Resume Project 1 - EDA Bank Loan Default Risk Analysis

In [1]:
```
# Import Python Libraries

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

In [2]:
```
# Supress Warnings

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

In [3]:
```
import os
os.getcwd() # if you want to change the working directory
```

Out[3]:
```
'C:\\Users\\Sai Chetan'
```

In [11]:
```
# Read CSV File Applicaton:

appDF = pd.read_csv(r'D:\00 - Naresh IT\1 - Materials\Jypter\14 - Kaggle Resume Projec
appDF.head()

# D:\00 - Naresh IT\1 - Materials\Jypter\14 - Kaggle Resume Project Bank Loan Risk Man
```

Out[11]:

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

5 rows × 122 columns

In [12]:
```
# Read CSV Files Previous:

prevDF = pd.read_csv(r'D:\00 - Naresh IT\1 - Materials\Jypter\14 - Kaggle Resume Proje
prevDF.head()

# D:\00 - Naresh IT\1 - Materials\Jypter\14 - Kaggle Resume Project Bank Loan Risk Man
```

Out[12]:

	SK_ID_PREV	SK_ID_CURR	NAME_CONTRACT_TYPE	AMT_ANNUITY	AMT_APPLICATION	AMT_CREDI
0	2030495	271877	Consumer loans	1730.430	17145.0	17145
1	2802425	108129	Cash loans	25188.615	607500.0	679671
2	2523466	122040	Cash loans	15060.735	112500.0	136444
3	2819243	176158	Cash loans	47041.335	450000.0	470790
4	1784265	202054	Cash loans	31924.395	337500.0	404055

5 rows × 37 columns

In [16]:
```
appDF.shape # Database dimension Application CSV
```

Out[16]:
```
(307511, 122)
```

In [15]:
```
prevDF.shape # Database dimension Previous CSV
```

Out[15]:
```
(1670214, 37)
```

In [20]:
```
appDF.size # Database size Application CSV
```

Out[20]:
```
37516342
```

In [18]:
```
prevDF.size # Database size Previous CSV
```

Out[18]:
```
61797918
```

In [22]:
```
# Database column types

appDF.info(verbose=True) # After using verbose only able to see columns info
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Data columns (total 122 columns):
 #    Column                          Dtype
---   ------                          -----
 0    SK_ID_CURR                      int64
 1    TARGET                          int64
 2    NAME_CONTRACT_TYPE              object
 3    CODE_GENDER                     object
 4    FLAG_OWN_CAR                    object
 5    FLAG_OWN_REALTY                 object
 6    CNT_CHILDREN                    int64
 7    AMT_INCOME_TOTAL                float64
 8    AMT_CREDIT                      float64
 9    AMT_ANNUITY                     float64
 10   AMT_GOODS_PRICE                 float64
 11   NAME_TYPE_SUITE                 object
 12   NAME_INCOME_TYPE                object
 13   NAME_EDUCATION_TYPE             object
 14   NAME_FAMILY_STATUS              object
 15   NAME_HOUSING_TYPE               object
 16   REGION_POPULATION_RELATIVE      float64
 17   DAYS_BIRTH                      int64
 18   DAYS_EMPLOYED                   int64
 19   DAYS_REGISTRATION               float64
 20   DAYS_ID_PUBLISH                 int64
 21   OWN_CAR_AGE                     float64
 22   FLAG_MOBIL                      int64
 23   FLAG_EMP_PHONE                  int64
 24   FLAG_WORK_PHONE                 int64
 25   FLAG_CONT_MOBILE                int64
 26   FLAG_PHONE                      int64
 27   FLAG_EMAIL                      int64
 28   OCCUPATION_TYPE                 object
 29   CNT_FAM_MEMBERS                 float64
 30   REGION_RATING_CLIENT            int64
 31   REGION_RATING_CLIENT_W_CITY     int64
 32   WEEKDAY_APPR_PROCESS_START      object
 33   HOUR_APPR_PROCESS_START         int64
 34   REG_REGION_NOT_LIVE_REGION      int64
 35   REG_REGION_NOT_WORK_REGION      int64
 36   LIVE_REGION_NOT_WORK_REGION     int64
 37   REG_CITY_NOT_LIVE_CITY          int64
 38   REG_CITY_NOT_WORK_CITY          int64
 39   LIVE_CITY_NOT_WORK_CITY         int64
 40   ORGANIZATION_TYPE               object
 41   EXT_SOURCE_1                    float64
 42   EXT_SOURCE_2                    float64
 43   EXT_SOURCE_3                    float64
 44   APARTMENTS_AVG                  float64
 45   BASEMENTAREA_AVG                float64
 46   YEARS_BEGINEXPLUATATION_AVG     float64
 47   YEARS_BUILD_AVG                 float64
 48   COMMONAREA_AVG                  float64
 49   ELEVATORS_AVG                   float64
 50   ENTRANCES_AVG                   float64
 51   FLOORSMAX_AVG                   float64
 52   FLOORSMIN_AVG                   float64
 53   LANDAREA_AVG                    float64
 54   LIVINGAPARTMENTS_AVG            float64
```

55	LIVINGAREA_AVG	float64
56	NONLIVINGAPARTMENTS_AVG	float64
57	NONLIVINGAREA_AVG	float64
58	APARTMENTS_MODE	float64
59	BASEMENTAREA_MODE	float64
60	YEARS_BEGINEXPLUATATION_MODE	float64
61	YEARS_BUILD_MODE	float64
62	COMMONAREA_MODE	float64
63	ELEVATORS_MODE	float64
64	ENTRANCES_MODE	float64
65	FLOORSMAX_MODE	float64
66	FLOORSMIN_MODE	float64
67	LANDAREA_MODE	float64
68	LIVINGAPARTMENTS_MODE	float64
69	LIVINGAREA_MODE	float64
70	NONLIVINGAPARTMENTS_MODE	float64
71	NONLIVINGAREA_MODE	float64
72	APARTMENTS_MEDI	float64
73	BASEMENTAREA_MEDI	float64
74	YEARS_BEGINEXPLUATATION_MEDI	float64
75	YEARS_BUILD_MEDI	float64
76	COMMONAREA_MEDI	float64
77	ELEVATORS_MEDI	float64
78	ENTRANCES_MEDI	float64
79	FLOORSMAX_MEDI	float64
80	FLOORSMIN_MEDI	float64
81	LANDAREA_MEDI	float64
82	LIVINGAPARTMENTS_MEDI	float64
83	LIVINGAREA_MEDI	float64
84	NONLIVINGAPARTMENTS_MEDI	float64
85	NONLIVINGAREA_MEDI	float64
86	FONDKAPREMONT_MODE	object
87	HOUSETYPE_MODE	object
88	TOTALAREA_MODE	float64
89	WALLSMATERIAL_MODE	object
90	EMERGENCYSTATE_MODE	object
91	OBS_30_CNT_SOCIAL_CIRCLE	float64
92	DEF_30_CNT_SOCIAL_CIRCLE	float64
93	OBS_60_CNT_SOCIAL_CIRCLE	float64
94	DEF_60_CNT_SOCIAL_CIRCLE	float64
95	DAYS_LAST_PHONE_CHANGE	float64
96	FLAG_DOCUMENT_2	int64
97	FLAG_DOCUMENT_3	int64
98	FLAG_DOCUMENT_4	int64
99	FLAG_DOCUMENT_5	int64
100	FLAG_DOCUMENT_6	int64
101	FLAG_DOCUMENT_7	int64
102	FLAG_DOCUMENT_8	int64
103	FLAG_DOCUMENT_9	int64
104	FLAG_DOCUMENT_10	int64
105	FLAG_DOCUMENT_11	int64
106	FLAG_DOCUMENT_12	int64
107	FLAG_DOCUMENT_13	int64
108	FLAG_DOCUMENT_14	int64
109	FLAG_DOCUMENT_15	int64
110	FLAG_DOCUMENT_16	int64
111	FLAG_DOCUMENT_17	int64
112	FLAG_DOCUMENT_18	int64
113	FLAG_DOCUMENT_19	int64
114	FLAG_DOCUMENT_20	int64

```
 115  FLAG_DOCUMENT_21              int64
 116  AMT_REQ_CREDIT_BUREAU_HOUR    float64
 117  AMT_REQ_CREDIT_BUREAU_DAY     float64
 118  AMT_REQ_CREDIT_BUREAU_WEEK    float64
 119  AMT_REQ_CREDIT_BUREAU_MON     float64
 120  AMT_REQ_CREDIT_BUREAU_QRT     float64
 121  AMT_REQ_CREDIT_BUREAU_YEAR    float64
dtypes: float64(65), int64(41), object(16)
memory usage: 286.2+ MB
```

In [23]:
```
prevDF.info()
```

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

In [24]:
```
# Checking the numeric variables of the dataframes
appDF.describe()
```

Out[24]:

