IMDB INSIGHTS

USING SQL AND PYTHON

Author: Diego Hernández Jiménez

Index

Summary

Using publicly available datasets from the Internet Movie Database (IMDb) I’ve created my own database.

I’ve connected the database from a Google Colab envirnoment and I’ve written some queries to extract interesting data.

I’ve supported the exploratory analysis with plots generated with the Python library seaborn.

Software

- Ubuntu 22.04.1 LTS (GNU/Linux 5.10.16.3-microsoft-standard-WSL2 x86_64)
- SQLite 3.37.2
- Ipython-sql 0.3.9 (Google Colab)
- Python 3.7.13 (Google Colab)
- Pandas 1.3.5 (Google Colab)
- Seaborn 0.11.2 (Google Colab)
- Matplotlib 3.2.2 (Google Colab)

Procedure:

Gathering data

```
mkdir imdb_data # create folder for datasets

cd imdb_data # go to folder

wget
https://datasets.imdbws.com/title.ratings.tsv.gz
# download specific dataset

gzip --decompress title.ratings.tsv.gz # extract
the data
```

Procedure:

Creating the database and importing the tables

```
sqlite3 imdb.db # create database and open sqlite

sqlite> CREATE TABLE ratings(
  tconst TEXT,
  averageRating NUMERIC,
  numVotes INT
); # create table schema

sqlite> CREATE INDEX movieID_rat ON
ratings(tconst); # indices allow faster queries

sqlite> .mode tabs  # to allow tab separated
files

sqlite> .import title.ratings.tsv ratings # fill
table with data from tsv file. This step can be
done without previously manually creating the
table (but it Will infere data types)
```

Procedure:

Preparing Google Colab environment

```
%pip install ipython-sql


# to be able to access drive files
from google.colab import drive
drive.mount('/content/drive’)


%load_ext sql
%sql sqlite:////content/drive/MyDrive/Colab_Not
ebooks/sql_imdb/imdb.db
# location of database file


import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

Procedure:

Executing queries and plotting

(more on jupyter file)

```
%%sql

SELECT
  ROUND(AVG(averageRating),3) AS avg_rating,
  MIN(averageRating) AS min_rating,
  MAX(averageRating) AS max_rating,
  SUBSTR(CAST(startYear as TEXT),3,1) || '0s' AS
decade
FROM basics
  INNER JOIN ratings ON basics.tconst == ratings.
tconst
WHERE startYear >= 1900 AND startYear <= 1999
AND startYear != '\N' AND titleType == 'movie'
GROUP BY decade
ORDER BY decade ASC;
```

Procedure:

Executing queries and plotting

avg_rating	min_rating	max_rating	decade
4.241	2.5	7.4	00s
5.886	1.0	9.2	10s
6.095	1.0	9.3	20s
6.078	1.1	9.4	30s
6.174	1.0	9.5	40s
6.269	1.2	9.5	50s
6.17	1.0	9.4	60s
5.976	1.1	9.8	70s
5.996	1.0	9.8	80s
6.0	1.0	9.8	90s

Learnings

- Use SQLite from command line interface
- Integrate SQL + Python in Colab

future directions

- Design more complex (and efficient) queries
- Use other DBMS software like PostgreSQL