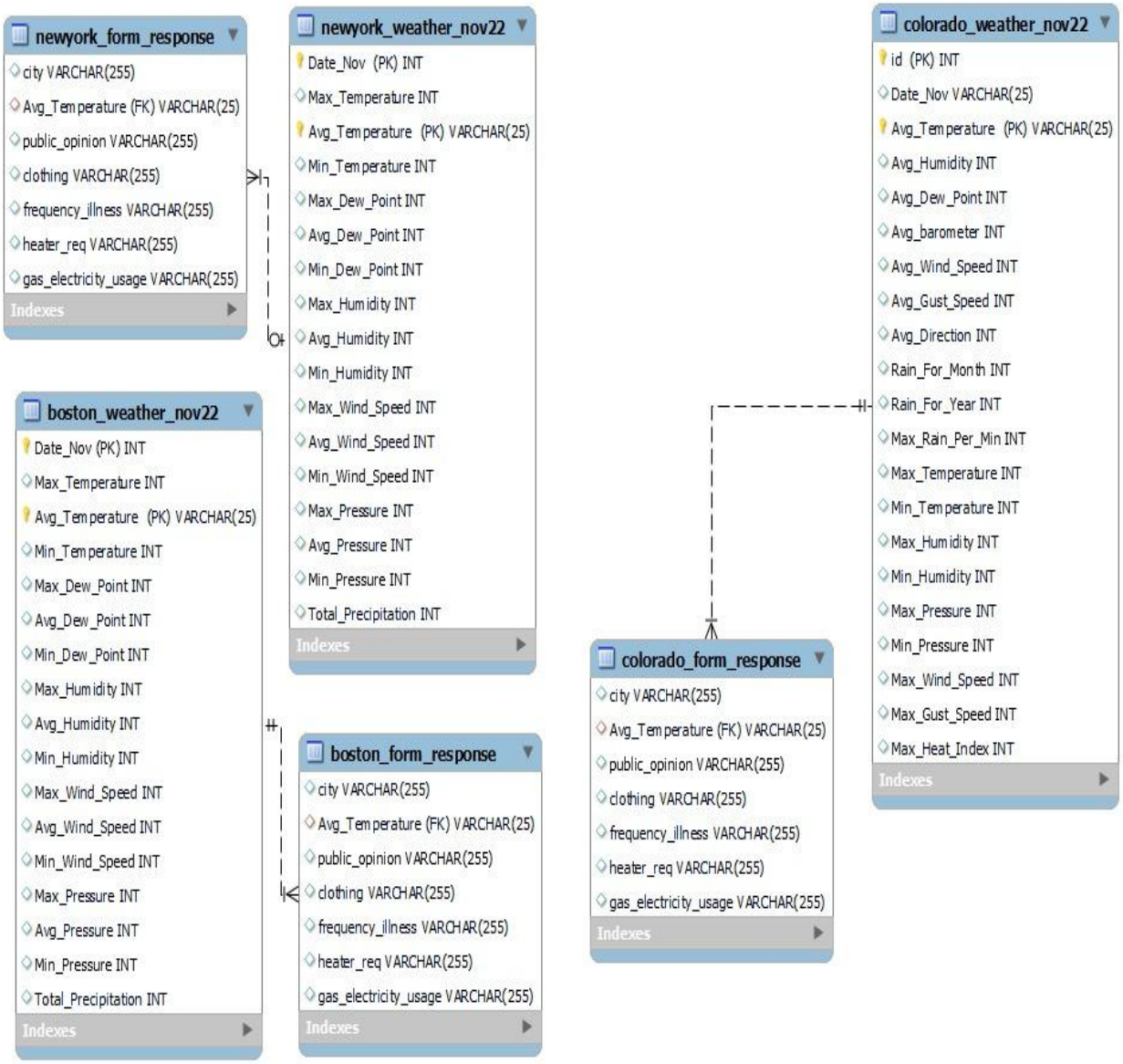
# Assignment 3 - Gathering, Scraping, Munging and Cleaning Data

**Team name : errorFree**
**Group Members:**
**Ritik Bhandari(002738904)**
**Rishi Shelly(002771020)**
**Dhiral Mayavanshi(002706357 )**

**Database Schema: ER Diagram**



## Data Cleaning using Python Code:

```python
    f_index = [index[i] for i in range(len(index)) if len(index[i]) > 6]
    data = [df_list[i][1:] for i in range(len(df_list)) if len(df_list[i][1:]) == 19]

from datetime import datetime

final_index = [datetime.strptime(str(f_index[i]), '%Y%m%d').strftime('%Y-%m-%d') for i in range(len(f_index))]

# Data Cleaning - Renaming the columns
col = ['Average temperature (°F)', 'Average humidity (%)',
       'Average dewpoint (°F)', 'Average barometer (in)',
       'Average windspeed (mph)', 'Average gustspeed (mph)',
       'Average direction (°deg)', 'Rainfall for month (in)',
       'Rainfall for year (in)', 'Maximum rain per minute',
       'Maximum temperature (°F)', 'Minimum temperature (°F)',
       'Maximum humidity (%)', 'Minimum humidity (%)', 'Maximum pressure',
       'Minimum pressure', 'Maximum windspeed (mph)',
       'Maximum gust speed (mph)', 'Maximum heat index (°F)']
df = pd.DataFrame(data, columns=col, index=final_index)
df.head(30)
df.to_csv('November22_Colorado.csv')

# Data Cleaning - Deleting Multiple Duplicate Rows
df_state = pd.read_csv('November22_Colorado')
Dup_Rows = df_state[df_state.duplicated()]
DF_RM_DUP = df_state.drop_duplicates(keep='first')
print(DF_RM_DUP)
DF_RM_DUP.to_csv('Colorado_Weather_Nov22.csv')

# Data Cleaning - Checking for Null Values
print(DF_RM_DUP.isnull().sum())
```

## Code For Creating tables and inserting the data into MySQL Workbench:

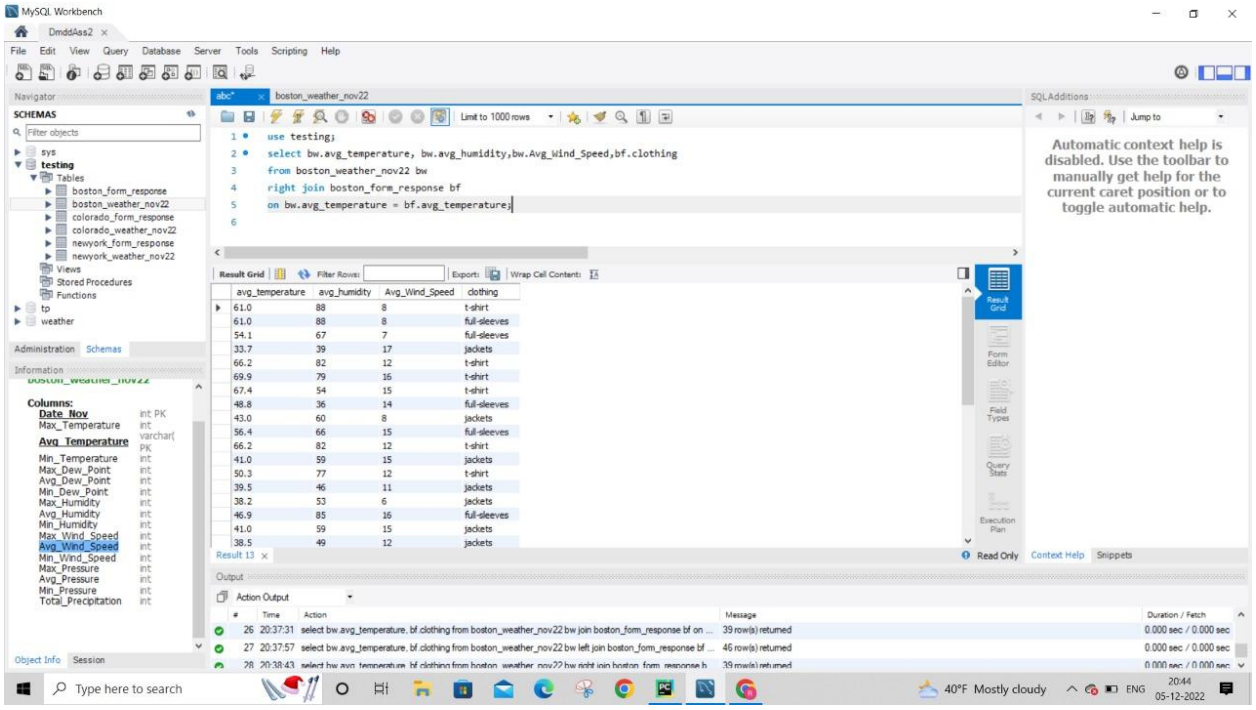
```python
mycursor = DmddAss2.cursor()
#Creating Tables for MySQL Workbench
mycursor.execute("CREATE TABLE Boston_Weather_Nov22 (Date_Nov INT ,Max_Temperature INT,"
                 " Avg_Temperature VARCHAR(25) PRIMARY KEY,Min_Temperature INT,"
                "Max_Dew_Point INT, Avg_Dew_Point INT,Min_Dew_Point INT,"
                 "Max_Humidity INT, Avg_Humidity INt, Min_Humidity INT,"
                  "Max_Wind_Speed INT, Avg_Wind_Speed INT,Min_Wind_Speed INT,"
                  "Max_Pressure INT, Avg_Pressure INT, Min_Pressure INT,Total_Precipitation INT)")

df = pd.read_csv(r'C:\Users\Asus\Desktop\BostonWeatherNov22.csv')
print(df)

# Inserting Data into MySQL WOrkbench
for i_row in df.iterrows():
    sql = "INSERT INTO testing.Boston_weather_Nov22 values(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)"
    mycursor.execute(sql,tuple(row))
    DmddAss2.commit()

DmddAss2= mysql.connector.connect(host='localhost', user='root', passwd='root@123', database='testing')
mycursor = DmddAss2.cursor()
#Creating Tables for MySQL Workbench
mycursor.execute("CREATE TABLE Boston_form_response (city VARCHAR(255), Avg_Temperature VARCHAR(25),FOREIGN KEY(Avg_Temperature)

df = pd.read_csv(r'C:\Users\Asus\Desktop\DmddAss3\Boston_form_response.csv')
print(df)
# Inserting Data into MySQL WOrkbench
for i_row in df.iterrows():
    sql = "INSERT INTO testing.Boston_form_response values(%s,%s,%s,%s,%s,%s,%s)"
    mycursor.execute(sql,tuple(row))
    DmddAss2.commit()
```

**Description :**

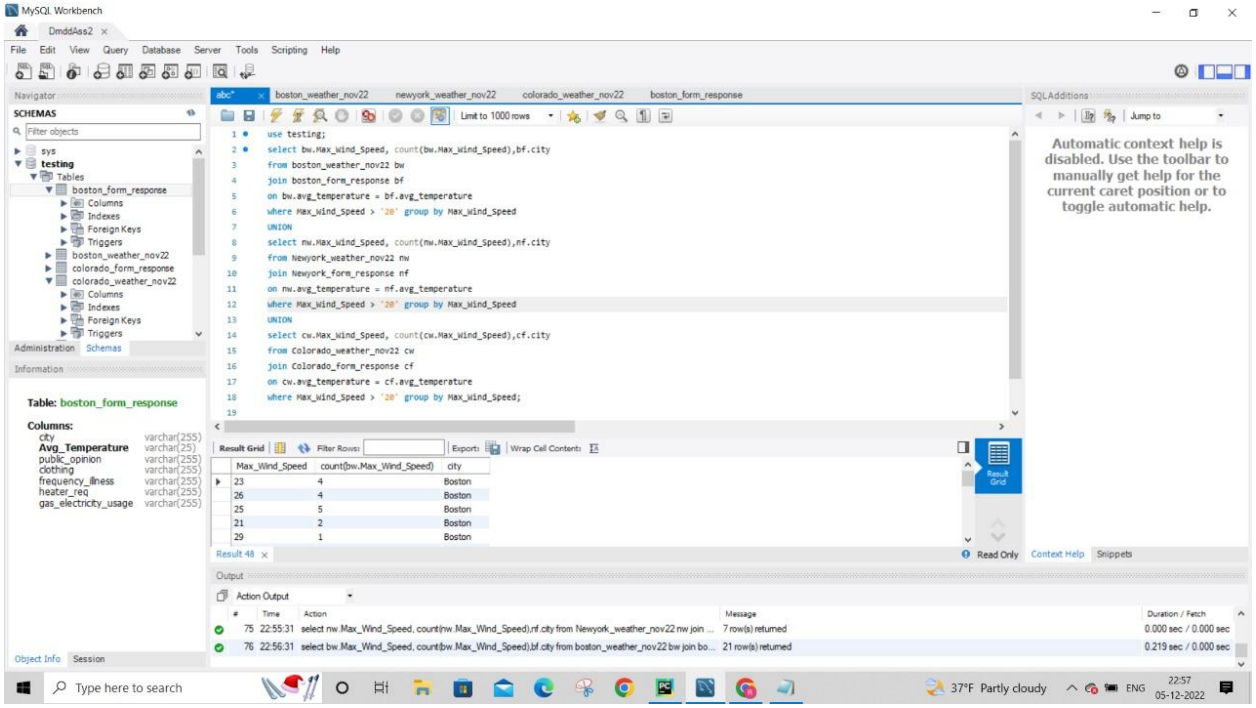
**Weather Analysis for 3 cities ,NewYork , Boston and Colorado was done. The data for Colorado was Scraped from the website - [Estes Park Weather - Home/Forecasts]() and data for Boston and New York was collected from - [East Boston, MA Weather History | Weather Underground (wunderground.com)]. Public Opinions and responses was collected from the people of these cities and then Analysis was performed using various MySQl Workbench. Data Visulaization of the collected data was also done to present the analyzed data in an effective manner.**

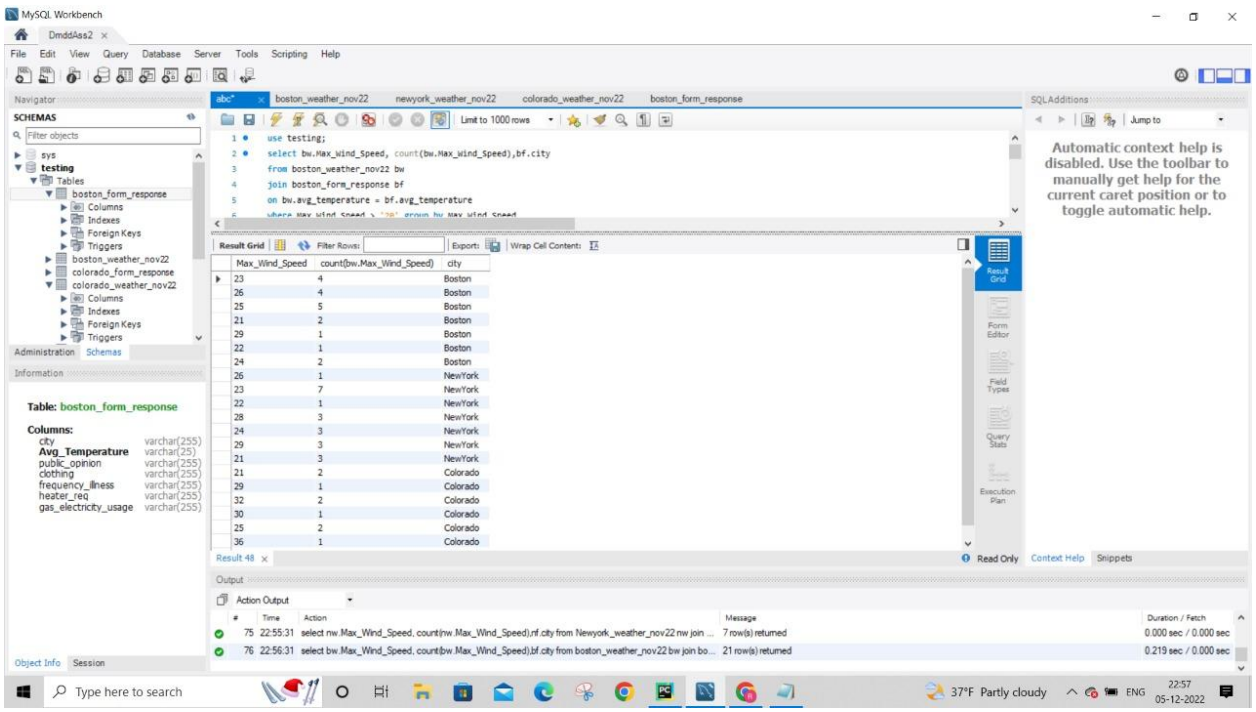
**Use Cases:**

**Use Case 1: Clothing preference of people in Boston according to it's weather conditions**

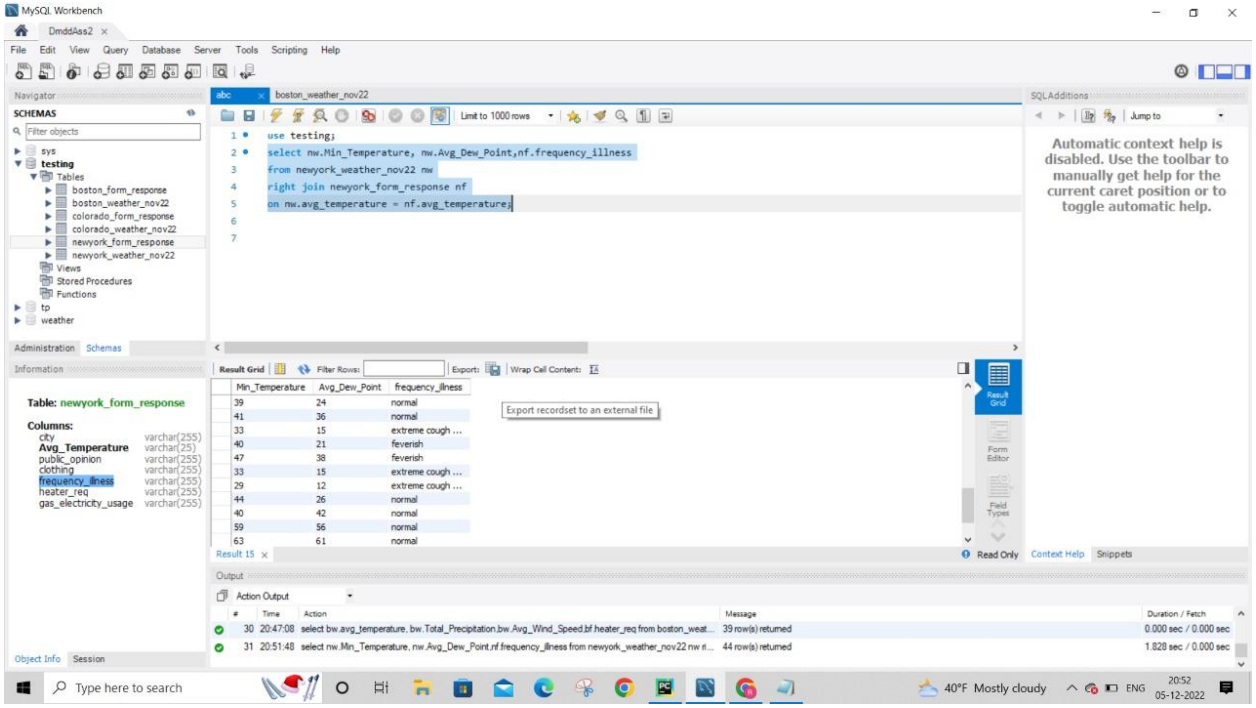



**Use Case 2: Which is the best city for generating win energy?**

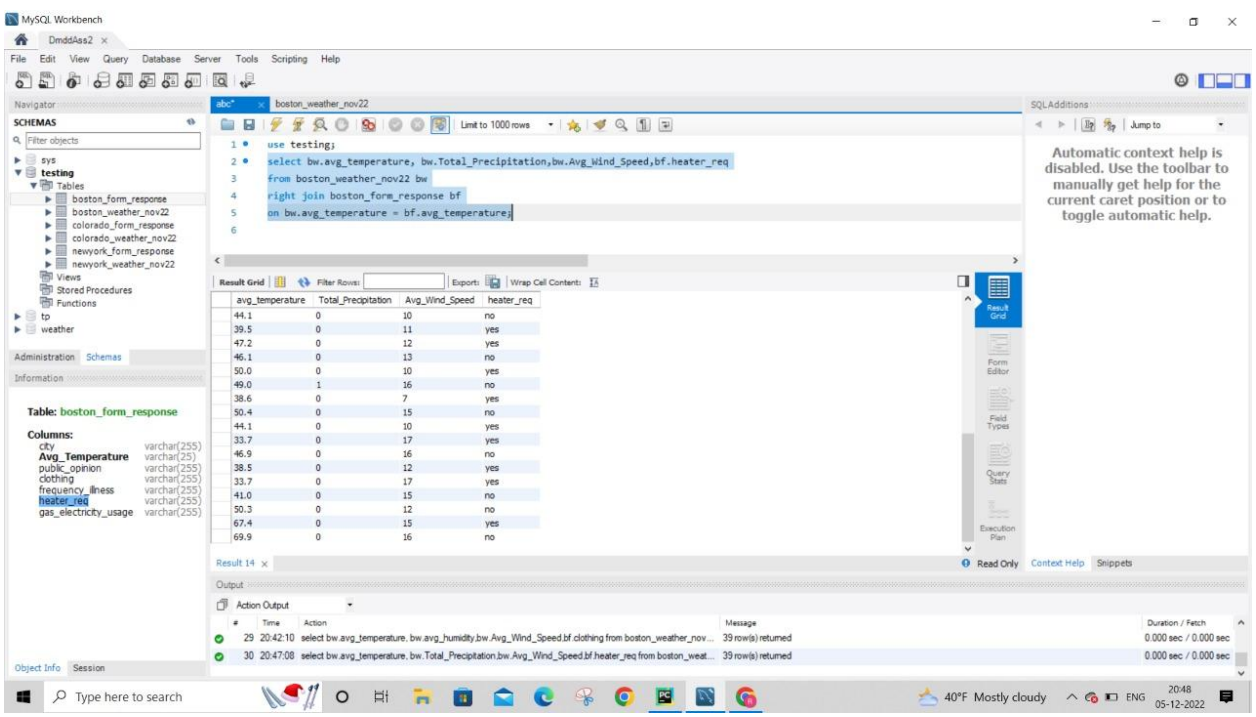


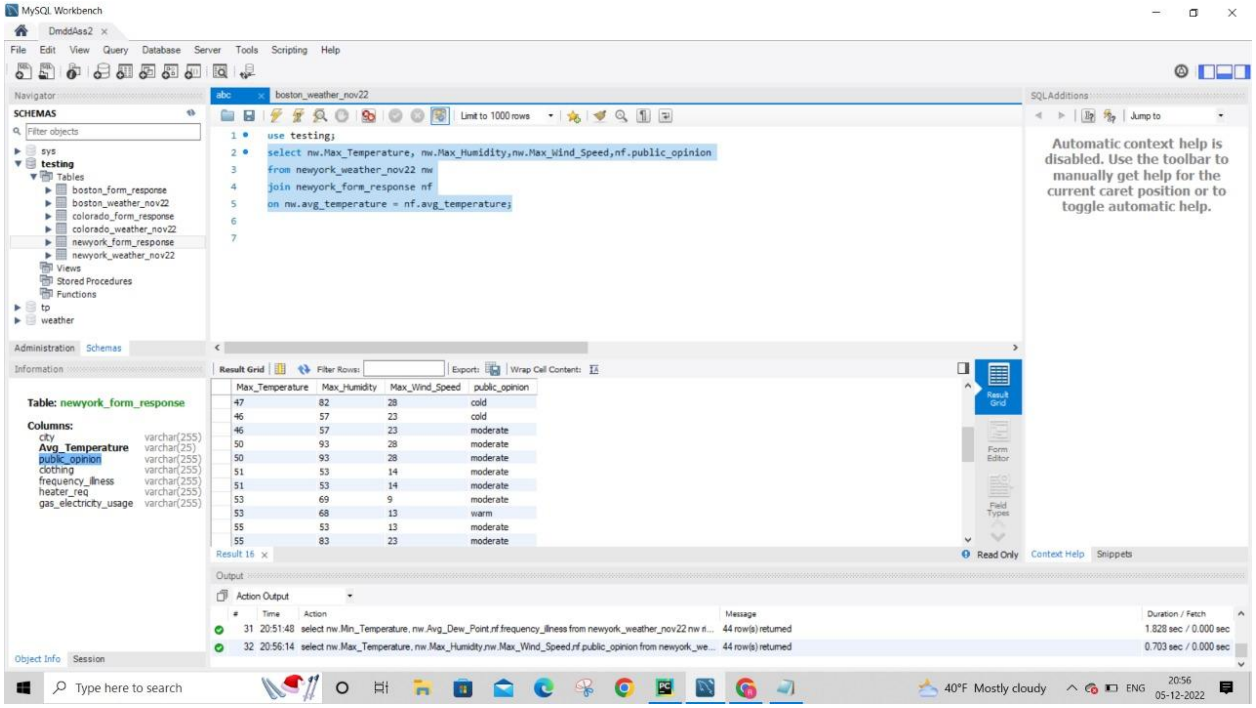

## USe Case 3: Rate of illness in NewYork due to low temperature and Dew Point



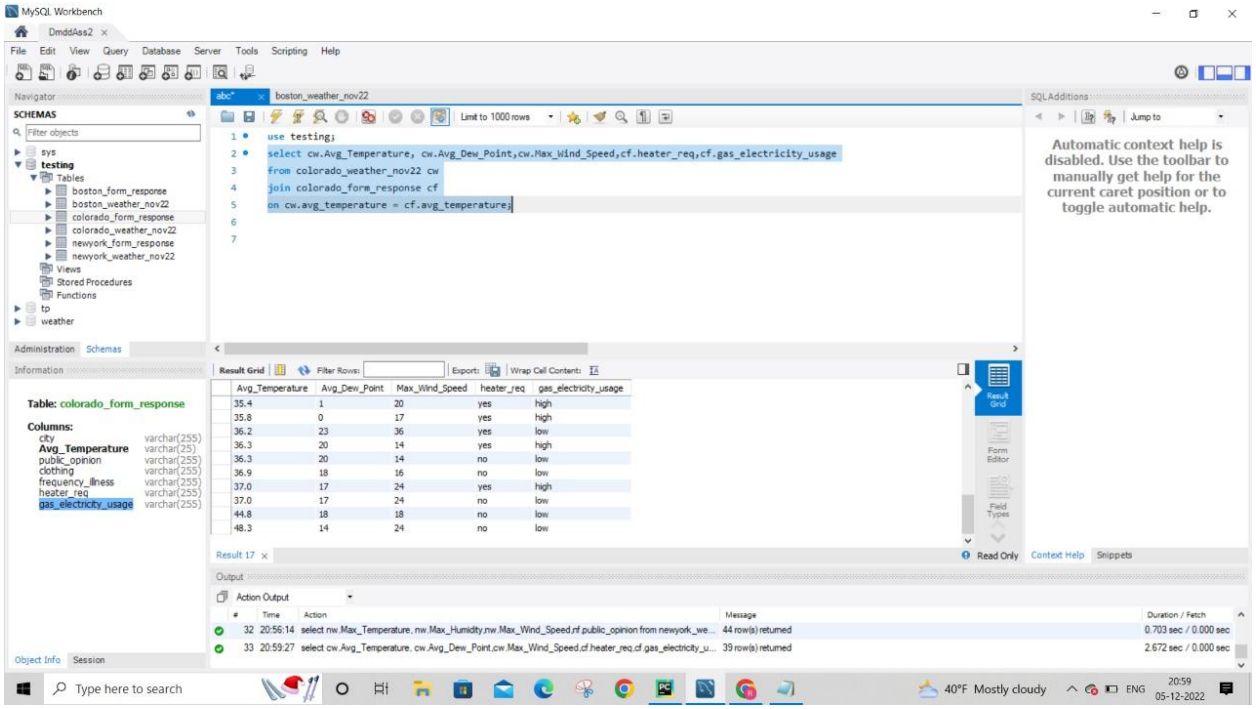

## Use Case 4 : Heater Requirement by the people of Boston according to it's weather conditions



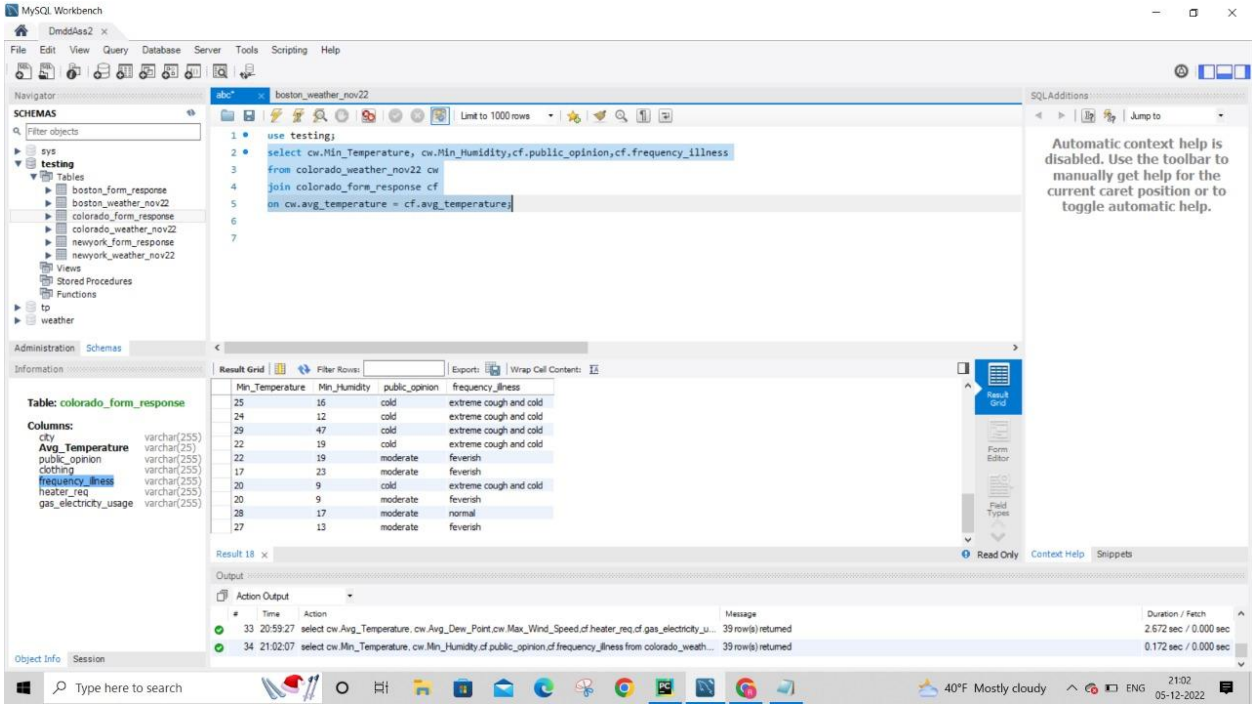
## Use Case 5: Public Opinion about the weather conditions in NewYork



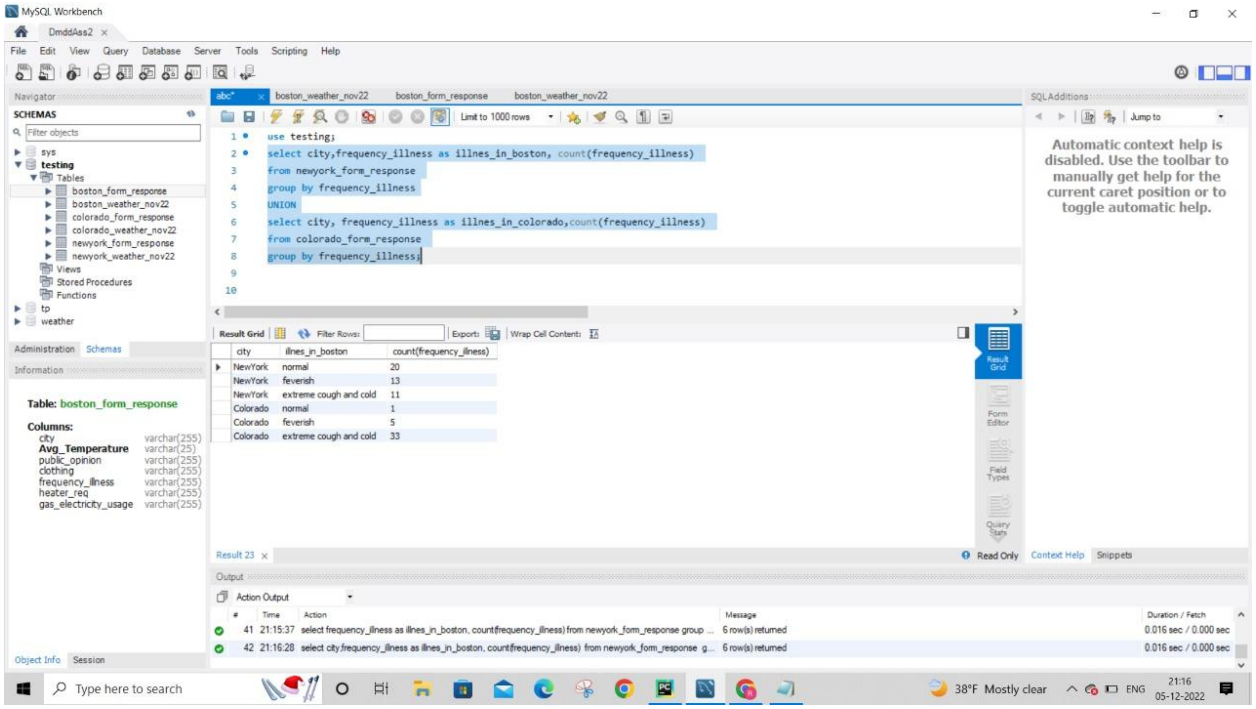

## USe Case 6: Heater, gas and electricity usage/requirement by the people of Colorado



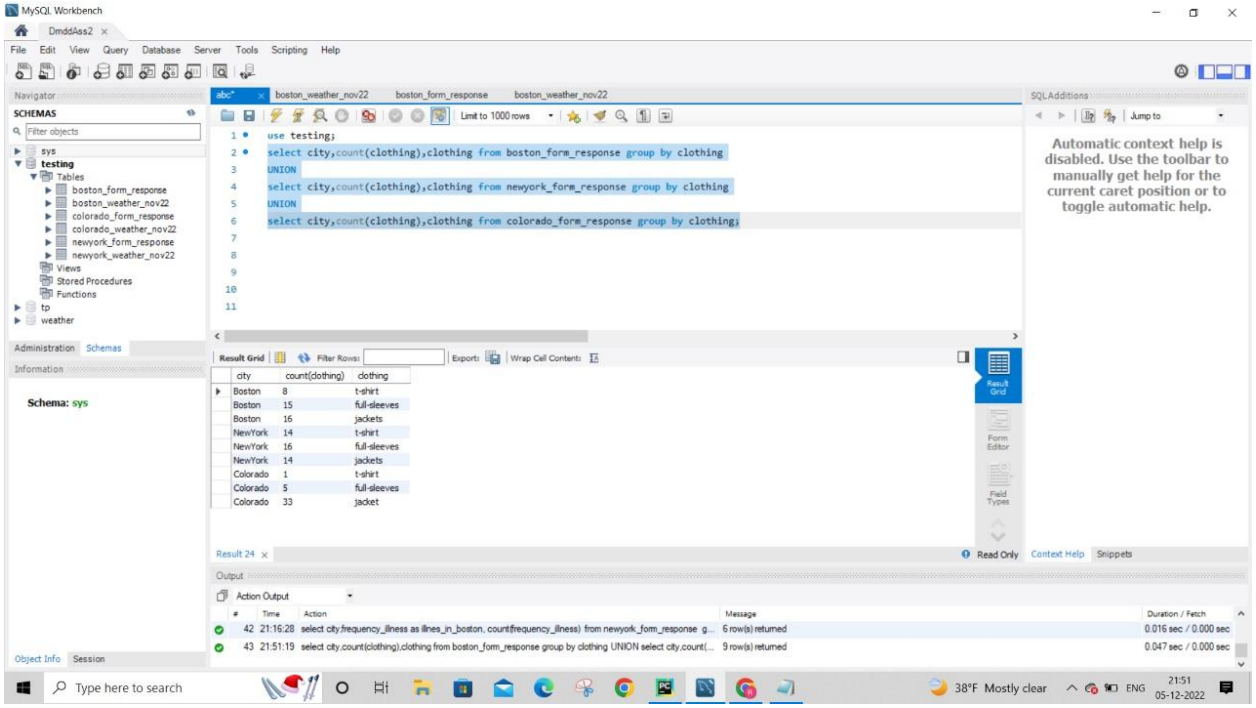
## USe Case 7: Colorado's public opinion and rate of illness during different weather conditions



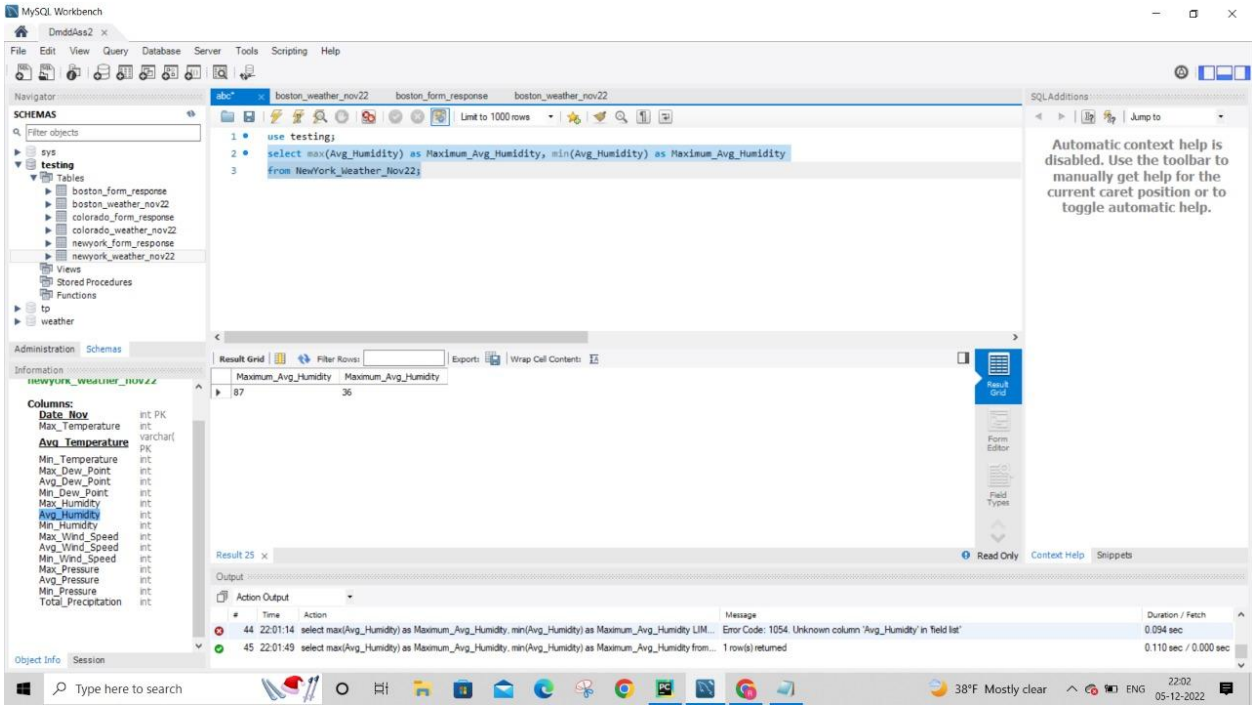

## Use Case 8: Comparison of Frequency of illness of NewYork and Colorado



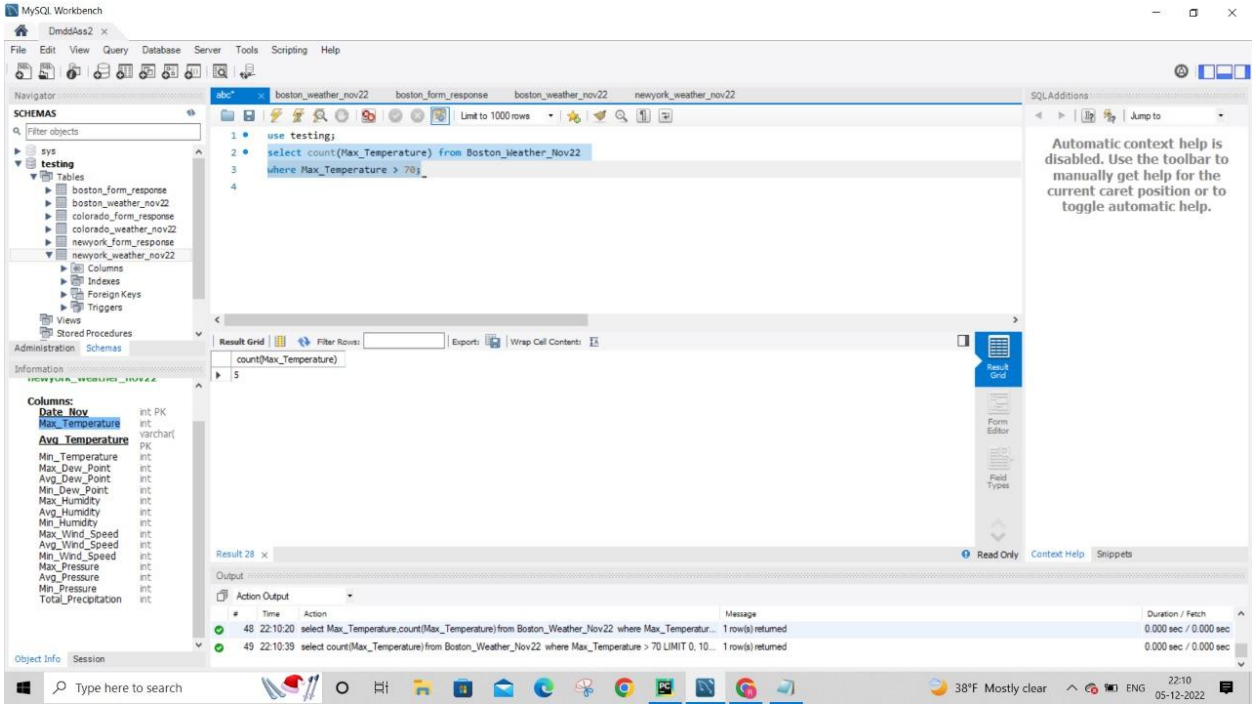
## Use Case 9: Comparison between the preferred clothing in NewYork, Boston and Colorado



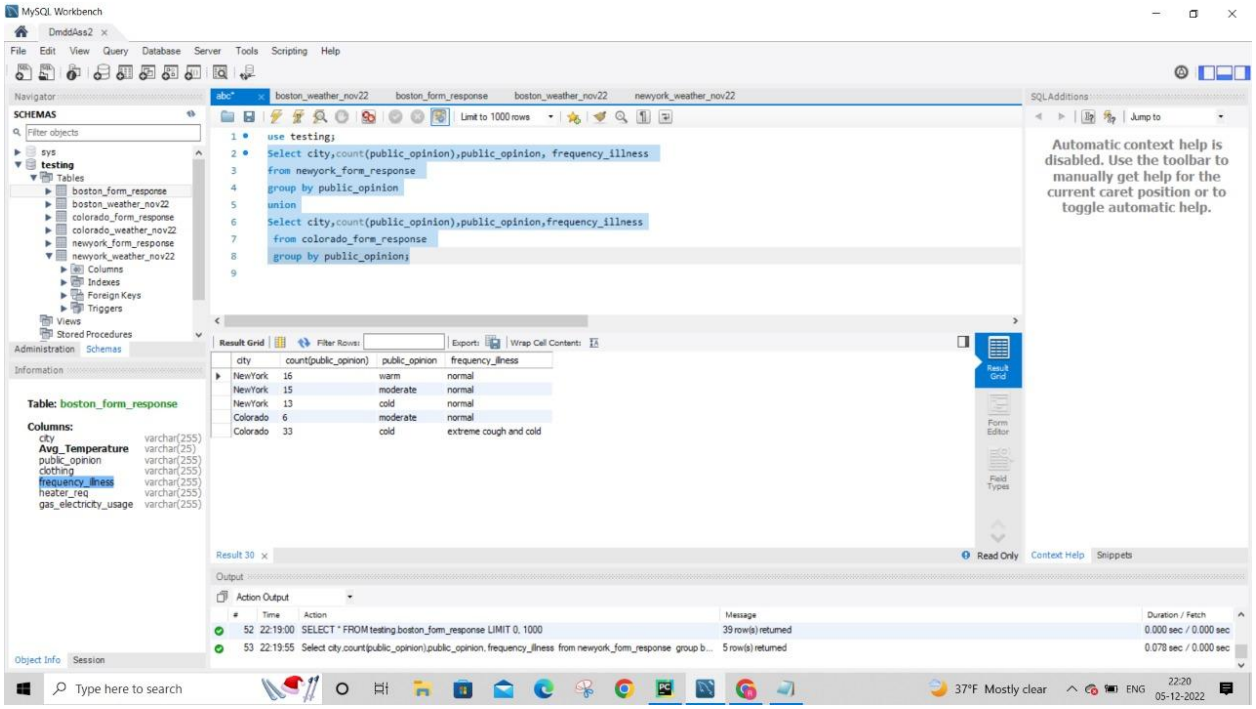

## Use Case 10: What is the Maximum and Minimum Humidity in NewYork during the month of Nov 22



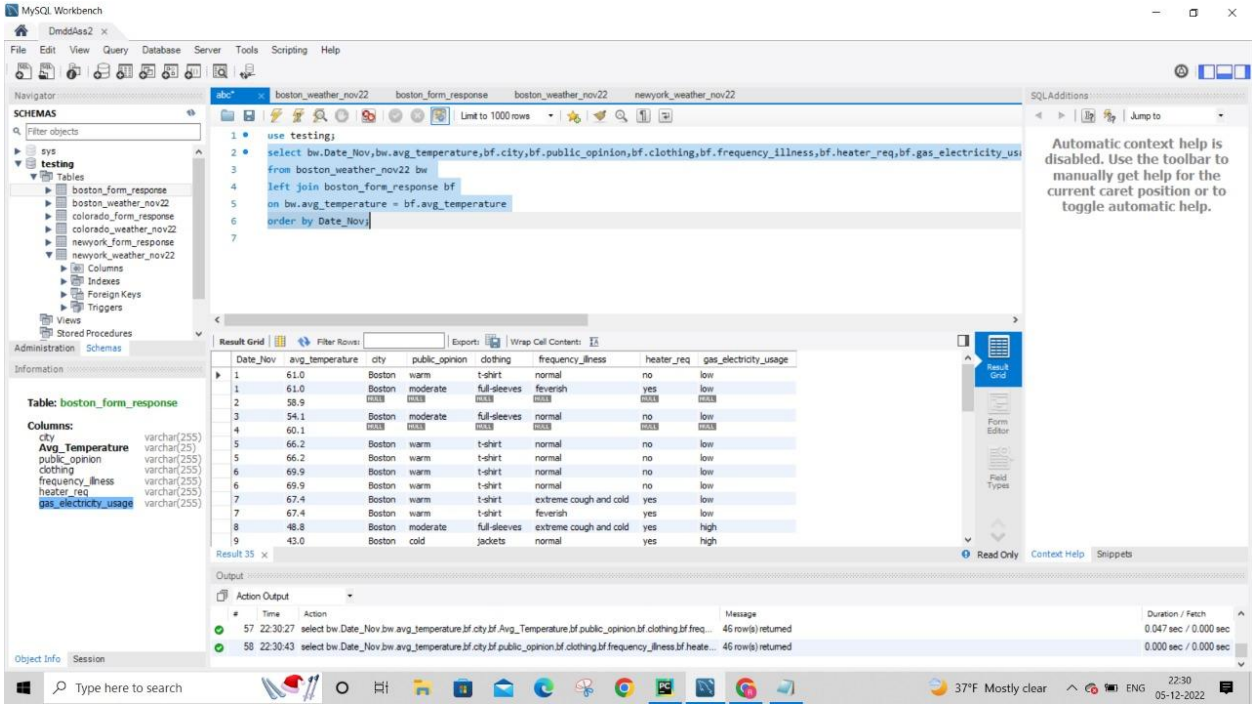
## Use Case 11: How many times Maximum Temperature was greater than 70F in Boston



