03. Evaluate Data Quality Issues in the Data Provided

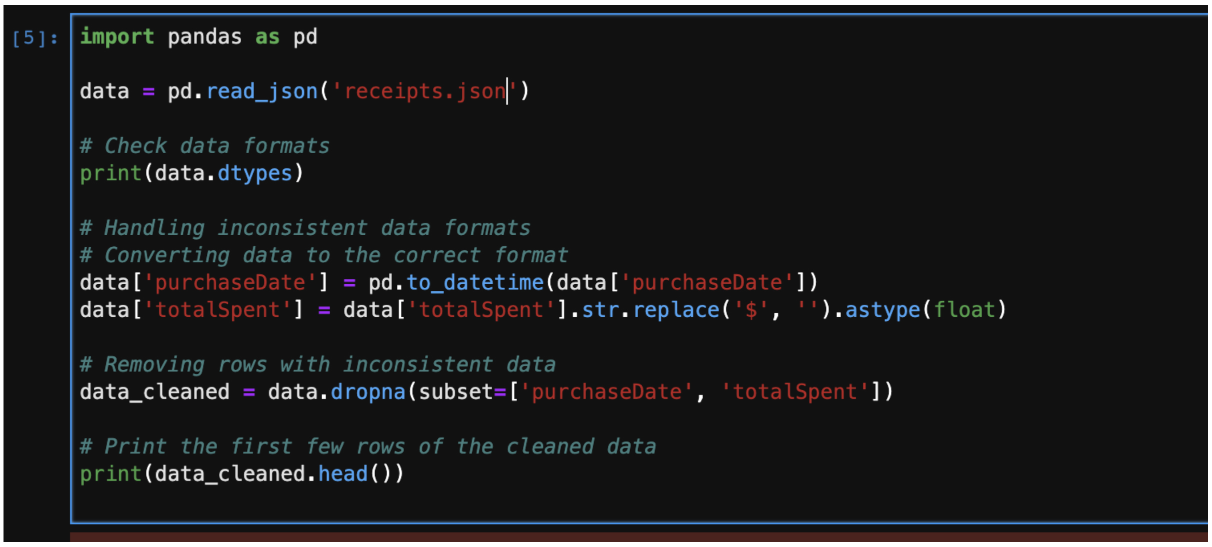
Since I encountered issues to run the JSON file, the screenshots of which I've uploaded as "Error Screenshots", I've theoretically answered the data quality issues that might arrive.

1. One common potential data quality issue that can be easily identified using Python is missing or null values in the dataset;

After identifying the missing values, there are several options for handling them:

- remove rows with missing values using the dropna() function.
- fill the missing values with a specified value (e.g., 0) using the fillna() function.
- interpolates missing values based on neighboring values using the interpolate() function.
- use more advanced imputation methods, such as mean, median, or machine learning-based imputation techniques.

Example of using python functions and libraries to remove null/missing values-



```
[5]: import pandas as pd

     data = pd.read_json('receipts.json')

     # Check data formats
     print(data.dtypes)

     # Handling inconsistent data formats
     # Converting data to the correct format
     data['purchaseDate'] = pd.to_datetime(data['purchaseDate'])
     data['totalSpent'] = data['totalSpent'].str.replace('$', '').astype(float)

     # Removing rows with inconsistent data
     data_cleaned = data.dropna(subset=['purchaseDate', 'totalSpent'])

     # Print the first few rows of the cleaned data
     print(data_cleaned.head())
```

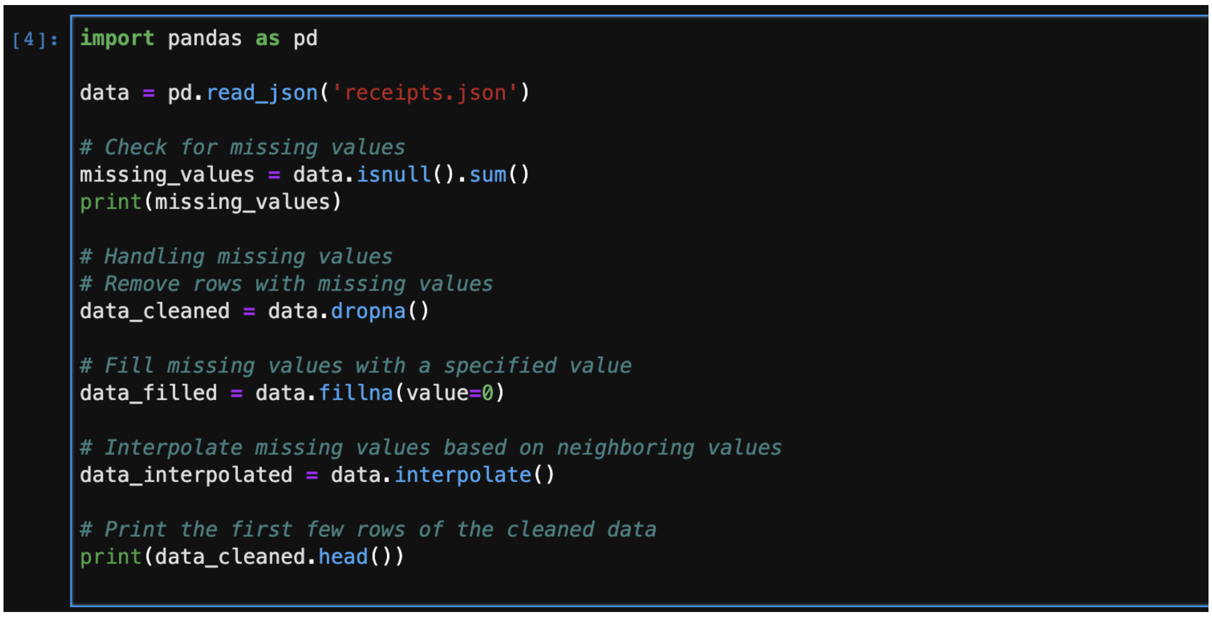
2. Another common data quality issue is inconsistent or incorrect data formats within the dataset.

Considering this coding challenge itself for example, I encountered data format issues due to JSON file not being proper, and file extraction issues from the zipped file downloaded via AWS. For this we could have made use of consistent and robust methods to convert into JSON or chosen CSV for easier flexibility through platforms, and uploaded accurately over the AWS buckets. Or, in some cases, date values that are stored in different formats or numeric values that contain non-numeric characters. We can identify and handle inconsistent data formats using Python;

- making use of various data format conversion functions like I tried using json_to_csv created under the JSON library, and other functions like to_csv under Pandas library

- converting the data to the correct format using functions like pd.to_datetime() to convert date strings to datetime objects, and astype() to convert numeric values to the desired data type
- removing rows with inconsistent data by using the dropna() function, specifying the subset of columns that need to have valid values

Example of using python functions and libraries to make the formats of both data and values consistent-



```
[4]: import pandas as pd

data = pd.read_json('receipts.json')

# Check for missing values
missing_values = data.isnull().sum()
print(missing_values)

# Handling missing values
# Remove rows with missing values
data_cleaned = data.dropna()

# Fill missing values with a specified value
data_filled = data.fillna(value=0)

# Interpolate missing values based on neighboring values
data_interpolated = data.interpolate()

# Print the first few rows of the cleaned data
print(data_cleaned.head())
```