Renaming and Combining

Data comes in from many sources. Help it all make sense together

Tutorial Data

Learn Tutorial
Pandas

Course step
6 of 6

Introduction

Oftentimes data will come to us with column names, index names, or other naming conventions that we are not satisfied with. In that case, you'll learn how to use pandas functions to change the names of the offending entries to something better.

You'll also explore how to combine data from multiple DataFrames and/or Series.

To start the exercise for this topic, please click here.

Renaming

The first function we'll introduce here is `rename()`, which lets you change index names and/or column names. For example, to change the `points` column in our dataset to `score`, we would do:

Show hidden code

In [2]:

```
reviews.rename(columns={'points': 'score'})
```

	country	description	designation	score	price	province	region_1	region_2	taster_name	taster_twitter_handle	title	varie
0	Italy	Aromas include tropical fruit, broom, brimston...	Vulkà Bianco	87	NaN	Sicily & Sardinia	Etna	NaN	Kerin O'Keefe	@kerinokeefe	Nicosia 2013 Vulkà Bianco (Etna)	Whit
1	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	87	15.0	Douro	NaN	NaN	Roger Voss	@vossroger	Quinta dos Avidagos 2011 Avidagos Red (Douro)	Portu
...	...	...	...	...	...	...	...	...	...	...	...	...
129969	France	A dry style of Pinot Gris, this is crisp with ...	NaN	90	32.0	Alsace	Alsace	NaN	Roger Voss	@vossroger	Domaine Marcel Deiss 2012 Pinot Gris (Alsace)	Pinot
129970	France	Big, rich and off-dry, this is powered by inte...	Lieu-dit Harth Cuvée Caroline	90	21.0	Alsace	Alsace	NaN	Roger Voss	@vossroger	Domaine Schoffit 2012 Lieu-dit Harth Cuvée Car...	Gew

`rename()` lets you rename index *or* column values by specifying a `index` or `column` keyword parameter, respectively. It supports a variety of input formats, but usually a Python dictionary is the most convenient. Here is an example using it to rename some elements of the index.

In [3]:

```
reviews.rename(index={0: 'firstEntry', 1: 'secondEntry'})
```

	country	description	designation	points	price	province	region_1	region_2	taster_name	taster_twitter_handle	title
firstEntry	Italy	Aromas include tropical fruit, broom, brimston...	Vulkà Bianco	87	NaN	Sicily & Sardinia	Etna	NaN	Kerin O'Keefe	@kerinokeefe	Nicosia 2013 Vulkà Bianco (Etna)
secondEntry	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	87	15.0	Douro	NaN	NaN	Roger Voss	@vossroger	Quinta dos Avidagos 2011 Avidagos Red (Douro)
...	...	...	...	...	...	...	...	...	...	...	...
129969	France	A dry style of Pinot Gris, this is crisp with ...	NaN	90	32.0	Alsace	Alsace	NaN	Roger Voss	@vossroger	Domaine Marcel Deiss 2012 Pinot Gris (Alsace)
129970	France	Big, rich and off-dry, this is powered by inte...	Lieu-dit Harth Cuvée Caroline	90	21.0	Alsace	Alsace	NaN	Roger Voss	@vossroger	Domaine Schoffit 2012 Lieu-dit Harth Cuvée Car...

You'll probably rename columns very often, but rename index values very rarely. For that, `set_index()` is usually more convenient.

Both the row index and the column index can have their own `name` attribute. The complimentary `rename_axis()` method may be used to change these names. For example:

```
In [4]:
reviews.rename_axis("wines", axis='rows').rename_axis("fields", axis='columns')
```

fields	country	description	designation	points	price	province	region_1	region_2	taster_name	taster_twitter_handle	title	vari
wines												
0	Italy	Aromas include tropical fruit, broom, brimston...	Vulkà Bianco	87	NaN	Sicily & Sardinia	Etna	NaN	Kerin O'Keefe	@kerinokeefe	Nicosia 2013 Vulkà Bianco (Etna)	Whi
1	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	87	15.0	Douro	NaN	NaN	Roger Voss	@vossroger	Quinta dos Avidagos 2011 Avidagos Red (Douro)	Port
...	...	...	...	...	...	...	...	...	...	...	...	...
129969	France	A dry style of Pinot Gris, this is crisp with ...	NaN	90	32.0	Alsace	Alsace	NaN	Roger Voss	@vossroger	Domaine Marcel Deiss 2012 Pinot Gris (Alsace)	Pinc
129970	France	Big, rich and off-dry, this is powered by inte...	Lieu-dit Harth Cuvée Caroline	90	21.0	Alsace	Alsace	NaN	Roger Voss	@vossroger	Domaine Schoffit 2012 Lieu-dit Harth Cuvée Car...	Gew

Combining

When performing operations on a dataset, we will sometimes need to combine different DataFrames and/or Series in non-trivial ways. Pandas has three core methods for doing this. In order of increasing complexity, these are `concat()`, `join()`, and `merge()`. Most of what `merge()` can do can also be done more simply with `join()`, so we will omit it and focus on the first two functions here.

The simplest combining method is `concat()`. Given a list of elements, this function will smush those elements together along an axis.

