

# Collecting weather data from an API

---

**Name:** Cuadra, Audrick Zander G.

**Section:** CPE22S3

**Date:** March 27, 2024

**Submitted to:** Engr. Roman Richard

---

## About the data

In this notebook, we will be collecting daily weather data from the [National Centers for Environmental Information \(NCEI\) API](#). We will use the Global Historical Climatology Network - Daily (GHCND) data set; see the documentation [here](#)..

Note: The NCEI is part of the National Oceanic and Atmospheric Administration (NOAA) and, as you can see from the URL for the API, this resource was created when the NCEI was called the NCDC. Should the URL for this change in the future, you can search for the NCEI weather API to find the updated one.

## ✓ Using the NCEI API

Paste your token below.

```
1 import requests
2
3 def make_request(endpoint, payload=None):
4     """
5     Make a request to a specific endpoint in the API
6     passing headers and optional payload.
7
8     Parameters:
9         - endpoint: The endpoint of the API you want to
10                    make GET request to.
11         - payload: A dictionary of data to pass along
12                   with the request.
13
14     Returns:
15         Response object.
16     """
17     return requests.get(
18         f'https://www.ncdc.noaa.gov/cdo-web/api/v2/{endpoint}',
19         headers={
20             'token': 'YNAdteQhHJsoetjhLCpXzrRhrGdGHvuc'
21         },
22         params=payload
23     )
```

## ✓ Collecting All Data Points for 2018 In NYC (Various Stations)

We can make a loop to query for all the data points one day at a time. Here we create a list of all the results:

```

1 import datetime
2
3 from IPython import display # for updating the cell dynamically
4
5 current = datetime.date(2018, 1, 1)
6 end = datetime.date(2019, 1, 1)
7
8 results = []
9
10 while current < end:
11     # update the cell with status information
12     display.clear_output(wait=True)
13     display.display(f'Gathering data for {str(current)}')
14
15     response = make_request(
16         'data',
17         {
18             'datasetid' : 'GHCND', # Global Historical Climatology Network - Daily (GH
19             'locationid' : 'CITY:US360019', # NYC
20             'startdate' : current,
21             'enddate' : current,
22             'units' : 'metric',
23             'limit' : 1000 # max allowed
24         }
25     )
26
27     if response.ok:
28         # we extend the list instead of appending to avoid getting a nested list
29         results.extend(response.json()['results'])
30
31     # update the current date to avoid an infinite loop
32     current += datetime.timedelta(days=1)

```

'Gathering data for 2018-12-31'

Now, we can create a dataframe with all this data. Notice there are multiples stations with values for each datatype on a given day. We don't know what the stations are, but we can look them up and add them to the data:

```

1 import pandas as p
2
3 df = p.DataFrame(results)
4 df.head()

```

	date	datatype	station	attributes	value
--	------	----------	---------	------------	-------

Save this data to a file:

```
1 df.to_csv('/content/nyc_weather_2018.csv', index=False)
```

and write it to the database:

```
1 import sqlite3
2
3 with sqlite3.connect('/content/weather.db') as connection:
4     df.to_sql(
5         'weather', connection, index=False, if_exists='replace'
6     )
```

For learning about merging dataframes, we will also get the data mapping station IDs to information about the station:

```
1 response = make_requests(
2     'stations',
3     {
4         'datasetid' : 'GHCND',
5         'locationid' : 'CITY:US360019',
6         'limit' : 1000
7     }
8 )
9
10 stations = p.DataFrame(response.json()['results'])[['id', 'name', 'latitude', 'longitu
11 stations.to_csv('/content/weather_stations.csv', index=False)
12
13 with sqlite3.connect('/content/weather.db') as connection:
14     stations.to_sql(
15         'stations', connection, index=False, if_exists='replace'
16     )
```

## ✓ Comments and Insights:

This module topic demonstrates collecting and managing data with the use of API. The module also demonstrates the manipulation of databases and csv simultaneously. Through this module I that dataframes can be written to databases with the use of sqlite.