TIME SERIES FORECASTING

BUSINESS REPORT

-----------------\*\*\*\*\*\*------------------

**BY:- ANIKET HIRGUDE**

***INDEX***

**1. Read the data as an appropriate Time Series data and plot the data.**

**2. Perform appropriate Exploratory Data Analysis to understand the data and also perform decomposition.**

**3. Split the data into training and test. The test data should start in 1991.**

**4. Build all the exponential smoothing models on the training data and evaluate the model using RMSE on the test data. Other models such as regression,naïve forecast models and simple average models. should also be built on the training data and check the performance on the test data using RMSE.**

**5. Check for the stationarity of the data on which the model is being built on using appropriate statistical tests and also mention the hypothesis for the statistical test. If the data is found to be non-stationary, take appropriate steps to make it stationary. Check the new data for stationarity and comment. Note: Stationarity should be checked at alpha = 0.05.**

**6. Build an automated version of the ARIMA/SARIMA model in which the parameters are selected using the lowest Akaike Information Criteria (AIC) on the training data and evaluate this model on the test data using RMSE.**

**7. Build ARIMA/SARIMA models based on the cut-off points of ACF and PACF on the training data and evaluate this model on the test data using RMSE.**

**8. Build a table (create a data frame) with all the models built along with their corresponding parameters and the respective RMSE values on the test data.**

**9. Based on the model-building exercise, build the most optimum model(s) on the complete data and predict 12 months into the future with appropriate confidence intervals/bands.**

***10. Comment on the model thus built and report your findings and suggest the measures that the company should be taking for future sales.***

**Problem 1**: SPARKLING

**1. Read the data as an appropriate Time Series data and plot the data.**

|  | **YearMonth** | **Sparkling** |
| --- | --- | --- |
| **0** | 1980-01 | 1686 |
| **1** | 1980-02 | 1591 |
| **2** | 1980-03 | 2304 |
| **3** | 1980-04 | 1712 |
| **4** | 1980-05 | 1471 |

TABLE 1

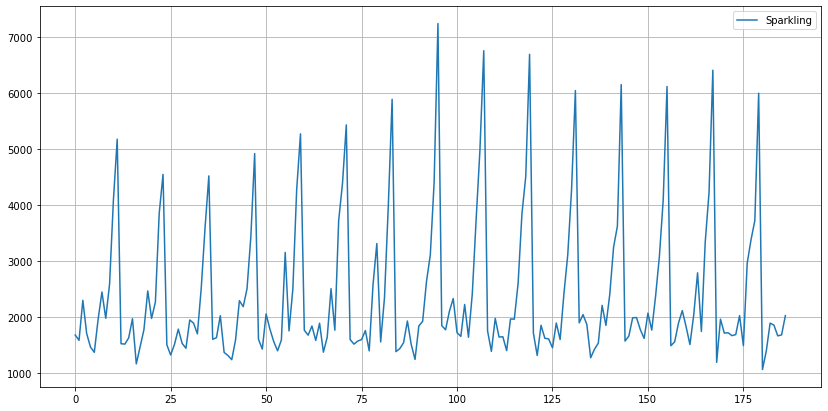
CHECKING NULL VALUES

YearMonth 0

Sparkling 0

dtype: int64

PLOT A GRAPH OF THE DATA



GRAPH 1

Though the above plot looks like a Time Series plot, notice that the X-Axis is not time. In order to make the X-Axis as a Time Series, we need to pass the date range manually through a command in Pandas.

CONVERT THE INTO MONTH

DatetimeIndex(['1980-01-31', '1980-02-29', '1980-03-31', '1980-04-30',

'1980-05-31', '1980-06-30', '1980-07-31', '1980-08-31',

'1980-09-30', '1980-10-31',

...

'1994-10-31', '1994-11-30', '1994-12-31', '1995-01-31',

'1995-02-28', '1995-03-31', '1995-04-30', '1995-05-31',

'1995-06-30', '1995-07-31'],

dtype='datetime64[ns]', length=187, freq='M')

MONTHLY DATASET

|  | **YearMonth** | **Sparkling** | **Time\_Stamp** |
| --- | --- | --- | --- |
| **0** | 1980-01 | 1686 | 1980-01-31 |
| **1** | 1980-02 | 1591 | 1980-02-29 |
| **2** | 1980-03 | 2304 | 1980-03-31 |
| **3** | 1980-04 | 1712 | 1980-04-30 |
| **4** | 1980-05 | 1471 | 1980-05-31 |

TABLE 2

MAKING INDEX OF TIMESTAMP

|  | **YearMonth** | **Sparkling** |
| --- | --- | --- |
| **Time\_Stamp** |  |  |
| **1980-01-31** | 1980-01 | 1686 |
| **1980-02-29** | 1980-02 | 1591 |
| **1980-03-31** | 1980-03 | 2304 |
| **1980-04-30** | 1980-04 | 1712 |
| **1980-05-31** | 1980-05 | 1471 |

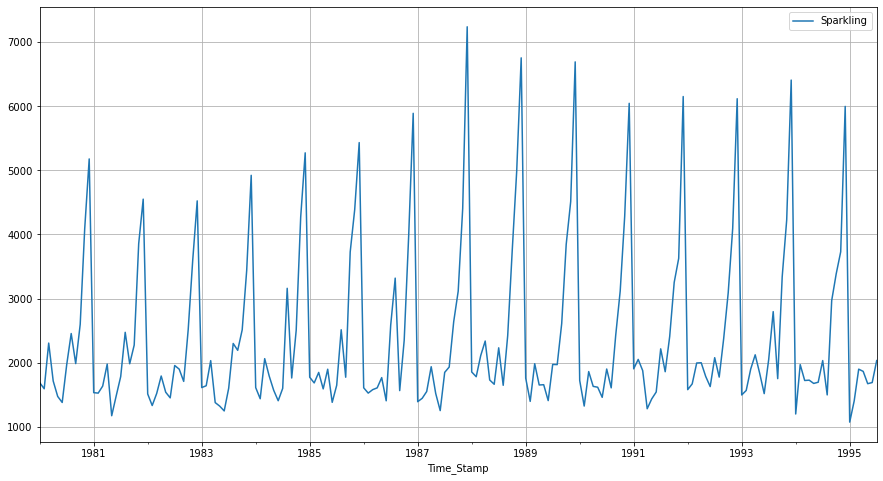
TABLE 3

FINAL DATASET

|  | **Sparkling** |
| --- | --- |
| **Time\_Stamp** |  |
| **1980-01-31** | 1686 |
| **1980-02-29** | 1591 |
| **1980-03-31** | 2304 |
| **1980-04-30** | 1712 |
| **1980-05-31** | 1471 |

TABLE 4

PLOTTING A GRAPH OF FINAL DATASET



GRAPH 2

# 2. Perform appropriate Exploratory Data Analysis to understand the data and also perform decomposition.

LETS DESCRIBE THE DATA

|  | **Sparkling** |
| --- | --- |
| **count** | 187.000000 |
| **mean** | 2402.417112 |
| **std** | 1295.111540 |
| **min** | 1070.000000 |
| **25%** | 1605.000000 |
| **50%** | 1874.000000 |
| **75%** | 2549.000000 |
| **max** | 7242.000000 |

TABLE 5

INFORMATION OF THE DATA

<class 'pandas.core.frame.DataFrame'>

DatetimeIndex: 187 entries, 1980-01-31 to 1995-07-31

Data columns (total 1 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Sparkling 187 non-null int64

dtypes: int64(1)

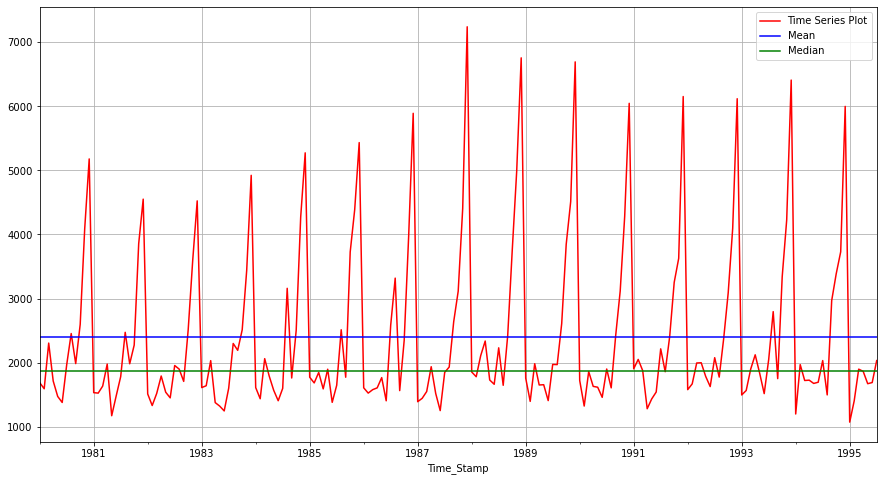
memory usage: 2.9 KB

SHAPE OF THE DAT

(187, 1)

In [189]:

PLOTTING A GRAPH TO SEE THE MEAN AND MEDIAN



GRAPH 3

CHECKING FOR THE INDEX

<bound method \_inherit\_from\_data.<locals>.method of DatetimeIndex(['1980-01-31', '1980-02-29', '1980-03-31', '1980-04-30',

'1980-05-31', '1980-06-30', '1980-07-31', '1980-08-31',

'1980-09-30', '1980-10-31',

...

'1994-10-31', '1994-11-30', '1994-12-31', '1995-01-31',

'1995-02-28', '1995-03-31', '1995-04-30', '1995-05-31',

'1995-06-30', '1995-07-31'],

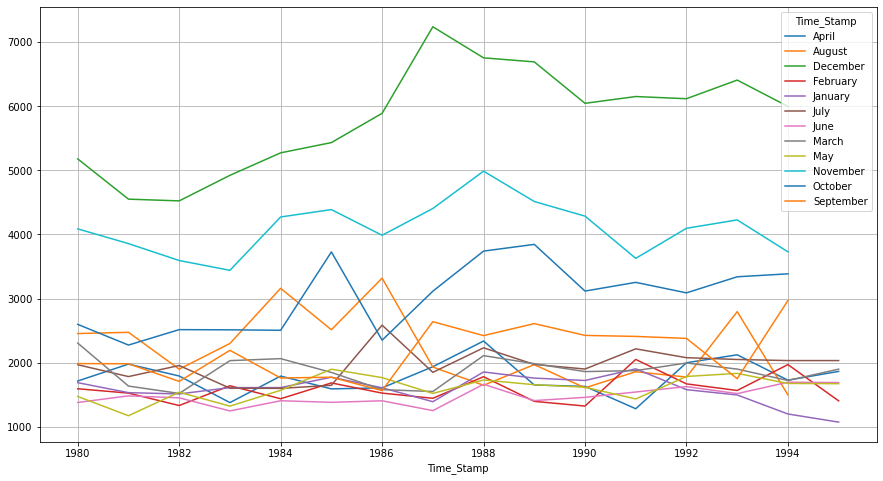
dtype='datetime64[ns]', name='Time\_Stamp', length=187, freq=None)>

CHECKING FOR THE MONTHLY DATASET

| **Time\_Stamp** | **April** | **August** | **December** | **February** | **January** | **July** | **June** | **March** | **May** | **November** | **October** | **September** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Time\_Stamp** |  |  |  |  |  |  |  |  |  |  |  |  |
| **1980** | 1712.0 | 2453.0 | 5179.0 | 1591.0 | 1686.0 | 1966.0 | 1377.0 | 2304.0 | 1471.0 | 4087.0 | 2596.0 | 1984.0 |
| **1981** | 1976.0 | 2472.0 | 4551.0 | 1523.0 | 1530.0 | 1781.0 | 1480.0 | 1633.0 | 1170.0 | 3857.0 | 2273.0 | 1981.0 |
| **1982** | 1790.0 | 1897.0 | 4524.0 | 1329.0 | 1510.0 | 1954.0 | 1449.0 | 1518.0 | 1537.0 | 3593.0 | 2514.0 | 1706.0 |
| **1983** | 1375.0 | 2298.0 | 4923.0 | 1638.0 | 1609.0 | 1600.0 | 1245.0 | 2030.0 | 1320.0 | 3440.0 | 2511.0 | 2191.0 |
| **1984** | 1789.0 | 3159.0 | 5274.0 | 1435.0 | 1609.0 | 1597.0 | 1404.0 | 2061.0 | 1567.0 | 4273.0 | 2504.0 | 1759.0 |
| **1985** | 1589.0 | 2512.0 | 5434.0 | 1682.0 | 1771.0 | 1645.0 | 1379.0 | 1846.0 | 1896.0 | 4388.0 | 3727.0 | 1771.0 |
| **1986** | 1605.0 | 3318.0 | 5891.0 | 1523.0 | 1606.0 | 2584.0 | 1403.0 | 1577.0 | 1765.0 | 3987.0 | 2349.0 | 1562.0 |
| **1987** | 1935.0 | 1930.0 | 7242.0 | 1442.0 | 1389.0 | 1847.0 | 1250.0 | 1548.0 | 1518.0 | 4405.0 | 3114.0 | 2638.0 |
| **1988** | 2336.0 | 1645.0 | 6757.0 | 1779.0 | 1853.0 | 2230.0 | 1661.0 | 2108.0 | 1728.0 | 4988.0 | 3740.0 | 2421.0 |
| **1989** | 1650.0 | 1968.0 | 6694.0 | 1394.0 | 1757.0 | 1971.0 | 1406.0 | 1982.0 | 1654.0 | 4514.0 | 3845.0 | 2608.0 |
| **1990** | 1628.0 | 1605.0 | 6047.0 | 1321.0 | 1720.0 | 1899.0 | 1457.0 | 1859.0 | 1615.0 | 4286.0 | 3116.0 | 2424.0 |
| **1991** | 1279.0 | 1857.0 | 6153.0 | 2049.0 | 1902.0 | 2214.0 | 1540.0 | 1874.0 | 1432.0 | 3627.0 | 3252.0 | 2408.0 |
| **1992** | 1997.0 | 1773.0 | 6119.0 | 1667.0 | 1577.0 | 2076.0 | 1625.0 | 1993.0 | 1783.0 | 4096.0 | 3088.0 | 2377.0 |
| **1993** | 2121.0 | 2795.0 | 6410.0 | 1564.0 | 1494.0 | 2048.0 | 1515.0 | 1898.0 | 1831.0 | 4227.0 | 3339.0 | 1749.0 |
| **1994** | 1725.0 | 1495.0 | 5999.0 | 1968.0 | 1197.0 | 2031.0 | 1693.0 | 1720.0 | 1674.0 | 3729.0 | 3385.0 | 2968.0 |
| **1995** | 1862.0 | NaN | NaN | 1402.0 | 1070.0 | 2031.0 | 1688.0 | 1897.0 | 1670.0 | NaN | NaN | NaN |

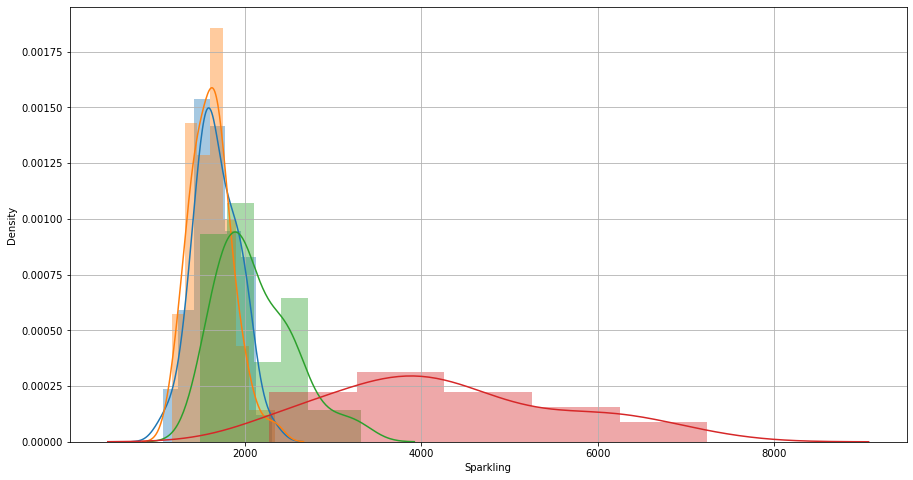
TABLE 6

QUARTERLY DATA

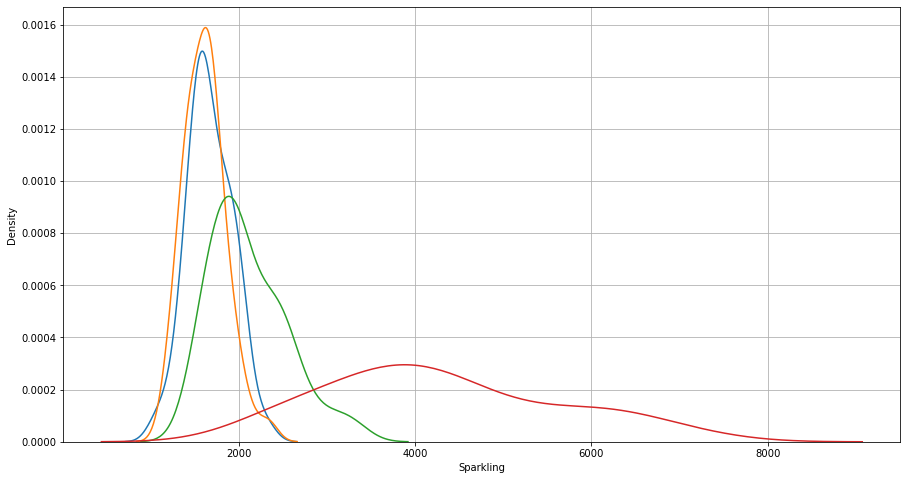


GRAPH 4

ON DISTPLOT

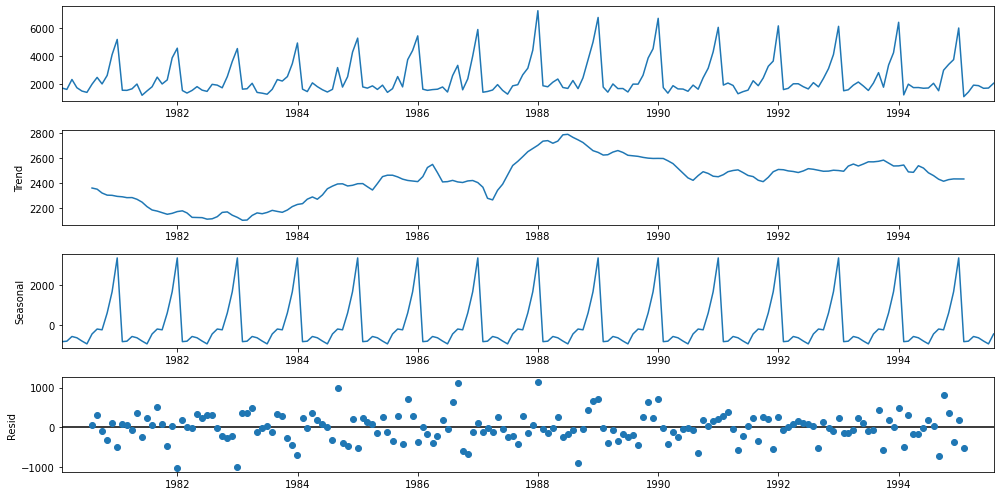


GRAPH 5



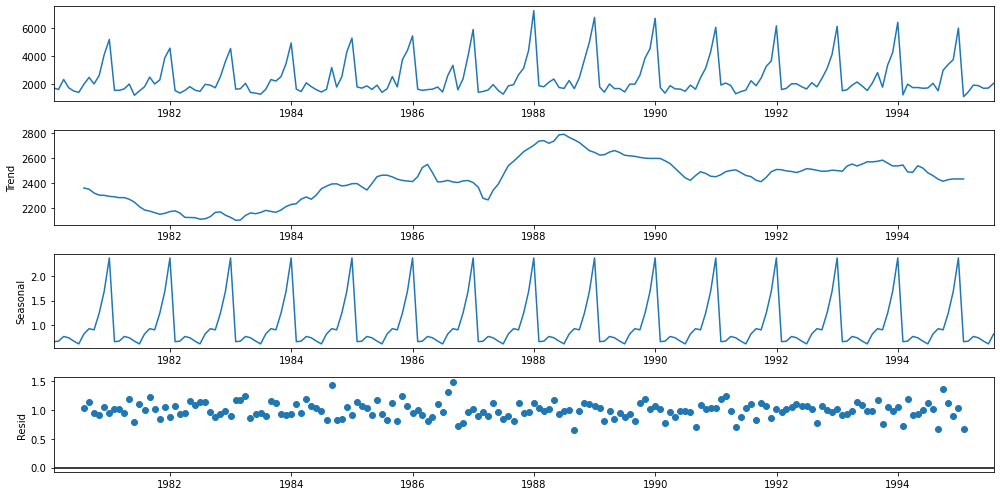
GRAPH 6

DECOMPOSITION FOR ADDITIVE



GRAPH 7

DECOMPOSITION FOR MULTIPICATIVE



GRAPH 8

# 3. Split the data into training and test. The test data should start in 1991.

SHAPE OF THE TRAIN AND TEST DATASET

(132, 1)

(55, 1)

In [29]:

TRAINING DATASET

|  | **Sparkling** |
| --- | --- |
| **Time\_Stamp** |  |
| **1980-01-31** | 1686 |
| **1980-02-29** | 1591 |
| **1980-03-31** | 2304 |
| **1980-04-30** | 1712 |
| **1980-05-31** | 1471 |
| **...** | ... |
| **1990-08-31** | 1605 |
| **1990-09-30** | 2424 |
| **1990-10-31** | 3116 |
| **1990-11-30** | 4286 |
| **1990-12-31** | 6047 |

TEST DATASET

|  | **Sparkling** |
| --- | --- |
| **Time\_Stamp** |  |
| **1991-01-31** | 1902 |
| **1991-02-28** | 2049 |
| **1991-03-31** | 1874 |
| **1991-04-30** | 1279 |
| **1991-05-31** | 1432 |
| **1991-06-30** | 1540 |
| **1991-07-31** | 2214 |
| **1991-08-31** | 1857 |
| **1991-09-30** | 2408 |
| **1991-10-31** | 3252 |
| **1991-11-30** | 3627 |
| **1991-12-31** | 6153 |
| **1992-01-31** | 1577 |
| **1992-02-29** | 1667 |
| **1992-03-31** | 1993 |
| **1992-04-30** | 1997 |
| **1992-05-31** | 1783 |
| **1992-06-30** | 1625 |
| **1992-07-31** | 2076 |
| **1992-08-31** | 1773 |
| **1992-09-30** | 2377 |
| **1992-10-31** | 3088 |
| **1992-11-30** | 4096 |
| **1992-12-31** | 6119 |
| **1993-01-31** | 1494 |
| **1993-02-28** | 1564 |
| **1993-03-31** | 1898 |
| **1993-04-30** | 2121 |
| **1993-05-31** | 1831 |
| **1993-06-30** | 1515 |
| **1993-07-31** | 2048 |
| **1993-08-31** | 2795 |
| **1993-09-30** | 1749 |
| **1993-10-31** | 3339 |
| **1993-11-30** | 4227 |
| **1993-12-31** | 6410 |
| **1994-01-31** | 1197 |
| **1994-02-28** | 1968 |
| **1994-03-31** | 1720 |
| **1994-04-30** | 1725 |
| **1994-05-31** | 1674 |
| **1994-06-30** | 1693 |
| **1994-07-31** | 2031 |
| **1994-08-31** | 1495 |
| **1994-09-30** | 2968 |
| **1994-10-31** | 3385 |
| **1994-11-30** | 3729 |
| **1994-12-31** | 5999 |
| **1995-01-31** | 1070 |
| **1995-02-28** | 1402 |
| **1995-03-31** | 1897 |
| **1995-04-30** | 1862 |
| **1995-05-31** | 1670 |
| **1995-06-30** | 1688 |
| **1995-07-31** | 2031 |

In [ ]:

# 4. Build all the exponential smoothing models on the training data and evaluate the model using RMSE on the test data. Other models such as regression,naïve forecast models and simple average models. should also be built on the training data and check the performance on the test data using RMSE.

SES - ETS(A, N, N) - Simple Exponential Smoothing with additive errors

The simplest of the exponentially smoothing methods is naturally called simple exponential smoothing (SES).

This method is suitable for forecasting data with no clear trend or seasonal pattern.

In Single ES, the forecast at time (t + 1) is given by Winters,1960

𝐹𝑡+1=𝛼𝑌𝑡+(1−𝛼)𝐹𝑡

Parameter 𝛼 is called the smoothing constant and its value lies between 0 and 1. Since the model uses only one smoothing constant, it is called Single Exponential Smoothing.

Note: Here, there is both trend and seasonality in the data. So, we should have directly gone for the Triple Exponential Smoothing but Simple Exponential Smoothing and the Double Exponential Smoothing models are built over here to get an idea of how the three types of models compare in this case.

SimpleExpSmoothing class must be instantiated and passed the training data.

The fit() function is then called providing the fit configuration, the alpha value, smoothing\_level. If this is omitted or set to None, the model will automatically optimize the value.

Let us check the parameters

{'smoothing\_level': 0.07029120741395753,

'smoothing\_trend': nan,

'smoothing\_seasonal': nan,

'damping\_trend': nan,

'initial\_level': 1764.0137027194548,

'initial\_trend': nan,

'initial\_seasons': array([], dtype=float64),

'use\_boxcox': False,

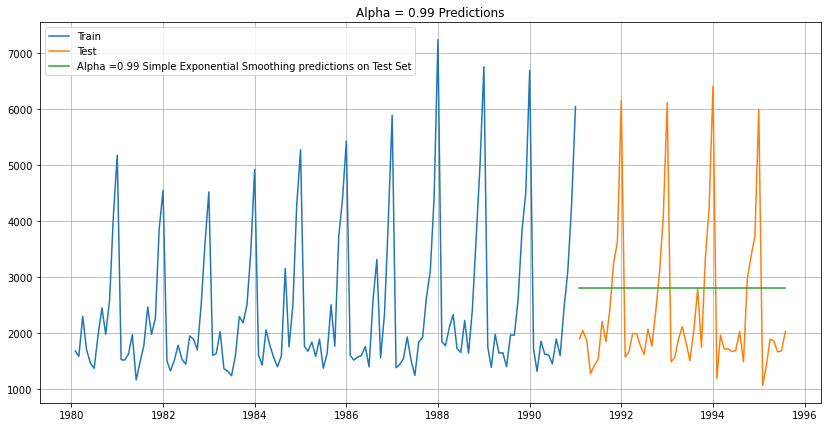
'lamda': None,

'remove\_bias': False}

Here, Python has optimized the smoothing level 0.07029

# Using the fitted model on the training set to forecast on the test set

## Plotting the Training data, Test data and the forecasted values

## Mean Absolute Percentage Error (MAPE) - Function Definition

In [35]:

## Mean Absolute Percentage Error (MAPE) - Function Definition

SES RMSE: 1338.0083842215713

SES RMSE (calculated using statsmodels): 1338.0083842215713

RESULT

Test RMSEAlpha=0.99,SES1338.008384

Holt - ETS(A, A, N) - Holt's linear method with additive errors

Double Exponential Smoothing

# Initializing the Double Exponential Smoothing Model

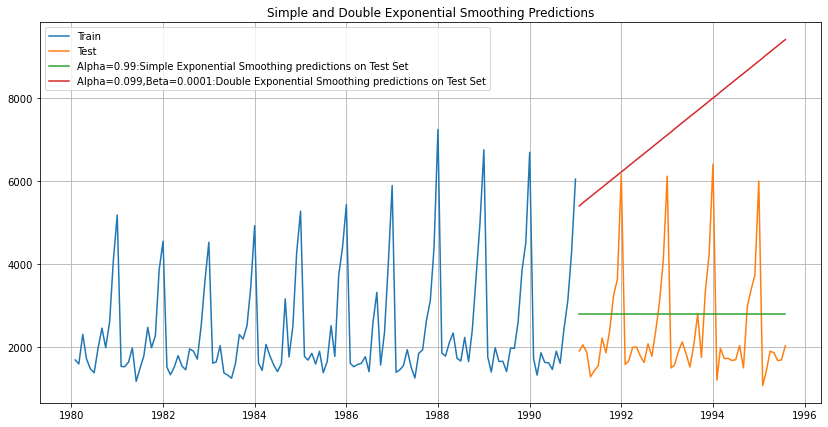
# Fitting the Simple Exponential Smoothing model and asking python to choose the optimal parameters

==Holt model Exponential Smoothing Estimated Parameters ==

{'smoothing\_level': 0.6649999999999999, 'smoothing\_trend': 0.0001, 'smoothing\_seasonal': nan, 'damping\_trend': nan, 'initial\_level': 1502.1999999999998, 'initial\_trend': 74.87272727272739, 'initial\_seasons': array([], dtype=float64), 'use\_boxcox': False, 'lamda': None, 'remove\_bias': False}

# Forecasting using this model for the duration of the test set

## Plotting the Training data, Test data and the forecasted values



In [42]:

MEAN SQUARE ERROR

DES RMSE: 5291.8798332269125

In [43]:

RESULT OF MEAN SQUARE ERROR

|  | **Test RMSE** |
| --- | --- |
| **Alpha=0.99,SES** | 1338.008384 |
| **Alpha=1,Beta=0.0189:DES** | 5291.879833 |

Holt-Winters - ETS(A, A, A) - Holt Winter's linear method with additive errors

# Initializing the Double Exponential Smoothing Model

==Holt Winters model Exponential Smoothing Estimated Parameters ==

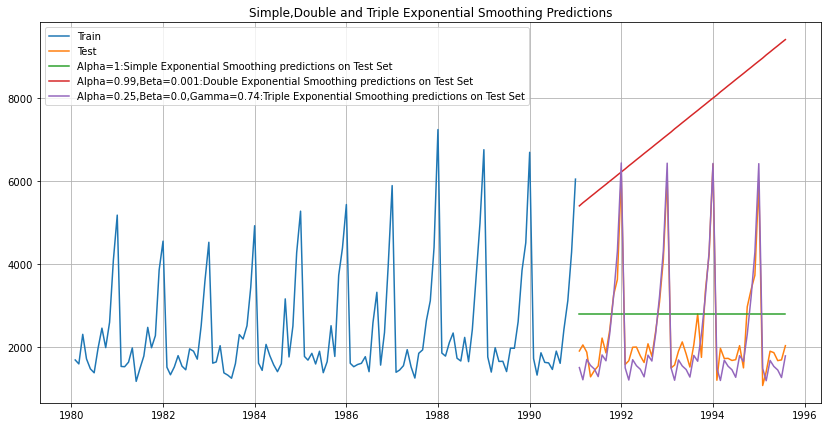
{'smoothing\_level': 0.10445172087746615, 'smoothing\_trend': 0.01093604014460117, 'smoothing\_seasonal': 0.48173366787974736, 'damping\_trend': nan, 'initial\_level': 2363.9138835485714, 'initial\_trend': -0.09301259445429122, 'initial\_seasons': array([-653.56600429, -736.53385593, -368.26577908, -483.5227771 ,

-826.11546154, -832.81081219, -386.32510381, 91.54924895,

-261.07990293, 265.28539181, 1579.92313798, 2619.37089909]), 'use\_boxcox': False, 'lamda': None, 'remove\_bias': False}

In [45]:

## Plotting the Training data, Test data and the forecasted values



In [47]:

MEAN SQUARE ERROR

TES RMSE: 377.73332145026274

In [48]:

RESULT

|  | **Test RMSE** |
| --- | --- |
| **Alpha=0.99,SES** | 1338.008384 |
| **Alpha=1,Beta=0.0189:DES** | 5291.879833 |
| **Alpha=0.25,Beta=0.0,Gamma=0.74:TES** | 377.733321 |

### Holt-Winters - ETS(A, A, M) - Holt Winter's linear method

# Initializing the Double Exponential Smoothing Model

==Holt Winters model Exponential Smoothing Estimated Parameters ==

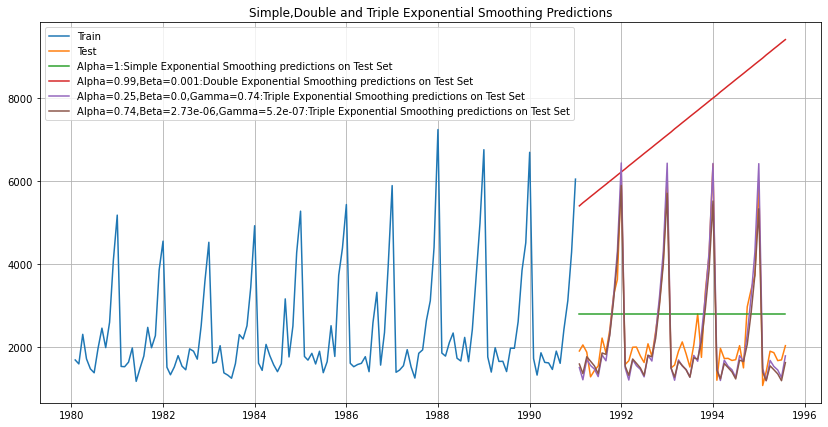
{'smoothing\_level': 0.111049435599969, 'smoothing\_trend': 0.04934441604042903, 'smoothing\_seasonal': 0.36234516019830915, 'damping\_trend': nan, 'initial\_level': 2356.499708505957, 'initial\_trend': -9.728309340605668, 'initial\_seasons': array([0.71397117, 0.68314042, 0.90585281, 0.80629489, 0.65638027,

0.65449343, 0.88653044, 1.13356122, 0.91978839, 1.21195457,

1.87113803, 2.37593129]), 'use\_boxcox': False, 'lamda': None, 'remove\_bias': False}

In [50]:

## Plotting the Training data, Test data and the forecasted values



In [52]:

TES\_am RMSE: 403.11949999600796

In [54]:

RESULT

|  | **Test RMSE** |
| --- | --- |
| **Alpha=0.99,SES** | 1338.008384 |
| **Alpha=1,Beta=0.0189:DES** | 5291.879833 |
| **Alpha=0.25,Beta=0.0,Gamma=0.74:TES** | 377.733321 |
| **Alpha=0.74,Beta=2.73e-06,Gamma=5.2e-07,Gamma=0:TES** | 403.119500 |

In [55]:

## Model 1: Linear Regression

Training Time instance

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132]

Test Time instance

[256, 257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310]

First few rows of Training Data

|  | **Sparkling** | **time** |
| --- | --- | --- |
| **Time\_Stamp** |  |  |
| **1980-01-31** | 1686 | 1 |
| **1980-02-29** | 1591 | 2 |
| **1980-03-31** | 2304 | 3 |
| **1980-04-30** | 1712 | 4 |
| **1980-05-31** | 1471 | 5 |

Last few rows of Training Data

|  | **Sparkling** | **time** |
| --- | --- | --- |
| **Time\_Stamp** |  |  |
| **1990-08-31** | 1605 | 128 |
| **1990-09-30** | 2424 | 129 |
| **1990-10-31** | 3116 | 130 |
| **1990-11-30** | 4286 | 131 |
| **1990-12-31** | 6047 | 132 |

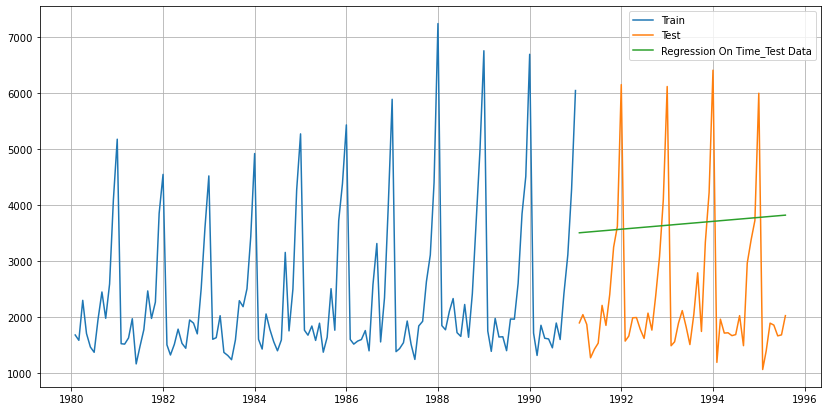
First few rows of Test Data

|  | **Sparkling** | **time** |
| --- | --- | --- |
| **Time\_Stamp** |  |  |
| **1991-01-31** | 1902 | 256 |
| **1991-02-28** | 2049 | 257 |
| **1991-03-31** | 1874 | 258 |
| **1991-04-30** | 1279 | 259 |
| **1991-05-31** | 1432 | 260 |

Last few rows of Test Data

|  | **Sparkling** | **time** |
| --- | --- | --- |
| **Time\_Stamp** |  |  |
| **1995-03-31** | 1897 | 306 |
| **1995-04-30** | 1862 | 307 |
| **1995-05-31** | 1670 | 308 |
| **1995-06-30** | 1688 | 309 |
| **1995-07-31** | 2031 | 310 |

PLOTING THE GRAPH OF LR



Defining the functions for calculating the accuracy metrics.

For RegressionOnTime forecast on the Test Data, RMSE is 1798.201

In [65]:

RESULT

|  | **Test RMSE** |
| --- | --- |
| **RegressionOnTime** | 1798.2007 |

In [66]:

## Model 2: Naive Approach: $\hat{y}\_{t+1} = y\_t$

For this particular naive model, we say that the prediction for tomorrow is the same as today and the prediction for day after tomorrow is tomorrow and since the prediction of tomorrow is same as today,therefore the prediction for day after tomorrow is also today.

