EXERCISE 11

DATA MANAGEMENT

Group 11

Alexandre Ducommun
Rabeb Ben Ramdhane
Abdallah Shukor

Professor

Iulian Ciorascu

22 December 2019

Project goal

The goal of the project is an analysis of YouTube data from various countries. Data will be imported from the website: https://www.kaggle.com/datasnaek/youtube-new . The project scripts realized will import, download and load the data into a database.

Project structure

The project has the following structure:

db/	the database folder
sources/	folder to store the downloaded data into csv format
transformed files/	folder to store the transformed data into csv format
unautomated sources/	folder to store the data into json format
api_key.txt	file that contains the YouTube API key
country_codes.txt	file that contains the country codes
load.py	python script to push data into the database
scraper.py	python script to scrap data from YouTube
transform.py	python script to transform the data before database push

Data extraction

To retrieve the data, we used the python script **scraper.py** from the author of Kaggle project of the website. We made a few modifications to adapt the script to match with our project structure (location to download the sources). The script scraps the data using the YouTube API key and download the data into csv format in the folder sources. The filename of the data contains the date and the country codes.

Concerning the json files that contains the categories for each country, we downloaded it manually and put them into the folder **unautomated sources**. We don't know where to retrieve these files others than the Kaggle website. So, an automated task could connect to an account on Kaggle and scraps the website to retrieve the json files.

Data transformation

Our script **transform.py** use 3 functions to transform data. These functions prepare the data put the result in a csv file into the **fransformed files** folder.

The function prepare_trending_videos add the attributes country_codes, publish_time, publish_date and remove the publishedAt attribute. The result of this preparation is a concatenated csv file of all the files in **sources** folder with the name **trending_video.csv**.

The function prepare_videos_tags split the tags from the trending_videos.csv into a new csv file videos_tags.csv. For each tag that has been split, we store it in a data frame with its trending_videos_id. When we push the data frame into a csv file, we added an index with the label video_tags_id that will be used as primary key in the future database.

The function prepare_categories open the json files in a data frame and extract the id, title and etag for each category. Then these attributes are pushed as a row in a csv files named videos_categories.csv.

Loading data

We used an SQLite database to store our data about trending videos. The database is stored into the **db** folder. There are 3 functions to push data into the database.

The function push_trending_videos connect to the database, create a cursor and create a table **trending_videos** where the columns match the attributes of the trending_videos.csv. We created a data frame with the data of the file (trending_videos.csv) and used the function to_sql to push the content into the table. Then we commit and close the connection.

The function push_videos_tags connect to the database, create a cursor and create a table **videos_tags** where the columns match the attributes of the videos_tags.csv. Like the previous function, a data frame of the videos_tags.csv is pushed into the created table with the to_sql function.

The last function push_videos_categories establish the connection to the database and create the table **push_videos_categories** where columns match with attribute of videos_categories.csv. The data frame of the videos_categories.csv is pushed into the table with the to_sql function.