In [ ]:
```
# This .ipynb file is edited by group 4 in IST 652 class
# Members are Kishan, Babatunde, Kapil, Hemanth Chowdary and Hang
import pandas as pd
import numpy as np
import requests
import matplotlib.pyplot as plt
```

Part 1

Dataset 1

In [ ]:
```
# Here's a dataset from UCI machine learning repository, named 'adult'
try:
    from ucimlrepo import fetch_ucirepo
except ImportError:
    %pip install ucimlrepo
# fetch dataset
adult = fetch_ucirepo(id=2)
adult=pd.concat([adult.data.features,adult.data.targets],axis=1)
adult.head()
```

In [ ]:
```
# URL:https://archive.ics.uci.edu/dataset/2/adult (need further action to download)
# Introduction: this dataset is extracted from 1994 census tract data, listing personal information like ed
# Column description: see https://cseweb.ucsd.edu/classes/sp15/cse190-c/reports/sp15/048.pdf for detailed i
adult.shape
# This dataset has 48842 rows with 15 columnss
```

Out[ ]:
```
(48842, 15)
```

Dataset 2

In [ ]:
```
# Name: Monthly state sales tax collections from data.gov
# dataset url: https://catalog.data.gov/dataset/sales-tax-collections-by-state
ds2_url='https://data.transportation.gov/api/views/3qgg-2u2a/rows.csv?accessType=DOWNLOAD'
```

```
ds2_df = pd.read_csv(ds2_url)
ds2_df.dtypes
```

Out[ ]:
```
state             object
month             object
year               int64
tax type          object
value            float64
fips state         int64
numeric month      int64
note             float64
id                object
dtype: object
```

In [ ]:
```
ds2_df.head()
```

Out[ ]:

	state	month	year	tax type	value	fips state	numeric month	note	id
0	Alabama	July	2023	motor fuel	83403043.0	1	7	NaN	1_2023_7
1	Alaska	July	2023	motor fuel	NaN	2	7	NaN	2_2023_7
2	Arizona	July	2023	motor fuel	NaN	4	7	NaN	4_2023_7
3	Arkansas	July	2023	motor fuel	48791407.0	5	7	NaN	5_2023_7
4	California	July	2023	motor fuel	754200000.0	6	7	NaN	6_2023_7

In [ ]:
```
# Introduction:
# "Monthly state sales tax collections is an experimental dataset published by the U.S. Census Bureau.
# It provides data for collections from sales taxes including motor fuel taxes. Data reported for a specifi
# Tax collections primarily rely on unaudited data collected from existing state reports or state data sou
# Secondarily, states report the data via the Quarterly Survey of State and Local Tax Revenue. Data are up

# This dataset contains information about sales tax collections across various U.S. states.
```

In [ ]:
```
ds2_df.info()
# Attributes included in this dataset are state name, month, year, tax type, and corresponding tax values.
# Some entries have missing values denoted as NaN.
# The dataset spans 2652 rows and 9 columns, with each row representing a specific sales tax record, provi
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2652 entries, 0 to 2651
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   state          2652 non-null   object 
 1   month          2652 non-null   object 
 2   year           2652 non-null   int64  
 3   tax type       2652 non-null   object 
 4   value          2151 non-null   float64
 5   fips state     2652 non-null   int64  
 6   numeric month  2652 non-null   int64  
 7   note           0 non-null      float64
 8   id             2652 non-null   object 
dtypes: float64(2), int64(3), object(4)
memory usage: 186.6+ KB
```

--This dataset contains information about sales tax collections across various U.S. states. It includes details such as state name, month, year, tax type, and corresponding values. Some entries have missing values denoted as NaN. The dataset spans 2652 rows and 9 columns, with each row representing a specific sales tax record, providing insights into tax collections for different states and time periods.

Part 2

API 1

In [ ]:
```
# API url: https://jsonplaceholder.typicode.com
# Introduction: "JSONPlaceholder is a free online REST API that you can use whenever you need some fake da
# It can be in a README on GitHub, for a demo on CodeSandbox, in code examples on Stack Overflow,
#  ...or simply to test things locally."
```

In [ ]:
```
# Sample code
base_url = "https://jsonplaceholder.typicode.com"
# This block of code request to get a list of users
users_response = requests.get(f"{base_url}/users")
users_data = users_response.json()
#This block of code request to get a list of posts by a specific user
```

```
user_id = 1
user_posts_response = requests.get(f"{base_url}/users/{user_id}/posts")
user_posts_data = user_posts_response.json()
```

In [ ]:
```
# Display a list of users as a Pandas DataFrame
users_df = pd.DataFrame(users_data)
users_df.head()
```

Out[ ]:

	id	name	username	email	address	phone	website	company
0	1	Leanne Graham	Bret	Sincere@april.biz	{'street': 'Kulas Light', 'suite': 'Apt. 556',...	1-770-736-8031 x56442	hildegard.org	{'name': 'Romaguera-Crona', 'catchPhrase': 'Mu...
1	2	Ervin Howell	Antonette	Shanna@melissa.tv	{'street': 'Victor Plains', 'suite': 'Suite 87...	010-692-6593 x09125	anastasia.net	{'name': 'Deckow-Crist', 'catchPhrase': 'Proac...
2	3	Clementine Bauch	Samantha	Nathan@yesenia.net	{'street': 'Douglas Extension', 'suite': 'Suit...	1-463-123-4447	ramiro.info	{'name': 'Romaguera-Jacobson', 'catchPhrase': ...
3	4	Patricia Lebsack	Karianne	Julianne.OConner@kory.org	{'street': 'Hoeger Mall', 'suite': 'Apt. 692',...	493-170-9623 x156	kale.biz	{'name': 'Robel-Corkery', 'catchPhrase': 'Mult...
4	5	Chelsey Dietrich	Kamren	Lucio_Hettinger@annie.ca	{'street': 'Skiles Walks', 'suite': 'Suite 351...	(254)954-1289	demarco.info	{'name': 'Keebler LLC', 'catchPhrase': 'User-c...

In [ ]:
```
# Display posts by a specific user in a DataFrame
user_posts_df = pd.DataFrame(user_posts_data)
user_posts_df.head()
```

Out[ ]:

	userId	id	title	body
0	1	1	sunt aut facere repellat provident occaecati e...	quia et suscipit\nsuscipit recusandae consequu...
1	1	2	qui est esse	est rerum tempore vitae\nsequi sint nihil repr...
2	1	3	ea molestias quasi exercitationem repellat qui...	et iusto sed quo iure\nvoluptatem occaecati om...
3	1	4	eum et est occaecati	ullam et saepe reiciendis voluptatem adipisci\...
4	1	5	nesciunt quas odio	repudiandae veniam quaerat sunt sed\nalias aut...

API 2

A description of the API you interacted with, mentioning its URL:

- API Name: NREL Alternative Fuel Stations
- URL: https://developer.nrel.gov/docs/transportation/alt-fuel-stations-v1/
- Description: The NREL Alternative Fuel Stations API provides detailed information on over 20,000 alternative fuel stations in the United States. The data includes specific locations, the types of fuel available, station names, and operating hours. This information is crucial for drivers, businesses, and service providers interested in alternative fuel (like electric, hydrogen, propane, etc.), promoting the transition to non-polluting, renewable energy sources.

Executing API Requests

In [ ]:
```
import requests
import pandas as pd
```

In [ ]:
```
api_key = "aNcdrYMCKeVhNCWnyGN4OwwBwPTQ5EsF9r2kmYzR"
url_CA_stations = f"https://developer.nrel.gov/api/alt-fuel-stations/v1.json?state=CA&api_key={api_key}"
url_NY_stations = f"https://developer.nrel.gov/api/alt-fuel-stations/v1.json?state=NY&api_key={api_key}"
```

Making request to the API

In [ ]:
```
response_CA = requests.get(url_CA_stations)
response_NY = requests.get(url_NY_stations)
```

Checking Response Status

In [ ]:
```
print("CA Status Code:", response_CA.status_code)
print("NY Status Code:", response_NY.status_code)
```

```
CA Status Code: 200
NY Status Code: 200
```

