Preprocesamiento

In [1]:
```
import pandas as pd
```

In [4]:
```
myData = pd.read_csv('Lago Datos.csv', header=0, low_memory=False)
print(myData.describee)
```

```
<bound method NDFrame.describe of        ODATEDW OSOURCE  TCODE STATE    ZIP MAILCODE
PVASTATE   DOB NOEXCH  \
0         9401     BOA      1    CA  91326      NaN      NaN  5202      0
1         9001     AMH      1    NC  27017      NaN      NaN     0      0
2         8701     BRY      0    CA  95953      NaN      NaN  2801      0
3         8601     NaN      0    FL  33176      NaN      NaN  2001      0
4         9401     CWR      0    AL  35603      NaN      NaN     0      0
...        ...     ...    ...   ...    ...      ...      ...   ...    ...
47687     9101     PBL      0    CA  94305      NaN      NaN  1401      0
47688     9201     HHH     28    CA  92581      NaN      NaN     0      0
47689     9601     ASE      1    AK  99504      NaN      NaN     0      0
47690     9601     DCD      1    TX  77379      NaN      NaN  5001      0
47691     8801     MCC      2    NC  28409      NaN      NaN  1801      0

      RECINHSE  ...     IDX HPHONE_D RFA_2R RFA_2F RFA_2A  MDMAUD_R  MDMAUD_F  \
0          NaN  ...  148535        0      L      2      G         X         X
1          NaN  ...   15078        1      L      4      E         X         X
2          NaN  ...  172556        1      L      4      E         X         X
3            X  ...    7112        1      L      2      F         X         X
4          NaN  ...   47784        0      L      1      F         X         X
...        ...  ...     ...      ...    ...    ...    ...       ...       ...
47687        X  ...   12322        1      L      3      G         L         2
47688      NaN  ...  156106        0      L      1      G         X         X
47689      NaN  ...  184568        0      L      1      G         X         X
47690      NaN  ...  122706        1      L      1      F         X         X
47691        X  ...  185114        1      L      1      G         C         1

      MDMAUD_A CLUSTER2 GEOCODE2
0            X      1.0        A
1            X     60.0        C
2            X     41.0        C
3            X     26.0        A
4            X     16.0        C
...        ...      ...      ...
47687        C      2.0        A
47688        X     35.0        A
47689        X     12.0        C
47690        X      2.0        A
47691        C     12.0        C

[47692 rows x 479 columns]>
```

Valores perdidos

In [5]:
```
print(myData.isnull().any().any())
```

```
True
```

Ordenando atributos por su tipo

In [7]:
```
tipos = myData.columns.to_series().groupby(myData.dtypes).groups
```

Litando variables categóricos

In [9]:
```
import numpy as np
colText = tipos[np.dtype('object')]
print(colText)
print(len(colText))
```

```
Index(['OSOURCE', 'STATE', 'ZIP', 'MAILCODE', 'PVASTATE', 'NOEXCH', 'RECINHSE',
       'RECP3', 'RECPGVG', 'RECSWEEP', 'MDMAUD', 'DOMAIN', 'AGEFLAG',
       'HOMEOWNR', 'CHILD03', 'CHILD07', 'CHILD12', 'CHILD18', 'GENDER',
       'MAJOR', 'COLLECT1', 'VETERANS', 'BIBLE', 'CATLG', 'HOMEE', 'PETS',
       'CDPLAY', 'STEREO', 'PCOWNERS', 'PHOTO', 'CRAFTS', 'FISHER', 'GARDENIN',
       'BOATS', 'WALKER', 'KIDSTUFF', 'CARDS', 'PLATES', 'PEPSTRFL', 'RFA_2',
       'RFA_3', 'RFA_4', 'RFA_5', 'RFA_6', 'RFA_7', 'RFA_8', 'RFA_9', 'RFA_10',
       'RFA_11', 'RFA_12', 'RFA_13', 'RFA_14', 'RFA_15', 'RFA_16', 'RFA_17',
       'RFA_18', 'RFA_19', 'RFA_20', 'RFA_21', 'RFA_22', 'RFA_23', 'RFA_24',
       'RFA_2R', 'RFA_2A', 'MDMAUD_R', 'MDMAUD_F', 'MDMAUD_A', 'GEOCODE2'],
      dtype='object')
68
```

Listando atributos numéricos

In [11]:
```
columnas = myData.columns
colNum = list(set(columnas)-set(colText))
print(colNum)
print(len(colNum))
```

```
['LFC7', 'ANC3', 'POP903', 'LSC2', 'PEC2', 'HC20', 'OCC6', 'IC12', 'HHD6', 'HHAS2',
'HHD2', 'HC11', 'HUPA6', 'EIC14', 'ADATE_6', 'RP4', 'ETHC5', 'LASTGIFT', 'OEDC6', 'AF
C4', 'HUPA4', 'RAMNT_22', 'DW2', 'RAMNT_5', 'RAMNT_3', 'AFC5', 'MC2', 'INCOME', 'AGE9
04', 'POP90C3', 'DW3', 'DATASRCE', 'HU5', 'AGEC4', 'MSA', 'IC4', 'MALEVET', 'MARR2',
'CHILC5', 'TIMELAG', 'OCC3', 'EIC3', 'RAMNT_14', 'RHP2', 'PUBDOITY', 'RDATE_17', 'RAM
NT_18', 'WEALTH1', 'ADATE_15', 'ETHC1', 'TPE5', 'LFC10', 'POP90C2', 'ADATE_13', 'HHD
5', 'ADATE_7', 'POP902', 'HUPA7', 'RAMNT_6', 'SEC1', 'ADATE_10', 'POP90C5', 'RDATE_1
4', 'IC14', 'ETH15', 'ANC15', 'HU3', 'HHD11', 'EIC4', 'HHD10', 'ETHC6', 'MARR4', 'RP
2', 'HC21', 'EIC11', 'ADATE_12', 'TPE2', 'HHN6', 'IC2', 'HIT', 'CLUSTER', 'EC8', 'IC
5', 'AGEC1', 'HUR2', 'RDATE_9', 'RFA_2F', 'RAMNT_24', 'RAMNT_23', 'CHIL3', 'ADATE_2
4', 'DW9', 'ETH14', 'LFC5', 'EC4', 'IC19', 'HC3', 'EIC16', 'EC1', 'OCC9', 'HHD1', 'HC
5', 'OEDC7', 'AGEC5', 'ANC13', 'RAMNT_17', 'ETH10', 'NUMPRM12', 'AGE901', 'RAMNT_20',
'MBCRAFT', 'EIC2', 'AGE', 'PUBGARDN', 'NEXTDATE', 'TPE11', 'RDATE_22', 'IC22', 'IDX',
'ADATE_8', 'TCODE', 'ANC1', 'HHAGE1', 'HU1', 'ETH6', 'AGEC6', 'HC6', 'NGIFTALL', 'EIC
13', 'RDATE_21', 'OCC12', 'MHUC1', 'LSC1', 'LFC4', 'ANC11', 'HC7', 'STATEGOV', 'OCC1
3', 'HC15', 'CHILC3', 'POP90C1', 'LFC9', 'EC2', 'ANC9', 'HC16', 'OEDC2', 'EC5', 'SEC
5', 'ETHC4', 'HHAGE3', 'ADATE_11', 'SOLIH', 'OCC10', 'GEOCODE', 'HUPA3', 'CHIL2', 'IC
8', 'VC2', 'HHAS4', 'OEDC3', 'AGEC2', 'RDATE_20', 'HHN4', 'IC10', 'HHAS3', 'POBC1',
'CARDPM12', 'LIFESRC', 'RDATE_6', 'ADATE_23', 'PUBOPP', 'RDATE_3', 'ETH11', 'ANC6',
'HHD8', 'HHAS1', 'AC1', 'MC1', 'HC4', 'HUR1', 'HVP4', 'AGE902', 'RAMNT_10', 'ADATE_
5', 'EIC9', 'AGE906', 'DW5', 'RDATE_13', 'DW6', 'HU4', 'HC10', 'HPHONE_D', 'MAXADAT
E', 'ADATE_4', 'SEC3', 'RAMNT_12', 'CHILC4', 'TPE8', 'SEC2', 'LFC1', 'TPE13', 'RDATE_
8', 'ETH16', 'HUPA5', 'EIC12', 'ADATE_19', 'ADATE_20', 'CARDGIFT', 'EC3', 'HC1', 'VOC
1', 'ADATE_2', 'RAMNT_13', 'PUBHLTH', 'MBGARDEN', 'RHP4', 'ETH12', 'HV3', 'EIC8', 'VO
C2', 'ETH3', 'ANC5', 'DW8', 'HVP1', 'CHIL1', 'MAGMALE', 'TPE1', 'ANC4', 'ANC14', 'RAM
NT_11', 'RDATE_5', 'MINRDATE', 'IC21', 'IC11', 'HVP2', 'TPE12', 'ETH1', 'AGE907', 'OE
DC4', 'RAMNT_21', 'FISTDATE', 'RAMNT_15', 'IC16', 'ADATE_16', 'HVP5', 'ETHC2', 'HV1',
'EC6', 'IC3', 'HU2', 'TPE7', 'HC14', 'HC18', 'AC2', 'ETH2', 'ETH5', 'MALEMILI', 'HHP
2', 'RP1', 'IC13', 'POP901', 'ADATE_17', 'RDATE_10', 'MAXRDATE', 'CARDPROM', 'RAMNT_
8', 'RAMNT_9', 'AFC6', 'AFC2', 'ETHC3', 'EIC5', 'HHD4', 'ANC8', 'ANC12', 'HHAGE2', 'P
EC1', 'MAGFAML', 'VIETVETS', 'SEC4', 'CHILC1', 'ADATE_14', 'VC4', 'DMA', 'IC1', 'AFC
1', 'HVP3', 'AVGGIFT', 'OEDC5', 'POP90C4', 'LOCALGOV', 'HHP1', 'HV4', 'DW4', 'NUMCHL
D', 'OCC7', 'OCC4', 'RDATE_18', 'ETH7', 'ANC2', 'RDATE_11', 'LFC6', 'LSC4', 'OCC11',
'MINRAMNT', 'PUBCULIN', 'IC17', 'ETH9', 'ADI', 'TPE9', 'ADATE_21', 'MBBOOKS', 'EIC1
0', 'HV2', 'HC2', 'AGE903', 'HHD9', 'CHILC2', 'ADATE_18', 'WEALTH2', 'RP3', 'HHN1',
'HVP6', 'LFC3', 'HC8', 'DW7', 'NUMPROM', 'RAMNTALL', 'IC7', 'HHN3', 'LFC8', 'HHD12',
'HC12', 'RDATE_7', 'RDATE_19', 'OCC8', 'EIC6', 'IC6', 'AGE905', 'DW1', 'OCC2', 'ADATE
_3', 'RDATE_4', 'HHN2', 'WWIIVETS', 'LASTDATE', 'RDATE_16', 'TPE4', 'EIC15', 'HHD7',
'EIC7', 'ETH13', 'HC19', 'FEDGOV', 'DOB', 'ETH4', 'AFC3', 'RAMNT_19', 'RDATE_15', 'RD
ATE_23', 'EIC1', 'IC9', 'HUPA1', 'AGEC7', 'OCC1', 'HHN5', 'ETH8', 'PUBNEWFN', 'RAMNT_
16', 'MHUC2', 'EC7', 'MAXRAMNT', 'IC15', 'CLUSTER2', 'MARR3', 'OCC5', 'TPE10', 'IC2
3', 'HHD3', 'ADATE_9', 'HUPA2', 'VC3', 'PUBPHOTO', 'HC17', 'POBC2', 'MAGFEM', 'MARR
1', 'VC1', 'RAMNT_7', 'SOLP3', 'TPE6', 'HC13', 'HC9', 'AGEC3', 'MC3', 'LSC3', 'IC20',
'ANC10', 'RDATE_24', 'TPE3', 'RDATE_12', 'VOC3', 'IC18', 'RHP1', 'RAMNT_4', 'LFC2',
'RHP3', 'OEDC1', 'MBCOLECT', 'ADATE_22', 'ANC7', 'ODATEDW']
411
```

Completamos valores perdidos en atributos numéricos

In [14]:
```
for c in colNum:
    mean = myData[c].mean()
    myData[c] = myData[c].fillna(mean)
```

Completando valores perdidos en atributos categóricos

In [15]:
```
for c in colText:
    mode = myData[c].mode()[0]
```

```
    myData[c] = myData[c].fillna(mode)
```

Después de completar los valores perdidos, observemos si hay todavía nulls

In [16]:
```
print(myData.isnull().any().any())
```

```
False
```

Exportamos en un nuevo archivo

In [17]:
```
myData.to_csv("datasetLimpio.csv", index=False)
```

In [ ]:

In [ ]: