Telco Customer Churn EDA

Each row represents a customer, each column contains customer's attributes described on the column Metadata.

The data set includes information about:

Customers who left within the last month – the column is called Churn

Services that each customer has signed up for – phone, multiple lines, internet, online security, online backup, device protection, tech support, and streaming TV and movies

Customer account information – how long they've been a customer, contract, payment method, paperless billing, monthly charges, and total charges

Demographic info about customers – gender, age range, and if they have partners and dependents

Data Set link: https://www.kaggle.com/datasets/blastchar/telco-customer-churn (https://www.kaggle.com/datasets/blastchar/telco-customer-churn)

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.ticker as mtick
import seaborn as sns
%matplotlib inline
```

In [2]:
```
telecom_data = pd.read_csv('WA_Fn-UseC_-Telco-Customer-Churn.csv')
```

In [39]:
```
telecom_data.head(15)
```

Out[39]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service
1	5575-GNVDE	Male	0	No	No	34	Yes	No
2	3668-QPYBK	Male	0	No	No	2	Yes	No
3	7795-CFOCW	Male	0	No	No	45	No	No phone service
4	9237-HQITU	Female	0	No	No	2	Yes	No
5	9305-CDSKC	Female	0	No	No	8	Yes	Yes
6	1452-KIOVK	Male	0	No	Yes	22	Yes	Yes
7	6713-OKOMC	Female	0	No	No	10	No	No phone service
8	7892-POOKP	Female	0	Yes	No	28	Yes	Yes
9	6388-TABGU	Male	0	No	Yes	62	Yes	No
10	9763-GRSKD	Male	0	Yes	Yes	13	Yes	No
11	7469-LKBCI	Male	0	No	No	16	Yes	No
12	8091-TTVAX	Male	0	Yes	No	58	Yes	Yes
13	0280-XJGEX	Male	0	No	No	49	Yes	Yes
14	5129-JLPIS	Male	0	No	No	25	Yes	No

15 rows × 21 columns

In [4]:
```
telecom_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   customerID        7043 non-null   object 
 1   gender            7043 non-null   object 
 2   SeniorCitizen     7043 non-null   int64  
 3   Partner           7043 non-null   object 
 4   Dependents        7043 non-null   object 
 5   tenure            7043 non-null   int64  
 6   PhoneService      7043 non-null   object 
 7   MultipleLines     7043 non-null   object 
 8   InternetService   7043 non-null   object 
 9   OnlineSecurity    7043 non-null   object 
 10  OnlineBackup      7043 non-null   object 
 11  DeviceProtection  7043 non-null   object 
 12  TechSupport       7043 non-null   object 
 13  StreamingTV       7043 non-null   object 
 14  StreamingMovies   7043 non-null   object 
 15  Contract          7043 non-null   object 
 16  PaperlessBilling  7043 non-null   object 
 17  PaymentMethod     7043 non-null   object 
 18  MonthlyCharges    7043 non-null   float64
 19  TotalCharges      7043 non-null   object 
 20  Churn             7043 non-null   object 
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
```

In [5]:
```
telecom_data.shape
```

Out[5]:
```
(7043, 21)
```

In [6]:
```
telecom_data.columns
```

Out[6]:
```
Index(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport',
       'StreamingTV', 'StreamingMovies', 'Contract', 'PaperlessBilling',
       'PaymentMethod', 'MonthlyCharges', 'TotalCharges', 'Churn'],
      dtype='object')
```

In [7]:
```
telecom_data.nunique()
```

Out[7]:
```
customerID          7043
gender                 2
SeniorCitizen          2
Partner                2
Dependents             2
tenure                73
PhoneService           2
MultipleLines          3
InternetService        3
OnlineSecurity         3
OnlineBackup           3
DeviceProtection       3
TechSupport            3
StreamingTV            3
StreamingMovies        3
Contract               3
PaperlessBilling       2
PaymentMethod          4
MonthlyCharges      1585
TotalCharges        6531
Churn                  2
dtype: int64
```

In [8]:
```
telecom_data.dtypes
```

Out[8]:
```
customerID          object
gender              object
SeniorCitizen        int64
Partner             object
Dependents          object
tenure               int64
PhoneService        object
MultipleLines       object
InternetService     object
OnlineSecurity      object
OnlineBackup        object
DeviceProtection    object
TechSupport         object
StreamingTV         object
StreamingMovies     object
Contract            object
PaperlessBilling    object
PaymentMethod       object
MonthlyCharges     float64
TotalCharges        object
Churn               object
dtype: object
```

TotalCharges column dtype should be float64

In [9]:
```
telecom_data.isnull().sum()
```

Out[9]:
```
customerID          0
gender              0
SeniorCitizen       0
Partner             0
Dependents          0
tenure              0
PhoneService        0
MultipleLines       0
InternetService     0
OnlineSecurity      0
OnlineBackup        0
DeviceProtection    0
TechSupport         0
StreamingTV         0
StreamingMovies     0
Contract            0
PaperlessBilling    0
PaymentMethod       0
MonthlyCharges      0
TotalCharges        0
Churn               0
dtype: int64
```

In [10]:
```
# Convert total Charges to numeric values
telecom_data['TotalCharges'] = pd.to_numeric(telecom_data['TotalCharges'], err
```

In [11]:
```
telecom_data.isnull().sum()
```

Out[11]:
```
customerID          0
gender              0
SeniorCitizen       0
Partner             0
Dependents          0
tenure              0
PhoneService        0
MultipleLines       0
InternetService     0
OnlineSecurity      0
OnlineBackup        0
DeviceProtection    0
TechSupport         0
StreamingTV         0
StreamingMovies     0
Contract            0
PaperlessBilling    0
PaymentMethod       0
MonthlyCharges      0
TotalCharges       11
Churn               0
dtype: int64
```

There is only 11 NAN values in TotalCharges column

In [12]:
```
telecom_data[telecom_data['TotalCharges'].isnull()]
```

Out[12]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLir
488	4472-LVYGI	Female	0	Yes	Yes	0	No	No phc serv
753	3115-CZMZD	Male	0	No	Yes	0	Yes	
936	5709-LVOEQ	Female	0	Yes	Yes	0	Yes	
1082	4367-NUYAO	Male	0	Yes	Yes	0	Yes	Y
1340	1371-DWPAZ	Female	0	Yes	Yes	0	No	No phc serv
3331	7644-OMVMY	Male	0	Yes	Yes	0	Yes	
3826	3213-VVOLG	Male	0	Yes	Yes	0	Yes	Y
4380	2520-SGTTA	Female	0	Yes	Yes	0	Yes	
5218	2923-ARZLG	Male	0	Yes	Yes	0	Yes	
6670	4075-WKNIU	Female	0	Yes	Yes	0	Yes	Y
6754	2775-SEFEE	Male	0	No	Yes	0	Yes	Y

11 rows × 21 columns

In [13]:
```
telecom_data.dropna(inplace=True)
```

In [14]:
```
telecom_data.isnull().sum()
```

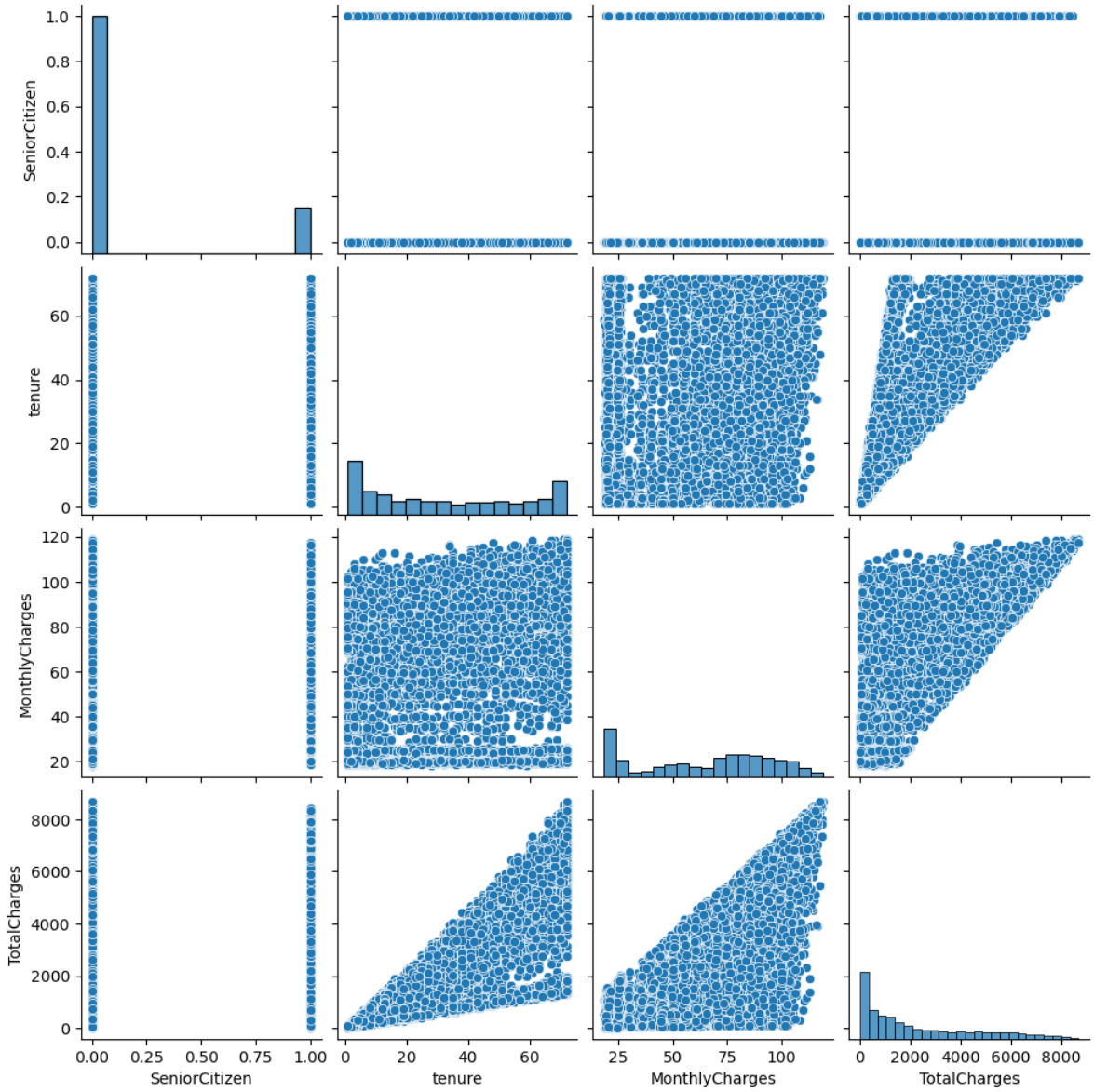
Out[14]:
```
customerID          0
gender              0
SeniorCitizen       0
Partner             0
Dependents          0
tenure              0
PhoneService        0
MultipleLines       0
InternetService     0
OnlineSecurity      0
OnlineBackup        0
DeviceProtection    0
TechSupport         0
StreamingTV         0
StreamingMovies     0
Contract            0
PaperlessBilling    0
PaymentMethod       0
MonthlyCharges      0
TotalCharges        0
Churn               0
dtype: int64
```

In [15]:
```
telecom_data.count()
```

Out[15]:
```
customerID          7032
gender              7032
SeniorCitizen       7032
Partner             7032
Dependents          7032