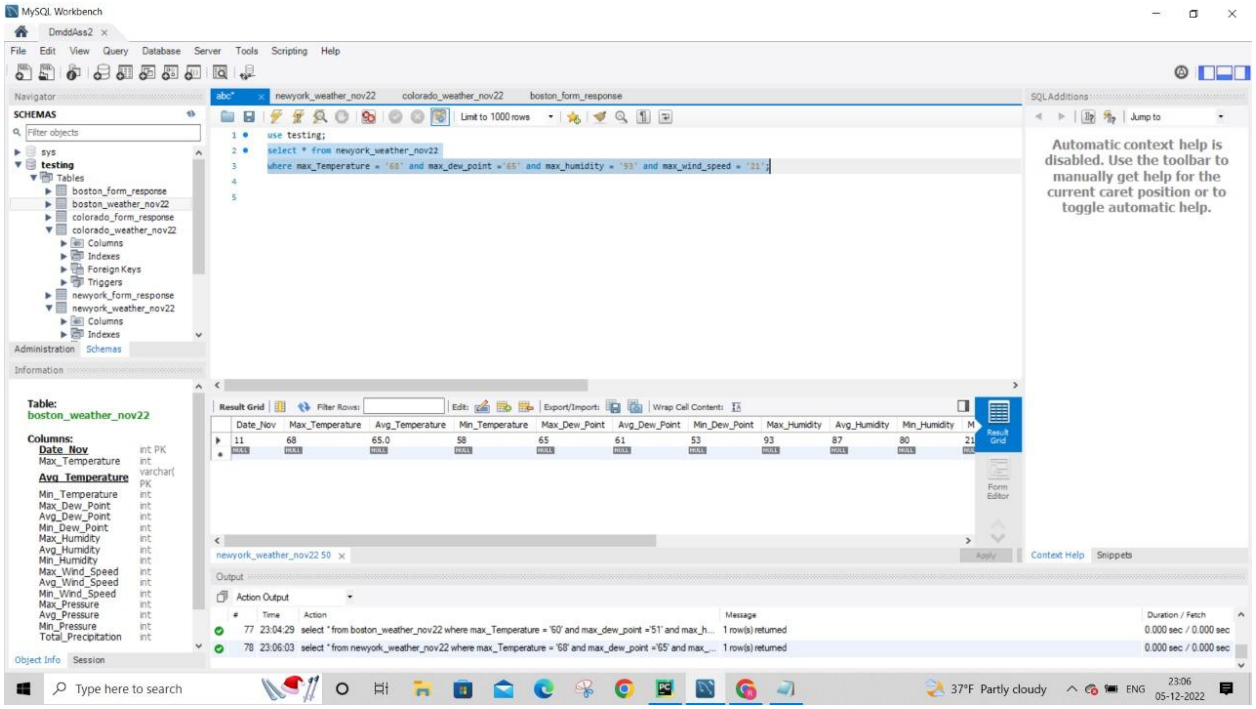

## Use Case 12: Which is the best city to visit in the month of November



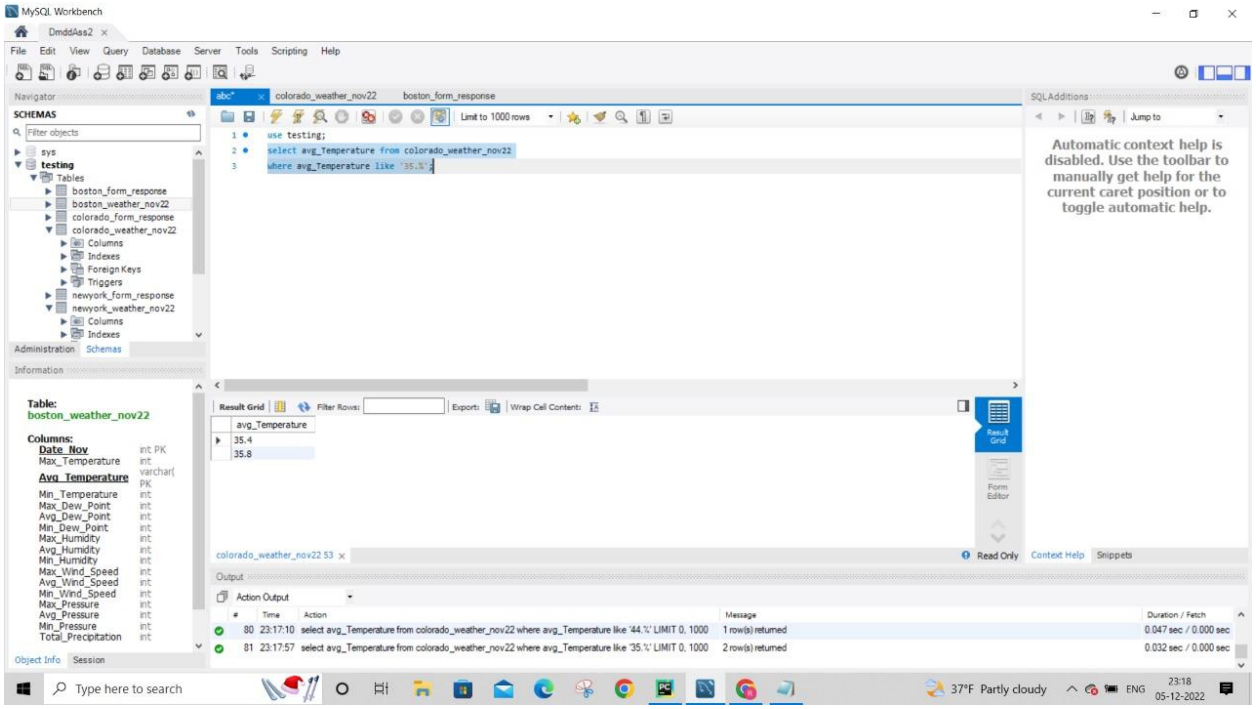
## Use Case 13: How does the Average Temperature of Boston in month of Nov affect the public




## Use Case 14: On which date did the city experience this particular weather conditions



**Use Case 15: how many times the Average Temperature of Colorado was around 33?**



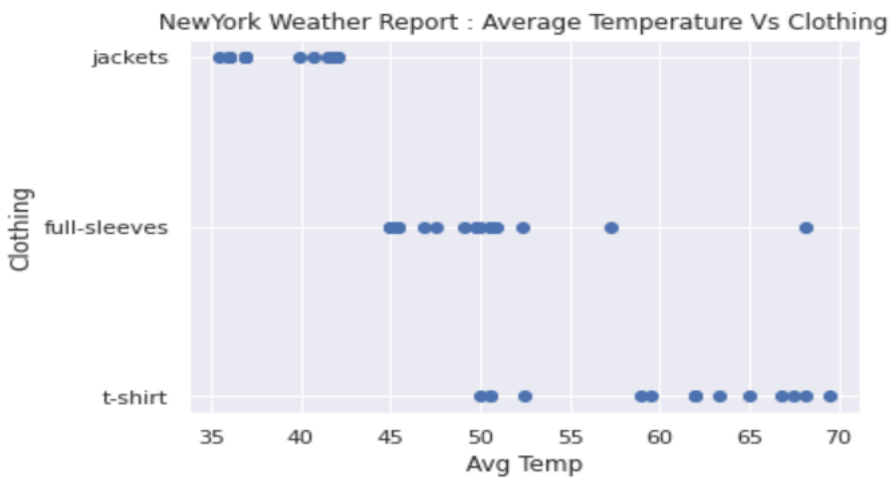

**Data Visualization and Representation:**

**Colorado URL:** [Assignment3_datavisualizationColorado.ipynb - Colaboratory (google.com)]
**New York URL:** [Assignment3_datavisualizationNewYork.ipynb - Colaboratory (google.com)]
**Boston URL:** [Copy of Assignment3_datavisualizationBoston.ipynb - Colaboratory (google.com)]

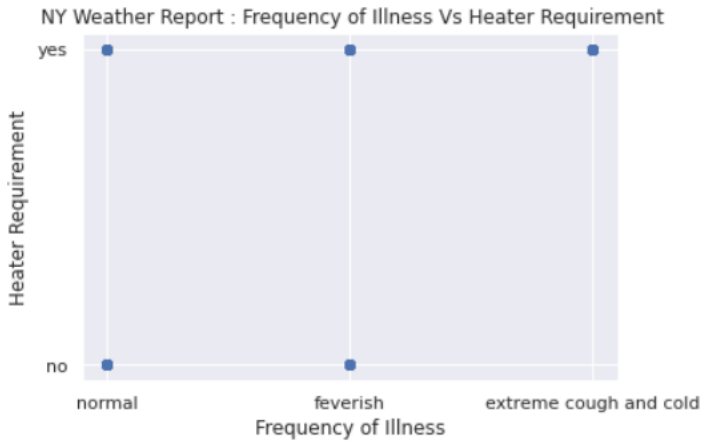
**New York Weather and Response Form Dataset Visualization:**

```
[ ]  data = pd.read_csv("/content/NewYork_form_response.csv")
```

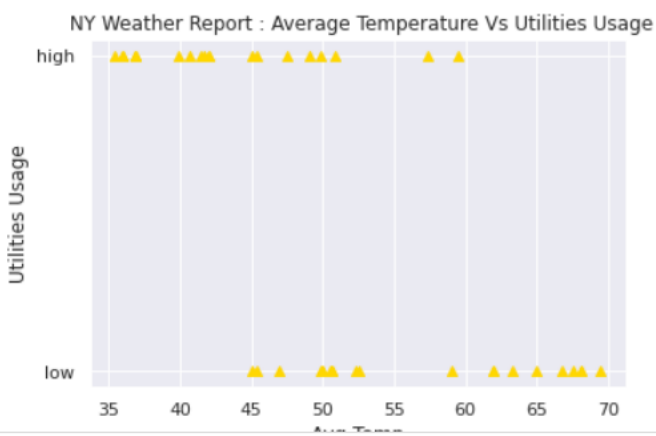
```
[ ]  import matplotlib.pyplot as plt

     plt.scatter(df.Avg_Temperature, df.clothing)
     plt.title(' NewYork Weather Report : Average Temperature Vs Clothing ')
     plt.xlabel('Avg Temp')
     plt.ylabel('Clothing')
     plt.show()
```



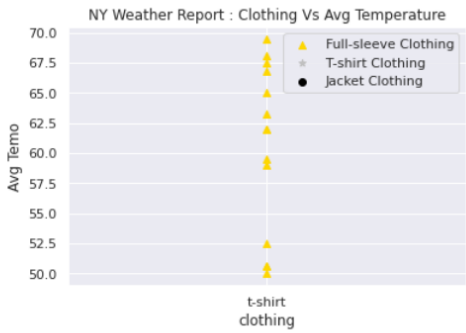
```python
import matplotlib.pyplot as plt

plt.scatter(df.frequency_illness, df.heater_req)
plt.title(' NY Weather Report : Frequency of Illness Vs Heater Requirement')
plt.xlabel('Frequency of Illness')
plt.ylabel('Heater Requirement')
plt.show()
```




```python
[ ] import matplotlib.pyplot as plt

    plt.scatter(df.Avg_Temperature, df.gas_electricity_usage, color='gold', marker= '^')
    plt.title(' NY Weather Report : Average Temperature Vs Utilities Usage')
    plt.xlabel('Avg Temp')
    plt.ylabel('Utilities Usage')
    plt.show()
```

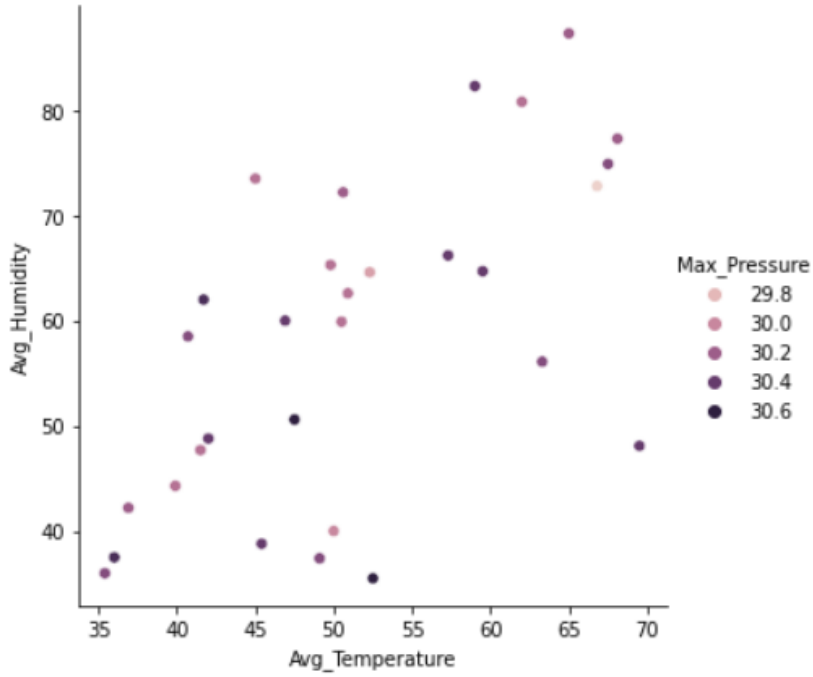


```
df_fullsleeve = df[df['clothing'] == 't-shirt']
df_tshirt = df[df['clothing'] == 'full-sleeve']
df_jacket = df[df['clothing'] == 'jacket']

plt.scatter(df_fullsleeve.clothing, df_fullsleeve.Avg_Temperature, label='Full-sleeve Clothing', color='gold', marker= '^')
plt.scatter(df_tshirt.clothing, df_tshirt.Avg_Temperature, label='T-shirt Clothing', color='silver', marker= '*')
plt.scatter(df_jacket.clothing, df_jacket.Avg_Temperature, label='Jacket Clothing', color='black', marker= 'o')
plt.title('NY Weather Report : Clothing Vs Avg Temperature')
plt.xlabel('clothing')
plt.ylabel('Avg Temo')
plt.legend()
plt.show()
```



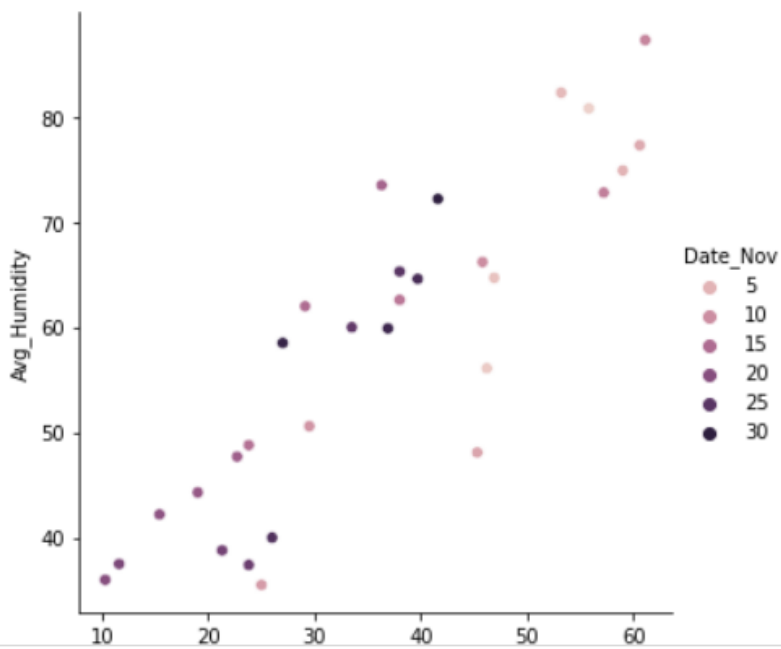
```
sns.relplot(x= 'Avg_Temperature',y='Avg_Humidity', hue='Max_Pressure',data=df)
```

```
<seaborn.axisgrid.FacetGrid at 0x7fb0c28ba040>
```



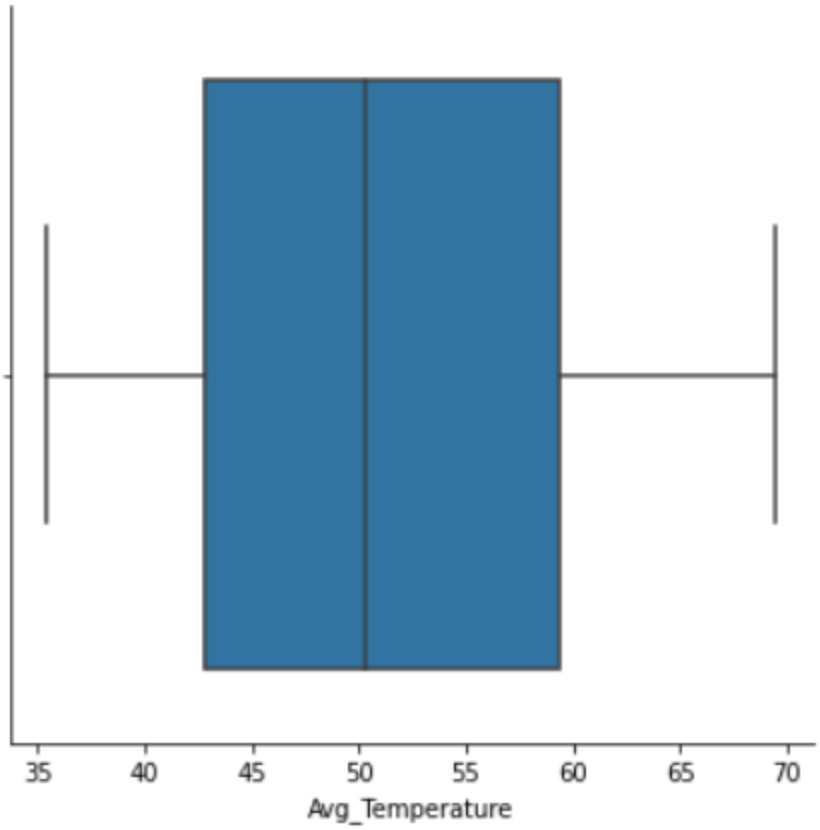

```
sns.relplot(x= 'Avg_Dew Point',y='Avg_Humidity', hue='Date_Nov',data=df)
```

```
<seaborn.axisgrid.FacetGrid at 0x7fb0b8f8eee0>
```



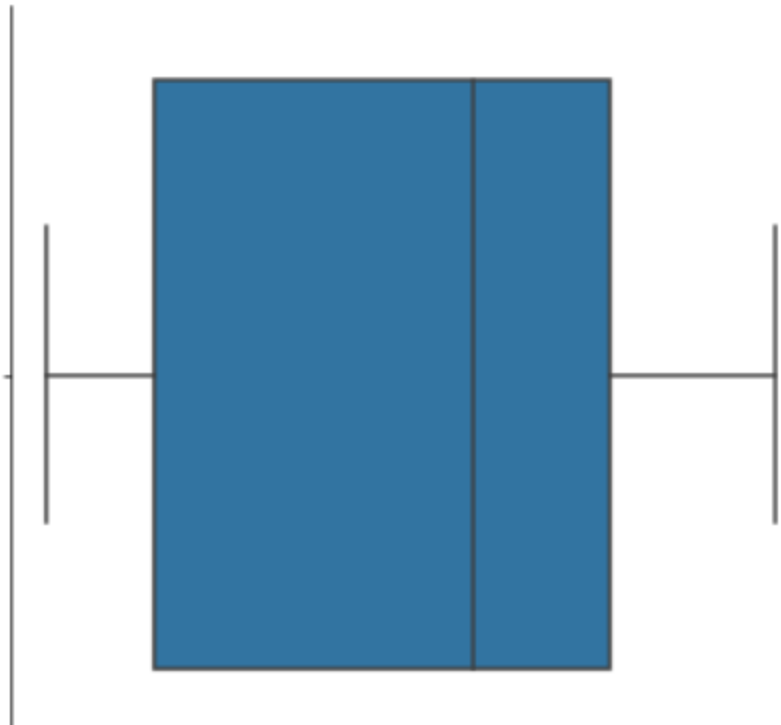
```
sns.catplot(x='Avg_Temperature', kind='box', data=df)
```

```
<seaborn.axisgrid.FacetGrid at 0x7fb0b88106d0>
```



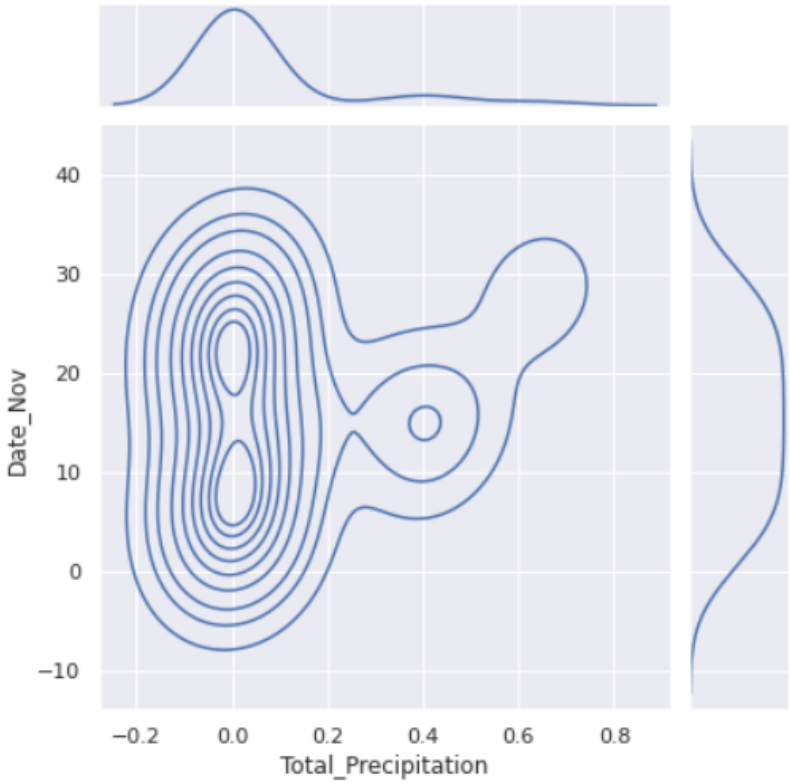

```
[ ] sns.catplot(x='Max_Humidity', kind='box', data=df)
```

```
<seaborn.axisgrid.FacetGrid at 0x7fb0b847b340>
```



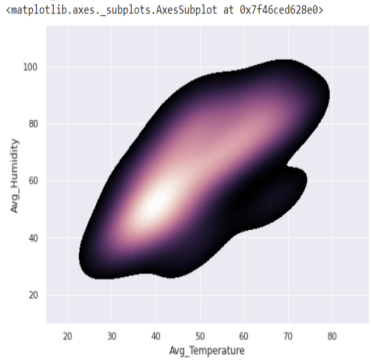
## Linear Regression:

```
import seaborn as sns
sns.set(color_codes=True)
sns.jointplot(x= "Total_Precipitation", y= "Date_Nov", data=df, kind= 'kde');
```

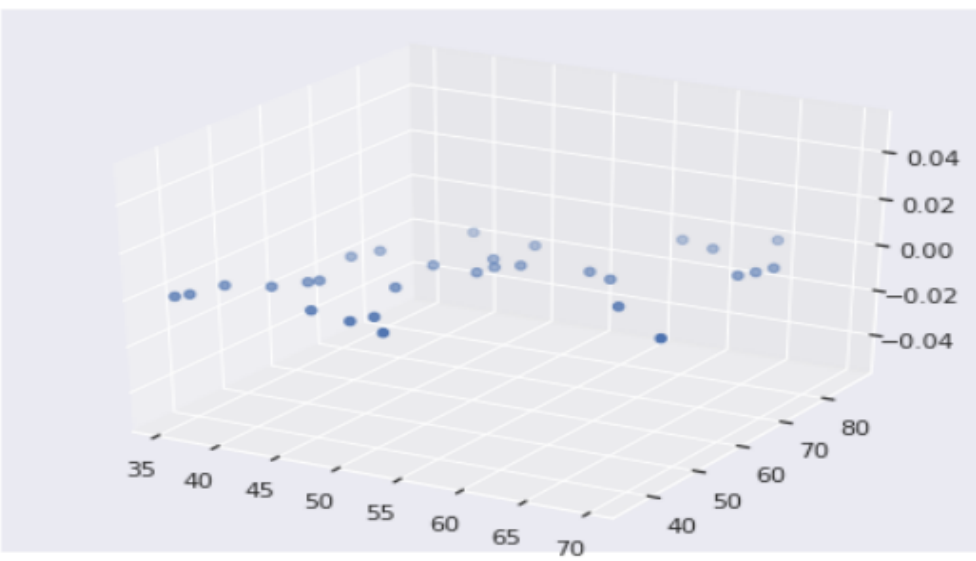



```
f, ax = plt.subplots(figsize=(8, 6))
cmap = sns.cubehelix_palette(as_cmap=True, dark=0, light=1, reverse=True)
sns.kdeplot(df.Avg_Temperature,df.Avg_Humidity, cmap=cmap, n_levels=60, shade=True)
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: y. From version 0.12, the only valid positional argument will be `data`, and passing other a
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7f46ced628e0>
```



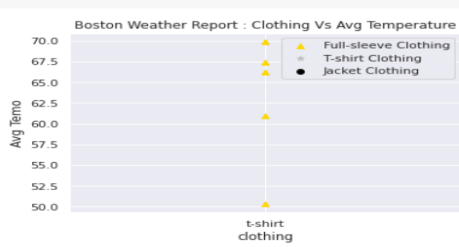
```python
from matplotlib import pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

font = {'size': 8}
plt.rc('font', **font)

fig = plt.figure()
three_d_plot = Axes3D(fig)
three_d_plot.scatter(df.Avg_Temperature,df.Avg_Humidity)
plt.show()
```



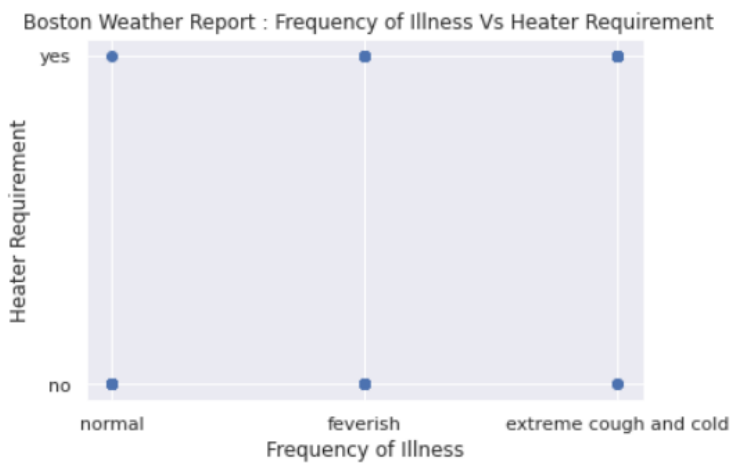

**Boston Weather and Response Form Dataset Visualization:**

**Scatter Plot:**

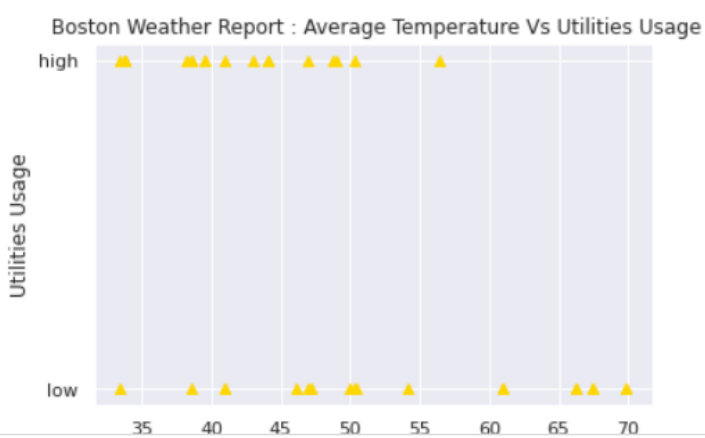
```python
df_fullsleeve = df[df['clothing'] == 't-shirt']
df_tshirt = df[df['clothing'] == 'full-sleeve']
df_jacket = df[df['clothing'] == 'jacket']

plt.scatter(df_fullsleeve.clothing, df_fullsleeve.Avg_Temperature, label='Full-sleeve Clothing', color='gold', marker= '^')
plt.scatter(df_tshirt.clothing, df_tshirt.Avg_Temperature, label='T-shirt Clothing', color='silver', marker= '*')
plt.scatter(df_jacket.clothing, df_jacket.Avg_Temperature, label='Jacket Clothing', color='black', marker= 'o')
plt.title('Boston Weather Report : Clothing Vs Avg Temperature')
plt.xlabel('clothing')
plt.ylabel('Avg Temo')
plt.legend()
plt.show()
```



```python
import matplotlib.pyplot as plt

plt.scatter(df.frequency_illness, df.heater_req)
plt.title(' Boston Weather Report : Frequency of Illness Vs Heater Requirement')
plt.xlabel('Frequency of Illness')
plt.ylabel('Heater Requirement')
plt.show()
```

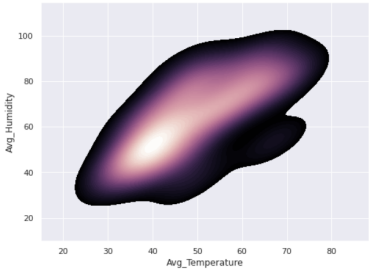



```python
[ ] import matplotlib.pyplot as plt

plt.scatter(df.Avg_Temperature, df.gas_electricity_usage, color='gold', marker= '^')
plt.title(' Boston Weather Report : Average Temperature Vs Utilities Usage')
plt.xlabel('Avg Temp')
plt.ylabel('Utilities Usage')
plt.show()
```

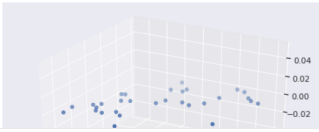


```python
f, ax = plt.subplots(figsize=(8, 6))
cmap = sns.cubehelix_palette(as_cmap=True, dark=0, light=1, reverse=True)
sns.kdeplot(df.Avg_Temperature,df.Avg_Humidity, cmap=cmap, n_levels=60, shade=True)
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: y. From version 0.12, the only valid positional argument will be `data`, and passing othe
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7f365a832880>
```

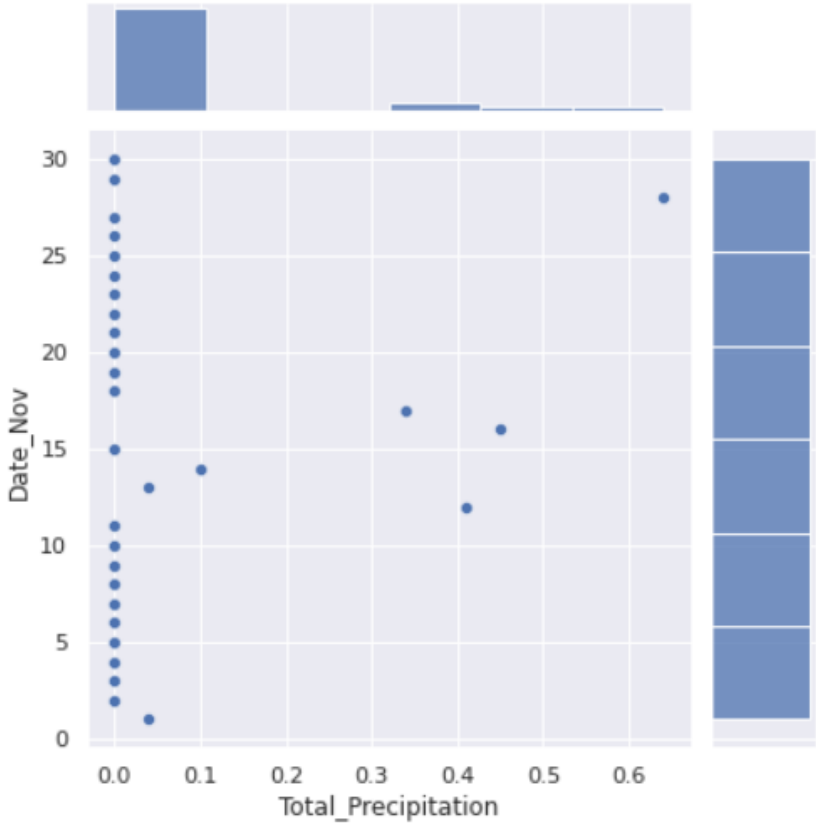



```python
from matplotlib import pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

font = {'size': 8}
plt.rc('font', **font)

fig = plt.figure()
three_d_plot = Axes3D(fig)
three_d_plot.scatter(df.Avg_Temperature,df.Avg_Humidity)
plt.show()
```

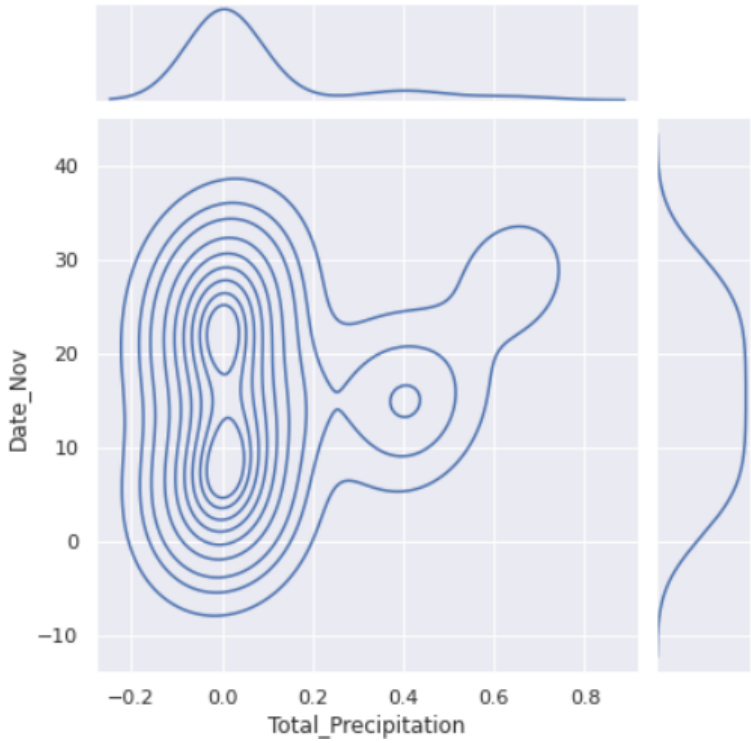


```python
import seaborn as sns
sns.set(color_codes=True)
sns.jointplot(x= "Total_Precipitation", y= "Date_Nov", data=df)
```

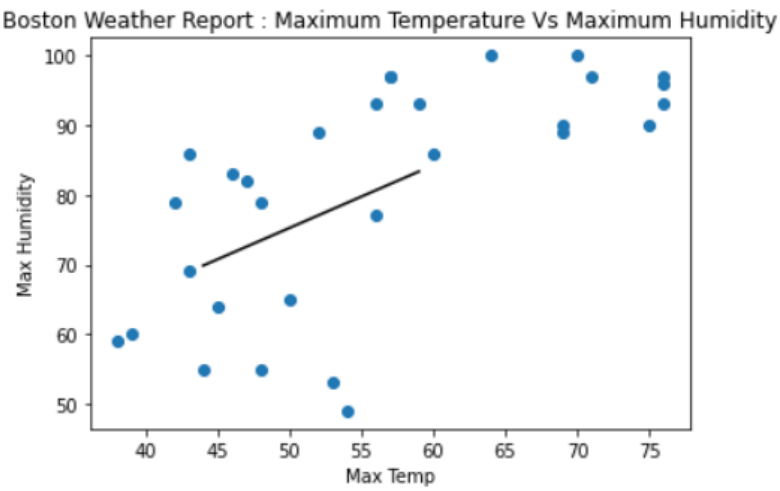
```
<seaborn.axisgrid.JointGrid at 0x7f3661fbf7f0>
```



```
[ ] import seaborn as sns
    sns.set(color_codes=True)
    sns.jointplot(x= "Total_Precipitation", y= "Date_Nov", data=df, kind= 'kde');
```



```python
# y = mx + b
# y = slope(x) + intercept

m, b, r, p, err = stats.linregress(df.Max_Temperature, df.Max_Humidity)
x = range(44, 60)
y = m * x + b
plt.plot(x, y, color='black')
plt.scatter(df.Max_Temperature, df.Max_Humidity)
plt.title('Boston Weather Report : Maximum Temperature Vs Maximum Humidity')
plt.xlabel('Max Temp')
plt.ylabel('Max Humidity')
plt.show()
```

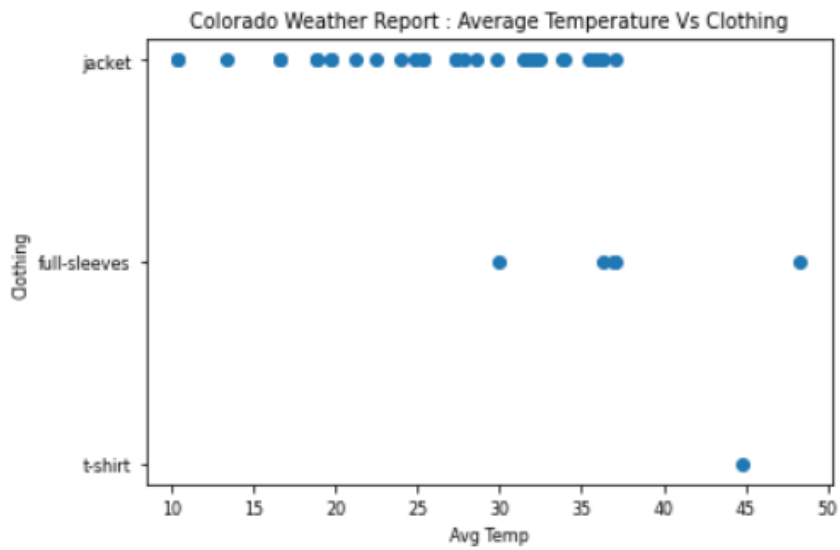



```python
[ ] from statsmodels.stats.diagnostic import het_breuschpagan
    from statsmodels.stats.diagnostic import het_white
    from statsmodels.formula.api import ols

    model = ols(formula='Max_Temperature~Max_Humidity', data=df).fit()

    white_test = het_white(model.resid, model.model.exog)
    breuschpagan_test = het_breuschpagan(model.resid, model.model.exog)

    output_df = pd.DataFrame(columns=['LM stat', 'LM p', 'F stat', 'F stat p'])
    output_df.loc['White'] = white_test
    output_df.loc['Breusch-Pagan'] = breuschpagan_test

    output_df
```

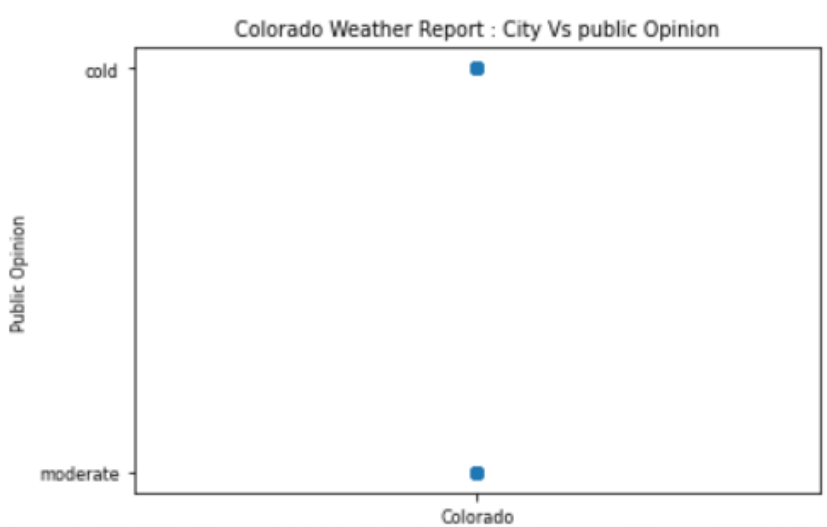
| | LM stat | LM p | F stat | F stat p |
|---|---|---|---|---|
| **White** | 0.200923 | 0.904420 | 0.091025 | 0.913274 |
| **Breusch-Pagan** | 0.192211 | 0.661083 | 0.180553 | 0.674146 |

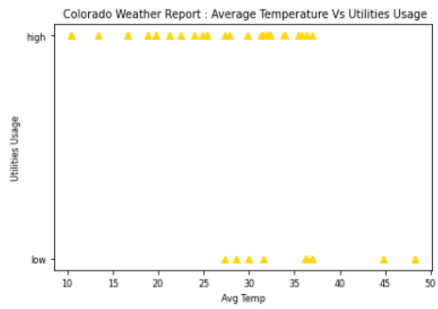
## Colorado Weather Data and Response Form : Data Visualization

```
[ ] import matplotlib.pyplot as plt

    plt.scatter(df.Avg_Temperature, df.clothing)
    plt.title(' Colorado Weather Report : Average Temperature Vs Clothing ')
    plt.xlabel('Avg Temp')
    plt.ylabel('Clothing')
    plt.show()
```

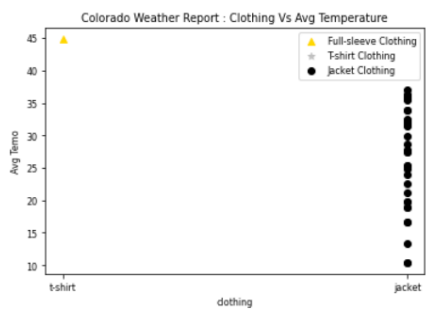



```
[ ] import matplotlib.pyplot as plt

    plt.scatter(df.City, df.public_opinion)
    plt.title(' Colorado Weather Report : City Vs public Opinion ')
    plt.xlabel('City')
    plt.ylabel('Public Opinion')
    plt.show()
```

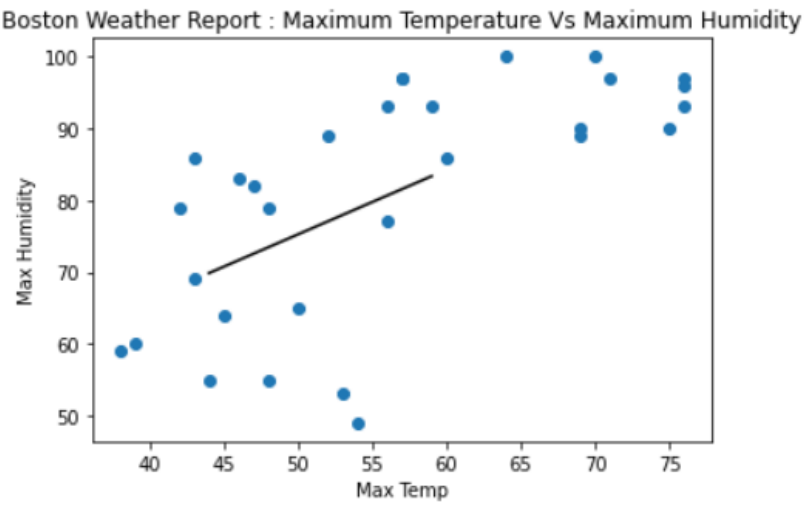




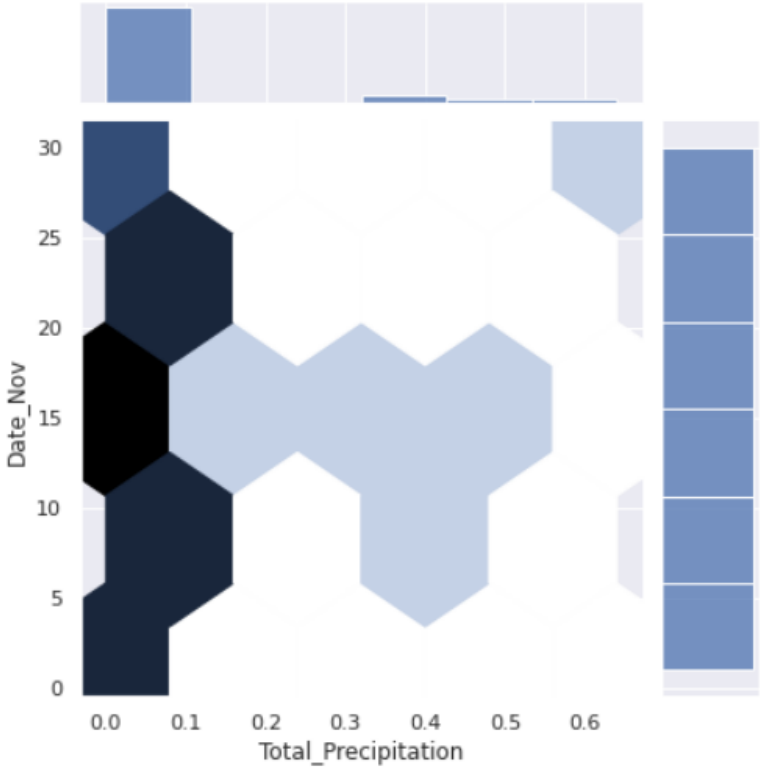

```
[ ] df_fullsleeve = df[df['clothing'] == 't-shirt']
    df_tshirt = df[df['clothing'] == 'full-sleeve']
    df_jacket = df[df['clothing'] == 'jacket']

    plt.scatter(df_fullsleeve.clothing, df_fullsleeve.Avg_Temperature, label='Full-sleeve Clothing', color='gold', marker= '^')
    plt.scatter(df_tshirt.clothing, df_tshirt.Avg_Temperature, label='T-shirt Clothing', color='silver', marker= '*')
    plt.scatter(df_jacket.clothing, df_jacket.Avg_Temperature, label='Jacket Clothing', color='black', marker= 'o')
    plt.title('Colorado Weather Report : Clothing Vs Avg Temperature')
    plt.xlabel('clothing')
    plt.ylabel('Avg Temo')
    plt.legend()
    plt.show()
```



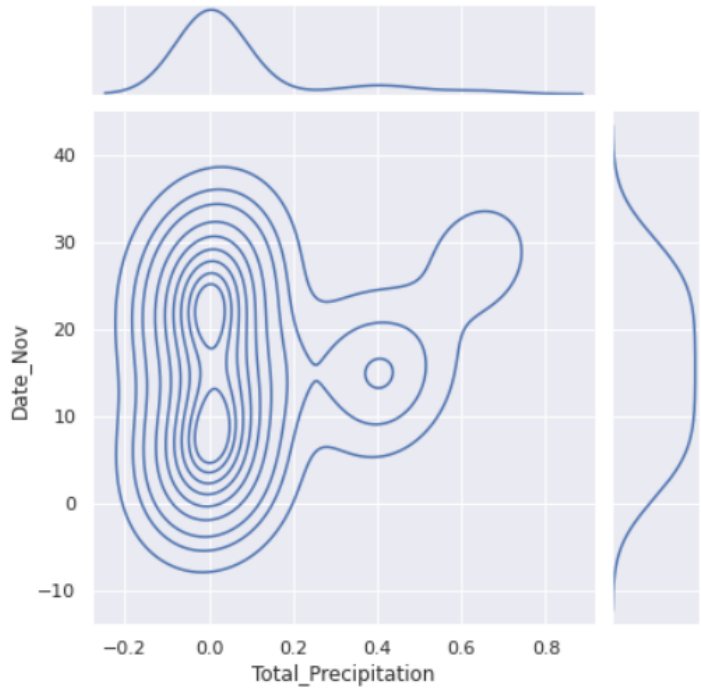
```
# y = mx + b
# y = slope(x) + intercept

m, b, r, p, err = stats.linregress(df.Max_Temperature, df.Max_Humidity)
x = range(44, 60)
y = m * x + b
plt.plot(x, y, color='black')
plt.scatter(df.Max_Temperature, df.Max_Humidity)
plt.title('Colorado Weather Report : Maximum Temperature Vs Maximum Humidity')
plt.xlabel('Max Temp')
plt.ylabel('Max Humidity')
plt.show()
```



```
[ ]  import seaborn as sns
     sns.set(color_codes=True)
     sns.jointplot(x= "Total_Precipitation", y= "Date_Nov", data=df, kind= 'hex');
```

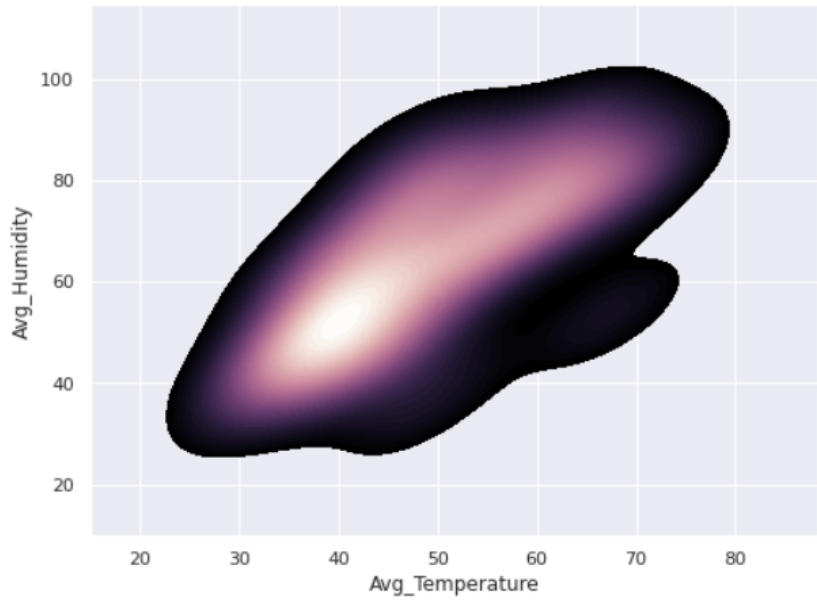


```
[ ] import seaborn as sns
    sns.set(color_codes=True)
    sns.jointplot(x= "Total_Precipitation", y= "Date_Nov", data=df, kind= 'kde');
```



```
f, ax = plt.subplots(figsize=(8, 6))
cmap = sns.cubehelix_palette(as_cmap=True, dark=0, light=1, reverse=True)
sns.kdeplot(df.Avg_Temperature,df.Avg_Humidity, cmap=cmap, n_levels=60, shade=True)
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the follow
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7f46ced628e0>
```




**Data Visualization Collabs:**

**Colorado URL:** [Assignment3_datavisualizationColorado.ipynb - Colaboratory (google.com)]
**New York URL:** [Assignment3_datavisualizationNewYork.ipynb - Colaboratory (google.com)]
**Boston URL:** [Copy of Assignment3_datavisualizationBoston.ipynb - Colaboratory (google.com)]

**GitHub URL:** [Dhiral25/Weather-Analysis (github.com)]