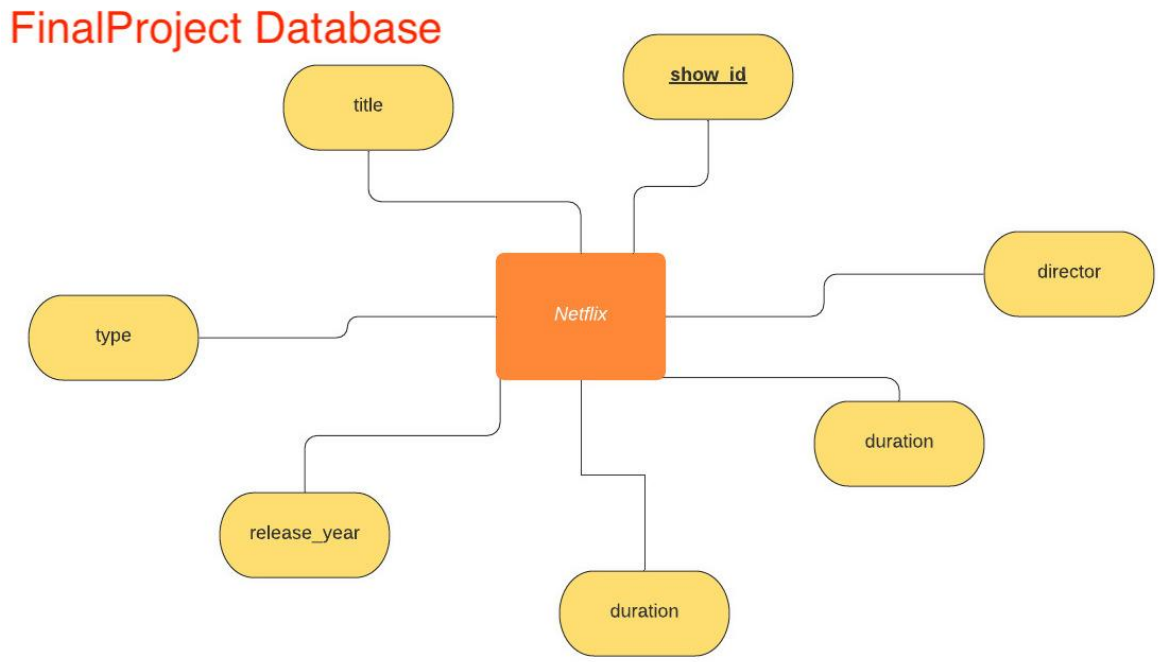
Vito Leone
12/9/21

Purpose:
The purpose of this database is to help users find shows or movies to watch on Netflix based on their specific interests. For example, If I wanted to watch a Stephen Speilburg movie tonight but I only had about 2 hours before I had to leave for dinner, I could search through this database and see that if I went on Netflix I could watch “The BFG”, “Indiana Jones and the Raiders of the Lost Ark”, “Indiana Jones and the Temple of Doom”, or “The Adventures of Tintin”. This database is great for people who know what their movie preferences are, but are still deciding on what to watch.

ER Diagram:



Schema:
Netflix(show_id, title, type, director, release_year, rating, duration);

- FDs: show_id -> title, type, director, release_year, rating, duration
    - Database was already in 3rd Normal Form

Questions and Corresponding Queries:
My queries contain many logic statements, many LIMIT clauses to keep the data at a readable size, some WHERE NOT clauses in order to exclude unwanted data, a couple of LIKE clauses to find keywords buried in strings, Use of the COUNT() and AVG() functions, and the use of GROUP BY and ORDER BY … DESC to organize specific data sets.

1, Find all the movies directed by Quentin Tarantino where the rating is PG-13 or R.
SELECT title FROM netflix WHERE director = "Quentin Tarantino" AND (rating = "PG-13" OR rating = "R");

2, Find 10 movies released between 2005 and 2010 that are longer than 120 minutes.
SELECT title FROM netflix WHERE type = "Movie" AND release_year > 2005 AND release_year < 2010 AND duration > 120 LIMIT 10;

3, Find the top 10 shows and release years that lasted longer than 4 seasons with a rating of TV-MA.
SELECT title, release_year FROM netflix WHERE type = "TV Show" AND duration > 4 AND rating = "TV-MA" LIMIT 10;

