МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ  
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕХНОЛОГИЙ И УПРАВЛЕНИЯ ИМЕНИ К.Г. РАЗУМОВСКОГО (ПЕРВЫЙ КАЗАЧИЙ УНИВЕРСИТЕТ)»  
(ФГБОУ ВО «МГУТУ ИМ. К.Г. РАЗУМОВСКОГО (ПКУ)»)  
УНИВЕРСИТЕТСКИЙ КОЛЛЕДЖ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Специальность 09.02.03 Программирование в компьютерных системах

КУРСОВОЙ ПРОЕКТ

МДК 02. Разработка и администрирование баз данных

МДК 02.02. Технология разработки и защиты баз данных

на тему: «Разработка информационной системы туристической фирмы»

Пояснительная записка

УКИТ.09.02.03.2019.403.18ПЗ

ПКС 305

|  |  |  |
| --- | --- | --- |
| Исполнитель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись) | Кошолапова А.Ю. |
| Руководитель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись) | Ларионова Е. А. |
| Норм контроль | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись) | Гусева Е. Л. |

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc27305809)

[1 ОБЩАЯ ЧАСТЬ 4](#_Toc27305810)

[1.1 Назначения и цели создания 4](#_Toc27305811)

[1.2 Обзор и анализ предметной области 4](#_Toc27305812)

[1.3 Жизненный цикл базы данных 6](#_Toc27305813)

[1.4 Выбор и характеристика СУБД 7](#_Toc27305814)

[1.5 Выбор и характеристика среды разработки приложения 8](#_Toc27305815)

[2 СПЕЦИАЛЬНАЯ ЧАСТЬ 10](#_Toc27305816)

[2.1 Определение основных задач, приводящих к цели 10](#_Toc27305817)

[2.2 Архитектура информационной системы 10](#_Toc27305818)

[2.3 Логическая модель базы данных 11](#_Toc27305819)

[2.4 Нормализация таблиц 11](#_Toc27305820)

[2.5 Описание таблиц 13](#_Toc27305821)

[2.6 Другие объекты базы данных 15](#_Toc27305822)

[2.7 Разработка приложения 15](#_Toc27305823)

[2.8 Инструкция пользователю 17](#_Toc27305824)

[ЗАКЛЮЧЕНИЕ 20](#_Toc27305825)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 21](#_Toc27305826)

[ПРИЛОЖЕНИЕ А](#_Toc27305827) [SQL скрипты 23](#_Toc27305828)

[ПРИЛОЖЕНИЕ Б](#_Toc27305829) [Код программы 30](#_Toc27305830)

ВВЕДЕНИЕ

Развитие рыночных отношений в экономике и научно-технический прогресс ускорили темпы внедрения во все сферы социально-экономической жизни общества последних достижений в области информационных технологий. Информация в современном мире превратилась в один из наиболее важных ресурсов, а информационные системы стали необходимым инструментом практически во всех сферах деятельности человека. Информационная система стала неотъемлемой частью функционирования практически любой организации, поэтому нет необходимости обсуждать вопрос актуальности разработки и внедрения информационных систем. Однако вопрос системности подходов к ее проектированию и качества разработки до сих пор является актуальным.

Туризм – прибыльная и доходная сфера, тесно взаимодействующая со многими другими отраслями и способствующая их развитию.

Актуальность настоящего исследования заключается в том, что разработка и внедрение автоматизированной системы работы с клиентами турфирмы предопределит успешную реализацию качественного туристического продукта потребителю и явится одним из основных источников существования туристического предприятия.

Поэтому целью курсового проекта является выполнение разработки приложения по автоматизированию туристической фирмы.

К основным задачам, которые нужно выполнить в ходе проекта, относится:

* освоение тонкостей проектирования баз данных,
* изучение Entity Framework,
* разработка функций для работы с базой данных,
* создание функций для работы с данными внутри приложения,
* проектирование интерфейса программы,
* написание функций для корректного сохранения и обновления данных,
* моделирование .

1. ОБЩАЯ ЧАСТЬ
   1. Назначения и цели создания

База данных информационной системы туристической фирмы предназначена для хранение и удобного использования пользовательских данных.

Основной целью является получение опыта в разработке и защите систем, в основе которых лежит база данных, а также

* 1. Обзор и анализ предметной области
     1. Аннотация

Невозможно представить крупную систему, которая не использовала бы базы данных для хранения информации. Сервисы по истории представляют собой отражение базы данных для клиента в удобном виде с дополнительным клиентским функционалом. Почему не может заменить систему по хранению информации таблицы exel, какой функционал имеется в уже реализованной системе и многое другое описано в данном разделе.

* + 1. Анализ сайта https://tourist.tez-tour.com/

Система tez-tour – пример хорошей реализации базы данных туристической фирмы. Основное меню состоит из из выбор тура, сортировки по датам, странам, видам отдыха, также тут сразу можно выбрать авиабилеты для запланированного отдыха (Рисунок 1). Для реализации подобного функционала необходимо задействовать больше времени, чем на курсовой проект. Также подобные проекты разрабатываются не в одиночку и не силами одних программистов, а используется целый ряд специалистов из различных областей

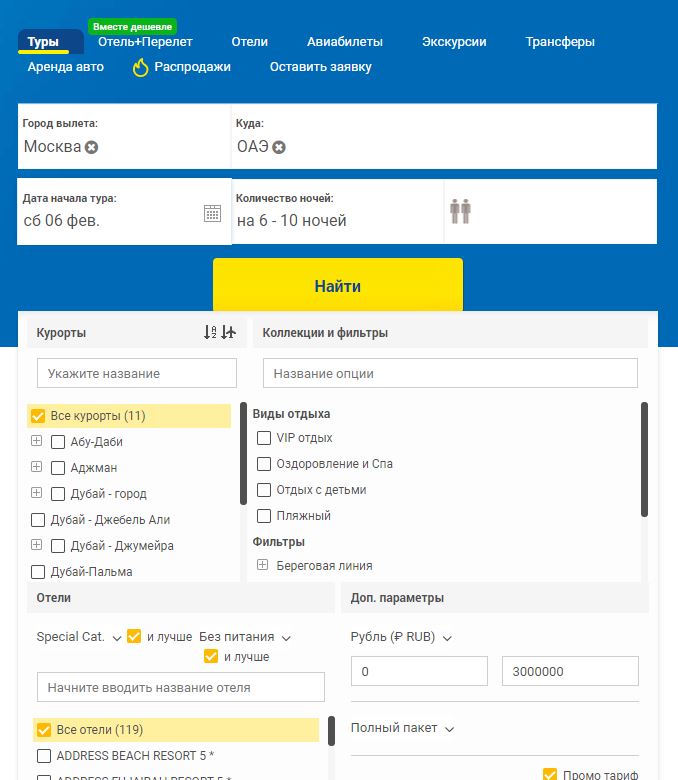


Рисунок 1 – Система tez-tour

Данная система закрепилась на рынке и является примером системы для реализации курсового проекта.

* + 1. Анализ ведения истории просмотров в exсel

Excel представляет собой обычные таблицы, в которых может храниться и анализироваться информация. Для системы же необходимы не просто таблицы, а еще и их связи.

Весь необходимый функционал может быть реализован на макросах, а запись в таблицы и проверка данных через внешнее приложение. Но таким образом это будет реализация СУБД, что не является темой курсового проекта.

В excel нет возможности выборки данных, вставки данных с множеством проверок по типам и значения, а также по связям. Таким образом, можно сделать вывод, что для реализации курсового проекта exel не подходит.

* + 1. Выводы по анализируемым системам

Был выполнен анализ инструментов, на основе которых можно создать курсовой проект. Отличным вариантом является access, так как имеет весь необходимый функционал. Плохим же вариантом является excel, так как в нем нет необходимого для системы функционала – возможностей СУБД.

Также была рассмотрена система-сервис tez-tour, который является полностью реализованной системой. Разумеется, не может идти речи о системе, которая хоть как-то будет конкурировать с данным сервисом, так как это не является целью курсового проекта.

* 1. Жизненный цикл базы данных

Жизненный цикл – это процесс проектирования, реализации и управлением базой данных. Он состоит из:

* анализа предметной области, который включает в себя планирование разработки, определение требований, сбор и анализ требований,
* проектирование базы данных , которое состоит из создания концептуальной модели, логической модели и физической модели,
* реализации, в процессе которой пишутся скрипты на создание базы данных, связей сущностей в базе данных, а так же разработки интерфейса приложения и его функций.

1.3.1 Выделение основных объектов

Можно выделить два основных объекта:

* пользователь,
* тур,

Данные объекты имеют в себе поля и связанные таблицы, которые определяются предметной областью. Например, для одной предметной области нет смысла расписывать подробно всех актеров фильмов и производить связь актеров и фильмов. Такой предметной областью является система истории просмотров фильмов. Для другой предметной области, к примеру, системе подбора актеров, такие связи необходимы.

* 1. Выбор и характеристика СУБД
     1. Аннотация

В данном разделе производится анализ различных СУБД. Для реализации базы данных отлично подходит access, но какая СУБД в итоге выбрана и почему не access описано в данном разделе.

* + 1. Выбор СУБД

В качестве СУБД отлично подходит access. В данной СУБД есть возможность реализации полной системы, используя графический интерфейс, но она не подходит для данного курсового проекта, так как не позволит получить опыт в составлении запросов и практики создания систем с двухуровневой архитектуры.

Следовательно, выбор падает на ms sql сервер. Данная СУБД имеет больший серверный функционал, но уже отсутствует реализации создания клиентского функционала используя лишь ms sql сервер.

* + 1. Обоснование выбора СУБД

Для разработки базы данных была выбрана среда Microsoft SQL Server Management Studio 2018, так как она является удобной для использования и не требует долгого освоения тонкостей для создания полноценной базы данных под программу, а также с ней можно работать, используя Entity Framework. Эта СУБД использует язык SQL (T-SQL) для написания необходимых сущностей и заполнения их данными.

При сравнении SQL Server с MySQL можно выделить несколько главных различий:

* первый предназначен для работы с платформой .Net, а второй для работы с PHP,
* синтаксис,
* данные SQL Server хранятся в единой системе Microsoft, в то время как данные MySQL хранятся на движках.

Весомыми плюсами SQL Server являются простота понимания принципов работы, синтаксиса, а также возможность создавать объекты баз данных либо с помощью скрипта, либо вручную, с помощью конструктора.

* 1. Выбор и характеристика среды разработки приложения
     1. Аннотация

В данном разделе производится выбор сред разработки приложения и базы данных, а также производится его обоснование.

* + 1. Выбор системы программирования
       1. База данных

Для создания базы данных подходит ssms. Оно позволяет в графическом режиме создавать базы данных, таблицы, настраивать связи и генерировать скрипты создания, удаления и резервного копирования.

В данной среде есть множество ошибок, но также есть и пути, следуя которым, данные ошибки не будут вызваны и произведена корректная работа.

* + - 1. Приложение

Для разработки приложения была выбрана среда программирования Microsoft Visual Studio 2019, потому что эта среда разработки обеспечивает возможность создания приложений с удобным интерфейсом с помощью языка C# и системы построения клиентских приложений WPF.Net Framework.

Microsoft Visual Studio 2019 – продукт компании Microsoft, включающий в себя интегрированную среду разработки программного обеспечения, а такжеряд других инструментальных средств.

Некоторые преимущества:

* удобный поиск ошибок,
* подробная документация об ошибках, располагающаяся на сайте компании Microsoft,
* возможность создания специальной библиотеки для тестирования.

.

* + 1. Сравнение с другими системами программирования
       1. База данных

Наиболее удобной является среда разработки mySql workbench, но оно подходит для mysql сервера. Эти СУБД имеют множество различий в типах данных, так что данная среда не подходит.

Существует также Erwin, но в почти свободном доступе есть его старая версия, которая прихотлива к кодировкам, выставленным в системе и может не корректно работать при смене кодировок.

* + - 1. Приложение

Очень удобным является язык программирования php и среда разработки phpStorm. Данный язык имеет неплохую оптимизацию с 7й версии и поддержку. Преимущества по php сравнению с c#:

* кроссплатформенность,
* более высокая скорость работы (не используется прослойка – виртуальная машина),
* больше возможностей в качестве web backend (c# универсалтный, а php затачивался как язык для работы backend веба)

Но все преимущества меркнут в одном основном преимуществе c# - он используется для квалификационной работы.

* + 1. Выводы по выбранной системе программирования

Выбранные средства – ssms и visual studio напрямую зависят не от удобства, а от принятых стандартов и целей создания курсового проекта. Нет сомнений, что написание курсового проекта в данных средах значительно повысят уровень подготовки для квалификационной работы.

1. СПЕЦИАЛЬНАЯ ЧАСТЬ
   1. Определение основных задач, приводящих к цели

Постановка задачи. Спроектировать средствами MS SQL SERVER базу данных и создать средствами C# приложение, позволяющее хранить информацию о турах: Страна, количество дней, стоимость путёвки, дата начала тура, анкетные и паспортные данные туристов. Предусмотреть возможность добавления, удаления, редактирования записей, определения количества имеющихся путёвок на заданный тур в указанную дату, выдачи списка туристов, купивших путёвки на заданный тур. Предусмотреть выдачу отчётов. Обеспечить защиту базы данных и приложения.

В результате выполнения этих работ, будет получен необходимый опыт в проектировании, создании и защите баз данных.

* 1. Нормализация таблиц

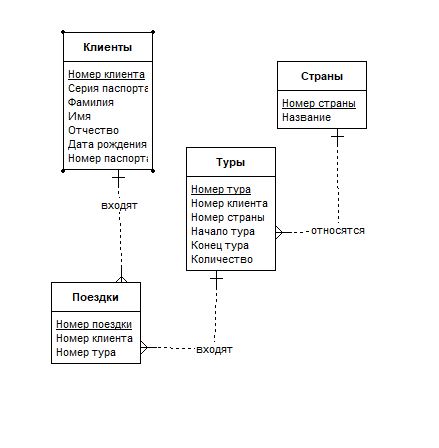


Рисунок 1 – Логическая модель базы данных

В данной базе была реализована связь многие ко многим между таблицами «Туры» и «Клиенты» с помощью третьей таблицы «Путевки»

* + 1. Физическая модель базы данных

На физической модели базы данных (Рисунок 2) можно определить обосновать нормализацию таблиц.

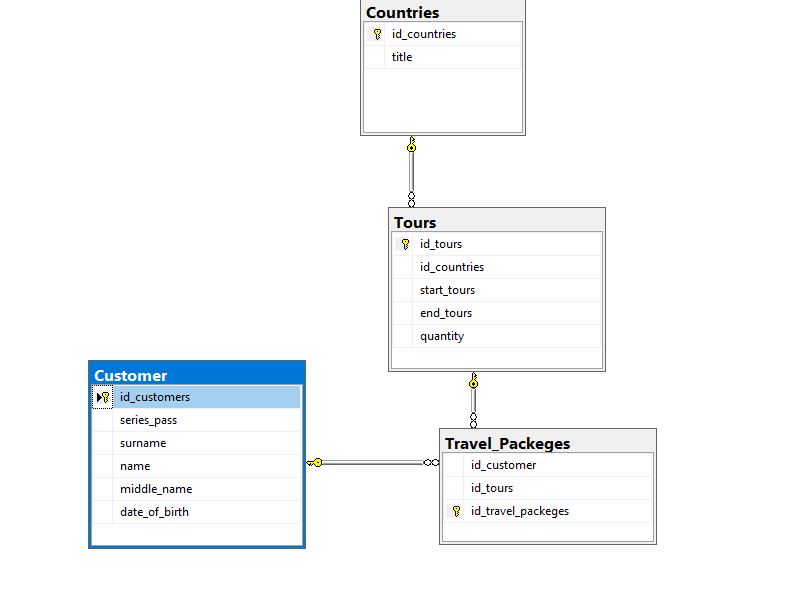


Рисунок 2 – Физическая модель базы данных

* + 1. Проверка нормальных форм таблицы фильмов

Представленные таблицы полностью соответствуют третьей нормальной форме, потому что в рамках данной предметной области нет смысла делить дату рождения в таблице «Клиенты», так как у нас нет задачи получить отдельно число, месяц и год. Так же мы видим, что в таблицах все атрибуты относятся к своим первичным ключам, это доказывает то , что таблицы находятся в третьей нормально форме.

* 1. Описание таблиц

Таблица «Customers»(Клиенты)

|  |  |  |  |
| --- | --- | --- | --- |
| Название в коде | Русское обозначение | Тип данных | Примечания |
| id\_customers | Номер клиента | Int | PK |
| series\_and\_number | Серия и номер паспорта | Int |  |
| surname | Фамилия | Nvarchar(50) |  |
| name | Имя | Nvarchar(50) |  |
| middle\_name | Отчество | Nvarchar(50) |  |
| date\_of\_birth | Дата рождения | date |  |

Таблица «Countries»(Страны)

|  |  |  |  |
| --- | --- | --- | --- |
| Название в коде | Русское обозначение | Тип данных | Примечания |
| id\_countries | Номер страны | Int | PK |
| title | Название | Varchar(50) |  |

Таблица «Tours»(Туры)

|  |  |  |  |
| --- | --- | --- | --- |
| Название в коде | Русское обозначение | Тип данных | Примечания |
| Id\_tours | Номер тура | Int | PK |
| Title\_tours | Название тура | Varchar(50) |  |
| id\_countries | Номер страны | Int | FK |
| start\_tour | Дата начала тура | Date |  |
| end\_tour | Дата окончания тура | Date |  |
| quantity | количество | Int |  |

Таблица «Travel\_Packeges»(Путевки)

|  |  |  |  |
| --- | --- | --- | --- |
| Название в коде | Русское обозначение | Тип данных | Примечания |
| id\_travel\_packeges | Номер путевки | Int | PK |
| id\_customers | Номер клиента | Int | FK |
| id\_tours | Номер тура | Int | FK |

* 1. Разработка приложения
     1. Функции приложения

Приложение имеет следующие функции:

* обеспечение клиентского интерфейса,
* взаимодействие с базой данных,
* валидация данных.

В рамках курсового проекта были реализованы следующие функции:

* получение данных для таблиц,
* проверка кода на инъекции,
* функция добавления тура,
  + 1. Требования к техническим средствам

Для функционирования системы подойдет любой компьютер с операционной системой windows 10.

* + 1. Требования к программным средствам

Для работы приложения необходимо следующее программное обеспечение:

* установленная платформа «.Net»,
* установленный ms sql сервер.
  + 1. Настройка информационной системы

Перед запуском системы необходимо исполнить скрипт «createDatabase.sql» для ms sql сервера, разрешить подключаться к базе данных по проверке подлинности widows. Данные манипуляции удобно производить в ssms.

ЗАКЛЮЧЕНИЕ

В процессе выполнения курсового проекта:

* проанализирована предметная область,
* произведен и обоснован подбор программных средств,
* создана физическая модель базы данных и реализована в ms sql сервере,
* обоснована третья форма одной из таблиц,
* расписаны таблицы базы данных с русским обозначением,
* произведена настройка ms sql сервера,
* разработано приложение,
* реализована валидация данных форм,
* продемонстрировано решение главной задачи.

Цели, поставленные в курсовой работе, были достигнуты не в полной мере. Параллельно с курсовой работой была произведена подготовка к квалификационной работе.

Реализованная система не может являться конкурентом уже существующим системам, но она является отличным опытом в разработке систем, основанных на клиент-серверной архитектуре.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. База знаний о взаимодействии с ms sql посредством c#. (2016). Web: https://o7planning.org/ru/10515/working-with-sql-server-database-using-csharp (дата обращения 15.11.2018).
2. Rob Miles. C# Programming Yellow Book. – М.: DCSUH. 2015.
3. База знаний о создании баз данных ms sql server. (2017). Web: https://docs.microsoft.com/ru-ru/sql/relational-databases/databases/create-a database?-View-sql-server-2016 (дата обращения 18.11.2018).
4. Артеменко Ю.Н. C# 7.0. Карманный справочник. – М.: «Альфа‑Книга». 2017.
5. База знаний работы с планировщиком задач в Windows. Web: https://public-pc.com/rabota-s-planirovshhikom-zadaniy-v-windows-10/ (дата обращения 20.11.2018).
6. База знаний о создании диаграмм в ms sql server. (2018). Web: https://remotehelper.ru/subd/ms-sql-server/ms-sql-server-kak-sozdat-diagrammu-bd-so-svyazyami (дата обращения 21.11.2018).
7. База знаний о ролях ms sql server. (2015). Web: https://professorweb.ru/my/sql-server/2012/level3/3\_12.php (дата обращения 23.11.2018).
8. База знаний об ограничении целостности данных ms sql server. (2016). Web: http://www.intuit.ru/studies/courses/5/5/lecture/138?page=2 (дата обращения 09.12.2018).
9. Worldskills Rassia.   
   Web: https://nationalteam.worldskills.ru/skills/proektirovanie-use-case-diagrammy-opredelenie-funktsionalnykh-vozmozhnostey-sistemy/

ПРИЛОЖЕНИЕ А

SQL скрипт на создание базы

USE [master]

GO

/\*\*\*\*\*\* Object: Database [ TA] Script Date: 31.01.2021 21:47:33 \*\*\*\*\*\*/

CREATE DATABASE [ TA]

CONTAINMENT = NONE

ON PRIMARY

( NAME = N' ta', FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA\ ta.mdf' , SIZE = 8192KB , MAXSIZE = UNLIMITED, FILEGROWTH = 65536KB )

LOG ON

( NAME = N' ta\_log', FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA\ ta\_log.ldf' , SIZE = 8192KB , MAXSIZE = 2048GB , FILEGROWTH = 65536KB )

GO

ALTER DATABASE [ TA] SET COMPATIBILITY\_LEVEL = 140

GO

IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))

begin

EXEC [ TA].[dbo].[sp\_fulltext\_database] @action = 'enable'

end

GO

ALTER DATABASE [ TA] SET ANSI\_NULL\_DEFAULT OFF

GO

ALTER DATABASE [ TA] SET ANSI\_NULLS OFF

GO

ALTER DATABASE [ TA] SET ANSI\_PADDING OFF

GO

ALTER DATABASE [ TA] SET ANSI\_WARNINGS OFF

GO

ALTER DATABASE [ TA] SET ARITHABORT OFF

GO

ALTER DATABASE [ TA] SET AUTO\_CLOSE OFF

GO

ALTER DATABASE [ TA] SET AUTO\_SHRINK OFF

GO

ALTER DATABASE [ TA] SET AUTO\_UPDATE\_STATISTICS ON

GO

ALTER DATABASE [ TA] SET CURSOR\_CLOSE\_ON\_COMMIT OFF

GO

ALTER DATABASE [ TA] SET CURSOR\_DEFAULT GLOBAL

GO

ALTER DATABASE [ TA] SET CONCAT\_NULL\_YIELDS\_NULL OFF

GO

ALTER DATABASE [ TA] SET NUMERIC\_ROUNDABORT OFF

GO

ALTER DATABASE [ TA] SET QUOTED\_IDENTIFIER OFF

GO

ALTER DATABASE [ TA] SET RECURSIVE\_TRIGGERS OFF

GO

ALTER DATABASE [ TA] SET DISABLE\_BROKER

GO

ALTER DATABASE [ TA] SET AUTO\_UPDATE\_STATISTICS\_ASYNC OFF

GO

ALTER DATABASE [ TA] SET DATE\_CORRELATION\_OPTIMIZATION OFF

GO

ALTER DATABASE [ TA] SET TRUSTWORTHY OFF

GO

ALTER DATABASE [ TA] SET ALLOW\_SNAPSHOT\_ISOLATION OFF

GO

ALTER DATABASE [ TA] SET PARAMETERIZATION SIMPLE

GO

ALTER DATABASE [ TA] SET READ\_COMMITTED\_SNAPSHOT OFF

GO

ALTER DATABASE [ TA] SET HONOR\_BROKER\_PRIORITY OFF

GO

ALTER DATABASE [ TA] SET RECOVERY FULL

GO

ALTER DATABASE [ TA] SET MULTI\_USER

GO

ALTER DATABASE [ TA] SET PAGE\_VERIFY CHECKSUM

GO

ALTER DATABASE [ TA] SET DB\_CHAINING OFF

GO

ALTER DATABASE [ TA] SET FILESTREAM( NON\_TRANSACTED\_ACCESS = OFF )

GO

ALTER DATABASE [ TA] SET TARGET\_RECOVERY\_TIME = 60 SECONDS

GO

ALTER DATABASE [ TA] SET DELAYED\_DURABILITY = DISABLED

GO

ALTER DATABASE [ TA] SET QUERY\_STORE = OFF

GO

USE [ TA]

GO

ALTER DATABASE SCOPED CONFIGURATION SET IDENTITY\_CACHE = ON;

GO

ALTER DATABASE SCOPED CONFIGURATION SET LEGACY\_CARDINALITY\_ESTIMATION = OFF;

GO

ALTER DATABASE SCOPED CONFIGURATION FOR SECONDARY SET LEGACY\_CARDINALITY\_ESTIMATION = PRIMARY;

GO

ALTER DATABASE SCOPED CONFIGURATION SET MAXDOP = 0;

GO

ALTER DATABASE SCOPED CONFIGURATION FOR SECONDARY SET MAXDOP = PRIMARY;

GO

ALTER DATABASE SCOPED CONFIGURATION SET PARAMETER\_SNIFFING = ON;

GO

ALTER DATABASE SCOPED CONFIGURATION FOR SECONDARY SET PARAMETER\_SNIFFING = PRIMARY;

GO

ALTER DATABASE SCOPED CONFIGURATION SET QUERY\_OPTIMIZER\_HOTFIXES = OFF;

GO

ALTER DATABASE SCOPED CONFIGURATION FOR SECONDARY SET QUERY\_OPTIMIZER\_HOTFIXES = PRIMARY;

GO

ALTER DATABASE [ TA] SET READ\_WRITE

GO

USE [ TA]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Counries] Script Date: 31.01.2021 21:52:24 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Counries](

[id\_countries] [int] NOT NULL,

[title] [nvarchar](50) NULL,

CONSTRAINT [PK\_Counries] PRIMARY KEY CLUSTERED

(

[id\_countries] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

USE [ TA]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Customers] Script Date: 31.01.2021 21:52:57 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Customers](

[id\_customers] [int] NOT NULL,

[series\_pass] [int] NULL,

[surename] [nvarchar](50) NULL,

[name] [nvarchar](50) NULL,

[middle\_name] [nvarchar](50) NULL,

[date\_of\_birth] [date] NULL,

[number\_pass] [int] NULL,

CONSTRAINT [PK\_Customers] PRIMARY KEY CLUSTERED

(

[id\_customers] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

USE [ TA]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Tours] Script Date: 31.01.2021 21:53:20 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Tours](

[id\_tours] [int] NOT NULL,

[id\_countres] [int] NULL,

[start\_tour] [date] NULL,

[end\_tour] [date] NULL,

[quantity] [int] NULL,

CONSTRAINT [PK\_Tours] PRIMARY KEY CLUSTERED

(

[id\_tours] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

USE [ TA]

GO

/\*\*\*\*\*\* Object: Table [dbo].[TravelPackeges] Script Date: 31.01.2021 21:53:34 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[TravelPackeges](

[id\_travel\_packeges] [int] NOT NULL,

[id\_customers] [int] NULL,

[id\_tours] [int] NULL,

CONSTRAINT [PK\_TravelPackeges] PRIMARY KEY CLUSTERED

(

[id\_travel\_packeges] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[TravelPackeges] WITH CHECK ADD CONSTRAINT [FK\_TravelPackeges\_Customers] FOREIGN KEY([id\_customers])

REFERENCES [dbo].[Customers] ([id\_customers])

GO

ALTER TABLE [dbo].[TravelPackeges] CHECK CONSTRAINT [FK\_TravelPackeges\_Customers]

GO

ALTER TABLE [dbo].[TravelPackeges] WITH CHECK ADD CONSTRAINT [FK\_TravelPackeges\_Tours] FOREIGN KEY([id\_customers])

REFERENCES [dbo].[Tours] ([id\_tours])

GO

ALTER TABLE [dbo].[TravelPackeges] CHECK CONSTRAINT [FK\_TravelPackeges\_Tours]

GO

ПРИЛОЖЕНИЕ Б

**Код программы**

MainWindow.xaml.cs

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace localMovieViewingHistory

{

public class Raiting

{

public int id { get; set; }

public string name { get; set; }

public string nameR { get; set; }

public double raiting { get; set; }

}

public class tours

{

public int id\_tours { get; set; }

public string title\_tours { get; set; }

public string start\_tours { get; set; }

public int duration { get; set; }

public string actors { get; set; }

public string dateStart { get; set; }

}

public class genre

{

public int id { get; set; }

public string name { get; set; }

}

public class view

{

public int id { get; set; }

public int idTours { get; set; }

public int stars { get; set; }

public string date { get; set; }

public string time { get; set; }

public string nameFilmOriginal { get; set; }

public string nameFilmRussian { get; set; }

public string description { get; set; }

}

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public bool checkQuery(string query)

{

if(query.Contains('`'))

{

return false;

}

return true;

}

SqlConnection connect = new SqlConnection("Data Source=localhost\\sqlexpress;Initial Catalog=TA;User ID=sa;Pwd=123;");

List<tours> tourss = new List<tours>();

List<genre> genres = new List<genre>();

List<view> views = new List<view>();

List<Raiting> raitings = new List<Raiting>();

public void getTours()

{

tourss = new List<tours>();

try

{

connect.Open();

SqlCommand getTours = connect.CreateCommand();

int minReturnDateActual = 0;

try

{

minReturnDateActual = Convert.ToInt32(TextBox2.Text);

}

catch

{

MessageBox.Show("Ошибка формата!!");

connect.Close();

TextBox2.Text = "";

return;

}

getTours.CommandText = string.Format("SELECT title\_tours,start\_tours, quantity FROM dbo.films WHERE(start\_tours LIKE '{0}%') AND(title\_tours LIKE '{1}%')", TextBox1.Text, Date1.SelectedDate, TextBox2.Text);

SqlDataReader toursReader = getTours.ExecuteReader();

while (toursReader.Read())

{

tours.Add(new tourss

{

start\_tours = (string)toursReader[1],

title\_tours = (string)toursReader[2],

actors = (string)toursReader[4],

dateStart = (string)toursReader[5]

});

}

tourTour.Items.Clear();

foreach (tours film in tourss)

{

tourTour.Items.Add(film);

}

}

finally

{

connect.Close();

}

}

public void getGenres()

{

genres = new List<genre>();

try

{

connect.Open();

SqlCommand getGenres = connect.CreateCommand();

getGenres.CommandText = string.Format("SELECT id, genre\_name FROM genres");

SqlDataReader genresReader = getGenres.ExecuteReader();

while (genresReader.Read())

{

genres.Add(new genre

{

id = (int)genresReader[0],

name = (string)genresReader[1]

});

}

} finally

{

connect.Close();

}

}

public MainWindow()

{

InitializeComponent();

filmsPanel.Visibility = Visibility.Hidden;

}

private void filmsGetterEvent(object sender, RoutedEventArgs e)

{

getTours();

}

public int getIdGenreFromName(string name)

{

foreach (genre genre in genres)

{

if (genre.name == name)

{

return genre.id;

}

}

return -1;

}

private void Button\_Click\_3(object sender, RoutedEventArgs e)

{

if (TextBox1.Text != "")

{

if (TextBox2.Text != "")

{

if (Date1.SelectedDate != null)

{

{

int idTours = 0;

try

{

connect.Open();

SqlCommand setFilm = connect.CreateCommand();

setFilm.CommandText = string.Format("INSERT INTO Tours(title\_tours, quantity, start\_tours) VALUES (N'{0}', N'{1}', {2}, N'{3}')", TextBox1.Text, TextBox2.Text, Date1.SelectedDate);

if (checkQuery(setFilm.CommandText))

{

setFilm.ExecuteNonQuery();

}

else

{

MessageBox.Show("Обноружена отака!");

}

}

catch

{

MessageBox.Show("Запись должна быть уникальна!");

}

finally

{

connect.Close();

}

try

{

connect.Open();

SqlCommand getNewTours = connect.CreateCommand();

getNewTours.CommandText = string.Format("SELECT id FROM Tours WHERE (title\_tours = N'{0}')", TextBox1.Text);

SqlDataReader readerNewTours = getNewTours.ExecuteReader();

readerNewTours.Read();

idTours = (int)readerNewTours[0];

}

catch

{

MessageBox.Show("");

}

finally

{

connect.Close();

TextBox1.Text = "";

TextBox2.Text = "";

Date1.Text = "";

}

}

}

else

{

int idTours = 0;

try

{

connect.Open();

SqlCommand setFilm = connect.CreateCommand();

setFilm.CommandText = string.Format("INSERT INTO films(name\_film\_original, name\_film\_russian, duration, actors, premiere\_date) VALUES (N'{0}', N'{1}', {2}, N'{3}', N'{4}')", TextBox1.Text, TextBox2.Text, Date1.SelectedDate);

setFilm.ExecuteNonQuery();

}

catch

{

MessageBox.Show("Запись должна быть уникальна!");

}

finally

{

connect.Close();

}

try

{

connect.Open();

SqlCommand getNewTours = connect.CreateCommand();

getNewTours.CommandText = string.Format("SELECT id FROM Tours WHERE (title\_tours = N'{0}')", TextBox1.Text);

SqlDataReader readerNewTours = getNewTours.ExecuteReader();

readerNewTours.Read();

idTours = (int)readerNewTours[0];

}

catch

{

MessageBox.Show("");

}

finally

{

connect.Close();

TextBox1.Text = "";

TextBox2.Text = "";

Date1.Text = "";

}

}

}

else

{

MessageBox.Show("Введите количество");

}

}

else

{

MessageBox.Show("Введите оригинальное название тура");

}

}

private void TextBox1\_TextChanged(object sender, TextChangedEventArgs e)

{

}

int selectedMenu;

private void TabControl\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if(selectedMenu!=menu.SelectedIndex)

{

selectedMenu = menu.SelectedIndex;

getTours();

getGenres();

}

}

}

}

MainWindow.xaml

<Window x:Class="localMovieViewingHistory.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:localMovieViewingHistory"

mc:Ignorable="d"

Title="Турестическое агенство" Height="480" Width="640" MinHeight="480" MinWidth="640">

<Grid>

<Grid.RowDefinitions>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

</Grid.ColumnDefinitions>

<Label x:Name="CurrentLogin" Content="" Grid.Row="0" Grid.Column="6" Grid.ColumnSpan="3"/>

<TabControl x:Name="menu" Grid.Row="0" Grid.Column="0" Grid.RowSpan="10" Grid.ColumnSpan="12" SelectionChanged="TabControl\_SelectionChanged" >

<TabItem x:Name="filmsPanel" Header="Туры" Height="60">

<Grid Background="AliceBlue">

<Grid.ColumnDefinitions>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

<RowDefinition></RowDefinition>

</Grid.RowDefinitions>

<Label VerticalAlignment="Bottom" Content="Туры" Grid.Row="0" Grid.Column="0" Grid.ColumnSpan="2"/>

<Label VerticalAlignment="Center" Content="Название" Grid.Row="1" Grid.Column="7" Grid.ColumnSpan="1" HorizontalAlignment="Right" />

<Label VerticalAlignment="Center" Content="Начало тура" Grid.Row="3" Grid.Column="7" Grid.ColumnSpan="2" HorizontalAlignment="Right" Margin="0,10,48,28" Grid.RowSpan="2"/>

<Label VerticalAlignment="Center" Content="Количество" Grid.Row="2" Grid.Column="7" Grid.ColumnSpan="2" HorizontalAlignment="Right" Margin="0,5,48,1"/>

<ListView SelectionChanged="getInfoOutTours" x:Name="toursTours" Grid.Row="1" Grid.Column="0" Grid.ColumnSpan="7" Grid.RowSpan="8" Margin="4 4 4 4">

<ListView.View>

<GridView>

<GridViewColumn DisplayMemberBinding="{Binding Path=origName}" >Название</GridViewColumn >

<GridViewColumn DisplayMemberBinding="{Binding Path=rusName}" >Начало тура</GridViewColumn>

<GridViewColumn DisplayMemberBinding="{Binding Path=actors}">Количество</GridViewColumn>

<GridViewColumn DisplayMemberBinding="{Binding Path=dateStart}" >Клиент</GridViewColumn>

</GridView>

</ListView.View>

</ListView>

<TextBox TextChanged="filmsGetterEvent" x:Name="TextBox1" Height="22" Grid.Row="1" Grid.Column="8" Grid.ColumnSpan="2" Grid.RowSpan="1" Margin="4 4 4 4"></TextBox>

<TextBox TextChanged="filmsGetterEvent" x:Name="TextBox2" Height="22" Grid.Row="2" Grid.Column="8" Grid.ColumnSpan="2" Grid.RowSpan="1" Margin="4 4 4 4"></TextBox>

<DatePicker x:Name="Date1" Height="22" Grid.Row="3" Grid.Column="8" Grid.ColumnSpan="2" Grid.RowSpan="1" Margin="4,10,4,0"></DatePicker>

<Button Content="Добавить" Grid.Row="5" Grid.Column="9" Grid.RowSpan="1" Grid.ColumnSpan="3" Height="30" Margin="10 0 10 0" MaxWidth="100" Click="Button\_Click\_3"/>

</Grid>

</TabItem>

</TabControl>

</Grid>

</Window>