Creating a DataFrame from Scratch

In this notebook, we'll explore some common methods to construct a dataframe from scratch.

For clarity we'll convert this data into different data structures and then create dataframes out of them.

```
| Name    | Age  | Department  |
|---------|------|-------------|
| Alice   | 25   | Marketing   |
| Bob     | 32   | Engineering |
| Charlie | 28   | Sales       |
| Denise  | 35   | Finance     |
```

Method 1: List

We can create a list of lists, where

- each nested list holds the data of row in the required order
- and another list to hold the name of columns.

```
import pandas as pd

data = [
    ['Alice', 25, 'Marketing'],
    ['Bob', 32, 'Engineering'],
    ['Charlie', 28, 'Sales'],
    ['Denise', 35, 'Finance']
]

column_names = ['Name', 'Age', 'Department']
```

Now we can use the `DataFrame` class of Pandas to generate the dataframe out of these lists.

```
df = pd.DataFrame(data, columns=column_names)
df
```

	Name	Age	Department
0	Alice	25	Marketing
1	Bob	32	Engineering
2	Charlie	28	Sales
3	Denise	35	Finance

Method 2: Dictionary

For this we can store the data column-wise, where

- each key represent the name of column and
- corresponding values is a list holds the data of that column.

```
data = {
    'Name': ['Alice', 'Bob', 'Charlie', 'Denise'],
    'Age': [25, 32, 28, 35],
    'Department': ['Marketing', 'Engineering', 'Sales', 'Finance']
}
```

```
df = pd.DataFrame(data)
df
```

	Name	Age	Department
0	Alice	25	Marketing
1	Bob	32	Engineering
2	Charlie	28	Sales
3	Denise	35	Finance

Method 3: Series in Pandas

Here we create a series of each column individually and then pass them within `pd.DataFrame` as a values to dictionary.

```
name_series = pd.Series(['Alice', 'Bob', 'Charlie', 'Denise'])
age_series = pd.Series([25, 32, 28, 35])
department_series = pd.Series(['Marketing', 'Engineering', 'Sales', 'Finance'])
```

```
df = pd.DataFrame({
    'Name': name_series,
    'Age': age_series,
    'Department': department_series
})
df
```

	Name	Age	Department
0	Alice	25	Marketing
1	Bob	32	Engineering
2	Charlie	28	Sales
3	Denise	35	Finance

Method 4: Numpy Arrays

This is very similar to the nested lists method.

```
data = np.array([
    ['Alice', 25, 'Marketing'],
    ['Bob', 32, 'Engineering'],
    ['Charlie', 28, 'Sales'],
    ['Denise', 35, 'Finance']
])

column_names = ['Name', 'Age', 'Department']
```

```
df = pd.DataFrame(data, columns=column_names)
df
```

	Name	Age	Department
0	Alice	25	Marketing
1	Bob	32	Engineering
2	Charlie	28	Sales
3	Denise	35	Finance

General methods for reading files -

We can create dataframe from the files containing data in different formats like csv, excel sheets, json, etc.

From CSV file

This is how our data looks in a csv file:

```
Name,Age,Department
Alice,25,Marketing
Bob,32,Engineering
Charlie,28,Sales
Denise,35,Finance
```

We can read through this and directly pass the path of corresponding csv file to `pd.read_csv` method as follows.

```
# creating .csv file with data
%%writefile data.csv

Name,Age,Department
Alice,25,Marketing
Bob,32,Engineering
Charlie,28,Sales
Denise,35,Finance
```

```
Overwriting data.csv
```

```
pd.read_csv("data.csv")
```

	Name	Age	Department
0	Alice	25	Marketing
1	Bob	32	Engineering
2	Charlie	28	Sales
3	Denise	35	Finance

From JSON file

Here is our original data in json format:

```
{'Name': {'0': 'Alice', '1': 'Bob', '2': 'Charlie', '3': 'Denise'},
 'Age': {'0': '25', '1': '32', '2': '28', '3': '35'},
 'Department': {'0': 'Marketing',
  '1': 'Engineering',
  '2': 'Sales',
  '3': 'Finance'}}
```

We can directly store this in a .json file and read it from that using the `pd.read_json` method.

```
# creating .csv file with data
%%writefile data.json

{"Name":{"0":"Alice","1":"Bob","2":"Charlie","3":"Denise"},"Age":{"0":"25","1":"32","2":"28","3":"35"},"Department":{"0":"Marketing","1":"Engineering","2":
```

```
Writing data.json
```

```
pd.read_json("data.json")
```

	Name	Age	Department
0	Alice	25	Marketing
1	Bob	32	Engineering
2	Charlie	28	Sales
3	Denise	35	Finance

There are many more methods to load data into a Pandas dataframe.

Refer this: https://pandas.pydata.org/docs/index.html
