Nicholas Carnival
03/04/19

I chose to have the key attribute for every movie be the ISAN (International Standard Audiovisual Number) that is associated with the movie because there are some movies that have the same name and could confuse the database. This number will allow anyone to find the exact movie they are looking for with a query. I also chose to have the reviewer link to another table that has information about every reviewer because most movies are reviewed by similar reviewers and we don't want to list those reviewers every time we list a movie in the database. Another situation where I did a similar thing was with the production company. There are many movies that have multiple production companies and all production companies make more than one movie and would end up being listed tons of times throughout the database. I also chose to split the information about stars into its own table because stars appear in different movies and each star has a bunch of information in the database. This is similar to how imdb does this because every actor on imdb has their own page with the movies that they have been in on that page.

It is smart to not list things over and over in a database because it will be hard to make changes to those entities. We stop from creating repetitive data by splitting the relationships into multiple tables. If we were to list everything in one table, it would take a really long time to change an actors death date because we would have to find every occurence of them in the database and change it accordingly. With this model we would only have to change the actors information in one place and all of that information could be accessed through every movie that they are associated with. A similar situation occurs for both the reviewers and the publication companies. Any data that is used by multiple movies is useful to split into its own table in order to cut down on having the same information twice in the database. It is much easier to change the data in one table than to change the data from multiple location in the same table.

The data that I chose to leave in the 'Movie' table consists of the: ISAN, Sequels, Director, Reviewer Name, Release Year, Movie Title, Production Company, Writers, and Star Names. I felt that all of this data would not be repeated often throughout the database because we are not worried about data that involves the director other than the directors name. This model also allows for changes to easily be made. If we wanted to start recording data about the director we could create a new table with information about each director and link this table to each movie where the directors are the same. This allows us to scale up this database with any amount of data about movies because we can add tables about each aspect of the movie without having to change the initial data held in our database. This is useful to us because we would rather add data than risk removing the wrong data.

The only issue with the table in its current state depends on how much information the client wants about the writers, sequels, and director(s). With the current state of the table we only care about the name(s) of the directors, writers, and sequels. As opposed to websites such as IMDB where they keep information about everyone who is involved with the movie. If we wanted to do this we would have to add some more tables and link them to each other. For example: if we wanted to keep track of the date of birth of each director, we would create a new table to hold information about directors and add all of the data that we want about the directors into that table. We would then create a relationship between our ‘Movies’ table and our ‘Directors’ table so that every movie doesn't have to store all of the data for each director. It saves space and keeps the database cleaner.

I hope this model suits your needs client!