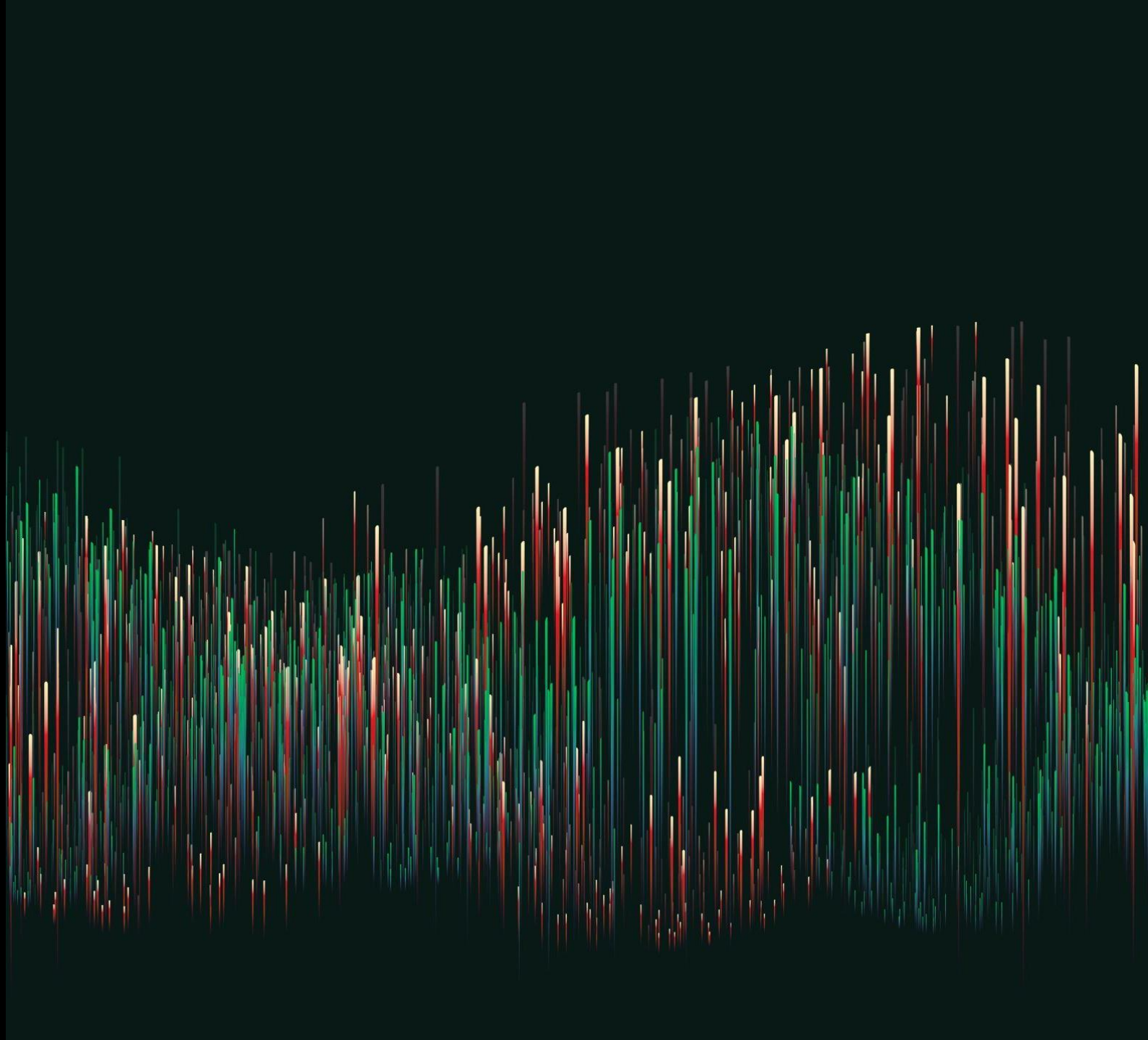
# CEIS110 Programming with Data Final Project

Present by Hector Acosta



# Introduction

**Background**

- Data generated from various sources is growing exponentially, creating a need for efficient storage, management, analysis, and security measures.
- This project focuses on data analysis using Python, divided into six parts. It involves analyzing weather data obtained from a cloud source, stored in a database, and processed using programming and data analytics tools.

**Scenario**

- Develop a software system to download weather data from a cloud source, store it in a database, and extract desired subsets for analysis.
- Utilize programming and data analytics to analyze the data, create visualizations, and make predictions.
- The project uses weather data based on my zip code.

# Software Inventory

**Software**

•**Excel:** Office 365 applications.

•**Python (Replit):** An online integrated development environment (IDE) for Python, providing a platform to write, compile, and run Python code. Replit includes built-in packages and libraries for various development tasks.

**Description of Replit:**

•Replit is a powerful online coding platform designed for collaborative programming and learning. It provides a full-featured IDE with capabilities to write, test, and deploy Python code. Replit also offers integrated support for version control, package management, and project sharing, making it ideal for educational and professional development projects.

# Planning and Design

**Objectives**

Use a flowchart as a design tool.

Develop a flowchart for the course project.

Generate a prototype design of a software system.

# Importance of Flowcharts

## Flowcharts

- Graphically explain a system using shapes, boxes, circles, and connecting arrows.
- Represent data flow, including inputs, outputs, and storage points.
- Commonly used to represent programs, algorithms, or any ordered process.

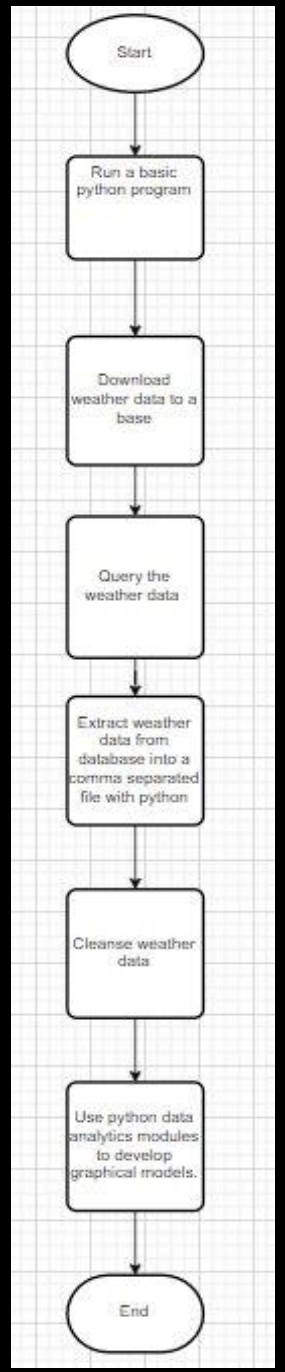
## Why Flowcharts Are Important

- Simplifies understanding of the process.
- Helps identify potential issues before implementation.
- Provides a clear roadmap for programming and debugging.

