**Министерство науки и высшего образования Российской Федерации** Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Владимирский государственный университет**

**имени Александра Григорьевича и Николая Григорьевича Столетовых» (ВлГУ)**

Колледж инновационных технологий и предпринимательства

**КАФЕДРА ФИЗИКИ И ПРИКЛАДНОЙ МАТЕМАТИКИ**

**ОТЧЕТ**

**ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ**

**ПО МОДУЛЮ ПМ.02 «Разработка и администрирование баз данных»**

Студента *Годовикова Вадима Артёмовича*

Группы ПКспк-220

Специальности 09.02.03 Программирование в компьютерных системах

Время прохождения практики

с «25» апреля 2023 г.

по «03» июля 2023 г.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Руководитель практики* | |  | *Руководитель практики* | |
| *от ВлГУ* | |  | *от профильной организации* | |
| *преподаватель КИТП* | |  | *Руководитель SEO-отдела* | |
| *(каф. ФиПМ)* | |  |  | |
|  | *М.Ю. Кабанова* |  |  | *А.А. Усков* |

Владимир, 2023

|  |  |
| --- | --- |
|  | Утверждаю  Зав. кафедрой С.И. Абрахин  « » 2023 г. |
|
|

**ЗАДАНИЕ**

на производственную практику по модулю ПМ.02 «Разработка и администрирование БД» студента ***Годовикова Вадима Артёмовича***

(фамилия, имя, отчество)

**3** курса, направления ***09.02.03 Программирование в компьютерных системах***

группы **ПКспк-220**

Предприятие ООО «ВладВеб»

Последовательность прохождение практики

1. получить задание;
2. пройти инструктаж по технике безопасности и охране труда;
3. выполнить все задания;
4. оформить отчет согласно требованиям.

За время прохождения практики необходимо:

1. Изучить вопросы, предусмотренные программой по всем разделам.

2. Ознакомиться с инфраструктурой организации – базы прохождения практики,

с программно-аппаратными возможностями места практики

3. Изучить и исследовать

а) анализ предметной области, целей и назначения БД; б) написание скрипта создания БД,

логических элементов; в) наполнение БД; г) разработка логики манипуляции данными;

д) выполнение обслуживания БД; е) проектирование, разработка, тестирование клиентского

приложения для работы с БД.

4. Получить навыки оформления отчета по результатам практики согласно требованиям к

оформлению документации

5. Задание по охране труда, технике безопасности и охране окружающей среды

изучить правила техники безопасности при работе за ПК, вопросы охраны труда на месте

прохождения практики

Отчет по практике составить к **01.07.2023 г.**

Задание выдал: преподаватель КИТП М.Ю. Кабанова

(Ф.И.О, руководителя практики от университета)

Задание получил: В.А. Годовиков

(подпись студента, дата)

Примечание: задание должно быть приложено к отчету по практике (вторым листом после титульного листа)

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc138629577)

[1 Инфраструктура организации 5](#_Toc138629578)

[2 АНАЛИЗ ЗАДАНИЯ 9](#_Toc138629579)

[3 Разработка базы данных и ее объектов 12](#_Toc138629580)

[4 ОБРАБОТКА ИСХОДНЫХ ДАННЫХ. ИМПОРТ ДАННЫХ 17](#_Toc138629581)

[5 Построение запросов разных типов к базе данных 21](#_Toc138629582)

[6 Обслуживание базы данных 24](#_Toc138629583)

[7 Решение административных функций в базе данных 27](#_Toc138629584)

[8 Разработка приложения для работы с базами данных 32](#_Toc138629585)

[9 Тестирование разработанного функционала 46](#_Toc138629586)

[9.1 Авторизация под существующими учетными данными 48](#_Toc138629587)

[9.2 Авторизация под несуществующими учетными данными 49](#_Toc138629588)

[9.3 Валидация на обязательность заполняемых полей при создании и редактировании записей 50](#_Toc138629589)

[9.4 Просмотр записей из разных таблиц 51](#_Toc138629590)

[9.5 Удаление записей из разных таблиц 53](#_Toc138629591)

[Заключение 55](#_Toc138629592)

[Список использованных источников 56](#_Toc138629593)

[Приложение А 57](#_Toc138629594)

[Приложение Б 59](#_Toc138629595)

[Приложение В 61](#_Toc138629596)

ВВЕДЕНИЕ

Производственная практика по модулю ПМ.02 «Разработка и администрирование баз данных» проходит с 25 апреля 2023 года по 3 июля 2023 года.

Целью практики является проектирование и разработка базы данных по предметной области онлайн-кинотеатр для просмотра фильмов, а также реализация клиентского приложения для взаимодействия с базой данных.

В базу данных будут входить фильмы и вся их информация, актёры и создатели данных фильмов. Также в базе данных будут находиться пользователи, которые могут смотреть фильмы и могут приобрести ту или иную подписку.

В качестве СУБД будет использоваться SQL Server – это система управления базами данных, в работе с которой используется язык программирования SQL [1]. СУБД SQL Server используются для создания, размещения, хранения и управления реляционными (табличными) базами данных на специальных серверах или в облаке.

Трудно представить, что раньше вся информация размещалась на бумажных носителях и архивы документов занимали подчас целые здания. Сейчас же пришел новый век, новая информационная эпоха, и теперь для хранения и обработки информации используются в основном электронные системы [2].

1 Инфраструктура организации

Местом прохождения практики является веб-студия «ВладВеб». Данная студия на рынке уже 15 лет, создаёт сайты разной сложности под разные задачи.

В офисе веб-студии находятся четыре персональных компьютера, список аппаратного обеспечения организации представлен в таблице 1.

Таблица 1 – Аппаратное обеспечение

|  |  |
| --- | --- |
| Название аппаратного обеспечения | Характеристики |
| Персональный компьютер | Intel Core i5-11400F 2.60GHz, NVIDIA GT 740m, Realtek Family Controller, HDD 2TB, SSD 512 GB |

Программное обеспечение, используемое в веб-студии, представлено в таблице 2.

Таблица 2 – Программное обеспечение

|  |  |  |
| --- | --- | --- |
| Название ПО | Характеристики | Применение |
| Microsoft Office 2010 | Офисный пакет приложений (MS Word, MS Excel, MS Publisher, MS PowerPoint, MS Visio, MS Access) | Офисный пакет приложений, созданных корпорацией Microsoft. В состав этого пакета входит программное обеспечение для работы с различными типами документов. |

Продолжение таблицы 2.

|  |  |  |
| --- | --- | --- |
| Антивирус Касперского | Антивирус | Решение для базовой защиты компьютера от вредоносных программ. Продукт обеспечивает защиту в режиме реального времени от основных информационных угроз -- как известных, так и новых. |
| WinRAR | Архиватор | Архиватор файлов в форматы RAR и ZIP для 32-разрядных и 64-разрядных операционных систем Windows. |
| Mozilla Firefox | Интернет-браузер | Cвободный браузер, входящий в набор программ Mozilla Application Suite, разработкой и распространением которого занимается Mozilla Corporation. |

Физическая схема локальной вычислительной сети организации отображена на рисунке 1.

Логическая схема локальной вычислительной сети организации представлена на рисунке 2.

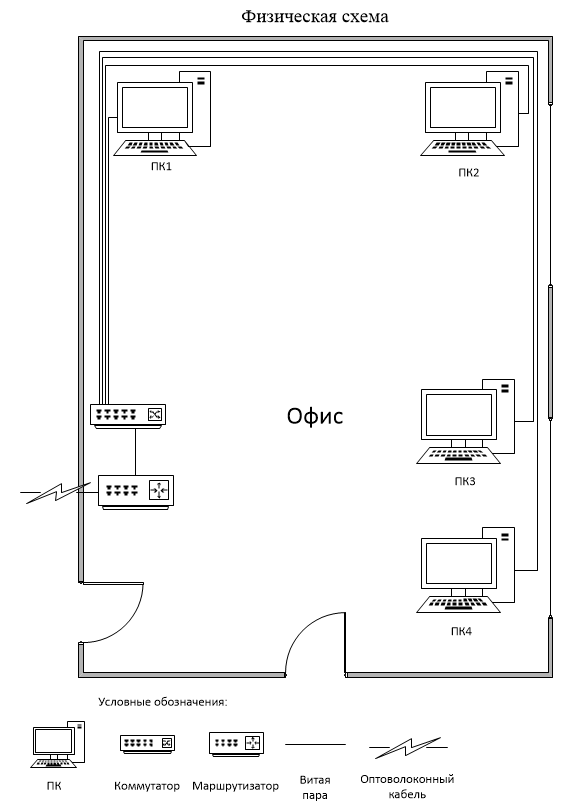


Рисунок 1 — Физическая схема ЛВС

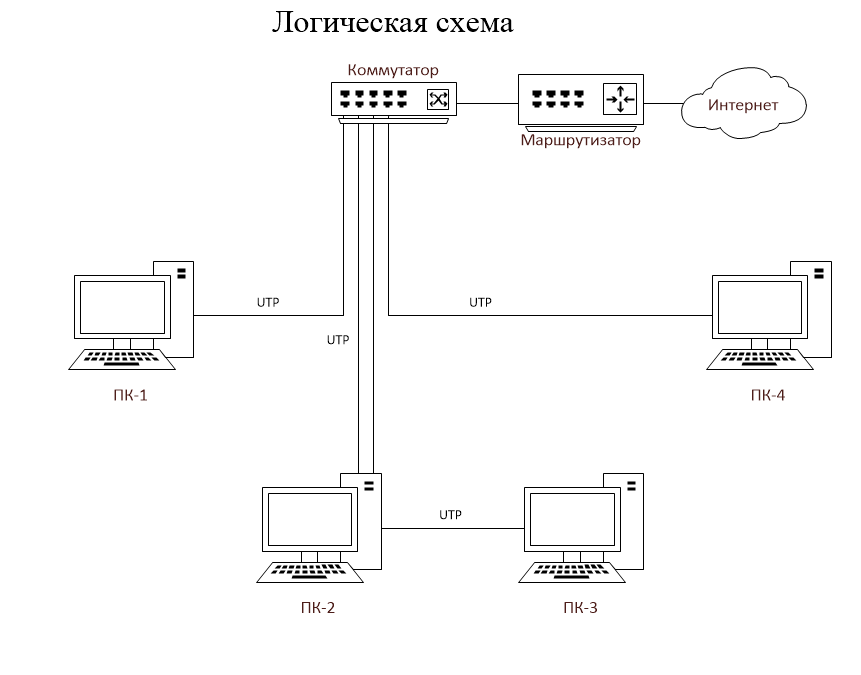


Рисунок 2 — Логическая схема ЛВС

Был произведён анализ места прохождения практики. Исследование локальной сети, а также организационных и технических моментов студии.

2 АНАЛИЗ ЗАДАНИЯ

При проектировании базы данных была проанализирована предметная область онлайн-кинотеатров.

В результате анализа были поставлены основные задачи, которые следует решить для разработки базы данных, такие как:

* Определить основные сущности базы данных.
* Описать атрибуты сущностей.
* Определить связи между сущностями.

Для реализации задач проектирования были построены диаграммы в нотации Чена и нотации Мартина.

Диаграмма в нотации Чена определяет основные сущности, ее атрибуты, ключевые поля и связи между сущностями, диаграмма изображена на рисунке 3.

Диаграмма в нотации Мартина показывает отношение между сущностями в виде таблиц базы данных с учётом первичных и внешних ключей, диаграмма изображена на рисунке 4.

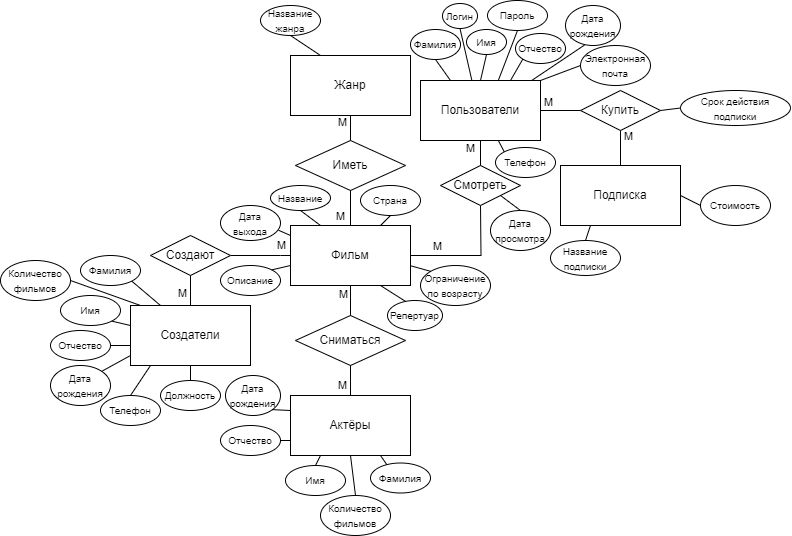


Рисунок 3 — Диаграмма базы данных в нотации Чена

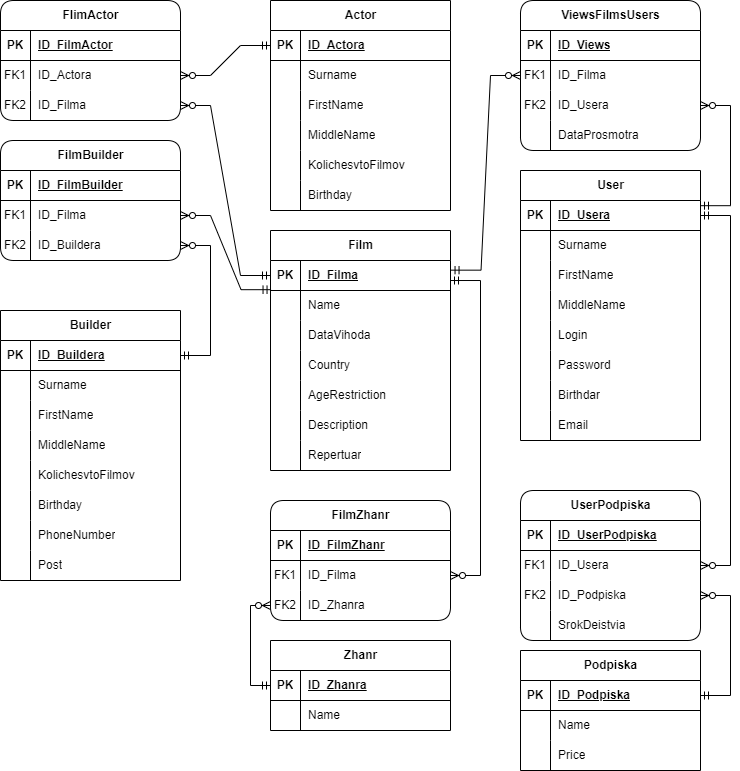


Рисунок 4 — Диаграмма базы данных в нотации Мартина

Совокупность диаграммы в нотации Чена и в нотации Мартина представляет собой ER-модель базы данных, по которой будет реализовываться база данных.

3 Разработка базы данных и ее объектов

В базе данных содержится следующий набор сущностей с присущими им атрибутами:

* Фильм (Film): уникальный идентификатор фильма, полное наименование фильма, страна создания фильма, ограничение по возрасту, описание, дата выхода, репертуар.
* Жанр (Zhanr): уникальный идентификатор жанра, наименование жанра.
* Актёр (Actor): уникальный идентификатор актёра, фамилия, имя, отчество актёра, количество фильмов в которых снимался актёр, дата рождения.
* Создатель (Builder): уникальный идентификатор создателя, фамилия, имя, отчество создателя, количество фильмов в которых участвовал при создании, дата рождения, номер телефона, должность.
* Пользователь (User): уникальный идентификатор пользователя, фамилия, имя, отчество пользователя, уникальный логин для входа в приложении, пароль, дата рождения, электронная почта.
* Подписка (Podpiska): уникальный идентификатор подписки, название подписки, цена подписки.

Словарь данных для создания базы данных представлен в таблице 3.

Таблица 3. Словарь данных

|  |  |  |  |
| --- | --- | --- | --- |
| Film | | | |
| Ключ | Поле | Обязательное (да/нет) | Примечание |
| Первичный ключ | ID\_Filma | Да | Код фильма |
|  | Name | Да | Название фильма |
|  | DataVihoda | Да | Дата выхода фильма |
|  | Country | Да | Страна создания фильма |

Продолжение таблицы 3

|  |  |  |  |
| --- | --- | --- | --- |
|  | AgeRestriction | Да | Возрастное ограничение фильма |
|  | Description | Да | Описание фильма |
|  | Repertuar | Да | Репертуар фильма. Может иметь значения такие как: фильм, мультфильм, короткометражка, сериал. |
| Zhanr | | | |
| Первичный ключ | ID\_Zhanra | Да | Код жанра |
|  | Name | Да | Название жанра |
| FilmZhanr | | | |
| Первичный ключ | ID\_FilmZhanr | Да | Код жанра фильма |
| Внешний ключ | ID\_Filma | Да | Внешний ключ на таблицу Film |
| Внешний ключ | ID\_Zhanra | Да | Внешний ключ на таблицу Zhanr |
| Actor | | | |
| Первичный ключ | ID\_Actora | Да | Код актёра |
|  | Surname | Да | Фамилия |
|  | FirstName | Да | Имя |
|  | MiddleName | Нет | Отчество |
|  | KolichestvoFilmov | Нет | Количество фильмов, в которых снимался актёр |
|  | Birthday | Нет | Дата рождения |
| FilmActor | | | |
| Первичный ключ | ID\_FilmActor | Да | Код актёра фильма |
| Внешний ключ | ID\_Actora | Да | Внешний ключ на таблицу Actor |
| Внешний ключ | ID\_Filma | Да | Внешний ключ на таблицу Film |
| Builder | | | |
| Первичный ключ | ID\_Builder | Да | Код создателя фильма |
|  | Surname | Да | Фамилия |

Продолжение таблицы 3

|  |  |  |  |
| --- | --- | --- | --- |
|  | FirstName | Да | Имя |
|  | MiddleName | Нет | Отчество |
|  | KolichestvoFilmov | Нет | Количество фильмов, в которых участвовал при создании |
|  | Birthday | Нет | Дата рождения |
|  | PhoneNumber | Нет | Номер телефона |
|  | Post | Да | Должность |
| FilmBuilder | | | |
| Первичный ключ | ID\_FilmBuilder | Да | Код создателя фильма |
| Внешний ключ | ID\_Filma | Да | Внешний ключ на таблицу Film |
| Внешний ключ | ID\_Builder | Да | Внешний ключ на таблицу Builder |
| User | | | |
| Первичный ключ | ID\_Usera | Да | Код пользователя |
|  | Surname | Да | Фамилия |
|  | FirstName | Да | Имя |
|  | MiddleName | Нет | Отчество |
|  | Login | Да | Логин для входа |
|  | Password | Да | Пароль для входа |
|  | Birthday | Да | Дата рождения |
|  | Email | Да | Электронная почта |
| ViewsFilmsUsers | | | |
| Первичный ключ | ID\_Views | Да | Код просмотра |
| Внешний ключ | ID\_Filma | Да | Внешний ключ на таблицу Film |
| Внешний ключ | ID\_Usera | Да | Внешний ключ на таблицу User |
|  | DataProsmotra | Да | Дата просмотра фильма |
| Podpiska | | | |
| Первичный ключ | ID\_Podpiska | Да | Код подписки |
|  | Name | Да | Название подписки |
|  | Price | Да | Цена подписки |
| UserPodpiska | | | |

Продолжение таблицы 3

|  |  |  |  |
| --- | --- | --- | --- |
| Первичный ключ | ID\_UserPodpiska | Да | Код покупки подписки |
| Внешний ключ | ID\_Usera | Да | Внешний ключ на таблицу User |
| Внешний ключ | ID\_Podpiska | Да | Внешний ключ на таблицу Podpiska |
|  | SrokDeistvia | Да | Срок действия подписки |

Код создания базы данных и таблиц представлен в приложении А.

Итоговая диаграмма базы данных, сгенерированная после создания базы данных, отображена на рисунке 5.

На диаграмме базы данных можно проследить связи между сущностями базы данных. Связь многие-ко-многим организована через создание ассоциативной таблица с внешними ключами на таблицы, между которыми организована связь один-ко-многим.

Разработанная схема базы данных находится в:

* первой нормальной форме (1НФ), так как в качестве доменов выступают только скалярные значения, и информация в таблицах не дублируется. Во всех таблицах есть идентификатор ID в качестве первичного ключа.
* второй и третьей нормальных формах (2НФ и 3НФ), каждый не ключевой атрибут неприводимо и нетранзитивно зависит от первичного ключа. Схема базы данных находится в нормальной форме Бойса-Кода.

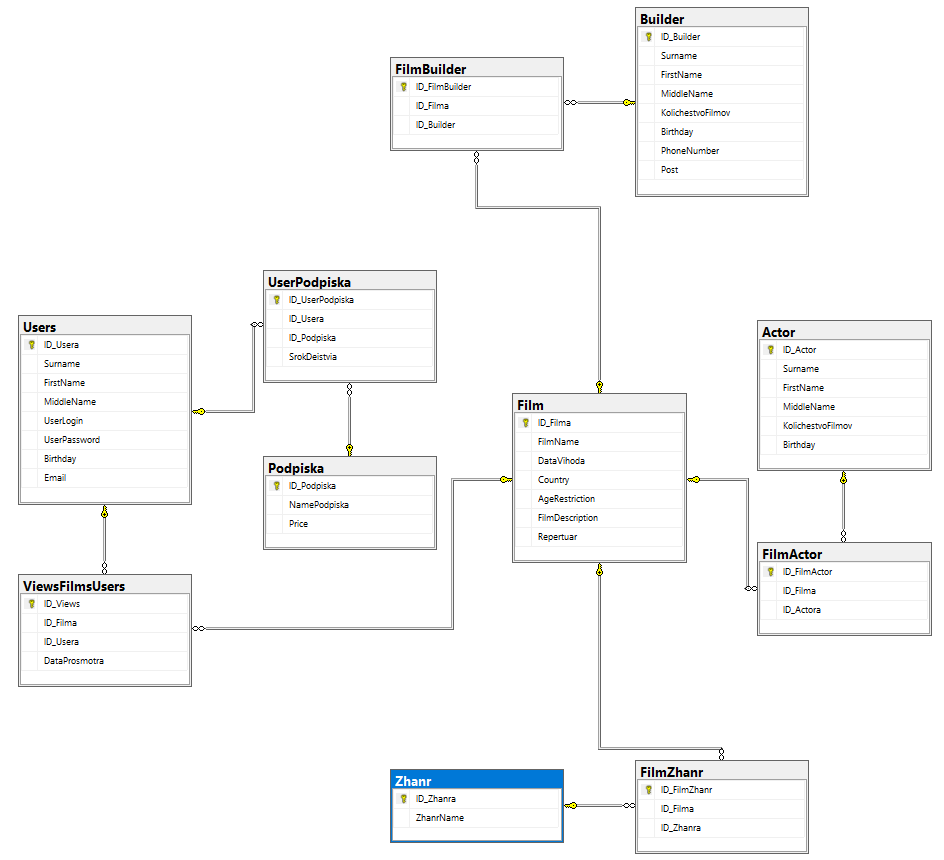


Рисунок 5 — Диаграмма базы данных

Разработанная база данных отвечает требованиям целостности данных, так как при создании таблиц был использован декларативный метод, что означает, что в базу данных невозможно записать некорректные данные.

4 ОБРАБОТКА ИСХОДНЫХ ДАННЫХ. ИМПОРТ ДАННЫХ

Для импорта данных в базу данных использовался Microsoft Excel. Целостность и корректность данных была достигнута с помощью нескольких ограничений при составлении таблицы Excel, таких как:

* В качестве данных в каждой ячейке таблицы выступают только скалярные значения и данные не дублируются, в каждой таблице имеется уникальный идентификатор строки.
* Каждый не ключевой атрибут неприводимо и нетранзитивно зависит от уникального идентификатора [3].

Процесс импорта данных изображен на рисунках 6-12.

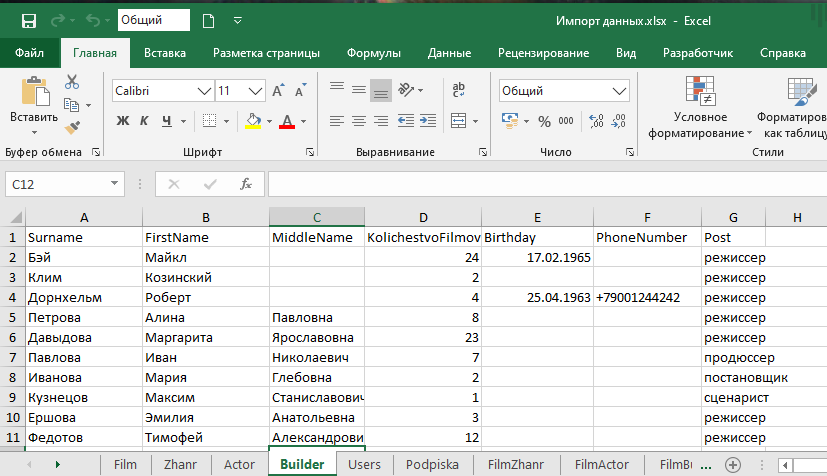


Рисунок 6 — Окно интерфейса документа Excel

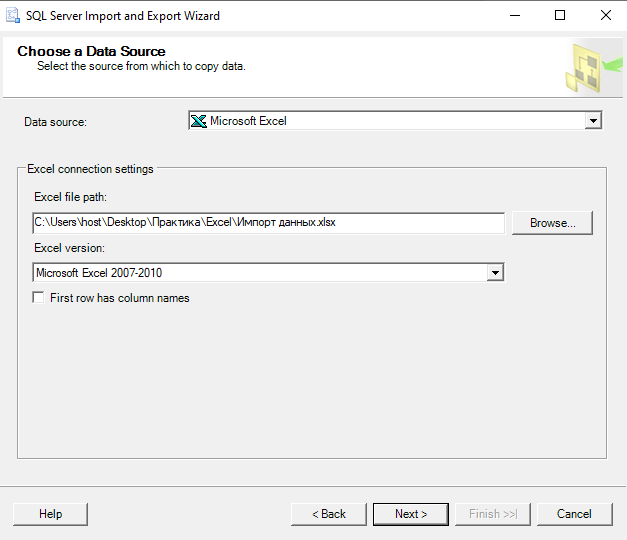


Рисунок 7 — Окно выбора источника данных

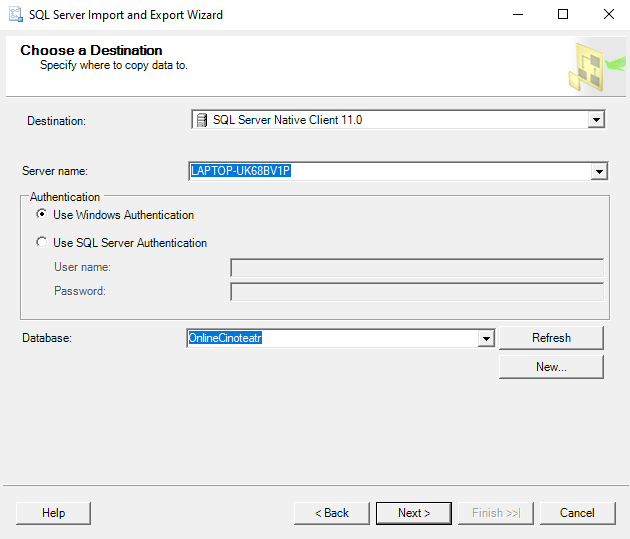


Рисунок 8 — Окно выбора назначения

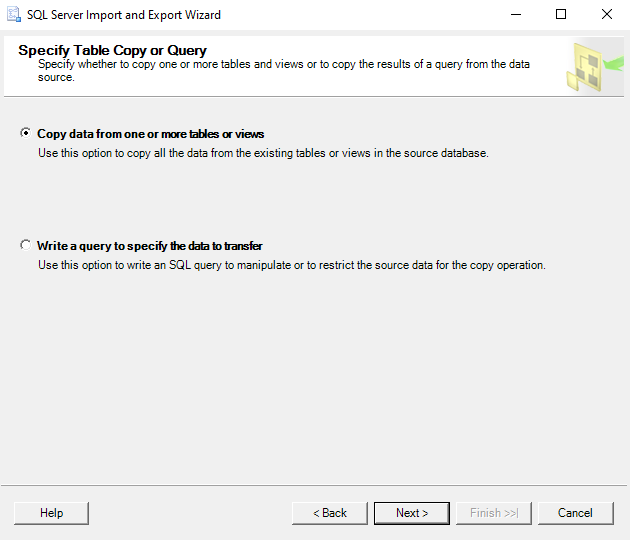


Рисунок 9 — Окно выбора копирования таблицы или запроса

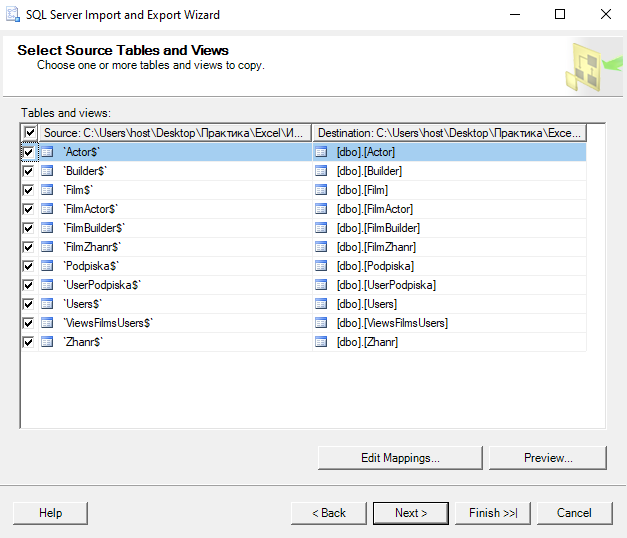


Рисунок 10 — Выбор исходных таблиц и представлений

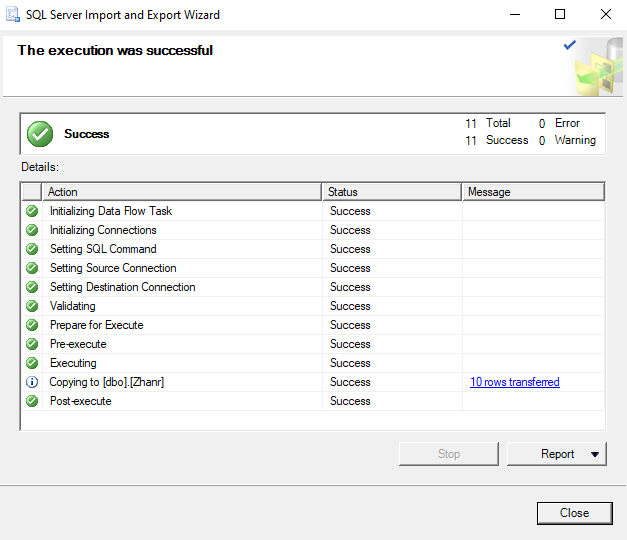


Рисунок 11 — Процесс импорта



Рисунок 12 — Результат импорта

С помощью импорта данных данные были успешно импортированы в разработанную базу данных.

5 Построение запросов разных типов к базе данных

В базе данных реализована хранимая процедура, скалярная функция и представление.

Хранимая процедура имеет название CheckUserPodpiska [4]. Она предназначена для того, чтобы вывести в одну таблицу информацию о пользователей, которые купили определенную подписку.

Данная процедура путем объединения таблиц Users, Podpiska, UsersPodpiska облегчает поиск информации. В результате хранимой процедуры формируется информация, в которую входят такие поля, как: фамилия пользователя, имя пользователя, логин пользователя, название купленной подписки и срок действия подписки. Также процедура имеет входной параметр, который определяет нужную для поиска подписку. Результат выполнения хранимой процедуры представлен на рисунке 13.

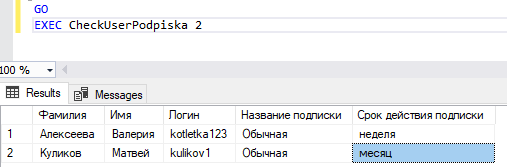


Рисунок 13 — Результат работы хранимой процедуры

В базе данных реализована скалярная функция KolvoUserBuyPodpiska, которая рассчитывает количество пользователей, которые купили определённую подписку [5]. Данные извлекаются из таблиц Users, Podpiska и UsersPodpiska. Выходным параметром скалярной функции является числовое значение, которое определяет тип подписки. Результат выполнения скалярной функции представлен на рисунке 14.

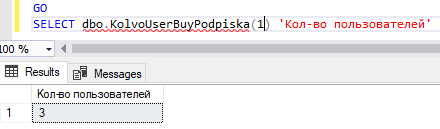


Рисунок 14 — Результат работы скалярной функции

В базе данных реализовано представление UsersViews, которое содержит информацию о просмотрах пользователями фильмов и содержит следующие поля: название фильма, фамилия пользователя, имя пользователя, логин пользователя, дата просмотра. Данные извлекаются из таблиц таких как: Users, Film, ViewsFilmUsers Содержимое представления отображено на рисунке 15.

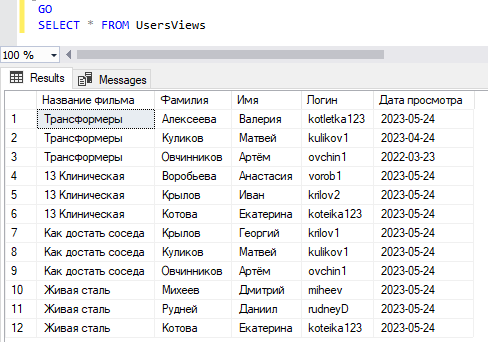


Рисунок 15 — Содержимое представления

Оператор SQL UPDATE предназначен для обновления (редактирования) данных в таблице. Он применяется, когда в той или иной строке таблицы уже записаны некоторые данные и нужно внести в них изменения.

В базе данных было реализовано обновление данных. Данный скрипт позволяет увеличить стоимость подписки на определённую сумму. Пример работы скрипта: первоначальные исходные данные находятся на рисунке 16.

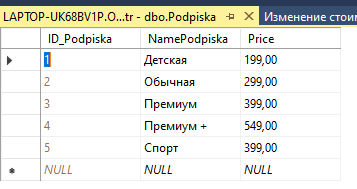


Рисунок 16 — Результат работы обновления данных

Листинг кода данного sql скрипта показан на рисунке 17.

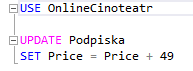


Рисунок 17 — Результат работы скалярной функции

Исходные данные после выполнения данного SQL-скрипта показаны на рисунке 18.



Рисунок 18 — Результат работы скалярной функции

Тем самым стоимость каждой подписки увеличилась на 50.

SQL запросы представлены в приложении Б.

6 Обслуживание базы данных

Индекс – это поле, по которому оптимизирован поиск данных. Таблицы могут иметь огромное количество строк, которые хранятся в произвольном порядке. Без индекса поиск нужных строк идёт по порядку (последовательно), что на больших объёмах данных отнимает много времени.

В базе данных OnliceCinoteatr был создан индекс FindZhanr, позволяющий ускорить поиск конкретных жанров по их названию в таблицу Zhanr. В качестве столбца был использован ZhanrName. SQL-скрипт создания индекса представлен в листинге 1, а результат его выполнения на рисунке 19.

Листинг 1 – Создание индекса FindZhanr

CREATE INDEX FindZhanr ON dbo.Zhanr (ZhanrName ASC)

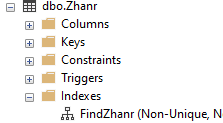


Рисунок 19 — Созданный индекс в дереве объектов

Если же индексом пользовались уже долгое время и он, так скажем устарел, то для улучшения производительности работы самого индекса, его можно перестроить. SQL-скрипт для перестройки индекса FindZhanr и его результат представлен на рисунке 20.

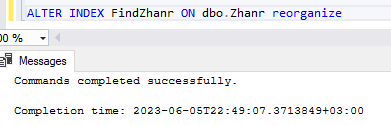


Рисунок 20 — Результат работы перестройки индекса

Также данный индекс можно удалить. SQL-скрипт и его выполнение показано на рисунке 21.

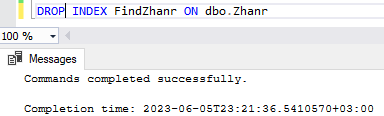


Рисунок 21 — Результат работы удаления индекса

Триггер – это такой специальный тип хранимой процедуры, которая вызывается автоматически при выполнении определённого действия над таблицей или представлением, в частности, при добавлении, изменении или удалении данных, то есть при выполнении команд INSERT, UPDATE, DELETE [6].

В реализуемой базе данных OnlineCinoteatr создан триггер UsersDelete, который записывает в служебную таблицу DeleteUsers тех пользователей, которые были по определённым причинам удалены из таблицы Users. При удалении пользователя, информация о нём записывается в служебную таблицу, результат работы триггера показан на рисунке 22.

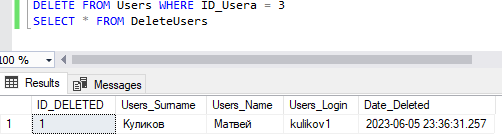


Рисунок 22 — Результат работы триггера

Листинг кода создания служебной таблицы и триггера UsersDelete представлены в приложении Б.

7 Решение административных функций в базе данных

Для реализации логического функционала доступа к данным различных пользователей, были определены следующие роли:

* ClientOC, представляющая собой роль клиента приложения, которая может просматривать такие таблицы, как Film, Zhanr, Actor, Builder, Podpiska;
* Administrator, представляющая собой администратора базы данных и имеющая неограниченные возможности по просмотру, редактированию и удалению записей из всех таблиц, а то есть: Actor, Builder, Film, FilmActor, FilmBuilder, FilmZhanr, Podpiska, UserPodpiska, Users, ViewFilmsUsers, Zhanr. А также выполнению хранимых процедур и функций.

Созданные роли и пользователи базы данных в дереве объектов представлены на рисунке 23.

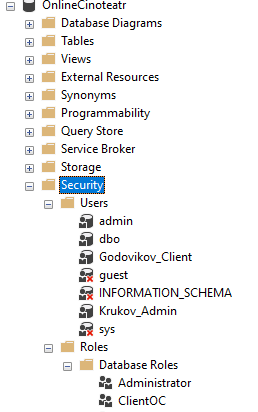


Рисунок 23 — Созданные роли и пользователи

Права роли Administrator представлены на рисунке 24.

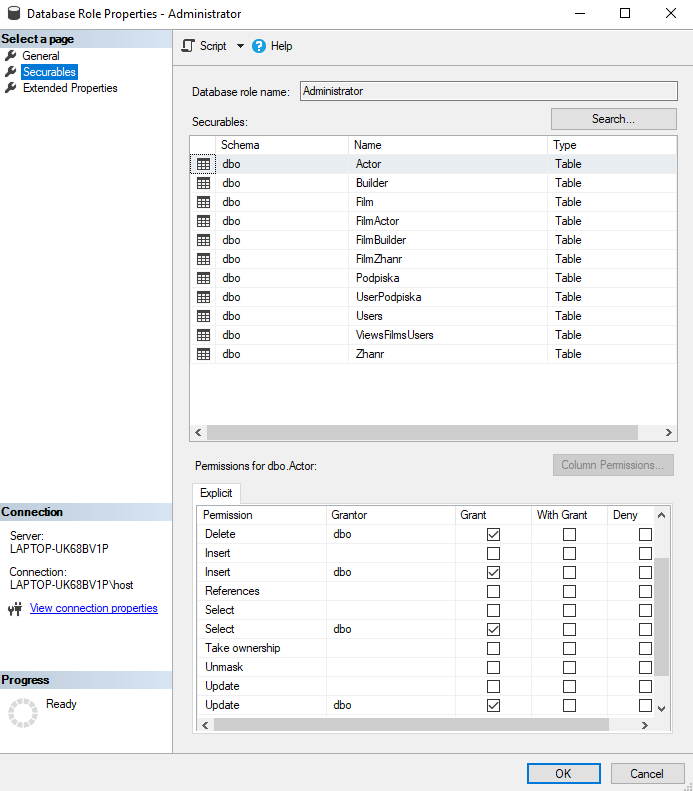


Рисунок 24 — Права роли Administrator

Права роли ClientOC представлены на рисунке 25.

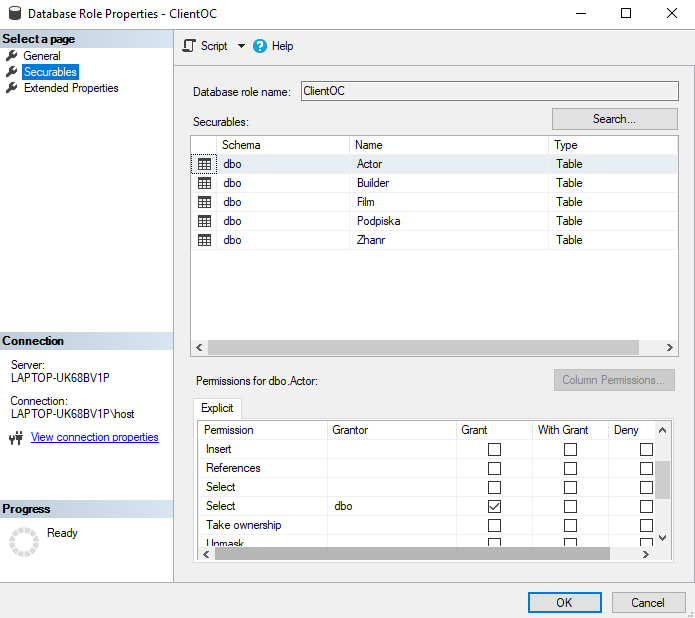


Рисунок 25 — Права роли ClientOC

Также в базе данных были созданы два пробных пользователя, таких как Krukov\_Admin с привязанной к ему ролью Administrator и Godovikov\_Client с ролью ClientOC.

Пользователи и роли предоставлены на рисунках 26-27.

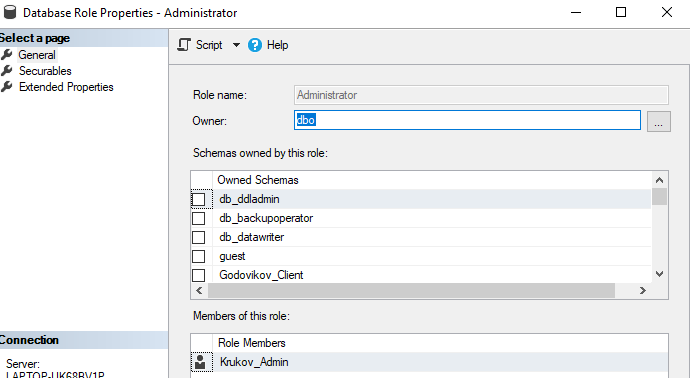


Рисунок 26 — Пользователи роли Administrator

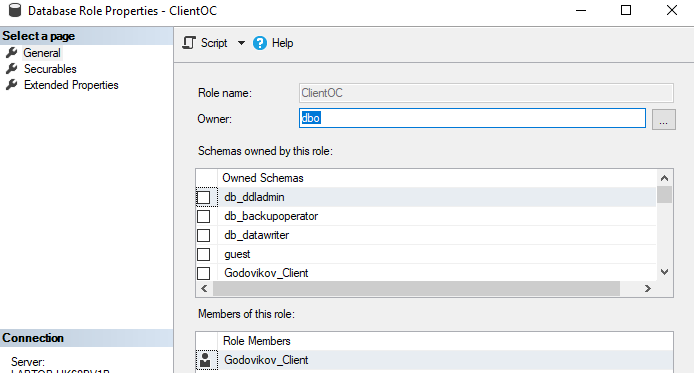


Рисунок 27 — Пользователи роли ClientOC

Были определены роли базы данных, их права, а также были созданы пробные пользователи на каждую роль. SQL-скрипты для создания ролей и пользователей, а также выдачи прав, представлены в приложении В.

Резервное копирование позволяет сохранить структуру базы данных, а самое главное – её содержимое, в любом месте на диске. Цель резервного копирования – это обезопасить себя от потери накопленных данных.

Для того, чтобы выполнить резервное копирование базы данных, необходимо выполнить представленный в листинге 2 SQL-скрипт, который сохранит резервную копию базы данных по указанному пути на диске компьютера.

Листинг 2 – Создание резервной копии базы данных

BACKUP DATABASE [OnlineCinoteatr]

TO DISK N'C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\Backup\OnlineCinoteatr.bak'

WITH NOFORMAT, NOINIT, NAME = N'OnlineCinoteatr База данных BACKUP',

SKIP, NOREWIND, NOUNLOAD, STATS = 10

GO

Для удобства применения данного скрипта также целесообразно разработать командный файл с расширением .bat, при выполнении которого данный SQL-скрипт будет выполняться. Код командного файла представлен в листинге 3.

Листинг 3 – Код командного файла для выполнения SQL-скрипта для создания резервной копии

sqlcmd -s LAPTOP-UK68BV1P –i C:\Users\host\Desktop\Практика\Запросы\Backup\backup.sql

Также был разработан SQL-скрипт, который позволяет восстановить базу данных из резервной копии, представленный в листинге 4.

Листинг 4 – Восстановление базы данных из резервной копии

USE [master]

RESTORE DATABASE [OnlineCinoteatr]

FROM DISK = N'C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\Backup\OnlineCinoteatr.bak'

WITH FILE = 1, NOUNLOAD, STATS = 5

GO

Как и на примере предыдущего SQL-скрипта, был создан командный файл, упрощающий работу с данным SQL-скриптом для восстановления базы данных из резервной копии, код которого представлен в листинге 5.

Листинг 5 – Код командного файла для выполнения SQL-скрипта для восстановления базы данных из резервной копии

sqlcmd -s LAPTOP-UK68BV1P –i C:\Users\host\Desktop\Практика\Запросы\Backup\restore.sql

Таким образом, были разработаны как SQL-скрипты для создания резервной копии базы данных и ее восстановления, так и командные файлы, упрощающие этот процесс.

8 Разработка приложения для работы с базами данных

Для клиентского приложения была разработана диаграмма прецендентов, которая предоставлена на рисунке 28.

