**Explore and Clean the Data**

As with any dataset, the first steps are going to be data exploration and data cleaning. We need to get a better understanding of what we're dealing with. Since you've gone through this process a number of times before on previous days, can you tackle the following challenges on your own?

Challenge 1

Answer these questions about how the data is structured:

1. How many rows and columns does the dataset contain?
2. Are there any NaN values present?
3. Are there any duplicate rows?
4. What are the data types of the columns?

Challenge 2

Convert the USD\_Production\_Budget, USD\_Worldwide\_Gross, and USD\_Domestic\_Gross columns to a numeric format by removing $ signs and ,.

Note that *domestic* in this context refers to the United States.

Challenge 3

Convert the Release\_Date column to a Pandas Datetime type.

.

.

..

...

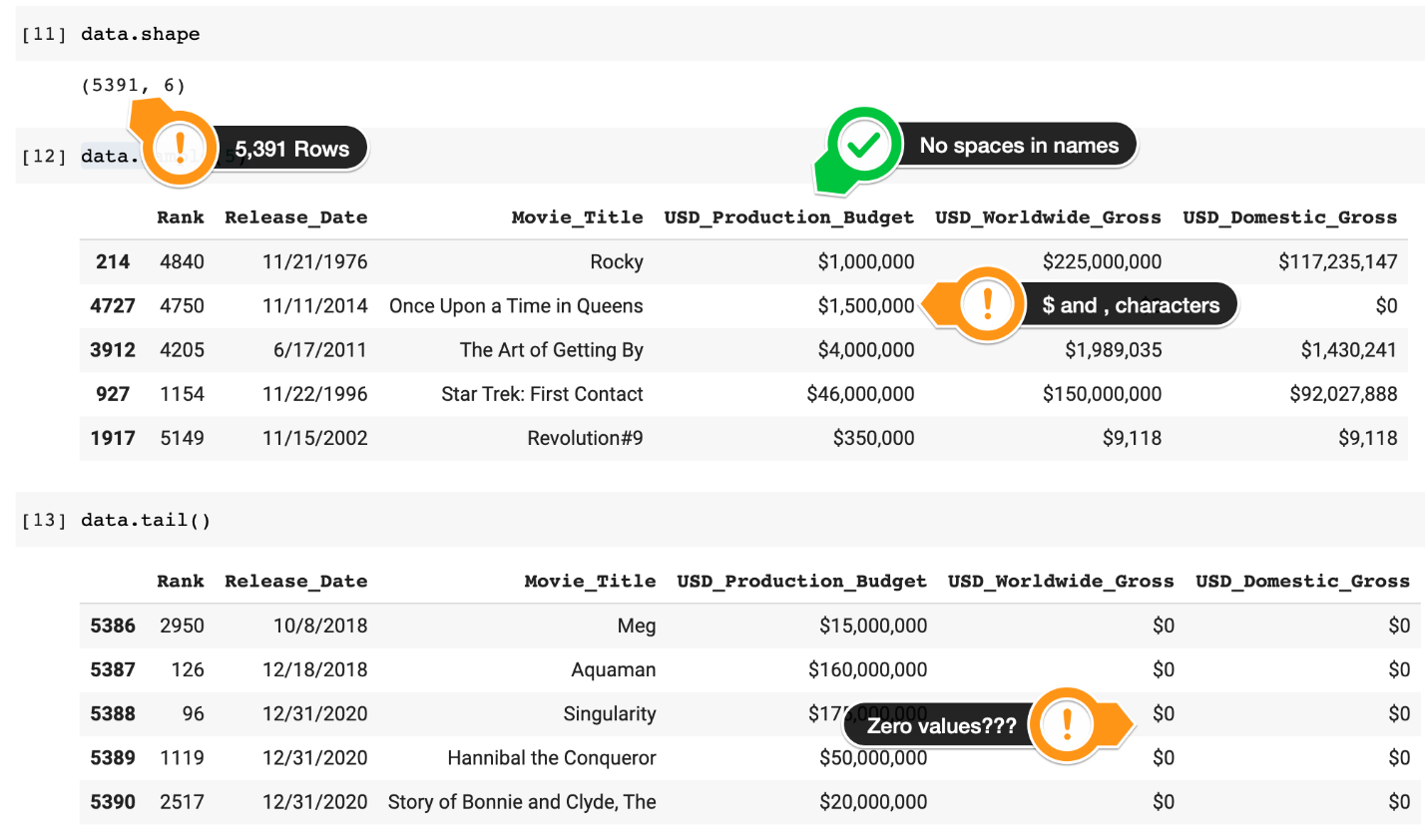
..

