

This notebook is an exercise in the [Data Cleaning \(https://www.kaggle.com/learn/data-cleaning\)](https://www.kaggle.com/learn/data-cleaning) course. You can reference the tutorial at [this link \(https://www.kaggle.com/alexisbcook/parsing-dates\)](https://www.kaggle.com/alexisbcook/parsing-dates).

In this exercise, you'll apply what you learned in the **Parsing dates** tutorial.

Setup

The questions below will give you feedback on your work. Run the following cell to set up the feedback system.

```
In [1]: from learntools.core import binder
binder.bind(globals())
from learntools.data_cleaning.ex3 import *
print("Setup Complete")
```

Setup Complete

Get our environment set up

The first thing we'll need to do is load in the libraries and dataset we'll be using. We'll be working with a dataset containing information on earthquakes that occurred between 1965 and 2016.

```
In [2]: # modules we'll use
import pandas as pd
import numpy as np
import seaborn as sns
import datetime

# read in our data
earthquakes = pd.read_csv("../input/earthquake-database/database.csv")

# set seed for reproducibility
np.random.seed(0)
```

1) Check the data type of our date column

You'll be working with the "Date" column from the `earthquakes` dataframe. Investigate this column now: does it look like it contains dates? What is the dtype of the column?

```
In [18]: # TODO: Your code here!
earthquakes['Date'].head()
```

```
Out[18]: 0    01/02/1965
1    01/04/1965
2    01/05/1965
3    01/08/1965
4    01/09/1965
Name: Date, dtype: object
```

Once you have answered the question above, run the code cell below to get credit for your work.

```
In [4]: # Check your answer (Run this code cell to receive credit!)
q1.check()
```

Correct:

The "Date" column in the `earthquakes` DataFrame does have dates. The dtype is "object".

```
In [5]: # Line below will give you a hint
#q1.hint()
```

2) Convert our date columns to datetime

Most of the entries in the "Date" column follow the same format: "month/day/four-digit year". However, the entry at index 3378 follows a completely different pattern. Run the code cell below to see this.

```
In [6]: earthquakes[3378:3383]
```

Out[6]:

	Date	Time	Latitude	Longitude	Type	Depth	Depth Error	Depth Seismic Stations
3378	1975-02-23T02:58:41.000Z	1975-02-23T02:58:41.000Z	8.017	124.075	Earthquake	623.0	NaN	NaN
3379	02/23/1975	03:53:36	-21.727	-71.356	Earthquake	33.0	NaN	NaN
3380	02/23/1975	07:34:11	-10.879	166.667	Earthquake	33.0	NaN	NaN
3381	02/25/1975	05:20:05	-7.388	149.798	Earthquake	33.0	NaN	NaN
3382	02/26/1975	04:48:55	85.047	97.969	Earthquake	33.0	NaN	NaN

5 rows × 21 columns

This does appear to be an issue with data entry: ideally, all entries in the column have the same format. We can get an idea of how widespread this issue is by checking the length of each entry in the "Date" column.

```
In [7]: date_lengths = earthquakes.Date.str.len()
date_lengths.value_counts()
```

```
Out[7]: 10    23409
        24      3
        Name: Date, dtype: int64
```

Looks like there are two more rows that has a date in a different format. Run the code cell below to obtain the indices corresponding to those rows and print the data.

```
In [8]: indices = np.where([date_lengths == 24])[1]
print('Indices with corrupted data:', indices)
earthquakes.loc[indices]
```

Indices with corrupted data: [3378 7512 20650]

```
Out[8]:
```

	Date	Time	Latitude	Longitude	Type	Depth	Depth Error	Depth Seism Station
3378	1975-02-23T02:58:41.000Z	1975-02-23T02:58:41.000Z	8.017	124.075	Earthquake	623.0	NaN	Na
7512	1985-04-28T02:53:41.530Z	1985-04-28T02:53:41.530Z	-32.998	-71.766	Earthquake	33.0	NaN	Na
20650	2011-03-13T02:23:34.520Z	2011-03-13T02:23:34.520Z	36.344	142.344	Earthquake	10.1	13.9	289

3 rows × 21 columns

Given all of this information, it's your turn to create a new column "date_parsed" in the earthquakes dataset that has correctly parsed dates in it.

Note: When completing this problem, you are allowed to (but are not required to) amend the entries in the "Date" and "Time" columns. Do not remove any rows from the dataset.

```
In [19]: # TODO: Your code here

earthquakes.loc[3378, "Date"] = "02/23/1975"
earthquakes.loc[7512, "Date"] = "04/28/1985"
earthquakes.loc[20650, "Date"] = "03/13/2011"
earthquakes['date_parsed'] = pd.to_datetime(earthquakes['Date'], format="%m/%d, %Y")

# Check your answer
q2.check()
```

Correct

```
In [10]: # Lines below will give you a hint or solution code
#q2.hint()
#q2.solution()
```

3) Select the day of the month

Create a Pandas Series `day_of_month_earthquakes` containing the day of the month from the "date_parsed" column.

```
In [20]: # try to get the day of the month from the date column
day_of_month_earthquakes = earthquakes['date_parsed'].dt.day

# Check your answer
q3.check()
```

Correct

```
In [12]: # Lines below will give you a hint or solution code
#q3.hint()
#q3.solution()
```

4) Plot the day of the month to check the date parsing

Plot the days of the month from your earthquake dataset.

```
In [13]: # TODO: Your code here!
```

Does the graph make sense to you?

```
In [14]: # Check your answer (Run this code cell to receive credit!)
q4.check()
```

Correct:

The graph should make sense: it shows a relatively even distribution in days of the month, which is what we would expect.

```
In [15]: # Line below will give you a hint
#q4.hint()
```

(Optional) Bonus Challenge

For an extra challenge, you'll work with a [Smithsonian dataset](https://www.kaggle.com/smithsonian/volcanic-eruptions) (<https://www.kaggle.com/smithsonian/volcanic-eruptions>) that documents Earth's volcanoes and their eruptive history over the past 10,000 years

Run the next code cell to load the data.

```
In [16]: volcanos = pd.read_csv("../input/volcanic-eruptions/database.csv")
```

Try parsing the column "Last Known Eruption" from the `volcanos` dataframe. This column contains a mixture of text ("Unknown") and years both before the common era (BCE, also known as BC) and in the common era (CE, also known as AD).

```
In [17]: volcanos['Last Known Eruption'].sample(5)
```

```
Out[17]: 764      Unknown
1069     1996 CE
34      1855 CE
489     2016 CE
9       1302 CE
Name: Last Known Eruption, dtype: object
```

(Optional) More practice

If you're interested in graphing time series, [check out this tutorial](https://www.kaggle.com/residentmario/time-series-plotting-optional) (<https://www.kaggle.com/residentmario/time-series-plotting-optional>).

You can also look into passing columns that you know have dates in them the `parse_dates` argument in `read_csv`. (The documentation [is here](https://pandas.pydata.org/pandas-docs/stable/generated/pandas.read_csv.html) (https://pandas.pydata.org/pandas-docs/stable/generated/pandas.read_csv.html)). Do note that this method can be very slow, but depending on your needs it may sometimes be handy to use.

Keep going

In the next lesson, learn how to [work with character encodings](https://www.kaggle.com/alexisbcook/character-encodings) (<https://www.kaggle.com/alexisbcook/character-encodings>).

Have questions or comments? Visit the [course discussion forum](https://www.kaggle.com/learn/data-cleaning/discussion) (<https://www.kaggle.com/learn/data-cleaning/discussion>) to chat with other learners.