International Space Station Weather Analysis

By: Julia Couch, Selin Fidan, and Michelle Karls
Repository link: https://github.com/mekarls/fa20-finalproject.git

How we started & Where we are now

Initial Goals

- ✓ Use Twitter API to get data on tweets from Joe Biden & Donald Trump
- ✓ Use Daylight API (Is the sun out?)
- ✓ Use Weather API (what is the weather while tweeting?)

Final Result & Goals

- ✓ Use ISS API to get data points
- ✓ Send data to Weather API to decide whether valid
- ✓ Get Weather & Daylight API data
- ✓ Create visual summaries of data

Major Issue: Twitter API more difficult to use than we thought, switched to ISS Location API

Final Focal Question

How can weather and daylight data be analyzed in the context of the location of the International Space Station?

Biggest problem we faced:

- ISS not always near a weather station- used ISS tracker to ensure it was over land.
- https://spotthestation.nasa.gov/tracking_map.cfm)
- Ended up needing to collect 700+ raw data points to get 102 usable points.
- Comprehensive list of documentation at the end of the presentation.

Database Tables

API_Data.db

ISS_Raw

706 total data points

	date	time	latitude	longitude
	Filter	Filter	Filter	Filter
510	2020-12-05	18:50:31	-51.4214	138.5774
511	2020-12-05	18:51:30	-50.9104	144.3499
512	2020-12-05	18:52:29	-50.0975	149.9539
513	2020-12-05	18:53:28	-49.0010	155.3297
514	2020-12-05	18:54:27	-47.6437	160.4354
515	2020-12-05	18:55:26	-46.0501	165.2478
516	2020-12-05	18:56:26	-44.2460	169.7585
517	2020-12-05	18:57:25	-42.2562	173.9721
518	2020-12-05	18:58:24	-40.0853	177.9342
519	2020-12-05	18:59:23	-37.7909	-178.4017
520	2020-12-05	19:00:23	-35.3746	-174.9789
521	2020-12-06	01:49:40	50.8050	-141.0052

http://open-notify.org/Open-Notify-API/ISS-Location-Now/

ISS_Data

102 data points

	date	time	latitude	longitude
	Filter	Filter	Filter	Filter
1	2020-12-03	15:06:32	39.9070	69.0450
2	2020-12-03	15:11:34	27.0118	85.4354
3	2020-12-03	18:06:22	50.0829	-6.3673
4	2020-12-03	18:06:52	49.5453	-3.5461
5	2020-12-03	18:07:23	48.9237	-0.7489
6	2020-12-03	18:07:54	48.2417	1.9287
7	2020-12-03	18:08:24	47.4932	4.5292
8	2020-12-03	18:08:55	46.6680	7.0900
9	2020-12-03	18:09:26	45.7964	9.5278
10	2020-12-03	18:10:27	43.9061	14.1226
11	2020-12-03	18:10:57	42.8959	16.2839
12	2020-12-03	18:11:27	41.8416	18.3698
13	2020-12-03	18:12:28	39.5550	22.4202
14	2020-12-03	18:13:30	37.1400	26.1569

	id	time_zone
	Filter	Filter
9	9	Asia/Dushanbe
10	10	Asia/Karachi
11	11	Asia/Kathmandu
12	12	Europe/London
13	13	Etc/GMT
14	14	Europe/Paris
15	15	Europe/Zurich
16	16	Europe/Rome
17	17	Etc/GMT-1
18	18	Europe/Zagreb
19	19	Europe/Athens
20	20	Asia/Riyadh
21	21	Africa/Brazzaville
22	22	Africa/Douala
23	23	America/Detroit

Time_Zones:

*shared id with Weather

Weather

*shared id with Time_Zones

	time_zone	wind_dir	temp	max_temp	windspeed	precipitation	dew	humidity	conditions
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	9	250.0	40.9	40.9	8.9	0.0	35.5	80.94	Overcast
2	11	270.0	64.3	64.3	6.9	0.0	46.3	52.0	Overcast
3	12	330.0	46.8	46.8	39.1	0.0	39.6	75.9	Clear
4	13	330.0	47.2	47.2	13.9	0.0	40.7	78.09	Clear
5	14	190.0	44.5	44.5	13.9	0.0	40.9	87.09	Overcast
6	14	180.0	44.5	44.5	19.7	0.0	43.3	95.48	Overcast
7	14	180.0	40.9	40.9	11.4	0.0	35.5	80.94	Clear
8	15	150.0	35.5	35.5	3.4	0.0	30.1	80.51	Clear
9	16	110.0	35.5	35.5	3.4	0.0	35.5	100.0	Overcast
10	17	240.0	44.5	44.5	4.7	0.0	44.5	100.0	Overcast
11	18	310.0	49.9	49.9	9.2	0.0	46.3	87.38	Overcast
12	17	150.0	57.1	57.1	6.9	0.0	46.3	67.14	Partially cloudy
13	19	110.0	57.1	57.1	16.1	0.0	48.1	71.84	Overcast
14	19	50.0	57.1	57.1	8.1	0.0	46.3	67.14	Clear
15	19	40.0	60.7	60.7	12.8	0.0	55.3	82.38	Overcast
16	20	90.0	71.5	71.5	3.4	0.0	26.5	18.56	Clear
17	20	250.0	80.5	80.5	10.3	0.0	49.9	34.43	Clear
18	21	250.0	75.1	75.1	3.4	0.0	71.5	88.6	Clear
19	22	NULL	82.3	82.3	2.2	0.0	76.9	83.78	Overcast

https://www.visualcrossing.com/weather-api

Daylight

	sunrise	sunset	solar_noon	day_length
	Filter	Filter	Filter	Filter
1	2:28:49 AM	11:59:01 AM	7:13:55 AM	09:30:12
2	12:52:13 AM	11:24:27 AM	6:08:20 AM	10:32:14
3	8:06:19 AM	4:24:59 PM	12:15:39 PM	08:18:40
4	7:52:43 AM	4:16:00 PM	12:04:22 PM	08:23:17
5	7:38:55 AM	4:07:25 PM	11:53:10 AM	08:28:30
6	7:25:27 AM	3:59:28 PM	11:42:27 AM	08:34:01
7	7:12:07 AM	3:52:00 PM	11:32:03 AM	08:39:53
8	6:58:45 AM	3:44:52 PM	11:21:48 AM	08:46:07
9	6:45:49 AM	3:38:17 PM	11:12:03 AM	08:52:28
10	6:20:56 AM	3:26:24 PM	10:53:40 AM	09:05:28
11	6:09:01 AM	3:21:02 PM	10:45:01 AM	09:12:01
12	5:57:22 AM	3:15:59 PM	10:36:40 AM	09:18:37
13	5:34:26 AM	3:06:30 PM	10:20:28 AM	09:32:04
14	5:12:53 AM	2:58:09 PM	10:05:31 AM	09:45:16

https://sunrise-sunset.org/api

How to run the code:

How to retrieve data

- In api_code.py, begin with inserting the number of ISS raw data points you want to collect (max 25), line 46

```
for _ in range(10):
```

- Next, in main(), uncomment the function “create_ISS_tables(cur, conn)”

```
177
178 def main():
179
180     # Database and Tables
181     cur, conn = setUpDatabase('API_Data.db')
182
183     create_iss_tables(cur, conn)
184
185     # create_weather_tables(cur, conn)
186
187     # weather(cur, conn)
188     create_daylight_table(cur, conn)
189
190 if __name__ == '__main__':
191     main()
192
```

- Run. Now, you have your raw data inserted in the database. Comment out create_ISS_table(cur, conn)

- Next, let's process the data to see if it's valid by running it through a weather API
  - Check the database to see what increment of raw data you need to check
  - Replace line 84 with this increment

```
    for i in data[691:707]: # x[191:216] x[216:241] x[2
        # print(i)
        dat, tim, lat, lon= i[0], i[1], i[2], i[3]
```

- Uncomment weather in main(), run code, comment it back out.

```
def main():

    # Database and Tables
    cur, conn = setUpDatabase('API_Data.db')

    # create_iss_tables(cur, conn)

    # create_weather_tables(cur, conn)

    weather(cur, conn)
```

- Now in the create_daylight_table(), increment your data in line 158, the same increment as in weather()

```
# print(data)
for d in data[105:106]: #increment manually
    lat, long, date = d[0], d[1], d[2]
    print(lat, long, date)
```

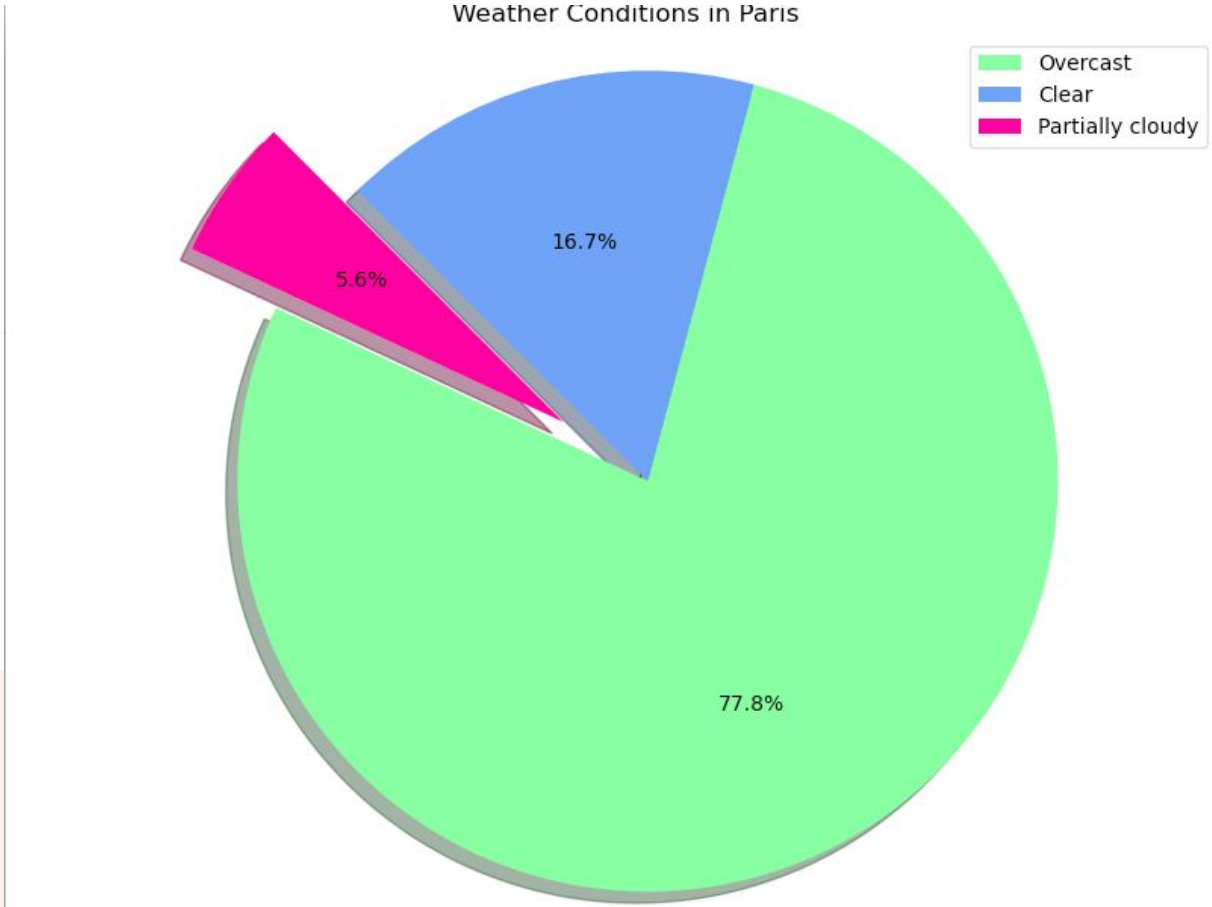
- Uncomment create_daylight_table(cur, conn) in main() and run. Now you have collected data, confirmed its usability, got more information from two APIs and inserted all of this into the database.

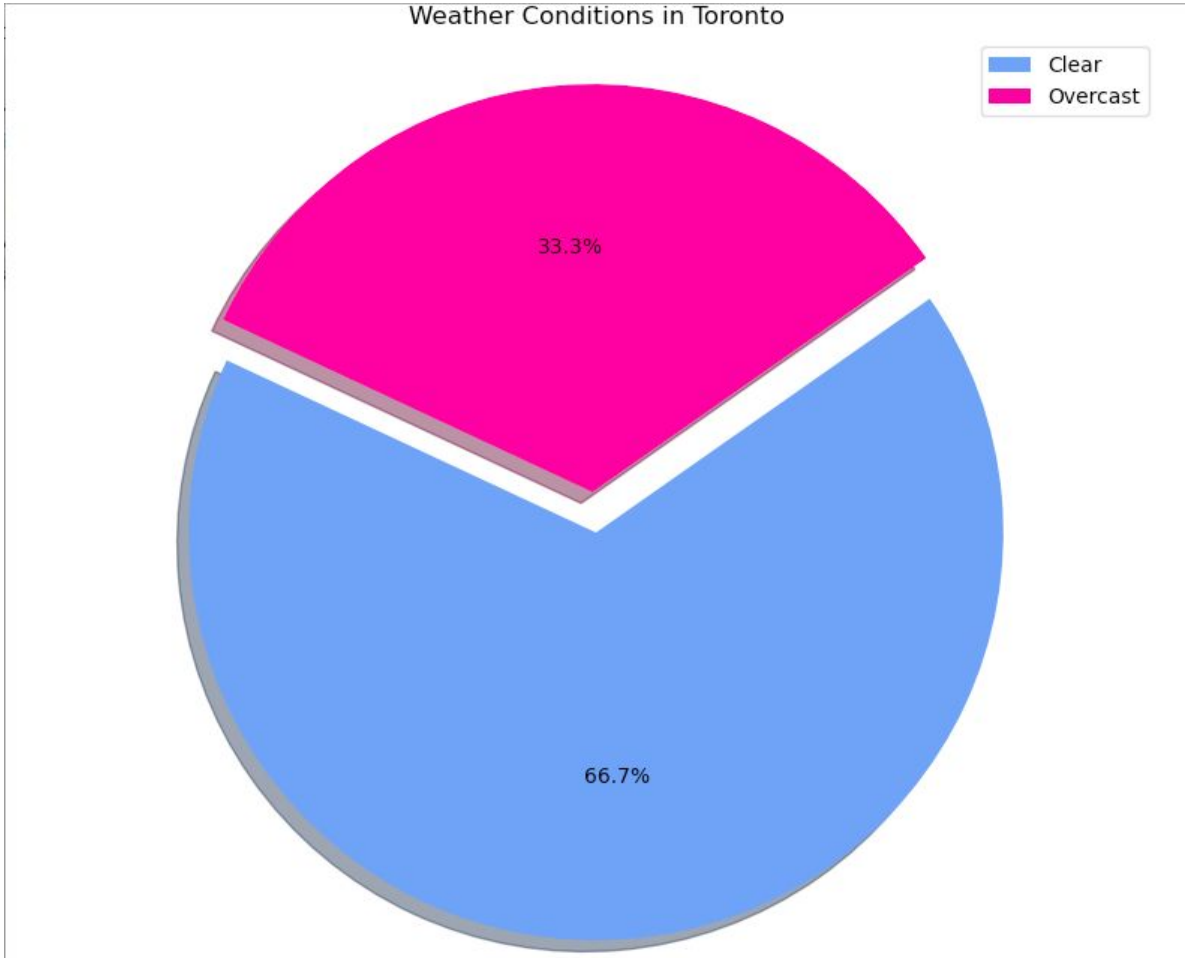
```
178  def main():
179
180      # Database and Tables
181      cur, conn = setUpDatabase('API_Data.db')
182
183      # create_iss_tables(cur, conn)
184
185      # create_weather_tables(cur, conn)
186      #
187      # weather(cur, conn)
188      create_daylight_table(cur, conn)
```

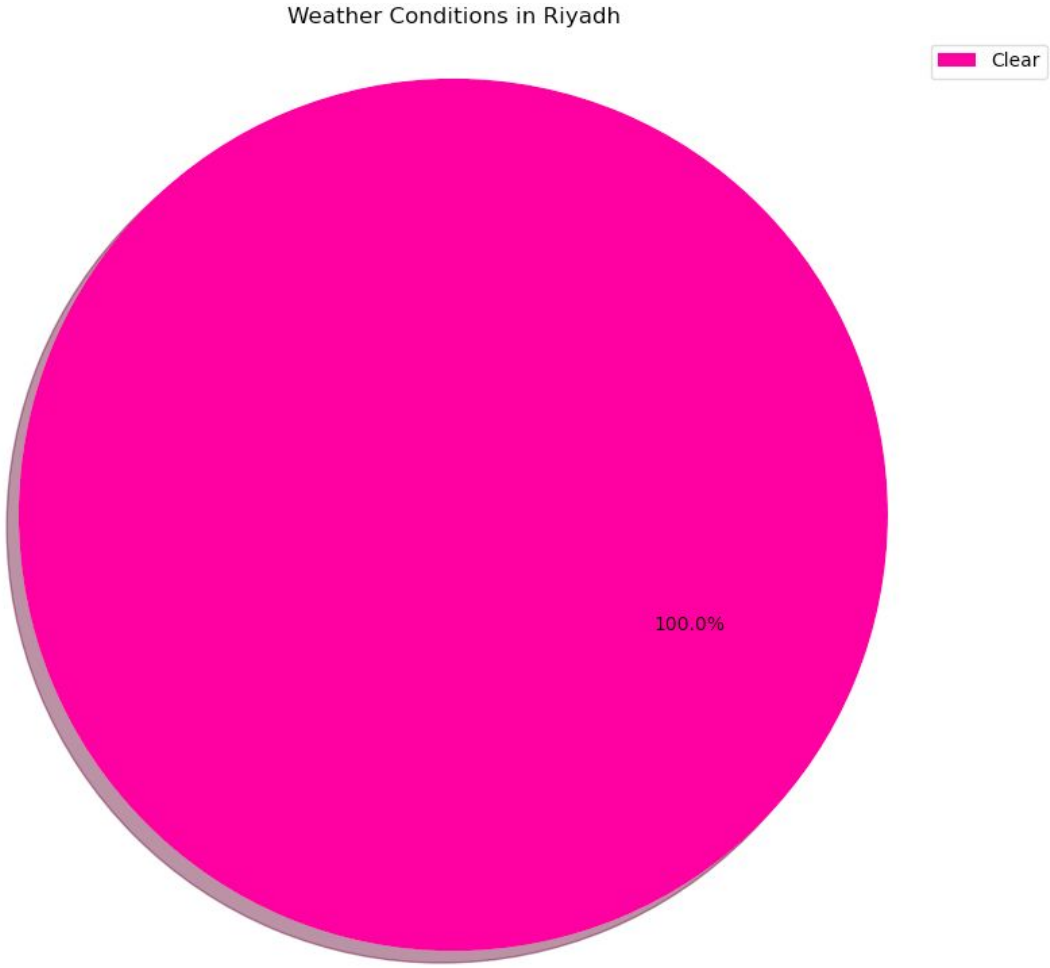
- ✓ Now, in order to see the visualizations of this data, open “visualizations.py”
- ✓ Run code
- ✓ Close out each graph to reveal the next!

Visualizations

found in “visualizations.py”


Weather Conditions in Paris
Overcast
Clear
Partially cloudy
16.7%
5.6%
77.8%


Weather Conditions in Toronto
Clear
Overcast
33.3%
66.7%

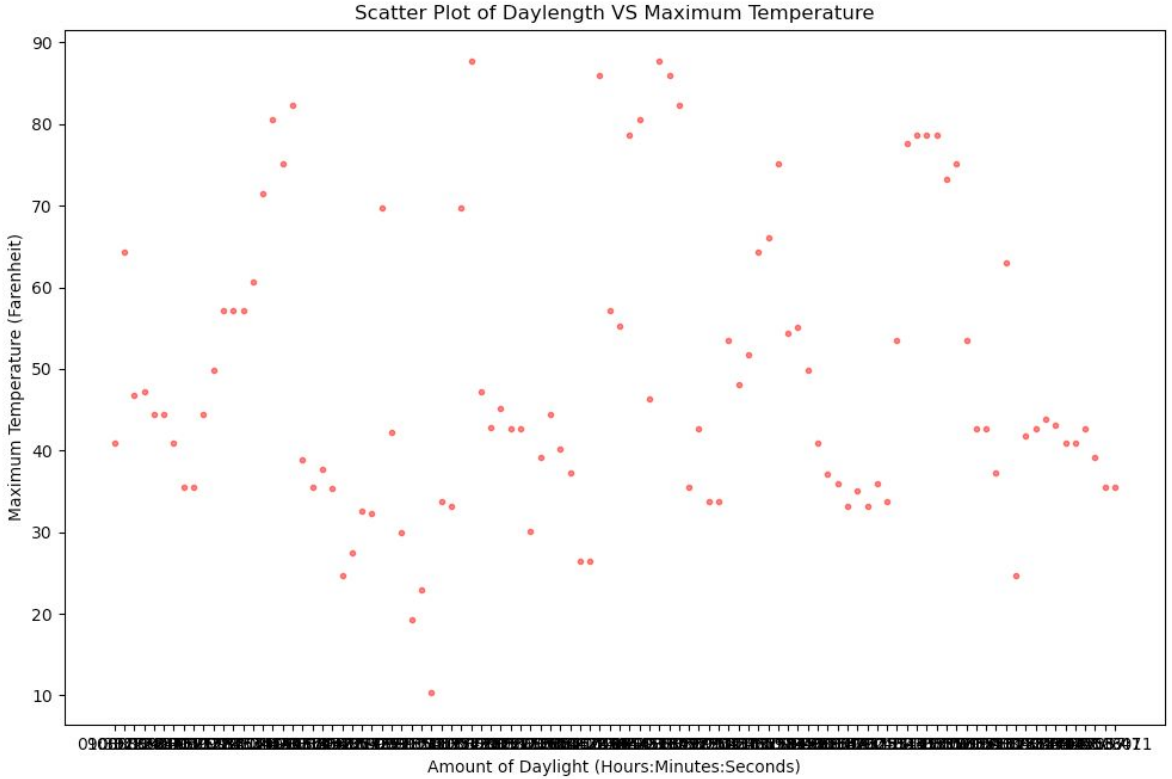

Weather Conditions in Riyadh
Clear
100.0%

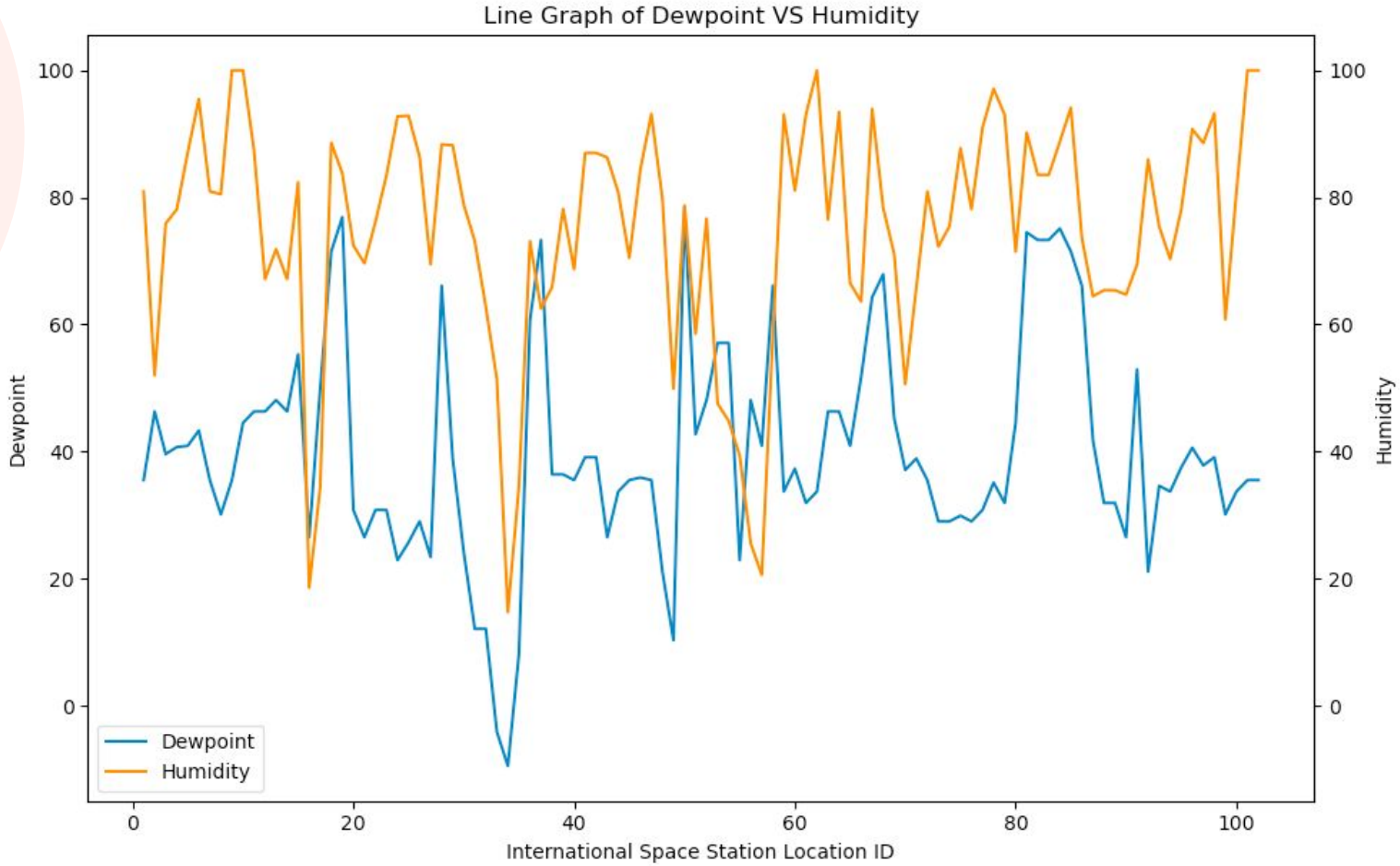
Explanatory/ Calculation File

percentages.csv

```
1  time_zone, overcast, clear, partially cloudy
2  paris, 77.78, 16.67, 5.56
3  toronto, 66.67, 33.33, 0.0
4  riyadh, 100.0, 0.0, 0.0
```

*The percentages used for one set of visualizations


Scatter Plot of Daylength VS Maximum Temperature
Maximum Temperature (Farenheit)
90
80
70
60
50
40
30
20
10
Amount of Daylight (Hours:Minutes:Seconds)


Line Graph of Dewpoint VS Humidity
Dewpoint
Humidity
International Space Station Location ID
Dewpoint
Humidity
0
20
40
60
80
100

What does each function do?

def setupDatabase(db_name):

This function takes in the name of a non-existing database, and creates one, returning cur & conn.

def ISS_position():

This function requests data from the ISS API.

Inputs: none

Outputs: formatted string of unix time, latitude, and longitude for that particular location

Called: in create_iss_tables()

def create_iss_tables(cur, conn):

Creates two tables, ISS_Raw and ISS_Data.

Inputs: cur and conn (database connection)

Outputs: no return statement,writes 25 max data to ISS_Raw, created tables

def create_weather_tables(cur, conn):

This function creates a two new tables, timezone & weather, in the database.

Input: cur, conn Output: no return statement, created tables

def weather(cur, conn):

This function comprises the bulk of our code. First, it selects all data from ISS_Raw. Next, it slices only the data we want to test. It runs each data point through the Weather API and checks whether there is an error in the return object. If there is no error, and there are actual values (as opposed to None) it writes the original data point (time, lat, long) to ISS_Data. If the time zone for that specific data point is not already in Time_Zones table, it inserts the time zone and a primary integer key. Finally, it inserts the returned non-null weather values into the Weather table, with a foreign key to Time_Zones.

Inputs: cur, conn

Outputs: no return statement, writes to 3 different database tables

def create_daylight_table (cur, conn):

This function creates a table in the database for daylight, creates a list from ISS_data, returns response from daylight API, adds it into weather table in db.

Input: cur, conn

Output: no return statement, creates daylight table, inserts data.

def main():

This function calls all other functions (except for iss_position()) to run the program.

Input: none

Output: cur, conn

def conditionsPiechart(cur, conn):

Creates ‘percentages.csv’ (calculations file). Uses INNER JOIN to pull conditions from Weather and time_zone from Time_Zones connected by the foreign key time zone id. We chose the 3 most common time_zones to analyze. Iterates through returned list of tuples (time zone, weather condition) for Paris, Toronto, and Riyadh and creates 3 dictionaries of condition frequency for each. Frequency percentages are calculated for each time zone, and are written to the csv file. 0.00 is added in place of no value. Lastly, the function creates a pie chart of weather conditions for each time zone.

Input: cur, conn Output: csv file, 3 pie charts

def daylengthVSmaxtemp(cur, conn):

This function requests data from the Daylight and Weather tables.

Inputs: curr and conn (database connection)

Outputs: formatted day_length and max_temp into lists and returns a scatter plot

def dewpointVShumidity(cur, conn):

This function requests specific data from the Weather table.

Inputs: cur and conn (database connection)

Outputs: formatted dew, humidity, and ISS location ID into lists and returns a line graph with two axes.

def main():

This is the main function in visualizations.py that calls cur and conn and connects it to API_Data.db to request data for visuals

Resource Documentation

Date	Issue	Resource	Result
11-25	database locked	https://stackoverflow.com/questi	save and close database, solved
11-26	unix to datetime?	https://www.w3resource.com/py	got format , solved
11-1	iss doesn't always collect valid data that would run through weather api	https://spotthestation.nasa.gov/t	most frustrating thing, had to track ISS location and collect 700+ raw data points
12-1	unique vs primary key	https://www.geeksforgeeks.org/	implemented primary key into time_zones, solved
12-2	unique text values?	https://stackoverflow.com/questi	utilized INSERT INTO OR IGNORE, solved
12-3	problems with returning weather api object	https://stackoverflow.com/questi	did not help, example given was too different
12-3	problems with converting api response	https://stackoverflow.com/questi	worked, but had to dumps() into a json object to use json editor- double reformat necessary
12-5	append multiple items to a list at once?	https://stackoverflow.com/questi	learned .extend(), solved!
12-7	issues creating pie charts	https://pythonspot.com/matplotli	solved! formatting issues
12-7	issues with plotly	https://plotly.com/python/pie-ch	not resolved, did not end up using plotly
12-10	how to approach a non-existent index when you need a certain list length?	https://stackoverflow.com/questi	did not need, forgot that .append() existed to add items to a list
10-Dec	forgot how to sort a dictionary	https://stackoverflow.com/questi	solved! Just needed a reminder on sorting

Thanks!

Questions? Comments? Thank you for listening!