DBMS ASSIGNMENT 5-Views

Views Exercises (Movie-Rating Database)

Schema:

Movie (mID, title, year, director)

English: There is a movie with ID number *mID*, a *title*, a release *year*, and a *director*.

Reviewer (rID, name)

English: The reviewer with ID number *rID* has a certain *name*.

Rating (rID, mID, stars, ratingDate)

English: The reviewer *rID* gave the movie *mID* a number of *stars* rating (1-5) on a certain *ratingDate*.

Each exercise asks you to create a view, and then write a query using that view, perhaps along with previously created views and/or the base tables.

1. Create a view called ***TNS*** containing title-name-stars triples, where the movie (title) was reviewed by a reviewer (name) and received the rating (stars). Then referencing only view ***TNS*** and table *Movie*, write a SQL query that returns the lastest year of any movie reviewed by Chris Jackson. You may assume movie names are unique.

2. Referencing view ***TNS*** from Exercise 1 and no other tables, create a view ***RatingStats*** containing each movie title that has at least one rating, the number of ratings it received, and its average rating. Then referencing view ***RatingStats*** and no other tables, write a SQL query to find the title of the highest-average-rating movie with at least three ratings.

3. Create a view ***Favorites*** containing rID-mID pairs, where the reviewer with rID gave the movie with mID the highest rating he or she gave any movie. Then referencing only view ***Favorites*** and tables ***Movie*** and ***Reviewer***, write a SQL query to return reviewer-reviewer-movie triples where the two (different) reviewers have the movie as their favorite. Return each pair once, i.e., don't return a pair and its inverse.

4. Create a view ***REVIEW*** *containing* reviewer name, movie title, stars, and ratingDate. Also, sort the data, first by reviewer name, then by movie title, and lastly by number of stars.

5. Create a view ***stars*** containing title,For each movie that has at least one rating, find the highest number of stars that movie received. Return the movie title and number of stars. Sort by movie title.

6. Create a view ***pairs*** *containing names of both reviewers.* For all pairs of reviewers such that both reviewers gave a rating to the same movie, return the names of both reviewers. Eliminate duplicates, don't pair reviewers with themselves, and include each pair only once. For each pair, return the names in the pair in alphabetical order.

7. Create a view ***director*** *containing title, director.* Some directors directed more than one movie. For all such directors, return the titles of all movies directed by them, along with the director name. Sort by director name, then movie title.

8. Create a view **HighLow** containing title, For each movie, return the title and the 'rating spread', that is, the difference between highest and lowest ratings given to that movie. Sort by rating spread from highest to lowest, then by movie title.