# Flowcharts

- The process for the flowchart for this project:

  a. Run a basic python program
  b. Download weather data to a database.
  c. Query the weather data
  d. Extract weather data from database into a comma separated file with python
  e. Cleanse weather data
  f. Use python data analytics modules to develop graphical models



# Phyton Program

- Write a basic Python program.

**Becoming Familiar with Python**

- Write a Python program to ask for user input (name, city, temperature).
- Display a message based on the temperature input.

Module2Project.py › ...

```python
#Module 2 Project
#Hector Acosta 5/9/2024
# Inputs
name = input("What is your name? ")
city = input("What city do you live in? ")
temperature = float(input("What is the temperature? "))
#Process and Outputs
if temperature > 70:
    print("Hello", name, "it is nice where you live")
else:
    print("Hello", name, "it is too cold where you live")
```

Run

```
--- Which file would you like to run? (Enter
0 - EXIT
1 - Main.py
2 - myownlearning.py
3 - Church.py
4 - week2practice.py
5 - Module2Project.py
Enter Option: 5
--- Running Module2Project.py ---
What is your name? Hector
What city do you live in? Vineland
What is the temperature? 61
Hello Hector it is too cold where you live
```

# Downloading Weather Data

**Objectives:**

- Practice writing and executing Python programs
- Learn to download data from the cloud using an API
- Create a relational database

**Overview:**

- Utilize the US National Oceanic and Atmospheric Administration (NOAA) API for weather data.
- Develop a Python program to download and store weather observations in a local database.
- Analyze and prepare weather data for later use.

# Python Code for Downloading Weather Data

**Python Code Overview**

- The Python code connects to the NOAA API, retrieves weather data, and stores it in a SQLite database.
- Key sections include setting up API parameters, connecting to the database, creating the table, and inserting data.
- Build WeatherDb.py Code

Module3Project.py › ... Format

```python
#Purpose: Build weather database from NOAA data
#Name: Hector Acosta
#Date: 5/13/24
#   See https://pypi.org/project/noaa-sdk/ for details on noaa_sdk package used

from noaa_sdk import noaa
import sqlite3
import datetime

# parameters for retrieving NOAA weather data
zipCode = "08361"  # change to your postal code
country = "US"
```

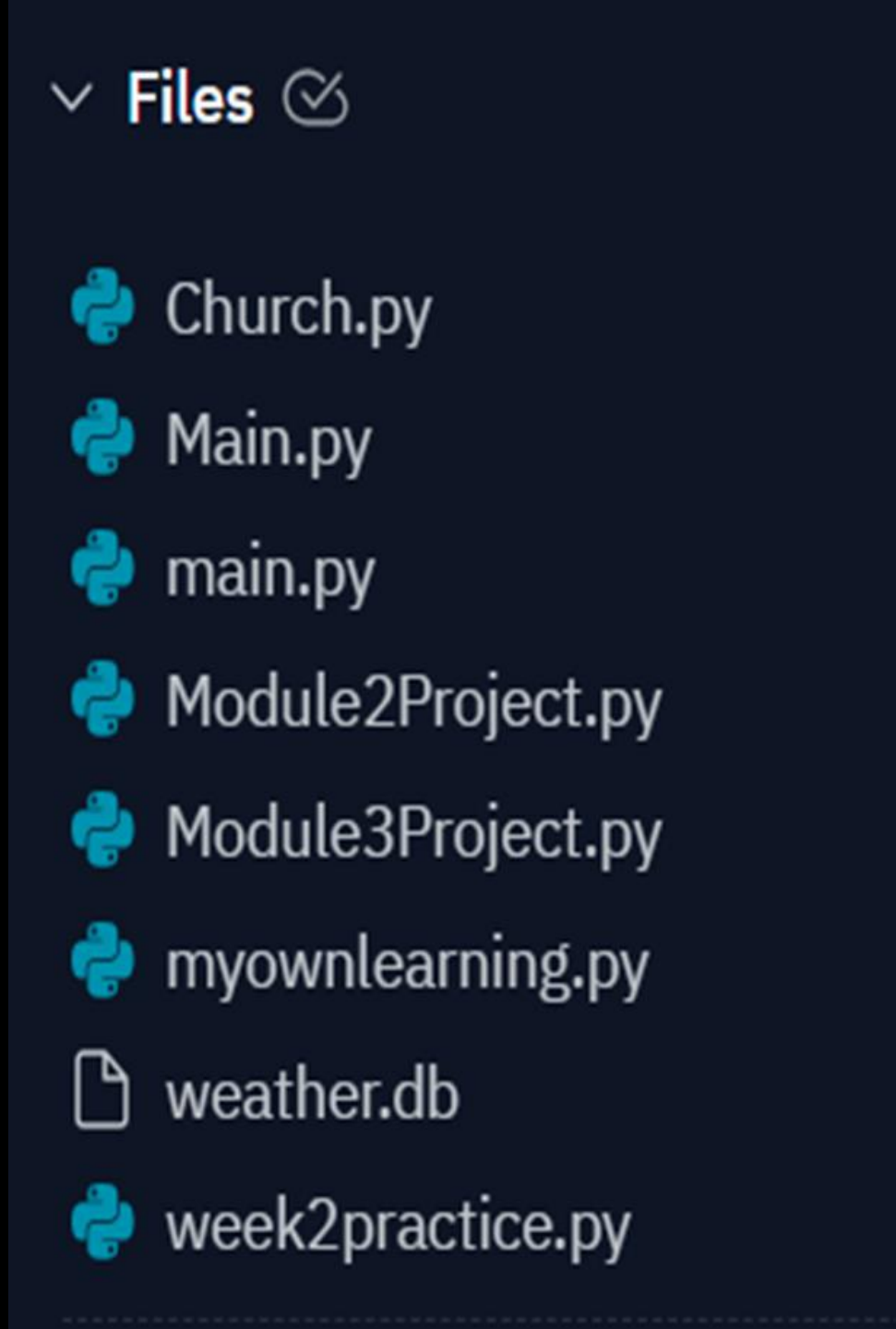
# Program Execution

- After running the program, the console output confirms the successful retrieval and insertion of weather data into the database.
- The number of rows inserted may vary depending on the data available.

```
--- Which file would you like to run? (En
0 - EXIT
1 - Main.py
2 - myownlearning.py
3 - Church.py
4 - week2practice.py
5 - Module2Project.py
6 - Module3Project.py
Enter Option: 6
--- Running Module3Project.py ---
Preparing database...
Database prepared
Getting weather data...
Inserting rows...
212 rows inserted
Database load complete!
```

# Verifying Database Creation

- The next step is to verify the weather.db file has been created in the project folder.
- This file contains the weather data fetched and stored by the program.



