In [7]:
```
import pandas as pd
import numpy as np
```

In [3]:
```
import matplotlib.pyplot as plt
import seaborn as sms
```

In [ ]:
```
##import data_set
```

In [16]:
```
data_df = pd.read_csv("./Data.csv")
```

In [9]:
```
data_df.head()
```

Out[9]:

	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	Germany	40.0	NaN	Yes

In [ ]:
```
#Xử lí dữ liệu bị Nan
```

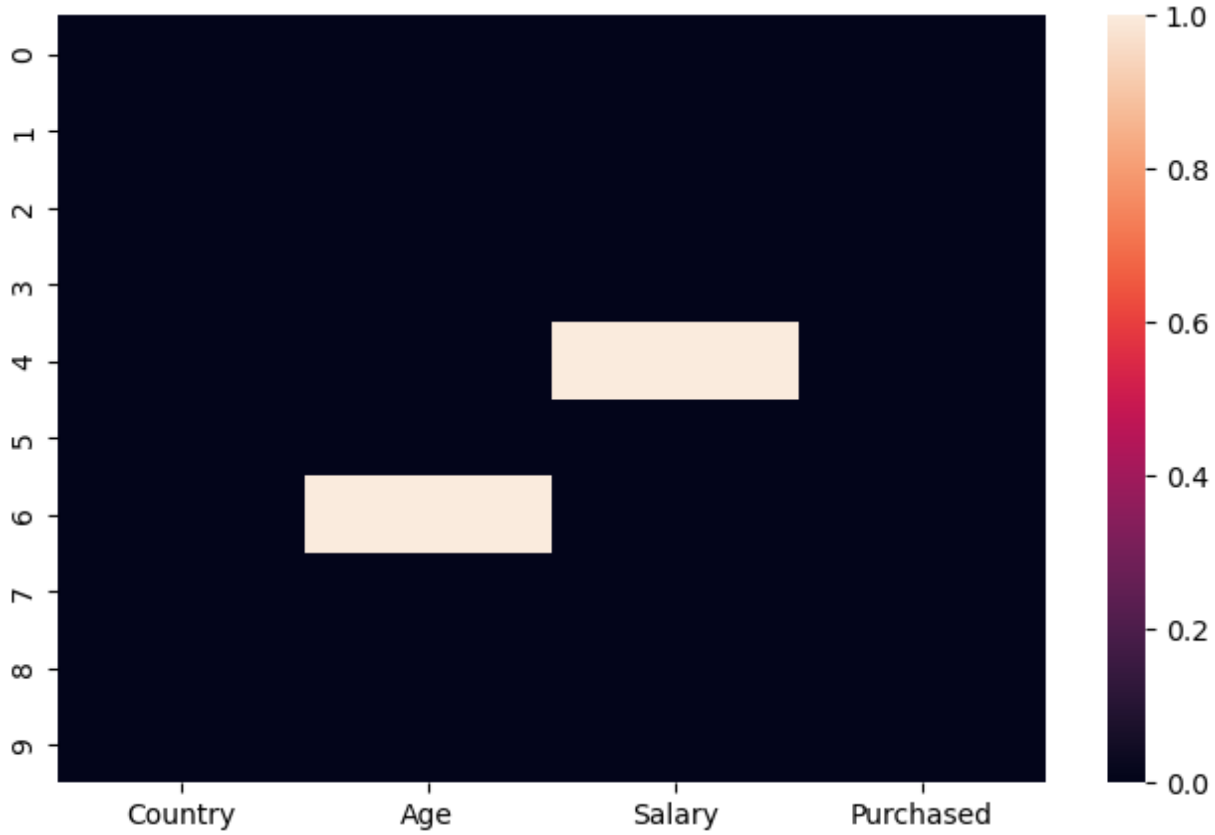
In [10]:
```
data_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Country    10 non-null     object 
 1   Age        9 non-null      float64
 2   Salary     9 non-null      float64
 3   Purchased  10 non-null     object 
dtypes: float64(2), object(2)
memory usage: 452.0+ bytes
```

In [29]:
```
for col in data_df.columns:
    Missing_data=data_df[col].isna().sum()
    Missing_percent=Missing_data/len(data_df)*100
    print(f"Cột {col} có {Missing_percent} % dữ liệu bị Nan")
```

```
Cột Country có 0.0 % dữ liệu bị Nan
Cột Age có 10.0 % dữ liệu bị Nan
Cột Salary có 10.0 % dữ liệu bị Nan
Cột Purchased có 0.0 % dữ liệu bị Nan
```

In [11]:
```
fix,ag=plt.subplots(figsize=(8,5))
sms.heatmap(data_df.isna(),);
```



In [19]:
```
data_df = pd.read_csv("./Data.csv")
```

In [28]:
```
x= data_df.iloc[:,:-1].values
x
```

Out[28]:
```
array([['France', 44.0, 72000.0],
       ['Spain', 27.0, 48000.0],
       ['Germany', 30.0, 54000.0],
       ['Spain', 38.0, 61000.0],
       ['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]], dtype=object)
```

In [ ]:
```
#x= data_df.iloc[:,:-1] lấy các giá trị trong dataset, ngoại trừ giá trị cuối
#x chứa các giá trị hoành độ
```

In [27]:
```
y= data_df.iloc[:,-1].values
y
```

Out[27]:
```
array(['No', 'Yes', 'No', 'No', 'Yes', 'Yes', 'No', 'Yes', 'No', 'Yes'],
      dtype=object)
```

In [ ]:
```
#y chứa các giá trị mua hàng hay ko, tung độ
```

In [41]:
```
from sklearn.impute import SimpleImputer
#Tạo class chứa các giá trị Nan
imputer= SimpleImputer(missing_values=np.nan,strategy="mean")
imputer.fit(x[:,-1:3])
```

```
x[:,-1:3]=imputer.transform(x[:, -1:3])
x
```

Out[41]:
```
array([['France', 44.0, 72000.0],
       ['Spain', 27.0, 48000.0],
       ['Germany', 30.0, 54000.0],
       ['Spain', 38.0, 61000.0],
       ['Germany', 40.0, 63777.77777777778],
       ['France', 35.0, 58000.0],
       ['Spain', nan, 52000.0],
       ['France', 48.0, 79000.0],
       ['Germany', 50.0, 83000.0],
       ['France', 37.0, 67000.0]], dtype=object)
```

In [ ]:
```
#Mã hóa dữ liệu danh mục
#Chuyển hóa dữ liệu dạng string sang dạng numberic
#encoding indenpendent varible(x) mã hóa biến độc lập
```

In [42]:
```
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
ct=ColumnTransformer(transformers=[('encoder',OneHotEncoder(),[0])],remainder="passthr
X= ct.fit_transform(x)
X
```

Out[42]:
```
array([[1.0, 0.0, 0.0, 44.0, 72000.0],
       [0.0, 0.0, 1.0, 27.0, 48000.0],
       [0.0, 1.0, 0.0, 30.0, 54000.0],
       [0.0, 0.0, 1.0, 38.0, 61000.0],
       [0.0, 1.0, 0.0, 40.0, 63777.77777777778],
       [1.0, 0.0, 0.0, 35.0, 58000.0],
       [0.0, 0.0, 1.0, nan, 52000.0],
       [1.0, 0.0, 0.0, 48.0, 79000.0],
       [0.0, 1.0, 0.0, 50.0, 83000.0],
       [1.0, 0.0, 0.0, 37.0, 67000.0]], dtype=object)
```

In [ ]:
```
#encoding denpendent varible(y) biến phụ thuộc mã hóa
```

In [49]:
```
y= data_df.iloc[:,-1].values
y
```

Out[49]:
```
array(['No', 'Yes', 'No', 'No', 'Yes', 'Yes', 'No', 'Yes', 'No', 'Yes'],
      dtype=object)
```

In [51]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
y=le.fit_transform(y)
y
```

Out[51]:
```
array([0, 1, 0, 0, 1, 1, 0, 1, 0, 1], dtype=int64)
```

In [ ]:
```
#Spliting the dataset(x=data input,y=output) into the training set and test set
```

In [52]:
```
from sklearn.model_selection import train_test_split
np.random.seed(42)
X_train,X_test,y_train,y_test=train_test_split(X,y, test_size=0.2,)
```

In [53]:
```
X_train
```

Out[53]:
```
array([[1.0, 0.0, 0.0, 35.0, 58000.0],
       [1.0, 0.0, 0.0, 44.0, 72000.0],
       [1.0, 0.0, 0.0, 48.0, 79000.0],
       [0.0, 1.0, 0.0, 30.0, 54000.0],
       [1.0, 0.0, 0.0, 37.0, 67000.0],
       [0.0, 1.0, 0.0, 40.0, 63777.77777777778],
       [0.0, 0.0, 1.0, 38.0, 61000.0],
       [0.0, 0.0, 1.0, nan, 52000.0]], dtype=object)
```

In [54]:
```
y_train
```

Out[54]:
```
array([1, 0, 1, 0, 1, 1, 0, 0], dtype=int64)
```

In [55]:
```
X_test
```

Out[55]:
```
array([[0.0, 1.0, 0.0, 50.0, 83000.0],
       [0.0, 0.0, 1.0, 27.0, 48000.0]], dtype=object)
```

In [56]:
```
y_test
```

Out[56]:
```
array([0, 1], dtype=int64)
```

In [ ]:
```
#Feature Scalling
```

In [58]:
```
from sklearn.preprocessing import StandardScaler
sc= StandardScaler()
X_train[:,3:]=sc.fit_transform(X_train[:,3:])
X_train
```

Out[58]:
```
array([[1.0, 0.0, 0.0, -0.7061388043040211, -0.6260377781240922],
       [1.0, 0.0, 0.0, 0.9415184057386947, 1.013042950055349],
       [1.0, 0.0, 0.0, 1.673810499091013, 1.8325833141450698],
       [0.0, 1.0, 0.0, -1.6215039209944186, -1.0943465576039326],
       [1.0, 0.0, 0.0, -0.339992757627862, 0.4276569757055486],
       [0.0, 1.0, 0.0, 0.20922631238637662, 0.05040823668012205],
       [0.0, 0.0, 1.0, -0.15691973428978245, -0.274806193514212],
       [0.0, 0.0, 1.0, nan, -1.328500947343853]], dtype=object)
```

In [59]:
```
X_test=sc.transform(X_train[:,3:])
X_test
```

Out[59]:
```
array([[-0.7061388 , -0.62603778],
       [ 0.94151841,  1.01304295],
       [ 1.6738105 ,  1.83258331],
       [-1.62150392, -1.09434656],
       [-0.33999276,  0.42765698],
       [ 0.20922631,  0.05040824],
       [-0.15691973, -0.27480619],
       [        nan, -1.32850095]])
```

In [ ]: