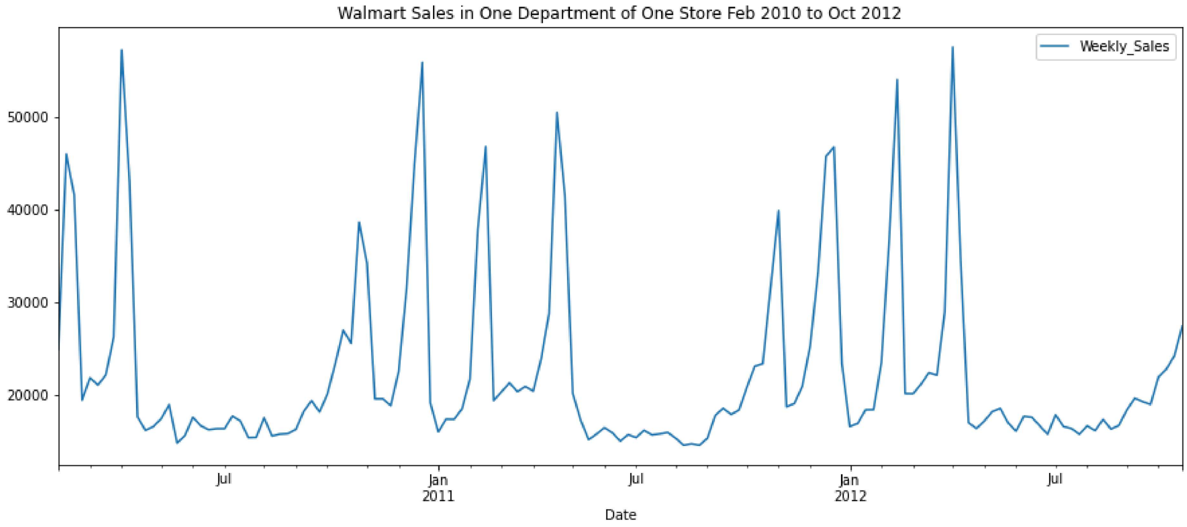
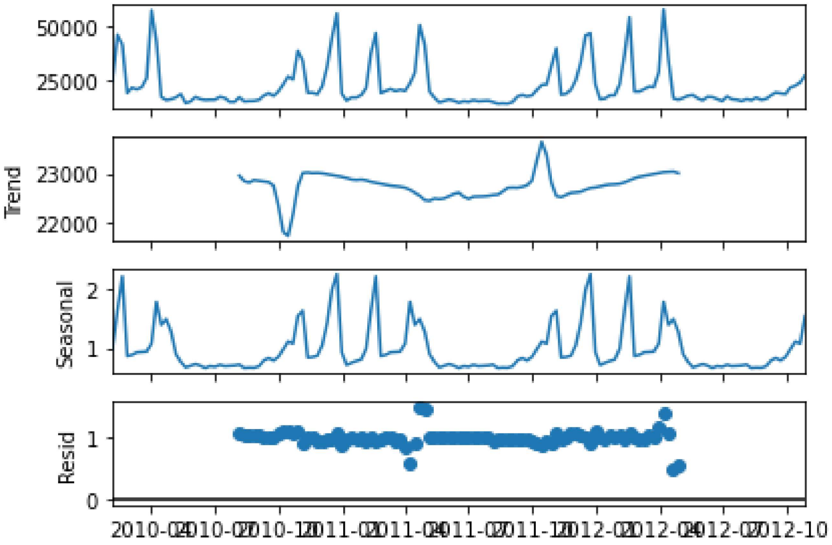
In [7]:

```
# Plotting
dat_grouped.plot(figsize=(15, 6), title="Walmart Sales in One Department o
plt.show()
```



In [8]:

```
# Seasonal Decomposition
result = seasonal_decompose(dat_grouped, model='multiplicative')
fig = result.plot()
plt.show()
```



In [9]:

```
# ADF Test
result_adf = adfuller(dat_grouped['Weekly_Sales'])
print('ADF Statistic:', result_adf[0])
print('p-value:', result_adf[1])
```

```
ADF Statistic: -2.5221638686500403
p-value: 0.11017432282586914
```

In [10]:

```
# Log Transformation
dat_grouped['Weekly_Sales_log'] = np.log(dat_grouped['Weekly_Sales'])
```

In [11]:

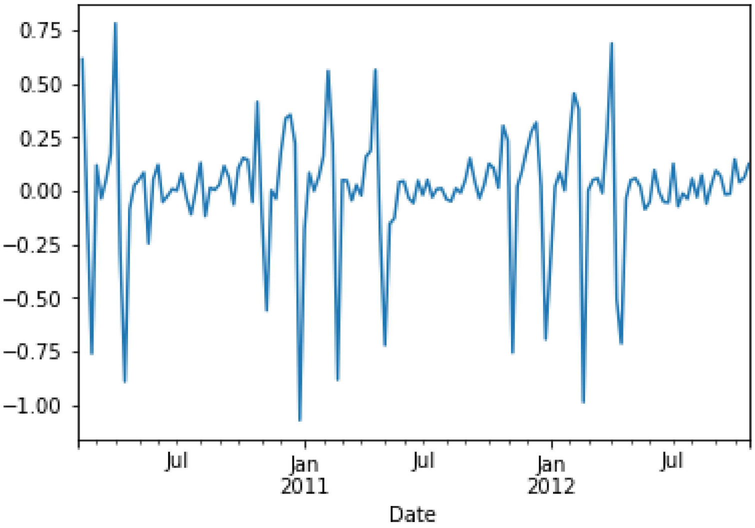
```
# Calculate first differences
dat_grouped["Weekly_Sales_diff"] = dat_grouped['Weekly_Sales_log'].diff()

```

In [12]:

```
dat_grouped["Weekly_Sales_diff"].plot()
```

Out[12]:

```
<AxesSubplot:xlabel='Date'>
```



In [13]:

```
# Convert to growth rates (multiply by 100)
dat_grouped['Weekly_Sales_growth'] = dat_grouped['Weekly_Sales_diff'] * 10

```

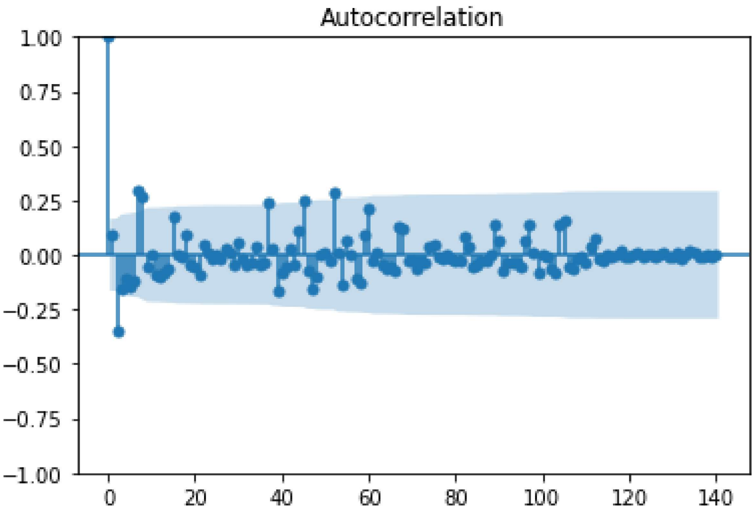
In [14]:

```
print(dat_grouped.head())
```

```
            Weekly_Sales  Weekly_Sales_log  Weekly_Sales_diff  \
Date
2010-02-05      24924.50         10.123607                NaN
2010-02-12      46039.49         10.737255           0.613648
2010-02-19      41595.55         10.635748          -0.101506
2010-02-26      19403.54          9.873211          -0.762538
2010-03-05      21827.90          9.990944           0.117733

            Weekly_Sales_growth
Date
2010-02-05                  NaN
2010-02-12            61.364825
2010-02-19           -10.150632
2010-02-26           -76.253767
2010-03-05            11.773344
```

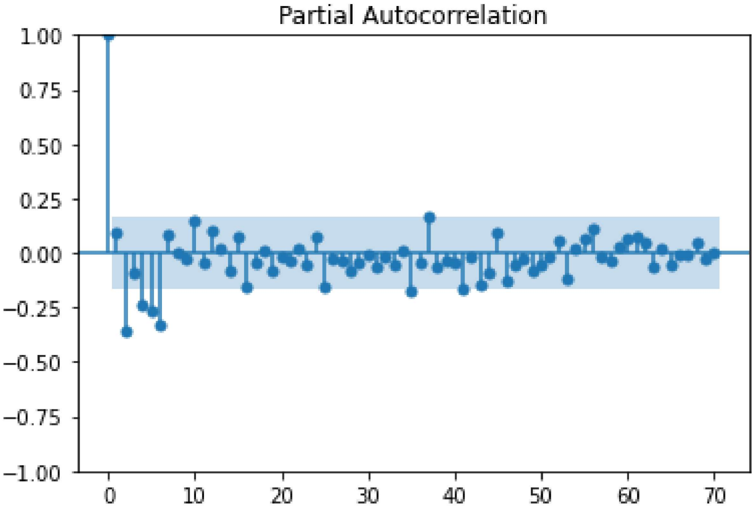
In [16]:

```
#acf plot
plot_acf(dat_grouped['Weekly_Sales_growth'].dropna(), lags=140)
plt.show()
```



In [18]:

```
#pacf plot
plot_pacf(dat_grouped['Weekly_Sales_growth'].dropna(), lags=70)
plt.show()
```



In [ ]:

```
# SARIMA Model
train = dat_grouped.loc[:'2012-09-01']['Weekly_Sales']

# Use auto_arima to find optimal SARIMA parameters
sarima_model = auto_arima(train, seasonal=True, m=12, trace=True, suppress

# Training the Model
sarima_model.fit(train)
```