LAB3 REPORT

Shaoze Yang

Initialize database

```
create table name
(
nconst varchar(10) not null,
primaryName text not null,
birthYear varchar(4) not null,
deathYear varchar(4) not null,
primaryProfession text not null,
knownForTitles text not null,
primary key(nconst)
);
```

```
create table akas
(
titleId text not null,
ordering INT not null,
title text not null,
region varchar(3) not null,
language varchar(3) not null,
types text not null,
attributes text not null,
isOriginalTitle BOOL not null
);
```

```
create table basics
(
tconst varchar(10) not null,
titleType text not null,
primaryTitle text not null,
originalTitle text not null,
isAdult BOOL not null,
startYear varchar(4) not null,
endYear varchar(4) not null,
runtimeMinutes INT not null,
genres text not null,
primary key(tconst)
);
```

```
create table crew
(
tconst varchar(10) not null,
directors text not null,
writers text not null,
primary key(tconst)
);
```

```
create table episode
(
    tconst varchar(10) not null,
    parentTconst varchar(10) not null,
    seasonNumber INT not null,
    episodeNumber INT not null,
    primary key(tconst)
);
```

```
create table principals
(
    tconst varchar(10) not null,
    ordering INT not null,
    nconst varchar(10) not null,
    category text not null,
    job text not null,
    characters text not null
);
```

```
create table ratings
(
    tconst varchar(10) not null,
    averageRating FLOAT not null,
    numVotes INT not null,
    primary key(tconst)
);
```

e

Verifying the Data

- The oldest movie

```
SELECT *
FROM basics
WHERE startYear = (SELECT MIN(startYear) FROM basics);
```

```
tt3155794 short Passage de Venus  Passage de Venus  0 1874  \N  1 Documentary,Short
```

- The the longest movie in 2009

```
SELECT *
FROM basics
WHERE
titleType = "movie"
AND runtimeMinutes <> "\N" AND startYear = 2009
AND runtimeMinutes = (SELECT MAX(runtimeMinutes) FROM basics WHERE titleType =
"movie" AND runtimeMinutes <> "\N" AND startYear = 2009);
```

```
tt1458948|movie|Native of Owhyhee|Native of
Owhyhee|0|2009|\N|390|Documentary,History
```

- The year with the most movies

```
SELECT startYear
FROM basics
WHERE titleType = 'movie' AND startYear <> '\N'
GROUP BY startYear
HAVING COUNT(startYear) = (
    SELECT MAX(count_value)
    FROM (
        SELECT startYear, COUNT(startYear) AS count_value
        FROM basics
        WHERE titleType = 'movie' AND startYear <> '\N'
        GROUP BY startYear
    ) AS subquery
);
```

```
2019
```

- The name of the person who contains in the most movies

```
SELECT primaryName FROM name
WHERE nconst =
(SELECT nconst
FROM
( SELECT *
  FROM principals RIGHT JOIN
  basics ON principals.tconst = basics.tconst
  WHERE basics.titleType = 'movie'
)
GROUP BY nconst
ORDER BY COUNT(nconst) DESC LIMIT 1);
```

```
Ilaiyaraaja
```

- The principal crew of the movie with highest average ratings and more than 500 votes

```
SELECT * FROM crew
WHERE tconst =
(
SELECT tconst FROM
    (
        SELECT ratings.* FROM
        ratings RIGHT JOIN
        basics ON ratings.tconst = basics.tconst
        WHERE basics.titleType = 'movie'
    )
WHERE
numVotes > 500
ORDER BY averageRating DESC LIMIT 1
);
```

```
tt14039792|nm11823961|\N
```

```
SELECT primaryName
FROM name WHERE
nconst = 'nm11823961';
```

```
Narra Sivanageswararao
```

- (Advanced) The count of each Pair<BirthYear, DeathYear> of the people

```
SELECT birthYear, deathYear, COUNT(*) AS row_count
FROM (
  SELECT * FROM
  name WHERE
  birthYear <> '\N' AND deathYear <> '\N'
)
GROUP BY birthYear, deathYear
ORDER BY row_count DESC;
```

LIMIT 10:

```
1930|2018|126
1933|2020|125
1929|2015|120
1928|2013|118
1930|2013|118
1931|2020|116
1928|2010|115
1928|2017|115
1927|2015|114
1925|2015|113
```

Interaction with Python

```
create table name_profession
(
nconst varchar(10) not null,
profession text not null,
foreign key(nconst) references name(nconst)
);
```

```
import sqlite3
from tqdm import tqdm

con = sqlite3.connect("imdb.sqlite3")
cur = con.cursor()

profession_list = cur.execute("SELECT nconst, primaryProfession FROM name").fetchall()

for professions in tqdm(profession_list):
    for profession in professions[1].split(','):
        data = (professions[0], profession)
        cur.execute("INSERT INTO name_profession VALUES(?,?)", data)
con.commit()
con.close()
```

Advanced Analysis

- The top 3 most common professions among these people and also the average life span of these three professions

```
SELECT profession,
COUNT(*) as frequency FROM name_profession
WHERE profession <> ""
GROUP BY profession
ORDER BY frequency DESC LIMIT 3;
```

```
actor|1208604
actress|729427
miscellaneous|50393
```

```
SELECT AVG(name.deathYear - name.birthYear)
FROM name_profession
JOIN name ON name.nconst = name_profession.nconst
WHERE name_profession.profession = 'actor' AND name.birthYear <> "\N" AND
name.deathYear <> "\N";
```

```
70.1, 73.5, 72.9
```

- The top 3 most popular (received most votes) genres

```
import sqlite3
from tqdm import tqdm

con = sqlite3.connect("./var/imdb.sqlite3")
cur = con.cursor()

try :
    cur.execute(
    '''
    CREATE TABLE title_genre
    (tconst varchar(10) not null,
    genre text not null,
    foreign key(tconst) references basics(tconst))
    '''
    )
    con.commit()
except: pass

genre_list = cur.execute("SELECT tconst, genres FROM basics").fetchall()

for genres in tqdm(genre_list):
    for genre in genres[1].split(','):
        data = (genres[0], genre)
        cur.execute("INSERT INTO title_genre VALUES(?,?)", data)

con.commit()
con.close()
```

```sql
SELECT t1.genre, SUM(t2.numVotes) AS sum_value
FROM title_genre AS t1
JOIN ratings AS t2 ON t1.tconst = t2.tconst
GROUP BY t1.genre
ORDER BY sum_value DESC
LIMIT 3;
```

```
Drama|466282920
Action|302449006
Comedy|286389971
```

- The average time span (endYear - startYear) of the titles for each person

```sql
import sqlite3
from tqdm import tqdm

con = sqlite3.connect("./var/imdb.sqlite3")
cur = con.cursor()

cur.execute(
'''
CREATE TABLE title
(nconst varchar(10) not null,
primaryName text not null,
title text not null,
foreign key(nconst) references name(nconst))
'''
)
con.commit()

title_list = cur.execute("SELECT nconst, primaryName, knownForTitles FROM
name").fetchall()

for titles in tqdm(title_list):
    for title in titles[2].split(','):
        data = (titles[0], titles[1], title)
        cur.execute("INSERT INTO title VALUES(?,?,?)", data)
con.commit()
con.close()
```

```sql
SELECT title .primaryName, AVG(basics.endYear-basics.startYear) AS avg_value
FROM title
JOIN basics  ON title .title = basics.tconst
GROUP BY title .nconst
ORDER BY avg_value DESC
LIMIT 10;
```

Output:

```
Marcel Linden|68.0
Willem Wansink|68.0
Oliver Bovelet|68.0
Shogo Akagawa|68.0
Kjell Göstein Resi|68.0
Gemma Casadevall|68.0
Ludger Karthausen|68.0
Piotr Buras|68.0
Vera Thiemann|68.0
Gil Yaron|68.0
```