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNU
count	307511.000000	307511.000000	307511.000000	3.075110e+05	3.075110e+05	307499.000
mean	278180.518577	0.080729	0.417052	1.687979e+05	5.990260e+05	27108.573
std	102790.175348	0.272419	0.722121	2.371231e+05	4.024908e+05	14493.737
min	100002.000000	0.000000	0.000000	2.565000e+04	4.500000e+04	1615.500
25%	189145.500000	0.000000	0.000000	1.125000e+05	2.700000e+05	16524.000
50%	278202.000000	0.000000	0.000000	1.471500e+05	5.135310e+05	24903.000
75%	367142.500000	0.000000	1.000000	2.025000e+05	8.086500e+05	34596.000
max	456255.000000	1.000000	19.000000	1.170000e+08	4.050000e+06	258025.500

8 rows × 106 columns

In [25]:
```
prevDF.describe()
```

Out[25]:

	SK_ID_PREV	SK_ID_CURR	AMT_ANNUITY	AMT_APPLICATION	AMT_CREDIT	AMT_DOWN_PA
count	1.670214e+06	1.670214e+06	1.297979e+06	1.670214e+06	1.670213e+06	7.743
mean	1.923089e+06	2.783572e+05	1.595512e+04	1.752339e+05	1.961140e+05	6.697
std	5.325980e+05	1.028148e+05	1.478214e+04	2.927798e+05	3.185746e+05	2.092
min	1.000001e+06	1.000010e+05	0.000000e+00	0.000000e+00	0.000000e+00	-9.000
25%	1.461857e+06	1.893290e+05	6.321780e+03	1.872000e+04	2.416050e+04	0.000
50%	1.923110e+06	2.787145e+05	1.125000e+04	7.104600e+04	8.054100e+04	1.638
75%	2.384280e+06	3.675140e+05	2.065842e+04	1.803600e+05	2.164185e+05	7.740
max	2.845382e+06	4.562550e+05	4.180581e+05	6.905160e+06	6.905160e+06	3.060

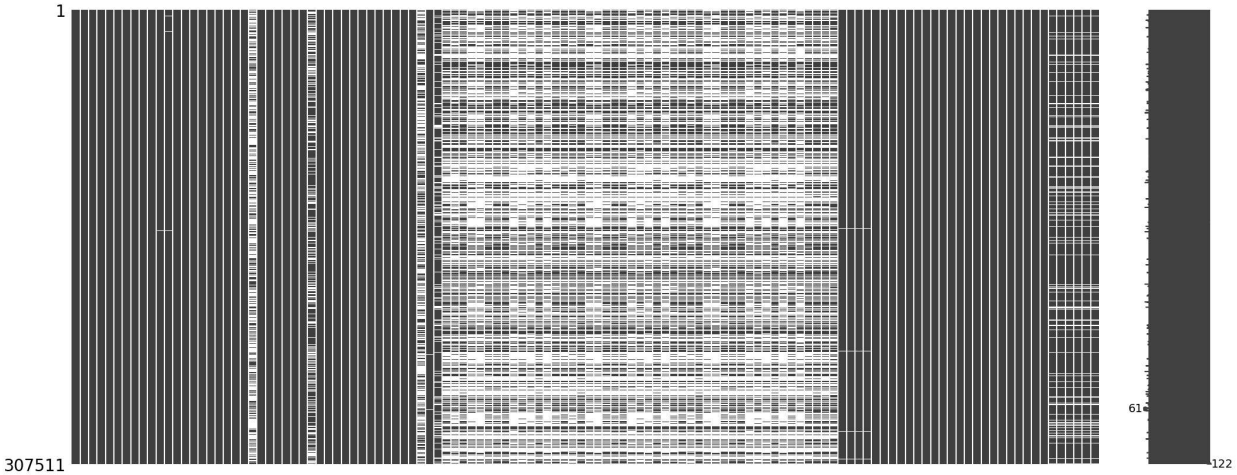
8 rows × 21 columns

Data Cleaning & Manipulation

In [41]:
```
# Null Value Calculation
# applicationDF Missing values

import missingno as mn # Used this import when we have more columns in dataset. Check
mn.matrix(appDF)
```