Processing the JSON Responses

In [ ]:
```
data_CA = response_CA.json()
data_NY = response_NY.json()
```

Creating Data Frames

In [ ]:
```
df_CA = pd.DataFrame(data_CA['fuel_stations'])
df_NY = pd.DataFrame(data_NY['fuel_stations'])
```

Displaying the Data

In [ ]:
```
print(df_CA.head())
print(df_NY.head())
```

```
  access_code                                    access_days_time  \
0      public  24 hours daily; Customers must set up a PG&E a...
1      public  8am-4pm M-F; Customers must set up a PG&E acco...
2      public  24 hours daily; Customers must set up a PG&E a...
3      public  24 hours daily; Customers must set up a PG&E a...
4     private                                               None

  access_detail_code cards_accepted date_last_confirmed expected_date  \
0        KEY_ALWAYS     Proprietor          2023-01-10          None
1        KEY_ALWAYS     Proprietor          2023-10-12          None
2        KEY_ALWAYS     Proprietor          2023-10-12          None
3        KEY_ALWAYS     Proprietor          2023-10-12          None
4              None           None          2023-05-03          None

  fuel_type_code         groups_with_access_code   id   open_date  ...  \
0            CNG  Public - Card key at all times  792  1995-05-15  ...
1            CNG  Public - Card key at all times  798  1995-05-15  ...
2            CNG  Public - Card key at all times  809  1992-05-15  ...
3            CNG  Public - Card key at all times  810  1992-05-15  ...
4            CNG                         Private  813  1995-05-15  ...

  rd_blended_with_biodiesel rd_max_biodiesel_level nps_unit_name  \
0                      None                  None          None
1                      None                  None          None
2                      None                  None          None
3                      None                  None          None
4                      None                  None          None

  access_days_time_fr intersection_directions_fr bd_blends_fr  \
0                None                       None         None
1                None                       None         None
2                None                       None         None
3                None                       None         None
4                None                       None         None

        groups_with_access_code_fr ev_pricing_fr federal_agency  \
0  Public - Carte-clé en tout temps          None            NaN
1  Public - Carte-clé en tout temps          None            NaN
2  Public - Carte-clé en tout temps          None            NaN
3  Public - Carte-clé en tout temps          None            NaN
4                           Privé          None            NaN
```

```
   ev_network_ids
0             NaN
1             NaN
2             NaN
3             NaN
4             NaN

[5 rows x 73 columns]
  access_code                                    access_days_time  \
0      public  24 hours daily; call 866-809-4869 for Clean En...
1      public  24 hours daily; call 866-809-4869 for Clean En...
2      public  8am-6pm M-F; call 718-204-4048 to arrange for ...
3      public  8am-8pm M-F; call 718-204-4048 to arrange for ...
4      public  7am-11pm M-F, 7am-3pm Sat-Sun; call 718-204-40...

   access_detail_code                                      cards_accepted  \
0  CREDIT_CARD_ALWAYS       CleanEnergy D FuelMan M V Voyager Wright_Exp
1  CREDIT_CARD_ALWAYS  CleanEnergy D FleetOne FuelMan M Proprietor V ...
2         KEY_ALWAYS                                                None
3         KEY_ALWAYS                                                None
4         KEY_ALWAYS                                                None

  date_last_confirmed expected_date fuel_type_code  \
0          2023-09-14          None            CNG
1          2022-12-13          None            CNG
2          2023-10-12          None            CNG
3          2023-10-12          None            CNG
4          2023-10-12          None            CNG

             groups_with_access_code   id   open_date  ...  \
0  Public - Credit card at all times  108  2016-07-15  ...
1  Public - Credit card at all times  112  1988-01-15  ...
2     Public - Card key at all times  129  1998-01-15  ...
3     Public - Card key at all times  130  1996-12-03  ...
4     Public - Card key at all times  132  1998-01-15  ...

  rd_blended_with_biodiesel rd_max_biodiesel_level nps_unit_name  \
0                      None                  None          None
1                      None                  None          None
2                      None                  None          None
3                      None                  None          None
4                      None                  None          None
```

```
  access_days_time_fr intersection_directions_fr bd_blends_fr  \
0                None                       None         None
1                None                       None         None
2                None                       None         None
3                None                       None         None
4                None                       None         None

              groups_with_access_code_fr ev_pricing_fr federal_agency  \
0  Public - Carte de crédit en tout temps          None            NaN
1  Public - Carte de crédit en tout temps          None            NaN
2        Public - Carte-clé en tout temps          None            NaN
3        Public - Carte-clé en tout temps          None            NaN
4        Public - Carte-clé en tout temps          None            NaN

   ev_network_ids
0             NaN
1             NaN
2             NaN
3             NaN
4             NaN

[5 rows x 73 columns]
```