tenure              7032
PhoneService        7032
MultipleLines       7032
InternetService     7032
OnlineSecurity      7032
OnlineBackup        7032
DeviceProtection    7032
TechSupport         7032
StreamingTV         7032
StreamingMovies     7032
Contract            7032
PaperlessBilling    7032
PaymentMethod       7032
MonthlyCharges      7032
TotalCharges        7032
Churn               7032
dtype: int64
```

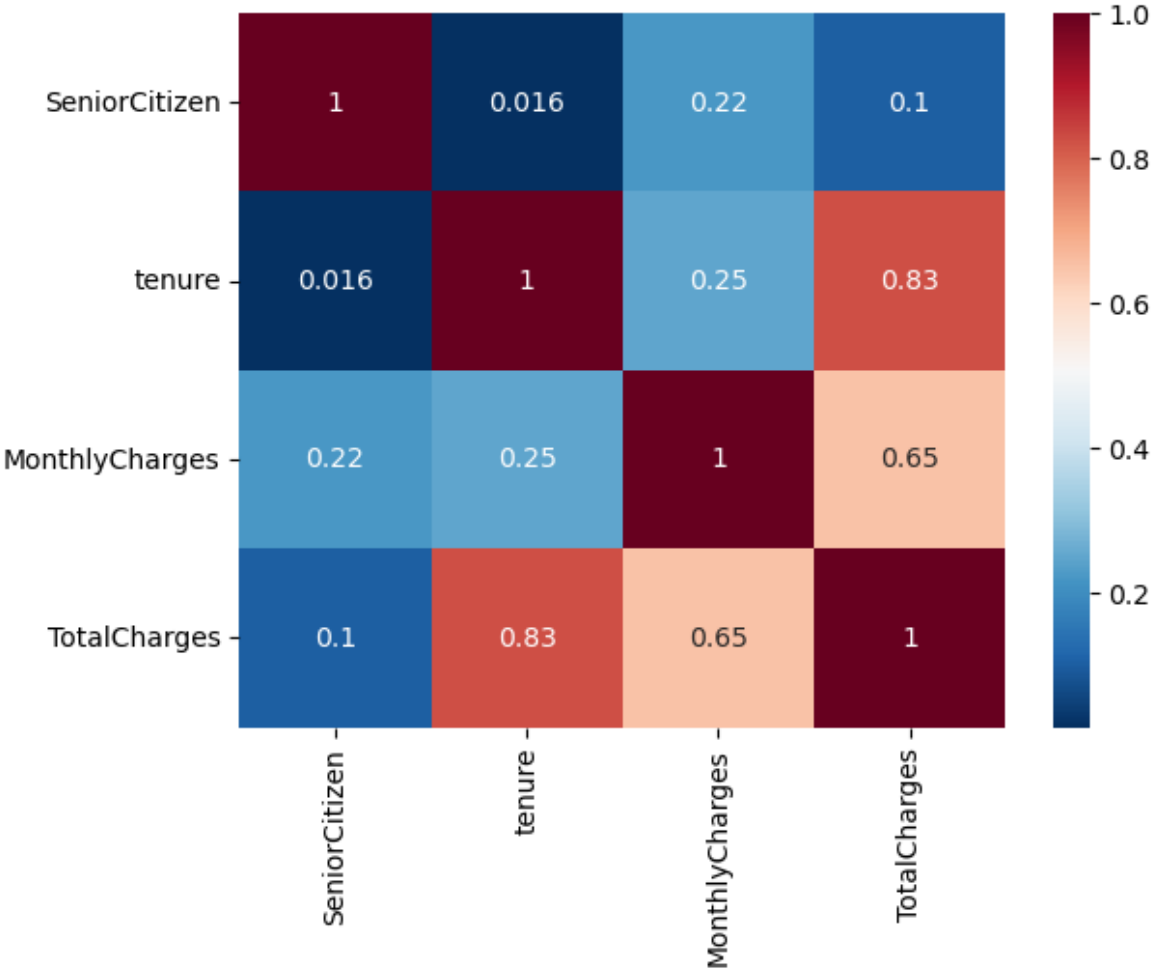
In [16]:
```
sns.pairplot(data=telecom_data)
```

Out[16]:
```
<seaborn.axisgrid.PairGrid at 0x18c6293c220>
```



In [17]:
```
sns.heatmap(data=telecom_data.corr(), annot=True, cmap='RdBu_r')
```

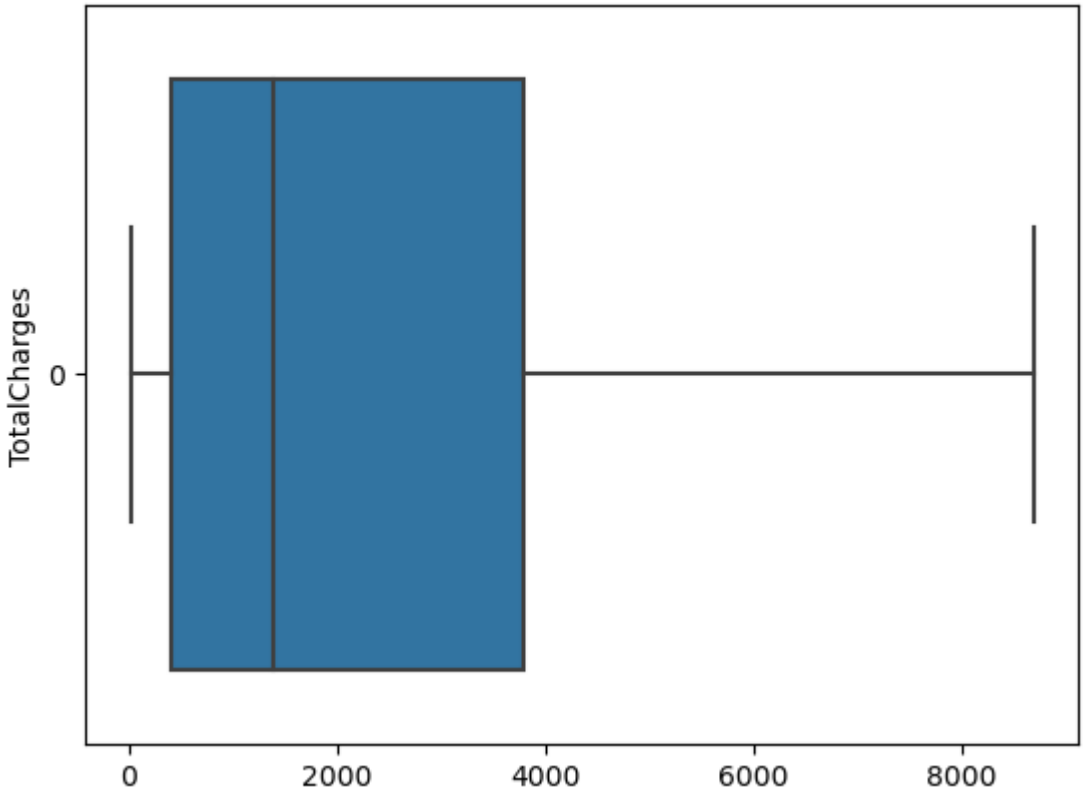
Out[17]: <AxesSubplot:>



We can understand from this that when tenture and monthlycharges increase TotalCharges increases.

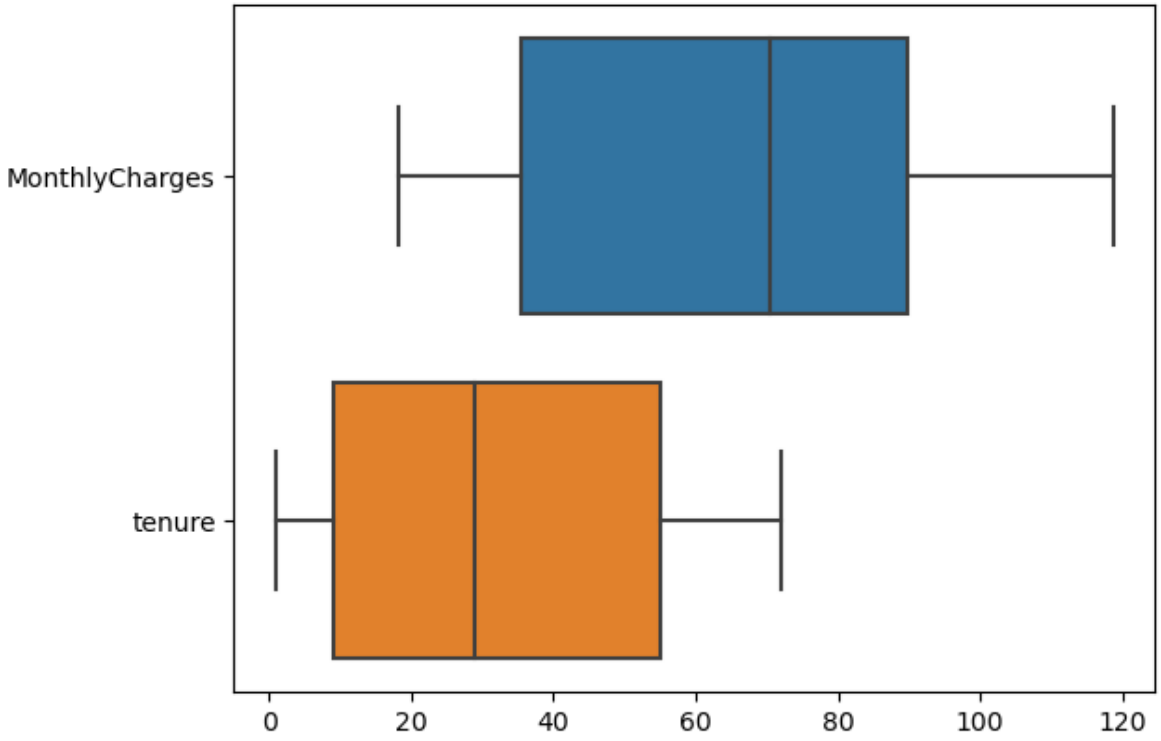
In [18]:
```
sns.boxplot(data=telecom_data['TotalCharges'], orient='h')
plt.ylabel('TotalCharges')
```

Out[18]: Text(0, 0.5, 'TotalCharges')



In [19]:
```
sns.boxplot(data=telecom_data[['MonthlyCharges', 'tenure']], orient='h')
```

Out[19]:
```
<AxesSubplot:>
```



Gender Distribution

In [20]:
```
gender = (telecom_data['gender'].value_counts() * 100) / len(telecom_data['gen
gender
```

Out[20]:
```
Male      50.469283
Female    49.530717
Name: gender, dtype: float64
```

In [21]:
```
gender.index
```

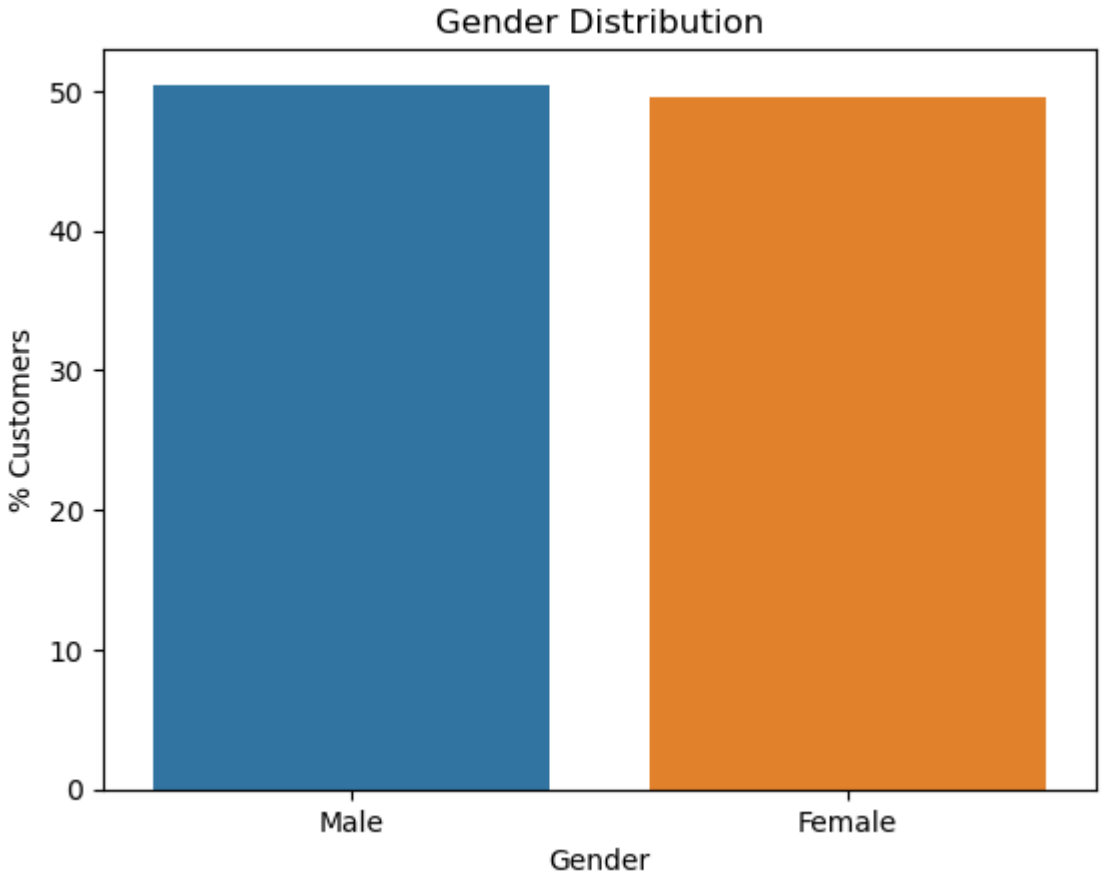
Out[21]:
```
Index(['Male', 'Female'], dtype='object')
```

In [22]:
```
gender.values
```

Out[22]:
```
array([50.46928328, 49.53071672])
```

In [23]:
```
sns.barplot(x=gender.index, y=gender.values)
plt.title('Gender Distribution')
plt.ylabel('% Customers')
plt.xlabel('Gender')
```

Out[23]:
```
Text(0.5, 0, 'Gender')
```

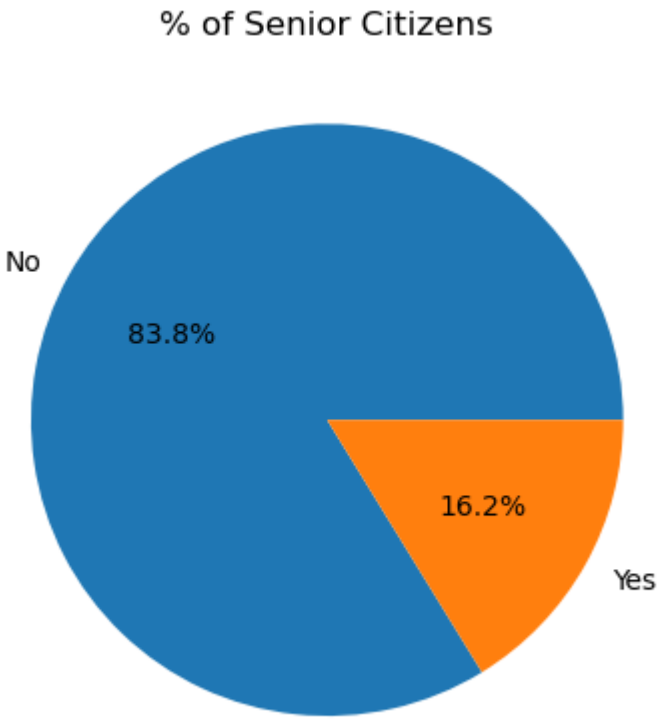


In [24]:
```
senior_citizen = (telecom_data['SeniorCitizen'].value_counts() * 100) / len(te
senior_citizen
```

Out[24]:
```
0    83.759954
1    16.240046
Name: SeniorCitizen, dtype: float64
```

In [25]:
```
plt.pie(senior_citizen, labels=['No', 'Yes'], autopct='%1.1f%%')
plt.title('% of Senior Citizens')
```

Out[25]: Text(0.5, 1.0, '% of Senior Citizens')



Dependents and parteners

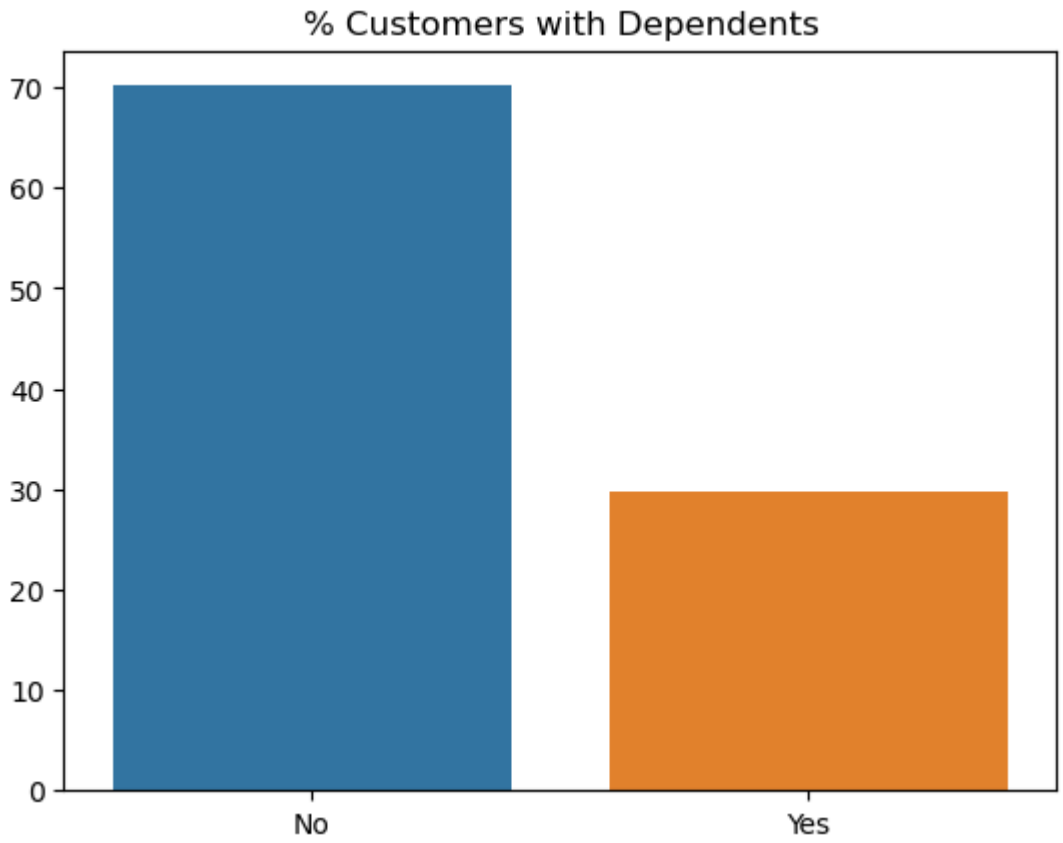
In [26]:
```
dependents = telecom_data['Dependents'].value_counts() * 100 / len(telecom_dat
print(dependents)
print("=============")

parteners = telecom_data['Partner'].value_counts() * 100 / len(telecom_data['P
print(parteners)
```

```
No     70.150739
Yes    29.849261
Name: Dependents, dtype: float64
=============
No     51.749147
Yes    48.250853
Name: Partner, dtype: float64
```

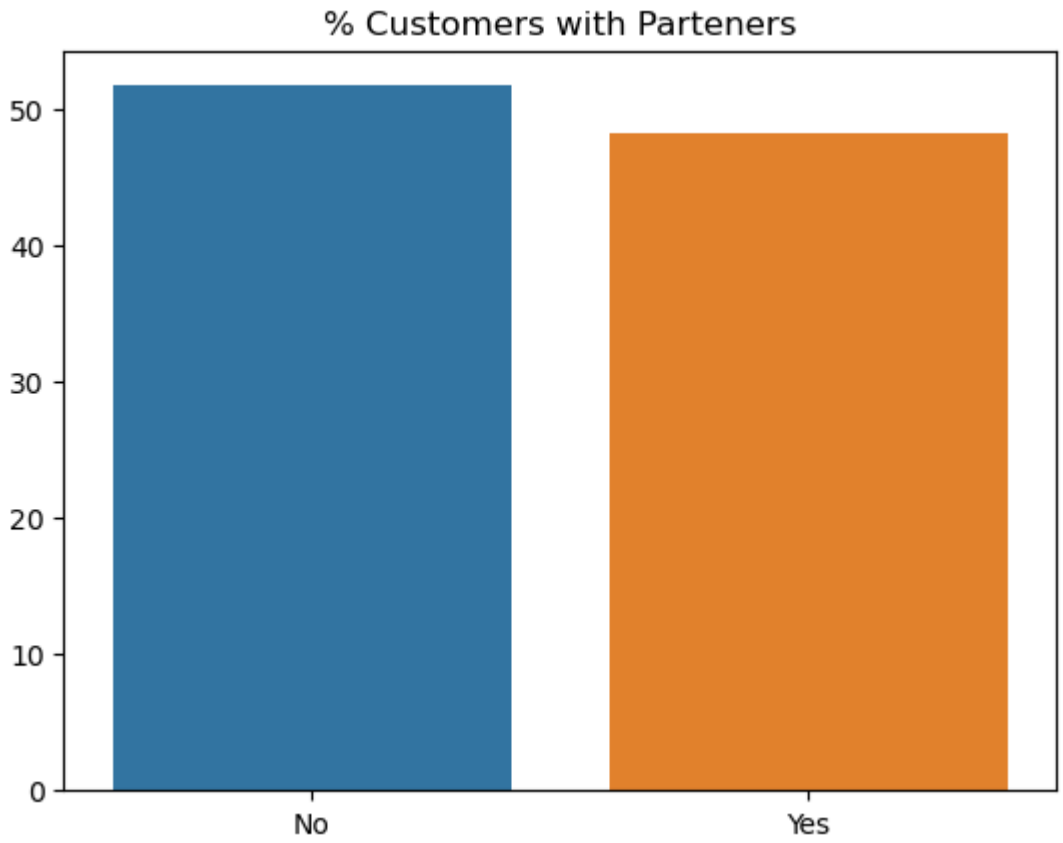
In [27]:
```
sns.barplot(x=dependents.index, y=dependents.values)
plt.title("% Customers with Dependents")
```

Out[27]:
```
Text(0.5, 1.0, '% Customers with Dependents')
```



In [28]:
```
sns.barplot(x=parteners.index, y=parteners.values)
plt.title("% Customers with Parteners")
```

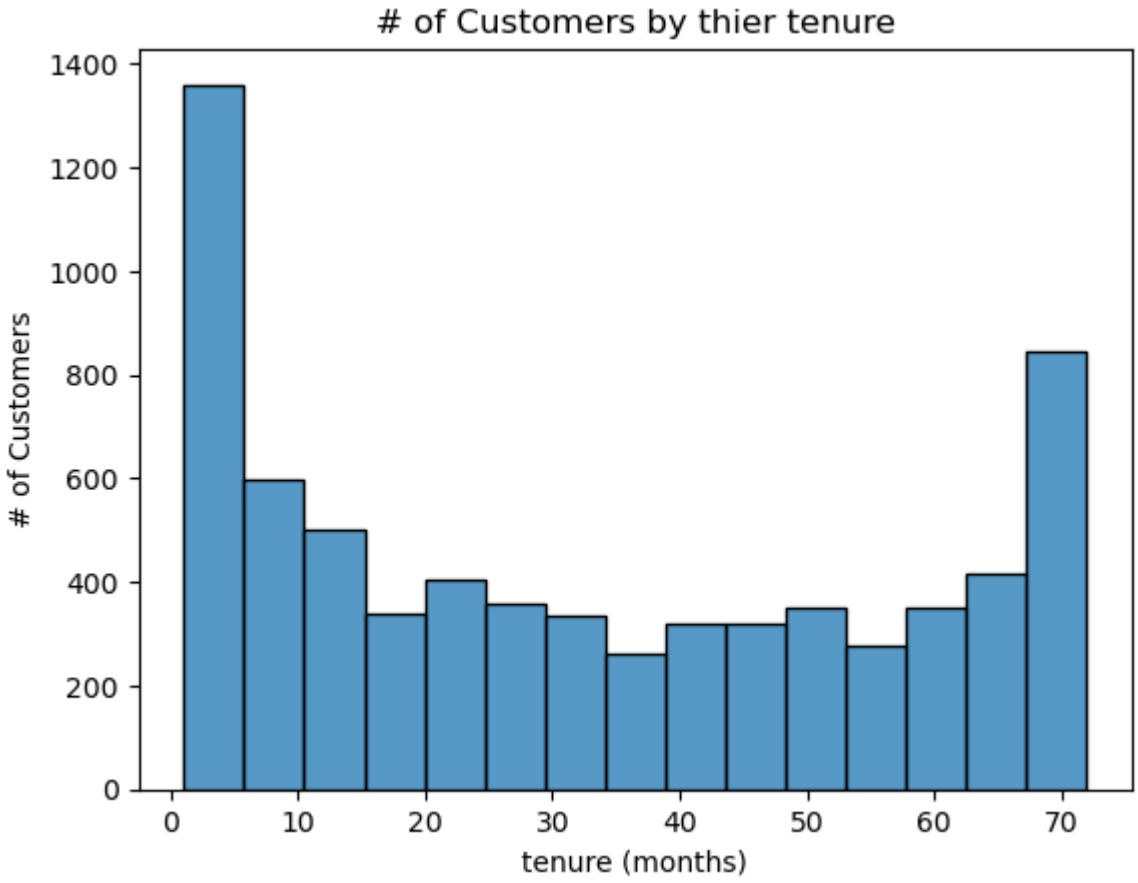
Out[28]:
```
Text(0.5, 1.0, '% Customers with Parteners')
```



Customer Account Information

In [29]:
```
sns.histplot(data=telecom_data, x='tenure')
plt.ylabel('# of Customers')
plt.xlabel('tenure (months)')
plt.title('# of Customers by thier tenure')
```

Out[29]:
```
Text(0.5, 1.0, '# of Customers by thier tenure')
```



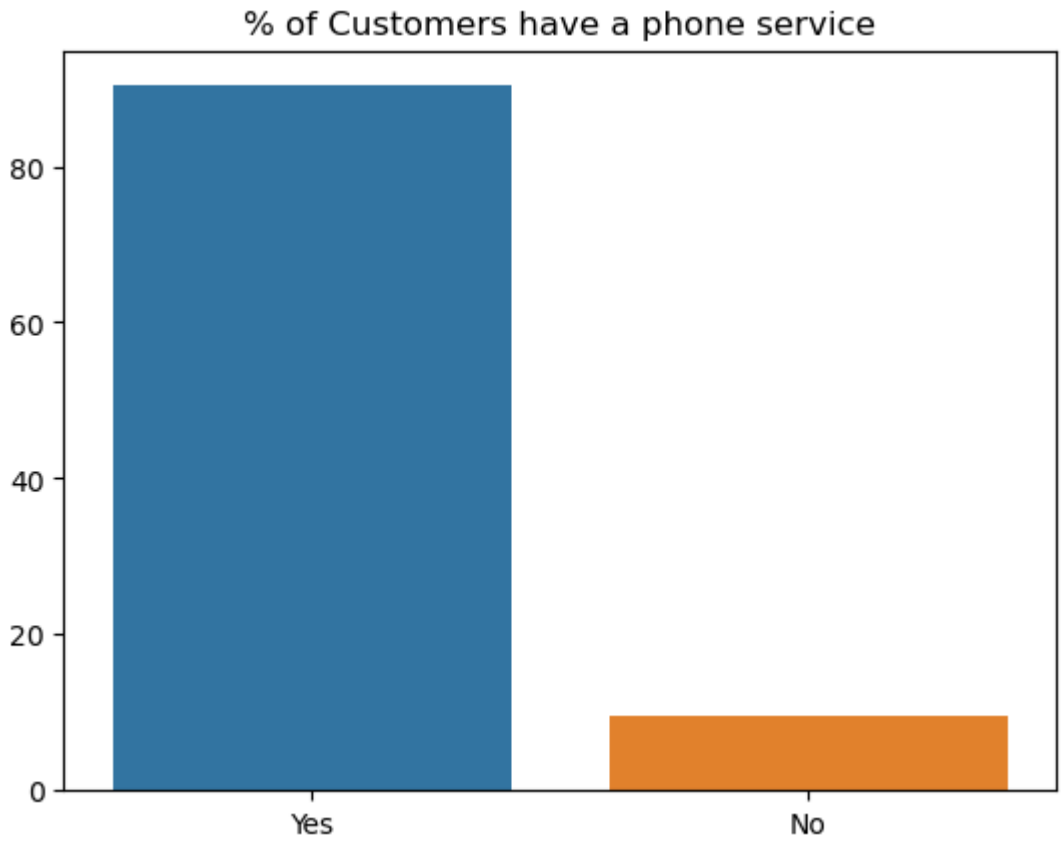
After looking to the distribution we can see that alot of customers stay with the company for a months and quiet many stay for 72 months.

In [30]:
```
phones = telecom_data['PhoneService'].value_counts() * 100 / len(telecom_data[
phones
```

Out[30]:
```
Yes    90.32992
No      9.67008
Name: PhoneService, dtype: float64
```

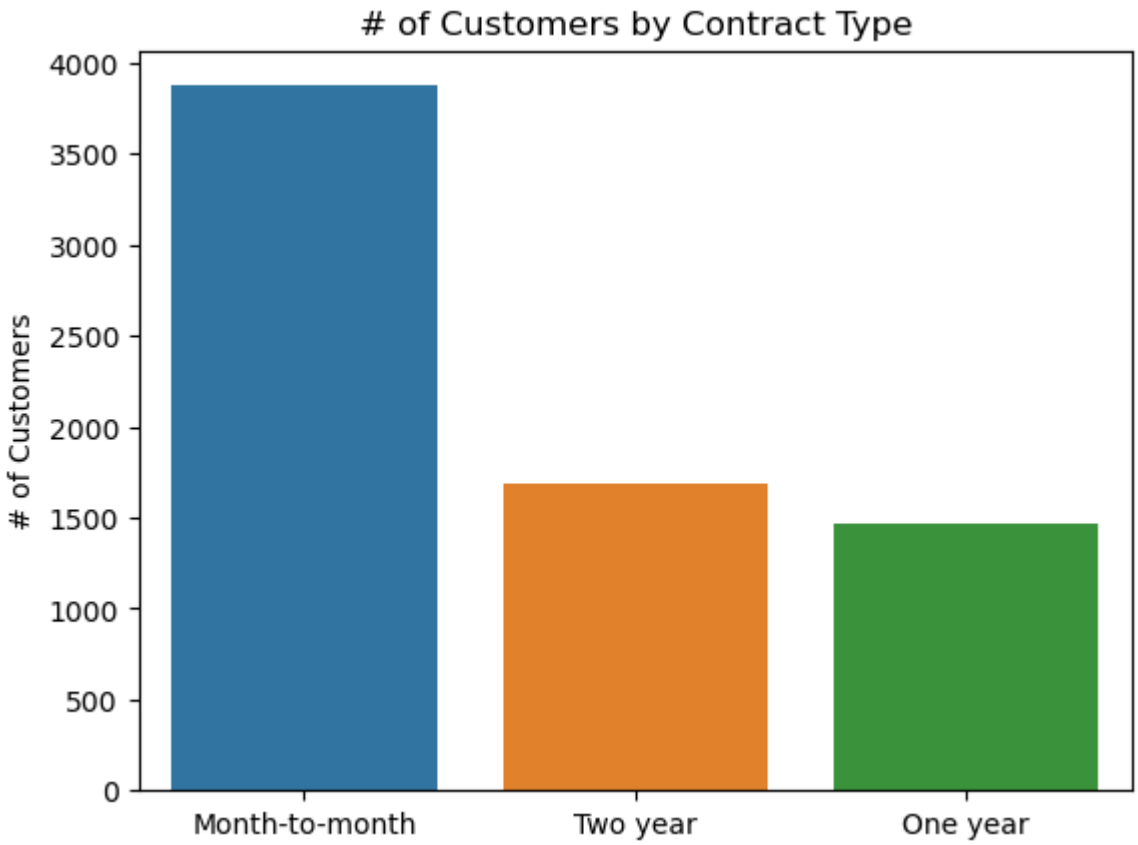
In [31]:
```
sns.barplot(x=phones.index, y=phones.values)
plt.title('% of Customers have a phone service')
```

Out[31]:
```
Text(0.5, 1.0, '% of Customers have a phone service')
```



In [32]:
```
contract = telecom_data['Contract'].value_counts()
sns.barplot(x=contract.index, y=contract.values)
plt.title('# of Customers by Contract Type')
plt.ylabel('# of Customers')
```

Out[32]:
```
Text(0, 0.5, '# of Customers')
```



As we can see most of customers are in Month-to-Month Contract.

In [33]:
```
month_to_month = telecom_data[telecom_data['Contract'] == 'Month-to-month']
one_year = telecom_data[telecom_data['Contract'] == 'One year']
two_year = telecom_data[telecom_data['Contract'] == 'Two year']
```

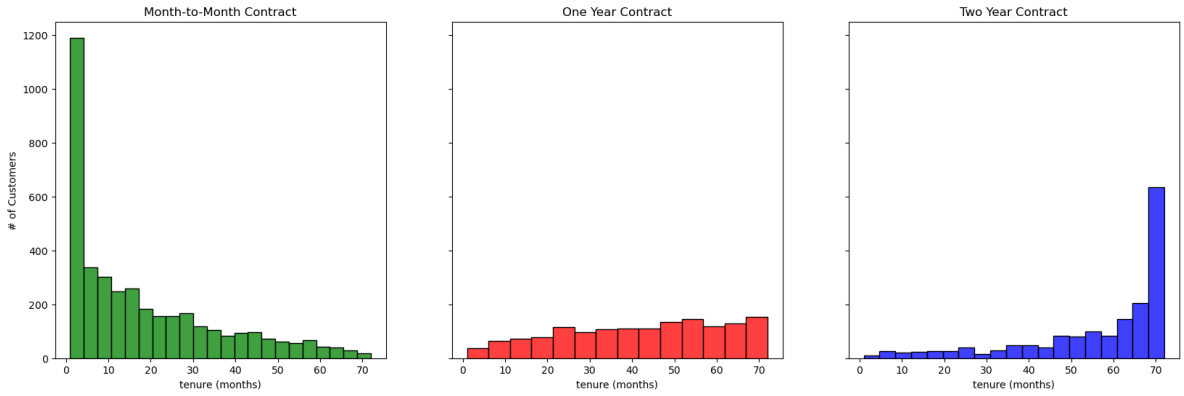
In [34]:
```
fig, (ax1, ax2, ax3) = plt.subplots(nrows=1, ncols=3, figsize=(20, 6), sharey=
ax1.set_ylabel('# of Customers')

sns.histplot(data=month_to_month, x='tenure', ax=ax1, color='green')
ax1.set_title('Month-to-Month Contract')
ax1.set_xlabel('tenure (months)')

sns.histplot(data=one_year, x='tenure', ax=ax2, color='red')
ax2.set_title('One Year Contract')
ax2.set_xlabel('tenure (months)')

sns.histplot(data=two_year, x='tenure', ax=ax3, color='blue')
ax3.set_title('Two Year Contract')
ax3.set_xlabel('tenure (months)')
```

Out[34]:
```
Text(0.5, 0, 'tenure (months)')
```



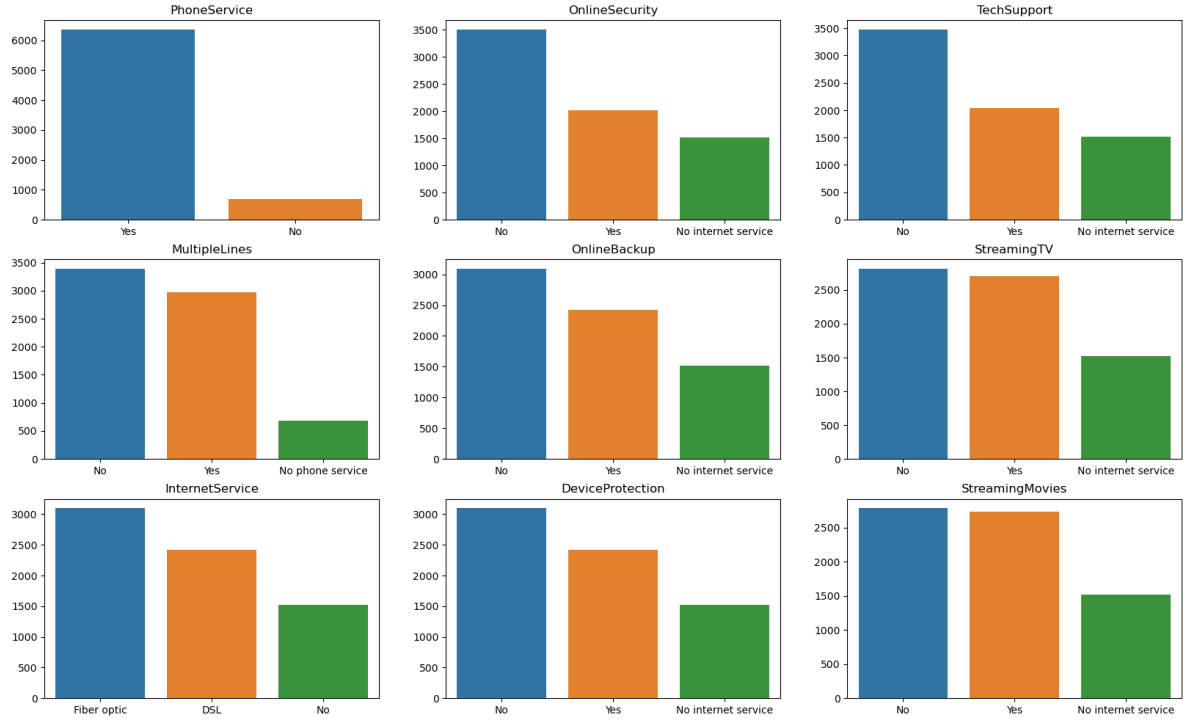
Most of the monthly contracts last for 1-2 months, while the 2 year contracts tend to last for about 70 months. This shows that the customers taking a longer contract are more loyal to the company and tend to stay with it for a longer period of time.

Various Services

In [35]:
```
services = ['PhoneService','MultipleLines','InternetService','OnlineSecurity',
            'OnlineBackup','DeviceProtection','TechSupport','StreamingTV','Stre

fig, axes = plt.subplots(nrows=3, ncols=3, figsize=(20, 12))

