Data Cleaning: Data cleansing or data cleaning is the process of detecting and correcting corrupt or inaccurate records from a recordset, table, or database and refers to identifying incomplete, incorrect, inaccurate or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data.

Bad data could be:

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

If we encounter such bad data in any of the columns, We will dismiss the whole row from the file. Especially, for duplicate values.

Or, we can replace bad data by taking the mean/median/mode of the other data of that column.

Data cleaning process

There are many process to clear data.

Clearing empty cell:

i) Drop the empty cell dropna(inplace = True)

ii) Fill the empty cell using specific value

iii) Fill by mean, median, mode

Data of Wrong Format

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:

i) remove the rows.

ii) convert all cells in the columns into the same format.

Wrong Data

- "Wrong data" does not have to be "empty cells" or "wrong format", it can just be wrong, like if someone registered "199" instead of "1.99".
- Sometimes you can spot wrong data by looking at the data set, because you have an expectation of what it should be.

Replacing Values

One way to fix wrong values is to replace them with something else.

Removing Rows

Another way of handling wrong data is to remove the rows that contains wrong data.

Removing Duplicates

Find the duplicates

To discover duplicates, we can use the duplicated() method.

Remove the duplicates

To remove duplicates, use the drop_duplicates() method.

Let a dataframe be,

```
import pandas as pd
df = pd.read_excel("dirtydata.xlsx")
new_df = df.dropna()
print(df)
#print(new_df)
```

```
    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
```

- To drop entire rows with null values, We can use ***.dropna()*** method. Note that there are several *"Nan"* values in lines like 18,22 and etc.

```
import pandas as pd
df = pd.read_excel("dirtydata.xlsx")
new_df = df.dropna()
print(new_df)
```

```
    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
```

- To replace a null value in the dataframe, we can use the ***".fillna("value", inplace = True)"*** method.
  Note that there are several *"Nan"* values in lines like 18,22 and etc. And now, They are replaced with 130.

```
1 import pandas as pd
2 df = pd.read_excel("dirtydata.xlsx")
3 df.fillna(130, inplace = True)
4 print(df)
```

```
    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
```

The problem with setting specific values is that it does not abide by the types of data that is in the columns.

- That is why, When we set any specific value, we should specify the columns also. We can do it using
  dataframe_name[“***Column_name***"].fillna("***Replace_value***",inplace = True)

```
import pandas as pd
df = pd.read_excel("dirtydata.xlsx")
df["Calories"].fillna(130, inplace = True)
df["Date"].fillna("3020/13/07",inplace = True)
print(df)
```

	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	3020/13/07	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

Note that there are several *“Nan”* values in lines like 18 and 28 in the calories column which is now replaced with 130.
And for date, 22th row of the date column had null value before but now it has a new date “30/13/07”

- Now, the problem with this technique is the specific value might not sync or blend in the other values of that column. If we insert a value that can be obtained using mean/median/mode on other values of that column, that value will sync in or blend in relatively better.

  We need to calculate the mean/median/mode first using this format.

  variabel _name = ***dataframe***["***Column_name***"].***technique***()

```
1 import pandas as pd
2 df = pd.read_excel("dirtydata.xlsx")
3 x = df["Calories"].mean()
4 df["Calories"].fillna(x, inplace = True)
5 df["Date"].fillna("2020/12/22",inplace = True)
6 print(df)
```

```
    Duration         Date  Pulse  Maxpulse  Calories
0         60  2020/12/01'    110       130    409.10
1         60  2020/12/02'    117       145    479.00
2         60  2020/12/03'    103       135    340.00
3         45  2020/12/04'    109       175    282.40
4         45  2020/12/05'    117       148    406.00
5         60  2020/12/06'    102       127    300.00
6         60  2020/12/07'    110       136    374.00
7        450  2020/12/08'    104       134    253.30
8         30  2020/12/09'    109       133    195.10
9         60  2020/12/10'     98       124    269.00
10        60  2020/12/11'    103       147    329.30
11        60  2020/12/12'    100       120    250.70
12        60  2020/12/12'    100       120    250.70
13        60  2020/12/13'    106       128    345.30
14        60  2020/12/14'    104       132    379.30
15        60  2020/12/15'     98       123    275.00
16        60  2020/12/16'     98       120    215.20
17        60  2020/12/17'    100       120    300.00
18        45  2020/12/18'     90       112    304.68
19        60  2020/12/19'    103       123    323.00
20        45  2020/12/20'     97       125    243.00
21        60  2020/12/21'    108       131    364.20
22        45   2020/12/22    100       119    282.00
23        60  2020/12/23'    130       101    300.00
24        45  2020/12/24'    105       132    246.00
25        60  2020/12/25'    102       126    334.50
26        60     20201226    100       120    250.00
27        60  2020/12/27'     92       118    241.00
28        60  2020/12/28'    103       132    304.68
29        60  2020/12/29'    100       132    280.00
30        60  2020/12/30'    102       129    380.30
31        60  2020/12/31'     92       115    243.00
```

  Note that there are several *“Nan”* values in lines like 18,22 and etc. And now, They are replaced with the mean value of the column.
  We need to do all mean/median/mode operations and select the best fit value.

Formatting wrong data

To fix the format of the wrong data, we need to convert the whole column into the appropriate format.

Eg: We have a date here but in the wrong format. We need to format it in a DateTime format.

	Duration	Date	Pulse	Maxpulse	Calories
0	60	2020/12/01'	110	130	409.10
1	60	2020/12/02'	117	145	479.00
2	60	2020/12/03'	103	135	340.00
3	45	2020/12/04'	109	175	282.40
4	45	2020/12/05'	117	148	406.00
5	60	2020/12/06'	102	127	300.00
6	60	2020/12/07'	110	136	374.00
7	450	2020/12/08'	104	134	253.30
8	30	2020/12/09'	109	133	195.10
9	60	2020/12/10'	98	124	269.00
10	60	2020/12/11'	103	147	329.30
11	60	2020/12/12'	100	120	250.70
12	60	2020/12/12'	100	120	250.70
13	60	2020/12/13'	106	128	345.30
14	60	2020/12/14'	104	132	379.30
15	60	2020/12/15'	98	123	275.00
16	60	2020/12/16'	98	120	215.20
17	60	2020/12/17'	100	120	300.00
18	45	2020/12/18'	90	112	304.68
19	60	2020/12/19'	103	123	323.00
20	45	2020/12/20'	97	125	243.00
21	60	2020/12/21'	108	131	364.20
22	45	2020/12/22	100	119	282.00
23	60	2020/12/23'	130	101	300.00
24	45	2020/12/24'	105	132	246.00
25	60	2020/12/25'	102	126	334.50
26	60	20201226	100	120	250.00
27	60	2020/12/27'	92	118	241.00
28	60	2020/12/28'	103	132	304.68
29	60	2020/12/29'	100	132	280.00
30	60	2020/12/30'	102	129	380.30
31	60	2020/12/31'	92	115	243.00

We can do using this syntax,

new_dataframe["***Column_name***"] = pd.to_***NewFormat***(***old_dataframe***["***Column_name***"]).dt.***date/time*** [Optional based on actually what we want to see]

Later, We can convert it into a string and print it.

```
1 import pandas as pd
2 df = pd.read_excel("dirtydata.xlsx")
3 x = df["Calories"].mean()
4 df["Calories"].fillna(x, inplace = True)
5 df["Date"].fillna("2020/12/22",inplace = True)
6 df["Date"]=pd.to_datetime(df["Date"]).dt.date
7 print(df)
```

	Duration	Date	Pulse	Maxpulse	Calories
0	60	2020-12-01	110	130	409.10
1	60	2020-12-02	117	145	479.00
2	60	2020-12-03	103	135	340.00
3	45	2020-12-04	109	175	282.40
4	45	2020-12-05	117	148	406.00
5	60	2020-12-06	102	127	300.00
6	60	2020-12-07	110	136	374.00
7	450	2020-12-08	104	134	253.30
8	30	2020-12-09	109	133	195.10
9	60	2020-12-10	98	124	269.00
10	60	2020-12-11	103	147	329.30
11	60	2020-12-12	100	120	250.70
12	60	2020-12-12	100	120	250.70

```
1 import pandas as pd
2 df = pd.read_excel("dirtydata.xlsx")
3 x = df["Calories"].mean()
4 df["Calories"].fillna(x, inplace = True)
5 df["Date"].fillna("2020/12/22",inplace = True)
6 df["Date"]=pd.to_datetime(df["Date"]).dt.time
7 print(df)
```

	Duration	Date	Pulse	Maxpulse	Calories
0	60	00:00:00	110	130	409.10
1	60	00:00:00	117	145	479.00
2	60	00:00:00	103	135	340.00
3	45	00:00:00	109	175	282.40
4	45	00:00:00	117	148	406.00
5	60	00:00:00	102	127	300.00
6	60	00:00:00	110	136	374.00
7	450	00:00:00	104	134	253.30
8	30	00:00:00	109	133	195.10
9	60	00:00:00	98	124	269.00
10	60	00:00:00	103	147	329.30
11	60	00:00:00	100	120	250.70

- If we want to replace a specific value within a row/column, We can do that using this format,

data_frame.loc[***"row_number", "column_name"***] = ***new_value***

	Duration	Date	Pulse	Maxpulse	Calories
0	60	2020-12-01	110	130	409.10
1	60	2020-12-02	117	145	479.00
2	60	2020-12-03	103	135	340.00
3	45	2020-12-04	109	175	282.40
4	45	2020-12-05	117	148	406.00
5	60	2020-12-06	102	127	300.00
6	60	2020-12-07	110	136	374.00
7	450	2020-12-08	104	134	253.30
8	30	2020-12-09	109	133	195.10
9	60	2020-12-10	98	124	269.00
10	60	2020-12-11	103	147	329.30
11	60	2020-12-12	100	120	250.70
12	60	2020-12-12	100	120	250.70
13	60	2020-12-13	106	128	345.30
14	60	2020-12-14	104	132	379.30
15	60	2020-12-15	98	123	275.00
16	60	2020-12-16	98	120	215.20
17	60	2020-12-17	100	120	300.00
18	45	2020-12-18	90	112	304.68
19	60	2020-12-19	103	123	323.00

Lets say, We want to change this ***450*** into ***45***. We need to set the exact row number (7 Here) & Column name (“Duration”)

```
import pandas as pd
df = pd.read_excel("dirtydata.xlsx")
x = df["Calories"].mean()
df["Calories"].fillna(x, inplace = True)
df["Date"].fillna("2020/12/22",inplace = True)
df["Date"]=pd.to_datetime(df["Date"]).dt.date
df.loc[7, "Duration"] = 45
print(df)
```

	Duration	Date	Pulse	Maxpulse	Calories
0	60	2020-12-01	110	130	409.10
1	60	2020-12-02	117	145	479.00
2	60	2020-12-03	103	135	340.00
3	45	2020-12-04	109	175	282.40
4	45	2020-12-05	117	148	406.00
5	60	2020-12-06	102	127	300.00
6	60	2020-12-07	110	136	374.00
7	45	2020-12-08	104	134	253.30
8	30	2020-12-09	109	133	195.10
9	60	2020-12-10	98	124	269.00
10	60	2020-12-11	103	147	329.30
11	60	2020-12-12	100	120	250.70
12	60	2020-12-12	100	120	250.70
13	60	2020-12-13	106	128	345.30
14	60	2020-12-14	104	132	379.30

- If we have duplicate data, the Entire row that is. We need to drop those duplicate rows and keep only one because of no point in having the same data over and over again.

  Before dismissing the rows, We need to identify those first. We can do that using this format,

  Dataframe.duplicated()

```
1 import pandas as pd
2 df = pd.read_excel("dirtydata.xlsx")
3 x = df["Calories"].mean()
4 df["Calories"].fillna(x, inplace = True)
5 df["Date"].fillna("2020/12/22",inplace = True)
6 df["Date"]=pd.to_datetime(df["Date"]).dt.date
7 df.loc[7, "Duration"] = 45
8 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
```

  All of them are unique except the 12th column. We don't know which column's duplicate this one is but this one is duplicate for sure

  But in this case, It is the duplicate of 11th number column.

```
 11        60  2020/12/12'    100       120     250.7
 12        60  2020/12/12'    100       120     250.7
```

- We can drop these duplicate values using

data_frame.drop_duplicates(inplace = True)

```
import pandas as pd
df = pd.read_excel("dirtydata.xlsx")
x = df["Calories"].mean()
df["Calories"].fillna(x, inplace = True)
df["Date"].fillna("2020/12/22",inplace = True)
df["Date"]=pd.to_datetime(df["Date"]).dt.date
df.loc[7, "Duration"] = 45
df.drop_duplicates(inplace = True)
print(df)
```

```
    Duration        Date  Pulse  Maxpulse  Calories
0         60  2020-12-01    110       130    409.10
1         60  2020-12-02    117       145    479.00
2         60  2020-12-03    103       135    340.00
3         45  2020-12-04    109       175    282.40
4         45  2020-12-05    117       148    406.00
5         60  2020-12-06    102       127    300.00
6         60  2020-12-07    110       136    374.00
7         45  2020-12-08    104       134    253.30
8         30  2020-12-09    109       133    195.10
9         60  2020-12-10     98       124    269.00
10        60  2020-12-11    103       147    329.30
11        60  2020-12-12    100       120    250.70
13        60  2020-12-13    106       128    345.30
14        60  2020-12-14    104       132    379.30
15        60  2020-12-15     98       123    275.00
16        60  2020-12-16     98       120    215.20
17        60  2020-12-17    100       120    300.00
18        45  2020-12-18     90       112    304.68
19        60  2020-12-19    103       123    323.00
```

We can now see there’s no longer any 12th number row.