

**ГОСУДАРСТВЕННОЕ УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«ИНСТИТУТ БИЗНЕСА
БЕЛОРУССКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА»**

Кафедра цифровых систем и технологий

Курсовая работа

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ АПТЕКИ

Севрюк Александры Петровны
студентки 3 курса группы 852
специальности «Управление
информационными ресурсами»

Научный руководитель:
профессор, к.т.н.
Ю.Н. Силкович

Минск, 2021

**ГОСУДАРСТВЕННОЕ УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«ИНСТИТУТ БИЗНЕСА БЕЛОРУССКОГО ГОСУДАРСТВЕННОГО
УНИВЕРСИТЕТА»**

Кафедра цифровых систем и технологий

ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ

Студентка Севрюк Александра Петровна

Группа 852 УИР

1. Тема «Разработка информационной системы аптеки»
2. Срок представления курсовой работы к защите 10.05.2021
3. Исходные данные для научного исследования

Информационная система разрабатывается в архитектуре «клиент – сервер». В качестве средства для разработки серверной части приложения используется СУБД, поддерживающая реляционную модель данных. Разработанная база данных должна содержать активное содержимое в виде хранимых процедур, пользовательских функций и триггеров для выполнения задач основных бизнес-процессов в предметной области. Клиентская часть обеспечивает доступ пользователя к результатам обработки информации в базе данных.

4. Содержание курсовой работы

Реферат

Введение

1. Описание и анализ предметной области.

2. Разработка концептуальной модели БД (ER-диаграмма). Нормализация.

3. Разработка логической модели БД.

4. Создание серверной части приложения. Разработка запросов, триггеров, хранимых процедур.

5. Разработка клиентской части приложения.

Заключение

Список использованных источников

Приложения (при необходимости)

5. Документы, представляемые в электронном виде

1. Файлы серверной части БД, файлы скриптов создания основных объектов БД.

2. Файлы проекта клиентской части приложения.

3. Презентация

Руководитель курсовой работы _____ Ю.Н. Силкович 22.02.2021

Задание принял к исполнению _____ А.П. Севрюк 22.02.2021

РЕФЕРАТ

БАЗЫ ДАННЫХ, ИНФОРМАЦИОННАЯ СИСТЕМА, АПТЕКА, АВТОМАТИЗАЦИЯ, ЯЗЫК C#, ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ, MS SQL SERVER

Цель курсовой работы: закрепление, углубление, систематизация теоретических знаний, полученных в процессе изучения дисциплины «Системы баз данных», а также развитие практических навыков разработки автоматизированной информационной системы с помощью MS SQL Server и платформы .NET (язык C#).

Объект исследования: информационная система аптеки.

Предмет исследования: разработка автоматизированной информационной системы с помощью MS SQL Server и платформы .NET (язык C#).

Методы исследования: описательный, аналитический, практический (создание информационной системы).

Исследования и разработки: проведен анализ средств разработки интерфейса и сделан выбор СУБД, проведен анализ предметной области, рассмотрены методы работы с информацией (добавление, изменение, удаление), разработан программный код и интерфейс приложения.

Элементы научной новизны: разработана база данных в SQL Server.

Область возможного применения: рассмотренные методы и средства создания автоматизированной информационной системы могут быть использованы при создании информационной системы для различных сфер деятельности, разработанная информационная система может быть внедрена на аптечном предприятии.

Технико-экономическая, социальная значимость: разработанная информационная система поможет автоматизировать работу аптек.

Автор работы подтверждает, что приведенный в ней материал правильно и объективно отражает принципы создания автоматизированной информационной системы, а все заимствованные из литературных и других источников теоретические, методологические и методические положения сопровождаются ссылками на их авторов.

(подпись студента)

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ	5
ГЛАВА 1 ВЫБОР ПРЕДМЕТНОЙ ОБЛАСТИ И СРЕДСТВ РАЗРАБОТКИ..	6
1.1 Выбор предметной области.	6
1.2 Выбор средств разработки.	7
ГЛАВА 2 ПРОЕКТИРОВАНИЕ РАЗРАБАТЫВАЕМОЙ БАЗЫ ДАННЫХ .	10
2.1 Концептуальное проектирование.	10
2.2 Логическое и физическое проектирование.	12
ГЛАВА 3 РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ.....	15
3.1 Разработка запросов для создания БД, таблиц и заполнения их данными.....	15
3.2 Разработка запросов манипулирования данными.	16
3.3 Разработка запросов на выборку данных, создание представлений.	18
3.4 Разработка триггеров, пользовательских функций и хранимых процедур.....	22
3.5 Администрирование базы данных.	29
3.6 Разработка интерфейса приложения.....	31
ЗАКЛЮЧЕНИЕ	37
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	39

