# Python - Pandas

By Menna Jaheen

Pandas is **a Python library used for working with data sets**. It has functions for analyzing, cleaning, exploring, and manipulating data.

## Why Use Pandas?

- Pandas allows us to analyze big data and make conclusions based on statistical theories.
- Pandas can clean messy data sets, and make them readable and relevant.
- Relevant data is very important in data science.

## What Can Pandas Do?

### Pandas gives you answers about the data. Like:

- Is there a correlation between two or more columns?
- What is average value?
- Max value?
- Min value?

---

### In the following exercise I wrote some comments explaining each sell

[1]:
```
import pandas as pd
```

[2]:
```
df = pd.read_csv('survey_results_public.csv')
```

[3]:
```
#data frames are the back bone of pandas and they are just some rows and columns of data
#If we're using native python and not pandas , we can represent df using dictionaries and lists
#we can simply create a dicionary with values and keys , then import the pandas library and type this :
#df = pd.DataFrame(your dicionary name)
df
```

[4]:
```
df.shape # it specifies the number of rows and columns
```

[4]:
```
(65437, 114)
```

[5]:
```
df.info() # details about data : number of rows , columns , their datatypes
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 65437 entries, 0 to 65436
Columns: 114 entries, ResponseId to JobSat
dtypes: float64(13), int64(1), object(100)
memory usage: 56.9+ MB
```

[6]:
```
#If you want to display all of the columns :
#114 ----> total number of columns
pd.set_option('display.max_columns',114)
```

[15]:
```
df.shape # it specifies the number of rows and columns
```

[15]:
```
(65437, 114)
```

[16]:
```
df.info() # details about data : number of rows , columns , their datatypes
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 65437 entries, 0 to 65436
Columns: 114 entries, ResponseId to JobSat
dtypes: float64(13), int64(1), object(100)
memory usage: 56.9+ MB
```

[17]:
```
#If you want to display all of the columns :
#114 ----> total number of columns
pd.set_option('display.max_columns',114)
```

[18]:
```
# If we want to display only the first five rows of the df :
#you can also pass a value to the head( ) for the number of columns
df.head(5)
```

## Key Components:

### DataFrame

- A two-dimensional labeled data structure with columns of potentially different types.
- **Creating a DataFrame:**

```
df = pd.DataFrame({
    'A': [1, 2, 3],
    'B': [4, 5, 6]
})
```

- **Accessing data:**

```
df['A']  # Accesses the column 'A'
df.loc[0]  # Accesses the first row
```

## Basic Functions

1. **Reading and Writing Data:**
   - **Read CSV File:**

     ```
     df = pd.read_csv('file.csv')
     ```

   - **Write DataFrame to CSV File:**

     ```
     df.to_csv('file.csv', index=False)
     ```

2. **Data Exploration:**
   - **View First Few Rows:**

     ```
     df.head()  # Shows the first 5 rows
     ```

   - **View Last Few Rows:**

     ```
     df.tail()  # Shows the last 5 rows
     ```

   - **DataFrame Info:**

     ```
     df.info()  # Displays summary information about the Dat
     ```

```
aFrame
```

- **Descriptive Statistics:**

```
df.describe()  # Provides summary statistics for numeri
cal columns
```

3. **Data Selection and Filtering:**
   - **Select Columns:**

```
df['A']  # Selects column 'A'
```

   - **Select Rows by Index:**

```
df.loc[0]  # Selects the row with index 0
```

   - **Conditional Filtering:**

```
df[df['A'] > 2]  # Filters rows where the value in colu
mn 'A' is greater than 2
```

4. **Data Manipulation:**
   - **Add New Column:**

```
df['C'] = [7, 8, 9]
```

   - **Drop Columns:**

```
df.drop('C', axis=1, inplace=True)
```

   - **Fill Missing Values:**

```
df.fillna(value=0, inplace=True)
```

- **Drop Missing Values:**

```
df.dropna(inplace=True)
```