**INFO 6210 Movie Database**

*YURU LI*

*ZHIHUI XIN*

Abstract

The domain of our project is movie. Our goal is to use the data we get from the internet (social media, web scraper, API, raw data, twitter, Instagram, etc.) to build a database which include movie information, actor/actress Instagram data, movie recommend twitter account’s tweet data, Oscar winner’s data, etc. Also, we are going to create a newsfeed as an interface to query this data.

**Keywords**

Movie, database, social media, movie information

I. Introduction

When user input an old movie name or an actor name, we do some algorithm by connecting the data tables, then returns an in-theater movie that the user may want to see, at the same time show some related comments or recent news.

People now really love watching movie on internet. Some people have specific aim of what to watch, other people have no idea and they really want to watch movies to relax.

They might have favorite actors/actress, but they don’t really know what movies they play, and sometimes people just want to watch special genre like action/comedy. So we build the database of movies, and people could search by genre/actors/title. The database will help people find some related movies. It helps people to make a better choice.

II. Data Resource

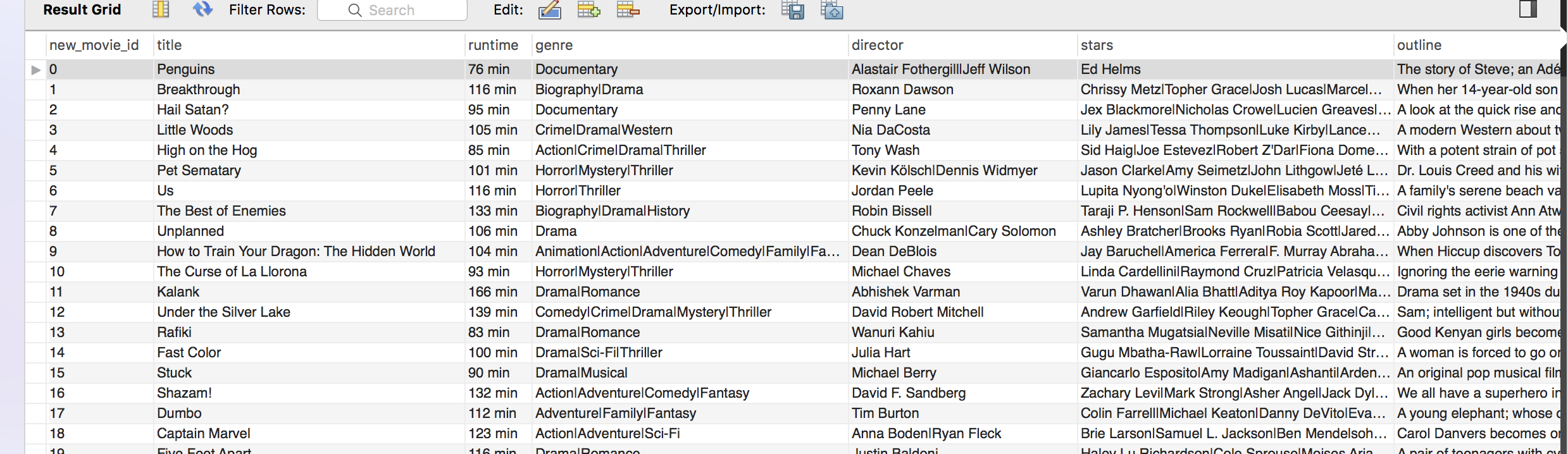
1. **Old movie information from Kaggle**

We got past ten years movie information from Kaggle, but it has too much useless information, and the format need to be cleaned.



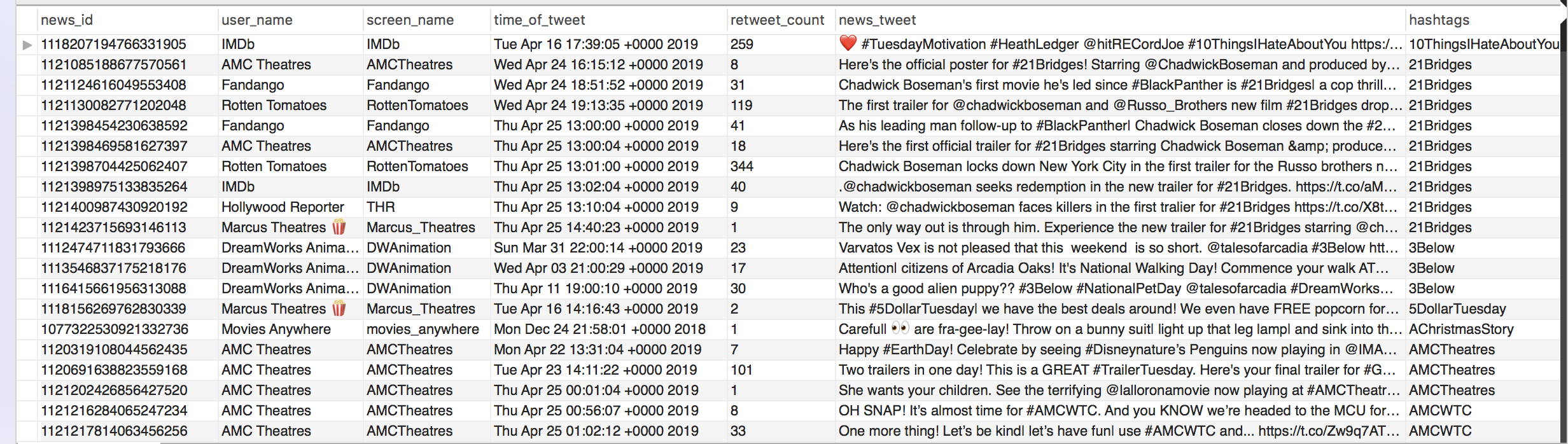
1. **New movie information from IMDB**

From IMDB, we use web scraper to get the information of latest movies. And we add real-time updated function.



**3. Movie context information from Twitter**

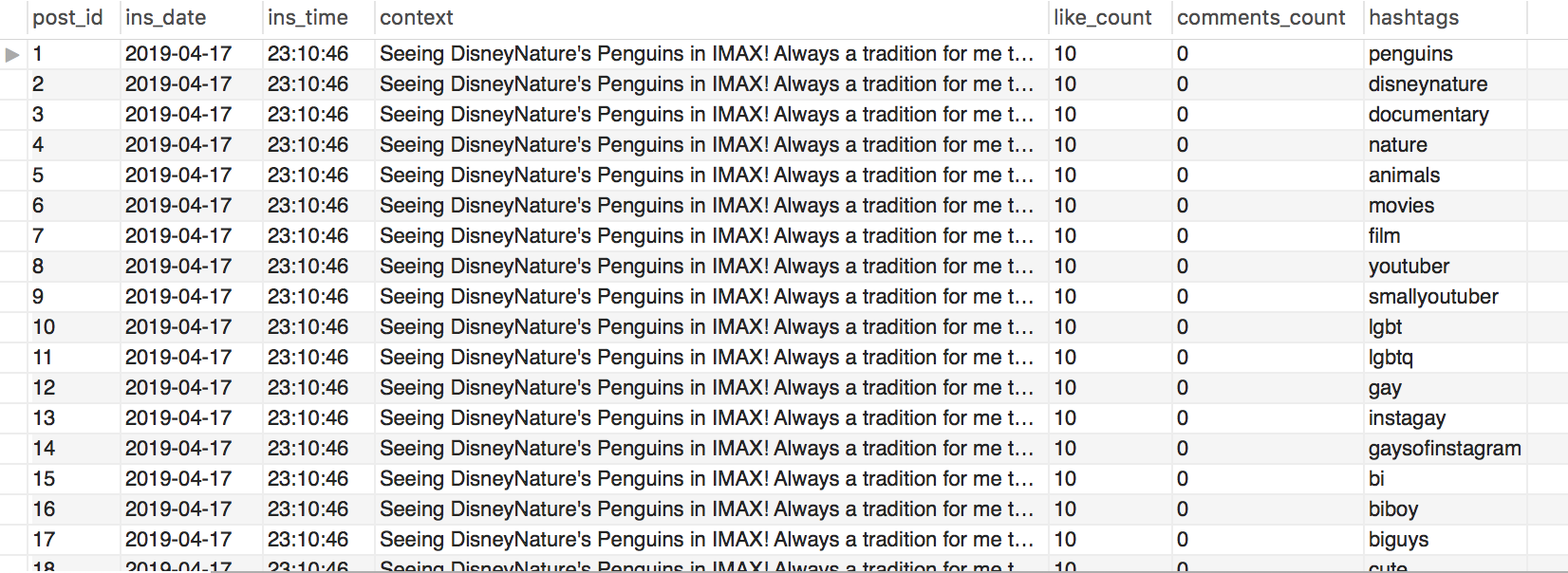
From IMDB, we can get all the new movie titles, use these names as Twitter hashtags names to get the tweet of those movies, get retweet number, time of tweet, etc.