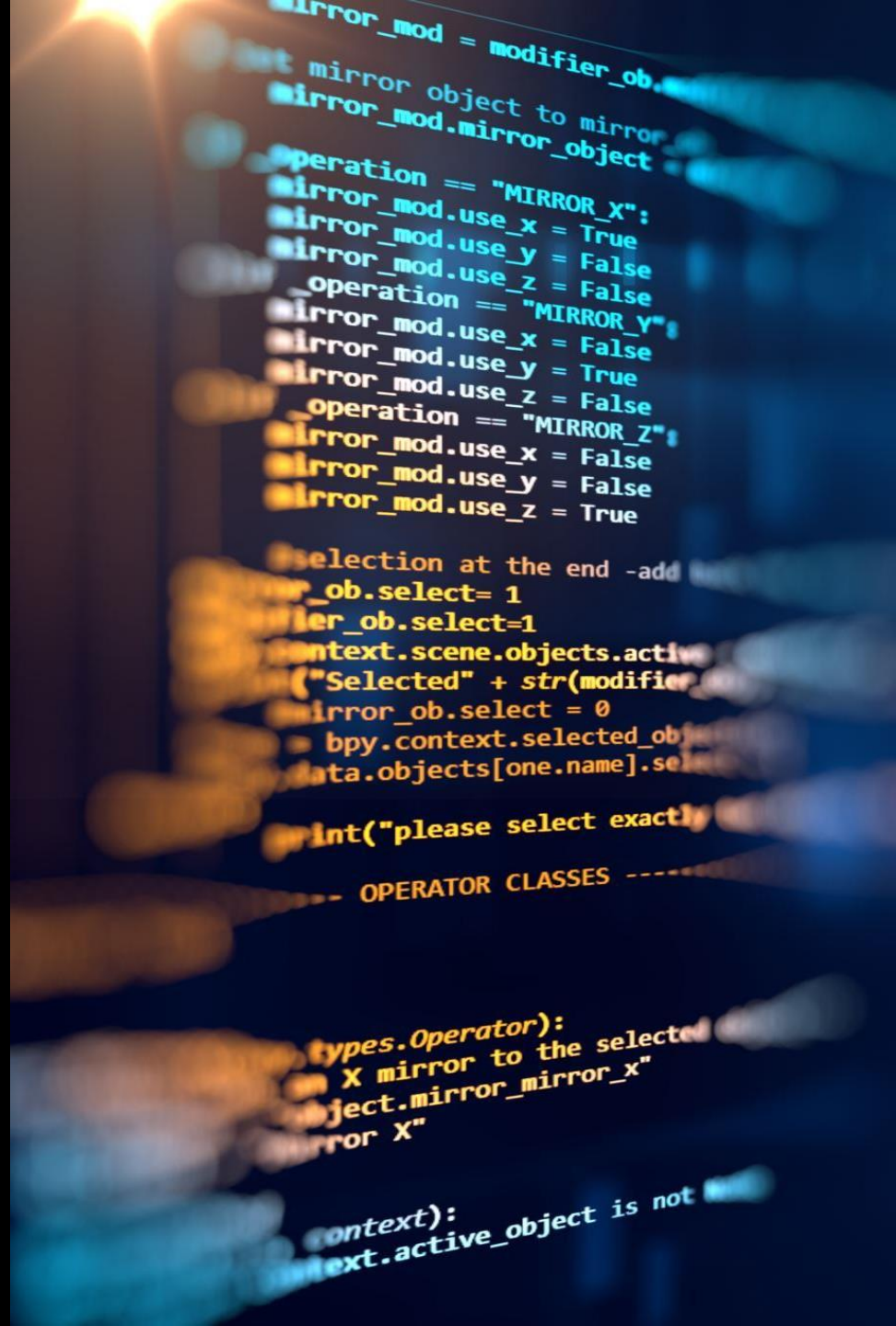
# Introduction to Querying the Database with SQL

**Objectives:**

- Connect to a database using Python
- Execute simple SQL SELECT queries
- View data retrieved from the database using queries

**Overview:**

- SQL (Structured Query Language) is used for interacting with relational databases.
- Python programs can issue SQL commands to databases and display the results to users.
- This stage of the project focuses on using Python to query the database and retrieve weather data.



# Query to Retrieve All Columns and All Rows

The first query retrieves all rows and columns from the observations table.

This is achieved using the SQL command: SELECT * FROM observations;.

The result provides a comprehensive view of all the data stored in the table.

Module4Project.py > ...

```python
#Purpose: Query database using SQL
#Name: Hector Acosta
#Date: 5/21/24
#   Run BuildWeatherDB.py to build weather database before running this program

import sqlite3
import pandas as pd


#file names for database and output file
dbFile = "weather.db"

#format output
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
pd.set_option('display.max_colwidth', None)
pd.set_option('display.expand_frame_repr', False)

#connect to and query weather database
conn = sqlite3.connect(dbFile)
#Create SQL command
selectCmd = " SELECT * FROM observations ORDER BY timestamp; "


#print out the query
result = pd.read_sql_query(selectCmd, conn)
print(result)
```

Console  Shell  Module4Project.py  +

Run  Ask AI  11s on 21:24:04, 05/21

```
Enter Option: 7
--- Running Module4Project.py ---
                  timestamp  windSpeed  temperature  relativeHumidity  windDirection  barometricPressure  visibility          textDescription
0    2024-05-07T00:54:00+00:00       0.00         20.0         93.389863            0.0              101250       12870           Mostly Cloudy
1    2024-05-07T01:54:00+00:00       0.00         18.9         93.336012            0.0              101220       11270           Partly Cloudy
2    2024-05-07T03:09:00+00:00       0.00         17.8         96.284527            0.0              101150        4830                Fog/Mist
3    2024-05-07T03:54:00+00:00       0.00         17.2         96.881098            0.0              101120        6440
4    2024-05-07T04:54:00+00:00       0.00         17.2         96.881098            0.0              101080        4830                Fog/Mist
5    2024-05-07T05:37:00+00:00       0.00         17.2         96.881098            0.0              101020        4020                Fog/Mist
6    2024-05-07T05:54:00+00:00       0.00         17.2         96.881098            0.0              101020        4020                Fog/Mist
7    2024-05-07T06:15:00+00:00       0.00         16.7         96.253553            0.0              101020        3220                Fog/Mist
8    2024-05-07T06:24:00+00:00       0.00         17.2         96.881098            0.0              101020        4830                Fog/Mist
9    2024-05-07T06:54:00+00:00       0.00         17.2         96.881098            0.0              100980        4020                Fog/Mist
10   2024-05-07T07:54:00+00:00       0.00         17.2         96.881098            0.0              100950        4020                Fog/Mist
11   2024-05-07T08:44:00+00:00       0.00         17.2         96.881098            0.0              100950        4020                Fog/Mist
12   2024-05-07T09:32:00+00:00       0.00         17.8         93.281507            0.0              100920        4020                Fog/Mist
13   2024-05-07T10:01:00+00:00       0.00         16.7         96.253553            0.0              100950        1210                Fog/Mist
14   2024-05-07T10:20:00+00:00       0.00         16.7        100.000000            0.0              100920        2010                Fog/Mist
15   2024-05-07T10:54:00+00:00       0.00         17.2         96.881098            0.0              100950        2820                Fog/Mist
16   2024-05-07T11:28:00+00:00       0.00         18.3         93.306364            0.0              100950        3220                Fog/Mist
17   2024-05-07T11:54:00+00:00       0.00         18.3         93.306364            0.0              100950        4020                Fog/Mist
18   2024-05-07T12:18:00+00:00       0.00         19.4         90.472847            0.0              100980        6440                Fog/Mist
19   2024-05-07T12:54:00+00:00      11.16         20.6         78.351777           20.0              100980       12870
20   2024-05-07T13:54:00+00:00       7.56         22.2         68.375390           90.0              100980       16090
21   2024-05-07T14:45:00+00:00       0.00         22.8         63.854514            0.0              100980       16090           Mostly Cloudy
22   2024-05-07T14:54:00+00:00       0.00         23.3         61.951861            0.0              100980       16090
23   2024-05-07T15:54:00+00:00       5.40         24.4         57.986289            NaN              100920       16090                  Cloudy
24   2024-05-07T16:30:00+00:00       0.00         25.0         51.787727            0.0              100850       16090                  Cloudy
25   2024-05-07T16:54:00+00:00       7.56         25.0         53.830642          150.0              100810       16090           Mostly Cloudy
26   2024-05-07T17:54:00+00:00       5.40         25.0         57.760700            NaN              100780       16090                  Cloudy
27   2024-05-07T18:54:00+00:00       7.56         25.0         51.787727            NaN              100750       16090                  Cloudy
28   2024-05-07T19:54:00+00:00      11.16         24.4         49.979117            NaN              100680       16090           Mostly Cloudy
29   2024-05-07T20:54:00+00:00       5.40         23.9         53.550674            NaN              100710       16090                   Clear
30   2024-05-07T21:54:00+00:00       7.56         21.7         61.189733            NaN              100710       16090           Mostly Cloudy
```

# Query to Retrieve Lowest and Highest Temperatures

- The second query retrieves the lowest and highest temperatures observed in the data set.
- This is achieved using the SQL command: SELECT MIN(temperature), MAX(temperature) FROM observations;.
- The result shows the minimum and maximum temperatures in Celsius, as provided by the NOAA weather service.

```python
#Purpose: Query database using SQL
#Name: Hector Acosta
#Date: 5/21/24
#   Run BuildWeatherDB.py to build weather database before running this program

import sqlite3
import pandas as pd


#file names for database and output file
dbFile = "weather.db"

#format output
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
pd.set_option('display.max_colwidth', None)
pd.set_option('display.expand_frame_repr', False)

#connect to and query weather database
conn = sqlite3.connect(dbFile)
#create SQL command
selectCmd = " SELECT MIN(temperature), MAX(temperature) FROM observations; "


#print out the query
result = pd.read_sql_query(selectCmd, conn)
print(result)
```

```
--- Running Module4Project.py ---
   MIN(temperature)  MAX(temperature)
0               2.8              30.0
```

# Query to Retrieve All Clear Days

- The third query retrieves the temperature and windspeed for records where the weather description is 'Clear'.
- This is achieved using the SQL command: SELECT temperature, windspeed, textDescription FROM observations WHERE textDescription = 'Clear';.
- The result shows the relevant data for all clear weather observations.

```
--- Running Module4Project.py ---
    temperature  windSpeed textDescription
0          13.9       0.00           Clear
1          15.6       5.40           Clear
2          17.8      16.56           Clear
3          20.0       9.36           Clear
4          20.0      11.16           Clear
5          12.8       9.36           Clear
6           2.8       5.40           Clear
7           3.3       0.00           Clear
8           3.9       0.00           Clear
9           3.3       0.00           Clear
10          3.9       0.00           Clear
11          6.1       0.00           Clear
12          6.7       0.00           Clear
13          7.2       5.40           Clear
14         10.0      11.16           Clear
15         12.2      18.36           Clear
16         13.9      18.36           Clear
17         15.0      11.16           Clear
18         12.8      14.76           Clear
19         10.6      16.56           Clear
20         20.6      11.16           Clear
21         15.6       5.40           Clear
22         12.8       0.00           Clear
23         16.1       7.56           Clear
24         14.4      11.16           Clear
25         16.7       5.40           Clear
26         18.9       7.56           Clear
27         22.8       9.36           Clear
28         23.9       5.40           Clear
29         25.6      11.16           Clear
30         27.2      12.96           Clear
31         28.9      25.92           Clear
32         29.4      25.92           Clear
33         30.0      22.32           Clear
34         30.0      18.36           Clear
35         29.4      25.92           Clear
36         25.0      11.16           Clear
37         22.2       9.36           Clear
38         20.0       9.36           Clear
39         17.2       0.00           Clear
40         23.9       5.40           Clear
```

Module4Project.py › ...

```python
#Purpose: Query database using SQL
#Name: Hector Acosta
#Date: 5/21/24
#   Run BuildWeatherDB.py to build weather database before running this program

import sqlite3
import pandas as pd


#file names for database and output file
dbFile = "weather.db"

#format output
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
pd.set_option('display.max_colwidth', None)
pd.set_option('display.expand_frame_repr', False)

#connect to and query weather database
conn = sqlite3.connect(dbFile)
#Create SQL command
selectCmd = "SELECT temperature, windspeed, textDescription FROM observations where textDescription = 'Clear'; "

#print out the query
result = pd.read_sql_query(selectCmd, conn)
print(result)
```

# Querying and Manipulating Data with SQL and Python

1

**Retrieve weather data using SQL embedded in Python:**

- Learn to connect to a SQLite database and execute SQL queries from within a Python script.
- Understand how to fetch data from a database and handle it in Python.

2

**Cleanse and manipulate the data:**

- Identify and remove invalid or missing data entries.
- Transform data, such as converting temperatures from Celsius to Fahrenheit.

3

**Save the data as a CSV file:**

- Export the cleaned and transformed data into a CSV format that is compatible with Excel and other data analysis tools.

4

**Visualize the data in Excel:**

- Import the CSV file into Excel.
- Create charts to visually analyze the weather data trends over a specific period.

# Python Code for Data Extraction

- The Python script connects to the `weather.db` database and extracts temperature and humidity data.
- Data is cleansed and transformed by removing invalid values and converting temperatures to Fahrenheit.
- The cleansed data is saved into a CSV file.

Module5Project.py > ...

```python
#Purpose: Extract temperature, humidity data from weather database into CSV file
#Name: Hector Acosta
#Date: 5/28/2024
#   Run BuildWeatherDB.py to build weather database before running this program
import sqlite3
#convert Celsius temperature to Fahrenheit
def convertCtoF(tempC):
    return (tempC*9.0/5.0) + 32.0
#file names for database and output file
dbFile = "weather.db"
output_file_name='formatdata1.csv'
#connect to and query weather database and
conn = sqlite3.connect(dbFile)
#create cursor to execute SQL commands
cur = conn.cursor()
selectCmd = """ SELECT temperature, relativeHumidity FROM observations
                ORDER BY timestamp; """
cur.execute(selectCmd)
allRows = cur.fetchall()
#limit the number of rows output to half
rowCount = len(allRows)//2 # double slash does integer division
rows = allRows[:rowCount]

#write data to output file
with open(output_file_name,"w+") as outf:
    outf.write('Celsius,Fahrenheit,Humidity')
    outf.write('\n')
    for row in rows:
        tempC = row[0]
        if tempC is None:       #handle missing temperature value
            continue
        else:
            tempF = convertCtoF(tempC)
            outf.write(str(tempC)+',')
            outf.write(str(tempF)+',')
        humidity = row[1]
        if humidity is None:   #handle missing humidity value
            outf.write('\n')
        else:
            outf.write(str(humidity)+'\n') #print data to file separated by commas
```

# CSV Data in Excel

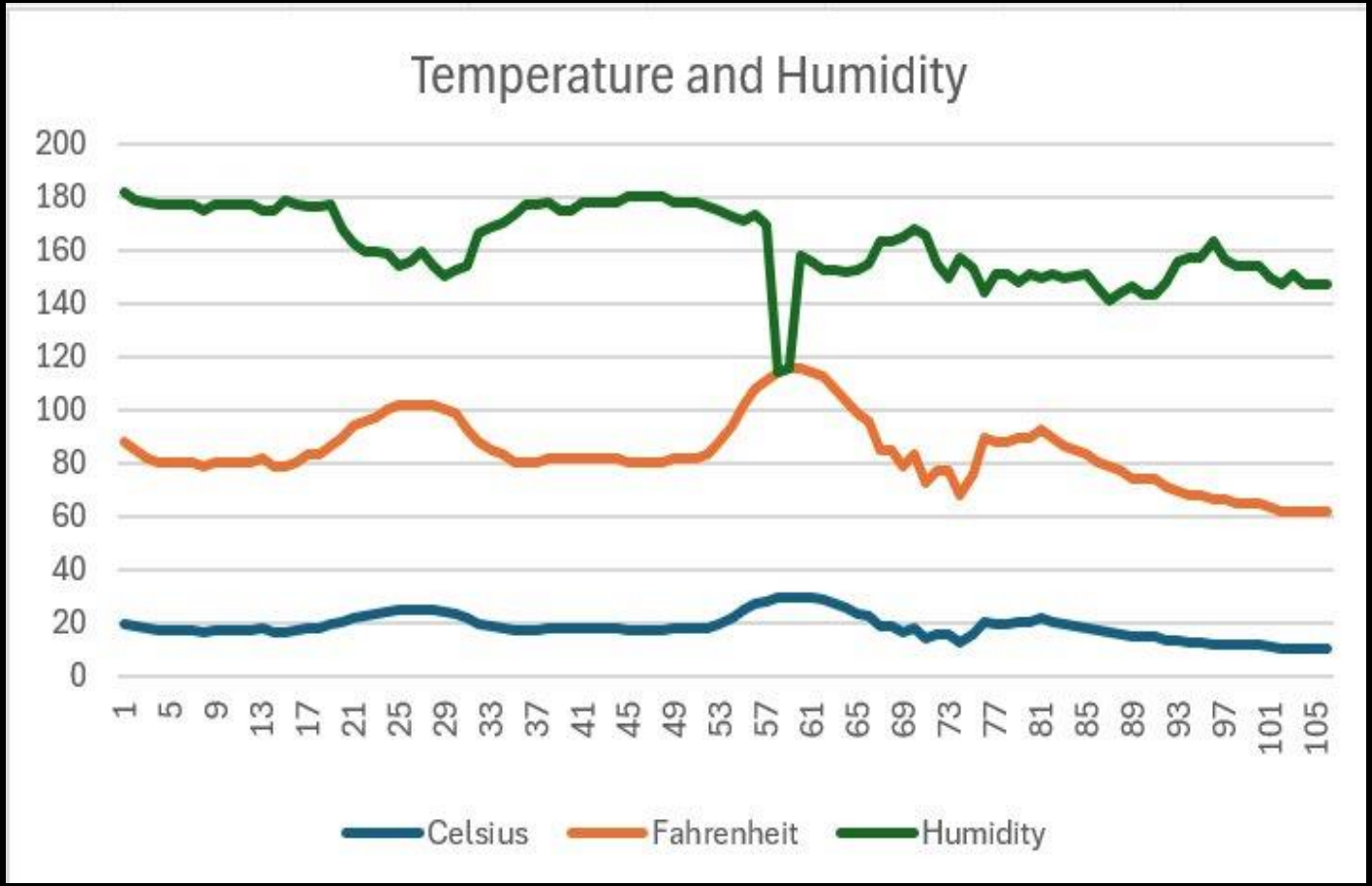
Data Retrieved and Converted to CSV Format

- The extracted data is saved in formatdata1.csv and formatdata2.csv.
- The CSV file shows the data in three columns: Celsius, Fahrenheit, and Humidity.
- This data can now be used for further analysis and visualization.

| Celsius | Fahrenheit | Humidity |
|---|---|---|
| 20 | 68 | 93.3899 |
| 18.9 | 66.02 | 93.336 |
| 17.8 | 64.04 | 96.2845 |
| 17.2 | 62.96 | 96.8811 |
| 17.2 | 62.96 | 96.8811 |
| 17.2 | 62.96 | 96.8811 |
| 17.2 | 62.96 | 96.8811 |
| 16.7 | 62.06 | 96.2536 |
| 17.2 | 62.96 | 96.8811 |
| 17.2 | 62.96 | 96.8811 |
| 17.2 | 62.96 | 96.8811 |
| 17.2 | 62.96 | 96.8811 |
| 17.8 | 64.04 | 93.2815 |
| 16.7 | 62.06 | 96.2536 |
| 16.7 | 62.06 | 100 |
| 17.2 | 62.96 | 96.8811 |
| 18.3 | 64.94 | 93.3064 |
| 18.3 | 64.94 | 93.3064 |
| 19.4 | 66.92 | 90.4728 |
| 20.6 | 69.08 | 78.3518 |
| 22.2 | 71.96 | 68.3754 |

# Temperature and Humidity Chart

- The data from the CSV file is used to create a line chart in Excel.
- The chart visualizes the temperature and humidity data over time.
- This helps in analyzing weather patterns and trends.



# Develop Graphical Models and Interpret Results

**Objective:**
To develop graphical models using Python data analytics modules and interpret results from the data analysis.

**Data Collection and Preparation:**
The Python script connects to the weather.db database and extracts temperature and humidity data.

- Data is cleansed and transformed by removing invalid values and converting temperatures to Fahrenheit.
- The cleansed data is saved into a CSV file.

**Project Overview:**
This project focuses on developing graphical models and interpreting results using pandas, the data analytics module in Python.

- Created a histogram of humidity data from the second period to analyze the distribution and frequency of humidity values across the dataset.
- Generated box plots for week 2 data for Celsius, Fahrenheit, and Humidity to identify the range and distribution of temperature and humidity data.
- Analyzed if the highest humidity values correspond to specific temperature ranges in the provided data.
- Based on the data and weather forecast, predicted variations in temperature and humidity for the next few days in zipcode 08361.

# Plot #1 - Histogram

**Purpose:**

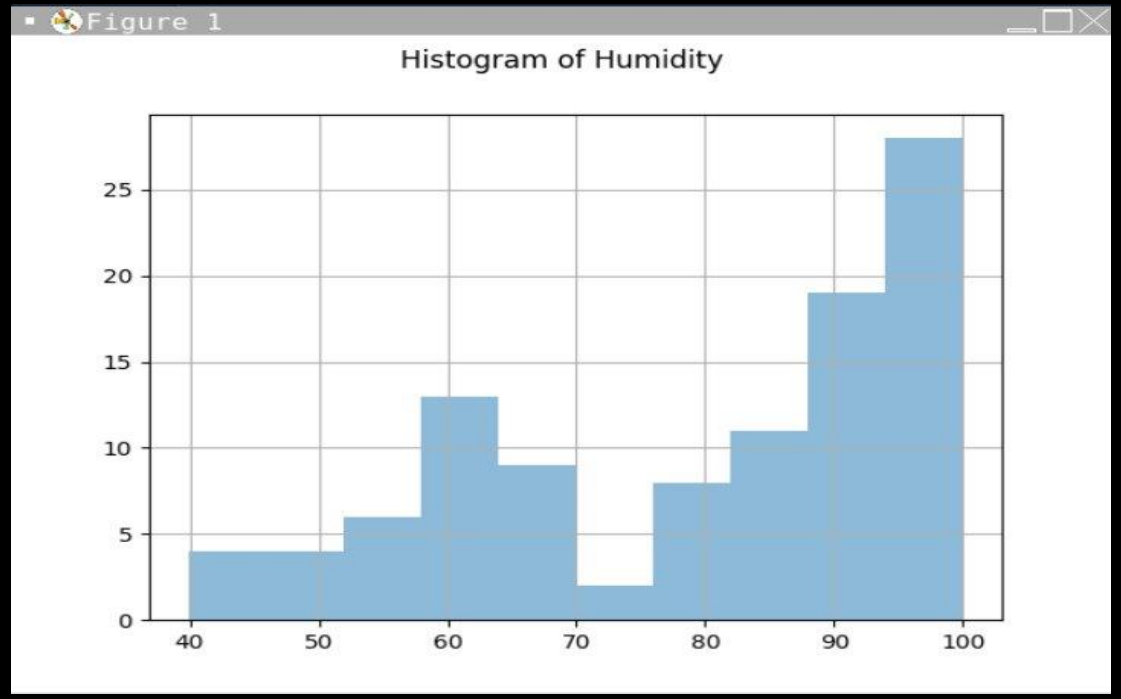
Create a histogram of humidity data from the second period.

**Analysis:**

Display the distribution of humidity data.

Understand the frequency of humidity values across the dataset.

```python
#Purpose: Create a histogram of humidity data from the second period
#Name: Hector Acosta
#Date: 6/3/2024
import pandas as pd
import matplotlib.pyplot as plt
df1 = pd.read_csv("formatdata1.csv")
df1['Humidity'].hist(bins=10, alpha=0.5); plt.suptitle('Histogram of Humidity')
plt.show()
```



# Plot #2 - Box Plot

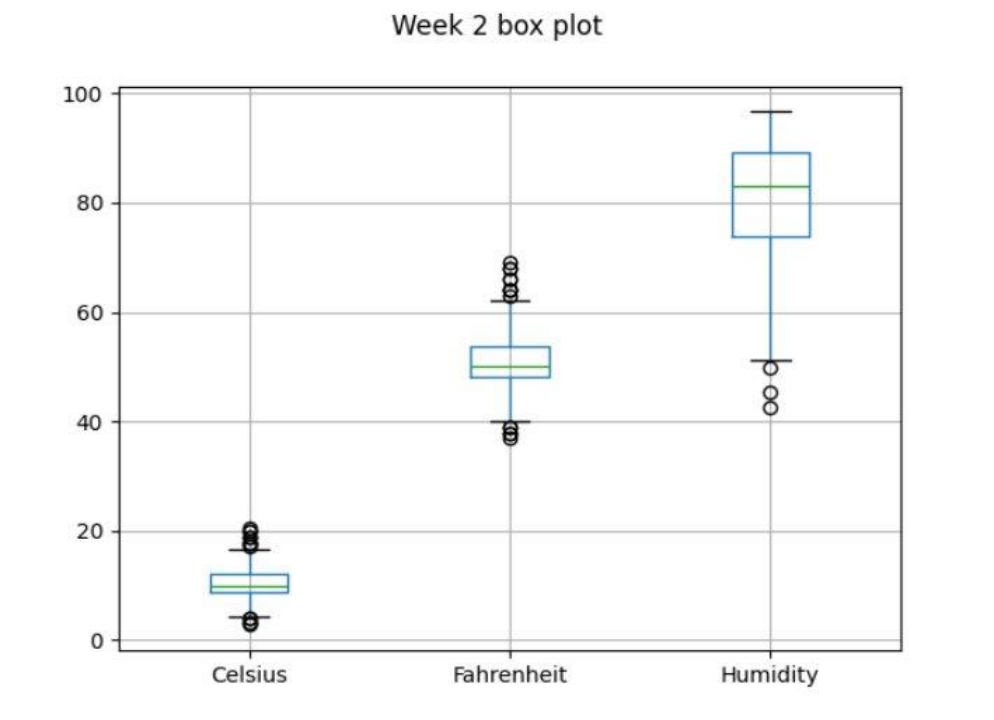
```python
#Name: Hector Acosta
#Date: 6/3/2024
import pandas as pd
import matplotlib.pyplot as plt
df2 = pd.read_csv("formatdata2.csv")
df2.boxplot(); plt.suptitle("Week 2 box plot")
plt.show()
```

**Purpose:**

Create box plots for week 2 data for Celsius, Fahrenheit, and Humidity.

**Analysis:**

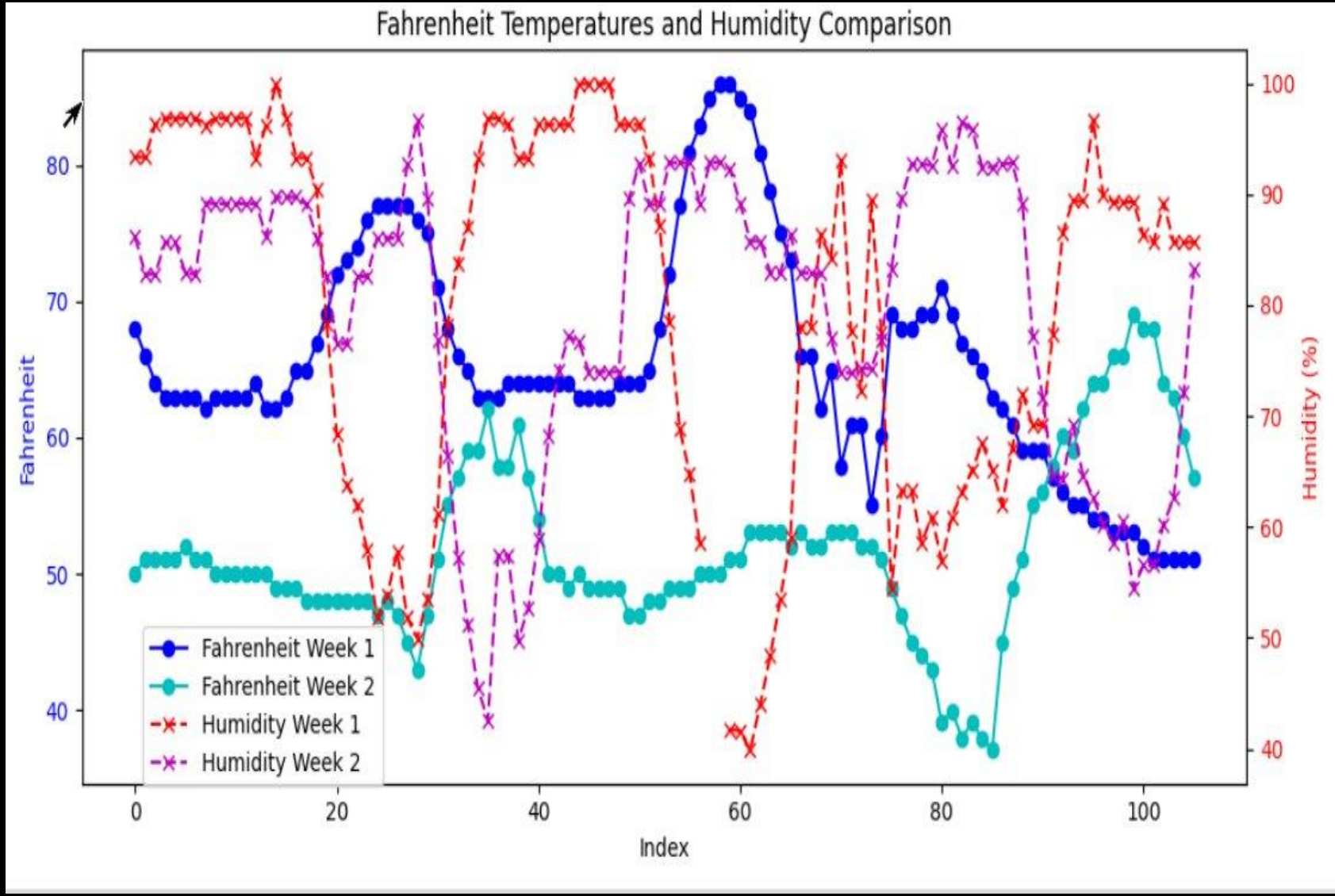
- Identify the range and distribution of temperature and humidity data.
- Compare variability between Celsius, Fahrenheit, and Humidity.



