CSC343H1 F 20209: INTRODUCTION TO DATABASES

YouTube Trending Video Database Analysis

Chuyi Hou: 1004197834
Yuchen Tong: 1003534669

October, 2020

Computer Science
UNIVERSITY OF TORONTO

Division of Engineering Science
UNIVERSITY OF TORONTO

Contents

1 Dataset and Relational Schema (Phase 1)

1.1 Domain

We want to study the trends of Youtube videos from different countries.

1.2 Dataset

The dataset can be downloaded manually via https://www.kaggle.com/datasnaek/youtube-new or using API: kaggle datasets download -d datasnaek/youtube-new

The raw dataset consists of ten json files for category ids and ten csv files of YouTube trending videos from ten different countries. Before implementing the schema, we should create country IDs and convert publish date so that it can be compared with trending date, split and count numbers of tags for each video, and find category names for each category ID.

We will be using the following columns of the csv files:

video_id, trending_date, title, channel_title, category_id, publish_time, tags, views, likes, dislikes

Last but not the least, we should get a sense of what those tags stand for and what categories of contents they represent.

1.3 Investigative Questions

- Find top n^* videos that are simultaneously trending in different countries given a specific time interval (Report: vID,vTitle,countryName,categName,tags,time interval).
- Given a specific time interval, find the trending categories that at least n^* countries have in common. (Report: categName, Time interval)
- Find the all time top n^* tags for each country. (Report: tag, countryName)

$*$: Means variable.

1.4 Schema

Country(countryID, countryName)
Video(vID, vTitle, categID, countryID)
Category(categID, categName)
Channel(channelTitle, vID)
Publish(vID, pubTime, tags, categID, channelTitle)
Trending(trendingDate, vID)
Popularity(vID, views, likes, dislikes)

Publish[vID] $\subseteq$ Video[vID]
Channel[vID] $\subseteq$ Video[vID]
Popularity[vID] $\subseteq$ Publish[vID]
Video[countryID] $\subseteq$ Country[countryID]
Publish[categID] $\subseteq$ Category[categID]
Publish[channelTitle, vID] $\subseteq$ Channel[channelTitile, vID]

2 Schema Implementation (Phase 2)

2.1 SQL Schema

```
drop schema if exists TrendingYouTube cascade;
create schema TrendingYouTube;
set search_path to TrendingYouTube;

create domain ctryID as smallint
    check (value >= 1 and value <= 10);

create domain hits as bigint
    check (value >= 0);

create domain ctryName as text
    check (value in ('Canada', 'Germany', 'France',
        'United Kingdom', 'India', 'Japan', 'Korea',
        'Mexico', 'Russia', 'United States'));

create domain catID as smallint
    check (value >= 1 and value <= 30);

create table country(
    countryID ctryID primary key not NULL,
    countryName ctryName unique not NULL
);

create table category(
    categID catID primary key not NULL,
    categName text unique not NULL
);

create table video(
    vID character(11) primary key not NULL,
    vTitle text not NULL,
    categID catID not NULL,
    countryID ctryID not NULL,
    foreign key (countryID) references country

);

create table channel(
    channelTitle text not NULL,
    vID character(11) not NULL,
    primary key (channelTitle, vID),
    foreign key (vID) references video
);

create table publish(
    vID character(11) primary key not NULL,
    pubTime date not NULL,
    tags text not NULL,
    categID catID not NULL,
    channelTitle text not NULL,
```

```
    foreign key (categID) references category,
    foreign key (channelTitle, vID) references channel
);

create table trending(
    trendingDate date not NULL,
    vID character(11) references video not NULL,
    primary key (trendingDate, vID)
);

create table popularity(
    vID character(11) primary key not NULL,
    view hits not NULL,
    likes hits not NULL,
    dislikes hits not NULL,
    foreign key (vID) references publish
);
```

