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.

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:

In [1]:
```
import pandas as pd

reviews = pd.read_csv("datasets/winemag-data-130k-v2.csv", index_col=0)
```

In [2]:
```
reviews.rename(columns={'points': 'score'})
```

Out[2]:

	country	description	designation	score	price	province	region_1	region
0	Italy	Aromas include tropical fruit, broom, brimston...	Vulkà Bianco	87	NaN	Sicily & Sardinia	Etna	N
1	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	87	15.0	Douro	NaN	N
2	US	Tart and snappy, the flavors of lime flesh and...	NaN	87	14.0	Oregon	Willamette Valley	Willame Vall
3	US	Pineapple rind, lemon pith and orange blossom ...	Reserve Late Harvest	87	13.0	Michigan	Lake Michigan Shore	N
4	US	Much like the regular bottling from 2012, this...	Vintner's Reserve Wild Child Block	87	65.0	Oregon	Willamette Valley	Willame Vall
...	...	...	...	...	...	...	...	
65494	France	Made from young vines from the Vaulorent porti...	Fourchaume Premier Cru	90	45.0	Burgundy	Chablis	N
65495	Australia	This is a big, fat, almost sweet-tasting Caber...	NaN	90	22.0	South Australia	McLaren Vale	N
65496	US	Much improved over the unripe 2005, Fritz's 20...	Estate	90	20.0	California	Dry Creek Valley	Sonor
65497	US	This wine wears its 15.8% alcohol	Block 24	90	31.0	California	Napa Valley	Na

	country	description	designation	score	price	province	region_1	region
		better than ...						
65498	Spain	A unique take on Manzanilla Sherry, which is o...	Manzanilla	90	10.0	Andalucia	Jerez	N

65499 rows × 13 columns

`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'})
```

Out[3]:

	country	description	designation	points	price	province	region_1	
firstEntry	Italy	Aromas include tropical fruit, broom, brimston...	Vulkà Bianco	87	NaN	Sicily & Sardinia	Etna	
secondEntry	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	87	15.0	Douro	NaN	
2	US	Tart and snappy, the flavors of lime flesh and...	NaN	87	14.0	Oregon	Willamette Valley	W
3	US	Pineapple rind, lemon pith and orange blossom ...	Reserve Late Harvest	87	13.0	Michigan	Lake Michigan Shore	
4	US	Much like the regular bottling from 2012, this...	Vintner's Reserve Wild Child Block	87	65.0	Oregon	Willamette Valley	W
...	...	...	...	...	...	...	...	
65494	France	Made from young vines from the Vaulorent porti...	Fourchaume Premier Cru	90	45.0	Burgundy	Chablis	
65495	Australia	This is a big, fat, almost sweet-tasting Caber...	NaN	90	22.0	South Australia	McLaren Vale	
65496	US	Much improved over the unripe 2005, Fritz's 20...	Estate	90	20.0	California	Dry Creek Valley	
65497	US	This wine wears its 15.8% alcohol	Block 24	90	31.0	California	Napa Valley	

	country	description	designation	points	price	province	region_1
		better than ...					
65498	Spain	A unique take on Manzanilla Sherry, which is o...	Manzanilla	90	10.0	Andalucia	Jerez

65499 rows × 13 columns

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='columr
```

Out[4]:

fields wines	country	description	designation	points	price	province	region_1	regior
0	Italy	Aromas include tropical fruit, broom, brimston...	Vulkà Bianco	87	NaN	Sicily & Sardinia	Etna	N
1	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	87	15.0	Douro	NaN	N
2	US	Tart and snappy, the flavors of lime flesh and...	NaN	87	14.0	Oregon	Willamette Valley	Willame Va
3	US	Pineapple rind, lemon pith and orange blossom ...	Reserve Late Harvest	87	13.0	Michigan	Lake Michigan Shore	N
4	US	Much like the regular bottling from 2012, this...	Vintner's Reserve Wild Child Block	87	65.0	Oregon	Willamette Valley	Willame Va
...	...	...	...	...	...	...	...	
65494	France	Made from young vines from the Vaulorent porti...	Fourchaume Premier Cru	90	45.0	Burgundy	Chablis	N
65495	Australia	This is a big, fat, almost sweet-tasting Caber...	NaN	90	22.0	South Australia	McLaren Vale	N
65496	US	Much improved over the unripe 2005, Fritz's 20...	Estate	90	20.0	California	Dry Creek Valley	Sonc
[illegible]	US	This wine wears its	Block 24	90	31.0	California	Napa Valley	Na

fields	country	description	designation	points	price	province	region_1	regior
wines								
		15.8% alcohol better than ...						
65498	Spain	A unique take on Manzanilla Sherry, which is o...	Manzanilla	90	10.0	Andalucia	Jerez	N

65499 rows × 13 columns

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("datasets/CAvideos.csv")
british_youtube = pd.read_csv("datasets/GBvideos.csv")

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

Out[5]:

	video_id	trending_date	title	channel_title	category_id	
0	n1WpP7iowLc	17.14.11	Eminem - Walk On Water (Audio) ft. Beyoncé	EminemVEVO	10	1(
1	0dBIkQ4Mz1M	17.14.11	PLUSH - Bad Unboxing Fan Mail	iDubbbzTV	23	13
2	5qpjK5DgCt4	17.14.11	Racist Superman | Rudy Mancuso, King Bach & Le...	Rudy Mancuso	23	12
3	d380meD0W0M	17.14.11	I Dare You: GOING BALD!?	nigahiga	24	1
4	2Vv-BfVoq4g	17.14.11	Ed Sheeran - Perfect (Official Music Video)	Ed Sheeran	10	0
...	...	...	...	...	...	
38911	l884wKofd54	18.14.06	Enrique Iglesias - MOVE TO MIAMI (Official Vid...	EnriqueIglesiasVEVO	10	0
38912	IP8k2xkhOdI	18.14.06	Jacob Sartorius - Up With It (Official Music V...	Jacob Sartorius	10	1
38913	ll-an3K9pjg	18.14.06	Anne-Marie - 2002 [Official Video]	Anne-Marie	10	0
38914	-DRsfNObKIQ	18.14.06	Eleni Foureira - Fuego - Cyprus - LIVE - First...	Eurovision Song Contest	24	0

	video_id	trending_date	title	channel_title	category_id	
38915	4YFo4bdMO8Q	18.14.06	KYLE - Ikuyo feat. 2 Chainz & Sophia Black [A...	SuperDuperKyle	10	11

79797 rows × 16 columns

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='_CA', rsuffix='_GB')
```

Out[6]:

title	trending_date	video_id_CA	channel_title_CA	category_id_CA	publis
!! THIS VIDEO IS NOTHING BUT PAIN !! | Getting Over It - Part 7	18.04.01	PNn8sECd7io	Markiplier	20	03T19:
#1 Fortnite World Rank - 2,323 Solo Wins!	18.09.03	DvPW66IFhMI	AlexRamiGaming	20	09T07
#1 Fortnite World Rank - 2,330 Solo Wins!	18.10.03	EXEaMjFeiEk	AlexRamiGaming	20	10T06
#1 MOST ANTICIPATED VIDEO (Timber Frame House Raising)	17.20.12	bYvQmusLaxw	Pure Living for Life	24	20T02
	17.21.12	bYvQmusLaxw	Pure Living for Life	24	20T02
...	...	...	...	...	
😲 She Is So Nervous But BLOWS The ROOF After Taking on OPERA Song! | Britain´s Got Talent 2018	18.02.05	WttN1Z0XF4k	How Talented	24	28T19:
	18.29.04	WttN1Z0XF4k	How Talented	24	28T19:
	18.30.04	WttN1Z0XF4k	How Talented	24	28T19:
🚨 BREAKING NEWS 🔴 Raja Live all Slot Channels Welcome 🎰	18.07.05	Wt9Gkpmbt44	TheBigJackpot	24	07T06:
🚨 Active Shooter at YouTube Headquarters - LIVE BREAKING NEWS COVERAGE	18.04.04	Az72jrKbANA	Right Side Broadcasting Network	25	03T23

40900 rows × 28 columns

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.

Your turn

If you haven't started the exercise, you can start now.