

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
Федеральное государственное бюджетное образовательное учреждение высшего
образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет «Вычислительная техника» (наименование)	Кафедра «Системы автоматизированного проектирования» (наименование)
Направление подготовки	09.03.01 «Информатика и вычислительная техника» (код и наименование)
Профиль	«Системы автоматизированного проектирования» (наименование)

БАКАЛАВРСКАЯ РАБОТА
на тему

Разработка приложения для учета нововведений на
сайте

Студент РД Камелин Д.В.
(подпись, дата) (фамилия, инициалы)

Руководитель ИИ-8.06.2017 Семилетова И.А.
(подпись, дата) (фамилия, инициалы)

Нормоконтролёр ИИ.806.САПР ПГУ ИИ 8.06.17 Богдан А.С.
(должность, место работы) (подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой _____
(подпись) (фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК _____
(подпись) (фамилия, инициалы)

Пенза 2017

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	5
1. АНАЛИЗ СУЩЕСТВУЮЩИХ СИСТЕМ АНАЛОГИЧНОГО НАЗНАЧЕНИЯ	7
1.1. Анализ существующих систем.....	7
1.2. Анализ технического задания.....	10
2. РАЗРАБОТКА СТРУКТУРЫ ПРИЛОЖЕНИЯ.....	15
2.1. Проектирование базы данных	15
2.1.1. Построение концептуальной модели.....	15
3. РАЗРАБОТКА ПРОГРАММНОГО И ИНФОРМАЦИОННОГО ОБЕСПЕЧЕНИЯ.....	21
3.1. Разработка хранимых процедур	21
3.2. Физическая модель	23
3.3. Создание SQL запроса на создание базы данных	25
3.4. Разработка приложения для работы с базой данных	26
3.4.1. Описание классов.....	26
3.4.2. Визуализация.....	29
4. ТЕСТИРОВАНИЕ	34
4.1. Тестирование серверной части.....	34
4.2. Тестирование приложения.....	36
ЗАКЛЮЧЕНИЕ	37
СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ.....	38
ПРИЛОЖЕНИЕ А	39
ПРИЛОЖЕНИЕ Б.....	47
ПРИЛОЖЕНИЕ В	49

ВВЕДЕНИЕ

Данные окружают нас повсюду. Мы получаем их из средств массовой информации или прогуливаясь по улице, натыкаемся на них на рабочем месте или дома, листая страницы в сети Интернет. Поскольку каждый современный человек постоянно встречается с огромным количеством информации, ему нужно эту информацию накапливать, хранить, сортировать, упорядочивать или удалять.

База данных - некоторый упорядоченный массив информации. Чаще всего для представления данных используют таблицы. В ячейках расположена информация, а столбцы и строки содержат признаки, по которым происходит упорядочивание. Важной проблемой для правильного построения базы данных является выяснение наборов признаков, которые являются общими для предметной области. Именно в зависимости от значения этих признаков отделяются экземпляры каждой конкретной сущности.

Система управления базами данных (СУБД) — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных [1].

Данные недостаточно просто получить, необходимо еще производить над ними некоторые операции. В первую очередь, имеющуюся упорядоченную информацию нужно сохранить. В дальнейшем нужно получать именно те данные, которые требуются пользователю в конкретный момент времени. Для этого управляющая система должна обладать неким набором алгоритмов сортировки, а также разнообразными фильтрами. При правильном построении СУБД это все обеспечивает, а также реализует другие требуемые функции.

Главной целью этой выпускной квалификационной работы является разработка приложения для ведения товарооборота на складе. В данной работе будут описаны этапы разработки и тестирования этого программного обеспечения.

					ВКР.090301.09.13 81 17	Лист
						5
Изм.	Лист	№ докум.	Подпись	Дата		

Программа спроектирована и разработана в Visual Studio 2015 на языке программирования С#.Серверная часть приложения будет реализована в СУБД Microsoft SQL Server 2012 средствами языка Transact-SQL.

					<i>ВКР.090301.09.13 81 17</i>	Лист
						6
Изм.	Лист	№ докум.	Подпись	Дата		

1. АНАЛИЗ СУЩЕСТВУЮЩИХ СИСТЕМ АНАЛОГИЧНОГО НАЗНАЧЕНИЯ

В этом разделе подробно описаны основные требования к разрабатываемому приложению.

1.1. Анализ существующих систем

Существует множество систем для ведения товарооборота. Большинство из них либо слишком большие и комплексные, либо написаны под конкретную предметную область и использование их в условиях, даже незначительно отличающихся от тех, для которых они созданы, может быть невозможно. Мною были выбраны несколько систем для сравнения.

Облачный сервис «Мой склад».

Данная система позволяет вести учет товаров в магазине или торговой сети, на складе. Она поддерживает фискальный регистратор и штрих-сканнер, не требует наличия собственного сервера, проста во внедрении и предоставляет пробный период – 14 дней. Далее требуется абонентская плата. Программа требует подключения к сети Интернет, так как данные сохраняются в облачном хранилище. Зато она крайне нетребовательна к аппаратным средствам и может запускаться практически на любом устройстве. Среди преимуществ данного программного продукта можно выделить:

- Возможность выполнения широкого спектра задач, связанных с учетом товаров, ведением базы клиентов в Excel, контролем над финансовыми потоками;
- Достаточно простой запуск сервиса в эксплуатацию: можно интуитивно понять, как вести учет, для работы с программой и ведения учета не нужны специальные умения и знания;
- В программе предусмотрена возможность вести учет товаров дистанционно;

					ВКР.090301.09.13 81 17	Лист
						7
Изм.	Лист	№ докум.	Подпись	Дата		

- Квалифицированная техническая поддержка;
- Защита коммерческой информации, конфиденциальность данных.

Одним из главных недостатков, на мой взгляд, является требование постоянного Интернет соединения, так как, несмотря на развитие технологий, все еще остаются зоны, в которых отсутствует подключение к сети Интернет или сигнал слишком слаб.

Программа учета товарооборота «VsShop».

Данная программа предназначена для ведения товарооборота в небольших магазинах. Пользователь может без специального обучения и подготовки применять данную программу для своих нужд. Также программа относится к серии portable программа, то есть ее не нужно устанавливать в систему. Программа чудесно работает с флеш носителя, что обеспечивает дополнительные удобства в плане мобильности. Приложение работает под управлением системы Windows XP или выше. Никаких дополнительных драйверов для работы с базой данных не требуется.

Основные возможности программы:

- Автоматизация учета оборотной ведомости;
- Автоматизация ведения таблицы проверок;
- Календарь товарооборота;
- Инструментарий ведения переучета товаров;
- Ведение статистики реализации товаров.

Данная программа является платной. В отличие от предыдущей системы оплата происходит только один раз.

					<i>ВКР.090301.09.13 81 17</i>	Лист
						8
Изм.	Лист	№ докум.	Подпись	Дата		

Таблица 1 - Достоинства и недостатки систем аналогов

Критерий	«Мой склад»	«VsShop»
Оформление интерфейса	+	+
Дружественность интерфейса	+	-
Ведение учета товаров	+	+
Вывод отчетности в документ	+	+
Расчет денежного оборота	+	+
Потребность в инсталляции	+	-
Необходимость оплаты	+	+

Расшифровка знаков:

- «+» Данный пункт удовлетворяет полностью;
- «-» Данный пункт не удовлетворяет.

Мое приложение также обладает дружелюбным интерфейсом, ведет учет товаров и выводит отчеты в документ, но не ведет расчет денежного оборота. Кроме всего прочего, мое приложение будет являться полностью бесплатным и ее не нужно устанавливать в систему.

1.2. Анализ технического задания

Основные требования к базе данных:

- Удовлетворяет всем требованиям пользователей к содержимому базы данных;
- Гарантирует непротиворечивость и целостность данных;
- Обеспечивает естественное, легкое для восприятия структурирование информации;
- Удовлетворяет требованиям пользователей к производительности базы данных.

Основные требования к пользовательскому приложению:

- Простой и понятный интерфейс;
- Управление вызовами хранимых процедур базы данных;
- Изменение, добавление и удаление записей в базе данных;
- Вывод, поиск и сортировка данных.

Результатом работы должно быть приложение для работы с базой данных.

Выбранные технологии:

- Microsoft Visual Studio 2015;
- Microsoft SQL Server 2012;
- Язык программирования C#;
- Microsoft Visio 2010;
- ERwin Data Modeler.

Подробнее о выбранных технологиях.

Microsoft Visual Studio – продукт Microsoft, включающий интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данная программа позволяет разрабатывать различные приложения как консольные, так и с графическим интерфейсом, например, Windows Forms, а также веб-приложения и веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET

					ВКР.090301.09.13 81 17	Лист
						10
Изм.	Лист	№ докум.	Подпись	Дата		

Framework, Xbox, Windows Phone .NET Compact Framework и Silverlight. Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик способен работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio позволяет создавать и подключать плагины для расширения функциональности почти на любом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual SourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения (например, клиент Team Explorer для работы с Team Foundation Server) [2].

MicrosoftSQLServer –система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основным используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка [3].

Язык программирования C# — объектно-ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров под руководством Андерса Хейлсберга в компании Microsoft как язык разработки приложений для платформы Microsoft .NETFramework и впоследствии был стандартизирован как ECMA-334 и ISO/IEC 23270.

					<i>ВКР.090301.09.13 81 17</i>	Лист
						11
Изм.	Лист	№ докум.	Подпись	Дата		

C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.

Переняв многое от своих предшественников — языков C++, Pascal, Модула, Smalltalk и, в особенности, Java — C#, опираясь на практику их использования, исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем, например, C# в отличие от C++ не поддерживает множественное наследование классов (между тем допускается множественное наследование интерфейсов) [4].

Microsoft Visio – векторный графический редактор, редактор диаграмм и блок-схем для Windows [5].

Ядром любой базы данных является модель данных. Модель данных представляет собой множество структур данных, ограничений целостности и операций манипулирования данными. Различные объекты предметной области и их связи друг с другом могут быть представлены с помощью модели данных.

Рассмотрим три основных типа моделей данных: иерархическую, сетевую и реляционную.

Иерархическая структура представляет совокупность элементов, связанных между собой по определенным правилам. Объекты, связанные иерархическими отношениями, образуют ориентированный граф (перевернутое дерево).

К основным понятиям иерархической структуры относятся: уровень, элемент (узел), связь. Узел — это совокупность атрибутов данных, описывающих некоторый объект. На схеме иерархического дерева узлы представляются вершинами графа. Каждый узел на более низком уровне связан только с одним узлом, находящимся на более высоком уровне. Иерархическое дерево имеет

только одну вершину (корень дерева), не подчиненную никакой другой вершине и находящуюся на самом верхнем (первом) уровне. Зависимые (подчиненные) узлы находятся на втором, третьем и т.д. уровнях. Количество деревьев в базе данных определяется числом корневых записей.

К каждой записи базы данных существует только один (иерархический) путь от корневой записи.

В сетевой структуре при тех же основных понятиях (уровень, узел, связь) каждый элемент может быть связан с любым другим элементом.

Реляционная модель данных

Понятие реляционный (англ. relation — отношение) связано с разработками известного американского специалиста в области систем баз данных Е. Кодда.

Эти модели характеризуются простотой структуры данных, удобным для пользователя табличным представлением и возможностью использования формального аппарата алгебры отношений и реляционного исчисления для обработки данных.

Реляционная модель ориентирована на организацию данных в виде двумерных таблиц. Каждая реляционная таблица представляет собой двумерный массив и обладает следующими свойствами:

- каждый элемент таблицы — один элемент данных;
- все столбцы в таблице однородные, т.е. все элементы в столбце имеют одинаковый тип (числовой, символьный и т.д.) и длину;
- каждый столбец имеет уникальное имя;
- одинаковые строки в таблице отсутствуют;
- порядок следования строк и столбцов может быть произвольным.

Отношения представлены в виде таблиц, строки которых соответствуют кортежам или записям, а столбцы — атрибутам отношений, доменам, полям.

Поле, каждое значение которого однозначно, определяет соответствующую запись, и называется простым ключом (ключевым полем). Если записи

					ВКР.090301.09.13 81 17	Лист
						13
Изм.	Лист	№ докум.	Подпись	Дата		

однозначно определяются значениями нескольких полей, то такая таблица базы данных имеет составной ключ.

Чтобы связать две реляционные таблицы, необходимо ключ первой таблицы ввести в состав ключа второй таблицы (возможно совпадение ключей); в противном случае нужно ввести в структуру первой таблицы внешний ключ — ключ второй таблицы.

Анализ существующих решений показал, что выгоднее использовать реляционную модель представления данных, так как она отличается простотой структуры данных и наглядностью их предоставления.

					<i>ВКР.090301.09.13 81 17</i>	Лист
						14
Изм.	Лист	№ докум.	Подпись	Дата		

2. РАЗРАБОТКА СТРУКТУРЫ ПРИЛОЖЕНИЯ

В этом разделе подробно описывается этап проектирования приложения.

2.1. Проектирование базы данных

2.1.1. Построение концептуальной модели

Начать проектирование стоит с определения границ предметной области. Границами являются сам склад и происходящие в нем процессы: поставки и вывозы товаров. Исходя из этого, можно выделить следующие бизнес-процессы:

- Поставка. Происходит занесение новой поставки в соответствующую таблицу, занесение товара в таблицу, если данного товара там не было; если товар уже есть, то увеличивается количество товара.
- Вывоз. Происходит занесение нового вывоза в соответствующую таблицу, еще вычитается количество вывезенных товаров.

Более подробно эти бизнес-процессы представлены в таблице 2.

Таблица 2 – Бизнес-процессы предметной области.

№	Название	Исполнитель	Входные данные		Выходные данные		Ограничения на БП
			Поставщик	Содержание	Потребитель	Содержание	
1	Поставка	Заведующий складом	Заведующий складом	Имя поставщика, дата, наименование товаров, артикул, тип, цвет, количество, единицы измерения, цена, номер склада	Клиент	ID поставщика, ID товара, ID склада, ID поставки	

продолжение табл. 2

№	Название	Исполнитель	Входные данные		Выходные данные		Ограничения на БП
			Поставщик	Содержание	Потребитель	Содержание	
2	Вывоз	Заведующий складом	Заведующий складом	Имя покупателя, дата, наименование товара, цвет количество	Клиент	Отчет о покупке, ID вывоза	

На основе описания бизнес-процессов строится исходная концептуальная модель данных, элементами которой являются сущности и связи между ними. Более подробное их описание представлено в таблицах 3 и 4.

Таблица 3 – Сущности предметной области.

№	Наименование	Описание	Тип сущности	Количество экземпляров	Изменяемость
1	Товары	ID, ID склада, наименование, артикул, тип, цвет, единицы измерения, количество, цена	стержневая	10	5 в день
2	Склады	ID, номер склада, месторасположение	справочная	3	1 в год
3	Клиенты	ID, имя, адрес, телефон	справочная	6	2 в день
4	Поставки	ID, ID клиента, дата	стержневая	5	10 в день
5	Вывоз	ID, ID клиента, дата	стержневая	4	10 в день
6	Смежная таблица	ID, ID товара, ID поставки	ассоциативная	20	40 в день

продолжение табл. 3

№	Наименование	Описание	Тип сущности	Количество экземпляров	Изменяемость
7	Смежная таблица 2	ID, ID товара, ID вывоза	ассоциативная	24	40 в день

Таблица 4- Связи между сущностями.

№	Наименование главное сущности	Наименование подчинённой сущности	Смысл связи	Тип связи	Ограничения
1	Склад	Товары		1:M	
2	Клиенты	Поставки		1:M	
3	Клиенты	Вывозы		1:M	
4	Товары	Смежная таблица		1:M	
5	Товары	Смежная таблица 2		1:M	
6	Поставки	Смежная таблица		1:M	
7	Вывозы	Смежная таблица 2		1:M	

EntityRelation-модель – модель данных, позволяющая описывать концептуальные системы, представляет собой графическую нотацию, основанную на блоках и соединяющих их линиях, с помощью которых можно описывать объекты и отношения между ними. В этом смысле ER-модель является метамоделью данных, то есть средством описания моделей данных [6]. На основе выделенных сущностей и связей строится инфологическая модель, представленная на рисунке 1 в виде диаграммы сущность-связь.

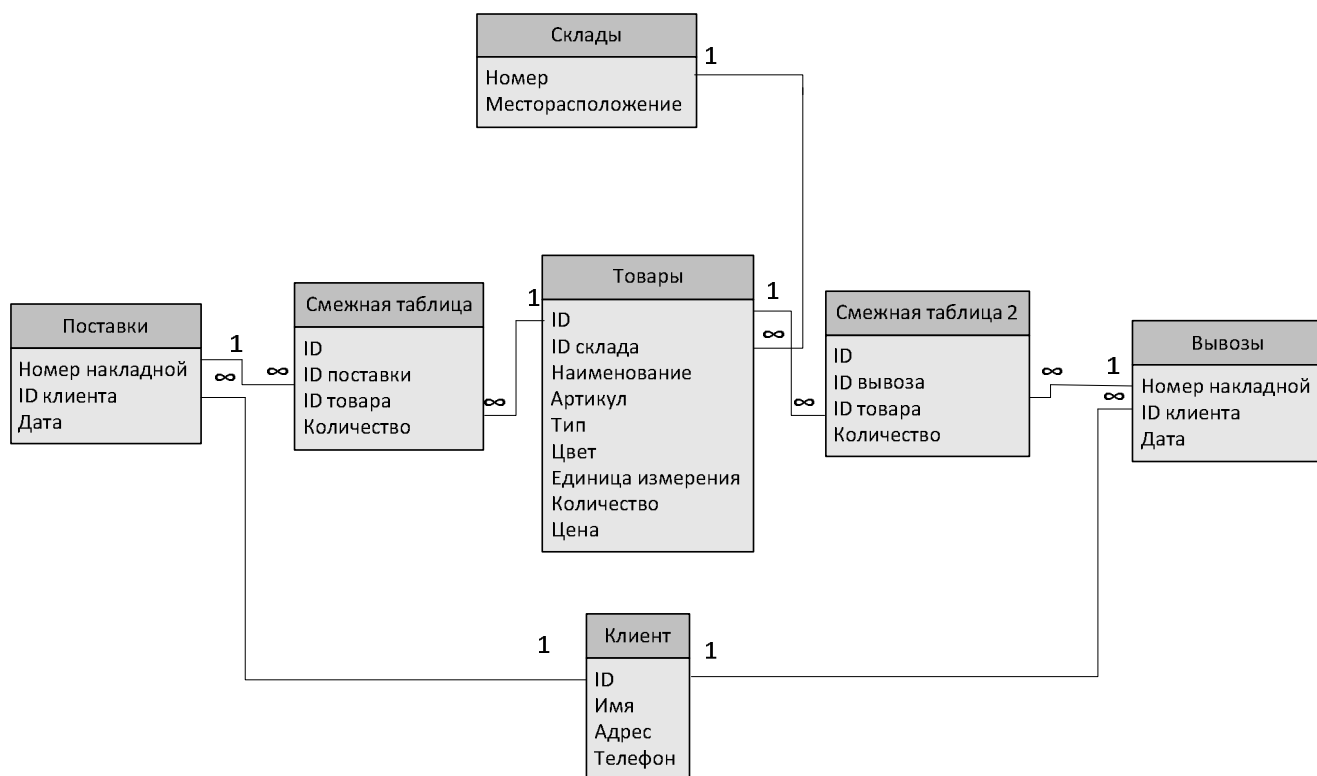


Рисунок 1 – Диаграмма сущность-связь

В таблицах 5-11 представлены пояснения к сущностям.

Таблица 5 – Склады.

№	Наименование	Тип	Роль атрибута в пределах сущности	Обязательность	Ограничения
1	Номер	Целый	Первичный ключ	Not NULL	«0-9»
3	Место	Символьный		NULL	«А-Я»

Таблица 6 – Поставки.

№	Наименование	Тип	Роль атрибута в пределах сущности	Обязательность	Ограничения
1	Номер накладной	Целый	Первичный ключ	Not NULL	«0-9»
2	ID клиента	Целый	Внешний ключ	Not NULL	«0-9»
3	Дата	Дата		NULL	«0-9»

Таблица 7 – Вывозы.

№	Наименование	Тип	Роль атрибута в пределах сущности	Обязательность	Ограничения
1	Номер накладной	Целый	Первичный ключ	Not NULL	«0-9»
2	ID клиента	Целый	Внешний ключ	Not NULL	«0-9»
3	Дата	Дата		NULL	«0-9»

Таблица 8 – Товары.

№	Наименование	Тип	Роль атрибута в пределах сущности	Обязательность	Ограничения
1	ID	Целый	Первичный ключ	Not NULL	«0-9»
2	Номер склада	Целый	Внешний ключ	Not NULL	«0-9»
3	Наименование	Символьный		NULL	«А-Я»
4	Артикул	Целый		NULL	«0-9»
5	Тип	Символьный		NULL	«А-Я»
6	Цвет	Символьный		NULL	«А-Я»
7	Единицы измерения	Символьный		Not NULL	«А-Я»
8	Количество	Целый		Not NULL	«0-9»
9	Цена	Целый		Not NULL	«0-9»

Таблица 9 – Клиенты.

№	Наименование	Тип	Роль атрибута в пределах сущности	Обязательность	Ограничения
1	ID	Целый	Первичный ключ	Not NULL	«0-9»
2	Имя	Целый		Not NULL	«А-Я»
3	Адрес	Символьный		NULL	«А-Я»
4	Телефон	Символьный		Not NULL	«0-9»

Таблица 10 – Смежная таблица.

№	Наименование	Тип	Роль атрибута в пределах сущности	Обязательность	Ограничения
1	ID	Целый	Первичный ключ	Not NULL	«0-9»
2	ID поставки	Целый	Внешний ключ	Not NULL	«0-9»
3	ID товара	Целый	Внешний ключ	Not NULL	«0-9»

Таблица 11 – Смежная таблица 2.

№	Наименование	Тип	Роль атрибута в пределах сущности	Обязательность	Ограничения
1	ID	Целый	Первичный ключ	Not NULL	«0-9»
2	ID вывоза	Целый	Внешний ключ	Not NULL	«0-9»
3	ID товара	Целый	Внешний ключ	Not NULL	«0-9»

3. Разработка программного и информационного обеспечения

3.1. Разработка хранимых процедур

Перечень хранимых процедур представлен в таблице 12. Более подробно они представлены в приложении Б.

Таблица 12 – Хранимые процедуры.

№ п/п	Имя процедуры	Назначение процедуры	Краткое описание работы процедуры	Параметры
1	Supdate	Вывод поставок по дате	Вывод всех поставок, дата которых равна выбранной	Дата
2	Supclient	Вывод поставок по клиенту	Вывод всех поставок, у которых одинаково имя клиента	Имя
3	Remdate	Вывод вывозов по дате	Вывод всех вывозов, дата которых равна выбранной	Дата
4	Remclient	Вывод вывозов по клиенту	Вывод всех вывозов, у которых одинаково имя клиента	Имя
5	Add_sup	Добавление поставки	Добавление новой поставки. У каждой поставки выбирается дата, имя клиента и перечень товаров.	Имя клиента Дата Наименование товара Артикул Цвет Количество Единицы

продолжение табл. 12

№ п/п	Имя процедуры	Назначение процедуры	Краткое описание работы процедуры	Параметры
				измерения Цена Номер склада
6	Add_rem	Добавление вывоза	Добавление нового вывоза. У каждого вывоза выбирается дата, имя клиента и перечень товаров.	Имя клиента Дата Наименование товара Цвет Количество
7	Add_client	Добавление клиента	Добавляет клиента в базу	Имя Адрес Номер телефона
8	Del_client	Удаление клиента	Удаляет клиента из базы	Имя
9	Add_war	Добавление склада	Добавляют склад в базу	Номер склада Месторасполо- жение
10	Rem_war	Удаление склада	В таблицу Room по ID_class добавляет столбец Name (название класса)	Номер склада
11	Search_client	Поиск по клиенту	Вывод клиентов, имя которых содержит текст, введенный в строку поиска	Имя клиента

№ п/п	Имя процедуры	Назначение процедуры	Краткое описание работы процедуры	Параметры
12	Search_goods	Поиск по товару	Вывод товаров, наименование которых содержит текст, введенный в строку поиска	Наименование товара

3.2. Физическая модель

Физическое проектирование базы данных - процесс подготовки описания реализации базы данных на вторичных запоминающих устройствах; на этом этапе рассматриваются основные отношения, организация файлов и индексов, предназначенных для обеспечения эффективного доступа к данным, а также все связанные с этим ограничения целостности и средства защиты.

Физическое проектирование является третьим и последним этапом создания проекта базы данных, при выполнении которого проектировщик принимает решения о способах реализации разрабатываемой базы данных. Во время предыдущего этапа проектирования была определена логическая структура базы данных (которая описывает отношения и ограничения в рассматриваемой прикладной области). Хотя эта структура не зависит от конкретной целевой СУБД, она создается с учетом выбранной модели хранения данных, например, реляционной, сетевой или иерархической. Однако, приступая к физическому проектированию базы данных, прежде всего, необходимо выбрать конкретную целевую СУБД. Поэтому физическое проектирование неразрывно связано с конкретной СУБД. Между логическим и физическим проектированием существует постоянная обратная связь, так как решения, принимаемые на этапе физического проектирования с целью повышения производительности системы, способны повлиять на структуру логической модели данных.

Как правило, основной целью физического проектирования базы данных является описание способа физической реализации логического проекта базы данных [7].

В случае реляционной модели данных под этим подразумевается следующее:

- создание набора реляционных таблиц и ограничений для них на основе информации, представленной в глобальной логической модели данных;
- обеспечение оптимальной производительности СУБД с помощью определения конкретных структур хранения данных и методов доступа к ним;
- разработка методов и средств защиты разрабатываемой системы.

В данном дипломном проекте физическая модель для базы данных разрабатывалась с использованием CASE-средства ErWinDataModeler. CASE-средство ErWinDataModeler поддерживает стандарт методологии IDEF1X.

					<i>ВКР.090301.09.13 81 17</i>	Лист
						24
Изм.	Лист	№ докум.	Подпись	Дата		

3.3.Создание SQL запроса на создание базы данных

После создания физической модели базы данных, был сгенерирован скрипт на языке запросов Transact SQL, создающий таблицы и поля базы данных (см. приложение А).

					<i>ВКР.090301.09.13 81 17</i>	Лист
						25
Изм.	Лист	№ докум.	Подпись	Дата		

3.4. Разработка приложения для работы с базой данных

3.4.1. Описание классов

Интерфейс пользователя клиентского приложения ИС реализован в виде графического оконного интерфейса. Для отображения используются несколько окон, которые выполнены с помощью Windows Forms. Листинги приложения приведены в приложении В.

Список классов с описанием представлен ниже:

- main.cs – главное меню. Из него можно попасть в меню просмотра клиентов/товаров/ складов, в меню добавления поставки/вывоза, в меню добавления клиентов и складов, в меню удаления или выйти из приложения. В окне присутствуют 5 элементов:
 - Button1 – переход в view.cs;
 - Button2 – переход в add.cs;
 - Button3 – переход в add_new.cs;
 - Button4 – переход в delete.cs;
 - Button5 – выход из приложения;
- view.cs – окно просмотра клиентов/товаров/ складов. Здесь можно посмотреть список клиентов/товаров/ складов, переключаясь с помощью соответствующих кнопок. В форме присутствуют 5 элементов:
 - Button1 – переключиться на просмотр клиентов (по умолчанию);
 - Button2 – переключиться на просмотр товаров;
 - Button3 – переключиться на просмотр складов;
 - Button4 – возврат в main.cs;
 - dataGridView1 – поле для отображения данных;
- add.cs – меню поставки/вывоза. Отсюда можно перейти к добавлению новых поставок/вывозов. В форме присутствуют 3 элемента:

- Button1 – переход в форму;
- Button2 – добавление нового вызова;
- Button3 – возвращение в main.cs;
- Supply.cs – меню для добавления новой поставки. В форме присутствуют 15 элементов:
 - datePicker1 – выбор даты поставки;
 - TextBox1 – поле для ввода номера накладной;
 - ComboBox1 – поле для выбора клиента;
 - dataGridView1 – поле для отображения доставленных товаров;
 - TextBox2 – поле для ввода наименования товара;
 - TextBox3 – поле для ввода номера склада товара;
 - TextBox4 – поле для ввода артикула товара;
 - TextBox5 – поле для ввода типа товара;
 - TextBox6 – поле для ввода цвета товара;
 - TextBox7 – поле для ввода количества товара;
 - TextBox8 – поле для ввода единиц измерения товара;
 - TextBox9 – поле для ввода стоимости товара;
 - Button1 – кнопка для добавления товара в список доставленных;
 - Button2 – кнопка для завершения поставки;
 - Button3 – кнопка для возврата в форму add.cs;
- Removal.cs – меню для добавления нового вывоза. В форме присутствуют 7 элементов:
 - datePicker1 – выбор даты вывоза;
 - TextBox1 – поле для ввода номера накладной;
 - ComboBox1 – поле для выбора клиента;
 - dataGridView1 – поле для отображения вывозимых товаров;
 - dataGridView1 – поле для отображения всех товаров;
 - Button1 – кнопка для завершения вывоза;

- Button2 – кнопка для возврата в форму add.cs;
- add_new.cs – меню для добавления. Отсюда можно добавить новых клиентов или новый склад и выйти в главное меню. В окне присутствуют 3 элемента:
 - Button1 - переход в new_client.cs;
 - Button2 - переход в new_war.cs;
 - Button3–возвращение в main.cs;
- new_client.cs – меню добавления нового клиента. Добавить нового клиента можно, введя данные в соответствующие поля и нажав кнопку ОК. В окне присутствуют 5 элементов:
 - Button1–кнопка ОК;
 - Button2–возврат в add_new.cs;
 - Textbox1 –поле для ввода имени;
 - Textbox2 –поле для ввода адреса;
 - Textbox3 –поле для ввода телефона;
- new_war.cs–меню добавления нового склада. Добавить новый склад можно, введя данные в соответствующие поля и нажав кнопку ОК. В форме присутствуют 4 элемента:
 - Button1 – кнопка ОК;
 - Button2 – возврат в add_new.cs;
 - Textbox1 – поле для ввода номера склада;
 - Textbox2 – поле для ввода расположения склада;
- delete.cs - окно удаления. В окне присутствуют 7 элементов:
 - Button1 – выполнить поиск;
 - Button2 – переключиться на клиентов;
 - Button3 – переключиться на склады;
 - Button4 – выполнить удаление;
 - Button5 - возврат в main.cs;

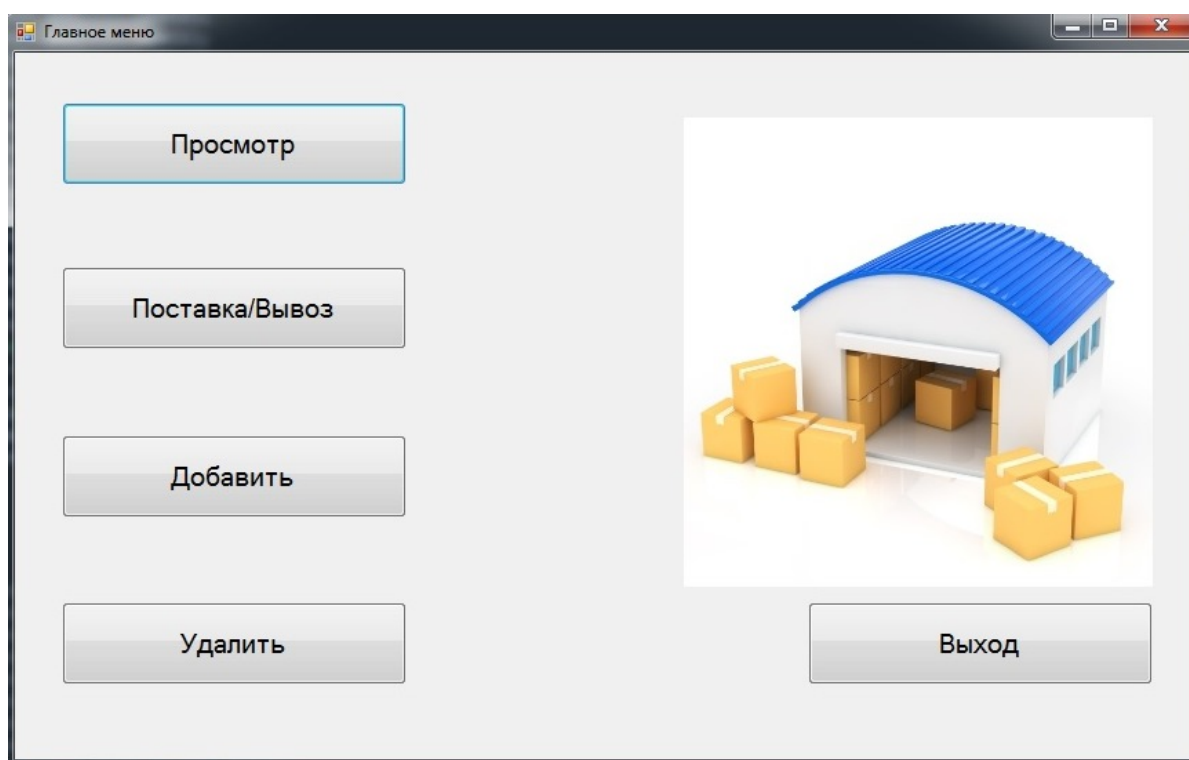


Рисунок 2 – Главное меню

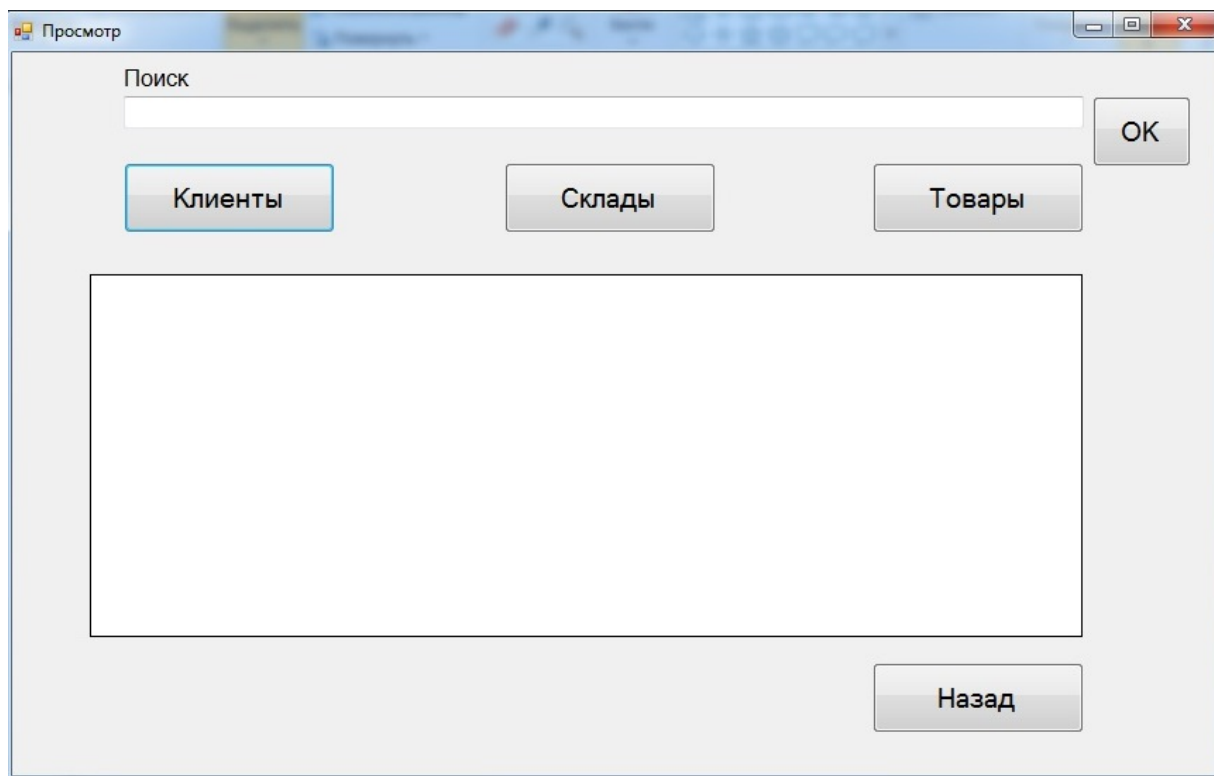


Рисунок 3 – Просмотр

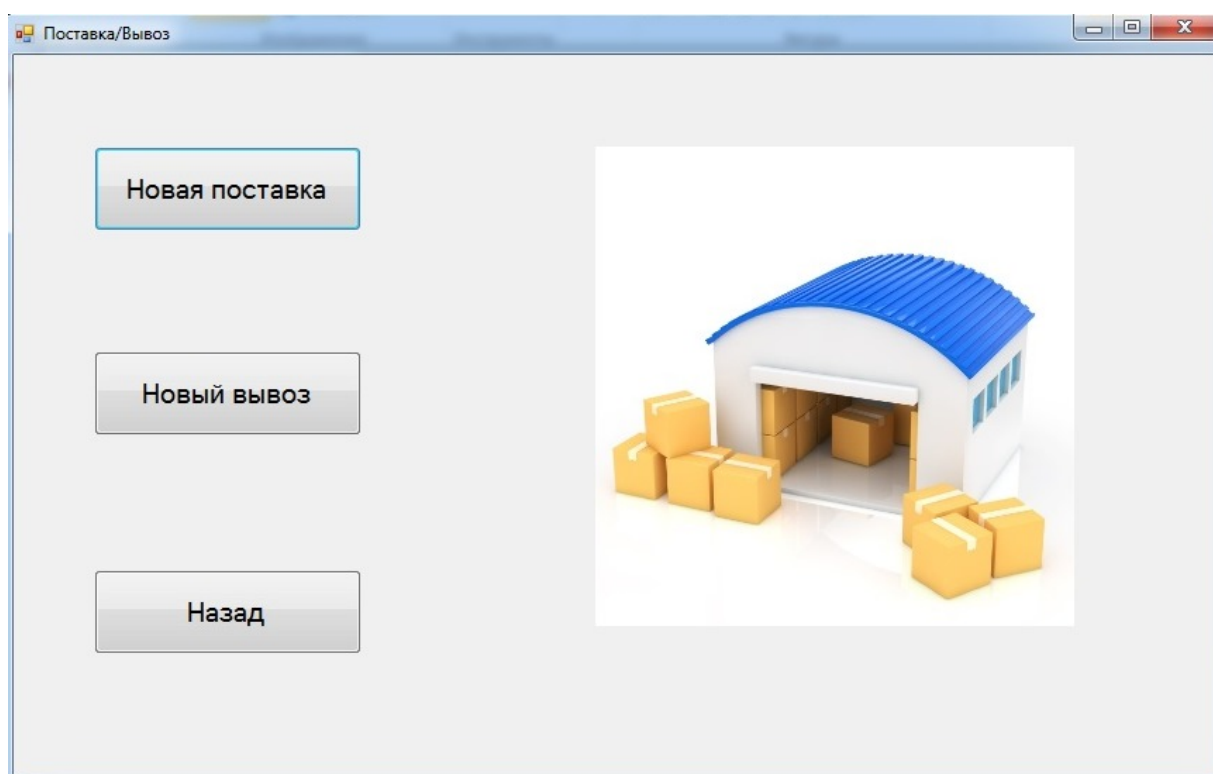


Рисунок 4 – Поставка/Вывоз

Новая поставка

Дата
9 июня 2017 г.

Номер накладной

Имя клиента

Наименование	Номер склада	Артикул	Тип	Цвет	Количество	Единицы	Цена
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

OK

Завершить Назад

Рисунок 5 – Новая поставка

Новый вывоз

Дата
9 июня 2017 г.

Имя клиента

<input type="text"/>
<input type="text"/>

Завершить Назад

Рисунок 6 – Новый вывоз

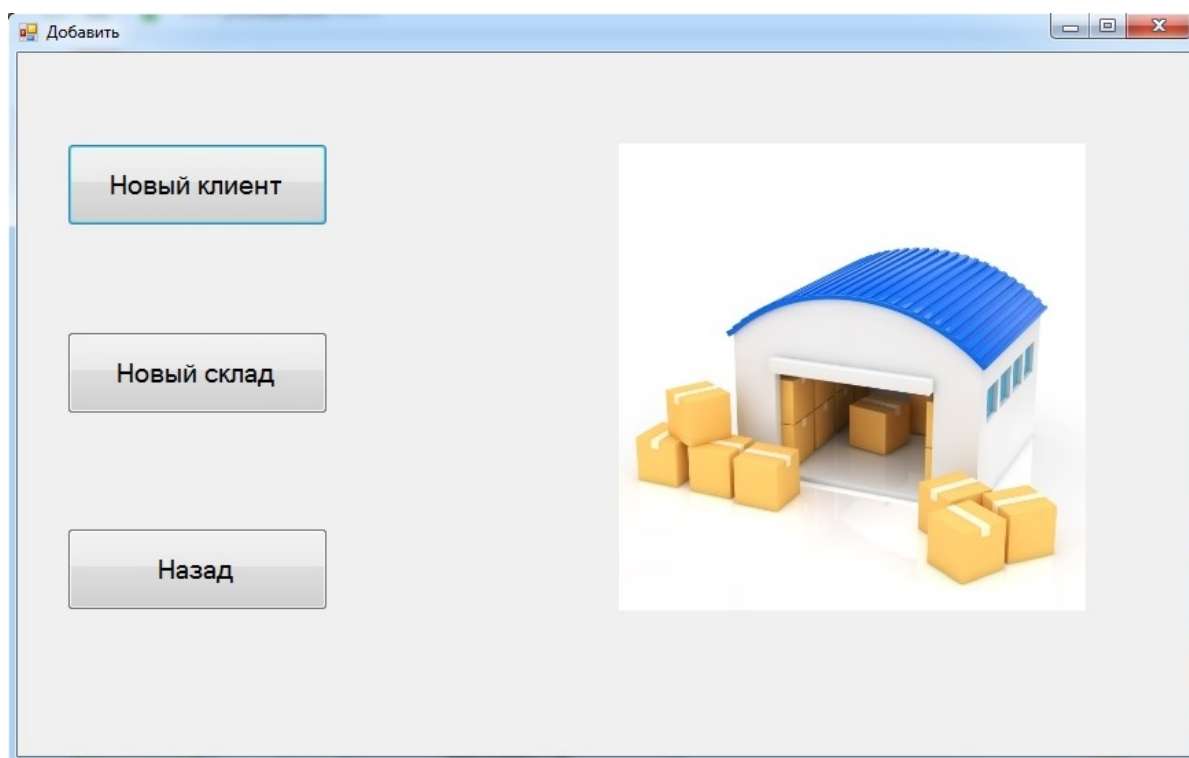


Рисунок 7 – Меню добавления

Рисунок 8 – Добавление нового клиента

Новый склад

Номер

Месторасположение

OK Назад

Рисунок 9 – Добавление нового склада

Удаление

Поиск

OK

Клиенты Склады

Удалить Назад

Рисунок 10 – Меню удаления

4. Тестирование

Тестирование программного обеспечения важный этап разработки. На этом этапе выявляются ошибки, недоработки программного обеспечения.

Тестирование имеет две главные цели:

- Продемонстрировать разработчикам и заказчикам, что программа соответствует требованиям;
- Выявить ситуации, в которых поведение программы является неправильным, нежелательным или не соответствующим спецификации.

4.1. Тестирование серверной части

Таблица 13 – Тестирование серверной части

№ п/п	Назначение теста	Результаты теста
1	Проверка хранимой процедуры Supdate	Процедура работает без ошибок.
2	Проверка хранимой процедуры Supclient	Процедура работает без ошибок.
3	Проверка хранимой процедуры Remdate	Процедура работает без ошибок.
4	Проверка хранимой процедуры Remclient	Процедура работает без ошибок.
5	Проверка хранимой процедуры Add_sup	Процедура работает без ошибок.
6	Проверка хранимой процедуры Add_rem	Процедура работает без ошибок.

продолжение табл. 13

№ п/п	Назначение теста	Результаты теста
7	Проверка хранимой процедурыAdd_war	Процедура работает без ошибок.
8	Проверка хранимой процедурыAdd_client	Процедура работает без ошибок.
9	Проверка хранимой процедурыDel_client	Процедура работает без ошибок.
10	Проверка хранимой процедурыRem_war	Процедура работает без ошибок.
11	Проверка хранимой процедурыSearch_Client	Процедура работает без ошибок.
12	Проверка хранимой процедурыSearch_goods	Процедура работает без ошибок.

4.2. Тестирование приложения

Таблица 14 – Тестирование клиентского приложения

№ п/п	Назначение теста	Описание теста	Результаты теста
1	Запуск программы	Происходит запуск программы	Программа запускается без ошибок
2	Проверка работы меню	Выбирается любой пункт меню	Открытие необходимого окна происходит без ошибок
3	Проверка работы с базой данных	Возможность просмотра баз данных, добавления и удаления записей	Программа работает без ошибок.

В ходе тестирования был выявлен недостаток в классе add.cs. Не работала кнопка возврата в главное меню. Данная неисправность была устранена.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы был пройден полный путь разработки программного обеспечения. Спроектирована база данных системы, разработан пользовательский интерфейс.

Так же спроектирован и реализован прототип программного обеспечения для ведения товарооборота. В нем использована разработанная структура базы данных, наполнена тестовым набором данных и оттестирована на основных функциях системы. На этапе тестирования не было выявлено серьезных недостатков в работе программы. Отсюда можно сделать вывод, что тестирование успешно пройдено.

В дальнейшем прототип программного обеспечения планируется доработать: доработать дизайн интерфейса, сделать его более простым для понимания пользователем.

В заключении можно сказать, что все поставленные задачи выполнены.

					<i>ВКР.090301.09.13 81 17</i>	Лист
						37
Изм.	Лист	№ докум.	Подпись	Дата		

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Система управления базами данных, Википедия [Электронный ресурс]. Режим доступа: свободный, https://ru.wikipedia.org/wiki/Система_управления_базами_данных (Дата обращения: 20.05.2017).
2. Microsoft Visual Studio, Википедия [Электронный ресурс]. Режим доступа: свободный, https://ru.wikipedia.org/wiki/Microsoft_Visual_Studio (Дата обращения: 25.05.2017).
3. MicrosoftSQLServer, Википедия [Электронныйресурс]. Режим доступа: свободный, https://ru.wikipedia.org/wiki/Microsoft_SQL_Server (Дата обращения: 26.05.2017).
4. CSharp, Википедия [Электронныйресурс]. Режим доступа: свободный, https://ru.wikipedia.org/wiki/C_Sharp (Дата обращения: 27.05.2017).
5. MicrosoftVisio, Википедия [Электронныйресурс]. Режим доступа: свободный, https://ru.wikipedia.org/wiki/Microsoft_Visio (Дата обращения: 29.05.2017).
6. ER-модель [Электронный ресурс]. Режим доступа: свободный, <https://ru.wikipedia.org/wiki/ER-модель> (дата обращения: 24.05.2017).
7. Физическое проектирование базы данных [Электронный ресурс]. Режим доступа: свободный, <http://bourabai.ru/dbt/dbms/03.htm> (дата обращения: 24.05.2017). <https://ru.wikipedia.org/wiki/ER-модель>

ПРИЛОЖЕНИЕ А

SQL-скрипт запроса, создающего таблицу.

```
CREATE TABLE Clients
```

```
(  
    ID_client      integer NOT NULL ,  
    Name           varchar(20) NULL ,  
    Adress         varchar(20) NULL ,  
    Phone          integer NULL  
)  
go
```

```
ALTER TABLE Clients
```

```
    ADD CONSTRAINT XPKClients PRIMARY KEY CLUSTERED  
(ID_client ASC)  
go
```

```
CREATE TABLE Goods
```

```
(  
    Name           varchar(20) NULL ,  
    Type           varchar(20) NULL ,  
    Color          varchar(20) NULL ,  
    Unit           varchar(20) NULL ,  
    Quantity       integer NULL ,  

```

					<i>BKP.090301.09.13 81 17</i>	Лист
						39
Изм.	Лист	№ докум.	Подпись	Дата		

```

        Articul      integer NULL ,
        ID_goods     integer NOT NULL ,
        Price        integer NULL ,
        ID_war       integer NOT NULL
    )
go

```

```

ALTER TABLE Goods
    ADD CONSTRAINT XPKGoods PRIMARY KEY CLUSTERED
(ID_goods ASC,ID_war ASC)
go

```

```

CREATE TABLE Removal
(
    ID_rem          integer NOT NULL ,
    Date            datetime NULL ,
    ID_war          integer NULL ,
    ID_client       integer NOT NULL
)
go

```

```

ALTER TABLE Removal

```

					BKP.090301.09.13 81 17	Лист
						40
Изм.	Лист	№ докум.	Подпись	Дата		

```

        ADD CONSTRAINT XPKRemoval PRIMARY KEY CLUSTERED
(ID_rem ASC,ID_client ASC)

go

```

```

CREATE TABLE Removal_1
(
    ID            integer NOT NULL ,
    ID_goods      integer NOT NULL ,
    ID_rem        integer NOT NULL ,
    ID_war        integer NOT NULL ,
    ID_client     integer NOT NULL
)

go

```

```

ALTER TABLE Removal_1

        ADD CONSTRAINT XPKRemoval_1 PRIMARY KEY CLUSTERED
(ID ASC,ID_goods ASC,ID_rem ASC,ID_war ASC,ID_client ASC)

go

```

```

CREATE TABLE Supply
(
    ID_sup        integer NOT NULL ,

```

					<i>BKP.090301.09.13 81 17</i>	Лист
						41
Изм.	Лист	№ докум.	Подпись	Дата		

```

        Date            datetime NULL ,
        ID_war           integer NOT NULL ,
        ID_client        integer NOT NULL
    )
go

```

```

ALTER TABLE Supply
    ADD CONSTRAINT XPKSupply PRIMARY KEY CLUSTERED
(ID_sup ASC,ID_war ASC,ID_client ASC)
go

```

```

CREATE TABLE Supply_1
(
    ID_sup            integer NOT NULL ,
    ID_goods          integer NOT NULL ,
    ID                integer NOT NULL ,
    ID_war            integer NOT NULL ,
    ID_client         integer NOT NULL
)
go

```

```

ALTER TABLE Supply_1

```

					BKP.090301.09.13 81 17	Лист
Изм.	Лист	№ докум.	Подпись	Дата		42

```

        ADD CONSTRAINT XPKSupply_1 PRIMARY KEY CLUSTERED
(ID_goods ASC,ID_sup ASC,ID ASC,ID_war ASC,ID_client ASC)

go

```

```

CREATE TABLE Warehouses
(
    Number          integer NULL ,
    Placing          varchar(20) NULL ,
    ID_war           integer NOT NULL
)

go

```

```

ALTER TABLE Warehouses

        ADD CONSTRAINT XPKWarehouses PRIMARY KEY CLUSTERED
(ID_war ASC)

go

```

```

ALTER TABLE Goods

        ADD CONSTRAINT R_11 FOREIGN KEY (ID_war) REFERENCES
Warehouses(ID_war)

        ON DELETE NO ACTION

```

					<i>BKP.090301.09.13 81 17</i>	Лист
						43
Изм.	Лист	№ докум.	Подпись	Дата		

ON UPDATE NO ACTION

go

ALTER TABLE Removal

ADD CONSTRAINT R_13 FOREIGN KEY (ID_client) REFERENCES
Clients(ID_client)

ON DELETE NO ACTION

ON UPDATE NO ACTION

go

ALTER TABLE Removal_1

ADD CONSTRAINT R_8 FOREIGN KEY (ID_goods,ID_war)
REFERENCES Goods(ID_goods,ID_war)

ON DELETE NO ACTION

ON UPDATE NO ACTION

go

ALTER TABLE Removal_1

					<i>BKP.090301.09.13 81 17</i>	Лист
						44
Изм.	Лист	№ докум.	Подпись	Дата		

```

ADD CONSTRAINT R_9 FOREIGN KEY (ID_rem,ID_client)
REFERENCES Removal(ID_rem,ID_client)
ON DELETE NO ACTION
ON UPDATE NO ACTION

```

go

```

ALTER TABLE Supply
ADD CONSTRAINT R_10 FOREIGN KEY (ID_war) REFERENCES
Warehouses(ID_war)
ON DELETE NO ACTION
ON UPDATE NO ACTION

```

go

```

ALTER TABLE Supply
ADD CONSTRAINT R_12 FOREIGN KEY (ID_client) REFERENCES
Clients(ID_client)
ON DELETE NO ACTION
ON UPDATE NO ACTION

```

go

					<i>BKP.090301.09.13 81 17</i>	Лист
						45
Изм.	Лист	№ докум.	Подпись	Дата		

```

ALTER TABLE Supply_1
    ADD CONSTRAINT R_6 FOREIGN KEY (ID_goods,ID_war)
REFERENCES Goods(ID_goods,ID_war)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION

go

```

```

ALTER TABLE Supply_1
    ADD CONSTRAINT R_7 FOREIGN KEY (ID_sup,ID_war,ID_client)
REFERENCES Supply(ID_sup,ID_war,ID_client)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION

go

```

					<i>BKP.090301.09.13 81 17</i>	Лист
Изм.	Лист	№ докум.	Подпись	Дата		46

ПРИЛОЖЕНИЕ Б

Хранимые процедуры

dbo.Add_war

```
ALTER PROCEDURE [dbo].[Add_war]
```

```
    @adr nvarchar(50),
```

```
    @num int
```

```
AS
```

```
    INSERT INTO Warehouses(Placing, Number)
```

```
    VALUES (@adr, @num)
```

```
    SELECT SCOPE_IDENTITY()
```

```
    dbo.Add_client
```

```
ALTER PROCEDURE [dbo].[Add_client]
```

```
    @name nvarchar(50),
```

```
    @adress nvarchar(50),
```

```
    @phone int
```

```
AS
```

```
    INSERT INTO Clients(Name, Adress, Phone)
```

```
    VALUES (@name, @adress, @phone)
```

dbo.Delete_war

```
ALTER PROCEDURE [dbo].[Delete_war]
```

```
    @id int
```

```
AS
```

```
BEGIN
```

```
    Delete from Warehouses
```

```
    WHERE [Warehouses].Number = @id
```

```
END
```

dbo.Delete_client

```
ALTER PROCEDURE [dbo].[Delete_client]
```

					<i>BKP.090301.09.13 81 17</i>	Лист
						47
Изм.	Лист	№ докум.	Подпись	Дата		

```

@id int
AS
BEGIN
Delete from Clients
WHERE [Clients].ID_client = @id
END

```

					BKP.090301.09.13 81 17	Лист
						48
Изм.	Лист	№ докум.	Подпись	Дата		

ПРИЛОЖЕНИЕ В

Листинг программы

Класс Main.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Storage
{
    public partial class main : Form
    {
        public main()
        {
            InitializeComponent();
        }
        private void Form1_Load(object sender, EventArgs e)
        {

        }

        private void button1_Click(object sender, EventArgs e)
        {
            view Form = new view();
            Form.Show();
        }
        private void button2_Click(object sender, EventArgs e)
        {
            add form2 = new add();
            form2.Show();
        }
        private void button3_Click(object sender, EventArgs e)
        {
            add_new add_new = new add_new();
        }
    }
}
```

					ВКР.090301.09.13 81 17	Лист
Изм.	Лист	№ докум.	Подпись	Дата		49

```

        if
(Application.OpenForms.OfType<add_new>().Count()<2)
        {
            add_new.Show();
        }
    }

    private void button5_Click(object sender, EventArgs e)
    {
        if (MessageBox.Show("Уверены?", this.Text,
            MessageBoxButtons.YesNo,
MessageBoxIcon.Question) != DialogResult.No)
            this.Close();
    }

    private void button6_Click(object sender, EventArgs e)
    {
        Delete Form1 = new Delete();
        Form1.Show();
    }

}
}

```

Класс Add.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Storage
{
    public partial class add : Form
    {

```

					BKP.090301.09.13 81 17	Лист
						50
Изм.	Лист	№ докум.	Подпись	Дата		

```

public add()
{
    InitializeComponent();
}

private void button1_Click(object sender, EventArgs e)
{
    Supply form1 = new Supply();
    form1.Show();
}

private void button3_Click(object sender, EventArgs e)
{
    this.Close();
}

private void add_Load(object sender, EventArgs e)
{
}

private void button2_Click(object sender, EventArgs e)
{
    Removal form = new Removal();
    form.Show();
}
}
}

```

Класс Add_new.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

```

namespace Storage

					BKP.090301.09.13 81 17	Лист
						51
Изм.	Лист	№ докум.	Подпись	Дата		

```

{
    public partial class add_new : Form
    {
        public add_new()
        {
            InitializeComponent();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            new_client new_client = new new_client();
            new_client.Show();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            new_war new_war = new new_war();
            new_war.Show();
        }
    }
}

```

Класс New_clients.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace Storage
{

```

					BKP.090301.09.13 81 17	Лист
Изм.	Лист	№ докум.	Подпись	Дата		52

```

public partial class new_client : Form
{
    string connectionString = @"Data Source=USER-
ПК\SQLEXPRESS; Initial Catalog=Storage; Integrated
Security=SSPI;";
    int ph;
    string name;
    string address;
    public new_client()
    {
        InitializeComponent();

        private void button1_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void textBox1_TextChanged(object sender,
        EventArgs e)
        {
        }

        private void button3_Click(object sender, EventArgs e)
        {
            name = Convert.ToString(textBox1.Text);
            adress = Convert.ToString(textBox2.Text);
            ph = Convert.ToInt32(textBox3.Text);
            try
            {
                SqlConnection con = new SqlConnection();
                con.ConnectionString = connectionString;
                SqlCommand cmd = con.CreateCommand();
                cmd.CommandType = CommandType.StoredProcedure;
                cmd.CommandText = "[Add_client]";
                cmd.Parameters.Clear();
                cmd.Parameters.Add("@name",
                SqlDbType.VarChar).Value = name;
                cmd.Parameters.Add("@adress",
                SqlDbType.VarChar).Value = adress;
            }
            catch { }
        }
    }
}

```

					<i>BKP.090301.09.13 81 17</i>	Лист
						53
Изм.	Лист	№ докум.	Подпись	Дата		

```

        cmd.Parameters.Add("@phone",
SqlDbType.Int).Value = ph;
        con.Open();
        cmd.ExecuteNonQuery();
        con.Close();

    }

}

}
}

```

Класс New_war.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace Storage
{
    public partial class new_war : Form
    {
        string connectionString = @"Data Source=USER-
ПК\SQLEXPRESS; Initial Catalog=Storage; Integrated
Security=SSPI;";
        string place;
        int num;
        public new_war()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)

```

```

    {
        this.Close();
    }

    private void button2_Click(object sender, EventArgs e)
    {
        num = Convert.ToInt32(textBox1.Text);
        place = Convert.ToString(textBox2.Text);
        try
        {
            SqlConnection con = new SqlConnection();
            con.ConnectionString = connectionString;
            SqlCommand cmd = con.CreateCommand();
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.CommandText = "[Add_war]";
            cmd.Parameters.Clear();
            cmd.Parameters.Add("@num", SqlDbType.Int).Value
= num;
            cmd.Parameters.Add("@adr",
SqlDbType.VarChar).Value = place;

            con.Open();
            cmd.ExecuteNonQuery();
            con.Close();

        }
        catch { };

    }

    private void textBox1_TextChanged(object sender,
EventArgs e)
    {
    }

}

```