Data cleaning means fixing bad data in your data set.

Bad data could be:

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

```
import pandas as pd

df = pd.read_csv('diabetes.csv')

print(df.to_string())
```

```
494           3    102.0          74.0           0      0.0  29.5                    0.121  32.0        0
495           7    187.0          50.0          33    392.0  33.9                    0.826  34.0        1
496           3    173.0          78.0          39    185.0  33.8                    0.970  31.0        1
497          10     94.0          72.0          18      0.0  23.1                    0.595  56.0        0
498           1    108.0          60.0          46    178.0  35.5                    0.415  24.0        0
499           5     97.0          76.0          27      0.0  35.6                    0.378  52.0        1
500           4     83.0          86.0          19      0.0  29.3                    0.317   NaN        0
501           1    114.0          66.0          36    200.0  38.1                    0.289  21.0        0
502           1    149.0          68.0          29    127.0  29.3                    0.349  42.0        1
503           5    117.0          86.0          30    105.0  39.1                    0.251  42.0        0
504           1    111.0           NaN           0      0.0  32.8                    0.265  45.0        0
505           4    112.0          78.0          40      0.0  39.4                    0.236  38.0        0
506           1    116.0          78.0          29    180.0  36.1                    0.496   NaN        0
507           0    141.0          84.0          26      0.0  32.4                    0.433  22.0        0
508           2    175.0          88.0           0      0.0  22.9                    0.326  22.0        0
509           2     92.0          52.0           0      0.0  30.1                    0.141  22.0        0
510           3    130.0           NaN          23     79.0  28.4                    0.323  34.0        1
511           8    120.0          86.0           0      0.0  28.4                    0.259  22.0        1
512           2    174.0          88.0          37    120.0  44.5                    0.646  24.0        1
513           2    106.0          56.0          27    165.0  29.0                    0.426  22.0        0
514           2    105.0          75.0           0      0.0  23.3                    0.560   NaN        0
515           4     95.0          60.0          32      0.0  35.4                    0.284  28.0        0
516           0    126.0          86.0          27    120.0  27.4                    0.515  21.0        0
517           8     65.0          72.0          23      0.0  32.0                    0.600  42.0        0
518           2     99.0           NaN          17    160.0  36.6                    0.453  21.0        0
519           1    102.0          74.0           0      0.0  39.5                    0.293  42.0        1
520          11    120.0          80.0          37    150.0  42.3                    0.785  48.0        1
521           3    102.0          44.0          20     94.0  30.8                    0.400  26.0        0
522           1    109.0          58.0          18    116.0  28.5                    0.219  22.0        0
523           9    140.0          94.0           0      0.0  32.7                    0.734  45.0        1
524          13    153.0          88.0          37    140.0  40.6                    1.174  39.0        0
525          12    100.0          84.0          33    105.0  30.0                    0.488   NaN        0
526           1    147.0          94.0          41      0.0  49.3                    0.358  27.0        1
527           1     81.0          74.0          41     57.0  46.3                    1.096  32.0        0
528           3    187.0           NaN          22    200.0  36.4                    0.408  36.0        1
529           6    162.0          62.0           0      0.0  24.3                    0.178  50.0        1
530           4    136.0          70.0           0      0.0  31.2                    1.182  22.0        1
531           1    121.0          78.0          39     74.0  39.0                    0.261   NaN        0
532           3    108.0          62.0          24      0.0  26.0                    0.223  25.0        0
533           0    181.0           NaN          44    510.0  43.3                    0.222  26.0        1
534           8    154.0          78.0          32      0.0  32.4                    0.443  45.0        1
535           1    128.0          88.0          39    110.0  36.5                    1.057  37.0        1
536           7    137.0          90.0          41      0.0  32.0                    0.391   NaN        0
537           0    123.0          72.0           0      0.0  36.3                    0.258  52.0        1
538           1    106.0          76.0           0      0.0  37.5                    0.197  26.0        0
539           6    190.0           NaN           0      0.0  35.5                    0.278  66.0        1
540           2     88.0          58.0          26     16.0  28.4                    0.766  22.0        0
541           9    170.0          74.0          31      0.0  44.0                    0.403  43.0        1
542           9     89.0          62.0           0      0.0  22.5                    0.142   NaN        0
543          10    101.0          76.0          48    180.0  32.9                    0.171  63.0        0
544           2    122.0           NaN          27      0.0  36.8                    0.340  27.0        0
545           5    121.0          72.0          23    112.0  26.2                    0.245  30.0        0
546           1    126.0          60.0           0      0.0   NaN                    0.349  47.0        1
547           1     93.0          70.0          31      0.0  30.4                    0.315   NaN        0
548           7    137.0          90.0          41      0.0  32.0                    0.391   NaN        0
549           1    121.0          78.0          39     74.0  39.0                    0.261   NaN        0
550           0    123.0          72.0           0      0.0  36.3                    0.258  52.0        1
551           9    170.0          74.0          31      0.0  44.0                    0.403  43.0        1
```

The dropna(inplace = True) will NOT return a new DataFrame, but it will remove all rows containing NULL values from the original DataFrame.

```
df.dropna(inplace = True)

print(df.to_string())
```

```
480   0   151.0   90.0   46    0.0  42.1   0.371  21.0  1
481   6   109.0   60.0   27    0.0  25.0   0.206  27.0  0
482  12   121.0   78.0   17    0.0  26.5   0.259  62.0  0
483   2   129.0    0.0    0    0.0  38.5   0.304  41.0  0
484   4   110.0   76.0   20  100.0  28.4   0.118  27.0  0
485   6    80.0   80.0   36    0.0  39.8   0.177  28.0  0
486  10   115.0    0.0    0    0.0   0.0   0.261  30.0  1
487   2   127.0   46.0   21  335.0  34.4   0.176  22.0  0
488   9   164.0   78.0    0    0.0  32.8   0.148  45.0  1
489   2    93.0   64.0   32  160.0  38.0   0.674  23.0  1
490   3   158.0   64.0   13  387.0  31.2   0.295  24.0  0
491   5   126.0   78.0   27   22.0  29.6   0.439  40.0  0
492  10   129.0   62.0   36    0.0  41.2   0.441  38.0  1
493   0   134.0   58.0   20  291.0  26.4   0.352  21.0  0
494   3   102.0   74.0    0    0.0  29.5   0.121  32.0  0
495   7   187.0   50.0   33  392.0  33.9   0.826  34.0  1
496   3   173.0   78.0   39  185.0  33.8   0.970  31.0  1
497  10    94.0   72.0   18    0.0  23.1   0.595  56.0  0
498   1   108.0   60.0   46  178.0  35.5   0.415  24.0  0
499   5    97.0   76.0   27    0.0  35.6   0.378  52.0  1
501   1   114.0   66.0   36  200.0  38.1   0.289  21.0  0
502   1   149.0   68.0   29  127.0  29.3   0.349  42.0  1
503   5   117.0   86.0   30  105.0  39.1   0.251  42.0  0
505   4   112.0   78.0   40    0.0  39.4   0.236  38.0  0
507   0   141.0   84.0   26    0.0  32.4   0.433  22.0  0
508   2   175.0   88.0    0    0.0  22.9   0.326  22.0  0
509   2    92.0   52.0    0    0.0  30.1   0.141  22.0  0
511   8   120.0   86.0    0    0.0  28.4   0.259  22.0  1
512   2   174.0   88.0   37  120.0  44.5   0.646  24.0  1
513   2   106.0   56.0   27  165.0  29.0   0.426  22.0  0
515   4    95.0   60.0   32    0.0  35.4   0.284  28.0  0
516   0   126.0   86.0   27  120.0  27.4   0.515  21.0  0
517   8    65.0   72.0   23    0.0  32.0   0.600  42.0  0
519   1   102.0   74.0    0    0.0  39.5   0.293  42.0  1
520  11   120.0   80.0   37  150.0  42.3   0.785  48.0  1
521   3   102.0   44.0   20   94.0  30.8   0.400  26.0  0
522   1   109.0   58.0   18  116.0  28.5   0.219  22.0  0
523   9   140.0   94.0    0    0.0  32.7   0.734  45.0  1
524  13   153.0   88.0   37  140.0  40.6   1.174  39.0  0
526   1   147.0   94.0   41    0.0  49.3   0.358  27.0  1
527   1    81.0   74.0   41   57.0  46.3   1.096  32.0  0
529   6   162.0   62.0    0    0.0  24.3   0.178  50.0  1
530   4   136.0   70.0    0    0.0  31.2   1.182  22.0  1
532   3   108.0   62.0   24    0.0  26.0   0.223  25.0  0
534   8   154.0   78.0   32    0.0  32.4   0.443  45.0  1
535   1   128.0   88.0   39  110.0  36.5   1.057  37.0  1
537   0   123.0   72.0    0    0.0  36.3   0.258  52.0  1
538   1   106.0   76.0    0    0.0  37.5   0.197  26.0  0
540   2    88.0   58.0   26   16.0  28.4   0.766  22.0  0
541   9   170.0   74.0   31    0.0  44.0   0.403  43.0  1
543  10   101.0   76.0   48  180.0  32.9   0.171  63.0  0
545   5   121.0   72.0   23  112.0  26.2   0.245  30.0  0
550   0   123.0   72.0    0    0.0  36.3   0.258  52.0  1
551   9   170.0   74.0   31    0.0  44.0   0.403  43.0  1
```

Replace Empty Values Another way of dealing with empty cells is to insert a new value instead.

This way you do not have to delete entire rows just because of some empty cells.

The fillna() method allows us to replace empty cells with a value:

```
import pandas as pd

df = pd.read_csv('diabetes.csv')

df.fillna(130, inplace = True)

print(df.to_string())
```

512	2	174.0	88.0	37	120.0	44.5	0.646	24.0	1
513	2	106.0	56.0	27	165.0	29.0	0.426	22.0	0
514	2	105.0	75.0	0	0.0	23.3	0.560	130.0	0
515	4	95.0	60.0	32	0.0	35.4	0.284	28.0	0
516	0	126.0	86.0	27	120.0	27.4	0.515	21.0	0
517	8	65.0	72.0	23	0.0	32.0	0.600	42.0	0
518	2	99.0	130.0	17	160.0	36.6	0.453	21.0	0
519	1	102.0	74.0	0	0.0	39.5	0.293	42.0	1
520	11	120.0	80.0	37	150.0	42.3	0.785	48.0	1
521	3	102.0	44.0	20	94.0	30.8	0.400	26.0	0
522	1	109.0	58.0	18	116.0	28.5	0.219	22.0	0
523	9	140.0	94.0	0	0.0	32.7	0.734	45.0	1
524	13	153.0	88.0	37	140.0	40.6	1.174	39.0	0
525	12	100.0	84.0	33	105.0	30.0	0.488	130.0	0
526	1	147.0	94.0	41	0.0	49.3	0.358	27.0	1
527	1	81.0	74.0	41	57.0	46.3	1.096	32.0	0
528	3	187.0	130.0	22	200.0	36.4	0.408	36.0	1
529	6	162.0	62.0	0	0.0	24.3	0.178	50.0	1
530	4	136.0	70.0	0	0.0	31.2	1.182	22.0	1
531	1	121.0	78.0	39	74.0	39.0	0.261	130.0	0
532	3	108.0	62.0	24	0.0	26.0	0.223	25.0	0
533	0	181.0	130.0	44	510.0	43.3	0.222	26.0	1
534	8	154.0	78.0	32	0.0	32.4	0.443	45.0	1
535	1	128.0	88.0	39	110.0	36.5	1.057	37.0	1
536	7	137.0	90.0	41	0.0	32.0	0.391	130.0	0
537	0	123.0	72.0	0	0.0	36.3	0.258	52.0	1
538	1	106.0	76.0	0	0.0	37.5	0.197	26.0	0
539	6	190.0	130.0	0	0.0	35.5	0.278	66.0	1
540	2	88.0	58.0	26	16.0	28.4	0.766	22.0	0
541	9	170.0	74.0	31	0.0	44.0	0.403	43.0	1
542	9	89.0	62.0	0	0.0	22.5	0.142	130.0	0
543	10	101.0	76.0	48	180.0	32.9	0.171	63.0	0
544	2	122.0	130.0	27	0.0	36.8	0.340	27.0	0
545	5	121.0	72.0	23	112.0	26.2	0.245	30.0	0
546	1	126.0	60.0	0	0.0	130.0	0.349	47.0	1
547	1	93.0	70.0	31	0.0	30.4	0.315	130.0	0
548	7	137.0	90.0	41	0.0	32.0	0.391	130.0	0
549	1	121.0	78.0	39	74.0	39.0	0.261	130.0	0
550	0	123.0	72.0	0	0.0	36.3	0.258	52.0	1
551	9	170.0	74.0	31	0.0	44.0	0.403	43.0	1

Replace Only For Specified Columns

The example above replaces all empty cells in the whole Data Frame.

To only replace empty values for one column, specify the column name for the DataFrame:

```
import pandas as pd

df = pd.read_csv('diabetes.csv')

df["Age"].fillna(130, inplace = True)

print(df.to_string())
```

```
528    3   187.0     NaN   22  200.0  36.4  0.408  36.0    1
529    6   162.0    62.0    0    0.0  24.3  0.178  50.0    1
530    4   136.0    70.0    0    0.0  31.2  1.182  22.0    1
531    1   121.0    78.0   39   74.0  39.0  0.261 130.0    0
532    3   108.0    62.0   24    0.0  26.0  0.223  25.0    0
533    0   181.0     NaN   44  510.0  43.3  0.222  26.0    1
534    8   154.0    78.0   32    0.0  32.4  0.443  45.0    1
535    1   128.0    88.0   39  110.0  36.5  1.057  37.0    1
536    7   137.0    90.0   41    0.0  32.0  0.391 130.0    0
537    0   123.0    72.0    0    0.0  36.3  0.258  52.0    1
538    1   106.0    76.0    0    0.0  37.5  0.197  26.0    0
539    6   190.0     NaN    0    0.0  35.5  0.278  66.0    1
540    2    88.0    58.0   26   16.0  28.4  0.766  22.0    0
541    9   170.0    74.0   31    0.0  44.0  0.403  43.0    1
542    9    89.0    62.0    0    0.0  22.5  0.142 130.0    0
543   10   101.0    76.0   48  180.0  32.9  0.171  63.0    0
544    2   122.0     NaN   27    0.0  36.8  0.340  27.0    0
545    5   121.0    72.0   23  112.0  26.2  0.245  30.0    0
546    1   126.0    60.0    0    0.0   NaN  0.349  47.0    1
547    1    93.0    70.0   31    0.0  30.4  0.315 130.0    0
548    7   137.0    90.0   41    0.0  32.0  0.391 130.0    0
549    1   121.0    78.0   39   74.0  39.0  0.261 130.0    0
550    0   123.0    72.0    0    0.0  36.3  0.258  52.0    1
551    9   170.0    74.0   31    0.0  44.0  0.403  43.0    1
```

Replace Using Mean, Median, or Mode

A common way to replace empty cells, is to calculate the mean, median or mode value of the column.

Pandas uses the mean() median() and mode() methods to calculate the respective values for a specified column:

```
import pandas as pd

df = pd.read_csv('diabetes.csv')

x = df["Age"].mean()

df["Age"].fillna(x, inplace = True)

print(df.to_string())
```

```
540   2    88.0   58.0   26    16.0  28.4   0.766  22.000000  0
541   9   170.0   74.0   31     0.0  44.0   0.403  43.000000  1
542   9    89.0   62.0    0     0.0  22.5   0.142  33.253283  0
543  10   101.0   76.0   48   180.0  32.9   0.171  63.000000  0
544   2   122.0    NaN   27     0.0  36.8   0.340  27.000000  0
545   5   121.0   72.0   23   112.0  26.2   0.245  30.000000  0
546   1   126.0   60.0    0     0.0   NaN   0.349  47.000000  1
547   1    93.0   70.0   31     0.0  30.4   0.315  33.253283  0
548   7   137.0   90.0   41     0.0  32.0   0.391  33.253283  0
549   1   121.0   78.0   39    74.0  39.0   0.261  33.253283  0
550   0   123.0   72.0    0     0.0  36.3   0.258  52.000000  1
551   9   170.0   74.0   31     0.0  44.0   0.403  43.000000  1
```

```
import pandas as pd

df = pd.read_csv('diabetes.csv')

x = df["Age"].median()

df["Age"].fillna(x, inplace = True)

print(df.to_string())
```

```
494   3   102.0   74.0    0     0.0  29.5   0.121  32.0  0
495   7   187.0   50.0   33   392.0  33.9   0.826  34.0  1
496   3   173.0   78.0   39   185.0  33.8   0.970  31.0  1
497  10    94.0   72.0   18     0.0  23.1   0.595  56.0  0
498   1   108.0   60.0   46   178.0  35.5   0.415  24.0  0
499   5    97.0   76.0   27     0.0  35.6   0.378  52.0  1
500   4    83.0   86.0   19     0.0  29.3   0.317  29.0  0
501   1   114.0   66.0   36   200.0  38.1   0.289  21.0  0
502   1   149.0   68.0   29   127.0  29.3   0.349  42.0  1
503   5   117.0   86.0   30   105.0  39.1   0.251  42.0  0
504   1   111.0    NaN    0     0.0  32.8   0.265  45.0  0
505   4   112.0   78.0   40     0.0  39.4   0.236  38.0  0
506   1   116.0   78.0   29   180.0  36.1   0.496  29.0  0
507   0   141.0   84.0   26     0.0  32.4   0.433  22.0  0
508   2   175.0   88.0    0     0.0  22.9   0.326  22.0  0
509   2    92.0   52.0    0     0.0  30.1   0.141  22.0  0
510   3   130.0    NaN   23    79.0  28.4   0.323  34.0  1
511   8   120.0   86.0    0     0.0  28.4   0.259  22.0  1
512   2   174.0   88.0   37   120.0  44.5   0.646  24.0  1
513   2   106.0   56.0   27   165.0  29.0   0.426  22.0  0
514   2   105.0   75.0    0     0.0  23.3   0.560  29.0  0
515   4    95.0   60.0   32     0.0  35.4   0.284  28.0  0
516   0   126.0   86.0   27   120.0  27.4   0.515  21.0  0
517   8    65.0   72.0   23     0.0  32.0   0.600  42.0  0
518   2    99.0    NaN   17   160.0  36.6   0.453  21.0  0
519   1   102.0   74.0    0     0.0  39.5   0.293  42.0  1
520  11   120.0   80.0   37   150.0  42.3   0.785  48.0  1
521   3   102.0   44.0   20    94.0  30.8   0.400  26.0  0
522   1   109.0   58.0   18   116.0  28.5   0.219  22.0  0
523   9   140.0   94.0    0     0.0  32.7   0.734  45.0  1
524  13   153.0   88.0   37   140.0  40.6   1.174  39.0  0
525  12   100.0   84.0   33   105.0  30.0   0.488  29.0  0
526   1   147.0   94.0   41     0.0  49.3   0.358  27.0  1
527   1    81.0   74.0   41    57.0  46.3   1.096  32.0  0
528   3   187.0    NaN   22   200.0  36.4   0.408  36.0  1
529   6   162.0   62.0    0     0.0  24.3   0.178  50.0  1
530   4   136.0   70.0    0     0.0  31.2   1.182  22.0  1
531   1   121.0   78.0   39    74.0  39.0   0.261  29.0  0
532   3   108.0   62.0   24     0.0  26.0   0.223  25.0  0
533   0   181.0    NaN   44   510.0  43.3   0.222  26.0  1
534   8   154.0   78.0   32     0.0  32.4   0.443  45.0  1
535   1   128.0   88.0   39   110.0  36.5   1.057  37.0  1
536   7   137.0   90.0   41     0.0  32.0   0.391  29.0  0
537   0   123.0   72.0    0     0.0  36.3   0.258  52.0  1
538   1   106.0   76.0    0     0.0  37.5   0.197  26.0  0
539   6   190.0    NaN    0     0.0  35.5   0.278  66.0  1
540   2    88.0   58.0   26    16.0  28.4   0.766  22.0  0
541   9   170.0   74.0   31     0.0  44.0   0.403  43.0  1
542   9    89.0   62.0    0     0.0  22.5   0.142  29.0  0
543  10   101.0   76.0   48   180.0  32.9   0.171  63.0  0
544   2   122.0    NaN   27     0.0  36.8   0.340  27.0  0
545   5   121.0   72.0   23   112.0  26.2   0.245  30.0  0
546   1   126.0   60.0    0     0.0   NaN   0.349  47.0  1
547   1    93.0   70.0   31     0.0  30.4   0.315  29.0  0
548   7   137.0   90.0   41     0.0  32.0   0.391  29.0  0
549   1   121.0   78.0   39    74.0  39.0   0.261  29.0  0
550   0   123.0   72.0    0     0.0  36.3   0.258  52.0  1
551   9   170.0   74.0   31     0.0  44.0   0.403  43.0  1
```

```
import pandas as pd

df = pd.read_csv('diabetes.csv')

x = df["Age"].mode()[0]
```

```
df["Age"].fillna(x, inplace = True)

print(df.to_string())
```

```
     494            3    102.0          74.0            0      0.0  29.5                    0.121  32.0        0
     495            7    187.0          50.0           33    392.0  33.9                    0.826  34.0        1
     496            3    173.0          78.0           39    185.0  33.8                    0.970  31.0        1
     497           10     94.0          72.0           18      0.0  23.1                    0.595  56.0        0
     498            1    108.0          60.0           46    178.0  35.5                    0.415  24.0        0
     499            5     97.0          76.0           27      0.0  35.6                    0.378  52.0        1
     500            4     83.0          86.0           19      0.0  29.3                    0.317  22.0        0
     501            1    114.0          66.0           36    200.0  38.1                    0.289  21.0        0
