AMERICAN UNIVERSITY
OF MADABA
الجامعـــــة الأميركيّـــــة
في مادبـا

Develop Capabilities and Talents
Not-for-Profit University

Project: Weather data set

Ghazal Helal

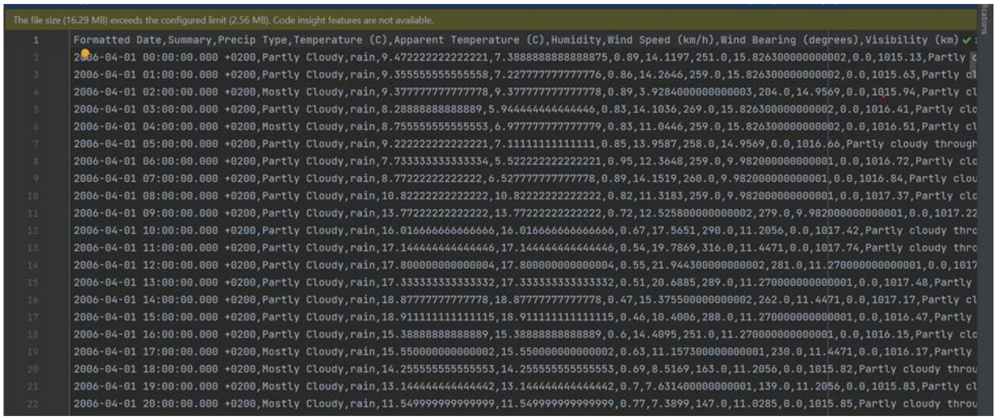
Course: Data Engineering - 402201_1

Domain: Weather data set

Abstract: weather was chosen as a domain to get data set from reliable historical sources and that give us data we dealt with in various ways in our project starting from exploring and analyzing the data in python language to build a pipeline for it and lastly bring all the processed data to database under the name weather and these methods can be applied to different data sets and different domains to make the best benefit out of it.

Weather data set: include measurements of the temperature, humidity, wind speed, and precipitation. Many other methods, such as weather stations, weather balloons, and satellites, may be used to gather the data.

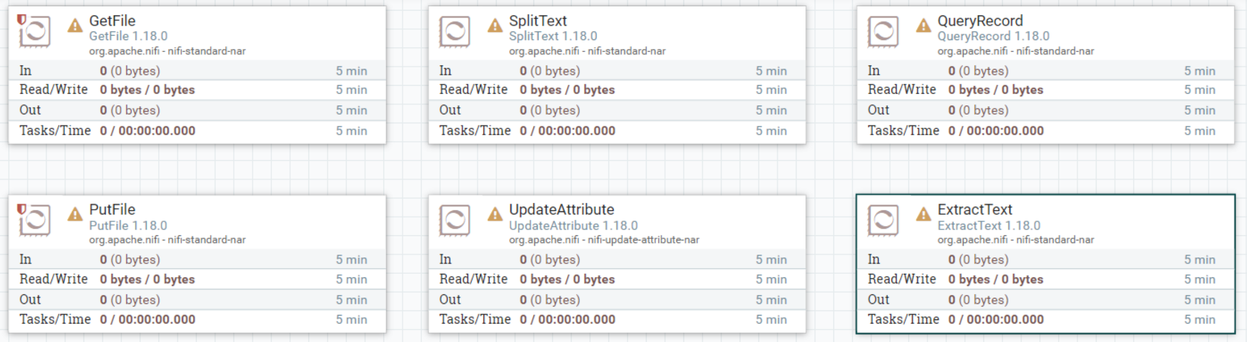
Weather.CSV (data set):



The file size (16.29 MB) exceeds the configured limit (2.56 MB). Code insight features are not available.

```
Formatted Date,Summary,Precip Type,Temperature (C),Apparent Temperature (C),Humidity,Wind Speed (km/h),Wind Bearing (degrees),Visibility (km)
2006-04-01 00:00:00.000 +0200,Partly Cloudy,rain,9.472222222222221,7.3888888888888875,0.89,14.1197,251.0,15.826300000000002,0.0,1015.13,Partly c
2006-04-01 01:00:00.000 +0200,Partly Cloudy,rain,9.355555555555558,7.227777777777776,0.86,14.2646,259.0,15.826300000000002,0.0,1015.63,Partly cl
2006-04-01 02:00:00.000 +0200,Mostly Cloudy,rain,9.377777777777778,9.377777777777778,0.89,3.9284000000000003,204.0,14.9569,0.0,1015.94,Partly cl
2006-04-01 03:00:00.000 +0200,Partly Cloudy,rain,8.288888888888889,5.944444444444446,0.83,14.1036,269.0,15.826300000000002,0.0,1016.41,Partly clo
2006-04-01 04:00:00.000 +0200,Mostly Cloudy,rain,8.755555555555553,6.977777777777779,0.83,11.0446,259.0,15.826300000000002,0.0,1016.51,Partly cl
2006-04-01 05:00:00.000 +0200,Partly Cloudy,rain,9.222222222222221,7.111111111111111,0.85,13.9587,258.0,14.9569,0.0,1016.66,Partly cloudy through
2006-04-01 06:00:00.000 +0200,Partly Cloudy,rain,7.733333333333334,5.522222222222221,0.95,12.3648,259.0,9.982000000000001,0.0,1016.72,Partly clo
2006-04-01 07:00:00.000 +0200,Partly Cloudy,rain,8.772222222222222,6.527777777777778,0.89,14.1519,260.0,9.982000000000001,0.0,1016.84,Partly clou
2006-04-01 08:00:00.000 +0200,Partly Cloudy,rain,10.822222222222222,10.822222222222222,0.82,11.3183,259.0,9.982000000000001,0.0,1017.37,Partly clo
2006-04-01 09:00:00.000 +0200,Partly Cloudy,rain,13.772222222222222,13.772222222222222,0.72,12.525800000000002,279.0,9.982000000000001,0.0,1017.22
2006-04-01 10:00:00.000 +0200,Partly Cloudy,rain,16.016666666666666,16.016666666666666,0.67,17.5651,290.0,11.2056,0.0,1017.42,Partly cloudy thro
2006-04-01 11:00:00.000 +0200,Partly Cloudy,rain,17.144444444444446,17.144444444444446,0.54,19.7869,316.0,11.4471,0.0,1017.74,Partly cloudy thro
2006-04-01 12:00:00.000 +0200,Partly Cloudy,rain,17.800000000000004,17.800000000000004,0.55,21.944300000000002,281.0,11.270000000000001,0.0,1017
2006-04-01 13:00:00.000 +0200,Partly Cloudy,rain,17.333333333333332,17.333333333333332,0.51,20.6885,289.0,11.270000000000001,0.0,1017.48,Partly
2006-04-01 14:00:00.000 +0200,Partly Cloudy,rain,18.877777777777778,18.877777777777778,0.47,15.375500000000002,262.0,11.4471,0.0,1017.17,Partly cl
2006-04-01 15:00:00.000 +0200,Partly Cloudy,rain,18.911111111111115,18.911111111111115,0.46,10.4006,288.0,11.270000000000001,0.0,1016.47,Partly
2006-04-01 16:00:00.000 +0200,Partly Cloudy,rain,15.388888888888889,15.388888888888889,0.6,14.4095,251.0,11.270000000000001,0.0,1016.15,Partly clo
2006-04-01 17:00:00.000 +0200,Mostly Cloudy,rain,15.550000000000002,15.550000000000002,0.63,11.157300000000001,230.0,11.4471,0.0,1016.17,Partly
2006-04-01 18:00:00.000 +0200,Mostly Cloudy,rain,14.255555555555553,14.255555555555553,0.69,8.5169,163.0,11.2056,0.0,1015.82,Partly cloudy throu
2006-04-01 19:00:00.000 +0200,Mostly Cloudy,rain,13.144444444444442,13.144444444444442,0.7,7.631400000000001,139.0,11.2056,0.0,1015.83,Partly cl
2006-04-01 20:00:00.000 +0200,Mostly Cloudy,rain,11.549999999999999,11.549999999999999,0.77,7.3899,147.0,11.0285,0.0,1015.85,Partly cloudy throu
```

1) Building a data pipeline using Apache Nifi:



GetFile
GetFile 1.18.0
org.apache.nifi - nifi-standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

SplitText
SplitText 1.18.0
org.apache.nifi - nifi-standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

QueryRecord
QueryRecord 1.18.0
org.apache.nifi - nifi-standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

PutFile
PutFile 1.18.0
org.apache.nifi - nifi-standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

UpdateAttribute
UpdateAttribute 1.18.0
org.apache.nifi - nifi-update-attribute-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

ExtractText
ExtractText 1.18.0
org.apache.nifi - nifi-standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

A) **Get file:** A "GetFile" processor should be added to a NiFi flow once it has been created. The CSV file containing the weather data set will be retrieved from a particular place on your computer using this processor.

B) **SplitText:** A "SplitText" processor should be connected to the "GetFile" processor. To allow for independent processing of each line, the CSV file will be divided into separate lines using the "SplitText" processor.

C) **QueryRecord:** To query a record in a CSV weather data set in Apache NiFi, you can use the QueryRecord processor. This processor allows you to apply an SQL-like query to a record and output the resulting records to a specified relationship.

D) **Extract text:** With the help of this processor, you may extract particular text patterns from a FlowFile's contents and output those patterns to a particular relationship.

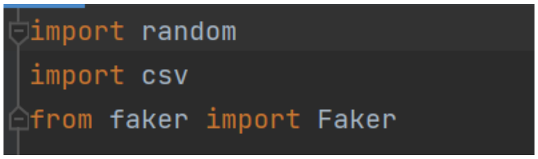
E) **UpdateAttribute:** This processor enables you to change the value of an existing attribute as well as set, delete, or remove attributes from a FlowFile.

F) **Putfile:** Using the routing capability, the PutFile processor may also be set up to write data to various directories dependent on the timestamp or location of the flowfile, for example.

2) Generate data from a defined data source with 5,000 rows and 10-columns.

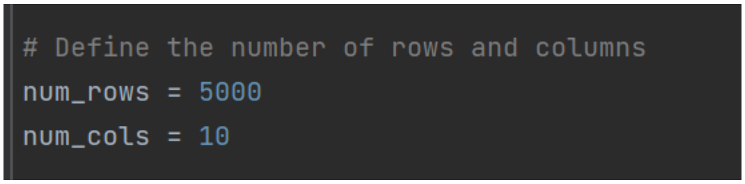
Using several libraries and modules, you may create data in Python from a given data source with 5,000 rows and 10 columns. Here is an illustration of data generation utilizing the integrated random and csv modules:

Using several libraries and modules, you may create data in Python from a given data source with 5,000 rows and 10 columns. Here is an illustration of data generation utilizing the integrated random and csv modules:



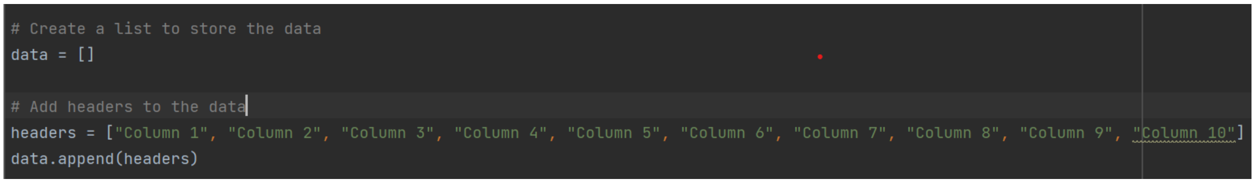
```python
import random
import csv
from faker import Faker
```

A) Import the random and csv modules at the top of your Python script.



