Описание предметной области «кинофильмы»

Пусть требуется разработать информационную систему для автоматизации учета и анализа кинопродукции. Система должна предусматривать режимы ведения системного каталога, отражающего перечень жанров фильмов, по которым имеются киноленты в базе данных. Внутри базы данных жанры фильмов в системном каталоге могут иметь уникальный внутренний номер и полное наименование. Каждый фильм может содержать сведения из нескольких жанров. Каждый фильм в базе данных может присутствовать в нескольких экземплярах. Каждый фильм, хранящийся в базе данных, характеризуется следующими параметрами:

- уникальный ID;
- название;
- киностудия;
- год выпуска;
- страна;
- продолжительность;
- бюджет;
- количество жанров фильма.

Фильмы могут иметь одинаковые названия, но они различаются по уникальному ID.

В базе данных ведется учет пользователей.

На каждого пользователя в базу данных записываются следующие сведения:

- ФИО;
- номер телефона.

Каждому пользователю присваивается уникальный номер учетной записи.

Каждый пользователь может оценивать несколько фильмов. Пользователь не должен одновременно оценивать более одного экземпляра фильма с одним названием.

Каждый фильм в базе данных может присутствовать в нескольких экземплярах. Каждый экземпляр имеет следующие характеристики:

- уникальный идентификатор экземпляра;
- информация о фильме, которая совпадает с уникальным ID из описания фильма;
- место хранения в базе.

В случае оценки экземпляра фильма пользователем в базе данных хранится специальная запись, в которой должны быть записаны следующие сведения:

- номер учетной записи пользователя, который оценил фильм;
- дата оценки;
- цифра оценки.

Предусмотреть следующие ограничения в информационной системе:

1. Фильм может не иметь одного или нескольких актеров.
2. В базе данных могут быть зарегистрированы пользователи младше 18 лет.
3. В базе присутствуют фильмы, выпущенные с 1900 по текущий год.
4. Каждый пользователь может оценивать не более 10 фильмов одновременно.
5. Каждый пользователь при регистрации в системе должен указать контактную информацию для связи.
6. Каждый жанр может содержать ссылки на множество фильмов, но каждый фильм может относиться к разным жанрам.

Следующие группы пользователей должны работать с информационной системой:

- администраторы базы данных;
- пользователи системы;
- аналитики.

При работе с системой администраторы базы данных должны иметь возможность решать следующие задачи:

1. Добавлять новые фильмы и регистрировать их в базе данных.

2. Относить фильмы к одному или нескольким жанрам.

3. Вести учет актеров и их ролей в фильмах.

4. Проводить оценку фильмов, то есть назначение новых идентификаторов новым фильмам, и помещая их в соответствующие разделы базы данных, записывать место хранения каждого экземпляра.

5. Проводить дополнительную каталогизацию, если поступило несколько экземпляров фильма, который уже есть в базе данных, при этом информация о фильме в реестре не вносится, а каждому новому экземпляру присваивается новый идентификатор и для него определяется место на полке базы данных.

6. Вести учет оценок фильмов пользователями, при этом предполагается два режима работы: оценка фильма пользователем и прием от него отзывов о фильме обратно в базу данных. При оценке фильма фиксируется, когда и какой экземпляр фильма был оценен данным пользователем и к какому сроку пользователь должен вернуть этот экземпляр фильма. При оценке фильма на наличие свободного экземпляра его конкретный номер может определяться по заданному уникальному ID фильма или инвентарный номер может быть известен заранее. Не требуется вести «историю» оценки фильма, то есть требуется отражать только текущее состояние базы данных. При приеме оценки, возвращаемой пользователем, проверяется соответствие возвращаемого инвентарного номера фильма выданному инвентарному номеру, и он ставится на свое старое место в базе данных.

7. Проводить удаление утерянных пользователем фильмов по специальному акту удаления или замены, подписанному администратором базы данных.

8. Проводить закрытие учетной записи пользователя, то есть удаление данных о нем, если пользователь хочет выйти из базы данных и не является должником, то есть за ним не числится ни одного экземпляра фильма.

Пользователи системы должны иметь возможность решать следующие задачи:

1. Просматривать системный каталог, то есть перечень всех жанров фильмов, фильмы по которым есть в базе данных.

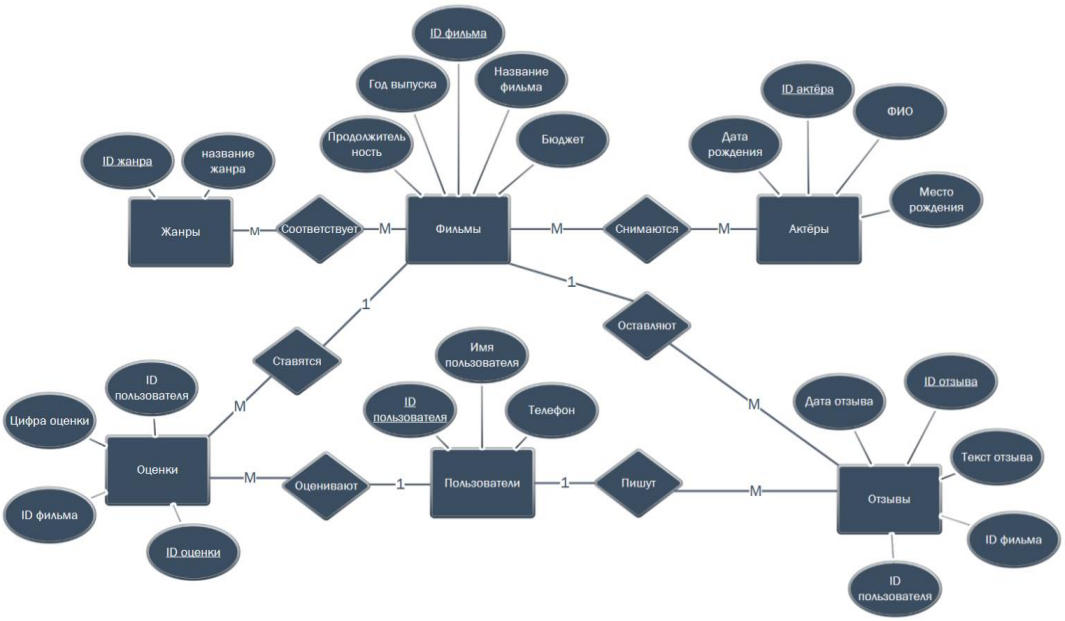
2. По выбранному жанру получить общий перечень фильмов, которые участвуют в базе данных.

3. Для выбранного фильма получить инвентарный номер свободного экземпляра фильма и сообщение о том, что свободных экземпляров фильма нет. В случае отсутствия свободных экземпляров фильма пользователь должен иметь возможность узнать дату ближайшего предполагаемого возврата экземпляра данного фильма. Пользователь не может узнать информацию о том, у кого в настоящий момент экземпляр данного фильма находится на руках (в целях обеспечения конфиденциальности держателей требуемого фильма).

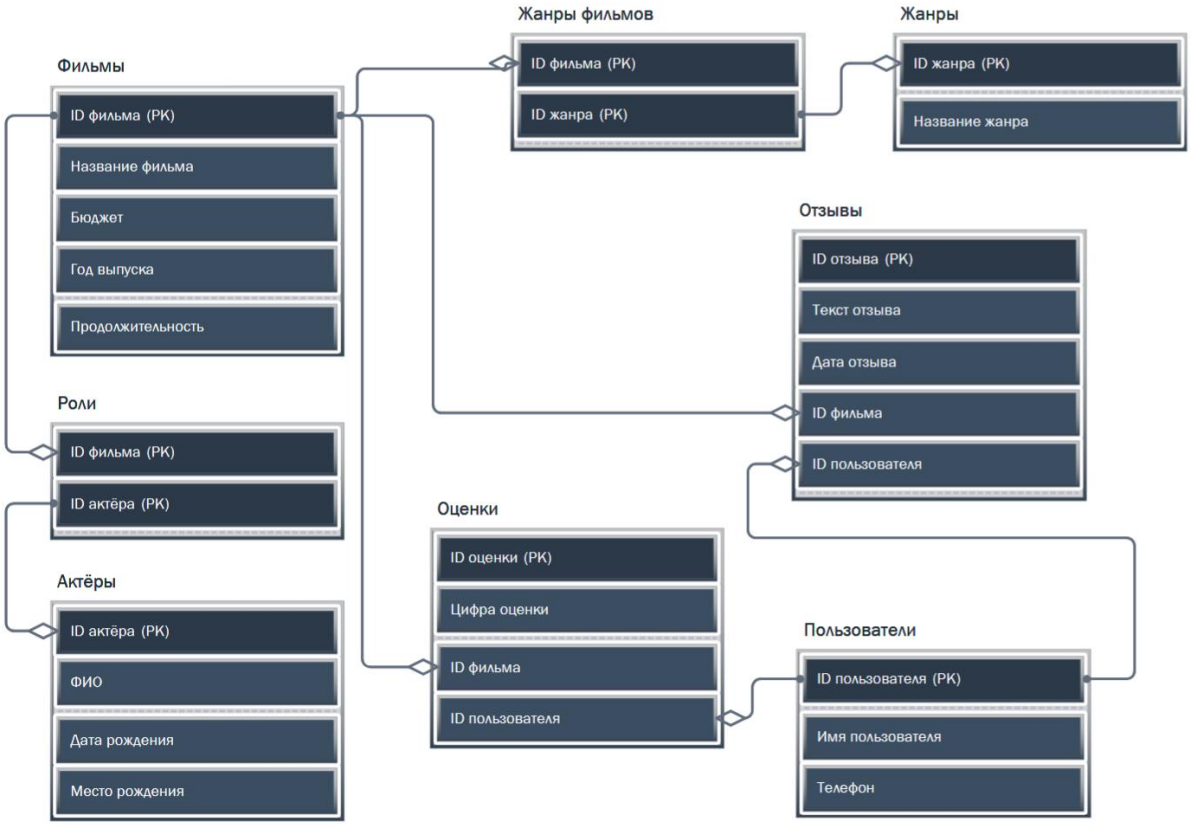
4. По выбранному актеру получить список фильмов, в которых он участвует в базе данных.

Аналитики базы данных должны иметь возможность получать сведения о пользователях-посетителях базы данных, которые не вернули вовремя взятые фильмы; сведения о фильмах, которые не являются популярными, т. е. все экземпляры которых находятся на руках у пользователей; сведения о стоимости конкретного фильма, для того чтобы установить возможность возмещения стоимости утерянного фильма или возможность замены его другой кинолентой; сведения о наиболее популярных фильмах, то есть таких, все экземпляры которых находятся на руках у пользователей.

ER-диаграмма:



Блок-схема:



SQL-файл:

```
create database if not exists db_films;
use db_films;

create table if not exists movies(
        id_movie int not null auto_increment,
    movie_name varchar(100),
    movie_budget int not null default 0,
    movie_release_date date,
    movie_duration int not null default 0,
    primary key (id_movie)
);

create table if not exists actors(
        id_actor int not null auto_increment,
```

```
    actor_name varchar(100),
    actor_birthdate date,
    actor_birthplace varchar(100),
    primary key (id_actor)
);

create table if not exists roles(
        id_movie int not null,
    id_actor int not null,
    primary key (id_movie, id_actor),
    constraint fk_movies_actors
            foreign key (id_movie)
            references movies(id_movie),
        constraint fk_actors_movies
            foreign key (id_actor)
       references actors(id_actor)
);

create table if not exists genres(
        id_genre int not null auto_increment,
    genre_name varchar(50),
    primary key(id_genre)
);

create table if not exists genres_of_movies(
        id_movie int not null,
    id_genre int not null,
    primary key (id_movie, id_genre),
    constraint fk_movie_genres
            foreign key (id_movie)
       references movies(id_movie),
```

```
        constraint fk_genre_movies
                foreign key (id_genre)
        references genres(id_genre)
);

create table if not exists users(
        id_user int not null auto_increment,
    user_name varchar(50),
    user_phone int,
    primary key(id_user)
);

create table if not exists reviews(
        id_review int not null auto_increment,
    review_text varchar(1000),
    review_date date,
    id_movie int not null,
    id_user int not null,
    primary key(id_review),
    constraint fk_movie_users
                foreign key(id_movie)
        references movies(id_movie),
        constraint fk_user_movies
                foreign key(id_user)
        references users(id_user)
);

create table if not exists rates(
        id_rate int not null auto_increment,
    rate_number int not null default 10,
    id_movie int not null,
```

```
    id_user int not null,
    primary key(id_rate),
    constraint fk_movie_user
                foreign key(id_movie)
        references movies(id_movie),
         constraint fk_user_movie
                foreign key(id_user)
        references users(id_user)
);
```

Полученная таблица из этого кода: