SWE 521 – Database Systems
Assignment 2 Report

Student Yana Krasovska
ID: 2022719111
Date: 05.06.2023

This submission is for assignment 2 “Movie Database” of course 521 Database Systems. The submission consists of:

- this report with the description of the schema refinement.
- updated Entity-Relationship diagrams as part of the report.
- A README.txt file with explanations of how to run the code.
- Java project source files.

Contents of this report:

1. Schema refinement details:
    a. Description of the schema refinement, including CHECK constraints.
    b. Functional dependencies and normal forms of relations.
2. Updated ER and logical ER diagrams.

1. Schema refinement details

a. Description of the schema refinement

The schema was refined during the implementation of the required use cases. Based on the use cases described in the Requirements section of the assignment, the following relations/attributes were removed/modified/introduced from/in/to the schema to perform the requested operations:

- Nationality – was made an attribute of Director relation to avoid unnecessary complexity not justified by the required use cases.

- Agreement – was removed as not required for the described use cases.

- District attribute – was removed from Theatre and other relations as not instrumental for the required use cases.

- Average_rating attribute – introduced in the Movie relation to handle the appropriate use cases, with a CHECK constraint for values between 0 and 5 for the average_rating. This is the only additional CHECK constraint introduced; the rest of the CHECK constraints persisted from the previous version of the database.

- Database manager, Genre, Rating Platform – these entities remain in the ER diagram, but the *tables* were removed from the database (as instructed in the requirements, they are hard-coded now). Consequently, Foreign keys referencing Rating Platform and Genre were removed from other relations.

Additionally, for User, Director, and Audience relations, transactions were used for cases when multiple of these relations were modified.

To implement the use case 3 correctly, the Database Managers were made to only be able to delete audiences, not director.

To implement the use case involving rating movies correctly, the use case for Audience to “Subscribe to platform” was introduced and handled.

b. Functional dependencies and normal forms of tables.

Users (username, password, name, surname, account_type_

Functional dependencies (non-trivial):
{username} ⟶ {password, name, surname, account_type} ==>

{username} ⟶ {password }, {username} ⟶ { name}, {username}⟶ { surname},
{username}⟶ { account_type}
Candidate keys: {username}; {username, password}; {username, name, surname}; {username, account_type}
All attributes are part of a candidate key here and so are prime attributes (3NF); username is the minimal candidate key and is the superkey for each functional dependency (BCNF).

Director (username, nationality, platform_id, account_type)

Functional dependencies (non-trivial):
{username} ⟶ {nationality, platform_id, account_type} ==>
{username}⟶ {nationality }, {username} ⟶ {platform_id}, {username} ⟶ {account_type},

Candidate keys: {username}

Each non-prime attribute in this relation depends on the username (2NF); non-prime attributes do not determine other non-prime attributes (3NF); and username is the superkey for each functional dependency (BCNF).

Audience (username, account_type)

Functional dependencies:
{username} ⟶ { account_type}
This table is in BCNF, obviously.

Theatre (theatre_id, theatre_name, capacity)

Functional dependencies:
{theatre_id}⟶ { theatre_name, capacity} ==>
{theatre_id}⟶ { theatre_name}, {theatre_id} ⟶{ capacity}

Candidate keys: {theatre_id}, {theatre_name, capacity}

All attributes in this relation are prime attributes (3NF); and theatre_id is the superkey for each functional dependency (BCNF).

Subscription (audience_username, platform_id)
This table has no non-trivial functional dependencies. It is obviously in BCNF.

MovieGenreList (movie_id, genre_id)
This table has no non-trivial functional dependencies. It is obviously in BCNF.

WatchedMovie (audience_username, movie_id)
This table has no non-trivial functional dependencies. It is obviously in BCNF.

Movie (movie_id, movie_name, director_username, platform_id, prequel_movie_id, duration, average_rating)

Functional dependencies:
{movie_id} ⟶ {movie_name, director_username, prequel_movie_id, duration, average_rating}
==> {movie_id} ⟶ {movie_name}, {movie_id} ⟶ {director_username},
{movie_id} ⟶ {prequel_movie}, {movie_id} ⟶ {duration}
{movie_id} ⟶ {average_rating}
{director_username} ⟶ {platform_id}
Since {movie_id} ⟶ {director_username} ==> {movie_id} ⟶ {platform_id}
{prequel_movie_id} ⟶ {movie_id}

Candidate keys: {movie_id}

Platform_id as a non-prime attribute depends on part (director_username) of the {movie_name, director_username} candidate key, which violates 2NF. Platform_id is required in this relation because of the constraint in Rating table which makes sure that rating is given to the movie on the correct platform.

MovieSession (session_id, movie_id, theatre_id, screening_date, from_timeslot, to_timeslot)

Functional dependencies:
{session_id} ⟶ {movie_id, theatre_id, screening_date, from_timeslot, to_timeslot}
==> {session_id} ⟶ {movie_id}, {theatre_id}, {screening_date}, {from_timeslot}, {to_timeslot}

Candidate keys: {session_id}

Each non-prime attribute depends on the minimal candidate key session_id (2NF); non-prime attributes do not determine other non-key attributes (3NF); and session_id is the superkey for each functional dependency (BCNF).

Ticket (audience_username, session_id, movie_id, prequel_movie_id,)

Functional dependencies:
{Audience_username, session_id} ⟶ {movie_id, prequel_movie_id}
{Prequel_movie_id} ⟶ {movie_id}, {session_id} ⟶ {movie_id, prequel_movie_id}
{Movie_id} ⟶ {prequel_movie_id}

Candidate keys: {audience_username, session_id}, {session_id, movie_id}, {session_id, prequel_movie_id}

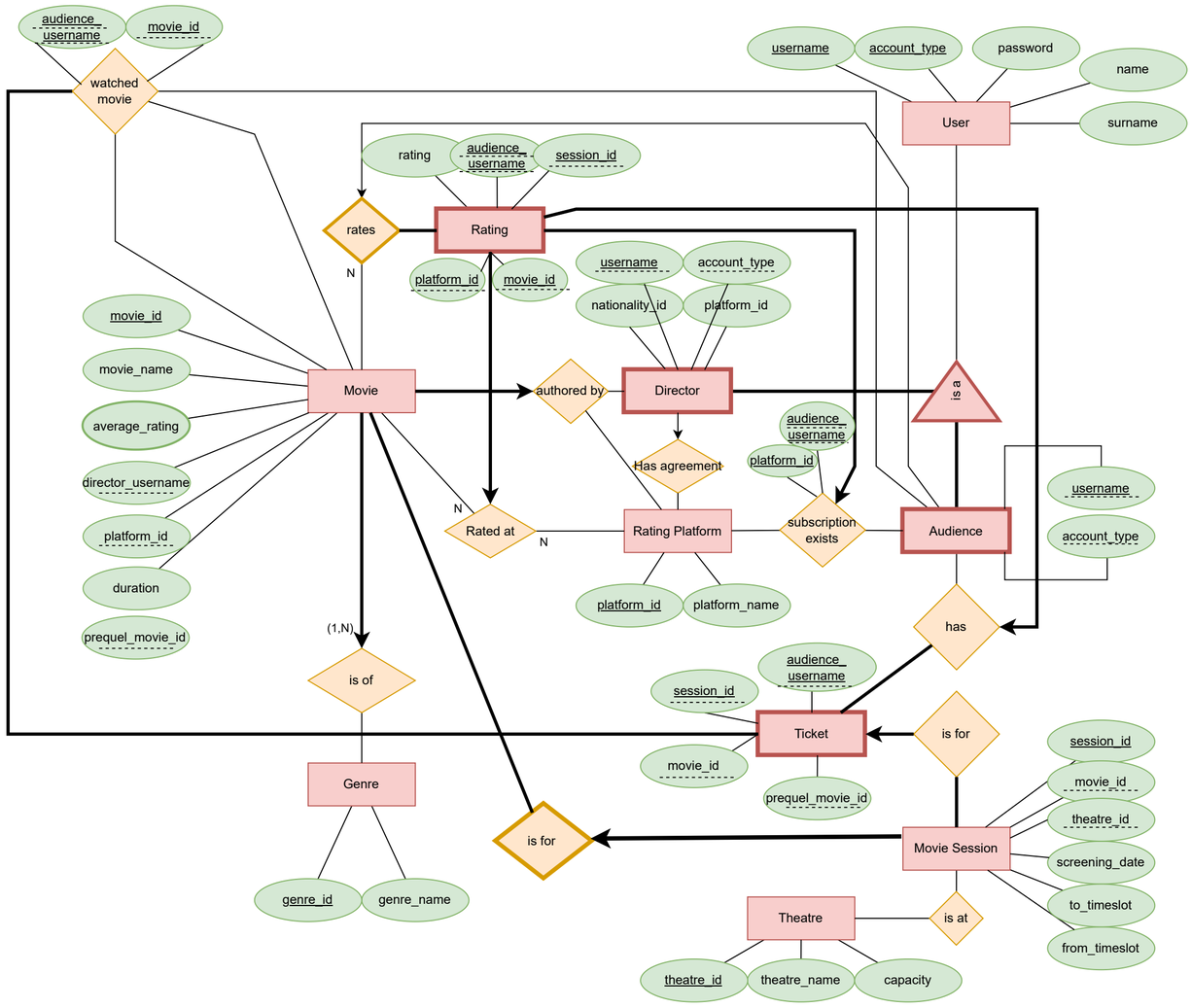
All attributes in this relation are prime attributes (3NF); all attributes in this relation depend on the superkey {audience_username, session_id} (BCNF).

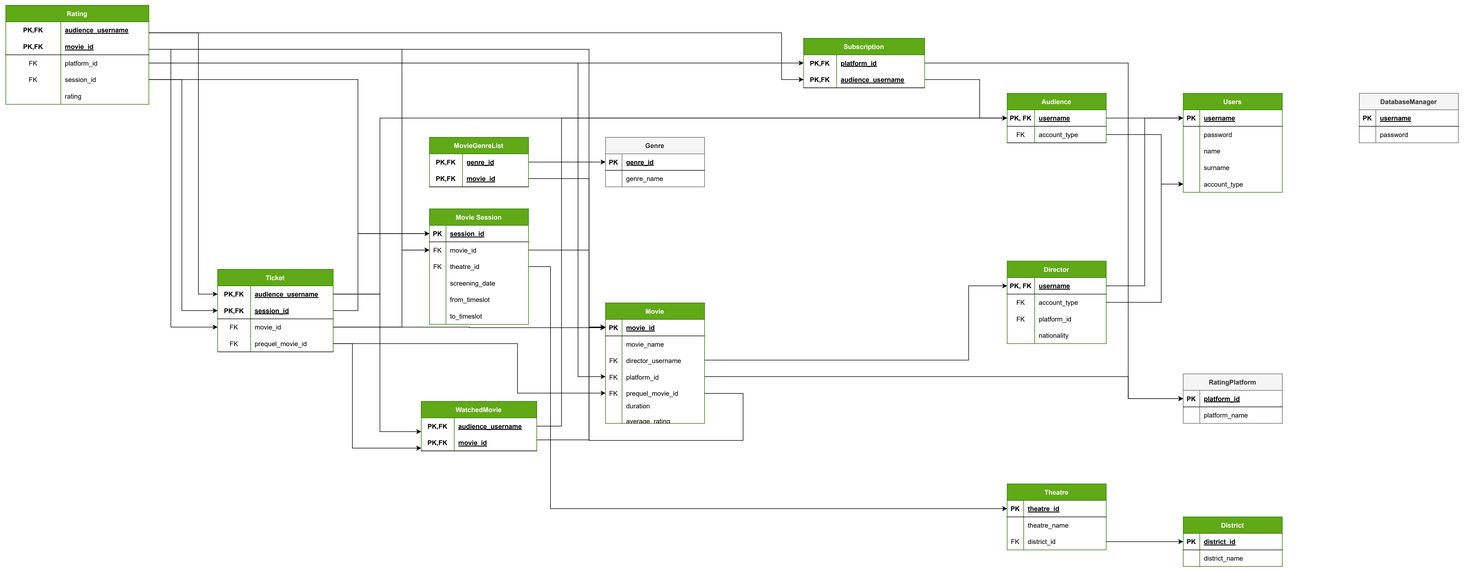
Rating (audience_username, movie_id, session_id, platform_id, rating)

Functional dependencies:
{audience_username, movie_id} ⟶ {rating}
{session_id} ⟶ {movie_id}, {session_id} ⟶ {platform_id}
{movie_id} ⟶ {platform_id}

Candidate keys: {audience_username, movie_id}
Non-prime attributes are session_id, platform_id, rating.
Platform_id depends on the part (movie_id) of a candidate key, which violates 2NF. We need the platform_id in this relation to reference both Subscription and Movie relations to make sure that audience has Subscription for the same Platform where the Movie is listed, otherwise Rating cannot be given.

2. Updated ER and logical ER diagrams


audience_username
movie_id
watched movie
username
account_type
password
name
surname
User
rating
audience_username
session_id
rates
Rating
N
platform_id
movie_id
username
account_type
nationality_id
platform_id
movie_id
movie_name
average_rating
director_username
platform_id
duration
prequel_movie_id
Movie
authored by
Director
is a
audience_username
platform_id
Has agreement
N
Rated at
N
Rating Platform
subscription exists
Audience
username
account_type
platform_id
platform_name
has
(1,N)
is of
audience_username
session_id
Ticket
is for
movie_id
prequel_movie_id
session_id
movie_id
theatre_id
screening_date
to_timeslot
from_timeslot
Genre
is for
Movie Session
Theatre
is at
genre_id
genre_name
theatre_id
theatre_name
capacity


Rating
PK,FK audience_username
PK,FK movie_id
FK platform_id
FK session_id
rating
Subscription
PK,FK platform_id
PK,FK audience_username
Audience
PK, FK username
FK account_type
Users
PK username
password
name
surname
account_type
DatabaseManager
PK username
password
MovieGenreList
PK,FK genre_id
PK,FK movie_id
Genre
PK genre_id
genre_name
Movie Session
PK session_id
FK movie_id
FK theatre_id
screening_date
from_timeslot
to_timeslot
Ticket
PK,FK audience_username
PK,FK session_id
FK movie_id
FK prequel_movie_id
Director
PK, FK username
FK account_type
FK platform_id
nationality
Movie
PK movie_id
movie_name
FK director_username
FK platform_id
FK prequel_movie_id
duration
average_rating
RatingPlatform
PK platform_id
platform_name
WatchedMovie
PK,FK audience_username
PK,FK movie_id
Theatre
PK theatre_id
theatre_name
FK district_id
District
PK district_id
district_name