Twitter hashtag

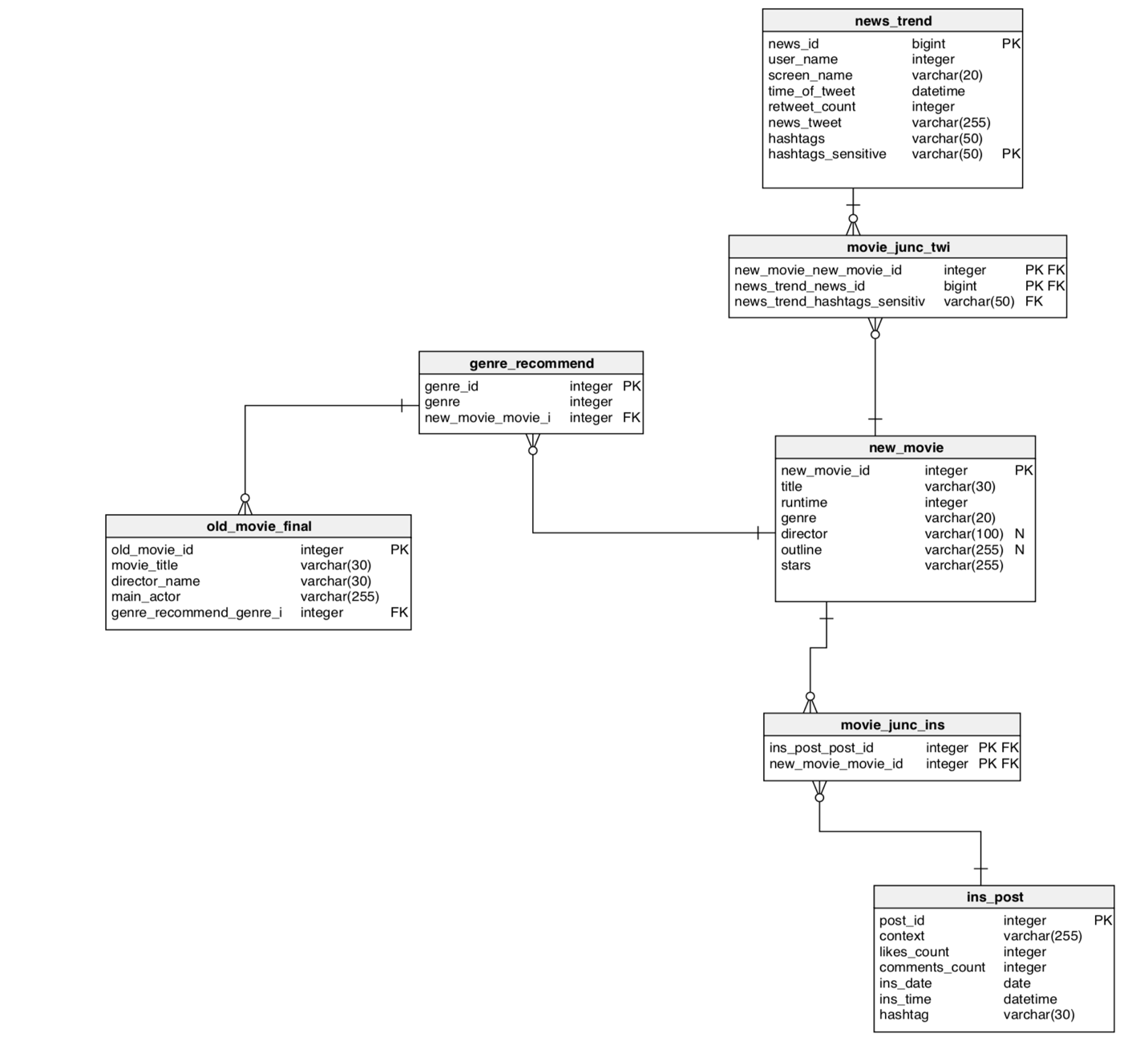
**4. Movie fans information from Instagram**

We use new movies name got from IMDB as hashtag and we use instaloader to get ins information those post posts in specific hashtags. In order to simplify data, we just extract those who posts and get posts likes, post comments and so on. And there might be some related hashtags.

Inshashtag

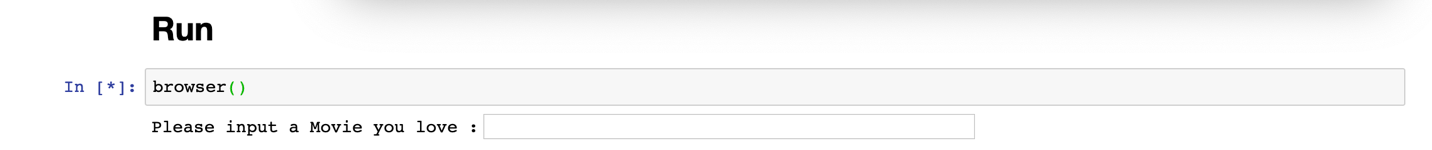


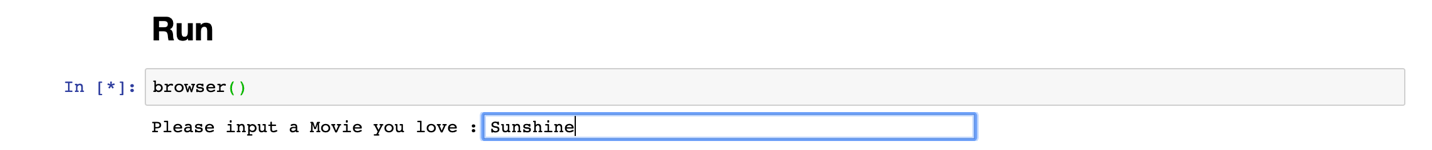
III. ER Diagram



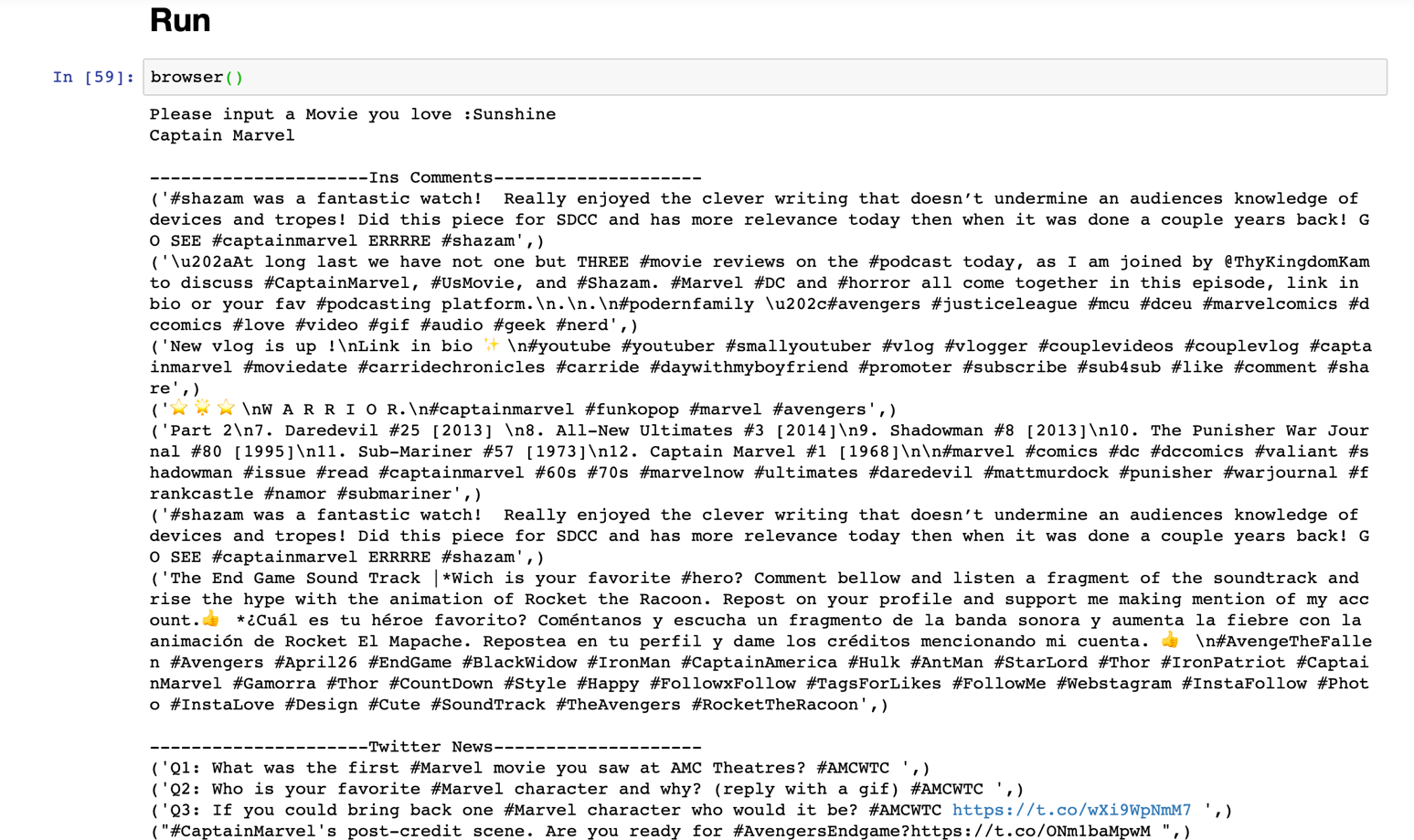
IV. Code with Documentation

Python browser

****

<<Sunshine>> is the movie I love most, we input it into the browser

We get the result <<Captain Marvel>> which has the same genre with <<Sunshine>>.

****

V. News and Trend

The Data for news trend is collected from twitter, I make sure the posts are related to movie by collecting data only from movie news account or movie company account. And I do some following steps to make sure every time I refresh the code, the feedback is the news(posts) about the movie which are in-theater right now:

1. Use web scraper to get all the in-theater movies;
2. Collected all the latest 100 posts of each news account, manage those posts using their hashtags.
3. Using hashtags to match the news with the in-theater movies.

At the time I am writing this example, the in-theater movies are as follows:

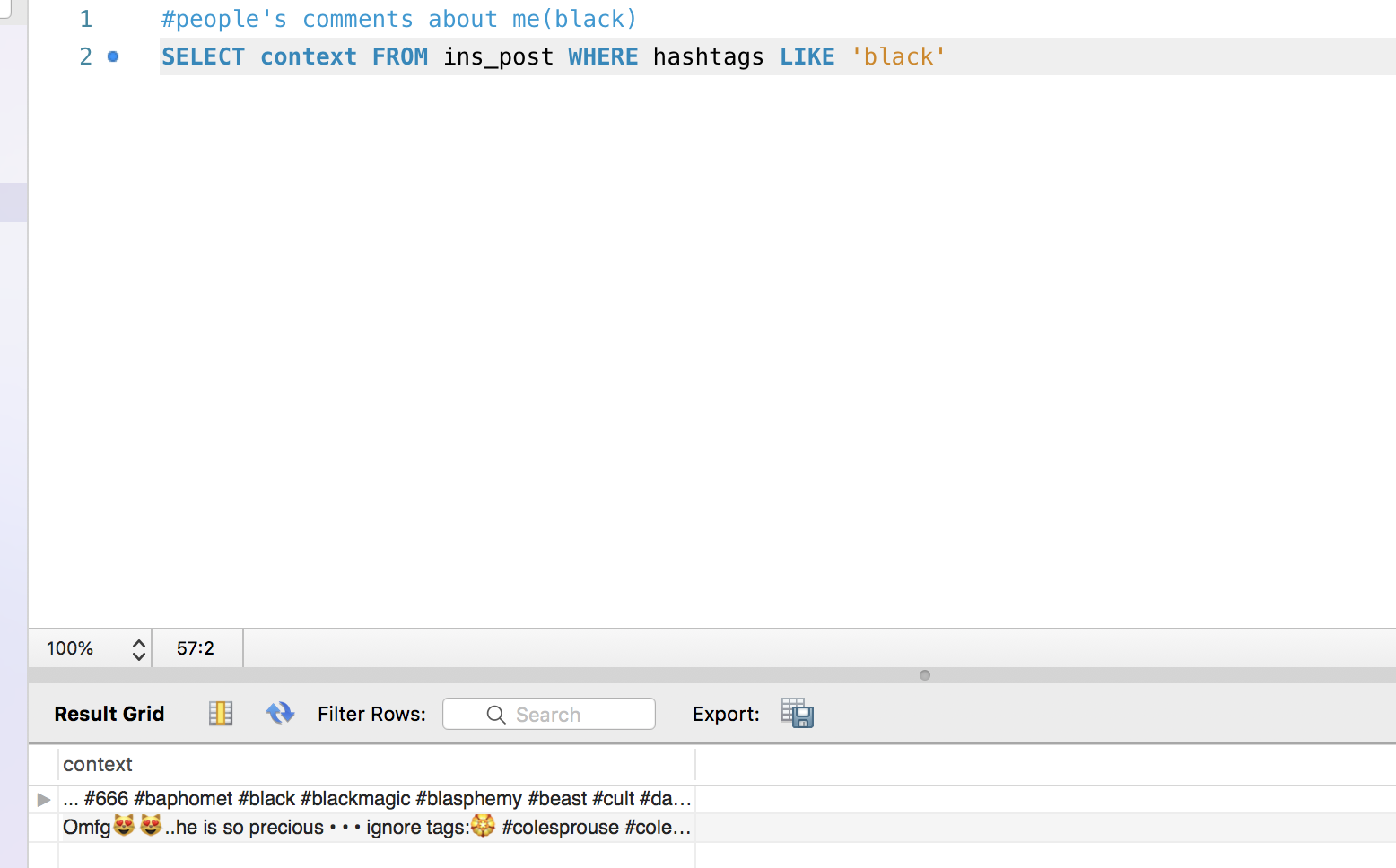


The news examples are as follow:

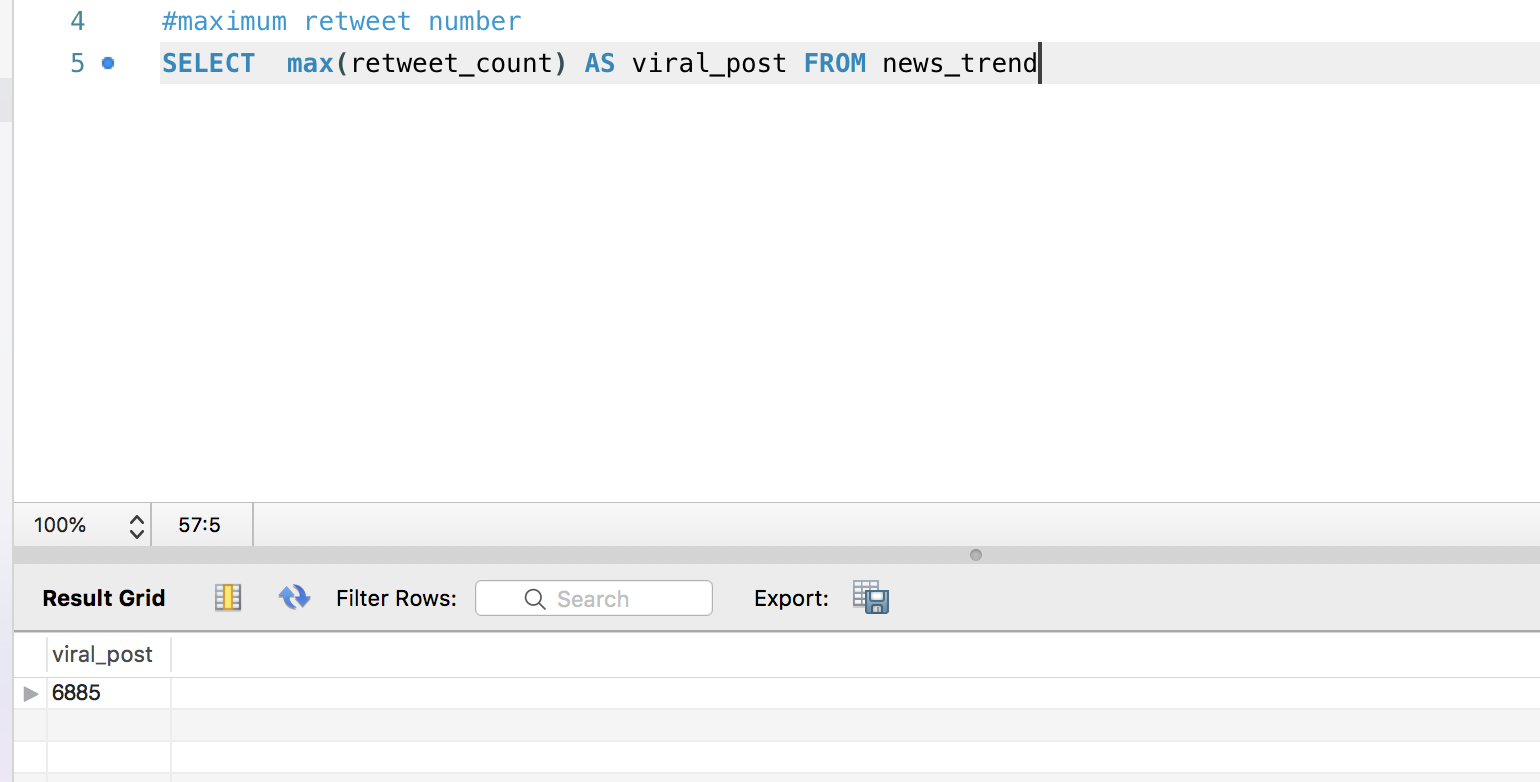


Each news(post) is tagged by at least a tag. And when we join these 2 tables, we get all Latest news related the movie.

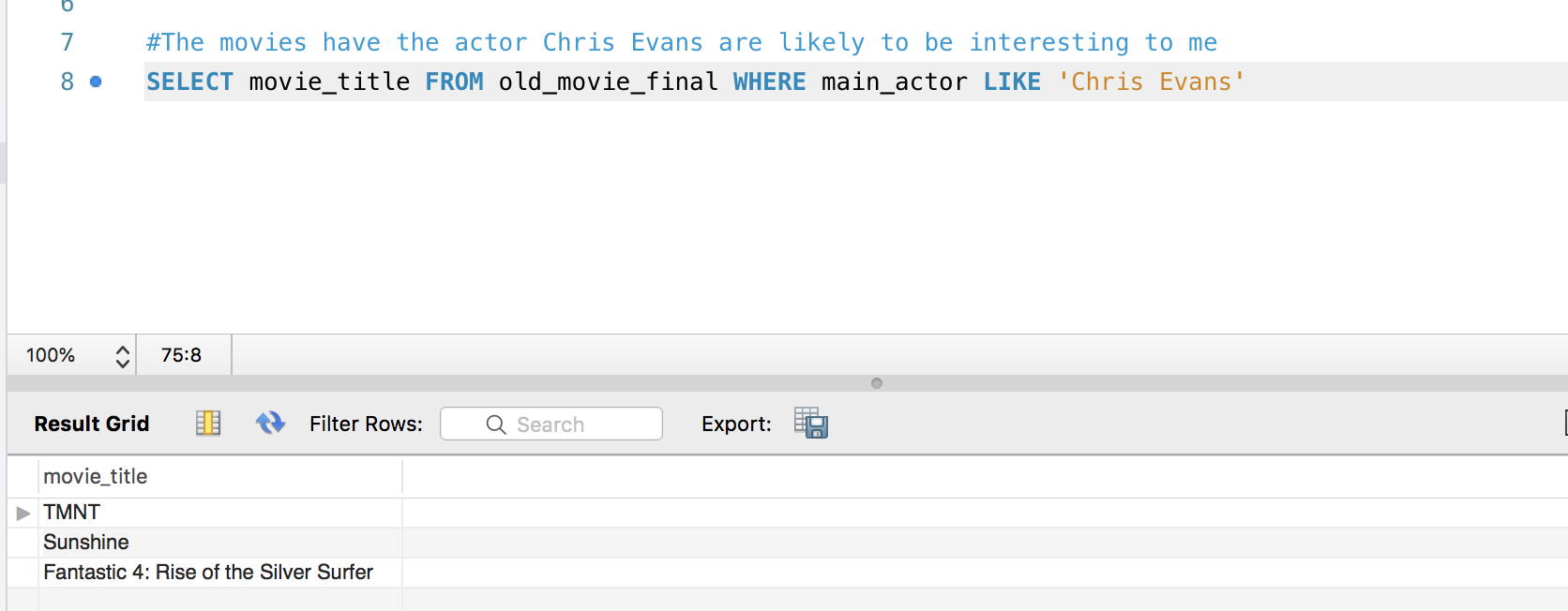
VI. Use case to answer those question

1.What are people saying about me (somebody)?