Диаграмма прецендентов – диаграмма, описывающая функционал разрабатываемой программной системы, который доступен каждой группе пользователей.

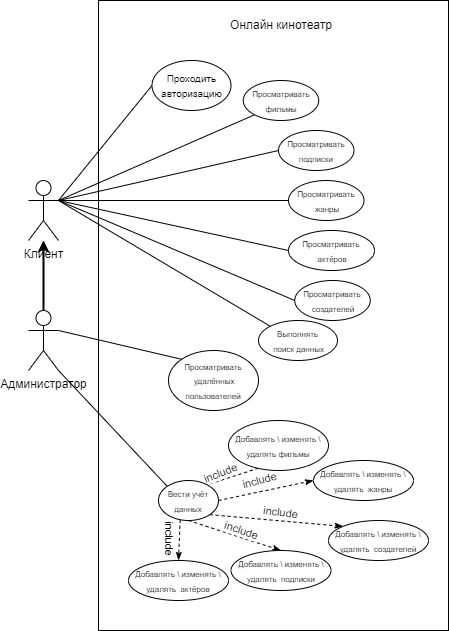


Рисунок 28 – Диаграмма прецедентов

Из представленной выше диаграммы прецендентов была построена диаграмма вариантов использования. Изображение диаграммы использования представлено на рисунке 29.

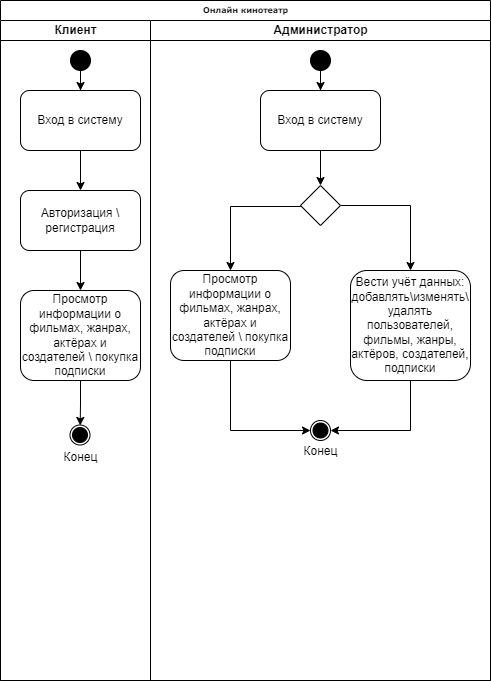


Рисунок 29 – Диаграмма вариантов использования

Клиентское приложение состоит из форм авторизации и регистрации, главной формы и набора форм для добавления и изменения данных. Каждая форма представляет собой пользовательский интерфейс, который помогает взаимодействовать с базой данных.

При запуске приложения запускается форма выбора, представлена на рисунке 30. С помощью которой, пользователь выбирает, зарегистрироваться или авторизоваться. Также данная форма располагает в себе кнопку «Выйти», что означает выход из приложения.



Рисунок 30 – Форма выбора

Форма авторизации содержит поля для ввода логина и пароля пользователя, а также кнопку «Войти» и кнопку «Назад», которая вернёт. Форма авторизации изображена на рисунке 31.

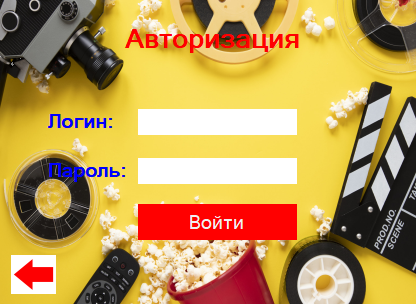


Рисунок 31 – Форма авторизации

При нажатии по кнопке авторизации на сервер отправляется запрос авторизации. По результату этого запроса определяется, имеет ли пользователь возможность пользоваться данным клиентским приложением.

При отказе в доступе или при вводе некорректных данных пользователю отображается ошибка.

При получении доступа, открывается главная форма.

Форма регистрации содержит в себе три поля для ввода логина и пароля, а также подтверждение пароля и кнопок «Выполнить» и «Назад». Изображение формы представлено на рисунке 32.

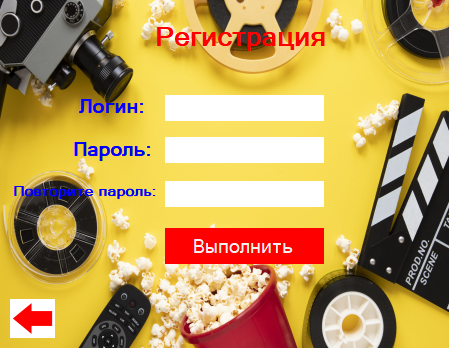


Рисунок 32 – Форма регистрации

При нажатии на кнопку «Выполнить», на сервер отправляется запрос о регистрации пользователя под ролью клиента. После чего перекидывает на форму авторизации.

Приложение содержит одну главную форму, с помощью которой осуществляется взаимодействие с базой данных для обычных пользователей и администратора. Только у пользователей функционал данной формы меньше, чем у администратора. В приложении имеются вспомогательные формы, для добавления и изменения записей в существующих таблицах базы данных, но доступ к ним обычный пользователь не имеет.

Главная форма с функционалом для пользователя, который авторизовался в качестве администратора, изображена на рисунке 33. Содержит в себе:

* Кнопку просмотра фильмов, включающая в себя таблицу для просмотра фильмов.
* Кнопку просмотра жанров, включающая в себя таблицу для просмотра жанров.
* Кнопку просмотра актёров, включающая в себя таблицу для просмотра актёров.
* Кнопку просмотра создателей, включающая в себя таблицу для просмотра создателей.
* Кнопку просмотра подписок, включающая в себя таблицу для просмотра подписок.
* Кнопку просмотра пользователей, включающая в себя таблицу для просмотра пользователей.
* Кнопку просмотра удалённых пользователей, включающая в себя служебную таблицу, нужна для того, чтобы сохранить целостность базы данных и не утерять данные о пользователей, которые уже давно не заходили в сеть или удалили свои аккаунты.
* Кнопку просмотра общей таблицы, включающая в себя представление из всех данных, связанных между собой, чтобы было проще находить нужный нам для себя фильм, по названию, жанру, актёру, режиссёру и т.д.
* Вспомогательную кнопку «Просмотр пользователей с определённой подпиской», включающая в себя вспомогательную форму, которая содержит в себе хранимую процедуру, информацию о пользователей с определённой подпиской, подписку нужно указать.
* Вспомогательную кнопку «Подсчёт пользователей с определённой подпиской», включающая в себя вспомогательную форму, которая содержит в себе скалярную функцию, информацию о количестве пользователей, которые купили определённую подписку, подписку нужно указать.
* Вспомогательную кнопку «Просмотры пользователей», включающая в себя вспомогательную форму, которая содержит в себе представление с информацией о просмотрах, т.е. кто из пользователей посмотрел фильмы.
* Вспомогательную кнопку «плюс к стоимости подписок», включающая в себя вспомогательную форму, которая содержит в себе функцию с обновлением данных в таблице «Подписка», изменяет стоимость подписок на определённую сумму, сумму нужно указать.
* Кнопку «Выйти», которая позволит выйти из приложения.

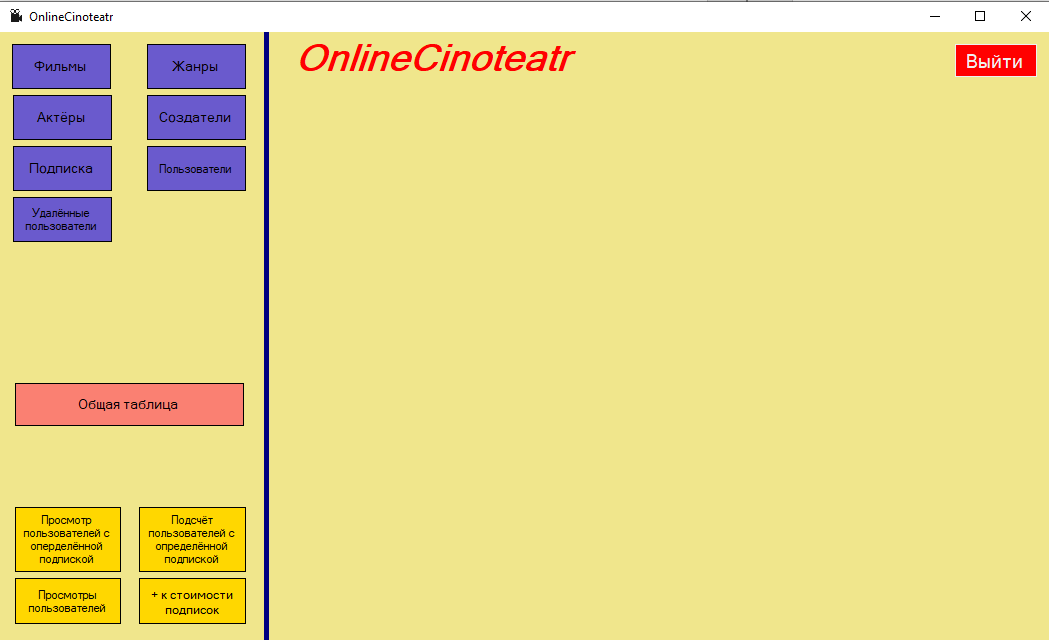


Рисунок 33 – Главная форма (администратор)

Для просмотра данных из таблиц базы данных, нужно нажать на любую кнопку просмотра. Таблицы для просмотра идентичны таблицам в форме клиента, но клиент не имеет возможность просматривать таблицы «Пользователи» и «Удалённые пользователи».

Просмотр таблиц базы данных представлен на рисунке 34.

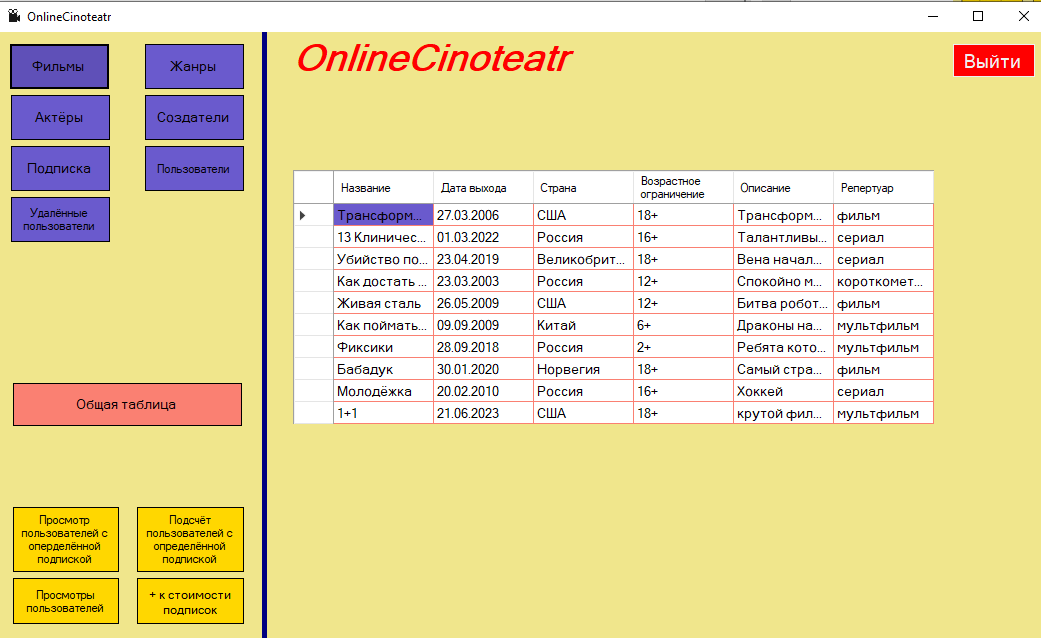


Рисунок 34 – Просмотр таблицы «Фильмы»

При просмотре таблиц, таких как: фильмы, жанры, актёры, создатели, подписка и пользователи правым кликом мыши можно вызвать контекстное меню, с помощью которого можно редактировать записи в таблице, а т.е. удалять, добавлять и изменять. Вызов контекстного меню показан на рисунке 35.

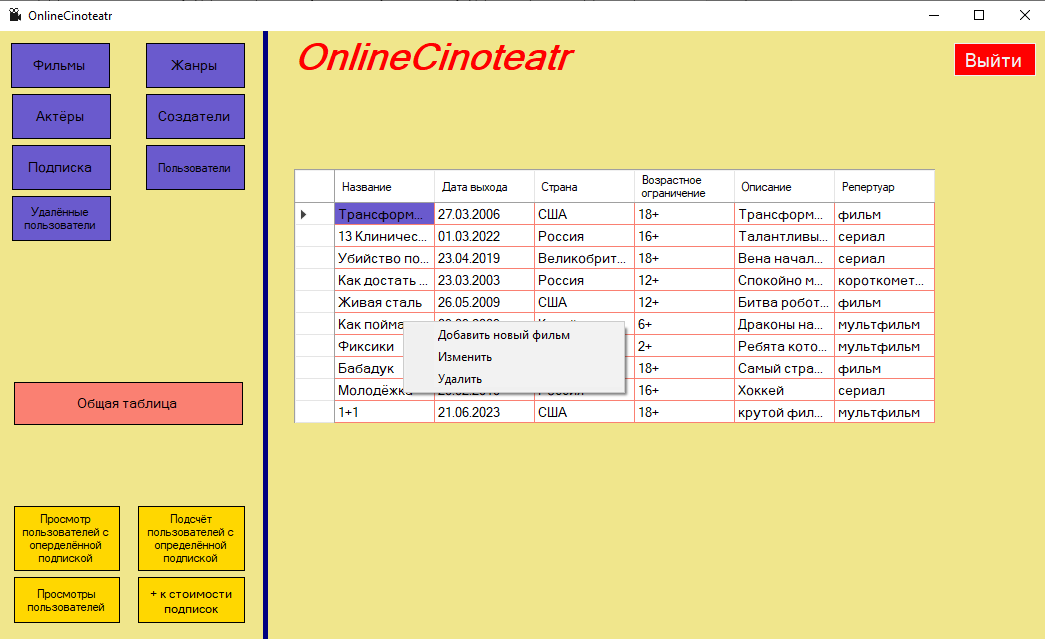


Рисунок 35 – Вызов контекстного меню

Вызывать контекстное меню может только пользователь, авторизовавшийся в качестве администратора.

Для добавления и изменения записей в таблице были разработаны вспомогательные формы, которые упрощают процесс добавления и изменения данных в таблице.

Форма добавления фильма представлена на рисунке 36. Открыв её, можно увидеть поля для ввода названия фильма, дата выходы, страна создания фильма, возрастное ограничение, описание, а также репертуар. Добавлять и изменять фильмы может только пользователь, авторизовавшийся, как администратор.

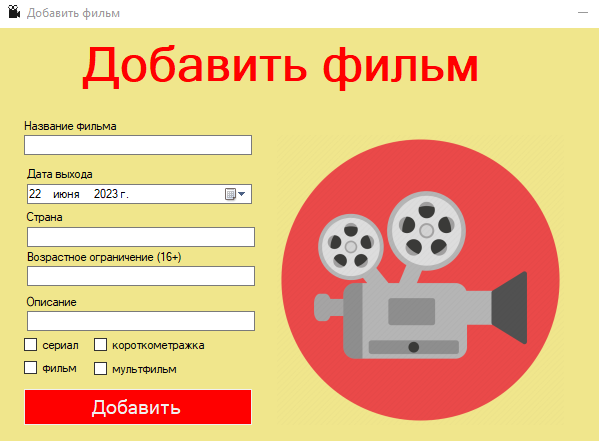


Рисунок 36 – Форма добавления фильма

Форма изменения данных о фильме аналогична форме добавления, но при открытии она заполняется информацией, которую можно редактировать. Форма изменения данных о фильме изображена на рисунке 37.

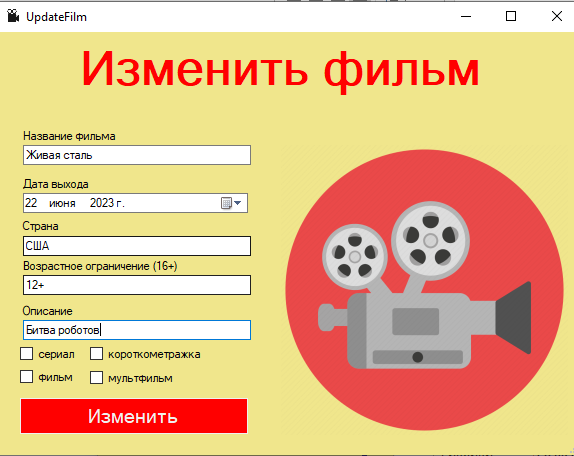


Рисунок 37 – Форма изменения фильма

При нажатии на кнопку «Общая таблица» появляется возможность выполнить поиск по таблице со всей информацией. Изображение общей таблицы показано на рисунке 38.

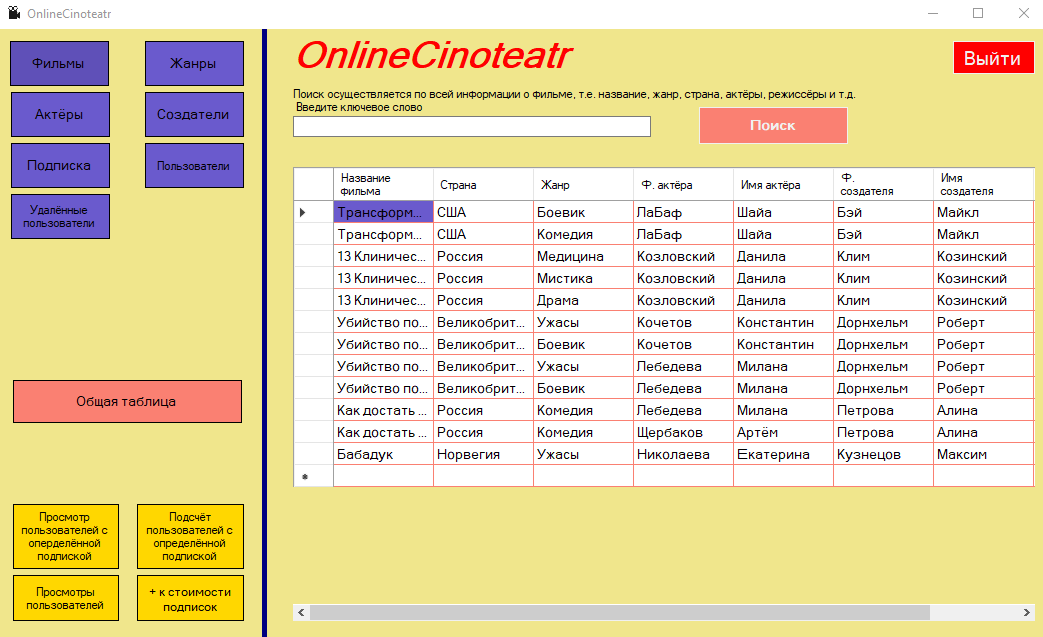


Рисунок 38 – Общая таблица

Поиском могут воспользоваться любые пользователи, которые авторизовались в клиентское приложение.

При вводе в поле определённое ключевое слово и при нажатии на кнопку «Поиск», происходит поиск по таблице. Изображение поиска предоставлено на рисунке 39.

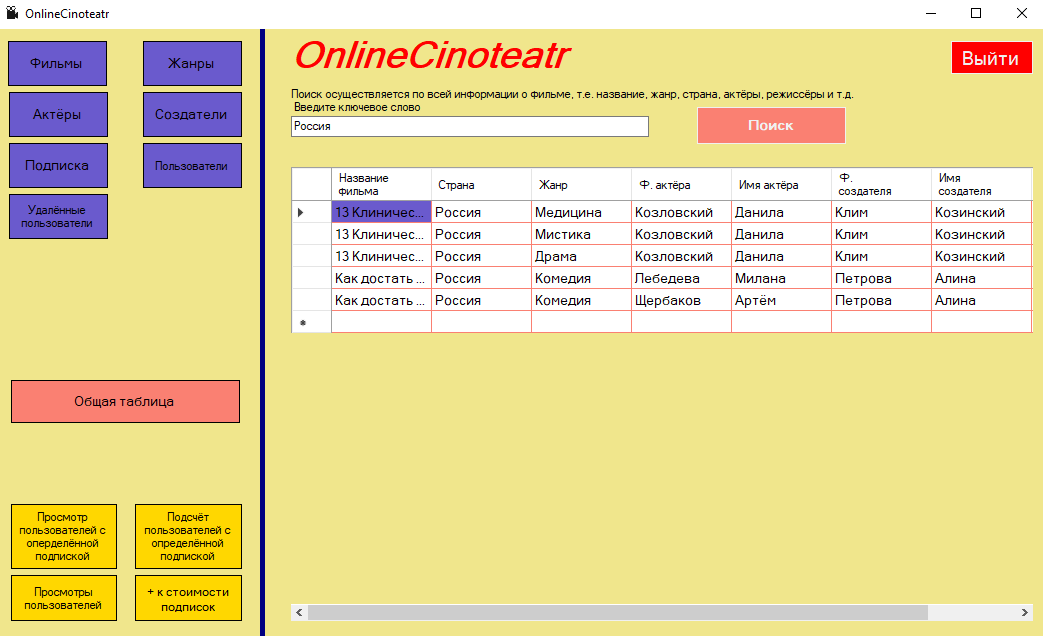


Рисунок 39 – Поиск

Для быстрого поиска нужной информации по записям базы данных были реализованы четыре вспомогательные кнопки, которые доступны только пользователю администратору.

При нажатии на вспомогательную кнопку «Просмотр пользователей с определённой подпиской» открывается вспомогательная форма с информацией хранимой процедуры. Вспомогательная форма с информацией хранимой процедуры представлена на рисунке 40.

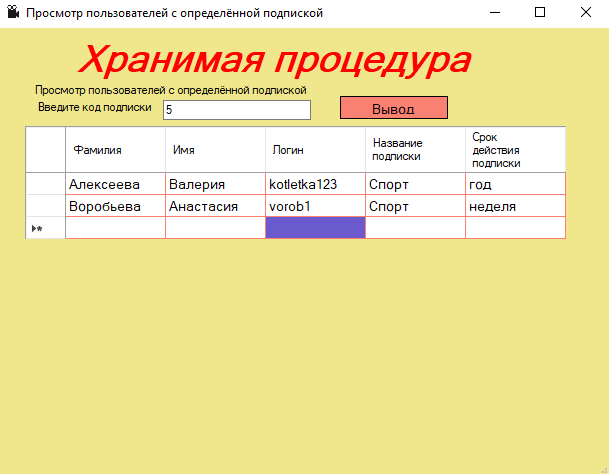


Рисунок 40 – Вспомогательная форма «Хранимая процедура»

В данной форме находится поле для ввода кода подписки из базы данных и кнопка «Вывод», после ввода и нажатии на кнопку, появляется таблица с пользователями, которые купили данную подписку.

При нажатии на вспомогательную кнопку «Подсчёт пользователей с определённой подпиской» открывается вспомогательная форма с информацией скалярной функции. Вспомогательная форма с информацией скалярной функции предоставлена на рисунке 41.

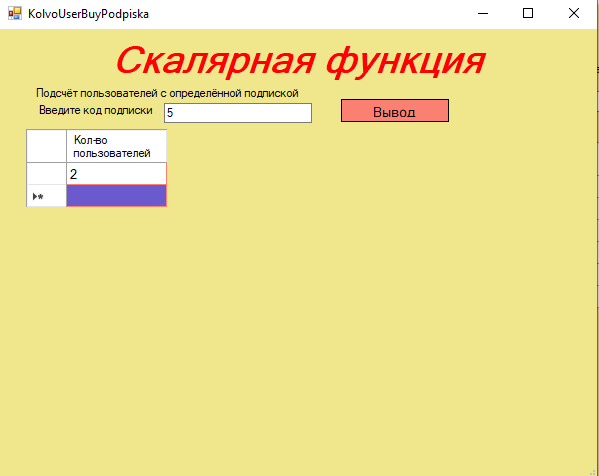


Рисунок 41 – Вспомогательная форма «Скалярная функция»

В данной форме находится поле для ввода кода подписки из базы данных и кнопка «Вывод», после ввода и нажатии на кнопку, появляется таблица с числом пользователей, которые купили данную подписку.

При нажатии на вспомогательную кнопку «Просмотры пользователей» открывается вспомогательная форма с информацией о представлении. Вспомогательная форма с информацией о представлении представлена на рисунке 42.

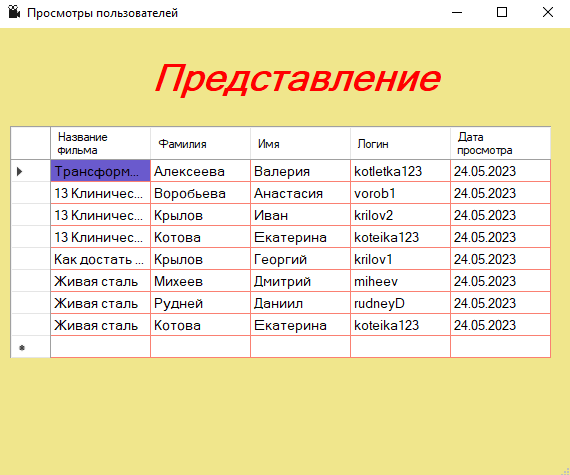


Рисунок 42 – Вспомогательная форма «Представление»

Данная форма содержит в себе только таблицу с информацией о пользователях, которые посмотрели тот или иной фильм и когда посмотрели.

При нажатии на вспомогательную кнопку «плюс к стоимости подписки» открывается вспомогательная форма, в которой можно изменить стоимость всех подписок, указав в поле сумму и нажав на кнопку обновить. Изображение вспомогательной формы обновления данных представлено на рисунке 43.

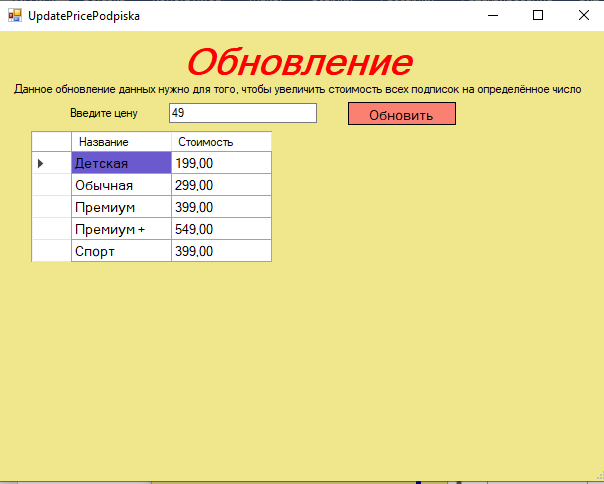


Рисунок 43 – Вспомогательная форма «Обновление»

При авторизации пользователем в качестве клиента с главной формы пропадают вспомогательные кнопки и некоторые кнопки для просмотра, а также клиенту невозможно вызвать контекстное меню для редактирования записей в таблицах. Изображение главной формы представлено на рисунке 44.



Рисунок 44 – Вспомогательная форма «Представление»

Главная форма с функционалом для пользователя, который авторизовался в качестве клиента, содержит в себе:

* Кнопку просмотра фильмов, включающая в себя таблицу для просмотра фильмов.
* Кнопку просмотра жанров, включающая в себя таблицу для просмотра жанров.
* Кнопку просмотра актёров, включающая в себя таблицу для просмотра актёров.
* Кнопку просмотра создателей, включающая в себя таблицу для просмотра создателей.
* Кнопку просмотра подписок, включающая в себя таблицу для просмотра подписок.
* Кнопку просмотра общей таблицы, включающая в себя представление из всех данных, связанных между собой, чтобы было проще находить нужный нам для себя фильм, по названию, жанру, актёру, режиссёру и т.д.
* Кнопку «Выйти», которая позволит выйти из приложения.

9 Тестирование разработанного функционала

**Тестирование программного обеспечения** — процесс исследования программного обеспечения (ПО) с целью получения информации о качестве продукта.

Дефект (Баг) – расхождение ожидаемого и фактического результатов.

Вид тестирования – совокупность активностей, направленных на тестирование заданных характеристик системы или её части, основанная на конкретных целях.

По запуску кода на исполнения тестирование программного обеспечения разделяют на:

* Статическое тестирование – процесс тестирования, который проводится для верификации практически любого артефакта разработки: программного кода компонент, требований, системных спецификаций, функциональных спецификаций, документов проектирования и архитектуры программных систем и их компонентов.
* Динамическое тестирование – тестирование проводится на работающей системе, не может быть осуществлено без запуска программного кода приложения.

По доступу к коду и архитектуре тестирование программного обеспечения делят на:

* Тестирование белого ящика – метод тестирования ПО, который предполагает полный доступ к коду проекта.
* Тестирование серого ящика – метод тестирования ПО, который предполагает частичный доступ к коду проекта (комбинация White Box и Black Box методов).
* Тестирование чёрного ящика – метод тестирования ПО, который не предполагает доступа (полного или частичного) к системе. Основывается на работе исключительно с внешним интерфейсом тестируемой системы.

По уровню детализации приложения тестирование программного обеспечения делиться на:

* Модульное тестирование – проводится для тестирования какого-либо одного логически выделенного и изолированного элемента (модуля) системы в коде. Проводится самими разработчиками, так как предполагает полный доступ к коду.
* Интеграционное тестирование – тестирование, направленное на проверку корректности взаимодействия нескольких модулей, объединенных в единое целое.
* Системное тестирование – процесс тестирования системы, на котором проводится не только функциональное тестирование, но и оценка характеристик качества системы — ее устойчивости, надежности, безопасности и производительности.
* Приёмочное тестирование – проверяет соответствие системы потребностям, требованиям и бизнес-процессам пользователя.

По принципам работы с приложением тестирование программного обеспечения разделяют на:

* Позитивное тестирование — тестирование, при котором используются только корректные данные.
* Негативное тестирование — тестирование приложения, при котором используются некорректные данные и выполняются некорректные операции.

По степени автоматизации тестирования программного обеспечения разделяют на:

* Ручное тестирование.
* Автоматизированное тестирование.

В зависимости от исполнителей тестирование программного обеспечения подразделяют на:

* Альфа-тестирование – является ранней версией программного продукта. Может выполняться внутри организации-разработчика с возможным частичным привлечением конечных пользователей.

Бета-тестирование - программное обеспечение, выпускаемое для ограниченного количества пользователей. Главная цель — получить отзывы клиентов о продукте и внести соответствующие изменения.

9.1 Авторизация под существующими учетными данными

Для проверки работоспособности приложения при авторизации под существующими учетными данными было произведено тестирование ПО, результат тестирования представлен в таблице 4.

Таблица 4 – Тестирование авторизации под существующими данными

|  |  |
| --- | --- |
| Входные данные | Логин: Krukov\_Admin Пароль: Krukov\_Admin |
| Ожидаемый результат | Открытие главной формы с функционалом администратора |

Продолжение таблицы 4

|  |  |
| --- | --- |
| Фактический результат | Открытие главной формы с функционалом администратора |

В ходе проведения тестирования программного обеспечения по авторизации под существующими данными расхождение ожидаемого и фактического результата не было установлено.

9.2 Авторизация под несуществующими учетными данными

Для проверки работоспособности приложения при авторизации под несуществующими учетными данными было произведено тестирование ПО, результат тестирования представлен в таблице 5.

Таблица 5 – Тестирование авторизации под несуществующими данными

|  |  |
| --- | --- |
| Входные данные | Логин: Kotleta Пароль: Krukov\_Admin |

Продолжение таблицы 5

|  |  |
| --- | --- |
| Ожидаемый результат | Сообщение об ошибке |
| Фактический результат | Сообщение об ошибке |

В ходе проведения тестирования программного обеспечения по авторизации под несуществующими данными расхождение ожидаемого и фактического результата не было установлено.

9.3 Валидация на обязательность заполняемых полей при создании и редактировании записей

Для проверки работоспособности приложения при вводе данных в формы было произведено тестирование программного обеспечения, результат тестирования представлен в таблице 6.

Таблица 6 – Тестирование ввода данных в формы

|  |  |
| --- | --- |
| Входные данные | Название фильма: -, дата выхода: 22.06.2023, страна: Россия, возрастное ограничение: 14+, описание: Привет, репертуар: фильм. |
| Ожидаемый результат | Сообщение об ошибке |
| Фактический результат | Сообщение об ошибке |

В ходе проведения тестирования программного обеспечения при вводе данных в формы расхождение ожидаемого и фактического результата не было установлено.

9.4 Просмотр записей из разных таблиц

Для проверки работоспособности просмотра записей из разных таблиц было проведено тестирование программного обеспечения, результат тестирования представлен в таблицах 7-9.

Таблица 7 – Тестирование просмотра записей в таблице фильмы

|  |  |
| --- | --- |
| Входные данные | Кнопка просмотра: «Фильмы» |
| Ожидаемый результат | Открытие таблицы с информацией о фильмах |

Продолжение таблицы 7

|  |  |
| --- | --- |
| Фактический результат | Открытие таблицы с информацией о фильмах |

Таблица 8 – Тестирование просмотра записей в таблице жанры

|  |  |
| --- | --- |
| Входные данные | Кнопка просмотра: «Жанры» |
| Ожидаемый результат | Открытие таблицы с информацией о жанрах |
| Фактический результат | Открытие таблицы с информацией о жанрах |

Таблица 9 – Тестирование просмотра записей в таблице актёры

|  |  |
| --- | --- |
| Входные данные | Кнопка просмотра: «Актёры» |
| Ожидаемый результат | Открытие таблицы с информацией об актёрах |

Продолжение таблицы 9

|  |  |
| --- | --- |
| Фактический результат | Открытие таблицы с информацией об актёрах |

В ходе проведения тестирования программного обеспечения по просмотру записей в разных таблицах расхождение ожидаемого и фактического результата не было установлено.

9.5 Удаление записей из разных таблиц

Для проверки работоспособности оповещения об удалении записей из разных таблиц было проведено тестирование программного обеспечения, результат тестирования представлен в таблице 10.

Таблица 10 – Тестирование удаления записей в таблице техники

|  |  |
| --- | --- |
| Входные данные | Наименование жанра: Драма |
| Ожидаемый результат | Открытие сообщения информации об удалении |

Продолжение таблицы 10

|  |  |
| --- | --- |
| Фактический результат | Открытие сообщения информации об удалении |

В ходе проведения тестирования программного обеспечения по удалению записей в разных таблицах расхождение ожидаемого и фактического результата не было установлено.

Заключение

В ходе выполнения производственной практики была изучена предметная область онлайн кинотеатров для создания базы данных и последующей её внедрения в пользовательское приложение.

Целью производственной практики было разработать базу данных на тему «Онлайн кинотеатр» и разработать с интеграцией клиентское приложение.

Цель была выполнена. Основные задачи работы: создание ER-модели реляционной модели данных, создание диаграммы Мартина, создание хранимой процедуры, представления, скалярной функции, триггеров и разработка клиентского приложение с интеграцией ранее созданной базы данных.

В процессе выполнения производственной практики были закреплены навыки разработки базы данных на Microsoft SQL Server, а также создания клиентского приложения используя Entity Framework [7].

Разработанная база данных может облегчить поиск фильмов пользователям интернета.

Список использованных источников

1. SQL за 20 минут // Proglib URL: https://proglib.io/p/sql-for-20-minutes (дата обращения 01.05.2023)
2. Ефимова М. З., Зуйкова А. Как устроен язык SQL и почему он так востребован // Practicum Yandex URL: https://practicum.yandex.ru/blog/chto-takoe-sql/ (дата обращения 01.05.2023)
3. Описание основных приемов нормализации базы данных // Microsoft URL: https://learn.microsoft.com/ru-ru/office/troubleshoot/access/database-normalization-description (дата обращения: 04.05.2023).
4. Руководство по созданию хранимых процедур // Microsoft URL: https://learn.microsoft.com/ru-ru/sql/relational-databases/stored-procedures/stored-procedures-database-engine?view=sql-server-linux-ver15 (дата обращения 04.05.2023)
5. Руководство по созданию скалярных функций // Skeitol URL: https://skeitol.ru/articles/hranimye\_funkcii\_transact\_sql/ (дата обращения 06.05.2023)
6. Руководство по представлении триггеров в MS SQL SERVER 2022 // Metanit URL: https://metanit.com/sql/sqlserver/12.1.php (дата обращения 09.05.2023)
7. Обзор Entity Framework Core – EF Core // Microsoft learn URL: https://learn.microsoft.com/ru-ru/ef/core/ (дата обращения 30.05.2023)

Приложение А

-- Создание базы данных с названием "OnlineCinoteatr"

CREATE DATABASE OnlineCinoteatr

USE OnlineCinoteatr

-- Создание таблицы Film

CREATE TABLE Film (

ID\_Filma INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

FilmName VARCHAR(100) NOT NULL,

DataVihoda DATE DEFAULT GETDATE() NOT NULL,

Country VARCHAR(50) NOT NULL,

AgeRestriction VARCHAR(5) NOT NULL,

FilmDescription VARCHAR(1000) NOT NULL,

Repertuar VARCHAR(25) NOT NULL, CHECK(Repertuar IN ('фильм', 'мультфильм', 'сериал', 'короткометражка')),

);

-- Создание таблицы Zhanr

CREATE TABLE Zhanr (

ID\_Zhanra INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

ZhanrName VARCHAR(50) NOT NULL,

);

-- Создание таблицы Actor

CREATE TABLE ACTOR (

ID\_Actor INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

Surname VARCHAR(50) NOT NULL,

FirstName VARCHAR(50) NOT NULL,

MiddleName VARCHAR(50),

KolichestvoFilmov INT, CHECK (KolichestvoFilmov > 0),

Birthday DATETIME DEFAULT GETDATE(),

);

-- Создание таблицы Builder

CREATE TABLE Builder (

ID\_Builder INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

Surname VARCHAR(50) NOT NULL,

FirstName VARCHAR(50) NOT NULL,

MiddleName VARCHAR(50),

KolichestvoFilmov INT, CHECK (KolichestvoFilmov > 0),

Birthday DATETIME DEFAULT GETDATE(),

PhoneNumber VARCHAR(12), CHECK (PhoneNumber LIKE '+[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'),

Post VARCHAR(50) NOT NULL,

);

-- Создание таблицы User

CREATE TABLE Users (

ID\_Usera INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

Surname VARCHAR(50) NOT NULL,

FirstName VARCHAR(50) NOT NULL,

MiddleName VARCHAR(50),

UserLogin VARCHAR(50) NOT NULL,

UserPassword VARCHAR(50) NOT NULL,

Birthday DATETIME DEFAULT GETDATE(),

Email VARCHAR(50) NOT NULL,

);

-- Создание таблицы Podpiska

CREATE TABLE Podpiska (

ID\_Podpiska INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

NamePodpiska VARCHAR(50) NOT NULL,

Price DECIMAL(10,2),

);

-- Создание ассоциативных таблиц и опредление связей между таблицами

-- Создание таблицы FilmZhanr

CREATE TABLE FilmZhanr (

ID\_FilmZhanr INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

ID\_Filma INT NOT NULL,

ID\_Zhanra INT NOT NULL,

CONSTRAINT fk\_FilmZhanr\_Film FOREIGN KEY (ID\_Filma) References Film

ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT fk\_FilmZhanr\_Zhanr FOREIGN KEY (ID\_Zhanra) References Zhanr

ON UPDATE CASCADE ON DELETE CASCADE,

);

-- Создание таблицы FilmActor

CREATE TABLE FilmActor (

ID\_FilmActor INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

ID\_Filma INT NOT NULL,

ID\_Actora INT NOT NULL,

CONSTRAINT fk\_FilmActor\_Film FOREIGN KEY (ID\_Filma) References Film

ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT fk\_FilmActor\_Actor FOREIGN KEY (ID\_Actora) References Actor

ON UPDATE CASCADE ON DELETE CASCADE,

);

-- Создание таблицы FilmBuilder

CREATE TABLE FilmBuilder (

ID\_FilmBuilder INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

ID\_Filma INT NOT NULL,

ID\_Builder INT NOT NULL,

CONSTRAINT fk\_FilmBuilder\_Film FOREIGN KEY (ID\_Filma) References Film

ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT fk\_FilmBuilder\_Builder FOREIGN KEY (ID\_Builder) References Builder

ON UPDATE CASCADE ON DELETE CASCADE,

);

-- Создание таблицы ViewsFilmsUsers

CREATE TABLE ViewsFilmsUsers (

ID\_Views INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

ID\_Filma INT NOT NULL,

ID\_Usera INT NOT NULL,

DataProsmotra DATETIME DEFAULT GETDATE() NOT NULL,

CONSTRAINT fk\_ViewsFilmsUsers\_Film FOREIGN KEY (ID\_Filma) References Film

ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT fk\_ViewsFilmsUsers\_Users FOREIGN KEY (ID\_Usera) References Users

ON UPDATE CASCADE ON DELETE CASCADE,

);

-- Создание таблицы UserPodpiska

CREATE TABLE UserPodpiska (

ID\_UserPodpiska INT IDENTITY(1,1) NOT NULL PRIMARY KEY,

ID\_Usera INT NOT NULL,

ID\_Podpiska INT NOT NULL,

SrokDeistvia VARCHAR(50), CHECK (SrokDeistvia IN ('неделя','месяц','год','неограничено')),

CONSTRAINT fk\_UserPodpiska\_Podpiska FOREIGN KEY (ID\_Podpiska) References Podpiska

ON UPDATE CASCADE ON DELETE CASCADE,

CONSTRAINT fk\_UserPodpiska\_Users FOREIGN KEY (ID\_Usera) References Users

ON UPDATE CASCADE ON DELETE CASCADE,

);

Приложение Б

Создание служебной таблицы

CREATE TABLE DeleteUsers

(

ID\_DELETED INT IDENTITY(1,1) PRIMARY KEY NOT NULL,

Users\_Surname VARCHAR(50) NOT NULL,

Users\_Name VARCHAR(50) NOT NULL,

Users\_Login VARCHAR(50) NOT NULL,

Date\_Deleted DATETIME

);

Создание триггера UsersDelete

GO

CREATE TRIGGER UsersDelete ON Users

AFTER DELETE

AS

BEGIN

INSERT INTO DeleteUsers(Users\_Surname, Users\_Name, Users\_Login, Date\_Deleted)

SELECT deleted.Surname 'Фамилия', deleted.FirstName 'Имя', deleted.UserLogin 'Логин', GETDATE() AS 'Дата удаления'

FROM deleted

END

Запрос хранимой процедуры

USE OnlineCinoteatr

GO

CREATE PROCEDURE CheckUserPodpiska

@idPodpiska INT

AS

BEGIN

SELECT us.Surname 'Фамилия', us.FirstName 'Имя', us.UserLogin 'Логин', p.NamePodpiska 'Название подписки', up.SrokDeistvia 'Срок действия подписки'

FROM Users us, Podpiska p, UserPodpiska up

WHERE us.ID\_Usera = up.ID\_Usera AND

p.ID\_Podpiska = up.ID\_Podpiska AND

p.ID\_Podpiska = @idPodpiska

END

GO

EXEC CheckUserPodpiska 2

Запрос скалярной функции

USE OnlineCinoteatr

GO

CREATE FUNCTION dbo.KolvoUserBuyPodpiska

(@idPodpiska INT)

RETURNS INT

BEGIN

RETURN (SELECT COUNT(1)

FROM Users us, Podpiska p, UserPodpiska up

WHERE us.ID\_Usera = up.ID\_Usera AND

p.ID\_Podpiska = up.ID\_Podpiska AND

p.ID\_Podpiska = @idPodpiska)

END

Запрос представления

USE OnlineCinoteatr

GO

CREATE VIEW UsersViews

AS

SELECT f.FilmName 'Название фильма', us.Surname 'Фамилия', us.FirstName 'Имя',

us.UserLogin 'Логин', vws.DataProsmotra 'Дата просмотра'

FROM Users us, Film f, ViewsFilmsUsers vws

WHERE us.ID\_Usera = vws.ID\_Usera AND

f.ID\_Filma = vws.ID\_Filma

Приложение В

USE OnlineCinoteatr

GO

EXEC sp\_addrole 'Administrator'

EXEC sp\_addrole 'ClientOC'

GO

GRANT DELETE, INSERT, UPDATE, SELECT ON Actor TO Administrator

GRANT DELETE, INSERT, UPDATE, SELECT ON Builder TO Administrator

GRANT DELETE, INSERT, UPDATE, SELECT ON Film TO Administrator

GRANT DELETE, INSERT, UPDATE, SELECT ON FilmActor TO Administrator

GRANT DELETE, INSERT, UPDATE, SELECT ON FilmBuilder TO Administrator

GRANT DELETE, INSERT, UPDATE, SELECT ON FilmZhanr TO Administrator

GRANT DELETE, INSERT, UPDATE, SELECT ON Podpiska TO Administrator

GRANT DELETE, INSERT, UPDATE, SELECT ON UserPodpiska TO Administrator

GRANT DELETE, INSERT, UPDATE, SELECT ON Users TO Administrator

GRANT DELETE, INSERT, UPDATE, SELECT ON ViewsFilmsUsers TO Administrator

GRANT DELETE, INSERT, UPDATE, SELECT ON Zhanr TO Administrator

GO

GRANT SELECT ON Actor TO ClientOC

GRANT SELECT ON Film TO ClientOC

GRANT SELECT ON Builder TO ClientOC

GRANT SELECT ON Podpiska TO ClientOC

GRANT SELECT ON Zhanr TO ClientOC

GO

EXEC sp\_addlogin 'Krukov\_Admin', 'Krukov\_Admin', 'OnlineCinoteatr'

USE OnlineCinoteatr

EXEC sp\_adduser 'Krukov\_Admin', 'Krukov\_Admin'

EXEC sp\_addrolemember 'Administrator', 'Krukov\_Admin'

GO

EXEC sp\_addlogin 'Godovikov\_Client', 'Godovikov\_Client', 'OnlineCinoteatr'

USE OnlineCinoteatr

EXEC sp\_adduser 'Godovikov\_Client', 'Godovikov\_Client'

EXEC sp\_addrolemember 'ClientOC', 'Godovikov\_Client'