DSCI 551 – Fall 2021

Homework #1: Firebase, JSON, and Data Modeling

Deadline: 9/19, Sunday (100 points)

In this homework, consider managing data for a film rental app in Firebase realtime database. The data are stored in five tables in CSV format: *film*, *film_category*, *category*, *actor*, and *film_actor*. Note that *film_category* is a relationship table connecting film to category; and *film_actor* is also a relationship table connecting film and actor.

Suppose that the app needs to support two access patterns:

- Pattern 1: Given a film category (e.g., family), find titles, release_year, rating (e.g., PG-13), rental_rate, and rental_duration about all the films in that category. Order the films by title alphabetically.
- Pattern 2: Given an actor name (its full name, e.g., "ed chase"), find the title and release years of all films the actor has acted in. Order the result by title alphabetically.

Requirements:

- For each query in both patterns, only one round trip (send request and receive response) is permitted to the Firebase server.
- The response should not include irrelevant data, e.g., asking for family films, but the response from Firebase also contains non-family films, or asking for films by Ed Chase, but the responses contained films **not** by Ed Chase.
- You should NOT download the entire database and process the query locally.

Tasks:

1. [60 points] Write a Python script "load.py" that transforms and loads the data into Firebase. Your script may assume that all CSV files are located at the current directory (where you execute the script).
2. [20 points] Write a Python script "film.py" for access pattern 1.
   Execution format:
   ```
   python film.py <category>
   ```
   e.g., <category> may be family, comedy, sci-fi, etc. (case insensitive)

   Output format (print your results, one film per line as tuple; results are case insensitive):
   ```
   ('AMADEUS HOLY', '2006', 'PG', '0.99', '6')
   ('AMERICAN CIRCUS', '2006', 'R', '4.99', '3')
   …
   ```
   If no films in a given category, output:
   ```
   No results found.
   ```

3. [20 points] Write a Python script "actor.py" for pattern 2.
   Execution format:

```
python actor.py <actor full name>
```

e.g., <actor full name> may be "ed chase".

Output format (print your results, one film per line as tuple; results are case insensitive):

```
('ALONE TRIP', '2006')
('ARMY FLINTSTONES', '2006')
('ARTIST COLDBLOODED', '2006')
('BOONDOCK BALLROOM', '2006')
...
```

If the actor did not play in any films, output:

```
No results found.
```

Permitted libraries: pandas, requests, json, sys.

Submissions:

- Above 3 scripts.
- A document (word/pdf) explaining why your program sends only one request to Firebase per query; and if the response contains irrelevant data.
- A JSON dump of your Firebase database for this app.
- A screenshot of your Firebase, showing the structure of your database.
- Missing any files results in 30 pts penalty.
- Include all files mentioned above in a zipped folder and name it Firstname_Lastname_hw1.zip (e.g. Wensheng_Wu_hw1.zip). Do not create subfolders under the main folder.
- Submit online. See syllabus for late penalty!
- **PLEASE TEST YOUR SCRIPT ON EC2 AND MAKE SURE IT RUNS!**