2.2 Import Schema Demo on SQL

```
csc343h-houchuyi=> \i schema.ddl
DROP SCHEMA
CREATE SCHEMA
SET
CREATE DOMAIN
CREATE DOMAIN
CREATE DOMAIN
CREATE DOMAIN
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
CREATE TABLE
csc343h-houchuyi=> \d
                List of relations
     Schema      |    Name    | Type  |  Owner
-----------------+------------+-------+----------
 trendingyoutube | category   | table | houchuyi
 trendingyoutube | channel    | table | houchuyi
 trendingyoutube | country    | table | houchuyi
 trendingyoutube | popularity | table | houchuyi
 trendingyoutube | publish    | table | houchuyi
 trendingyoutube | trending   | table | houchuyi
 trendingyoutube | video      | table | houchuyi
(7 rows)

csc343h-houchuyi=> \d category
          Table "trendingyoutube.category"
  Column   | Type  | Collation | Nullable | Default
-----------+-------+-----------+----------+---------
 categid   | catid |           | not null |
 categname | text  |           | not null |
```

```
Indexes:
    "category_pkey" PRIMARY KEY, btree (categid)
    "category_categname_key" UNIQUE CONSTRAINT, btree (categname)
Referenced by:
    TABLE "publish" CONSTRAINT "publish_categid_fkey" FOREIGN KEY
    (categid) REFERENCES category(categid)

csc343h-houchuyi=> \d channel
                Table "trendingyoutube.channel"
    Column    |     Type      | Collation | Nullable | Default
--------------+---------------+-----------+----------+---------
 channeltitle | text          |           | not null |
 vid          | character(11) |           | not null |
Indexes:
    "channel_pkey" PRIMARY KEY, btree (channeltitle, vid)
Foreign-key constraints:
    "channel_vid_fkey" FOREIGN KEY (vid) REFERENCES video(vid)
Referenced by:
    TABLE "publish" CONSTRAINT "publish_channeltitle_fkey" FOREIGN KEY
    (channeltitle, vid) REFERENCES channel(channeltitle, vid)

csc343h-houchuyi=> \d country
             Table "trendingyoutube.country"
   Column    |   Type   | Collation | Nullable | Default
-------------+----------+-----------+----------+---------
 countryid   | ctryid   |           | not null |
 countryname | ctryname |           | not null |
Indexes:
    "country_pkey" PRIMARY KEY, btree (countryid)
    "country_countryname_key" UNIQUE CONSTRAINT, btree (countryname)
Referenced by:
    TABLE "video" CONSTRAINT "video_countryid_fkey" FOREIGN KEY (countryid)
    REFERENCES country(countryid)

csc343h-houchuyi=> \d popularity
            Table "trendingyoutube.popularity"
  Column  |     Type      | Collation | Nullable | Default
----------+---------------+-----------+----------+---------
 vid      | character(11) |           | not null |
 view     | hits          |           | not null |
 likes    | hits          |           | not null |
 dislikes | hits          |           | not null |
Indexes:
    "popularity_pkey" PRIMARY KEY, btree (vid)
Foreign-key constraints:
    "popularity_vid_fkey" FOREIGN KEY (vid) REFERENCES publish(vid)

csc343h-houchuyi=> \d publish
                Table "trendingyoutube.publish"
    Column    |     Type      | Collation | Nullable | Default
--------------+---------------+-----------+----------+---------
 vid          | character(11) |           | not null |
 pubtime      | date          |           | not null |
 tags         | text          |           | not null |
```

```
 categid      | catid         |           | not null |
 channeltitle | text          |           | not null |
Indexes:
    "publish_pkey" PRIMARY KEY, btree (vid)
Foreign-key constraints:
    "publish_categid_fkey" FOREIGN KEY (categid) REFERENCES category(categid)
    "publish_channeltitle_fkey" FOREIGN KEY (channeltitle, vid) REFERENCES
    channel(channeltitle, vid)
Referenced by:
    TABLE "popularity" CONSTRAINT "popularity_vid_fkey" FOREIGN KEY (vid)
    REFERENCES publish(vid)

csc343h-houchuyi=> \d trending
               Table "trendingyoutube.trending"
    Column    |     Type      | Collation | Nullable | Default
--------------+---------------+-----------+----------+---------
 trendingdate | date          |           | not null |
 vid          | character(11) |           | not null |
Indexes:
    "trending_pkey" PRIMARY KEY, btree (trendingdate, vid)
Foreign-key constraints:
    "trending_vid_fkey" FOREIGN KEY (vid) REFERENCES video(vid)

csc343h-houchuyi=> \d video
               Table "trendingyoutube.video"
  Column   |     Type      | Collation | Nullable | Default
-----------+---------------+-----------+----------+---------
 vid       | character(11) |           | not null |
 vtitle    | text          |           | not null |
 categid   | catid         |           | not null |
 countryid | ctryid        |           | not null |
Indexes:
    "video_pkey" PRIMARY KEY, btree (vid)
Foreign-key constraints:
    "video_countryid_fkey" FOREIGN KEY (countryid) REFERENCES country(countryid)
Referenced by:
    TABLE "channel" CONSTRAINT "channel_vid_fkey" FOREIGN KEY (vid)
    REFERENCES video(vid)
    TABLE "trending" CONSTRAINT "trending_vid_fkey" FOREIGN KEY (vid)
    REFERENCES video(vid)
```