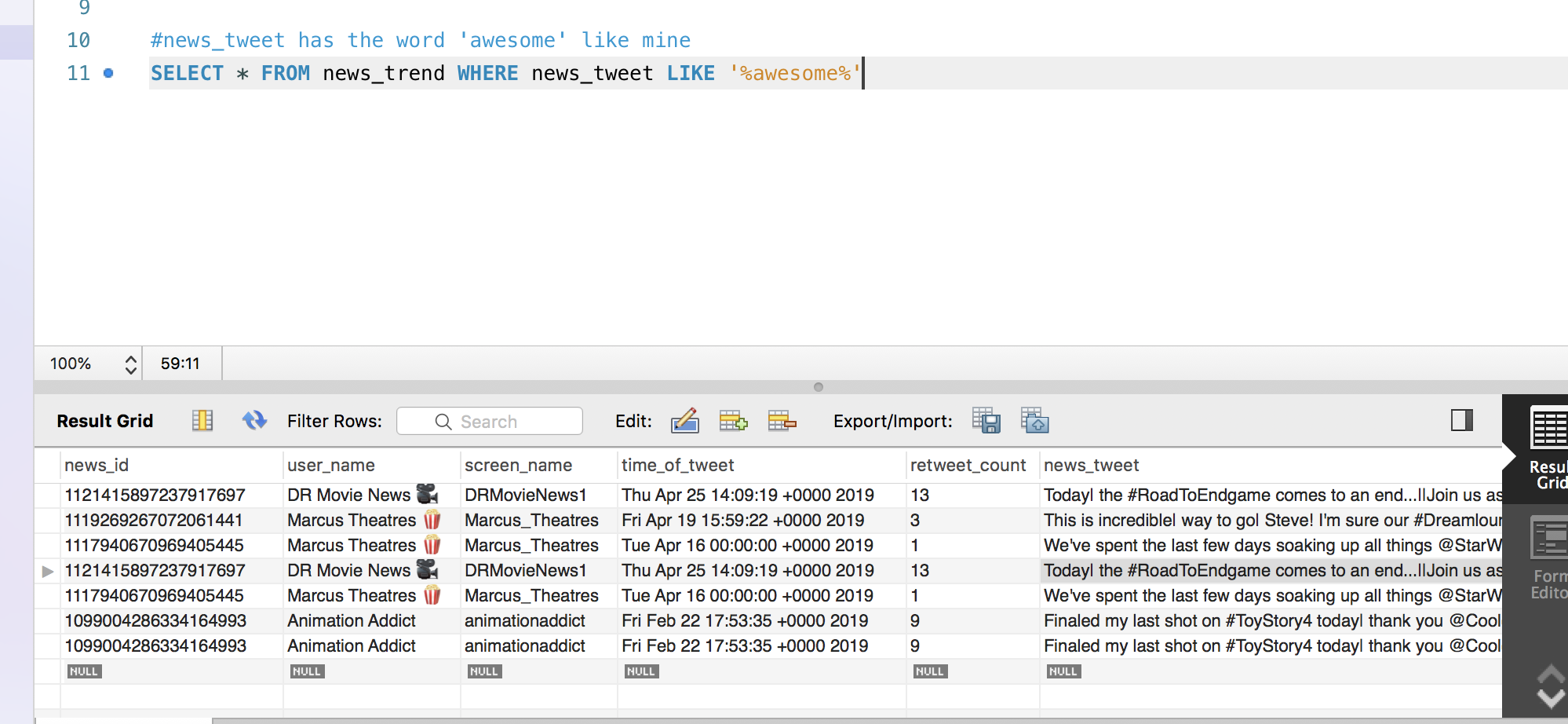
2.How viral are my post?

6885 retweet, so viral!!

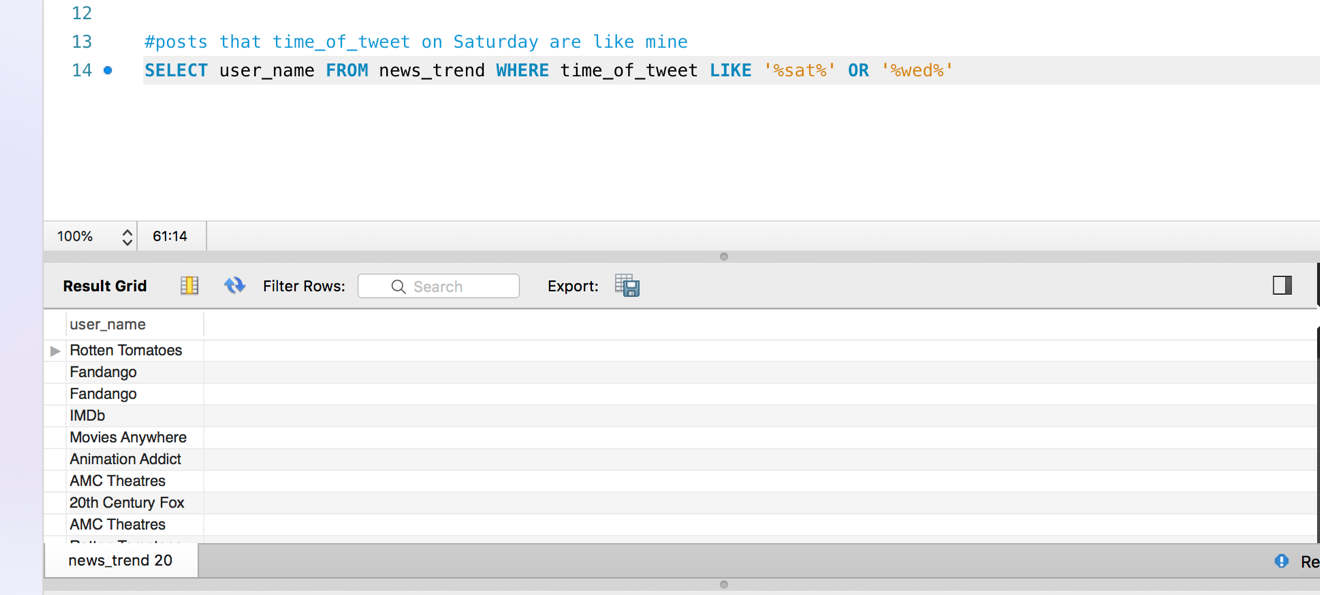
3.What posts are likely to be interesting to me?



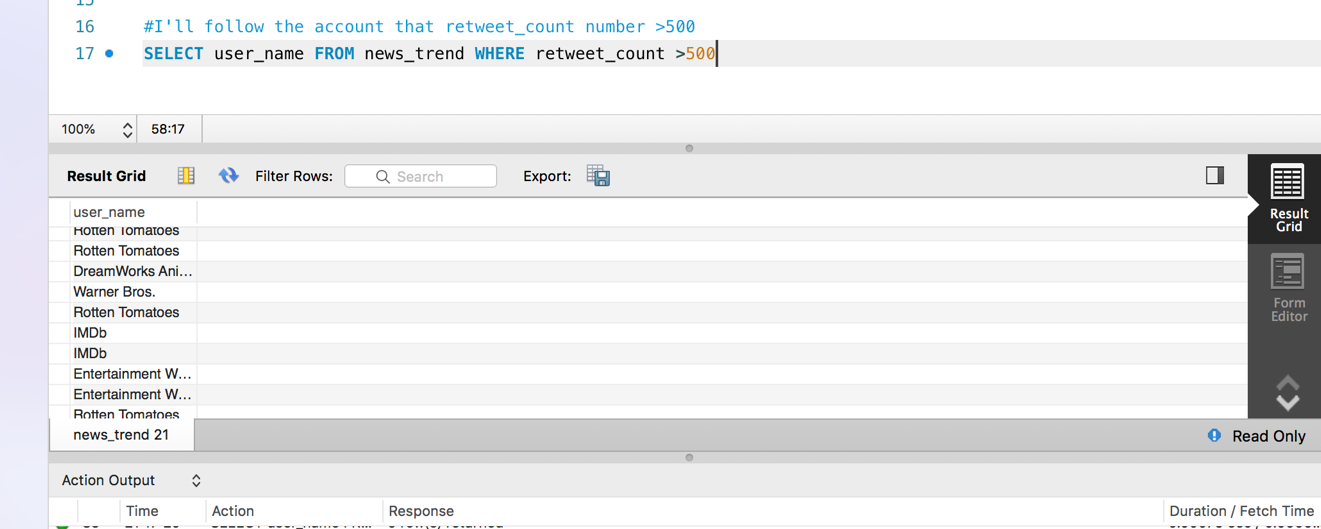
4.What posts are like mine?



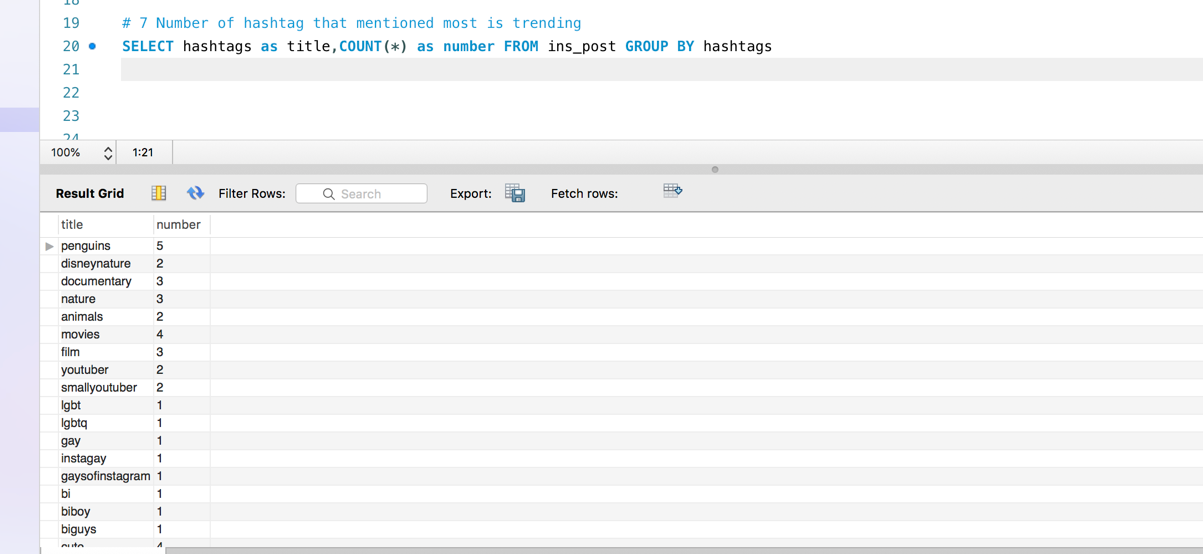
5.What users post like me?

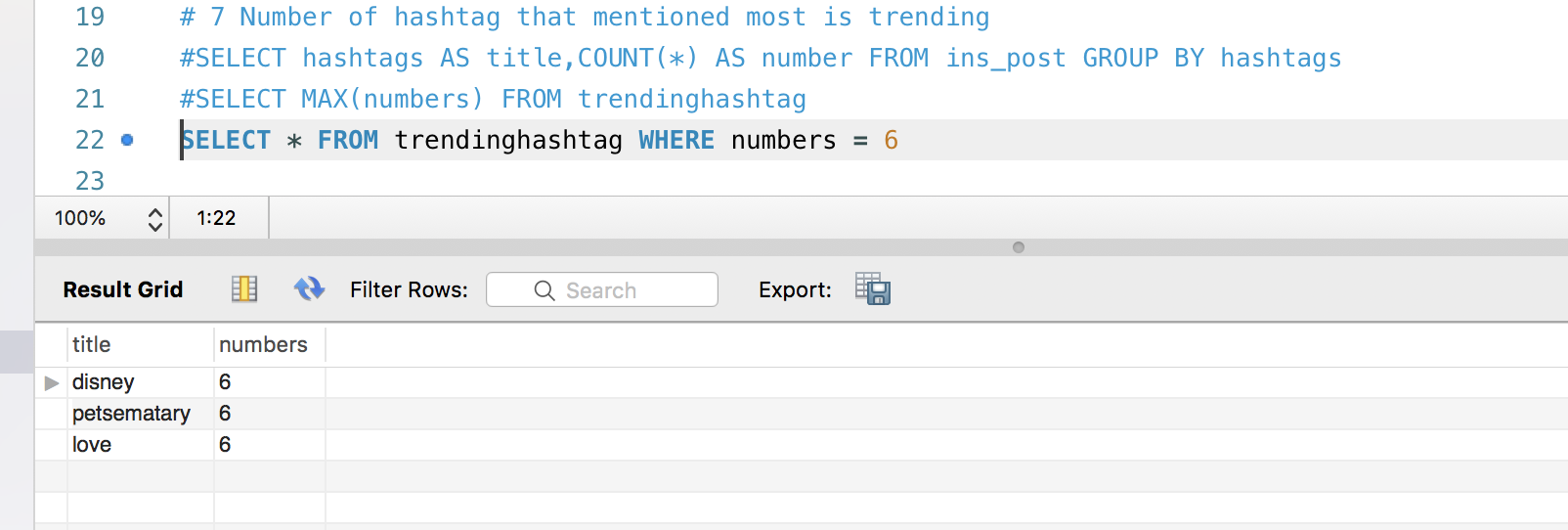


6.Who should I be following?

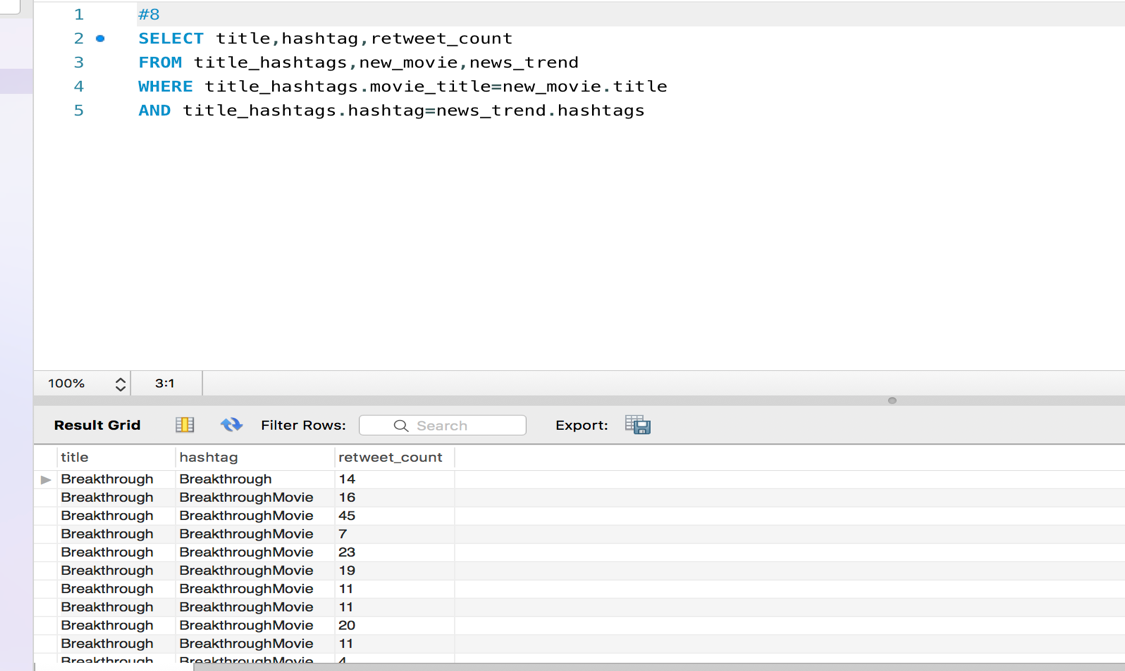


7.What topics are trending in my domain?

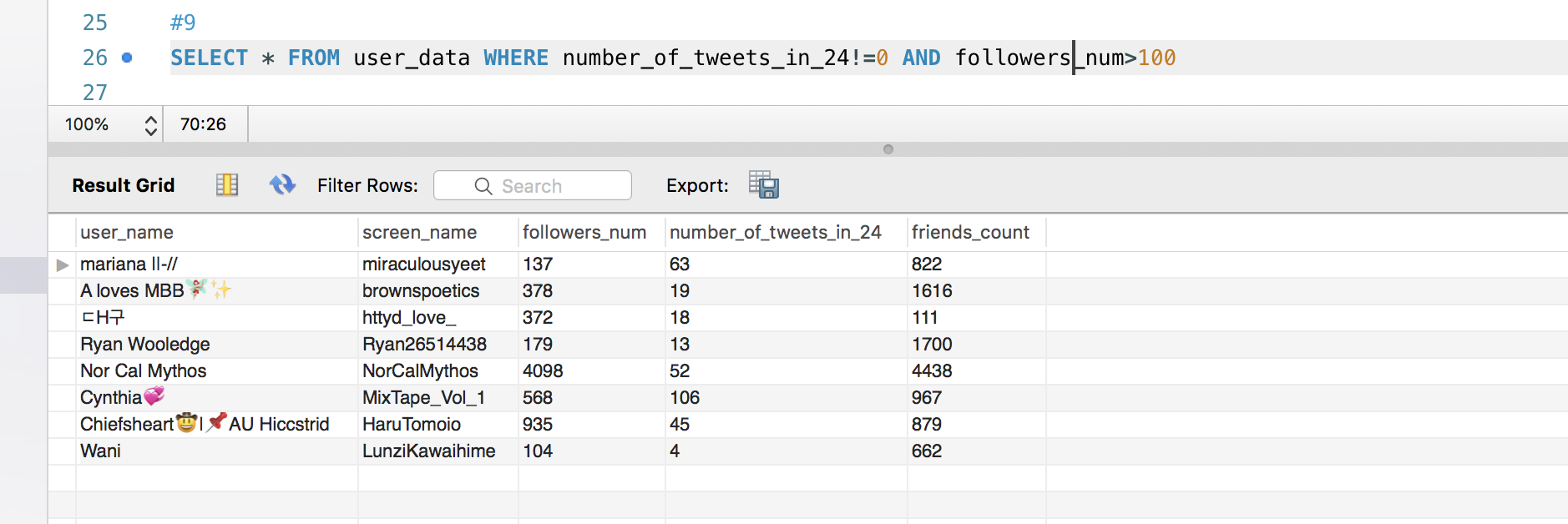




8.What keywords/hashtags should I add to my post?



9.Should I follow somebody back?



VII. Fuzzy search

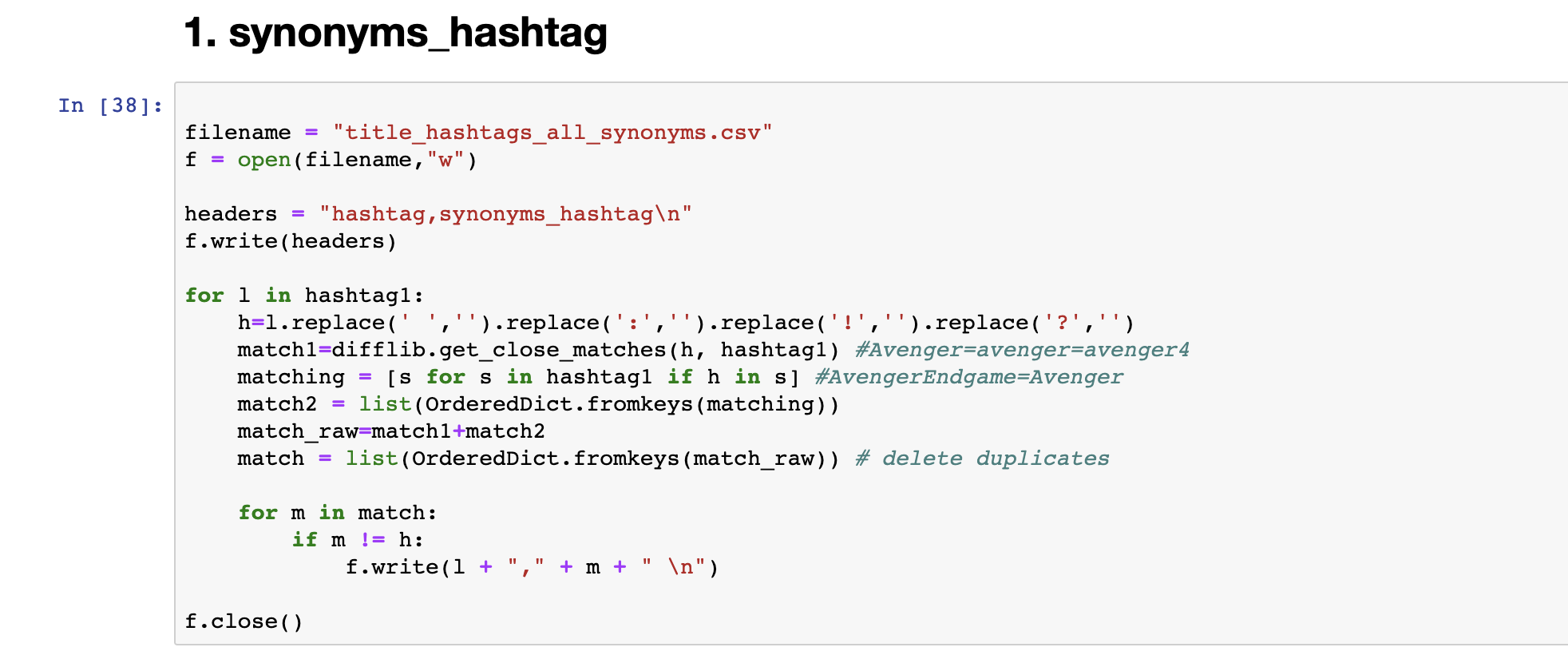
Basically two packages are used to build these tag tables, one is difflib, another one is fuzzywuzzy.

Difflib can find similar word with a given string from a given list;

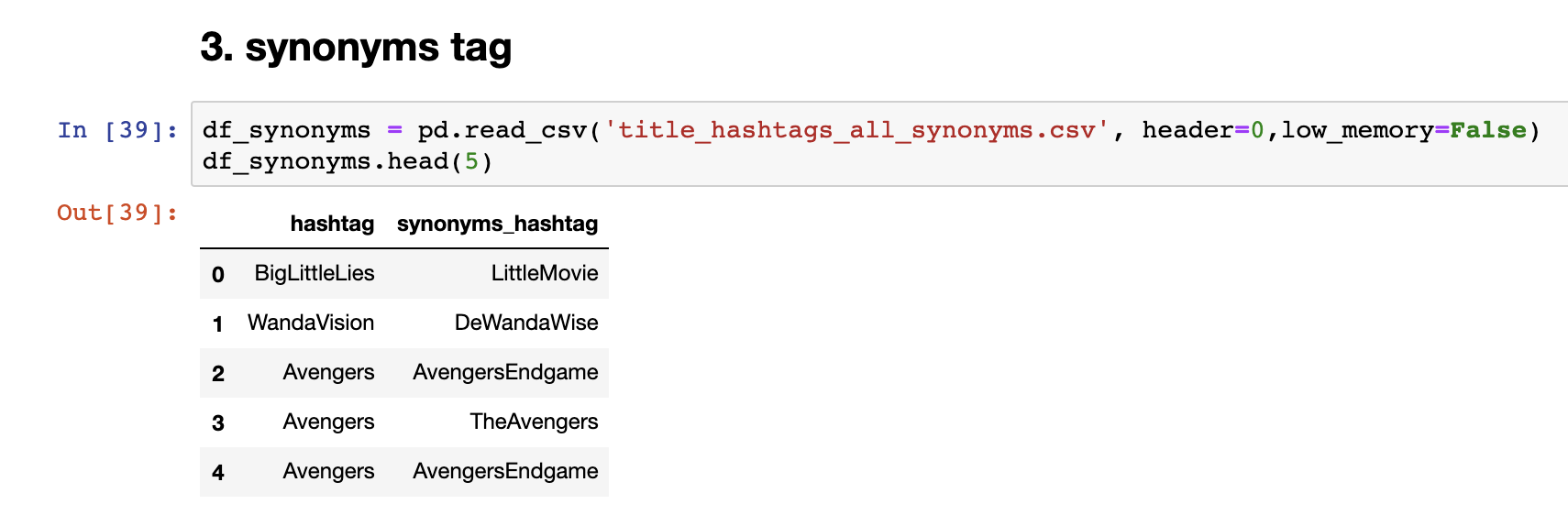
Fuzzywuzzy can value the similar rate of two strings

**1. Synonyms**

Code:



result (5 examples are shown)



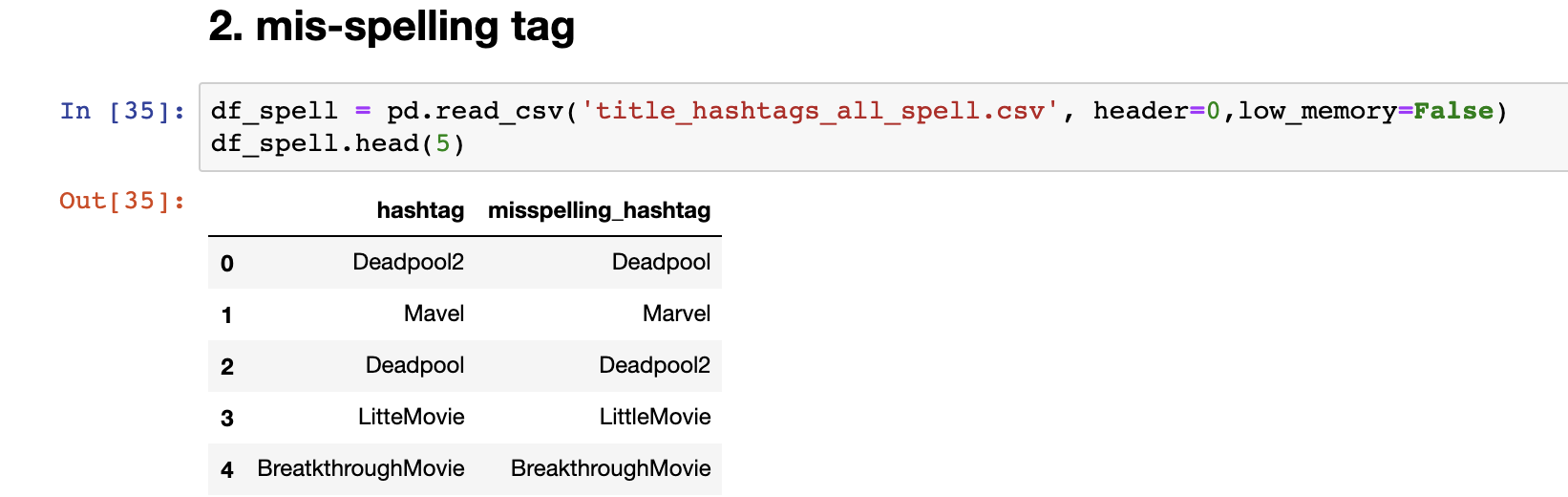
**2. Mis-spellings**

**Basically it’s finding word that similar rate >90%, the value 90% is after several trys I found it as a perfect value for check mis-spelling word**

**Code:**



Result (5 examples are shown)



**3. Semantic information**

First I have a list of movie titles, a list of TV series, a list of all hashtags.

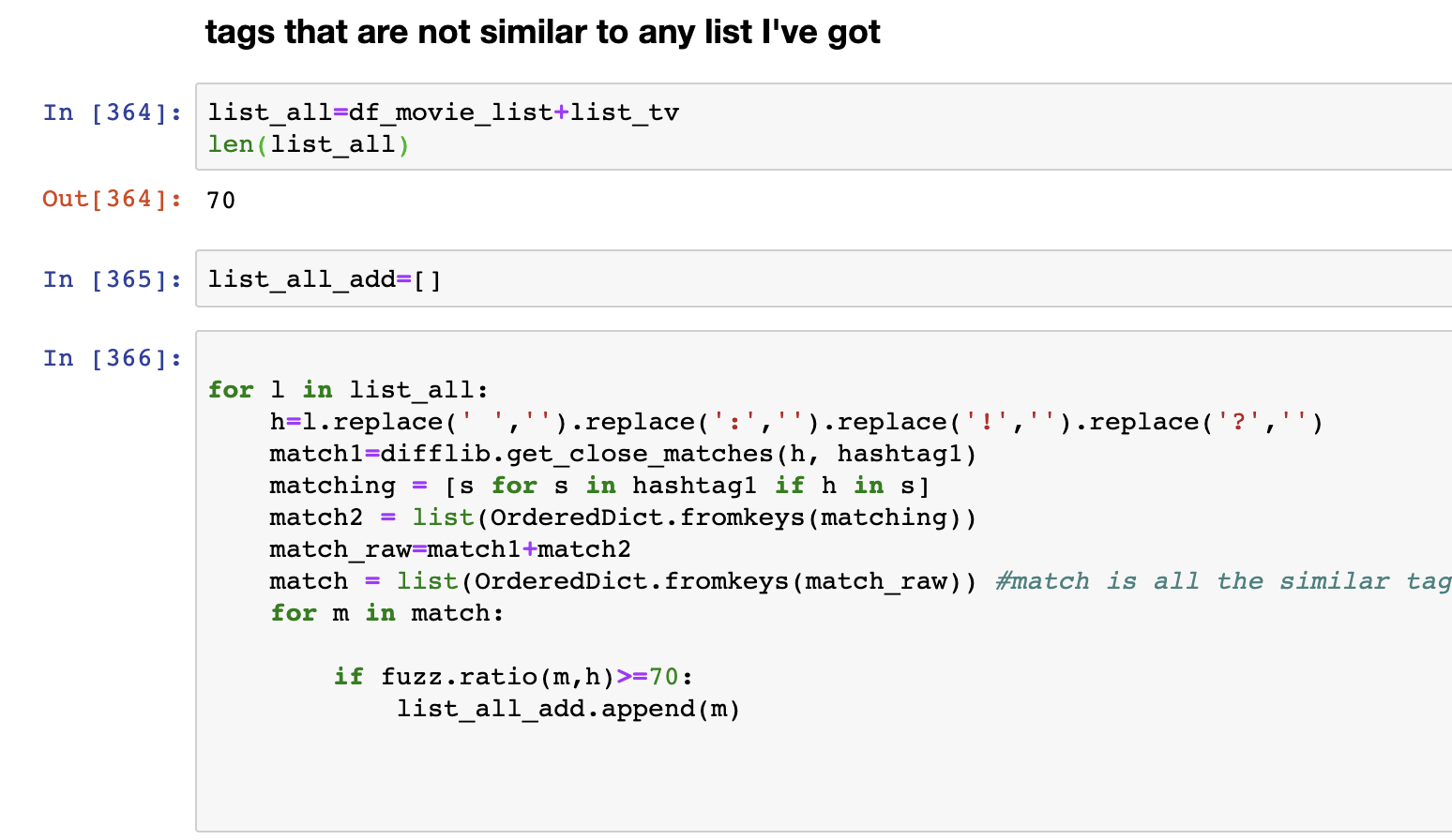
From all hashtags list, those who are 70% similar to movie titles are movies title tags;

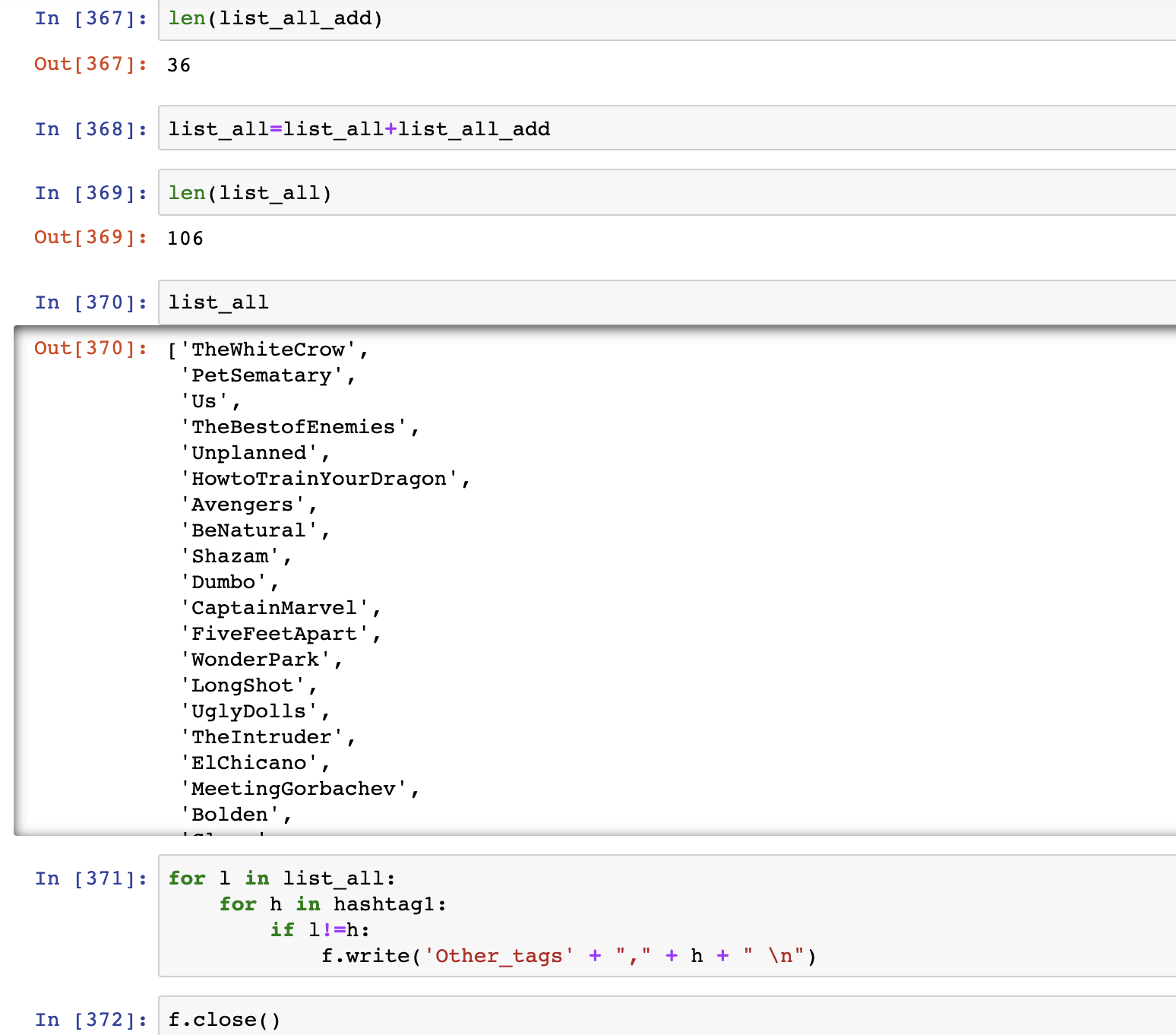
From all hashtags list, those who are 70% similar to TV series are TV series title tags;

The rest are in other category

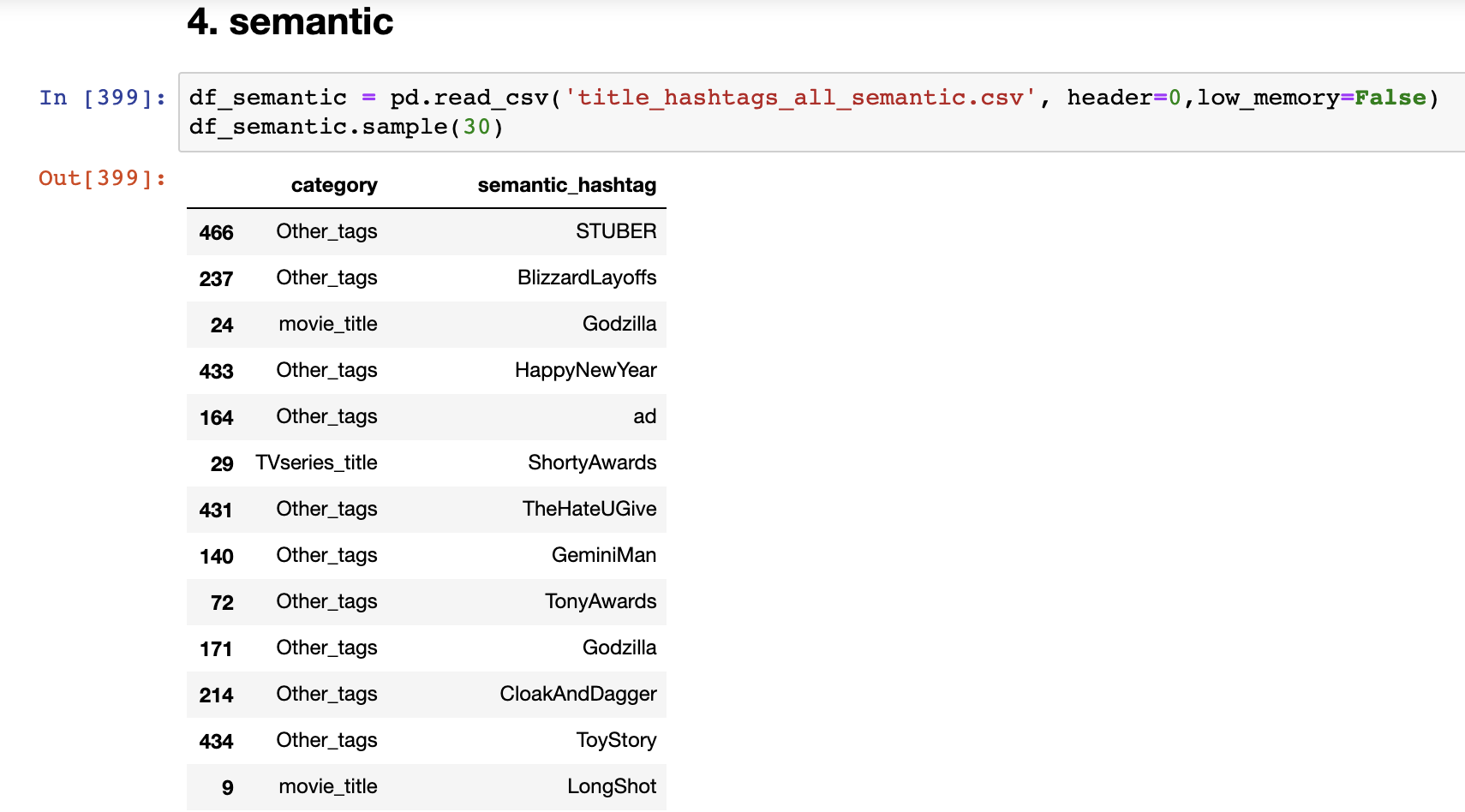








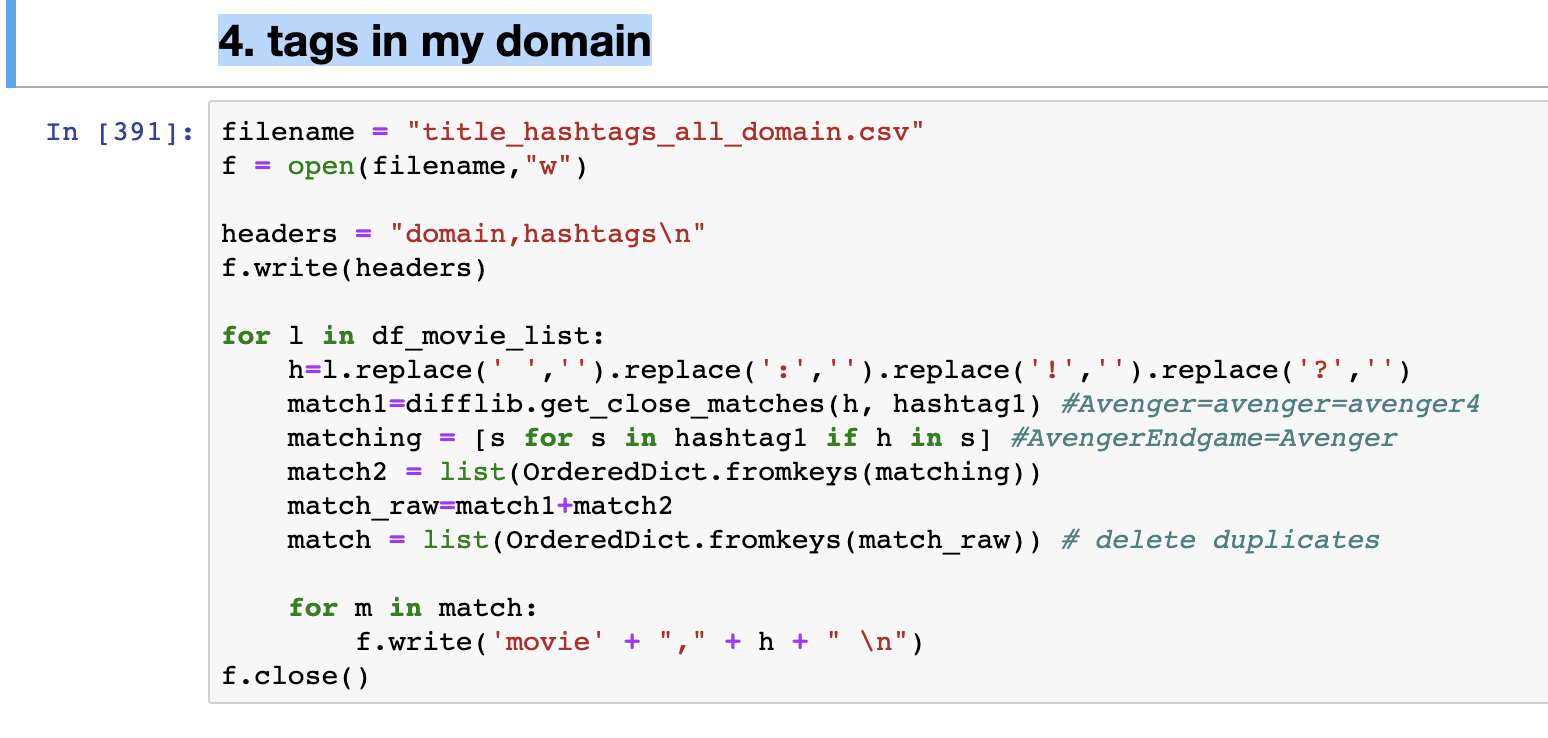
Result



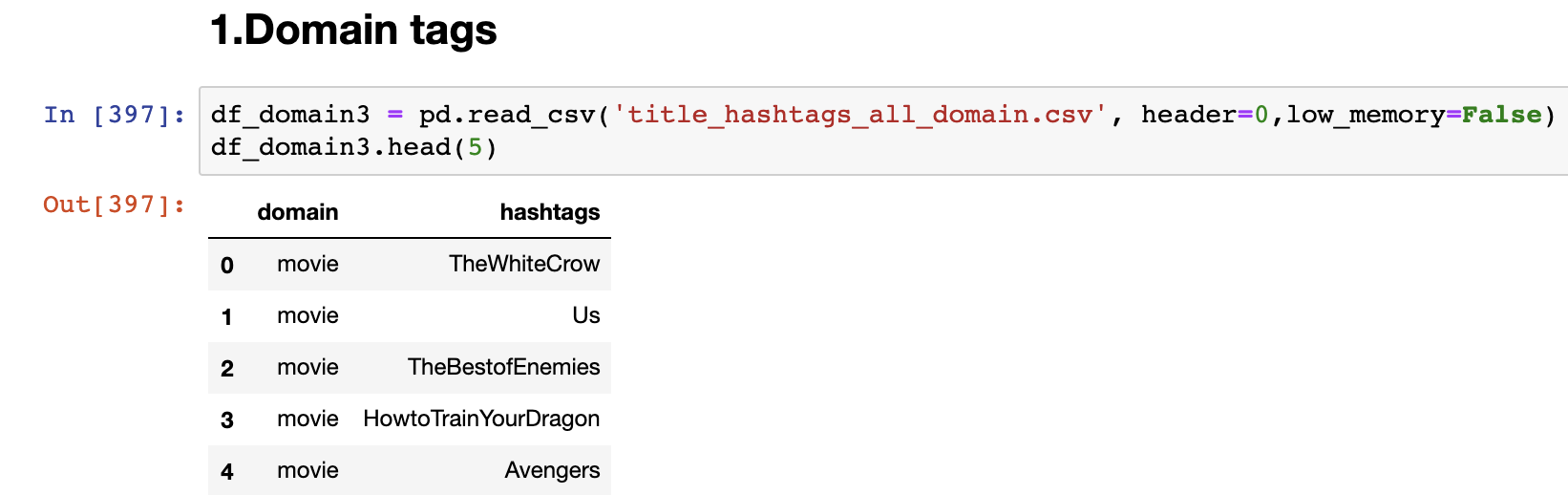
**4. tags in my domain**

All the tag in domain movie

Code



Result



VIII. Result

We create the database about latest movies and related old movies information. Then we write code in python to clean data, and then put into SQL to build connection of our database. We write functions to help users to find the movie they would like to watch.

IX. Conclusion and Difficulties

We create the database and use python to clean and then build connection of our tables and information. Then we write switch case to call database. Therefore, when user search information, we will call function to get the information they want.

However, the most difficult thing is that we don’t know how to create a website. And there are limits of using twitter API. So, from this project, we have clear aim of what to learn and do in the future: consolidate our knowledge, try to learn how to create the website.

X. Citation

1. Kaggle: <https://www.kaggle.com/beaubellamy/ski-resort>

2. Instagram API: <https://instaloader.github.io/index.html>

3.Twitter API: <https://developer.twitter.com/en/docs.html> 4.<http://www.omdbapi.com/>

5.<https://www.themoviedb.org/documentation/api>

6.<https://github.com/NIKBEARBROWN/INFO_6210>

|  |
| --- |
| MIT License |
|  |  |
|  | Copyright (c) 2019 YURU LI, ZHIHUI XIN |
|  |  |
|  | Permission is hereby granted, free of charge, to any person obtaining a copy |
|  | of this software and associated documentation files (the "Software"), to deal |
|  | in the Software without restriction, including without limitation the rights |
|  | to use, copy, modify, merge, publish, distribute, sublicense, and/or sell |
|  | copies of the Software, and to permit persons to whom the Software is |
|  | furnished to do so, subject to the following conditions: |
|  |  |
|  | The above copyright notice and this permission notice shall be included in all |
|  | copies or substantial portions of the Software. |
|  |  |
|  | THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR |
|  | IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, |
|  | FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE |
|  | AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER |
|  | LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, |
|  | OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE |
|  | SOFTWARE. |