.

.

**Solution for Challenge 1**

With any new dataset, it's a good idea to do some standard checks and conversions. I typically always first look at .shape, .head(), .tail(), .info() and .sample().  Here's what I'm spotting already:



There are thousands of entries in the DataFrame - one entry for each movie. We'll have some challenges formatting the data before we can do more analysis because we have non-numeric characters in our budget and revenue columns.

We can check for NaN values with this line:

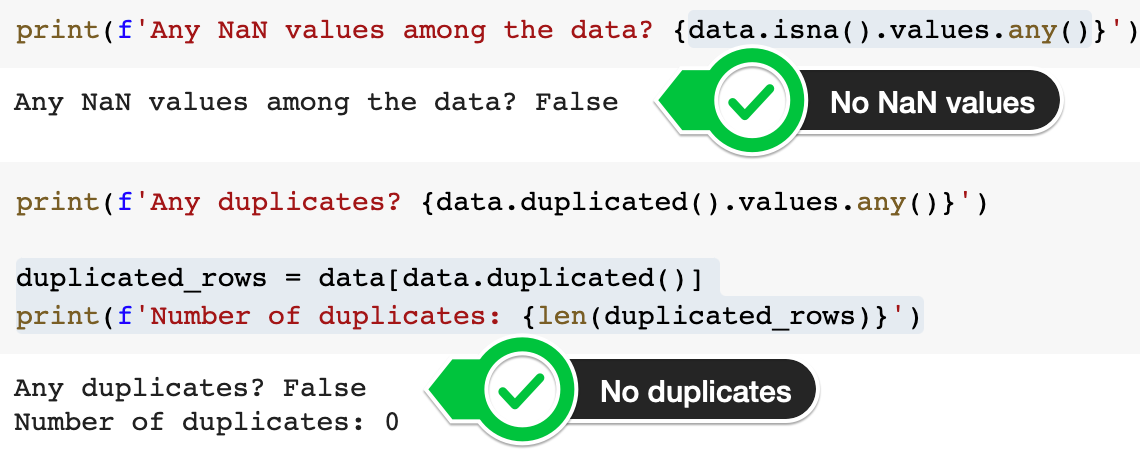
1. data.isna().values.any()

And check for duplicates with this line:

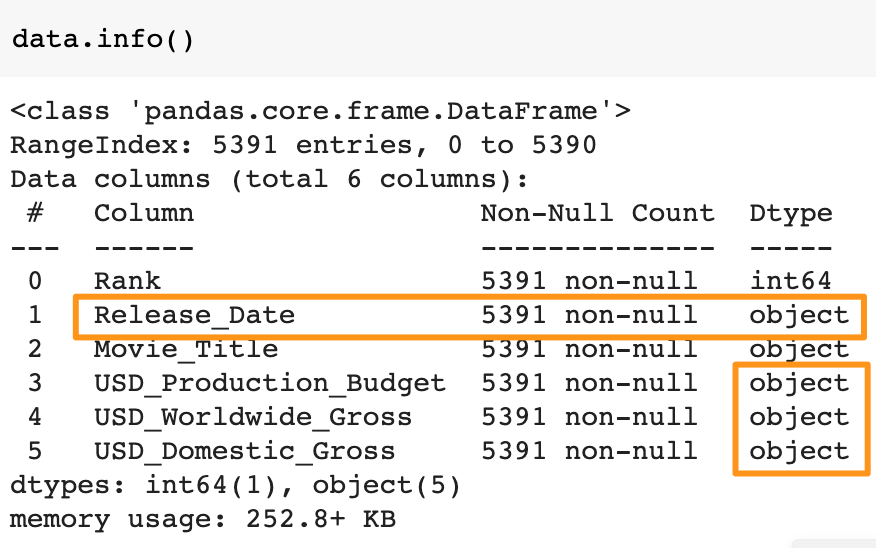
1. data.duplicated().values.any()

We can see the total number of duplicates by creating a subset and looking at the length of that subset:

1. duplicated\_rows = data[data.duplicated()]
2. len(duplicated\_rows)



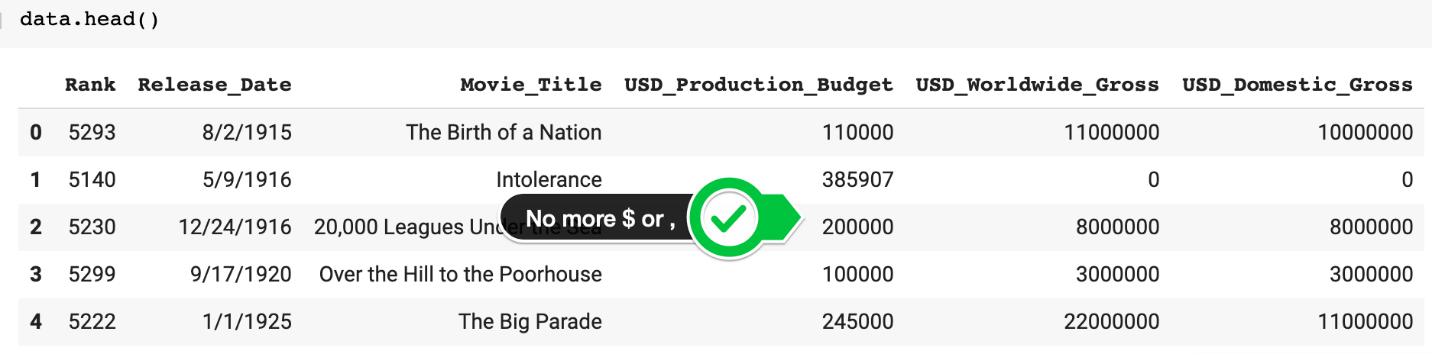
The fact that there are no duplicates or NaN (not-a-number) values in the dataset will make our job easier. We can also see if there are null values in .info(), which also shows us that we need to do some type conversion.



**Solution for Challenge 2**

In order to convert the data in the budget and revenue columns and remove all the non-numeric characters, we can use a nested for loop. We create two Python lists: the characters to remove and the column names. Inside the nested loop we can combine .replace() and .to\_numeric() to achieve our goal.

1. chars\_to\_remove = [',', '$']
2. columns\_to\_clean = ['USD\_Production\_Budget',
3. 'USD\_Worldwide\_Gross',
4. 'USD\_Domestic\_Gross']
6. for col in columns\_to\_clean:
7. for char in chars\_to\_remove:
8. # Replace each character with an empty string
9. data[col] = data[col].astype(str).str.replace(char, "")
10. # Convert column to a numeric data type
11. data[col] = pd.to\_numeric(data[col])



**Solution for Challenge 3**

To convert the Release\_Date column to a DateTime object, all we need to do is call the to\_datetime() function.

1. data.Release\_Date = pd.to\_datetime(data.Release\_Date)

When we check .info() again we see that the columns now have the desired data type. This allows us to proceed with the next parts of our analysis.

