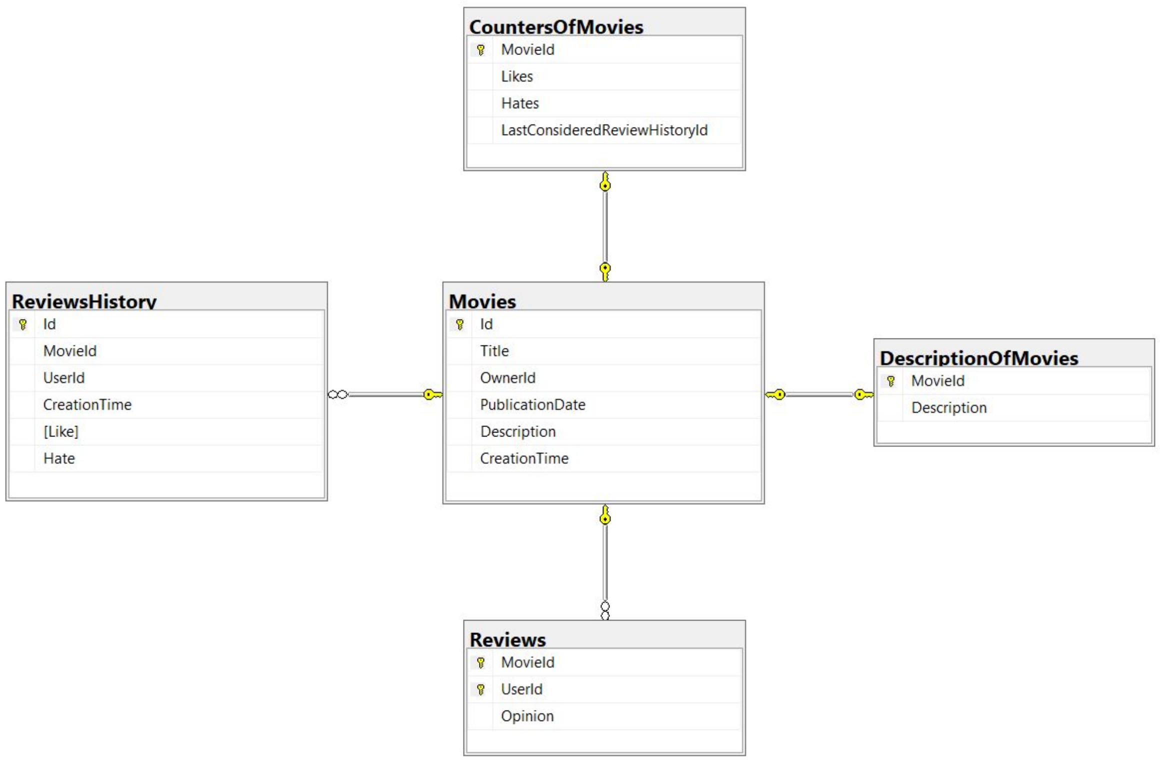
Database Model

The current implementation utilizes tables in an RDBMS (SQL server) even though a high traffic version of the site would require a queue mechanism to record review actions and not directly persist them in DB.



ReviewsHistory records each review action of the user :

Like	column [Like] = 1
Unlike	column [Like] = -1
Hate	column Hate = 1
Unhate	column Hate = -1

Reviews record the review opinion of the user:

- Like
- Unlike
- Neutral

CountersOfMovies records Like and Hate counters per movie. The table is updated by **UpdateCountersJob** which runs every 30 seconds on the server. **UpdateCountersJob** records the last utilized Id of ReviewHistory table in CountersOfMovies table so that consecutive reads do not recount everything, but only the newly added review actions.

DescriptionOfMovies records the description of the movie when the length of the description exceeds 300 characters. This is done so that a truncated version of the description is stored in the Movies table resulting in fewer reads when the list is loaded from the database server.

Movies record the main movie data including Publication Date, Creation Date, and OwnerId.

Like or Hate Concept:

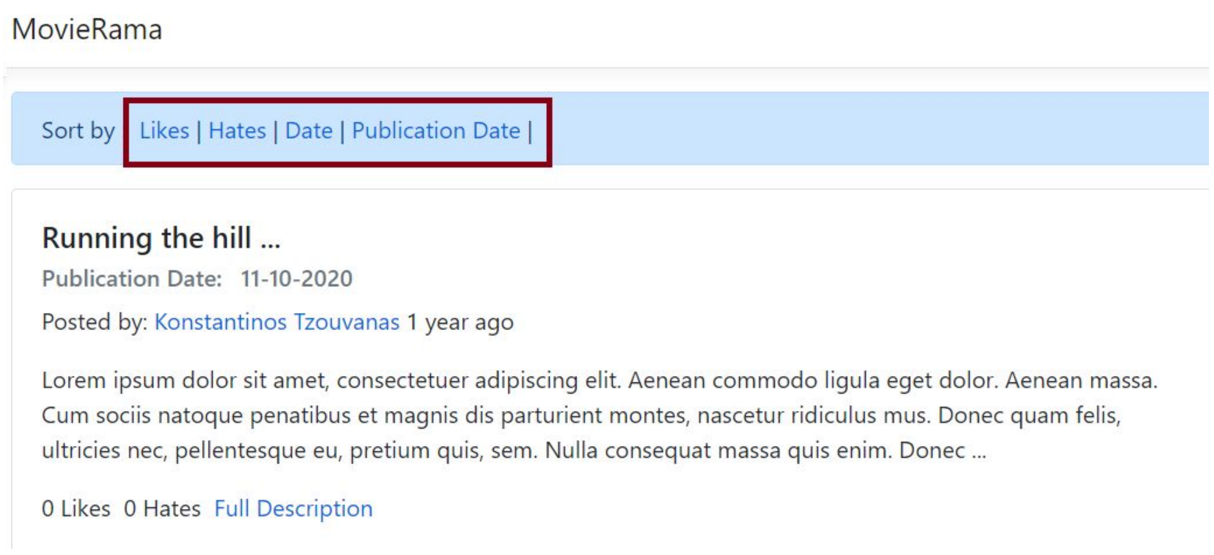
- The user likes a movie.
- Ajax request is sent to the server with review action and Id of the Movie.
- The server updates review opinion to Like in Reviews and inserts review action is ReviewsHistory.
- **UpdateCountersJob** executes updating movie CountersOfMovies
- Page refresh shows the updated movie counters in Like, Hate links. (1, 0)

- The user unlikes the same movie.
- Ajax request is sent to the server with review action and Id of the Movie.
- The server updates review opinion to Neutral in Reviews and inserts review action is ReviewsHistory.
- **UpdateCountersJob** executes updating movie CountersOfMovies
- Page refresh shows the updated movie counters in Like, Hate links. (0, 0)

- Every Like review action is recording 1 in Like column of ReviewsHistory
- Every, Unlike review action, is recording -1 in Like column of ReviewsHistory
- Every Hate review action is recording 1 in the Hate column of ReviewsHistory
- Every Unhate review action is recording -1 in the Hate column of ReviewsHistory

Summing the information per movie **UpdateCountersJob** can correctly increase or reduce the number of likes or hates per movie.

Sorting Behavior


MovieRama

Sort by Likes | Hates | Date | Publication Date |

Running the hill ...

Publication Date: 11-10-2020

Posted by: Konstantinos Tzouvanas 1 year ago

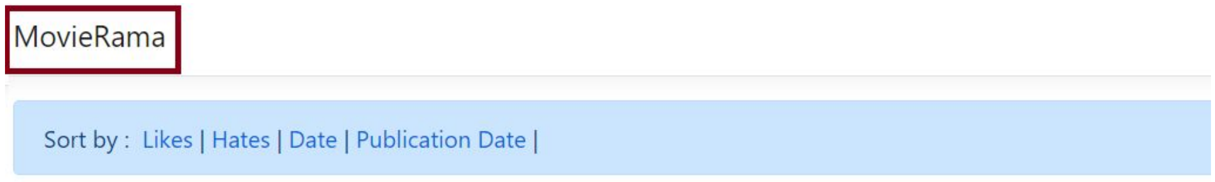
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium quis, sem. Nulla consequat massa quis enim. Donec ...

0 Likes 0 Hates Full Description

When the user clicks one of the **sort links**, data are sorted in ascending based on the clicked link. Re-clicking the same button swaps to the opposite ordering. As a result, the user can see data both sorted Asc or Desc by simply re-clicking.

Both the publication date and creation date are available as sort fields. The publication date is the date registered by the user when a movie is created (thus it can be a day in the past), while the creation date is updated by the system (with value {now}) when the movie is created.

Movies of User Behavior
When the user selects a specific movie owner, all movies of the owner are presented. Going back to the main list takes place by clicking the MovieRama title.


MovieRama
Sort by : Likes | Hates | Date | Publication Date |