# Analysis

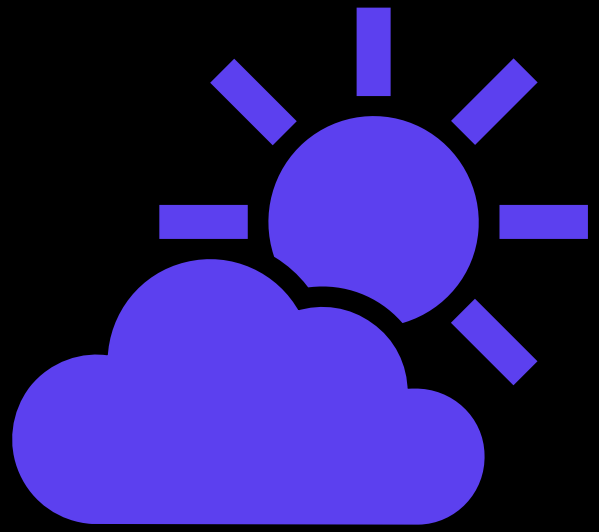
Do the highest humidity values correspond to specific temperature ranges?

- In both weeks, the highest humidity values correspond to specific temperature ranges: Week 1 displays high humidity at around 44.96°F, 45.0°F, and 46.9°F. Week 2 exhibits high humidity at around 37.94°F, 39.02°F, and 42.98°F. As as result, it can be concluded that the highest humidity values do correspond to specific temperature ranges in the provided data.



# Prediction

- Based on my data and the provided weather forecast for the next few days, here's a prediction and analysis for temperature and humidity variations over the next few days in zipcode 08361:
- Temperature Variations:
  - Mostly Sunny (Actual and Forecast): - Highs: 81°F to 84°F - Lows: 54°F to 61°F - Humidity Expectation: During the day, lower humidity at around 40-50%. As temperatures cool at night, expect humidity to rise to around 60-70%.
  - Cloudy (Actual and Forecast): - Highs: 77°F to 85°F - Lows: 64°F to 69°F - Humidity Expectation: Cloudy conditions will bring higher humidity. Daytime humidity may range from 50-70%, while nighttime humidity could be around 70-80%.
  - Thunderstorms (Forecast): - High: 82°F - Low: 64°F - Humidity Expectation: Expect higher humidity during thunderstorms, potentially reaching 70-80% during the day and remaining high at night. AM Showers and PM
  - Showers (Forecast): - Highs: 79°F to 83°F - Lows: 62°F to 66°F - Humidity Expectation: Showers will bring higher humidity during the day, likely around 70-80%, and remaining high at night.
  - Partly Cloudy (Forecast): - Highs: 78°F to 82°F - Lows: 60°F to 67°F - Humidity Expectation: Moderate humidity during the day, possibly around 50-60%, and increasing slightly at night to 60-70%.
- Humidity and Temperature Relationship: - If Temperature Goes Up: As temperatures rise, the air's moisture capacity increases, leading to lower relative humidity.
- For example, expect humidity to decrease from around 70% to 50% as temperatures rise from 77°F to 85°F. - If Temperature Goes Down: Cooler temperatures reduce the air's moisture capacity, resulting in higher relative humidity. For example, expect humidity to increase from around 50% to 70% as temperatures drop from 85°F to 64°F.



# Challenges

- In this project, I utilize Excel and Replit for data analysis, encountering several challenges typical for newcomers to Python. First, connecting to the database posed an initial hurdle, requiring research into database connection strings and basic SQL commands. Subsequently, data cleansing and transformation demanded proficiency with pandas functions to handle cleaning and unit conversion effectively. Generating graphical models for visualization purposes necessitated a deep dive into matplotlib and pandas plotting functionalities to create meaningful representations of the data. Upon producing these visuals, interpreting results became paramount, entailing a grasp of statistical methods and domain knowledge to glean insights from the data trends. Forecasting and prediction presented additional challenges, demanding an understanding of time series analysis and forecasting models to make accurate predictions and interpretations. Handling file operations, debugging and troubleshooting, optimizing code efficiency, and ensuring clear documentation and reporting were also critical aspects of the project, each requiring dedicated study and practice to overcome.

# Carrer Skills

- **Database Management:** Understanding database structures and executing SQL queries for cybersecurity analysis.
- **Data Cleansing and Transformation:** Cleaning and preparing data for analysis to detect security threats.
- **Data Visualization:** Creating visual representations of cybersecurity data (e.g., network traffic analysis, intrusion detection) using tools like matplotlib and pandas.
- **Statistical Analysis:** Analyzing patterns and trends in cybersecurity data to identify anomalies and potential threats.
- **Forecasting and Prediction:** Using time series analysis and forecasting models to predict cybersecurity trends and potential threats.
- **File Operations:** Handling and analyzing log files, packet captures, and other cybersecurity data sources.
- **Debugging and Troubleshooting:** Identifying and resolving security issues in scripts and applications.
- **Code Optimization:** Writing efficient code for security tools and scripts, focusing on performance and reliability.
- **Documentation and Reporting:** Documenting security findings, incidents, and providing clear, actionable reports.
- **Project Management:** Managing cybersecurity projects, including planning, execution, and meeting security objectives.

These skills are crucial for a career in cybersecurity, providing a strong foundation for threat detection, incident response, and overall security management.

# Conclusion

- This project has provided a comprehensive introduction to data analysis using Python, focusing on weather data retrieved from a cloud source and stored in a local database. Utilizing tools like Excel and Replit, I encountered and overcame various challenges typical for newcomers to Python and data analytics. These challenges included connecting to databases, cleansing and transforming data, creating graphical models, and interpreting results.
- Through this project, I gained valuable career skills in database management, data cleansing, visualization, statistical analysis, and more. Looking forward, these skills are not only relevant for further educational pursuits but also for a career in cybersecurity. The ability to manage and analyze data effectively is crucial in identifying and mitigating security threats.
- This project has laid a solid foundation for my future studies and career aspirations in cybersecurity, emphasizing the importance of continuous learning and practical application of programming and data analytics skills.
- In essence, this project has equipped me with the fundamental skills and knowledge needed to pursue further studies and eventually excel in the field of cybersecurity.

# Resources

- **Live Lectures from DeVry University Professors**
    - **Edwin Hill**
    - **Saeed Jellouli**

These live lectures provided valuable insights and practical knowledge that were instrumental in completing this project successfully.