Cleaning and Ogranizing

In [ ]:
```
# Example: Selecting only a few columns to display
columns_to_display = ['station_name', 'street_address', 'fuel_type_code', 'access_days_time']
print(df_CA[columns_to_display].head())
print(df_NY[columns_to_display].head())
```

```
                          station_name      street_address fuel_type_code  \
0   PG&E - Grass Valley Service Center  788 Taylorville Rd            CNG
1     PG&E - Santa Cruz Service Center         615 7th Ave            CNG
2        PG&E - Salinas Service Center     390 E Alisal St            CNG
3     PG&E - San Carlos Service Center  275 Industrial Way            CNG
4  PG&E - San Francisco Service Center       536 Treat Ave            CNG

                                  access_days_time
0  24 hours daily; Customers must set up a PG&E a...
1  8am-4pm M-F; Customers must set up a PG&E acco...
2  24 hours daily; Customers must set up a PG&E a...
3  24 hours daily; Customers must set up a PG&E a...
4                                               None
                             station_name          street_address  \
0  Clean Energy - Greenpoint - National Grid         287 Maspeth Ave
1                  Canarsie - National Grid         8424 Ditmas Ave
2       Con Edison - Van Nest Service Center      1615 Bronxdale Ave
3            Con Edison - Rye Service Center  178 Theodore Fremd Ave
4  Con Edison - College Point Service Center         124-15 31st Ave

  fuel_type_code                                    access_days_time
0            CNG  24 hours daily; call 866-809-4869 for Clean En...
1            CNG  24 hours daily; call 866-809-4869 for Clean En...
2            CNG  8am-6pm M-F; call 718-204-4048 to arrange for ...
3            CNG  8am-8pm M-F; call 718-204-4048 to arrange for ...
4            CNG  7am-11pm M-F, 7am-3pm Sat-Sun; call 718-204-40...
```

b. The API requests that you submitted and how you submitted them:

1. First API Request:

- Purpose: Retrieve data on alternative fuel stations in California.
- Request URL: https://developer.nrel.gov/api/alt-fuel-stations/v1.json?state=CA&api_key=%7Bapi_key%7D
- Method: Used the requests library in Python to send an HTTP GET request to the API's endpoint with my API key and specified parameters (state=CA).
- Response: The API returned a JSON object containing data on various fuel stations in California, including their names, addresses, types of fuel available, and access times.

2. Second API Request:

- Purpose: Retrieve data on alternative fuel stations in New York.
- Request URL: https://developer.nrel.gov/api/alt-fuel-stations/v1.json?state=NY&api_key=%7Bapi_key%7D
- Method: Similarly, I used the requests library to send a GET request with parameters (state=NY) to the same endpoint.
- Response: The API returned a JSON object specific to New York, with details on the fuel stations located within the state.

Further Data Processing:

After obtaining the responses, I used the pandas library in Python for data manipulation and analysis. I transformed the JSON response into a pandas DataFrame to facilitate easier data handling and visualization. To make the information reader-friendly and concise, I selected specific columns for display, giving anyone reviewing the data a clear view of the station's essential details without overwhelming them with the complete dataset.

The key columns displayed were:

- 'station_name'
- 'street_address'
- 'fuel_type_code'
- 'access_days_time'

These columns were chosen to provide a quick overview of each station's identity, location, services offered, and operational schedule, critical for potential consumers seeking fueling services.

This formatted response directly addresses the assignment's textual questions and provides context surrounding the code you've written. It ensures that the person grading or reviewing your work understands the logic and purpose behind each section of your code. Make sure to include this explanatory text in your notebook, typically in markdown cells, to narrate the story of what you're doing and why.