e+06  279264.000000   10879.920000     7.155000e+04
75%    2.388632e+06  368527.500000   19669.140000     1.800000e+05
max    2.845367e+06  456254.000000  234478.395000     3.826372e+06

         AMT_CREDIT  AMT_DOWN_PAYMENT  AMT_GOODS_PRICE  \
count  4.999900e+04      2.480100e+04     3.925500e+04
mean   1.885429e+05      6.557571e+03     2.151414e+05
std    3.084736e+05      1.744458e+04     3.024993e+05
min    0.000000e+00      0.000000e+00     0.000000e+00
25%    2.605500e+04      0.000000e+00     4.941000e+04
50%    7.890750e+04      1.566000e+03     1.040175e+05
75%    1.981058e+05      7.875000e+03     2.250000e+05
max    4.104351e+06      1.035000e+06     3.826372e+06

       HOUR_APPR_PROCESS_START  NFLAG_LAST_APPL_IN_DAY  RATE_DOWN_PAYMENT  \
count              49999.000000            49999.000000       24801.000000
mean                  12.478330                0.996500           0.079083
std                    3.333012                0.059058           0.107658
min                    0.000000                0.000000           0.000000
25%                   10.000000                1.000000           0.000000
50%                   12.000000                1.000000           0.049732
75%                   15.000000                1.000000           0.108909
max                   23.000000                1.000000           0.944776

       …  RATE_INTEREST_PRIVILEGED  DAYS_DECISION  SELLERPLACE_AREA  \
count  …                165.000000   49999.000000      4.999900e+04
mean   …                  0.787674    -900.112622      4.016558e+02
```

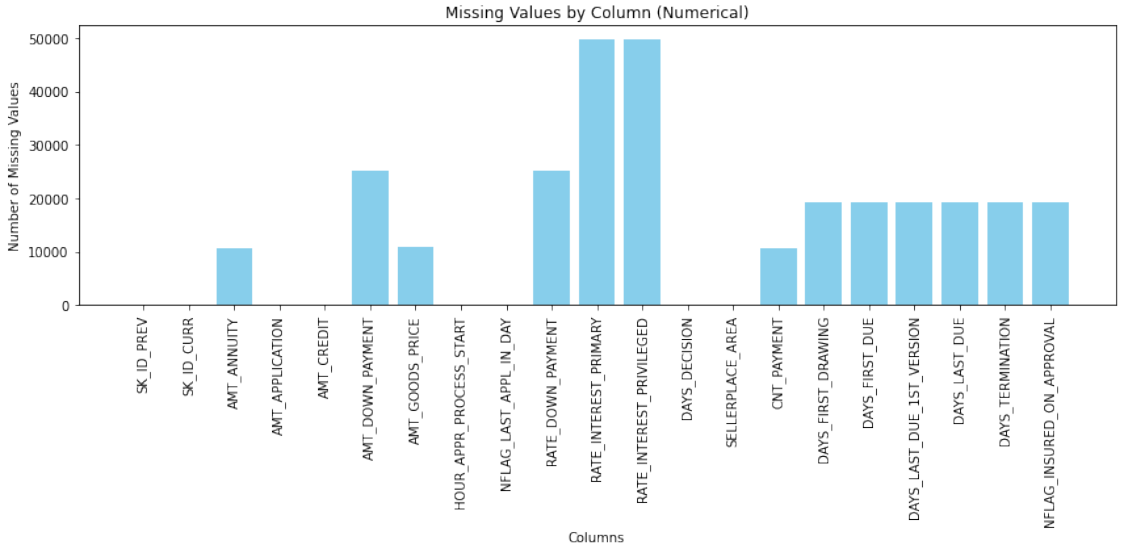
```
std    …                  0.091985     786.531303      1.793772e+04
min    …                  0.424419   -2922.000000     -1.000000e+00
25%    …                  0.715645   -1335.000000     -1.000000e+00
50%    …                  0.835095    -599.000000      1.000000e+01
75%    …                  0.852537    -292.000000      1.000000e+02
max    …                  0.867336      -2.000000      4.000000e+06

        CNT_PAYMENT  DAYS_FIRST_DRAWING  DAYS_FIRST_DUE  \
count  39407.000000        30839.000000    30839.000000
mean      15.555891       344485.142806    14217.240150
std       13.985174        84683.650627    73348.984383
min        0.000000        -2910.000000    -2891.000000
25%        6.000000       365243.000000    -1642.000000
50%       12.000000       365243.000000     -822.000000
75%       18.000000       365243.000000     -404.000000
max       60.000000       365243.000000   365243.000000

       DAYS_LAST_DUE_1ST_VERSION  DAYS_LAST_DUE  DAYS_TERMINATION  \
count                30839.000000   30839.000000      30839.000000
mean                 31528.148611   76724.982101      81666.162586
std                 103691.881189  149757.893750     153101.159809
min                  -2800.000000   -2850.000000      -2844.000000
25%                  -1270.000000   -1337.000000      -1293.000000
50%                   -366.000000    -536.000000       -500.000000
75%                    113.000000     -71.000000        -45.000000
max                 365243.000000  365243.000000     365243.000000

       NFLAG_INSURED_ON_APPROVAL
count                30839.000000
mean                     0.322352
std                      0.467384
min                      0.000000
25%                      0.000000
50%                      0.000000
75%                      1.000000
max                      1.000000

[8 rows x 21 columns]
```

[40]:
```
missing_values = numerical_cols.isnull().sum()
print("Missing Values:")
print(missing_values)
```

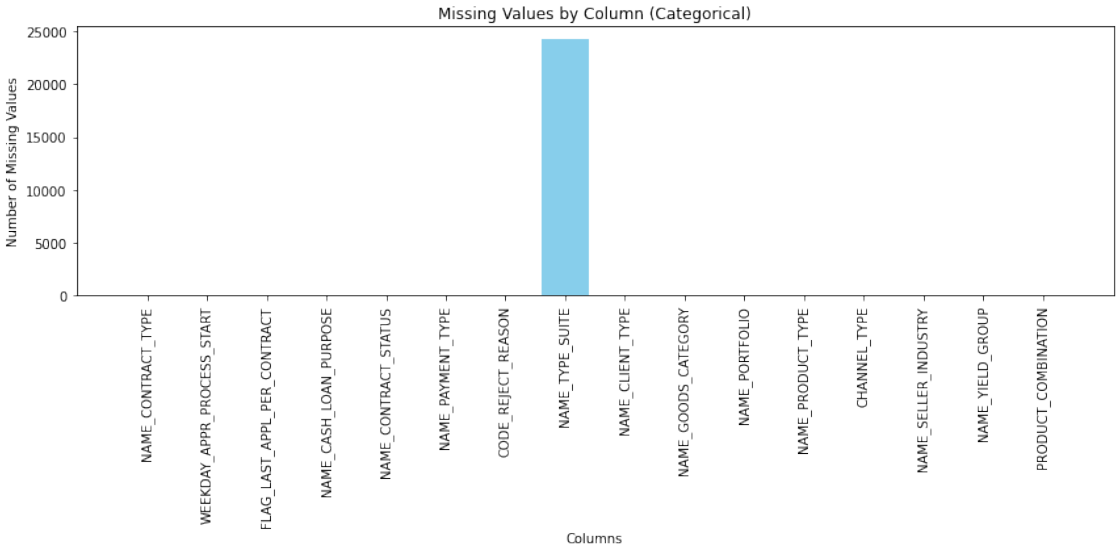
```
Missing Values:
SK_ID_PREV                   0
SK_ID_CURR                   0
AMT_ANNUITY                10592
AMT_APPLICATION              0
```

```
AMT_CREDIT                       0
AMT_DOWN_PAYMENT             25198
AMT_GOODS_PRICE              10744
HOUR_APPR_PROCESS_START          0
NFLAG_LAST_APPL_IN_DAY           0
RATE_DOWN_PAYMENT            25198
RATE_INTEREST_PRIMARY        49834
RATE_INTEREST_PRIVILEGED     49834
DAYS_DECISION                    0
SELLERPLACE_AREA                 0
CNT_PAYMENT                  10592
DAYS_FIRST_DRAWING           19160
DAYS_FIRST_DUE               19160
DAYS_LAST_DUE_1ST_VERSION    19160
DAYS_LAST_DUE                19160
DAYS_TERMINATION             19160
NFLAG_INSURED_ON_APPROVAL    19160
dtype: int64
```

[41]:
```
missing_values = numerical_cols.isnull().sum()

# Create a bar chart to visualize missing values
plt.figure(figsize=(12, 6))
plt.bar(missing_values.index, missing_values.values, color='skyblue')
plt.xlabel('Columns')
plt.ylabel('Number of Missing Values')
plt.title('Missing Values by Column (Numerical)')
plt.xticks(rotation=90)  # Rotate x-axis labels for better readability
plt.tight_layout()
plt.show()
```



[42]:
```
categorical_cols = pre_app_data.select_dtypes(include='object')

# Calculate the number of missing values for each categorical column
missing_values = categorical_cols.isnull().sum()

# Create a bar chart to visualize missing values
plt.figure(figsize=(12, 6))
plt.bar(missing_values.index, missing_values.values, color='skyblue')
plt.xlabel('Columns')
plt.ylabel('Number of Missing Values')
plt.title('Missing Values by Column (Categorical)')
plt.xticks(rotation=90)  # Rotate x-axis labels for better readability
plt.tight_layout()
plt.show()
```



[43]:
```
null_counts2 = pre_app_data.isnull().sum()/49999*100
```

[44]:
```
null_counts2
```

[44]:
```
SK_ID_PREV                    0.000000
SK_ID_CURR                    0.000000
NAME_CONTRACT_TYPE            0.000000
AMT_ANNUITY                  21.184424
AMT_APPLICATION               0.000000
AMT_CREDIT                    0.000000
AMT_DOWN_PAYMENT             50.397008
AMT_GOODS_PRICE              21.488430
```

```
WEEKDAY_APPR_PROCESS_START      0.000000
HOUR_APPR_PROCESS_START         0.000000
FLAG_LAST_APPL_PER_CONTRACT     0.000000
NFLAG_LAST_APPL_IN_DAY          0.000000
RATE_DOWN_PAYMENT              50.397008
RATE_INTEREST_PRIMARY          99.669993
RATE_INTEREST_PRIVILEGED       99.669993
NAME_CASH_LOAN_PURPOSE          0.000000
NAME_CONTRACT_STATUS            0.000000
DAYS_DECISION                   0.000000
NAME_PAYMENT_TYPE               0.000000
CODE_REJECT_REASON              0.000000
NAME_TYPE_SUITE                48.486970
NAME_CLIENT_TYPE                0.000000
NAME_GOODS_CATEGORY             0.000000
NAME_PORTFOLIO                  0.000000
NAME_PRODUCT_TYPE               0.000000
CHANNEL_TYPE                    0.000000
SELLERPLACE_AREA                0.000000
NAME_SELLER_INDUSTRY            0.000000
CNT_PAYMENT                    21.184424
NAME_YIELD_GROUP                0.000000
PRODUCT_COMBINATION             0.016000
DAYS_FIRST_DRAWING             38.320766
DAYS_FIRST_DUE                 38.320766
DAYS_LAST_DUE_1ST_VERSION      38.320766
DAYS_LAST_DUE                  38.320766
DAYS_TERMINATION               38.320766
NFLAG_INSURED_ON_APPROVAL      38.320766
dtype: float64
```

[45]:
```python
threshold = 0.3 * len(pre_app_data)

# Filter columns with missing values exceeding the threshold
columns_to_drop = pre_app_data.columns[pre_app_data.isnull().sum() > threshold]

# Drop the selected columns
pre_app_data_new = pre_app_data.drop(columns=columns_to_drop)

# Verify the changes
print(pre_app_data_new.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_GOODS_PRICE WEEKDAY_APPR_PROCESS_START  \
0     17145.0          17145.0                   SATURDAY
1    679671.0         607500.0                   THURSDAY
2    136444.5         112500.0                    TUESDAY
3    470790.0         450000.0                     MONDAY
4    404055.0         337500.0                   THURSDAY

   HOUR_APPR_PROCESS_START FLAG_LAST_APPL_PER_CONTRACT  …  NAME_CLIENT_TYPE  \
0                       15                           Y  …          Repeater
1                       11                           Y  …          Repeater
2                       11                           Y  …          Repeater
3                        7                           Y  …          Repeater
4                        9                           Y  …          Repeater

  NAME_GOODS_CATEGORY NAME_PORTFOLIO  NAME_PRODUCT_TYPE  \
0              Mobile            POS                XNA
1                 XNA           Cash             x-sell
2                 XNA           Cash             x-sell
3                 XNA           Cash             x-sell
4                 XNA           Cash            walk-in

             CHANNEL_TYPE SELLERPLACE_AREA NAME_SELLER_INDUSTRY CNT_PAYMENT  \
0             Country-wide              35         Connectivity        12.0
1          Contact center              -1                  XNA        36.0
2  Credit and cash offices              -1                  XNA        12.0
3  Credit and cash offices              -1                  XNA        12.0
4  Credit and cash offices              -1                  XNA        24.0

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

[5 rows x 26 columns]
```

[46]:
```
pre_app_data_new.shape
```

[46]:
```
(49999, 26)
```

[47]:
```
drop_columns2 =␣
 ↪['WEEKDAY_APPR_PROCESS_START','HOUR_APPR_PROCESS_START','FLAG_LAST_APPL_PER_CONTRACT']
```

[48]:
```
pre_app_data_new = pre_app_data_new.drop(columns = drop_columns2)
```

[49]:
```
pre_app_data_new.shape
```

[49]:
```
(49999, 23)
```

[50]:
```
pre_app_data_new.head()
```

[50]:
```
   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_GOODS_PRICE  NFLAG_LAST_APPL_IN_DAY NAME_CASH_LOAN_PURPOSE  \
0     17145.0          17145.0                       1                     XAP
1    679671.0         607500.0                       1                     XNA
2    136444.5         112500.0                       1                     XNA
3    470790.0         450000.0                       1                     XNA
4    404055.0         337500.0                       1                 Repairs

  NAME_CONTRACT_STATUS  …  NAME_CLIENT_TYPE NAME_GOODS_CATEGORY  \
0             Approved  …          Repeater              Mobile
1             Approved  …          Repeater                 XNA
2             Approved  …          Repeater                 XNA
3             Approved  …          Repeater                 XNA
4              Refused  …          Repeater                 XNA

  NAME_PORTFOLIO NAME_PRODUCT_TYPE            CHANNEL_TYPE SELLERPLACE_AREA  \
0            POS               XNA            Country-wide               35
1           Cash            x-sell          Contact center               -1
2           Cash            x-sell  Credit and cash offices               -1
3           Cash            x-sell  Credit and cash offices               -1
4           Cash           walk-in  Credit and cash offices               -1

  NAME_SELLER_INDUSTRY CNT_PAYMENT  NAME_YIELD_GROUP       PRODUCT_COMBINATION
0         Connectivity        12.0            middle  POS mobile with interest
1                  XNA        36.0       low_action          Cash X-Sell: low
2                  XNA        12.0              high         Cash X-Sell: high
3                  XNA        12.0            middle       Cash X-Sell: middle
4                  XNA        24.0              high         Cash Street: high

[5 rows x 23 columns]
```

[51]:
```
columns_converts = ['DAYS_DECISION']

pre_app_data_new[columns_converts] = pre_app_data_new[columns_converts].abs()
```

[52]:
```
pre_app_data_new['DAYS_DECISION'].describe()
```

[52]:
```
count    49999.000000
mean       900.112622
std        786.531303
min          2.000000
25%        292.000000
50%        599.000000
75%       1335.000000
max       2922.000000
Name: DAYS_DECISION, dtype: float64
```

[53]:
```
pre_app_data_new.isnull().sum().sort_values(ascending =True)
```

[53]:
```
SK_ID_PREV                   0
NAME_SELLER_INDUSTRY         0
SELLERPLACE_AREA             0
CHANNEL_TYPE                 0
NAME_PRODUCT_TYPE            0
NAME_PORTFOLIO               0
NAME_GOODS_CATEGORY          0
NAME_CLIENT_TYPE             0
CODE_REJECT_REASON           0
NAME_YIELD_GROUP             0
NAME_PAYMENT_TYPE            0
NAME_CONTRACT_STATUS         0
NAME_CASH_LOAN_PURPOSE       0
NFLAG_LAST_APPL_IN_DAY       0
AMT_CREDIT                   0
AMT_APPLICATION              0
NAME_CONTRACT_TYPE           0
SK_ID_CURR                   0
DAYS_DECISION                0
PRODUCT_COMBINATION          8
AMT_ANNUITY              10592
CNT_PAYMENT              10592
AMT_GOODS_PRICE          10744
dtype: int64
```

[54]:
```
numeric_columns = ['AMT_ANNUITY', 'CNT_PAYMENT', 'AMT_GOODS_PRICE']
for column in numeric_columns:
    pre_app_data_new[column].fillna(pre_app_data_new[column].mean(),␣
 ↪inplace=True)

# Fill missing values in categorical columns with the mode
categorical_columns = ['PRODUCT_COMBINATION']
for column in categorical_columns:
```

```
    pre_app_data_new[column].fillna(pre_app_data_new[column].mode()[0],␣
↪inplace=True)
```

[55]:
```
pre_app_data_new.isnull().sum()
```

[55]:
```
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
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
```

0.2.3 merge data

[56]:
```
merged_data = app_data.merge(pre_app_data, on='SK_ID_CURR', how='inner')
```

[57]:
```
merged_data.head()
```

[57]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE_x	CODE_GENDER	FLAG_OWN_CAR	\
0	100007	0	Cash loans	M	N	
1	100009	0	Cash loans	F	Y	
2	100012	0	Revolving loans	M	N	
3	100026	0	Cash loans	F	N	
4	100027	0	Cash loans	F	N	

	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT_x	\
0	Y	0	121500.0	513000.0	
1	Y	1	171000.0	1560726.0	
2	Y	0	135000.0	405000.0	

```
3               N             1          450000.0      497520.0
4               Y             0           83250.0      239850.0

       NAME_INCOME_TYPE  …  NAME_SELLER_INDUSTRY CNT_PAYMENT  \
0               Working  …  Consumer electronics        18.0
1  Commercial associate  …  Consumer electronics        12.0
2               Working  …          Connectivity        12.0
3               Working  …  Consumer electronics         6.0
4             Pensioner  …                   XNA        12.0

  NAME_YIELD_GROUP             PRODUCT_COMBINATION  DAYS_FIRST_DRAWING  \
0             high               Cash Street: high            365243.0
1       low_action  POS household without interest            365243.0
2             high        POS mobile with interest            365243.0
3       low_normal  POS household without interest            365243.0
4       low_normal                Cash X-Sell: low                 NaN

   DAYS_FIRST_DUE  DAYS_LAST_DUE_1ST_VERSION  DAYS_LAST_DUE  DAYS_TERMINATION  \
0          -834.0                     -324.0         -354.0            -347.0
1          -418.0                      -88.0          -88.0             -84.0
2         -1641.0                    -1311.0        -1401.0           -1397.0
3         -1396.0                    -1246.0        -1246.0           -1243.0
4             NaN                        NaN            NaN               NaN

   NFLAG_INSURED_ON_APPROVAL
0                        0.0
1                        0.0
2                        0.0
3                        0.0
4                        NaN

[5 rows x 65 columns]
```

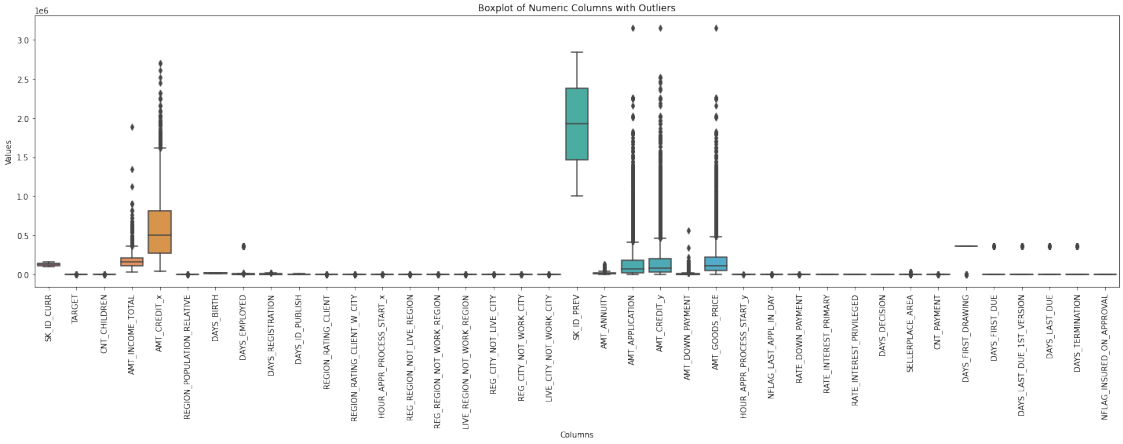
[58]:
```
merged_data.columns
```

[58]:
```
Index(['SK_ID_CURR', 'TARGET', 'NAME_CONTRACT_TYPE_x', 'CODE_GENDER',
       'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL',
       'AMT_CREDIT_x', 'NAME_INCOME_TYPE', 'NAME_EDUCATION_TYPE',
       'NAME_FAMILY_STATUS', 'NAME_HOUSING_TYPE', 'REGION_POPULATION_RELATIVE',
       'DAYS_BIRTH', 'DAYS_EMPLOYED', 'DAYS_REGISTRATION', 'DAYS_ID_PUBLISH',
       'REGION_RATING_CLIENT', 'REGION_RATING_CLIENT_W_CITY',
       'WEEKDAY_APPR_PROCESS_START_x', 'HOUR_APPR_PROCESS_START_x',
       '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', 'SK_ID_PREV', 'NAME_CONTRACT_TYPE_y',
       'AMT_ANNUITY', 'AMT_APPLICATION', 'AMT_CREDIT_y', 'AMT_DOWN_PAYMENT',
```

```
 'AMT_GOODS_PRICE', 'WEEKDAY_APPR_PROCESS_START_y',
 'HOUR_APPR_PROCESS_START_y', 'FLAG_LAST_APPL_PER_CONTRACT',
 'NFLAG_LAST_APPL_IN_DAY', 'RATE_DOWN_PAYMENT', 'RATE_INTEREST_PRIMARY',
 'RATE_INTEREST_PRIVILEGED', 'NAME_CASH_LOAN_PURPOSE',
 'NAME_CONTRACT_STATUS', 'DAYS_DECISION', 'NAME_PAYMENT_TYPE',
 'CODE_REJECT_REASON', 'NAME_TYPE_SUITE', 'NAME_CLIENT_TYPE',
 'NAME_GOODS_CATEGORY', 'NAME_PORTFOLIO', 'NAME_PRODUCT_TYPE',
 'CHANNEL_TYPE', 'SELLERPLACE_AREA', 'NAME_SELLER_INDUSTRY',
 'CNT_PAYMENT', 'NAME_YIELD_GROUP', 'PRODUCT_COMBINATION',
 'DAYS_FIRST_DRAWING', 'DAYS_FIRST_DUE', 'DAYS_LAST_DUE_1ST_VERSION',
 'DAYS_LAST_DUE', 'DAYS_TERMINATION', 'NFLAG_INSURED_ON_APPROVAL'],
dtype='object')
```

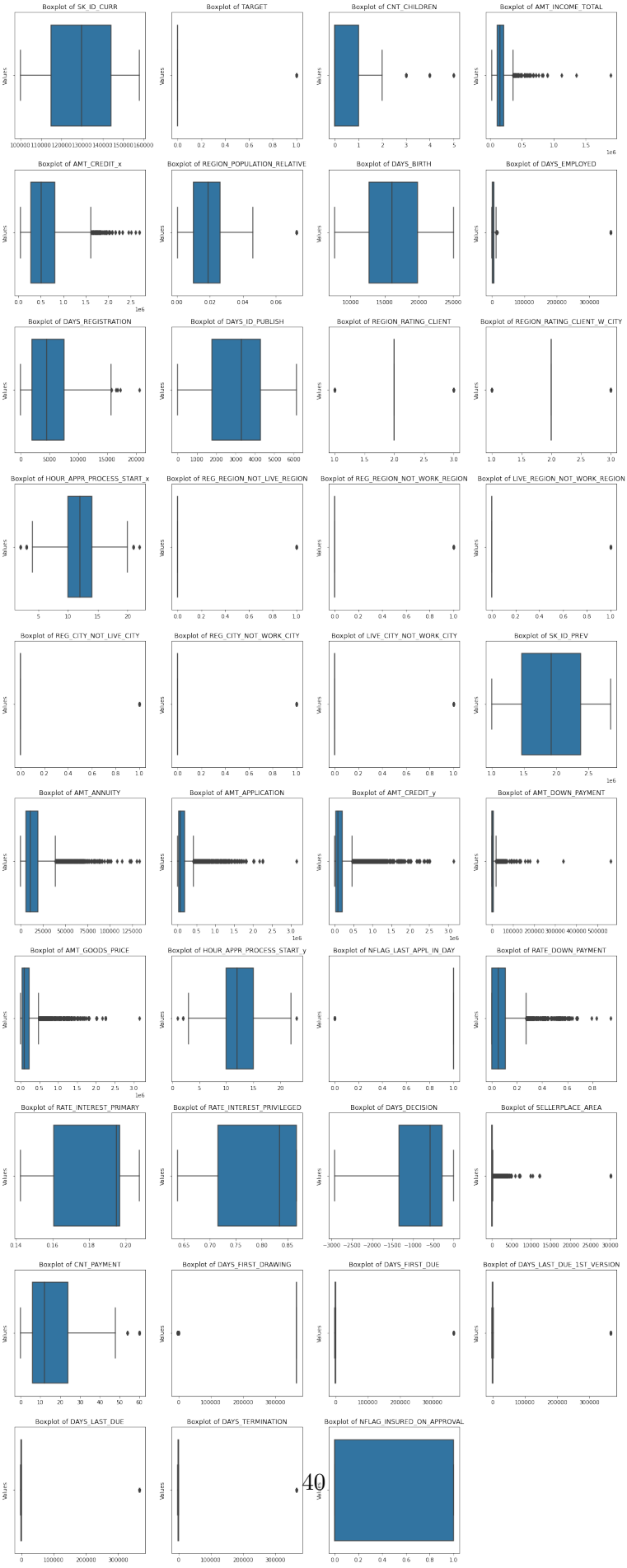
0.2.4 Outliers

[59]:
```
numeric_columns = merged_data.select_dtypes(include=['int64', 'float64'])

# Create a boxplot to visualize outliers
plt.figure(figsize=(20, 8))
sns.boxplot(data=numeric_columns)
plt.xticks(rotation=90)  # Rotate x-axis labels for better readability
plt.title('Boxplot of Numeric Columns with Outliers')
plt.xlabel('Columns')
plt.ylabel('Values')
plt.tight_layout()
plt.show()
```



[60]:
```
import math

numeric_columns = merged_data.select_dtypes(include=['int64', 'float64'])
```

```
# Calculate the number of rows and columns needed for the grid
num_cols = numeric_columns.shape[1]
num_rows = math.ceil(num_cols / 4)

# Create subplots to accommodate all columns
plt.figure(figsize=(16, num_rows * 4))  # Adjust the figsize as needed
for i, column in enumerate(numeric_columns.columns, 1):
    plt.subplot(num_rows, 4, i)  # Use num_rows and 4
    sns.boxplot(x=numeric_columns[column])
    plt.title(f'Boxplot of {column}')
    plt.xlabel('')
    plt.ylabel('Values')
plt.tight_layout()
plt.show()
```


Boxplot of SK_ID_CURR
Boxplot of TARGET
Boxplot of CNT_CHILDREN
Boxplot of AMT_INCOME_TOTAL
Boxplot of AMT_CREDIT_x
Boxplot of REGION_POPULATION_RELATIVE
Boxplot of DAYS_BIRTH
Boxplot of DAYS_EMPLOYED
Boxplot of DAYS_REGISTRATION
Boxplot of DAYS_ID_PUBLISH
Boxplot of REGION_RATING_CLIENT
Boxplot of REGION_RATING_CLIENT_W_CITY
Boxplot of HOUR_APPR_PROCESS_START_x
Boxplot of REG_REGION_NOT_LIVE_REGION
Boxplot of REG_REGION_NOT_WORK_REGION
Boxplot of LIVE_REGION_NOT_WORK_REGION
Boxplot of REG_CITY_NOT_LIVE_CITY
Boxplot of REG_CITY_NOT_WORK_CITY
Boxplot of LIVE_CITY_NOT_WORK_CITY
Boxplot of SK_ID_PREV
Boxplot of AMT_ANNUITY
Boxplot of AMT_APPLICATION
Boxplot of AMT_CREDIT_y
Boxplot of AMT_DOWN_PAYMENT
Boxplot of AMT_GOODS_PRICE
Boxplot of HOUR_APPR_PROCESS_START_y
Boxplot of NFLAG_LAST_APPL_IN_DAY
Boxplot of RATE_DOWN_PAYMENT
Boxplot of RATE_INTEREST_PRIMARY
Boxplot of RATE_INTEREST_PRIVILEGED
Boxplot of DAYS_DECISION
Boxplot of SELLERPLACE_AREA
Boxplot of CNT_PAYMENT
Boxplot of DAYS_FIRST_DRAWING
Boxplot of DAYS_FIRST_DUE
Boxplot of DAYS_LAST_DUE_1ST_VERSION
Boxplot of DAYS_LAST_DUE
Boxplot of DAYS_TERMINATION
Boxplot of NFLAG_INSURED_ON_APPROVAL
Values

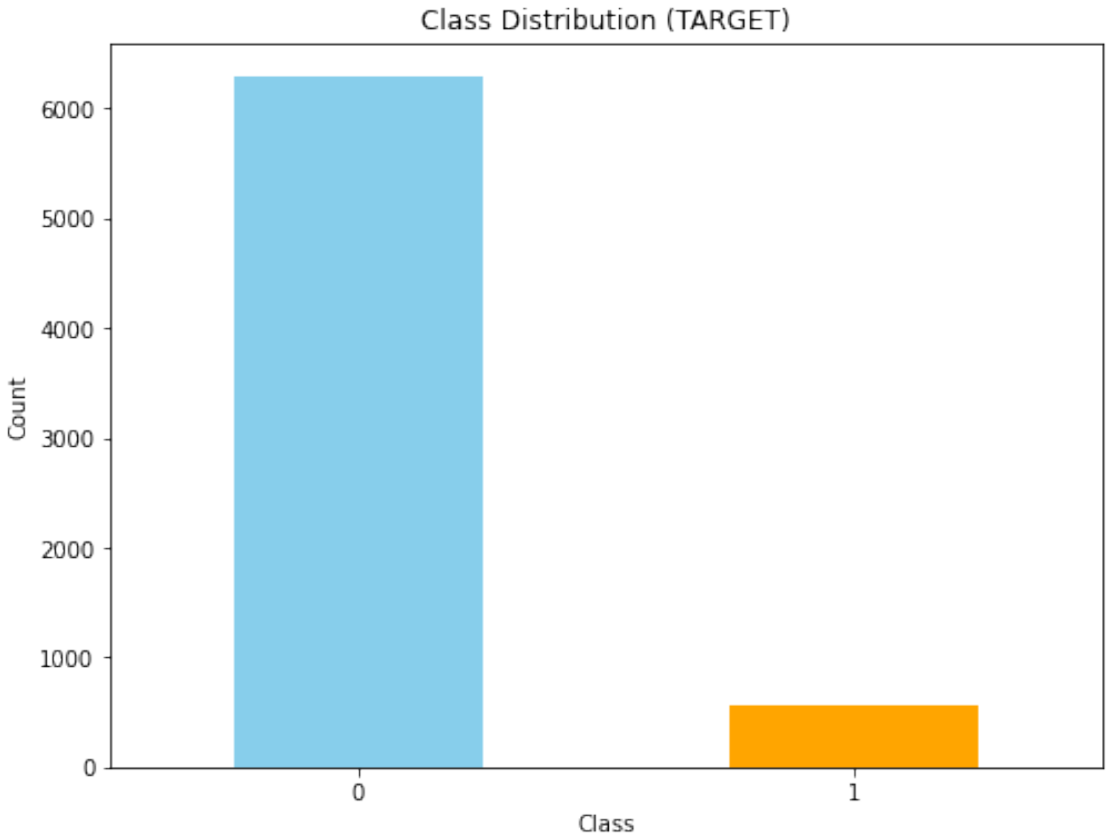
[61]:
```
file_path = 'C:/Users/ANINDYA DAS/OneDrive/Desktop/Tranity/Bank Loan Case Study/
 ↪merged_data.csv'

# Save the merged data to a CSV file
merged_data.to_csv(file_path, index=False)
```

[62]:
```
merged_data['TARGET'].value_counts()
```

[62]:
```
0    6287
1     554
Name: TARGET, dtype: int64
```

[63]:
```
class_distribution = merged_data['TARGET'].value_counts()

# Print the class distribution
print("Class Distribution:")
print(class_distribution)

# Step 2: Visualize class distribution
plt.figure(figsize=(8, 6))
merged_data['TARGET'].value_counts().plot(kind='bar', color=['skyblue',␣
 ↪'orange'])
plt.title('Class Distribution (TARGET)')
plt.xlabel('Class')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.show()

# Step 3: Calculate class imbalance ratio
majority_class_count = merged_data['TARGET'].value_counts().max()
minority_class_count = merged_data['TARGET'].value_counts().min()
imbalance_ratio = majority_class_count / minority_class_count

print(f'Imbalance Ratio: {imbalance_ratio:.2f}')
```

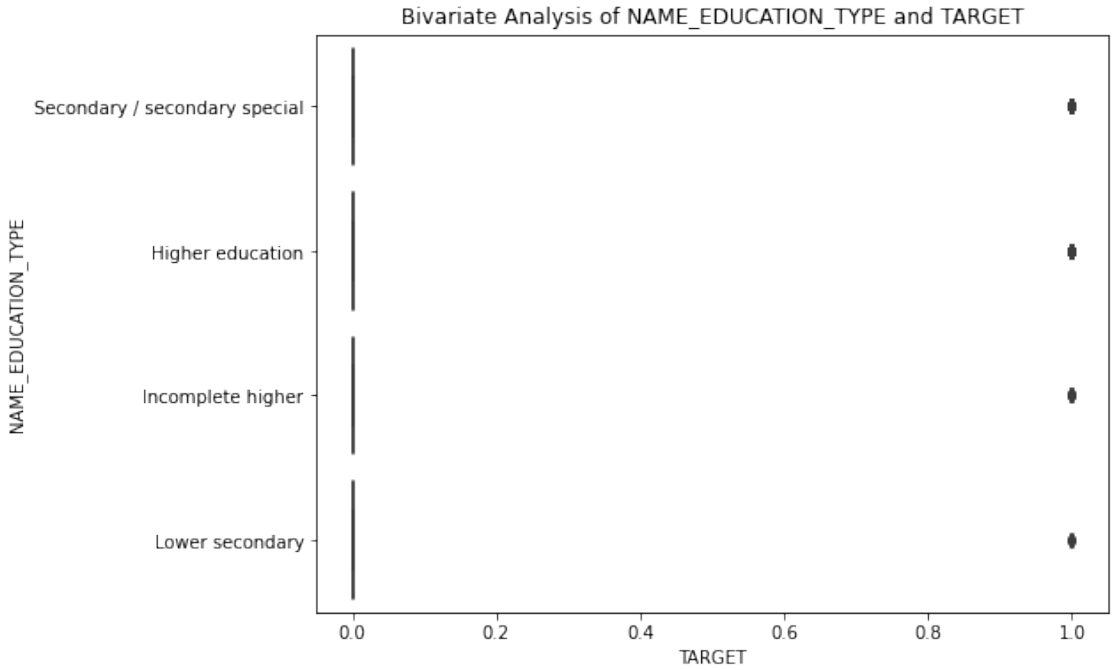
```
Class Distribution:
0    6287
1     554
Name: TARGET, dtype: int64
```

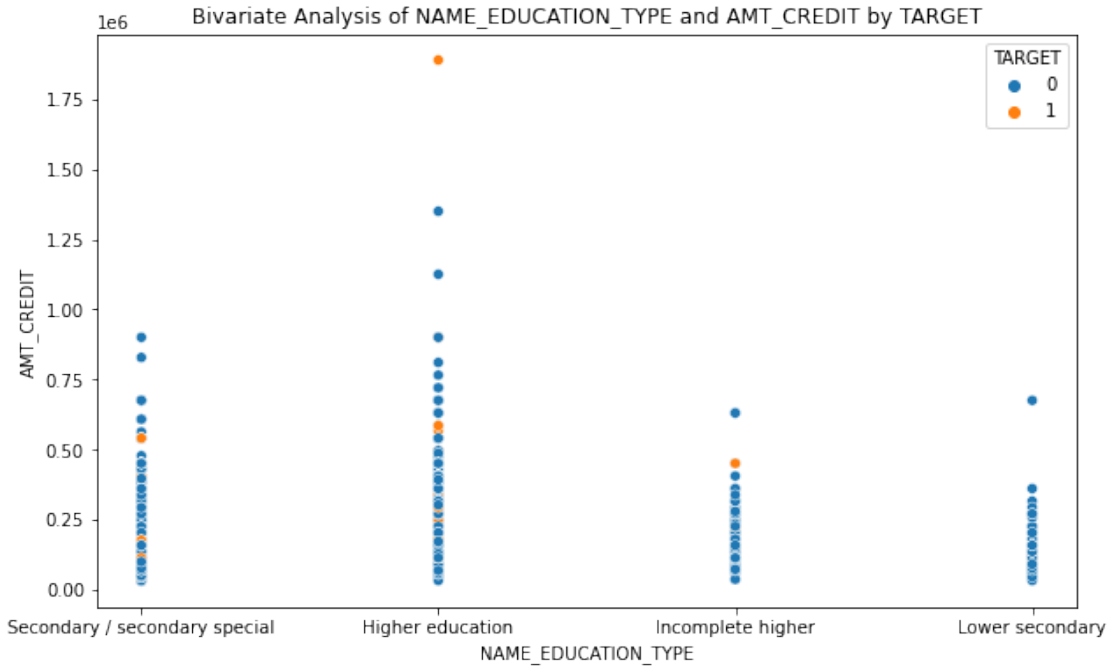



```
Imbalance Ratio: 11.35
```

[ ]:

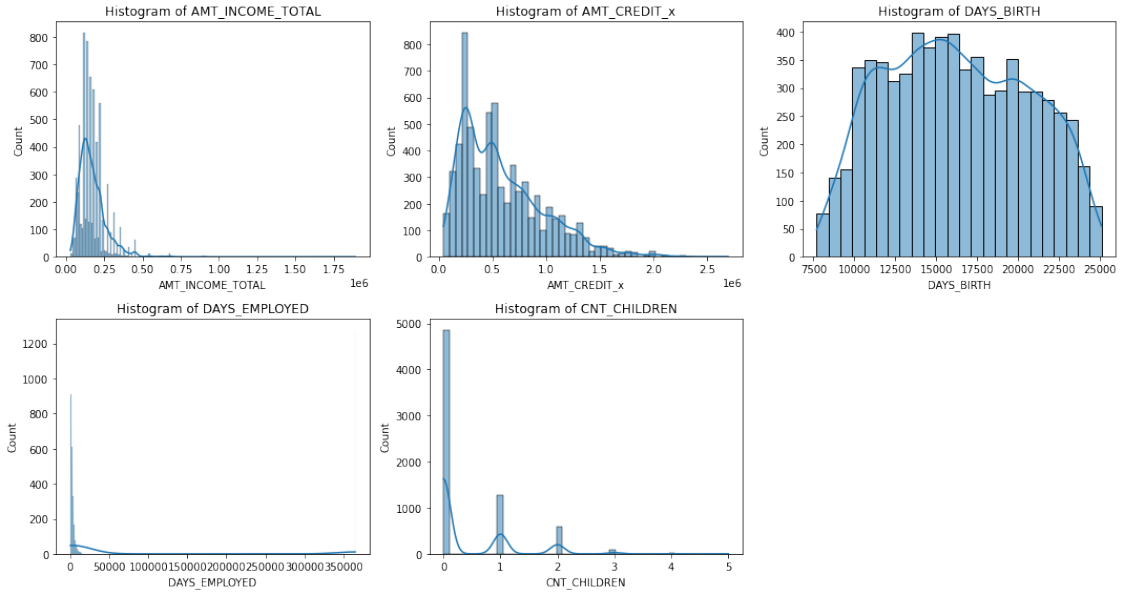
[64]:
```
plt.figure(figsize=(8, 6))
sns.boxplot(data=merged_data, x='TARGET', y='NAME_EDUCATION_TYPE')
plt.title('Bivariate Analysis of NAME_EDUCATION_TYPE and TARGET')
plt.xlabel('TARGET')
plt.ylabel('NAME_EDUCATION_TYPE')
plt.show()

# Bivariate analysis between two numeric variables (e.g., AMT_INCOME_TOTAL and␣
 ↪AMT_CREDIT)
plt.figure(figsize=(10, 6))
sns.scatterplot(data=merged_data, x='NAME_EDUCATION_TYPE',␣
 ↪y='AMT_INCOME_TOTAL', hue='TARGET')
plt.title('Bivariate Analysis of NAME_EDUCATION_TYPE and AMT_CREDIT by TARGET')
plt.xlabel('NAME_EDUCATION_TYPE')
plt.ylabel('AMT_CREDIT')
plt.show()
```

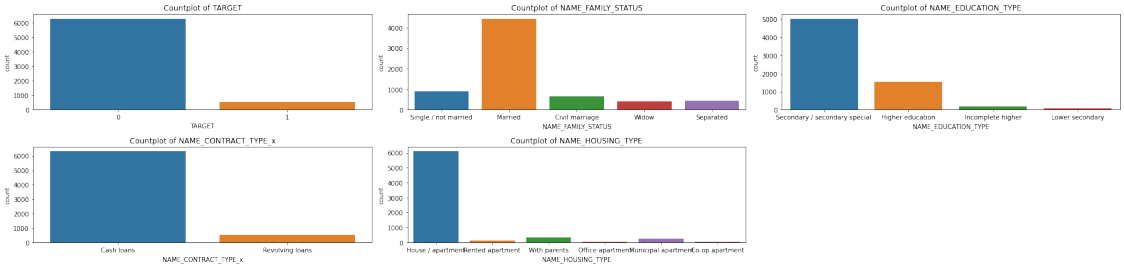




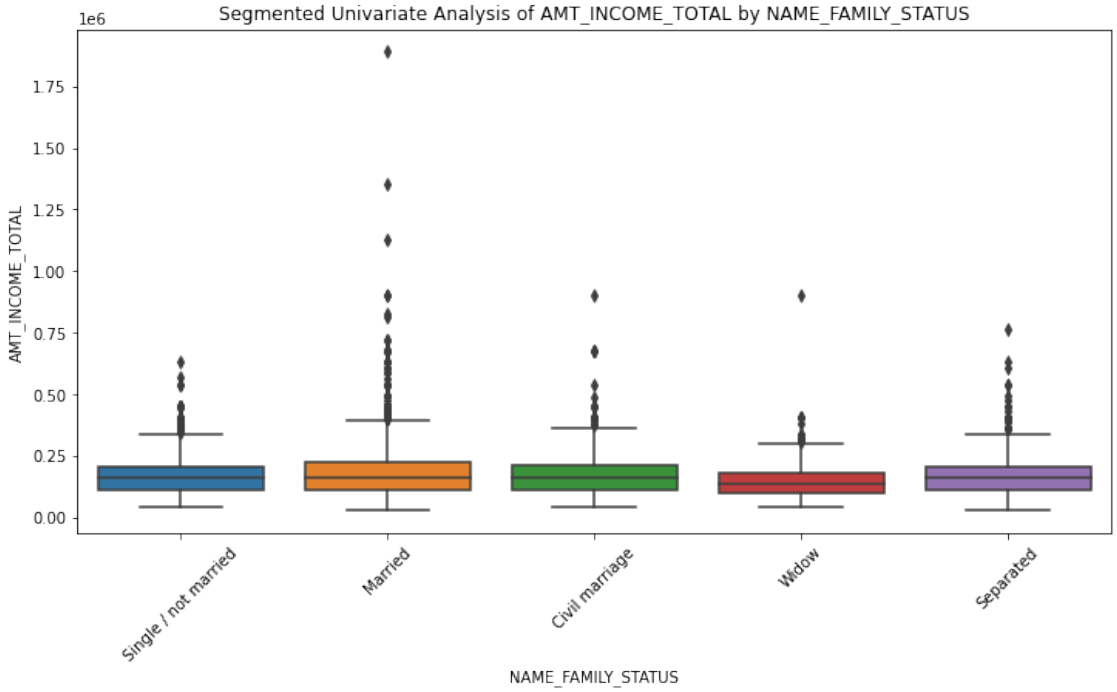
[65]:
```
selected_columns = [
    'TARGET', 'AMT_INCOME_TOTAL', 'AMT_CREDIT_x', 'DAYS_BIRTH',
    'DAYS_EMPLOYED', 'CNT_CHILDREN', 'NAME_FAMILY_STATUS',
```

```
    'NAME_EDUCATION_TYPE', 'NAME_CONTRACT_TYPE_x', 'NAME_HOUSING_TYPE'
]

# Univariate Analysis - Numeric Columns
numeric_columns = [
    'AMT_INCOME_TOTAL', 'AMT_CREDIT_x', 'DAYS_BIRTH',
    'DAYS_EMPLOYED', 'CNT_CHILDREN'
]

# Histograms for numeric columns
plt.figure(figsize=(15, 8))
for i, col in enumerate(numeric_columns, 1):
    plt.subplot(2, 3, i)
    sns.histplot(merged_data[col], kde=True)
    plt.title(f'Histogram of {col}')

plt.tight_layout()
plt.show()

# Univariate Analysis - Categorical Columns
categorical_columns = [
    'TARGET', 'NAME_FAMILY_STATUS',
    'NAME_EDUCATION_TYPE', 'NAME_CONTRACT_TYPE_x', 'NAME_HOUSING_TYPE'
]

# Countplots for categorical columns
plt.figure(figsize=(25, 6))
for i, col in enumerate(categorical_columns, 1):
    plt.subplot(2, 3, i)
    sns.countplot(data=merged_data, x=col)
    plt.title(f'Countplot of {col}')

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

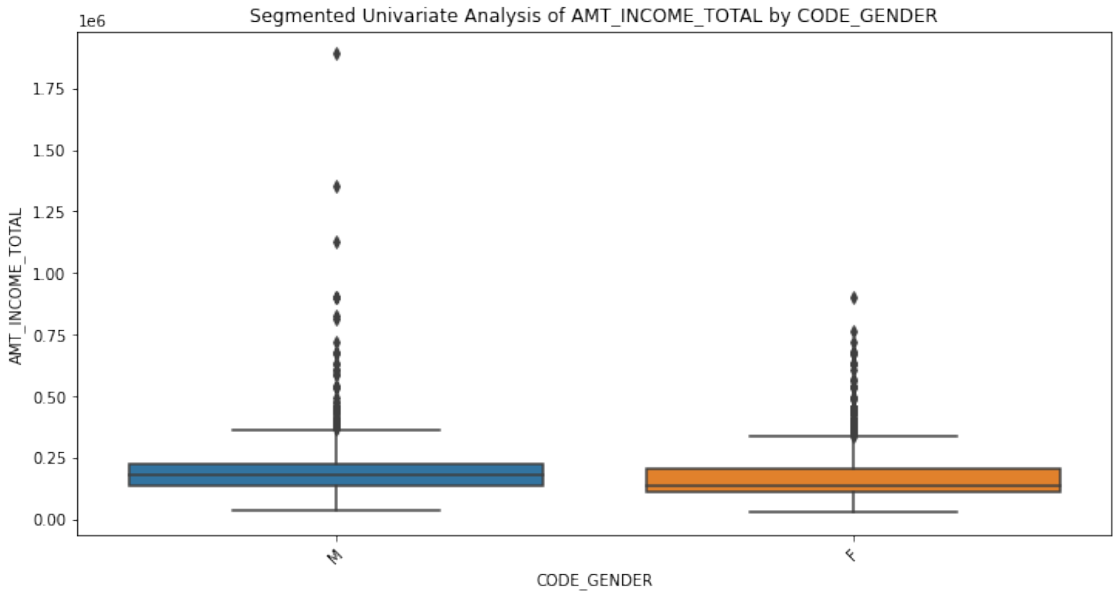




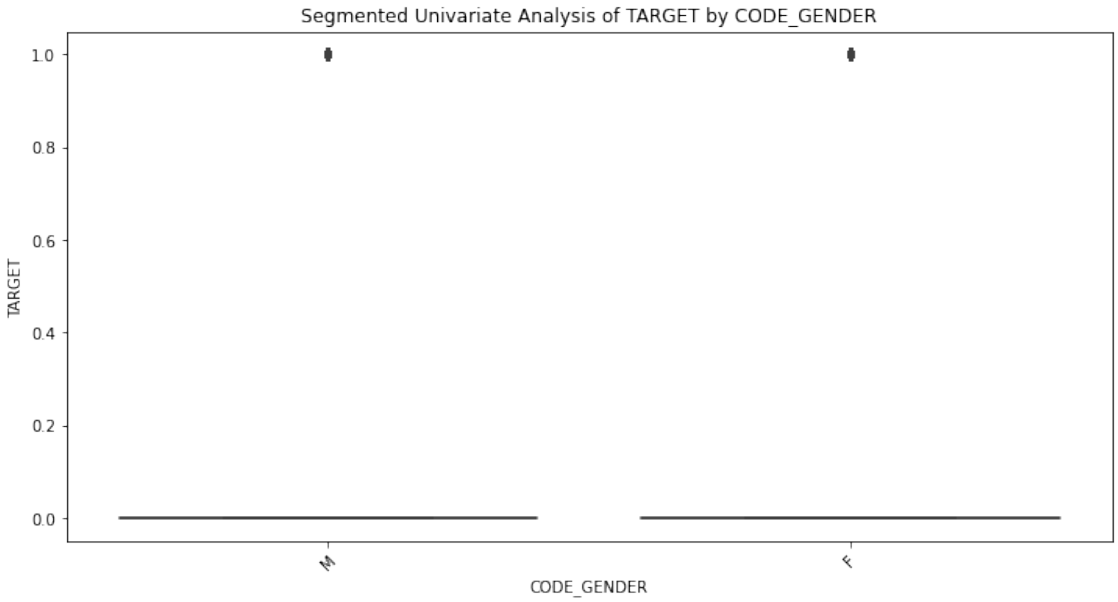
[66]:
```
segmenting_variable = 'NAME_FAMILY_STATUS'  # Variable for segmentation
target_variable = 'AMT_INCOME_TOTAL'  # Variable to analyze within segments

# Segmented Univariate Analysis - Numeric Variable within Segments
plt.figure(figsize=(12, 6))
sns.boxplot(data=merged_data, x=segmenting_variable, y=target_variable)
plt.xticks(rotation=45)
plt.title(f'Segmented Univariate Analysis of {target_variable} by␣
 ↪{segmenting_variable}')
plt.show()
```



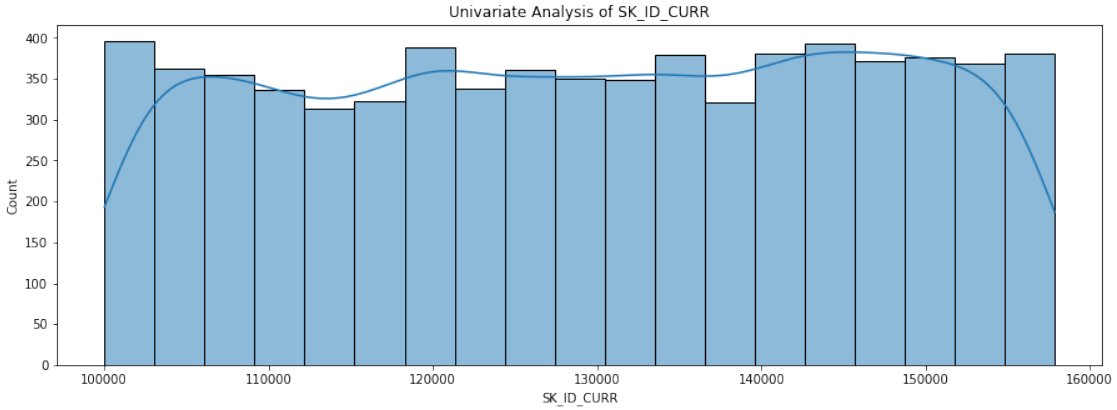
[67]:
```
segmenting_variable = 'CODE_GENDER'  # Variable for segmentation
target_variable = 'AMT_INCOME_TOTAL'  # Variable to analyze within segments

# Segmented Univariate Analysis - Numeric Variable within Segments
plt.figure(figsize=(12, 6))
sns.boxplot(data=merged_data, x=segmenting_variable, y=target_variable)
plt.xticks(rotation=45)
plt.title(f'Segmented Univariate Analysis of {target_variable} by␣
  ↪{segmenting_variable}')
plt.show()
```

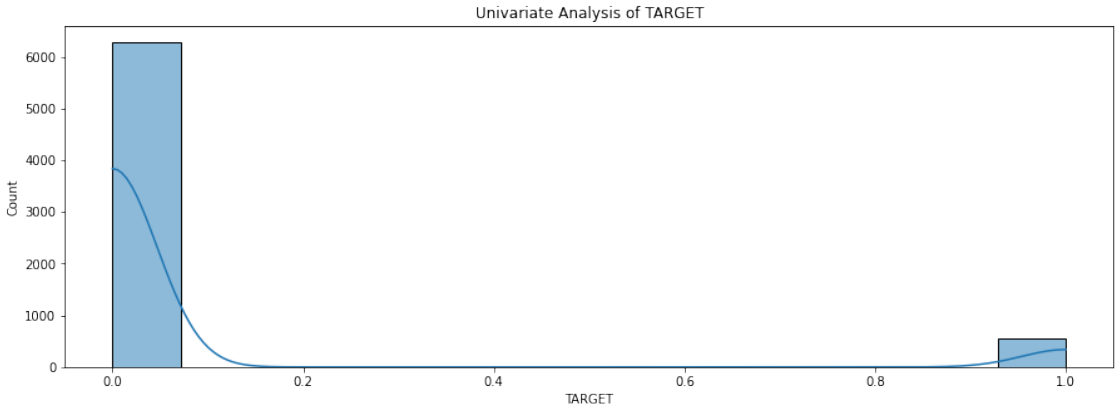


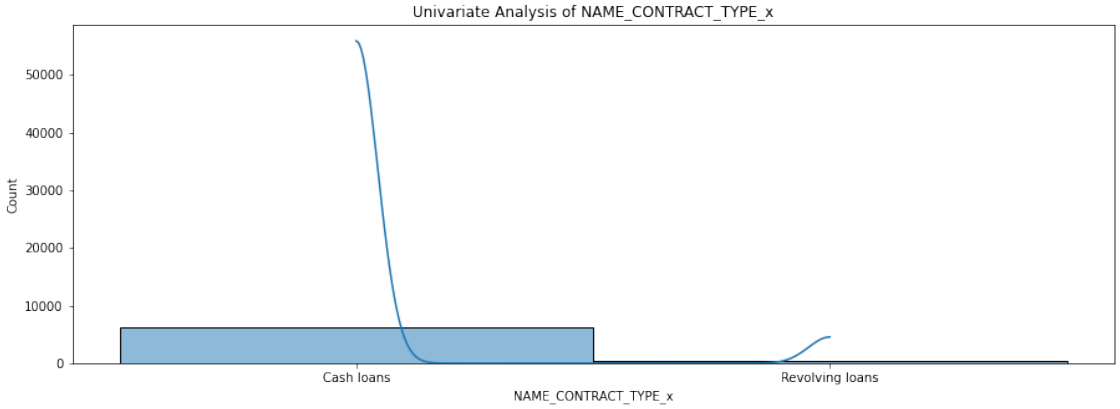
[68]:
```
segmenting_variable = 'CODE_GENDER'  # Variable for segmentation
target_variable = 'TARGET'  # Variable to analyze within segments

# Segmented Univariate Analysis - Numeric Variable within Segments
plt.figure(figsize=(12, 6))
sns.boxplot(data=merged_data, x=segmenting_variable, y=target_variable)
plt.xticks(rotation=45)
plt.title(f'Segmented Univariate Analysis of {target_variable} by␣
  ↪{segmenting_variable}')
plt.show()
```

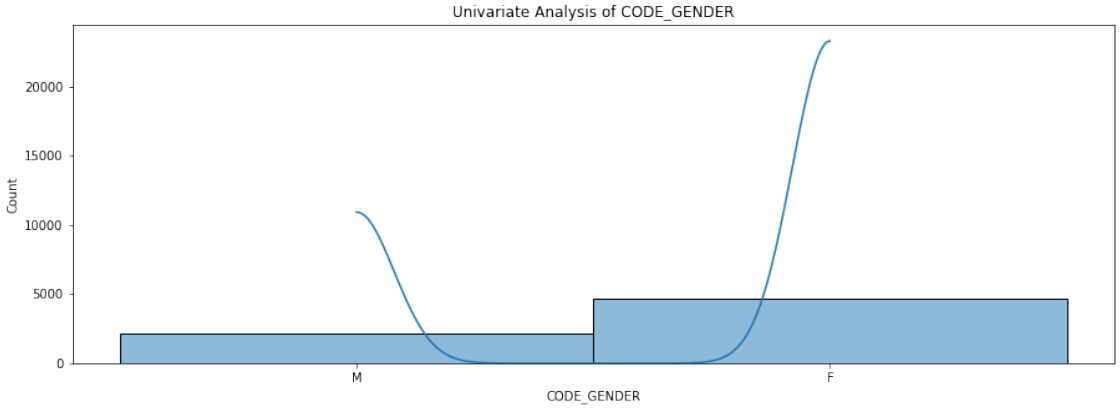


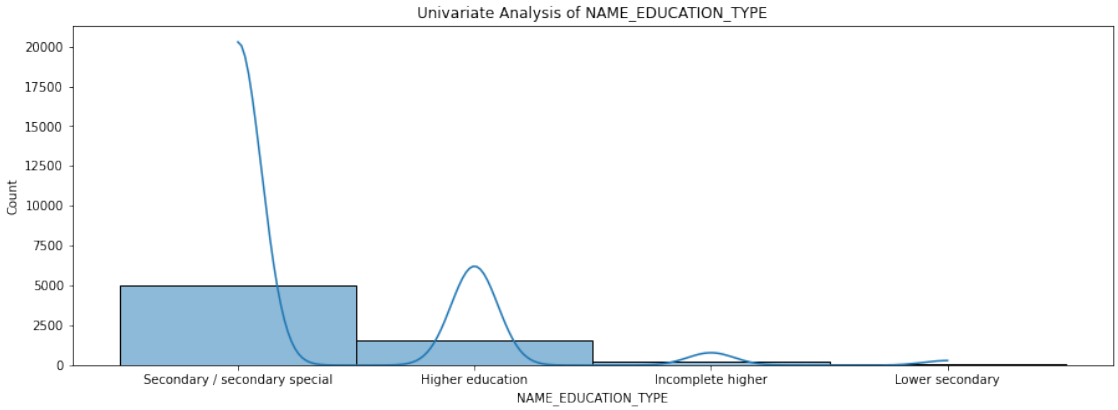
[69]:
```
columns_to_analyze = [
    'SK_ID_CURR',
    'TARGET',
    'NAME_CONTRACT_TYPE_x',
    'CODE_GENDER',
    'NAME_EDUCATION_TYPE',
    'FLAG_OWN_REALTY',
    'CNT_CHILDREN',
    'AMT_INCOME_TOTAL',
    'AMT_CREDIT_x',
    # Add more columns as needed
]

# Loop through columns for univariate analysis
for column in columns_to_analyze:
    plt.figure(figsize=(15, 5))
    sns.histplot(data=merged_data, x=column, kde=True)
    plt.title(f'Univariate Analysis of {column}')
    plt.show()
```

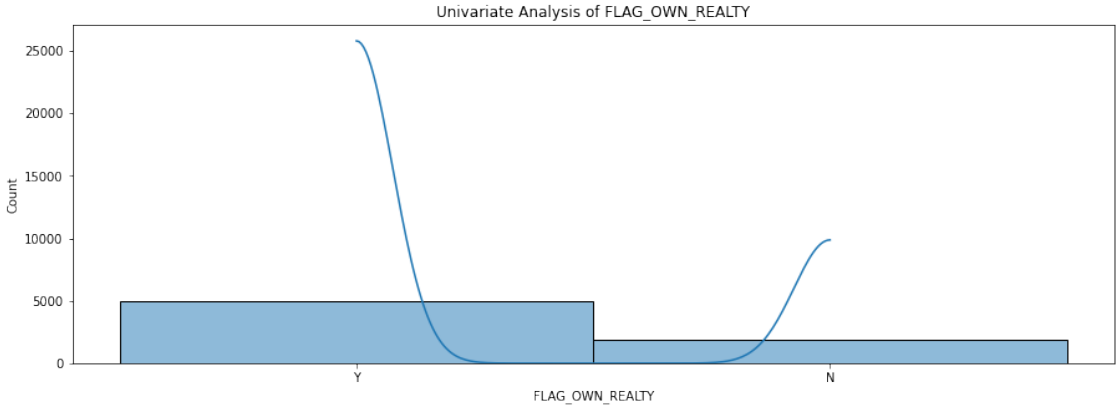


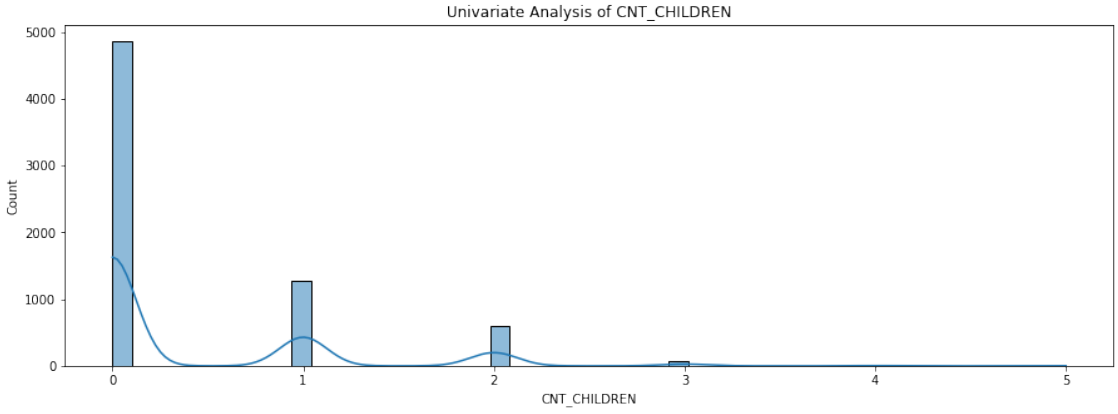

Univariate Analysis of TARGET
Count
TARGET
0
1000
2000
3000
4000
5000
6000
0.0
0.2
0.4
0.6
0.8
1.0

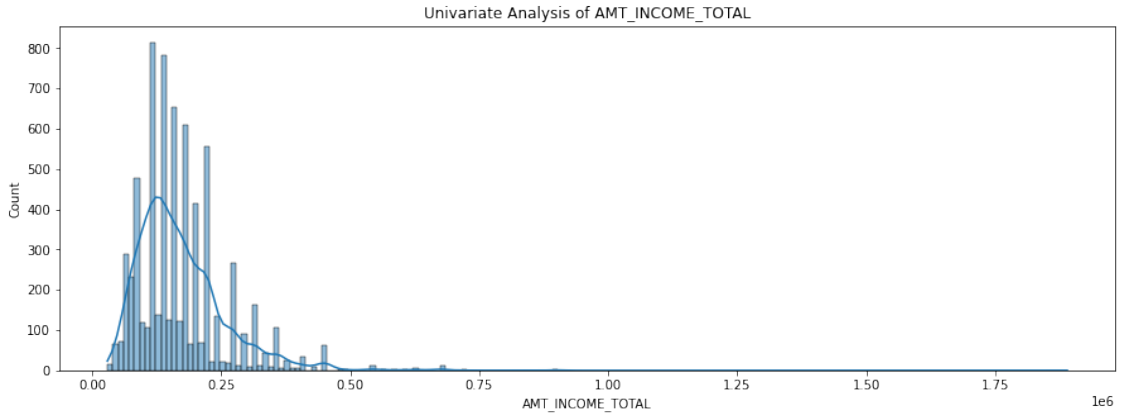

Univariate Analysis of NAME_CONTRACT_TYPE_x
Count
NAME_CONTRACT_TYPE_x
0
10000
20000
30000
40000
50000
Cash loans
Revolving loans

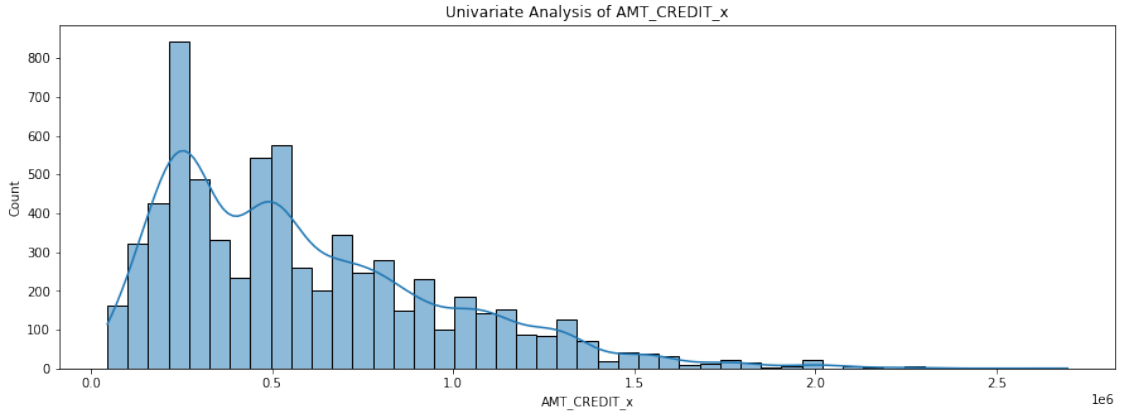

Univariate Analysis of CODE_GENDER
Count
CODE_GENDER
0
5000
10000
15000
20000
M
F


Univariate Analysis of NAME_EDUCATION_TYPE
Count
NAME_EDUCATION_TYPE
Secondary / secondary special
Higher education
Incomplete higher
Lower secondary


Univariate Analysis of FLAG_OWN_REALTY
Count
FLAG_OWN_REALTY
Y
N

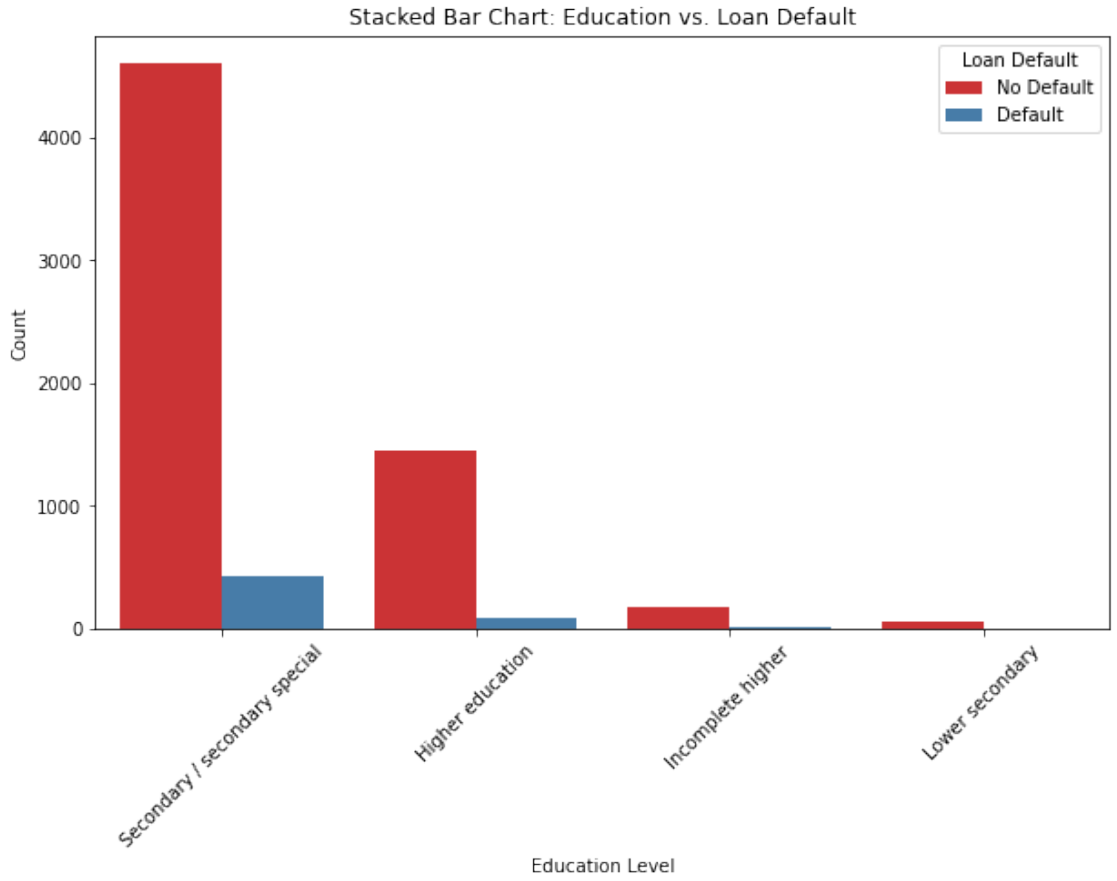

Univariate Analysis of CNT_CHILDREN
Count
CNT_CHILDREN

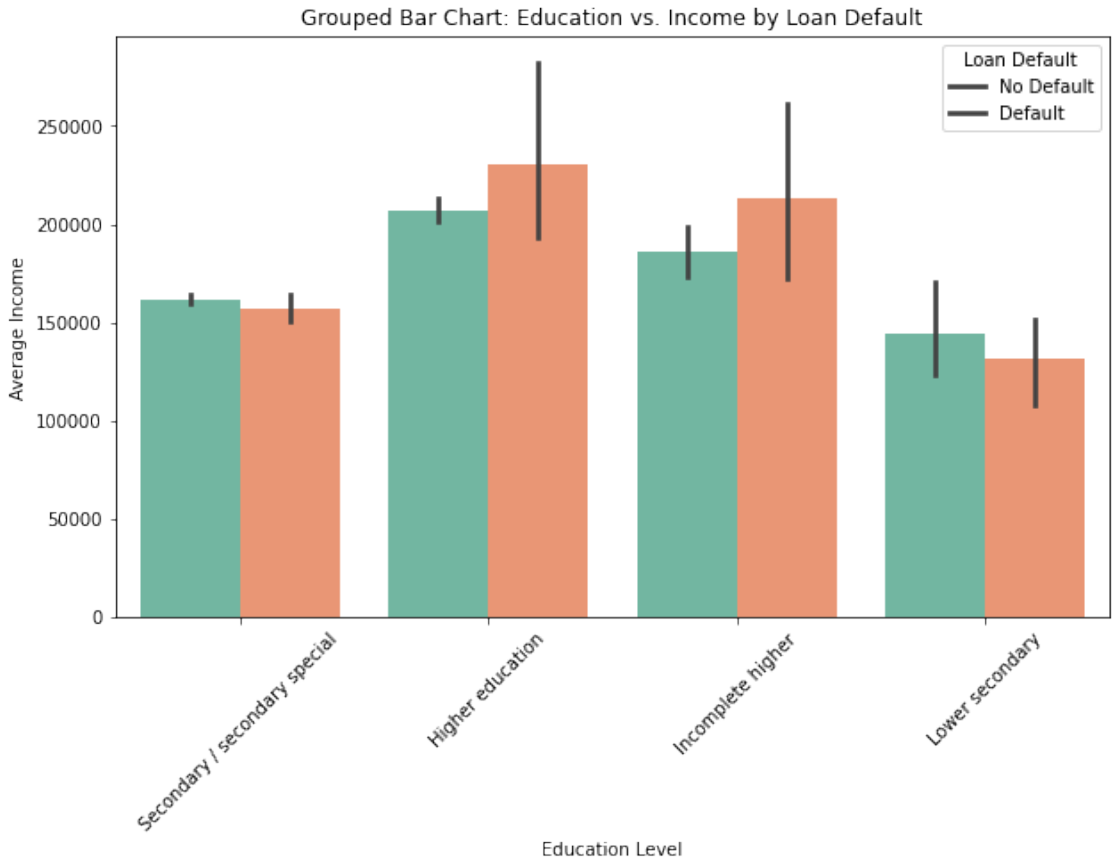


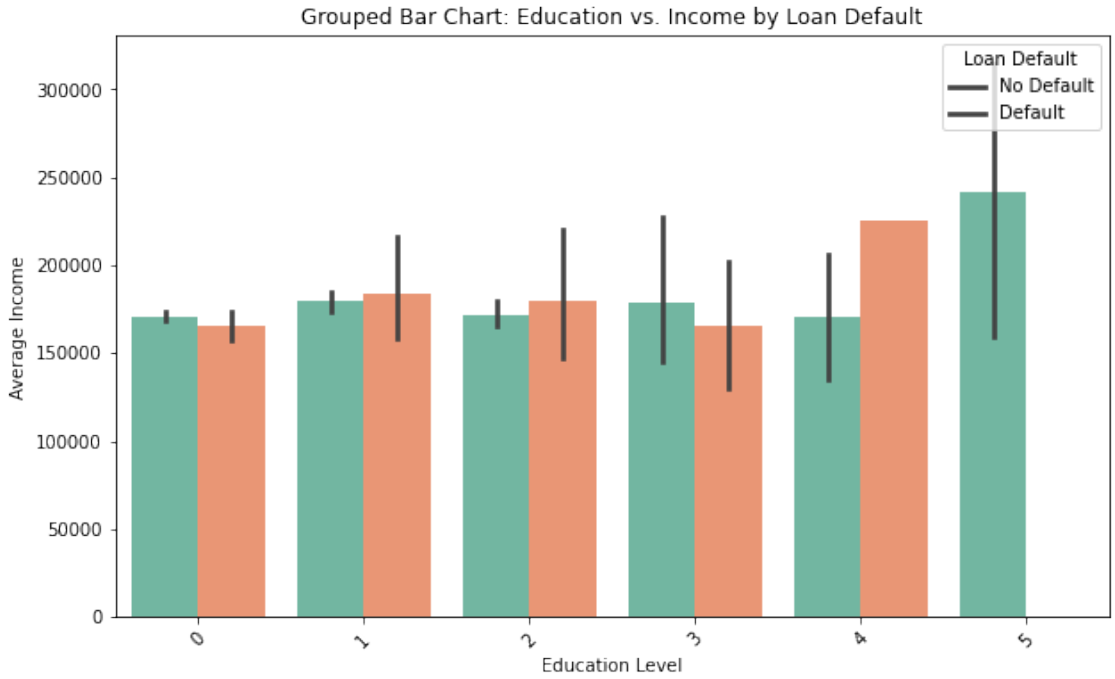


[70]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=merged_data, x='NAME_EDUCATION_TYPE', hue='TARGET',␣
 ↪palette='Set1')
plt.title('Stacked Bar Chart: Education vs. Loan Default')
plt.xlabel('Education Level')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.legend(title='Loan Default', loc='upper right', labels=['No Default',␣
 ↪'Default'])
plt.show()

# Create a grouped bar chart for comparing variable distributions across␣
 ↪scenarios
plt.figure(figsize=(10, 6))
sns.barplot(data=merged_data, x='NAME_EDUCATION_TYPE', y='AMT_INCOME_TOTAL',␣
 ↪hue='TARGET', palette='Set2')
plt.title('Grouped Bar Chart: Education vs. Income by Loan Default')
```

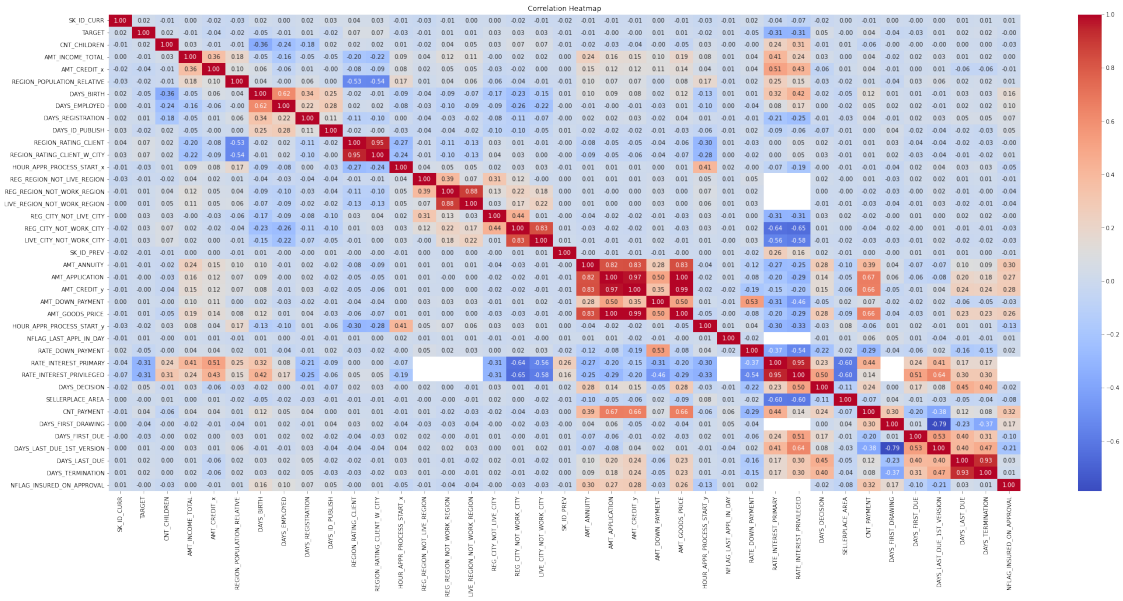
```
plt.xlabel('Education Level')
plt.ylabel('Average Income')
plt.xticks(rotation=45)
plt.legend(title='Loan Default', loc='upper right', labels=['No Default',␣
 ↪'Default'])
plt.show()

plt.figure(figsize=(10, 6))
sns.barplot(data=merged_data, x='CNT_CHILDREN', y='AMT_INCOME_TOTAL',␣
 ↪hue='TARGET', palette='Set2')
plt.title('Grouped Bar Chart: Education vs. Income by Loan Default')
plt.xlabel('Education Level')
plt.ylabel('Average Income')
plt.xticks(rotation=45)
plt.legend(title='Loan Default', loc='upper right', labels=['No Default',␣
 ↪'Default'])
plt.show()
```




Grouped Bar Chart: Education vs. Income by Loan Default
Loan Default
No Default
Default
Average Income
0
50000
100000
150000
200000
250000
Secondary / secondary special
Higher education
Incomplete higher
Lower secondary
Education Level


Grouped Bar Chart: Education vs. Income by Loan Default
Loan Default
No Default
Default
Average Income
0
50000
100000
150000
200000
250000
300000
0
1
2
3
4
5
Education Level

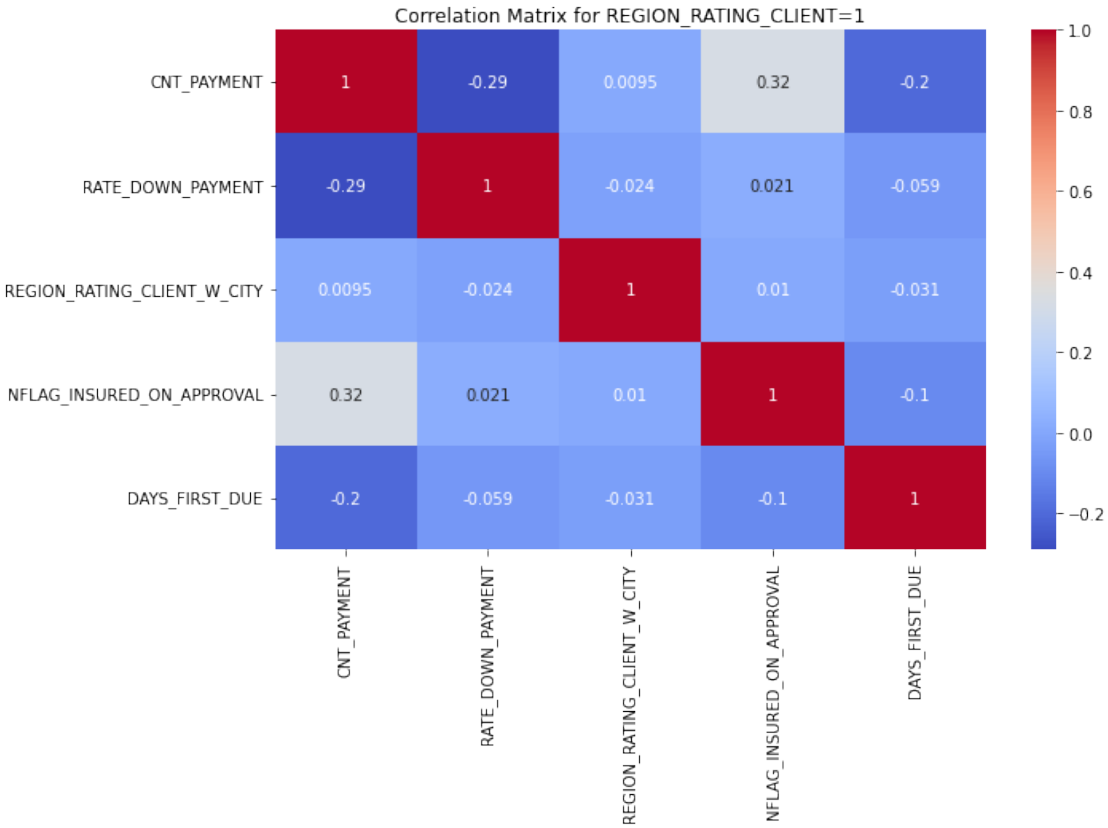
[71]:
```
correlation_matrix = merged_data.corr()

# Create a heatmap
plt.figure(figsize=(35, 15))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()
```



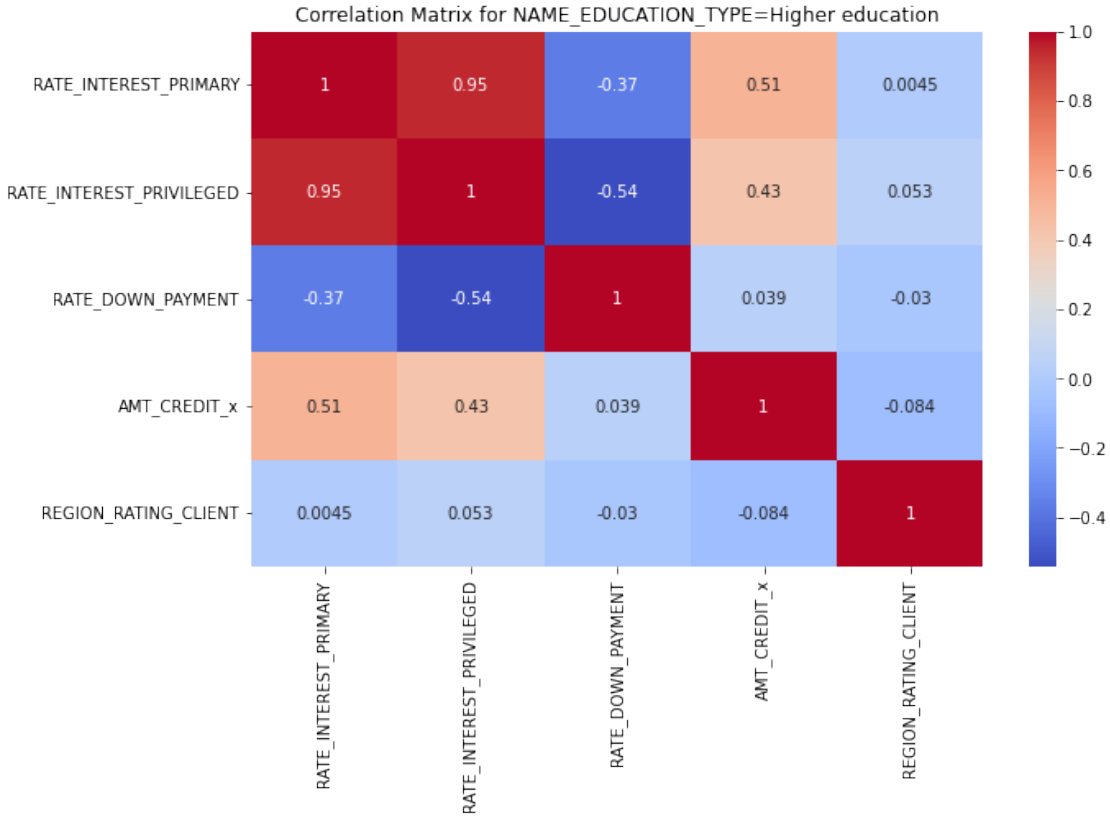
[72]:
```
def visualize_top_correlations(merged_data, scenario_variable, scenario_value,␣
↪target_variable, top_n=5):
    # Filter the DataFrame for the specific scenario
    scenario_df = merged_data[app_data[scenario_variable] == scenario_value]

    # Calculate correlations with the target variable
    correlations = scenario_df.corr()[target_variable].drop(target_variable)

    # Get the top correlated variables
    top_correlations = correlations.abs().nlargest(top_n)

    # Create a heatmap for the top correlated variables
    plt.figure(figsize=(10, 6))
    sns.heatmap(merged_data[top_correlations.index].corr(), annot=True,␣
↪cmap='coolwarm')
    plt.title(f'Correlation Matrix for {scenario_variable}={scenario_value}')
    plt.show()
```

```
# Example usage for different scenarios
visualize_top_correlations(merged_data, 'REGION_RATING_CLIENT', 1, 'TARGET')
visualize_top_correlations(merged_data, 'NAME_EDUCATION_TYPE', 'Higher␣
 ↪education', 'TARGET')
visualize_top_correlations(merged_data, 'NAME_INCOME_TYPE', 'Working', 'TARGET')
```

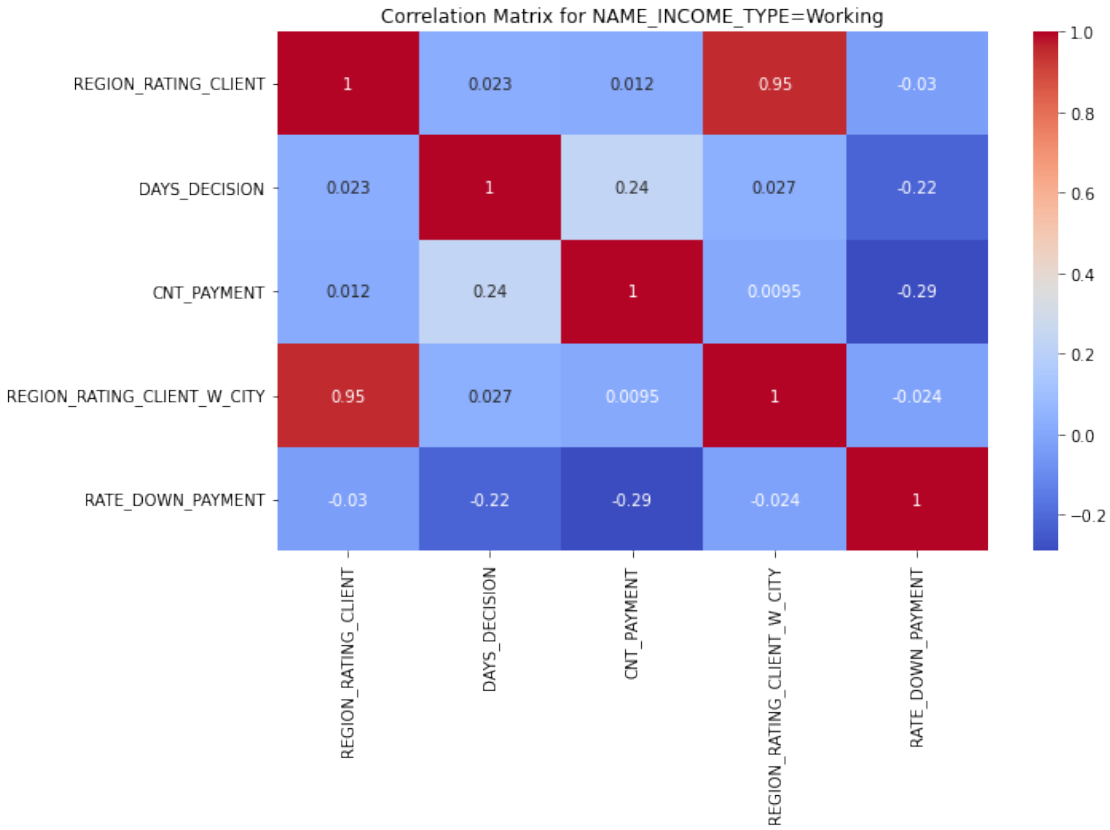
```
C:\Users\ANINDYA DAS\AppData\Local\Temp\ipykernel_14968\2942921166.py:3:
UserWarning: Boolean Series key will be reindexed to match DataFrame index.
  scenario_df = merged_data[app_data[scenario_variable] == scenario_value]
```




```
C:\Users\ANINDYA DAS\AppData\Local\Temp\ipykernel_14968\2942921166.py:3:
UserWarning: Boolean Series key will be reindexed to match DataFrame index.
  scenario_df = merged_data[app_data[scenario_variable] == scenario_value]
```




```
C:\Users\ANINDYA DAS\AppData\Local\Temp\ipykernel_14968\2942921166.py:3:
UserWarning: Boolean Series key will be reindexed to match DataFrame index.
  scenario_df = merged_data[app_data[scenario_variable] == scenario_value]
```

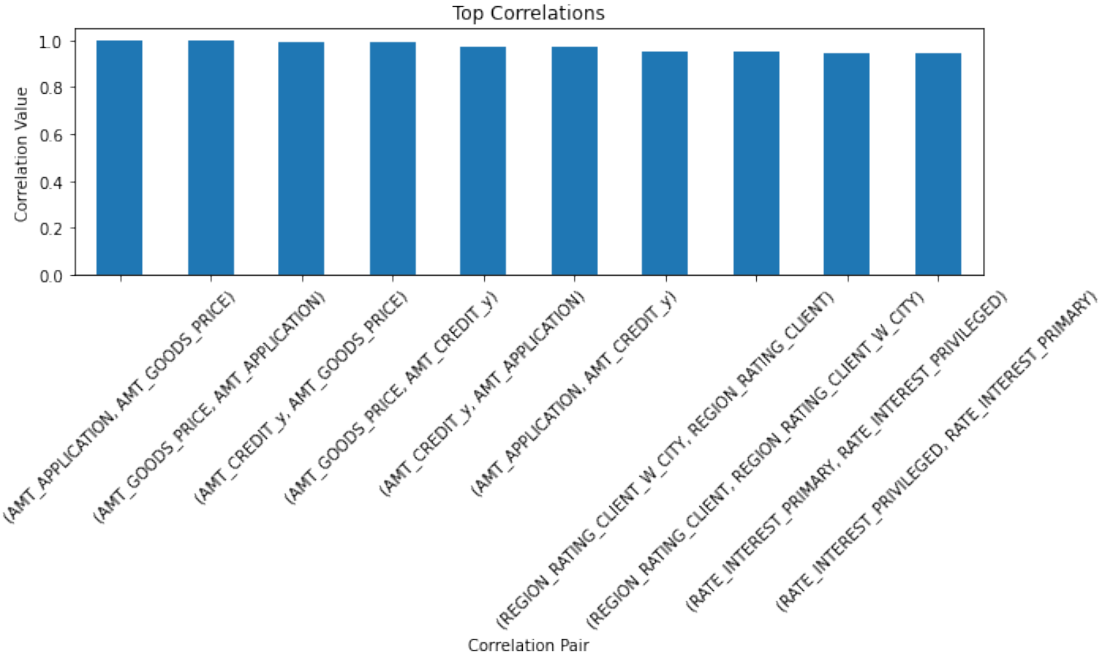


[73]:
```
correlation_matrix = merged_data.corr()

# Flatten the correlation matrix and sort by absolute correlation value
correlation_values = correlation_matrix.abs().unstack().
 ↪sort_values(ascending=False)

# Filter out correlations with themselves (diagonal elements)
correlation_values = correlation_values[correlation_values.index.
 ↪get_level_values(0) != correlation_values.index.get_level_values(1)]

# Get the top N correlations (e.g., top 10)
top_correlations = correlation_values.head(10)

# Print the top correlations
print(top_correlations)
```

```
AMT_APPLICATION          AMT_GOODS_PRICE          0.999981
AMT_GOODS_PRICE          AMT_APPLICATION          0.999981
AMT_CREDIT_y             AMT_GOODS_PRICE          0.993420
AMT_GOODS_PRICE          AMT_CREDIT_y             0.993420
AMT_CREDIT_y             AMT_APPLICATION          0.974375
```

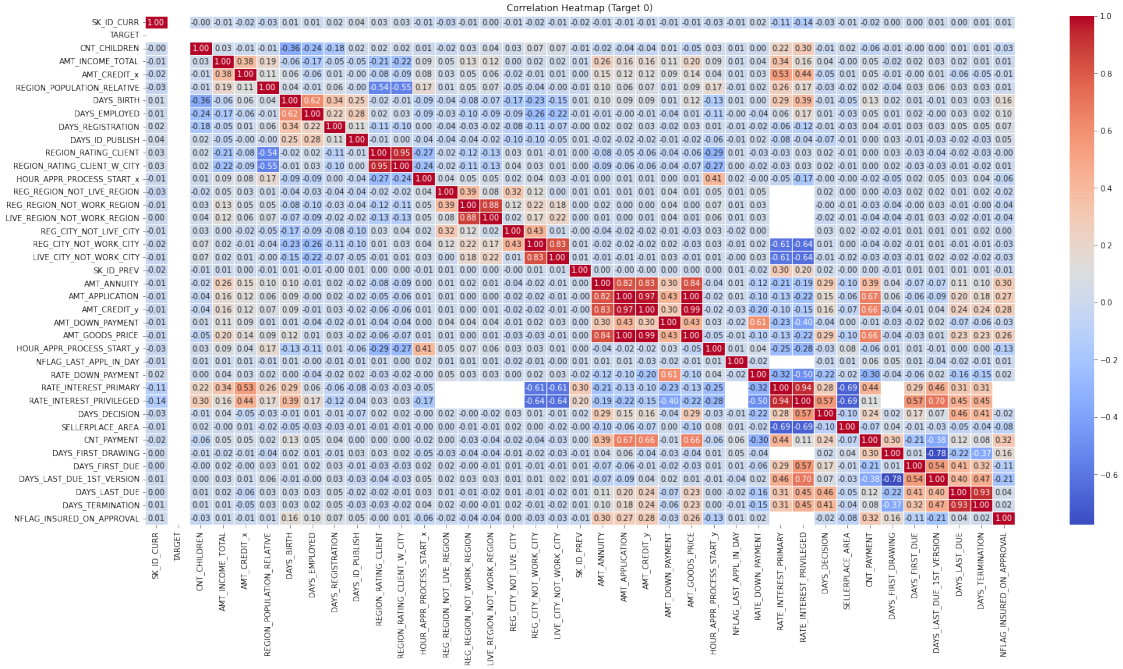
```
AMT_APPLICATION              AMT_CREDIT_y                  0.974375
REGION_RATING_CLIENT_W_CITY  REGION_RATING_CLIENT          0.950614
REGION_RATING_CLIENT         REGION_RATING_CLIENT_W_CITY   0.950614
RATE_INTEREST_PRIMARY        RATE_INTEREST_PRIVILEGED      0.947970
RATE_INTEREST_PRIVILEGED     RATE_INTEREST_PRIMARY         0.947970
dtype: float64
```

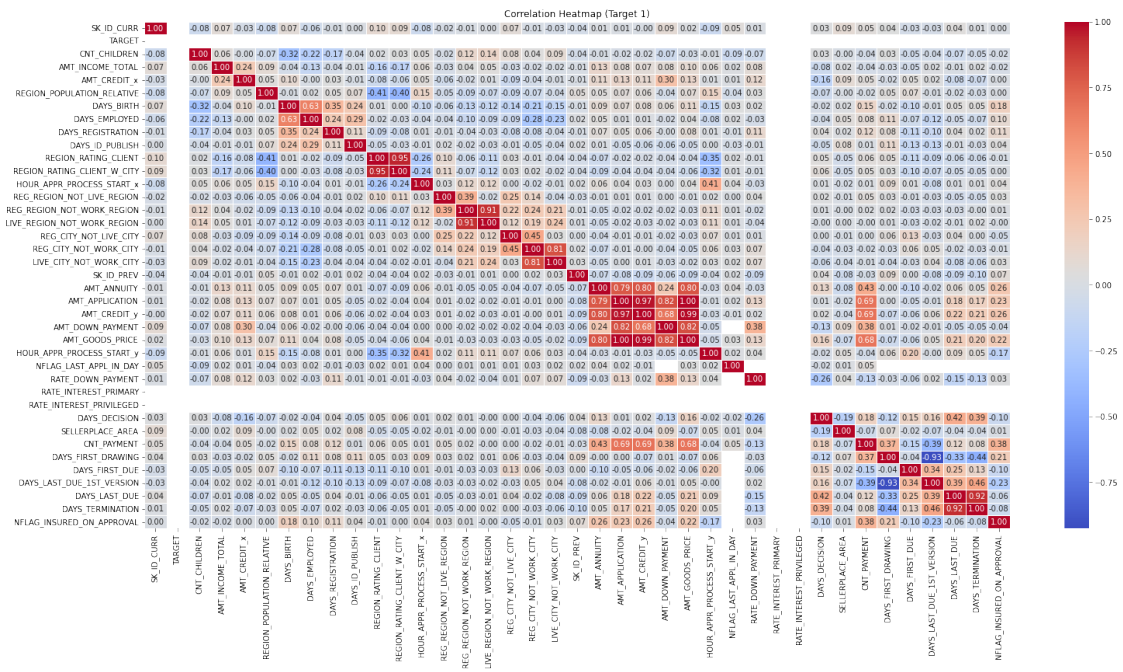
[74]:
```
plt.figure(figsize=(10, 6))
top_correlations.plot(kind='bar')
plt.title('Top Correlations')
plt.xlabel('Correlation Pair')
plt.ylabel('Correlation Value')
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
plt.tight_layout()

# Show the plot
plt.show()
```



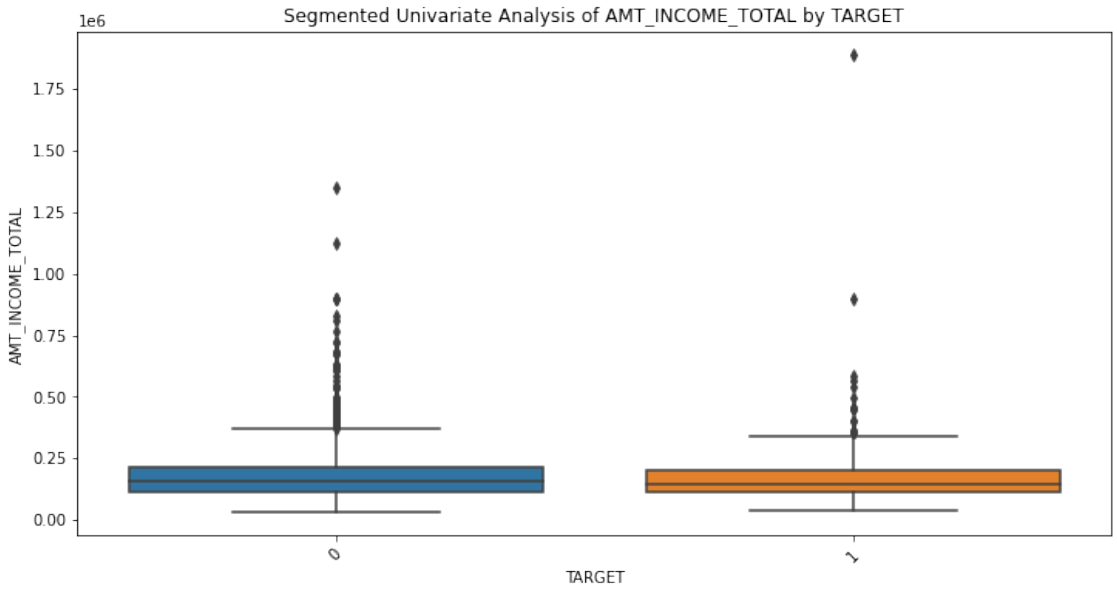
[75]:
```
# Calculate the correlation matrix when target is 0
corr_target_0 = merged_data[merged_data['TARGET'] == 0].corr()

# Calculate the correlation matrix when target is 1
corr_target_1 = merged_data[merged_data['TARGET'] == 1].corr()
```

[80]:
```
def plot_heatmap(corr_matrix, title):
    plt.figure(figsize=(25, 12))
    sns.heatmap(corr_matrix, cmap='coolwarm', annot=True, fmt=".2f",␣
 ↪linewidths=0.5)
    plt.title(title)
    plt.show()

# Create correlation matrices for target 0 and target 1
corr_matrix_target_0 = merged_data[merged_data['TARGET'] == 0].corr()
corr_matrix_target_1 = merged_data[merged_data['TARGET'] == 1].corr()

# Plot heatmap for correlations when target is 0
plot_heatmap(corr_matrix_target_0, 'Correlation Heatmap (Target 0)')

# Plot heatmap for correlations when target is 1
plot_heatmap(corr_matrix_target_1, 'Correlation Heatmap (Target 1)')
```

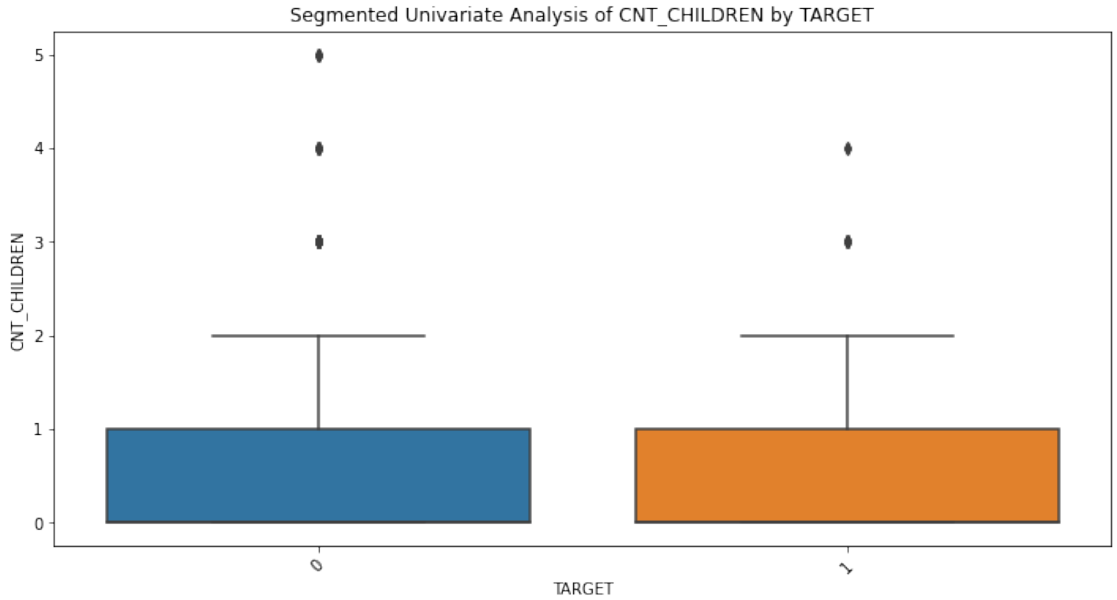




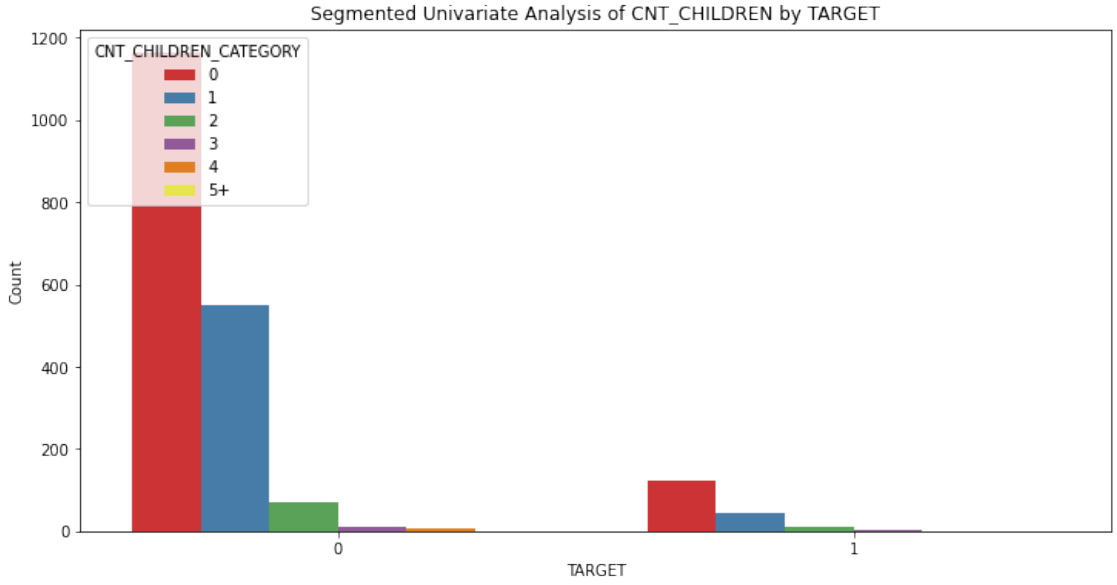
[81]:
```
segmenting_variable = 'TARGET'
target_variable = 'AMT_INCOME_TOTAL'  # Variable to analyze within segments

# Segmented Univariate Analysis - Numeric Variable within Segments
plt.figure(figsize=(12, 6))
sns.boxplot(data=merged_data, x=segmenting_variable, y=target_variable)
plt.xticks(rotation=45)
plt.title(f'Segmented Univariate Analysis of {target_variable} by␣
  ↪{segmenting_variable}')
plt.show()
```



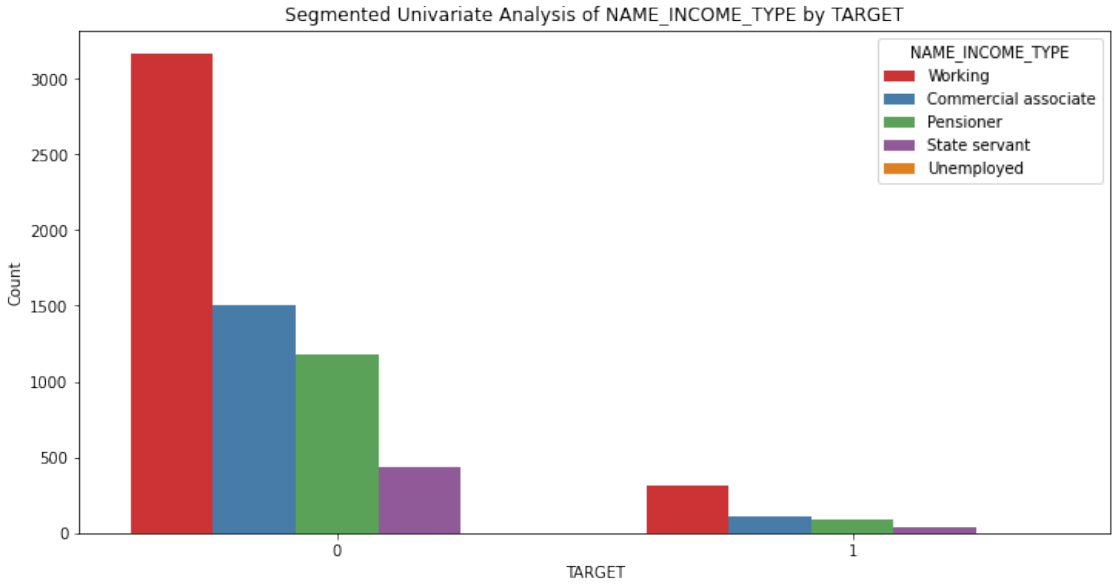
[82]:
```
segmenting_variable = 'TARGET'
target_variable = 'CNT_CHILDREN'  # Variable to analyze within segments

# Segmented Univariate Analysis - Numeric Variable within Segments
plt.figure(figsize=(12, 6))
sns.boxplot(data=merged_data, x=segmenting_variable, y=target_variable)
plt.xticks(rotation=45)
plt.title(f'Segmented Univariate Analysis of {target_variable} by␣
  ↪{segmenting_variable}')
plt.show()
```



[83]:
```
bins = [0, 1, 2, 3, 4, 5, float('inf')]
labels = ['0', '1', '2', '3', '4', '5+']
merged_data['CNT_CHILDREN_CATEGORY'] = pd.cut(merged_data['CNT_CHILDREN'],␣
 ↪bins=bins, labels=labels)

# Create a stacked bar chart
plt.figure(figsize=(12, 6))
sns.countplot(data=merged_data, x='TARGET', hue='CNT_CHILDREN_CATEGORY',␣
 ↪palette='Set1')
plt.xticks(rotation=0)
plt.title('Segmented Univariate Analysis of CNT_CHILDREN by TARGET')
plt.xlabel('TARGET')
plt.ylabel('Count')
plt.legend(title='CNT_CHILDREN_CATEGORY')
plt.show()
```



[89]:
```
plt.figure(figsize=(12, 6))
sns.countplot(data=merged_data, x='TARGET', hue='NAME_INCOME_TYPE',␣
 ↪palette='Set1')
plt.xticks(rotation=0)
plt.title('Segmented Univariate Analysis of NAME_INCOME_TYPE by TARGET')
plt.xlabel('TARGET')
plt.ylabel('Count')
plt.legend(title='NAME_INCOME_TYPE')
plt.show()
```


Segmented Univariate Analysis of NAME_INCOME_TYPE by TARGET
NAME_INCOME_TYPE
Working
Commercial associate
Pensioner
State servant
Unemployed
Count
3000
2500
2000
1500
1000
500
0
0
1
TARGET

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: