

01 Data_manipulation_with_pandas

Pandas is the world's most popular library, used for everything from data manipulation to data analysis. How to manipulate dataframes, extracting, filtering and transforming real-world datasets for analysis were shown in this course.

Course Outline

1. **Chapter 1: DataFrames**
 - Sorting and Subsetting
 - Creating new columns
2. **Chapter 2: Aggregating Data**
 - Summary statistics
 - Counting
 - Grouped summary statistics
3. **Chapter 3: Slicing and Indexing Data**
 - Subsetting using slicing
 - Indexes and subsetting using indexes
4. **Chapter 4: Creating and Visualizing Data**
 - Plotting
 - Handling missing data
 - Reading data into a DataFrame

From <https://github.com/ishtiakrongon/Datacamp-Data_manipulation_with_pandas/tree/main>

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.

;

Data Science: is a branch of computer science where we study how to store, use and analyze data for deriving information from it.

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?

Pandas are also able to delete rows that are not relevant, or contains wrong values, like empty or NULL values. This is called *cleaning* the data.

02 Pandas Series

A pandas series is like a column in a table. It is a one dimensional array holding data of any type.

```
# Create a simple Pandas Series from a list:
import pandas as pd
a = [1, 7, 2]
myvar = pd.Series(a)
print(myvar)
```

```
0    1
1    7
2    2
dtype: int64
```

```
[4] # printing the first value at 0 index in the series
print(myvar[2])
```

```
2
```

```
[8] # converting a dictionary (key:value pair) into a series
import pandas as pd

calories = {"day1": 420, "day2": 380, "day3": 390}

myvar = pd.Series(calories, index = ["day1", "day2"])

print(myvar)
```

```
day1    420
day2    380
dtype: int64
```

```
import pandas as pd

revisiontime = { "saturday": 4, "sunday": 3, "monday": 4}

myvar = pd.Series(revisiontime, index = ["saturday"])

print(myvar)
```

```
saturday    4
dtype: int64
```

03 Pandas DataFrame

A Pandas DataFrame is a 2 dimensional data structure, like a 2 dimensional array, or a table with rows and columns.

Create a simple panda dataframe:

```
import pandas as pd

data = {
    "calories": [420, 380, 390],
    "duration": [50, 40, 45]
}

# load data into a DataFrame object:
df = pd.DataFrame(data)

print(df)
```

	calories	duration
0	420	50
1	380	40
2	390	45

Locate Row:

Pandas use the loc attribute to return one or more specified row(s)

```
[2] import pandas as pd

data = {
    "calories": [420, 380, 390],
    "duration": [50, 40, 45]
}

# load data into a DataFrame object:
df = pd.DataFrame(data)

print(df)

print(df.loc[0])
```

	calories	duration
0	420	50
1	380	40
2	390	45

calories 420
duration 50
Name: 0, dtype: int64

Use a list of index. Return row 0 and 1:



```
# use a list of indexes
import pandas as pd

data = {
    "calories": [420, 380, 390],
    "duration": [50, 40, 45]
}

df = pd.DataFrame(data)

print(df.loc[[0, 1]])
```

	calories	duration
0	420	50
1	380	40

04 Pandas Reading Files- CSV, JSON

Replace 'path/to/homelessness.csv' with the path to your csv file:

```
import pandas as pd

# Replace 'path/to/homelessness.csv' with the path to your csv file
df = pd.read_csv('/content/drive/MyDrive/Data/homelessness.csv')

# Print the first 5 rows of the DataFrame
print(df.head())
```

	Duration	Pulse	Maxpulse	Calories
0	60	110	130	409.1
1	60	117	145	479.0
2	60	103	135	340.0
3	45	109	175	282.4
4	45	117	148	406.0

Panda Analysing Data

```
#Print information about the data:
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 169 entries, 0 to 168
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype  
---  -
0    Duration    169 non-null    int64  
1    Pulse       169 non-null    int64  
2    Maxpulse    169 non-null    int64  
3    Calories    164 non-null    float64
dtypes: float64(1), int64(3)
memory usage: 5.4 KB
None
```

The `info()` method also tells us how many Non-Null values there are present in each column, and in our data set it seems like there are 164 of 169 Non-Null values in the "Calories" column.

Which means that there are 5 rows with no value at all, in the "Calories" column, for whatever reason.

Empty values, or Null values, can be bad when analyzing data, and you should consider removing rows with empty values. This is a step towards what is called *cleaning data*, and you will learn more about that in the next chapters.

06 Data Cleaning

Data cleaning means fixing bad data in your data set.

Bad data could be:

- Empty cells
- Data in wrong format
- Wrong data
- Duplicates

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
18	45	'2020/12/18'	90	112	NaN
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
22	45	NaN	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	2020/12/26	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	NaN
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

The data set contains some empty

cells ("Date" in row 22, and "Calories" in row 18 and 28).

The data set contains the wrong format ("Date" in row 26).

The data set contains wrong data ("Duration" in row 7).

The data set contains duplicates (row 11 and 12).



```
#Load the CSV into a DataFrame:  
# use to_string() to print the entire DataFrame.  
import pandas as pd  
  
df = pd.read_csv('/content/drive/MyDrive/Data Analytics Bootcamp No  
  
print(df.to_string())  
#Keep an eye on the NaN value i.e. empty values NaN= Not a Number
```

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
18	45	'2020/12/18'	90	112	NaN
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
22	45	NaN	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	NaN
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

Drop rows that contain empty cells

```
new_df = df.dropna()
```

```
# Return a new Data Frame with no empty cells:

import pandas as pd

df = pd.read_csv('/content/drive/MyDrive/Data Analytics Bootcamp Notes/WK 10 Pan

new_df = df.dropna()

print(new_df.to_string())
# result show that some rows have been removed (row 18, 22 and 28).

#These rows had cells with empty values.
```

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

✓ 0s completed at 16:20

Replace null values with 130:

```
df.fillna(130, inplace = True)
```

```

# Replace NULL values with the number 130:

import pandas as pd

df = pd.read_csv('/content/drive/MyDrive/Data Analytics Bootcamp Notes/WK 10 Panda,

df.fillna(130, inplace = True)
print(df.to_string())

#Notice in the result: empty cells got the value 130 (in row 18, 22 and 28).

```

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
18	45	'2020/12/18'	90	112	130.0
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
22	45	130	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	130.0
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

Drop missing date:
Row 22 was removed.

```

import pandas as pd

df = pd.read_csv('/content/drive/MyDrive/Data Analytics Bootcamp')

df['Date'] = pd.to_datetime(df['Date'])

df.dropna(subset=['Date'], inplace = True)

print(df.to_string())

```

```

➡

```

	Duration	Date	Pulse	Maxpulse	Calories
0	60	2020-12-01	110	130	409.1
1	60	2020-12-02	117	145	479.0
2	60	2020-12-03	103	135	340.0
3	45	2020-12-04	109	175	282.4
4	45	2020-12-05	117	148	406.0
5	60	2020-12-06	102	127	300.0
6	60	2020-12-07	110	136	374.0
7	450	2020-12-08	104	134	253.3
8	30	2020-12-09	109	133	195.1
9	60	2020-12-10	98	124	269.0
10	60	2020-12-11	103	147	329.3
11	60	2020-12-12	100	120	250.7
12	60	2020-12-12	100	120	250.7
13	60	2020-12-13	106	128	345.3
14	60	2020-12-14	104	132	379.3
15	60	2020-12-15	98	123	275.0
16	60	2020-12-16	98	120	215.2
17	60	2020-12-17	100	120	300.0
18	45	2020-12-18	90	112	NaN
19	60	2020-12-19	103	123	323.0
20	45	2020-12-20	97	125	243.0
21	60	2020-12-21	108	131	364.2
23	60	2020-12-23	130	101	300.0
24	45	2020-12-24	105	132	246.0
25	60	2020-12-25	102	126	334.5
26	60	2020-12-26	100	120	250.0
27	60	2020-12-27	92	118	241.0
28	60	2020-12-28	103	132	NaN
29	60	2020-12-29	100	132	280.0
30	60	2020-12-30	102	129	380.3
31	60	2020-12-31	92	115	243.0

Cleaning Data of Wrong Format

Cells with data of wrong format can make it difficult, or even impossible, to analyze data.

To fix it, you have two options: remove the rows, or convert all cells in the columns into the same format.

we have two cells with the wrong format

Pandas has a `to_datetime()` method for this:

```
df['Date'] = pd.to_datetime(df['Date'])
```

This line focuses on a specific column called "Date" and changes its format to datetime, which is a special way of representing dates and times in Python, so you can do cool things with them later.

```
✓ [15] import pandas as pd
js      df['Date'] = pd.to_datetime(df['Date'])
        print(df.to_string())
        df = pd.read_csv('/content/drive/MyDrive/WK 10 Panda/Resource
```

	Duration	Date	Pulse	Maxpulse	Calories
0	60	2020-12-01	110	130	409.1
1	60	2020-12-02	117	145	479.0
2	60	2020-12-03	103	135	340.0
3	45	2020-12-04	109	175	282.4
4	45	2020-12-05	117	148	406.0
5	60	2020-12-06	102	127	300.0
6	60	2020-12-07	110	136	374.0
7	450	2020-12-08	104	134	253.3
8	30	2020-12-09	109	133	195.1
9	60	2020-12-10	98	124	269.0
10	60	2020-12-11	103	147	329.3
11	60	2020-12-12	100	120	250.7
12	60	2020-12-12	100	120	250.7
13	60	2020-12-13	106	128	345.3
14	60	2020-12-14	104	132	379.3
15	60	2020-12-15	98	123	275.0
16	60	2020-12-16	98	120	215.2
17	60	2020-12-17	100	120	300.0
18	45	2020-12-18	90	112	NaN
19	60	2020-12-19	103	123	323.0
20	45	2020-12-20	97	125	243.0
21	60	2020-12-21	108	131	364.2
22	45	NaT	100	119	282.0
23	60	2020-12-23	130	101	300.0
24	45	2020-12-24	105	132	246.0
25	60	2020-12-25	102	126	334.5
26	60	2020-12-26	100	120	250.0
27	60	2020-12-27	92	118	241.0
28	60	2020-12-28	103	132	NaN
29	60	2020-12-29	100	132	280.0
30	60	2020-12-30	102	129	380.3
31	60	2020-12-31	92	115	243.0

Fixing wrong data

in row 7, the duration is 450, but for all the other rows the duration is between 30 and 60.

1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
**7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
18	45	'2020/12/18'	90	112	NaN
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
22	45	NaN	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	NaN
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

'df.loc[7, 'Duration'] = 45':

```
[20] import pandas as pd

df = pd.read_csv('/content/drive/MyDrive/WK 10 Panda/Resources-2024020

df.loc[7, 'Duration'] = 45

print(df.to_string())
```

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	45	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
18	45	'2020/12/18'	90	112	NaN
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
22	45	NaN	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	NaN
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

Discovering duplicates

Row 11 and 12 are duplicates

0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
**11	60	'2020/12/12'	100	120	250.7
**12	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
18	45	'2020/12/18'	90	112	NaN
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
22	45	NaN	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	NaN
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

The `duplicated()` method returns a Boolean values for each row:

Returns True for every row that is a duplicate, otherwise False:

```
print(df.duplicated())
```

```
0    False
1    False
2    False
3    False
4    False
5    False
6    False
7    False
8    False
9    False
10   False
11   False
12    True
13   False
14   False
15   False
```

Removing duplicates

`drop_duplicates()`

The (`inplace = True`) will make sure that the method does NOT return a *new* DataFrame, but it will remove all duplicates from the *original* DataFrame.

```
# 2. Load the data from a CSV file (open the data book)
df = pd.read_csv('/content/drive/MyDrive/WK 10 Panda/Resc

# 3. Remove duplicate rows (keep only unique pages in the
df.drop_duplicates(inplace=True)

# 4. Display the updated DataFrame (show the book without
print(df.to_string())
```

	Duration	Date	Pulse	Maxpulse	Calories
0	60	'2020/12/01'	110	130	409.1
1	60	'2020/12/02'	117	145	479.0
2	60	'2020/12/03'	103	135	340.0
3	45	'2020/12/04'	109	175	282.4
4	45	'2020/12/05'	117	148	406.0
5	60	'2020/12/06'	102	127	300.0
6	60	'2020/12/07'	110	136	374.0
7	450	'2020/12/08'	104	134	253.3
8	30	'2020/12/09'	109	133	195.1
9	60	'2020/12/10'	98	124	269.0
10	60	'2020/12/11'	103	147	329.3
11	60	'2020/12/12'	100	120	250.7
13	60	'2020/12/13'	106	128	345.3
14	60	'2020/12/14'	104	132	379.3
15	60	'2020/12/15'	98	123	275.0
16	60	'2020/12/16'	98	120	215.2
17	60	'2020/12/17'	100	120	300.0
18	45	'2020/12/18'	90	112	NaN
19	60	'2020/12/19'	103	123	323.0
20	45	'2020/12/20'	97	125	243.0
21	60	'2020/12/21'	108	131	364.2
22	45	NaN	100	119	282.0
23	60	'2020/12/23'	130	101	300.0
24	45	'2020/12/24'	105	132	246.0
25	60	'2020/12/25'	102	126	334.5
26	60	20201226	100	120	250.0
27	60	'2020/12/27'	92	118	241.0
28	60	'2020/12/28'	103	132	NaN
29	60	'2020/12/29'	100	132	280.0
30	60	'2020/12/30'	102	129	380.3
31	60	'2020/12/31'	92	115	243.0

```
import pandas as pd

df = pd.read_csv('/content/drive/MyDrive/WK 10
Panda/Resources-20240202T091819Z-001/Resources/data2.csv')

#to print the entire DataFrame.
print(df.to_string())

#Return a new Data Frame with no empty cells:
new_df = df.dropna()
print(new_df.to_string())
```



```
#Remove all rows with NULL values:
df.dropna(inplace = True)
print(df.to_string())

# Replace NULL values in the "Calories" columns with the number 130.
This operation inserts 130 in empty cells in the "Calories" column (row
18 and 28).
df["Calories"].fillna(130, inplace = True)
print(df.to_string())

# 1 Convert a column to datetime format
df['Date'] = pd.to_datetime(df['Date'])
# 2 Remove rows with missing dates (tidy up the book)
df.dropna(subset=['Date'], inplace=True)
print(df.to_string())
#3 Display the updated DataFrame (show the cleaned book)
print(df.to_string())

# Update a specific value in the DataFrame (make a change in the book)
df.loc[7, 'Duration'] = 45
print(df.to_string())

#1 Check for duplicate rows (find identical pages in the book)
print(df.duplicated())
# 2 Remove duplicate rows (keep only unique pages in the book)
df.drop_duplicates(inplace=True)
print(df.to_string())
```

07 Correlation and Plotting

<https://www.machinelearningplus.com/plots/top-50-matplotlib-visualizations-the-master-plots-python/#1.-Scatter-plot>

08 Flight Delay

Problem: Our dataset contains data from the Bureau of the United States of transportation about the arrival, delay, and cancellation of domestic flights from July 2019 to July 2022. We will investigate and highlight which airlines and airports have the most delay and cancellation over time. We will also analyse the causes of the delays and cancellation. At last, we will see whether COVID-19 pandemic had an impact on the overwall flight cancellations.

1. Mount the drive

```
From google.colab import drive
drive.mount('/content/drive')
```

2. Import pandas as library

```
rom google.colab import drive
drive.mount('/content/drive')
```

3. Explore the dataset, print first 5 rows

```
df1.head()
```

```
df1.head()
```

```
year month
```

```
0 2022
```

```
1 2022
```

```
2 2022
```

```
3 2022
```

```
4 2022
```

4. Explore the last 5 rows

```
df1.tail()
```

```
[ ] df1.tail()
```

	year	month	carrier
1675	2022	7	Y
1676	2022	7	Y
1677	2022	7	Y
1678	2022	7	Y
1679	2022	7	Y

- Combine all three files into one document using `.concat()` function

```
df= pd.concat([df1, df2, df3], ignore_index=True)
```

- Explore first few rows of the whole dataset

```
df.head(10)
```

	year	month	carrier	carrier_name	airport	airport_name	ar
0	2022	7	9E	Endeavor Air Inc.	ABE	Allentown/Bethlehem/Easton, PA: Lehigh Valley ...	
1	2022	7	9E	Endeavor Air Inc.	ABY	Albany, GA: Southwest Georgia Regional	
2	2022	7	9E	Endeavor Air Inc.	ACK	Nantucket, MA: Nantucket Memorial	
3	2022	7	9E	Endeavor Air Inc.	AEX	Alexandria, LA: Alexandria International	
4	2022	7	9E	Endeavor Air Inc.	AGS	Augusta, GA: Augusta Regional at Bush Field	
5	2022	7	9E	Endeavor Air Inc.	ALB	Albany, NY: Albany International	
6	2022	7	9E	NaN	ATL	Atlanta, GA: Hartsfield-Jackson Atlanta Intern...	
7	2022	7	9E	NaN	ATW	Appleton, WI: Appleton International	
8	2022	7	9E	Endeavor Air Inc.	AUS	Austin, TX: Austin - Bergstrom International	
9	2022	7	9E	Endeavor Air Inc.	AVL	Asheville, NC: Asheville Regional	

- Explore how many rows and columns the dataset has

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 42461 entries, 0 to 42460
Data columns (total 16 columns):
#   Column                Non-Null Count  Dtype
---  -
0   year                   42461 non-null  int64
1   month                  42461 non-null  int64
2   carrier                42461 non-null  object
3   carrier_name           42445 non-null  object
4   airport                42461 non-null  object
5   airport_name           42461 non-null  object
6   arr_flights            42354 non-null  float64
7   arr_del15              42183 non-null  float64
8   arr_cancelled          42354 non-null  float64
9   arr_diverted           42354 non-null  float64
10  arr_delay              42354 non-null  float64
11  carrier_delay           42354 non-null  float64
12  weather_delay          42354 non-null  float64
13  nas_delay              42354 non-null  float64
14  security_delay         42354 non-null  float64
15  delays                 0 non-null      float64
dtypes: float64(10), int64(2), object(4)
memory usage: 5.2+ MB
```

8. df.shape

This returns only the number of rows and columns from the dataset.

```
df.shape

(42461, 16)
```

9. Run statistics using the .describe() function

```
df.describe()
```

	year	month	arr_flights	arr_del15	arr_cancelled	arr_diverted	arr_delay	carrier_delay	weather_delay	nas_delay	security_delay	delays
count	42461.000000	42461.000000	42354.000000	42183.000000	42354.000000	42354.000000	42354.000000	42354.000000	42354.000000	42354.000000	42354.000000	0.0
mean	2020.096324	6.529521	290.053950	41.470095	9.251641	0.624357	2757.859210	1007.217996	177.938495	587.135383	6.496647	NaN
std	0.815651	3.341909	836.485771	130.737062	70.613366	3.069780	9664.845862	3405.101443	742.131883	2784.605383	45.371314	NaN
min	2019.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	NaN
25%	2020.000000	4.000000	33.000000	3.000000	0.000000	0.000000	158.000000	49.000000	0.000000	9.000000	0.000000	NaN
50%	2020.000000	7.000000	81.000000	10.000000	0.000000	0.000000	566.000000	211.000000	0.000000	80.000000	0.000000	NaN
75%	2021.000000	9.000000	194.000000	28.000000	3.000000	0.000000	1737.000000	705.000000	99.000000	282.000000	0.000000	NaN
max	2022.000000	12.000000	21873.000000	3506.000000	4951.000000	154.000000	305694.000000	119425.000000	26428.000000	200000.000000	3760.000000	NaN

- Year and months have float datatype

Data Transformation

10. Convert the datatype of month and years to strings

```
df_copy['month'].astype(str)
df_copy['year'].astype(str)
df_copy['date'].astype(str)

0      2022-7
1      2022-7
2      2022-7
3      2022-7
4      2022-7
...
42456  2020-8
42457  2020-8
42458  2020-8
42459  2020-8
42460  2020-8
Name: date, Length: 42461, dtype: object
```

date

2022-7

2022-7

2022-7

2022-7

2022-7

2022-7

2022-7

2022-7