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

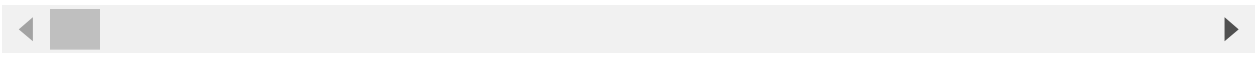
In [2]:
```
pd.set_option('display.max_columns', 300) # to display all the columns
pd.set_option('display.max_rows', 300) # to display all the rows
pd.set_option('display.width', 1000)
```

In [3]:
```
apply=pd.read_csv("application_data.csv")
# for i in apply.columns:
#     print(i)
prev_apply1=pd.read_csv("previous_application.csv")
apply
```

Out[3]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG
0	100002	1	Cash loans	M	N	
1	100003	0	Cash loans	F	N	
2	100004	0	Revolving loans	M	Y	
3	100006	0	Cash loans	F	N	
4	100007	0	Cash loans	M	N	
...	...	...	...	...	...	
307506	456251	0	Cash loans	M	N	
307507	456252	0	Cash loans	F	N	
307508	456253	0	Cash loans	F	N	
307509	456254	1	Cash loans	F	N	
307510	456255	0	Cash loans	F	N	

307511 rows × 122 columns



In [4]:
```
apply.shape
apply.dtypes.value_counts()
```

Out[4]:
```
float64    65
int64      41
object     16
dtype: int64
```

In [5]:
```
apply.head()
```

Out[5]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN
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	

In [6]:
```
apply.describe()
```

Out[6]:

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_A
count	307511.000000	307511.000000	307511.000000	3.075110e+05	3.075110e+05	307499
mean	278180.518577	0.080729	0.417052	1.687979e+05	5.990260e+05	27108
std	102790.175348	0.272419	0.722121	2.371231e+05	4.024908e+05	14493
min	100002.000000	0.000000	0.000000	2.565000e+04	4.500000e+04	1615
25%	189145.500000	0.000000	0.000000	1.125000e+05	2.700000e+05	16524
50%	278202.000000	0.000000	0.000000	1.471500e+05	5.135310e+05	24903
75%	367142.500000	0.000000	1.000000	2.025000e+05	8.086500e+05	34596
max	456255.000000	1.000000	19.000000	1.170000e+08	4.050000e+06	258025

In [7]:
```
apply.agg(['nunique'])
```

Out[7]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLA
nunique	307511	2	2	3	2	

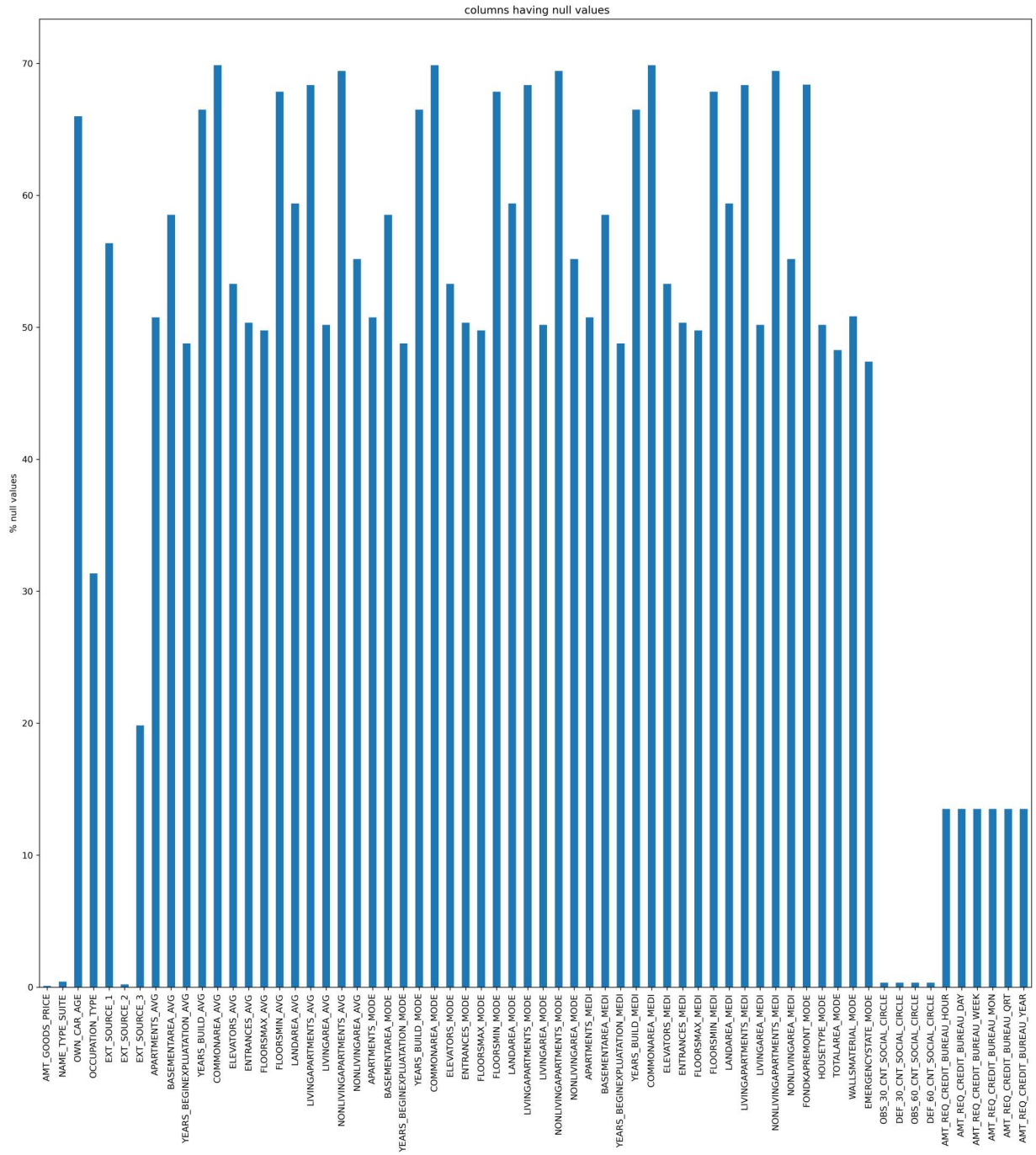
In [8]:
```
output = round(apply.isnull().sum()/len(apply.index)*100,2)
output
# Null_col = apply1.columns[apply1.isnull().mean()>0]
# Null_col
output = output[output>0]
output
```

Out[8]:

AMT_GOODS_PRICE	0.09
NAME_TYPE_SUITE	0.42
OWN_CAR_AGE	65.99
OCCUPATION_TYPE	31.35
EXT_SOURCE_1	56.38
EXT_SOURCE_2	0.21
EXT_SOURCE_3	19.83
APARTMENTS_AVG	50.75
BASEMENTAREA_AVG	58.52
YEARS_BEGINEXPLUATATION_AVG	48.78
YEARS_BUILD_AVG	66.50
COMMONAREA_AVG	69.87
ELEVATORS_AVG	53.30
ENTRANCES_AVG	50.35
FLOORSMAX_AVG	49.76
FLOORSMIN_AVG	67.85
LANDAREA_AVG	59.38
LIVINGAPARTMENTS_AVG	68.35
LIVINGAREA_AVG	50.19
NONLIVINGAPARTMENTS_AVG	69.43
NONLIVINGAREA_AVG	55.18
APARTMENTS_MODE	50.75
BASEMENTAREA_MODE	58.52
YEARS_BEGINEXPLUATATION_MODE	48.78
YEARS_BUILD_MODE	66.50
COMMONAREA_MODE	69.87
ELEVATORS_MODE	53.30
ENTRANCES_MODE	50.35
FLOORSMAX_MODE	49.76
FLOORSMIN_MODE	67.85
LANDAREA_MODE	59.38
LIVINGAPARTMENTS_MODE	68.35
LIVINGAREA_MODE	50.19
NONLIVINGAPARTMENTS_MODE	69.43
NONLIVINGAREA_MODE	55.18
APARTMENTS_MEDI	50.75
BASEMENTAREA_MEDI	58.52
YEARS_BEGINEXPLUATATION_MEDI	48.78
YEARS_BUILD_MEDI	66.50
COMMONAREA_MEDI	69.87
ELEVATORS_MEDI	53.30
ENTRANCES_MEDI	50.35
FLOORSMAX_MEDI	49.76
FLOORSMIN_MEDI	67.85
LANDAREA_MEDI	59.38
LIVINGAPARTMENTS_MEDI	68.35
LIVINGAREA_MEDI	50.19
NONLIVINGAPARTMENTS_MEDI	69.43

```
NONLIVINGAREA_MEDI             55.18
FONDKAPREMONT_MODE             68.39
HOUSETYPE_MODE                 50.18
TOTALAREA_MODE                 48.27
WALLSMATERIAL_MODE             50.84
EMERGENCYSTATE_MODE            47.40
OBS_30_CNT_SOCIAL_CIRCLE        0.33
DEF_30_CNT_SOCIAL_CIRCLE        0.33
OBS_60_CNT_SOCIAL_CIRCLE        0.33
DEF_60_CNT_SOCIAL_CIRCLE        0.33
AMT_REQ_CREDIT_BUREAU_HOUR     13.50
AMT_REQ_CREDIT_BUREAU_DAY      13.50
AMT_REQ_CREDIT_BUREAU_WEEK     13.50
AMT_REQ_CREDIT_BUREAU_MON      13.50
AMT_REQ_CREDIT_BUREAU_QRT      13.50
AMT_REQ_CREDIT_BUREAU_YEAR     13.50
dtype: float64
```

In [9]:
```
plt.figure(figsize= (20,20),dpi=300)
output.plot(kind = 'bar')
plt.title (' columns having null values')
plt.ylabel('% null values')
plt.show()
```



HERE NULL VALUE PERCENTAGES OF EACH COLUMN IS SHOWN THROUGH THIS GRAPH

BASED ON THIS GRAPH,WE CAN SEE WHICH COLUMNS HAVE MORE THAN 50% AND LESS THAN 50% NULL VALUES

In [10]:
```
null=apply.isnull().mean().sort_values(ascending=False)
null*100
apply.columns[apply.isnull().mean()<=0.15]
```

Out[10]:
```
Index(['SK_ID_CURR', 'TARGET', 'NAME_CONTRACT_TYPE', 'CODE_GENDER', 'FLAG_OWN_C
AR', 'FLAG_OWN_REALTY', 'CNT_CHILDREN', 'AMT_INCOME_TOTAL', 'AMT_CREDIT', 'AMT_
ANNUITY', 'AMT_GOODS_PRICE', 'NAME_TYPE_SUITE', 'NAME_INCOME_TYPE', 'NAME_EDUCA
TION_TYPE', 'NAME_FAMILY_STATUS', 'NAME_HOUSING_TYPE', 'REGION_POPULATION_RELAT
IVE', 'DAYS_BIRTH', 'DAYS_EMPLOYED', 'DAYS_REGISTRATION', 'DAYS_ID_PUBLISH', 'F
LAG_MOBIL', 'FLAG_EMP_PHONE', 'FLAG_WORK_PHONE', 'FLAG_CONT_MOBILE', 'FLAG_PHON
E', 'FLAG_EMAIL', 'CNT_FAM_MEMBERS', 'REGION_RATING_CLIENT', 'REGION_RATING_CLI
ENT_W_CITY', 'WEEKDAY_APPR_PROCESS_START', 'HOUR_APPR_PROCESS_START', 'REG_REGI
ON_NOT_LIVE_REGION', 'REG_REGION_NOT_WORK_REGION', 'LIVE_REGION_NOT_WORK_REGIO
N', 'REG_CITY_NOT_LIVE_CITY', 'REG_CITY_NOT_WORK_CITY', 'LIVE_CITY_NOT_WORK_CIT
Y', 'ORGANIZATION_TYPE', 'EXT_SOURCE_2', 'OBS_30_CNT_SOCIAL_CIRCLE', 'DEF_30_CN
T_SOCIAL_CIRCLE', 'OBS_60_CNT_SOCIAL_CIRCLE', 'DEF_60_CNT_SOCIAL_CIRCLE', 'DAYS
_LAST_PHONE_CHANGE', '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_1
0', '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', 'AMT_REQ_CRED
IT_BUREAU_HOUR', 'AMT_REQ_CREDIT_BUREAU_DAY', 'AMT_REQ_CREDIT_BUREAU_WEEK', 'AM
T_REQ_CREDIT_BUREAU_MON', 'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_CREDIT_BUREAU_Y
EAR'],
      dtype='object')
```

In [11]:
```
default=apply[apply["TARGET"]==1]
non_default=apply[apply["TARGET"]==0]
default
# DATA HAS BEEN DIVIDED INTO DEFAULTERS AND NON-DEFAULTERS
```

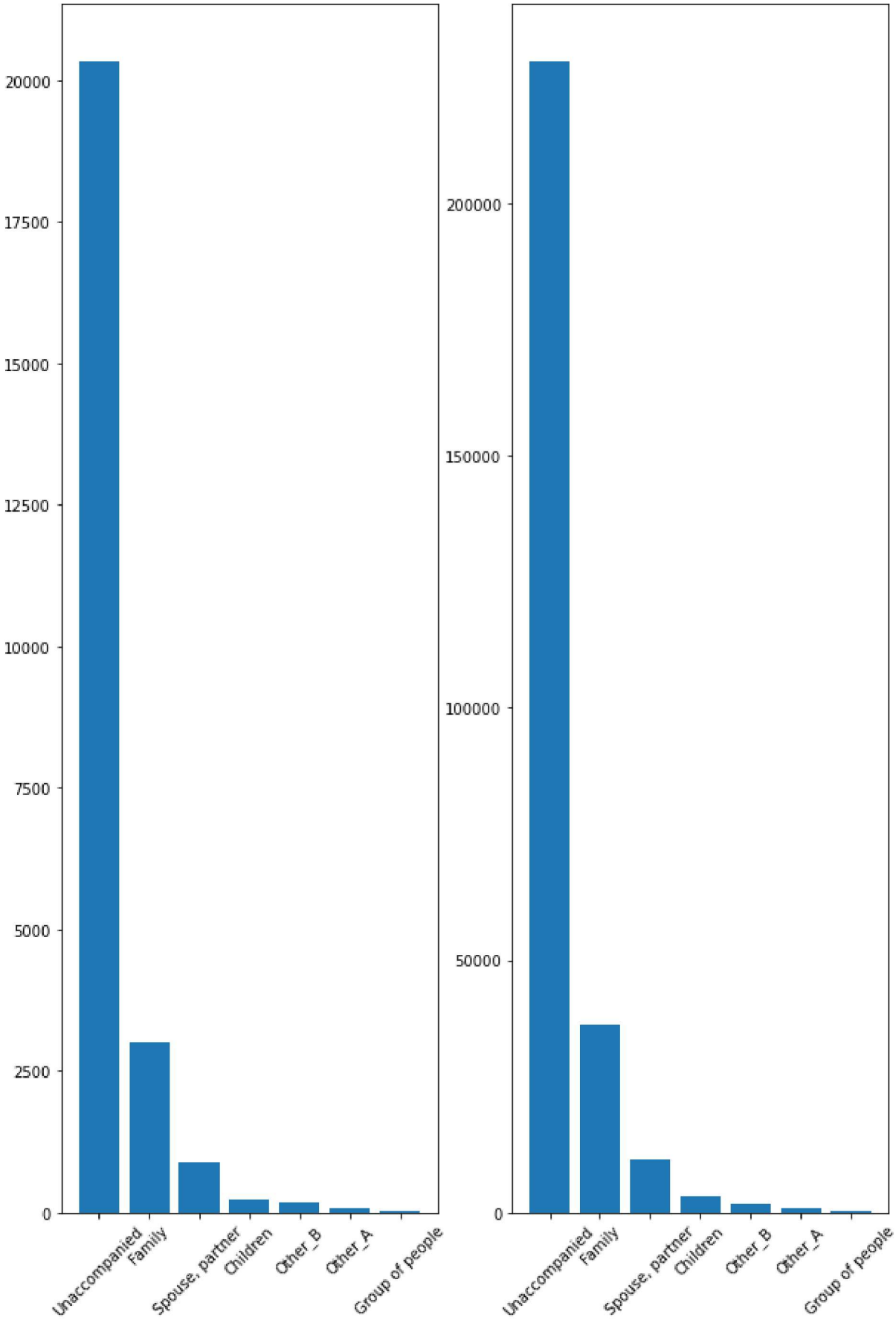
Out[11]:

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG
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	
...	...	...	...	...	...	
307448	456186	1	Cash loans	M	N	
307475	456215	1	Cash loans	F	N	
307481	456225	1	Cash loans	M	N	
307489	456233	1	Cash loans	F	N	
307509	456254	1	Cash loans	F	N	

24825 rows × 122 columns

In [12]:
```
suite1=default["NAME_TYPE_SUITE"].value_counts(ascending=False)
suite2=non_default["NAME_TYPE_SUITE"].value_counts(ascending=False)

fig,(ax1,ax2)=plt.subplots(1,2,figsize=(10,15))
#sns.countplot(suite1,data=default)

plt.subplot(1,2,1)
plt.bar(suite1.keys(),suite1.values)
#ax1.set_xticklabels(ax1.get_xticklabels(), rotation=90)
plt.xticks(rotation=45)
plt.subplot(1,2,2)
plt.xticks(rotation=45)
#ax2.set_xticklabels(ax2.get_xticklabels(), rotation=90)
plt.bar(suite2.keys(),suite2.values)
```

Out[12]:
```
<BarContainer object of 7 artists>
```



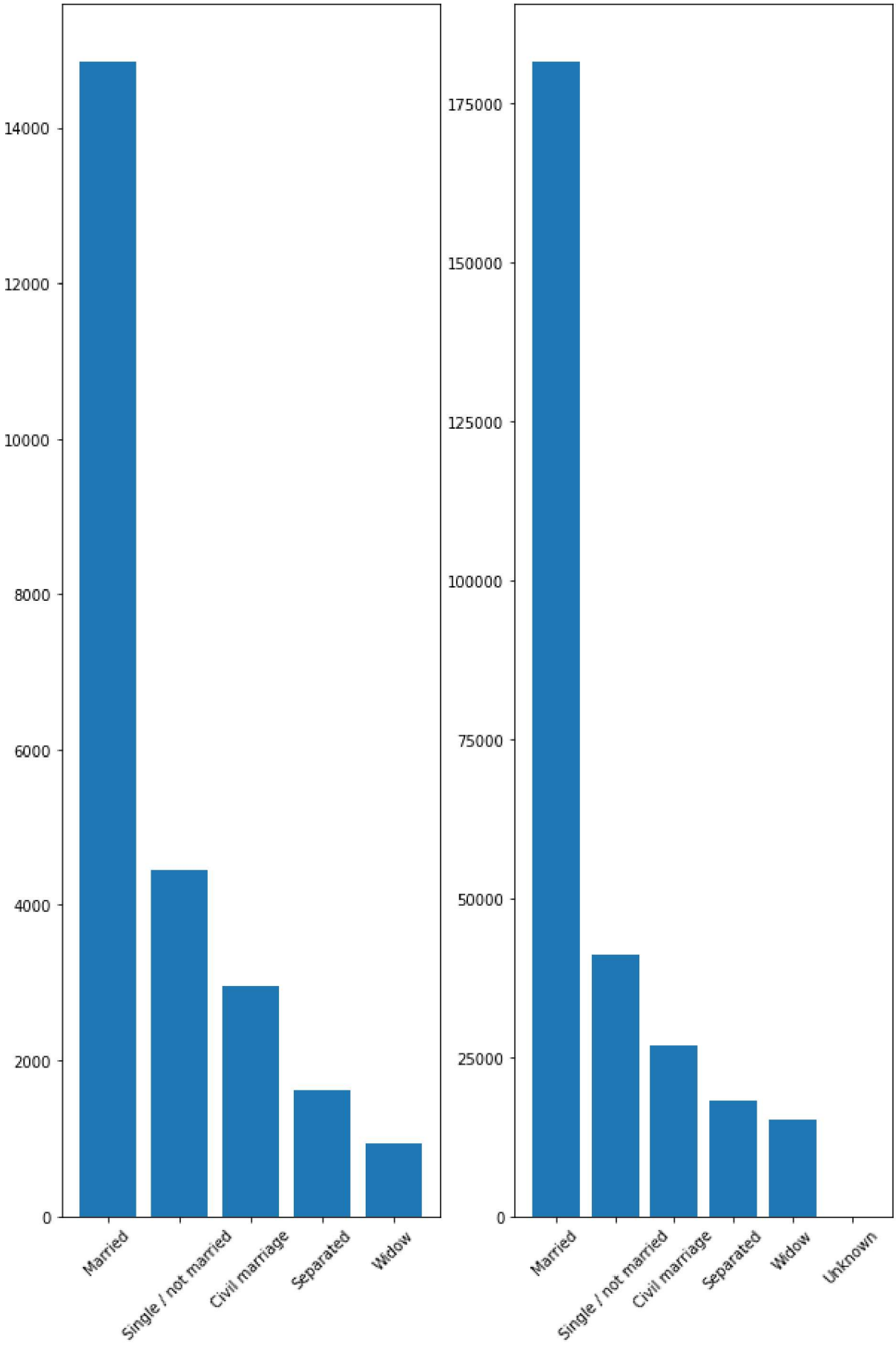
UNACCOMAPNIED FAMILIES ARE THE ONES

WHO APPLY MORE LOANS THAN OTHERS AND

WE CAN SEE THAT NON-DEFAULTERS ARE MORE IN NUMBER THAN IN DEFAULTERS IN THIS CATEGORY

In [13]:
```
status1=default["NAME_FAMILY_STATUS"].value_counts(ascending=False)
status2=non_default["NAME_FAMILY_STATUS"].value_counts(ascending=False)

fig,(ax1,ax2)=plt.subplots(1,2,figsize=(10,15))
#sns.countplot(suite1,data=default)

plt.subplot(1,2,1)
plt.bar(status1.keys(),status1.values)
plt.xticks(rotation=45)
plt.subplot(1,2,2)
plt.xticks(rotation=45)
plt.bar(status2.keys(),status2.values)
```

Out[13]:
```
<BarContainer object of 6 artists>
```



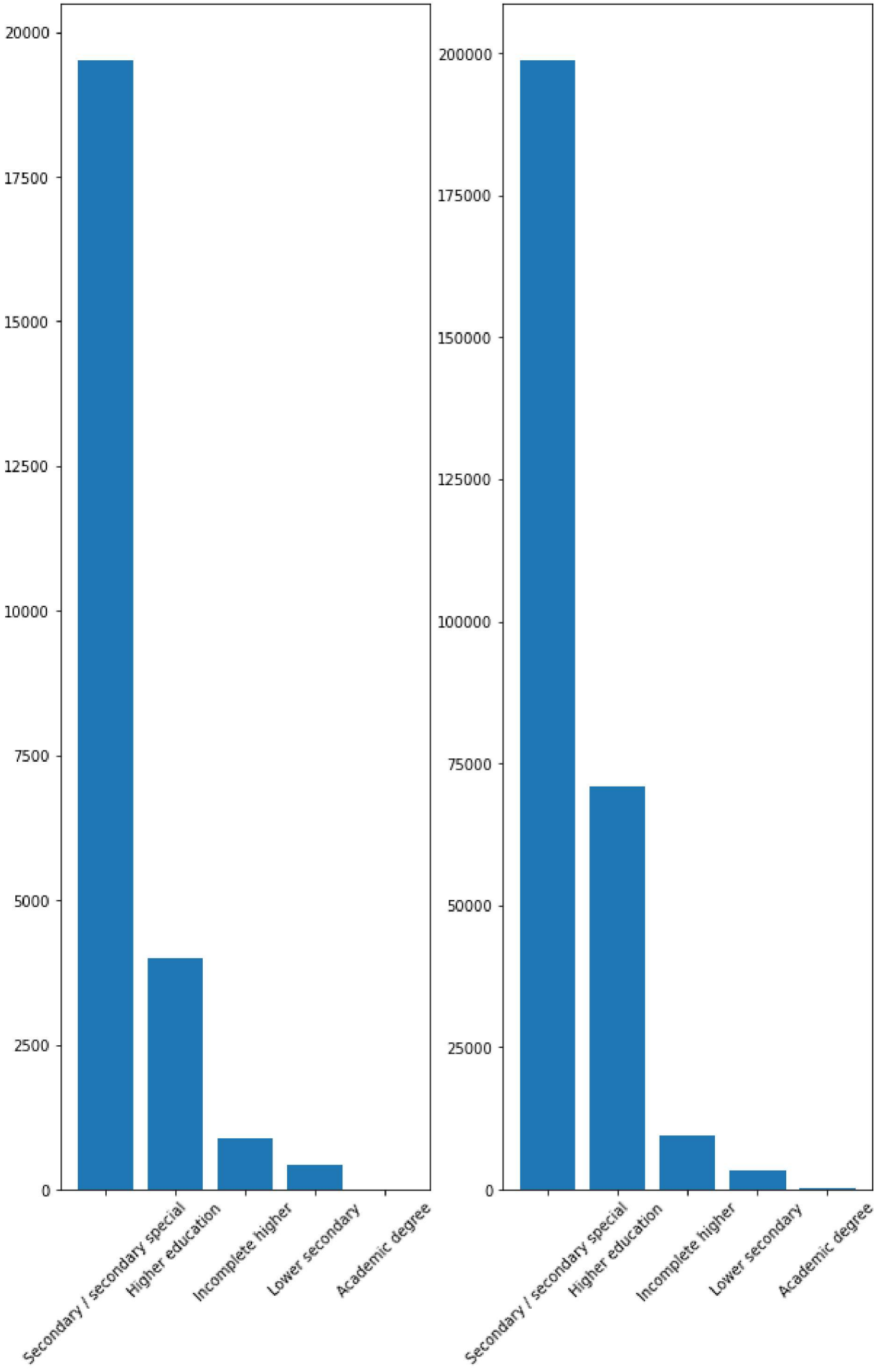
MARRIED PEOPLE APPLY MORE LOAN THAN

OTHERS

IN SINGLE/NOT-MARRIED CATEGORY,DEFAULTERS ARE MORE IN NUMBER THAN NON-DEFAULTERS

In [14]:
```
edu1=default["NAME_EDUCATION_TYPE"].value_counts(ascending=False)
edu2=non_default["NAME_EDUCATION_TYPE"].value_counts(ascending=False)

fig,(ax1,ax2)=plt.subplots(1,2,figsize=(10,15))
#sns.countplot(suite1,data=default)

plt.subplot(1,2,1)
plt.bar(edu1.keys(),edu1.values)
plt.xticks(rotation=45)
plt.subplot(1,2,2)
plt.xticks(rotation=45)
plt.bar(edu2.keys(),edu2.values)
```

Out[14]:
```
<BarContainer object of 5 artists>
```

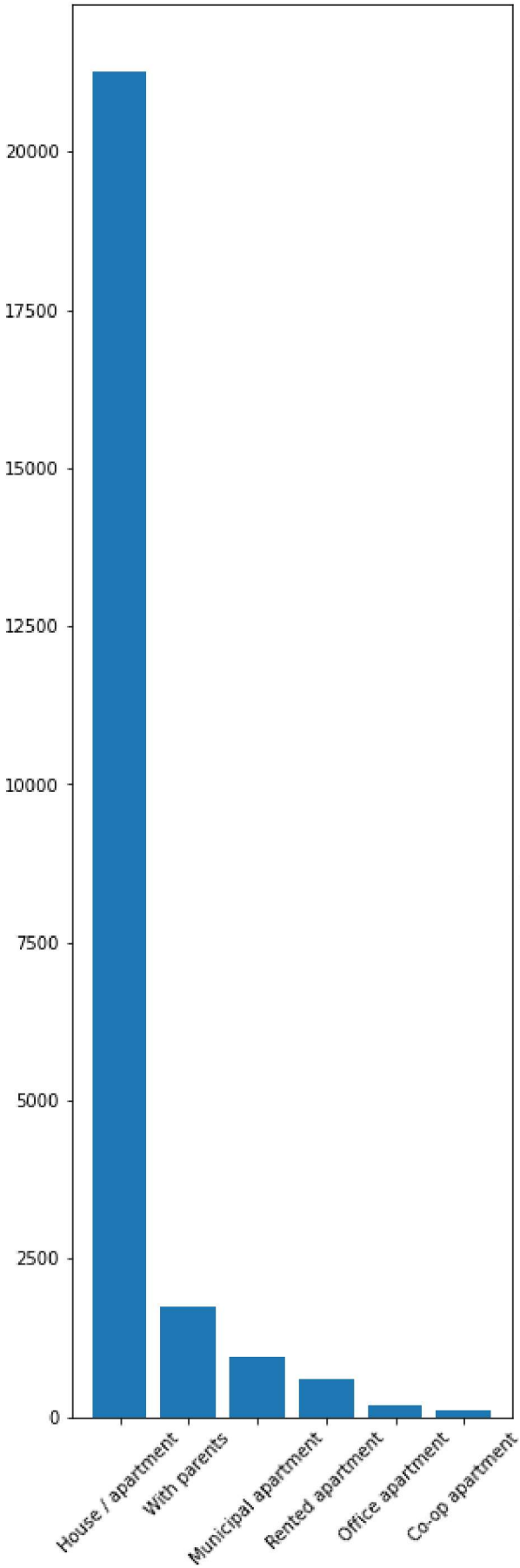

20000
17500
15000
12500
10000
7500
5000
2500
0
Secondary / secondary special
Higher education
Incomplete higher
Lower secondary
Academic degree
200000
175000
150000
125000
100000
75000
50000
25000
0
Secondary / secondary special
Higher education
Incomplete higher
Lower secondary
Academic degree

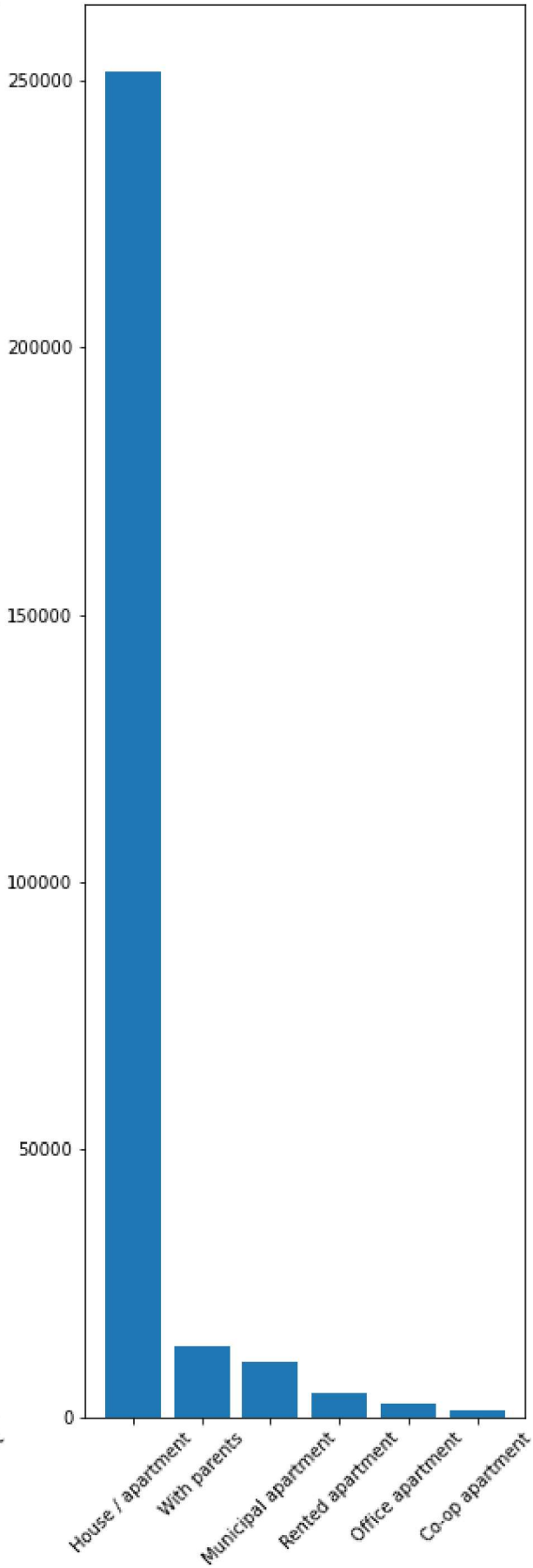
SECONDARY/SECONDARY SPECIAL APPLY MORE LOANS THAN OTHERS AND TEND TO BE MORE DEFAULT THAN OTHER CATEGORY STUDENTS.

MOST OF THE HIGHER EDUCATION STUDENTS ARE NON-DEFAULTERS.

In [15]:
```
housing1=default["NAME_HOUSING_TYPE"].value_counts(ascending=False)
housing2=non_default["NAME_HOUSING_TYPE"].value_counts(ascending=False)

fig,(ax1,ax2)=plt.subplots(1,2,figsize=(10,15))
#sns.countplot(suite1,data=default)

plt.subplot(1,2,1)
plt.bar(housing1.keys(),housing1.values)
plt.xticks(rotation=45)
plt.subplot(1,2,2)
plt.xticks(rotation=45)
plt.bar(housing2.keys(),housing2.values)
```

Out[15]:
```
<BarContainer object of 6 artists>
```

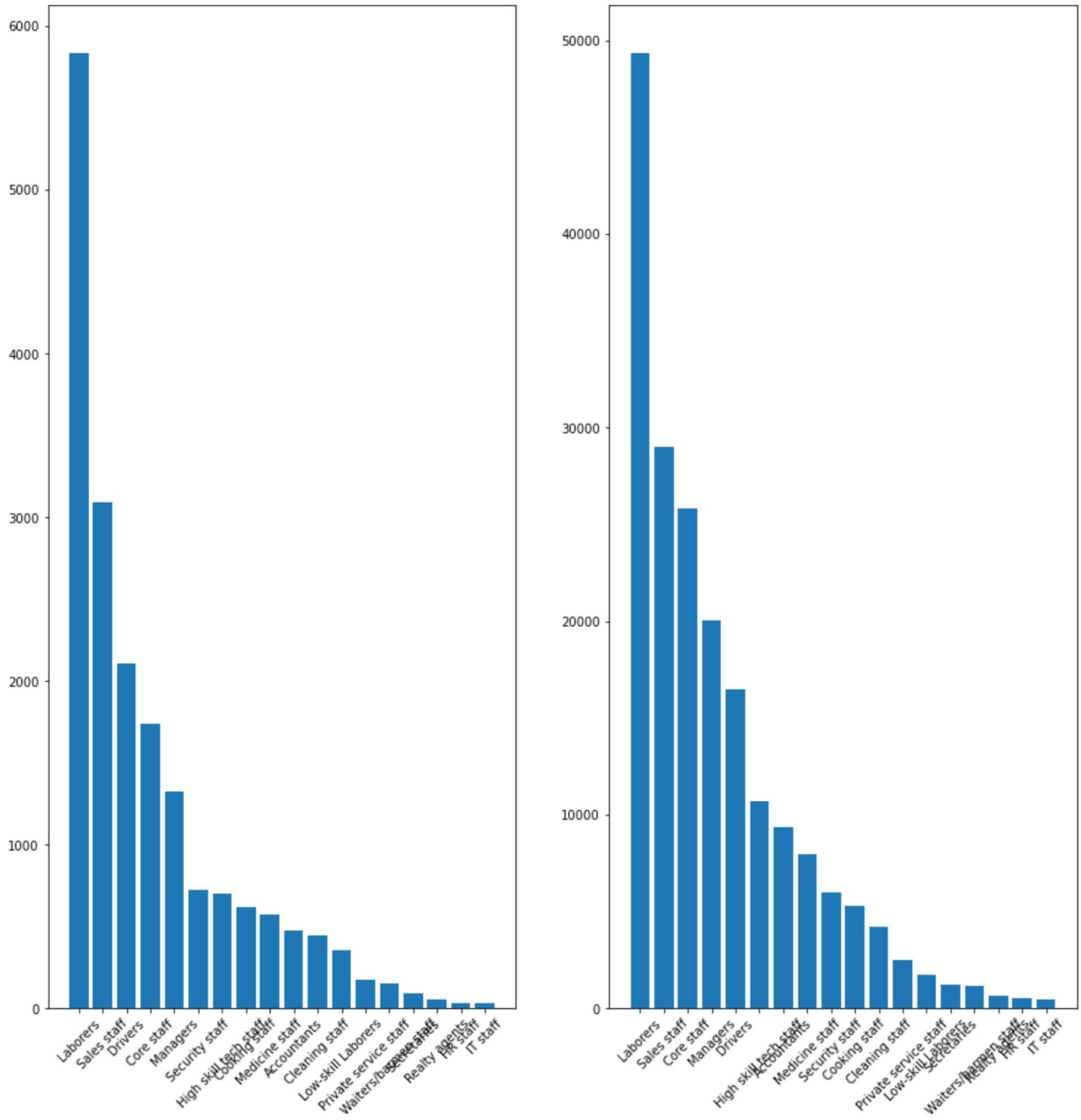

20000
17500
15000
12500
10000
7500
5000
2500
0
House / apartment
With parents
Municipal apartment
Rented apartment
Office apartment
Co-op apartment


250000
200000
150000
100000
50000
0
House / apartment
With parents
Municipal apartment
Rented apartment
Office apartment
Co-op apartment

In [16]:
```
# PEOPLE LIVING IN HOUSE/APARTMENT APPLY MORE LOANS WHEN COMPARED TO OTHER HOUSIN
# DEFAULTERS RATE IS HIGHER THAN NON-DEFAULTERS FOR PEOPLE WHO LIVE WITH THEIR PA
```

In [17]:
```
occupation1=default["OCCUPATION_TYPE"].value_counts(ascending=False)
occupation2=non_default["OCCUPATION_TYPE"].value_counts(ascending=False)

# fig,ax=plt.subplots(1,2)
#sns.countplot(suite1,data=default)
plt.figure(figsize=(15,15))
plt.subplot(1,2,1)

plt.bar(occupation1.keys(),occupation1.values)
plt.xticks(rotation=45)
plt.subplot(1,2,2)
plt.xticks(rotation=45)
plt.bar(occupation2.keys(),occupation2.values)
plt.figure(figsize=(15,15))
```

Out[17]:
```
<Figure size 1080x1080 with 0 Axes>
```

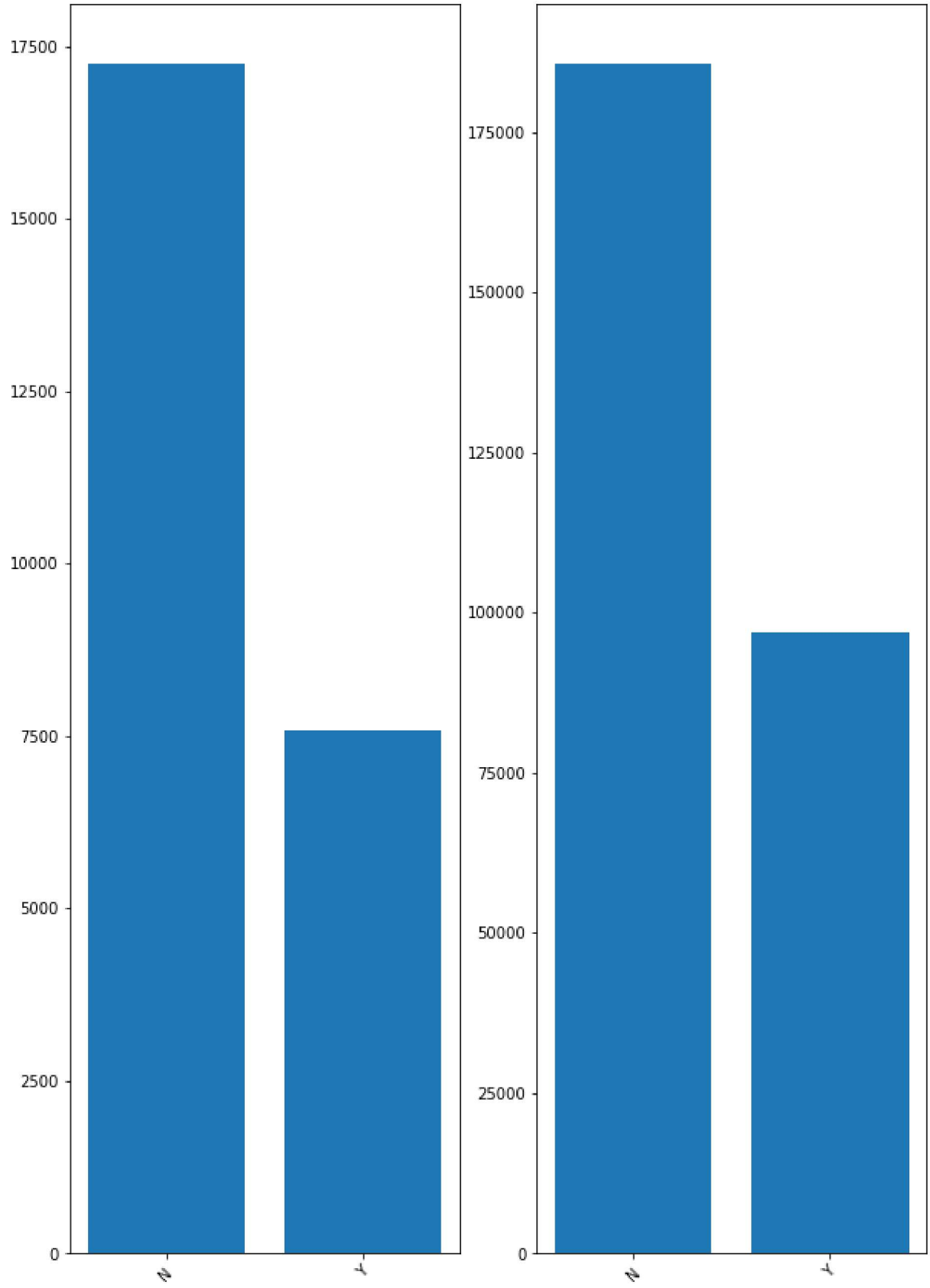


```
<Figure size 1080x1080 with 0 Axes>
```

In [18]:
```
# LABORERES APPLY MORE LOAN THAN OTHER PROFESSIONS.
# IT STAFF TENDS TO APPLY MINIMAL LOAN THAN OTHERS.
```

In [19]:
```
car1=default["FLAG_OWN_CAR"].value_counts(ascending=False)
car2=non_default["FLAG_OWN_CAR"].value_counts(ascending=False)

fig,(ax1,ax2)=plt.subplots(1,2,figsize=(10,15))
#sns.countplot(suite1,data=default)

plt.subplot(1,2,1)
plt.bar(car1.keys(),car1.values)
plt.xticks(rotation=45)
plt.subplot(1,2,2)
plt.xticks(rotation=45)
plt.bar(car2.keys(),car2.values)
```

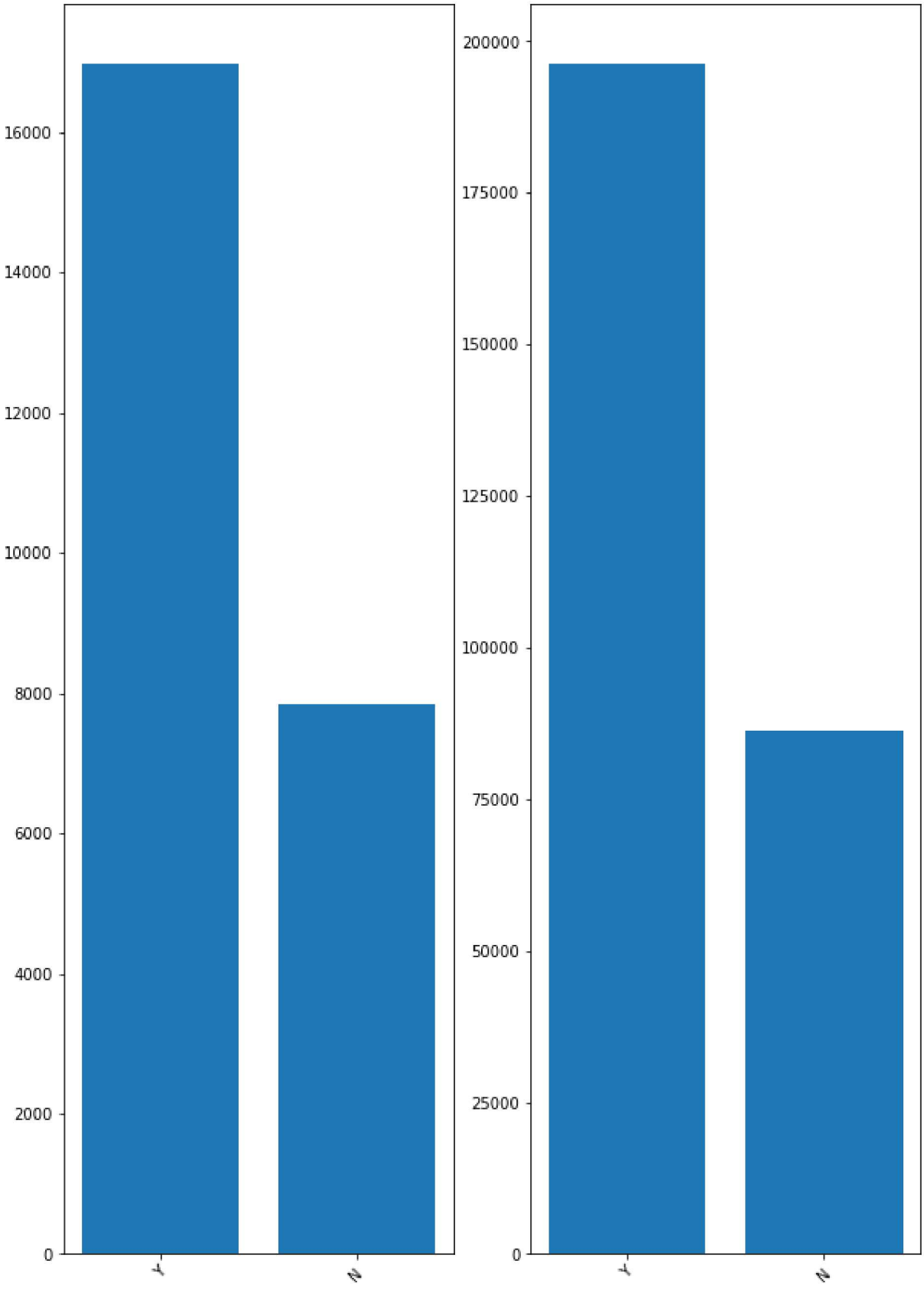
Out[19]:
```
<BarContainer object of 2 artists>
```



MORE THAN 70% OF PEOPLE WHO HAVE CARS ARE DEFUALTERS AND NEARLY 60 % OF PEOPLE ARE NON-DEFAULTERS.

In [20]:
```
realty1=default["FLAG_OWN_REALTY"].value_counts(ascending=False)
realty2=non_default["FLAG_OWN_REALTY"].value_counts(ascending=False)

fig,(ax1,ax2)=plt.subplots(1,2,figsize=(10,15))
#sns.countplot(suite1,data=default)

plt.subplot(1,2,1)
plt.bar(realty1.keys(),realty1.values)
plt.xticks(rotation=45)
plt.subplot(1,2,2)
plt.xticks(rotation=45)
plt.bar(realty2.keys(),realty2.values)
```

Out[20]:
```
<BarContainer object of 2 artists>
```



GENERALLY PEOPLE WHO HAVE HOUSE/APARTMENT TEND TO APPLY FOR LOANS.

In [21]:
```
default["AGE"]=default["DAYS_BIRTH"].abs()//365.25
non_default["AGE"]=non_default["DAYS_BIRTH"].abs()//365.25
default["AGE"]
# CALCULATION OF AGE USING NUMBER OF DAYS_BIRTH
```

```
<ipython-input-21-44657ae7db9e>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  default["AGE"]=default["DAYS_BIRTH"].abs()//365.25
<ipython-input-21-44657ae7db9e>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  non_default["AGE"]=non_default["DAYS_BIRTH"].abs()//365.25
```

Out[21]:
```
0         25
26        51
40        47
42        36
81        67
          ..
307448    27
307475    36
307481    56
307489    45
307509    32
Name: AGE, Length: 24825, dtype: int64
```

In [ ]:
```
```

In [22]:
```
default["AGE"].describe()
```

Out[22]:
```
count    24825.000000
mean        40.252931
std         11.474598
min         21.000000
25%         31.000000
50%         39.000000
75%         49.000000
max         68.000000
Name: AGE, dtype: float64
```

In [23]:
```
default["AGE_bins"]=pd.cut(default['AGE'],bins=np.arange(20,71,5))
non_default["AGE_bins"]=pd.cut(non_default['AGE'],bins=np.arange(20,71,5))
default["AGE_bins"]
```

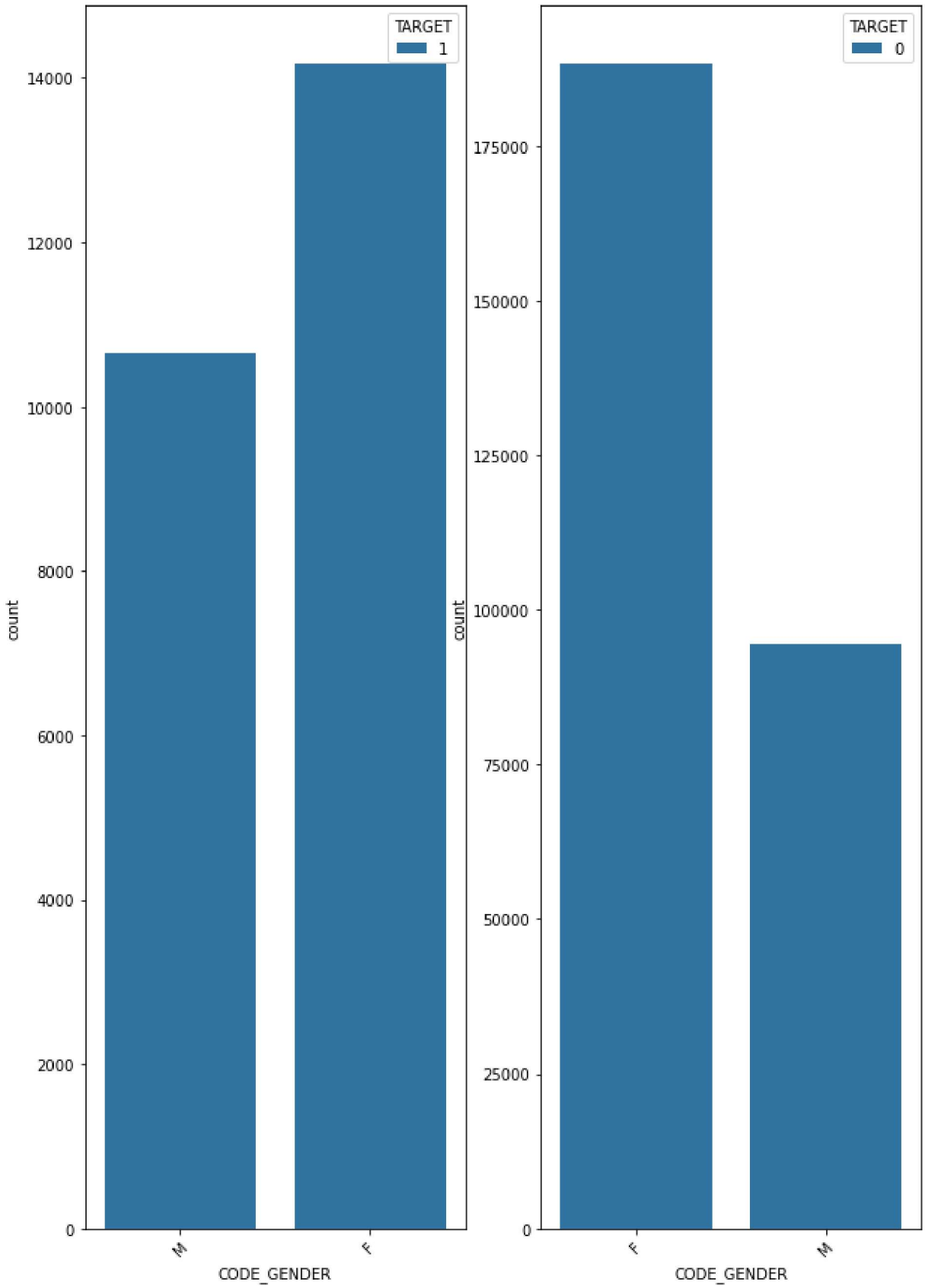
```
<ipython-input-23-23572d9d4f0e>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  default["AGE_bins"]=pd.cut(default['AGE'],bins=np.arange(20,71,5))
<ipython-input-23-23572d9d4f0e>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  non_default["AGE_bins"]=pd.cut(non_default['AGE'],bins=np.arange(20,71,5))
```

Out[23]:
```
0         (20, 25]
26        (50, 55]
40        (45, 50]
42        (35, 40]
81        (65, 70]
             ...
307448    (25, 30]
307475    (35, 40]
307481    (55, 60]
307489    (40, 45]
307509    (30, 35]
Name: AGE_bins, Length: 24825, dtype: category
Categories (10, interval[int64]): [(20, 25] < (25, 30] < (30, 35] < (35, 40]
... (50, 55] < (55, 60] < (60, 65] < (65, 70]]
```

In [24]:
```
temp_default=default[default["CODE_GENDER"]!="XNA"]
temp_non_default=non_default[non_default["CODE_GENDER"]!="XNA"]
gender=temp_default["CODE_GENDER"]
gender1=temp_non_default["CODE_GENDER"]
# drop_rows=gender[g["CODE_GENDER"]=="XNA"]
fig,ax=plt.subplots(1,2,figsize=(10,15))
plt.subplot(1,2,1)
plt.xticks(rotation=45)
sns.countplot(gender,data=default,hue="TARGET")
plt.subplot(1,2,2)
plt.xticks(rotation=45)
sns.countplot(gender1,data=non_default,hue="TARGET")
```

```
C:\Users\P.Vijay Srinivasan\anaconda3\lib\site-packages\seaborn\_decorators.py:
36: FutureWarning: Pass the following variable as a keyword arg: x. From versio
n 0.12, the only valid positional argument will be `data`, and passing other ar
guments without an explicit keyword will result in an error or misinterpretatio
n.
  warnings.warn(
C:\Users\P.Vijay Srinivasan\anaconda3\lib\site-packages\seaborn\_decorators.py:
36: FutureWarning: Pass the following variable as a keyword arg: x. From versio
n 0.12, the only valid positional argument will be `data`, and passing other ar
guments without an explicit keyword will result in an error or misinterpretatio
n.
  warnings.warn(
```

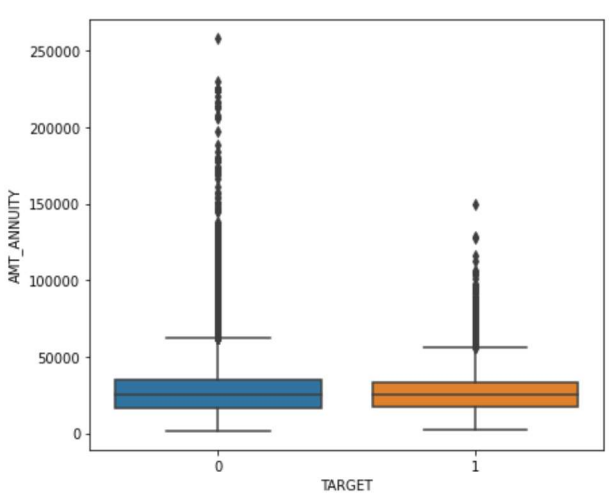
Out[24]: <AxesSubplot:xlabel='CODE_GENDER', ylabel='count'>

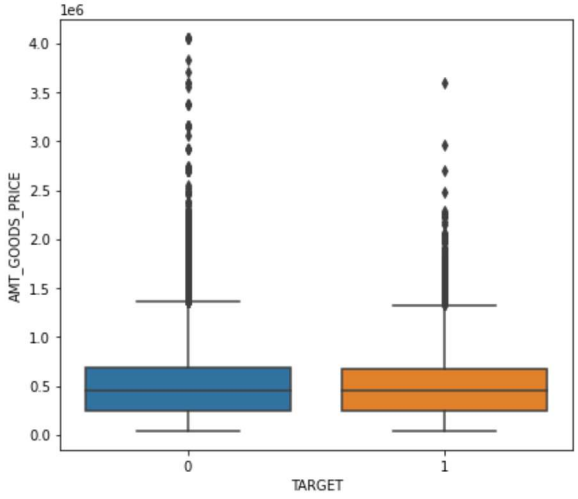


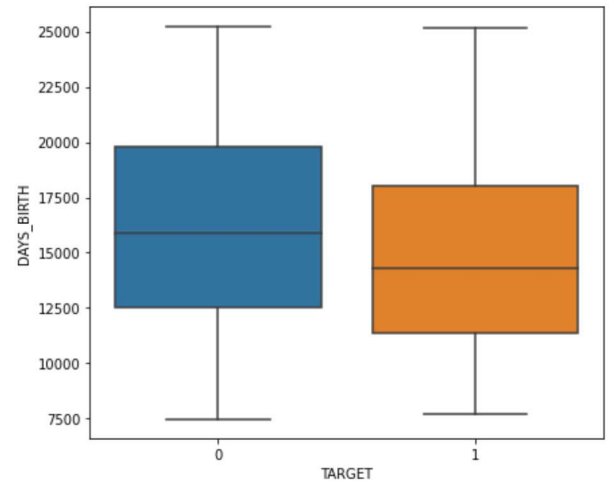
Females tend to apply more loans than males

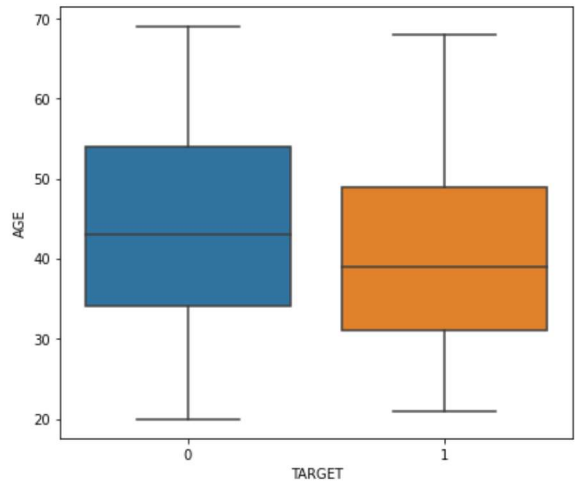
In [25]:
```
apply['DAYS_BIRTH'] = apply['DAYS_BIRTH'].abs()
apply['DAYS_EMPLOYED'] =apply['DAYS_EMPLOYED'].abs()
apply["AGE"]=apply["DAYS_BIRTH"].abs()//365.25
apply["AGE_bins"]=pd.cut(apply['AGE'],bins=np.arange(20,71,5))
```

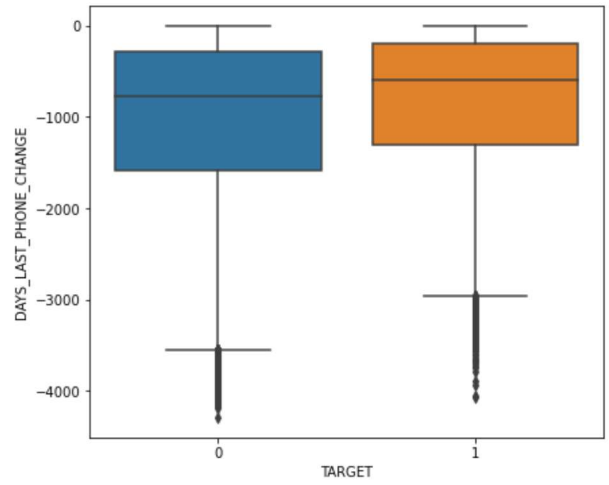
In [26]:
```
features = ['AMT_ANNUITY','AMT_GOODS_PRICE','DAYS_BIRTH','AGE','DAYS_LAST_PHONE_C
plt.figure(figsize = (15, 20))

for i in enumerate(features):
    plt.subplot(3, 2, i[0]+1)
#     plt.subplots_adjust(hspace=0.5)
    sns.boxplot(x = 'TARGET', y = i[1], data =apply)
```

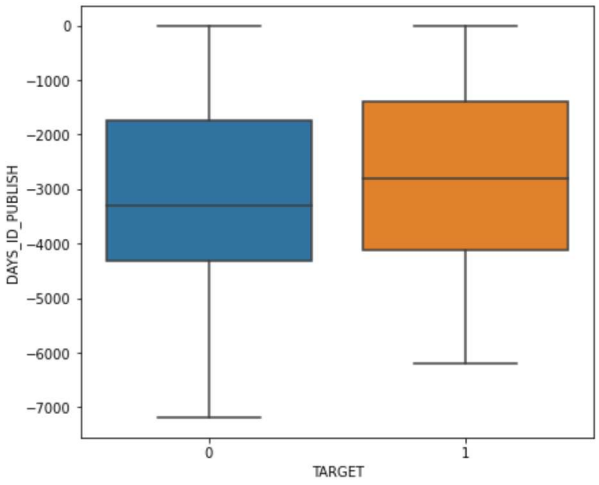












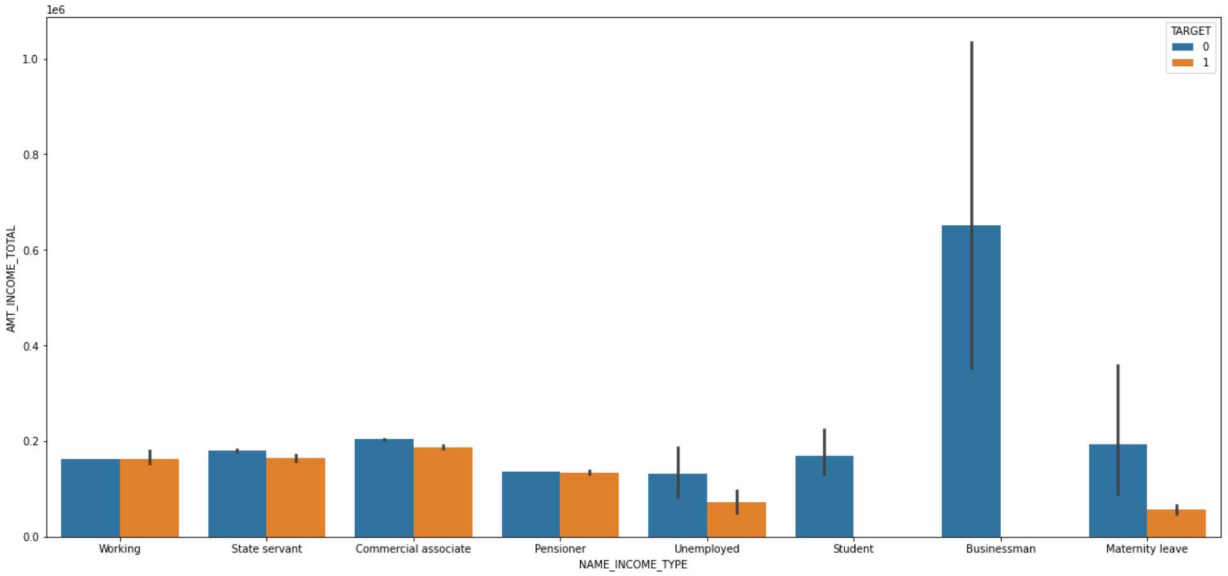
The people from age between 25 and 45 are having higher probability of repayment

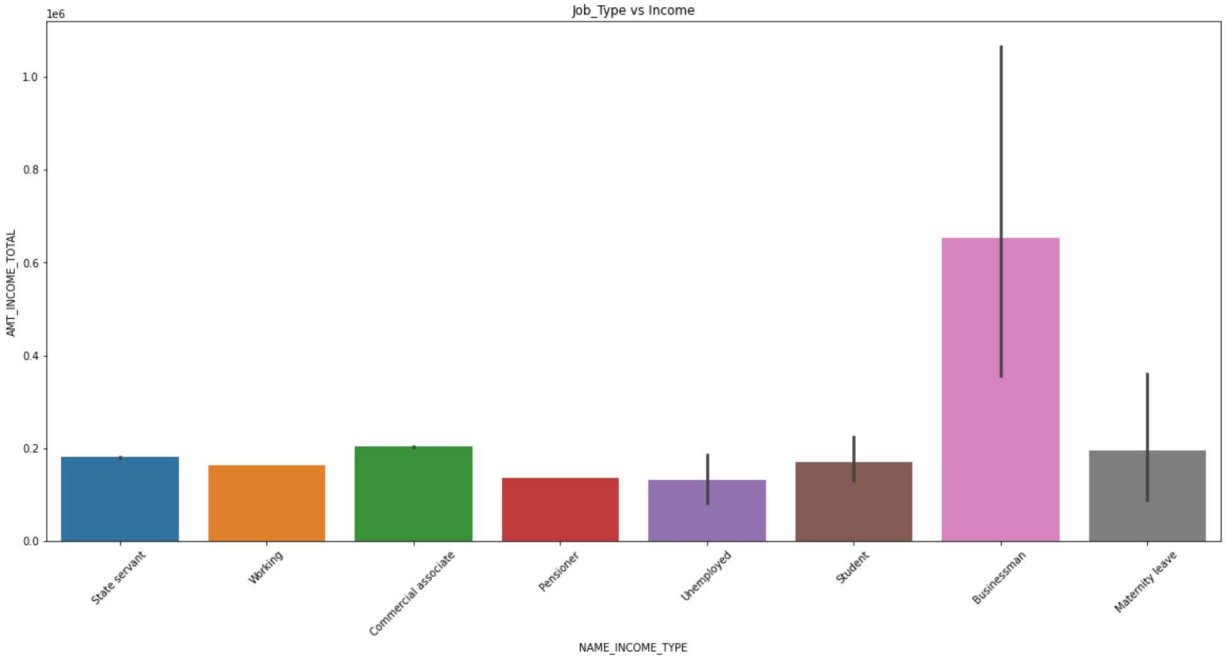
Less outliers is observed in Days_Birth and days_id_publish

Some outliers are observed in In 'AMT_ANNUITY','AMT_GOODS_PRICE','DAYS_EMI DAYS_LAST_PHONE_CHANGE in the dataset

people changing ID in recent days are prone to be default

In [27]:
```
fig,ax=plt.subplots(2,2,figsize=(20,20))
plt.yscale('log')
plt.subplot(2,1,1)
# sns.barplot(data=default, x='NAME_INCOME_TYPE',y='AMT_INCOME_TOTAL',orient='v')
sns.barplot(data=apply, x='NAME_INCOME_TYPE',y='AMT_INCOME_TOTAL',hue="TARGET",or

plt.subplot(2,1,2)
sns.barplot(data=non_default, x='NAME_INCOME_TYPE',y='AMT_INCOME_TOTAL',orient='v
plt.title('Job_Type vs Income')
plt.xticks(rotation=45)
plt.show()
```



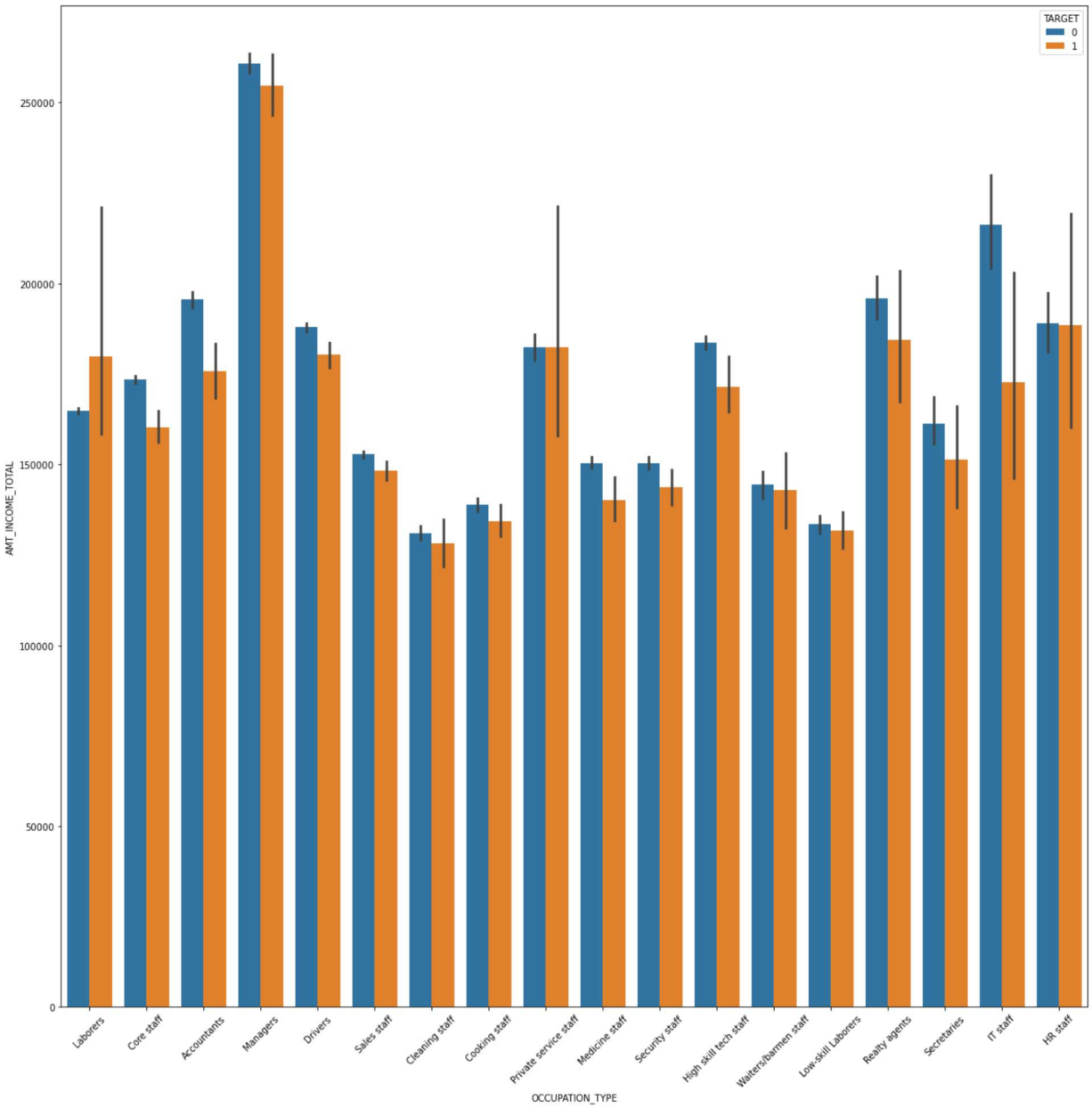


As we can see that business people have higher incomes than others and also they tend to apply for more loans and also repay it.

In [28]:
```
print("OCCUPATION TYPE vs ANNUAL INCOME")
plt.figure(figsize=(20,20))
plt.yscale('linear')
plt.xticks(rotation=45)
sns.barplot(data=apply, x='OCCUPATION_TYPE',y='AMT_INCOME_TOTAL',hue="TARGET",ori
# REFERENCE FOR GRAPH COLORS:0-MEANS DEFAULTERS AND 1 MEANS NON-DEFAULTERS
```

```
OCCUPATION TYPE vs ANNUAL INCOME
```

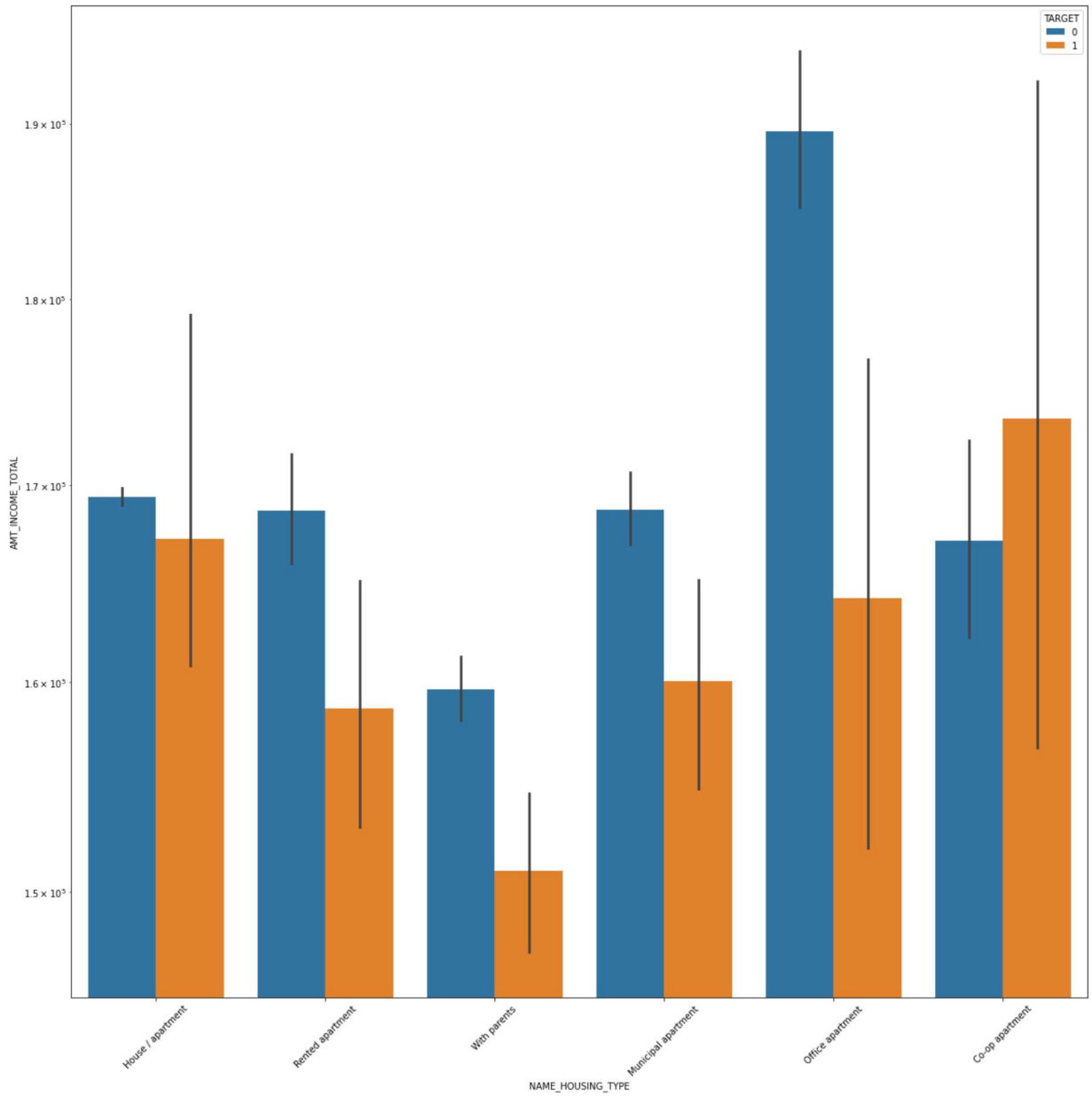
Out[28]:
```
<AxesSubplot:xlabel='OCCUPATION_TYPE', ylabel='AMT_INCOME_TOTAL'>
```



Managers recieve higher income per annum and also they tend to default more than other peple.

Cleaning staff recieve lowest among others and they also tend to default but less compared to others.

In [29]:
```
# fig,ax=plt.subplots(2,2,figsize=(20,20))
plt.figure(figsize=(20,20))
plt.xticks(rotation=45)
plt.yscale('log')
# plt.subplot(2,1,1)
sns.barplot(data=apply, x='NAME_HOUSING_TYPE',y='AMT_INCOME_TOTAL',hue="TARGET",c
plt.show()
# REFERENCE FOR GRAPH COLORS:0-MEANS DEFAULTERS AND 1 MEANS NON-DEFAULTERS
```



People living in office apartment who don't have any payment difficulties tend to recieve more

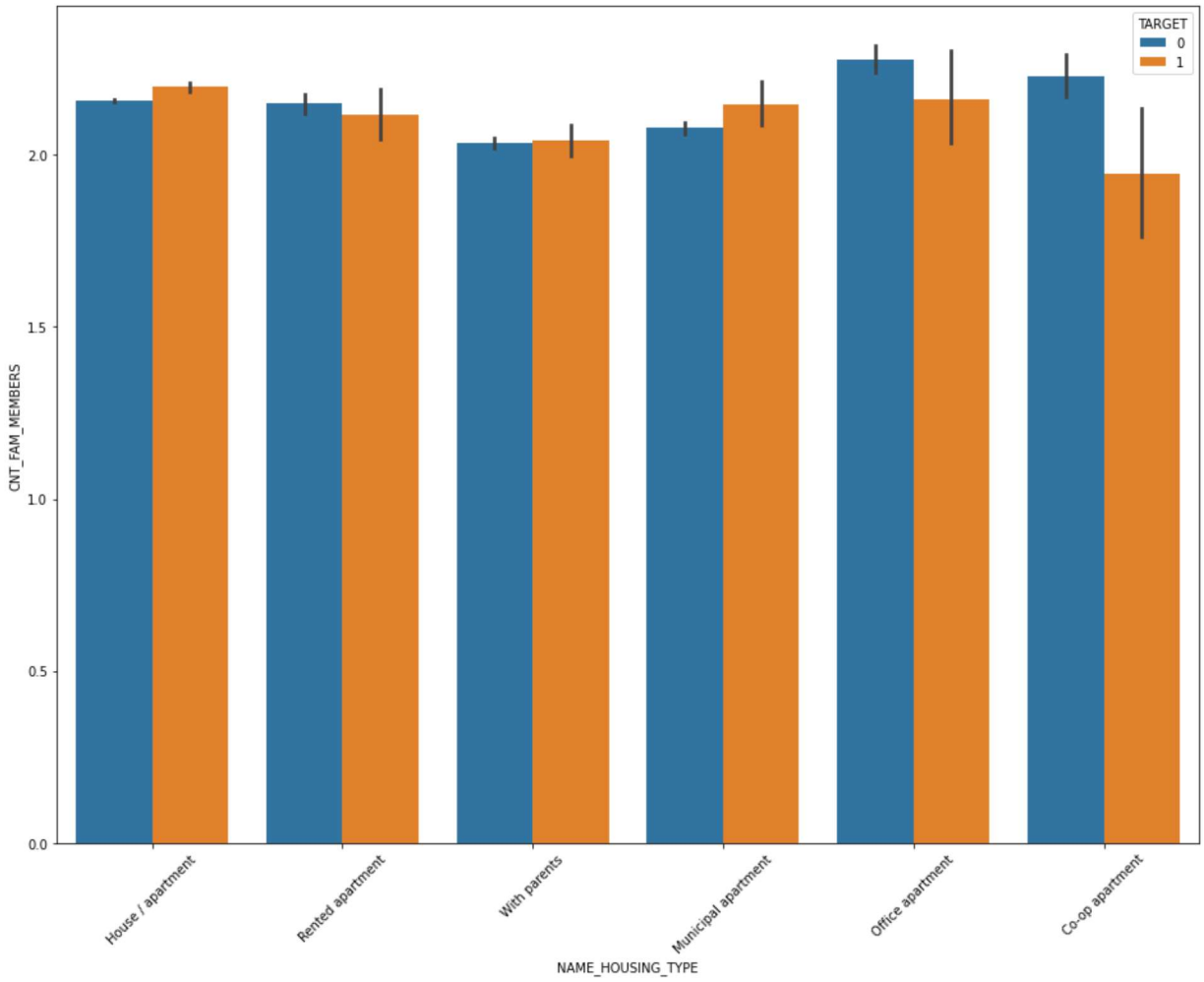
income so we can infer

that as they recieve good income they can repay the loan within due date and same with people who live in rented and municipal

apartments

People living in co-op apartment have more payement difficulties and also they recieve low income.

In [30]:
```
plt.figure(figsize=(16,12))
plt.xticks(rotation=45)
# plt.ylim(1,16)
# plt.yscale('linear')
sns.barplot(data=apply, x='NAME_HOUSING_TYPE',y='CNT_FAM_MEMBERS',hue="TARGET",or
# plt.title('Income amount vs Education Status')
plt.show()
# REFERENCE FOR GRAPH COLORS:0-MEANS DEFAULTERS AND 1 MEANS NON-DEFAULTERS
```



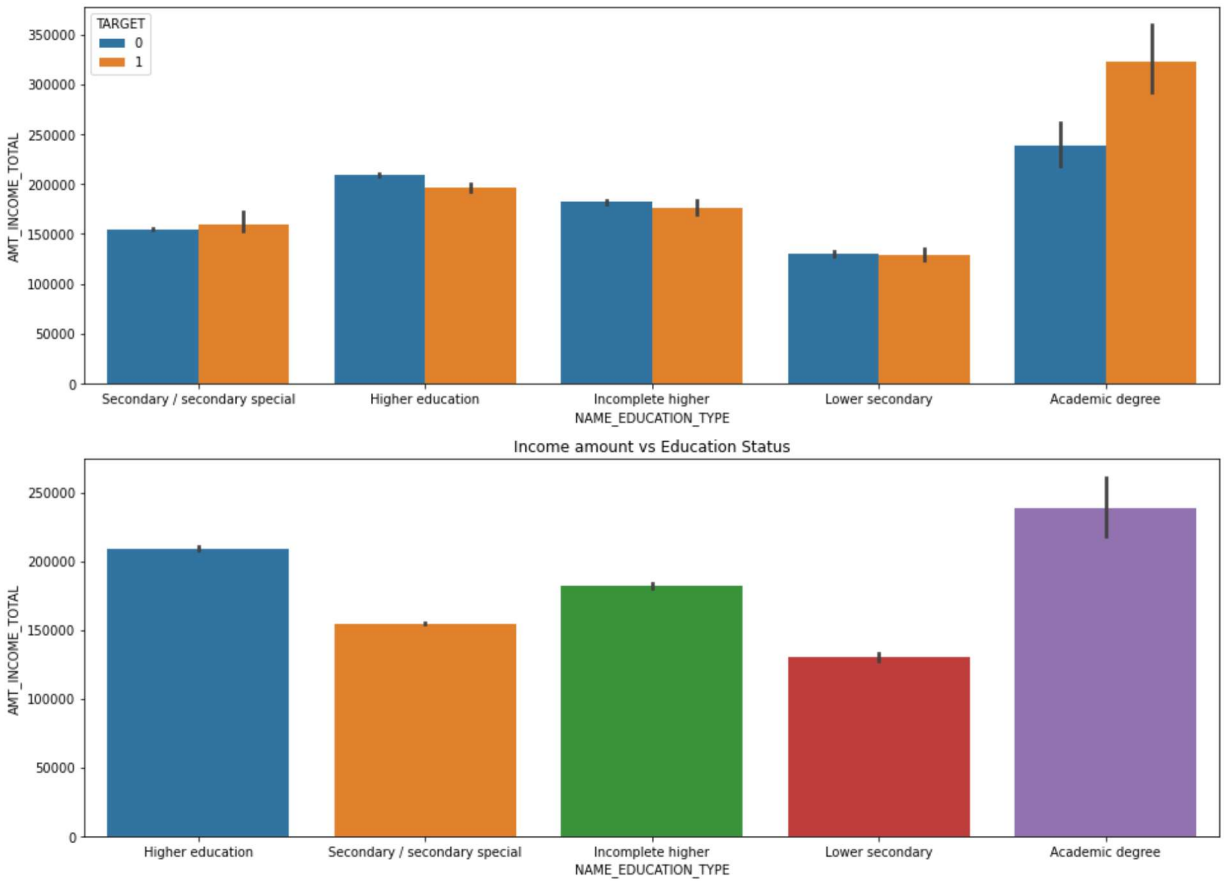
We can see that most of the applicants have more than 2 family members

As from the befoore graph we can conclude that most people who live in office apartments and rented apartments tend to be non_defaulters and have more than 2 family members

and people who live in co-op apartment have more payment difficulties and those people have <=2 family members.

Those who have their own house/apartment also apply loans but most of them tend to be defaulters

In [31]:
```
# plt.figure(figsize=(16,12))
# plt.xticks(rotation=90)
fig,ax=plt.subplots(1,2,figsize=(16,12))
plt.yscale('log')
plt.subplot(2,1,1)
sns.barplot(data=apply, x='NAME_EDUCATION_TYPE',y='AMT_INCOME_TOTAL',hue="TARGET"
plt.subplot(2,1,2)
sns.barplot(data=non_default, x='NAME_EDUCATION_TYPE',y='AMT_INCOME_TOTAL',orient
plt.title('Income amount vs Education Status')
plt.show()
```

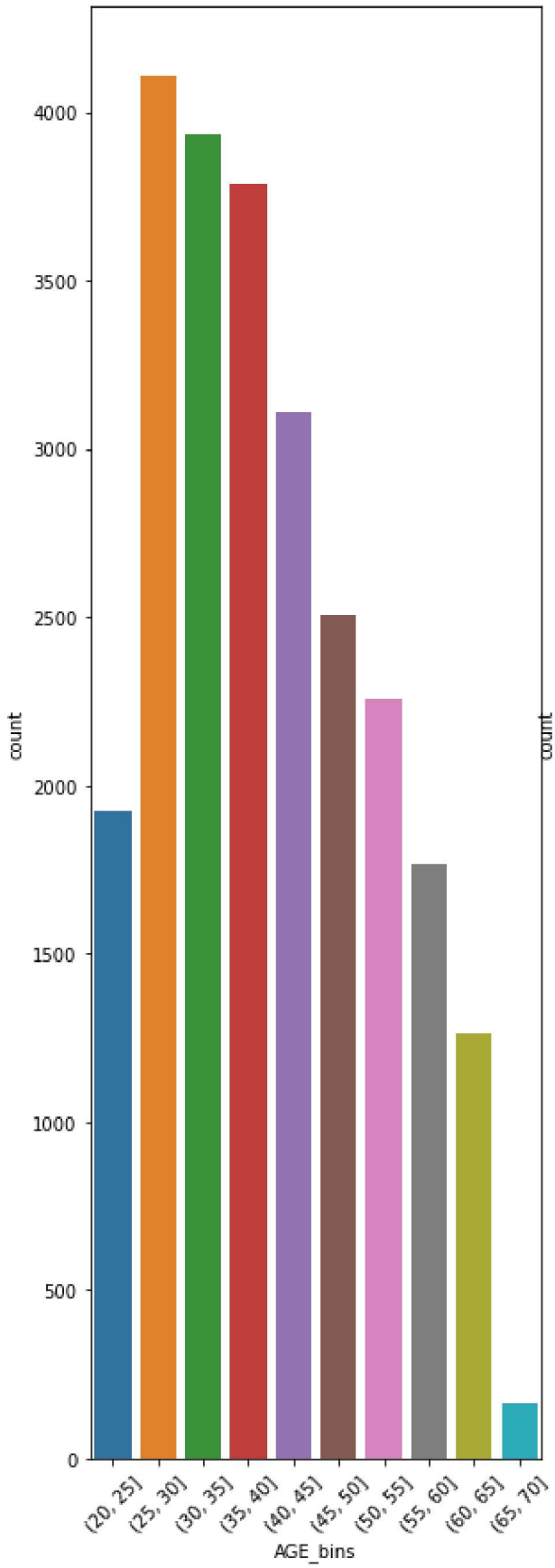


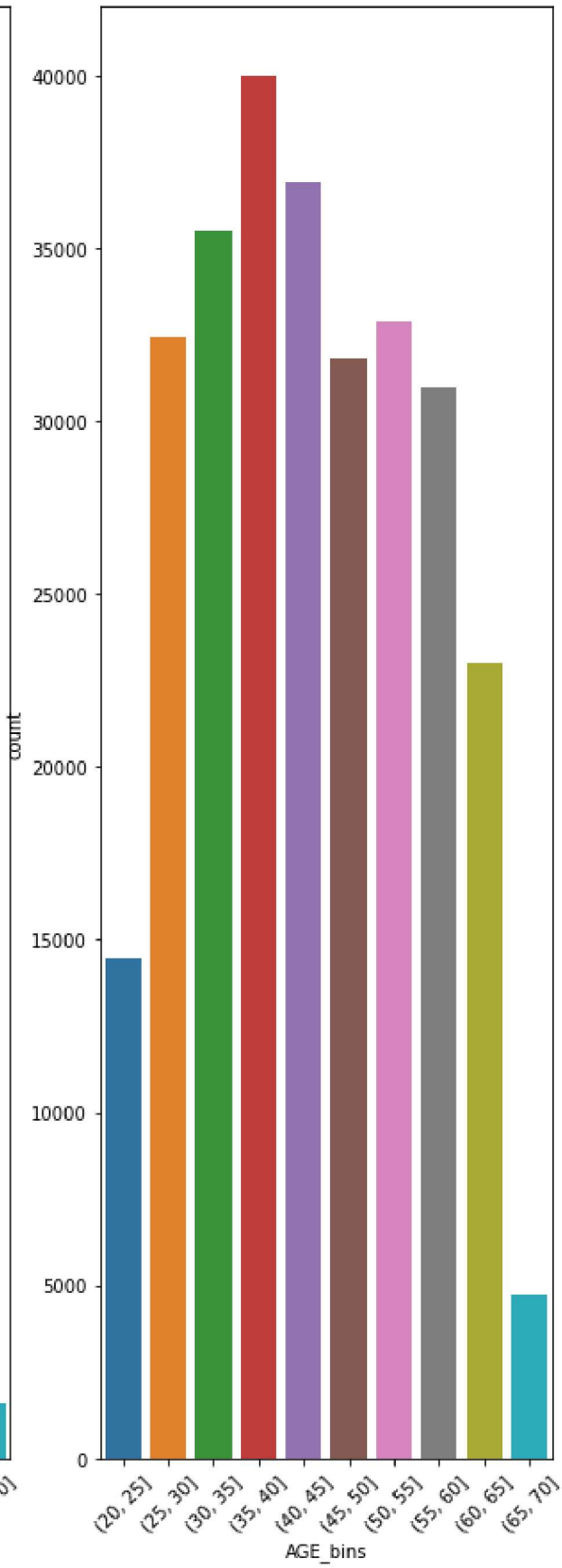
In [32]:
```
# From the previous count plot about name_education_type,most people who have pay
# and their annual money is less than others.
# In both cases,people who have completed their academic degree tend to have more
# Most of the People who complete their higher education don't have payment diffi
```

In [33]:
```
age=default["AGE_bins"]
age1=non_default["AGE_bins"]
fig,ax=plt.subplots(1,2,figsize=(10,15))
plt.subplot(1,2,1)
plt.xticks(rotation=45)
sns.countplot(age,data=default)
plt.subplot(1,2,2)
plt.xticks(rotation=45)
sns.countplot(age1,data=non_default)
```

```
C:\Users\P.Vijay Srinivasan\anaconda3\lib\site-packages\seaborn\_decorators.py:
36: FutureWarning: Pass the following variable as a keyword arg: x. From versio
n 0.12, the only valid positional argument will be `data`, and passing other ar
guments without an explicit keyword will result in an error or misinterpretatio
n.
  warnings.warn(
C:\Users\P.Vijay Srinivasan\anaconda3\lib\site-packages\seaborn\_decorators.py:
36: FutureWarning: Pass the following variable as a keyword arg: x. From versio
n 0.12, the only valid positional argument will be `data`, and passing other ar
guments without an explicit keyword will result in an error or misinterpretatio
n.
  warnings.warn(
```

Out[33]:
```
<AxesSubplot:xlabel='AGE_bins', ylabel='count'>
```

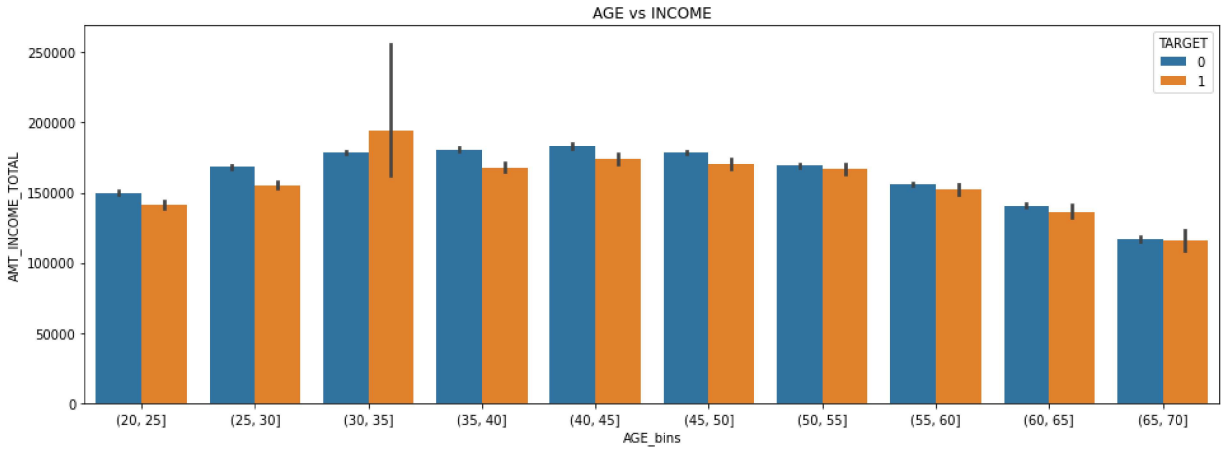




AGE_GROUPS FROM 25-45 TEND TO APPLY MORE LOANS.

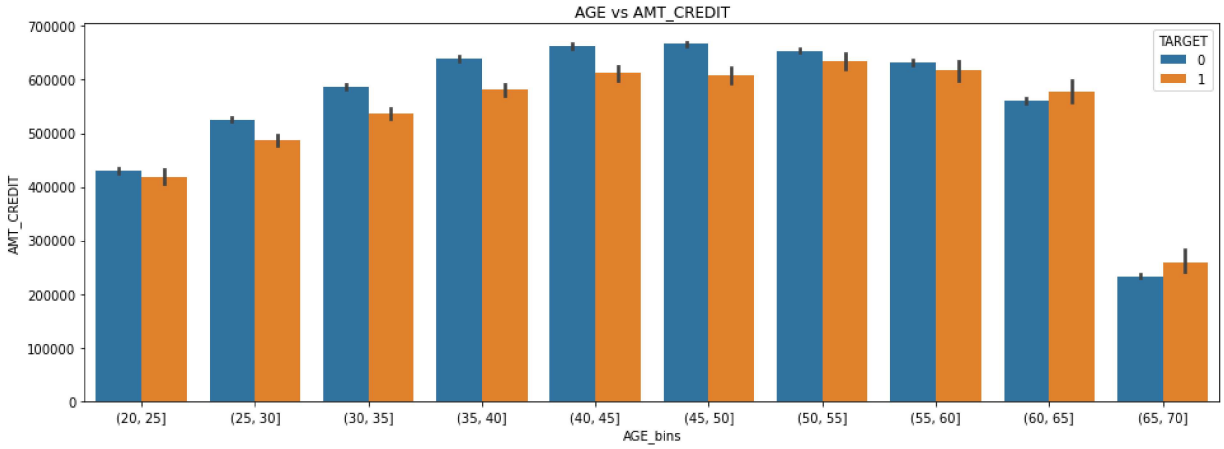
AMONG THEM, PEOPLE OF AGE 25-30 ARE MORE DEFAULT WHEREAS 35-45 ARE THE PEOPLE WHO ARE LIKELY TO BE LESS DEFAULT

In [34]:
```
apply["AGE"]=apply["DAYS_BIRTH"].abs()//365.25
apply["AGE_bins"]=pd.cut(apply['AGE'],bins=np.arange(20,71,5))
plt.figure(figsize=(16,12))
plt.yscale('log')
plt.subplot(2,1,1)
sns.barplot(data=apply, x='AGE_bins',y='AMT_INCOME_TOTAL', hue ='TARGET',orient='
plt.title('AGE vs INCOME')
plt.show()
```



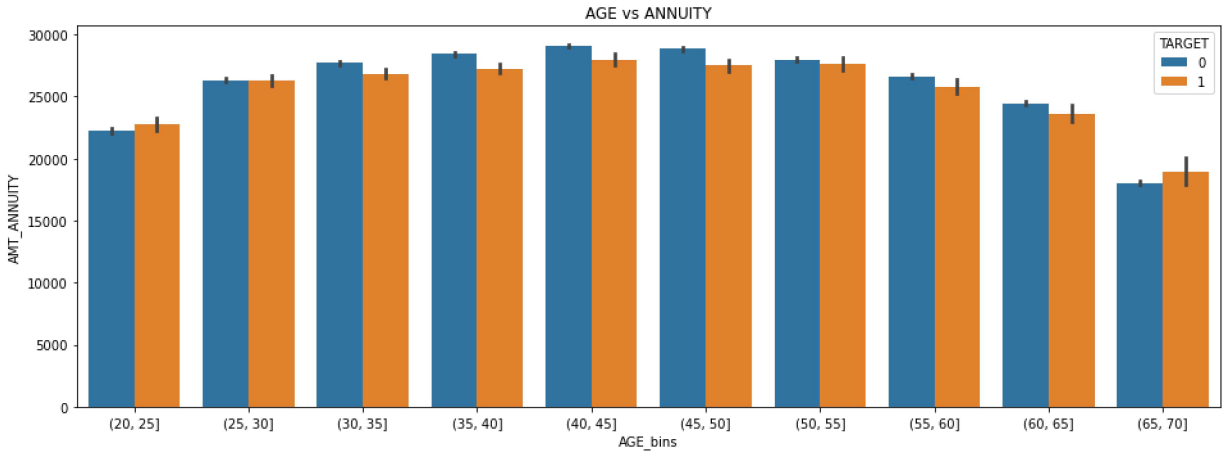
From the above graph we can see that most of the people of age between 30-35 earns more than others and surprisingly some of them tend to be defaulters.

In [35]:
```
apply["AGE"]=apply["DAYS_BIRTH"].abs()//365.25
apply["AGE_bins"]=pd.cut(apply['AGE'],bins=np.arange(20,71,5))
plt.figure(figsize=(16,12))
plt.yscale('log')
plt.subplot(2,1,1)
sns.barplot(data=apply, x='AGE_bins',y='AMT_CREDIT', hue ='TARGET',orient='v')
plt.title('AGE vs AMT_CREDIT')
plt.show()
```



From the above graph we can see that People of age 40-60 tend to apply AMT_Credit to reduce their income tax bill

In [36]:
```
apply["AGE"]=apply["DAYS_BIRTH"].abs()//365.25
apply["AGE_bins"]=pd.cut(apply['AGE'],bins=np.arange(20,71,5))
plt.figure(figsize=(16,12))
plt.yscale('log')
plt.subplot(2,1,1)
sns.barplot(data=apply, x='AGE_bins',y='AMT_ANNUITY', hue ='TARGET',orient='v')
plt.title('AGE vs ANNUITY')
plt.show()
```

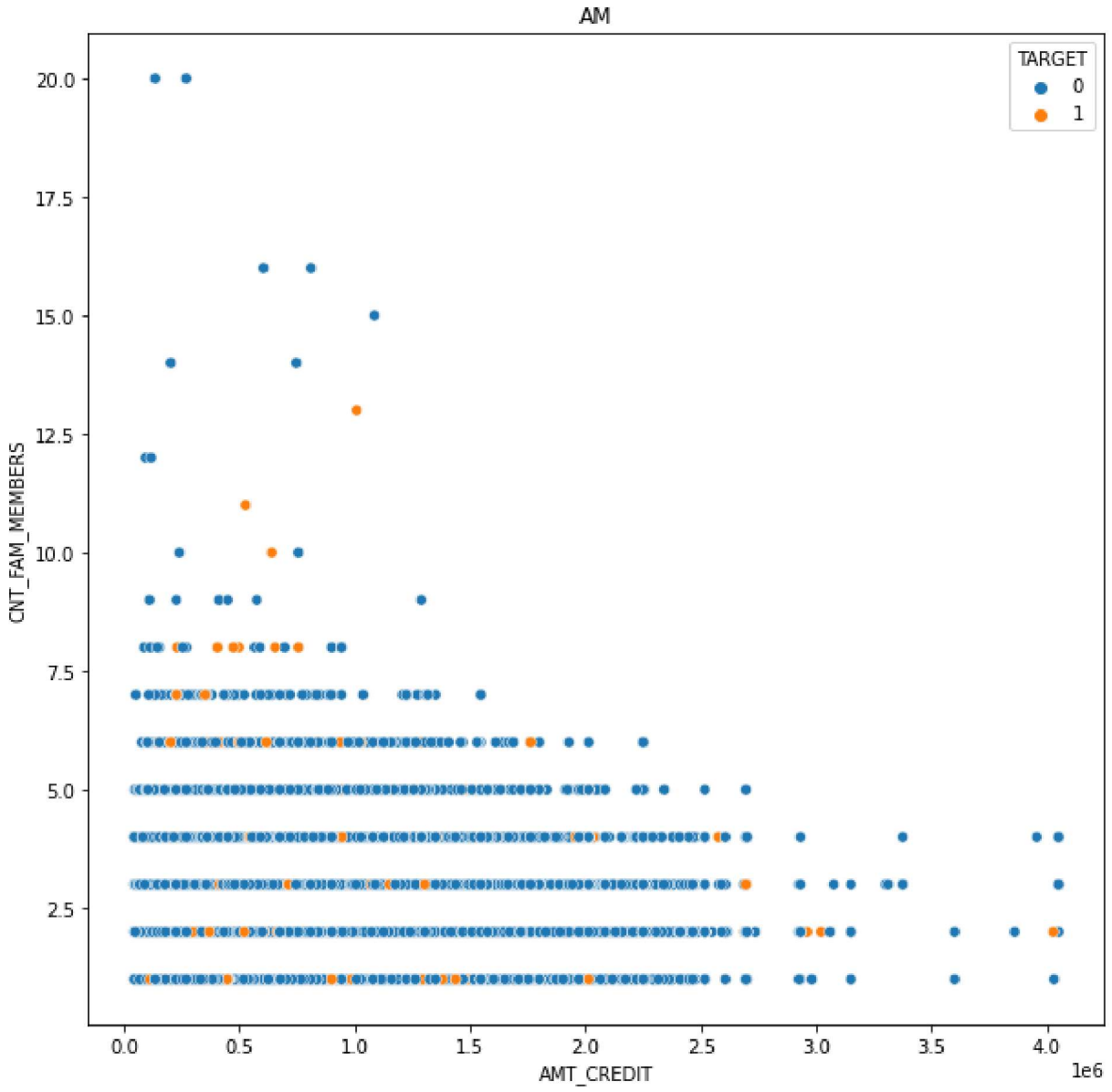


From the graph we can see that people of age 40-45 are having higher rate of annuity.

Old people of age 65-70 have low annuity rate.

In [41]:
```
# fig,(ax1,ax2,ax3) = plt.subplots(1,3,figsize=(20,6))
plt.figure(figsize=(10,10))
sns.scatterplot(x='AMT_CREDIT', y='CNT_FAM_MEMBERS',hue="TARGET",data=apply)
# sns.scatterplot(x='AMT_CREDIT', y='CNT_FAM_MEMBERS',hue="TARGET",data=default,
# sns.scatterplot(x='AMT_CREDIT', y='CNT_FAM_MEMBERS',hue="TARGET",data=non_defau
plt.title("AM")
ax1.set_xlabel('AMT_CREDIT')
ax1.set_ylabel('CNT_FAM_MEMBERS')
```

Out[41]:
```
Text(18.106250000000017, 0.5, 'CNT_FAM_MEMBERS')
```



if the family is small and amt-credit is low,we can assume that they are likely to be more default.

CONCLUSION: 1)Unaccompanied families are the one who apply more loans . Non defaulters are more than defaulters.Defaulters are more among people who are single

2)Married people usually apply more loans.

3)Student pursuing higher education tend to apply more loans and most of them also are non defaulters.

4)People living in house/apartment apply more loans than other people in different housing types

5)Among many number of occupations,laboreres tend to apply more loans.

6)Most of the people who have cars tend to be default more.

7)People who are of age 25-45 tend to apply more loans.Among them,people who are of age 25-30 tend to default more and people of age 30-45 are likely to be less defualt.

8)We can see that most of the people of age between 30-35recieves higher income than other age groups and surprisingly some of them tend to be defaulters.

RECOMMENDATION: 1)Bank should avoid giving loans to students who pursue secondary education as they are having difficulties in payment.

2)Banks can concentrate on giving loans to people living in house/apartment than other housing types because most of them tend to replay the loans(non deafulters)

3)Banks should avoid giving loans to people who live in co-op aparments as they have payment diffculties due to their less income.

4)Banks can focus giving loans to people who live in own house,office apartments as they have higher income and they have less default record.

5)Banks should avoid giving loans to people of age 25-30 as they have less income than people of age 31-40.

6)So banks can provide loans to people of age those who are above 30 as they recieve higher income with a well-experienced job so there are high chances that they can repay the loan without any payment difficulties.