3 Data Cleaning and Import (Phase 3)

3.1 Decisions

Before we execute the data cleaning and import process, we decided to make some modifications to our schema and choose a narrowed selection of the YouTube trending dataset. For the changes made in schema, since countryID for videos is representing that video being trending in that country, we changed:

$$Video(\underline{vID}, vTitle, categID, countryID) \rightarrow Video(\underline{vID}, vTitle, categID)$$

and

$$Trending(\underline{trendingDate, vID}) \rightarrow Trending(\underline{trendingDate, vID, countryID})$$

Moreover, we needed to change the foreign key accordingly:

$$Video[countryID] \subseteq Country[countryID] \rightarrow Trending[countryID] \subseteq Country[countryID]$$

For the new dataset, it only included three countries instead of 10 as initially proposed. They are Canada, the United States, and the United Kingdom. Notice that they are all English-speaking countries, hence we can access the dataset in a more understandable manner in terms of the output result that we are going to write queries on.

Next, in the process of data cleanning and import, we created two pieces of python code.
create_Category_from_json.py is responsible for extracting category ids and their corresponding names from json files, and *data_cleaning.py* is to extract relevant data from each country's csv file. During data cleaning, we encountered an issue with load some countries' csv files. This might due to the csv decoder being unable to process some foreign languages' characters. This issue was avoided since we now only considered three English-speaking countries.

3.2 Cleaning Steps

- For *create_Category_from_json.py*
  - Read all json files which contains category ID and category name mapping for different countries.
  - Create a overall key(Category ID) and value(Category name) mapping by adding each item from previous mapping to a python dictionary.
  - If there are exist same category name mapped by different category ID in different country, suffix a number(how many times the name is occuring) to avoid duplicates.
- For *data_cleaning.py*
  - We have 3 csv files containing YouTube trending video data for 3 countries respectively.
  - First, we create the Country.csv by assigning 1 to 3 to these 3 countries respectively, and then add a countryID column to the country's csv file (i.e. for Canada, its csv file will be added a countryID column with all ones).
  - Then, combine all countries's csv by stacking one on up of another and select (project) relevant columns. (notice that there could be cases where one video can be trending in multiple countries, hence duplicates might occur in the combined csv).
  - Finally, load the combined csv (3 countries data are in side with relevant columns) and extract columns for SQL tables accordingly. Drop duplicates based on keys that were defined in the schema before write to files.
  - Output cleaned csv files: {Video.csv, Channel.csv, Publish.csv, Trending.csv, Popularity.csv}
- Country.csv, with columns being countryID (1 to 3) and countryName,
- All cleaned files are: {Country.csv, Category.csv, Video.csv, Channel.csv, Publish.csv, Trending.csv, Popularity.csv}

3.3 create_Category_from_json.py

```
import json
import pandas as pd

filenames = ['CA_category_id','DE_category_id','FR_category_id','
    ↪ GB_category_id', 'IN_category_id','JP_category_id','KR_category_id',
    ↪  'US_category_id']

for f in filenames:

    with open('./dataset/'+f+'.json') as file:

        data_dict = json.load(file)

    category_id = {}

    for i in range(0, len(data_dict['items'])):
        id = data_dict['items'][i]['id']
        category = data_dict['items'][i]['snippet']['title']

        category_id[id] = category

df = pd.DataFrame(category_id.items(),columns=['categID','categName'])

df.to_csv('Category.csv',index=False)
```

3.4 data_cleaning.py

```
import pandas as pd

# Country.csv and Category.csv are made manually

all_filenames = ['CAvideos','DEvideos','FRvideos','GBvideos','INvideos','JPvideos','KRvideos','USvideos']
# 'MXvideos' 'RUvideos'
# first add one additional column 'countryID' to each csv files
countryID = 1
print('Start adding countryID column to each countries csv files')
for file in all_filenames:
    df = pd.read_csv('./dataset/'+file+'.csv',
                     usecols = ['video_id','trending_date','title','channel_title','category_id','publish_time','tags','views','likes','dislikes'])
    df["countryID"] = countryID
    df.to_csv('./dataset/' + file + 'ID.csv', index=False)
    countryID += 1
print('Adding succesfully')

print('Start Combining all countries csv files')
# next, we combine all csv files
combined_csv = pd.concat([pd.read_csv('./dataset/' + f + 'ID.csv') for f in all_filenames])
#export to csv
combined_csv.to_csv('./dataset/combined.csv', index=False)
print('Successfully Combined')


# then project desired columns and write
# define columns for each csv files
data = {'Video':['video_id','title','category_id','countryID'],
        'Channel':['channel_title','video_id'],
        'Publish':['video_id','publish_time','tags','category_id','channel_title'],
        'Trending':['trending_date','video_id'],
        'Popularity':['video_id','views','likes','dislikes']
        }

cols = [[   vID    , 'vTitle', 'categID', 'countryID'],
        ['channelTitle', 'vID'],
        ['vID', 'pubTime', 'tags', 'categID',   'channelTitle'],
        ['trendingDate', 'vID'],
        ['vID', 'views', 'likes', 'dislikes']]

keys = ['video_id', ['channel_title','video_id'],'video_id',['trending_date','video_id'],'video_id']
i = 0
print('Start Extracting Columns From the Combined csv to Each Tables Needed for SQL Import')
for table in data:
    df = pd.read_csv('./dataset/combined.csv')
```

```
    df = df[data[table]]

    # sorting by key
    df.sort_values(keys[i], inplace = True)

    # dropping ALL duplicte rows
    df.drop_duplicates(subset = keys[i],keep = False, inplace = True)

    #modify date datastyle for Trending table
    if table == 'Trending':
        dates = df.trending_date

        new_col = []
        for date in dates:
            ymd = date.split('.')
            new_col.append('20'+ymd[0]+'-'+ymd[1]+'-'+ymd[2])

        new = pd.DataFrame(new_col,columns=['trendingDate'])

        df.drop(columns=['trending_date'])

    # changing columns using .columns()
    df.columns = cols[i]

    df.to_csv(table+'.csv',index=False)
    i+=1
print('All Tables are Successfully Created')
```

3.5 Data Import Demo on SQL

4 Queries and Results (Phase 4)

4.1 Investigative Questions Revision

- Find top n^* videos that are simultaneously trending in different countries given a specific time interval (Report: vID,vTitle,categName,tags,time_interval,number_of_trendings).
- Find the monthly (from 2017-11 to 2018-06) top trending category that at least n^* countries have in common. (report: year, month, categName, number_of_trendings)
- Find channels that are trending in every month within a time interval (e.g. 2017-11 to 2018-03), and whose monthly total view is non-decreasing. (report: channelname)

$*$: Means variable.

4.2 What we have learned