Out[41]:
```
<Axes: >
```



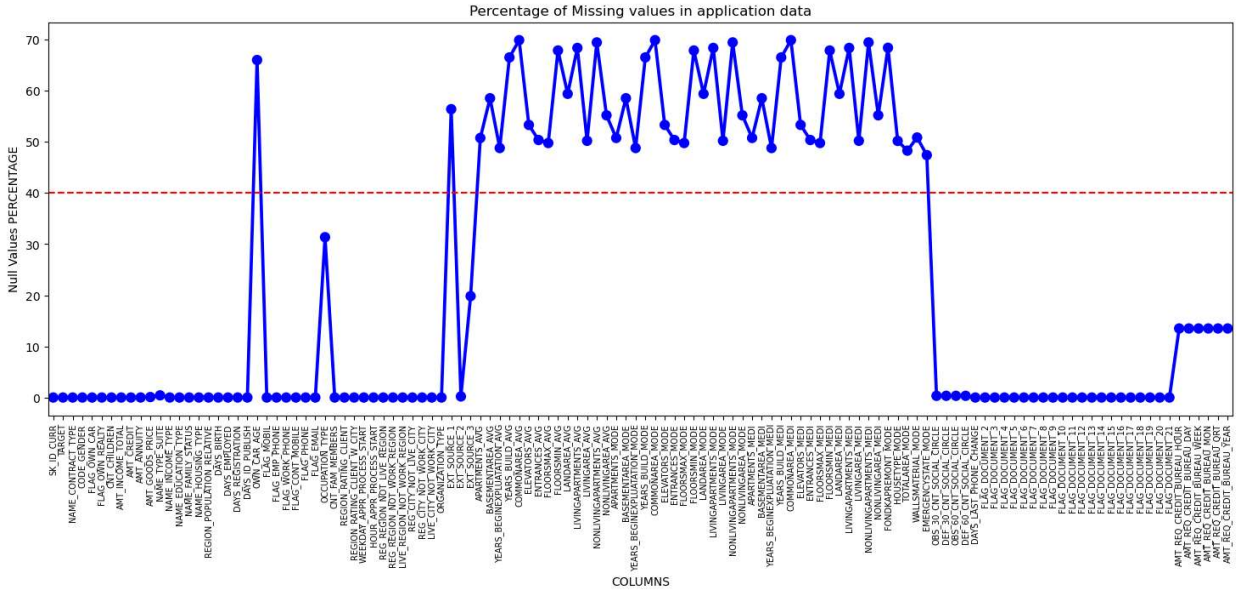
In [ ]:
```
#Insight:
'''
Based on the above Matrix, it is evidednt that the dataset has many missing values.
Let's check for each column what is the % of missing values
'''
```

In [42]:
```
# % null value in each column
round(appDF.isnull().sum() / appDF.shape[0] * 100.00,2)
```

Out[42]:
```
SK_ID_CURR                     0.0
TARGET                         0.0
NAME_CONTRACT_TYPE             0.0
CODE_GENDER                    0.0
FLAG_OWN_CAR                   0.0
                              ...
AMT_REQ_CREDIT_BUREAU_DAY     13.5
AMT_REQ_CREDIT_BUREAU_WEEK    13.5
AMT_REQ_CREDIT_BUREAU_MON     13.5
AMT_REQ_CREDIT_BUREAU_QRT     13.5
AMT_REQ_CREDIT_BUREAU_YEAR    13.5
Length: 122, dtype: float64
```

In [ ]:
```
#Insight:
'''There are many columns in applicationDF dataframe where missing value is more than
Let's plot the columns vs missing value % with 40% being the cut-off marks'''
```

In [43]:
```
null_applicationDF = pd.DataFrame((appDF.isnull().sum())*100/appDF.shape[0]).reset_inc
null_applicationDF.columns = ['Column Name', 'Null Values Percentage']
fig = plt.figure(figsize=(18,6))
ax = sns.pointplot(x="Column Name",y="Null Values Percentage",data=null_applicationDF,
plt.xticks(rotation =90,fontsize =7)
ax.axhline(40, ls='--',color='red')
plt.title("Percentage of Missing values in application data")
plt.ylabel("Null Values PERCENTAGE")
plt.xlabel("COLUMNS")
plt.show()
```



In [ ]:
```
#Insight:
'''
From the plot we can see the columns in which percentage of null values more than 40%
above the red line and the columns which have less than 40 % null values below the red
Let's check the columns which has more than 40% missing values
'''
```

In [45]:
```
# more than or equal to 40% empty rows columns
nullcol_40_application = null_applicationDF[null_applicationDF["Null Values Percentage
nullcol_40_application
```

Out[45]:

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

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

In [46]:
```
# How many columns have more than or euqal to 40% null values ?
len(nullcol_40_application)
```

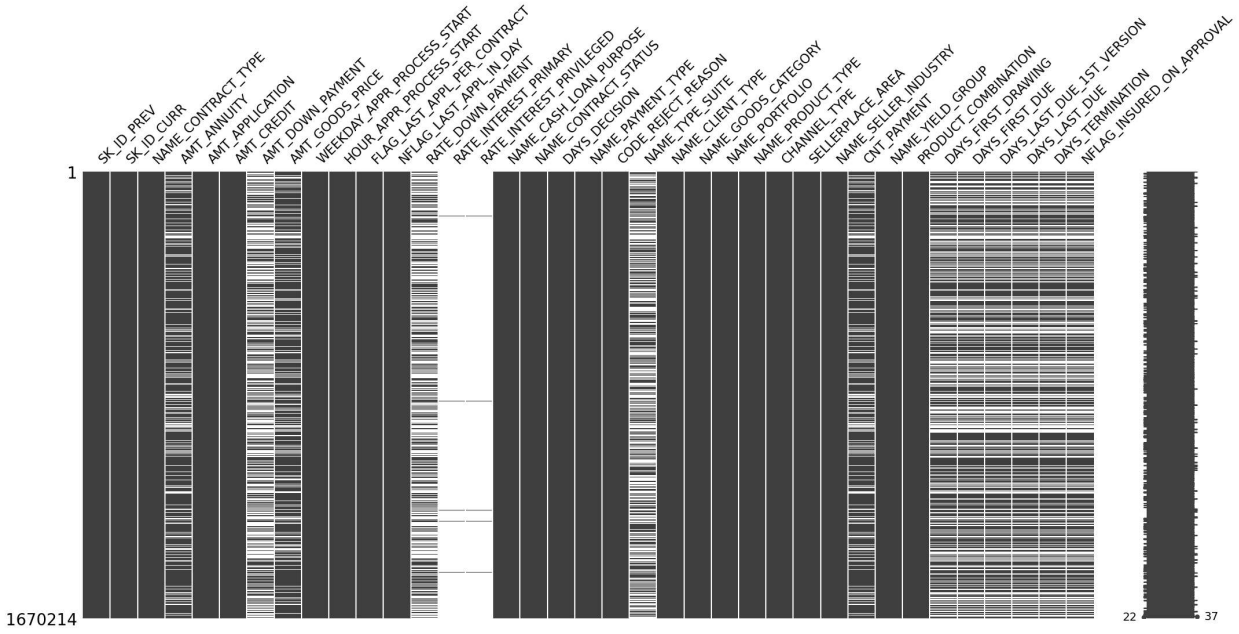
Out[46]:
```
49
```

In [ ]:
```
# Insight:
'''
Total of 49 columns are there which have more than 40% null values.Seems like most of
with high missing values are related to different area sizes on apartment owned/rented
'''
```

In [47]:
```
#  previousDF Missing Values

mn.matrix(prevDF)
```

Out[47]:
```
<Axes: >
```

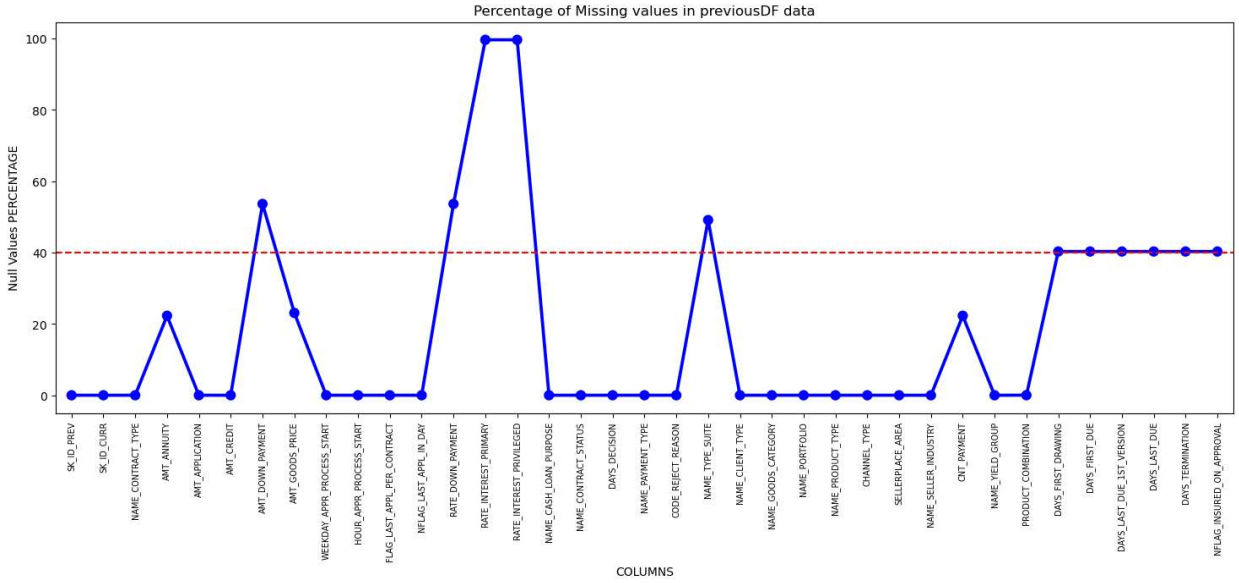


In [48]:
```
# checking the null value % of each column in previousDF dataframe
round(prevDF.isnull().sum() / prevDF.shape[0] * 100.00,2)
```

Out[48]:
```
SK_ID_PREV                    0.00
SK_ID_CURR                    0.00
NAME_CONTRACT_TYPE            0.00
AMT_ANNUITY                  22.29
AMT_APPLICATION               0.00
AMT_CREDIT                    0.00
AMT_DOWN_PAYMENT             53.64
AMT_GOODS_PRICE              23.08
WEEKDAY_APPR_PROCESS_START    0.00
HOUR_APPR_PROCESS_START       0.00
FLAG_LAST_APPL_PER_CONTRACT   0.00
NFLAG_LAST_APPL_IN_DAY        0.00
RATE_DOWN_PAYMENT            53.64
RATE_INTEREST_PRIMARY        99.64
RATE_INTEREST_PRIVILEGED     99.64
NAME_CASH_LOAN_PURPOSE        0.00
NAME_CONTRACT_STATUS          0.00
DAYS_DECISION                 0.00
NAME_PAYMENT_TYPE             0.00
CODE_REJECT_REASON            0.00
NAME_TYPE_SUITE              49.12
NAME_CLIENT_TYPE              0.00
NAME_GOODS_CATEGORY           0.00
NAME_PORTFOLIO                0.00
NAME_PRODUCT_TYPE             0.00
CHANNEL_TYPE                  0.00
SELLERPLACE_AREA              0.00
NAME_SELLER_INDUSTRY          0.00
CNT_PAYMENT                  22.29
NAME_YIELD_GROUP              0.00
PRODUCT_COMBINATION           0.02
DAYS_FIRST_DRAWING           40.30
DAYS_FIRST_DUE               40.30
DAYS_LAST_DUE_1ST_VERSION    40.30
DAYS_LAST_DUE                40.30
DAYS_TERMINATION             40.30
NFLAG_INSURED_ON_APPROVAL    40.30
dtype: float64
```

In [ ]:
```
# Insight:
'''
There are many columns in previousDF dataframe where missing value is more than 40%.
Let's plot the columns vs missing value % with 40% being the cut-off marks
'''
```

In [49]:
```
null_previousDF = pd.DataFrame((prevDF.isnull().sum())*100/prevDF.shape[0]).reset_inde
null_previousDF.columns = ['Column Name', 'Null Values Percentage']
fig = plt.figure(figsize=(18,6))
ax = sns.pointplot(x="Column Name",y="Null Values Percentage",data=null_previousDF,col
plt.xticks(rotation =90,fontsize =7)
ax.axhline(40, ls='--',color='red')
plt.title("Percentage of Missing values in previousDF data")
plt.ylabel("Null Values PERCENTAGE")
plt.xlabel("COLUMNS")
plt.show()
```



In [ ]:
```
# Insight:
'''
From the plot we can see the columns in which percentage of null values more than 40%
and the columns which have less than 40 % null values below the red line.
Let's check the columns which has more than 40% missing values
'''
```

In [50]:
```
# more than or equal to 40% empty rows columns
nullcol_40_previous = null_previousDF[null_previousDF["Null Values Percentage"]>=40]
nullcol_40_previous
```

Out[50]:

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

In [51]:
```
# How many columns have more than or euqal to 40% null values ?
len(nullcol_40_previous)
```

Out[51]:
```
11
```

In [ ]:
```
# Insight:
'''
```

```
Total of 11 columns are there which have more than 40% null values. These columns can
Before deleting these columns,
let's review if there are more columns which can be dropped or not[](http://)
'''
```

In [ ]:

In [ ]: