Collect Data from Files

In this case study, we are going to collect weather data from all kinds of sources. At the end, we will have an integrated csv file for further analysis in next courses.

```
import pandas as pd
```

Colunms

This is a simplified scenario, is that we know the data are following same structure. We don't have to worry about the order of columns.

In real life scenario, we may have to investigate and find out how columns across different resources.

1. Are they the same or different?
2. If they share the same name, do they represent the same attribute? For example, `price` in source 1, and `price` in source 2. How about `price` in source 1 is in US Dollars, and `price` in source 2 is in EURO?
3. If they have different names, do they really represent different attributes? For example, `DoB` in source 1 and `Birthday` in source 2. `STUID` in source 1 and `ID` in source 2.

TXT

```
df1 = pd.read_csv('/content/boulderdaily_Part1.txt')
df1
```

	year	month	day	tmax	tmin	precip	snow	snowcover
0	1897	1	1	-998	-998	-998.0	-998.0	-998
1	1897	1	2	-998	-998	-998.0	-998.0	-998
2	1897	1	3	-998	-998	-998.0	-998.0	-998
3	1897	1	4	-998	-998	-998.0	-998.0	-998
4	1897	1	5	-998	-998	-998.0	-998.0	-998
...	...	...	...	...	...	...	...	...
1093	1899	12	27	50	22	0.0	-998.0	-998
1094	1899	12	28	43	23	0.0	-998.0	-998
1095	1899	12	29	56	23	0.0	-998.0	-998
1096	1899	12	30	48	24	0.0	-998.0	-998
1097	1899	12	31	50	25	0.0	-998.0	-998

1098 rows × 8 columns

Excel

```
df2 = pd.read_excel('/content/boulderdaily_Part2.xlsx')
df2
```

	year	month	day	tmax	tmin	precip	snow	snowcover
0	1900	1	1	40	18	0.0	-998	-998
1	1900	1	2	47	20	0.0	-998	-998
2	1900	1	3	57	31	0.0	-998	-998
3	1900	1	4	47	33	0.0	-998	-998
4	1900	1	5	53	29	0.0	-998	-998
...	...	...	...	...	...	...	...	...
10975	1929	12	27	49	24	0.0	-998	-998
10976	1929	12	28	59	19	0.0	-998	-998
10977	1929	12	29	61	31	0.0	-998	-998
10978	1929	12	30	62	36	0.0	-998	-998
10979	1929	12	31	55	32	0.0	-998	-998

10980 rows × 8 columns

HTML

```
df3 = pd.read_html('/content/boulderdaily_Part3.html', header = 0)[0]
df3
```

	year	month	day	tmax	tmin	precip	snow	snowcover
0	1930	1	1	45	20	-999.00	-998	-998
1	1930	1	2	36	17	0.01	-998	-998
2	1930	1	3	54	13	0.00	-998	-998
3	1930	1	4	57	26	0.00	-998	-998
4	1930	1	5	58	28	0.00	-998	-998
...	...	...	...	...	...	...	...	...
3655	1939	12	27	21	2	0.12	-998	-998
3656	1939	12	28	45	14	0.00	-998	-998
3657	1939	12	29	49	26	0.00	-998	-998
3658	1939	12	30	57	36	0.00	-998	-998
3659	1939	12	31	47	24	-999.00	-998	-998

3660 rows × 8 columns

▾ json

```
df4 = pd.read_json('/content/boulderdaily_Part4.json')
df4
```

	year	month	day	tmax	tmin	precip	snow	snowcover
0	1940	1	1	28	20	-999.00	-998.0	-998
1	1940	1	2	44	20	0.00	-998.0	-998
2	1940	1	3	34	27	-999.00	-998.0	-998
3	1940	1	4	28	18	0.10	-998.0	-998
4	1940	1	5	22	5	0.31	-998.0	-998
...	...	...	...	...	...	...	...	...
3655	1949	12	27	57	27	0.00	0.0	0
3656	1949	12	28	65	38	0.00	0.0	0
3657	1949	12	29	60	34	0.00	0.0	0
3658	1949	12	30	62	34	0.00	0.0	0
3659	1949	12	31	54	25	0.00	0.0	0

3660 rows × 8 columns

HTML Table

```
df5 = pd.read_html('/content/boulderdaily_Part5.htm')[0]
df5
```

	year	month	day	tmax	tmin	precip	snow	snowcover
0	1950	1	1	59	35	0.00	0.0	0
1	1950	1	2	55	29	0.00	0.0	0
2	1950	1	3	32	-9	0.43	5.0	4
3	1950	1	4	16	-12	0.00	0.0	4
4	1950	1	5	30	-1	0.00	0.0	3
...	...	...	...	...	...	...	...	...
3655	1959	12	27	45	23	0.00	0.0	0
3656	1959	12	28	48	28	0.00	0.0	0
3657	1959	12	29	38	19	0.00	0.0	0
3658	1959	12	30	39	15	0.00	0.0	0
3659	1959	12	31	37	17	-999.00	-999.0	0

3660 rows × 8 columns

Database

```
import sqlite3

connection = sqlite3.connect('/content/boulderdaily_Part6.sqlite')
cursor = connection.cursor()

query = '''SELECT name FROM sqlite_master
WHERE type='table';'''

cursor.execute(query)
results = cursor.fetchall()
results
```

```
[('boulderdaily_Part6',)]
```

```
query = '''SELECT *
FROM boulderdaily_Part6'''
```

```
cursor.execute(query)
results = cursor.fetchall()
results
```

```
[(1960, 1, 1, 31, 15, 0.07, 3.0, 2),
 (1960, 1, 2, 32, 4, 0.0, 0.0, 1),
 (1960, 1, 3, 24, 5, 0.0, 0.0, 1),
 (1960, 1, 4, 30, 3, 0.0, 0.0, 1),
 (1960, 1, 5, 36, 10, 0.0, 0.0, -999),
 (1960, 1, 6, 51, 18, 0.0, 0.0, -999),
 (1960, 1, 7, 54, 37, 0.0, 0.0, -999),
 (1960, 1, 8, 63, 25, 0.0, 0.0, 0),
 (1960, 1, 9, 65, 35, 0.0, 0.0, 0),
 (1960, 1, 10, 55, 32, 0.0, 0.0, 0),
 (1960, 1, 11, 53, 27, 0.0, 0.0, 0),
 (1960, 1, 12, 54, 31, 0.0, 0.0, 0),
 (1960, 1, 13, 50, 27, 0.0, 0.0, 0),
 (1960, 1, 14, 39, 21, 0.25, 3.5, 3),
 (1960, 1, 15, 34, 13, 0.0, 0.0, 3),
 (1960, 1, 16, 25, 14, 0.05, 2.0, 5),
 (1960, 1, 17, 23, 16, 0.16, 1.8, 7),
 (1960, 1, 18, 21, 4, 0.01, 0.1, 5),
 (1960, 1, 19, 30, -2, 0.0, 0.0, 4),
 (1960, 1, 20, 41, 4, 0.0, 0.0, 4),
 (1960, 1, 21, 32, 11, 0.0, 0.0, 4),
 (1960, 1, 22, 23, 4, 0.0, 0.0, 4),
 (1960, 1, 23, 47, 15, 0.0, 0.0, 4),
 (1960, 1, 24, 48, 19, 0.0, 0.0, 3),
 (1960, 1, 25, 48, 32, 0.0, 0.0, 3),
 (1960, 1, 26, 55, 29, 0.0, 0.0, 2),
 (1960, 1, 27, 54, 33, -999.0, -999.0, 1),
 (1960, 1, 28, 47, 32, 0.0, 0.0, 0),
 (1960, 1, 29, 61, 23, 0.0, 0.0, 0),
 (1960, 1, 30, 60, 33, 0.0, 0.0, 0),
 (1960, 1, 31, 48, 36, 0.0, 0.0, 0),
 (1960, 2, 1, 48, 30, 0.01, 0.1, -999),
 (1960, 2, 2, 40, 23, -998.0, -998.0, -999),
 (1960, 2, 3, 43, 31, 0.19, 1.5, -999),
 (1960, 2, 4, 46, 26, 0.0, 0.0, 0),
 (1960, 2, 5, 46, 26, 0.0, 0.0, 0),
 (1960, 2, 6, 52, 32, 0.0, 0.0, 0),
 (1960, 2, 7, 63, 26, 0.0, 0.0, 0),
 (1960, 2, 8, 59, 37, 0.0, 0.0, 0),
 (1960, 2, 9, 55, 37, 0.0, 0.0, 0),
 (1960, 2, 10, 42, 22, 0.0, 0.0, 0),
 (1960, 2, 11, 41, 26, 0.01, -999.0, -999),
 (1960, 2, 12, 49, 18, 0.02, 0.5, -999),
 (1960, 2, 13, 53, 30, 0.0, 0.0, -999),
 (1960, 2, 14, 45, 25, 0.25, 4.2, 4),
 (1960, 2, 15, 48, 22, 0.0, 0.0, 2),
 (1960, 2, 16, 41, 29, 0.0, 0.0, 1),
 (1960, 2, 17, 35, 21, 0.0, 0.0, -999),
 (1960, 2, 18, 35, 12, 0.0, 0.0, 0),
 (1960, 2, 19, 34, 17, 0.0, 0.0, 0)
```

```
(1960, 2, 19, 34, 17, 0.0, 0.0, 6),
(1960, 2, 20, 33, 18, 0.06, 3.6, 2),
(1960, 2, 21, 46, 17, 0.0, 0.0, 1),
(1960, 2, 22, 39, 12, 0.3, 8.5, 9),
(1960, 2, 23, 21, 1, 0.07, 1.0, 7),
(1960, 2, 24, 21, 1, 0.0, 0.0, 6),
(1960, 2, 25, 18, 1, -999.0, -999.0, 6),
(1960, 2, 26, 14, 0, 0.03, 0.5, 6),
(1960, 2, 27, 12, 1, 0.05, 0.6, 7),
(1960, 2, 28, 12, -5, 0.01, 0.4, 7),
(1960, 2, 29, 24, -5, 0.04, 0.5, 7)
```

```
df6 = pd.DataFrame(results)
df6
```

	0	1	2	3	4	5	6	7
0	1960	1	1	31	15	0.07	3.0	2
1	1960	1	2	32	4	0.00	0.0	1
2	1960	1	3	24	5	0.00	0.0	1
3	1960	1	4	30	3	0.00	0.0	1
4	1960	1	5	36	10	0.00	0.0	-999
...	...	...	...	...	...	...	...	...
7315	1979	12	27	35	29	0.96	9.5	-998
7316	1979	12	28	30	25	0.44	9.0	-998
7317	1979	12	29	32	15	0.00	0.0	-998
7318	1979	12	30	44	13	0.00	0.0	-998
7319	1979	12	31	52	16	0.00	0.0	-998

7320 rows × 8 columns

```
df6.columns = ['year',  'month',    'day',  'tmax', 'tmin', 'precip',   'snow', 's
df6
```

	year	month	day	tmax	tmin	precip	snow	snowcover
0	1960	1	1	31	15	0.07	3.0	2
1	1960	1	2	32	4	0.00	0.0	1
2	1960	1	3	24	5	0.00	0.0	1
3	1960	1	4	30	3	0.00	0.0	1
4	1960	1	5	36	10	0.00	0.0	-999
...	...	...	...	...	...	...	...	...
7315	1979	12	27	35	29	0.96	9.5	-998
7316	1979	12	28	30	25	0.44	9.0	-998
7317	1979	12	29	32	15	0.00	0.0	-998
7318	1979	12	30	44	13	0.00	0.0	-998
7319	1979	12	31	52	16	0.00	0.0	-998

7320 rows × 8 columns

XML

```
df7 = pd.read_xml('/content/boulderdaily_Part7.xml')
df7
```

	year	month	day	tmax	tmin	precip	snow	snowcover
0	1980	1	1	51	29	0.0	0.0	-998
1	1980	1	2	42	29	-999.0	0.0	-998
2	1980	1	3	43	14	-999.0	0.0	-998
3	1980	1	4	43	31	0.0	0.0	-998
4	1980	1	5	57	25	0.0	0.0	-998
...	...	...	...	...	...	...	...	...
7315	1999	12	27	54	29	0.0	-998.0	-998
7316	1999	12	28	63	32	0.0	-998.0	-998
7317	1999	12	29	63	30	0.0	-998.0	-998
7318	1999	12	30	58	27	0.0	-998.0	-998
7319	1999	12	31	60	26	0.0	-998.0	-998

7320 rows × 8 columns

CSV

We have done this when we learned pandas. You can get the path of your csv file, and feed the path to the function `read_csv`.

```
df8 = pd.read_csv('/content/boulderdaily_Part8.csv')
df8
```

	year	month	day	tmax	tmin	precip	snow	snowcover
0	2000	1	1	54	29	0.00	-998.0	-998
1	2000	1	2	43	22	0.00	-998.0	-998
2	2000	1	3	36	19	0.08	-998.0	-998
3	2000	1	4	49	13	0.00	-998.0	-998
4	2000	1	5	47	26	0.00	-998.0	-998
...	...	...	...	...	...	...	...	...
8413	2022	12	27	62	32	0.00	0.0	-999
8414	2022	12	28	57	41	-999.00	0.0	0
8415	2022	12	29	41	21	1.26	9.3	9
8416	2022	12	30	37	19	0.00	0.0	8
8417	2022	12	31	53	23	0.00	0.0	6

8418 rows × 8 columns

Integration!

Now we need to concatenate all dataframes together.

```
df = pd.concat([df1, df2, df3, df4, df5, df6, df7, df8], ignore_index = True)
df
```

	year	month	day	tmax	tmin	precip	snow	snowcover
0	1897	1	1	-998	-998	-998.00	-998.0	-998
1	1897	1	2	-998	-998	-998.00	-998.0	-998
2	1897	1	3	-998	-998	-998.00	-998.0	-998
3	1897	1	4	-998	-998	-998.00	-998.0	-998
4	1897	1	5	-998	-998	-998.00	-998.0	-998
...	...	...	...	...	...	...	...	...
46111	2022	12	27	62	32	0.00	0.0	-999
46112	2022	12	28	57	41	-999.00	0.0	0
46113	2022	12	29	41	21	1.26	9.3	9
46114	2022	12	30	37	19	0.00	0.0	8
46115	2022	12	31	53	23	0.00	0.0	6

46116 rows × 8 columns

Now we save the integrated data

```
df.to_csv('/content/boulderdaily.csv', index = False)
```

test the data is successfully saved

```
test = pd.read_csv('/content/boulderdaily.csv')
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 46116 entries, 0 to 46115
Data columns (total 8 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   year       46116 non-null  int64  
 1   month      46116 non-null  int64  
 2   day        46116 non-null  int64  
 3   tmax       46116 non-null  int64  
 4   tmin       46116 non-null  int64  
 5   precip     46116 non-null  float64
 6   snow       46116 non-null  float64
 7   snowcover  46116 non-null  int64  
dtypes: float64(2), int64(6)
memory usage: 2.8 MB
```