In [1]:

```
# Basic Libraries
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt # we only need pyplot
sb.set() # set the default Seaborn style for graphics
```

Problem 1 : Predicting CentralAir using SalePrice

a) Plot the distribution of CentralAir to check the imbalance of Y against N. Print the ratio of the classes Y : N.

In [2]:

```
houseData = pd.read_csv('train.csv')
CentralAir = pd.DataFrame(houseData['CentralAir'])
houseData["CentralAir"].value_counts()

```

Out[2]:

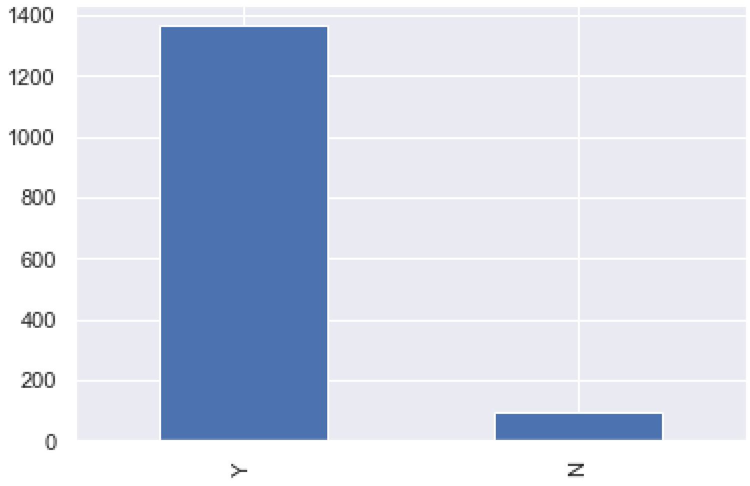
```
Y    1365
N      95
Name: CentralAir, dtype: int64
```

In [3]:

```
houseData["CentralAir"].value_counts().plot(kind='bar')

```

Out[3]:

```
<AxesSubplot:>
```



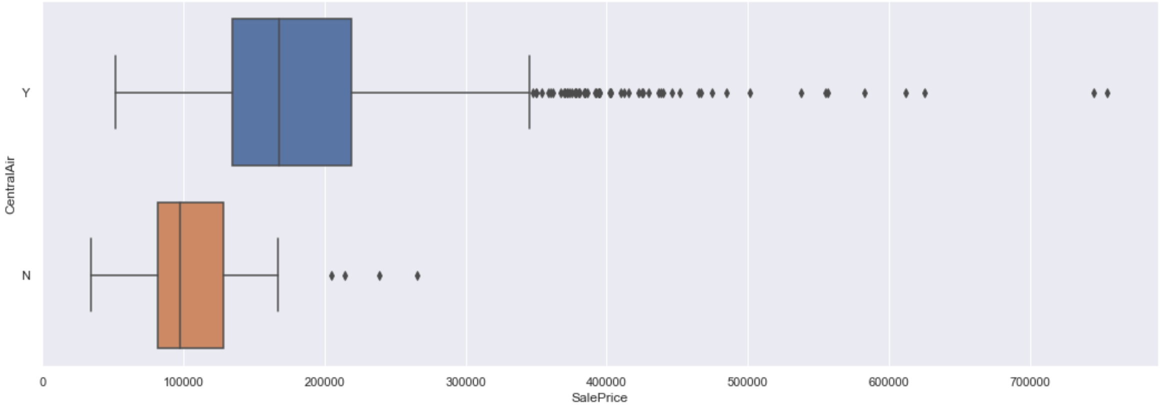
b) Plot CentralAir against SalePrice using any appropriate bivariate plot to note the mutual relationship.

In [4]:

```
# Create a joint dataframe by concatenating
SalePrice = pd.DataFrame(houseData['SalePrice'])
concatDF = pd.concat([SalePrice, CentralAir], axis = 1).reindex(SalePric

# Joint Boxplot of saleprice against central air
f = plt.figure(figsize=(18, 6))
sb.boxplot(x = "SalePrice", y = "CentralAir", data = concatDF, orient = '
```

Out[4]:

```
<AxesSubplot:xlabel='SalePrice', ylabel='CentralAir'>
```



c) Import Classification Tree model from Scikit-Learn : from sklearn.tree import DecisionTreeClassifier

In [5]:

```
# Import Decision Tree Classifier model from Scikit-Learn
from sklearn.tree import DecisionTreeClassifier

# Create a Decision Tree Classifier object
dectree = DecisionTreeClassifier(max_depth = 2)
```

d) Partition the dataset houseData into two “random” portions : Train Data (1100 rows) and Test Data (360 rows).

In [6]:

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix

# Extract Response and Predictors
X = pd.DataFrame(houseData['SalePrice'])
y = pd.DataFrame(houseData['CentralAir'])

# Split the Dataset into random Train and Test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 36

# Check the sample sizes
print("Train Set :", X_train.shape, y_train.shape)
print("Test Set  :", X_test.shape, y_test.shape)
```

```
Train Set : (1100, 1) (1100, 1)
Test Set  : (360, 1) (360, 1)
```

e) Training : Fit a Decision Tree model on the Train Dataset to predict the class (Y/N) of CentralAir using SalePrice.

In [7]:

```
# Decision Tree using Train Data
dectree = DecisionTreeClassifier(max_depth = 2)  # create the decision t
dectree.fit(X_train, y_train)                    # train the decision tr
```

Out[7]:

```
DecisionTreeClassifier(max_depth=2)
```

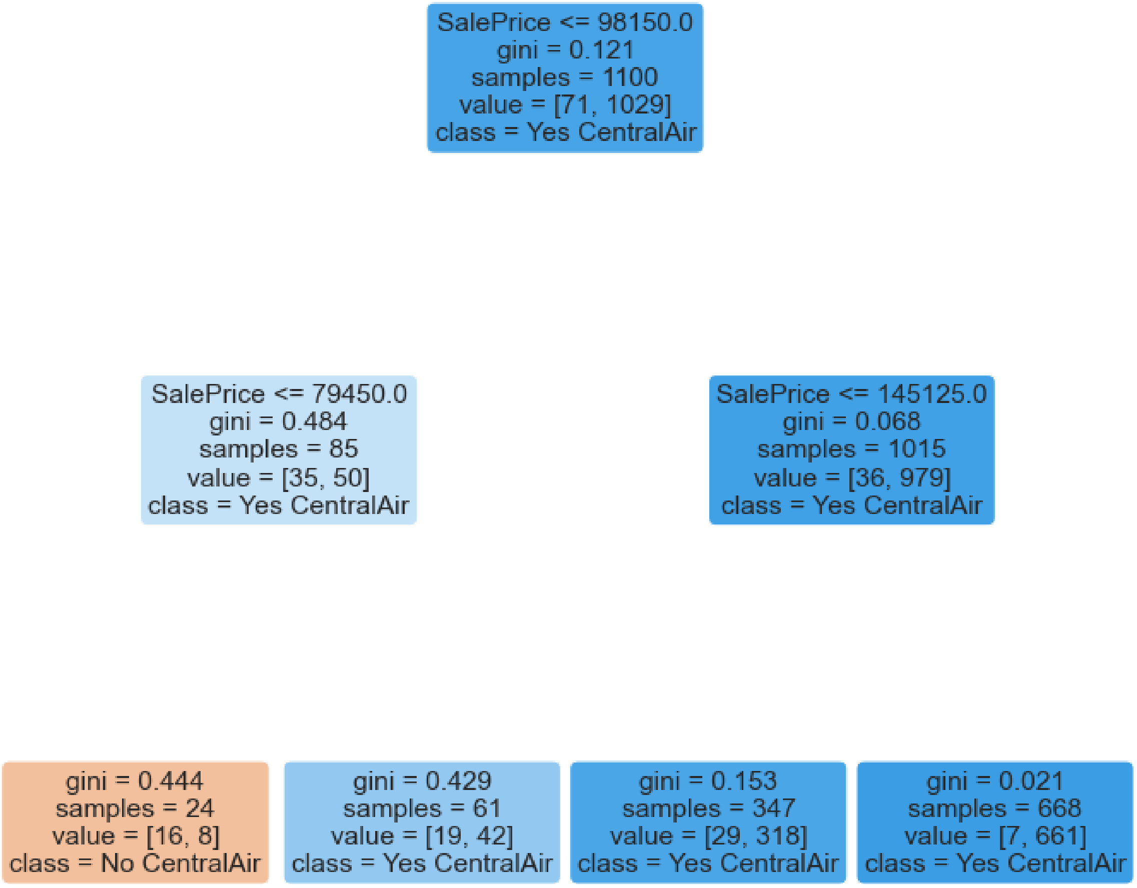
f) Visualize the Decision Tree model using the plot_tree function : from sklearn.tree import plot_tree

In [8]:

```python
# Plot the trained Decision Tree
from sklearn.tree import plot_tree

f = plt.figure(figsize=(12,12))
plot_tree(dectree, filled=True, rounded=True,
          feature_names=["SalePrice"],
          class_names=["No CentralAir","Yes CentralAir"])
```

Out[8]:

```
[Text(334.8, 543.6, 'SalePrice <= 98150.0\ngini = 0.121\nsamples = 1100\nva
lue = [71, 1029]\nclass = Yes CentralAir'),
 Text(167.4, 326.16, 'SalePrice <= 79450.0\ngini = 0.484\nsamples = 85\nval
ue = [35, 50]\nclass = Yes CentralAir'),
 Text(83.7, 108.72000000000003, 'gini = 0.444\nsamples = 24\nvalue = [16,
8]\nclass = No CentralAir'),
 Text(251.10000000000002, 108.72000000000003, 'gini = 0.429\nsamples = 61\n
value = [19, 42]\nclass = Yes CentralAir'),
 Text(502.20000000000005, 326.16, 'SalePrice <= 145125.0\ngini = 0.068\nsam
ples = 1015\nvalue = [36, 979]\nclass = Yes CentralAir'),
 Text(418.5, 108.72000000000003, 'gini = 0.153\nsamples = 347\nvalue = [29,
318]\nclass = Yes CentralAir'),
 Text(585.9, 108.72000000000003, 'gini = 0.021\nsamples = 668\nvalue = [7,
661]\nclass = Yes CentralAir')]
```



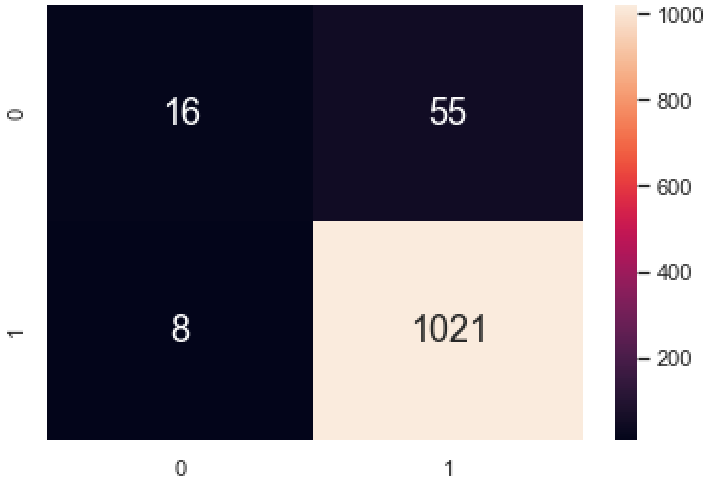
g) Predict CentralAir for the train dataset using the Decision Tree model and plot the Two-Way Confusion Matrix

In [9]:

```python
# Predict  CentralAir corresponding to Saleprice
y_train_pred = dectree.predict(X_train)


# Plot the two-way Confusion Matrix
from sklearn.metrics import confusion_matrix
sb.heatmap(confusion_matrix(y_train, y_train_pred),
           annot = True, fmt=".0f", annot_kws={"size": 18})
```

Out[9]:

```
<AxesSubplot:>
```



h) Print accuracy measures of the Decision Tree model, including its Classification Accuracy, True Positive Rate, True Negative Rate, False Positive Rate and False Negative Rate, based on the confusion matrix on train data.

In [10]:

```
# Print the Classification Accuracy
print("Classification Accuracy \t:", dectree.score(X_train, y_train))

TN = confusion_matrix(y_train, y_train_pred)[0][0]
FP = confusion_matrix(y_train, y_train_pred)[0][1]
FN = confusion_matrix(y_train, y_train_pred)[1][0]
TP = confusion_matrix(y_train, y_train_pred)[1][1]

TPR = TP/(TP+FP)
FPR = FP/(TP+FP)
TNR = TN/(TN+FN)
FNR = FN/(TN+FN)

print("True Positive Rate \t:", TPR)
print("False Positive Rate \t:", FPR)
print("True Negative Rate \t:", TNR)
print("False Negative Rate \t:", FNR)
```

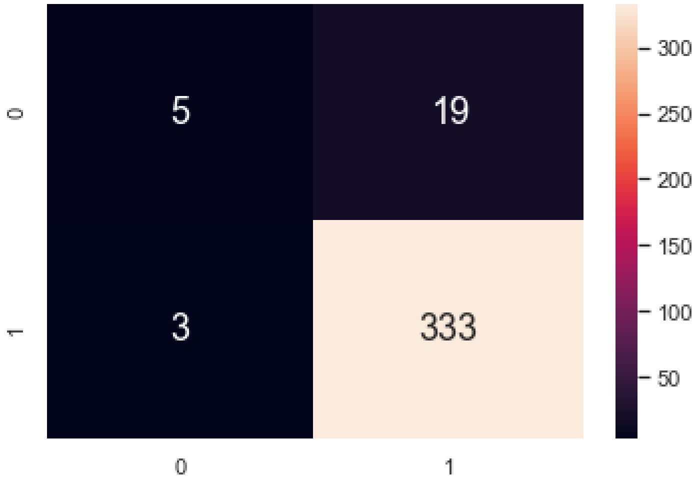
```
Classification Accuracy         : 0.9427272727272727
True Positive Rate      : 0.9488847583643123
False Positive Rate     : 0.05111524163568773
True Negative Rate      : 0.6666666666666666
False Negative Rate     : 0.3333333333333333
```

i) Predict CentralAir for the test dataset using the Decision Tree model and plot the Two-Way Confusion Matrix.

In [11]:

```python
# Predict  CentralAir corresponding to Saleprice
y_test_pred = dectree.predict(X_test)


# Plot the two-way Confusion Matrix
from sklearn.metrics import confusion_matrix
sb.heatmap(confusion_matrix(y_test, y_test_pred),
           annot = True, fmt=".0f", annot_kws={"size": 18})
```

Out[11]: `<AxesSubplot:>`



j) Print accuracy measures of the Decision Tree model, including its Classification Accuracy, True Positive Rate, True Negative Rate, False Positive Rate and False Negative Rate, based on the confusion matrix on test data.

In [12]:

```python
# Print the Classification Accuracy
print("Classification Accuracy \t:", dectree.score(X_test, y_test))

TN = confusion_matrix(y_test, y_test_pred)[0][0]
FP = confusion_matrix(y_test, y_test_pred)[0][1]
FN = confusion_matrix(y_test, y_test_pred)[1][0]
TP = confusion_matrix(y_test, y_test_pred)[1][1]

TPR  = TP/(TP+FP)
FPR  = FP/(TP+FP)
TNR  = TN/(TN+FN)
FNR  = FN/(TN+FN)

print("True Positive Rate \t:", TPR)
print("False Positive Rate \t:", FPR)
print("True Negative Rate \t:", TNR)
print("False Negative Rate \t:", FNR)
```

```
Classification Accuracy         : 0.9388888888888889
True Positive Rate      : 0.9460227272727273
False Positive Rate     : 0.05397727272727273
True Negative Rate      : 0.625
False Negative Rate     : 0.375
```

Problem 2

CentralAir AGAINST GrLivArea

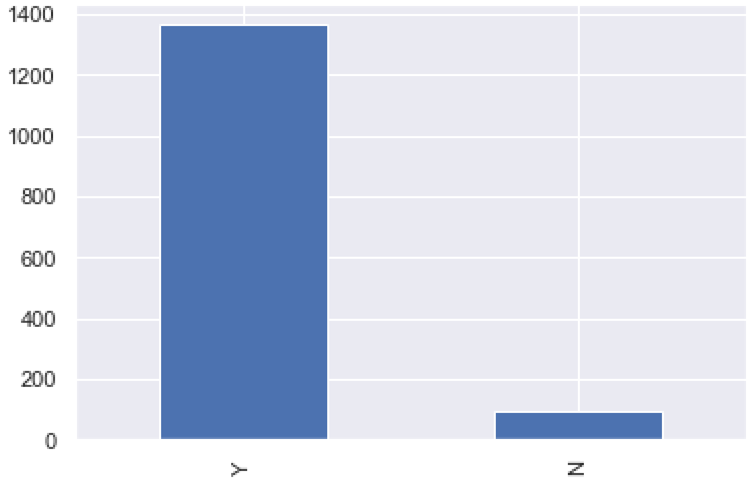
In [13]:

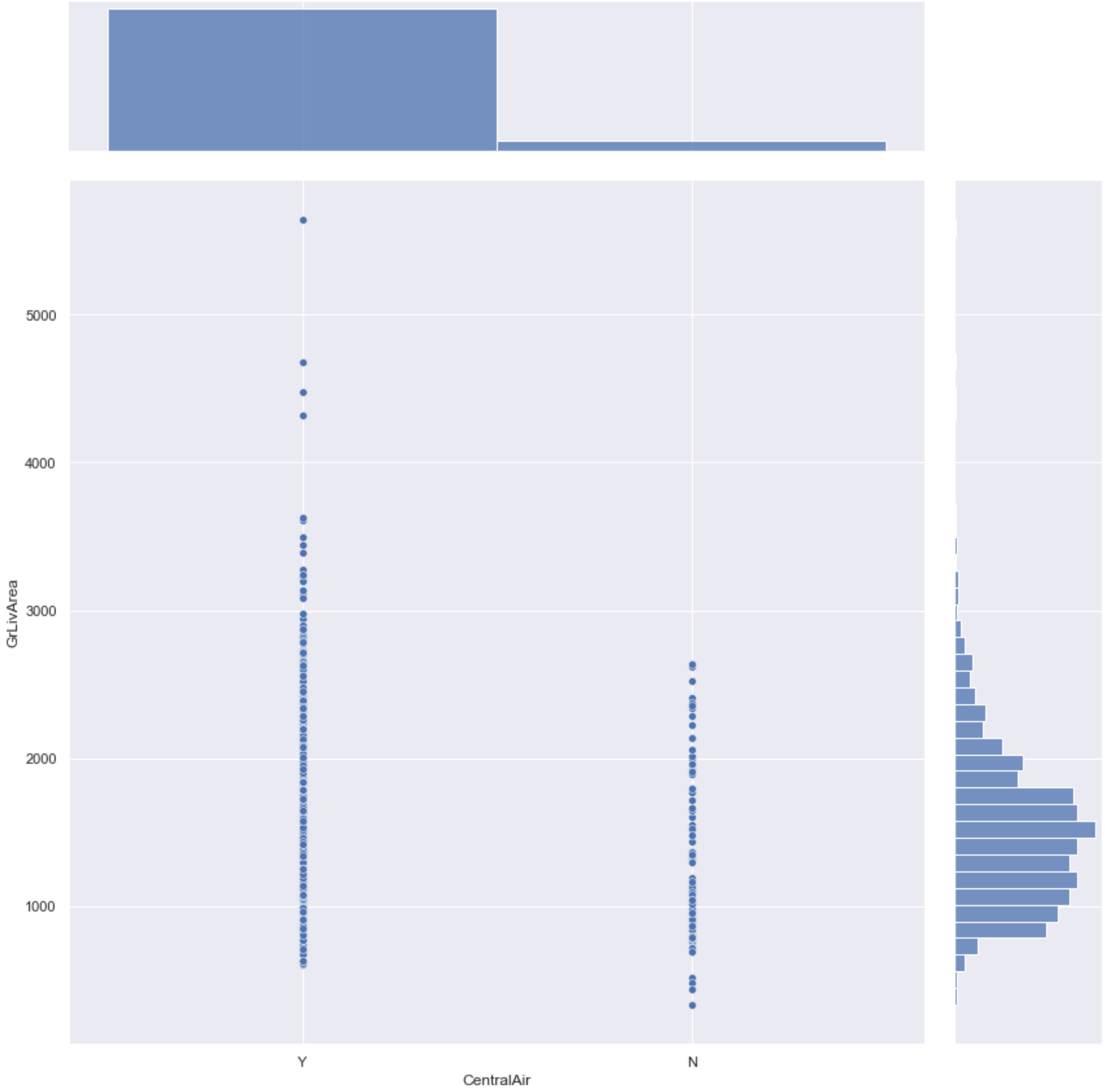
```
houseData = pd.read_csv('train.csv')
CentralAir = pd.DataFrame(houseData['CentralAir'])
houseData["CentralAir"].value_counts()

houseData["CentralAir"].value_counts().plot(kind='bar')

sb.jointplot(data = houseData,x = "CentralAir", y = "GrLivArea",  height

# Create a joint dataframe by concatenating
GrLivArea = pd.DataFrame(houseData['GrLivArea'])
concatDF = pd.concat([ GrLivArea, CentralAir], axis = 1).reindex(GrLivAr

# Joint Boxplot of saleprice against central air
f = plt.figure(figsize=(18, 6))
sb.boxplot(x = "GrLivArea", y = "CentralAir", data = concatDF, orient = '

# Import Decision Tree Classifier model from Scikit-Learn
from sklearn.tree import DecisionTreeClassifier

# Create a Decision Tree Classifier object
dectree = DecisionTreeClassifier(max_depth = 2)

from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix

# Extract Response and Predictors
X = pd.DataFrame(houseData['GrLivArea'])
y = pd.DataFrame(houseData['CentralAir'])

# Split the Dataset into random Train and Test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 36

# Check the sample sizes
print("Train Set :", X_train.shape, y_train.shape)
print("Test Set  :", X_test.shape, y_test.shape)


# Decision Tree using Train Data
dectree = DecisionTreeClassifier(max_depth = 2)  # create the decision t
dectree.fit(X_train, y_train)                    # train the decision tr
```

```
Train Set : (1100, 1) (1100, 1)
Test Set  : (360, 1) (360, 1)
```

Out[13]:

```
DecisionTreeClassifier(max_depth=2)
```


1400
1200
1000
800
600
400
200
0
Y
N


5000
4000
3000
2000
1000
GrLivArea
Y
N
CentralAir

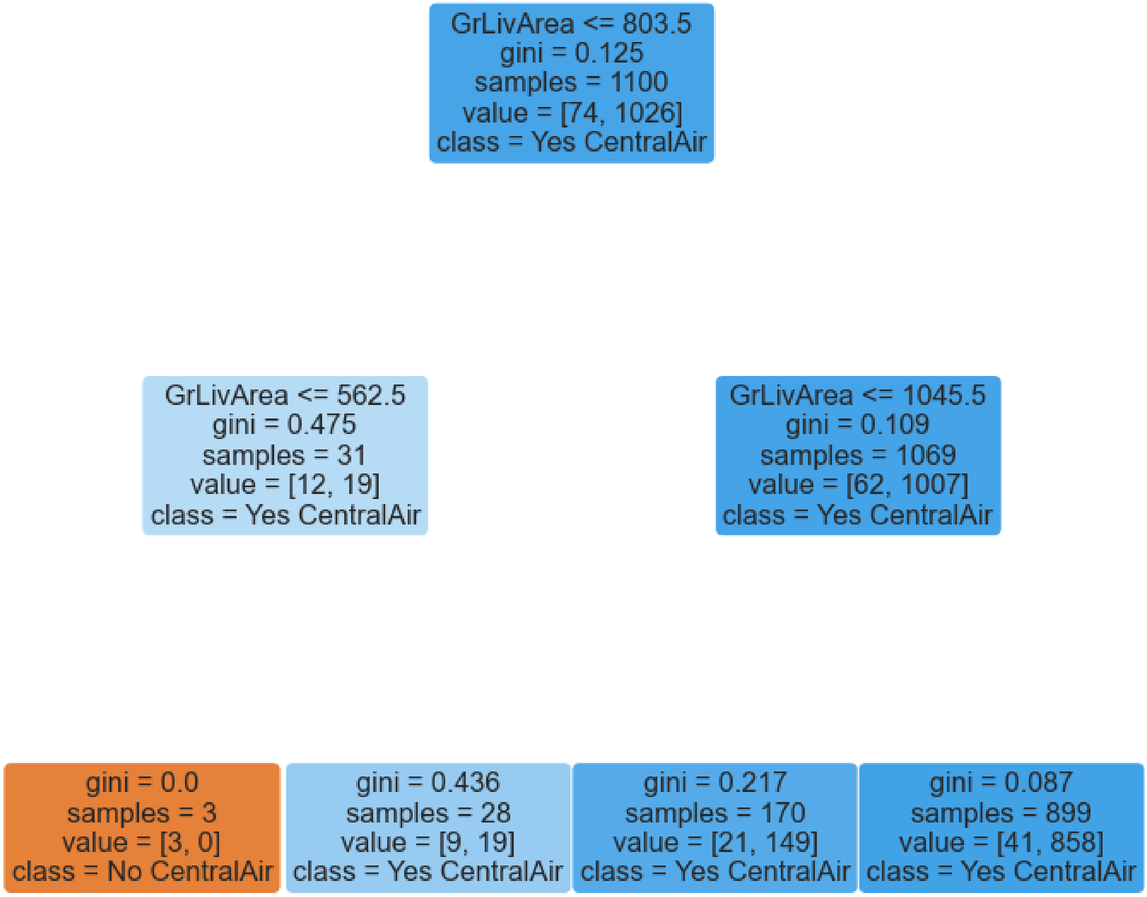
Y

In [14]:

```
# Plot the trained Decision Tree
from sklearn.tree import plot_tree

f = plt.figure(figsize=(12,12))
plot_tree(dectree, filled=True, rounded=True,
          feature_names=["GrLivArea"],
          class_names=["No CentralAir","Yes CentralAir"])
```

Out[14]:

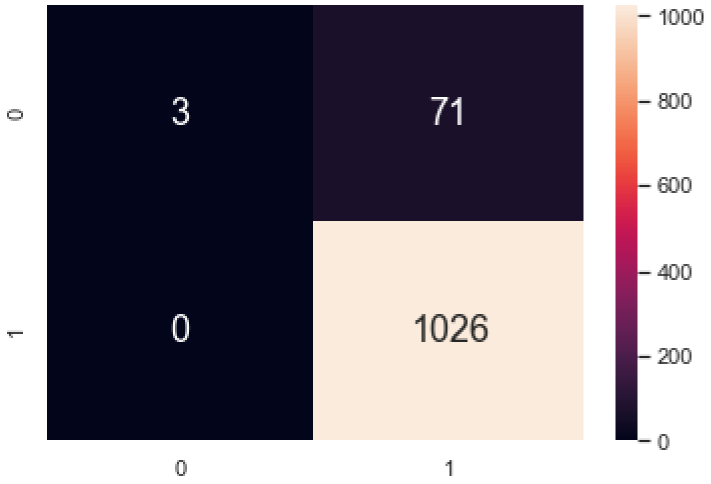
```
[Text(334.8, 543.6, 'GrLivArea <= 803.5\ngini = 0.125\nsamples = 1100\nvalu
e = [74, 1026]\nclass = Yes CentralAir'),
 Text(167.4, 326.16, 'GrLivArea <= 562.5\ngini = 0.475\nsamples = 31\nvalue
= [12, 19]\nclass = Yes CentralAir'),
 Text(83.7, 108.72000000000003, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]\nc
lass = No CentralAir'),
 Text(251.10000000000002, 108.72000000000003, 'gini = 0.436\nsamples = 28\n
value = [9, 19]\nclass = Yes CentralAir'),
 Text(502.20000000000005, 326.16, 'GrLivArea <= 1045.5\ngini = 0.109\nsampl
es = 1069\nvalue = [62, 1007]\nclass = Yes CentralAir'),
 Text(418.5, 108.72000000000003, 'gini = 0.217\nsamples = 170\nvalue = [21,
149]\nclass = Yes CentralAir'),
 Text(585.9, 108.72000000000003, 'gini = 0.087\nsamples = 899\nvalue = [41,
858]\nclass = Yes CentralAir')]
```



In [15]:

```python
# Predict  CentralAir corresponding to Saleprice
y_train_pred = dectree.predict(X_train)


# Plot the two-way Confusion Matrix
from sklearn.metrics import confusion_matrix
sb.heatmap(confusion_matrix(y_train, y_train_pred),
           annot = True, fmt=".0f", annot_kws={"size": 18})

# Print the Classification Accuracy
print("Classification Accuracy \t:", dectree.score(X_train, y_train))

TN = confusion_matrix(y_train, y_train_pred)[0][0]
FP = confusion_matrix(y_train, y_train_pred)[0][1]
FN = confusion_matrix(y_train, y_train_pred)[1][0]
TP = confusion_matrix(y_train, y_train_pred)[1][1]

TPR = TP/(TP+FP)
FPR = FP/(TP+FP)
TNR = TN/(TN+FN)
FNR = FN/(TN+FN)

print("True Positive Rate \t:", TPR)
print("False Positive Rate \t:", FPR)
print("True Negative Rate \t:", TNR)
print("False Negative Rate \t:", FNR)
```

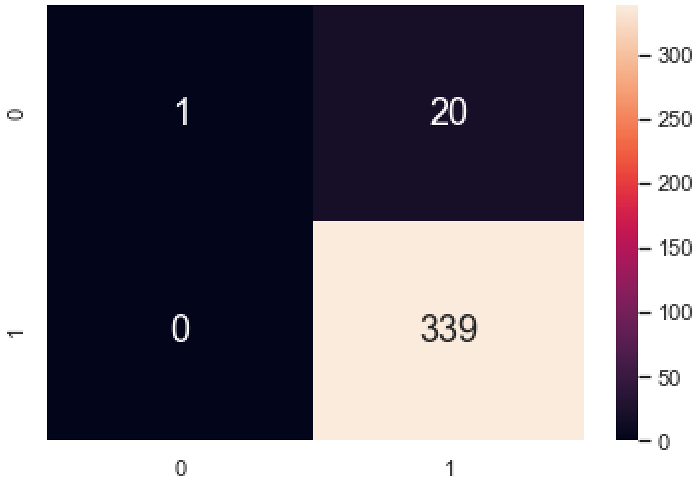
```
Classification Accuracy         : 0.9354545454545454
True Positive Rate      : 0.935278030993619
False Positive Rate     : 0.06472196900638104
True Negative Rate      : 1.0
False Negative Rate     : 0.0
```



In [16]:

```python
# Predict  CentralAir corresponding to Saleprice
y_test_pred = dectree.predict(X_test)


# Plot the two-way Confusion Matrix
from sklearn.metrics import confusion_matrix
sb.heatmap(confusion_matrix(y_test, y_test_pred),
           annot = True, fmt=".0f", annot_kws={"size": 18})
# Print the Classification Accuracy
print("Classification Accuracy \t:", dectree.score(X_test, y_test))

TN = confusion_matrix(y_test, y_test_pred)[0][0]
FP = confusion_matrix(y_test, y_test_pred)[0][1]
FN = confusion_matrix(y_test, y_test_pred)[1][0]
TP = confusion_matrix(y_test, y_test_pred)[1][1]

TPR = TP/(TP+FP)
FPR = FP/(TP+FP)
TNR = TN/(TN+FN)
FNR = FN/(TN+FN)

print("True Positive Rate \t:", TPR)
print("False Positive Rate \t:", FPR)
print("True Negative Rate \t:", TNR)
print("False Negative Rate \t:", FNR)
```

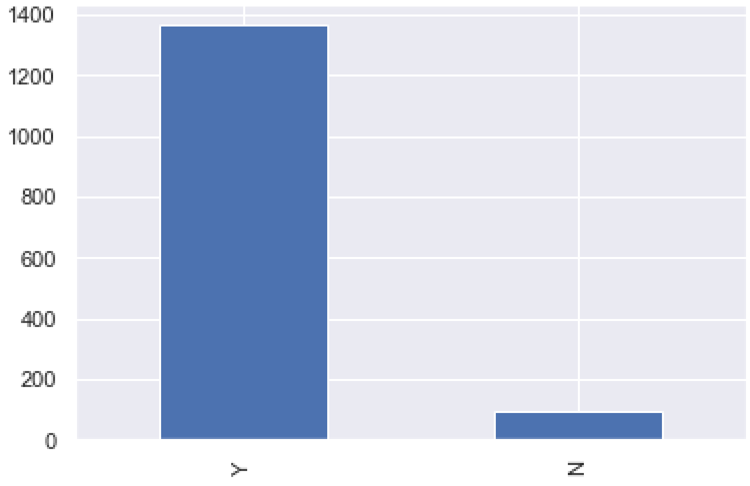
```
Classification Accuracy 	: 0.9444444444444444
True Positive Rate 	: 0.9442896935933147
False Positive Rate 	: 0.055710306406685235
True Negative Rate 	: 1.0
False Negative Rate 	: 0.0
```

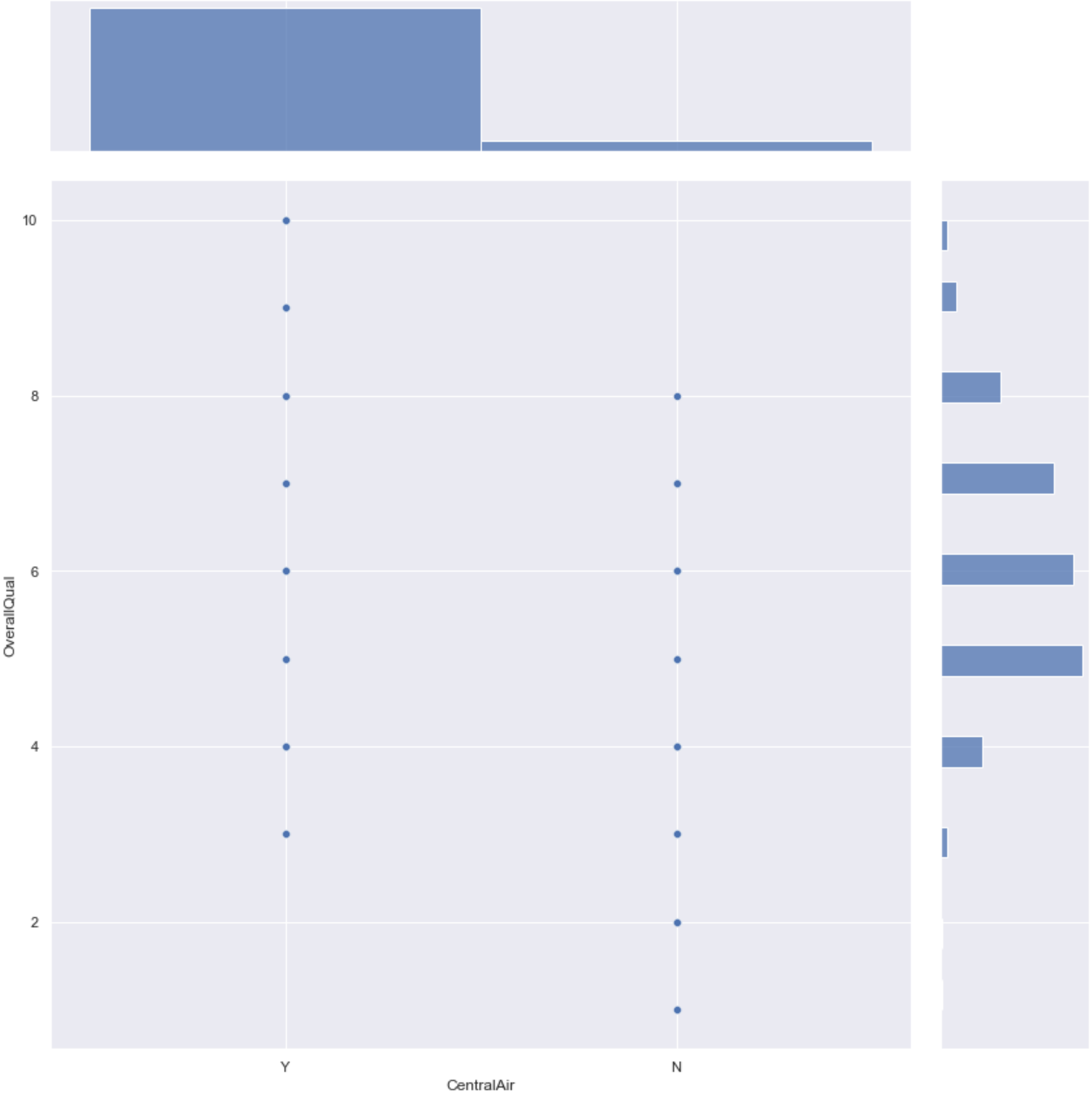


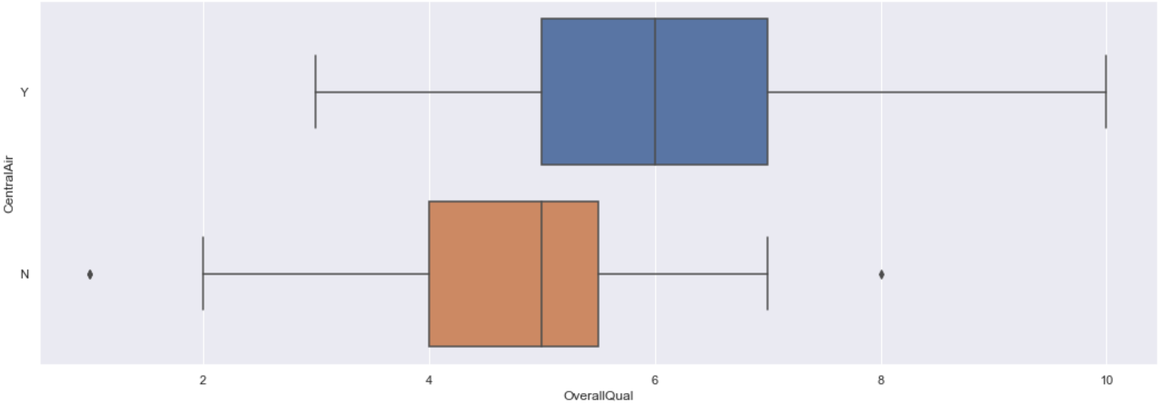
CentralAir AGAINST OverallQual

In [17]:

```
houseData = pd.read_csv('train.csv')
CentralAir = pd.DataFrame(houseData['CentralAir'])
houseData["CentralAir"].value_counts()

houseData["CentralAir"].value_counts().plot(kind='bar')

sb.jointplot(data = houseData,x = "CentralAir", y = "OverallQual",  heig

# Create a joint dataframe by concatenating
OverallQual = pd.DataFrame(houseData['OverallQual'])
concatDF = pd.concat([OverallQual, CentralAir], axis = 1).reindex(Overal

# Joint Boxplot of saleprice against central air
f = plt.figure(figsize=(18, 6))
sb.boxplot(x = "OverallQual", y = "CentralAir", data = concatDF, orient 

# Import Decision Tree Classifier model from Scikit-Learn
from sklearn.tree import DecisionTreeClassifier

# Create a Decision Tree Classifier object
dectree = DecisionTreeClassifier(max_depth = 2)

from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix

# Extract Response and Predictors
X = pd.DataFrame(houseData['OverallQual'])
y = pd.DataFrame(houseData['CentralAir'])
```




10
8
6
4
2
OverallQual
Y
N
CentralAir


Y
N
CentralAir
2
4
6
8
10
OverallQual

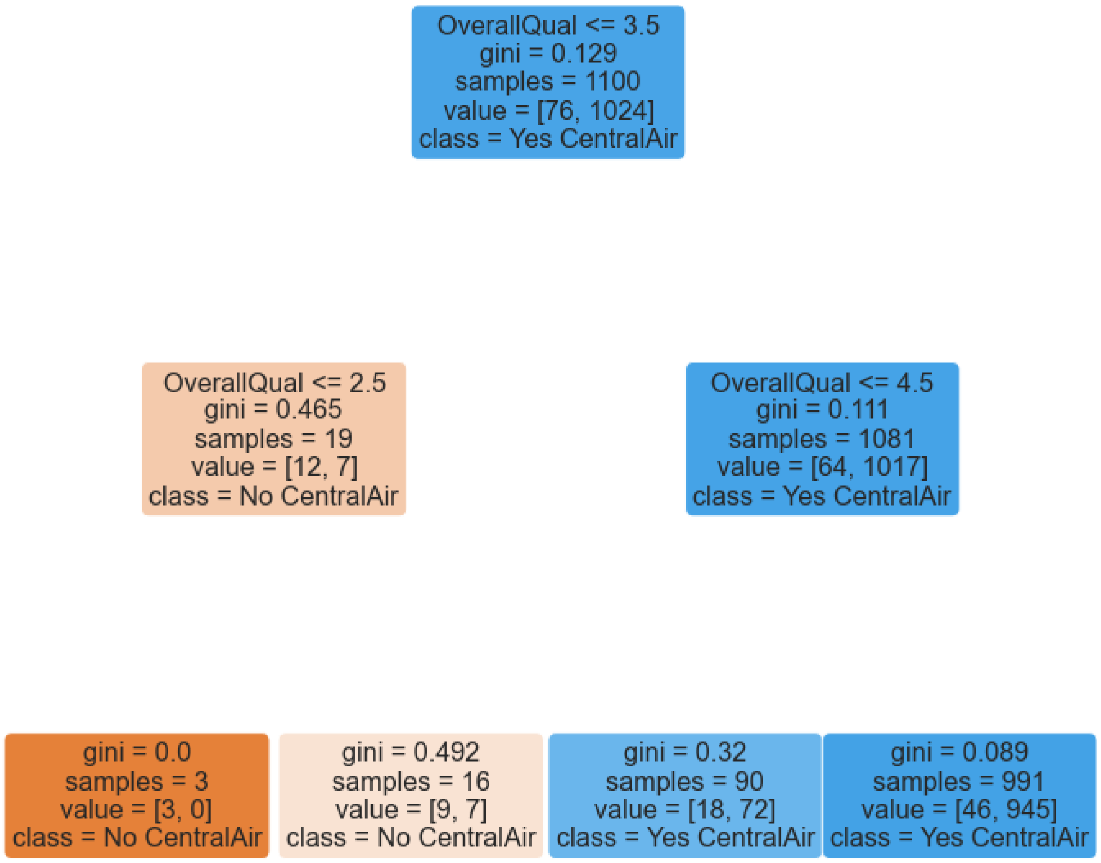
In [18]:

```python
# Split the Dataset into random Train and Test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 36

# Check the sample sizes
print("Train Set :", X_train.shape, y_train.shape)
print("Test Set  :", X_test.shape, y_test.shape)


# Decision Tree using Train Data
dectree = DecisionTreeClassifier(max_depth = 2)  # create the decision t
dectree.fit(X_train, y_train)                    # train the decision tr

# Plot the trained Decision Tree
from sklearn.tree import plot_tree

f = plt.figure(figsize=(12,12))
plot_tree(dectree, filled=True, rounded=True,
          feature_names=["OverallQual"],
          class_names=["No CentralAir","Yes CentralAir"])
```

```
Train Set : (1100, 1) (1100, 1)
Test Set  : (360, 1) (360, 1)
```

Out[18]:

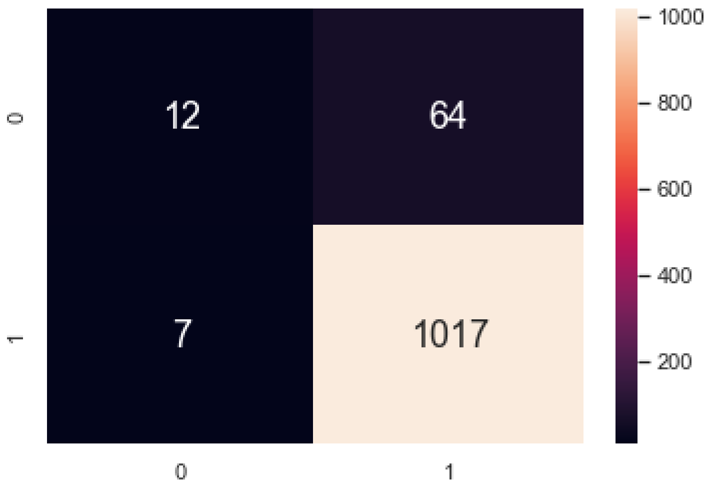
```
[Text(334.8, 543.6, 'OverallQual <= 3.5\ngini = 0.129\nsamples = 1100\nvalu
e = [76, 1024]\nclass = Yes CentralAir'),
 Text(167.4, 326.16, 'OverallQual <= 2.5\ngini = 0.465\nsamples = 19\nvalue
= [12, 7]\nclass = No CentralAir'),
 Text(83.7, 108.72000000000003, 'gini = 0.0\nsamples = 3\nvalue = [3, 0]\nc
lass = No CentralAir'),
 Text(251.10000000000002, 108.72000000000003, 'gini = 0.492\nsamples = 16\n
value = [9, 7]\nclass = No CentralAir'),
 Text(502.20000000000005, 326.16, 'OverallQual <= 4.5\ngini = 0.111\nsample
s = 1081\nvalue = [64, 1017]\nclass = Yes CentralAir'),
 Text(418.5, 108.72000000000003, 'gini = 0.32\nsamples = 90\nvalue = [18, 7
2]\nclass = Yes CentralAir'),
 Text(585.9, 108.72000000000003, 'gini = 0.089\nsamples = 991\nvalue = [46,
945]\nclass = Yes CentralAir')]
```


OverallQual <= 3.5
gini = 0.129
samples = 1100
value = [76, 1024]
class = Yes CentralAir
OverallQual <= 2.5
gini = 0.465
samples = 19
value = [12, 7]
class = No CentralAir
OverallQual <= 4.5
gini = 0.111
samples = 1081
value = [64, 1017]
class = Yes CentralAir
gini = 0.0
samples = 3
value = [3, 0]
class = No CentralAir
gini = 0.492
samples = 16
value = [9, 7]
class = No CentralAir
gini = 0.32
samples = 90
value = [18, 72]
class = Yes CentralAir
gini = 0.089
samples = 991
value = [46, 945]
class = Yes CentralAir

In [19]:

```
# Predict  CentralAir corresponding to Saleprice
y_train_pred = dectree.predict(X_train)


# Plot the two-way Confusion Matrix
from sklearn.metrics import confusion_matrix
sb.heatmap(confusion_matrix(y_train, y_train_pred),
           annot = True, fmt=".0f", annot_kws={"size": 18})

# Print the Classification Accuracy
print("Classification Accuracy \t:", dectree.score(X_train, y_train))

TN = confusion_matrix(y_train, y_train_pred)[0][0]
FP = confusion_matrix(y_train, y_train_pred)[0][1]
FN = confusion_matrix(y_train, y_train_pred)[1][0]
TP = confusion_matrix(y_train, y_train_pred)[1][1]

TPR = TP/(TP+FP)
FPR = FP/(TP+FP)
TNR = TN/(TN+FN)
FNR = FN/(TN+FN)

print("True Positive Rate \t:", TPR)
print("False Positive Rate \t:", FPR)
print("True Negative Rate \t:", TNR)
print("False Negative Rate \t:", FNR)
```

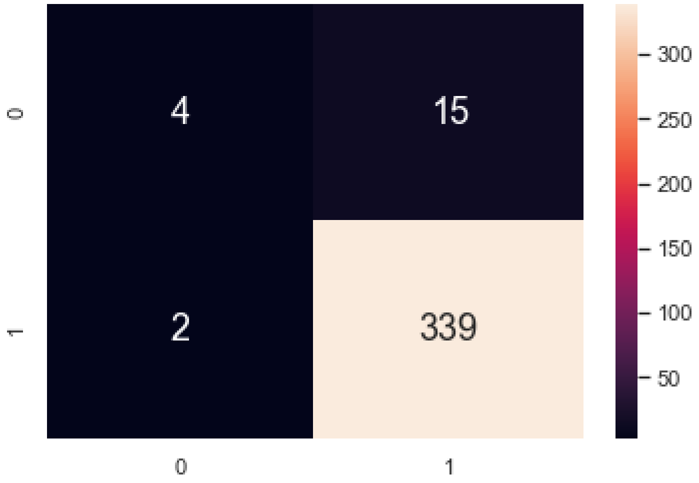
```
Classification Accuracy         : 0.9354545454545454
True Positive Rate      : 0.940795559666975
False Positive Rate     : 0.05920444033302498
True Negative Rate      : 0.631578947368421
False Negative Rate     : 0.3684210526315789
```



In [20]:

```python
# Predict  CentralAir corresponding to Saleprice
y_test_pred = dectree.predict(X_test)


# Plot the two-way Confusion Matrix
from sklearn.metrics import confusion_matrix
sb.heatmap(confusion_matrix(y_test, y_test_pred),
           annot = True, fmt=".0f", annot_kws={"size": 18})
# Print the Classification Accuracy
print("Classification Accuracy \t:", dectree.score(X_test, y_test))

TN = confusion_matrix(y_test, y_test_pred)[0][0]
FP = confusion_matrix(y_test, y_test_pred)[0][1]
FN = confusion_matrix(y_test, y_test_pred)[1][0]
TP = confusion_matrix(y_test, y_test_pred)[1][1]

TPR = TP/(TP+FP)
FPR = FP/(TP+FP)
TNR = TN/(TN+FN)
FNR = FN/(TN+FN)

print("True Positive Rate \t:", TPR)
print("False Positive Rate \t:", FPR)
print("True Negative Rate \t:", TNR)
print("False Negative Rate \t:", FNR)

```

```
Classification Accuracy         : 0.9527777777777777
True Positive Rate      : 0.9576271186440678
False Positive Rate     : 0.0423728813559322
True Negative Rate      : 0.6666666666666666
False Negative Rate     : 0.3333333333333333
```

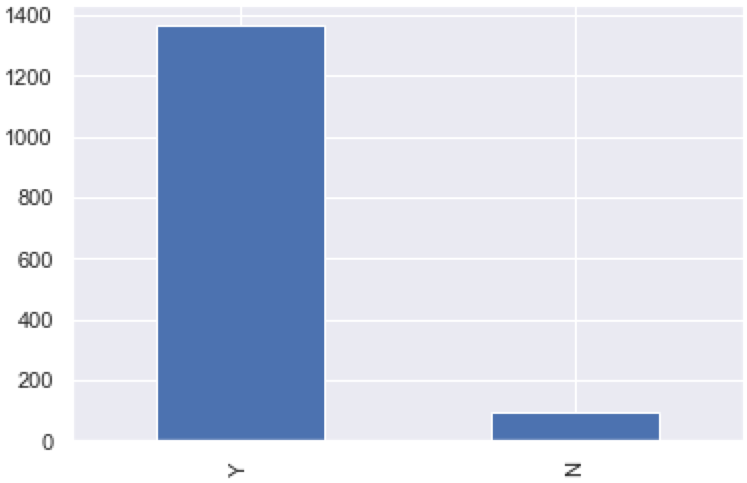


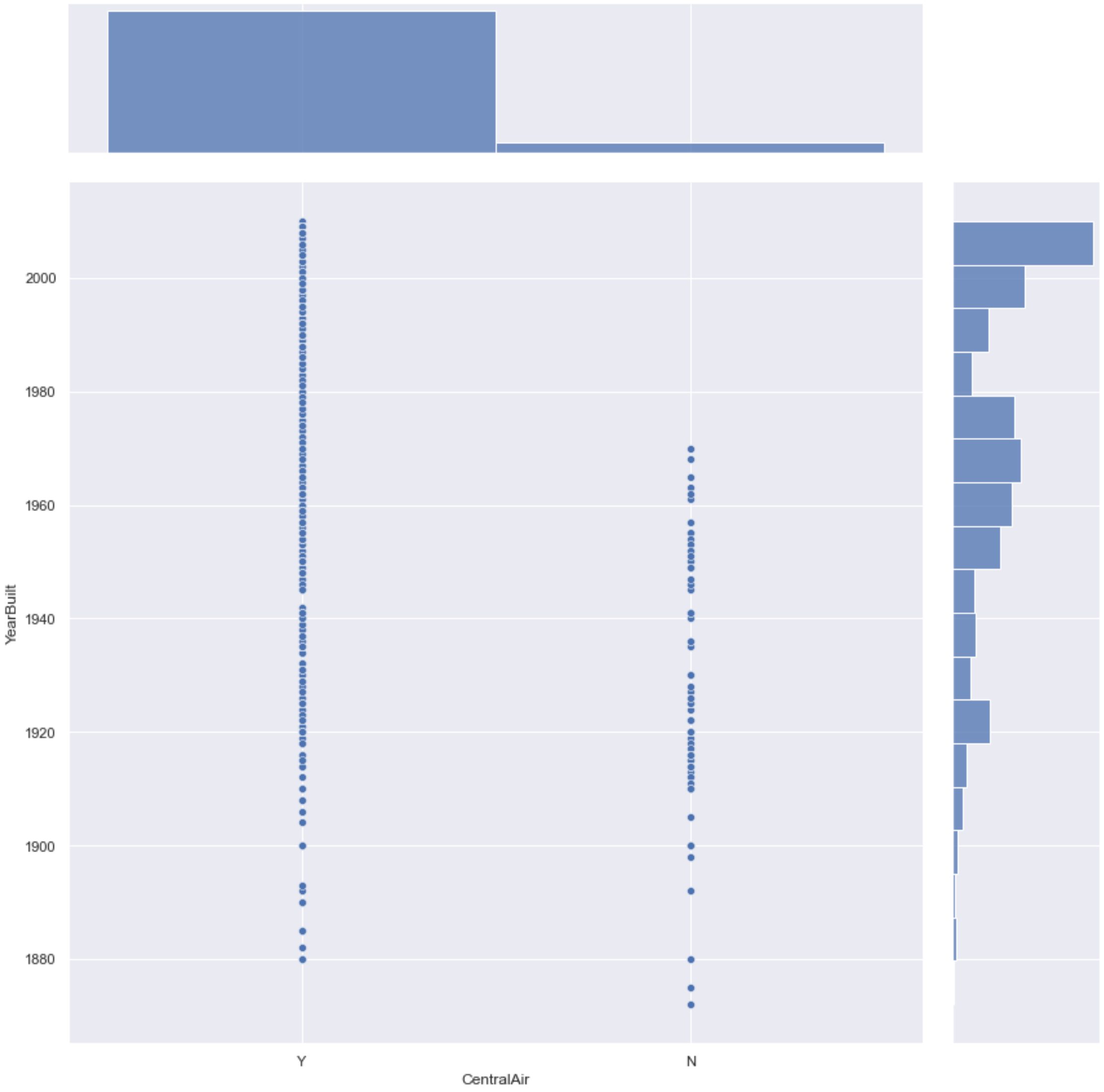
CentralAir AGAINST YearBuilt

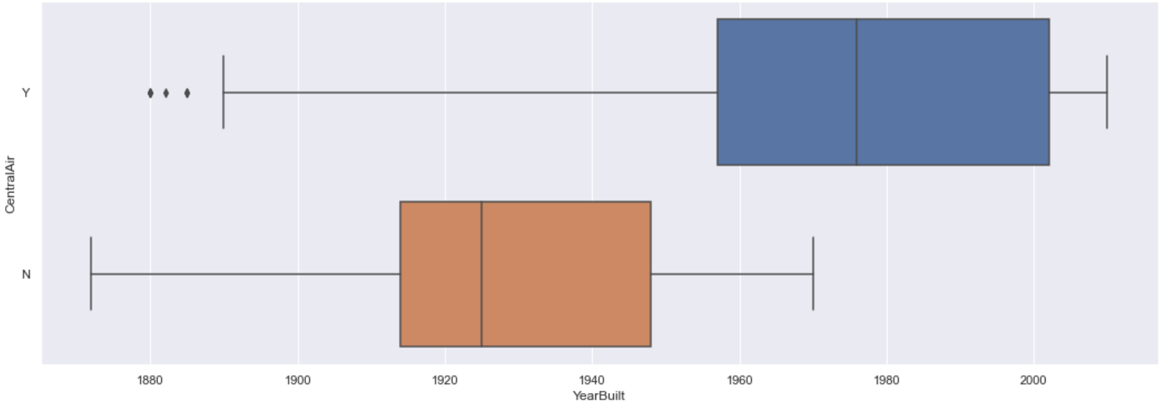
In [21]:

```python
houseData = pd.read_csv('train.csv')
CentralAir = pd.DataFrame(houseData['CentralAir'])
houseData["CentralAir"].value_counts()

houseData["CentralAir"].value_counts().plot(kind='bar')

sb.jointplot(data = houseData,x = "CentralAir", y = "YearBuilt",  height

# Create a joint dataframe by concatenating
YearBuilt = pd.DataFrame(houseData['YearBuilt'])
concatDF = pd.concat([YearBuilt, CentralAir], axis = 1).reindex(YearBuil

# Joint Boxplot of saleprice against central air
f = plt.figure(figsize=(18, 6))
sb.boxplot(x = "YearBuilt", y = "CentralAir", data = concatDF, orient = '

# Import Decision Tree Classifier model from Scikit-Learn
from sklearn.tree import DecisionTreeClassifier

# Create a Decision Tree Classifier object
dectree = DecisionTreeClassifier(max_depth = 2)

from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix

# Extract Response and Predictors
X = pd.DataFrame(houseData['YearBuilt'])
y = pd.DataFrame(houseData['CentralAir'])

# Split the Dataset into random Train and Test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 36

# Check the sample sizes
print("Train Set :", X_train.shape, y_train.shape)
print("Test Set  :", X_test.shape, y_test.shape)

```

```
Train Set : (1100, 1) (1100, 1)
Test Set  : (360, 1) (360, 1)
```




2000
1980
1960
1940
1920
1900
1880
YearBuilt
Y
N
CentralAir

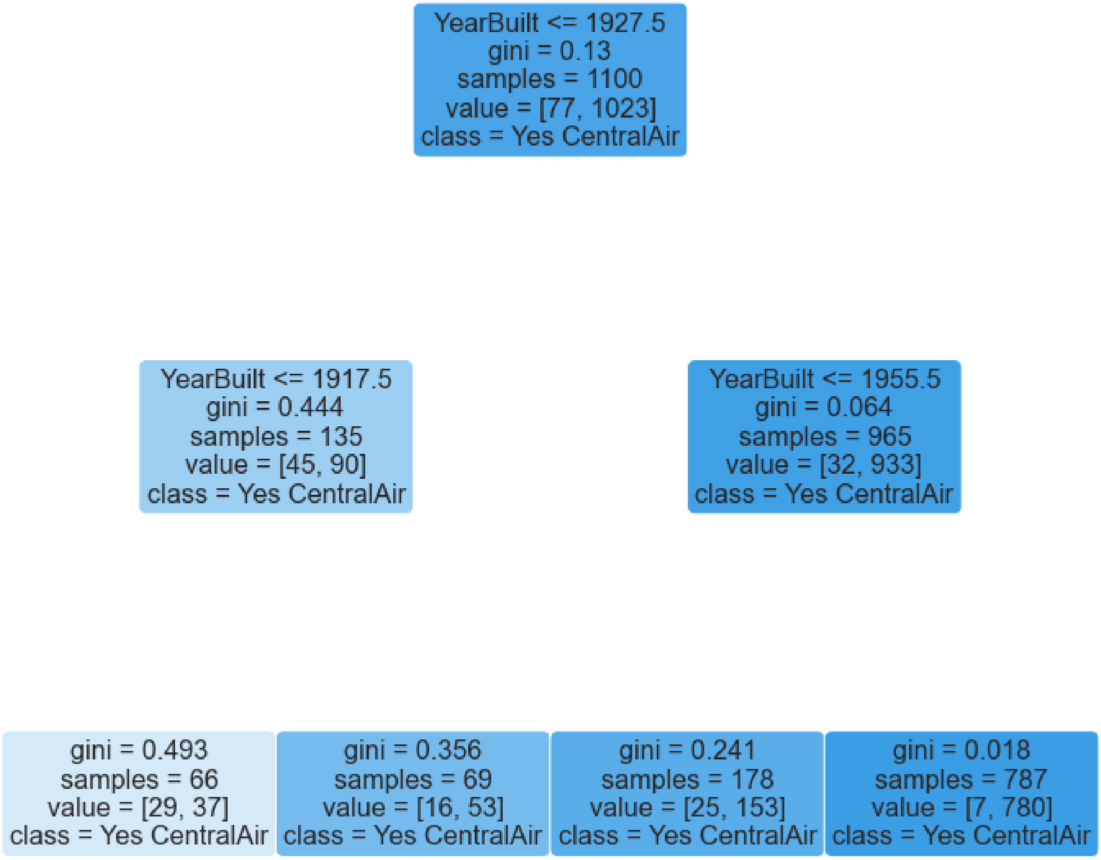

Y
N
CentralAir
1880
1900
1920
1940
1960
1980
2000
YearBuilt

In [22]:

```python
# Decision Tree using Train Data
dectree = DecisionTreeClassifier(max_depth = 2)  # create the decision tr
dectree.fit(X_train, y_train)                    # train the decision tre

# Plot the trained Decision Tree
from sklearn.tree import plot_tree

f = plt.figure(figsize=(12,12))
plot_tree(dectree, filled=True, rounded=True,
          feature_names=["YearBuilt"],
          class_names=["No CentralAir","Yes CentralAir"])
```

Out[22]:

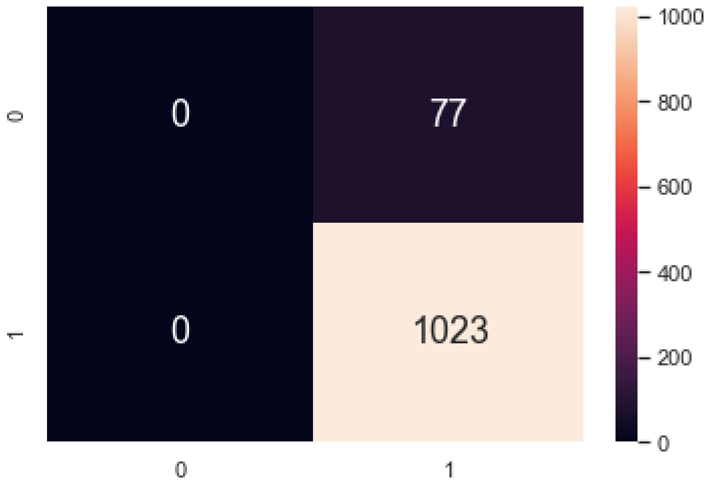
```
[Text(334.8, 543.6, 'YearBuilt <= 1927.5\ngini = 0.13\nsamples = 1100\nvalu
e = [77, 1023]\nclass = Yes CentralAir'),
 Text(167.4, 326.16, 'YearBuilt <= 1917.5\ngini = 0.444\nsamples = 135\nval
ue = [45, 90]\nclass = Yes CentralAir'),
 Text(83.7, 108.72000000000003, 'gini = 0.493\nsamples = 66\nvalue = [29, 3
7]\nclass = Yes CentralAir'),
 Text(251.10000000000002, 108.72000000000003, 'gini = 0.356\nsamples = 69\n
value = [16, 53]\nclass = Yes CentralAir'),
 Text(502.20000000000005, 326.16, 'YearBuilt <= 1955.5\ngini = 0.064\nsampl
es = 965\nvalue = [32, 933]\nclass = Yes CentralAir'),
 Text(418.5, 108.72000000000003, 'gini = 0.241\nsamples = 178\nvalue = [25,
153]\nclass = Yes CentralAir'),
 Text(585.9, 108.72000000000003, 'gini = 0.018\nsamples = 787\nvalue = [7,
780]\nclass = Yes CentralAir')]
```


YearBuilt <= 1927.5
gini = 0.13
samples = 1100
value = [77, 1023]
class = Yes CentralAir
YearBuilt <= 1917.5
gini = 0.444
samples = 135
value = [45, 90]
class = Yes CentralAir
YearBuilt <= 1955.5
gini = 0.064
samples = 965
value = [32, 933]
class = Yes CentralAir
gini = 0.493
samples = 66
value = [29, 37]
class = Yes CentralAir
gini = 0.356
samples = 69
value = [16, 53]
class = Yes CentralAir
gini = 0.241
samples = 178
value = [25, 153]
class = Yes CentralAir
gini = 0.018
samples = 787
value = [7, 780]
class = Yes CentralAir

In [23]:

```
# Predict  CentralAir corresponding to Saleprice
y_train_pred = dectree.predict(X_train)


# Plot the two-way Confusion Matrix
from sklearn.metrics import confusion_matrix
sb.heatmap(confusion_matrix(y_train, y_train_pred),
           annot = True, fmt=".0f", annot_kws={"size": 18})

# Print the Classification Accuracy
print("Classification Accuracy \t:", dectree.score(X_train, y_train))

TN = confusion_matrix(y_train, y_train_pred)[0][0]
FP = confusion_matrix(y_train, y_train_pred)[0][1]
FN = confusion_matrix(y_train, y_train_pred)[1][0]
TP = confusion_matrix(y_train, y_train_pred)[1][1]

TPR = TP/(TP+FP)
FPR = FP/(TP+FP)
TNR = TN/(TN+FN)
FNR = FN/(TN+FN)

print("True Positive Rate \t:", TPR)
print("False Positive Rate \t:", FPR)
print("True Negative Rate \t:", TNR)
print("False Negative Rate \t:", FNR)
```

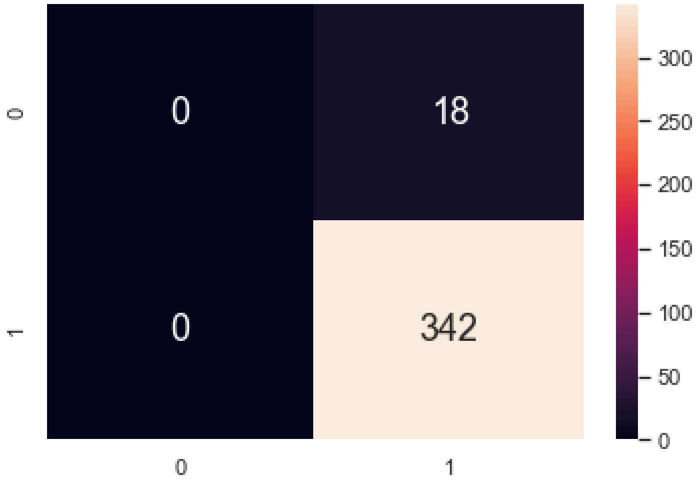
```
Classification Accuracy          : 0.93
True Positive Rate       : 0.93
False Positive Rate      : 0.07
True Negative Rate       : nan
False Negative Rate      : nan

<ipython-input-23-3724eeecaa86>:20: RuntimeWarning: invalid value encounter
ed in longlong_scalars
  TNR = TN/(TN+FN)
<ipython-input-23-3724eeecaa86>:21: RuntimeWarning: invalid value encounter
ed in longlong_scalars
  FNR = FN/(TN+FN)
```



In [24]:

```python
# Predict  CentralAir corresponding to Saleprice
y_test_pred = dectree.predict(X_test)


# Plot the two-way Confusion Matrix
from sklearn.metrics import confusion_matrix
sb.heatmap(confusion_matrix(y_test, y_test_pred),
           annot = True, fmt=".0f", annot_kws={"size": 18})
# Print the Classification Accuracy
print("Classification Accuracy \t:", dectree.score(X_test, y_test))

TN = confusion_matrix(y_test, y_test_pred)[0][0]
FP = confusion_matrix(y_test, y_test_pred)[0][1]
FN = confusion_matrix(y_test, y_test_pred)[1][0]
TP = confusion_matrix(y_test, y_test_pred)[1][1]

TPR = TP/(TP+FP)
FPR = FP/(TP+FP)
TNR = TN/(TN+FN)
FNR = FN/(TN+FN)

print("True Positive Rate \t:", TPR)
print("False Positive Rate \t:", FPR)
print("True Negative Rate \t:", TNR)
print("False Negative Rate \t:", FNR)

```

```
Classification Accuracy         : 0.95
True Positive Rate      : 0.95
False Positive Rate     : 0.05
True Negative Rate      : nan
False Negative Rate     : nan


<ipython-input-24-a4f71b0bf701>:19: RuntimeWarning: invalid value encounter
ed in longlong_scalars
  TNR = TN/(TN+FN)
<ipython-input-24-a4f71b0bf701>:20: RuntimeWarning: invalid value encounter
ed in longlong_scalars
  FNR = FN/(TN+FN)
```



Problem 3 : Best Uni-Variate Model to Predict CentralAir

Compare and contrast the four models in terms of Classification Accuracy, True Positive Rate and False Positive Rate on both Train and Test Data to comment on which univariate classification tree you think is the best to predict “CentralAir”.

CentralAir Against YearBuilt

Train Data: Classification Accuracy : 0.9409090909090909 True Positive Rate : 1 False Positive Rate : 1 True Negative Rate : 0 False Negative Rate : 0

Test Data: Classification Accuracy : 0.9166666666666666 True Positive Rate : 1 False Positive Rate : 1 True Negative Rate : 0 False Negative Rate : 0

CentralAir Against OverallQual

Train Data: Classification Accuracy : 0.94 True Positive Rate : 0.9444444444444444 False Positive Rate : 0.05555555555555555 True Negative Rate : 0.7 False Negative Rate : 0.3

Test Data: Classification Accuracy : 0.9388888888888889 True Positive Rate : 0.9464788732394366 False Positive Rate : 0.05352112676056338 True Negative Rate : 0.4 False Negative Rate : 0.6

CentralAir Against GrLivArea

Train Data: Classification Accuracy : 0.9436363636363636 True Positive Rate : 0.9434822242479489 False Positive Rate : 0.056517775752051046 True Negative Rate : 1.0 False Negative Rate : 0.0

Test Data: Classification Accuracy : 0.9194444444444444 True Positive Rate : 0.9192200557103064 False Positive Rate : 0.0807799442896936 True Negative Rate : 1.0 False Negative Rate : 0.0

Central Air Against SalePrice

Train Data: Classification Accuracy : 0.9445454545454546 True Positive Rate : 0.945821854912764 False Positive Rate : 0.05417814508723599 True Negative Rate : 0.8181818181818182 False Negative Rate : 0.18181818181818182

Test Data: Classification Accuracy : 0.9361111111111111 True Positive Rate : 0.9353932584269663 False Positive Rate : 0.06460674157303371 True Negative Rate : 1.0 False Negative Rate : 0.0

Conclusion

For CentralAir against YearBuilt, both the True and False positive rate for both train and test set are 1 / nearing 1; which is abnormally high. Hence, although the prediction is mostly positive, it may not be true, hence, YearBuilt is not reliable to predict CentralAir.

Among the three remaining factors, the True positive rate and False positive rate for both train and test set data are very similar, thus making it more difficult to pinpoint which is more reliable.

The classification accuracy of the test and train data for all 4 variables are very similar, differing only by the 2nd decimal place. SalePrice has the highest Classification Accuracy compared to the other variables, making it a bit more accurate.

Thus, SalePrice should be used to predict CentralAir.