for i, item in enumerate(services):
    if i < 3:
        sns.barplot(x=telecom_data[item].value_counts().index, y=telecom_data[
        axes[i, 0].set_title(item)
    elif (i >= 3) and (i < 6):
        sns.barplot(x=telecom_data[item].value_counts().index, y=telecom_data[
        axes[i-3, 1].set_title(item)
    elif i < 9:
        sns.barplot(x=telecom_data[item].value_counts().index, y=telecom_data[
        axes[i-6, 2].set_title(item)
```



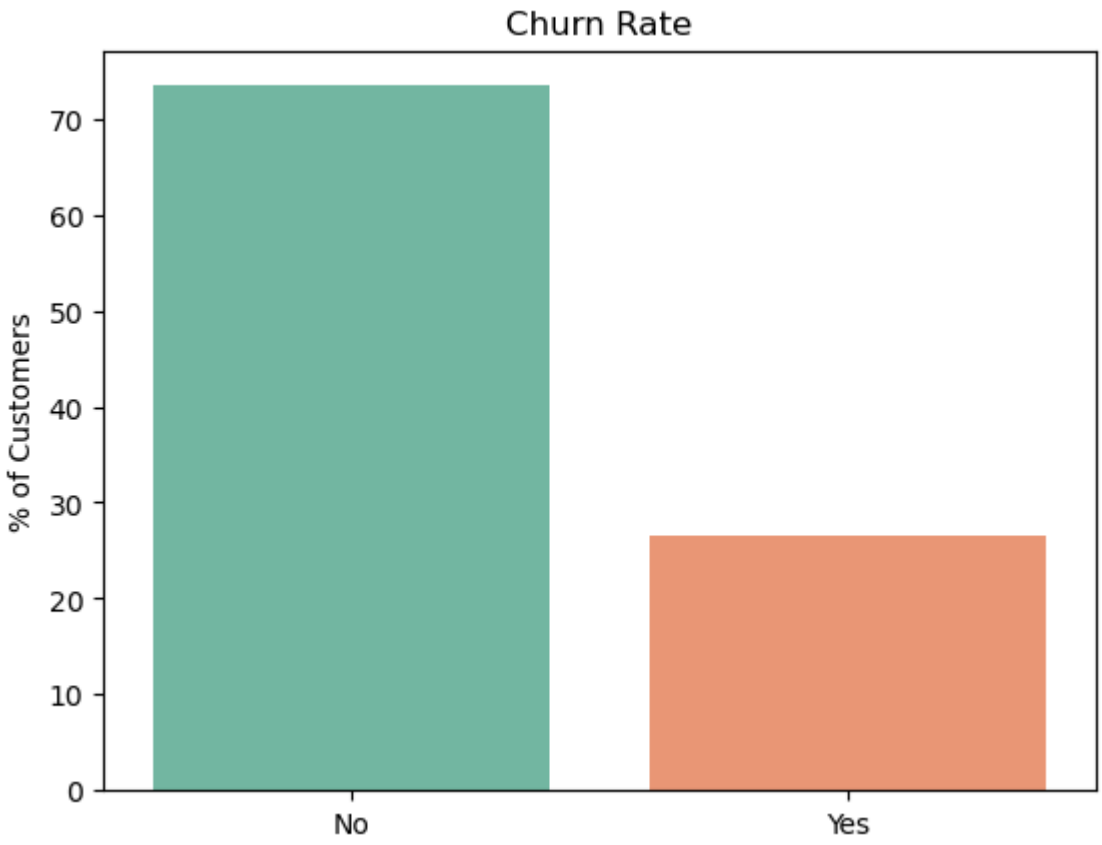
Churn

In [36]:
```
churn = telecom_data['Churn'].value_counts() * 100 / len(telecom_data['Churn']
churn
```

Out[36]:
```
No     73.421502
Yes    26.578498
Name: Churn, dtype: float64
```

In [37]:
```
sns.barplot(x=churn.index, y=churn.values, palette="Set2")
plt.title('Churn Rate')
plt.ylabel('% of Customers')
```

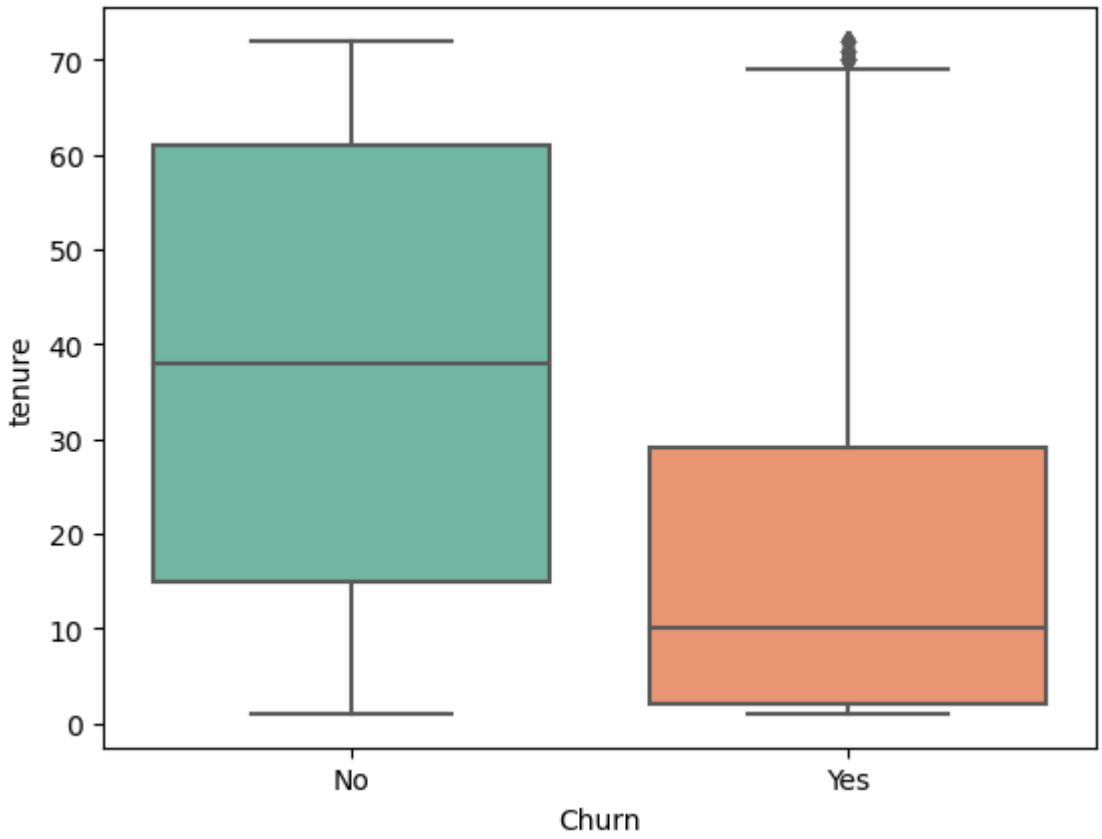
Out[37]:
```
Text(0, 0.5, '% of Customers')
```



About 74% of customers don't churn.

In [38]:
```
sns.boxplot(x=telecom_data['Churn'], y=telecom_data['tenure'], palette='Set2')
```

Out[38]:
```
<AxesSubplot:xlabel='Churn', ylabel='tenure'>
```



We can see that customers who don't churn tend to stay for a longer tenure.