

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Факультет «Информатика и вычислительная техника»

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

|  |  |  |
| --- | --- | --- |
|  | | |
| Зав. кафедрой | | «ПОВТиАС» |
| \_\_\_\_\_\_\_\_\_\_\_\_ | | В.В. Долгов |
| (подпись) | | (И.О.Ф.) |
| «\_\_\_» | \_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г. | |

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовой работе по дисциплине (модулю) базы данных

(наименование учебной дисциплины (модуля))

Автор проекта И.А. Андросов

подпись, дата И.О.Ф.

Обозначение отчета КР.420000.000 Группа ВПР31

Направление 09.03.04 «Программная инженерия»

Профиль Программное обеспечение вычислительной техники и автоматизированных систем

Руководитель проекта: М.В. Гранков

подпись, дата И.О.Ф

Оценка

дата подпись

Ростов-на-Дону

2022



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Факультет «Информатика и вычислительная техника»

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

**ЗАДАНИЕ**

к курсовой работе по дисциплине (модулю) базы данных

Обучающийся Андросов Илья Алексеевич

Обозначение проекта КР.420000.000 Группа ВПР31

Срок представления проекта (работы) к защите \_\_\_\_\_ « » 2022 г.

Содержание индивидуального задания:

Разработать для интернет-магазина кинофильмов систему базы данных и комплекс программ по управлению.

Руководитель проекта М.В. Гранков

подпись, дата И.О.Ф

Задание принял к исполнению И.А. Андросов

подпись, дата И.О.Ф

**Содержание**

[1 Внешняя модель 6](#_Toc150367660)

[1.1 Описание предметной области конечными пользователями 6](#_Toc150367661)

[1.2 Описание ограничений (ключей, зависимостей, связей и пр.), имеющих место в предметной области; 8](#_Toc150367662)

[1.3 Описание способов, форм обработки и представление сведений о хранимой в базе данных информации (не менее 3-х запросов) 9](#_Toc150367663)

[1.4 Дополнительные требования 10](#_Toc150367664)

[1.5 Модель предметной области в виде схемы “Объекты связи” 10](#_Toc150367665)

[2 Логическая модель 11](#_Toc150367666)

[2.1 Схемы базовых отношений 11](#_Toc150367667)

[2.2 Домены атрибутов отношений 11](#_Toc150367668)

[2.3 Множество функциональных зависимостей 12](#_Toc150367669)

[2.4 Неприводимое множество функциональных зависимостей 13](#_Toc150367670)

[2.5 Множество супер-ключей 16](#_Toc150367671)

[2.6 Множество потенциальных ключей 17](#_Toc150367672)

[2.7 Выбор первичных ключей 19](#_Toc150367673)

[2.8 Обосновать выбор внешних ключей 20](#_Toc150367674)

[2.9 Нормализация отношений предметной области 21](#_Toc150367675)

[2.10 Предикаты для проверки целостности базы данных 22](#_Toc150367676)

[2.11 Описание требуемых виртуальных отношений 24](#_Toc150367677)

[2.12 Реляционные выражения 25](#_Toc150367678)

[3 Физическая модель 26](#_Toc150367679)

**Постановка задачи**

Разработать систему базы данных для цифрового магазина кинофильмов.

Для достижения поставленной цели необходимо решить следующие задачи:

* составить внешние представления конечных пользователей, описание объектов предметной области, их атрибутов и связей между объектами, функциональных зависимостей, способов, форм обработки и представления сведений о хранимой в БД информации;
* составить логическую модель предметной области с использованием реляционной модели, разработать схемы базовых отношений, описать домены атрибутов всех отношений, построить множество функциональных зависимостей, построить неприводимое множество функциональных зависимостей, построить множество супер – ключей, по множеству супер - ключей построить множество потенциальных ключей, доказать их потенциальность, выбрать первичные ключи, обосновать выбор, провести нормализацию отношений до уровня не ниже 3НФ (на каждом этапе нормализации обосновывать выбор первичных ключей для каждого отношения), разработать предикат для проверки целостности базы данных, описать требуемые виртуальные отношения, разработать реляционные выражения для реализации запросов;
* составить физическую модель, которая должна содержать тексты программ для создания и поддержки всех используемых отношений, визуализацию запросов в виде экранных форм и документов (при реализации внутренней модели особое внимание уделить выполнению следующих правил Кодда: физическая независимость данных; логическая независимость данных; дистрибутивная независимость), разрабатываемое программное обеспечение должно реализовывать сетевой, многопользовательский способ доступа, содержать инструкции по инсталляции программного обеспечения и базовых отношений, требуется обеспечить простое изменение основных констант, характеризующих предметную область;
* разработать сопроводительную документацию.

# 1 Внешняя модель

В данном разделе рассматриваются внешние представления пользователей по управлению интернет-магазином кинофильмов, важные аспекты предметной области, функциональные зависимости.

Существуют три уровня абстракции в архитектуре ANSI/X3/SPARC базы данных:

* Уровень представлений (внешняя модель базы данных).
* Концептуальный уровень - концептуальная база данных (концептуальная (логическая) модель базы данных).
* Физический уровень - физическая база данных (физическая модель базы данных).

## 1.1 Описание предметной области конечными пользователями

В рамках выполнения курсовой работы выбрана предметная область — приложение магазина кинофильмов. Из перспективы конечных пользователей данной области выделяются следующие роли: администратор и покупатель. Давайте рассмотрим предметную область с точки зрения этих пользователей.

**Администратор:**

Администратор осуществляет добавление новых фильмов, предоставляя информацию о названии, описании, киностудии, режиссере, цене, скидке, жанре, размере и коде фильма. Также имеется возможность редактирования уже существующих фильмов, где изменяются параметры, такие как название, описание, режиссер, киностудия, цена, скидка, жанр, размер и код фильма. При необходимости, администратор добавляет киностудии или режиссеров, указывая наименование, адрес, страну и код киностудии или режиссера. Также имеется функционал назначения пользователя администратором, а также изменения его имени, фамилии, страны, баланса и пароля. Администратор может удалять пользователей, киностудии, режиссеров, фильмы, жанры и страны. Возможна также добавление нового жанра или страны. Иногда необходимо удалить все купленные фильмы у пользователя.

Ключевыми атрибутами администратора являются логин, пароль, имя, фамилия, тип пользователя.

В рассматриваемой предметной области связь между администратором, фильмами, покупателями, киностудиями и режиссерами представлена в виде ER-диаграммы(рис.1.1).

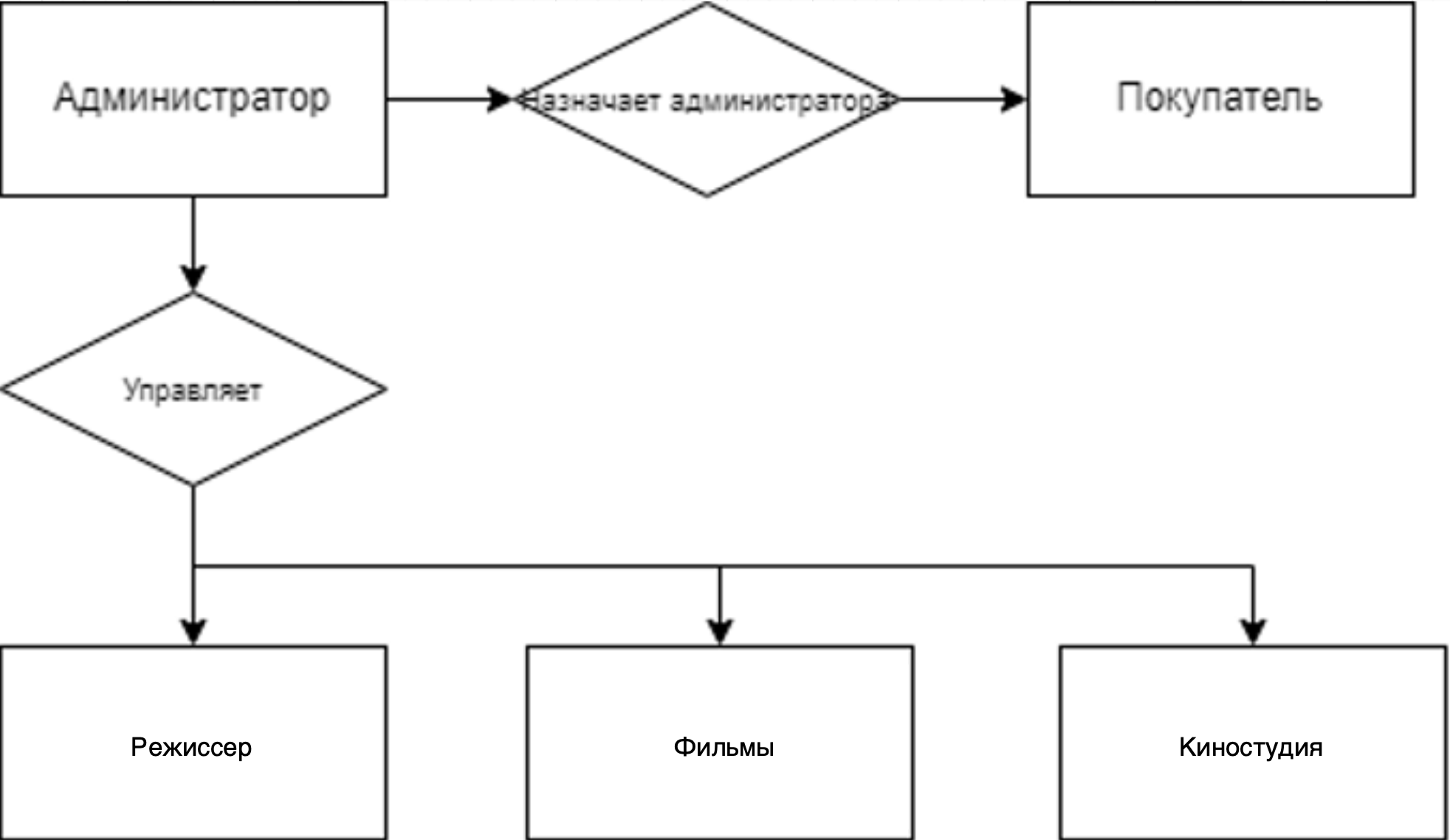


Рисунок 1.1 – связь между администратором и другими объектами

**Покупатель:**

В процессе использования интернет-магазина покупатель имеет возможность ознакомиться с ассортиментом фильмов, сортировать их по жанрам. При выборе конкретного фильма, покупатель видит его основные характеристики, такие как название, описание, киностудия, режиссер, цена, скидка, жанр и размер. Выбранный фильм можно приобрести, и он появляется в библиотеке покупателя. В библиотеке отображаются все купленные фильмы, с возможностью получения более подробной информации о киностудии и режиссере. Также в библиотеке можно скачивать фильмы. Раздел профиля покупателя содержит информацию о логине, имени, фамилии, стране и балансе. Также доступна функция изменения пароля, страны, удаления всех фильмов из библиотеки и пополнения баланса.

Ключевыми атрибутами покупателя являются логин, пароль, имя, фамилия, баланс и тип пользователя.

Связь между покупателем, фильмами и профилем представлена на следующей ER-диаграмме (рис. 1.2).

Рисунок 1.2 – Связь между покупателем и фильмами

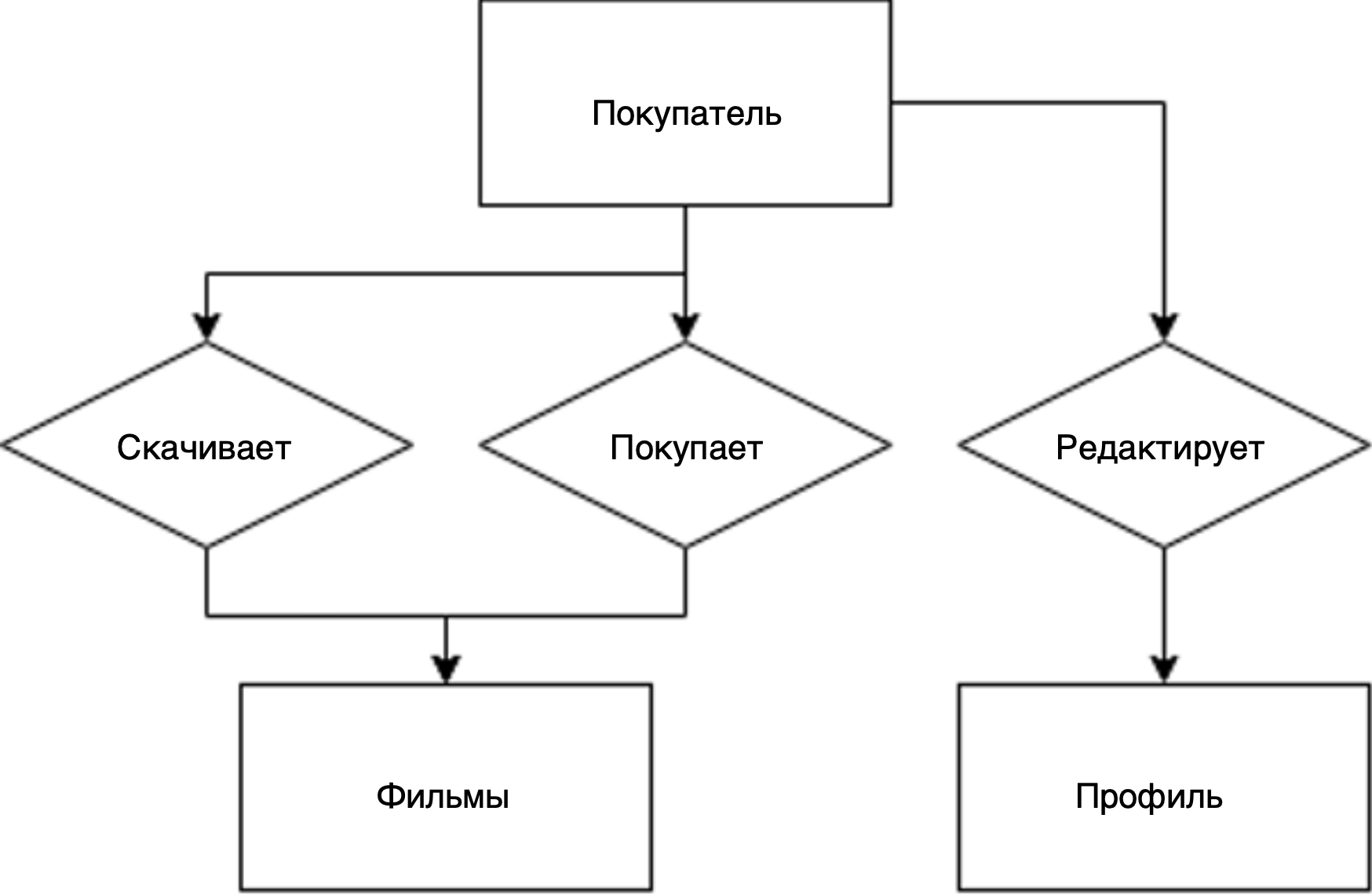


Рисунок 1.2 – связь между покупателем и другими объектами

## 1.2 Описание ограничений (ключей, зависимостей, связей и пр.), имеющих место в предметной области;

Рассмотрим список ограничений, полученных на этапе построения внешней модели:

1. Ни один пользователь не может владеть одним фильмом дважды.
2. Ни один фильм не может иметь двух и более режиссеров или киностудию.
3. Ни один фильм не может иметь больше одного жанра.
4. Ни один пользователь не может иметь больше одной страны.
5. Ни один режиссер или киностудия не может иметь больше одной страны.
6. Ни у одного пользователя не должен совпадать логин.

## 1.3 Описание способов, форм обработки и представление сведений о хранимой в базе данных информации (не менее 3-х запросов)

Рассмотрим отчёты, предоставляемые для руководителей магазина в виде таблиц:

1. Список всех фильмов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название фильма | Цена | Жанр | Режиссер | Киностудия | Размер |

Таблица 1.1 – отчет о всех фильмах

1. Список пользователей

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Логин | Имя | Фамилия | Страна | Баланс |

Таблица 1.2 – отчет о всех пользователях

1. Список фильмов по одному жанру

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название фильма | Цена | Режиссер | Киностудия | Размер |

Таблица 1.3 – отчет о всех фильмах одного жанра

## 1.4 Дополнительные требования

## 1.5 Модель предметной области в виде схемы “Объекты связи”

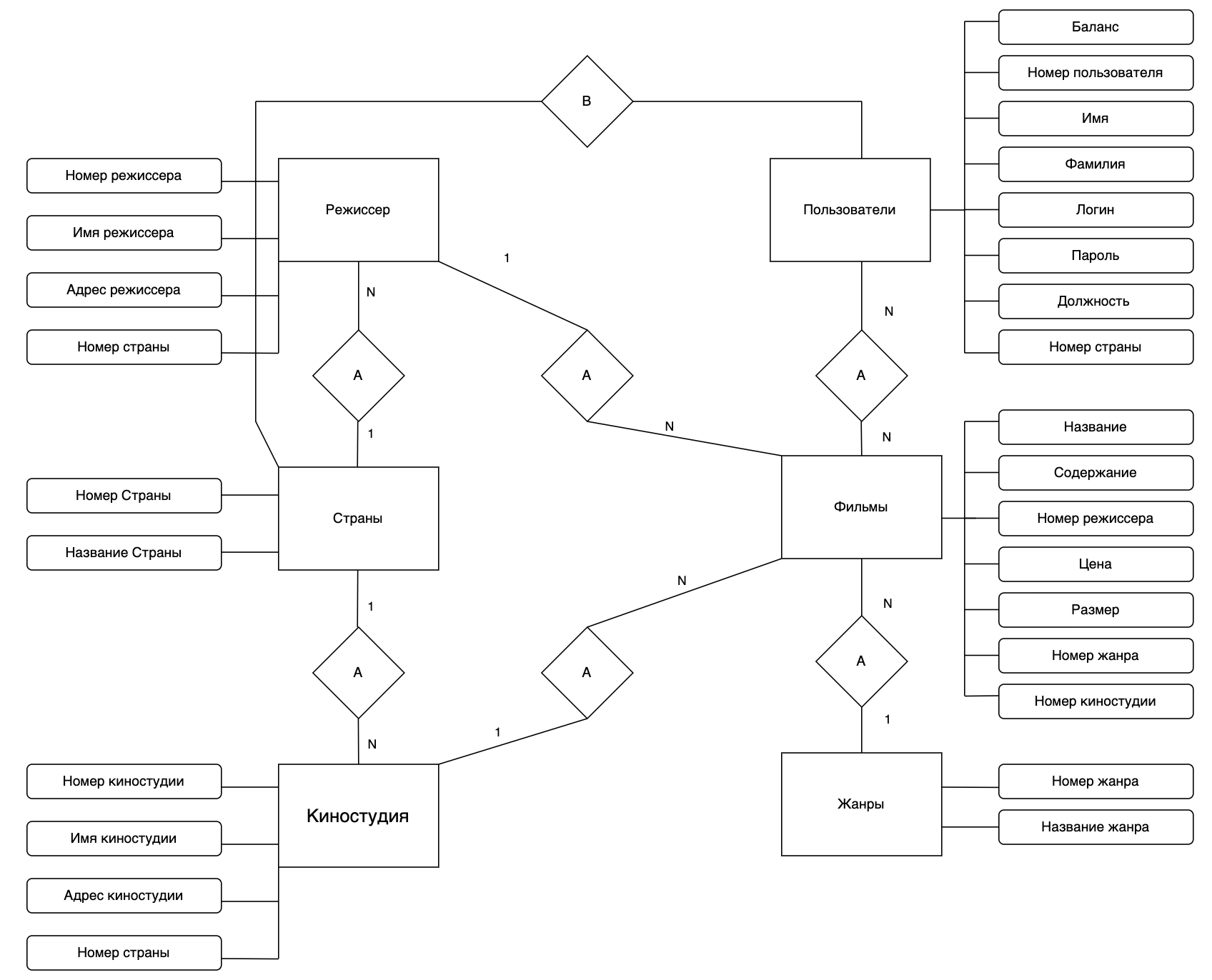
****

Рисунок 1.2 – модель предметной области магазина фильмов в виде схемы “Объекты связи”

# 2 Логическая модель

В данном разделе производится построение множества схем отношений, функциональных зависимостей, супер-ключей, производится нормализация отношений.

## 2.1 Схемы базовых отношений

Пользователь::={ **номер пользователя**, логин, пароль, имя, фамилия, тип пользователя, баланс, страна::={ **номер страны**, название страны}}

Фильмы::={ **номер фильма**, название фильма, описание, цена, скидка, размер, режиссер::={**номер режиссера**, название режиссера, адрес режиссера, страна::={ **номер страны**, название страны }}, киностудия::={ **номер киностудии**, название киностудии, адрес киностудии, страна::={ **номер страны**, название страны }}, жанр::={**номер жанра**, название жанра}

## 2.2 Домены атрибутов отношений

Отношение Пользователи:

* Номер пользователя - числа от 0
* Логин - последовательность символов до 32
* Пароль - последовательность символов до 32
* Имя - последовательность символов до 32
* Фамилия - последовательность символов до 32
* Номер страны - числа от 0
* Тип пользователя - последовательность символов до 10
* Баланс - числа от 0 включительно

Отношение Фильма:

* Номер фильма - числа от 0
* Название фильма - последовательность символов до 32
* Описание - текст
* Номер режиссера - числа от 0
* Номер киностудии - числа от 0
* Цена - числа от 0
* Скидка - числа от 0 включительно
* Номер жанра - числа от 0
* Размер - число от 0

Отношение Страны

* Номер Страны - числа от 0
* Название страны - последовательность символов до 20

Отношение Режиссера

* Номер Режиссера - числа от 0
* Название Режиссера - последовательность символов до 32
* Адрес Режиссера - последовательность символов до 20
* Номер страны - числа от 0

Отношение Киностудии

* Номер Киностудии - числа от 0
* Название Киностудии - последовательность символов до 32
* Адрес Киностудии - последовательность символов до 20
* Номер страны - числа от 0

Отношение Жанры

* Номер жанра - числа от 0
* Название жанра - последовательность символов до 32

## 2.3 Множество функциональных зависимостей

На основании списка утверждений, полученного после этапа построения внешней модели, описанного в главе 1.2, построим множество функциональных зависимостей:

S = {

{номер пользователя, номер фильма}→{ логин, пароль, имя, фамилия, тип пользователя, баланс, номер страны, название фильма, описание, цена, скидка, размер, номер жанра, номер режиссера, номер киностудии},

{номер фильма}→{название фильма, описание, цена, скидка, размер, номер жанра, номер режиссера, номер киностудии},

{номер жанра}→{название жанра},

{номер режиссера}→{название киностудии, адрес киностудии, номер страны},

{номер киностудии}→{название киностудии, адрес киностудии, номер страны},

{номер страны}→{название страны},

{номер пользователя}→{логин, пароль, имя, фамилия, тип пользователя, баланс, номер страны}

}

## 2.4 Неприводимое множество функциональных зависимостей

Множество ФЗ называется неприводимым тогда и только тогда, когда:

* Правая (зависимая) часть каждой ФЗ множества S содержит только один атрибут.
* В левой части каждой ФЗ множества S не может быть опущен ни один атрибут без изменения замыкания S+.
* Ни одна ФЗ в S не может быть опущена из S без изменения S+.

S0 = {

{нп, нф}→{ лгн, прл, имя, фам, тип, блнс, нс, назф, опис, цен, скдк, рзмр, нж, нр, нкс}

{номер пользователя, номер фильма}→{ логин, пароль, имя, фамилия, тип пользователя, баланс, номер страны, название фильма, описание, цена, скидка, размер, номер жанра, номер режиссера, номер киностудии},

{нф}→{назф, опис, цен, скдк, рзмр, нж, нр, нкс}

{номер фильма}→{название фильма, описание, цена, скидка, размер, номер жанра, номер режиссера, номер киностудии},

{нж}→{назж}

{номер жанра}→{название жанра},

{нр}→{назкс, адркс, нс}

{номер режиссера}→{название киностудии, адрес киностудии, номер страны},

{нкс}→{назкс, адркс, нс}

{номер киностудии}→{название киностудии, адрес киностудии, номер страны},

{нс}→{назс}

{номер страны}→{название страны},

{нп}→{лгн, прл, имя, фам, тип, блнс, нс}

{номер пользователя}→{логин, пароль, имя, фамилия, тип пользователя, баланс, номер страны}

}

Раскрываем определяемые множества с помощью правила декомпозиции

S1 = {

{нп,нф}→{лгн},{нп,нф}→{прл},{нп, нф}→{имя},{нп, нф}→{фам},

{нп,нф}→{тип},{нп,нф}→{блнс},{нп,нф}→{нс},{нп,нф}→{назф},

{нп,нф}→{опис},{нп,нф}→{цен},{нп,нф}→{скдк},{нп,нф}→{рзмр},

{нп,нф}→{нж},{нп,нф}→{нр},{нп,нф}→{нкс}

{нф}→{назф},{нф}→{опис},{нф}→{цен},{нф}→{скдк},{нф}→{рзмр}{нф}→{нж},{нф}→{нр},{нф}→{нкс}

{нж}→{назж}

{нр}→{назф},{нр}→{адрр},{нр}→{нс},

{нкс}→{назкс},{нкс}→{адркс},{нкс}→{нс},

{нс}→{назс},

{нп}→{лгн},{нп}→{прл},{нп}→{имя},{нп}→{фам},{нп}→{тип},{нп}→{блнс},{нп}→{нс}

}

Удаляем повторяющиеся зависимости по правилу рефлексивности (определяющие {нп}, {нп, ни})

S2 = {

{нф}→{назф},{нф}→{опис},{нф}→{цен},{нф}→{скдк},{нф}→{рзмр}{нф}→{нж},{нф}→{нр},{нф}→{нкс}

{нж}→{назж}

{нр}→{назф},{нр}→{адрр},{нр}→{нс},

{нкс}→{назкс},{нкс}→{адркс},{нкс}→{нс},

{нс}→{назс},

{нп}→{лгн},{нп}→{прл},{нп}→{имя},{нп}→{фам},{нп}→{тип},{нп}→{блнс},{нп}→{нс}

}

Дальнейшее преобразование множества функциональных зависимостей без изменения замыкания множества функциональной зависимости невозможно, S2 является неприводимым множеством.

## 2.5 Множество супер-ключей

*Суперключом* отношения R называется множество атрибутов отношения R, которое содержит как подмножество хотя бы один потенциальный ключ (т.е. ключ, который обладает свойствами: 1 – уникальности, 2 – не избыточности).

Т.о. суперключи – это такие подмножества Кi множества атрибутов отношения R, что функциональная зависимость:

 , где А= { a1 , a2 ,… an} – множество атрибутов R, т.е. Кi → A.

Определим супер-ключ для отношения Купленные фильмы

1. Предположим, что атрибуты номер пользователя и номер фильма образуют потенциальный ключ
2. Проверим потенциальность ключа {нп, нф} в п.2.6
3. Добавим к этому потенциальному ключу атрибут дата покупки.
4. Ключ {нп, нф, дата} обладает свойством уникальности, т.к. с одного номера пользователя к одному фильму может быть записана только одна дата покупки.

Так же ключ {нп, нф, дата} обладает свойством избыточности, т.к. {нп, нф} → {дата}.

1. На основании описанных выше множеств, построим супер-ключ КФИЛ = {{нп, нф}, {нп, нф, дата}} – супер-ключ отношения Купленные фильмы.

## 2.6 Множество потенциальных ключей

Пусть R – некоторая переменная отношения, тогда потенциальный ключ K для R это подмножество атрибутов R, всегда обладающее следующими свойствами:

* Свойство уникальности: нет двух различных кортежей в текущем значении переменной R с одинаковыми значениями K.
* Свойство не избыточности: никакое из подмножеств K не обладает свойством уникальности
* На практике в отношениях чаще всего встречается один потенциальный ключ, который выбирается первичным.

Определим потенциальные ключи отношения Купленные фильмы:

КФИЛ = {{ нп, нф}, { нп, нф, лгн, прл, имя, фам, тип, блнс, нс, назф, опис, цен, скдк, рзмр, нж, нр, нкс}}

Имеется множество атрибутов A = {нп, нф, лгн, прл, фам, тип, блнс, нс, назф, опис, цен, скдк, рзмр, нж, нр, нкс}, подмножество ФЗ {{нп}→{лгн},{нп}→{прл},{нп}→{имя},{нп}→{фам},{нп}→{тип},{нп}→{блнс},{нп}→{нс}, {нф}→{назф},{нф}→{опис},{нф}→{цен},{нф}→{скдк},{нф}→{рзмр}{нф}→{нж},{нф}→{нр},{нф}→{нкс}}.

Следует начать с множества с наименьшей мощностью, то есть с множества { нп, нф}:

Построим замыкание { нп, нф}+ для атрибута { нп, нф}:

1. { нп, нф}+ = { нп, нф}

Переберем все ФЗ подмножества:

1. {нп}→{лгн}∧{нп}⊂{нп, нф}+∧{лгн}⊄{нп, нф}+=>{нп, нф}+ = {нп,нф}+∪{лгн} = {нп, нп, лгн}
2. {нп}→{прл}∧{нп}⊂{нп, нф}+∧{прл}⊄{нп, нф}+=>{нп, нф}+ = {нп,нф}+∪{прл} = {нп, нф, лгн, прл}
3. {нп}→{имя}∧{нп}⊂{нп, нф}+∧{имя}⊄{нп, нф}+=>{нп, нф}+ = {нп,нф}+∪{имя} = {нп, нф, лгн, прл, имя}
4. {нп}→{фам}∧{нп}⊂{нп, нф}+∧{фам}⊄{нп, нф}+=>{нп, нф}+ = {нп,нф}+∪{фам} = {нп, нф, лгн, прл, фам}
5. {нп}→{тип}∧{нп}⊂{нп, нф}+∧{тип}⊄{нп, нф}+=>{нп, нф}+ = {нп,нф}+∪{тип} = {нп, нф, лгн, прл, фам, тип}
6. {нп}→{блнс}∧{нп}⊂{нп, нф}+∧{блнс}⊄{нп, нф}+=>{нп, нф}+ = {нп,нф}+∪{блнс} = {нп, нф, лгн, прл, фам, тип, блнс}
7. {нп}→{нс}∧{нп}⊂{нп, нф}+∧{нс}⊄{нп, нф}+=>{нп, нф}+ = {нп,нф}+∪{нс} = {нп, нф, лгн, прл, фам, тип, блнс, нс}
8. {нф}→{назф}∧{нф}⊂{нп, нф}+∧{назф}⊄{нп, нф}+=>{нп, нф}+ = {нп,нф}+∪{назф} = {нп, нф, лгн, прл, фам, тип, блнс, нс, назф}
9. {нф}→{опис}∧{нф}⊂{нп, нф}+∧{опис}⊄{нп, нф}+=>{нп, нф}+ = {нп,нф}+∪{опис} = {нп, нф, лгн, прл, фам, тип, блнс, нс, назф, опис}
10. {нф}→{цен}∧{нф}⊂{нп, нф}+∧{цен}⊄{нп, нф}+=>{нп, нф}+ = {нп,нф}+∪{цен} = {нп, нф, лгн, прл, фам, тип, блнс, нс, назф, опис, цен}
11. {нф}→{скдк}∧{нф}⊂{нп, нф}+∧{скдк}⊄{нп, нф}+=>{нп, нф}+ = {нп,нф}+∪{скдк} = {нп, нф, лгн, прл, фам, тип, блнс, нс, назф, опис, цен, скдк}
12. {нф}→{рзмр}∧{нф}⊂{нп, нф}+∧{рзмр}⊄{нп, нф}+=>{нп, нф}+ = {нп,нф}+∪{рзмр} = {нп, нф, лгн, прл, фам, тип, блнс, нс, назф, опис, цен, скдк, рзмр}
13. {нф}→{нж}∧{нф}⊂{нп, нф}+∧{нж}⊄{нп, нф}+=>{нп, нф}+ = {нп,нф}+∪{нж} = {нп, нф, лгн, прл, фам, тип, блнс, нс, назф, опис, цен, скдк, рзмр, нж}
14. {нф}→{нр}∧{нп}⊂{нп, нф}+∧{нр}⊄{нп, нф}+=>{нп, нф}+ = {нп,нф}+∪{нр} = {нп, нф, лгн, прл, фам, тип, блнс, нс, назф, опис, цен, скдк, рзмр, нж, нр}
15. {нф}→{нкс}∧{нп}⊂{нп, нф}+∧{нкс}⊄{нп, нф}+=>{нп, нф}+ = {нп,нф}+∪{нкс} = {нп, нф, лгн, прл, фам, тип, блнс, нс, назф, опис, цен, скдк, рзмр, нж, нр, низ}

Множество { нп, нф }+ = A => { нп, нф } – потенциальный ключ отношения Купленные фильмы.

Проверим ключ { нп, нф } на неизбыточность:

1. Ключ {нп} – номер пользователя, не обладает свойством уникальности, т.к. у одного фильма, может быть, несколько копий для разных пользователей
2. Ключ {нф} – номер фильма, не обладает свойством уникальности, т.к. один пользователь может владеть несколькими копиями разных фильмов.

Множество {нп, нф} имеет наименьшее значение мощности и является потенциальным ключом, дальнейшая проверка оставшихся множеств супер-ключа на потенциальность бессмысленна.

Множество {нп} не является потенциальным ключом, т.к. зная только номер пользователя мы не можем определить владеет ли конкретный пользователь конкретным фильмом, так же это работает и в обратную сторону, зная только множество {нф} мы не можем определить каким конкретно пользователь владеет фильмом. Соответственно, использую доказательство от противного, мы можем сделать вывод, что подмножества множества {нп, нф} не являются потенциальным ключом.

Множество {нп, нф} не является избыточным, так как хранит уникальную комбинацию данных и нормализован.

## 2.7 Выбор первичных ключей

**Первичный ключ** - множество атрибутов, значение которых уникальным образом идентифицирует кортеж в отношении.

Первичным ключом отношения **Купленные фильмы** выбирается множество атрибутов **Номер Фильма** и **Номер Пользователя**, потому что данное множество атрибутов уникально для каждого кортежа отношения, определяет все множество атрибутов отношения, то есть является потенциальным ключом, имеющим наименьшую мощность множества.

## 2.8 Обосновать выбор внешних ключей

Первичный ключ — множество атрибутов, значения которых уникальным образом идентифицируют кортеж в отношении.

Обоснование внешних ключей:

Пользователь и Страна:

Внешний ключ: Номер страны в таблице Пользователи.

Обоснование: Каждый пользователь принадлежит определенной стране, и атрибут Номер страны в таблице Пользователи связывает пользователей с их странами проживания.

Фильмы и Режиссер:

Внешний ключ: Номер режиссера в таблице Фильмы.

Обоснование: Фильм связан с режиссером, и атрибут Номер режиссера в таблице Фильмы устанавливает эту связь.

Фильмы и Киностудия:

Внешний ключ: Номер киностудии в таблице Фильмы.

Обоснование: Каждый фильм принадлежит киностудии, и атрибут Номер киностудии в таблице Фильмы создает связь с соответствующей киностудией.

Режиссеры и Страна:

Внешний ключ: Номер страны в таблице Режиссеры.

Обоснование: Режиссер связан с его страной происхождения, и атрибут Номер страны в таблице Режиссеры определяет эту связь.

Киностудии и Страна:

Внешний ключ: Номер страны в таблице Киностудии.

Обоснование: Киностудия связана с ее страной, и атрибут Номер страны в таблице Киностудии устанавливает эту связь.

Фильмы и Жанры:

Внешний ключ: Номер жанра в таблице Фильмы.

Обоснование: Фильм относится к определенному жанру, и атрибут Номер жанра в таблице Фильмы формирует эту связь.

Таким образом, использование внешних ключей обеспечивает связанные отношения между таблицами, согласно структуре базы данных, и поддерживает целостность данных.

## 2.9 Нормализация отношений предметной области

**Исходное отношение**

Пользователь::={ **номер пользователя**, логин, пароль, имя, фамилия, тип пользователя, баланс, страна::={ **номер страны**, название страны}}

Фильмы::={ **номер фильма**, название фильма, описание, цена, скидка, размер, режиссер::={**номер режиссера**, название режиссера, адрес режиссера, страна::={ **номер страны**, название страны }}, киностудии::={ **номер киностудии**, название киностудии, адрес киностудии, страна::={ **номер страны**, название страны }}, жанр::={**номер жанра**, название жанра}

**Приводим отношения к 1НФ**

Раскрываем составные атрибуты.

Купленные Фильмы = { **номер пользователя**, **номер фильма**, логин, пароль, имя, фамилия, тип пользователя, баланс, номер страны, название фильма, описание, цена, скидка, размер, номер жанра, номер режиссера, номер киностудии, дата}

Фильмы = {**номер фильма**, название фильма, описание, цена, скидка, размер, номер жанра, номер режиссера, номер киностудии},

Жанры = {**номер жанра**, название жанра},

Режиссер = {**номер режиссера**, название киностудии, адрес киностудии, номер страны},

Киностудия = {**номер киностудии**, название киностудии, адрес киностудии, номер страны},

Страны = {**номер страны**, название страны},

Пользователи = {**номер пользователя**, логин, пароль, имя, фамилия, тип пользователя, баланс, номер страны}

**Приводим отношения к 2НФ**

В отношении Купленных фильмов, информация о названии фильма, описании и т.д хранится, пока фильм продается, также информация от логине, пароле и т.д хранится, пока пользователь зарегистрирован. Для решения этой проблемы преобразуем Купленные фильмы в супер-ключ.

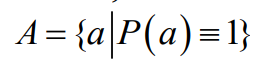
Купленные Фильмы = { **номер пользователя**, **номер фильма,** дата}

**Приводим отношения к 3НФ**

Во всех отношениях выполняется нетранзитивная независимость, то есть все неключевые атрибуты взаимно независимы, и при изменении одного из атрибутов, остальные атрибуты останутся неизменными.

## 2.10 Предикаты для проверки целостности базы данных

Как известно, множество может быть задано с помощью предиката:

 , где: P(a)  - неопределенное высказывание (предикат).

Можно показать, что реляционному выражению можно поставить в соответствие некоторый предикат, который определит некоторое отношение, совпадающее с результатом вычисления данного реляционного выражения.

Исчисление предикатов необходимо производить на множестве некоторых переменных. Первоначально реляционное исчисление было основано на переменных кортежа

* Переменная кортежа – это переменная, которая изменяется на некотором отношении..
* Реляционное исчисление, основанное на переменных кортежа, называется исчислением кортежей.
* Альтернативное исчисление, определённое на переменных доменов (т.е. переменных, которые изменяются не на отношениях, а на доменах) называется исчислением доменов.

Предикат проверки целостности для отношения Фильмы:

A={нф, назф, опис, цен, скдк, рзмр, нж, нр, нкс | P(скдк) ≥ 0 & P(скдк) ≤ 100}

**Условия целостности для отношения Пользователи:**

Тип пользователя может быть либо “user” либо “admin”

Номер пользователя уникальным образом идентифицирует кортеж отношения, является уникальным для каждого кортежа.

Все атрибуты должны иметь значения.

**Условия целостности для отношения Жанры:**

Номер жанра уникальным образом идентифицирует кортеж в отношении, уникален для каждого кортежа.

Название жанра должно быть уникальным.

Все атрибуты должны иметь значения.

**Условия целостности для отношения Страны:**

Номер страны уникальным образом идентифицирует кортеж в отношении, уникален для каждого кортежа.

Все атрибуты должны иметь значения.

**Условия целостности для отношения Киностудии:**

Номер киностудии уникальным образом идентифицирует кортеж в отношении, уникален для каждого кортежа.

Все атрибуты, кроме отчества покупателя, должны иметь значения.

**Условия целостности для отношения Режиссер:**

Номер режиссера уникальным образом идентифицирует кортеж в отношении, уникален для каждого кортежа.

Все атрибуты должны быть иметь значения.

**Условия целостности для отношения Купленные фильмы:**

Номер пользователя и номер фильма уникальным образом идентифицирует кортеж в отношении, сочетание двух атрибутов уникально для каждого кортежа.

Все атрибуты, кроме номера шкафа, должны иметь значения.

**Условия целостности для отношения Фильмы:**

Номер фильма уникальным образом идентифицирует кортеж в отношении, уникален для каждого кортежа.

Скидка не может быть меньше или равно 0 и больше или равна 100:

Все атрибуты, кроме описания, должны иметь значения.

## 2.11 Описание требуемых виртуальных отношений

Виртуальные отношения – отношения, образованные с помощью реляционных выражений на основе одного или нескольких базовых отношений (определение представлений);

Пример: если некоторое значение вычисляется на основе значений нескольких атрибутов, то оно может быть явно представлено в одном из виртуальных представлений. Виртуальное отношение формируется на основе некоторого выражения реляционного языка. Виртуальное представление может использоваться для доступа как обычное отношение БД. Корректность информации, доступной через виртуальное представление обуславливается следующим (шестым) правилом.

Схемы виртуальных отношений:

1. Список всех фильмов = {название фильма, цена, жанр, режиссер, киностудия, размер}
2. Список пользователей = {логин, имя, фамилия, страна, баланс}
3. Список фильмов по одному жанру = {название фильма, цена, режиссер, киностудия, размер}

## 2.12 Реляционные выражения

Реляционное выражение над отношениями, составленное из реляционных операций. Результат вычисления этого выражения – тоже отношение.

Реляционные операции – операции над отношениями. Результатом любой реляционной операции является также отношение.

Разработаем реляционные выражения для реализации запросов.

1. Реляционное выражение для всех фильмов:

R – отношение Список фильмов

Π2,4,7,8,9,6(R) = {(w2,w4,w7,w8,w9,w6)| (v1,v2,…,vn) ∈ R|wj = vij, j=2,4,6,7,8,9}

1. Реляционное выражение для всех пользователей:

R – отношение Список пользователей

Π2,4,5,8,7(R) = {(w2,w4,w5,w8,w7)| (v1,v2,…,vn) ∈ R|wj = vij, j = 2,4,5,7,8}

1. Реляционное выражение для фильмов одного жанра:

R – отношение Список фильмов

σF(π2,4,7,8,9,6(R)) = {(v1,v2,…,vn) ∈ π2,4,7,8,9,6(R)| F(R) ≡ “жанр”}

# Физическая модель

В данной главе описывается реализация взаимодействия прикладного приложения с базой данных с использованием языка Python и PostgreSQL.

Реализация запроса на покупку фильма (рис 3.1):

def buy\_film(user\_id,game\_id):

try:

if (check\_if\_owned(user\_id,film\_id)==0):

if (get\_user\_balance(user\_id)>int(float(get\_film(film\_id)[4]) \* ((100 - get\_game(game\_id)[5]) / 100))): new\_balance=get\_user\_balance(user\_id)-int(float(get\_film(film\_id)[4]) \* ((100 - get\_film(film\_id)[5]) / 100))

cur.execute("UPDATE users SET currency = %s WHERE user\_id=%s",(new\_balance,user\_id))

cur.execute("INSERT INTO owned\_films(film\_id,user\_id) VALUES (%s,%s)", (film\_id,user\_id))

conn.commit()

return True

else: return "Недостаточно средств"

else:return "У вас уже есть этот фильм"

except Exception as e:return ("Error"+e)

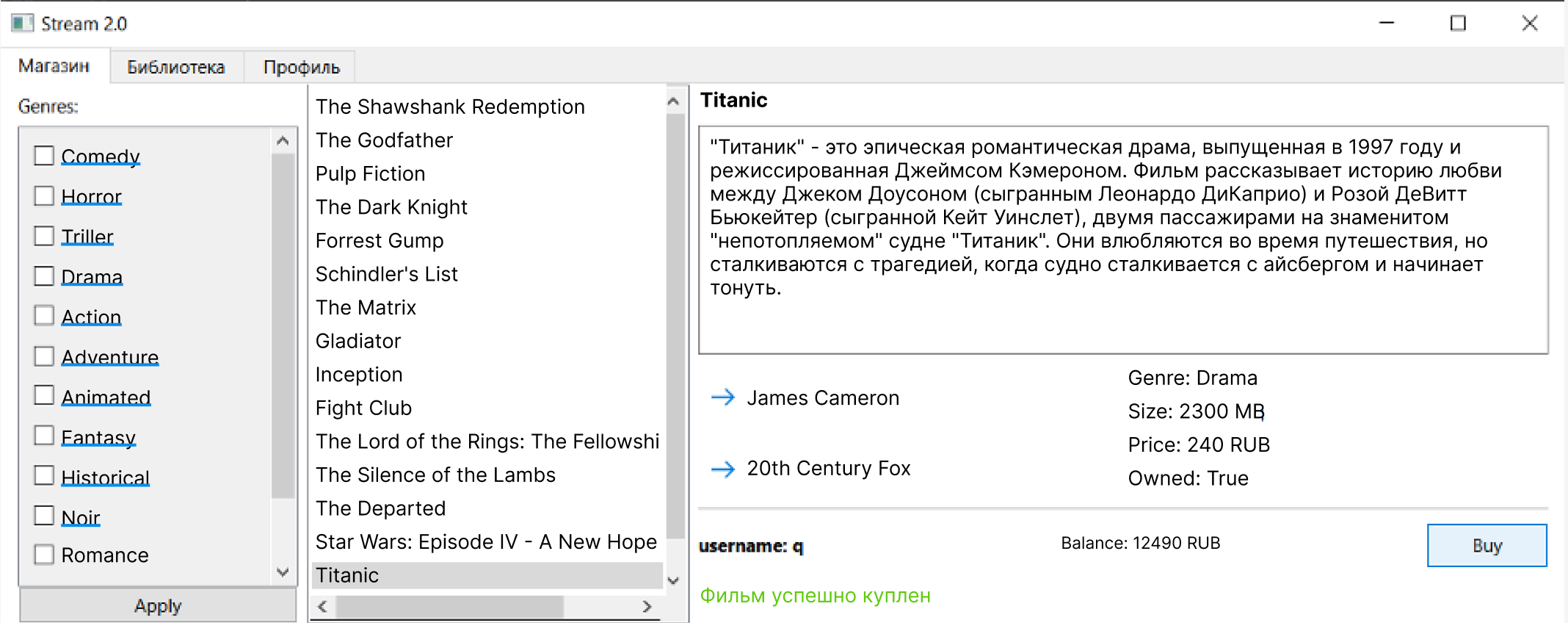


Рисунок 3.1 – покупка фильма

Поскольку сущность фильма, в результате нормализации, представляется двумя отношениями: Фильмы, Куплененные фильмы; то при покупки фильма реализуются два запроса в соответствующие отношения, однако в представлении конечного пользователя добавление товара является единым целым.

Реализация запроса на получение фильмов по конкретному жанру (рис 3.2):

def get\_films\_by\_genres(genres):

try:

films=[]

cur.execute("SELECT films.film\_name FROM films INNER JOIN genres f on f.genre\_id = films.genre\_id WHERE f.genre\_name = ANY(%s) ORDER BY film\_name",(genres,))

for film in cur:

films.append(game[0])

return films

except: print("ERROR to get films")

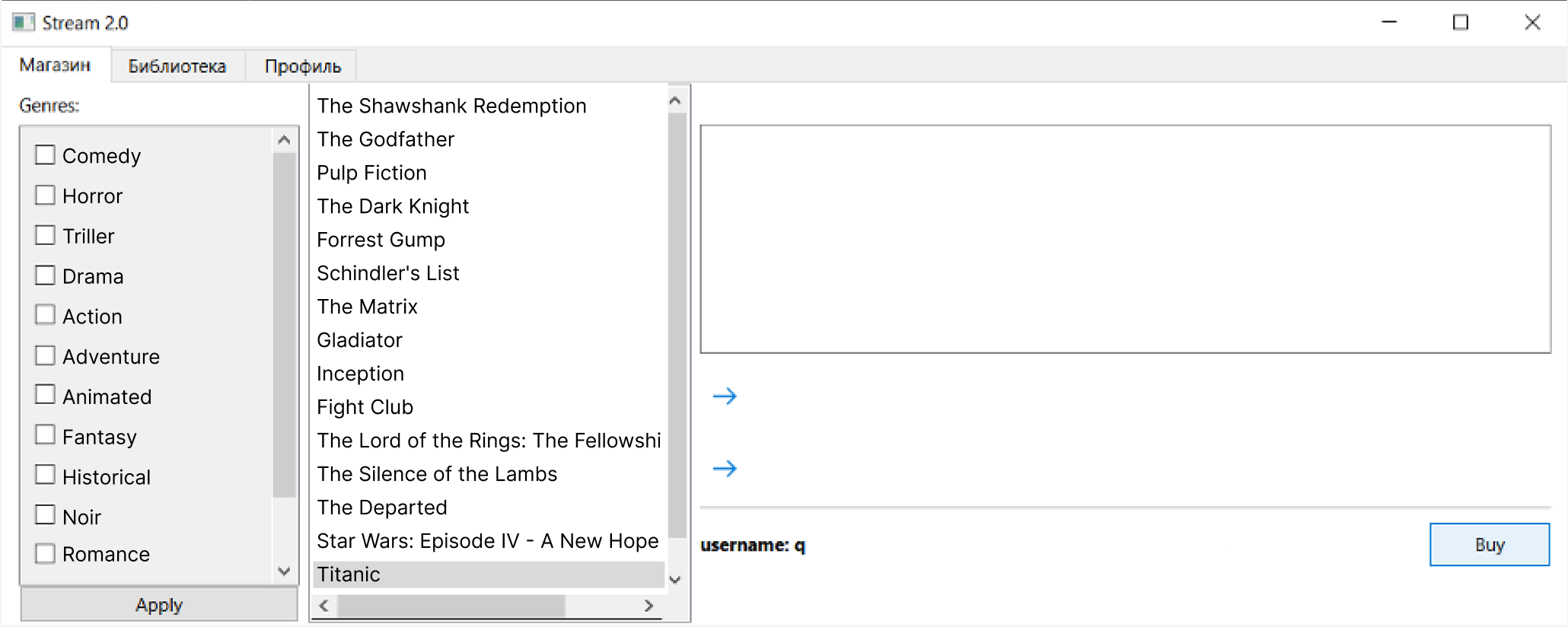


Рисунок 3.2 – отсортированный список фильмов

Программная реализация запроса на получение информации о режиссере (рис 3.3):

def get\_res(res\_id):

try:

cur.execute("SELECT res\_name, res\_adress, country\_id, res\_description FROM res WHERE res\_id=%s",( res\_id,))

res\_info=cur.fetchone()

return res\_info

except: return "Error"

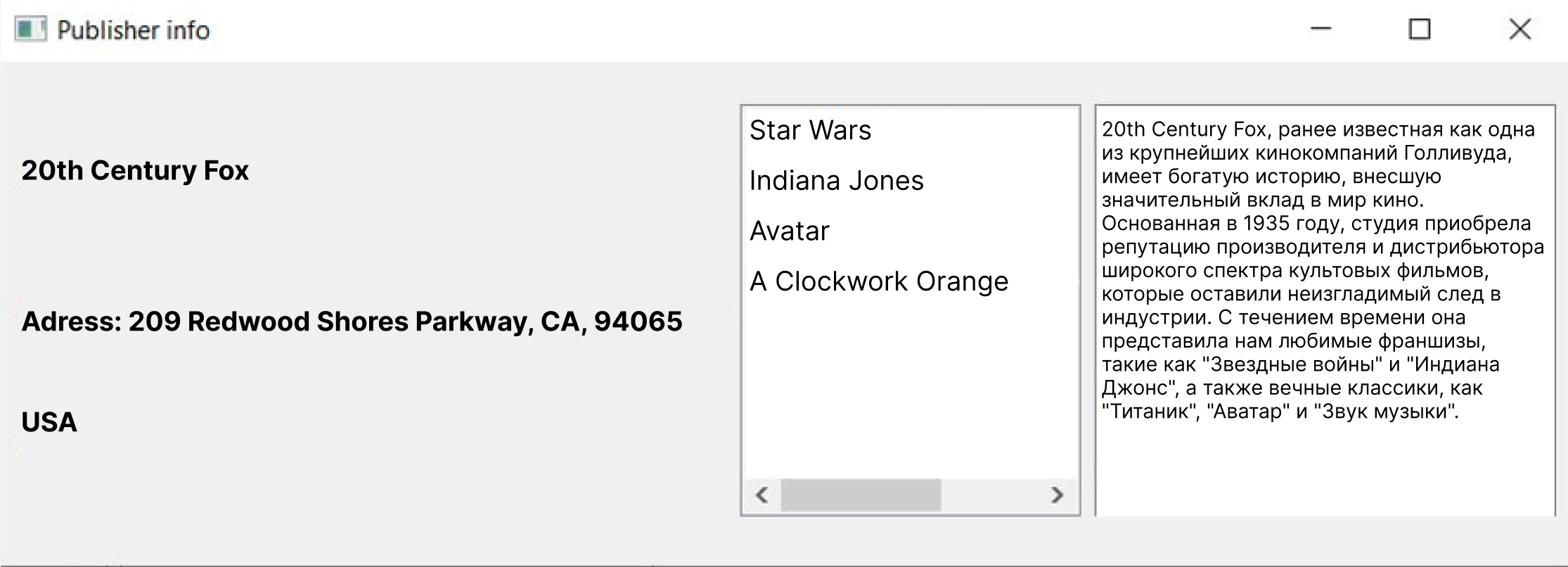


Рисунок 3.3 – информация о киностудии

Программная реализация запроса на получение информации об киностудии:

def get\_kstudio(kstudio\_id):

try:

cur.execute("SELECT kstudio\_name, kstudio\_adress, country\_id, kstudio\_description FROM kstudio WHERE kstudio\_id=%s", (kstudio\_id,))

kstudio\_info=cur.fetchone()

return kstudio\_info

except: return "Error"

Программная реализация запроса на получение списка купленных фильмов:

def get\_all\_films():

try:

films = []

cur.execute("SELECT film\_name FROM games ORDER BY film\_name")

for film in cur:

films.append(game[0])

return films

except:

print("ERROR to get films")

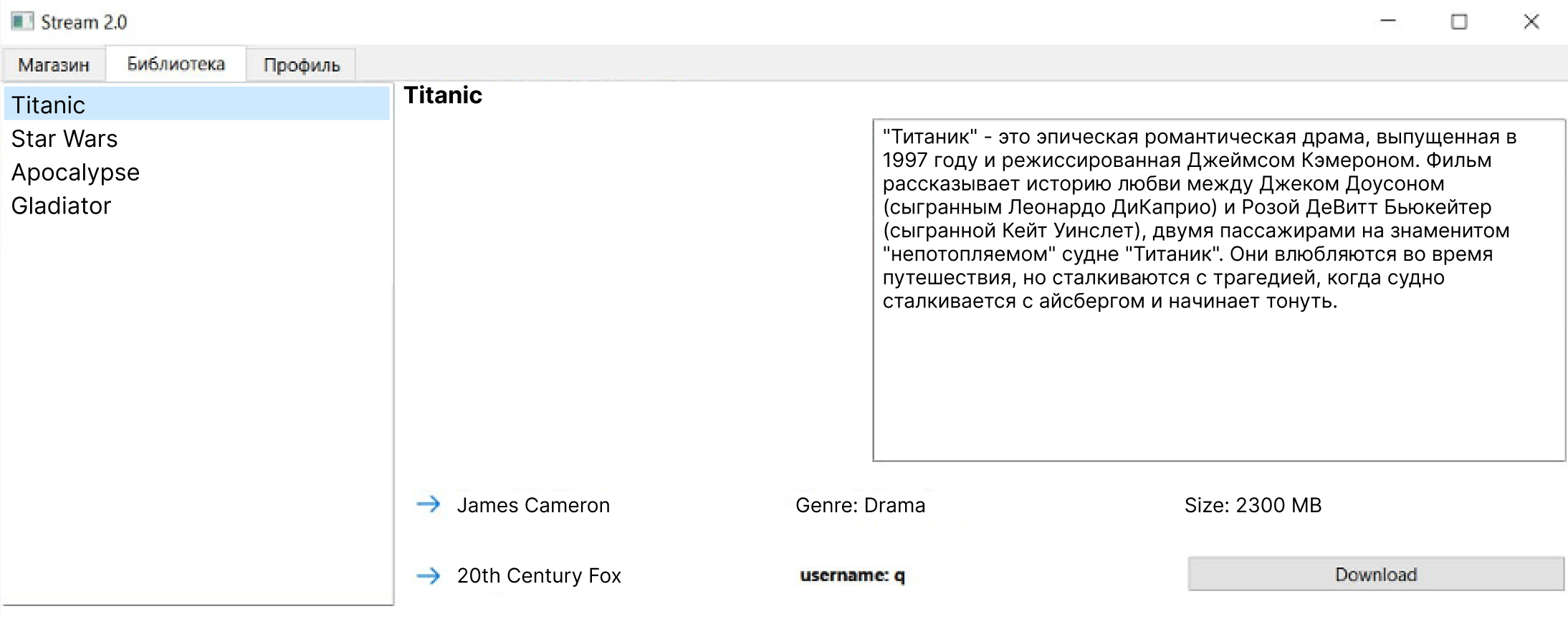


Рисунок 3.4 – список купленных фильмов

Программная реализация запроса на изменение пароля:

def change\_password(user\_id, password):

try:

cur.execute("UPDATE users SET password = %s WHERE user\_id = %s",(password,user\_id))

conn.commit()

return True

except: return False

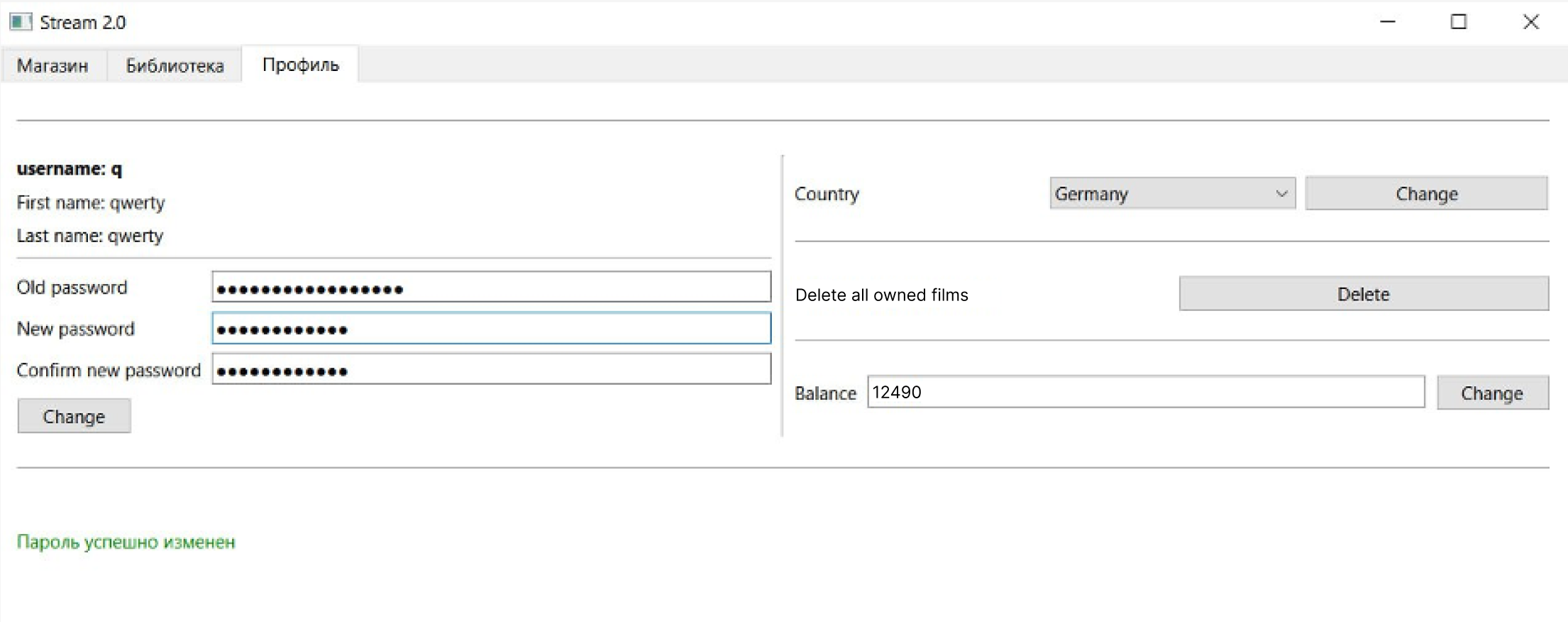


Рисунок 3.5 – изменение пароля

Программная реализация запроса на изменение страны:

def change\_country(user\_id,country\_id):

try:

cur.execute("UPDATE users SET country\_id = %s WHERE user\_id = %s",(country\_id,user\_id))

conn.commit()

return True

except: return False

Программная реализация запроса на удаление всех купленных фильмов:

def delete\_all\_owned\_films(user\_id):

try:

cur.execute("DELETE FROM owned\_films WHERE user\_id= %s", (user\_id,))

conn.commit()

return True

except: return False

Программная реализация запроса на изменение баланса:

def change\_balance(user\_id, balance):

try:

cur.execute("UPDATE users SET currency = %s WHERE user\_id = %s", (balance, user\_id))

conn.commit()

return True

except: return False

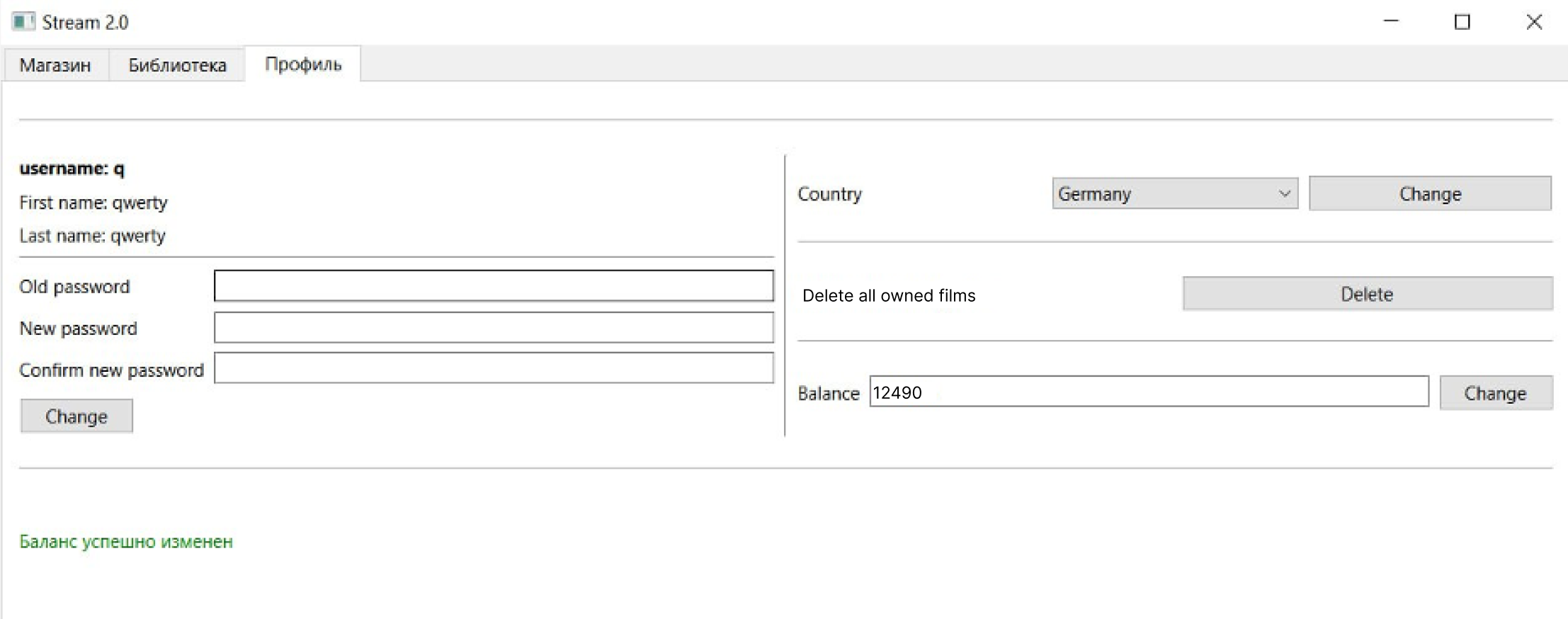


Рисунок 3.6 – изменение баланса