     502            1    149.0          68.0           29    127.0  29.3                    0.349  42.0        1
     503            5    117.0          86.0           30    105.0  39.1                    0.251  42.0        0
     504            1    111.0           NaN            0      0.0  32.8                    0.265  45.0        0
     505            4    112.0          78.0           40      0.0  39.4                    0.236  38.0        0
     506            1    116.0          78.0           29    180.0  36.1                    0.496  22.0        0
     507            0    141.0          84.0           26      0.0  32.4                    0.433  22.0        0
     508            2    175.0          88.0            0      0.0  22.9                    0.326  22.0        0
     509            2     92.0          52.0            0      0.0  30.1                    0.141  22.0        0
     510            3    130.0           NaN           23     79.0  28.4                    0.323  34.0        1
     511            8    120.0          86.0            0      0.0  28.4                    0.259  22.0        1
     512            2    174.0          88.0           37    120.0  44.5                    0.646  24.0        1
     513            2    106.0          56.0           27    165.0  29.0                    0.426  22.0        0
     514            2    105.0          75.0            0      0.0  23.3                    0.560  22.0        0
     515            4     95.0          60.0           32      0.0  35.4                    0.284  28.0        0
     516            0    126.0          86.0           27    120.0  27.4                    0.515  21.0        0
     517            8     65.0          72.0           23      0.0  32.0                    0.600  42.0        0
     518            2     99.0           NaN           17    160.0  36.6                    0.453  21.0        0
     519            1    102.0          74.0            0      0.0  39.5                    0.293  42.0        1
     520           11    120.0          80.0           37    150.0  42.3                    0.785  48.0        1
     521            3    102.0          44.0           20     94.0  30.8                    0.400  26.0        0
     522            1    109.0          58.0           18    116.0  28.5                    0.219  22.0        0
     523            9    140.0          94.0            0      0.0  32.7                    0.734  45.0        1
     524           13    153.0          88.0           37    140.0  40.6                    1.174  39.0        0
     525           12    100.0          84.0           33    105.0  30.0                    0.488  22.0        0
     526            1    147.0          94.0           41      0.0  49.3                    0.358  27.0        1
     527            1     81.0          74.0           41     57.0  46.3                    1.096  32.0        0
     528            3    187.0           NaN           22    200.0  36.4                    0.408  36.0        1
     529            6    162.0          62.0            0      0.0  24.3                    0.178  50.0        1
     530            4    136.0          70.0            0      0.0  31.2                    1.182  22.0        1
     531            1    121.0          78.0           39     74.0  39.0                    0.261  22.0        0
     532            3    108.0          62.0           24      0.0  26.0                    0.223  25.0        0
     533            0    181.0           NaN           44    510.0  43.3                    0.222  26.0        1
     534            8    154.0          78.0           32      0.0  32.4                    0.443  45.0        1
     535            1    128.0          88.0           39    110.0  36.5                    1.057  37.0        1
     536            7    137.0          90.0           41      0.0  32.0                    0.391  22.0        0
     537            0    123.0          72.0            0      0.0  36.3                    0.258  52.0        1
     538            1    106.0          76.0            0      0.0  37.5                    0.197  26.0        0
     539            6    190.0           NaN            0      0.0  35.5                    0.278  66.0        1
     540            2     88.0          58.0           26     16.0  28.4                    0.766  22.0        0
     541            9    170.0          74.0           31      0.0  44.0                    0.403  43.0        1
     542            9     89.0          62.0            0      0.0  22.5                    0.142  22.0        0
     543           10    101.0          76.0           48    180.0  32.9                    0.171  63.0        0
     544            2    122.0           NaN           27      0.0  36.8                    0.340  27.0        0
     545            5    121.0          72.0           23    112.0  26.2                    0.245  30.0        0
     546            1    126.0          60.0            0      0.0   NaN                    0.349  47.0        1
     547            1     93.0          70.0           31      0.0  30.4                    0.315  22.0        0
     548            7    137.0          90.0           41      0.0  32.0                    0.391  22.0        0
     549            1    121.0          78.0           39     74.0  39.0                    0.261  22.0        0
     550            0    123.0          72.0            0      0.0  36.3                    0.258  52.0        1
     551            9    170.0          74.0           31      0.0  44.0                    0.403  43.0        1
```

Remove rows with a NULL value in the "Age" column:

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

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

print(df.to_string())
```

```
561   1  114.0  66.0  36  200.0  38.1  0.289  21.0  0
502   1  149.0  68.0  29  127.0  29.3  0.349  42.0  1
503   5  117.0  86.0  30  105.0  39.1  0.251  42.0  0
504   1  111.0   NaN   0    0.0  32.8  0.265  45.0  0
505   4  112.0  78.0  40    0.0  39.4  0.236  38.0  0
507   0  141.0  84.0  26    0.0  32.4  0.433  22.0  0
508   2  175.0  88.0   0    0.0  22.9  0.326  22.0  0
509   2   92.0  52.0   0    0.0  30.1  0.141  22.0  0
510   3  130.0   NaN  23   79.0  28.4  0.323  34.0  1
511   8  120.0  86.0   0    0.0  28.4  0.259  22.0  1
512   2  174.0  88.0  37  120.0  44.5  0.646  24.0  1
513   2  106.0  56.0  27  165.0  29.0  0.426  22.0  0
515   4   95.0  60.0  32    0.0  35.4  0.284  28.0  0
516   0  126.0  86.0  27  120.0  27.4  0.515  21.0  0
517   8   65.0  72.0  23    0.0  32.0  0.600  42.0  0
518   2   99.0   NaN  17  160.0  36.6  0.453  21.0  0
519   1  102.0  74.0   0    0.0  39.5  0.293  42.0  1
520  11  120.0  80.0  37  150.0  42.3  0.785  48.0  1
521   3  102.0  44.0  20   94.0  30.8  0.400  26.0  0
522   1  109.0  58.0  18  116.0  28.5  0.219  22.0  0
523   9  140.0  94.0   0    0.0  32.7  0.734  45.0  1
524  13  153.0  88.0  37  140.0  40.6  1.174  39.0  0
526   1  147.0  94.0  41    0.0  49.3  0.358  27.0  1
527   1   81.0  74.0  41   57.0  46.3  1.096  32.0  0
528   3  187.0   NaN  22  200.0  36.4  0.408  36.0  1
529   6  162.0  62.0   0    0.0  24.3  0.178  50.0  1
530   4  136.0  70.0   0    0.0  31.2  1.182  22.0  1
532   3  108.0  62.0  24    0.0  26.0  0.223  25.0  0
533   0  181.0   NaN  44  510.0  43.3  0.222  26.0  1
534   8  154.0  78.0  32    0.0  32.4  0.443  45.0  1
535   1  128.0  88.0  39  110.0  36.5  1.057  37.0  1
537   0  123.0  72.0   0    0.0  36.3  0.258  52.0  1
538   1  106.0  76.0   0    0.0  37.5  0.197  26.0  0
539   6  190.0   NaN   0    0.0  35.5  0.278  66.0  1
540   2   88.0  58.0  26   16.0  28.4  0.766  22.0  0
541   9  170.0  74.0  31    0.0  44.0  0.403  43.0  1
543  10  101.0  76.0  48  180.0  32.9  0.171  63.0  0
544   2  122.0   NaN  27    0.0  36.8  0.340  27.0  0
545   5  121.0  72.0  23  112.0  26.2  0.245  30.0  0
546   1  126.0  60.0   0    0.0   NaN  0.349  47.0  1
550   0  123.0  72.0   0    0.0  36.3  0.258  52.0  1
551   9  170.0  74.0  31    0.0  44.0  0.403  43.0  1
```

Replacing Values

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

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

df.loc[545, 'Age'] = 45

print(df.to_string())
```

```
530   4   136.0   70.0   0   0.0  31.2  1.182  22.0  1
531   1   121.0   78.0  39  74.0  39.0  0.261   NaN  0
532   3   108.0   62.0  24   0.0  26.0  0.223  25.0  0
533   0   181.0    NaN  44 510.0  43.3  0.222  26.0  1
534   8   154.0   78.0  32   0.0  32.4  0.443  45.0  1
535   1   128.0   88.0  39 110.0  36.5  1.057  37.0  1
536   7   137.0   90.0  41   0.0  32.0  0.391   NaN  0
537   0   123.0   72.0   0   0.0  36.3  0.258  52.0  1
538   1   106.0   76.0   0   0.0  37.5  0.197  26.0  0
539   6   190.0    NaN   0   0.0  35.5  0.278  66.0  1
540   2    88.0   58.0  26  16.0  28.4  0.766  22.0  0
541   9   170.0   74.0  31   0.0  44.0  0.403  43.0  1
542   9    89.0   62.0   0   0.0  22.5  0.142   NaN  0
543  10   101.0   76.0  48 180.0  32.9  0.171  63.0  0
544   2   122.0    NaN  27   0.0  36.8  0.340  27.0  0
545   5   121.0   72.0  23 112.0  26.2  0.245  45.0  0
546   1   126.0   60.0   0   0.0   NaN  0.349  47.0  1
547   1    93.0   70.0  31   0.0  30.4  0.315   NaN  0
548   7   137.0   90.0  41   0.0  32.0  0.391   NaN  0
549   1   121.0   78.0  39  74.0  39.0  0.261   NaN  0
550   0   123.0   72.0   0   0.0  36.3  0.258  52.0  1
551   9   170.0   74.0  31   0.0  44.0  0.403  43.0  1
```

For small data sets you might be able to replace the wrong data one by one, but not for big data sets.

To replace wrong data for larger data sets you can create some rules

Loop through all values in the "Glucose" column.

If the value is higher than 120, set it to 120:

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

for x in df.index:
  if df.loc[x, "Glucose"] > 120:
    df.loc[x, "Glucose"] = 120

print(df.to_string())
```

```
543  10  101.0  76.0  48  180.0  32.9  0.171  63.0  0
544   2  120.0   NaN  27    0.0  36.8  0.340  27.0  0
545   5  120.0  72.0  23  112.0  26.2  0.245  30.0  0
546   1  120.0  60.0   0    0.0   NaN  0.349  47.0  1
547   1   93.0  70.0  31    0.0  30.4  0.315   NaN  0
548   7  120.0  90.0  41    0.0  32.0  0.391   NaN  0
549   1  120.0  78.0  39   74.0  39.0  0.261   NaN  0
550   0  120.0  72.0   0    0.0  36.3  0.258  52.0  1
551   9  120.0  74.0  31    0.0  44.0  0.403  43.0  1
```

Removing Rows

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

This way you do not have to find out what to replace them with, and there is a good chance you do not need them to do your analyses.

Delete rows where "Glucose" is higher than 120:

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

for x in df.index:
  if df.loc[x, "Glucose"] > 120:
    df.drop(x, inplace = True)

print(df.to_string())
```

```
439   2   89.0  90.0  30    0.0  33.5  0.292  42.0  0
440   4   99.0  68.0  38    0.0  32.8  0.145  33.0  0
441  13   76.0  60.0   0    0.0  32.8  0.180  41.0  0
443   2   68.0  70.0  32   66.0  25.0  0.187  25.0  0
445   6  114.0   0.0   0    0.0   0.0  0.189  26.0  0
448   3   87.0  60.0  18    0.0  21.8  0.444  21.0  0
449   1   97.0  64.0  19   82.0  18.2  0.299  21.0  0
450   3  116.0  74.0  15  105.0  26.3  0.107  24.0  0
451   0  117.0  66.0  31  188.0  30.8  0.493  22.0  0
452   0  111.0  65.0   0    0.0  24.6  0.660  31.0  0
454   0  107.0  76.0   0    0.0  45.3  0.686  24.0  0
455   1   86.0  66.0  52   65.0  41.3  0.917  29.0  0
456   6   91.0   0.0   0    0.0  29.8  0.501  31.0  0
457   1   77.0  56.0  30   56.0  33.3  1.251  24.0  0
459   0  105.0  90.0   0    0.0  29.6  0.197  46.0  0
460   0   57.0  60.0   0    0.0  21.7  0.735  67.0  0
463   8  100.0  74.0  40  215.0  39.4  0.661  43.0  1
465  10   90.0  85.0  32    0.0  34.9  0.825  56.0  1
466   4   84.0  90.0  23   56.0  39.5  0.159  25.0  0
467   1   88.0  78.0  29   76.0  32.0  0.365  29.0  0
473   1  116.0  70.0  28    0.0  27.4  0.204  21.0  0
475   1  119.0  44.0  47   63.0  35.5  0.280  25.0  0
476   6  108.0  44.0  20  130.0  24.0  0.813  35.0  0
477   2  118.0  80.0   0    0.0  42.9  0.693  21.0  1
481   6  109.0  60.0  27    0.0  25.0  0.206  27.0  0
484   4  110.0  76.0  20  100.0  28.4  0.118  27.0  0
485   6   80.0  80.0  36    0.0  39.8  0.177  28.0  0
486  10  115.0   0.0   0    0.0   0.0  0.261  30.0  1
489   2   93.0  64.0  32  160.0  38.0  0.674  23.0  1
494   3  102.0  74.0   0    0.0  29.5  0.121  32.0  0
497  10   94.0  72.0  18    0.0  23.1  0.595  56.0  0
498   1  108.0  60.0  46  178.0  35.5  0.415  24.0  0
499   5   97.0  76.0  27    0.0  35.6  0.378  52.0  1
500   4   83.0  86.0  19    0.0  29.3  0.317   NaN  0
501   1  114.0  66.0  36  200.0  38.1  0.289  21.0  0
503   5  117.0  86.0  30  105.0  39.1  0.251  42.0  0
504   1  111.0   NaN   0    0.0  32.8  0.265  45.0  0
505   4  112.0  78.0  40    0.0  39.4  0.236  38.0  0
506   1  116.0  78.0  29  180.0  36.1  0.496   NaN  0
509   2   92.0  52.0   0    0.0  30.1  0.141  22.0  0
511   8  120.0  86.0   0    0.0  28.4  0.259  22.0  1
513   2  106.0  56.0  27  165.0  29.0  0.426  22.0  0
514   2  105.0  75.0   0    0.0  23.3  0.560   NaN  0
515   4   95.0  60.0  32    0.0  35.4  0.284  28.0  0
517   8   65.0  72.0  23    0.0  32.0  0.600  42.0  0
518   2   99.0   NaN  17  160.0  36.6  0.453  21.0  0
519   1  102.0  74.0   0    0.0  39.5  0.293  42.0  1
520  11  120.0  80.0  37  150.0  42.3  0.785  48.0  1
521   3  102.0  44.0  20   94.0  30.8  0.400  26.0  0
522   1  109.0  58.0  18  116.0  28.5  0.219  22.0  0
525  12  100.0  84.0  33  105.0  30.0  0.488   NaN  0
527   1   81.0  74.0  41   57.0  46.3  1.096  32.0  0
532   3  108.0  62.0  24    0.0  26.0  0.223  25.0  0
538   1  106.0  76.0   0    0.0  37.5  0.197  26.0  0
540   2   88.0  58.0  26   16.0  28.4  0.766  22.0  0
542   9   89.0  62.0   0    0.0  22.5  0.142   NaN  0
543  10  101.0  76.0  48  180.0  32.9  0.171  63.0  0
547   1   93.0  70.0  31    0.0  30.4  0.315   NaN  0
```

Discovering Duplicates

Duplicate rows are rows that have been registered more than one time.

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

```
df=pd.read_csv("diabetes.csv")

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

```
494    False
495    False
496    False
497    False
498    False
499    False
500    False
501    False
502    False
503    False
504    False
505    False
506    False
507    False
508    False
509    False
510    False
511    False
512    False
513    False
514    False
515    False
516    False
517    False
518    False
519    False
520    False
521    False
522    False
523    False
524    False
525    False
526    False
527    False
528    False
529    False
530    False
531    False
532    False
533    False
534    False
535    False
536    False
537    False
538    False
539    False
540    False
541    False
542    False
543    False
544    False
545    False
546    False
547    False
548     True
549     True
550     True
551     True
```

Removing Duplicates

To remove duplicates, use the drop_duplicates() method.

```
df.drop_duplicates(inplace = True)

print(df.to_string())
```

502	1	149.0	68.0	29	127.0	29.3	0.349	42.0	1
503	5	117.0	86.0	30	105.0	39.1	0.251	42.0	0
504	1	111.0	NaN	0	0.0	32.8	0.265	45.0	0
505	4	112.0	78.0	40	0.0	39.4	0.236	38.0	0
506	1	116.0	78.0	29	180.0	36.1	0.496	NaN	0
507	0	141.0	84.0	26	0.0	32.4	0.433	22.0	0
508	2	175.0	88.0	0	0.0	22.9	0.326	22.0	0
509	2	92.0	52.0	0	0.0	30.1	0.141	22.0	0
510	3	130.0	NaN	23	79.0	28.4	0.323	34.0	1
511	8	120.0	86.0	0	0.0	28.4	0.259	22.0	1
512	2	174.0	88.0	37	120.0	44.5	0.646	24.0	1
513	2	106.0	56.0	27	165.0	29.0	0.426	22.0	0
514	2	105.0	75.0	0	0.0	23.3	0.560	NaN	0
515	4	95.0	60.0	32	0.0	35.4	0.284	28.0	0
516	0	126.0	86.0	27	120.0	27.4	0.515	21.0	0
517	8	65.0	72.0	23	0.0	32.0	0.600	42.0	0
518	2	99.0	NaN	17	160.0	36.6	0.453	21.0	0
519	1	102.0	74.0	0	0.0	39.5	0.293	42.0	1
520	11	120.0	80.0	37	150.0	42.3	0.785	48.0	1
521	3	102.0	44.0	20	94.0	30.8	0.400	26.0	0
522	1	109.0	58.0	18	116.0	28.5	0.219	22.0	0
523	9	140.0	94.0	0	0.0	32.7	0.734	45.0	1
524	13	153.0	88.0	37	140.0	40.6	1.174	39.0	0
525	12	100.0	84.0	33	105.0	30.0	0.488	NaN	0
526	1	147.0	94.0	41	0.0	49.3	0.358	27.0	1
527	1	81.0	74.0	41	57.0	46.3	1.096	32.0	0
528	3	187.0	NaN	22	200.0	36.4	0.408	36.0	1
529	6	162.0	62.0	0	0.0	24.3	0.178	50.0	1
530	4	136.0	70.0	0	0.0	31.2	1.182	22.0	1
531	1	121.0	78.0	39	74.0	39.0	0.261	NaN	0
532	3	108.0	62.0	24	0.0	26.0	0.223	25.0	0
533	0	181.0	NaN	44	510.0	43.3	0.222	26.0	1
534	8	154.0	78.0	32	0.0	32.4	0.443	45.0	1
535	1	128.0	88.0	39	110.0	36.5	1.057	37.0	1
536	7	137.0	90.0	41	0.0	32.0	0.391	NaN	0
537	0	123.0	72.0	0	0.0	36.3	0.258	52.0	1
538	1	106.0	76.0	0	0.0	37.5	0.197	26.0	0
539	6	190.0	NaN	0	0.0	35.5	0.278	66.0	1
540	2	88.0	58.0	26	16.0	28.4	0.766	22.0	0
541	9	170.0	74.0	31	0.0	44.0	0.403	43.0	1
542	9	89.0	62.0	0	0.0	22.5	0.142	NaN	0
543	10	101.0	76.0	48	180.0	32.9	0.171	63.0	0
544	2	122.0	NaN	27	0.0	36.8	0.340	27.0	0
545	5	121.0	72.0	23	112.0	26.2	0.245	30.0	0
546	1	126.0	60.0	0	0.0	NaN	0.349	47.0	1
547	1	93.0	70.0	31	0.0	30.4	0.315	NaN	0

✓ 0s completed at 10:11 AM