In [204]:
```
import types
import os
import pandas as pd
from botocore.client import Config
import ibm_boto3
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from scipy.stats import ttest_1samp

def __iter__(self): return 0


# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It incl
udes your credentials.
# You might want to remove those credentials before you share the notebook.

if os.environ.get('RUNTIME_ENV_LOCATION_TYPE') == 'external':
    endpoint_931d06c29a624816bb257512bafae77d = 'https://s3-api.us-geo.objects
torage.softlayer.net'
else:
    endpoint_931d06c29a624816bb257512bafae77d = 'https://s3-api.us-geo.objects
torage.service.networklayer.com'

client_931d06c29a624816bb257512bafae77d = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='CLHCS70tLsqmEcMAdIhGSZm4Jbxe096F6ncLB7vJaooa',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url=endpoint_931d06c29a624816bb257512bafae77d)

body = client_931d06c29a624816bb257512bafae77d.get_object(Bucket='telcocustome
rchurn-donotdelete-pr-h6ed2qaxbiolqe',Key='WA_Fn-UseC_-Telco-Customer-Churn_mo
dified.csv')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__,
body )

df_data_3 = pd.read_csv(body)
df_data_3.head()

data = df_data_3
data.head()
```

Out[204]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines
0	0002-ORFBO	Female	0	Yes	Yes	9	Yes	No
1	0003-MKNFE	Male	0	No	No	9	Yes	Yes
2	0004-TLHLJ	Male	0	No	No	4	Yes	No
3	0011-IGKFF	Male	1	Yes	No	13	Yes	No
4	0013-EXCHZ	Female	1	Yes	No	3	Yes	No

In [85]:
```
data.iloc[0]
```

Out[85]:
```
customerID            0002-ORFBO
gender                    Female
SeniorCitizen                  0
Partner                      Yes
Dependents                   Yes
tenure                         9
PhoneService                 Yes
MultipleLines                 No
InternetService              DSL
OnlineSecurity                No
OnlineBackup                 Yes
DeviceProtection              No
TechSupport                  Yes
StreamingTV                  Yes
StreamingMovies               No
Contract                One year
PaperlessBilling             Yes
PaymentMethod       Mailed check
MonthlyCharges              65.6
TotalCharges               593.3
Churn                         No
Name: 0, dtype: object
```

In [86]:
```
len(data)
```

Out[86]:
```
7043
```

In [87]:
```
data.shape
```

Out[87]:
```
(7043, 21)
```

In [88]:
```
data.columns
```

Out[88]:
```
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 [89]:
```
data.dtypes
```

Out[89]:
```
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        float64
Churn                object
dtype: object
```

In [90]:
```
data[data.isnull().any(axis=1)]
```

Out[90]:

customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines

In [91]:
```
### BEGIN SOLUTION
# Number of rows
print(data.shape[0])

# Column names
print(data.columns.tolist())

# Data types
print(data.dtypes)
### END SOLUTION
```

```
7043
['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents', 'tenure',
'PhoneService', 'MultipleLines', 'InternetService', 'OnlineSecurity', 'Online
Backup', 'DeviceProtection', 'TechSupport', 'StreamingTV', 'StreamingMovies',
'Contract', 'PaperlessBilling', 'PaymentMethod', 'MonthlyCharges', 'TotalChar
ges', 'Churn']
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        float64
Churn                object
dtype: object
```

In [94]:
```
data['TotalCharges'] = data['TotalCharges'].astype(float)
```

In [95]:
```
data.gender.value_counts()
```

Out[95]:
```
Male      3555
Female    3488
Name: gender, dtype: int64
```

In [1]:
```
#data.tenure.value_counts()
```

In [97]:
```
data["tenure"].mean()
```

Out[97]:
```
32.37114865824223
```

In [98]:
```
data["tenure"].median()
```

Out[98]:
```
29.0
```

In [99]:
```
data.SeniorCitizen.value_counts()
```

Out[99]:
```
0    5901
1    1142
Name: SeniorCitizen, dtype: int64
```

In [100]:
```
data.Partner.value_counts()
```

Out[100]:
```
No     3641
Yes    3402
Name: Partner, dtype: int64
```

In [101]:
```
data.Dependents.value_counts()
```

Out[101]:
```
No     4933
Yes    2110
Name: Dependents, dtype: int64
```

In [102]:
```
dps = data.PhoneService.value_counts()
print('Phone Service =', dps)
print('\n')
```

```
Phone Service = Yes    6361
No      682
Name: PhoneService, dtype: int64
```

In [103]:
```
data.MultipleLines.value_counts()
```

Out[103]:
```
No                  3390
Yes                 2971
No phone service     682
Name: MultipleLines, dtype: int64
```

In [104]:
```
dos = data.OnlineSecurity.value_counts()
dob = data.OnlineBackup.value_counts()
dpd = data.DeviceProtection.value_counts()
dts = data.TechSupport.value_counts()
dstv = data.StreamingTV.value_counts()

print(dos)
print('\n')
print(dob)
print('\n')
print(dpd)
print('\n')
print(dts)
print('\n')
print(dstv)
print('\n')
```

```
No                     3498
Yes                    2019
No internet service    1526
Name: OnlineSecurity, dtype: int64


No                     3088
Yes                    2429
No internet service    1526
Name: OnlineBackup, dtype: int64


No                     3095
Yes                    2422
No internet service    1526
Name: DeviceProtection, dtype: int64


No                     3473
Yes                    2044
No internet service    1526
Name: TechSupport, dtype: int64


No                     2810
Yes                    2707
No internet service    1526
Name: StreamingTV, dtype: int64
```

In [105]:
```
data.StreamingMovies.value_counts()
```

Out[105]:
```
No                     2785
Yes                    2732
No internet service    1526
Name: StreamingMovies, dtype: int64
```

In [16]:
```
ddc = data.Contract.value_counts()
#print('Data for Contract is:\n', ddc)
#print('\n')
#
dpb = data.PaperlessBilling.value_counts()
#print('Data for Paperless Billing is:\n', dpb)
#print('\n')

dpm = data.PaymentMethod.value_counts()
#print('Data for Payment Method is:\n', dpm)
#print('\n')


dmc = data.MonthlyCharges.value_counts()
#print('Data for Monthly Charges is:\n', dmc)
#print('\n')
dtc = data.TotalCharges.value_counts()
#print('Data for Total Charges is:\n', dtc)
#print('\n')
```

```
Error in data.Contract.value_counts(): could not find function "data.Contrac
t.value_counts"
Traceback:
```

In [107]:
```
data["MonthlyCharges"].mean()
```

Out[107]:
```
64.76169246059918
```

In [108]:
```
data["MonthlyCharges"].median()
```

Out[108]:
```
70.35
```

In [109]:
```
data["TotalCharges"].mean()
```

Out[109]:
```
2279.7343035638223
```

In [110]:
```
data["TotalCharges"].median()
```

Out[110]:
```
1394.55
```

In [111]:
```
data.dtypes
```

Out[111]:
```
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        float64
Churn                object
dtype: object
```

In [112]:
```
data['MonthlyCharges'].dtypes
```

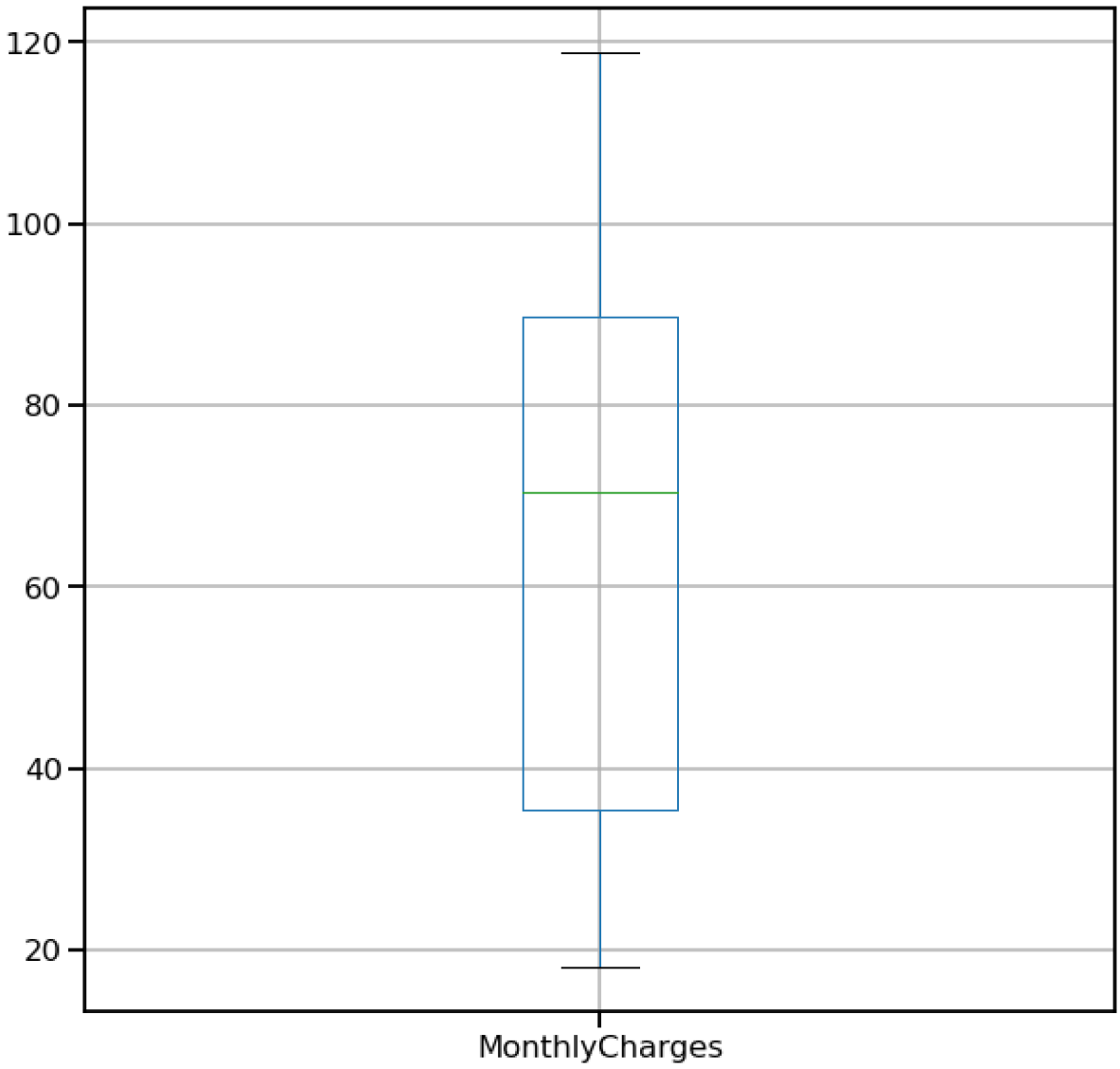
Out[112]:
```
dtype('float64')
```

In [113]:
```
data['TotalCharges'].dtypes
```

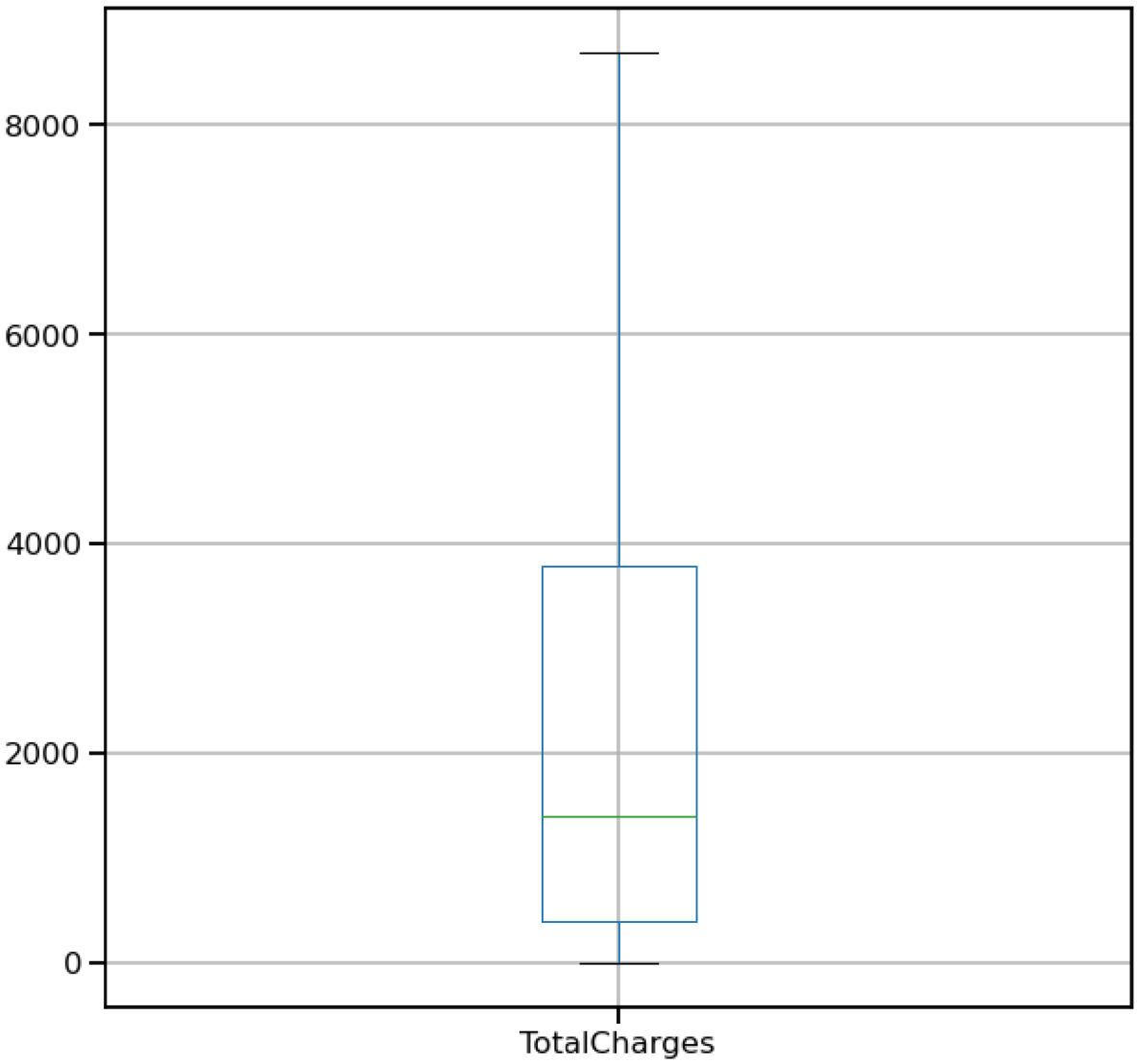
Out[113]:
```
dtype('float64')
```

In [114]:
```
#data['TotalCharges'] = data['TotalCharges'].astype(int)
#data['TotalCharges'] = pd.to_numeric(data['TotalCharges'])
```

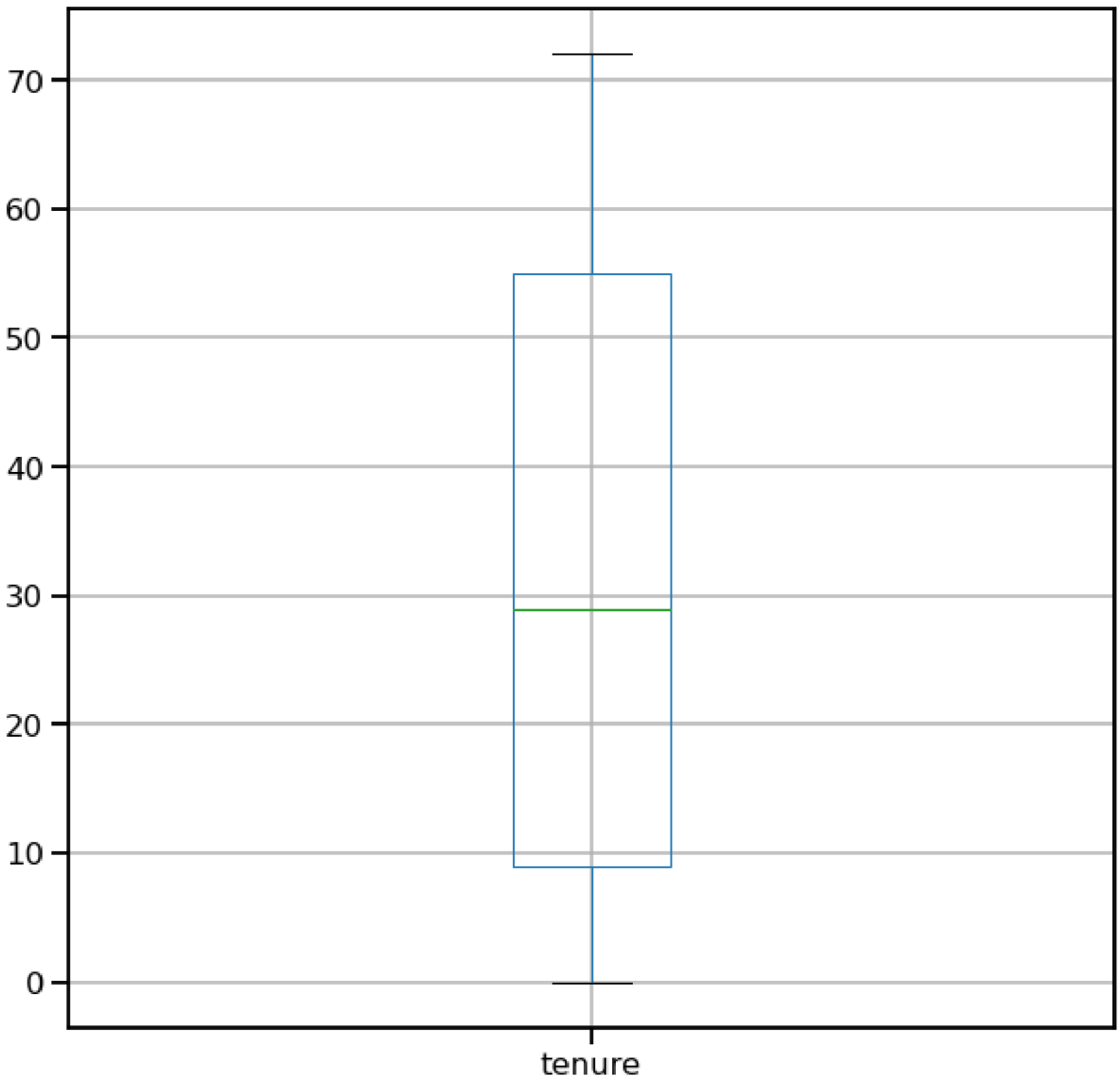
In [115]:
```
data.boxplot('MonthlyCharges', figsize = (10,10));
```



In [116]:
```
data.boxplot('TotalCharges', figsize = (10,10));
```



In [117]:
```
data.boxplot('tenure', figsize = (10,10));
```



In [118]:
```
data.head(5)
```

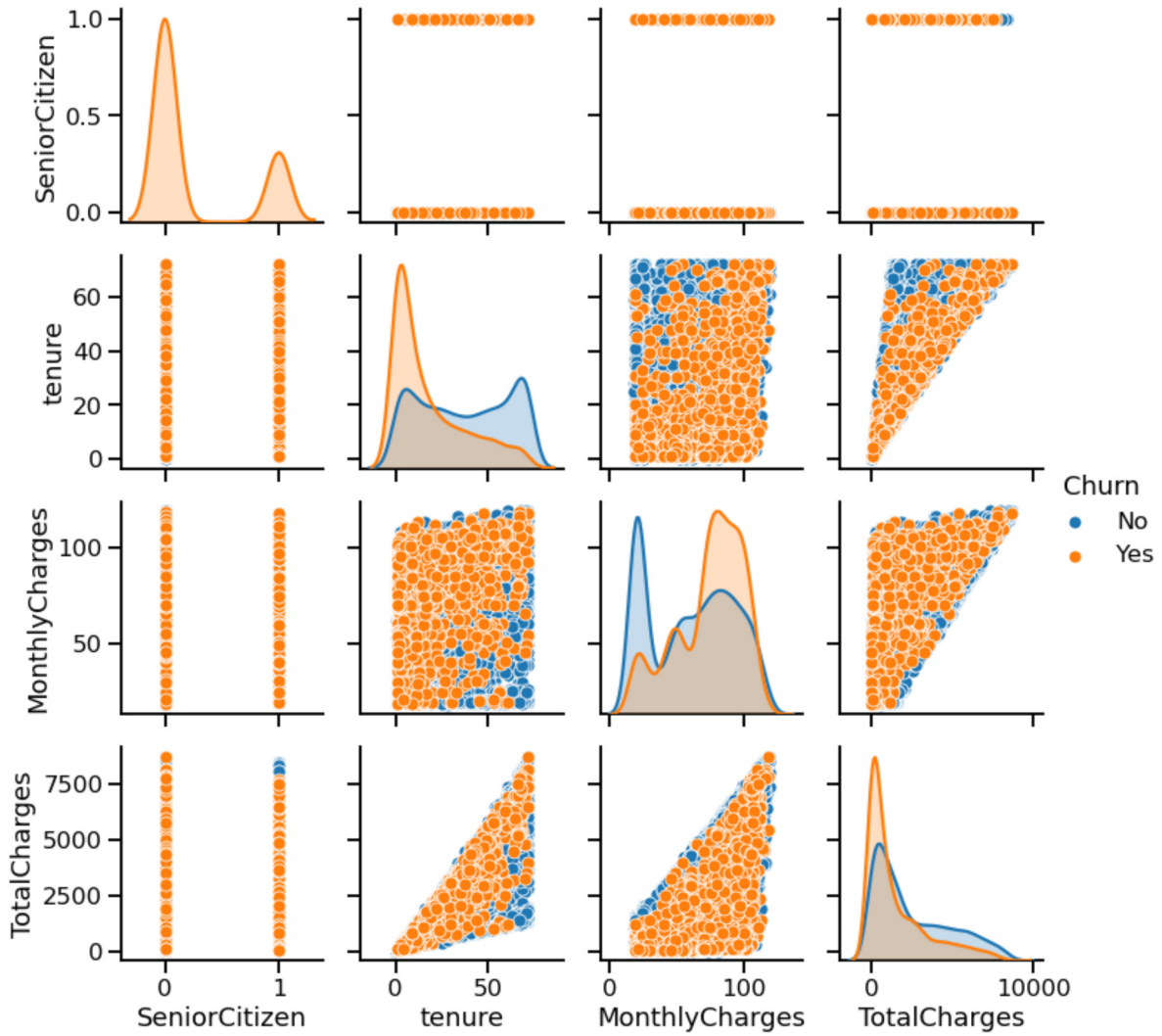
Out[118]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines
0	0002-ORFBO	Female	0	Yes	Yes	9	Yes	No
1	0003-MKNFE	Male	0	No	No	9	Yes	Yes
2	0004-TLHLJ	Male	0	No	No	4	Yes	No
3	0011-IGKFF	Male	1	Yes	No	13	Yes	No
4	0013-EXCHZ	Female	1	Yes	No	3	Yes	No

In [2]:
```
#data.set_index('Churn').head
#Leave off the .head for the full dataset
```

In [120]:
```
sns.set_context('talk')
sns.pairplot(data, hue='Churn');
```

```
/opt/conda/envs/Python-3.7-main/lib/python3.7/site-packages/seaborn/distribut
ions.py:369: UserWarning: Default bandwidth for data is 0; skipping density e
stimation.
  warnings.warn(msg, UserWarning)
```



In [121]:
```
#mean for tenure
mu = (data.mean()['tenure'])
print(mu)
```

```
32.37114865824223
```

In [122]:
```
#variance for tenure
print(data.var()['tenure'])
```

```
603.1681081237368
```

In [123]:
```
#standard deviation for tenure
stdTenure = data.std()['tenure']
print(data.std()['tenure'])
```

```
24.55948102309446
```

In [124]:
```
#mean,variance, standard deviation for monthly charges
muMC = (data.mean()['MonthlyCharges'])
varianceData = data.var()['MonthlyCharges']
stdDevData = data.std()['MonthlyCharges']
```

In [125]:
```
print(muMC)
print(varianceData)
print(stdDevData)
```

```
64.76169246059918
905.4109343405098
30.090047097678493
```

In [126]:
```
#mean,variance, standard deviation for total charges
muTC = (data.mean()['TotalCharges'])
varianceDataTC = data.var()['TotalCharges']
stdDevDataTC = data.std()['TotalCharges']
```

In [127]:
```
print(muTC)
print(varianceDataTC)
print(stdDevDataTC)
```