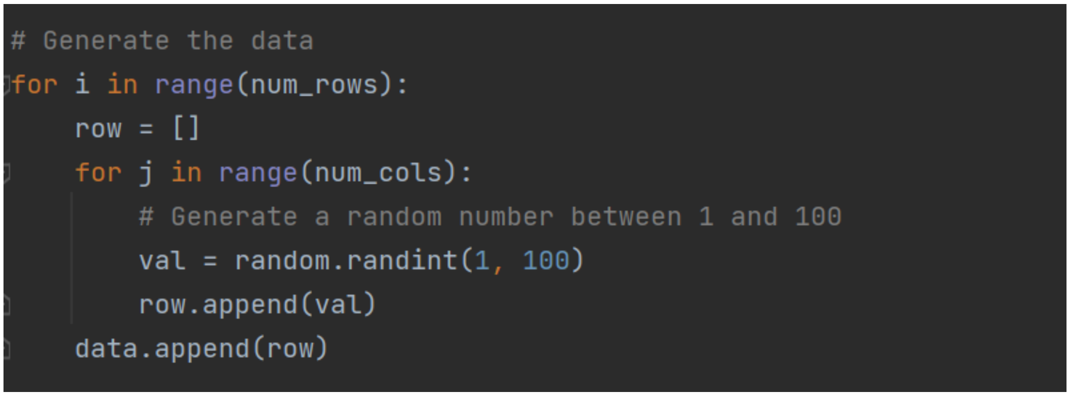
```python
# Define the number of rows and columns
num_rows = 5000
num_cols = 10
```

B) Define the number of rows and columns you want to generate.



```
# Create a list to store the data
data = []

# Add headers to the data
headers = ["Column 1", "Column 2", "Column 3", "Column 4", "Column 5", "Column 6", "Column 7", "Column 8", "Column 9", "Column 10"]
data.append(headers)
```

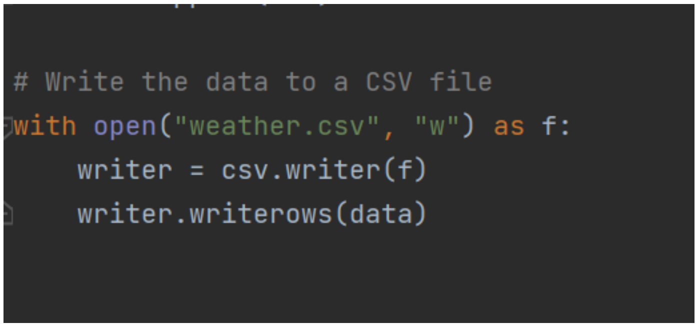
C) Create a list to store the data and add the headers to the list.



```
# Generate the data
for i in range(num_rows):
    row = []
    for j in range(num_cols):
        # Generate a random number between 1 and 100
        val = random.randint(1, 100)
        row.append(val)
    data.append(row)
```

D) Create the data using nested for loops. Both the outer and inner loops will repeat over the amount of rows and columns, respectively. To create random integers inside a specified range, use the random.randint() method. To the current row, add each number that is created.

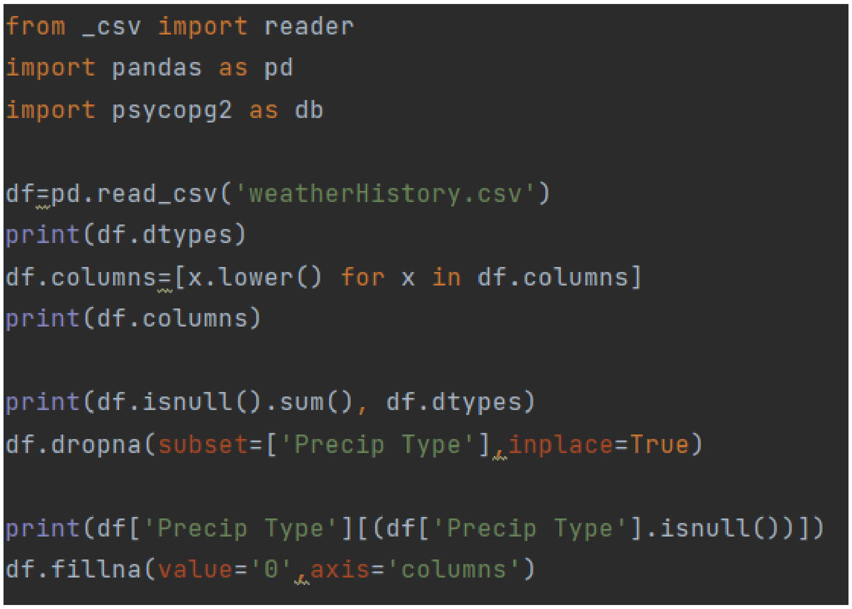
E) Write data to CSV file.



```
# Write the data to a CSV file
with open("weather.csv", "w") as f:
    writer = csv.writer(f)
    writer.writerows(data)
```

3) Analyzing the data set:

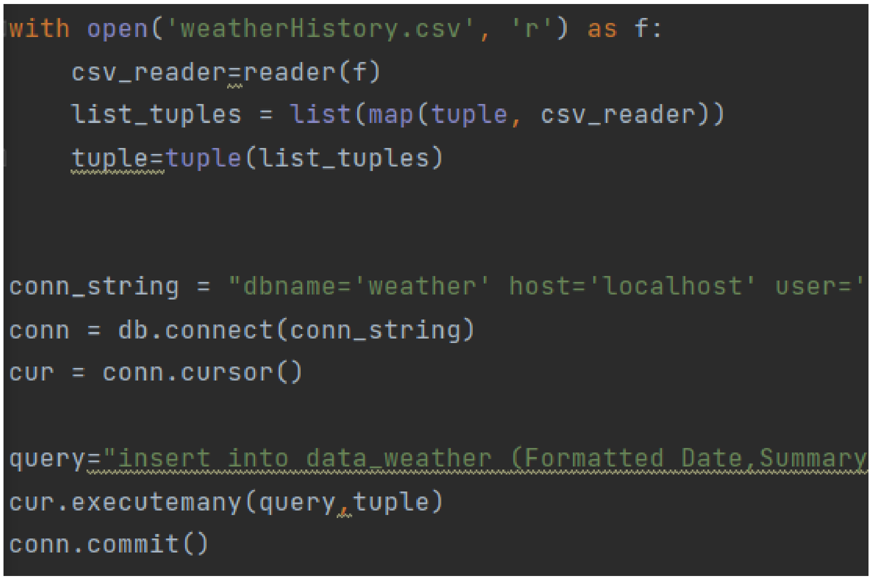
- We had the data set and then we applied various functions to explore the data like info(), dtypes() , columns() and more
- We analyzed the data for any odd or any null values. And found some null values in the column precip
- We used the fillna method to replace the null values by 0.



```
from _csv import reader
import pandas as pd
import psycopg2 as db

df=pd.read_csv('weatherHistory.csv')
print(df.dtypes)
df.columns=[x.lower() for x in df.columns]
print(df.columns)

print(df.isnull().sum(), df.dtypes)
df.dropna(subset=['Precip Type'],inplace=True)

print(df['Precip Type'][(df['Precip Type'].isnull())])
df.fillna(value='0',axis='columns')
```

4) The PgAdmin 4 database

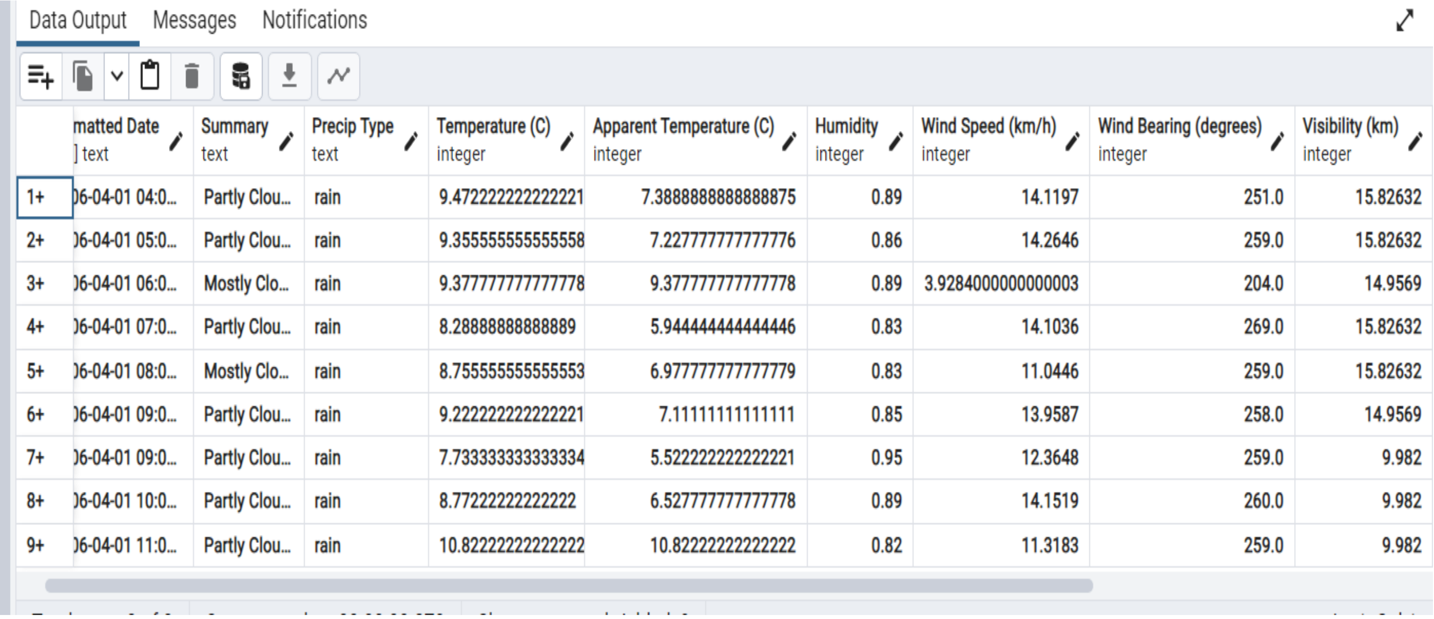
- First we establish connection with the database in the IDE as we did in the data engineering course.
- Next we created a cursor.
- And then we ceated a tuple of tuples to send our data to the table data_waether in the database weather.



```
with open('weatherHistory.csv', 'r') as f:
    csv_reader=reader(f)
    list_tuples = list(map(tuple, csv_reader))
    tuple=tuple(list_tuples)


conn_string = "dbname='weather' host='localhost' user='
conn = db.connect(conn_string)
cur = conn.cursor()

query="insert into data_weather (Formatted Date,Summary
cur.executemany(query,tuple)
conn.commit()
```

5) Results in the database

Here we have sample of the data in the pgAdmin that we sent from the commands in python :



	matted Date] text	Summary text	Precip Type text	Temperature (C) integer	Apparent Temperature (C) integer	Humidity integer	Wind Speed (km/h) integer	Wind Bearing (degrees) integer	Visibility (km) integer
1+	)6-04-01 04:0...	Partly Clou...	rain	9.472222222222221	7.388888888888875	0.89	14.1197	251.0	15.82632
2+	)6-04-01 05:0...	Partly Clou...	rain	9.355555555555558	7.227777777777776	0.86	14.2646	259.0	15.82632
3+	)6-04-01 06:0...	Mostly Clo...	rain	9.377777777777778	9.377777777777778	0.89	3.9284000000000003	204.0	14.9569
4+	)6-04-01 07:0...	Partly Clou...	rain	8.288888888888889	5.944444444444446	0.83	14.1036	269.0	15.82632
5+	)6-04-01 08:0...	Mostly Clo...	rain	8.755555555555553	6.977777777777779	0.83	11.0446	259.0	15.82632
6+	)6-04-01 09:0...	Partly Clou...	rain	9.222222222222221	7.111111111111111	0.85	13.9587	258.0	14.9569
7+	)6-04-01 09:0...	Partly Clou...	rain	7.733333333333334	5.522222222222221	0.95	12.3648	259.0	9.982
8+	)6-04-01 10:0...	Partly Clou...	rain	8.772222222222222	6.527777777777778	0.89	14.1519	260.0	9.982
9+	)6-04-01 11:0...	Partly Clou...	rain	10.822222222222222	10.822222222222222	0.82	11.3183	259.0	9.982

Lastly after analyzing and editing the data in different ways and also projected in many formats in the csv file and the database now we can use the data to make predictions based on the data set we have to the weather at any given time of the day if we have enough and proper info.