SQL movies

<https://rpubs.com/SulmanKhan/449883>

<https://gist.github.com/backpackerhh/2487a2c59789ef13099d>

<https://lagunita.stanford.edu/courses/DB/SQL/SelfPaced/courseware/ch-sql/seq-exercise-sql_movie_query_core/>

SELECT title FROM movie

WHERE director = "Steven Spielberg";

**Find all years that have a movie that received a rating of 4 or 5, and sort them in increasing order.**

SELECT DISTINCT movie.year

FROM movie JOIN rating

ON movie.mID = rating.mID

WHERE rating.stars >= 4

ORDER BY movie.year ASC;

**Find the titles of all movies that have no ratings.**

SELECT title from movie

LEFT JOIN rating ON

movie.mID = rating.mID

WHERE rating.mID is NULL;

**Some reviewers didn't provide a date with their rating. Find the names of all reviewers who have ratings with a NULL value for the date**.

SELECT name FROM reviewer

JOIN rating

ON reviewer.rID = rating.rID

WHERE ratingDate is NULL;

**Write a query to return the ratings data in a more readable format: 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.**

SELECT name AS "reviewer name", title AS "movie title", stars, ratingDate FROM reviewer

JOIN rating ON

reviewer.rID = rating.rID

JOIN movie ON

rating.mID = movie.mID

ORDER BY name ASC, title ASC, stars ASC;

**For all cases where the same reviewer rated the same movie twice and gave it a higher rating the second time, return the reviewer's name and the title of the movie.**

SELECT name, title

FROM Movie

INNER JOIN Rating R1 USING(mId)

INNER JOIN Rating R2 USING(rId, mId)

INNER JOIN Reviewer USING(rId)

WHERE R1.ratingDate < R2.ratingDate AND R1.stars < R2.stars;

**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.**

SELECT title, MAX(stars) FROM movie

JOIN rating ON

rating.mID = movie.mID

GROUP BY rating.mID, title

HAVING MAX(stars)

ORDER BY title ASC;

**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.**

SELECT title, (MAX(stars) - MIN(stars)) AS rating\_spread FROM movie

JOIN rating ON

rating.mID = movie.mID

GROUP BY rating.mID

ORDER BY rating\_spread DESC, title;

**Find the difference between the average rating of movies released before 1980 and the average rating of movies released after 1980. (Make sure to calculate the average rating for each movie, then the average of those averages for movies before 1980 and movies after. Don't just calculate the overall average rating before and after 1980.)**

SELECT AVG(Before1980.avg) - AVG(After1980.avg)

FROM (

SELECT AVG(stars) AS avg

FROM Movie

INNER JOIN Rating USING(mId)

WHERE year < 1980

GROUP BY mId

) AS Before1980, (

SELECT AVG(stars) AS avg

FROM Movie

INNER JOIN Rating USING(mId)

WHERE year > 1980

GROUP BY mId

) AS After1980;

<https://lagunita.stanford.edu/courses/DB/SQL/SelfPaced/courseware/ch-sql/seq-exercise-sql_movie_query_extra/>

**Find the names of all reviewers who rated Gone with the Wind.**

SELECT DISTINCT name FROM reviewer

JOIN rating ON

reviewer.rID = rating.rID

JOIN movie ON

rating.mID = movie.mID

WHERE movie.mID = "101";

**For any rating where the reviewer is the same as the director of the movie, return the reviewer name, movie title, and number of stars.**

SELECT name, title, stars FROM reviewer

JOIN rating ON

reviewer.rID = rating.rID

JOIN movie ON

rating.mID = movie.mID

WHERE name = director;

**Return all reviewer names and movie names together in a single list, alphabetized. (Sorting by the first name of the reviewer and first word in the title is fine; no need for special processing on last names or removing "The".)**

SELECT title FROM movie

UNION

SELECT name FROM reviewer

ORDER BY title, name;

<https://lagunita.stanford.edu/courses/DB/SQL/SelfPaced/courseware/ch-sql/seq-exercise-sql_social_query_core/>

<https://gist.github.com/DmitrySandalov/1354852>

**Find the names of all students who are friends with someone named Gabriel.**

SELECT DISTINCT name FROM Highschooler

WHERE ID in (SELECT ID1 FROM Friend WHERE ID2 in (SELECT ID FROM Highschooler WHERE name="Gabriel"));