This is useful when we have data in different DataFrame or Series objects but having the same fields (columns). One example: the YouTube Videos dataset, which splits the data up based on country of origin (e.g. Canada and the UK, in this example). If we want to study multiple countries simultaneously, we can use `concat()` to smush them together:

In [5]:

```
canadian_youtube = pd.read_csv("../input/youtube-new/CAvideos.csv")
british_youtube = pd.read_csv("../input/youtube-new/GBvideos.csv")

pd.concat([canadian_youtube, british_youtube])
```

Out[5]:

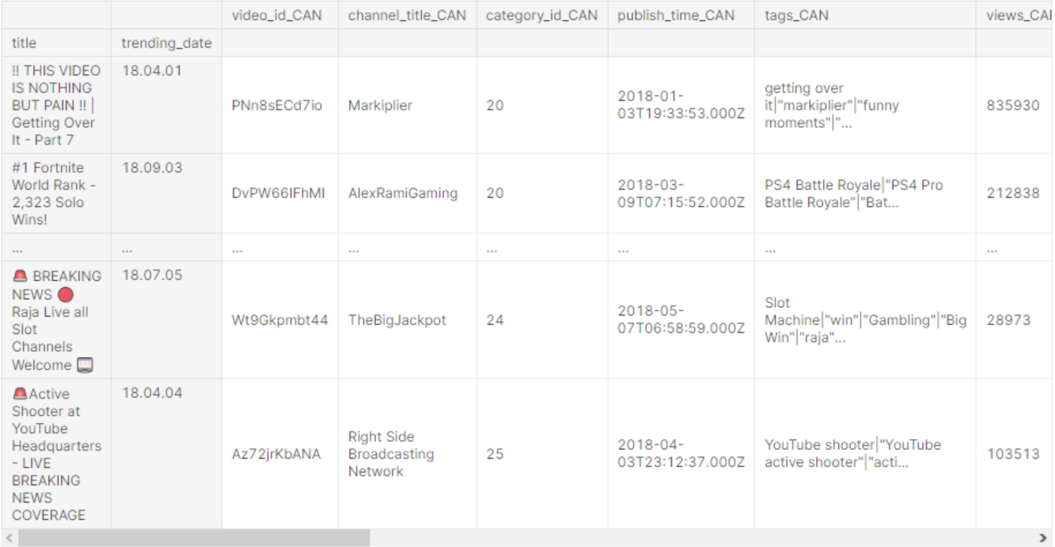
	video_id	trending_date	title	channel_title	category_id	publish_time	tags
0	n1WpP7iowLc	17.14.11	Eminem - Walk On Water (Audio) ft. Beyoncé	EminemVEVO	10	2017-11-10T17:00:03.000Z	Eminem|"Walk"|"On"|"Water"|"Aftermath/Sha
1	0dBIkQ4Mz1M	17.14.11	PLUSH - Bad Unboxing Fan Mail	iDubbbzTV	23	2017-11-13T17:00:00.000Z	plush|"bad unboxing"|"unboxing"|"fan mail"|"
...	...	...	...	...	...	...	...
38914	-DRsfNObKIQ	18.14.06	Eleni Foureira - Fuego - Cyprus - LIVE - First...	Eurovision Song Contest	24	2018-05-08T20:32:32.000Z	Eurovision Song Contest|"2018"|"Lisbon"|"C
38915	4YFo4bdMO8Q	18.14.06	KYLE - Ikuyo feat. 2 Chainz & Sophia Black [A...	SuperDuperKyle	10	2018-05-11T04:06:35.000Z	Kyle|"SuperDuperKyle"|"Ikuyo"|"2 Chainz"|"S

The middlemost combiner in terms of complexity is `join()`. `join()` lets you combine different DataFrame objects which have an index in common. For example, to pull down videos that happened to be trending on the same day in *both* Canada and the UK, we could do the following:

In [6]:

```
left = canadian_youtube.set_index(['title', 'trending_date'])
right = british_youtube.set_index(['title', 'trending_date'])

left.join(right, lsuffix='_CAN', rsuffix='_UK')
```



		video_id_CAN	channel_title_CAN	category_id_CAN	publish_time_CAN	tags_CAN	views_CA
title	trending_date						
!! THIS VIDEO IS NOTHING BUT PAIN !! | Getting Over It - Part 7	18.04.01	PNn8sECd7io	Markiplier	20	2018-01-03T19:33:53.000Z	getting over it|"markiplier"|"funny moments"|"...	835930
#1 Fortnite World Rank - 2,323 Solo Wins!	18.09.03	DvPW66IFhMI	AlexRamiGaming	20	2018-03-09T07:15:52.000Z	PS4 Battle Royale|"PS4 Pro Battle Royale"|"Bat...	212838
...	...	...	...	...	...	...	...
🚨 BREAKING NEWS 🔴 Raja Live all Slot Channels Welcome 🎰	18.07.05	Wt9Gkpmbt44	TheBigJackpot	24	2018-05-07T06:58:59.000Z	Slot Machine|"win"|"Gambling"|"Big Win"|"raja"...	28973
🚨Active Shooter at YouTube Headquarters - LIVE BREAKING NEWS COVERAGE	18.04.04	Az72jrKbANA	Right Side Broadcasting Network	25	2018-04-03T23:12:37.000Z	YouTube shooter|"YouTube active shooter"|"acti...	103513

The `lsuffix` and `rsuffix` parameters are necessary here because the data has the same column names in both British and Canadian datasets. If this wasn't true (because, say, we'd renamed them beforehand) we wouldn't need them.