Online Movie Theater

Created By:

Ivan Lohunkov(241069)

Pavlo Balan(241004)

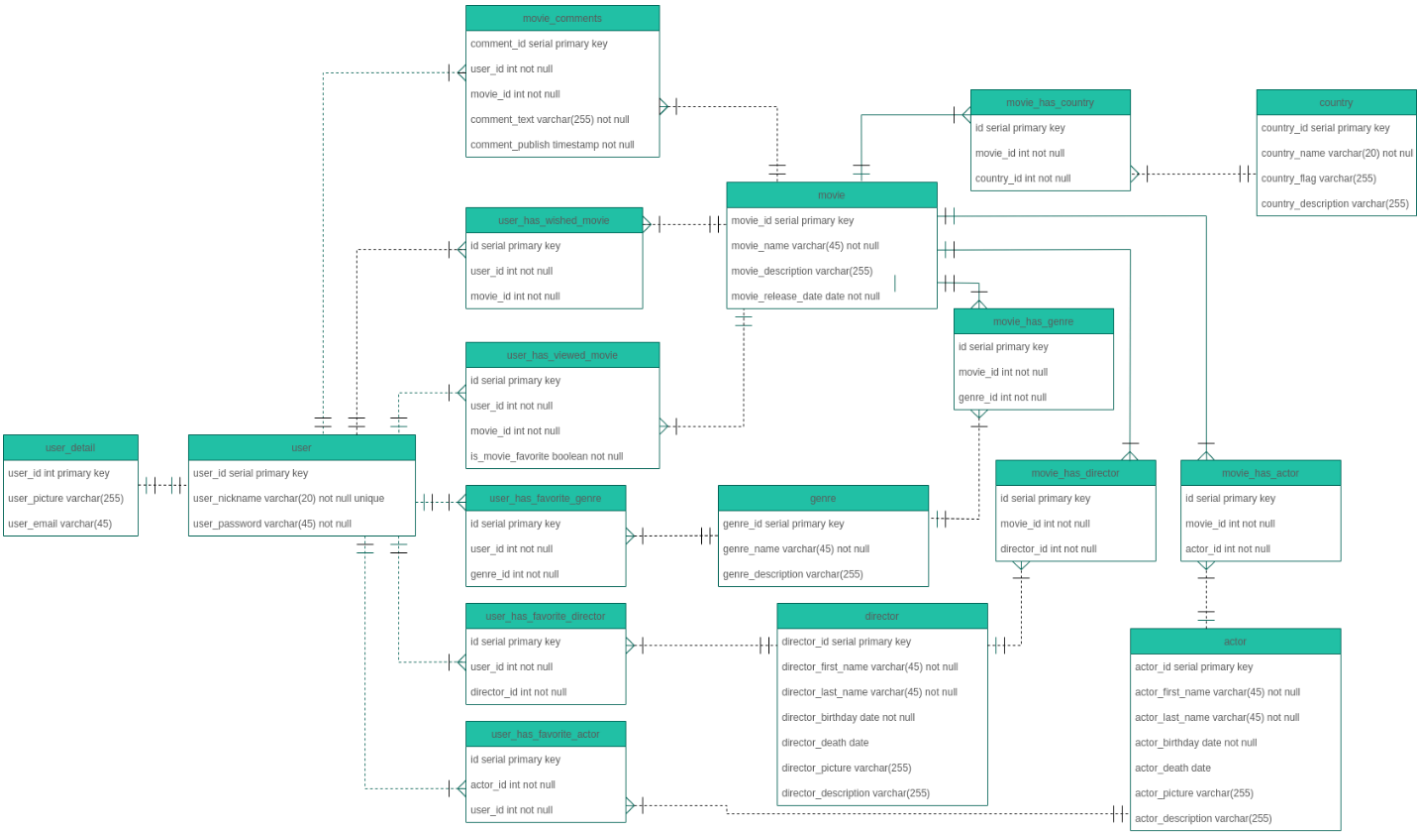
Application description

Our application is an online movie theater. It can find different kinds of movies that are stored in a database.

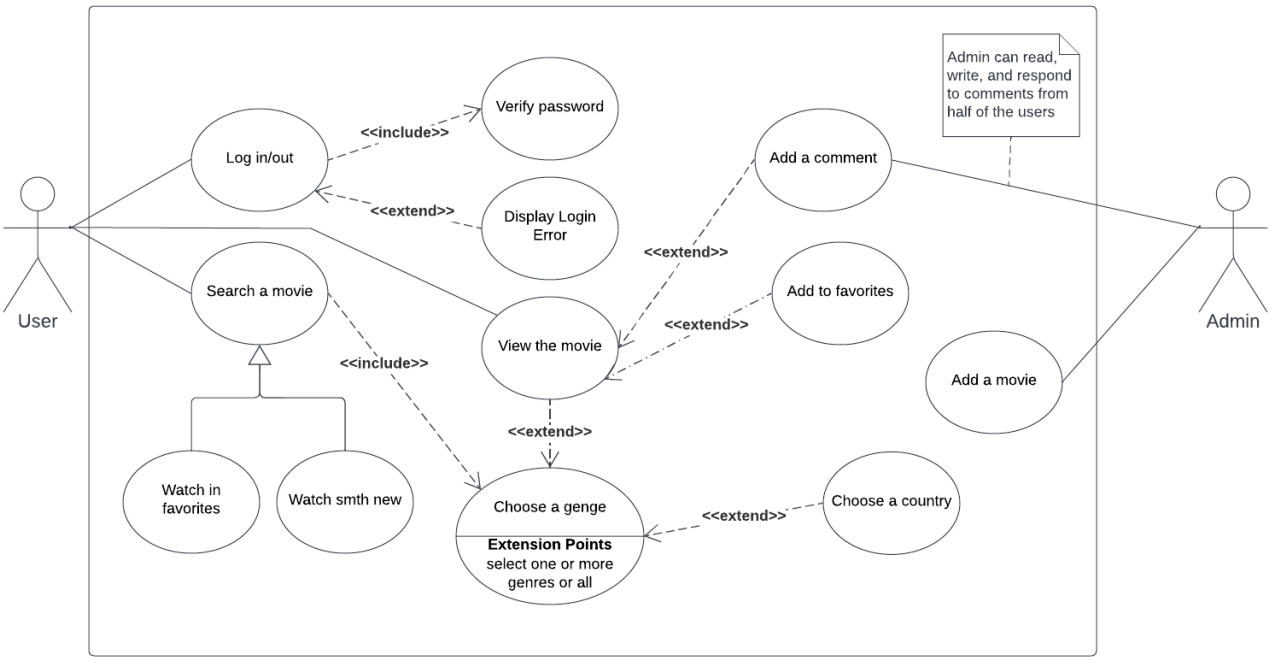
How the process itself takes place: The user registers/logs into our application, and enters the movie he is interested in into the search box. Or he can select the genre of film he is interested in (if he could not find the film from the recommended list), and the user can also select the country in which the particular film was released. After viewing, the user has the ability to add the movie to their favorites or, if the viewer has not finished the movie or has a desire to watch it in the future, they can add it to their wish list so the user can't lose the movie they found. There is also a function to add the director, actors, genres to your favorites. The user has his own profile in the app, where his nickname and all his favorite movies are stored. Our app also contains the data of actors and directors of films. This is done so that the user can immediately find out brief information about the film of interest.

The user can also write a comment about the movie after watching it. It is important to know that for each film the data of the actors and directors who worked on this film are attached, so that the user can immediately find everything he needs in the description. There is also a history of watched movies in our application. You can use it to see what the user has watched before.

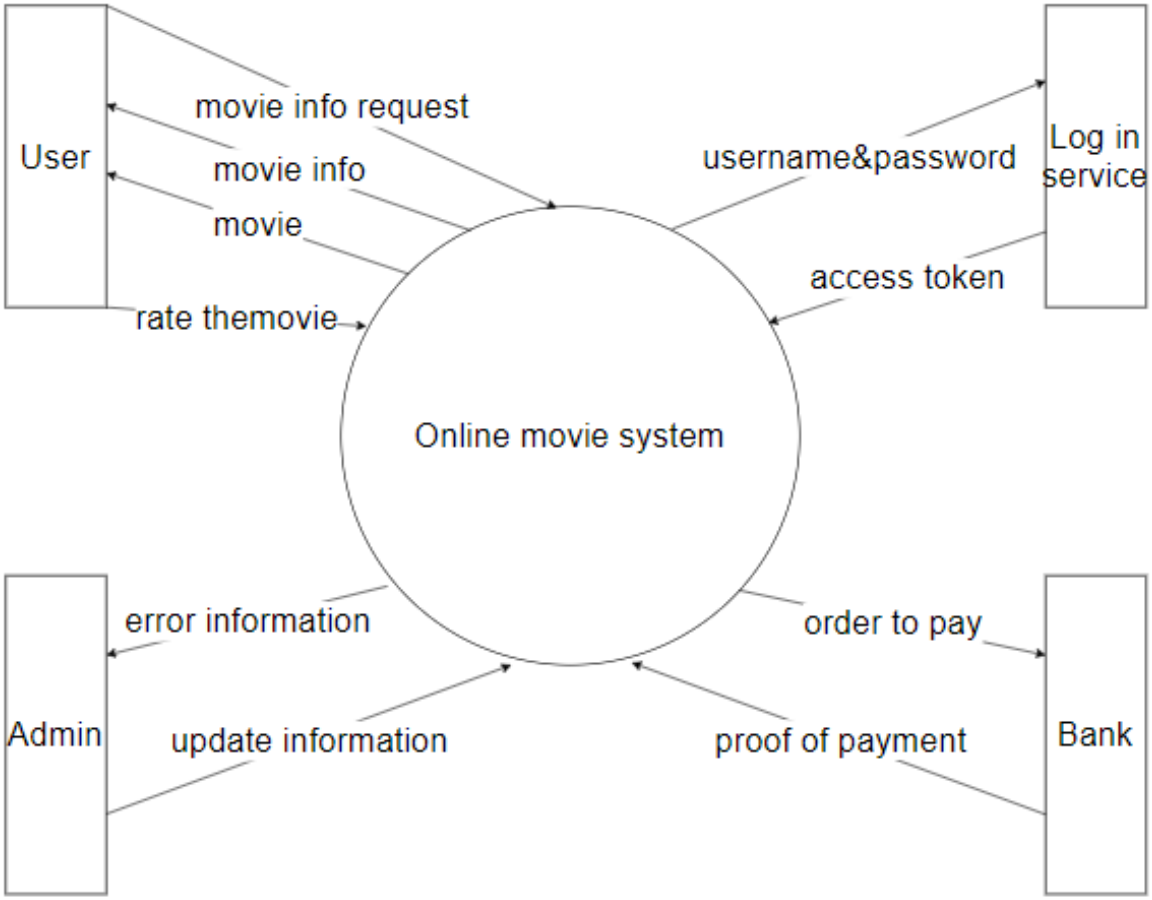
MovieDB ERD-Diagram



Use-case diagram



System-context diagram



List of suitable non-functional and functional requirements

- The system must allow users to watch movies.
- The system must allow users to find directors, actors, genres, movies.
- The system must allow users to add movies, actors, directors, genres to favorites list.
- The system shall store users.
- The system shall store user's comments.
- The system shall store user's

Description for each table

1. user - User of the online movie site + (login, password)
2. user detail - user data, such as mail and avatar
3. movie_comments - comments on the movie the user watched
4. user_has_wished_movie - A list of desired movies that the user wants to watch
5. user_has_viewed_movie - movies watched by the user
6. user_has_favorite_genre - Favorite movie genres are saved in the user profile
7. user_has_favorite_director - favorite directors are saved in the user profile
8. user_has_favorite_actor - Favorite actors are saved in the user profile
9. movie - list of films
10. genre - list of genres
11. director - list of directors
12. movie_has_country - the film has its own country of release
13. movie_has_genre - the film has its own genre
14. movie_has_director - the film has its own director
15. movie_has_actor - the film has its own actors
16. actor - list of actors in movies
17. country - the release range of a particular film

Description why do you think that your database is in 3rd normal form?

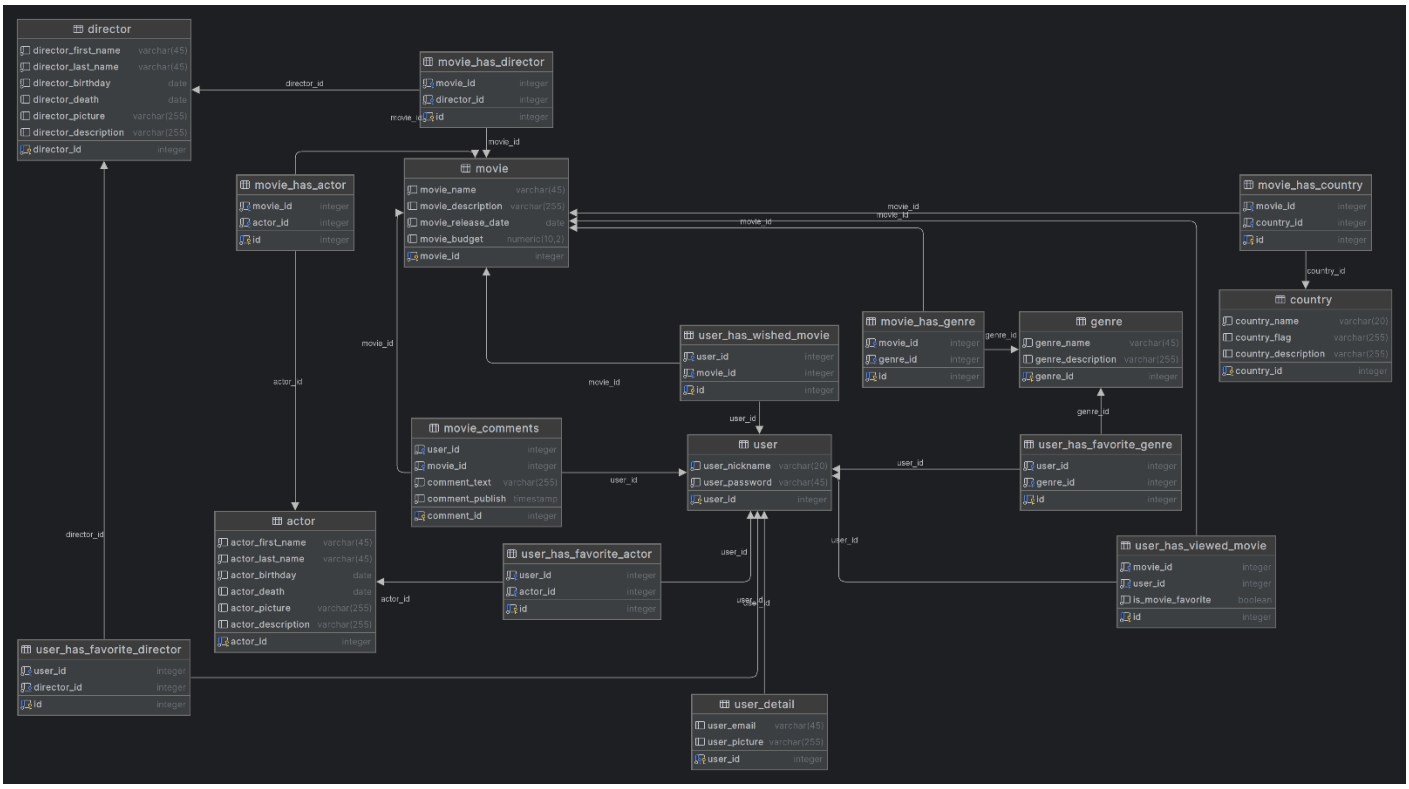
3NF (Third Normal Form) Rules

- Rule 1- Be in 2NF
- Rule 2- Has no transitive functional dependencies

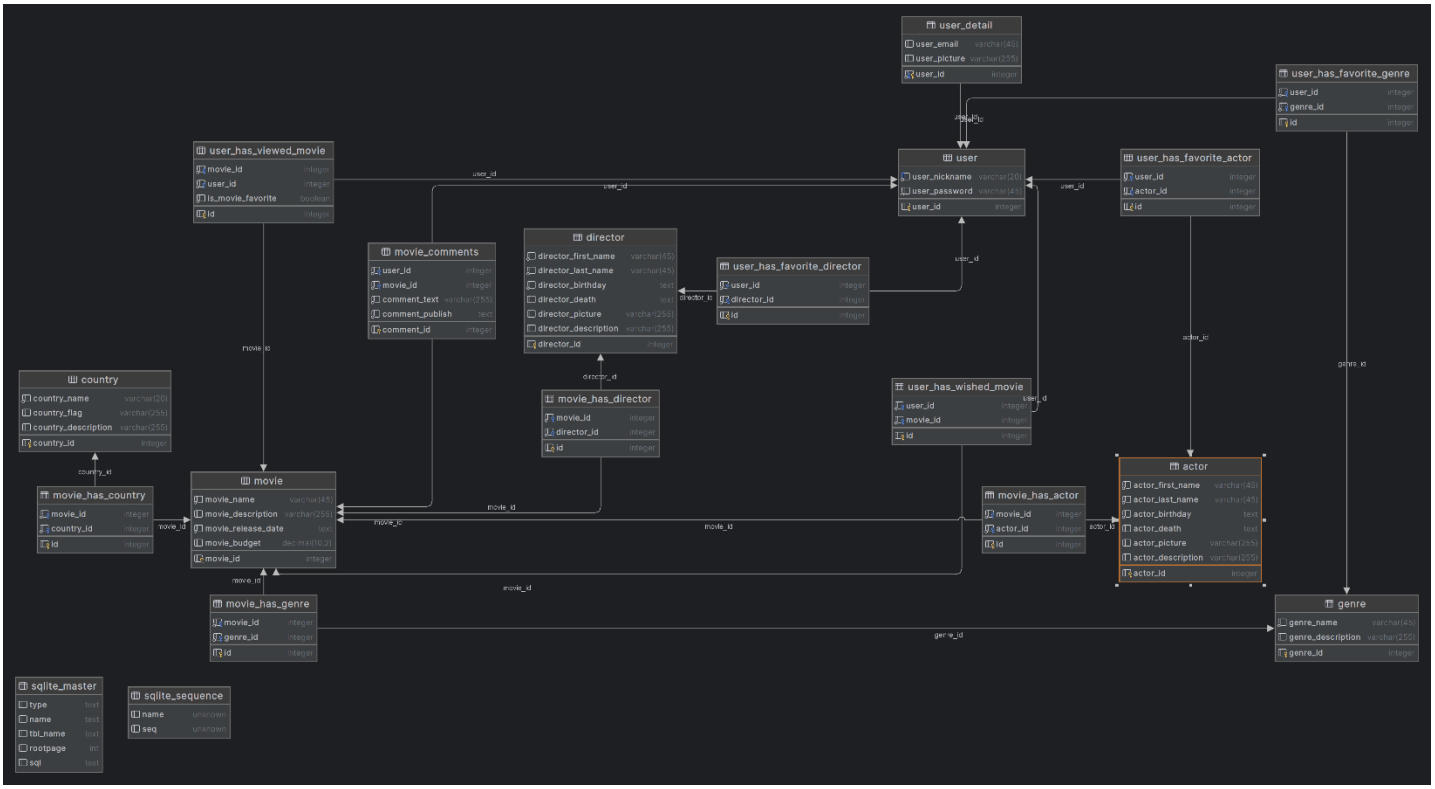
We consider our database to be the 3rd form because it divides one table into several auxiliary. There are no transitive functional dependencies, and hence our table is in 3NF.

Evidences:

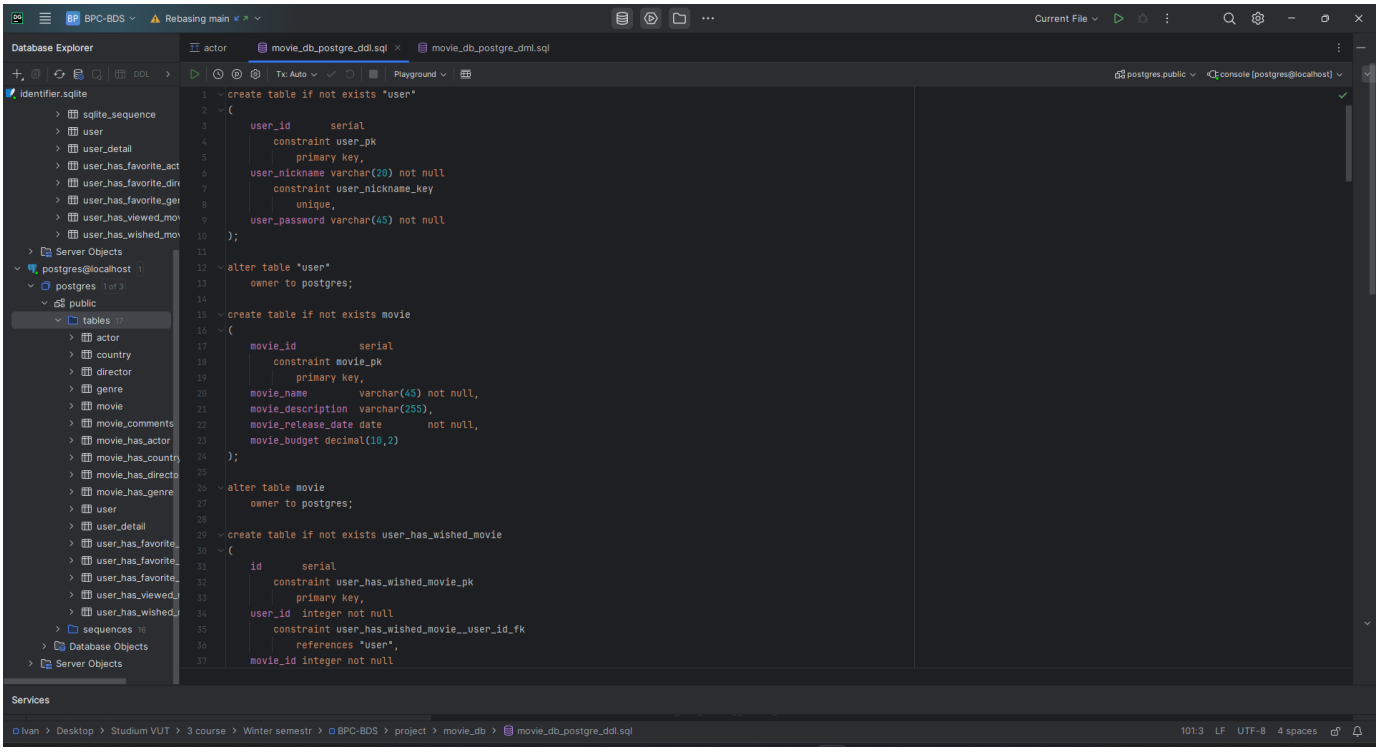
DataGrip PostgreSQL ERD-Diagram



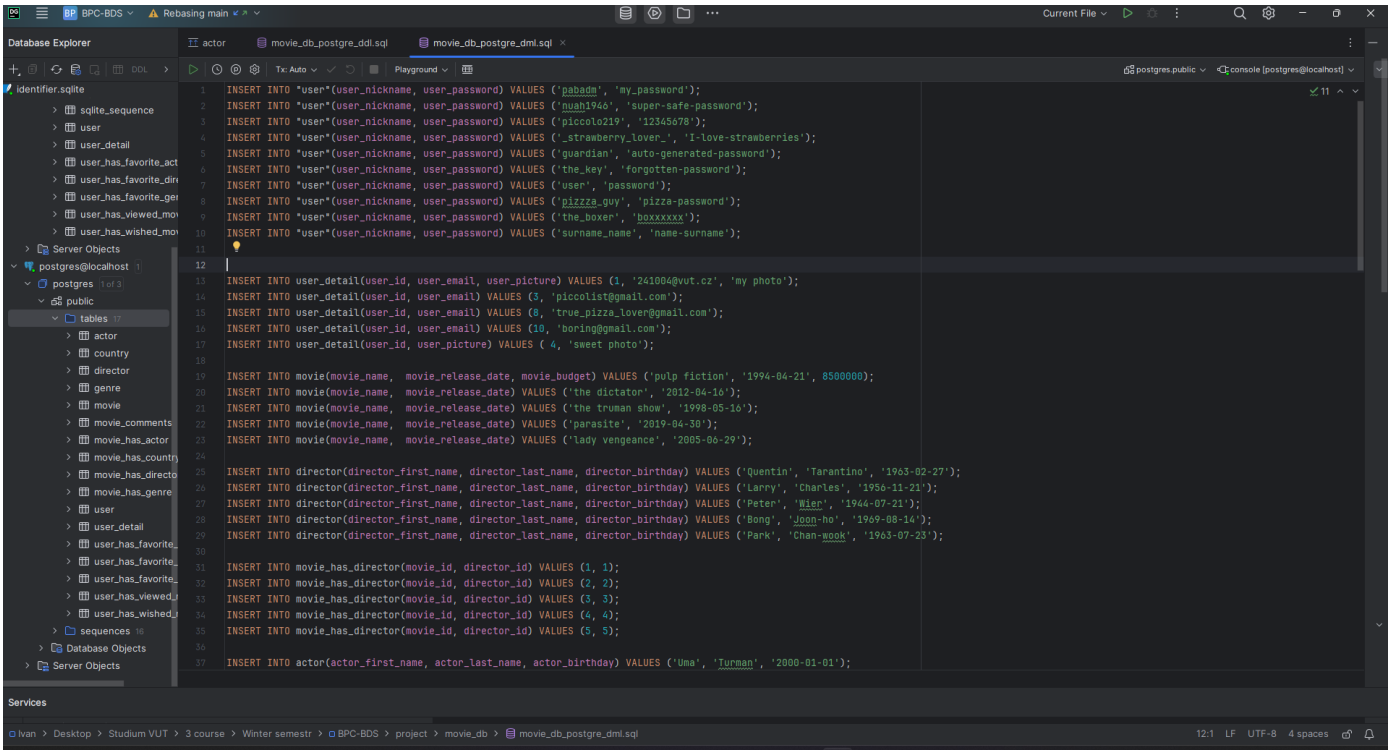
DataGrip PostgreSQL ERD-Diagram



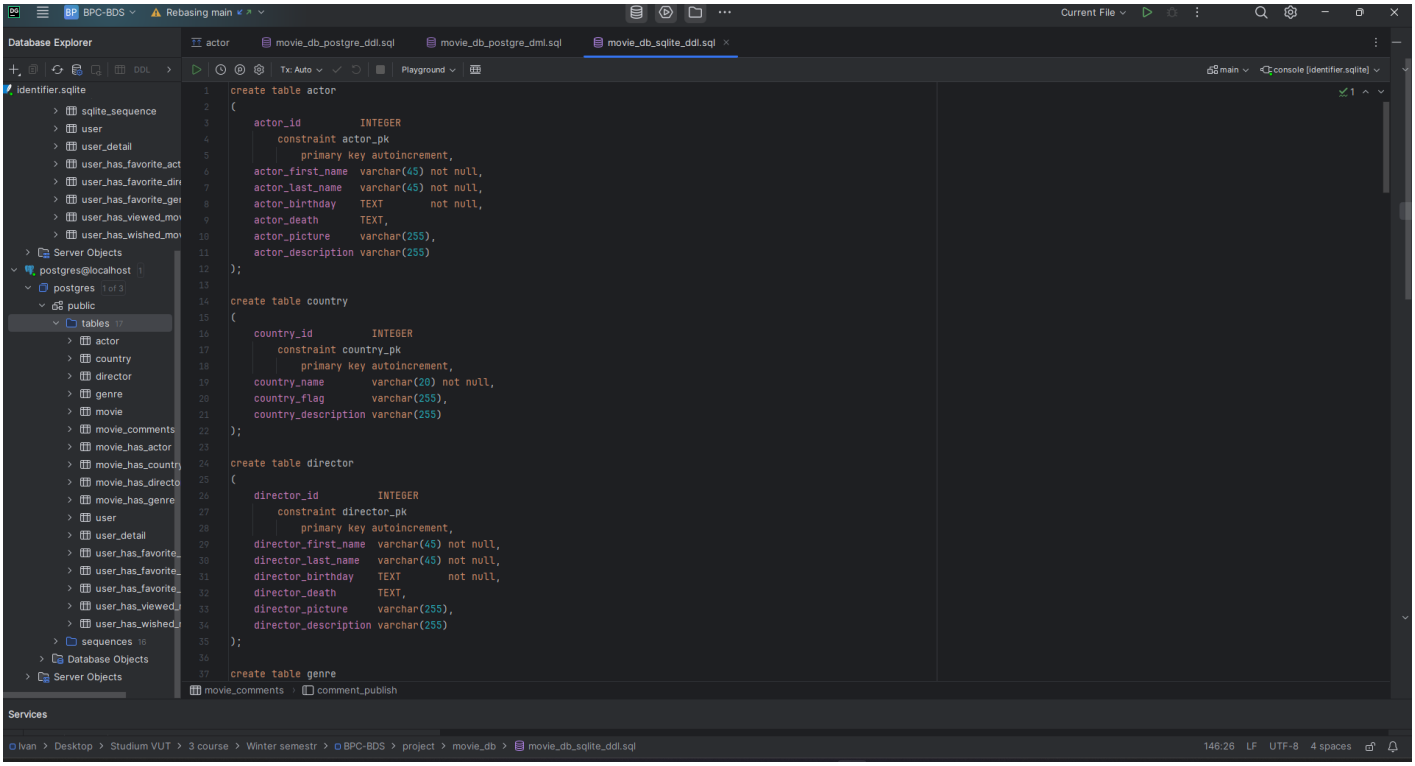
DataGrip screenshots:



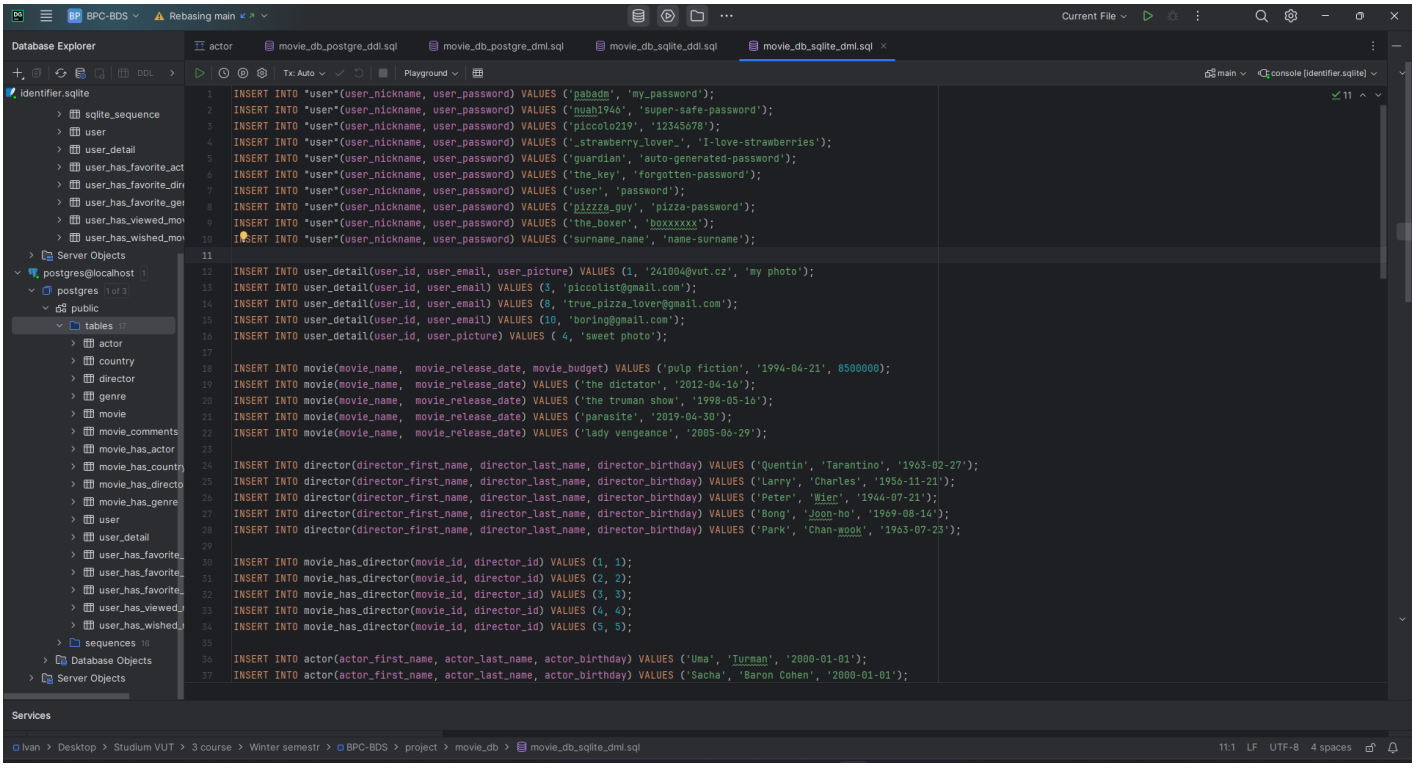
```
create table if not exists "user"
(
    user_id       serial
        constraint user_pk
            primary key,
    user_nickname varchar(28) not null
        constraint user_nickname_key
            unique,
    user_password varchar(45) not null
);

alter table "user"
    owner to postgres;

create table if not exists movie
(
    movie_id           serial
        constraint movie_pk
            primary key,
    movie_name         varchar(45) not null,
    movie_description  varchar(255),
    movie_release_date date        not null,
    movie_budget decimal(10,2)
);

alter table movie
    owner to postgres;

create table if not exists user_has_wished_movie
(
    id       serial
        constraint user_has_wished_movie_pk
            primary key,
    user_id  integer not null
        constraint user_has_wished_movie__user_id_fk
            references "user",
    movie_id integer not null
```



```
INSERT INTO "user"(user_nickname, user_password) VALUES ('pabadm', 'my_password');
INSERT INTO "user"(user_nickname, user_password) VALUES ('nuah1946', 'super-safe-password');
INSERT INTO "user"(user_nickname, user_password) VALUES ('piccolo219', '12345678');
INSERT INTO "user"(user_nickname, user_password) VALUES ('_strawberry_lover_', 'I-love-strawberries');
INSERT INTO "user"(user_nickname, user_password) VALUES ('guardian', 'auto-generated-password');
INSERT INTO "user"(user_nickname, user_password) VALUES ('the_key', 'forgotten-password');
INSERT INTO "user"(user_nickname, user_password) VALUES ('user', 'password');
INSERT INTO "user"(user_nickname, user_password) VALUES ('pizzza_guy', 'pizza-password');
INSERT INTO "user"(user_nickname, user_password) VALUES ('the_boxer', 'boxxxxxx');
INSERT INTO "user"(user_nickname, user_password) VALUES ('surname_name', 'name-surname');

INSERT INTO user_detail(user_id, user_email, user_picture) VALUES (1, '241004@vut.cz', 'my photo');
INSERT INTO user_detail(user_id, user_email) VALUES (3, 'piccolist@gmail.com');
INSERT INTO user_detail(user_id, user_email) VALUES (8, 'true_pizza_lover@gmail.com');
INSERT INTO user_detail(user_id, user_email) VALUES (10, 'boring@gmail.com');
INSERT INTO user_detail(user_id, user_picture) VALUES ( 4, 'sweet photo');

INSERT INTO movie(movie_name,  movie_release_date, movie_budget) VALUES ('pulp fiction', '1994-04-21', 8500000);
INSERT INTO movie(movie_name,  movie_release_date) VALUES ('the dictator', '2012-04-16');
INSERT INTO movie(movie_name,  movie_release_date) VALUES ('the truman show', '1998-05-16');
INSERT INTO movie(movie_name,  movie_release_date) VALUES ('parasite', '2019-04-30');
INSERT INTO movie(movie_name,  movie_release_date) VALUES ('lady vengeance', '2005-06-29');

INSERT INTO director(director_first_name, director_last_name, director_birthday) VALUES ('Quentin', 'Tarantino', '1963-02-27');
INSERT INTO director(director_first_name, director_last_name, director_birthday) VALUES ('Larry', 'Charles', '1956-11-21');
INSERT INTO director(director_first_name, director_last_name, director_birthday) VALUES ('Peter', 'Wier', '1944-07-21');
INSERT INTO director(director_first_name, director_last_name, director_birthday) VALUES ('Bong', 'Joon-ho', '1969-08-14');
INSERT INTO director(director_first_name, director_last_name, director_birthday) VALUES ('Park', 'Chan-wook', '1963-07-23');

INSERT INTO movie_has_director(movie_id, director_id) VALUES (1, 1);
INSERT INTO movie_has_director(movie_id, director_id) VALUES (2, 2);
INSERT INTO movie_has_director(movie_id, director_id) VALUES (3, 3);
INSERT INTO movie_has_director(movie_id, director_id) VALUES (4, 4);
INSERT INTO movie_has_director(movie_id, director_id) VALUES (5, 5);

INSERT INTO actor(actor_first_name, actor_last_name, actor_birthday) VALUES ('Uma', 'Turman', '2000-01-01');
```



```sql
create table actor
(
    actor_id          INTEGER
        constraint actor_pk
            primary key autoincrement,
    actor_first_name  varchar(45) not null,
    actor_last_name   varchar(45) not null,
    actor_birthday    TEXT        not null,
    actor_death       TEXT,
    actor_picture     varchar(255),
    actor_description varchar(255)
);

create table country
(
    country_id          INTEGER
        constraint country_pk
            primary key autoincrement,
    country_name        varchar(20) not null,
    country_flag        varchar(255),
    country_description varchar(255)
);

create table director
(
    director_id          INTEGER
        constraint director_pk
            primary key autoincrement,
    director_first_name  varchar(45) not null,
    director_last_name   varchar(45) not null,
    director_birthday    TEXT        not null,
    director_death       TEXT,
    director_picture     varchar(255),
    director_description varchar(255)
);

create table genre
```



```sql
INSERT INTO "user"(user_nickname, user_password) VALUES ('pabadm', 'my_password');
INSERT INTO "user"(user_nickname, user_password) VALUES ('nuan1946', 'super-safe-password');
INSERT INTO "user"(user_nickname, user_password) VALUES ('piccolo219', '12345678');
INSERT INTO "user"(user_nickname, user_password) VALUES ('_strawberry_lover_', 'I-love-strawberries');
INSERT INTO "user"(user_nickname, user_password) VALUES ('guardian', 'auto-generated-password');
INSERT INTO "user"(user_nickname, user_password) VALUES ('the_key', 'forgotten-password');
INSERT INTO "user"(user_nickname, user_password) VALUES ('user', 'password');
INSERT INTO "user"(user_nickname, user_password) VALUES ('pizzza_guy', 'pizza-password');
INSERT INTO "user"(user_nickname, user_password) VALUES ('the_boxer', 'boxxxxxx');
INSERT INTO "user"(user_nickname, user_password) VALUES ('surname_name', 'name-surname');

INSERT INTO user_detail(user_id, user_email, user_picture) VALUES (1, '241004@vut.cz', 'my photo');
INSERT INTO user_detail(user_id, user_email) VALUES (3, 'piccolist@gmail.com');
INSERT INTO user_detail(user_id, user_email) VALUES (8, 'true_pizza_lover@gmail.com');
INSERT INTO user_detail(user_id, user_email) VALUES (10, 'boring@gmail.com');
INSERT INTO user_detail(user_id, user_picture) VALUES ( 4, 'sweet photo');

INSERT INTO movie(movie_name,  movie_release_date, movie_budget) VALUES ('pulp fiction', '1994-04-21', 8500000);
INSERT INTO movie(movie_name,  movie_release_date) VALUES ('the dictator', '2012-04-16');
INSERT INTO movie(movie_name,  movie_release_date) VALUES ('the truman show', '1998-05-16');
INSERT INTO movie(movie_name,  movie_release_date) VALUES ('parasite', '2019-04-30');
INSERT INTO movie(movie_name,  movie_release_date) VALUES ('lady vengeance', '2005-06-29');

INSERT INTO director(director_first_name, director_last_name, director_birthday) VALUES ('Quentin', 'Tarantino', '1963-02-27');
INSERT INTO director(director_first_name, director_last_name, director_birthday) VALUES ('Larry', 'Charles', '1956-11-21');
INSERT INTO director(director_first_name, director_last_name, director_birthday) VALUES ('Peter', 'Wier', '1944-07-21');
INSERT INTO director(director_first_name, director_last_name, director_birthday) VALUES ('Bong', 'Joon-ho', '1969-08-14');
INSERT INTO director(director_first_name, director_last_name, director_birthday) VALUES ('Park', 'Chan-wook', '1963-07-23');

INSERT INTO movie_has_director(movie_id, director_id) VALUES (1, 1);
INSERT INTO movie_has_director(movie_id, director_id) VALUES (2, 2);
INSERT INTO movie_has_director(movie_id, director_id) VALUES (3, 3);
INSERT INTO movie_has_director(movie_id, director_id) VALUES (4, 4);
INSERT INTO movie_has_director(movie_id, director_id) VALUES (5, 5);

INSERT INTO actor(actor_first_name, actor_last_name, actor_birthday) VALUES ('Uma', 'Turman', '2000-01-01');
INSERT INTO actor(actor_first_name, actor_last_name, actor_birthday) VALUES ('Sacha', 'Baron Cohen', '2000-01-01');
```