# Google Trends ETL

## Introduction

This project consists on an ETL (Extract, Transform, Load) process.  
- Extract: download data from Google Trends.  
- Transform: keep important columns, append data for each month/year.  
- Load: upload files to Dropbox in chunks of 5GB.

## Files and scripts

* main.py: Main script to import, process and load the data.
* config.py: Defines configuration object that contains all values configured in the config.properties file.
* dropbox\_handler.py: Contains the code to download/upload files to Dropbox. It uses the library dropbox.
* files\_manager.py: Contains the code to manage files and divide them into chunks.
* process.py: Includes the code for transforming the downloaded files into files of the certain size (threshold configured).
* gtrends.py: Includes the code for stabilising the connection with Google Trends. It is used the library pytrends.
* config.properties: Configuration file.
* requirements.txt: Requirements for running the environment.

## Clone repository

The code is available in github in the following url: https://github.com/mikaelapisani/google-trends. In order to clone the repository in your machine execute:

git clone https://github.com/mikaelapisani/google-trends.git

## Configure environment

### Python

Python 3 is requeried, running the script in a virtualenv with the necessaries dependencies. Alternative, it is provided a Dockerfile in order to run the script in a docker container.

### Docker

Build image:

docker build -t gtrends-etl .

It is necessary to specify 3 volumes, which are the directories for the temporary folders to save data:  
- data folder: will contain monthly and daily information. - result folder: In case that the upload to dropbox fails, the results files will be at this directory. - tmp folder: In this folder the temporary files are downloaded to be processed.

Each of those folder should contain 2 directories inside, 'daily' (for daily data) and 'monthly' (for monthly data).

### Virtualenv

Substitute {google-trends-path} by the path where the code is downloaded.

python3 -m venv {google-trends-path}  
source {google-trends-path}/bin/activate  
cd {google-trends-path}   
pip install --upgrade pip  
pip install -r requirements.txt

docker run -ti -v <data\_dir>:/root/data -v <results\_dir>:/root/results -v <tmp\_dir>:/root/tmp gtrends-etl bash

Once inside the container follow the execution manual to execute the program.  
The configuration path would be at '/root/google-trends/config.properties'

### TTU cluster

1. It is necessary to have installed conda. Follow these guide if it is not installed:  
    http://www.depts.ttu.edu/hpcc/userguides/application\_guides/python.local\_installation.php
2. Create virtualenv:  
   cd $HOME/google-trends

. $HOME/conda/etc/profile.d/conda.sh

conda activate

pip install --upgrade pip

pip install -r requirements.txt

1. Create directories for tickers\_folder, data\_folder\_monthly, data\_folder\_daily, result\_folder\_monthly, result\_folder\_daily, tmp\_folder\_monthly and tmp\_folder\_daily. Suggested paths:

mkdir $HOME/data/ # tickers\_folder  
mkdir $HOME/data/monthly/ # data\_folder\_monthly  
mkdir $HOME/data/daily/ # data\_folder\_daily  
mkdir -p $HOME/results/monthly/ # result\_folder\_monthly  
mkdir $HOME/results/daily/ # result\_folder\_daily  
mkdir -p $HOME/tmp/monthly/ # tmp\_folder\_monthly  
mkdir $HOME/tmp/daily/ # tmp\_folder\_daily

1. Edit config.properties file with the corresponding paths.
2. To run the import step, execute:  
    qsub $HOME/google-trends/mpi/mpi\_import.sh
3. To run the process step, execute:  
    qsub $HOME/google-trends/mpi/mpi\_process.sh

#### MPI

In order to run the jobs in the cluster it is used MPI.  
It can be found more information about how to run jobs in the cluster in the following link: http://www.depts.ttu.edu/hpcc/userguides/general\_guides/job\_submission.php  
In this case it is configured only 1 process as it has to be serial.  
In the future, the code can be modified in order to process in parallel the information and clean the data faster. Meanwhile it could be executed in different computers changing the configuration to import or process different categories.

#### Commands

- list jobs: qstat

- kill job: qdel <JOB\_ID>

- see output: tail -f <PROJECT\_NAME>.\*

- see information about a failed job: qacct -j <JOB\_ID>

The project name is in the file mpi/mpi.sh in the -N parameter: "MPI\_gtrends"

### Configuration

**Log Configuration**  
- log\_level: Log level (INFO, WARNING, ERROR)

**Dropbox Configuration**  
- *access\_token:* Access token generated by Dropbox to access/upload files.  
- *dropbox\_timeout:* Timeout for HTTP connection by the Dropbox API.  
- *dropbox\_chunck:* Chunk size to send data to Dropbox.  
- *tickers\_path:* Path in Dropbox where the tickers file is located.  
- *dropbox\_folder\_upload:* Path for upload result files.  
- *data\_folder\_monthly\_dropbox:* Path for raw monthly data.  
- *data\_folder\_daily\_dropbox:* Path for raw daily data.

**Local Configuration**  
- *tickers\_folder:* Local folder's path where ticker file is stored locally. Observe that if you want to download other tickers' list different from the one that is in Dropbox, you can edit this file locally with the list wanted. This can be convinient when you want a smaller list of tickers.  
- *data\_folder\_monthly:* Local folder's path for monthly data.  
- *data\_folder\_daily:* Local folder's path for daily data.  
- *result\_folder:* Local folder's path for results files.  
Files are saved temporary in this folder before uploading to Dropbox.  
- *result\_folder\_monthly:* Local folder's path for monthly results files.  
- *result\_folder\_daily:* Local folder's path for daily results files.  
- *tmp\_folder\_monthly:* Temporary folder's path to download raw monthly data from Dropbox to process.  
- *tmp\_folder\_daily:* Temporary folder's path to download raw daily data from Dropbox to process.

**Google Trends Configuration**  
- *encoding:* Encoding used for download the data.  
- *tz:* Timezone for download data.  
- *categories:* List of categories to download. With its' correspond type (monthly/daily).  
The format should be like: category\_number:category\_type,category\_number:category\_type.  
Example: 1283:monthly,107:monthly,107:daily,278:monthly,278:daily  
- *year\_from:* Year from which the data should be downloaded.  
- *year\_until:* Year until which the data should be downloaded.  
- *geo:* Indicates geographical area from the data. In the case to want all areas, this field should be empty.  
- *gtrends\_timeout\_connect:* Timeout for opening HTTP connection with Google Trends.  
- *gtrends\_timeout\_read:* Timeout for HTTP connection for reading the data to download.  
- *retries:* Indicates how many times should retry when the connection fails.  
- *backoff\_factor:* Seconds between attempts after the second retry.  
- *output\_size\_mb:* Threshold in MB for the files to be uploaded.  
- *prefix:* Prefix for the result files.

**Observations:**

* The folder paths should end with '/'.
* Notice that you can edit in categories, only one category that you want to import/process. Being able to process faster the information in several computers. You can also edit the tickers files, in order to assign chunks of tickers to different computers.

### Extraction step

For the extraction step, csv files are imported from Google Trends, saving them locally in two temporary directories. One directory for monthly files and another for daily files. After downloaded each file is uploaded to dropbox. The temporary files are removed from the local directory once uploaded to dropbox.

A pseudo-code is provided to understand how the files are downloaded.

for ticker in ticker:  
 for category in categories:  
 if (category\_type=='monthly'):  
 download\_monthly(year\_from, year\_to, ticker, category)  
 else:  
 download\_daily(year\_from, year\_to, ticker, category)

In summary, for each ticker,category and category type the data is downloaded for the configured frame composed by year\_from and year\_to and uploaded to dropbox.  
In dropbox there would be 2 folders: monthly and daily. Inside each folder, there would be a folder per category, and inside this last folder the downloaded files would be available.

**Observations:**  
1. *Data cleaning:* In order to clean the data, a transformation is used in this script, keeping only the columns of interest. In addition, the date format is transformed from 'Y-M-d' to 'Y-M' in the case of monthly files.

1. *Handle errors:*

* As Google trends has a limit for the amount of files that can be downloaded per day, which is not fixed. In case that the file could not be downloaded an error message would appear in the log and the excecution would be stopped. Next time that the script is executed, it would perform another attempt for this file.
* In order to check if the file should be downloaded or not, it is checked if the file is already in the dropbox folder, in that case it is not downloaded.
* In case that the upload to Dropbox fails, the files would be availabe in the data folder configured for monthly and daily. Those files should be uploaded to dropbox manually in the corresponding directory before continue with the process.
* The script output indicates if there are more files to download. In the case that all the files have been downloaded, the output would look as follows:

download\_all=True

Otherwise, it would be:

download\_all=False

#### Execution:

python main.py -c 'config.properties' --import=true &> output

The file output will contain the output from the execution. It can be seen using the following commands:

tail -f output

less output

### Transform step

For reasons of optimization, the transformation is done during the extraction and loading steps. In the extraction is used to clean the data and in the loading to append the content of the files into one.

### Load step

The loading step consist on reading the files for monthly and daily data and generate 2 different types of files. One that contains the information for all the months and other for all the days. The information is appended into one file until reaching the threshold configured in output\_size\_mb.  
Once the threshold is reached, the file is uploaded to Dropbox, and start to generate a new file. This process is repeated until all the files have been processed. This should be done for each category.

**Observations**  
1. *Transformation:* The appended information is sorted by date before uploading the file the information.  
2. *Handle errors:* In case that the upload to Dropbox fails, the result file would be saved in the results directory. Those files should be uploaded manually to dropbox.

#### Execution:

For executing the script it is necessary to indicate that it corresponds to the process step as well as indicate the category number.

python main.py -c 'config.properties' --process=true &> output

The file output will contain the output from the execution.

The file output will contain the output from the execution. It can be seen using the following command:

tail -f output

less output