```
2279.7343035638223
5138357.167812732
2266.7944696890213
```

In [128]:
```
#calculate the interval within which 95% of the observations would be expected
to occur.
interTenureLower = mu - (2*stdTenure)
interTenureUpper = mu + (2*stdTenure)
if interTenureLower < 0:
    interTenureLower = 0
print('interval is: ', interTenureLower, ' < ', mu, ' < ', interTenureUpper)
```

```
interval is:  0  <  32.37114865824223  <  81.49011070443115
```

In [129]:
```
#calculate the interval within which 95% of the observations would be expected
to occur.
interLowerMC = muMC - (2*stdDevData)
interUpperMC = muMC + (2*stdDevData)
if interLowerMC < 0:
    interLowerMC = 0
print('interval is: ', interLowerMC, ' < ', muMC, ' < ', interUpperMC)
```

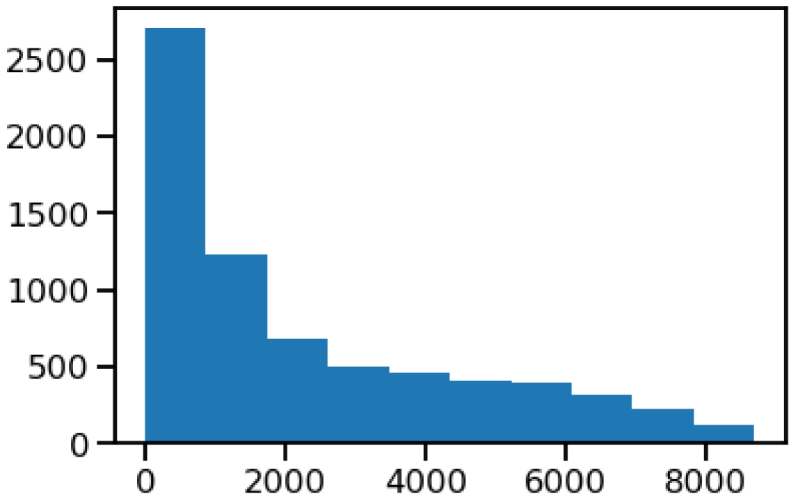
```
interval is:  4.581598265242192  <  64.76169246059918  <  124.94178665595616
```

In [130]:
```
#calculate the interval within which 95% of the observations would be expected
to occur.
interLowerTC = muTC - (2*stdDevDataTC)
interUpperTC = muTC + (2*stdDevDataTC)
if interLowerTC < 0:
    interLowerTC = 0
print('interval is: ', interLowerTC, ' < ', muTC, ' < ', interUpperTC)
```

```
interval is:  0  <  2279.7343035638223  <  6813.323242941865
```

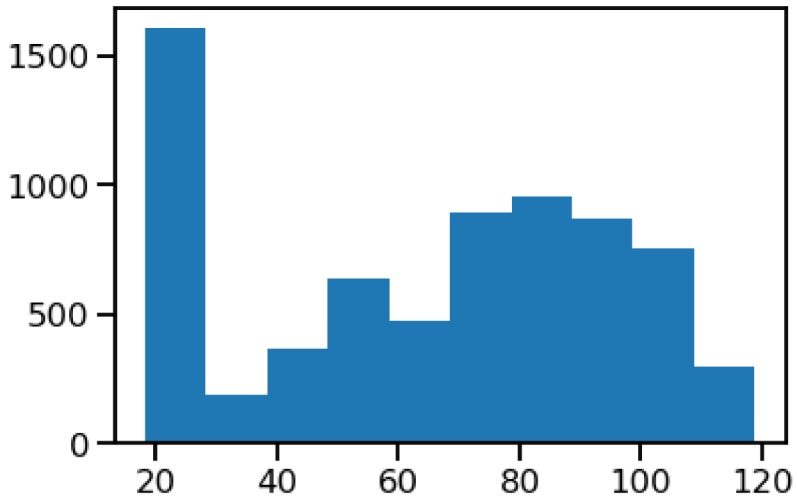
In [131]:
```
plt.hist(data.TotalCharges)
```

Out[131]:
```
(array([2701., 1227.,  685.,  503.,  460.,  414.,  396.,  311.,  224.,
         122.]),
 array([   0.  ,  868.48, 1736.96, 2605.44, 3473.92, 4342.4 , 5210.88,
        6079.36, 6947.84, 7816.32, 8684.8 ]),
 <a list of 10 Patch objects>)
```



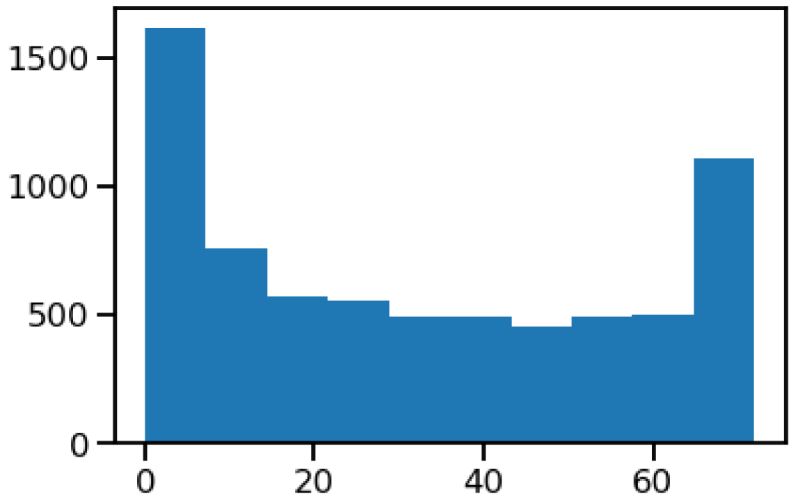
In [132]:
```
plt.hist(data.MonthlyCharges)
```

Out[132]:
```
(array([1606.,  191.,  365.,  639.,  473.,  895.,  953.,  869.,  758.,
         294.]),
 array([ 18.25,  28.3 ,  38.35,  48.4 ,  58.45,  68.5 ,  78.55,  88.6 ,
         98.65, 108.7 , 118.75]),
 <a list of 10 Patch objects>)
```



In [133]:
```
plt.hist(data.tenure)
```

Out[133]:
```
(array([1612.,  759.,  570.,  556.,  495.,  494.,  452.,  495.,  501.,
        1109.]),
 array([ 0. ,  7.2, 14.4, 21.6, 28.8, 36. , 43.2, 50.4, 57.6, 64.8, 72. ]),
 <a list of 10 Patch objects>)
```



In [134]:
```
data["Partner"].value_counts()
```

Out[134]:
```
No     3641
Yes    3402
Name: Partner, dtype: int64
```

In [135]:
```
#convert Partner from object to category then assign numerical values to Partner_category
data["Partner_category"] = data["Partner"].astype('category')
data.dtypes
```

Out[135]:
```
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        float64
Churn                object
Partner_category   category
dtype: object
```

In [136]:
```
data["Dependents"] = data["Dependents"].astype('category')
data["Dependents_category"] = data["Dependents"].cat.codes

data["MultipleLines"] = data["MultipleLines"].astype('category')
data["MultipleLines_category"] = data["MultipleLines"].cat.codes

data["PhoneService"] = data["PhoneService"].astype('category')
data["PhoneService_category"] = data["PhoneService"].cat.codes
data.head()
```

Out[136]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines
0	0002-ORFBO	Female	0	Yes	Yes	9	Yes	No
1	0003-MKNFE	Male	0	No	No	9	Yes	Yes
2	0004-TLHLJ	Male	0	No	No	4	Yes	No
3	0011-IGKFF	Male	1	Yes	No	13	Yes	No
4	0013-EXCHZ	Female	1	Yes	No	3	Yes	No

In [137]:
```
data["PhoneService_category"].value_counts()
```

Out[137]:
```
1    6361
0     682
Name: PhoneService_category, dtype: int64
```

In [17]:
```
#data["MonthlyCharges"].value_counts()
```

In [139]:
```
data["MultipleLines_category"].value_counts()
```

Out[139]:
```
0    3390
2    2971
1     682
Name: MultipleLines_category, dtype: int64
```

In [140]:
```
#one hot encoding Pandas supports this feature using get_dummies.
data_HotEncoded1 = pd.get_dummies(data, columns=["InternetService"])
```

In [18]:
```
#print(data_HotEncoded1)
```

In [19]:
```
#data_HotEncoded1.dtypes
```

In [20]:
```
#data.dtypes
```

In [11]:
```
#print(data_HotEncoded1['MonthlyCharges'])
```

In [12]:
```
#data["DeviceProtection"] = data["DeviceProtection"].astype('category')
#data["DeviceProtection_category"] = data["DeviceProtection"].cat.codes
#data.head()
```

In [13]:
```
#pd.set_option('display.max_rows', 50)
#print(data)
```

In [147]:
```
data["DeviceProtection_category"].value_counts()
```

Out[147]:
```
0    3095
2    2422
1    1526
Name: DeviceProtection_category, dtype: int64
```

In [148]:
```
#one hot encoding Pandas supports this feature using get_dummies.
data_HotEncoded1 = pd.get_dummies(data_HotEncoded1, columns=["TechSupport"])
```

In [4]:
```
#print(data_HotEncoded1)
```

In [150]:
```
#one hot encoding Pandas supports this feature using get_dummies.
data_HotEncoded1 = pd.get_dummies(data_HotEncoded1, columns=["StreamingTV"])
```

In [151]:
```
#one hot encoding Pandas supports this feature using get_dummies.
data_HotEncoded1 = pd.get_dummies(data_HotEncoded1, columns=["StreamingMovies"
])
```

In [152]:
```
#one hot encoding Pandas supports this feature using get_dummies.
data_HotEncoded1 = pd.get_dummies(data_HotEncoded1, columns=["Contract"])
```

In [153]:
```
print(data_HotEncoded1)
```

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	\
0	0002-ORFBO	Female	0	Yes	Yes	9	
1	0003-MKNFE	Male	0	No	No	9	
2	0004-TLHLJ	Male	0	No	No	4	
3	0011-IGKFF	Male	1	Yes	No	13	
4	0013-EXCHZ	Female	1	Yes	No	3	
...	...	...	...	...	...	...	
7038	9987-LUTYD	Female	0	No	No	13	
7039	9992-RRAMN	Male	0	Yes	No	22	
7040	9992-UJOEL	Male	0	No	No	2	
7041	9993-LHIEB	Male	0	Yes	Yes	67	
7042	9995-HOTOH	Male	0	Yes	Yes	63	

	PhoneService	MultipleLines	OnlineSecurity	OnlineBackup	\
0	Yes	No	No	Yes	
1	Yes	Yes	No	No	
2	Yes	No	No	No	
3	Yes	No	No	Yes	
4	Yes	No	No	No	
...	...	...	...	...	
7038	Yes	No	Yes	No	
7039	Yes	Yes	No	No	
7040	Yes	No	No	Yes	
7041	Yes	No	Yes	No	
7042	No	No phone service	Yes	Yes	

	DeviceProtection	PaperlessBilling	PaymentMethod	MonthlyCharges	\
0	No	Yes	Mailed check	65.60	
1	No	No	Mailed check	59.90	
2	Yes	Yes	Electronic check	73.90	
3	Yes	Yes	Electronic check	98.00	
4	No	Yes	Mailed check	83.90	
...	...	...	...	...	
7038	No	No	Mailed check	55.15	
7039	No	Yes	Electronic check	85.10	
7040	No	Yes	Mailed check	50.30	
7041	Yes	No	Mailed check	67.85	
7042	Yes	No	Electronic check	59.00	

	TotalCharges	Churn	Partner_category	Dependents_category	\
0	593.30	No	Yes	1	
1	542.40	No	No	0	
2	280.85	Yes	No	0	
3	1237.85	Yes	Yes	0	
4	267.40	Yes	Yes	0	
...	...	...	...	...	
7038	742.90	No	No	0	
7039	1873.70	Yes	Yes	0	
7040	92.75	No	No	0	
7041	4627.65	No	Yes	1	
7042	3707.60	No	Yes	1	

	MultipleLines_category	PhoneService_category	InternetService_DSL	\
0	0	1	1	
1	2	1	1	

```
2                          0                      1                     0
3                          0                      1                     0
4                          0                      1                     0
...                      ...                    ...                   ...
7038                       0                      1                     1
7039                       2                      1                     0
7040                       0                      1                     1
7041                       0                      1                     1
7042                       1                      0                     1

      InternetService_Fiber optic  InternetService_No  TechSupport_No  \
0                                0                   0               0
1                                0                   0               1
2                                1                   0               1
3                                1                   0               1
4                                1                   0               0
...                            ...                 ...             ...
7038                             0                   0               0
7039                             1                   0               1
7040                             0                   0               1
7041                             0                   0               0
7042                             0                   0               1

      TechSupport_No internet service  TechSupport_Yes  StreamingTV_No  \
0                                   0                1               0
1                                   0                0               1
2                                   0                0               1
3                                   0                0               0
4                                   0                1               0
...                               ...              ...             ...
7038                                0                1               1
7039                                0                0               1
7040                                0                0               1
7041                                0                1               1
7042                                0                0               0

      StreamingTV_No internet service  StreamingTV_Yes  StreamingMovies_No  \
0                                   0                1                   1
1                                   0                0                   0
2                                   0                0                   1
3                                   0                1                   0
4                                   0                1                   1
...                               ...              ...                 ...
7038                                0                0                   1
7039                                0                0                   0
7040                                0                0                   1
7041                                0                0                   0
7042                                0                1                   0

      StreamingMovies_No internet service  StreamingMovies_Yes  \
0                                       0                    0
1                                       0                    1
2                                       0                    0
3                                       0                    1
4                                       0                    0
...                                   ...                  ...
```

```
7038                                    0                    0
7039                                    0                    1
7040                                    0                    0
7041                                    0                    1
7042                                    0                    1

      Contract_Month-to-month  Contract_One year  Contract_Two year
0                           0                  1                  0
1                           1                  0                  0
2                           1                  0                  0
3                           1                  0                  0
4                           1                  0                  0
...                       ...                ...                ...
7038                        0                  1                  0
7039                        1                  0                  0
7040                        1                  0                  0
7041                        0                  0                  1
7042                        0                  0                  1

[7043 rows x 35 columns]
```

In [154]:
```
data_HotEncoded1.dtypes
```

Out[154]:
```
customerID                          object
gender                              object
SeniorCitizen                        int64
Partner                             object
Dependents                        category
tenure                               int64
PhoneService                      category
MultipleLines                     category
OnlineSecurity                      object
OnlineBackup                        object
DeviceProtection                    object
PaperlessBilling                    object
PaymentMethod                       object
MonthlyCharges                     float64
TotalCharges                       float64
Churn                               object
Partner_category                  category
Dependents_category                   int8
MultipleLines_category                int8
PhoneService_category                 int8
InternetService_DSL                  uint8
InternetService_Fiber optic          uint8
InternetService_No                   uint8
TechSupport_No                       uint8
TechSupport_No internet service      uint8
TechSupport_Yes                      uint8
StreamingTV_No                       uint8
StreamingTV_No internet service      uint8
StreamingTV_Yes                      uint8
StreamingMovies_No                   uint8
StreamingMovies_No internet service  uint8
StreamingMovies_Yes                  uint8
Contract_Month-to-month              uint8
Contract_One year                    uint8
Contract_Two year                    uint8
dtype: object
```

In [155]:
```
del data['OnlineBackup']
del data['PaperlessBilling']
del data['PaymentMethod']
```

In [156]:
```
#removed columns which do not contribute to customer churn such as billing met
hod, paperless billing, and online backup
data.dtypes
```

Out[156]:
```
customerID                  object
gender                      object
SeniorCitizen                int64
Partner                     object
Dependents                category
tenure                       int64
PhoneService              category
MultipleLines             category
InternetService             object
OnlineSecurity              object
DeviceProtection          category
TechSupport                 object
StreamingTV                 object
StreamingMovies             object
Contract                    object
MonthlyCharges             float64
TotalCharges               float64
Churn                       object
Partner_category          category
Dependents_category           int8
MultipleLines_category        int8
PhoneService_category         int8
DeviceProtection_category     int8
dtype: object
```

In [5]:
```
#removed columns which do not contribute to customer churn such as billing met
hod, paperless billing, and online backup
#data_HotEncoded1.dtypes
```

In [158]:
```
del data_HotEncoded1['OnlineBackup']
del data_HotEncoded1['PaperlessBilling']
del data_HotEncoded1['PaymentMethod']
```

In [159]:
```
pd.set_option("display.max_rows", None, "display.max_columns", None)
```

In [160]:
```
data.shape
```

Out[160]:
```
(7043, 23)
```

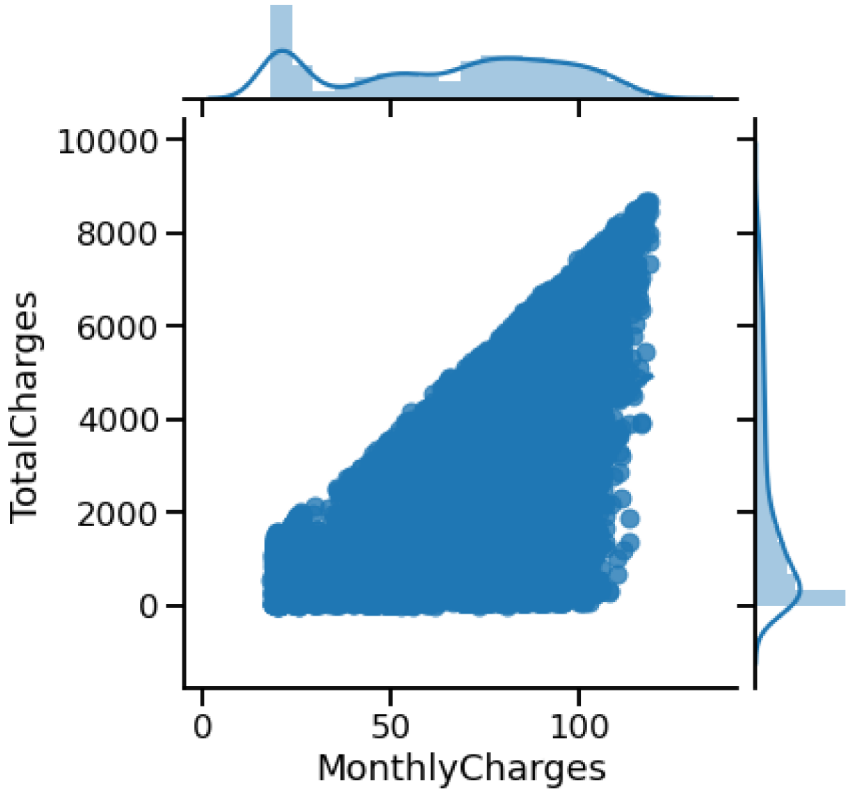
In [161]:
```
data_HotEncoded1.shape
```

Out[161]:
```
(7043, 32)
```

In [6]:
```
#print(data['MonthlyCharges'])
```

In [163]:
```
sns.jointplot(data=data, x="MonthlyCharges", y="TotalCharges", kind="reg")
```

Out[163]:
```
<seaborn.axisgrid.JointGrid at 0x7fe290994890>
```



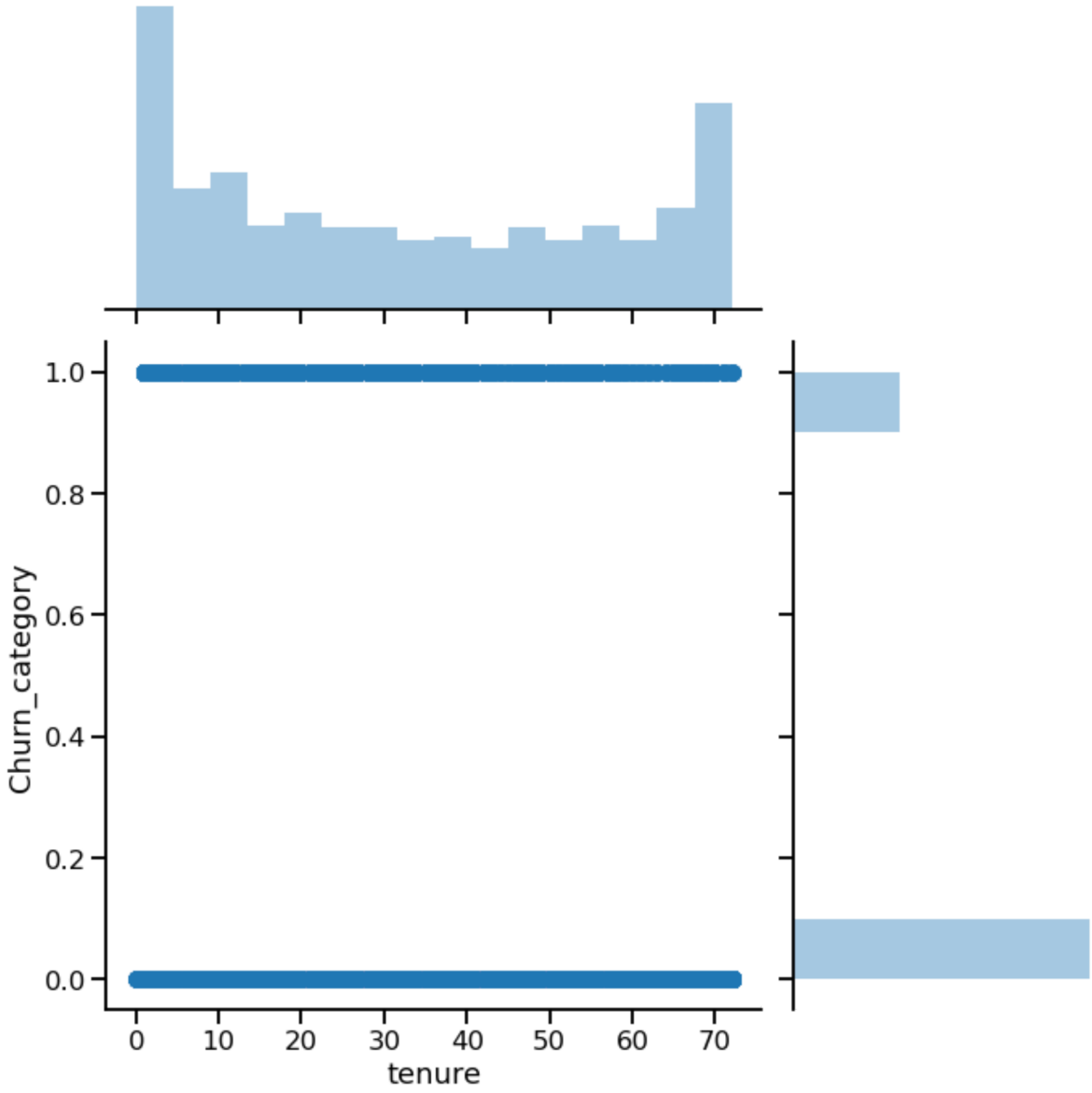
In [7]:
```
#print(data_HotEncoded1.head)
```

In [3]:
```
#data_HotEncoded1[['InternetService_Fiber optic','InternetService_DSL','Intern
etService_No']]
```

In [8]:
```
#Hypothesis 1: Customers who have long Tenures will not churn as frequently as
newer customers
#data_HotEncoded1["Churn"] = data_HotEncoded1["Churn"].astype('category')
#data_HotEncoded1["Churn_category"] = data_HotEncoded1["Churn"].cat.codes
#data_HotEncoded1[['tenure','Churn','Churn_category']]
```

In [167]:
```
sns.jointplot(data=data_HotEncoded1, x="tenure", y="Churn_category", height=10
, ratio=2, space=0.1, kind="scatter")
```

Out[167]: <seaborn.axisgrid.JointGrid at 0x7fe293a7cd90>



In [9]:
```
#If statement pandas
#df.loc[(df['First_name'] == 'Bill') | (df['First_name'] == 'Emma'), 'name_mat
ch'] = 'Match'
#df.loc[(df['First_name'] != 'Bill') & (df['First_name'] != 'Emma'), 'name_mat
ch'] = 'Mismatch'
#data_HotEncoded1.loc[(data_HotEncoded1['Churn_category'] > 0) & (data_HotEnco
ded1['tenure'] > 24), 'threshold_churn'] = 'threshold'
#data_HotEncoded1[['tenure','Churn_category','threshold_churn']]
```

In [172]:
```
data_HotEncoded1["threshold_churn"].value_counts()
```

Out[172]:
```
threshold    538
Name: threshold_churn, dtype: int64
```

In [173]:
```
prob_long_term_cust_will_churn = (data_HotEncoded1["threshold_churn"].value_co
unts())/len(data_HotEncoded1)
print(prob_long_term_cust_will_churn)
```

```
threshold    0.076388
Name: threshold_churn, dtype: float64
```

In [174]:
```
data_HotEncoded1["tenure"].mean()
```

Out[174]:
```
32.37114865824223
```

In [175]:
```
stdTenure_encoded = data_HotEncoded1.std()['tenure']
print(data_HotEncoded1.std()['tenure'])
```

```
24.55948102309446
```

In [176]:
```
#Probability of tenure and churn. The longer the tenure the less likelihood of
churn
#Probability = number of favorable outcomes / total number of outcomes
ProbChurnYesLowTenure = 1 - prob_long_term_cust_will_churn
print(ProbChurnYesLowTenure)
```

```
threshold    0.923612
Name: threshold_churn, dtype: float64
```

In [177]:
```
#long term churn probability
data_HotEncoded1.loc[(data_HotEncoded1['Churn_category'] == 0), 'churnZero'] =
'NoChurn'
# print the column data_HotEncoded1[['churnZero']]
LTCP = (data_HotEncoded1["churnZero"].value_counts())/len(data_HotEncoded1)
#long term tenure probability
data_HotEncoded1.loc[(data_HotEncoded1['tenure'] > 24), 'tenure24'] = '24'
# print the column data_HotEncoded1[['tenure24']]
LTTP = (data_HotEncoded1["tenure24"].value_counts())/len(data_HotEncoded1)
```

In [178]:
```
print("LTCP is",LTCP,"\nLTTP is",LTTP,"\nprob_long_term_cust_will_churn",prob_
long_term_cust_will_churn)
```

```
LTCP is NoChurn    0.73463
Name: churnZero, dtype: float64
LTTP is 24    0.544228
Name: tenure24, dtype: float64
prob_long_term_cust_will_churn threshold    0.076388
Name: threshold_churn, dtype: float64
```

In [179]:
```
#Bayes formula for customer churn longer than 24 months
#((probability of B given A is true) * probability of A ) / probability of B
fLTCP = float(LTCP)
fLTTP = float(LTTP)
fprob_long_term_cust_will_churn = float(prob_long_term_cust_will_churn)
BPTC = (fprob_long_term_cust_will_churn*fLTTP)/fLTCP
BPTC_float = "{:.4f}".format(BPTC)
print("Bayes probability for a long tenure customer to churn\n",BPTC_float)
```

```
Bayes probability for a long tenure customer to churn
 0.0566
```

In [180]:
```
data_HotEncoded1.dtypes
```

Out[180]:
```
customerID                              object
gender                                  object
SeniorCitizen                            int64
Partner                                 object
Dependents                            category
tenure                                   int64
PhoneService                          category
MultipleLines                         category
OnlineSecurity                          object
DeviceProtection                        object
MonthlyCharges                         float64
TotalCharges                           float64
Churn                                 category
Partner_category                      category
Dependents_category                       int8
MultipleLines_category                    int8
PhoneService_category                     int8
InternetService_DSL                      uint8
InternetService_Fiber optic              uint8
InternetService_No                       uint8
TechSupport_No                           uint8
TechSupport_No internet service          uint8
TechSupport_Yes                          uint8
StreamingTV_No                           uint8
StreamingTV_No internet service          uint8
StreamingTV_Yes                          uint8
StreamingMovies_No                       uint8
StreamingMovies_No internet service      uint8
StreamingMovies_Yes                      uint8
Contract_Month-to-month                  uint8
Contract_One year                        uint8
Contract_Two year                        uint8
Churn_category                            int8
threshold_churn                         object
churnZero                               object
tenure24                                object
dtype: object
```

In [181]:
```
selected_columns = data_HotEncoded1[["TechSupport_No","Churn","TechSupport_Yes","Churn_category"]]
data_tech_churn = selected_columns.copy()
data_tech_churn.dtypes
```

Out[181]:
```
TechSupport_No        uint8
Churn              category
TechSupport_Yes       uint8
Churn_category         int8
dtype: object
```

In [182]:
```
data_tech_churn["TechSupport_No"].value_counts()
```

Out[182]:
```
0    3570
1    3473
Name: TechSupport_No, dtype: int64
```

In [183]:
```
#Bayes formula for customer churn when there is no tech support
#((probability of B given A is true) * probability of A ) / probability of B
data_tech_churn.loc[(data_tech_churn['Churn_category'] == 1), 'churn_tech'] = 'quit'
ProbChurn = (data_tech_churn["churn_tech"].value_counts())/len(data_tech_churn)

data_tech_churn.loc[(data_tech_churn['TechSupport_No'] == 1), 'techNo'] = 'No'
ProbTechNo = (data_tech_churn["techNo"].value_counts())/len(data_tech_churn)

data_tech_churn.loc[(data_tech_churn['Churn_category'] == 1) & (data_tech_churn['TechSupport_No'] == 1), 'threshold_churn_tech'] = 'threshold'
prob_notechsprt_cust_will_churn = (data_tech_churn["threshold_churn_tech"].value_counts())/len(data_tech_churn)

fProbChurn = float(ProbChurn)
fProbTechNo = float(ProbTechNo)
fprob_notechsprt_cust_will_churn = float(prob_notechsprt_cust_will_churn)
BayesProbNoTechChurn = (fprob_notechsprt_cust_will_churn*fProbTechNo)/fProbChurn
print('probability of no tech support & churn turnover: ', prob_notechsprt_cust_will_churn)
print('probability of no tech support is: ', ProbTechNo)
floatBayesProbNoTechChurn = "{:.4f}".format(BayesProbNoTechChurn)
print('Bayes probability is: ', floatBayesProbNoTechChurn)
```

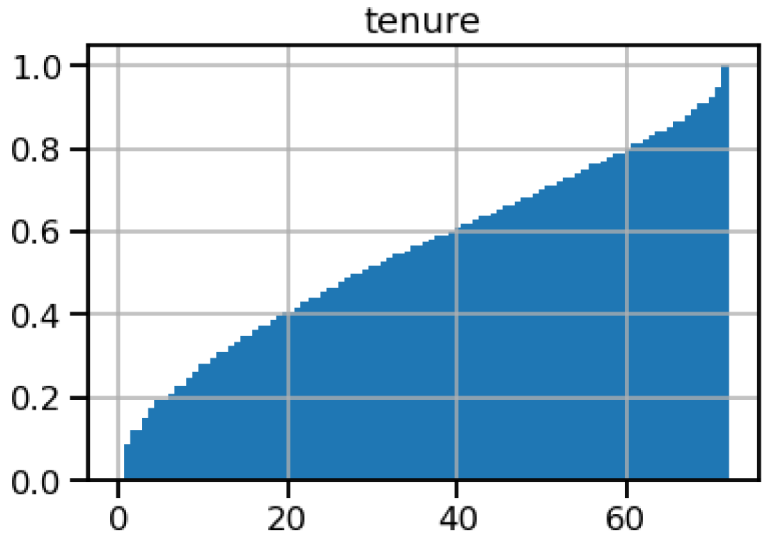
```
probability of no tech support & churn turnover:  threshold    0.20531
Name: threshold_churn_tech, dtype: float64
probability of no tech support is:  No    0.493114
Name: techNo, dtype: float64
Bayes probability is:  0.3815
```

In [184]:
```
selected_columns_tenure = data_HotEncoded1[["tenure"]]
dataSelectedColumntenure = selected_columns_tenure.copy()
dataSelectedColumntenure.dtypes
```

Out[184]:
```
tenure    int64
dtype: object
```

In [185]:
```
dataSelectedColumntenure.hist(cumulative=True, density=1, bins=100)

plt.show()
```

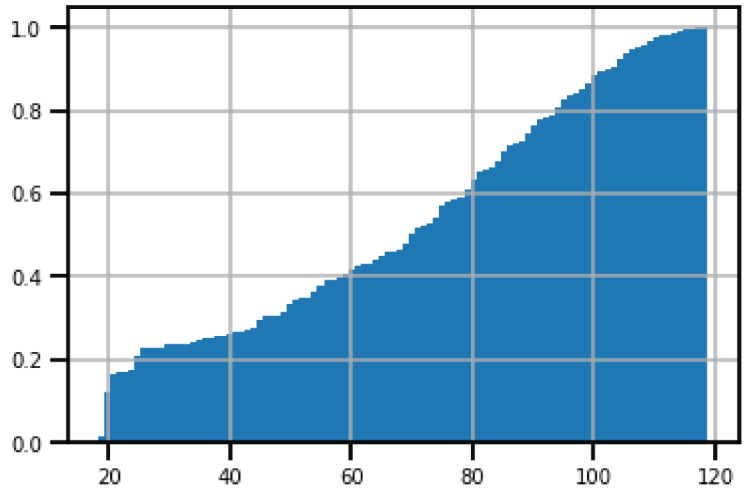


In [10]:
```
#Third set to test hypothesis monthly charges and churn
#selected_columns_month = data_HotEncoded1[["MonthlyCharges","Churn","Churn_ca
tegory"]]
#data_Month_churn_month = selected_columns_month.copy()
#data_Month_churn_month.loc[:,'MonthlyCharges']
```

In [190]:
```
#Third set to test hypothesis monthly charges and churn
print("length of column\n ", len(data_Month_churn_month))
print("print dtypes\n", data_Month_churn_month.dtypes)
```

```
length of column
  7043
print dtypes
 MonthlyCharges     float64
Churn              category
Churn_category         int8
dtype: object
```

In [191]:
```
plotMonthlyCharges = data_Month_churn_month.loc[:,'MonthlyCharges']
plotMonthlyCharges.hist(cumulative=True, density=1, bins=100, xlabelsize=10, y
labelsize=10)
#cdf plot
plt.show()
```



In [192]:
```
data_Month_churn_month.dtypes
```

Out[192]:
```
MonthlyCharges     float64
Churn             category
Churn_category        int8
dtype: object
```

In [193]:
```
#Bayes formula for customer churn when there is a low monthly bill
#((probability of B given A is true) * probability of A ) / probability of B
data_Month_churn_month.loc[(data_Month_churn_month['MonthlyCharges'] < 40.0),
'charges'] = '40mc'
ProbMonthChurn = (data_Month_churn_month["charges"].value_counts())/len(data_M
onth_churn_month)
print('probability of monthly charges being less than $40: ', ProbMonthChurn)

data_Month_churn_month.loc[(data_Month_churn_month['Churn_category'] == 1), 'C
C'] = 'No'
ProbChurnYes = (data_Month_churn_month["CC"].value_counts())/len(data_Month_ch
urn_month)
print('turnover probability in general: ', ProbChurnYes)

data_Month_churn_month.loc[(data_Month_churn_month['MonthlyCharges'] < 40.0) &
(data_Month_churn_month['Churn_category'] == 1), 'tcm'] = 'threshold_churn_mon
th'
prob_Month_churn_month = (data_Month_churn_month["tcm"].value_counts())/len(da
ta_Month_churn_month)
print('probability of monthly bill less than $40 and customer turnover: ', pro
b_Month_churn_month)

fProbMonthChurn = float(ProbMonthChurn)
fProbChurnYes = float(ProbChurnYes)
fprob_Month_churn_month = float(prob_Month_churn_month)
BayesProbNoMonthChurn = (fprob_Month_churn_month*fProbMonthChurn)/fProbChurnYe
s

floatBayesProbNoMonthChurn = "{:.4f}".format(BayesProbNoMonthChurn)
print('Bayes probability is: ', floatBayesProbNoMonthChurn)
```

```
probability of monthly charges being less than $40:  40mc    0.260826
Name: charges, dtype: float64
turnover probability in general:  No    0.26537
Name: CC, dtype: float64
probability of monthly bill less than $40 and customer turnover:  threshold_c
hurn_month    0.030243
Name: tcm, dtype: float64
Bayes probability is:  0.0297
```

In [194]:
```
#Paired Samples t-test
'''The paired sample t-test is also called dependent sample t-test. It's an un
ivariate test that tests for a
significant difference between 2 related variables. An example of this is if y
ou where to collect the blood
pressure for an individual before and after some treatment, condition, or time
point. This also tests if a dataset is
normally distributed. The null-hypothesis of this test is that the population
 is normally distributed.
Thus, if the p value is less than the chosen alpha level, then the null hypoth
esis is rejected and there is
evidence that the data tested are not normally distributed. statistic is close
to normality, p value is threshold usually < 0.05 if it is
normally distributed data'''

stats.shapiro(data_HotEncoded1['MonthlyCharges'])
```

```
/opt/conda/envs/Python-3.7-main/lib/python3.7/site-packages/scipy/stats/mores
tats.py:1681: UserWarning: p-value may not be accurate for N > 5000.
  warnings.warn("p-value may not be accurate for N > 5000.")
```

Out[194]:
```
ShapiroResult(statistic=0.9208902716636658, pvalue=0.0)
```

In [199]:
```
stats.shapiro(data_HotEncoded1['Churn_category'])
```

Out[199]:
```
ShapiroResult(statistic=0.5510977506637573, pvalue=0.0)
```

In [200]:
```
stats.ttest_rel(data_HotEncoded1['MonthlyCharges'], data_HotEncoded1['Churn_ca
tegory'])
```

Out[200]:
```
Ttest_relResult(statistic=180.37639409369268, pvalue=0.0)
```

In [196]:
```
stats.shapiro(data_HotEncoded1['tenure'])
```

Out[196]:
```
ShapiroResult(statistic=0.9037491083145142, pvalue=0.0)
```

In [206]:
```
#t test single parameter
mctest = data_HotEncoded1['MonthlyCharges']

tset, pval = ttest_1samp(mctest, 40)
print('p-values',pval)

if pval < 0.05:    # alpha value is 0.05 or 5%
   print(" we are rejecting null hypothesis")
else:
  print("we are accepting null hypothesis")
```

```
p-values 0.0
 we are rejecting null hypothesis
```

In [207]:
```
# t test comparison of 2 samples
'''week1_std = np.std(week1)
week2_std = np.std(week2)
print("week1 std value:",week1_std)
print("week2 std value:",week2_std)
ttest,pval = ttest_ind(week1,week2)
print("p-value",pval)
if pval <0.05:
  print("we reject null hypothesis")
else:
  print("we accept null hypothesis")'''
```

Out[207]:
```
'week1_std = np.std(week1)\nweek2_std = np.std(week2)\nprint("week1 std valu
e:",week1_std)\nprint("week2 std value:",week2_std)\nttest,pval = ttest_ind(w
eek1,week2)\nprint("p-value",pval)\nif pval <0.05:\n  print("we reject null h
ypothesis")\nelse:\n  print("we accept null hypothesis")'
```

In [208]:
```
'''The p-value is just the smallest significance level at which the null hypot
hesis would be rejected.
But once you have chosen a significance level, e.g. 0.05,it would be incorrect
to interpret p-values in reference to how close they are to your
significance level.'''
```

Out[208]:
```
'The p-value is just the smallest significance level at which the null hypoth
esis would be rejected. \nBut once you have chosen a significance level, e.g.
0.05,it would be incorrect to interpret p-values in reference to how close th
ey are to your\nsignificance level.'
```

In [209]:
```
'''For a Frequentist, probability of an event is the proportion of that event
 in long run. Most frequentist concepts comes from this idea (E.g. p-values, c
onfidence intervals)
For a Bayesian, probability is more epistemological. Which means that is his/h
er belief on the chance of an event occurring. This belief also known as prior
probability comes
from the previous experience, knowledge of literature e.t.c.
Bayesian inference use Bayes theorem to combine the prior probabilities and th
e likelihood from the data to get the posterior probability of the event.
Posterior probability (in lay terms) is the updated belief on the probability
 of an event happening given the prior and the data observed.'''
```

Out[209]:
```
'For a Frequentist, probability of an event is the proportion of that event i
n long run. Most frequentist concepts comes from this idea (E.g. p-values, co
nfidence intervals)\nFor a Bayesian, probability is more epistemological. Whi
ch means that is his/her belief on the chance of an event occurring. This bel
ief also known as prior probability comes \nfrom the previous experience, kno
wledge of literature e.t.c.\nBayesian inference use Bayes theorem to combine
the prior probabilities and the likelihood from the data to get the posterior
probability of the event.\nPosterior probability (in lay terms) is the update
d belief on the probability of an event happening given the prior and the dat
a observed.'
```

In [ ]: