```
import requests

def make_request(endpoint, payload=None):
    """
    Make a request to a specific endpoint on the weather API
    passing headers and optional payload.
    Parameters:
    - endpoint: The endpoint of the API you want to
    make a GET request to.
    - payload: A dictionary of data to pass along
    with the request.
    Returns:
    Response object.
    """
    return requests.get(
        f'https://www.ncdc.noaa.gov/cdo-web/api/v2/{endpoint}',
        headers={
            'token': 'pXBwHyopzPsefdNlOjlFekaxPowjdseD'
        },
        params=payload
    )
```

```
import datetime
from IPython import display

current = datetime.date(2018, 1, 1)
end = datetime.date(2019, 1, 1)
results = []

while current < end:
    # Update the cell with status information
    display.clear_output(wait=True)
    display.display(f'Gathering data for {str(current)}')

    response = make_request(
        'data',
        {
            'datasetid': 'GHCND',  # Global Historical Climatology Network - Daily (GHCND) dataset
            'locationid': 'CITY:US360019',  # NYC
            'startdate': current,
            'enddate': current,
            'units': 'metric',
            'limit': 1000  # max allowed
        }
    )

    if response.ok:
        # Extend the list instead of appending to avoid getting a nested list
        results.extend(response.json()['results'])

    # Update the current date to avoid an infinite loop
    current += datetime.timedelta(days=1)
```

```
'Gathering data for 2018-12-31'
```

```
import pandas as pd

df = pd.DataFrame(results)
df.head()
```

	date	datatype	station	attributes	value
0	2018-01-01T00:00:00	PRCP	GHCND:US1CTFR0039	„N,0800	0.0
1	2018-01-01T00:00:00	PRCP	GHCND:US1NJBG0015	„N,1050	0.0
2	2018-01-01T00:00:00	SNOW	GHCND:US1NJBG0015	„N,1050	0.0
3	2018-01-01T00:00:00	PRCP	GHCND:US1NJBG0017	„N,0920	0.0
4	2018-01-01T00:00:00	SNOW	GHCND:US1NJBG0017	„N,0920	0.0

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

```
import sqlite3

with sqlite3.connect('/content/weather.db') as connection:
  df.to_sql(
      'weather', connection, index = False, if_exists = 'replace'
  )
```

```
response = make_request(
    'stations',
    {
        'datasetid' : 'GHCND',
        'locationid' : 'CITY:US360019',
        'limit' : 1000
    }
)

stations = pd.DataFrame(response.json()['results'])[['id','name','latitude','longitude','elevation']]
stations.to_csv('/content/weather_by_station.csv', index=False)

with sqlite3.connect('/content/weather.db') as connection:
  stations.to_sql(
      'stations', connection, index=False, if_exists = 'replace'
  )
```

Comments

this module walks through the initial yet crucial steps of any data analysis project - collecting data. By focusing on the National Centers for Environmental Information (NCEI) API, it underscores the importance of accessing and retrieving data from APIs, a common source of data in many fields, from meteorology to social media analytics. The example of collecting daily weather data for New York City throughout 2018 exemplifies a practical scenario where automated data collection is not just convenient but necessary due to the volume of data.

Conclusion

For this module we use looping through dates to gather weather data for each day, storing this data in a structured format (CSV), and then transferring it into a SQLite database demonstrates a full pipeline from data collection to storage. This process is a cornerstone of data science workflows, preparing the ground for any subsequent analysis. Understanding these steps is crucial for anyone looking to work with time-series data or any large dataset that requires systematic collection and storage strategies.