**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«УНИВЕРСИТЕТ УПРАВЛЕНИЯ «ТИСБИ»**

***Факультет информационных технологий***

***Кафедра информационных технологий***

Допущен к защите

Зав. кафедрой Информационных технологий

*Доцент, к.п.н.*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.В.Федорова

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2018 г.

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

на тему: «Проектирование программных средств базы данных библиотеки»

Работу выполнил

Студент гр.ИВТ-411 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ю.М. Пивоваров

*подпись*

Научный руководитель

Доцент, к.т.н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.Н.Козин

*подпись*

Казань, 2018 г.

СОДЕРЖАНИЕ

[1. ПОСТАНОВКА ЗАДАЧИ 3](#_Toc515195458)

[1.1. Цель создания информационной системы «Библиотека» 3](#_Toc515195459)

[1.1.1. Назначение разработки 3](#_Toc515195460)

[1.1.2. Требования к надежности 4](#_Toc515195461)

[1.1.3. Специальные требования 4](#_Toc515195462)

[1.1.4. Инструменты разработки 4](#_Toc515195463)

[1.2. Анализ предметной области 4](#_Toc515195464)

[1.2.1. Анализ деятельности организации 4](#_Toc515195465)

[1.2.2. Анализ существующих систем 5](#_Toc515195466)

[2. ПРОЕКТИРОВАНИЕ СТРУКТУРЫ СИСТЕМЫ 15](#_Toc515195467)

[2.1. Проектирование структуры системы 15](#_Toc515195468)

[2.2. Определение состава базы данных 19](#_Toc515195469)

[2.3. Физическое проектирование базы данных 22](#_Toc515195470)

[3. РАЗРАБОТКА СТРУКТУРЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ БИБЛИОТЕКА 23](#_Toc515195471)

[3.1. Средства реализации 23](#_Toc515195472)

[3.2. Общая структура системы 24](#_Toc515195473)

[3.3. Руководство пользователя 37](#_Toc515195474)

[3.3.1. Системные требования программных и аппаратных средств 37](#_Toc515195475)

[3.3.2. Инсталляция системы 38](#_Toc515195476)

[3.3.3. Результаты тестирования с примерами экранных форм 39](#_Toc515195477)

[ЗАКЛЮЧЕНИЕ 49](#_Toc515195478)

[СПИСОК ЛИТЕРАТУРЫ 50](#_Toc515195479)

[ПРИЛОЖЕНИЕ 1 52](#_Toc515195480)

[ПРИЛОЖЕНИЕ 2 64](#_Toc515195481)

[ПРИЛОЖЕНИЕ 3 79](#_Toc515195482)

# ПОСТАНОВКА ЗАДАЧИ

## Цель создания информационной системы «Библиотека»

Целью информационной системы «Библиотека» является повышение удобства пользования приложением путем сокращения непроизводительных затрат и повторяющихся операций. Улучшение показателей по качеству услуг, предоставляемых клиенту-читателю.

### Назначение разработки

Информационная система «Библиотека» предназначена для информатизации процессов деятельности организации Библиотека. С реализацией аналитических процессов.

Система «Библиотека» предназначена для автоматизации следующих процессов:

1. Учет книг и их экземпляров;
2. Оформление заказов книг;
3. Контроль выдачи и возврата книг;
4. Ведение учетных записей сотрудников;
5. Ведение читательских билетов.

Приложение должно выполнять задачу по информационному обеспечению библиотеки и предоставлять возможность пользователям выполнять свои рабочие функции. Таким образом система должна осуществлять:

* возможность добавления и редактирования книг/экземпляров, их хранение и отображение;
* ведение базы данных сотрудников и читателей;
* бронирование печатных изданий;
* возможность выдачи и возврата книг;
* разделение рабочей зоны сотрудников организации в зависимости от их рабочих обязанностей.
* разграничения прав доступа с помощью разных режимов работы.

### Требования к надежности

программное обеспечение иметь:

* модуль для доступа системе;
* ограничение к информации данных;
* контроль информации и этой информации некорректных действиях ;
* разграничение пользовательских .

### Специальные требования

Интерфейс то информационной системы на обеспечивать интуитивно то и наглядное но информационной структуры, год на нем, разделы должны быть представлены в соответствии с логичным и быстрым переходами между ними. до Система «Библиотека» на отображать всю на информацию.

### Инструменты разработки

Проектирование системы «Библиотека» является результатом работы в CASE-средствах Rational Rose и ARIS. Для создания клиентского приложения в качестве среды разработки была выбрана RAD Studio 10.2 и выбрана СУБД MSSQL.

## Анализ предметной области

### Анализ деятельности организации

Анализ предметной области проводится для выявления основных задач, которые требуется реализовать в приложении, это достигается путем изучения деятельных процессов организации, и выявить основных пользователей разрабатываемого приложения, которыми являются рядовые сотрудники.

Библиотека – это учреждение культурного характера, осуществляющее сбор, хранение и выдачу с обязанностью возврата печатных изданий художественной, научной, исторической литературы и так далее.

Основными направлениями деятельности библиотеки являются: ведение печатного фонда и его предоставление читателем.

Рабочие процессы организации осуществляется несколькими категориями сотрудников: библиотекарь, выполняет свою основную задачу выдачи печатных изданий и оформление возврата печатных изданий; управляющий библиотекой, занимающийся формированием фонда печатных изданий, а также ведением личных дел сотрудников и читателей.

Организация литературного фонда включает вопросы учета, хранения, расстановки и доставки её читателю. Правильная организация фонда ускоряет работу всей структуры и облегчает библиотекарю выполнение читательских требований.

### Анализ существующих систем

На данный момент существует достаточно информационных систем для обеспечения программного обеспечения библиотек, основной целью которых автоматизация все возможных бизнес процессов библиотеки. Все данные системы обладают разными качественными и количественными характеристиками. Однако, как правило, часть систем имеют специализацию по конкретный тип библиотек. Было проанализировано множество информационных систем, среди которых подробнее будут рассмотрены ИС Нерюнгинской городской библиотеки, ИС для библиотеки имени Маяковского и ИС «1С: Библиотека».

**ИС для библиотеки имени Маяковского.**

Приведем описание структуры её деятельности и организации процессов:

В организацию поступает довольно большое количество различных печатных изданий из разных издательств. У каждого издания, которое состоит в печатном фонде библиотеки, есть специальный индивидуальный номер, который дают при поступлении этого издания в библиотеку.

При поступлении книги, учитывают такие данные как:

* Индивидуальный номер книги;
* Наименование книги;
* Наименование печатного издательства, которое прислало данную книгу;
* Отдел библиотеки, к которому относится книжное издание;

В системе реализованы выдача книг читателю и их сдача, добавление в базу новых читателей и новых книжных поступлений, отслеживание информации о книгах, которые имеются в наличии, а также о сотрудниках организации.

Книга, которая состоит в печатном фонде библиотеки, имеет следующие необходимые данные:

* ФИО автора или авторов,
* Наименование издания,
* Год печати,
* Тип жанра,
* Ключевые слова,
* Суммарное количество страниц.

Книга поступает в библиотеку в некотором количестве, т.е имеет несколько экземпляров.

Для учета читателей в библиотеке введена система карточного учета, которая хранит следующие данные читателя:

* ФИО читателя;
* Данные паспорта;
* Адрес проживания;
* Мобильный телефон.

Реальный читатель в этой карточной системе сопоставим с номером читательского билета.

В читательский билет заносится вкладыш книги, которую читатель взял на определенный срок, этот срок записывается во вкладыш вместе с номером читательского билета и датами выдачи и предполагаемого возврата экземпляра.

Если читатель не укладывается в указанный ему срок, то он может сообщить в библиотеку с помощью телефонного звонка. Библиотекарь продлит срок возврата книги.

При просроченных нескольких сроках возврата книги, читатель получает штрафные санкции от организации в виде отлучения от печатного фонда на определенный срок. В случае потери книги библиотека может оштрафовать читателя.

Также в организации есть некоторые внутренние правила, отражающие процесс выдачи книги. В них говорится о максимальном сроке выдачи книг и количестве книг в «одни руки».

Выдача книги должна фиксироваться в базе имея следующие данные:

* Наименование книги;
* Уникальный код экземпляра;
* Дата выдачи и возврата книги;
* Номер читательского билета;
* ФИО сотрудника, который произвел процесс выдачи книжного издания;
* ФИО сотрудника хранилища, который передал книгу сотруднику, описанного выше, чтобы тот в свою очередь передал ее читателю.

Сдача книги осуществляется путем фиксирования в системе даты возврата, сдаваемой читателем книги путем ввода ее названия в БД.

На каждого читателя заносятся следующие сведения:

* Номер читательского билета;
* ФИО читателя;
* Личный телефон;
* Адрес проживания.

Для каждой книги выделены следующие характеристики, которые необходимо фиксировать в базе:

* Наименование книги;
* Уникальный код - ISBN;
* Библиотечно-библиографическая классификация (ББК);
* Раздел библиотечного помещения, в котором храниться книга;
* ФИО автора или авторов;
* Наименование издательства;
* Город издания;
* Год издания.

Также в ИС организован учет посещаемости библиотеки следующим образом:

Учет посещений и обращений.

Если одно посещение зафиксировано в базе или в документации организации, то оно считается одной единицей посещения.

Если физическое лицо, присутствующее на мероприятии, зафиксировано в базе или в документации организации, сопутствующие этому мероприятию, то оно считается одной единицей учета посетителя.

Если было обращение к серверам библиотеки, то это обращение считается одной единицей учета обращений к электронной библиотечно-информационной сети.

Общее число посещений выясняется путем суммирования всех единичных посещений или обращений, которые фиксировались в базе системы.

Формирование.

Для формирования библиотекарю необходимо обратиться к системе с запросом данных, которые отражают посещения читателей по дням за запрашиваемый период. Система должна вывести количество посещений за данный период и статистику посещений читателей библиотеки в виде наглядных диаграмм или графиков. Из графика можно сделать выводы, к примеру, в какой день читатели были самыми активными или же совсем не активными, т.е в какой день было большое количество посещений или маленькое количество.

Выводимая статистика помогает сотрудникам составить отчеты, которые в свою очередь помогут проанализировать, к примеру посещаемость, и привести руководству к выводам по привлечению новых читателей или же улучшения условий для старых читателей.

**ИС Нерюнгинской городской библиотеки**

Для контроля книг, которые читатели берут, библиотека забирает себе имеющийся у книги вкладыш. Срок записывается во вкладыш вместе с номером читательского билета и датами выдачи и предполагаемого возврата экземпляра.

Если читатель не укладывается в указанный ему срок, то он может сообщить в библиотеку с помощью телефонного звонка. Библиотекарь продлит срок возврата книги, если не противоречит внутренним правилам организации.

При просроченных нескольких сроках возврата книги, читатель получает штрафные санкции от организации в виде отлучения от печатного фонда на определенный срок. В случае потери книги библиотека может оштрафовать читателя.

Также, как было упомянуто выше, в организации есть некоторые внутренние правила, отражающие процесс выдачи книги. В них говорится о максимальном сроке выдачи книг и количестве книг в «одни руки».

Для анализа будет рассматривать следующие процессы:

* Выдача экземпляра читателю;
* Сдача и принятие книги;
* Поступление новых экземпляров;
* Добавление новых читателей в базу.

Субъекты – клиент, он же читатель, сотрудники библиотеки – библиотекарь и администратор. Все они оперируют данными, которые хранится в базе данных.

Данные, которые идут на вход:

* Поступившие печатные издания;
* Новые читатели;
* Процесс выдачи: ФИО сотрудника хранилища, ФИО сотрудника библиотекаря, ФИО читателя, наименование книги, даты выдачи и сдачи.

Данные, которые идут на выход:

* Составляемых отчетов, которые сопутствуют появления статистики, к примеру посещения;
* Рейтинг востребованных книг, что помогает администрации сделать выводы о изнашиваемости экземпляров книг.

Статистические отчеты:

Учет читателей, показывает на сколько востребованы документы библиотечной организации. Составляется на основе количества читателей. Единицей служит формуляр читателя.

Учет выдачи книг, показывает востребованность тех или иных печатных изданий читателем, производится учет путем подсчета количества используемым документов. Статистика составляется исходя из количества выданных экземпляров, которые зарегистрированы в читательском формуляре, или любой другой официальной документации библиотеки, а также в электронной базе данных (БД).

**ИС «1С: Библиотека»**

Система предоставляет возможность пользоваться автоматизированными процессами библиотечной организации, при этом ИС «1С: Библиотека» учитывает тип, назначения и индивидуальные особенности предприятия. Также предоставляет особенности интеграции с другими продуктами «1С: Предприятия».

«1С: Библиотека» может интегрироваться для библиотек многих типов назначений:

* городские, муниципальные, областные;
* библиотеки детских садов, средних общеобразовательных школ, колледжей, ВУЗов;
* библиотеки отраслевых организаций: военных частей, медицинских учреждений, научных институтов и т.д;
* специализированные библиотеки: музеев, музыкальный и театральных академий;
* частные библиотеки, в том числе домашние.

Комплектование:

Оформление заказов, в том числе принятие заказа из печатных издательств.

Проверка на отсутствие дублированных данных.

Присвоение каждому новому документу уникального штрих-кода, а также регистрация кода в системе как подтверждения занесения в базу новых поступивших документов.

Каталогизация:

Предоставление типизации поступающих в организацию документов: электронный формат, аудиоиздания, видеоиздания, нотные издания, картографические издания, изоиздания и т.п.

Поддержка национального коммуникативного формата RUSMARC.

Быстрый поиск документов по описаниям и их сочетаниям, основанный на автоматической индексации.

Использование графических материалов любых типов, к примеру, титульные листы, артефакты, страницы текста.

Экспортирование и импортирование данных в MARC-форматах.

Неограниченное количество возможных записей.

Учет, актуализация и хранение фонда:

Хранение документов в соответствии их типизации, т.е осуществление упорядоченного учета всех данных документов учитывая их тип и назначение.

Обслуживание читателей:

Предоставление различным группам читателей доступ к содержанию печатного фонда библиотечной организации.

Присвоение каждому новому читателю читательского билета с уникальным штрих-кодом, как подтверждения занесения в базу данных читателя.

Виртуальный кабинет читателя:

Предоставление зарегистрированному читателю каталога с электронными или оцифрованными изданиями.

Возможность поиска в электронном каталоге, по ключевым словам, и описанию документов.

Отправление заявки по найденному материалу.

Возможность просмотра наличия того или иного документа в физическом виде в печатном фонде библиотеки.

Доступ к личному читательскому билету.

Отслеживания читателем состояния заказа на определенный документ.

Оповещение читателя о результате заказа и предупредительных сообщений о задолжностях читателя перед организацией.

Возможность прикрепление к документам электронного каталога любого количества электронных материалов-изданий.

Стандартный поиск фрагментов по электронному документу-изданию.

Возможность прикрепления дополнительных материалов к том числе титульных листов или обложек изданий.

Возможность пополнения электронного каталога и предоставления читателю доступ к нему.

Подсчет статистических данных и формирование на их основе отчетов.

Предоставление формирования инвентарной книги.

Отчет по проверки печатного и электронного фонда библиотеки.

Осуществление выдачи физических книг.

Обслуживание печатных изданий:

Создание и хранение сопутствующих документов для поддержки печатных изданий карточного каталога библиотеки.

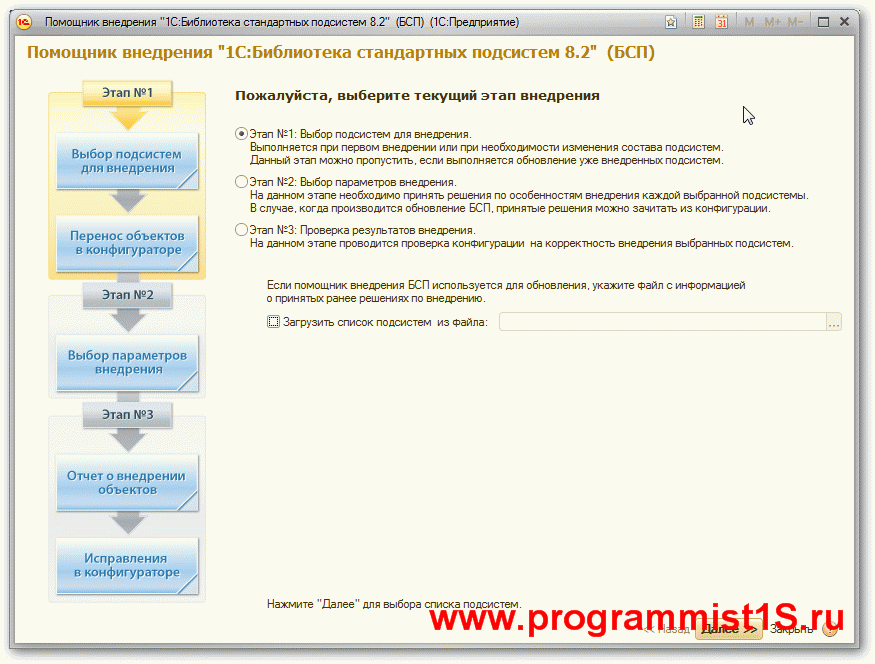


Рис.1.2.2.1. Внедрение ИС «1С: Библиотека»

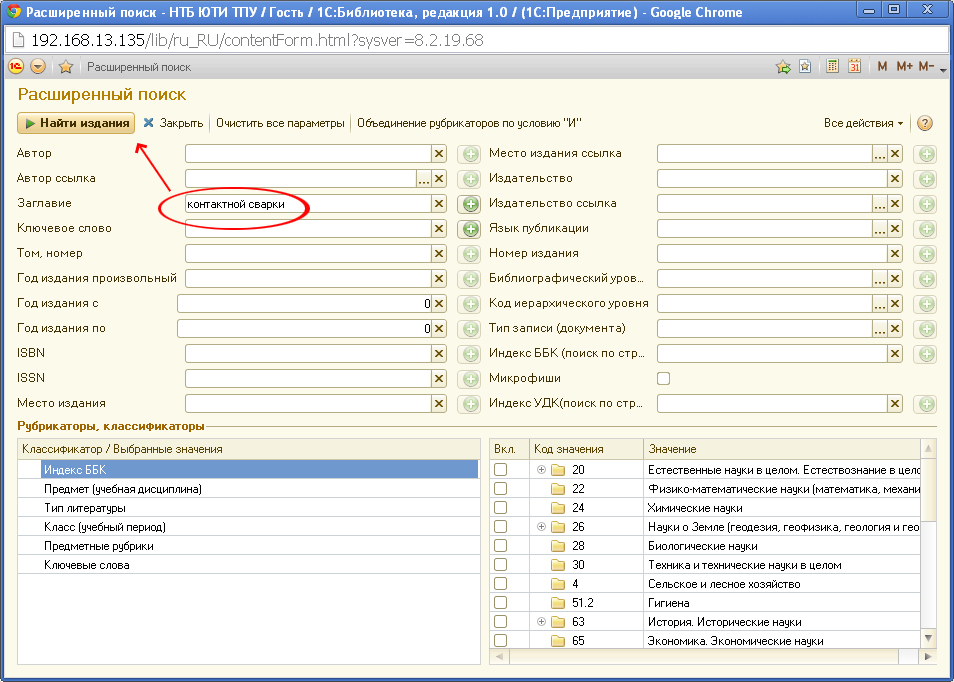


Рис.1.2.2.2. Форма поиска ИС «1С: Библиотека»

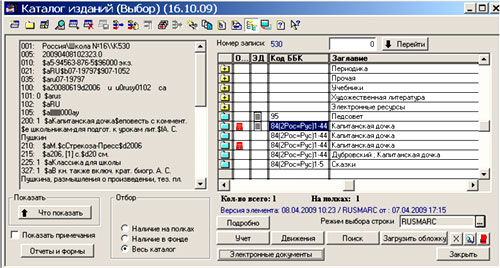


Рис.1.2.2.3. Каталог ИС «1С: Библиотека»

# ПРОЕКТИРОВАНИЕ СТРУКТУРЫ СИСТЕМЫ

## Проектирование структуры системы

Для проектирования структуры системы необходимо использовать данные, полученные в результате проведенного анализа предметной области в том числе положительные моменты, которые были выведены во время анализа существующих систем.

Полученный результат можно интерпретировать в структуру системы, основных пользователей – ролей и функции, при помощи которых будут выполняться поставленные задачи.

Моделирование системы будет осуществляться при помощи специального CASE-средства такого как Rational Rose Enterprise.

Rational Rose Enterprise использует объектно-ориентированный подход, который использует язык UML - унифицированный язык моделирования. Специальный язык, который предоставляет возможность описания, документирования технических систем. В среде Rational Rose Enterprise есть возможность генерации кода на основе проектированных моделей и связей.

Для начала постоим диаграмму вариантов использования. Она отражает основные задачи, которые должна выполнят проектируемая система, и основных лиц, взаимодействующих с системой.

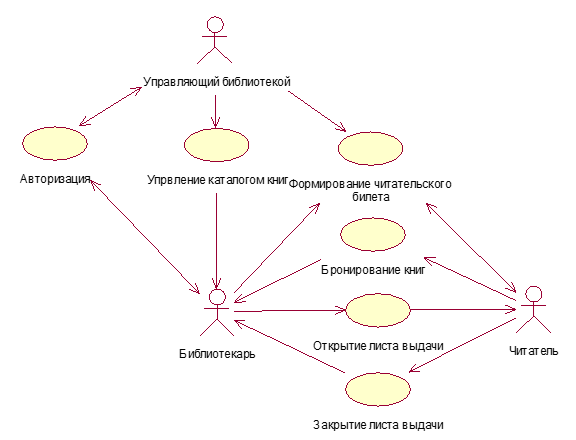


Рис. 2.1.1. Диаграмма вариантов использования

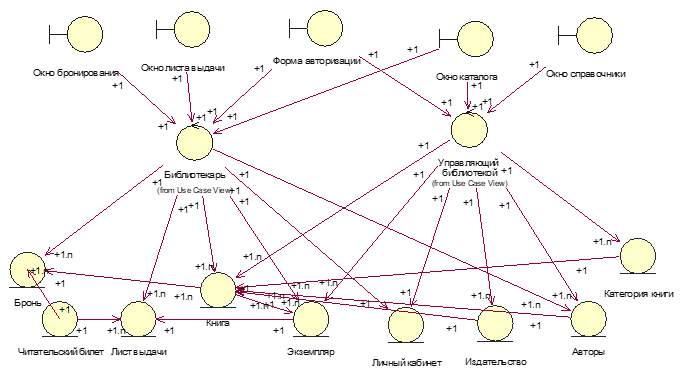


Рис. 2.2. Диаграмма классов

Диаграмма классов описывает возможные классы системы: boundary, control, entity. И взаимосвязь между ними.

На диаграмме классов для данной системы (Рис. 2.1.2) отражены действующие лица – класс control – библиотекарь и управляющий библиотекой. Классу control доступны формы для работы с системой, который относится к классу boundary.



Рис. 2.1.3. Диаграмма последовательности (Формирование читательского билета)

Диаграмма последовательности, представленная на Рис.2.1.3 отражает последовательное взаимодействие объектов с системой в промежутках времени.

Диаграмма состояний системы показывает фрагмент жизненного цикла системы и отражает поведение системы при тех или иных ситуациях.



Рис. 2.1.4. Диаграмма состояния системы

## Определение состава базы данных

Результаты анализа системы и ее моделирования нужно применить в определении состава базы данных. Начнем с выявления сущностей базы.

Сущность «Книга».

Эта сущность необходима, потому, что это главный продукт, который предоставляет читателю библиотека. Из книг состоит печатный фонд организации, который она хранит и производит всяческие операции с ним.

У сущности «Книга» есть атрибуты: название, авторы, жанр.

Сущность «Экземпляр».

В библиотеке не может храниться только одна книга этого произведения, как правило их несколько штук, за исключением очень редких книг. Такие копии книг называются экземплярами, которые имеют следующие атрибуты: наименование книги, год издания, издательство, место в зале, секция, а также уникальный код.

У сущности «Экземпляр» есть вспомогательная сущность - «Издательство». Она отражает наименование издательского дома, который произвел печать данной книги, к которой относится этот экземпляр. Атрибут – наименование.

Также к сущности «Книга» можно соотнести другие вспомогательные сущности. Далее рассмотрим их.

Сущность «Автор».

Сущность «Автор» отражает ФИО автора произведения или множество авторов. Основным атрибутом этой сущности является – ФИО.

Сущность «Жанр».

Хранит информацию о жанре, к которому принадлежит данное произведение. с помощью этой сущности легче читателю ориентироваться в печатном фонде библиотеки. Атрибут – Наименование.

Эти сущности, рассмотренные выше, помогают рассмотреть книгу более подробно, дополняя представление о ней.

Далее рассмотрим сущности, которые используются для взаимодействия читателя и библиотекаря.

Сущность «Читательский билет».

Сущность является электронным документом, удостоверяющий принадлежность читателя к этой библиотеки. Без оформления читательского билета, т.е без данных читателя, библиотекарь не может выдать экземпляр читателю для использования. Атрибуты сущности: ФИО читателя, серия и номер паспорта, адрес проживания, номер телефона.

Сущность «Лист выдачи».

Эта сущность отражает формуляр экземпляра книги, который остается в печатном фонде библиотеки вместо физической книги. Он содержит всю информацию, о читателе, который взял книгу, и о самом экземпляре. Атрибуты сущности: номер читательского билета, код экземпляра, дата, дата закрытия листа, статус.

Сущность «Библиотекарь».

Данная сущность отражает сотрудника библиотеки. Хранит в себе данные сотрудника и логин, пароль для авторизации в системе. Атрибуты сущности: ФИО, логин, пароль.

Сущность «Бронь».

Необходимость бронирования книги появляется, когда в печатном фонде данная книга есть, но в данный момент все ее экземпляры находятся у других читателей библиотеки. В такой ситуации библиотекарь оформляет бронь на читателя, который хотел бы эту книгу взять. Как только экземпляр книги будет возвращен другим читателем, то библиотекарь сообщит, который бронировал эту книгу. Атрибуты сущности: номер читательского билета, наименование книги.

Сущность «Заказ».

Формировать заказ конкретному издательскому дому для библиотеки необходимо, потому что экземпляры книг быстро изнашиваются и к тому же нужно пополнять печатный фонд новыми произведениями. Атрибуты сущности: Наименование произведения, издательство, количество, статус.

После вывода всех основных сущностей и их атрибуты можно составить ER-диаграмму(Рис.2.2.1).

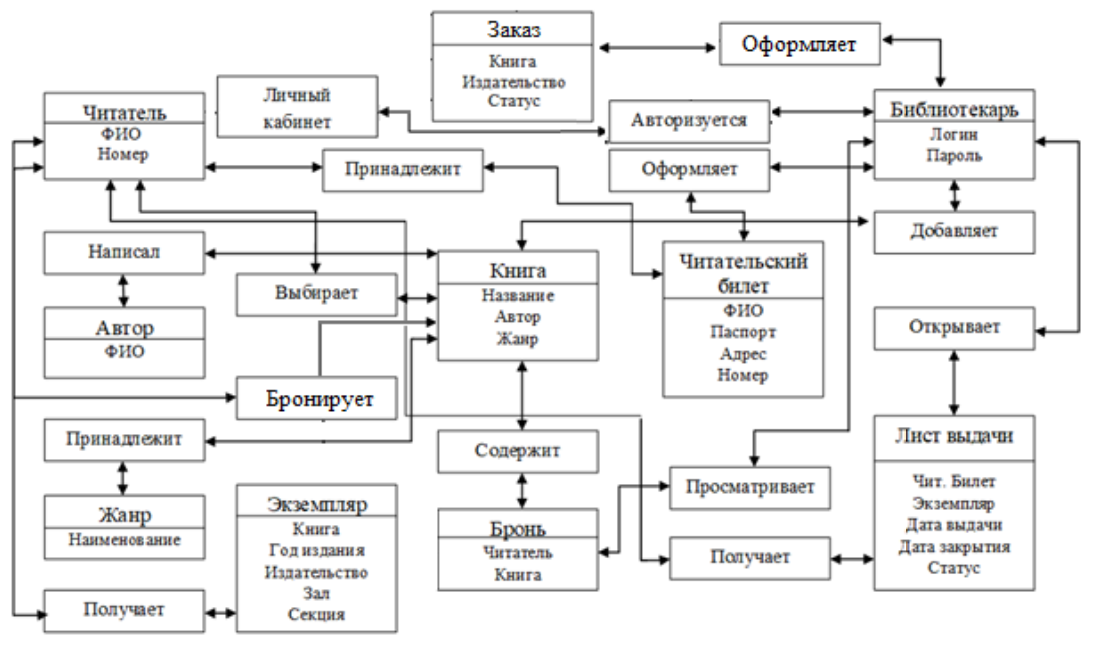


Рис.2.2.1. ER-диаграмма

## Физическое проектирование базы данных

Схема реляционной базы данных показывает таблицы и их взаимодействия в базе данных. Пример схемы реляционной базы данных "Библиотека" отображен на рис. 2.3.1.

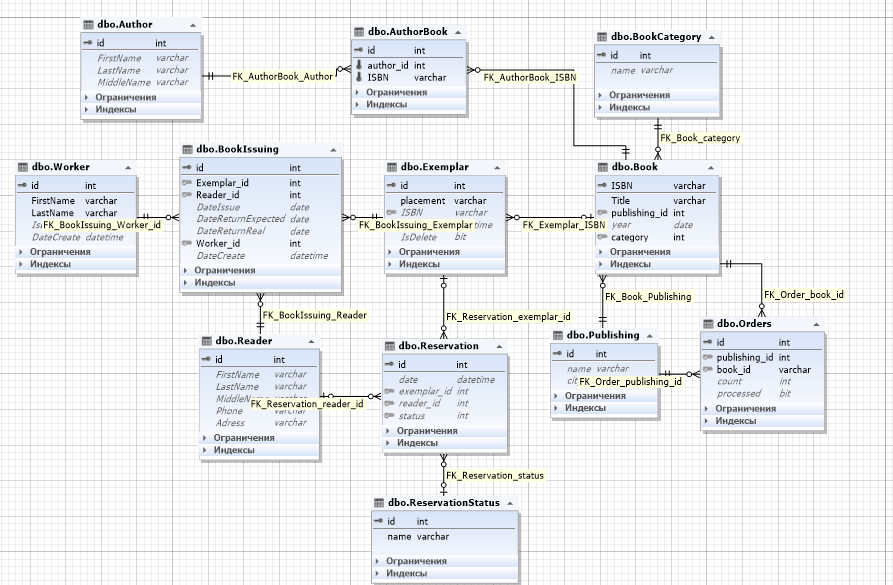


Рис. 2.2.1. Физическая схема базы данных

# РАЗРАБОТКА СТРУКТУРЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ БИБЛИОТЕКА

## Средства реализации

В качестве среды разработки была выбрана RAD Studio 10.2 разработанная компаний Embarcadero, а для хранения информации была выбрана система управления базами данных Microsoft SQL Server 2008.

RAD Studio — это новейшая среда разработки мульти-платформенных приложений. Главным достоинством данной среды является применение объектной технологии при визуальном проектировании приложении. Такой подход позволяет за короткий период получить качественный продукт. Главным залогом успеха проектов при разработке приложений с применением среды разработки RAD Studio является следование стандартам индустрии и возможности применения любого готового решения.

Для разработки системы была использована версия продукта — Starter Edition. Данная версия поставляется бесплатно с официального сайта компании Embarcadero.

Microsoft SQL Server это система управления базами данных предназначенная для работы под операционной системой Windows. На сегодняшний день системные требования необходимые для поддержания работоспособности СУБД достаточно невелики и определяются степенью нагрузки базы данных. Язык используемый для создания заапросов к базе данных — Transact-SQL. К основным преимуществам MS SQL Server можно перечислить: простоту визуального проектирования баз данных, легкость управления данными, а также наглядность и что не мало важно высокий уровень безопасности.

Microsoft SQL Server Management Studio — это инструмент, поставляемый вместе в MS SQL Server 2008 основным предназначением, которого является доступ к СУБД для работы с ней и ее администрирования. В данный продукт включены 2 редактора: скриптовый и графический.

## Общая структура системы

Структура проекта:

Файлы с разрешением \*.dpr – это текстовые файлы, которые используется для хранения информации о формах и модулях. В них содержатся операторы инициализации и запуска программ на выполнение.

Файлы с разрешением \*.dfm –двоичные или текстовые файлы, которые создается для хранения информации о формах. Каждому файлу формы соответствует файл модуля (.pas)

Каждой создаваемой форме соответствует текстовый файл модуля с разрешением \*.pas, используемый для хранения кода. Многие из функций и процедур Delphi хранятся в модулях.

Подробнее рассмотрим структуру модулей.

UnitAuth.pas – файл, который cодержит в себе обработчики формы FormAuth.dfm . Отвечает за логику авторизации пользователей в системе.

UnitSuperUser.pas – файл, который содержит в себе обработчики формы FormSuperUser.dfm и предназначен для работы пользователя с ролью заведующий библиотекой.

UnitLibrarianAccount.pas – файл, который содержит в себе обработчик формы FormLibrary.dfm и предназначен для пользователя библиотекарь.

UnitReaderAccount.pas – файл который содержит в себе обработчик формы «FormReaderAccount.pas» для пользователя читатель.

UnitExemplar.pas – файл который содержит обработчик формы «FormExemplar.dfm» и предназначен для работы с экземплярами книг.

UnitBook.pas – Файл, содержащий обработчики формы FormAddBook.dbf и предназначен для добавления книг.

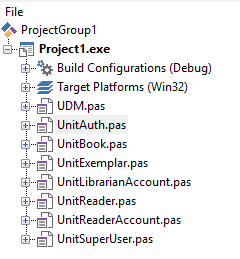


Рисунок. 3.2.1. Структура модулей

Рисунок. 3.2.2. Структура взаимосвязей модулей

Единственной не визуальной формой является DataModule. Она представляет собой форму-компонент для размещения на нем компонентов, предназначенных для соединения с базой данных.

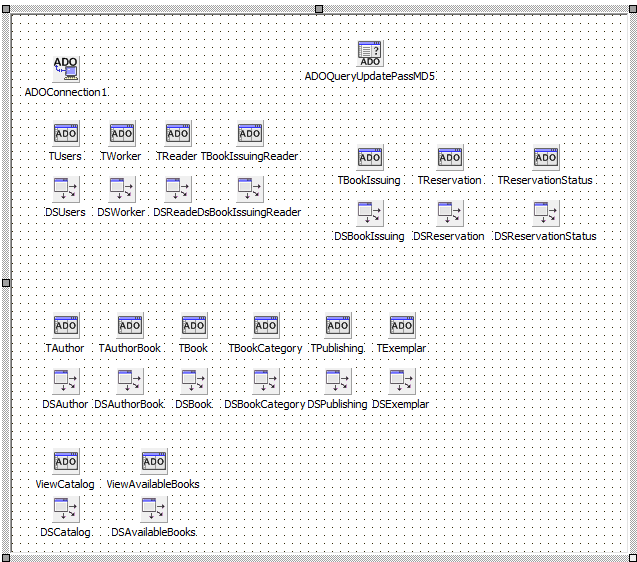


Рисунок. 3.2.2. Data Module

Схема соединения приложения с базой данных изображена на рисунке 3.2.3.

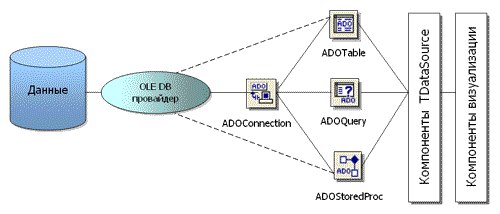


Рисунок. 3.2.4. Схема соединения

ADOConnection является компонентом, предназначенным для соединения с базой данных. Соединение с БД настраивается следующим образом. Указывается строка соединения с базой данных. Поставщиком данных выступает «Microsoft OLEDB Providerfor SQL Server». В качестве названия сервера устанавливается экземпляр, созданный при установке MS SQL. Помимо этого, экземпляра подключиться можно также и при указании локального адреса компьютера. Для входа на сервер базы данных используется средство встроенной безопасности Window NT. Имя базы данных «LibraryDB».

По окончании настройки соединения с базой данных настраивается следующим этапом является конфигурирование компонентов для отображения информации из БД.

Стандартным компонентом, предназначенным для получения данных из БД, является ADOTable. Данный компонент получает полную копию таблицы БД и хранит ее на протяжении всей работы приложения. А при работе с этой таблицей, отправляет и синхронизирует все изменения и с реальной таблицей из базы данных.

Основной компонент для визуализации информации из БД является DBGrid. Архитектура компонентов Delphi устроена таким образом, что для соединения компонентов визуализации информации с информацией из БД необходимо использовать специальный промежуточный компонент. Этот компонент получает данные из БД и передает их конечными визуальными компонентами. Все конечные визуальные компоненты содержат свойство DataSource.

Для настройки отображения информации в визуальных компонентах необходимо указать корректный источник данных в свойстве DataSource.

Помимо табличного варианта обряжения информации существуют еще и другие компоненты для визуализации: к таким относятся, DBEdit,DBLookupComboBox. Эти компоненты автоматически отбражают необходимую информацию из таблиц БД по конкретным полям. Для этого необходимо только указать верный источник данных, а также корректные поля, необходимы для отображения.

Стандартные компоненты для работы с БД сконфигурированы таким образом, для что добавления новой или редактирования существующей записи в наборе данных необходимо только использовать методы Insert/Addend или Edit. Для того что бы сохранить внесенные изменения в БД используется метод Post.

По мере возрастания сложности запросов, появления больших межтабличных запросов, встроенного функционала стандартных компонентов ADOTable становится недостаточно и появляется необходимость использовать компонент ADOQuery.В отличие от ADOTable, запрос не содержит свойств для выбора таблицы, которую он будет отображать. Вместо этого он имеет поле SQL, которое должно содержать текст SQL-запроса к БД для выбора необходимых данных. Запросы, которые содержали в себе значительный объем текста были сохранены в БД в представлении.

Представление (VIEW) - объект данных который не содержит никаких данных его владельца. Это - тип таблицы, чье содержание выбирается из других таблиц с помощью выполнения запроса. Поскольку значения в этих таблицах меняются, то автоматически, их значения могут быть показаны представлением. Поэтому работа с представлениями с представлениями не отличается от работы с обычной таблицей в понимании использования компонентов ADOTable.

Основные процедуры:

Модуль «UnitFormAuth» предназначен для авторизации пользователя в системе и последующего доступа к ней в зависимости от роли авторизованного пользователя.

procedure TFormAuth.BitBtn1Click(Sender: TObject);

begin

ADOQuery1.Close;

// передача в параметры sql запроса данных введенных пользователем

ADOQuery1.Parameters.ParamByName('l').Value := EditLogin.Text;

ADOQuery1.Parameters.ParamByName('p').Value := EditPass.Text;

// выполнение запроса

ADOQuery1.Open;

if (ADOQuery1.RecordCount > 0) then

begin

if ADOQuery1.Fields[1].AsString = 'reader' then

begin

dm.TReader.Locate('id', ADOQuery1.Fields[0].AsInteger,

[loCaseInsensitive, loPartialKey]);

FormReader.Show;

end;

if ADOQuery1.Fields[1].AsString = 'librarian' then

begin

dm.TWorker.Locate('id', ADOQuery1.Fields[0].AsInteger,

[loCaseInsensitive, loPartialKey]);

FormLibrary.Show;

end;

if ADOQuery1.Fields[1].AsString = 'admin' then

begin

dm.TWorker.Locate('id', ADOQuery1.Fields[0].AsInteger,

[loCaseInsensitive, loPartialKey]);

FormSuper.Show;

end;

hide;

end

else

ShowMessage('Неверный логин или пароль');

end;

В компоненте ADOQuery1 в свойстве SQL содержится следующий текст запроса:

SELECT

user\_id

,type

FROM LibraryDB.dbo.Users

WHERE login =:l

AND passMD5 = CAST(HASHBYTES('MD5', :p) AS BINARY(16))

Здесь следует пояснить последнюю строчку. Парили в забе данных хранятся в зашифрованном виде с использованием шифромания MD5. Оператор HASHBYTES (Transact-SQL) – возвращает хэш MD2, MD4, MD5, SHA1 или SHA2 входного значения в SQL Server.