4, Find the top 10 shows with the word “Love” in the title released between 1990 and 2000.
SELECT title FROM netflix WHERE type = "TV Show" AND title LIKE '%love%' AND release_year >= 1990 LIMIT 10;

5, Find directors who made movies longer than 180 minutes before 1970 and the movies' titles.
SELECT director, title FROM netflix WHERE duration > 180 AND release_year < 1970 AND NOT director LIKE "%,%";

6, Find 10 movies or TV shows that have been rated G and are longer than 180 minutes or TV-G and longer than 4 seasons.
SELECT title, type FROM netflix WHERE (rating = "G" AND duration > 210) OR (rating = "TV-G" AND duration > 4) LIMIT 10;

7, Find 15 unique directors who have either a show rated TV-MA or a movie rated R since 2021.
SELECT DISTINCT(director) FROM netflix WHERE release_year >= 2021 AND (rating = "R" OR rating = "TV-MA") AND NOT director = "" AND NOT director LIKE "%,%" LIMIT 15;

8, Find the average year of all the titles released between 2000 and this year.
SELECT AVG(release_year) FROM netflix WHERE release_year >= 2000;

9, Find the longest movie with its duration.
SELECT title, duration FROM netflix WHERE duration = (SELECT MAX(duration) FROM netflix WHERE duration > 99);

10, Find the top 10 TV shows that don’t have a listed director and lasted longer than 3 seasons.
SELECT title FROM netflix WHERE type = "TV Show" AND director = "" AND duration > 3 LIMIT 10;

11, Find the 5 most listed directors and how many works they have.
SELECT director, COUNT(director) FROM netflix WHERE NOT director = "" GROUP BY director ORDER BY COUNT(director) DESC LIMIT 5;

My Grade and Why?:

I believe that I deserve full credit for the final project due to the work I have put into creating this database through the terminal and its results in an environment on pyCharm.

Terminal Side:

First, I downloaded the .csv file containing the 8807 unique entities. In order to transfer the .csv file into the MYSQL database I create for this project, I had to reset the MYSQL permissions using the following commands:

```
SET GLOBAL local_infile=1;
quit
mysql --local-infile=1 -u vitol -p
```

Once my permissions were reset, I created my table, and added the .csv data into my database using the command:

```
LOAD DATA LOCAL INFILE '/Users/vitoleone1127/Downloads/netflix_titles.csv' INTO TABLE Netflix FIELDS TERMINATED BY ',' ENCLOSED BY '"';
```

After I loaded in my data, I decided I needed to remove some unnecessary attributes. The original database contained attributes that were extensive in length, listed multiple value types for each entity, and cluttered up the database making it difficult to read results. The columns were “release_date” which was in a string format; “cast” containing the entire cast each show; “listed_in” which showed every genre the show contained (usually 4-5); and “description” which was a paragraph long summary of the show. I removed the 4 troublesome attributes and proceeded to make changes. I then noticed that the duration column was not in a INT format, but rather in a string containing either ‘min’ or ‘Season(s)’ based on if it was a show or movie. This made it difficult to compare the lengths of movies or the amount of seasons each show had. I used the following commands to get rid of the string endings:

```
UPDATE netflix SET duration = REPLACE(duration, 'min', '') WHERE duration LIKE '%min';
UPDATE netflix SET duration = REPLACE(duration, 'Season', '') WHERE duration LIKE '%Season';
UPDATE netflix SET duration = REPLACE(duration, 'Seasons', '') WHERE duration LIKE '%Seasons';
```

I then saved the database into a dumpfile and tested it on another database I have to make sure it loads in data properly.

<u>PyCharm Side:</u>

As a student who has not taken any python courses, I feel as if this project gave me the skills to navigate PyCharm and to write/understand the code I am creating. To start, I had to find a way to create a new venv file in the FinalProject repository as it was missing when we were sent our repository. This took some time and tinkering but eventually I was able to create the venv folder and install the proper packages for mysql to run through python. Next, I used dumpfile to transfer the insertion data into my create.sql file. In order to get my questions to run and print as a user input application, I used a while loop which takes user input and either prints out the input's answer, or the list of questions until the user enters “quit”.

All in all, I feel as if I have put in about 8-10 hours of trial and error on both the terminal and pyCharm sides of this project, but as you can see, it has turned out nicely. Though I only have 1 table, the primary key show_id functionally determines every other attribute in the table. Because of this, I thought it was appropriate to only use 1 table, just as we discussed in our first final project meeting. That is why I feel as if I deserve full credit for this project.