Steps for Handling the missing value

1. Import Libraries
2. Load data
3. Seprate Input and Output attributes
4. Find the missing values and handle it in either way a. Removing data b. Imputation

In [1]:

```
# Step 1: Import Libraries

import numpy as np
import pandas as pd
from sklearn.impute import SimpleImputer

# Step 2: Load Data

datasets = pd.read_csv('Data_for_Missing_Values.csv')
print("\nData :\n",datasets)
print("\nData statistics\n",datasets.describe())
```

```
Data :
      Country   Age   Salary Purchased
0    France  44.0  72000.0        No
1     Spain  27.0  48000.0       Yes
2   Germany  30.0  54000.0        No
3     Spain  38.0  61000.0        No
4       NaN   NaN      NaN       NaN
5   Germany  40.0      NaN       Yes
6    France  35.0  58000.0       Yes
7     Spain   NaN  52000.0        No
8    France  48.0  79000.0       Yes
9   Germany  50.0  83000.0        No
10   France  37.0  67000.0       Yes
11    Spain  45.0  55000.0        No

Data statistics
              Age        Salary
count  10.000000     10.000000
mean   39.400000  62900.000000
std     7.515909  11892.574714
min    27.000000  48000.000000
25%    35.500000  54250.000000
50%    39.000000  59500.000000
75%    44.750000  70750.000000
max    50.000000  83000.000000
```

In [2]:

```
# Step 3: Seprate Input and Output attributes

# All rows, all columns except last
X = datasets.iloc[:, :-1].values

# Only last column
Y = datasets.iloc[:, -1].values

print("\n\nInput : \n", X)
print("\n\nOutput: \n", Y)
```

```
Input :
 [['France' 44.0 72000.0]
 ['Spain' 27.0 48000.0]
 ['Germany' 30.0 54000.0]
 ['Spain' 38.0 61000.0]
 [nan nan nan]
 ['Germany' 40.0 nan]
 ['France' 35.0 58000.0]
 ['Spain' nan 52000.0]
 ['France' 48.0 79000.0]
 ['Germany' 50.0 83000.0]
 ['France' 37.0 67000.0]
 ['Spain' 45.0 55000.0]]


Output:
 ['No' 'Yes' 'No' 'No' nan 'Yes' 'Yes' 'No' 'Yes' 'No' 'Yes' 'No']
```

In [3]:

```
# Step 4: Find the missing values and handle it in either way

# 4a. Removing the row with all null values

datasets.dropna(how='all',inplace=True)
print("\nNew Data :",datasets)
```

```
New Data :     Country   Age   Salary Purchased
0    France  44.0  72000.0        No
1     Spain  27.0  48000.0       Yes
2   Germany  30.0  54000.0        No
3     Spain  38.0  61000.0        No
5   Germany  40.0      NaN       Yes
6    France  35.0  58000.0       Yes
7     Spain   NaN  52000.0        No
8    France  48.0  79000.0       Yes
9   Germany  50.0  83000.0        No
10   France  37.0  67000.0       Yes
11    Spain  45.0  55000.0        No
```

In [4]:

```
# 4b. Imputation (Replacing null values with mean value of that attribute)

# All rows, all columns except last
new_X = datasets.iloc[:, :-1].values

# Only last column
new_Y = datasets.iloc[:, -1].values


# Using Imputer function to replace NaN values with mean of that parameter value
imputer = SimpleImputer(missing_values = np.nan,strategy = "mean")

# Fitting the data, function learns the stats
imputer = imputer.fit(new_X[:, 1:3])

# fit_transform() will execute those stats on the input ie. X[:, 1:3]
new_X[:, 1:3] = imputer.transform(new_X[:, 1:3])

# filling the missing value with mean
print("\n\nNew Input with Mean Value for NaN : \n\n", new_X)
```

```
New Input with Mean Value for NaN :

 [['France' 44.0 72000.0]
 ['Spain' 27.0 48000.0]
 ['Germany' 30.0 54000.0]
 ['Spain' 38.0 61000.0]
 ['Germany' 40.0 62900.0]
 ['France' 35.0 58000.0]
 ['Spain' 39.4 52000.0]
 ['France' 48.0 79000.0]
 ['Germany' 50.0 83000.0]
 ['France' 37.0 67000.0]
 ['Spain' 45.0 55000.0]]
```