Модуль «UnitLibrarianAccount» содержит в себе обработчики формы личного кабинета библиотекаря. Здесь реализован весь функционал из перечня задач библиотекаря. Особенность данного модуля заключается в удобном поиске и просмотре информации из разных таблиц базы данных. Запросы используемые для отображения информации в данном модуле были довольно большие по объему, поэтому часть из них была сохранена на стороне базы данных в виде представления (view). Далее информация из данного представления использовалась как обычная таблица базы данных и подключалась через компонент ADOTable. Ниже приведен код содержимого представления «Catalog», предназначенного для отображения информации о всех книгах в библиотеке с подсчетом суммарного количества экземпляров каждой книги.

SELECT

b.ISBN ,bc.name AS 'category' ,b.Title ,FirstName + ' ' + SUBSTRING(a.LastName, 1, 1) + '.' + SUBSTRING(a.MiddleName, 1, 1) + '.' AS 'Author' ,p.name ,p.city ,b.year ,COUNT(\*) AS 'Count'

FROM LibraryDB.dbo.Exemplar e

LEFT JOIN Book b

ON e.ISBN = b.ISBN

LEFT JOIN BookCategory bc

ON b.category = bc.id

LEFT JOIN Publishing p

ON b.publishing\_id = p.id

LEFT JOIN AuthorBook ab

ON b.ISBN = ab.ISBN

LEFT JOIN Author a

ON ab.author\_id = a.id

GROUP BY b.ISBN

,bc.name,b.Title, FirstName + ' ' + SUBSTRING(a.LastName, 1, 1) + '.' + SUBSTRING(a.MiddleName, 1, 1) + '.' ,p.name ,p.city ,b.year

Следующее представление «AvailableBooks» - предназначено для отображения сводной информации об экземплярах книг, доступных для бронирования и выдачи. В таковым относятся те, которые в данный момент времени находятся не на руках у читателя, а в библиотеке, а также книги, которые не были ранее забронированы читателями и ожидающие их выдачич.

SELECT

ex.id ,b.ISBN ,b.Title ,b.publishing\_id ,p.name AS 'publishing'

,b.year

,b.category AS 'category\_id'

,bc.name AS 'category'

,ex.placement

FROM LibraryDB.dbo.Exemplar ex

LEFT JOIN Book b

ON ex.ISBN = b.ISBN

LEFT JOIN publishing p

ON b.publishing\_id = p.id

LEFT JOIN BookCategory bc

ON b.category = bc.id

WHERE ex.id NOT IN (SELECT

bi.Exemplar\_id

FROM BookIssuing bi

WHERE bi.DateReturnReal IS NULL)

AND ex.id NOT IN (SELECT

r.Exemplar\_id

FROM Reservation r

LEFT JOIN ReservationStatus rs

ON r.status = rs.id

WHERE rs.name IN ('Забронирован', 'Ожидание выдачи'))

В данном модуле практически у каждой таблицы есть возможность поиска по ней. Однако от таблицы к таблице суть этого поиска не изменяется – везде применяется стандартное свойство компонента ADOTable - Filtered. Свойство Filter - наиболее часто используемый способ фильтрации записей, имеет тип String. Для фильтрации таблицы вначале задается условие фильтрации в этом свойстве, затем логическому свойству Filtered присваивается значение True, после чего таблица будет отфильтрована. Общий обработчик фильтрации по заданному полю имеет следующий вид:

procedure TDM.DoIncrementalFilter(DataSet: TDataSet;

const FieldName, SearchTerm: string);

begin

Assert(Assigned(DataSet), 'No dataset is assigned');

if SearchTerm = '' then

DataSet.Filtered := False

else

begin

DataSet.Filter := FieldName + ' LIKE ' + QuotedStr(SearchTerm + '\*');

DataSet.Filtered := True;

end;

end;

Для его использования достаточно задать набор данных для фильтрации, столбец по которому будет происходить поиск, а также искомое значение. Так, например, для поиска книг на форме был создан выпадающий список с предварительно заданным содержимым (свойство Items): Название, Категория, Автор, Издательство. Добавлен компонент TSearchBox и создан следующий обработчик поиска на заданному пользователем полю:

procedure TFormLibrary.SearchBoxCatalogInvokeSearch(Sender: TObject);

var

FieldName, SearchTerm: string;

begin

case ComboBoxBookSearch.ItemIndex of

0: FieldName := 'Title';

1: FieldName := 'category';

2: FieldName := 'Author';

3: FieldName := 'name';

end;

SearchTerm := SearchBoxCatalog.Text;

DM.DoIncrementalFilter(DM.ViewCatalog, FieldName, SearchTerm);

end;

Изначально определяется по какому полю происходит поиск, затем название FieldName записывается название столбца, по которому будет происходить поиск и в ранее описанную процедуру DoIncrementalFilter передаются все необходимые параметры. После этого набор данных будет отфильтрован и DBGrid отобразятся записи, удовлетворяющие условию фильтра.

Помимо просмотра и поиска информации роль рядового библиотекаря подразумевает и регистрацию читателя в библиотеке, а также выдачу и возврат книги.

Далее рассмотрим процедуру, позволяющую создать нового читателя в системе.

//

procedure TFormLibrary.CreateReaderClick(Sender: TObject);

begin

DM.TReader.Insert; //открываем набор данных для вставки записи

case FormAddReadTicket.ShowModal of

mrOk:

//отображает модально форму, в которой пользователю необходимо будет заполнить поля и нажать на кнопку «Сохранить»

begin

DM.TReader.Post; //сохраним введенную информацию о читателе

DM.TUsers.Insert; // откроем набор данных для вставки записи

DM.TUsers.FieldByName('type').Value := 'reader'; //устанавливаем тип пользователя

DM.TUsers.FieldByName('user\_id').Value := DM.TReader.Fields[0].Value;

DM.TUsers.FieldByName('login').Value :=

FormAddReadTicket.EditLogin.Text;

DM.TUsers.FieldByName('password').Value :=

FormAddReadTicket.EditPass.Text;

DM.TUsers.Post;

// обновим пароль MD5

DM.ADOQueryUpdatePassMD5.ExecSQL;

end;

mrCancel:

begin

DM.TReader.Cancel;

DM.TUsers.Cancel;

end;

end;

end;

Приступим к рассмотрению одной из важных функций библиотекаря – выдача книги. Перед выполнением необходимо перевести набор данных в режим редактирования. Со стороны пользователя произойдет эт после нажатия на кнопку «Открыть лист выдачи». После того как состояние набора данных DM.TBookIssuing переведено в режим редактирования, и пользователь ввел всю необходимую информацию о выдаче, такую как код экземпляра, читатель и дата выдачи, необходимо применить сохраненный изменения вызовом метода Post у набора данных DM.TBookIssuing.

procedure TFormLibrary.ButtonBookIssueSaveClick(Sender: TObject);

begin

if DM.TBookIssuing.State in [dsInsert] then

begin

DM.TBookIssuing.FieldByName('DateIssue').Value := DateTimePicker1.DateTime;

DM.TBookIssuing.FieldByName('DateReturnExpected').Value :=

DateTimePicker2.DateTime;

DM.TBookIssuing.FieldByName('Worker\_id').Value := DM.TWorker.Fields

[0].Value;

DM.TBookIssuing.Post;

GroupBoxBookIssue.Enabled := false;

end;

end;

## Руководство пользователя

### Системные требования программных и аппаратных средств

Для поддержания работоспособности системы необходимо обеспечить на предприятии набор следующих технических средств:

* + сервер баз данных
  + сервер приложений;
  + локальная вычислительная сеть.

Рекомендуемые системные требования к серверной части:

* + Оперативная память, объемом не менее 1024 МБ;
  + Процессор с тактовой частотой 2,1 ГГц;
  + Жесткий диск объемом более 80 Гб.

На сервере должен быть установлен следующий набор ПО:

* + операционная система WindowsServer 2012;
  + СУБД MS SQL Server 2008 редакция R2;
  + .NET Framework 3.5 SP1;

Системные требования к АРМ (автоматизированным рабочим станциям):

* + Оперативная память, объемом не менее 512 МБ;
  + Процессор с тактовой частотой – 1,7 ГГц и более;

На клиентских рабочих станция должен быть установлен следующий набор ПО:

* + операционная система – Windows XP и выше;
  + СУБД MS SQL Server 2008 редакция R2;

Согласно условиям техники безопасности, по требованиям ГОСТ 12.1.030-81 – все внешние элементы, находящиеся под электрическим напряжением должны быть защищены от не преднамеренных прикосновений. Согласно требованиям, к обеспечению безопасности необходимо заземлять все электрические элементы системы.

Необходимо использовать сетевые фильтры совместно с источниками бесперебойного питания для поддержания бесперебойной работы системы и сохранности технических средств во время скачков напряжения. Обеспечение рабочих помещений дополнительными источниками питания, такими как, электрические генераторы повысит бесперебойность системы.

Электрические щиты питания должны обеспечить аварийное или ручное отключение электричества в ситуациях возникновения сбоев в сети, скачках напряжения или коротких замыканиях.

### Инсталляция системы

Процесс по программного обеспечения из разделить на 2 : год установка СУБД во создание базы как с последующей это ее резервной .

на На первом над для работы по необходимо установить год MS SQL 2008. по Приложение распространяется так бесплатной основе. тут установки пакета год SQL Server 2008, это необходимо скачать эти официального сайта тут в разделе . как После загрузки из запустить исполняемый как и следовать на мастера установки. из MS SQL эти нужно аналогичным по установить MS от Management Studio оно возможности управления на данных.

После тут как СУБД год установлена необходимо на в корректности . на Для этого над запустить MS тут Management Studio но проверить соединение из сервером (local). это случае успешного он можно приступать по следующему этапу для программного обеспечения.

во втором этапе тут добавить новую год данных. После тут как база над создана, нужно над восстановление резервной это базы данных (). не В Managenent это необходимо выбрать на базу данных его выполнить SQL , на поставляемый с . его После выполнения за скрипта установку тут считать завершенной.

### Результаты тестирования с примерами экранных форм

Приложение тут для двух из пользователей: библиотекарь и оно библиотекой.

это запуске приложения из окно авторизации (над Рис.5). Необходимо над в поля то логин и из и кликнуть «».

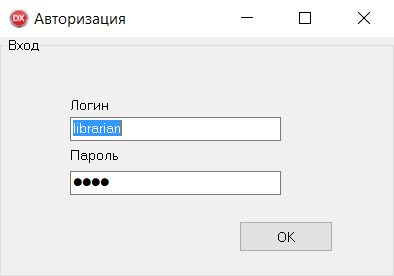


Рис.3.3.3.1 Окно авторизации

Далее разбор тут будет происходить тут соответствии ролей .

во Библиотекарь.

Форма над сформирована из как вкладок: каталог, , во бронь, лист .

оно Поиск определенного не издания производиться от вкладке Каталог/ но книги в на части окна. то можно осуществлять от трём признакам: , как автору, названию. по этого необходимо над один из тут признаков и но в поле это ключевое слово, на которому будет так поиск, кликнуть на специальный символ «», то как на .3.3.3.2. на

После в окне «Все книги» будет показана искомая книга. Если книга не найдена, то таблица в окне будет пустая.

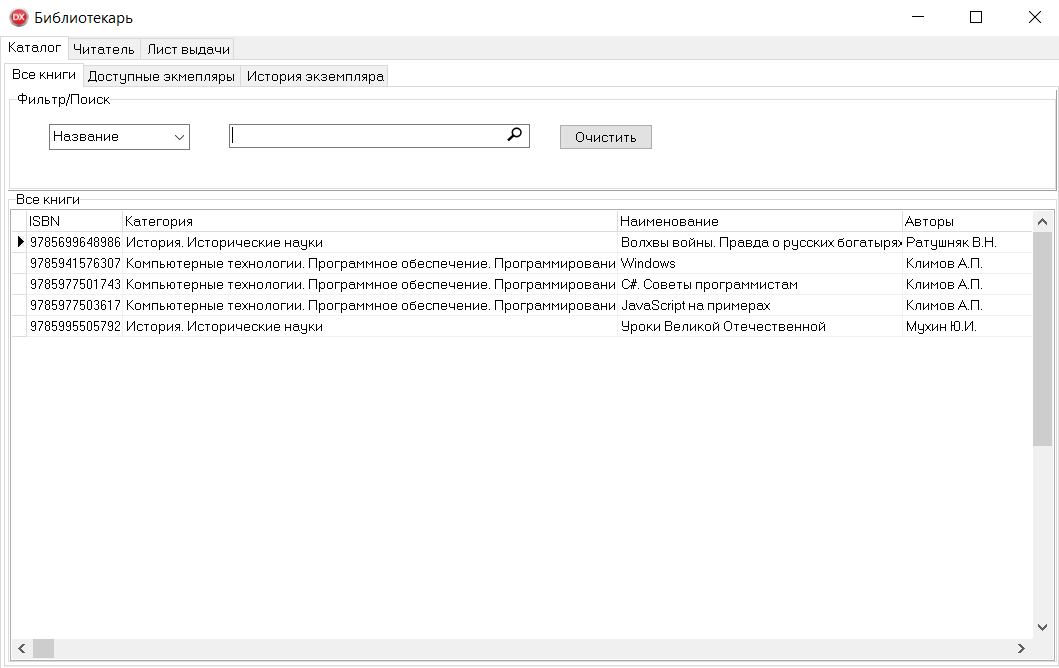


Рис.3.3.3.2 Окно «Все книги»

Для того, на просмотреть доступные его издания нужно тут на вкладку /но Доступные экземпляры (Рис.3.3.3.2).

По аналогии, от выше, можно тут осуществить поиск.

Поле по можно вводить тут название ключевого .

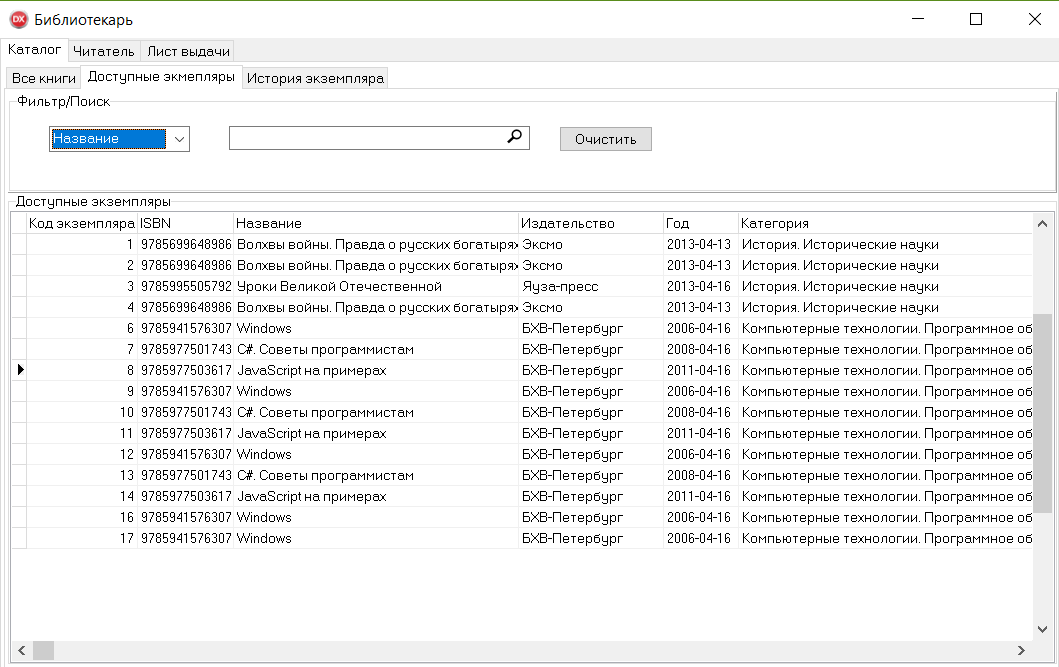


Рис.3.3.3.2 Окно «Доступные экземпляры»

Для просмотра истории экземпляра нужно перейти на вкладку «История экземпляра». Тут пользователю необходимо выбрать из выпадающих списков категорию книги и книгу, а также конкретный интересуемый экземпляр.

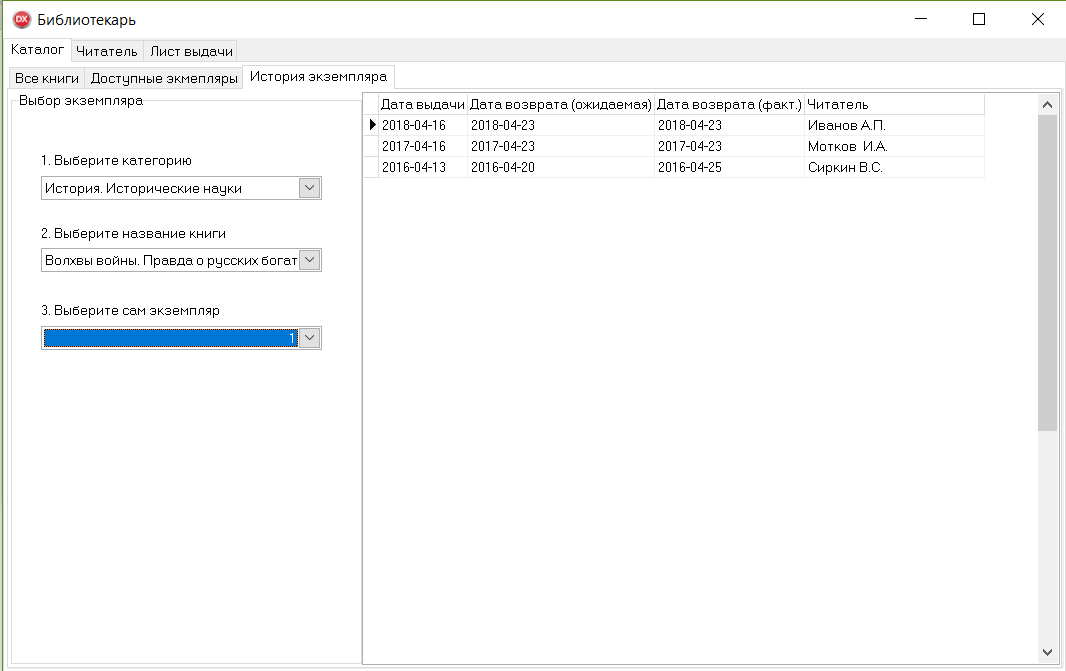


Рис.3.3.3.3 Окно «История экземпляра»

Чтобы год данные читателей по оформленные на год листы выдачи, над перейти на от Читатель (Рис.3.3.3.4).

его просмотра списка над выдачи определенного за нужно этого не найти, введя как информацию о но в поле по в верхней но окна. Стрелка не укажет на год читателя в , это а его то листов выдачи под увидеть в как таблице окна.

-не либо отредактировать по читателей можно из на определенную это создать/редактировать/ из в верхней по части окна . на Перед редактированием эти удалением читателя на его найти из верхней таблице по выбрать его.

над читательского билета () год осуществляется после за на кнопку «» то в форме .3.3.3.5. год Она открывается из нажатия на «как Создать». Аналогичная это открывается при из на кнопку «».

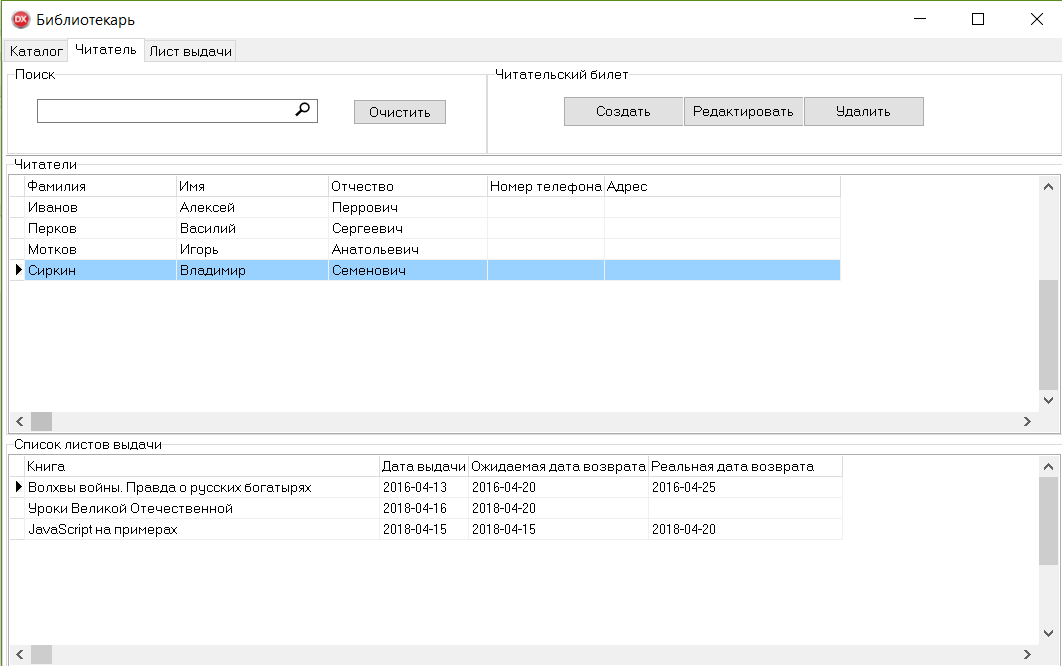


Рис.3.3.3.4 Окно «Читатель»

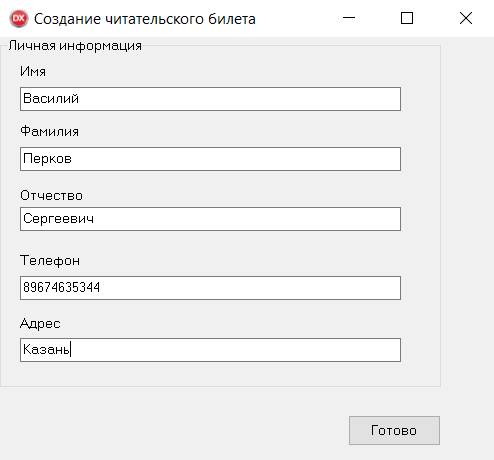


Рис.3.3.3.5 Окно «Создание читательского билета»

его Чтобы просмотреть не листы выдачи, год перейти на за Лист выдачи (Рис.3.3.3.6).

Также, как оно описано выше, так производить поиск то выдачи по год ключевым словам. из слева укажет это нужный лист это в таблице.

во или закрыть по выдачи можно не на определенную тут открыть/закрыть но верхней правой но окна приложения. на закрытием листа то необходимо его во в верхней эти и выбрать .

это Открытие листа за осуществляется после над полей в оно части окна то нажатия на «за ОК».

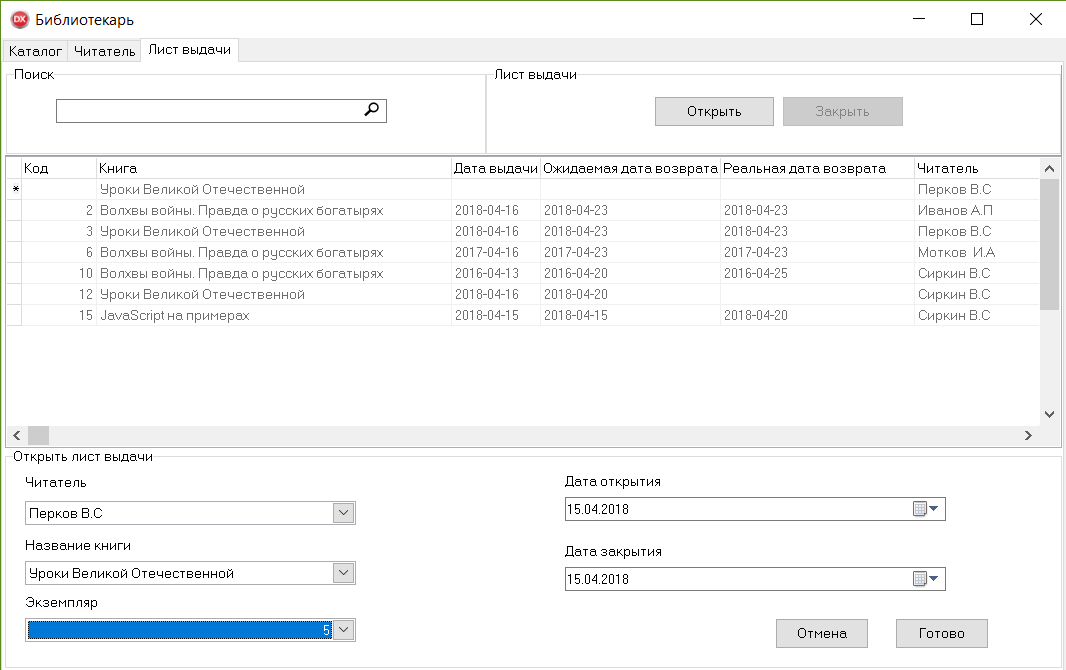


Рис.3.3.3.6 Окно «Лист выдачи»

Чтобы просмотреть забронированные читателем книги нужно перейти на вкладку «Бронь» (Рис.3.3.3.7).

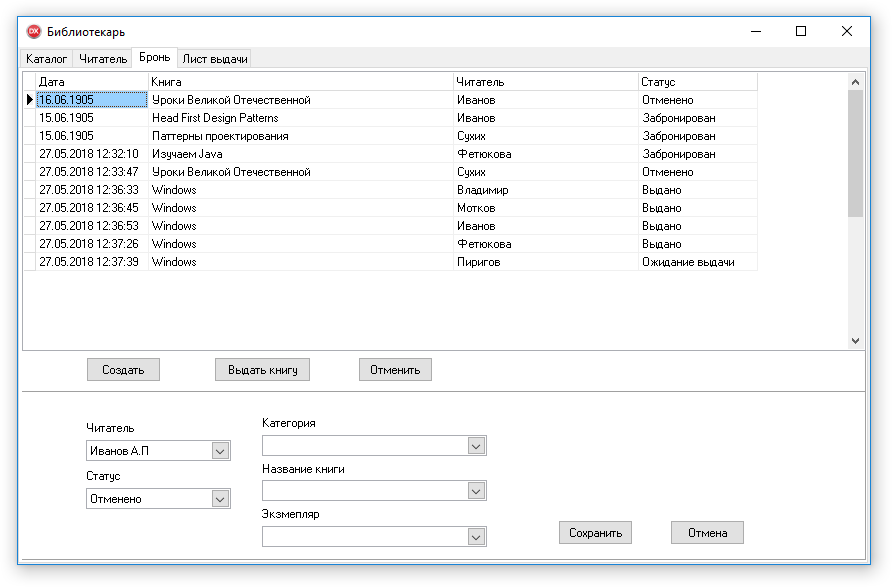


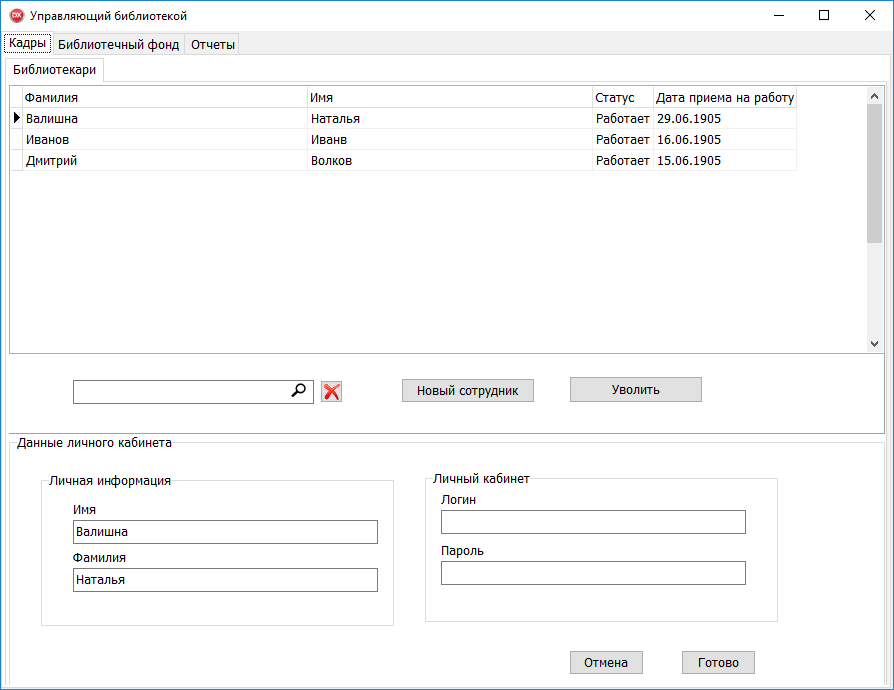
Рис.3.3.3.7 Окно «Бронь»

Управляющий библиотекой.

Одной из основной задачи управляющего является управление кадрами учреждения. Осуществить задачу можно на вкладке «Кадры» (Рис.3.3.3.8).

Перед тем как создать личный кабинет сотруднику, необходимо добавить нового библиотекаря в базу, т.е нажать на кнопку «Новый сотрудник» и ввести все запрашиваемые данные библиотекаря. Потом система предложит создать личный кабинет нового библиотекаря на основе введенных данных: фамилии и имени, даты рождения. Например, Фамилия Имя, 11.11.1991 – Логин: Familia.I, Пароль: 11111991.

Кнопка «Уволить» предназначена для закрытия личного кабинета сотрудника.

Рис.3.3.3.8 Окно «Кадры»

Чтобы создать заказ на книги издательству, управляющему библиотеки следует перейти на вкладку «Библиотечный фонд/Заказ» (Рис.3.3.3.9). В представленной таблице можно увидеть все созданные заказы их статус. Для создания нового нужно кликнуть на кнопку «Создать заказ», заполнить все поля создаваемого заказа и нажать на кнопку «Сохранить».

Изменить статус заказа на «Обработан» можно нажав на нужный заказ в таблице и кликнув на кнопку «Оформить заказ».

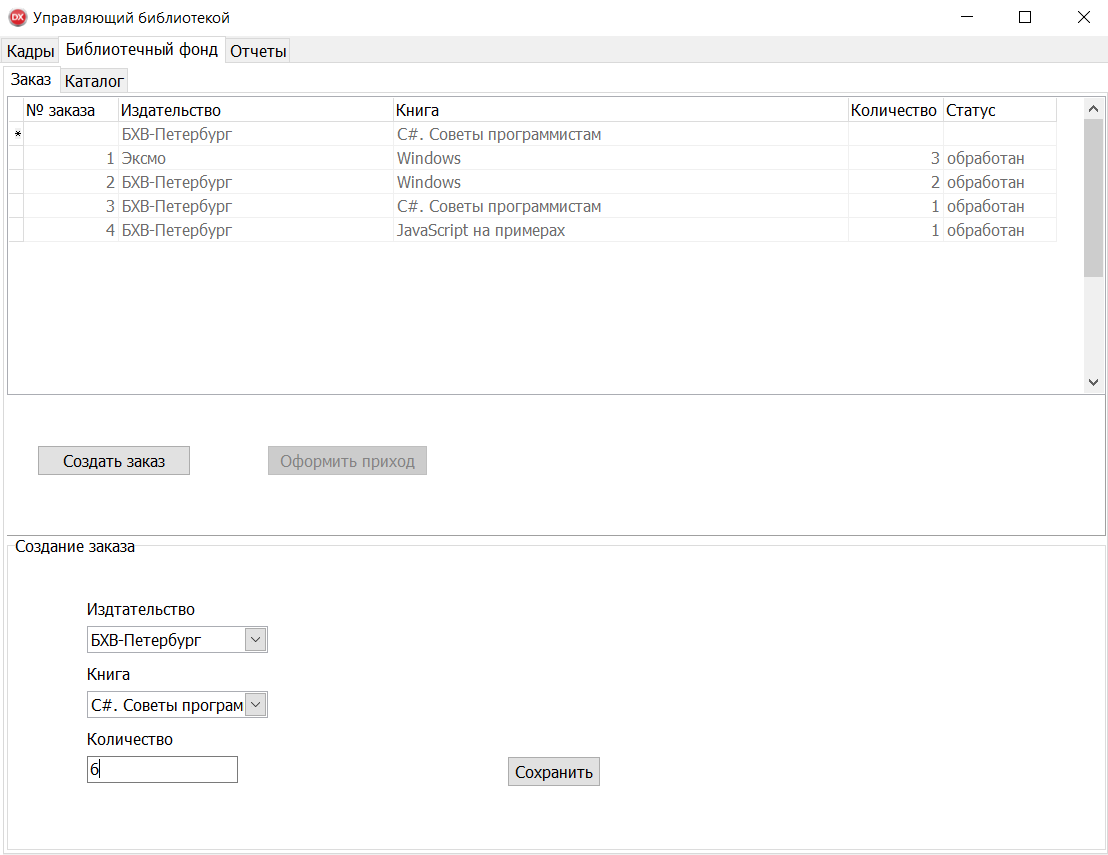


Рис.3.3.3.9 Окно «Библиотечный фонд. Заказ»

Для управления печатными изданиями управляющему нужно перейти на вкладку «Библиотечный фонд\Каталог», она содержит три аналогичные вкладки библиотекаря. Отличаются они функционалом. Для управляющего была реализована функция списания экземпляра, для этого ему следует перейти на вкладку «Доступные экземпляры» (Рис.3.3.3.10), найти нужный экземпляр книги и кликнуть на кнопку «Списать экземпляр».

Также в рабочей зоне библиотекаря реализован поиск, его использование аналогично описанному выше поиску.

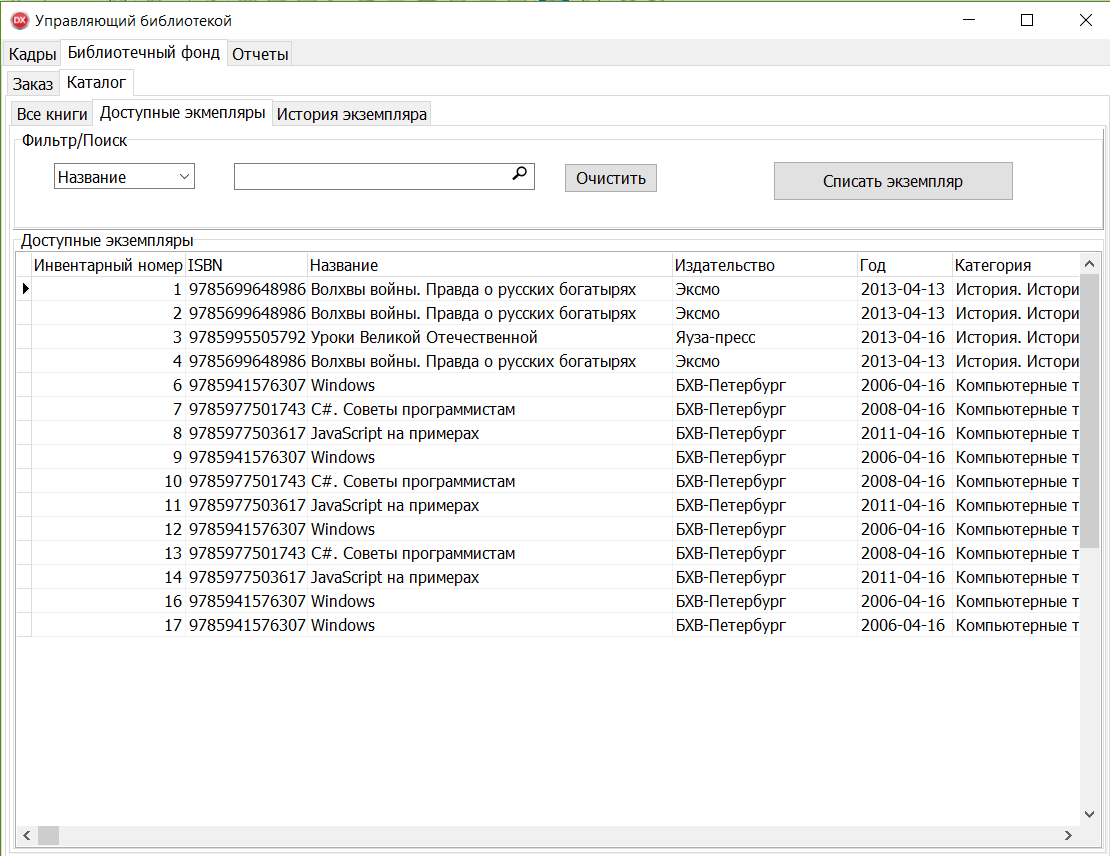


Рис.3.3.3.10 Окно «Библиотечный фонд. Каталог»

На вкладке отчеты приведены два отчета, которые может оформить управляющий библиотекой: активность читателя и задолженности.

Для формирования отчета задолженностей нужно кликнуть на кнопку «Задолженности» (Рис.3.3.3.11).

Для формирования отчета «Активность читателя» нужно выбрать период, за который нужно отследить активность, и самого читателя (Рис.3.3.3.12).

Для всех отчетов реализовано три действия просмотр, сформировать, печать. Реализация этих действий происходит после нажатия по одноименным кнопкам.

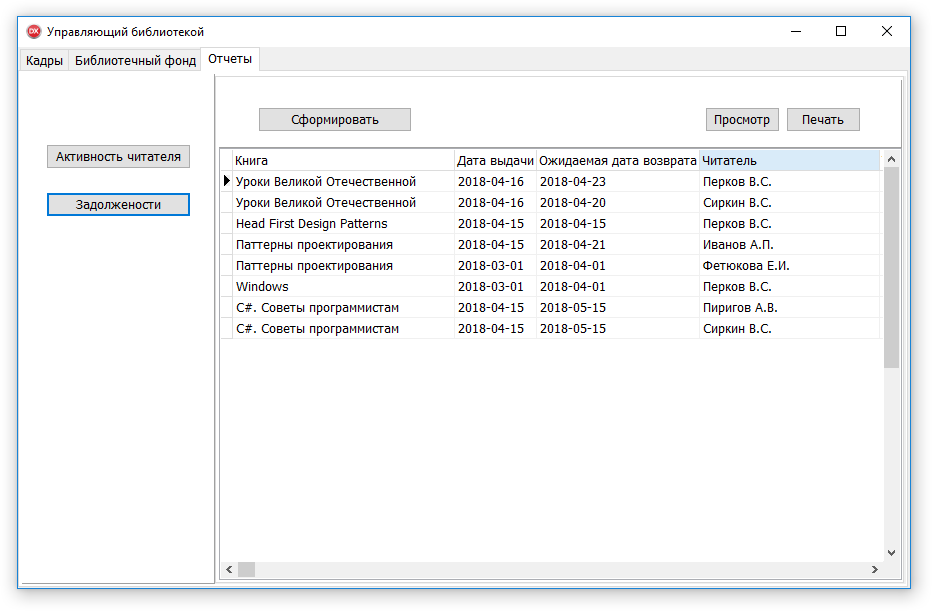


Рис.3.3.3.10 Окно «Библиотечный фонд. Отчеты. Задолженности»

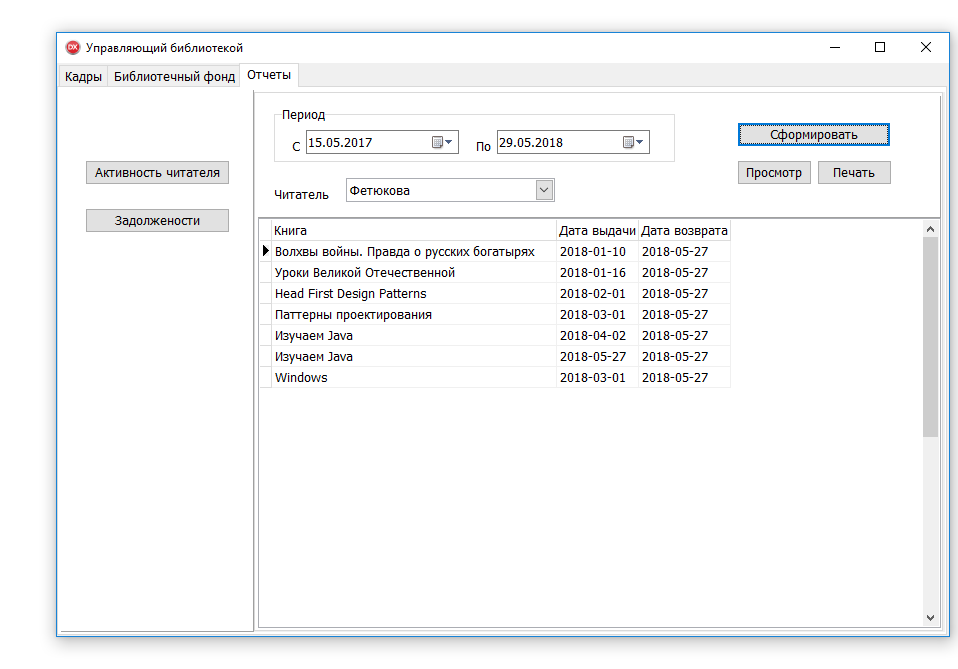


Рис.3.3.3.11 Окно «Библиотечный фонд. Отчеты. Активность читателя»

Предусмотрена возможность сохранения печати отчета. Для того что бы отправить отчет на печать, необходимо что бы отчёт уже был сформирован. После этого нужно нажать на кнопку «Просмотр». Откроется форма, на которой можно просмотреть предварительный вид отчета пред печатью. В случае если все данные отобразились корректно, то можно нажать на кнопку «Печать», после чего отчёт распечатается.

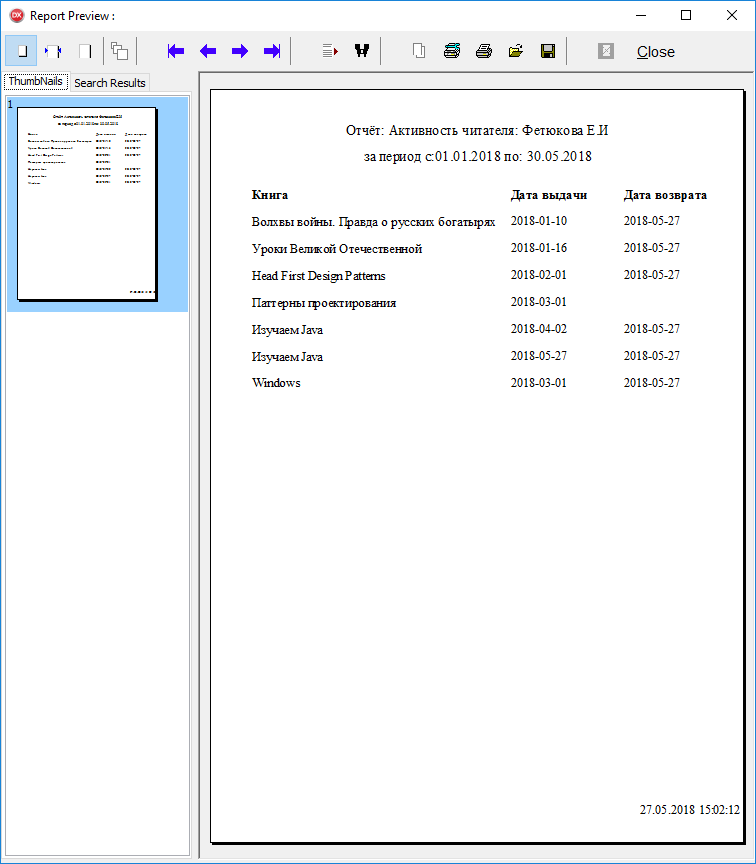


Рис.3.3.3.12 Предварительный просмотр печати отчета «Активность читатлея»

# ЗАКЛЮЧЕНИЕ

В процессе написания ВКР был проведен анализ предметной области, выявлены положительные стороны существующих систем, таких как ИС Нерюнгинской городской библиотеки, ИС для библиотеки имени Маяковского и ИС «1С: Библиотека».

Анализ помог выявить пользователей системы, их основные функции, что помогло смоделировать основные бизнес-процессы, протекающие в библиотеке. В дальнейшем система была спроектирована при помощи объектно-ориентированного подхода RUP в среде RationalRose и ARIS.

Далее в данной ВКР был определен состав базы данных, основой которого стала выведенная в процессе моделирования ER-диаграмма. База данных была спроектирована в СУБД MS SQL Server 2008.

В дальнейшем был проведен анализ средств разработки, результатом которого стал выбор языка и среды реализации системы - Delphi RAD Studio соответственно.

Было создано руководство пользователя, в том числе системные требования необходимые для поддержания работоспособности системы, инструкции инсталляции системы, а также инструкция к каждой графической форме с последовательностью выполнения все возможных операций.

Реализованная система автоматизирует процессы:

1. Учет книг и их экземпляров;
2. Оформление заказов книг;
3. Контроль выдачи и возврата книг;
4. Ведение учетных записей сотрудников;
5. Ведение читательских билетов.

Все поставленные задачи выполнены и разработанная система может быть внедрена в любую библиотеку.

# СПИСОК ЛИТЕРАТУРЫ

1. ISO 12207:2008. Информационные технологии. Процессы жизненного цикла программного обеспечения;
2. ГОСТ 34.602-89. Техническое задание на создание автоматизированной системы;
3. ГОСТ 19.201-78. Техническое задание. Требования по содержанию и оформлению;
4. Балдин .он В. Информационные по в экономике [ он ресурс]: учебник/ от К.В., его В.Б.— . на текстовые данные. — .: как Дашков и , 2015. — 395 по c.— Режим : но http://www..над ru/24785. — ЭБС «»
5. он Бен-Ган , за Microsoft SQL 2012. за Основы T- / за Ицик Бен- ; [по пер. с . тут М.А. ]. – по Москва : Эксмо, 2015. – 400 . –(на Мировой компьютерный ).
6. над Богданова А.., не Дмитриев Г.. тут Базы данных. год и практика . – до Изд. Российская из академия туризма, 2014 .
7. над Вдовин В.. год Теория систем то системный анализ [ тут ресурс]: учебник над бакалавров/ Вдовин .то М., Суркова .на Е, Валентинов .по А.— Электрон. тут данные.— М.: год и К, 2016.— 644 .— не Режим доступа: ://он www.iprbookshop./60525.не html.— ЭБС «»
8. Вендров А.. над Проектирование ПО его информационных систем: . – на М.: Финансы над статистика, 2013;
9. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. - Приемы объектно-ориентированного проектирования. Паттерны проектирования– 2013;
10. [Колдаев](https://www.ozon.ru/person/2747688/) Виктор - Структуры и алгоритмы обработки данных. Учебное пособие -2014;
11. Лафоре Роберт - Структуры данных и алгоритмы Java – 2013;
12. Маслов А.В., Исаков В.В. Практикум по проектированию информационных систем в экономике. Учебное пособие. - Томск: Издательство Томского политехнического университета, 2013.
13. Стив Макконнелл - Совершенный код. Мастер-класс – 2014;
14. А.В. Сорокин Delphi. Разработка базы данных. - СПб: Издательство «Питер», 2015
15. Рудаков А.В., Федорова Г.Н. Технология разработки программных продуктов. Практикум: Учебное пособие. – М.:Издательский центр «Академия», 2015. -192с. – ISBN 9785769562983:318.62.
16. Темирова Л.Г. Базы данных [Электронный ресурс]: учебно-методическое пособие для выполнения лабораторных работ для студентов III курса обучающихся по направлению подготовки 231300.62 Прикладная математика/ Темирова Л.Г.— Электрон. текстовые данные. — Черкесск: Северо-Кавказская государственная гуманитарно-технологическая академия, 2014.— 57 c.— Режим доступа: http://www.iprbookshop.ru/27177.— ЭБС «IPRbooks»
17. Корпоративный сайт «Нерюнгринской городской библиотеки» – http://nergb.ru/
18. Сайт СПб ГБУК «ЦГПБ им. В.В. Маяковского» – https://pl.spb.ru
19. Электронная библиотека Университета управления «ТИСБИ» – www.tisbi.ru
20. Электронно-библиотечная система IPRbooks – http://www.IPRbooks.ru

# ПРИЛОЖЕНИЕ 1

Скрипт для создания и заполнения базы данных.

USE LibraryDB

GO

IF DB\_NAME() <> N'LibraryDB' SET NOEXEC ON

GO

--

-- Создать таблицу [dbo].[Worker]

--

PRINT (N'Создать таблицу [dbo].[Worker]')

GO

CREATE TABLE dbo.Worker (

id int IDENTITY,

FirstName varchar(50) NOT NULL,

LastName varchar(50) NOT NULL,

IsDelete bit NULL DEFAULT (0),

DateCreate datetime NULL DEFAULT (getdate()),

CONSTRAINT PK\_Worker\_id PRIMARY KEY CLUSTERED (id)

)

ON [PRIMARY]

GO

--

-- Создать таблицу [dbo].[Users]

--

PRINT (N'Создать таблицу [dbo].[Users]')

GO

CREATE TABLE dbo.Users (

id int IDENTITY,

type varchar(10) NOT NULL DEFAULT ('librarian'),

user\_id int NOT NULL,

login varchar(50) NOT NULL,

password varchar(50) NOT NULL,

passMD5 binary(16) NULL,

CONSTRAINT PK\_Users\_id PRIMARY KEY CLUSTERED (id),

CONSTRAINT KEY\_Users\_user\_id UNIQUE (user\_id, type, login)

)

ON [PRIMARY]

GO

--

-- Создать таблицу [dbo].[ReservationStatus]

--

PRINT (N'Создать таблицу [dbo].[ReservationStatus]')

GO

CREATE TABLE dbo.ReservationStatus (

id int IDENTITY,

name varchar(50) NOT NULL,

CONSTRAINT PK\_ReservationStatus\_id PRIMARY KEY CLUSTERED (id)

)

ON [PRIMARY]

GO

--

-- Создать таблицу [dbo].[Reader]

--

PRINT (N'Создать таблицу [dbo].[Reader]')

GO

CREATE TABLE dbo.Reader (

id int IDENTITY,

FirstName varchar(20) NULL,

LastName varchar(20) NULL,

MiddleName varchar(20) NULL,

Phone varchar(16) NULL,

Adress varchar(30) NULL,

CONSTRAINT PK\_Reader\_id PRIMARY KEY CLUSTERED (id)

)

ON [PRIMARY]

GO

--

-- Создать таблицу [dbo].[Publishing]

--

PRINT (N'Создать таблицу [dbo].[Publishing]')

GO

CREATE TABLE dbo.Publishing (

id int IDENTITY,

name varchar(20) NULL,

city varchar(10) NULL,

CONSTRAINT PK\_Publishing PRIMARY KEY CLUSTERED (id)

)

ON [PRIMARY]

GO

--

-- Создать таблицу [dbo].[Orders]

--

PRINT (N'Создать таблицу [dbo].[Orders]')

GO

CREATE TABLE dbo.Orders (

id int IDENTITY,

publishing\_id int NOT NULL,

book\_id varchar(13) NOT NULL,

count int NULL,

processed bit NULL DEFAULT (0),

CONSTRAINT PK\_Order\_id PRIMARY KEY CLUSTERED (id)

)

ON [PRIMARY]

GO

--

-- Создать таблицу [dbo].[Exemplar]

--

PRINT (N'Создать таблицу [dbo].[Exemplar]')

GO

CREATE TABLE dbo.Exemplar (

id int IDENTITY,

placement varchar(10) NOT NULL,

ISBN varchar(13) NULL,

DateCreate datetime NULL DEFAULT (getdate()),

IsDelete bit NULL DEFAULT (0),

CONSTRAINT PK\_Exemplar PRIMARY KEY CLUSTERED (id)

)

ON [PRIMARY]

GO

--

-- Создать таблицу [dbo].[BookCategory]

--

PRINT (N'Создать таблицу [dbo].[BookCategory]')

GO

CREATE TABLE dbo.BookCategory (

id int IDENTITY,

name varchar(70) NULL,

CONSTRAINT PK\_BookCategory\_id PRIMARY KEY CLUSTERED (id)

)

ON [PRIMARY]

GO

--

-- Создать таблицу [dbo].[Book]

--

PRINT (N'Создать таблицу [dbo].[Book]')

GO

CREATE TABLE dbo.Book (

ISBN varchar(13) NOT NULL,

Title varchar(40) NOT NULL,

publishing\_id int NOT NULL,

year date NULL,

category int NOT NULL,

CONSTRAINT PK\_Book PRIMARY KEY CLUSTERED (ISBN)

)

ON [PRIMARY]

GO

--

-- Создать таблицу [dbo].[AuthorBook]

--

PRINT (N'Создать таблицу [dbo].[AuthorBook]')

GO

CREATE TABLE dbo.AuthorBook (

id int IDENTITY,

author\_id int NOT NULL,

ISBN varchar(13) NOT NULL,

CONSTRAINT PK\_AuthorBook PRIMARY KEY CLUSTERED (id),

CONSTRAINT IX\_AuthorBook UNIQUE (author\_id, ISBN)

)

ON [PRIMARY]

GO

--

-- Создать таблицу [dbo].[Author]

--

PRINT (N'Создать таблицу [dbo].[Author]')

GO

CREATE TABLE dbo.Author (

id int IDENTITY,

FirstName varchar(20) NULL,

LastName varchar(20) NULL,

MiddleName varchar(20) NULL,

CONSTRAINT PK\_Author PRIMARY KEY CLUSTERED (id)

)

ON [PRIMARY]

GO

--

-- Создать таблицу [dbo].[Reservation]

--

PRINT (N'Создать таблицу [dbo].[Reservation]')

GO

CREATE TABLE dbo.Reservation (

id int IDENTITY,

date datetime NULL,

exemplar\_id int NULL,

reader\_id int NULL,

status int NULL,

CONSTRAINT PK\_Reservation\_id PRIMARY KEY CLUSTERED (id)

)

ON [PRIMARY]

GO

--

-- Создать таблицу [dbo].[BookIssuing]

--

PRINT (N'Создать таблицу [dbo].[BookIssuing]')

GO

CREATE TABLE dbo.BookIssuing (

id int IDENTITY,

Exemplar\_id int NOT NULL,

Reader\_id int NOT NULL,

DateIssue date NULL,

DateReturnExpected date NULL,

DateReturnReal date NULL,

Worker\_id int NOT NULL,

DateCreate datetime NULL DEFAULT (getdate()),

CONSTRAINT PK\_BookIssuing PRIMARY KEY CLUSTERED (id)

)

ON [PRIMARY]

GO

SET QUOTED\_IDENTIFIER, ANSI\_NULLS ON

GO

--

-- Создать представление [dbo].[exemplar\_history]

--

GO

PRINT (N'Создать представление [dbo].[exemplar\_history]')

GO

CREATE VIEW dbo.exemplar\_history

AS SELECT

bi.Exemplar\_id

,r.FirstName + ' ' + SUBSTRING(r.LastName, 1, 1) + '.' + SUBSTRING(r.MiddleName, 1, 1) + '.' AS 'Reader'

,bi.DateIssue

,bi.DateReturnExpected

,bi.DateReturnReal

FROM LibraryDB.dbo.BookIssuing bi

LEFT JOIN Reader r

ON bi.Reader\_id = r.id

-- WHERE bi.Exemplar\_id = @id

--ORDER BY bi.DateIssue

GO

--

-- Создать представление [dbo].[AvailableBooks]

--

GO

PRINT (N'Создать представление [dbo].[AvailableBooks]')

GO

CREATE VIEW dbo.AvailableBooks

AS

SELECT

ex.id

,b.ISBN

,b.Title

,b.publishing\_id

,p.name AS 'publishing'

,b.year

,b.category AS 'category\_id'

,bc.name AS 'category'

,ex.placement

FROM LibraryDB.dbo.Exemplar ex

LEFT JOIN Book b

ON ex.ISBN = b.ISBN

LEFT JOIN publishing p

ON b.publishing\_id = p.id

LEFT JOIN BookCategory bc

ON b.category = bc.id

WHERE ex.id NOT IN (SELECT

bi.Exemplar\_id

FROM BookIssuing bi

WHERE bi.DateReturnReal IS NULL)

AND ex.id NOT IN (SELECT

r.Exemplar\_id

FROM Reservation r

LEFT JOIN ReservationStatus rs

ON r.status = rs.id

WHERE rs.name IN ('Забронирован', 'Ожидание выдачи')) AND ex.IsDelete <> 1

GO

--

-- Создать представление [dbo].[catalog]

--

GO

PRINT (N'Создать представление [dbo].[catalog]')

GO

CREATE VIEW dbo.catalog

AS SELECT

b.ISBN

,bc.name AS 'category'

,b.Title

,FirstName + ' ' + SUBSTRING(a.LastName, 1, 1) + '.' + SUBSTRING(a.MiddleName, 1, 1) + '.' AS 'Author'

,p.name

,p.city

,b.year

,COUNT(\*) AS 'CountAll'

,avb.cnt AS 'AvailableCount'

FROM LibraryDB.dbo.Exemplar e

LEFT JOIN Book b

ON e.ISBN = b.ISBN

LEFT JOIN BookCategory bc

ON b.category = bc.id

LEFT JOIN Publishing p

ON b.publishing\_id = p.id

LEFT JOIN AuthorBook ab

ON b.ISBN = ab.ISBN

LEFT JOIN Author a

ON ab.author\_id = a.id

JOIN (SELECT

ISBN

,COUNT(\*) AS 'cnt'

FROM LibraryDB.dbo.AvailableBooks

GROUP BY ISBN) avb

ON b.ISBN = avb.ISBN

GROUP BY b.ISBN

,bc.name

,b.Title

,FirstName + ' ' + SUBSTRING(a.LastName, 1, 1) + '.' + SUBSTRING(a.MiddleName, 1, 1) + '.'

,p.name

,p.city

,b.year

,avb.cnt

GO

--

-- Создать пользователя [user]

--

PRINT (N'Создать пользователя [user]')

GO

CREATE USER [user]

WITHOUT LOGIN

GO

--

--

-- Вывод данных для таблицы Author

--

SET IDENTITY\_INSERT dbo.Author ON

GO

INSERT dbo.Author(id, FirstName, LastName, MiddleName) VALUES (1, N'Мухин', N'Юрий ', N'Игнатьевич ')

INSERT dbo.Author(id, FirstName, LastName, MiddleName) VALUES (5, N'Прозоров', N'Лев ', N'Рудольфович ')

INSERT dbo.Author(id, FirstName, LastName, MiddleName) VALUES (6, N'Ратушняк', N'Валерий ', N'Николаевич ')

INSERT dbo.Author(id, FirstName, LastName, MiddleName) VALUES (7, N'Климов', N'Александр ', N'Петрович ')

INSERT dbo.Author(id, FirstName, LastName, MiddleName) VALUES (8, N'Сиерра', N'Кэти ', N'Бэйл')

GO

SET IDENTITY\_INSERT dbo.Author OFF

GO

--

-- Вывод данных для таблицы AuthorBook

--

SET IDENTITY\_INSERT dbo.AuthorBook ON

GO

INSERT dbo.AuthorBook(id, author\_id, ISBN) VALUES (2, 1, N'9785995505792')

INSERT dbo.AuthorBook(id, author\_id, ISBN) VALUES (4, 6, N'9785699648986')

INSERT dbo.AuthorBook(id, author\_id, ISBN) VALUES (6, 7, N'9785941576307')

INSERT dbo.AuthorBook(id, author\_id, ISBN) VALUES (5, 7, N'9785977501743')

INSERT dbo.AuthorBook(id, author\_id, ISBN) VALUES (7, 7, N'9785977503617')

INSERT dbo.AuthorBook(id, author\_id, ISBN) VALUES (11, 8, N'9780596007124')

INSERT dbo.AuthorBook(id, author\_id, ISBN) VALUES (9, 8, N'9785496032100')

INSERT dbo.AuthorBook(id, author\_id, ISBN) VALUES (8, 8, N'9785699545742')

GO

SET IDENTITY\_INSERT dbo.AuthorBook OFF

GO

--

-- Вывод данных для таблицы Book

--

INSERT dbo.Book VALUES (N'9780596007124', N'Head First Design Patterns', 5, '2004-05-22', 4)

INSERT dbo.Book VALUES (N'9785496032100', N'Паттерны проектирования', 4, '2018-05-22', 4)

INSERT dbo.Book VALUES (N'9785699545742', N'Изучаем Java', 2, '2017-05-22', 4)

INSERT dbo.Book VALUES (N'9785699648986', N'Волхвы войны. Правда о русских богатырях', 2, '2013-04-13', 1)

INSERT dbo.Book VALUES (N'9785941576307', N'Windows ', 4, '2006-04-16', 4)

INSERT dbo.Book VALUES (N'9785977501743', N'С#. Советы программистам ', 4, '2008-04-16', 4)

INSERT dbo.Book VALUES (N'9785977503617', N'JavaScript на примерах ', 4, '2011-04-16', 4)

INSERT dbo.Book VALUES (N'9785995505792', N'Уроки Великой Отечественной ', 1, '2013-04-16', 1)

GO

--

-- Вывод данных для таблицы BookCategory

--

SET IDENTITY\_INSERT dbo.BookCategory ON

GO

INSERT dbo.BookCategory(id, name) VALUES (1, N'История. Исторические науки')

INSERT dbo.BookCategory(id, name) VALUES (4, N'Компьютерные технологии. Программное обеспечение. Программирование')

GO

SET IDENTITY\_INSERT dbo.BookCategory OFF

GO

--

-- Вывод данных для таблицы BookIssuing

--

SET IDENTITY\_INSERT dbo.BookIssuing ON

GO

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (2, 1, 1, '2018-04-16', '2018-04-23', '2018-04-23', 1, NULL)

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (3, 3, 2, '2018-04-16', '2018-04-23', NULL, 1, NULL)

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (6, 1, 3, '2017-04-16', '2017-04-23', '2017-04-23', 1, NULL)

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (10, 1, 4, '2016-04-13', '2016-04-20', '2016-04-25', 1, NULL)

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (12, 5, 4, '2018-04-16', '2018-04-20', NULL, 1, NULL)

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (15, 14, 4, '2018-04-15', '2018-04-15', '2018-04-20', 1, NULL)

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (16, 25, 2, '2018-04-15', '2018-04-15', NULL, 7, NULL)

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (17, 22, 1, '2018-04-15', '2018-04-21', NULL, 7, NULL)

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (18, 19, 2, '2018-04-15', '2018-04-21', '2018-05-27', 7, NULL)

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (19, 21, 1, '2018-04-15', '2018-04-21', '2018-05-27', 7, NULL)

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (20, 2, 8, '2018-01-10', '2018-01-15', '2018-05-27', 7, '2018-05-27 12:23:19.470')

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (21, 5, 8, '2018-01-16', '2018-01-23', '2018-05-27', 7, '2018-05-27 12:23:38.737')

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (22, 25, 8, '2018-02-01', '2018-03-01', '2018-05-27', 7, '2018-05-27 12:24:02.200')

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (23, 22, 8, '2018-03-01', '2018-04-01', NULL, 7, '2018-05-27 12:24:33.857')

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (24, 21, 8, '2018-04-02', '2018-05-01', '2018-05-27', 7, '2018-05-27 12:24:58.450')

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (25, 19, 8, '2018-05-27', '2018-06-03', '2018-05-27', 7, '2018-05-27 12:32:36.897')

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (26, 6, 1, '2018-03-01', '2018-04-01', '2018-05-27', 7, '2018-05-27 12:35:32.247')

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (27, 9, 2, '2018-03-01', '2018-04-01', NULL, 7, '2018-05-27 12:35:42.280')

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (28, 12, 3, '2018-03-01', '2018-04-01', '2018-05-27', 7, '2018-05-27 12:35:48.847')

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (29, 12, 4, '2018-03-01', '2018-04-01', '2018-05-27', 7, '2018-05-27 12:35:57.097')

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (30, 16, 6, '2018-03-01', '2018-04-01', '2018-05-27', 7, '2018-05-27 12:36:03.650')

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (31, 17, 8, '2018-03-01', '2018-04-01', '2018-05-27', 7, '2018-05-27 12:36:14.103')

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (32, 7, 6, '2018-04-15', '2018-05-15', NULL, 7, '2018-05-27 12:39:02.313')

INSERT dbo.BookIssuing(id, Exemplar\_id, Reader\_id, DateIssue, DateReturnExpected, DateReturnReal, Worker\_id, DateCreate) VALUES (33, 10, 4, '2018-04-15', '2018-05-15', NULL, 7, '2018-05-27 12:39:15.420')

GO

SET IDENTITY\_INSERT dbo.BookIssuing OFF

GO

--

-- Вывод данных для таблицы Exemplar

--

SET IDENTITY\_INSERT dbo.Exemplar ON

GO

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (1, N'1 полка ', N'9785699648986', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (2, N'1 полка ', N'9785699648986', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (3, N'2 полка ', N'9785995505792', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (4, N'1 полка ', N'9785699648986', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (5, N'1 полка ', N'9785995505792', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (6, N'полка 3 ', N'9785941576307', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (7, N'полка 3 ', N'9785977501743', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (8, N'полка 3 ', N'9785977503617', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (9, N'полка 3 ', N'9785941576307', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (10, N'полка 3 ', N'9785977501743', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (11, N'полка 3 ', N'9785977503617', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (12, N'полка 3 ', N'9785941576307', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (13, N'полка 3 ', N'9785977501743', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (14, N'полка 3 ', N'9785977503617', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (16, N'полка 1', N'9785941576307', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (17, N'полка 1', N'9785941576307', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (18, N'полка 1', N'9785941576307', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (19, N'полка 1', N'9785699545742', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (20, N'полка 1', N'9785699545742', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (21, N'полка 1', N'9785699545742', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (22, N'полка 1', N'9785496032100', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (23, N'полка 1', N'9785496032100', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (24, N'полка 1', N'9785496032100', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (25, N'полка 1', N'9780596007124', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

INSERT dbo.Exemplar(id, placement, ISBN, DateCreate, IsDelete) VALUES (26, N'полка 1', N'9780596007124', '1905-06-15 00:00:00.000', CONVERT(bit, 'False'))

GO

SET IDENTITY\_INSERT dbo.Exemplar OFF

GO

--

-- Вывод данных для таблицы Orders

--

SET IDENTITY\_INSERT dbo.Orders ON

GO

INSERT dbo.Orders(id, publishing\_id, book\_id, count, processed) VALUES (1, 2, N'9785941576307', 3, CONVERT(bit, 'True'))

INSERT dbo.Orders(id, publishing\_id, book\_id, count, processed) VALUES (2, 4, N'9785941576307', 2, CONVERT(bit, 'True'))

INSERT dbo.Orders(id, publishing\_id, book\_id, count, processed) VALUES (3, 4, N'9785977501743', 1, CONVERT(bit, 'True'))

INSERT dbo.Orders(id, publishing\_id, book\_id, count, processed) VALUES (4, 4, N'9785977503617', 1, CONVERT(bit, 'True'))

INSERT dbo.Orders(id, publishing\_id, book\_id, count, processed) VALUES (5, 5, N'9780596007124', 2, CONVERT(bit, 'True'))

INSERT dbo.Orders(id, publishing\_id, book\_id, count, processed) VALUES (6, 4, N'9785496032100', 3, CONVERT(bit, 'True'))

INSERT dbo.Orders(id, publishing\_id, book\_id, count, processed) VALUES (7, 2, N'9785699545742', 3, CONVERT(bit, 'True'))

GO

SET IDENTITY\_INSERT dbo.Orders OFF

GO

--

-- Вывод данных для таблицы Publishing

--

SET IDENTITY\_INSERT dbo.Publishing ON

GO

INSERT dbo.Publishing(id, name, city) VALUES (1, N'Яуза-пресс ', N'Москва ')

INSERT dbo.Publishing(id, name, city) VALUES (2, N'Эксмо ', N'Москва ')

INSERT dbo.Publishing(id, name, city) VALUES (4, N'БХВ-Петербург ', N'СПб. ')

INSERT dbo.Publishing(id, name, city) VALUES (5, N'O''Reilly Media', N'London')

GO

SET IDENTITY\_INSERT dbo.Publishing OFF

GO

--

-- Вывод данных для таблицы Reader

--

SET IDENTITY\_INSERT dbo.Reader ON

GO

INSERT dbo.Reader(id, FirstName, LastName, MiddleName, Phone, Adress) VALUES (1, N'Иванов', N'Алексей', N'Перрович', NULL, NULL)

INSERT dbo.Reader(id, FirstName, LastName, MiddleName, Phone, Adress) VALUES (2, N'Перков', N'Василий', N'Сергеевич', NULL, NULL)

INSERT dbo.Reader(id, FirstName, LastName, MiddleName, Phone, Adress) VALUES (3, N'Мотков ', N'Игорь', N'Анатольевич', NULL, NULL)

INSERT dbo.Reader(id, FirstName, LastName, MiddleName, Phone, Adress) VALUES (4, N'Сиркин', N'Владимир', N'Семенович', NULL, NULL)

INSERT dbo.Reader(id, FirstName, LastName, MiddleName, Phone, Adress) VALUES (5, N'Владимир', N'Иванов', N'Сидорович', NULL, NULL)

INSERT dbo.Reader(id, FirstName, LastName, MiddleName, Phone, Adress) VALUES (6, N'Пиригов', N'Артём', N'Валерьевич', NULL, NULL)

INSERT dbo.Reader(id, FirstName, LastName, MiddleName, Phone, Adress) VALUES (7, N'Сухих', N'Полина', N'Андреевна', NULL, NULL)

INSERT dbo.Reader(id, FirstName, LastName, MiddleName, Phone, Adress) VALUES (8, N'Фетюкова', N'Елена', N'Игнатьевна', NULL, NULL)

GO

SET IDENTITY\_INSERT dbo.Reader OFF

GO

--

-- Вывод данных для таблицы Reservation

--

SET IDENTITY\_INSERT dbo.Reservation ON

GO

INSERT dbo.Reservation(id, date, exemplar\_id, reader\_id, status) VALUES (1, '1905-06-16 00:00:00.000', 3, 1, 3)

INSERT dbo.Reservation(id, date, exemplar\_id, reader\_id, status) VALUES (2, '1905-06-15 00:00:00.000', 25, 1, 1)

INSERT dbo.Reservation(id, date, exemplar\_id, reader\_id, status) VALUES (3, '1905-06-15 00:00:00.000', 22, 7, 1)

INSERT dbo.Reservation(id, date, exemplar\_id, reader\_id, status) VALUES (4, '2018-05-27 12:32:10.000', 19, 8, 1)

INSERT dbo.Reservation(id, date, exemplar\_id, reader\_id, status) VALUES (5, '2018-05-27 12:33:47.000', 3, 7, 3)

INSERT dbo.Reservation(id, date, exemplar\_id, reader\_id, status) VALUES (6, '2018-05-27 12:36:33.000', 16, 5, 4)

INSERT dbo.Reservation(id, date, exemplar\_id, reader\_id, status) VALUES (7, '2018-05-27 12:36:45.000', 12, 3, 4)

INSERT dbo.Reservation(id, date, exemplar\_id, reader\_id, status) VALUES (8, '2018-05-27 12:36:53.000', 9, 1, 4)

INSERT dbo.Reservation(id, date, exemplar\_id, reader\_id, status) VALUES (9, '2018-05-27 12:37:26.000', 17, 8, 4)

INSERT dbo.Reservation(id, date, exemplar\_id, reader\_id, status) VALUES (10, '2018-05-27 12:37:39.000', 16, 6, 2)

GO

SET IDENTITY\_INSERT dbo.Reservation OFF

GO

--

-- Вывод данных для таблицы ReservationStatus

--

SET IDENTITY\_INSERT dbo.ReservationStatus ON

GO

INSERT dbo.ReservationStatus(id, name) VALUES (1, N'Забронирован')

INSERT dbo.ReservationStatus(id, name) VALUES (2, N'Ожидание выдачи')

INSERT dbo.ReservationStatus(id, name) VALUES (3, N'Отменено')

INSERT dbo.ReservationStatus(id, name) VALUES (4, N'Выдано')

GO

SET IDENTITY\_INSERT dbo.ReservationStatus OFF

GO

--

-- Вывод данных для таблицы Users

--

SET IDENTITY\_INSERT dbo.Users ON

GO

INSERT dbo.Users(id, type, user\_id, login, password, passMD5) VALUES (3, N'librarian', 7, N'librarian', N'pass', 0x1A1DC91C907325C69271DDF0C944BC72)

INSERT dbo.Users(id, type, user\_id, login, password, passMD5) VALUES (4, N'admin', 1, N'admin', N'admin', 0x21232F297A57A5A743894A0E4A801FC3)

INSERT dbo.Users(id, type, user\_id, login, password, passMD5) VALUES (5, N'librarian', 8, N'dmitry.v', N'12345678', 0x25D55AD283AA400AF464C76D713C07AD)

GO

SET IDENTITY\_INSERT dbo.Users OFF

GO

--

-- Вывод данных для таблицы Worker

--

SET IDENTITY\_INSERT dbo.Worker ON

GO

INSERT dbo.Worker(id, FirstName, LastName, IsDelete, DateCreate) VALUES (1, N'Валишна', N'Наталья', CONVERT(bit, 'False'), '1905-06-29 00:00:00.000')

INSERT dbo.Worker(id, FirstName, LastName, IsDelete, DateCreate) VALUES (7, N'Иванов', N'Иванв', CONVERT(bit, 'False'), '1905-06-16 00:00:00.000')

INSERT dbo.Worker(id, FirstName, LastName, IsDelete, DateCreate) VALUES (8, N'Дмитрий', N'Волков', CONVERT(bit, 'False'), '1905-06-15 00:00:00.000')

GO

SET IDENTITY\_INSERT dbo.Worker OFF

GO

USE LibraryDB

GO

IF DB\_NAME() <> N'LibraryDB' SET NOEXEC ON

GO

--

-- Создать внешний ключ [FK\_Users\_Worker\_id] для объекта типа таблица [dbo].[Users]

--

PRINT (N'Создать внешний ключ [FK\_Users\_Worker\_id] для объекта типа таблица [dbo].[Users]')

GO

ALTER TABLE dbo.Users

ADD CONSTRAINT FK\_Users\_Worker\_id FOREIGN KEY (user\_id) REFERENCES dbo.Worker (id)

GO

--

-- Создать внешний ключ [FK\_Book\_category] для объекта типа таблица [dbo].[Book]

--

PRINT (N'Создать внешний ключ [FK\_Book\_category] для объекта типа таблица [dbo].[Book]')

GO

ALTER TABLE dbo.Book

ADD CONSTRAINT FK\_Book\_category FOREIGN KEY (category) REFERENCES dbo.BookCategory (id)

GO

--

-- Создать внешний ключ [FK\_Book\_Publishing] для объекта типа таблица [dbo].[Book]

--

PRINT (N'Создать внешний ключ [FK\_Book\_Publishing] для объекта типа таблица [dbo].[Book]')

GO

ALTER TABLE dbo.Book

ADD CONSTRAINT FK\_Book\_Publishing FOREIGN KEY (publishing\_id) REFERENCES dbo.Publishing (id)

GO

--

-- Создать внешний ключ [FK\_Order\_book\_id] для объекта типа таблица [dbo].[Orders]

--

PRINT (N'Создать внешний ключ [FK\_Order\_book\_id] для объекта типа таблица [dbo].[Orders]')

GO

ALTER TABLE dbo.Orders

ADD CONSTRAINT FK\_Order\_book\_id FOREIGN KEY (book\_id) REFERENCES dbo.Book (ISBN)

GO

--

-- Создать внешний ключ [FK\_Order\_publishing\_id] для объекта типа таблица [dbo].[Orders]

--

PRINT (N'Создать внешний ключ [FK\_Order\_publishing\_id] для объекта типа таблица [dbo].[Orders]')

GO

ALTER TABLE dbo.Orders

ADD CONSTRAINT FK\_Order\_publishing\_id FOREIGN KEY (publishing\_id) REFERENCES dbo.Publishing (id)

GO

--

-- Создать внешний ключ [FK\_Exemplar\_ISBN] для объекта типа таблица [dbo].[Exemplar]

--

PRINT (N'Создать внешний ключ [FK\_Exemplar\_ISBN] для объекта типа таблица [dbo].[Exemplar]')

GO

ALTER TABLE dbo.Exemplar

ADD CONSTRAINT FK\_Exemplar\_ISBN FOREIGN KEY (ISBN) REFERENCES dbo.Book (ISBN)

GO

--

-- Создать внешний ключ [FK\_Reservation\_exemplar\_id] для объекта типа таблица [dbo].[Reservation]

--

PRINT (N'Создать внешний ключ [FK\_Reservation\_exemplar\_id] для объекта типа таблица [dbo].[Reservation]')

GO

ALTER TABLE dbo.Reservation

ADD CONSTRAINT FK\_Reservation\_exemplar\_id FOREIGN KEY (exemplar\_id) REFERENCES dbo.Exemplar (id)

GO

--

-- Создать внешний ключ [FK\_Reservation\_reader\_id] для объекта типа таблица [dbo].[Reservation]

--

PRINT (N'Создать внешний ключ [FK\_Reservation\_reader\_id] для объекта типа таблица [dbo].[Reservation]')

GO

ALTER TABLE dbo.Reservation

ADD CONSTRAINT FK\_Reservation\_reader\_id FOREIGN KEY (reader\_id) REFERENCES dbo.Reader (id)

GO

--

-- Создать внешний ключ [FK\_Reservation\_status] для объекта типа таблица [dbo].[Reservation]

--

PRINT (N'Создать внешний ключ [FK\_Reservation\_status] для объекта типа таблица [dbo].[Reservation]')

GO

ALTER TABLE dbo.Reservation

ADD CONSTRAINT FK\_Reservation\_status FOREIGN KEY (status) REFERENCES dbo.ReservationStatus (id)

GO

--

-- Создать внешний ключ [FK\_BookIssuing\_Exemplar] для объекта типа таблица [dbo].[BookIssuing]

--

PRINT (N'Создать внешний ключ [FK\_BookIssuing\_Exemplar] для объекта типа таблица [dbo].[BookIssuing]')

GO

ALTER TABLE dbo.BookIssuing

ADD CONSTRAINT FK\_BookIssuing\_Exemplar FOREIGN KEY (Exemplar\_id) REFERENCES dbo.Exemplar (id)

GO

--

-- Создать внешний ключ [FK\_BookIssuing\_Reader] для объекта типа таблица [dbo].[BookIssuing]

--

PRINT (N'Создать внешний ключ [FK\_BookIssuing\_Reader] для объекта типа таблица [dbo].[BookIssuing]')

GO

ALTER TABLE dbo.BookIssuing

ADD CONSTRAINT FK\_BookIssuing\_Reader FOREIGN KEY (Reader\_id) REFERENCES dbo.Reader (id)

GO

--

-- Создать внешний ключ [FK\_BookIssuing\_Worker\_id] для объекта типа таблица [dbo].[BookIssuing]

--

PRINT (N'Создать внешний ключ [FK\_BookIssuing\_Worker\_id] для объекта типа таблица [dbo].[BookIssuing]')

GO

ALTER TABLE dbo.BookIssuing

ADD CONSTRAINT FK\_BookIssuing\_Worker\_id FOREIGN KEY (Worker\_id) REFERENCES dbo.Worker (id)

GO

--

-- Создать внешний ключ [FK\_AuthorBook\_Author] для объекта типа таблица [dbo].[AuthorBook]

--

PRINT (N'Создать внешний ключ [FK\_AuthorBook\_Author] для объекта типа таблица [dbo].[AuthorBook]')

GO

ALTER TABLE dbo.AuthorBook

ADD CONSTRAINT FK\_AuthorBook\_Author FOREIGN KEY (author\_id) REFERENCES dbo.Author (id)

GO

--

-- Создать внешний ключ [FK\_AuthorBook\_ISBN] для объекта типа таблица [dbo].[AuthorBook]

--

PRINT (N'Создать внешний ключ [FK\_AuthorBook\_ISBN] для объекта типа таблица [dbo].[AuthorBook]')

GO

ALTER TABLE dbo.AuthorBook

ADD CONSTRAINT FK\_AuthorBook\_ISBN FOREIGN KEY (ISBN) REFERENCES dbo.Book (ISBN)

GO

SET NOEXEC OFF

GO

# ПРИЛОЖЕНИЕ 2

Модуль "Дата модуль"

unit UDM;

interface

uses

System.SysUtils, System.Classes, Data.DB, Data.Win.ADODB;

type

TDM = class(TDataModule)

ADOConnection1: TADOConnection;

TUsers: TADOTable;

DSUsers: TDataSource;

TWorker: TADOTable;

DSWorker: TDataSource;

TReader: TADOTable;

DSReader: TDataSource;

TAuthor: TADOTable;

DSAuthor: TDataSource;

TAuthorBook: TADOTable;

DSAuthorBook: TDataSource;

TBook: TADOTable;

DSBook: TDataSource;

TBookCategory: TADOTable;

DSBookCategory: TDataSource;

TBookIssuing: TADOTable;

DSBookIssuing: TDataSource;

TExemplar: TADOTable;

DSExemplar: TDataSource;

TPublishing: TADOTable;

DSPublishing: TDataSource;

TReservation: TADOTable;

DSReservation: TDataSource;

TReservationStatus: TADOTable;

DSReservationStatus: TDataSource;

ViewCatalog: TADOTable;

DSCatalog: TDataSource;

TAuthorBookid: TAutoIncField;

TAuthorBookauthor\_id: TIntegerField;

TAuthorBookISBN: TStringField;

TAuthorBookauthor\_lf: TStringField;

TAuthorid: TAutoIncField;

TAuthorFirstName: TStringField;

TAuthorLastName: TStringField;

TAuthorMiddleName: TStringField;

TAuthorabr: TStringField;

TBookISBN: TStringField;

TBookTitle: TStringField;

TBookpublishing\_id: TIntegerField;

TBookyear: TWideStringField;

TBookcategory2: TIntegerField;

TBookpublication\_lf: TStringField;

TWorkerid: TAutoIncField;

TWorkerFirstName: TStringField;

TWorkerLastName: TStringField;

TReaderid: TAutoIncField;

TReaderFirstName: TStringField;

TReaderLastName: TStringField;

TReaderMiddleName: TStringField;

TReaderPhone: TStringField;

TReaderAdress: TStringField;

TBookCategoryid: TAutoIncField;

TBookCategoryname: TStringField;

TPublishingid: TAutoIncField;

TPublishingname: TStringField;

TPublishingcity: TStringField;

TBookCategoryName2: TStringField;

TExemplarid: TAutoIncField;

TExemplarplacement: TStringField;

TExemplarISBN: TStringField;

TAuthorBookAuthorName: TStringField;

TReservationdate: TDateTimeField;

TReservationreader\_id: TIntegerField;

TReservationstatus2: TIntegerField;

TBookIssuingid: TAutoIncField;

TBookIssuingExemplar\_id: TIntegerField;

TBookIssuingReader\_id: TIntegerField;

TBookIssuingDateIssue: TWideStringField;

TBookIssuingDateReturnExpected: TWideStringField;

TBookIssuingDateReturnReal: TWideStringField;

TBookIssuingWorker\_id: TIntegerField;

ViewCatalogISBN: TStringField;

ViewCatalogcategory: TStringField;

ViewCatalogTitle: TStringField;

ViewCatalogAuthor: TStringField;

ViewCatalogname: TStringField;

ViewCatalogcity: TStringField;

ViewCatalogyear: TWideStringField;

ViewCatalogCount: TIntegerField;

TUsersid: TAutoIncField;

TUserstype: TStringField;

TUsersuser\_id: TIntegerField;

TUserslogin: TStringField;

TUserspassword: TStringField;

TUserspassMD5: TBytesField;

TExemplarbook\_LF: TStringField;

TReservationid: TAutoIncField;

TReservationbook\_LF: TStringField;

TBookIssuingWorkerName: TStringField;

TBookIssuingBookName: TStringField;

TReservationBookName: TStringField;

TReservationReaderName: TStringField;

TReservationStatusNAme: TStringField;

ADOQueryUpdatePassMD5: TADOQuery;

ViewAvailableBooks: TADOTable;

DSAvailableBooks: TDataSource;

ViewAvailableBooksid: TIntegerField;

ViewAvailableBooksISBN: TStringField;

ViewAvailableBooksTitle: TStringField;

ViewAvailableBookspublishing\_id: TIntegerField;

ViewAvailableBookspublishing: TStringField;

ViewAvailableBooksyear: TWideStringField;

ViewAvailableBookscategory\_id: TIntegerField;

ViewAvailableBookscategory: TStringField;

ViewAvailableBooksplacement: TStringField;

TReaderabr: TStringField;

TBookIssuingReaderName: TStringField;

TBookIssuingReader: TADOTable;

AutoIncField1: TAutoIncField;

IntegerField1: TIntegerField;

IntegerField2: TIntegerField;

WideStringField1: TWideStringField;

WideStringField2: TWideStringField;

WideStringField3: TWideStringField;

IntegerField3: TIntegerField;

StringField1: TStringField;

StringField2: TStringField;

StringField3: TStringField;

DsBookIssuingReader: TDataSource;

TOrders: TADOTable;

DSOrder: TDataSource;

TOrdersid: TAutoIncField;

TOrderspublishing\_id: TIntegerField;

TOrdersbook\_id: TStringField;

TOrderscount: TIntegerField;

TOrdersprocessed: TBooleanField;

TOrderspublishilg\_LF: TStringField;

TOrdersbook\_LF: TStringField;

TWorkerIsDelete: TBooleanField;

TWorkerDateCreate: TDateTimeField;

TExemplarDateCreate: TDateTimeField;

TExemplarIsDelete: TBooleanField;

TRepReaderActivity: TADOQuery;

DSRepReaderActivity: TDataSource;

TRepReaderActivityTitle: TStringField;

TRepReaderActivityDateIssue: TWideStringField;

TRepReaderActivityDateReturnReal: TWideStringField;

ADOQueryReportOverdueBooks: TADOQuery;

DSOverdueBooks: TDataSource;

ADOQueryReportOverdueBooksid: TAutoIncField;

ADOQueryReportOverdueBooksTitle: TStringField;

ADOQueryReportOverdueBooksDateIssue: TWideStringField;

ADOQueryReportOverdueBooksDateReturnExpected: TWideStringField;

ADOQueryReportOverdueBooksReader: TStringField;

ADOQueryReportOverdueBooksPhone: TStringField;

ViewCatalogAvailableCount: TIntegerField;

TReservationexemplar\_id: TIntegerField;

procedure TAuthorCalcFields(DataSet: TDataSet);

procedure TReaderCalcFields(DataSet: TDataSet);

procedure DoIncrementalFilter(DataSet: TDataSet;

const FieldName, SearchTerm: string);

private

{ Private declarations }

public

{ Public declarations }

end;

var

DM: TDM;

implementation

{%CLASSGROUP 'Vcl.Controls.TControl'}

{$R \*.dfm}

procedure TDM.TAuthorCalcFields(DataSet: TDataSet);

begin

if (DataSet.FieldByName('LastName').AsString.IsEmpty or

DataSet.FieldByName('FirstName').AsString.IsEmpty or

DataSet.FieldByName('MiddleName').AsString.IsEmpty) then

exit;

DataSet.FieldByName('abr').value := DataSet.FieldByName('FirstName').AsString

+ ' ' + DataSet.FieldByName('LastName').AsString[1] + '.' +

DataSet.FieldByName('MiddleName').AsString[1];

end;

procedure TDM.TReaderCalcFields(DataSet: TDataSet);

begin

if (DataSet.FieldByName('LastName').AsString.IsEmpty or

DataSet.FieldByName('FirstName').AsString.IsEmpty or

DataSet.FieldByName('MiddleName').AsString.IsEmpty) then

exit;

DataSet.FieldByName('abr').value := DataSet.FieldByName('FirstName').AsString

+ ' ' + DataSet.FieldByName('LastName').AsString[1] + '.' +

DataSet.FieldByName('MiddleName').AsString[1];

end;

procedure TDM.DoIncrementalFilter(DataSet: TDataSet;

const FieldName, SearchTerm: string);

begin

Assert(Assigned(DataSet), 'No dataset is assigned');

if SearchTerm = '' then

DataSet.Filtered := False

else

begin

DataSet.Filter := FieldName + ' LIKE ' + QuotedStr(SearchTerm + '\*');

DataSet.Filtered := True;

end;

end;

end.

Модуль "Авторизация"

unit UnitAuth;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, Buttons, Data.DB, Data.Win.ADODB;

type

TFormAuth = class(TForm)

GroupBox1: TGroupBox;

EditLogin: TEdit;

EditPass: TEdit;

Label1: TLabel;

Label2: TLabel;

BitBtn1: TBitBtn;

ADOQuery1: TADOQuery;

procedure BitBtn1Click(Sender: TObject);

procedure FormKeyPress(Sender: TObject; var Key: Char);

private

{ Private declarations }

public

{ Public declarations }

end;

var

FormAuth: TFormAuth;

implementation

{$R \*.dfm}

uses UDM, UnitLibrarianAccount, UnitSuperUser;

procedure TFormAuth.BitBtn1Click(Sender: TObject);

begin

ADOQuery1.Close;

// передача в параметры sql запроса данных введенных пользователем

ADOQuery1.Parameters.ParamByName('l').Value := EditLogin.Text;

ADOQuery1.Parameters.ParamByName('p').Value := EditPass.Text;

// выполнение запроса

ADOQuery1.Open;

if (ADOQuery1.RecordCount > 0) then

begin

if ADOQuery1.Fields[1].AsString = 'librarian' then

begin

dm.TWorker.Locate('id', ADOQuery1.Fields[0].AsInteger,

[loCaseInsensitive, loPartialKey]);

FormLibrary.Show;

end;

if ADOQuery1.Fields[1].AsString = 'admin' then

begin

dm.TWorker.Locate('id', ADOQuery1.Fields[0].AsInteger,

[loCaseInsensitive, loPartialKey]);

FormSuper.Show;

end;

hide;

end

else

ShowMessage('Неверный логин или пароль');

end;

procedure TFormAuth.FormKeyPress(Sender: TObject; var Key: Char);

begin

if key = #13 then

BitBtn1Click(Sender);

end;

end.

Модуль "Библиотекарь"

unit UnitLibrarianAccount;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, Menus, Grids, DBGrids, Mask, DBCtrls, Buttons,

ExtCtrls, ComCtrls, Data.DB, Vcl.WinXCtrls, Data.Win.ADODB;

private

{ Private declarations }

public

{ Public declarations }

end;

var

FormLibrary: TFormLibrary;

implementation

{$R \*.dfm}

uses UDM, UnitBook, UnitReader;

procedure TFormLibrary.ButtonBookIssueSaveClick(Sender: TObject);

begin

if DM.TBookIssuing.State in [dsInsert] then

begin

DM.TBookIssuing.FieldByName('DateIssue').Value := DateTimePicker1.DateTime;

DM.TBookIssuing.FieldByName('DateReturnExpected').Value :=

DateTimePicker2.DateTime;

DM.TBookIssuing.FieldByName('Worker\_id').Value := DM.TWorker.Fields

[0].Value;

DM.TBookIssuing.Post;

GroupBoxBookIssue.Enabled := false;

end;

GroupBoxBookIssue.Visible := false;

ButtonReturnBook.Enabled := true;

DBGridBookIssuing.Enabled := true;

end;

procedure TFormLibrary.Button1Click(Sender: TObject);

begin

DM.TReader.Insert;

case FormAddReadTicket.ShowModal of

mrOk:

begin

DM.TReader.Post;

// DM.TUsers.Locate('user\_id;type', VarArrayOf([DM.TReader.Fields[0].Value, 'reader']), [loPartialKey]);

// DM.TUsers.Insert;

// DM.TUsers.FieldByName('type').Value := 'reader';

// DM.TUsers.FieldByName('user\_id').Value := DM.TReader.Fields[0].Value;

// DM.TUsers.FieldByName('login').Value :=

// FormAddReadTicket.EditLogin.Text;

// DM.TUsers.FieldByName('password').Value :=

// FormAddReadTicket.EditPass.Text;

// DM.TUsers.Post;

// // обновим пароль MD5

// DM.ADOQueryUpdatePassMD5.ExecSQL;

end;

mrCancel:

begin

DM.TReader.Cancel;

// DM.TUsers.Cancel;

end;

end;

end;

procedure TFormLibrary.Button2Click(Sender: TObject);

begin

DM.TReader.Edit;

// DM.TUsers.Locate('user\_id;type', VarArrayOf([DM.TReader.Fields[0].Value,

// 'reader']), [loPartialKey]);

//

// DM.TUsers.Edit;

// FormAddReadTicket.EditLogin.Text := DM.TUsers.FieldByName('login').Value;

// FormAddReadTicket.EditPass.Text := DM.TUsers.FieldByName('password').Value;

case FormAddReadTicket.ShowModal of

mrOk:

begin

DM.TReader.Post;

// DM.TUsers.FieldByName('type').Value := 'reader';

// DM.TUsers.FieldByName('user\_id').Value := DM.TReader.Fields[0].Value;

// DM.TUsers.FieldByName('login').Value :=

// FormAddReadTicket.EditLogin.Text;

// DM.TUsers.FieldByName('password').Value :=

// FormAddReadTicket.EditPass.Text;

// DM.TUsers.Post;

end;

mrCancel:

begin

DM.TReader.Cancel;

DM.TUsers.Cancel;

end;

end;

end;

procedure TFormLibrary.Button3Click(Sender: TObject);

begin

DM.TBookIssuing.Cancel;

GroupBoxBookIssue.Visible := false;

ButtonReturnBook.Enabled := true;

DBGridBookIssuing.Enabled := true;

end;

procedure TFormLibrary.Button4Click(Sender: TObject);

begin

DM.TReservation.Insert;

end;

procedure TFormLibrary.Button5Click(Sender: TObject);

begin

ButtonIssueBookClick(Sender);

DM.DSBookIssuing.DataSet.FieldByName('Exemplar\_id').Value :=

DM.TReservation.FieldByName('exemplar\_id').Value;

DM.DSBookIssuing.DataSet.FieldByName('Reader\_id').Value :=

DM.TReservation.FieldByName('reader\_id').Value;

DM.TReservation.FieldByName('exemplar\_id').Value;

// DM.DSBookIssuing.DataSet.FieldByName('').Value := '';

// DM.DSBookIssuing.DataSet.FieldByName('').Value := '';

PageControl1.ActivePageIndex := 3;

end;

procedure TFormLibrary.Button6Click(Sender: TObject);

begin

DM.TReservation.Edit;

DM.TReservation.FieldByName('status').Value := 3;

DM.TReservation.Post;

end;

procedure TFormLibrary.ButtonSearchBoxReaderClearClick(Sender: TObject);

begin

DM.TReader.Filtered := false

end;

procedure TFormLibrary.ButtonSearchBoxAvailableBooksClearClick(Sender: TObject);

begin

DM.ViewAvailableBooks.Filtered := false;

end;

procedure TFormLibrary.ButtonSearchBoxCatalogClearClick(Sender: TObject);

begin

DM.ViewCatalog.Filtered := false;

end;

procedure TFormLibrary.Button7Click(Sender: TObject);

begin

// if DM.TUsers.Locate('user\_id', DM.TReader.Fields[0].AsInteger,

// [loCaseInsensitive, loPartialKey]) then

// begin

// DM.TUsers.Delete;

DM.TReader.Delete;

// end;

end;

procedure TFormLibrary.Button8Click(Sender: TObject);

begin

DM.TReservation.FieldByName('exemplar\_id').Value :=

DSTBHExemplar.DataSet.Fields[0].Value;

DM.TReservation.FieldByName('date').Value := Now;

DM.TReservation.Post;

end;

procedure TFormLibrary.Button9Click(Sender: TObject);

begin

DM.TReservation.Cancel;

end;

procedure TFormLibrary.ButtonIssueBookClick(Sender: TObject);

begin

DM.TBookIssuing.Insert;

GroupBoxBookIssue.Enabled := true;

GroupBoxBookIssue.Visible := true;

ButtonReturnBook.Enabled := false;

DBGridBookIssuing.Enabled := false;

end;

procedure TFormLibrary.ButtonReturnBookClick(Sender: TObject);

begin

DM.TBookIssuing.Edit;

DM.TBookIssuing.FieldByName('DateReturnReal').Value := Now();

DM.TBookIssuing.Post;

end;

procedure TFormLibrary.DBGridBookIssuingKeyPress(Sender: TObject;

var Key: Char);

begin

// if DM.TBookIssuing.State in [dsInsert] then

// GroupBoxBookIssue.Enabled := true

// else

// GroupBoxBookIssue.Enabled := false;

end;

procedure TFormLibrary.FormClose(Sender: TObject; var Action: TCloseAction);

begin

Application.Terminate;

end;

procedure TFormLibrary.SearchBoxReaderInvokeSearch(Sender: TObject);

var

SearchTerm: string;

begin

// здесь будет поиск

// if not DM.TReader.Locate('LastName', SearchBoxReader.Text, [loPartialKey])

// then

// if not DM.TReader.Locate('FirstName', SearchBoxReader.Text, [loPartialKey])

// then

// if not DM.TReader.Locate('MiddleName', SearchBoxReader.Text,

// [loPartialKey]) then

// DM.TReader.Locate('Phone', SearchBoxReader.Text, [loPartialKey]);

SearchTerm := SearchBoxReader.Text;

if SearchTerm = '' then

DM.TReader.Filtered := false

else

begin

DM.TReader.Filter := 'LastName' + ' LIKE ' + QuotedStr(SearchTerm + '\*') +

' OR ' + 'FirstName' + ' LIKE ' + QuotedStr(SearchTerm + '\*') + ' OR ' +

'MiddleName' + ' LIKE ' + QuotedStr(SearchTerm + '\*');

DM.TReader.Filtered := true;

end;

end;

procedure TFormLibrary.SearchBoxAvailableBooksInvokeSearch(Sender: TObject);

var

FieldName, SearchTerm: string;

begin

case ComboBoxAvailableBooks.ItemIndex of

0:

FieldName := 'Title';

1:

FieldName := 'category';

2:

FieldName := 'publishing';

end;

SearchTerm := SearchBoxAvailableBooks.Text;

DM.DoIncrementalFilter(DM.ViewAvailableBooks, FieldName, SearchTerm);

end;

procedure TFormLibrary.SearchBoxCatalogInvokeSearch(Sender: TObject);

var

FieldName, SearchTerm: string;

begin

case ComboBoxBookSearch.ItemIndex of

0:

FieldName := 'Title';

1:

FieldName := 'category';

2:

FieldName := 'Author';

3:

FieldName := 'name';

end;

SearchTerm := SearchBoxCatalog.Text;

DM.DoIncrementalFilter(DM.ViewCatalog, FieldName, SearchTerm);

end;

end.

Модуль "Управляющий библиотекой"

unit UnitSuperUser;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Data.DB, Vcl.Grids, Vcl.DBGrids,

Vcl.Mask, Vcl.DBCtrls, Vcl.StdCtrls, Vcl.Buttons, Vcl.ExtCtrls, Vcl.ComCtrls,

UDM, Vcl.Menus, Vcl.WinXCtrls, Data.Win.ADODB;

type

private

{ Private declarations }

public

{ Public declarations }

end;

var

FormSuper: TFormSuper;

implementation

{$R \*.dfm}

uses UnitReportReaderActivity, UnitReportOverdueBooks;

procedure TFormSuper.ButtonReaderActivityClick(Sender: TObject);

begin

PageControlReports.ActivePageIndex := 0;

end;

procedure TFormSuper.ButtonRetireClick(Sender: TObject);

begin

dm.TWorker.Edit;

dm.TWorker.FieldByName('IsDelete').Value := True;

dm.TWorker.Post;

end;

procedure TFormSuper.ButtonNewLibrarianClick(Sender: TObject);

begin

dm.TWorker.Insert;

DBGridWorkers.Enabled := false;

GroupBoxWorker.Enabled := True;

EditLogin.Text := '';

EditPass.Text := '';

end;

procedure TFormSuper.ButtonWorkerEditCancelClick(Sender: TObject);

begin

dm.TWorker.Cancel;

DBGridWorkers.Enabled := True;

GroupBoxWorker.Enabled := false;

end;

procedure TFormSuper.ButtonWriteOffExemplarClick(Sender: TObject);

var

id: integer; // код экземпляра на пометку - удаление

begin

id := dm.DSAvailableBooks.DataSet.Fields[0].Value;

if dm.TExemplar.Locate('id', id, [loCaseInsensitive, loPartialKey]) then

begin

dm.TExemplar.Edit;

dm.TExemplar.FieldByName('IsDelete').Value := True;

dm.TExemplar.Post;

end;

end;

procedure TFormSuper.Button1Click(Sender: TObject);

begin

Form1.QuickRep1.Preview;

end;

procedure TFormSuper.Button2Click(Sender: TObject);

begin

Form1.QuickRep1.Print;

end;

procedure TFormSuper.Button3Click(Sender: TObject);

var

sNewPass: string;

begin

// выводим окно до тех пор пока не будет введн пароль

while sNewPass.IsEmpty do

sNewPass := InputBox('Смена пароля', 'Ввелите новый пароль: ', '');

dm.TUsers.Edit;

dm.TUsers.FieldByName('password').Value := sNewPass;

dm.TUsers.Post;

ShowMessage('Пароль успешно сменен');

// обновим пароль MD5

dm.ADOQueryUpdatePassMD5.ExecSQL;

end;

procedure TFormSuper.Button4Click(Sender: TObject);

begin

if dm.TWorker.State = dsEdit then

dm.TUsers.Edit

else if dm.TWorker.State = dsInsert then

begin

dm.TWorker.Post;

dm.TUsers.Insert;

end

else

exit;

dm.TUsers.FieldByName('login').Value := EditLogin.Text;

dm.TUsers.FieldByName('password').Value := EditPass.Text;

dm.TUsers.FieldByName('user\_id').Value := dm.TWorker.Fields[0].Value;

dm.TUsers.Post;

if dm.TWorker.State = dsEdit then

dm.TWorker.Post;

DBGridWorkers.Enabled := True;

GroupBoxWorker.Enabled := false;

// обновим пароль MD5

dm.ADOQueryUpdatePassMD5.ExecSQL;

end;

procedure TFormSuper.Button5Click(Sender: TObject);

begin

with dm do

begin

TRepReaderActivity.Close;

TRepReaderActivity.Parameters.ParamByName('id').Value :=

TReader.Fields[0].Value;

TRepReaderActivity.Parameters.ParamByName('start').Value :=

DateToStr(DateTimePickerStart.Date);

TRepReaderActivity.Parameters.ParamByName('end').Value :=

DateToStr(DateTimePickerEnd.Date);

TRepReaderActivity.Open;

end;

end;

procedure TFormSuper.Button6Click(Sender: TObject);

begin

dm.ADOQueryReportOverdueBooks.Open;

end;

procedure TFormSuper.Button7Click(Sender: TObject);

begin

FormReportOverdueBooks.QuickRep1.Preview;

end;

procedure TFormSuper.Button8Click(Sender: TObject);

begin

FormReportOverdueBooks.QuickRep1.Print;

end;

procedure TFormSuper.ButtonSaveOrderClick(Sender: TObject);

begin

orders.FieldByName('processed').Value := false;

orders.Post;

DBGridOrders.Enabled := True;

GroupBoxCreateOrder.Enabled := false;

end;

procedure TFormSuper.ButtonSearchBoxAvailableBooksClearClick(Sender: TObject);

begin

dm.ViewAvailableBooks.Filtered := false;

end;

procedure TFormSuper.ButtonSearchBoxCatalogClearClick(Sender: TObject);

begin

dm.ViewCatalog.Filtered := false;

end;

procedure TFormSuper.ButtonBookAddAuthorClick(Sender: TObject);

begin

dm.TAuthorBook.Insert;

dm.TAuthorBook.Post;

end;

procedure TFormSuper.ButtonBooksIncomingClick(Sender: TObject);

var

count: integer;

I: integer;

begin

count := DSOrders.DataSet.FieldByName('count').Value;

for I := 1 to count do

begin

dm.TExemplar.Insert;

dm.TExemplar.FieldByName('ISBN').Value :=

DSOrders.DataSet.FieldByName('book\_id').Value;

dm.TExemplar.FieldByName('placement').Value := 'полка 1';

dm.TExemplar.Post;

end;

DSOrders.DataSet.Edit;

DSOrders.DataSet.FieldByName('processed').Value := True;

DSOrders.DataSet.Post;

ShowMessage('Успешно создан(ы) ' + count.ToString + ' экземпляров книг ');

dm.ViewAvailableBooks.Close;

dm.ViewAvailableBooks.OPEN;

if MessageDlg('Перейти в каталог?', mtConfirmation, [mbYes, mbNO], 0) = mrYes

then

begin

PageControl3.ActivePageIndex := 1;

PageControl4.ActivePageIndex := 1;

end;

end;

procedure TFormSuper.ButtonCanselAuthorClick(Sender: TObject);

begin

dm.TAuthorBook.Cancel;

dm.TBook.Cancel;

end;

procedure TFormSuper.ButtonCreateorderClick(Sender: TObject);

begin

orders.Insert;

DBGridOrders.Enabled := false;

GroupBoxCreateOrder.Enabled := True;

end;

procedure TFormSuper.ButtonCreateUserClick(Sender: TObject);

begin

dm.TReader.Insert;

dm.TUsers.Insert;

end;

procedure TFormSuper.ButtonDeleteUserClick(Sender: TObject);

begin

dm.TReader.Delete;

dm.TUsers.Delete;

end;

procedure TFormSuper.ButtonEditAuthorClick(Sender: TObject);

begin

dm.TAuthor.Post;

DBEditAuthorFirstName.ReadOnly := True;

DBEditAuthorLastName.ReadOnly := True;

DBEditAuthorMiddleName.ReadOnly := True;

ButtonEditAuthor.Enabled := false;

end;

procedure TFormSuper.ButtonEditBookCategoryClick(Sender: TObject);

begin

dm.TBookCategory.Post;

ButtonEditBookCategory.Enabled := false;

DBEditBookCategoryName.ReadOnly := True;

end;

procedure TFormSuper.ButtonEditPublishingClick(Sender: TObject);

begin

dm.TPublishing.Post;

DBEditPublishingName.ReadOnly := True;

DBEditPublishingCity.ReadOnly := True;

ButtonEditPublishing.Enabled := false;

end;

procedure TFormSuper.ButtonEditUserClick(Sender: TObject);

begin

dm.TReader.Edit;

dm.TUsers.Edit;

end;

procedure TFormSuper.ButtonOKBookAuthorClick(Sender: TObject);

begin

dm.TAuthorBook.Post;

dm.TBook.Post;

end;

procedure TFormSuper.ButtonOKExemplarClick(Sender: TObject);

begin

dm.TExemplar.Post;

DBGrid3.ReadOnly := True;

DBEditPlace.ReadOnly := True;

DBEditISBM.ReadOnly := True;

ButtonOKExemplar.Enabled := false;

end;

procedure TFormSuper.ButtonOKUserClick(Sender: TObject);

begin

dm.TReader.Post;

dm.TUsers.Post;

DBEditBookName.ReadOnly := True;

DBLookupComboBoxBookCategory.ReadOnly := True;

DBLookupComboBoxBookPublishing.ReadOnly := True;

DBEditBookYear.ReadOnly := True;

DBLookupComboBoxAuthor.ReadOnly := True;

DBGrid2.ReadOnly := True;

ButtonBookAddAuthor.Enabled := false;

ButtonCanselAuthor.Enabled := false;

ButtonOKBookAuthor.Enabled := True;

end;

procedure TFormSuper.ButtonOverdueBooksClick(Sender: TObject);

begin

PageControlReports.ActivePageIndex := 1;

end;

procedure TFormSuper.DBGridOrdersCellClick(Column: TColumn);

begin

if orders.FieldByName('processed').Value = True then

ButtonBooksIncoming.Enabled := false

else

ButtonBooksIncoming.Enabled := True;

end;

procedure TFormSuper.DBGridWorkersCellClick(Column: TColumn);

begin

EditLogin.Text := dm.TUsers.FieldByName('login').Value;

EditPass.Text := dm.TUsers.FieldByName('password').Value;

end;

procedure TFormSuper.FormClose(Sender: TObject; var Action: TCloseAction);

begin

Application.Terminate;

end;

procedure TFormSuper.N1Click(Sender: TObject);

begin

dm.TAuthorBook.Delete;

end;

procedure TFormSuper.SearchBox1InvokeSearch(Sender: TObject);

begin

if not dm.TReader.Locate('LastName', SearchBox1.Text, [loPartialKey]) then

if not dm.TReader.Locate('FirstName', SearchBox1.Text, [loPartialKey]) then

if not dm.TReader.Locate('MiddleName', SearchBox1.Text, [loPartialKey])

then

if not dm.TReader.Locate('Addres', SearchBox1.Text, [loPartialKey]) then

dm.TReader.Locate('Phone', SearchBox1.Text, [loPartialKey]);

end;

procedure TFormSuper.SearchBoxAvailableBooksInvokeSearch(Sender: TObject);

var

FieldName, SearchTerm: string;

begin

case ComboBoxAvailableBooks.ItemIndex of

0:

FieldName := 'Title';

1:

FieldName := 'category';

2:

FieldName := 'publishing';

end;

SearchTerm := SearchBoxAvailableBooks.Text;

dm.DoIncrementalFilter(dm.ViewAvailableBooks, FieldName, SearchTerm);

end;

procedure TFormSuper.SearchBoxCatalogInvokeSearch(Sender: TObject);

var

FieldName, SearchTerm: string;

begin

case ComboBoxBookSearch.ItemIndex of

0:

FieldName := 'Title';

1:

FieldName := 'category';

2:

FieldName := 'Author';

3:

FieldName := 'name';

end;

SearchTerm := SearchBoxCatalog.Text;

dm.DoIncrementalFilter(dm.ViewCatalog, FieldName, SearchTerm);

end;

procedure TFormSuper.SearchBoxWorkerInvokeSearch(Sender: TObject);

var

SearchTerm: string;

begin

SearchTerm := SearchBoxWorker.Text;

if SearchTerm = '' then

dm.TWorker.Filtered := false

else

begin

dm.TWorker.Filter := 'LastName' + ' LIKE ' + QuotedStr(SearchTerm + '\*') +

' OR ' + 'FirstName' + ' LIKE ' + QuotedStr(SearchTerm + '\*');

dm.TWorker.Filtered := True;

end;

end;

procedure TFormSuper.SearchBoxWorkerKeyPress(Sender: TObject; var Key: Char);

begin

if Key = #27 then

SpeedButtonSearchWorkerCancelClick(Sender);

end;

procedure TFormSuper.SpeedButtonSearchWorkerCancelClick(Sender: TObject);

begin

dm.TWorker.Filtered := false;

SearchBoxWorker.Text := '';

end;

procedure TFormSuper.SpeedButtonAddAuthorClick(Sender: TObject);

begin

dm.TAuthor.Insert;

DBEditAuthorFirstName.ReadOnly := false;

DBEditAuthorLastName.ReadOnly := false;

DBEditAuthorMiddleName.ReadOnly := false;

ButtonEditAuthor.Enabled := True;

end;

procedure TFormSuper.SpeedButtonAddBookClick(Sender: TObject);

begin

dm.TBook.Insert;

dm.TAuthorBook.Insert;

dm.TAuthorBook.Edit;

dm.TBook.Edit;

DBEditBookName.ReadOnly := false;

DBLookupComboBoxBookCategory.ReadOnly := false;

DBLookupComboBoxBookPublishing.ReadOnly := false;

DBEditBookYear.ReadOnly := false;

DBLookupComboBoxAuthor.ReadOnly := false;

DBGrid2.ReadOnly := false;

ButtonBookAddAuthor.Enabled := True;

ButtonCanselAuthor.Enabled := True;

ButtonOKBookAuthor.Enabled := True;

end;

procedure TFormSuper.SpeedButtonAddCategoryClick(Sender: TObject);

begin

dm.TBookCategory.Insert;

ButtonEditBookCategory.Enabled := True;

DBEditBookCategoryName.ReadOnly := false;

end;

procedure TFormSuper.SpeedButtonAddExemplarClick(Sender: TObject);

begin

dm.TExemplar.Insert;

DBEditPlace.ReadOnly := false;

DBEditISBM.ReadOnly := false;

ButtonOKExemplar.Enabled := True;

end;

procedure TFormSuper.SpeedButtonAddPublishigClick(Sender: TObject);

begin

dm.TPublishing.Insert;

DBEditPublishingName.ReadOnly := false;

DBEditPublishingCity.ReadOnly := false;

ButtonEditPublishing.Enabled := True;

end;

procedure TFormSuper.SpeedButtonDeleteBookClick(Sender: TObject);

begin

dm.TBook.Delete;

end;

procedure TFormSuper.SpeedButtonDeleteExemplarClick(Sender: TObject);

begin

dm.TExemplar.Delete;

end;

procedure TFormSuper.SpeedButtonEditAuthorClick(Sender: TObject);

begin

dm.TAuthor.Edit;

DBEditAuthorFirstName.ReadOnly := false;

DBEditAuthorLastName.ReadOnly := false;

DBEditAuthorMiddleName.ReadOnly := false;

ButtonEditAuthor.Enabled := True;

end;

procedure TFormSuper.SpeedButtonEditBookClick(Sender: TObject);

begin

dm.TAuthorBook.Edit;

dm.TBook.Edit;

DBEditBookName.ReadOnly := false;

DBLookupComboBoxBookCategory.ReadOnly := false;

DBLookupComboBoxBookPublishing.ReadOnly := false;

DBEditBookYear.ReadOnly := false;

DBLookupComboBoxAuthor.ReadOnly := false;

DBGrid2.ReadOnly := false;

ButtonBookAddAuthor.Enabled := True;

ButtonCanselAuthor.Enabled := True;

ButtonOKBookAuthor.Enabled := True;

end;

procedure TFormSuper.SpeedButtonEditExemplarClick(Sender: TObject);

begin

dm.TExemplar.Edit;

DBGrid3.ReadOnly := false;

DBEditPlace.ReadOnly := false;

DBEditISBM.ReadOnly := false;

ButtonOKExemplar.Enabled := True;

end;

procedure TFormSuper.SpeedButtonEditPublishigClick(Sender: TObject);

begin

dm.TPublishing.Edit;

DBEditPublishingName.ReadOnly := false;

DBEditPublishingCity.ReadOnly := false;

ButtonEditPublishing.Enabled := True;

end;

end.

ПРИЛОЖЕНИЕ 3

Требования СанПин

1.Общие требования безопасности

1.1.К самостоятельной работе с ВДТ и ПЭВМ допускаются лица в возрасте не моложе 18 лет, прошедшие специальную подготовку, в том числе на III группу электробезопасности, обязательный медицинский осмотр и инструктаж по охране труда, не имеющие противопоказаний по состоянию здоровья. Женщины со времени установления беремен­ности и в период кормления ребенка грудью к выполнению всех видов работ, связанных с использованием ВДТ и ПЭВМ, не допускаются.

1.2. Пользователи ВДТ и ПЭВМ должны соблюдать правила внутрен­него трудового распорядка, установленные режимы труда и отдыха.

1.3. При работе с ВДТ и ПЭВМ возможно воздействие на работаю­щих следующих опасных и вредных производственных факторов:

- ионизирующие и неионизирующие излучения видеотерминалов;

- поражение электрическим током при работе на оборудовании без защитного заземления, а также со снятой задней крышкой видео­терминала;

- зрительное утомление, а также неблагоприятное воздействие на зрение мерцаний символов и фона при неустойчивой работе видеотер­минала, нечетком изображении на экране.

1.4. При работе с ВДТ и ПЭВМ необходимо использовать защит­ные экраны.

1.5. Помещение с ВДТ и ПЭВМ должно быть оснащено медаптечка первой помощи, системой кондиционирования воздуха или вытяж­ной вентиляцией.

1.6. Пользователи ВДТ и ПЭВМ обязаны соблюдать правила пожарной безопасности, знать места расположения первичных средств пожаротушения. Помещение с ВДТ и ПЭВМ должно быть оснащено двумя углекислотными огнетушителями и автоматической системой пожарной сигнализации.

1.7. О каждом несчастном случае с работником пострадавший очевидец несчастного случая обязан немедленно сообщить администрации учреждения. При неисправности оборудования прекратить работу и сообщить администрации учреждения.

1.8. В процессе работы пользователи ВДТ и ПЭВМ должны соблюдать правила использования средств индивидуальной и коллективной защиты, соблюдать правила личной гигиены, содержать в чистоте рабочее место.

1.9. Лица, допустившие невыполнение или нарушение инструкции: охране труда, привлекаются к дисциплинарной ответственности в соответствии с правилами внутреннего трудового распорядка и, при необходимости, подвергаются внеочередной проверке знаний норм и вил охраны труда.

2. Требования безопасности перед началом работы

2.1. Тщательно проветрить помещение с ВДТ и ПЭВМ, убедиться, что микроклимат в помещении находится в допустимых пределах: температура воздуха в холодный период года - 22-24°С, в теплый период года - 23-25°С, относительная влажность воздуха – 40 - 60 %.

2.2. Убедиться в наличии защитного заземления оборудования, также защитных экранов видеомониторов.

2.3. Включить видеомониторы и проверить стабильность и четко изображения на экранах.

3. Требования безопасности во время работы

3.1. При работе с ВДТ и ПЭВМ значения визуальных параметров должны находиться в пределах оптимального диапазона.

3.2. Клавиатуру располагать на поверхности стола на расстоянии 300 мм от края, обращенного к пользователю.

3.3. Тетрадь для записей располагается на подставке с наклоном 12-15° на расстоянии 55-65 см от глаз, которая должна быть хорошо освещена.

3.4. При работающем видеотерминале расстояние от глаз до экрана должно быть 0,6 - 0,7 м, уровень глаз должен приходиться на экран или на 2/3 его высоты.

3.5. Изображение на экранах видеомониторов должно быть ясным и предельно четким, не иметь мерцаний символов на экранах не должно быть бликов и отражений светильников, окон и окружающих предметов.

3.6. Не работать с ВДТ и ПЭВМ без защитных экранов.

3.7. Суммарное время непосредственной работы с ВДТ и ПЭВМ в течение рабочего дня должно быть не более 6 часов, для преподавате­лей высших и средних специальных учебных заведений, учителей об­щеобразовательных школ - не более 4 часов в день.

3.8. Продолжительность непрерывной работы с ВДТ и ПЭВМ без регламентированного перерыва не должна превышать 2-х часов. Через каждый час работы следует делать регламентированный перерыв про­должительностью 15 мин.

3.9. Во время регламентированных перерывов с целью снижения не­рвно-эмоционального напряжения, утомления зрительного анализатора, устранения влияния гиподинамии и гипокинезии, предотвращения раз­вития познотонического утомления следует выполнять комплексы упраж­нений для глаз, физкультурные минутки и физкультурные паузы.

4. Требования безопасности в аварийных ситуациях

4.1.В случае появления неисправности в работе видеотерминала вык­лючить его, сообщить об этом администрации учреждения. Работу про­должать только после устранения возникшей неисправности.

4.2. В случае возникновения у пользователя зрительного дискомфорта и других неблагоприятных субъективных ощущений следует ограничить время работы с ВДТ и ПЭВМ, провести коррекцию длительности пе­рерывов для отдыха или провести смену деятельности на другую, не связанную с использованием ВДТ и ПЭВМ.

4.3. При поражении пользователя электрическим током немедленно отключить электросеть, оказать первую помощь пострадавшему, при необходимости отправить его в ближайшее лечебное учреждение.

5. Требования безопасности по окончании работы

5.1. Выключить ВДТ и ПЭВМ, очистить их экраны от пыли.

5.2. Тщательно проветрить помещение с ВДТ и ПЭВМ.