Assignment #3: Movie Dictionary Database

HW3_toStudent.ipynb

1) Introduction

Assignment objectives:

This assignment aims to help students to practice their programming skills using “dictionary” in python.

Instructions

1. Save a copy in Drive (This is very important, otherwise, your work will be lost.)
2. Rename this notebook to HW3_xxxxxxxxxx.ipynb, where xxxxxxxxxx is your student ID.
    a. DO NOT submit a Colab link
3. Make sure that you do not peek at other people's code.
4. Make sure that nobody looks at your code.
5. If your submission is similar to others, you will get 0 points.
6. If your program cannot be run, it cannot be graded resulting in 0 points.

Submission

- File → Download → Download file.ipynb
- Submit the download file to MyCourseVille before 23.59 of Saturday 5th November 2022

2) Problem Statement: Movie Dictionary Database

Redis is a famous dictionary database that can store unstructured data, where each record (row) contains a different number of attributes. In this problem, we aim to imitate Redis by storing “movie data” in Python’s dictionary.

Figure 1 shows an example of 4 movies loaded from the file “import_movies_small.redis”. There are two files in the notebook.

- “import_movies.redis”: It is a full database with hundreds of movies.
- “Import_movies_small.redis”: It is a small sample database in Figure 1 containing only 4 movies.

There can be only 9 attributes for each movie as shown in Figure 2. Different movies can have different numbers of attributes. From Figure 1, HSET and movie_id are always in the beginning of each movie’s record. Then, it is followed by a pair of (key, value). Please note that the order doesn’t need to be as in Figure 2.

Function1: load_data_to_movie_dict(lines)

- Input: a list of lines that has been read from file
- Return: a dictionary of movie data, where key is movie_id

Write a function to load data into a movie dictionary (assume the variable “movies”) as the structure shown in Figure 3. Figure 4 shows an example of the movie_id = 1 (movies[1]).

If you load data correctly, your results should be similar to the expected output in Section 3 Outputs. The followings are hints based on Figure 1 & Figure 2.

- The movie_id must be parsed and changed to be integer, e.g., “movie:1” → 1.
- As in “movie:297” in Figure 1, there can be a special character (“\"”) in the text file; thus, you need to remove “\"” (replace with blank).
- There can be the attribute names (Figure 2) in the content (e.g., attributes “title” and “plot”). You may try to split by “"” (double quote) and then “ ” (space) before checking the attribute name.
- If a movie already existed in the movie dictionary, you should not load that movie into the dictionary again.

```
HSET "movie:1" title "Guardians of the Galaxy" genre "Action" votes 704613 rating
8.1 release_year 2014 plot "A group of intergalactic criminals must pull together to
stop a fanatical warrior with plans to purge the universe." poster
"https://m.media-amazon.com/images/M/MV5BMTAwMjU5OTgxNjZeQTJeQWpwZ15BbWU4MDUxNDYxODE
x._V1_SX300.jpg" ibmdb_id "tt2015381"
HSET "movie:2" title "Interstellar" genre "Adventure" votes 961763 rating 8.6
release_year 2014 plot "A team of explorers travel through a wormhole in space in an
attempt to ensure humanity's survival." poster
"https://m.media-amazon.com/images/M/MV5BZjdkOTU3MDktN2IxOS00OGEyLWFmMjktY2FiMmZkNWI
yODZiXkEyXkFqcGdeQXVyMTMxODk2OTU@._V1_SX300.jpg" ibmdb_id "tt0816692"
HSET "movie:7" title "X-Men: Days of Future Past" genre "Action" votes 524078 rating
8.0 release_year 2014
HSET "movie:297" title "Pod livnem pul" genre "Drama" votes 33 rating 6.6
release_year 2006  plot "Crimea Ukraine ca 1942. WWII. An elite squad of
\"razvedchiks\" - army scouts - is sent deep behind German lines on a series of
dangerous but vital reconnaissance missions for the Red Army." poster
"https://m.media-amazon.com/images/M/MV5BNDlkZmUwMGEtMTJmNC00ODlmLTk3NjYtMDc1MjViOWR
lM2YxXkEyXkFqcGdeQXVyNjExMjE5OTM@._V1_SX300.jpg" ibmdb_id "tt0902116"
```

Figure 1. Example of movie data from the file “import_movies_small.redis”. There are 4 movies, where blue highlight refers to movie_id and yellow highlights refer to attributes (as shown in Figure2). Please note that HSET must be ignored.

Attribute	Description
`movie_id`	the unique ID of the movie, internal to this database
`title`	the title of the movie
`plot`	a summary of the movie
`genre`	the genre of the movie (for now a movie will only have single genre)
`release_year`	the year the movie was released (as a numerical value)
`rating`	the average rating from the public (numerical value)
`votes`	the number of votes
`poster`	link to the movie poster
`imdb_id`	ID of the movie in the IMDB database

Figure 2. Possible attributes of each movie. Attributes “movie_id” must be int, “rating” and “votes” must be float, Attribute “release_year” must be int, and other attributes are string.

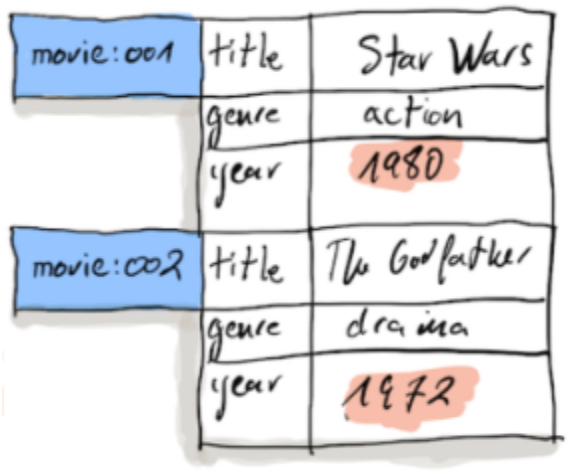

Figure 3. A structure of movie dictionary (movies), where movie_id is a key.

```
{'title': 'Guardians of the Galaxy',
 'genre': 'Action',
 'votes': 704613.0,
```

```
 'rating': 8.1,
 'release_year': 2014,
 'plot': 'A group of intergalactic criminals must pull together to stop
a fanatical warrior with plans to purge the universe.',
 'poster':
'https://m.media-amazon.com/images/M/MV5BMTAwMjU5OTgxNjZeQTJeQWpwZ15BbWU
4MDUxNDYxODEx._V1_SX300.jpg',
 'ibmdb_id': 'tt2015381'}
```

Figure 4. Example of the movie dictionary for the movie_id = 1 (movies[1])

Function2: summarize_movies_by_genre(movies):

- Input: a dictionary of movie data
- Return: a dictionary of genre & its movies, where key is a genre

Then, write a function to summarize movies by genre into a dictionary (assume the variable "movies_by_genre"), where key is genre and value is a list of movie_id's. Note that the list of movie_id's must be sorted ascendingly, e.g., 'movie:550', 'movie:955', 'movie:959'.

- Hint: Some movies may not have an attribute "genre", so you must choose only movies with this attribute ("genre").

```
'Sport': [550, 955, 959],
 'Romance': [553, 882],
 'History': [568, 878],
 'Musical': [600, 651],
```

Figure 5. Example of the movies_by_genre dictionary

Function3: calcualte_genre_stats(movies, movies_by_genre)

- Input: a dictionary of genre & its movies
- Return: a dictionary of genre's statistics, where key is a genre

Finally, write a function to calculate genre statistics (assume the variable "genre_stats"), where results are (1) the number of movies, (2) an average of rating, and (3) an average of votes. For an average, there are 2 decimal points. As shown in Figure 6, the result is a dictionary, where the key is genre containing 3 attributes ['num', 'rating', 'votes'].

```
'Action': {'num': 186, 'rating': 6.51, 'votes': 187336.3},
 'Adventure': {'num': 49, 'rating': 6.62, 'votes': 145701.29},
 'Crime': {'num': 49, 'rating': 6.55, 'votes': 135958.43},
 'Biography': {'num': 34, 'rating': 7.22, 'votes': 80150.53},
 'Horror': {'num': 33, 'rating': 5.93, 'votes': 41901.18},
```

Figure 6. Example of the genre_stats dictionary

3) Inputs & Outputs

Inputs

- Movie data as in Figure 1 from the file “import_movies_small.redis”. This data is already provided in the template file “HW3_toStudent.ipynb”

Functions

- Function1: load_data_to_movie_dict(lines)
- Function2: summarize_movies_by_genre(movies):
- Function3: calcualte_genre_stats(movies, movies_by_genre)

- You can have your own functions, but must still maintain all above functions (cannot modify them)!
- Function: print_ordered_dict(data, top='', details=True) is provided.
    - If details=True, it shows the detailed data in the dictionary.
    - If details=False, it shows only the number of attributes in the dictionary.

Outputs

- The dictionary “movies” contains movie data.
- The dictionary “movies_by_genre” contains a list of sorted movies by genre.
- The dictionary “genre_stats” contains statistics of each genre.
- **You can check parts of the results here [updated 28th Oct 2022].** There are 6 sheets as follows:
    - Sheet 0 shows an output of the small sample data (import_movies_small.redis).
    - Sheets 1-5 shows an output of the full database (import_movies.redis).