תרגיל 2 במערכות מסד נתונים – מפרט

22.05.18

מגישים:

סהר כהן, 206824088

גיל צימרמן, 318179496

בסיס הנתונים:

בשימוש בסיס הנתונים נמצא טיפוס חדש **movie_rating**. זהו enum בעל שני מופעים: 'LIKE' ו – 'DISLIKE'.

1) Viewers: טבלת כל הצופים במערכת. מורכבת משני Attributes:

- viewer_id: מספר מזהה של צופה במערכת.
- viewer_name: שם של צופה במערכת.

מגבלות:

- viewer_id הוא המפתח הראשי,
- viewer_id > 0,
- viewer_name אינו NULL.

2) Movies: טבלת כל הסרטים במערכת. מורכבת משלושה Attributes:

- movie_id: מספר מזהה של סרט במערכת.
- movie_name: שם של סרט במערכת.
- movie_description: תיאור של סרט במערכת.

מגבלות:

- movie_id הוא המפתח הראשי,
- movie_id > 0,
- movie_name אינו NULL (לא אופציונאלי),
- movie_description אינו NULL (לא אופציונאלי).

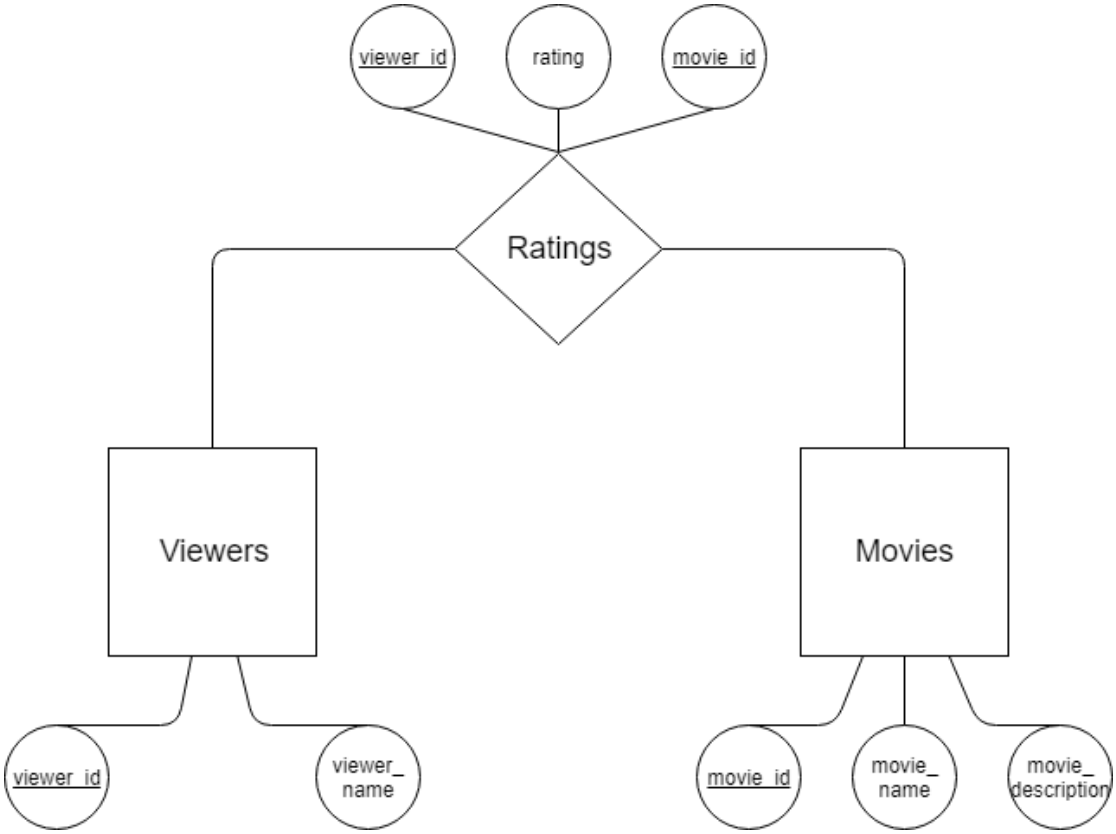
3) Ratings: יחס המתאר את קשר הצפייה בין צופה וסרט במערכת. מורכבת משולשה Attributes:

- viewer_id: מספר מזהה של צופה אשר קיים בטבלת הצופים.
- movie_id: מספר מזהה של סרט אשר קיים בטבלת הסרטים.
- rating: דירוג הצופה עבור הסרט: מטיפוס movie_rating.

מגבלות:

- viewer_id הוא מפתח זר המתאים למזהה של צופה בטבלה Viewers.
- movie_id הוא מפתח זר המתאים למזהה של צופה בטבלה Movies.
- viewer_id, movie_id הם המפתח הראשי,
- movie_rating יכול לקבל את הערך NULL (זוהי לא מגבלה, אלא הערה)
- כאשר נמחק צופה / סרט מהמערכת: הסר את כל ה – tupples בטבלת הדירוגים המתאימים לצופה / סרט אשר הוסרו (ON CASCADE delete).

להלן שרטוט ERD של בסיס הנתונים:



בקובץ הקוד Solution.java מופיעות כל השאילות של הפתרון, יחד עם הערות המסבירות את הסמנטיקה שמאחורי שאילתות ה – SQL בעבודה עם בסיס הנתונים. להלן פעולות ה – API: עם הסבר עבור השאילות:

CRUP API:

createViewer: הוספה (INSERT) של צופה נתון לתוך הטבלה Viewers.

getViewer: החזרה (SELECT) של הצופה המתאים ב – Viewers, לפי viewer_id נתון.

deleteViewer: מחיקה (DELETE) של צופה נתון מתוך הטבלה Viewers.

updateViewer: עדכון (UPDATE) של צופה נתון בטבלה Viewers.

createMovie: הוספה (INSERT) של סרט נתון לתוך הטבלה Movies.

getMovie: החזרה (SELECT) של הסרט המתאים ב – Movies, לפי movie_id נתון.

deleteMovie: מחיקה (DELETE) של סרט נתון בטבלה Movies.

updateMovie: עדכון (UPDATE) של סרט נתון בטבלה Movies.

BASIC API:

addView: הוספה (INSERT) של שלישייה (viewer_id, movie_id, NULL) לתוך הטבלה Ratings. יש לשים לב שבזמן <u>ההוספה</u>: לא קיים ערך דירוג movie_rating של הצופה עבור הסרט, ולכן הערך הנכנס הוא NULL.

removeView: מחיקה (DELETE) של השורה ובה הצופה הנתון והסרט הנתון.

getMovieViewCount: ספירה (COUNT) של מספר השורות בהן הופיע סרט נתון – הדבר מציין את מספר הצפיות על ידי צופים שונים שצבר הסרט הנתון.

addMovieRating: עדכון (UPDATE) של דירוג הצופה הנתון עבור הסרט הנתון, עם ערך דירוג חדש נתון ('LIKE' או 'DISLIKE').

removeMovieRating: עדכון (UPDATE) של דירוג הצופה הנתון עבור הסרט הנתון, עם ערך דירוג לא קיים NULL.

getMovieLikesCount: ספירה (COUNT) של מספר ה – tuples בטבלה בהן מופיע הסרט עם ה – movie_id הנתון, והדירוג הוא 'LIKE'. הדבר שקול למספר ה LIKEs שקיבל הסרט על ידי צופים שונים.

getMovieDislikesCount: ספירה (COUNT) של מספר ה – tuples בטבלה בהן מופיע הסרט עם ה – movie_id הנתון, והדירוג הוא 'DISLIKE'. הדבר שקול למספר ה DISLIKEs שקיבל הסרט על ידי צופים שונים.

ADVANCED API:

getSimilarViewers:

השאילתה עושה שימוש בטבלה Ratings וטבלאות הנוצרות בשאילתה
ViewerMovies - מחזירה את רשימת הסרטים שראה הצופה שהוכנס כקלט

SimilarViewers – מבצעת JOIN בין הטבלה ViewerMovies וRatings ומבצעת COUNT על movie_id.
השאילתה מחזירה את כמות הצפיות של כל צופה אחר עבור כל סרט שצפה הצופה שהוכנס כקלט

כעת השאילתה משתמשת בSimilarViewers כדי לקחת רק את הצופים שמספר הסרטים שראו גדול מ75% ממספר הסרטים ראה הצופה שהוכנס כקלט.

mostInfluencingViewers:

השאילתה עושה שימוש בטבלה Ratings וטבלאות Sub Queris:

ViewList – מחזירה את מספר הצפיות של כל צופה במערכת tuple(viewer_id,amount)

RatingList – מחזירה את מספר הפעמים שכל צופה במערכת דירג סרט tuple(viewer_id,amount)

השאילתה מבצעת LEFT JOIN בין ViewList ל RatingList כך שViewList בצד שמאל של הJOIN, זאת משום שכל צופה שראה סרט יכול לדרג אך כל צופה שדירג סרט בהכרח צפה בסרט. לאחר מכן השאילתה לוקחת רק את הviewer_id וממיינת לפי הנדרש.

getMoviesRecommendations:

השאילתה עושה שימוש בטבלה Ratings ובשאילתה SimilarViewrs כSub Query
תחילה עבור כל הצופים הדומים מסתכלים על כל הסרטים שצופה הקלט לא ראה ועבורם לוקחים רק את אלו שדורגו בLIKE וסופרים את כמות הלייקים. עושים RIGHT OUTER JOIN עם MovieList Sub Query אשר מחזיק את כל הסרטים שצפו הצופים הדומים אך צפה הקלט לא ראה, כך שנמצאים שם גם הסרטים שדורגו בלייק גם אלו שדורגו בDISLIKE וגם אלו שלא דורגו כלל. הSub Query נמצאת בצד ימין של הJOIN. לאחר הJOIN מבצעים מיון לפי הנדרש.

getConditionalRecommendations:

השאילתה משתמשת בטבלה Ratings ובשאילתה SimilarViewers אך עם החמרה בתנאים.
יוצרים מעטפת לשאילתה SimilarViewers ומוודאים שישארו רק הצופים הדומים שדירגו את סרט הקלט באותו דירוג כמו צופה הקלט.
לאחר מכן עבור כל המדרגים הדומים מסתכלים על כל הסרטים שצופה הקלט לא ראה ועבורם לוקחים רק את אלו שדורגו בLIKE וסופרים את כמות הלייקים. עושים RIGHT

OUTER JOIN עם MovieList Sub Query אשר מחזיק את כל הסרטים שצפו המדרגים הדומים אך צופה הקלט לא ראה, כך שנמצאים שם גם הסרטים שדורגו בלייק גם אלו שדורגו בDISLIKE וגם אלו שלא דורגו כלל. הSub Query נמצאת בצד ימין של הJOIN. לאחר הJOIN מבצעים מיון לפי הנדרש.