**Data Cleaning: Working with Time Stamps**

**Selecting an Individual Cell**

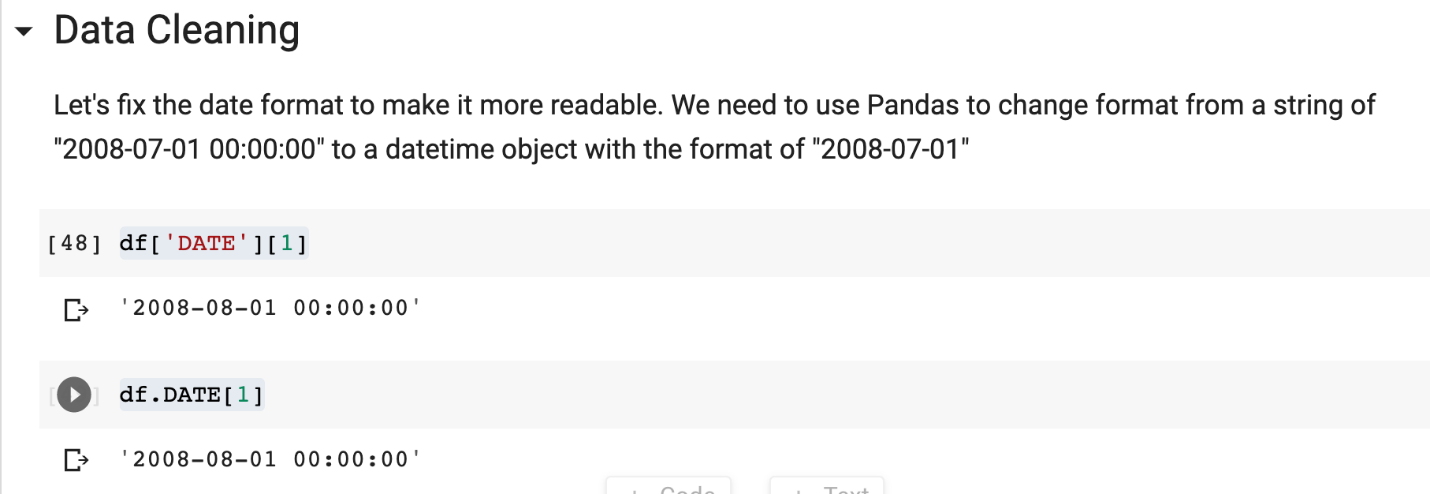
Let's take a closer look at the 'DATE' column in our DataFrame. We can use the double square bracket notation to look at the second entry in the column:

1. df['DATE'][1]

Alternatively, for column names no spaces, we can also use the dot-notation:

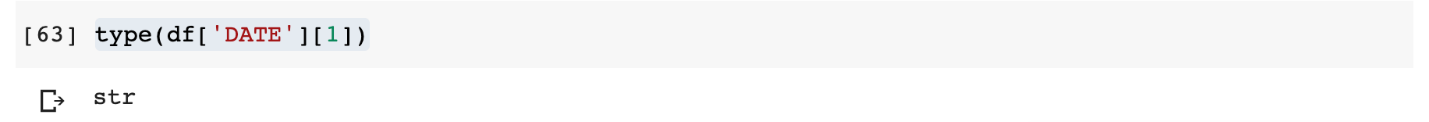
1. df.DATE[1]

I prefer the square bracket notation for column names since it's more flexible, but with the dot notation, you get to use autocomplete, which is also nice.



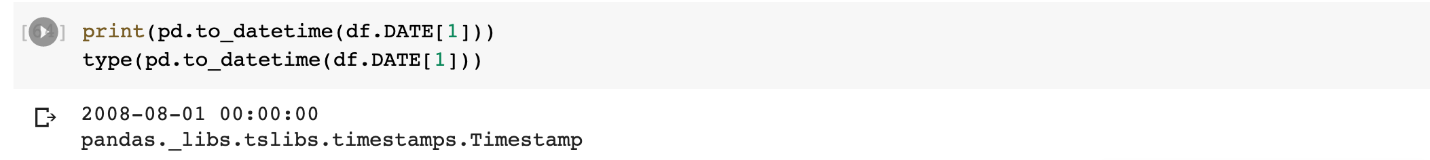
**Inspecting the Data Type**

When we type check the contents of this cell, we see that we are not dealing with a date object, but rather with a string.



This is not very handy. Not only will the string format always show the unnecessary 00:00:00, but we also don't get the benefit of working with Datetime objects, which know how to handle dates and times. Pandas can help us convert the string to a timestamp using the [to\_datetime()](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.to_datetime.html" \t "_blank) method.

Here's how we can convert the entry in our cell and check that it worked:



Let's use Pandas' to\_datetime() to convert the entire df['DATE'] column.



Excellent. Now we can start thinking about how to manipulate our data so that we get a one column per programming language. For all of that and more, I'll see you in the next lesson.