TAIL OF THE DATASET

|  | **Sparkling** |
| --- | --- |
| **Time\_Stamp** |  |
| **1990-08-31** | 1605 |
| **1990-09-30** | 2424 |
| **1990-10-31** | 3116 |
| **1990-11-30** | 4286 |
| **1990-12-31** | 6047 |

In [69]:

Time\_Stamp

1991-01-31 6047

1991-02-28 6047

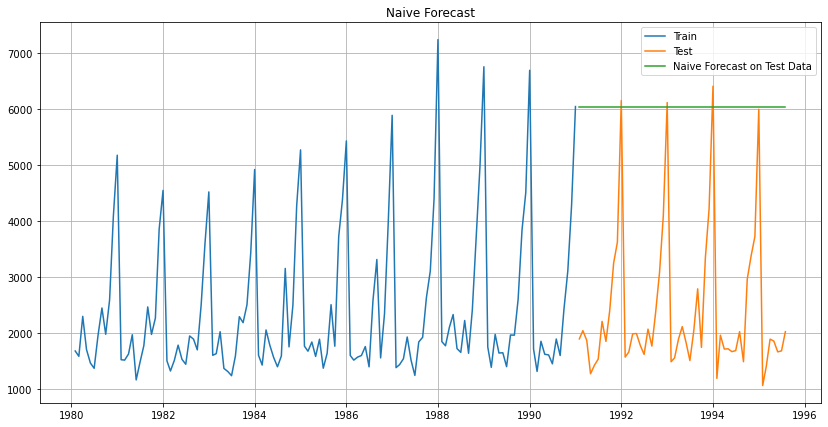
1991-03-31 6047

1991-04-30 6047

1991-05-31 6047

Name: naive, dtype: int64

In [70]:



In [71]:

### Model Evaluation

For RegressionOnTime forecast on the Test Data, RMSE is 3864.279

In [73]:

RESULT

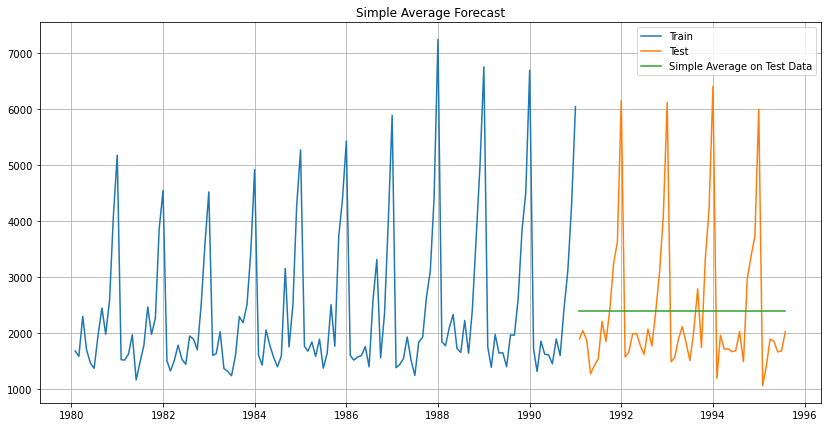
|  | **Test RMSE** |
| --- | --- |
| **RegressionOnTime** | 1798.200700 |
| **NaiveModel** | 3864.279352 |

In [ ]:

## Method 3: Simple Average

|  | **Sparkling** | **mean\_forecast** |
| --- | --- | --- |
| **Time\_Stamp** |  |  |
| **1991-01-31** | 1902 | 2403.780303 |
| **1991-02-28** | 2049 | 2403.780303 |
| **1991-03-31** | 1874 | 2403.780303 |
| **1991-04-30** | 1279 | 2403.780303 |
| **1991-05-31** | 1432 | 2403.780303 |

In [77]:



In [78]:

### Model Evaluation

For Simple Average forecast on the Test Data, RMSE is 1275.082

In [80]:

RESULT

|  | **Test RMSE** |
| --- | --- |
| **RegressionOnTime** | 1798.200700 |
| **NaiveModel** | 3864.279352 |
| **SimpleAverageModel** | 1275.081804 |

In [ ]:

# 5. Check for the stationarity of the data on which the model is being built on using appropriate statistical tests and also mention the hypothesis for the statistical test. If the data is found to be non-stationary, take appropriate steps to make it stationary. Check the new data for stationarity and comment. Note: Stationarity should be checked at alpha = 0.05.

## Check for stationarity of the whole Time Series data.

The Augmented Dickey-Fuller test is an unit root test which determines whether there is a unit root and subsequently whether the series is non-stationary.

The hypothesis in a simple form for the ADF test is:

𝐻0 : The Time Series has a unit root and is thus non-stationary.

𝐻1 : The Time Series does not have a unit root and is thus stationary.

We would want the series to be stationary for building ARIMA models and thus we would want the p-value of this test to be less than the 𝛼 value.

DF test statistic is -1.798

DF test p-value is 0.7055958459932372

Number of lags used 12

We see that at 5% significant level the Time Series is non-stationary.

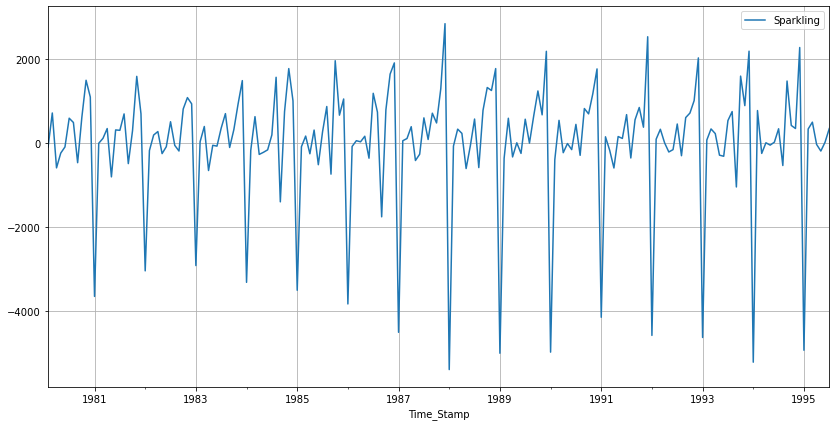
Let us take one level of differencing to see whether the series becomes stationary.

DF test statistic is -44.912

DF test p-value is 0.0

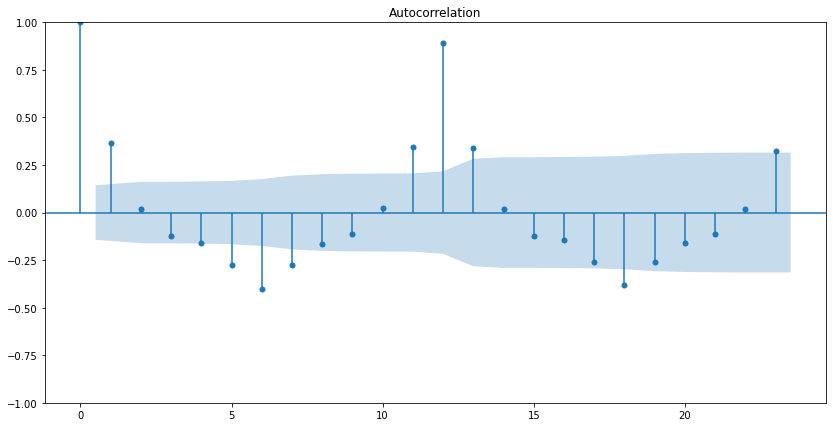
Number of lags used 10

Now, let us go ahead and plot the stationary series.

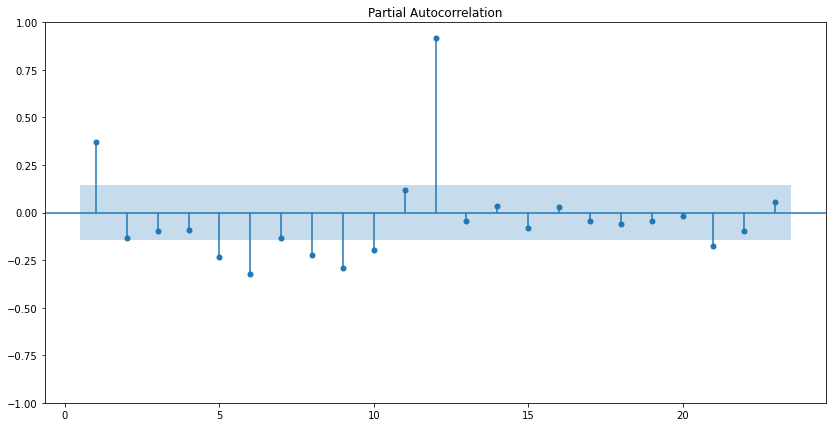


In [86]:

## Plot the Autocorrelation and the Partial Autocorrelation function plots on the whole data.

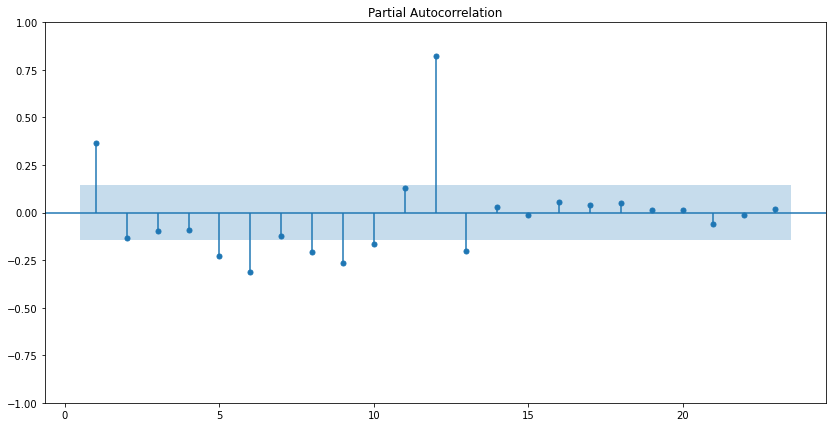


In [89]:



In [90]:

YWMEL



## Split the data into train and test and plot the training and test data.

Training Data is till the end of 2018. Test Data is from the beginning of 2019 to the last time stamp provided.

Int64Index([1980, 1981, 1982, 1983, 1984, 1985, 1986, 1987, 1988, 1989, 1990,

1991, 1992, 1993, 1994, 1995],

dtype='int64', name='Time\_Stamp')

In [92]:

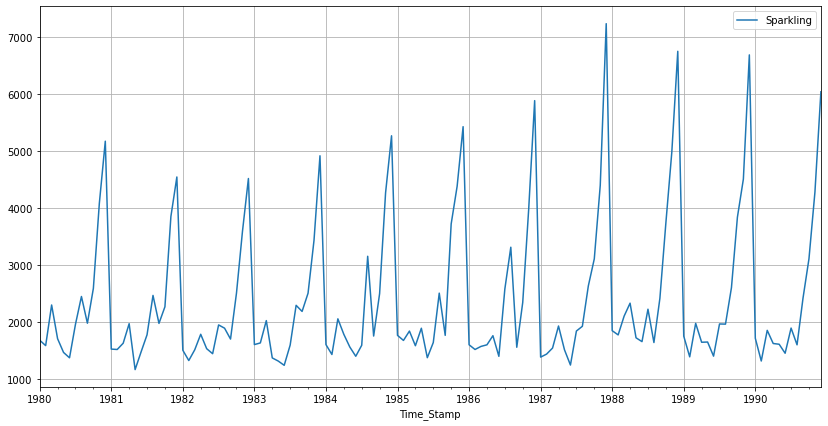
SHAPE OF THE TRAIN AND TEST DATA

(132, 1)

(55, 1)

## Check for stationarity of the Training Data Time Series.

Let us plot the training data



In [98]:

DF test statistic is -2.062

DF test p-value is 0.5674110388593689

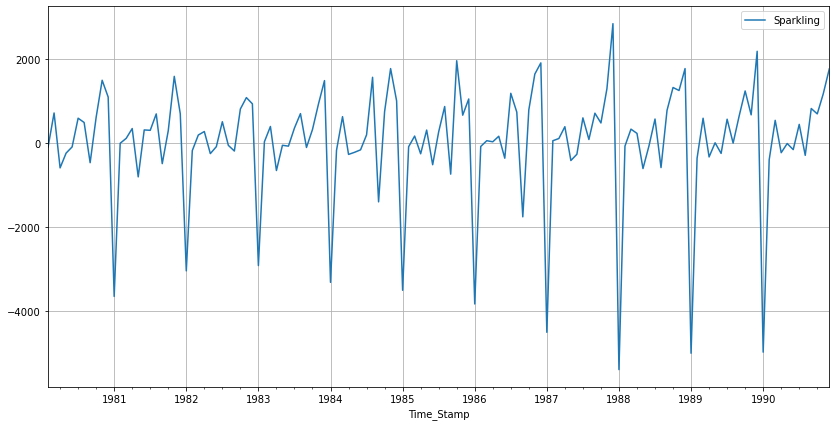
Number of lags used 12

The training data is non-stationary at 95% confidence level. Let us take a first level of differencing to stationarize the Time Series.

DF test statistic is -7.968

DF test p-value is 8.479210655514579e-11

Number of lags used 11



##### Note: If the series is non-stationary, stationarize the Time Series by taking a difference of the Time Series.

Then we can use this particular differenced series to train the ARIMA models. We do not need to worry about stationarity for the Test Data because we are not building any models on the Test Data, we are evaluating our models over there.

We can look at other kinds of transformations as part of making the time series stationary like taking logarithms.

TRAIN INFO

<class 'pandas.core.frame.DataFrame'>

DatetimeIndex: 132 entries, 1980-01-31 to 1990-12-31

Data columns (total 1 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Sparkling 132 non-null int64

dtypes: int64(1)

memory usage: 2.1 KB

# 6. Build an automated version of the ARIMA/SARIMA model in which the parameters are selected using the lowest Akaike Information Criteria (AIC) on the training data and evaluate this model on the test data using RMSE.

for demonstration purposes we are building an ARIMA model both by looking at the minimum AIC criterion and by looking at the ACF and the PACF plots.

## The following loop helps us in getting a combination of different parameters of p and q in the range of 0 and 2

## We have kept the value of d as 1 as we need to take a difference of the series to make it stationary.

Examples of the parameter combinations for the Model

Model: (0, 1, 0)

Model: (0, 1, 1)

Model: (0, 1, 2)

Model: (0, 1, 3)

Model: (1, 1, 0)

Model: (1, 1, 1)

Model: (1, 1, 2)

Model: (1, 1, 3)

Model: (2, 1, 0)

Model: (2, 1, 1)

Model: (2, 1, 2)

Model: (2, 1, 3)

Model: (3, 1, 0)

Model: (3, 1, 1)

Model: (3, 1, 2)

Model: (3, 1, 3)

#appending the AIC values and the model parameters to the previously created data frame

#for easier understanding and sorting of the AIC values

## Sort the above AIC values in the ascending order to get the parameters for the minimum AIC value

|  | **param** | **AIC** |
| --- | --- | --- |
| **10** | (2, 1, 2) | 2213.509212 |
| **15** | (3, 1, 3) | 2221.455189 |
| **14** | (3, 1, 2) | 2230.77343 |
| **11** | (2, 1, 3) | 2232.880487 |
| **9** | (2, 1, 1) | 2233.777626 |

In [106]:

ARIMA MATRIX

SARIMAX Results

==============================================================================

Dep. Variable: Sparkling No. Observations: 132

Model: ARIMA(2, 1, 2) Log Likelihood -1101.755

Date: Sat, 26 Nov 2022 AIC 2213.509

Time: 17:09:30 BIC 2227.885

Sample: 01-31-1980 HQIC 2219.351

- 12-31-1990

Covariance Type: opg

==============================================================================

coef std err z P>|z| [0.025 0.975]

------------------------------------------------------------------------------

ar.L1 1.3121 0.046 28.781 0.000 1.223 1.401

ar.L2 -0.5593 0.072 -7.741 0.000 -0.701 -0.418

ma.L1 -1.9917 0.109 -18.217 0.000 -2.206 -1.777

ma.L2 0.9999 0.110 9.109 0.000 0.785 1.215

sigma2 1.099e+06 1.99e-07 5.51e+12 0.000 1.1e+06 1.1e+06

===================================================================================

Ljung-Box (L1) (Q): 0.19 Jarque-Bera (JB): 14.46

Prob(Q): 0.67 Prob(JB): 0.00

Heteroskedasticity (H): 2.43 Skew: 0.61

Prob(H) (two-sided): 0.00 Kurtosis: 4.08

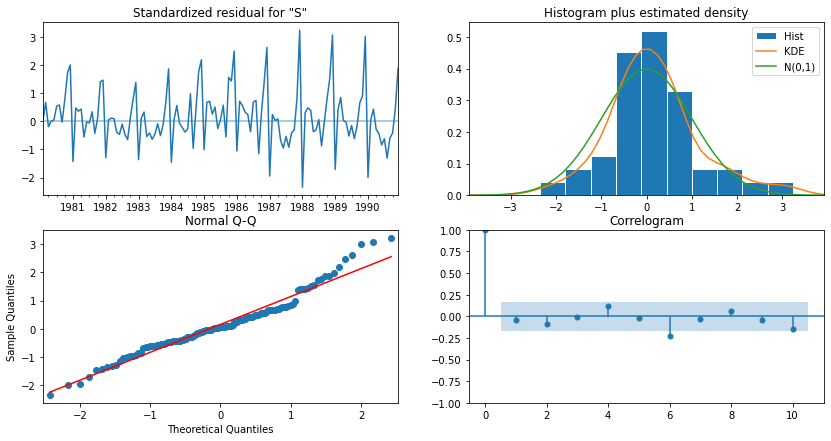
===================================================================================

Warnings:

[1] Covariance matrix calculated using the outer product of gradients (complex-step).

[2] Covariance matrix is singular or near-singular, with condition number 7.18e+27. Standard errors may be unstable.

Diagnostics plot.



In [108]:

## Predict on the Test Set using this model and evaluate the model.

## Mean Absolute Percentage Error (MAPE) - Function Definition

## Importing the mean\_squared\_error function from sklearn to calculate the RMSE

RMSE: 1299.979773687031

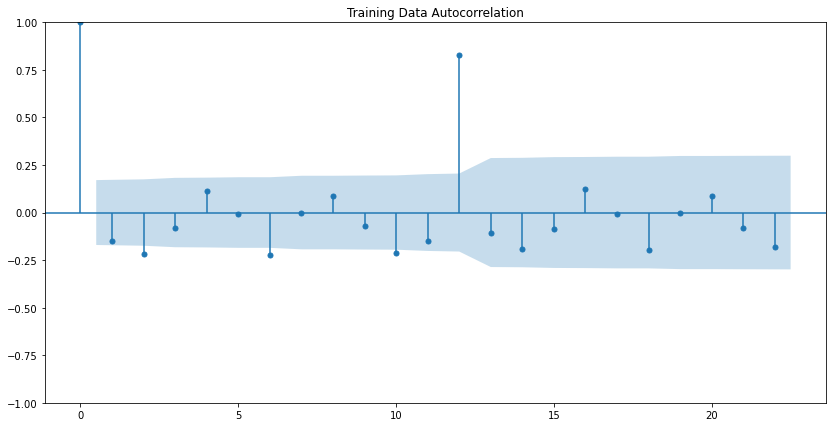
MAPE: 47.099973377448464

RESULT

|  | **RMSE** | **MAPE** |
| --- | --- | --- |
| **ARIMA(2,1,2)** | 1299.979774 | 47.099973 |

In [113]:

SARIMA



FOR EXAMPLE MODEL

Examples of the parameter combinations for the Model are

Model: (0, 1, 1)(0, 0, 1, 5)

Model: (0, 1, 2)(0, 0, 2, 5)

Model: (1, 1, 0)(1, 0, 0, 5)

Model: (1, 1, 1)(1, 0, 1, 5)

Model: (1, 1, 2)(1, 0, 2, 5)

Model: (2, 1, 0)(2, 0, 0, 5)

Model: (2, 1, 1)(2, 0, 1, 5)

Model: (2, 1, 2)(2, 0, 2, 5)

|  | **param** | **seasonal** | **AIC** |
| --- | --- | --- | --- |
| **71** | (2, 1, 1) | (2, 0, 2, 5) | 2002.228392 |
| **74** | (2, 1, 2) | (0, 0, 2, 5) | 2011.391206 |
| **77** | (2, 1, 2) | (1, 0, 2, 5) | 2013.068035 |
| **80** | (2, 1, 2) | (2, 0, 2, 5) | 2013.727537 |
| **20** | (0, 1, 2) | (0, 0, 2, 5) | 2015.836489 |

In [133]:

SAIMA METRIX

SARIMAX Results

==============================================================================================

Dep. Variable: Sparkling No. Observations: 132

Model: SARIMAX(2, 1, 1)x(2, 0, [1, 2], 5) Log Likelihood -993.114

Date: Sat, 26 Nov 2022 AIC 2002.228

Time: 17:12:36 BIC 2024.461

Sample: 01-31-1980 HQIC 2011.257

- 12-31-1990

Covariance Type: opg

==============================================================================

coef std err z P>|z| [0.025 0.975]

------------------------------------------------------------------------------

ar.L1 0.0495 0.108 0.459 0.646 -0.162 0.261

ar.L2 -0.4028 0.137 -2.938 0.003 -0.672 -0.134

ma.L1 -1.0447 0.046 -22.792 0.000 -1.135 -0.955

ar.S.L5 -1.7495 0.031 -56.206 0.000 -1.810 -1.688

ar.S.L10 -1.0390 0.028 -36.965 0.000 -1.094 -0.984

ma.S.L5 1.7155 0.149 11.498 0.000 1.423 2.008

ma.S.L10 0.9995 0.163 6.137 0.000 0.680 1.319

sigma2 7.987e+05 3.28e-07 2.43e+12 0.000 7.99e+05 7.99e+05

===================================================================================

Ljung-Box (L1) (Q): 2.47 Jarque-Bera (JB): 38.62

Prob(Q): 0.12 Prob(JB): 0.00

Heteroskedasticity (H): 2.48 Skew: 0.94

Prob(H) (two-sided): 0.01 Kurtosis: 5.07

===================================================================================

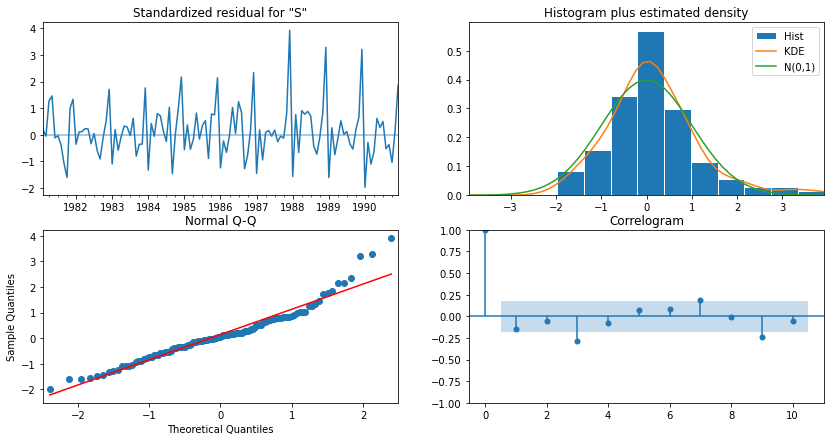
Warnings:

[1] Covariance matrix calculated using the outer product of gradients (complex-step).

[2] Covariance matrix is singular or near-singular, with condition number 1.01e+28. Standard errors may be unstable.

In [134]:

Diagnostics plot.



In [135]:

## Predict on the Test Set using this model and evaluate the model.

| **Sparkling** | **mean** | **mean\_se** | **mean\_ci\_lower** | **mean\_ci\_upper** |
| --- | --- | --- | --- | --- |
| **1991-01-31** | 3767.111822 | 968.955423 | 1867.994089 | 5666.229554 |
| **1991-02-28** | 2178.329021 | 973.157180 | 270.975997 | 4085.682044 |
| **1991-03-31** | 1881.587318 | 1032.297835 | -141.679261 | 3904.853897 |
| **1991-04-30** | 1712.951535 | 1032.318837 | -310.356206 | 3736.259275 |
| **1991-05-31** | 884.749947 | 1045.231973 | -1163.867075 | 2933.366970 |

In [138]:

RMSE: 1114.5969412125685

MAPE: 46.256971159192965

RESULT

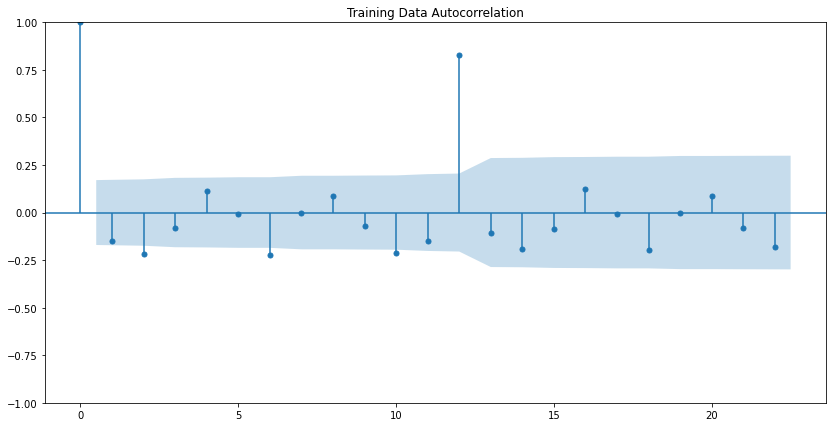
|  | **RMSE** | **MAPE** |
| --- | --- | --- |
| **ARIMA(2,1,2)** | 1299.979774 | 47.099973 |
| **SARIMA(1,1,3)(3,0,3,6)** | 872.912606 | 38.837631 |
| **ARIMA(4,1,4)** | 1218.084201 | 40.481177 |
| **SARIMA(2,1,1)(2,0,2,5)** | 1114.596941 | 46.256971 |

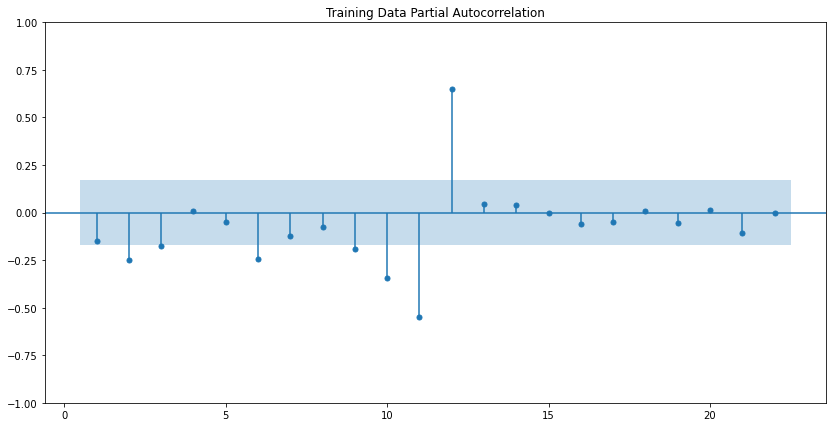
In [ ]:

# 7. Build ARIMA/SARIMA models based on the cut-off points of ACF and PACF on the training data and evaluate this model on the test data using RMSE.

ARIMA

Let us look at the ACF and the PACF plots





Here, we have taken alpha=0.05.

The Auto-Regressive parameter in an ARIMA model is 'p' which comes from the significant lag before which the PACF plot cuts-off to 4.

The Moving-Average parameter in an ARIMA model is 'q' which comes from the significant lag before the ACF plot cuts-off to 4.

By looking at the above plots, we will take the value of p and q to be 4 and 1 respectively.

SARIMAX Results

==============================================================================

Dep. Variable: Sparkling No. Observations: 132

Model: ARIMA(4, 1, 4) Log Likelihood -1097.900

Date: Sat, 26 Nov 2022 AIC 2213.800

Time: 17:24:07 BIC 2239.677

Sample: 01-31-1980 HQIC 2224.315

- 12-31-1990

Covariance Type: opg

==============================================================================

coef std err z P>|z| [0.025 0.975]

------------------------------------------------------------------------------

ar.L1 -0.4335 0.138 -3.141 0.002 -0.704 -0.163

ar.L2 -0.4357 0.066 -6.623 0.000 -0.565 -0.307

ar.L3 -0.4356 0.121 -3.589 0.000 -0.673 -0.198

ar.L4 0.5636 0.061 9.191 0.000 0.443 0.684

ma.L1 -0.0016 15.985 -0.000 1.000 -31.332 31.329

ma.L2 0.0068 31.889 0.000 1.000 -62.494 62.508

ma.L3 -0.0122 15.792 -0.001 0.999 -30.964 30.939

ma.L4 -0.9930 0.174 -5.700 0.000 -1.334 -0.652

sigma2 9.084e+05 3.38e-07 2.69e+12 0.000 9.08e+05 9.08e+05

===================================================================================

Ljung-Box (L1) (Q): 0.03 Jarque-Bera (JB): 0.87

Prob(Q): 0.87 Prob(JB): 0.65

Heteroskedasticity (H): 2.83 Skew: 0.17

Prob(H) (two-sided): 0.00 Kurtosis: 3.20

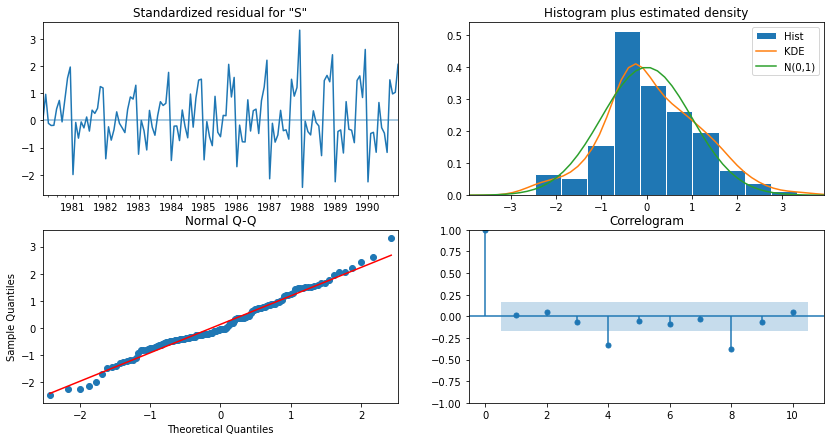
===================================================================================

Warnings:

[1] Covariance matrix calculated using the outer product of gradients (complex-step).

[2] Covariance matrix is singular or near-singular, with condition number 2.34e+29. Standard errors may be unstable.

Let us analyse the residuals from the various diagnostics plot.



In [143]:

## Predict on the Test Set using this model and evaluate the model.

RMSE: 1218.0842005744419

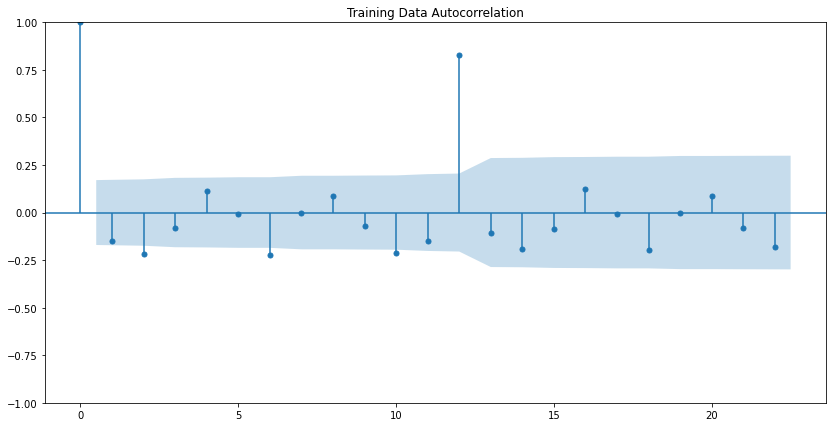
MAPE: 40.48117728094749

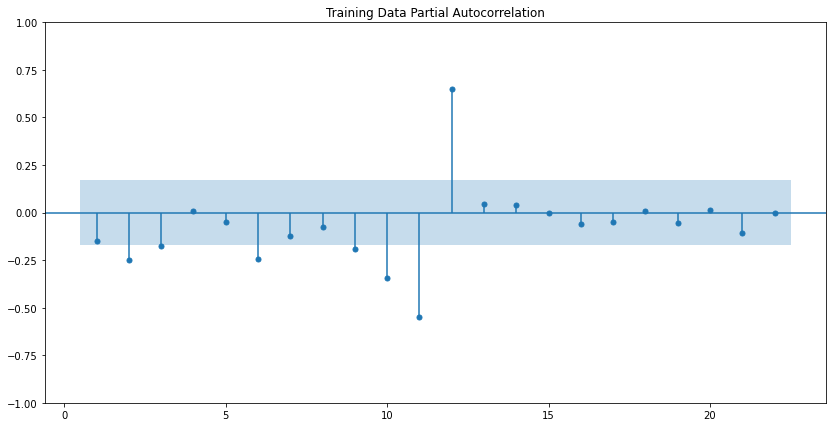
RESULT

|  | **RMSE** | **MAPE** |
| --- | --- | --- |
| **ARIMA(2,1,2)** | 1299.979774 | 47.099973 |
| **SARIMA(1,1,3)(3,0,3,6)** | 872.912606 | 38.837631 |
| **ARIMA(4,1,4)** | 1218.084201 | 40.481177 |
| **SARIMA(2,1,1)(2,0,2,5)** | 1114.596941 | 46.256971 |
| **ARIMA(4,1,4)** | 1218.084201 | 40.481177 |

In [147]:

#SARIMA





Here, we have taken alpha=0.05.

We are going to take the seasonal period as 3 or its multiple e.g. 6. We are taking the p value to be 3 and the q value also to be 3 as the parameters same as the ARIMA model.

The Auto-Regressive parameter in an SARIMA model is 'P' which comes from the significant lag after which the PACF plot cuts-off to 0.

The Moving-Average parameter in an SARIMA model is 'Q' which comes from the significant lag after which the ACF plot cuts-off to 3.

SARIMAX Results

=========================================================================================

Dep. Variable: Sparkling No. Observations: 132

Model: SARIMAX(3, 1, 3)x(0, 0, 3, 6) Log Likelihood -875.634

Date: Sat, 26 Nov 2022 AIC 1771.267

Time: 17:25:06 BIC 1798.181

Sample: 01-31-1980 HQIC 1782.182

- 12-31-1990

Covariance Type: opg

==============================================================================

coef std err z P>|z| [0.025 0.975]

------------------------------------------------------------------------------

ar.L1 -1.1904 0.170 -6.998 0.000 -1.524 -0.857

ar.L2 -0.2044 0.137 -1.489 0.136 -0.473 0.065

ar.L3 -0.0158 0.165 -0.096 0.923 -0.339 0.307

ma.L1 0.8224 0.207 3.968 0.000 0.416 1.229

ma.L2 -0.9771 0.098 -10.008 0.000 -1.168 -0.786

ma.L3 -0.7990 0.162 -4.928 0.000 -1.117 -0.481

ma.S.L6 -0.3335 0.157 -2.131 0.033 -0.640 -0.027

ma.S.L12 0.8407 0.141 5.962 0.000 0.564 1.117

ma.S.L18 -0.0408 0.187 -0.218 0.828 -0.408 0.326

sigma2 4.774e+05 7e-07 6.82e+11 0.000 4.77e+05 4.77e+05

===================================================================================

Ljung-Box (L1) (Q): 0.23 Jarque-Bera (JB): 1.26

Prob(Q): 0.63 Prob(JB): 0.53

Heteroskedasticity (H): 2.20 Skew: 0.24

Prob(H) (two-sided): 0.02 Kurtosis: 2.78

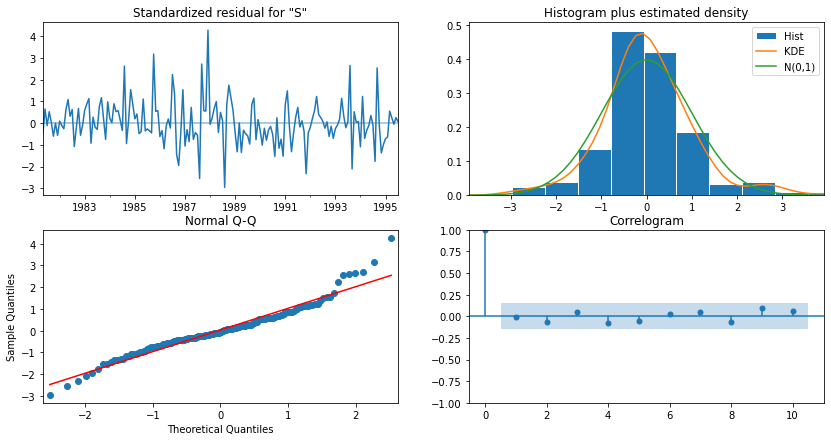
===================================================================================

Warnings:

[1] Covariance matrix calculated using the outer product of gradients (complex-step).

[2] Covariance matrix is singular or near-singular, with condition number 8.8e+26. Standard errors may be unstable.

Let us analyse the residuals from the various diagnostics plot.



In [ ]:

# 9. Based on the model-building exercise, build the most optimum model(s) on the complete data and predict 12 months into the future with appropriate confidence intervals/bands.

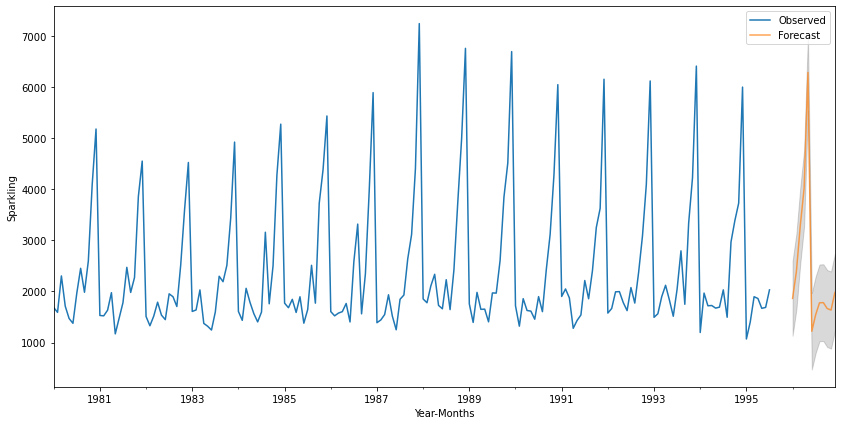
| **Sparkling** | **mean** | **mean\_se** | **mean\_ci\_lower** | **mean\_ci\_upper** |
| --- | --- | --- | --- | --- |
| **1995-08-31** | 1864.148257 | 372.572662 | 1133.919257 | 2594.377257 |
| **1995-09-30** | 2393.363665 | 378.431940 | 1651.650692 | 3135.076639 |
| **1995-10-31** | 3285.328488 | 379.282331 | 2541.948780 | 4028.708197 |
| **1995-11-30** | 4017.494940 | 380.130824 | 3272.452215 | 4762.537665 |
| **1995-12-31** | 6286.102218 | 380.977434 | 5539.400169 | 7032.804268 |

In [163]:

RMSE of the Full Model 531.977225771907

In [171]:

# plot the forecast along with the confidence band



In [173]:

RESULT

|  | **RMSE** | **MAPE** |
| --- | --- | --- |
| **ARIMA(2,1,2)** | 1299.979774 | 47.099973 |
| **SARIMA(1,1,3)(3,0,3,6)** | 872.912606 | 38.837631 |
| **ARIMA(4,1,4)** | 1218.084201 | 40.481177 |
| **SARIMA(2,1,1)(2,0,2,5)** | 1114.596941 | 46.256971 |
| **ARIMA(4,1,4)** | 1218.084201 | 40.481177 |
| **SARIMA(3,1,3)(0,0,3,6)** | 1175.140908 | 38.313509 |

\*\*\*

**Problem 2: ROSE**

# 1. Read the data as an appropriate Time Series data and plot the data.

HEAD OF THE DATA

|  | **YearMonth** | **Rose** |
| --- | --- | --- |
| **0** | 1980-01 | 112.0 |
| **1** | 1980-02 | 118.0 |
| **2** | 1980-03 | 129.0 |
| **3** | 1980-04 | 99.0 |
| **4** | 1980-05 | 116.0 |

In [29]:

WE FOUND TWO MISSING VALUES IN THE DATASET

YearMonth 0

Rose 2

dtype: int64

CONVERTED INTO NaN

In [ ]:

|  | **Rose** |
| --- | --- |
| **Time\_Stamp** |  |
| **1994-02-28** | 35.0 |
| **1994-03-31** | 42.0 |
| **1994-04-30** | 48.0 |
| **1994-05-31** | 44.0 |
| **1994-06-30** | 45.0 |
| **1994-07-31** | NaN |
| **1994-08-31** | NaN |
| **1994-09-30** | 46.0 |
| **1994-10-31** | 51.0 |
| **1994-11-30** | 63.0 |
| **1994-12-31** | 84.0 |

In [42]:

PLOTED THE VALUE

|  | **Rose** |
| --- | --- |
| **Time\_Stamp** |  |
| **1994-02-28** | 35.000000 |
| **1994-03-31** | 42.000000 |
| **1994-04-30** | 48.000000 |
| **1994-05-31** | 44.000000 |
| **1994-06-30** | 45.000000 |
| **1994-07-31** | 45.333333 |
| **1994-08-31** | 45.666667 |
| **1994-09-30** | 46.000000 |
| **1994-10-31** | 51.000000 |
| **1994-11-30** | 63.000000 |
| **1994-12-31** | 84.000000 |

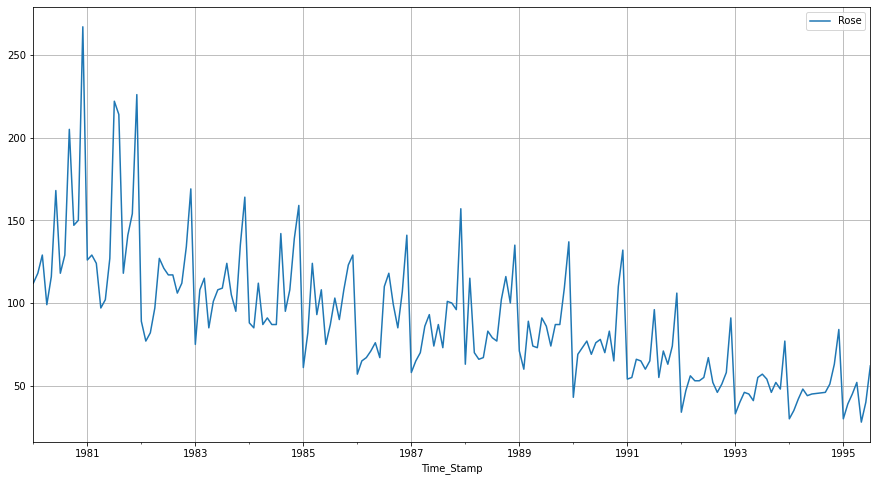
In [43]:

Rose 0

dtype: int64

In [44]:

PLOTTING THE GRAPH OF THE DATA



In [ ]:

# 2. Perform appropriate Exploratory Data Analysis to understand the data and also perform decomposition.

DESCRIBE THE DATA

|  | **Rose** |
| --- | --- |
| **count** | 187.000000 |
| **mean** | 89.914439 |
| **std** | 39.238325 |
| **min** | 28.000000 |
| **25%** | 62.500000 |
| **50%** | 85.000000 |
| **75%** | 111.000000 |
| **max** | 267.000000 |

In [ ]:

LETS CHECK THE INFORMATION OF THE DATA

<class 'pandas.core.frame.DataFrame'>

DatetimeIndex: 187 entries, 1980-01-31 to 1995-07-31

Data columns (total 1 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Rose 187 non-null float64

dtypes: float64(1)

memory usage: 2.9 KB

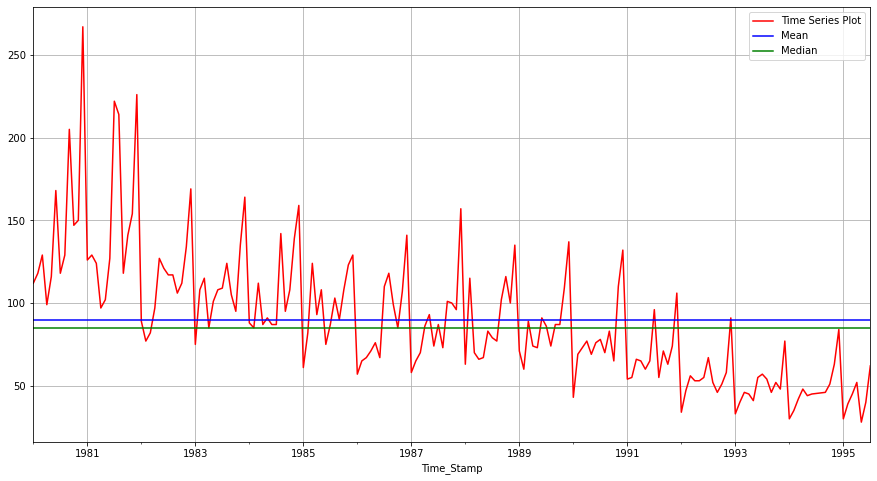
In [49]:

SHAPE

(187, 1)

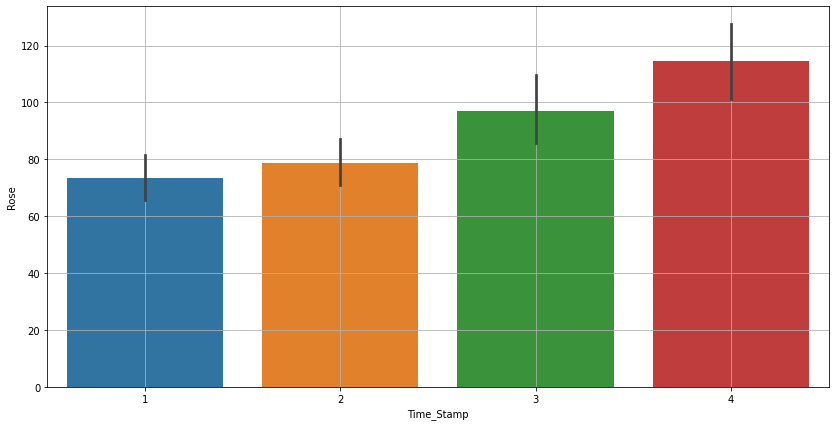
In [50]:

PLOTTING THE GRAPH WITH THE MEAN AND MEDIAN



In [117]:

BARPLOT OF QUARTERLY BASED



In [118]:

INDEX OF DATA

Out[118]:

<bound method \_inherit\_from\_data.<locals>.method of DatetimeIndex(['1980-01-31', '1980-02-29', '1980-03-31', '1980-04-30',

'1980-05-31', '1980-06-30', '1980-07-31', '1980-08-31',

'1980-09-30', '1980-10-31',

...

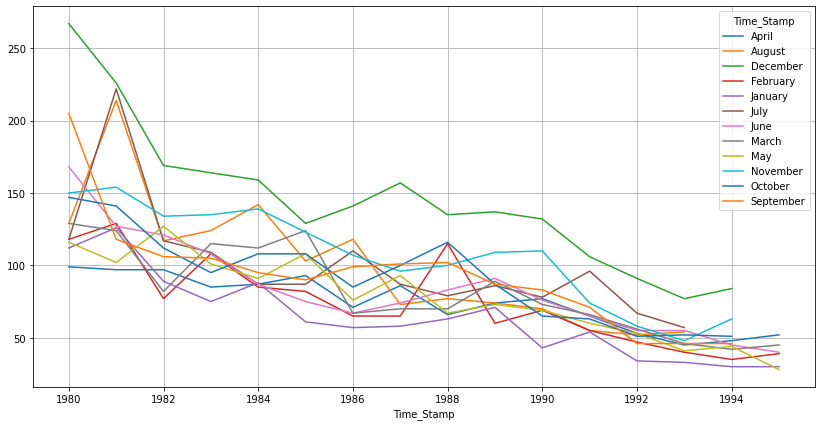
'1994-10-31', '1994-11-30', '1994-12-31', '1995-01-31',

'1995-02-28', '1995-03-31', '1995-04-30', '1995-05-31',

'1995-06-30', '1995-07-31'],

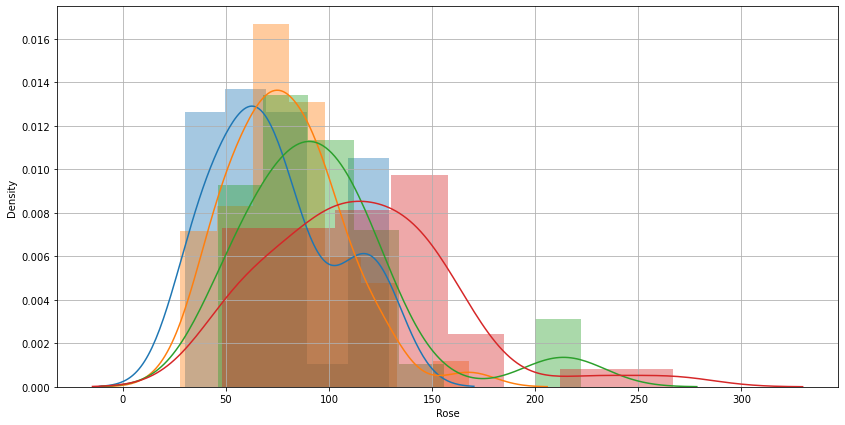
dtype='datetime64[ns]', name='Time\_Stamp', length=187, freq=None)>

QUARTERLY DATA



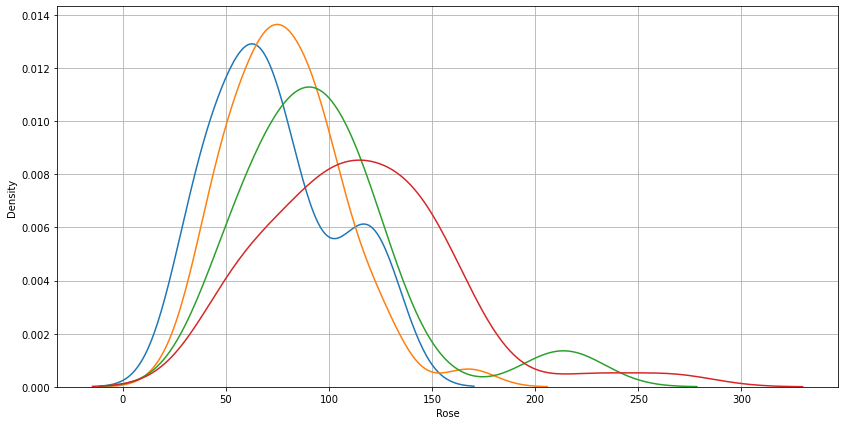
In [121]:

DISTPLOT



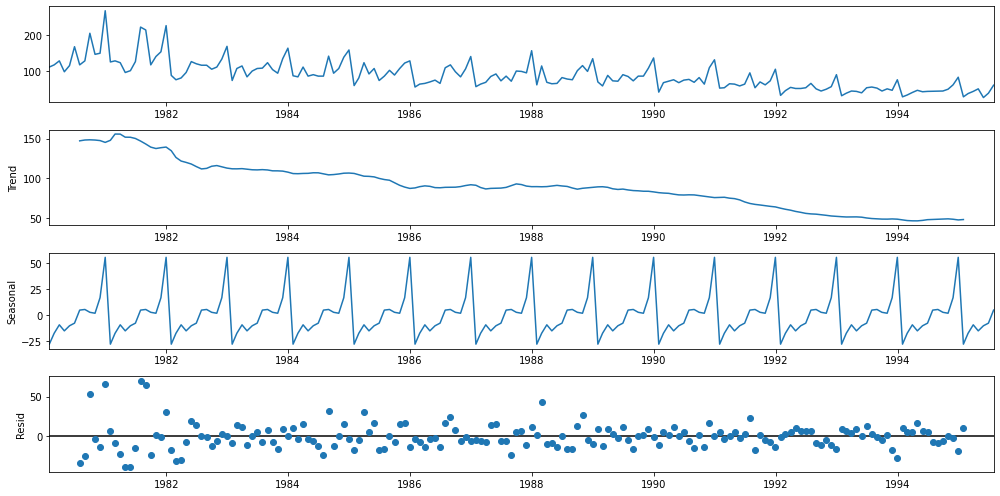
In [122]:

In [121]:

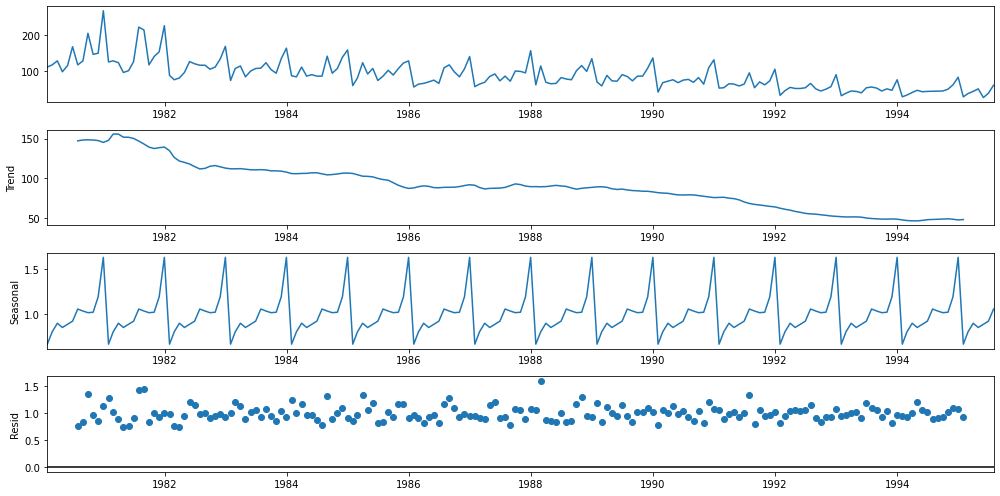


In [18]:

decomposition additive



decomposition MULTIPLCATIVE



# 3. Split the data into training and test. The test data should start in 1991.

SHAPE OF THE TRAIN AND TEST DATA

(132, 1)

(55, 1)

In [54]:

# 4. Build all the exponential smoothing models on the training data and evaluate the model using RMSE on the test data. Other models such as regression,naïve forecast models and simple average models. should also be built on the training data and check the performance on the test data using RMSE.

SES - ETS(A, N, N) - Simple Exponential Smoothing with additive errors

The simplest of the exponentially smoothing methods is naturally called simple exponential smoothing (SES).

This method is suitable for forecasting data with no clear trend or seasonal pattern.

In Single ES, the forecast at time (t + 1) is given by Winters,1960

𝐹𝑡+1=𝛼𝑌𝑡+(1−𝛼)𝐹𝑡

Parameter 𝛼 is called the smoothing constant and its value lies between 0 and 1. Since the model uses only one smoothing constant, it is called Single Exponential Smoothing.

Note: Here, there is both trend and seasonality in the data. So, we should have directly gone for the Triple Exponential Smoothing but Simple Exponential Smoothing and the Double Exponential Smoothing models are built over here to get an idea of how the three types of models compare in this case.

SimpleExpSmoothing class must be instantiated and passed the training data.

The fit() function is then called providing the fit configuration, the alpha value, smoothing\_level. If this is omitted or set to None, the model will automatically optimize the value.

# Fitting the Simple Exponential Smoothing model and asking python to choose the optimal parameters

## Let us check the parameters

{'smoothing\_level': 0.09874982660106467,

'smoothing\_trend': nan,

'smoothing\_seasonal': nan,

'damping\_trend': nan,

'initial\_level': 134.38704806164168,

'initial\_trend': nan,

'initial\_seasons': array([], dtype=float64),

'use\_boxcox': False,

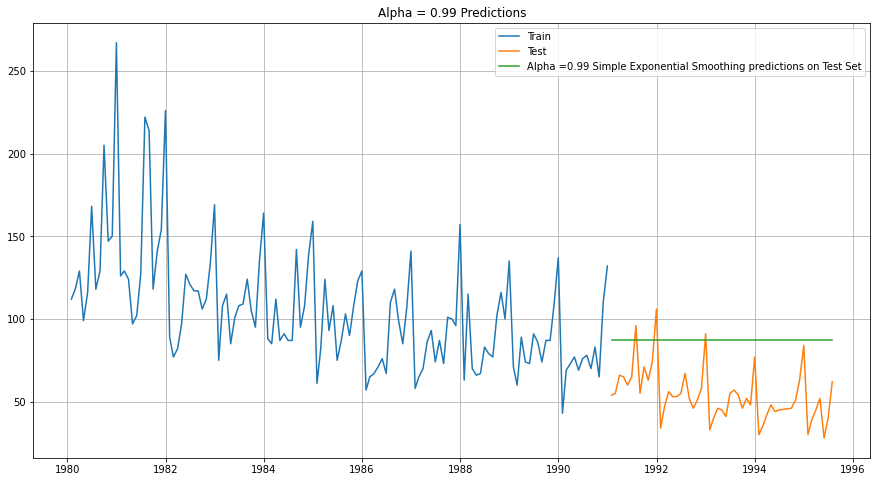
'lamda': None,

'remove\_bias': False}

Here, Python has optimized the smoothing level 0.09874

# Using the fitted model on the training set to forecast on the test set

## Plotting the Training data, Test data and the forecasted values



In [61]:

## Mean Absolute Percentage Error (MAPE) - Function Definition

SES RMSE: 36.79624028783191

SES RMSE (calculated using statsmodels): 36.7962402878319

RESULT

|  | **Test RMSE** |
| --- | --- |
| **Alpha=0.99,SES** | 36.79624 |

Holt - ETS(A, A, N) - Holt's linear method with additive errors

Double Exponential Smoothing

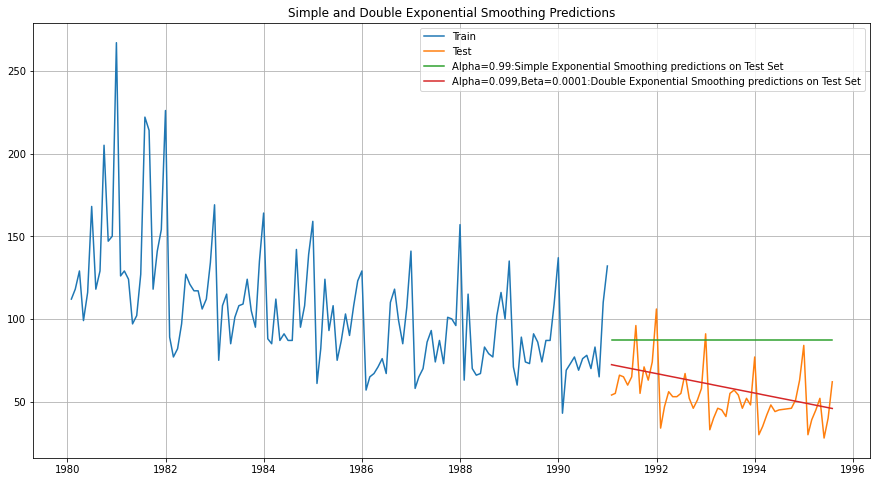
# Initializing the Double Exponential Smoothing Model

# Fitting the Simple Exponential Smoothing model and asking python to choose the optimal parameters

==Holt model Exponential Smoothing Estimated Parameters ==

{'smoothing\_level': 0.005627395225399259, 'smoothing\_trend': 6.722744943479034e-05, 'smoothing\_seasonal': nan, 'damping\_trend': nan, 'initial\_level': 137.53777460719618, 'initial\_trend': -0.4905093549686182, 'initial\_seasons': array([], dtype=float64), 'use\_boxcox': False, 'lamda': None, 'remove\_bias': False}

## Plotting the Training data, Test data and the forecasted values



In [69]:

DES RMSE: 15.394980179637058

In [70]:

RESULT

|  | **Test RMSE** |
| --- | --- |
| **Alpha=0.99,SES** | 36.79624 |
| **Alpha=1,Beta=0.0189:DES** | 15.39498 |

Holt-Winters - ETS(A, A, A) - Holt Winter's linear method with additive errors

# Initializing the Double Exponential Smoothing Model

# Fitting the Simple Exponential Smoothing model and asking python to choose the optimal parameters

==Holt Winters model Exponential Smoothing Estimated Parameters ==

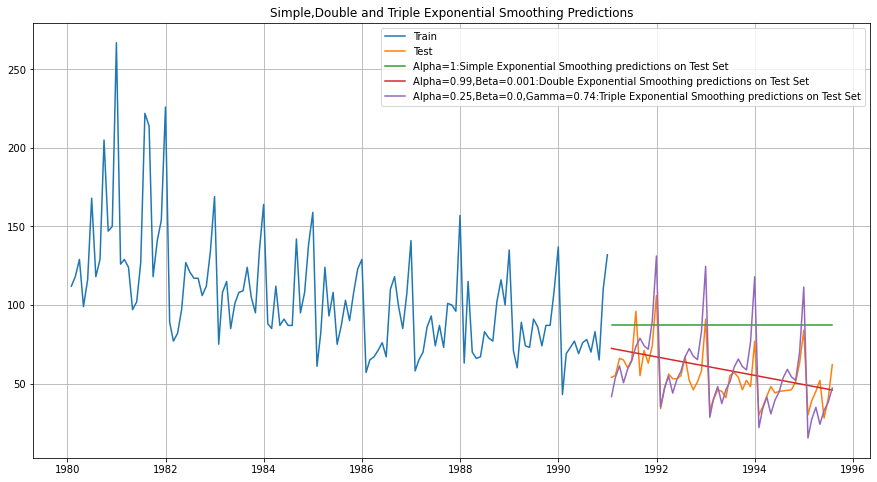
{'smoothing\_level': 0.09247995167383646, 'smoothing\_trend': 0.0, 'smoothing\_seasonal': 0.0, 'damping\_trend': nan, 'initial\_level': 147.1251209143706, 'initial\_trend': -0.5503329642408892, 'initial\_seasons': array([-31.99085126, -19.51359677, -11.31406667, -21.57574971,

-12.21264592, -6.5948571 , 3.15000859, 8.96777153,

4.77567092, 3.09390942, 21.52051931, 63.56076615]), 'use\_boxcox': False, 'lamda': None, 'remove\_bias': False}

In [72]:

## Plotting the Training data, Test data and the forecasted values



In [74]:

TES RMSE: 14.380700451545025

In [75]:

RESULT

|  | **Test RMSE** |
| --- | --- |
| **Alpha=0.99,SES** | 36.79624 |
| **Alpha=1,Beta=0.0189:DES** | 15.39498 |
| **Alpha=0.25,Beta=0.0,Gamma=0.74:TES** | 14.38070 |

### Holt-Winters - ETS(A, A, M) - Holt Winter's linear method

# Initializing the Double Exponential Smoothing Model

# Fitting the Simple Exponential Smoothing model and asking python to choose the optimal parameters

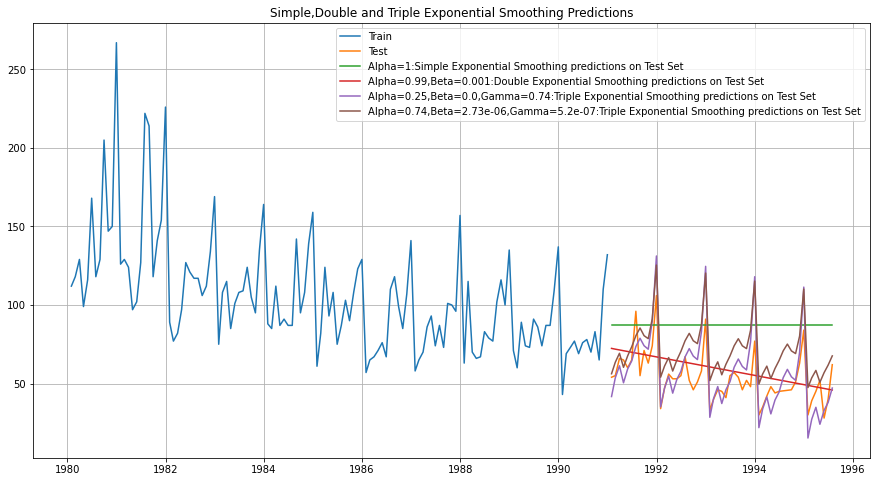
==Holt Winters model Exponential Smoothing Estimated Parameters ==

{'smoothing\_level': 0.07336915185646381, 'smoothing\_trend': 0.04366616789722176, 'smoothing\_seasonal': 8.092476708185191e-05, 'damping\_trend': nan, 'initial\_level': 142.12276158556045, 'initial\_trend': -0.8409491641808673, 'initial\_seasons': array([0.79010014, 0.89594527, 0.9795284 , 0.85607407, 0.9627955 ,

1.04820922, 1.15284393, 1.22759891, 1.16104173, 1.13748454,

1.32592572, 1.82801267]), 'use\_boxcox': False, 'lamda': None, 'remove\_bias': False}

## Plotting the Training data, Test data and the forecasted values



In [79]:

TES\_am RMSE: 19.867427460977897

In [81]:

RESULT

|  | **Test RMSE** |
| --- | --- |
| **Alpha=0.99,SES** | 36.796240 |
| **Alpha=1,Beta=0.0189:DES** | 15.394980 |
| **Alpha=0.25,Beta=0.0,Gamma=0.74:TES** | 14.380700 |
| **Alpha=0.74,Beta=2.73e-06,Gamma=5.2e-07,Gamma=0:TES** | 19.867427 |

In [82]:

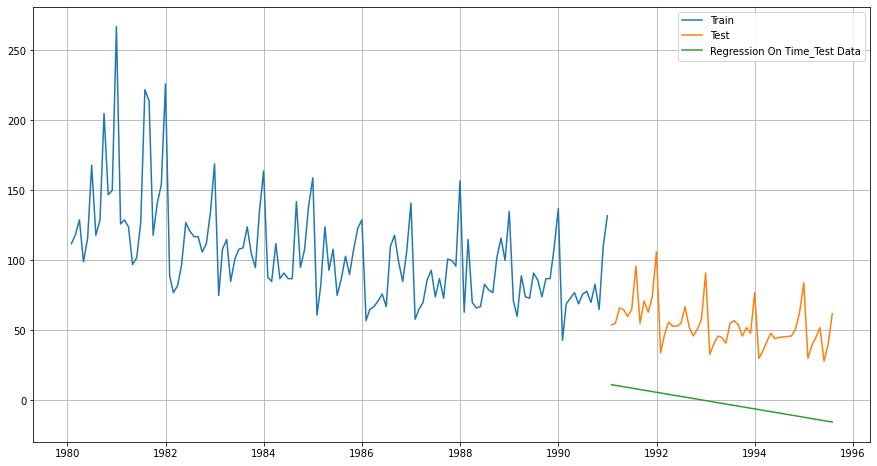
## Model 1: Linear Regression

Training Time instance

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132]

Test Time instance

[256, 257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310]



Defining the functions for calculating the accuracy metrics.

For RegressionOnTime forecast on the Test Data, RMSE is 57.790

In [95]:

RESULT

|  | **Test RMSE** |
| --- | --- |
| **RegressionOnTime** | 57.790054 |

In [96]:

## Model 2: Naive Approach: $\hat{y}\_{t+1} = y\_t$

For this particular naive model, we say that the prediction for tomorrow is the same as today and the prediction for day after tomorrow is tomorrow and since the prediction of tomorrow is same as today,therefore the prediction for day after tomorrow is also today.

TAIL OF THE DATA

|  | **Rose** |
| --- | --- |
| **Time\_Stamp** |  |
| **1990-08-31** | 70.0 |
| **1990-09-30** | 83.0 |
| **1990-10-31** | 65.0 |
| **1990-11-30** | 110.0 |
| **1990-12-31** | 132.0 |

In [100]:

NAVIE MODEL

Time\_Stamp

1991-01-31 132.0

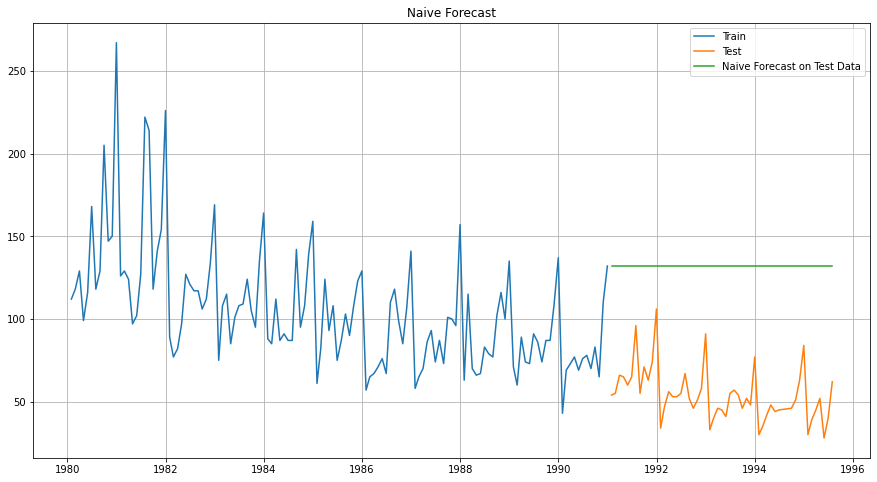
1991-02-28 132.0

1991-03-31 132.0

1991-04-30 132.0

1991-05-31 132.0

Name: naive, dtype: float64



In [102]:

### Model Evaluation

For RegressionOnTime forecast on the Test Data, RMSE is 79.719

In [105]:

RESULT

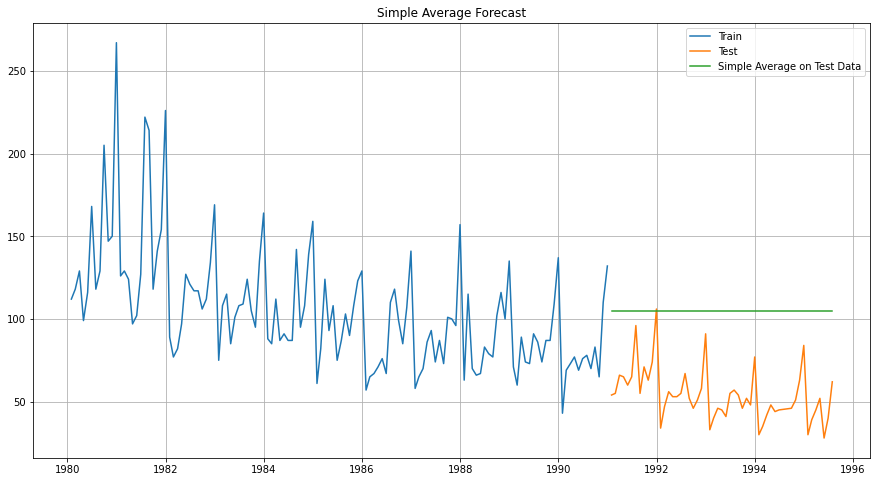
|  | **Test RMSE** |
| --- | --- |
| **RegressionOnTime** | 57.790054 |
| **NaiveModel** | 79.718773 |

In [106]:

## Method 3: Simple Average

|  | **Rose** | **mean\_forecast** |
| --- | --- | --- |
| **Time\_Stamp** |  |  |
| **1991-01-31** | 54.0 | 104.939394 |
| **1991-02-28** | 55.0 | 104.939394 |
| **1991-03-31** | 66.0 | 104.939394 |
| **1991-04-30** | 65.0 | 104.939394 |
| **1991-05-31** | 60.0 | 104.939394 |

In [109]:



In [110]:

### Model Evaluation

For Simple Average forecast on the Test Data, RMSE is 53.461

In [112]:

RESULT

|  | **Test RMSE** |
| --- | --- |
| **RegressionOnTime** | 57.790054 |
| **NaiveModel** | 79.718773 |
| **SimpleAverageModel** | 53.460570 |

# 5. Check for the stationarity of the data on which the model is being built on using appropriate statistical tests and also mention the hypothesis for the statistical test. If the data is found to be non-stationary, take appropriate steps to make it stationary. Check the new data for stationarity and comment. Note: Stationarity should be checked at alpha = 0.05.

## Check for stationarity of the whole Time Series data.

The Augmented Dickey-Fuller test is an unit root test which determines whether there is a unit root and subsequently whether the series is non-stationary.

The hypothesis in a simple form for the ADF test is:

𝐻0 : The Time Series has a unit root and is thus non-stationary.

𝐻1 : The Time Series does not have a unit root and is thus stationary.

We would want the series to be stationary for building ARIMA models and thus we would want the p-value of this test to be less than the 𝛼 value.

DF test statistic is -2.240

DF test p-value is 0.4671371627793157

Number of lags used 13

We see that at 5% significant level the Time Series is non-stationary.

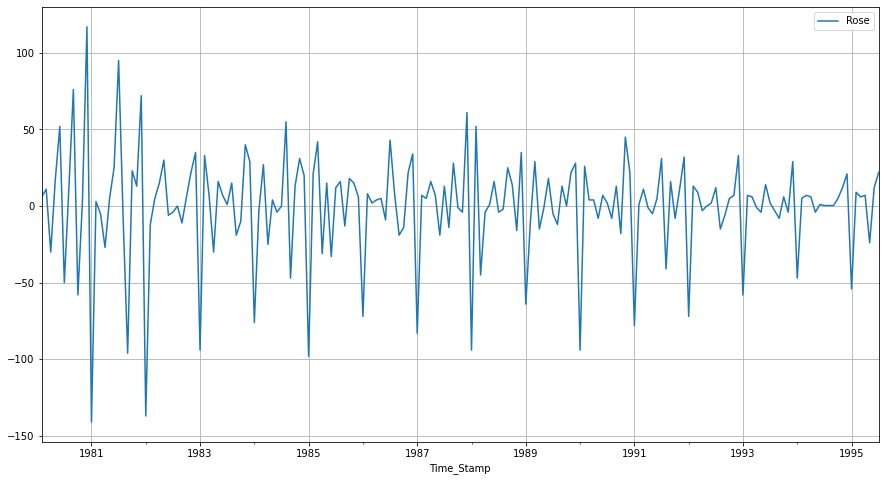
Let us take one level of differencing to see whether the series becomes stationary.

DF test statistic is -8.162

DF test p-value is 3.015976115826749e-11

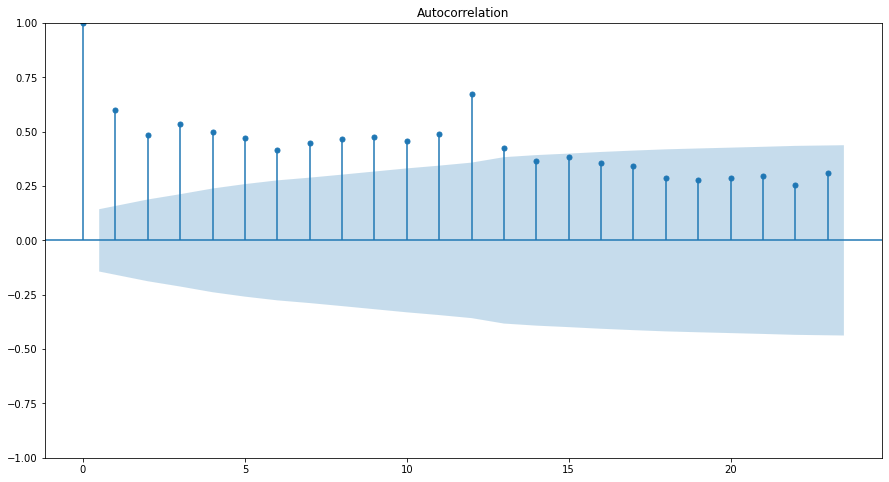
Number of lags used 12

Now, let us go ahead and plot the stationary series.

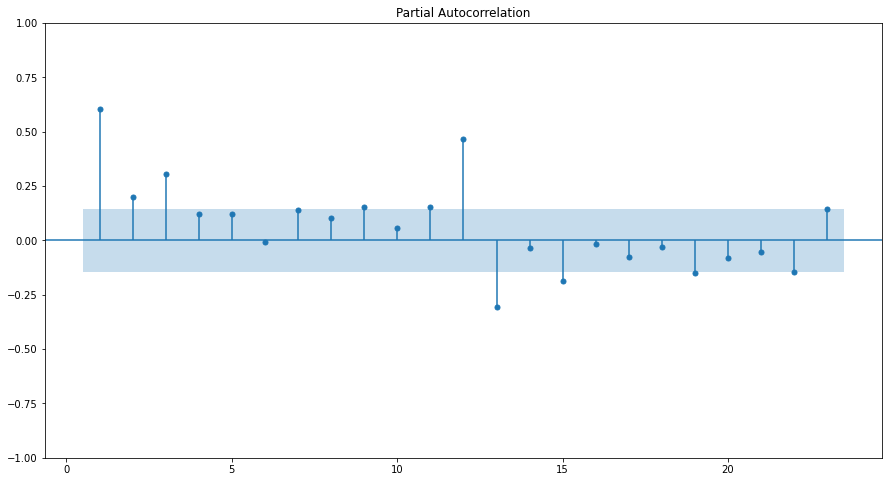


In [118]:

## Plot the Autocorrelation and the Partial Autocorrelation function plots on the whole data.

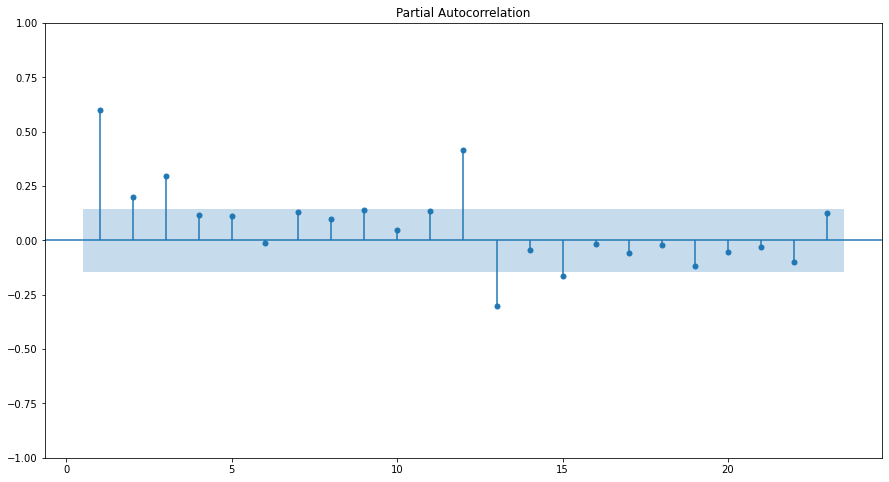


In [121]:



In [122]:

YWMLE



## Split the data into train and test and plot the training and test data.

Training Data is till the end of 2018. Test Data is from the beginning of 2019 to the last time stamp provided.

INDEX

Int64Index([1980, 1981, 1982, 1983, 1984, 1985, 1986, 1987, 1988, 1989, 1990,

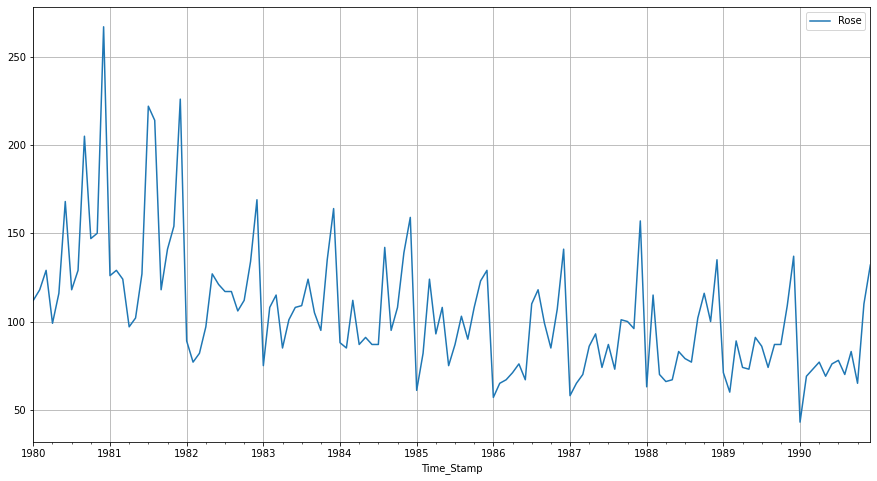
1991, 1992, 1993, 1994, 1995],

dtype='int64', name='Time\_Stamp')

## This is to display multiple data frames from one cell

## Check for stationarity of the Training Data Time Series.

Let us plot the training data once.



In [131]:

DF test statistic is -1.686

DF test p-value is 0.7569093051047051

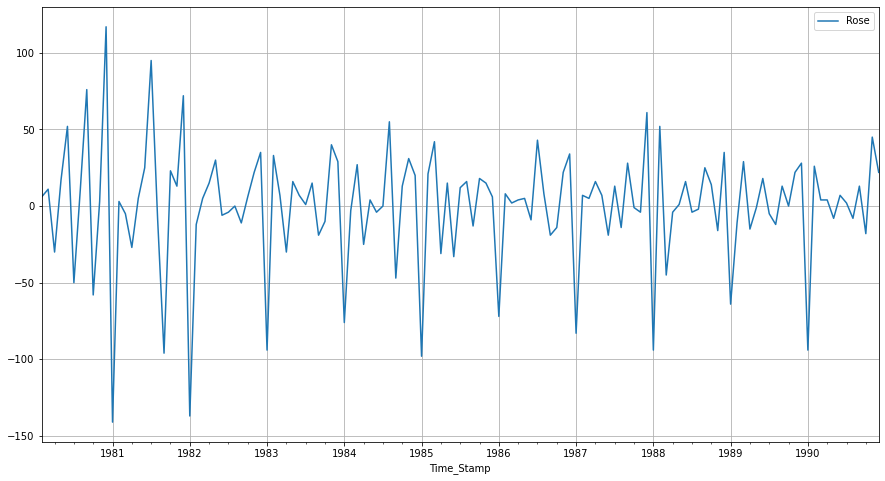
Number of lags used 13

The training data is non-stationary at 95% confidence level. Let us take a first level of differencing to stationarize the Time Series.

DF test statistic is -6.804

DF test p-value is 3.8948313567833765e-08

Number of lags used 12



##### Note: If the series is non-stationary, stationarize the Time Series by taking a difference of the Time Series.

Then we can use this particular differenced series to train the ARIMA models. We do not need to worry about stationarity for the Test Data because we are not building any models on the Test Data, we are evaluating our models over there.

We can look at other kinds of transformations as part of making the time series stationary like taking logarithms.

TRAIN INFO

<class 'pandas.core.frame.DataFrame'>

DatetimeIndex: 132 entries, 1980-01-31 to 1990-12-31

Data columns (total 1 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Rose 132 non-null float64

dtypes: float64(1)

memory usage: 2.1 KB

# 6. Build an automated version of the ARIMA/SARIMA model in which the parameters are selected using the lowest Akaike Information Criteria (AIC) on the training data and evaluate this model on the test data using RMSE.

for demonstration purposes we are building an ARIMA model both by looking at the minimum AIC criterion and by looking at the ACF and the PACF plots.

## The following loop helps us in getting a combination of different parameters of p and q in the range of 0 and 2

## We have kept the value of d as 1 as we need to take a difference of the series to make it stationary.

Examples of the parameter combinations for the Model

Model: (0, 1, 0)

Model: (0, 1, 1)

Model: (0, 1, 2)

Model: (0, 1, 3)

Model: (1, 1, 0)

Model: (1, 1, 1)

Model: (1, 1, 2)

Model: (1, 1, 3)

Model: (2, 1, 0)

Model: (2, 1, 1)

Model: (2, 1, 2)

Model: (2, 1, 3)

Model: (3, 1, 0)

Model: (3, 1, 1)

Model: (3, 1, 2)

Model: (3, 1, 3)

## Sort the above AIC values in the ascending order to get the parameters for the minimum AIC value

|  | **param** | **AIC** |
| --- | --- | --- |
| **11** | (2, 1, 3) | 1274.694837 |
| **15** | (3, 1, 3) | 1278.673068 |
| **2** | (0, 1, 2) | 1279.671529 |
| **6** | (1, 1, 2) | 1279.870723 |
| **3** | (0, 1, 3) | 1280.545376 |

In [139]:

SARIMAX Results

==============================================================================

Dep. Variable: Rose No. Observations: 132

Model: ARIMA(2, 1, 3) Log Likelihood -631.347

Date: Sun, 27 Nov 2022 AIC 1274.695

Time: 10:09:13 BIC 1291.946

Sample: 01-31-1980 HQIC 1281.705

- 12-31-1990

Covariance Type: opg

==============================================================================

coef std err z P>|z| [0.025 0.975]

------------------------------------------------------------------------------

ar.L1 -1.6779 0.084 -20.037 0.000 -1.842 -1.514

ar.L2 -0.7288 0.084 -8.701 0.000 -0.893 -0.565

ma.L1 1.0447 0.665 1.572 0.116 -0.258 2.347

ma.L2 -0.7719 0.135 -5.707 0.000 -1.037 -0.507

ma.L3 -0.9047 0.603 -1.500 0.134 -2.087 0.278

sigma2 859.3019 560.241 1.534 0.125 -238.749 1957.353

===================================================================================

Ljung-Box (L1) (Q): 0.02 Jarque-Bera (JB): 24.46

Prob(Q): 0.88 Prob(JB): 0.00

Heteroskedasticity (H): 0.40 Skew: 0.71

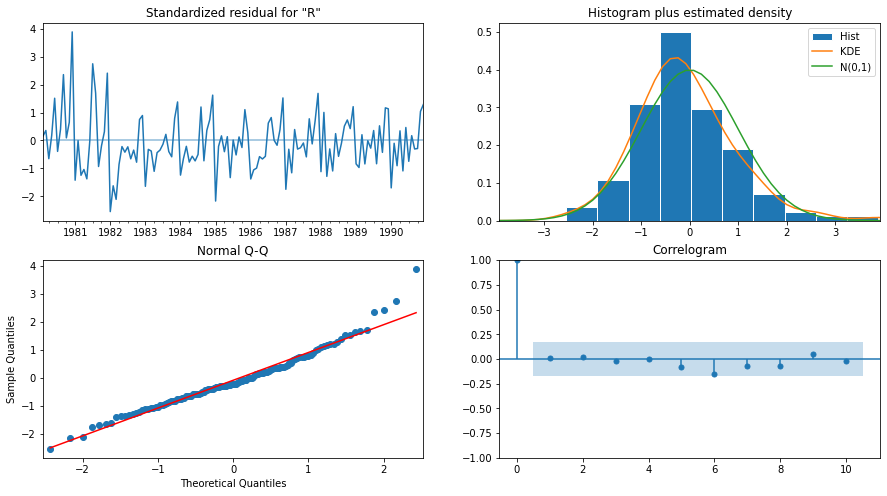
Prob(H) (two-sided): 0.00 Kurtosis: 4.57

===================================================================================

Warnings:

[1] Covariance matrix calculated using the outer product of gradients (complex-step).

Diagnostics plot.



In [141]:

## Predict on the Test Set using this model and evaluate the model.

## Mean Absolute Percentage Error (MAPE) - Function Definition

## Importing the mean\_squared\_error function from sklearn to calculate the RMSE

RMSE: 36.81246514792361

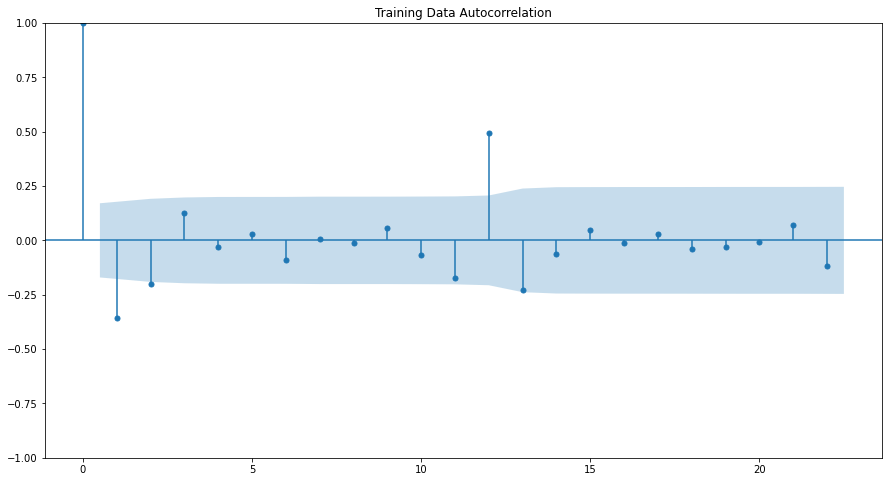
MAPE: 75.83765356015655

RESULT

|  | **RMSE** | **MAPE** |
| --- | --- | --- |
| **ARIMA(2,1,3)** | 36.812465 | 75.837654 |

In [146]:

#SARIMA



In [148]:

Examples of the parameter combinations for the Model are

Model: (0, 1, 1)(0, 0, 1, 5)

Model: (0, 1, 2)(0, 0, 2, 5)

Model: (1, 1, 0)(1, 0, 0, 5)

Model: (1, 1, 1)(1, 0, 1, 5)

Model: (1, 1, 2)(1, 0, 2, 5)

Model: (2, 1, 0)(2, 0, 0, 5)

Model: (2, 1, 1)(2, 0, 1, 5)

Model: (2, 1, 2)(2, 0, 2, 5)

In [149]:

SORT VALUE

|  | **param** | **seasonal** | **AIC** |
| --- | --- | --- | --- |
| **80** | (2, 1, 2) | (2, 0, 2, 5) | 1132.421381 |
| **26** | (0, 1, 2) | (2, 0, 2, 5) | 1134.372547 |
| **53** | (1, 1, 2) | (2, 0, 2, 5) | 1134.464636 |
| **23** | (0, 1, 2) | (1, 0, 2, 5) | 1136.146037 |
| **20** | (0, 1, 2) | (0, 0, 2, 5) | 1137.212194 |

In [154]:

SARIMAX Results

=========================================================================================

Dep. Variable: Rose No. Observations: 132

Model: SARIMAX(2, 1, 2)x(2, 0, 2, 5) Log Likelihood -557.211

Date: Sun, 27 Nov 2022 AIC 1132.421

Time: 10:12:59 BIC 1157.358

Sample: 01-31-1980 HQIC 1142.546

- 12-31-1990

Covariance Type: opg

==============================================================================

coef std err z P>|z| [0.025 0.975]

------------------------------------------------------------------------------

ar.L1 -0.7139 0.088 -8.139 0.000 -0.886 -0.542

ar.L2 0.2680 0.091 2.933 0.003 0.089 0.447

ma.L1 -8.265e-05 109.806 -7.53e-07 1.000 -215.216 215.216

ma.L2 -1.0001 109.457 -0.009 0.993 -215.533 213.532

ar.S.L5 0.2037 0.087 2.347 0.019 0.034 0.374

ar.S.L10 0.5215 0.074 7.014 0.000 0.376 0.667

ma.S.L5 -0.0002 86.321 -1.83e-06 1.000 -169.186 169.186

ma.S.L10 -0.9997 101.882 -0.010 0.992 -200.685 198.686

sigma2 536.1804 7.83e+04 0.007 0.995 -1.53e+05 1.54e+05

===================================================================================

Ljung-Box (L1) (Q): 0.22 Jarque-Bera (JB): 2.95

Prob(Q): 0.64 Prob(JB): 0.23

Heteroskedasticity (H): 0.52 Skew: -0.39

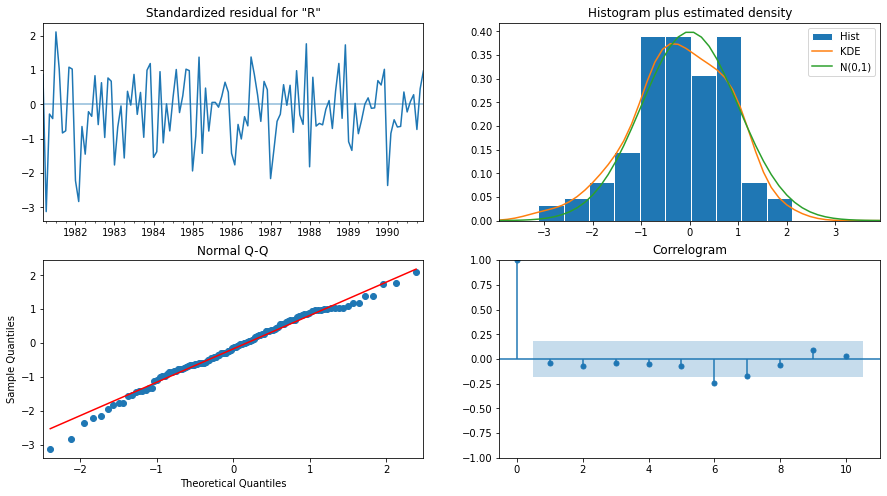
Prob(H) (two-sided): 0.04 Kurtosis: 3.08

===================================================================================

Warnings:

[1] Covariance matrix calculated using the outer product of gradients (complex-step).

Diagnostics PLOT



In [156]:

## Predict on the Test Set using this model and evaluate the model.

| **Rose** | **mean** | **mean\_se** | **mean\_ci\_lower** | **mean\_ci\_upper** |
| --- | --- | --- | --- | --- |
| **1991-01-31** | 99.923652 | 24.620531 | 51.668299 | 148.179006 |
| **1991-02-28** | 103.939011 | 25.581552 | 53.800090 | 154.077932 |
| **1991-03-31** | 92.823244 | 25.725826 | 42.401551 | 143.244937 |
| **1991-04-30** | 85.280526 | 25.719254 | 34.871714 | 135.689338 |
| **1991-05-31** | 95.582840 | 25.740912 | 45.131581 | 146.034100 |

In [159]:

RMSE: 38.72772381346243

MAPE: 79.64422170429181

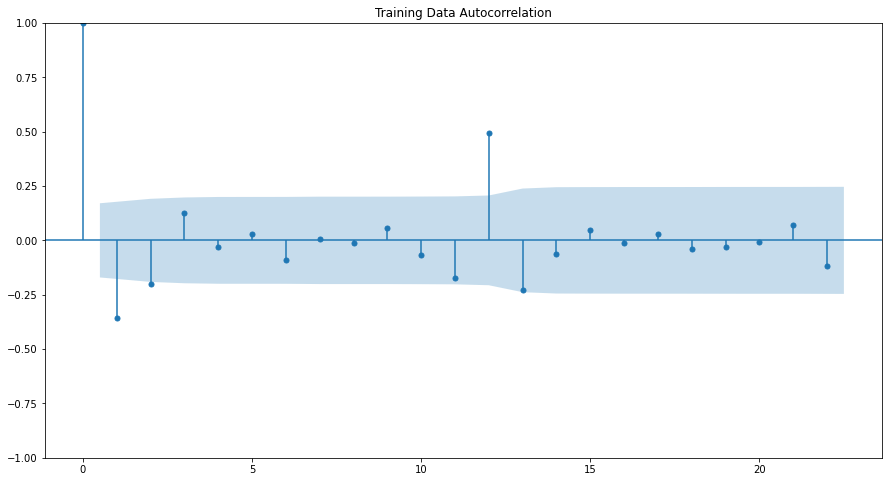
RESULT

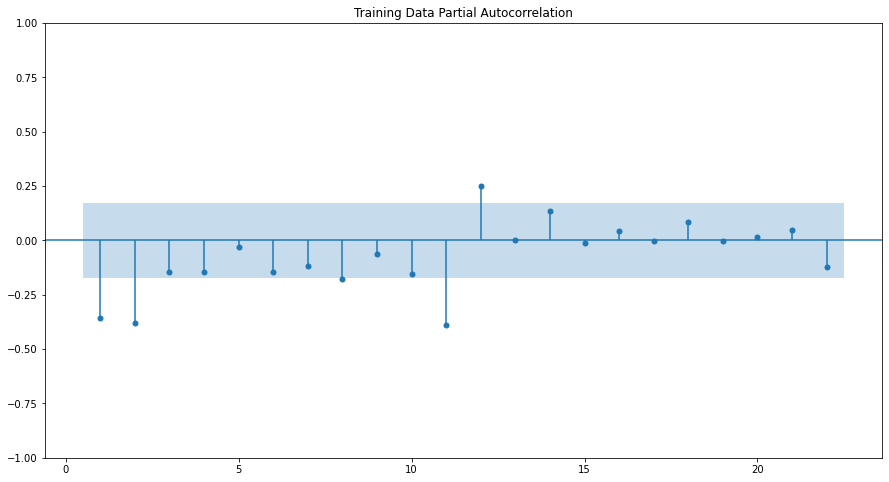
|  | **RMSE** | **MAPE** |
| --- | --- | --- |
| **ARIMA(2,1,3)** | 36.812465 | 75.837654 |
| **SARIMA(2,1,2)(2,0,2,5)** | 38.727724 | 79.644222 |

# 7. Build ARIMA/SARIMA models based on the cut-off points of ACF and PACF on the training data and evaluate this model on the test data using RMSE.

ARIMA

Let us look at the ACF and the PACF plots





Here, we have taken alpha=0.05.

The Auto-Regressive parameter in an ARIMA model is 'p' which comes from the significant lag before which the PACF plot cuts-off to 3.

The Moving-Average parameter in an ARIMA model is 'q' which comes from the significant lag before the ACF plot cuts-off to 3.

By looking at the above plots, we will take the value of p and q to be 4 and 1 respectively.

SARIMAX Results

==============================================================================

Dep. Variable: Rose No. Observations: 132

Model: ARIMA(4, 1, 4) Log Likelihood -632.146

Date: Sun, 27 Nov 2022 AIC 1282.292

Time: 10:15:54 BIC 1308.169

Sample: 01-31-1980 HQIC 1292.807

- 12-31-1990

Covariance Type: opg

==============================================================================

coef std err z P>|z| [0.025 0.975]

------------------------------------------------------------------------------

ar.L1 -0.5355 0.669 -0.801 0.423 -1.846 0.775

ar.L2 -0.8293 0.250 -3.323 0.001 -1.318 -0.340

ar.L3 -0.5475 0.652 -0.840 0.401 -1.825 0.730

ar.L4 0.1361 0.237 0.575 0.565 -0.327 0.600

ma.L1 -0.1821 0.666 -0.273 0.785 -1.488 1.124

ma.L2 0.2531 0.647 0.391 0.696 -1.015 1.522

ma.L3 -0.0664 0.663 -0.100 0.920 -1.365 1.232

ma.L4 -0.7017 0.573 -1.224 0.221 -1.826 0.422

sigma2 870.4120 86.423 10.072 0.000 701.027 1039.797

===================================================================================

Ljung-Box (L1) (Q): 0.01 Jarque-Bera (JB): 41.40

Prob(Q): 0.91 Prob(JB): 0.00

Heteroskedasticity (H): 0.38 Skew: 0.85

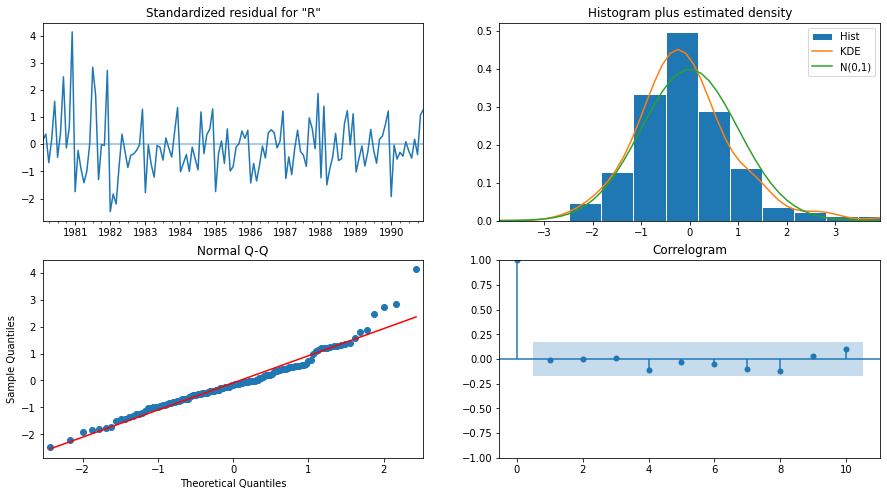
Prob(H) (two-sided): 0.00 Kurtosis: 5.16

===================================================================================

Warnings:

[1] Covariance matrix calculated using the outer product of gradients (complex-step).

Let us analyse the residuals from the various diagnostics plot.



## Predict on the Test Set using this model and evaluate the model.

RMSE: 36.808630051585226

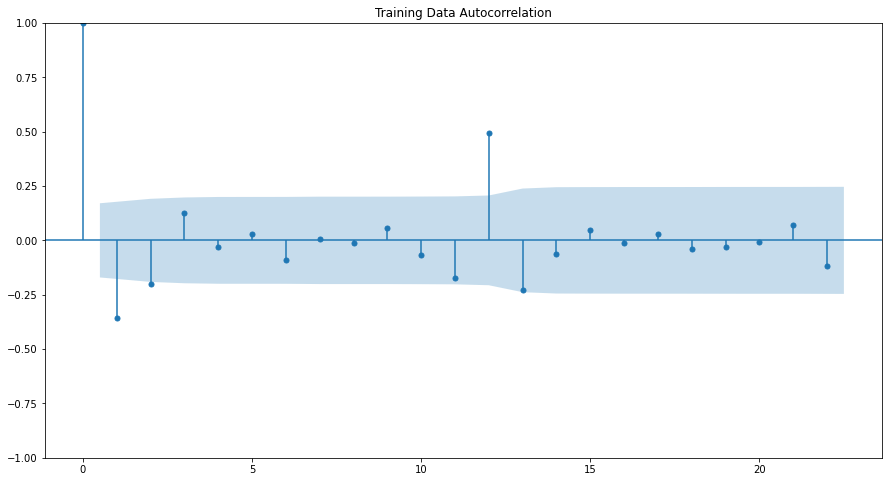
MAPE: 75.77236274328963

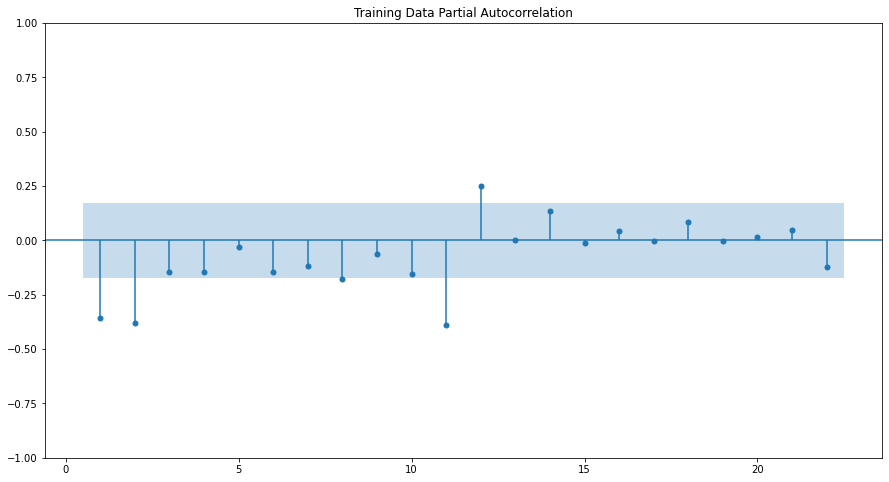
RESULT

|  | **RMSE** | **MAPE** |
| --- | --- | --- |
| **ARIMA(2,1,3)** | 36.812465 | 75.837654 |
| **SARIMA(2,1,2)(2,0,2,5)** | 38.727724 | 79.644222 |
| **ARIMA(3,1,3)** | 36.808630 | 75.772363 |

In [168]:

#SARIMA





Here, we have taken alpha=0.05.

We are going to take the seasonal period as 3 or its multiple e.g. 6. We are taking the p value to be 3 and the q value also to be 3 as the parameters same as the ARIMA model.

The Auto-Regressive parameter in an SARIMA model is 'P' which comes from the significant lag after which the PACF plot cuts-off to 0.

The Moving-Average parameter in an SARIMA model is 'Q' which comes from the significant lag after which the ACF plot cuts-off to 3.

SARIMAX Results

=========================================================================================

Dep. Variable: Rose No. Observations: 132

Model: SARIMAX(3, 1, 3)x(0, 0, 3, 6) Log Likelihood -487.076

Date: Sun, 27 Nov 2022 AIC 994.152

Time: 10:19:25 BIC 1021.065

Sample: 01-31-1980 HQIC 1005.066

- 12-31-1990

Covariance Type: opg

==============================================================================

coef std err z P>|z| [0.025 0.975]

------------------------------------------------------------------------------

ar.L1 0.1008 0.110 0.916 0.360 -0.115 0.317

ar.L2 0.6254 0.115 5.442 0.000 0.400 0.851

ar.L3 -0.1695 0.104 -1.637 0.102 -0.372 0.033

ma.L1 -0.9211 0.123 -7.483 0.000 -1.162 -0.680

ma.L2 -0.7848 0.150 -5.246 0.000 -1.078 -0.492

ma.L3 0.7398 0.125 5.915 0.000 0.495 0.985

ma.S.L6 -0.1352 0.143 -0.946 0.344 -0.415 0.145

ma.S.L12 0.5787 0.113 5.117 0.000 0.357 0.800

ma.S.L18 0.0666 0.153 0.435 0.664 -0.234 0.367

sigma2 409.6721 62.024 6.605 0.000 288.106 531.238

===================================================================================

Ljung-Box (L1) (Q): 0.00 Jarque-Bera (JB): 0.27

Prob(Q): 0.96 Prob(JB): 0.88

Heteroskedasticity (H): 0.78 Skew: -0.12

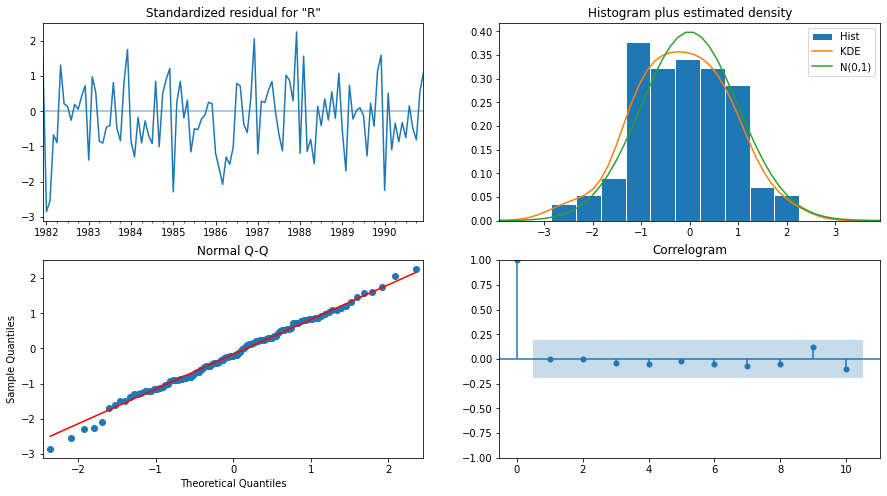
Prob(H) (two-sided): 0.45 Kurtosis: 2.93

===================================================================================

Warnings:

[1] Covariance matrix calculated using the outer product of gradients (complex-step).

Let us analyse the residuals from the various diagnostics plot.



In [172]:

## Predict on the Test Set using this model and evaluate the model.

RMSE: 29.721558504612833

MAPE: 60.37771409213758

RESULT

|  | **RMSE** | **MAPE** |
| --- | --- | --- |
| **ARIMA(2,1,3)** | 36.812465 | 75.837654 |
| **SARIMA(2,1,2)(2,0,2,5)** | 38.727724 | 79.644222 |
| **ARIMA(3,1,3)** | 36.808630 | 75.772363 |
| **SARIMA(3,1,3)(0,0,3,6)** | 29.721559 | 60.377714 |

# 8. Build a table (create a data frame) with all the models built along with their corresponding parameters and the respective RMSE values on the test data.

## Building the most optimum model on the Full Data.

SARIMAX Results

=========================================================================================

Dep. Variable: Rose No. Observations: 187

Model: SARIMAX(0, 1, 2)x(2, 0, 2, 6) Log Likelihood -734.147

Date: Sun, 27 Nov 2022 AIC 1482.294

Time: 10:21:07 BIC 1504.286

Sample: 01-31-1980 HQIC 1491.218

- 07-31-1995

Covariance Type: opg

==============================================================================

coef std err z P>|z| [0.025 0.975]

------------------------------------------------------------------------------

ma.L1 -0.7297 0.070 -10.350 0.000 -0.868 -0.592

ma.L2 -0.1899 0.066 -2.883 0.004 -0.319 -0.061

ar.S.L6 -0.0496 0.029 -1.687 0.092 -0.107 0.008

ar.S.L12 0.8766 0.030 29.411 0.000 0.818 0.935

ma.S.L6 0.1746 0.235 0.744 0.457 -0.286 0.635

ma.S.L12 -0.7825 0.194 -4.033 0.000 -1.163 -0.402

sigma2 283.5320 58.864 4.817 0.000 168.161 398.903

===================================================================================

Ljung-Box (L1) (Q): 0.19 Jarque-Bera (JB): 297.19

Prob(Q): 0.66 Prob(JB): 0.00

Heteroskedasticity (H): 0.17 Skew: 0.45

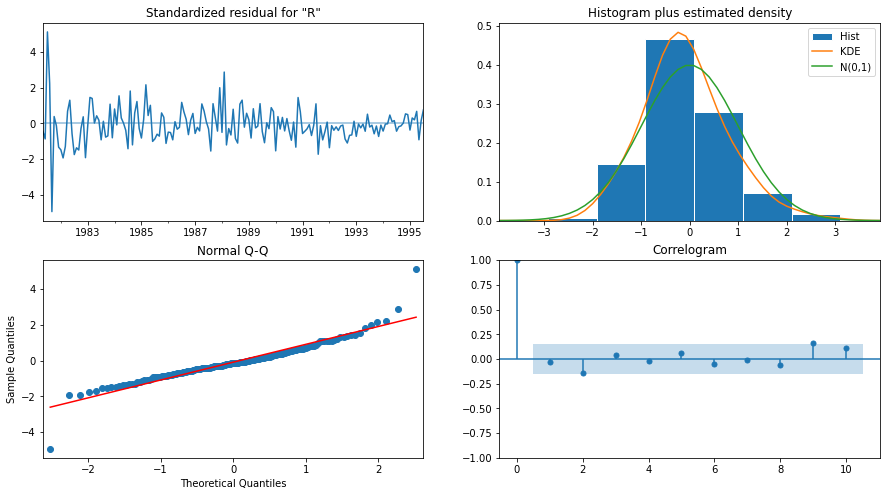
Prob(H) (two-sided): 0.00 Kurtosis: 9.40

===================================================================================

Warnings:

[1] Covariance matrix calculated using the outer product of gradients (complex-step).

Let us analyse the residuals from the various diagnostics plot.



# 9. Based on the model-building exercise, build the most optimum model(s) on the complete data and predict 12 months into the future with appropriate confidence intervals/bands.

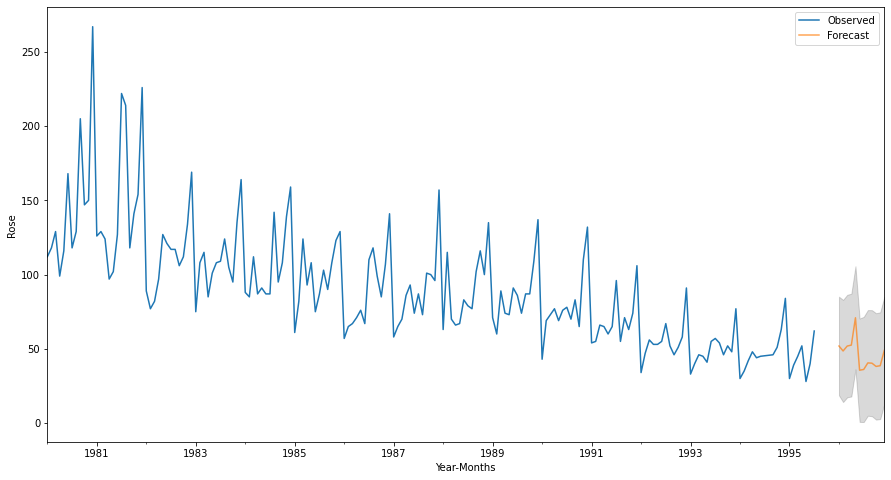
| **Rose** | **mean** | **mean\_se** | **mean\_ci\_lower** | **mean\_ci\_upper** |
| --- | --- | --- | --- | --- |
| **1995-08-31** | 51.906634 | 16.960539 | 18.664589 | 85.148679 |
| **1995-09-30** | 48.560881 | 17.562343 | 14.139321 | 82.982441 |
| **1995-10-31** | 51.876502 | 17.615196 | 17.351352 | 86.401653 |
| **1995-11-30** | 52.523982 | 17.667891 | 17.895551 | 87.152413 |
| **1995-12-31** | 70.922130 | 17.720430 | 36.190725 | 105.653534 |

In [181]:

RMSE of the Full Model 28.050812558421214

In [182]:

# plot the forecast along with the confidence band



In [184]

RESULT

|  | **RMSE** | **MAPE** |
| --- | --- | --- |
| **ARIMA(2,1,3)** | 36.812465 | 75.837654 |
| **SARIMA(2,1,2)(2,0,2,5)** | 38.727724 | 79.644222 |
| **ARIMA(3,1,3)** | 36.808630 | 75.772363 |
| **SARIMA(3,1,3)(0,0,3,6)** | 29.721559 | 60.377714 |

END \*\*\*