1) Importing

In [2]:

```
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

2.Load the Dataset

In [3]:

```
data=pd.read_csv("Churn_Modelling.csv")
```

In [4]:

```
data
```

Out[4]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	Estim
0	1	15634602	Hargrave	619	France	Female	42	2	0.00	1	1	1	1
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1	1
2	3	15619304	Onio	502	France	Female	42	8	159660.80	3	1	0	1
3	4	15701354	Boni	699	France	Female	39	1	0.00	2	0	0	
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	1	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	
9995	9996	15606229	Obijiaku	771	France	Male	39	5	0.00	2	1	0	
9996	9997	15569892	Johnstone	516	France	Male	35	10	57369.61	1	1	1	1
9997	9998	15584532	Liu	709	France	Female	36	7	0.00	1	0	1	
9998	9999	15682355	Sabbatini	772	Germany	Male	42	3	75075.31	2	1	0	
9999	10000	15628319	Walker	792	France	Female	28	4	130142.79	1	1	0	

10000 rows × 14 columns

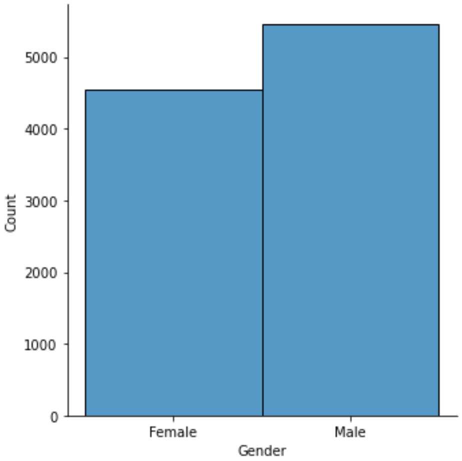
3.Visualizations

a) Univariate Analysis

In [8]:

```
sns.displot(data.Gender)
```

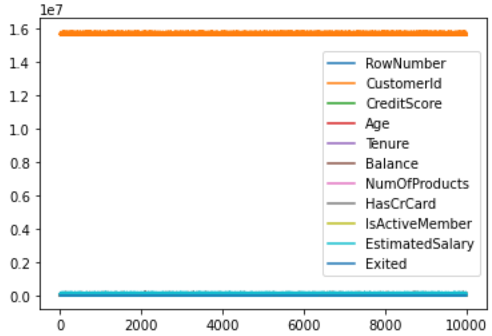
Out[8]:<seaborn.axisgrid.FacetGrid at 0x10a9e298>



In [10]:

```
data.plot.line()
```

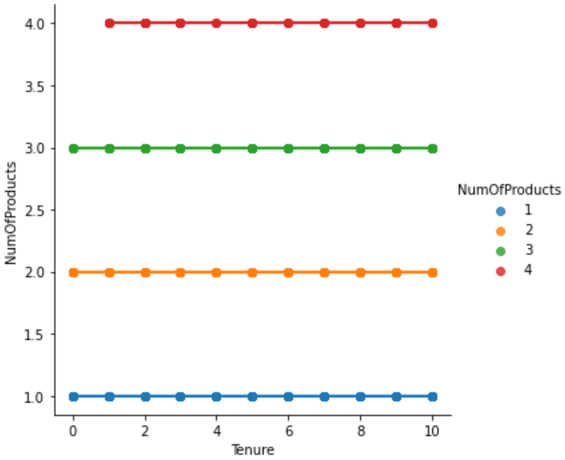
Out[10]:<AxesSubplot:>



In [11]:

```
sns.lmplot("Tenure","NumOfProducts",data,hue="NumOfProducts")
```

Out[11]:<seaborn.axisgrid.FacetGrid at 0x11293d0>



4)Perform descriptive statistics on the dataset.

In [12]:

```
data.describe()
```

Out[12]:

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember
count	10000.00000	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.00000	10000.000000
mean	5000.50000	1.569094e+07	650.528800	38.921800	5.012800	76485.889288	1.530200	0.70550	0.515100
std	2886.89568	7.193619e+04	96.653299	10.487806	2.892174	62397.405202	0.581654	0.45584	0.499797
min	1.00000	1.556570e+07	350.000000	18.000000	0.000000	0.000000	1.000000	0.00000	0.000000
25%	2500.75000	1.562853e+07	584.000000	32.000000	3.000000	0.000000	1.000000	0.00000	0.000000
50%	5000.50000	1.569074e+07	652.000000	37.000000	5.000000	97198.540000	1.000000	1.00000	1.000000
75%	7500.25000	1.575323e+07	718.000000	44.000000	7.000000	127644.240000	2.000000	1.00000	1.000000
max	10000.00000	1.581569e+07	850.000000	92.000000	10.000000	250898.090000	4.000000	1.00000	1.000000

5)Handle the Missing values.

In [13]:

```
data = pd.read_csv("Churn_Modelling.csv")
pd.isnull(data["Gender"])
```

Out[13]:

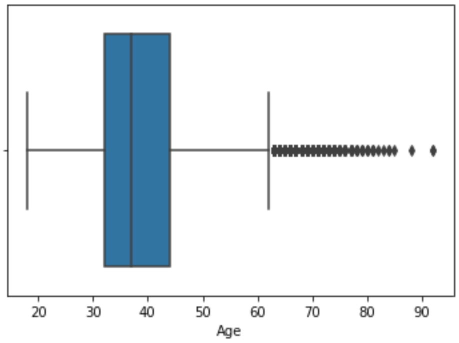
```
0       False
1       False
2       False
3       False
4       False
        ...  
9995    False
9996    False
9997    False
9998    False
9999    False
Name: Gender, Length: 10000, dtype: bool
```

6)Find the outliers and replace the outliers

In [14]:

```
sns.boxplot(data['Age'])
```

Out[14]:<AxesSubplot:xlabel='Age'>



In [15]:

```
data['Age']=np.where(data['Age']>50,40,data['Age'])
data['Age']
```

Out[15]:

```
0       42
1       41
2       42
3       39
4       43
        ..
9995    39
9996    35
9997    36
9998    42
9999    28
Name: Age, Length: 10000, dtype: int64
```

7) Check for Categorical columns and perform encoding.

In [17]:

```
pd.get_dummies(data, columns=["Gender", "Age"], prefix=["Age", "Gender"]).head()
```

Out[17]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	...	Gender_41	Gender_4
0	1	15634602	Hargrave	619	France	2	0.00	1	1	1	...	0	
1	2	15647311	Hill	608	Spain	1	83807.86	1	0	1	...	1	
2	3	15619304	Onio	502	France	8	159660.80	3	1	0	...	0	
3	4	15701354	Boni	699	France	1	0.00	2	0	0	...	0	
4	5	15737888	Mitchell	850	Spain	2	125510.82	1	1	1	...	0	

5 rows × 45 columns

8) Split the data into dependent and independent variables.

A) Split the data into Independent variables.

In [18]:

```
X = data.iloc[:, :-1].values
print(X)
```

```
[[1 15634602 'Hargrave' ... 1 1 101348.88]
 [2 15647311 'Hill' ... 0 1 112542.58]
 [3 15619304 'Onio' ... 1 0 113931.57]
 ...
 [9998 15584532 'Liu' ... 0 1 42085.58]
 [9999 15682355 'Sabbatini' ... 1 0 92888.52]
 [10000 15628319 'Walker' ... 1 0 38190.78]]
```

B) Split the data into Dependent variables.

In [19]:

```
Y = data.iloc[:, -1].values
print(Y)
```

```
[1 0 1 ... 1 1 0]
```