ВВЕДЕНИЕ

Трудно представить себе более благоприятную почву для внедрения новых компьютерных технологий, чем торговля. Многие задачи, которые возникают в ходе работы торговых предприятий, достаточно легко поддаются автоматизации. В современном обществе своевременная обработка информации способствует совершенствованию организации производства, оперативному и долгосрочному планированию, прогнозированию и анализу деятельности, что позволяет компаниям успешно конкурировать на рынке. Все это стало причиной появления большого ряда программ, средств и инструментов для их создания. Наиболее эффективно, быстро и качественно минимизировать затраты времени, материальных, трудовых ресурсов и упростить процесс обработки информации можно с помощью автоматизированных информационных систем. Это и обуславливает **актуальность** разработки автоматизированной информационной системы аптечного предприятия.

Объектом исследования данной курсовой работы является предприятие аптечной сети.

Предмет исследования – разработка информационной системы аптеки с помощью языка Transact-SQL и объектно-ориентированного языка программирования C#.

Цель курсовой работы – закрепление, углубление, систематизация теоретических знаний, полученных в процессе изучения дисциплины «Системы баз данных», а также развитие практических навыков разработки автоматизированной информационной системы с помощью MS SQL Server и платформы .NET (язык C#).

Для реализации поставленной цели были определены следующие **задачи**:

1. Описание и анализ предметной области;
2. Разработка концептуальной и логической моделей БД;
3. Создание базы данных в MS SQL Server;
4. Создание серверной части приложения;
5. Разработка клиентской части приложения.

При написании курсовой работы были использованы следующие **методы** исследования: описательный, аналитический, практический (создание информационной системы).

В процессе написания данной работы использовалась литература отечественных и зарубежных авторов, а также информация из сети Интернет.

ГЛАВА 1

ВЫБОР ПРЕДМЕТНОЙ ОБЛАСТИ И СРЕДСТВ РАЗРАБОТКИ

1.1 Выбор предметной области.

Перед началом разработки базы данных был проведен **анализ предметной области** разрабатываемой информационной системы, а именно предприятия аптечной сети. Были выделены **две группа пользователей** – сотрудники и клиенты аптеки.

Также был проведен анализ имеющихся проблем, а именно комплексный анализ функционирования аптеки, в результате которого был выявлен **ряд недостатков в функционировании**:

1. трудоемкий процесс оприходования товара;
2. ведение документации вручную;
3. высокая трудоемкость работы персонала аптеки с большим товарным ассортиментом;
4. сложности с учетом складских остатков;
5. отсутствие полноценного и детального учета движения товаров и финансовых средств на любой момент времени.

На основании этих недостатков в работе аптечного предприятия было принято решение о разработке и внедрении автоматизированной информационной системы.

Были выделены следующие **функциональные требования** [9]:

1. оперативное хранение и поиск информации о товарах и их продажах;
2. учет информации о поступлении в аптеку лекарственных средств от различных контрагентов, их хранении и поступлении в продажу;
3. расчет количества проданных товаров и выручки от их реализации;
4. предоставление возможности оценки эффективности работы предприятия;
5. эргономичный интерфейс.

Также в результате проведенного анализа было выделено **семь объектов** – оптовые заказы, товары, продажи, поставки, контрагенты, страны, места хранения.

В итоге было определено, что **целью** разрабатываемой информационной системы является автоматизация процессов сбора, обработки, хранения и анализа информации о продажах, поставках, контрагентах и местах хранения товаров, реализуемых в аптеке.

1.2 Выбор средств разработки.

В качестве средств разработки информационных систем будут рассмотрены следующие системы управления базами данных (СУБД): Oracle Database, Microsoft SQL Server и Microsoft Access [10]. Выбор этих СУБД связан с тем, что они наиболее распространены в настоящее время.

Oracle Database — СУБД, работающая с огромными объемами данных, обеспечивающая высокую производительность, масштабируемость, информационную безопасность и самоуправляемость. Применяется в корпоративных сетях распределенной обработки данных [6].

Microsoft SQL Server — мощная СУБД с поддержкой клиент-серверной архитектуры, позволяющая осуществлять тиражирование и параллельную обработку данных, обеспечивающая поддержку крупных БД на бюджетных компьютерах с поддержкой несмежного управления. Данная СУБД имеет гибкость и возможности масштабирования. БД, разработанные с использованием данного инструментального средства, можно эффективно использовать как на малых предприятиях в масштабах одного компьютера, так и в крупных организациях, где требуется поддержка больших БД [6].

Microsoft Office Access — одна из самых производительных, простых при использовании и гибких при настройке СУБД, поддерживающая работу нескольких пользователей, позволяющая хранить и обрабатывать данные, проверять правильность вводимых данных, разрабатывать формы для удобной работы с базой данных, создавать отчеты. Ms Access это полная в функциональном отношении СУБД, имеющая встроенный язык программирования Visual Basic Application [6].

В таблице 1 приведена сравнительная характеристика данных СУБД [6].
Таблица 1 – Сравнительная характеристика СУБД

Признак сравнения	Oracle Database	Microsoft SQL Server	Microsoft Access
Требования к аппаратному обеспечению	Процессор (минимум) Pentium 166; ОП не ниже 128 МВ; место на диске 1 Гб	Процессор (минимум) 600 МГц; ОП (минимум) 192Мб; место на диске 600 МВ	Процессор Pentium III (минимум) 233 МГц; ОП (минимум) 128 МБ; место на диске 400 МБ
Поддерживаемая модель данных	Универсальная модель данных	Реляционная модель данных	Реляционная модель данных
Поддерживаемые объекты БД	Таблицы, представления,	Таблицы, представления,	Таблицы, запрос, форма,

Окончание таблицы 1

	пользователь, последовательность, индекс, табличная область, кластер, роль, процедура, функция, пакет, триггер.	пользователь, индекс, процедура, функция, правила, ограничения, триггер.	отчет, макрос, модуль.
Технология создания БД и объектов БД	Визуально, с использованием SQL-скриптов	Визуально, с использованием SQL-скриптов	Визуально
Поддержка сервера БД	+	+	-
Средства поддержки ограничений целостности БД	Ключи, установление связей между таблицами, ограничения, триггер	Ключи, ограничения, триггер, транзакции	Ключи, установление связей между таблицами
Язык запросов	PL/SQL и ANSI SQL	T-SQL	SQL
Импорт и экспорт таблиц базы данных	+	-	+
Наличие встроенного языка для разработки приложений	PL/SQL	Элементы Microsoft Visual Basic for Application	Microsoft Visual Basic for Application
Функционирует под управлением операционных систем	Windows, Unix, Linux, MacOS	Windows	Windows
Простота/ сложность работы с СУБД	Сложно	Требуется определенных навыков	Просто
Поддержка многопользовательского режима	+	+	+
Кроссплатформенность	+	+	-

Из рассмотренных СУБД в качестве основного инструмента для создания информационной системы была выбрана СУБД Microsoft SQL Server, так как она обладает всеми средствами для создания и обеспечения работоспособности базы данных, поддерживает визуальную технологию создания объектов базы данных, стандарт языка SQL.

ГЛАВА 2

ПРОЕКТИРОВАНИЕ РАЗРАБАТЫВАЕМОЙ БАЗЫ ДАННЫХ

2.1 Концептуальное проектирование.

Целью концептуального проектирования является создание концептуальной модели данных исходя из представлений пользователей о предметной области.

Первым этапом разработки концептуальной схемы данных является **определение объектов**, которые существуют независимо от других. Такие объекты являются сущностями. Каждой сущности присвоим осмысленное имя, понятное пользователям. В результате проведенного анализа в главе 1 было выделено семь объектов: оптовые заказы, товары, продажи, поставки, контрагенты, страны, места хранения.

Далее мы определяем **связи между сущностями**. Определять будем только те связи между сущностями, которые необходимы для удовлетворения требований, определенных в первой главе к проекту базы данных. Связям присваиваем осмысленные имена, выраженные глаголами. В итоге мы строим концептуальную (инфологическую) схему данных в нотации Питера Чена (рис. 2.1).

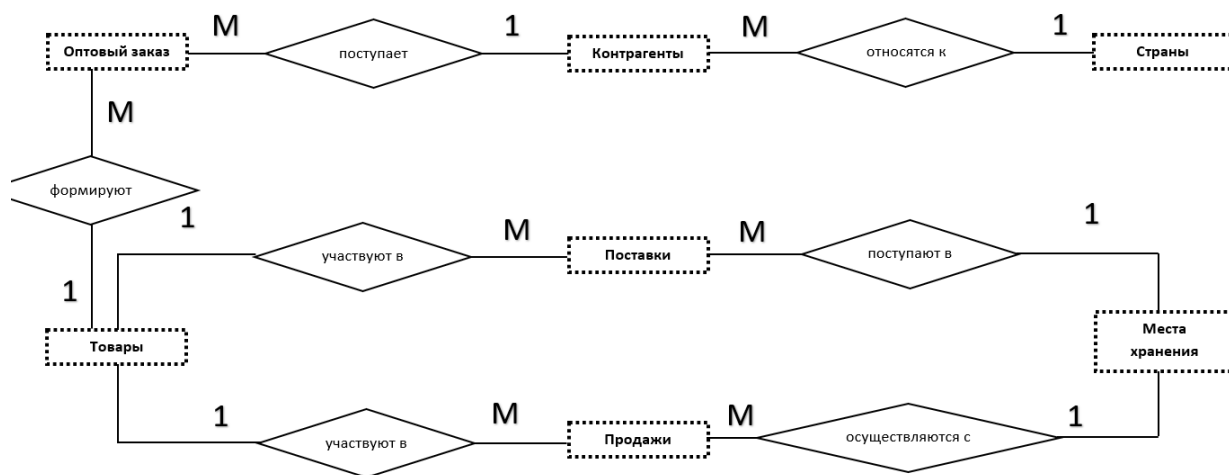


Рисунок 2.1 – Концептуальная схема данных в нотации Питера Чена

Следующий этап – это **определение атрибутов объектов**. На этом этапе мы выявляем все атрибуты, описывающие сущности, каждому атрибуту присваиваем осмысленное имя, понятное пользователю, задаем его тип и размерность значений, а также определяем первичные и внешние ключи для сущностей (таблица 2).

Таблица 2 – Описание атрибутов объектов

Имя атрибута	Тип атрибута и размерность значений	Первичные и внешние ключи
Места хранения		
Код склада	(целое число)	первичный ключ
Наименование	(строка, короткий текст: 50)	
Расположение	(строка, короткий текст: 100)	
ФИО материально ответственного лица	(строка, короткий текст: 70)	
Телефон	(строка, короткий текст: 15)	
Товары		
Код товара	(целое число)	первичный ключ
Артикул	(целое число)	
Наименование	(строка, короткий текст: 50)	
Единицы измерения	(строка, короткий текст: 10)	
Цена	(целое число, денежный, >0, рубли)	
Условия отпуска	(строка, короткий текст: 20)	
Контрагенты		
Код контрагента	(целое число)	первичный ключ
Наименование	(строка, короткий текст: 50)	
Страна	(целое число)	внешний ключ
Адрес	(строка, короткий текст: 50)	
Телефон	(строка, короткий текст: 15)	
Электронная почта	(строка, короткий текст: 30)	
ФИО контрагента	(строка, короткий текст: 50)	
Страны		
Код страны	(целое число)	первичный ключ
Название	(строка, короткий текст: 50)	
Рейтинг	(целое число, >0, <10)	
Поставки		
Код накладной	(целое число)	первичный ключ
Дата поставки	(дата и время, краткий формат даты)	
Контрагент	(строка, короткий текст: 50)	
Количество	(число, длинное целое)	
Цена	(целое число, денежный, >0, рубли)	

Окончание таблицы 2

Код склада	(целое число)	внешний ключ
Код товара	(целое число)	внешний ключ
Продажи		
Код накладной	(целое число)	первичный ключ
Дата продажи	(дата и время, краткий формат даты)	
Контрагент	(строка, короткий текст: 50)	
Количество	(число, длинное целое)	
Цена	(целое число, денежный, >0, рубли)	
Код склада	(целое число)	внешний ключ
Код товара	(целое число)	внешний ключ
Оптовый заказ		
Код заказа	(целое число)	первичный ключ
Дата заказа	(дата и время, краткий формат даты)	
Количество	(число, длинное целое)	
Код товара	(целое число)	внешний ключ
Код контрагента	(целое число)	внешний ключ

И последним шагом на этапе концептуального проектирования будет обсуждение концептуальной модели с конечными пользователями. Если будут обнаружены несоответствия предметной области, то в модель вносятся изменения.

2.2 Логическое и физическое проектирование.

Цель этапа логического проектирования – преобразование концептуальной модели на основе выбранной модели данных в логическую модель, не зависящую от особенностей используемой в дальнейшем СУБД для физической реализации базы данных.

Для начала мы выберем модель данных, в нашем случае это будет реляционная модель, в связи с наглядностью табличного представления данных и удобства работы с ними [4].

Далее мы определим и создадим набор таблиц для каждой сущности ER-модели. Установим связи между ними (рис. 2.2) посредством механизма первичных и внешних ключей.

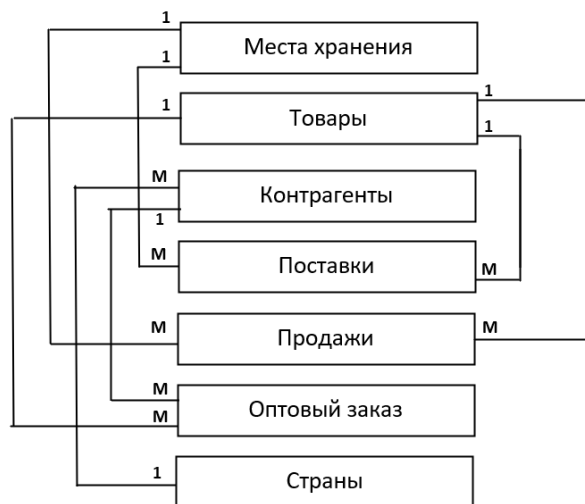


Рисунок 1.2 – Типы сущностей и связи между ними

Следующим шагом выполним нормализацию таблиц, т.е. проверим корректность структуры таблиц, созданных на предыдущем шаге, посредством применения к ним процедуры нормализации, т.е. приведем каждую из таблиц к 3НФ. В результате нормализации будет получен гибкий проект базы данных, позволяющий легко вносить в нее нужные расширения.

Далее выполняем проверку логической модели данных на предмет возможности выполнения всех предусмотренных транзакций. Используя ER-модель и установленные связи между первичными и внешними ключами, производим попытку выполнить все необходимые операции доступа к данным вручную. В результате проверки мы видим, что все операции успешно выполняются, значит составленная логическая модель является адекватной и не содержит ошибок.

В итоге окончательный вариант ER-модели данных, представляющей логическую модель данных будет выглядеть как показано на рисунке 2.3.

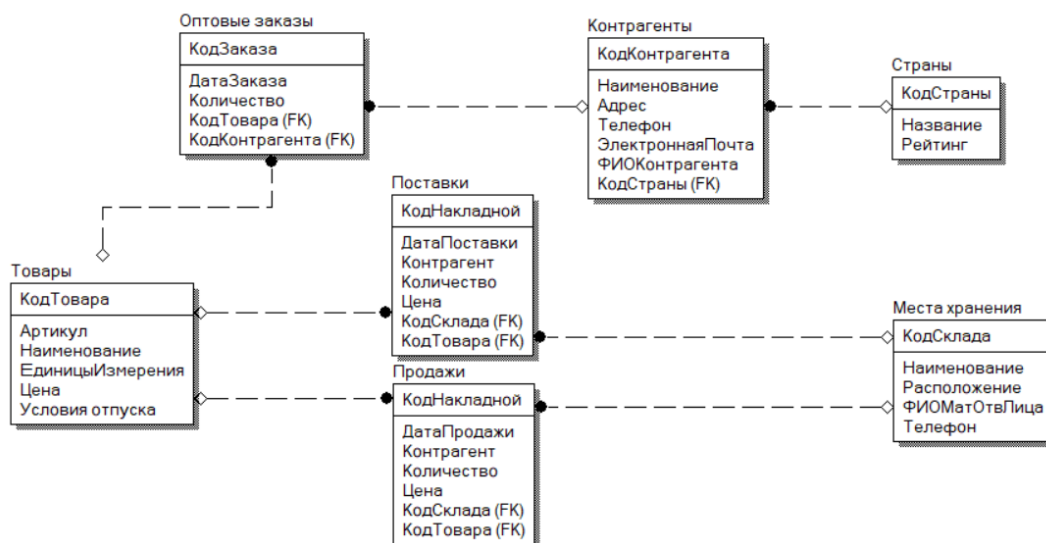


Рисунок 2.3 – Логическая ER-модель БД аптеки

Также мы можем средствами ERWin Data Modeler преобразовать логическую схему данных в физическую (рис. 2.4) [11].

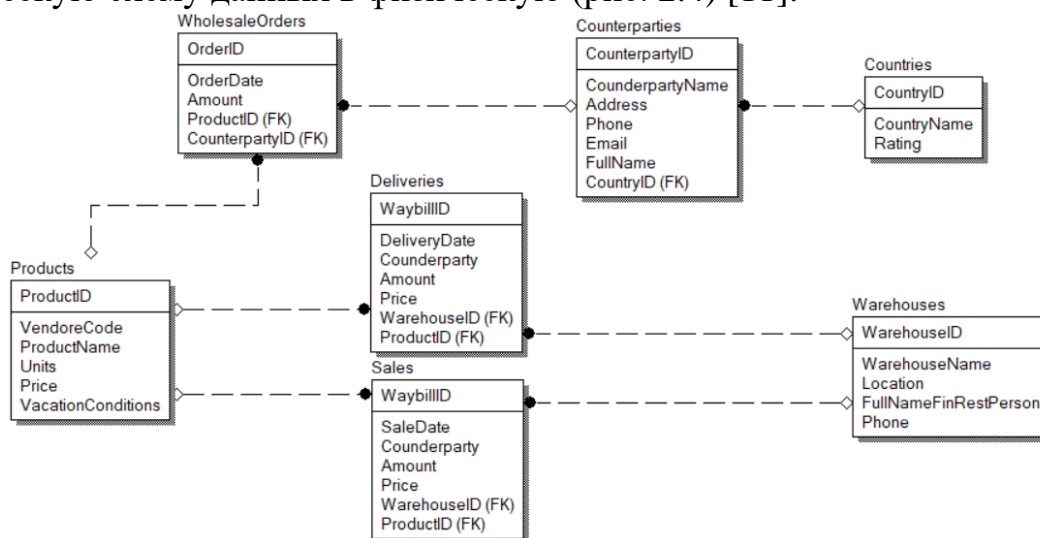


Рисунок 2.4 – Физическая ER-модель БД аптеки

Цель этапа физического проектирования – описание конкретной реализации базы данных, размещаемой во внешней памяти компьютера. Это описание структуры хранения данных и эффективных методов доступа к данным базы.

Выразим полученную в ERWin Data Modeler ER-диаграмму в терминах выбранной в первой главе СУБД, а именно MS SQL Server (рис. 2.5) [2].

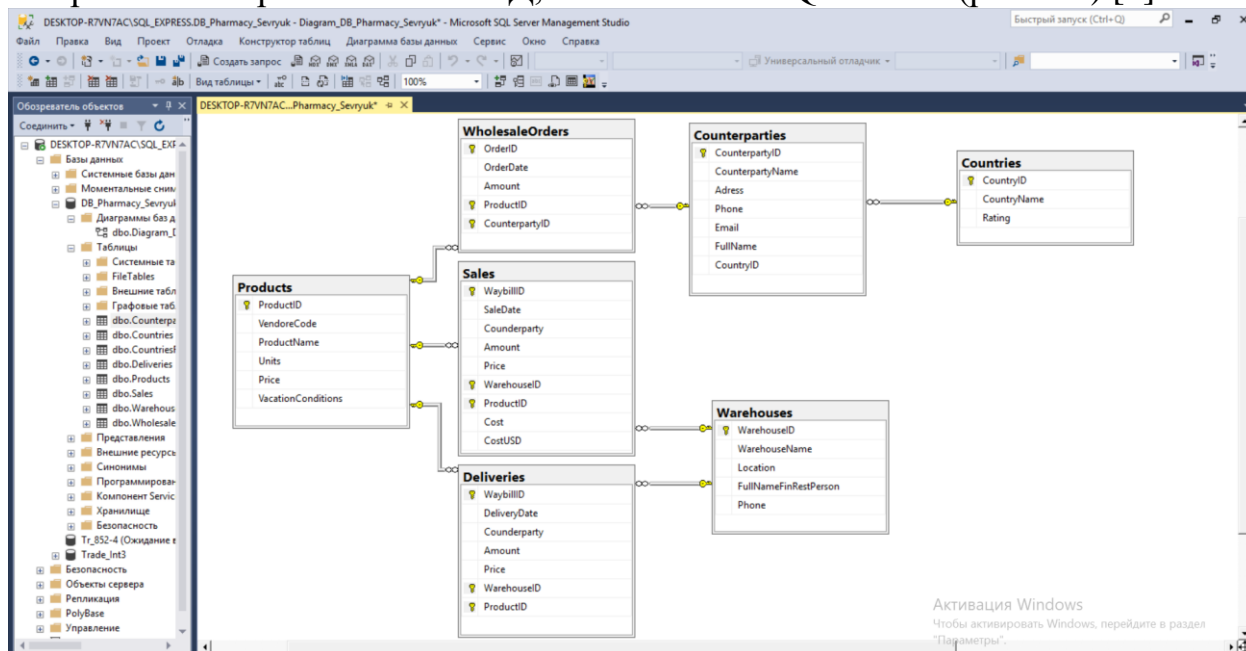


Рисунок 2.2 – Модель БД аптеки в MS SQL Server

ГЛАВА 3

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ

3.1 Разработка запросов для создания БД, таблиц и заполнения их данными.

Первым шагом в разработке информационной системы на языке SQL будет создание запроса для создания базы данных [1].

```
CREATE DATABASE DB_Pharmacy_Sevryuk
ON PRIMARY
    (NAME = DB_Pharmacy_Sevryuk,
    FILENAME =
'E:\first\SUBD\Course_work_Sevryuk\DB_Pharmacy_Sevryuk.mdf',
    SIZE = 5120KB,
    MAXSIZE = UNLIMITED,
    FILEGROWTH = 1024KB)
LOG ON
    (NAME = DB_Pharmacy_Sevryuk_log,
    FILENAME =
'E:\first\SUBD\Course_work_Sevryuk\DB_Pharmacy_Sevryuk.ldf',
    SIZE = 2048KB,
    MAXSIZE = 2048GB,
    FILEGROWTH = 10%)
GO
```

Далее были разработаны запросы для создания всех таблиц (с учетом ограничений целостности данных, ограничений на значения, значений по умолчанию и т.п.). В качестве примера приведен код запроса для создания таблицы «Продажи» [1].

```
CREATE TABLE Sales(
    WaybillID int NOT NULL,
    SaleDate date NOT NULL,
    Counterparty varchar (50) NOT NULL,
    Amount int NOT NULL CHECK (Amount>=0),
    Price money NOT NULL CHECK (Price>=0),
    WarehouseID int NOT NULL,
    ProductID int NOT NULL,
    PRIMARY KEY (
        WaybillID ASC,
        WarehouseID ASC,
        ProductID ASC),
    CONSTRAINT FK_Sales_Warehouses FOREIGN KEY (WarehouseID)
        REFERENCES dbo.Warehouses(WarehouseID)
    ON UPDATE CASCADE,
    CONSTRAINT FK_Sales_Products FOREIGN KEY (ProductID)
        REFERENCES dbo.Products(ProductID)
    ON UPDATE CASCADE
)
```

Были разработаны запросы для создания индексов для таблиц базы данных [1].

```
CREATE UNIQUE INDEX UIX_Products ON Products(ProductName)
```

```

CREATE UNIQUE INDEX UIX_Counterparties ON
Counterparties (CounterpartyName)
CREATE UNIQUE INDEX UIX_Countries ON Countries (CountryName)
CREATE UNIQUE INDEX UIX_Warehouses ON Warehouses (WarehouseName)
CREATE INDEX IX_Deliveries ON Deliveries (DeliveryDate,
Counterparty)
CREATE INDEX IX_Sales ON Sales (SaleDate, Counterparty)
CREATE INDEX IX_WholesaleOrders ON WholesaleOrders (OrderDate)

```

Следующим шагом были разработаны запросы на языке SQL для заполнения созданных таблиц данными. Все запросы однотипные, в качестве примера приведен код запроса для заполнения данными таблицы «Товары» [1].

```

INSERT INTO Products
VALUES
('1', '123', 'Парацетамол', 'Упаковка', '5.96', 'Без
рецепта'),
('2', '124', 'Анальгин', 'Упаковка', '1.24', 'Без рецепта'),
('3', '125', 'Зинерит', 'Упаковка', '32.64', 'По рецепту'),
('4', '126', 'Бепантен', 'Тюбик', '15.42', 'Без рецепта'),
('5', '127', 'Лизак', 'Упаковка', '4.81', 'Без рецепта'),
('6', '128', 'Антигриппин', 'Упаковка', '12.35', 'Без
рецепта')

```

Результат выполнения запроса представлен на рисунке 3.1.

ProductID	VendoreCode	ProductName	Units	Price	VacationCondi...
1	123	Парацетамол	Упаковка	5,9600	Без рецепта
2	124	Анальгин	Упаковка	1,2400	Без рецепта
3	125	Зинерит	Упаковка	32,6400	По рецепту
4	126	Бепантен	Тюбик	15,4200	Без рецепта
5	127	Лизак	Упаковка	4,8100	Без рецепта
6	128	Антигриппин	Упаковка	12,3500	Без рецепта
NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 3.1 – Заполненная данными таблица "Товары"

3.2 Разработка запросов манипулирования данными.

В качестве запроса на манипулирование данными был создан запрос, с помощью которого создается новая таблица «CounterpartiesWithCountries» путем объединения двух таблиц. В нашем случае происходит объединение таблиц «Контрагенты» и «Страны» по равенству «CountryID». В результирующую таблицу выводятся столбцы «Код контрагента», «Наименование контрагента», «Адрес», «Страна», «Рейтинг страны». Запрос создается с целью удобства одновременного просмотра контрагентов и их стран. Результат выполнения запроса представлен на рисунке 3.2.

```

SELECT C2.CounterpartyID, C2.CounterpartyName, C2.Adress,
C1.CountryName, C1.Rating
INTO CounterpartiesWithCountries
FROM Countries C1
LEFT JOIN Counterparties C2 ON C1.CountryID =
C2.CountryID

```


	CounterpartyID	Counterparty...	Adress	CountryName	Rating
▶	1	ООО "ФармВи...	г. Минск, ул. Т...	Беларусь	8
	2	ООО "Белфар...	г. Брест, ул. Со...	Беларусь	8
	4	ООО "Медикф...	г. Москва, ул. ...	Россия	7
	5	ООО "ФармМе...	г. Варшава, ул....	Польша	9
	6	ООО "ЧехМед...	г. Прага, ул. Че...	Чехия	9
	3	ООО "Фармекс"	г. Дрезден, ул. ...	Германия	10
*	NULL	NULL	NULL	NULL	NULL

Рисунок 3.2 – Результат выполнения запроса на создание таблицы

Далее была изменена структура созданной таблицы путем добавления в нее нового столбца с телефоном и заполнения его данными, взятыми из таблицы «Контрагенты», с помощью запроса на обновление. Результат выполнения запроса представлен на рисунке 3.3.

```
ALTER TABLE CounterpartiesWithCountries ADD Phone char(15) NULL
UPDATE CounterpartiesWithCountries
SET CounterpartiesWithCountries.Phone = (
    SELECT Counterparties.phone
    FROM Counterparties
    WHERE Counterparties.CounterpartyID =
CounterpartiesWithCountries.CounterpartyID)
```

	CounterpartyID	Counterparty...	Adress	CountryName	Rating	Phone
▶	1	ООО "ФармВи...	г. Минск, ул. Т...	Беларусь	8	+375291104294
	2	ООО "Белфар...	г. Брест, ул. Со...	Беларусь	8	+375291234567
	4	ООО "Медикф...	г. Москва, ул. ...	Россия	7	+79063514862
	5	ООО "ФармМе...	г. Варшава, ул....	Польша	9	+5695126515
	6	ООО "ЧехМед...	г. Прага, ул. Че...	Чехия	9	+75912364856
	3	ООО "Фармекс"	г. Дрезден, ул. ...	Германия	10	+789265894657
*	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 3.3 – Результат выполнения запроса на обновление данных

Был создан запрос на добавление в таблицу «CounterpartiesWithCountries» новых записей на тот случай, если у аптеки появились новые контрагенты и новые страны поставок. Результат выполнения запроса представлен на рисунке 3.4.

```
INSERT INTO CounterpartiesWithCountries
(CounterpartyID, CounterpartyName, Adress,
CountryName, Rating, Phone)
VALUES
(7, 'ООО "МедикалФранс"', 'г. Париж, ул. Французская,
д. 105', 'Франция', 8, '+33142967000'),
(8, 'ООО "МедикОмскФармэйшн"', 'г. Омск, ул.
Революционная, д. 1', 'Россия', 7, '+79253615859'),
(9, 'ООО "БританМед"', 'г. Манчестер, ул. Кингс Роуд,
д. 18', 'Великобритания', 10, '+44232567805')
```

	CounterpartyID	Counterparty...	Adress	CountryName	Rating	Phone
▶	1	ООО "ФармВи...	г. Минск, ул. Т...	Беларусь	9	+375291104294
	2	ООО "Белфар...	г. Брест, ул. Со...	Беларусь	9	+375291234567
	4	ООО "Медикф...	г. Москва, ул. ...	Россия	7	+79063514862
	5	ООО "ФармМе...	г. Варшава, ул....	Польша	9	+5695126515
	6	ООО "ЧехМед...	г. Прага, ул. Че...	Чехия	9	+75912364856
	3	ООО "Фармекс"	г. Дрезден, ул. ...	Германия	10	+789265894657
	7	ООО "Медика...	г. Париж, ул. Ф...	Франция	8	+33142967000
	8	ООО "МедикО...	г. Омск, ул. Ре...	Россия	7	+79253615859
	9	ООО "Британ...	г. Манчестер, ...	Великобритания	10	+44232567805
*	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 3.4 – Результат выполнения запроса на добавление данных

Был создан запрос на удаление записей из таблицы «CounterpartiesWithCountries» на тот случай, если аптека, например, перестала сотрудничать с Россией, поэтому нужно удалить из таблицы все записи, связанные с этой страной. Результат выполнения запроса представлен на рисунке 3.5.

```
DELETE FROM CounterpartiesWithCountries
WHERE CountryName = 'Россия'
```

	CounterpartyID	Counterparty...	Adress	CountryName	Rating	Phone
▶	1	ООО "ФармВи...	г. Минск, ул. Т...	Беларусь	9	+375291104294
	2	ООО "Белфар...	г. Брест, ул. Со...	Беларусь	9	+375291234567
	5	ООО "ФармМе...	г. Варшава, ул....	Польша	9	+5695126515
	6	ООО "ЧехМед...	г. Прага, ул. Че...	Чехия	9	+75912364856
	3	ООО "Фармекс"	г. Дрезден, ул. ...	Германия	10	+789265894657
	7	ООО "Медика...	г. Париж, ул. Ф...	Франция	8	+33142967000
	9	ООО "Британ...	г. Манчестер, ...	Великобритания	10	+44232567805
*	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 3.5 – Результат выполнения запроса на удаление данных

3.3 Разработка запросов на выборку данных, создание представлений.

Был создан запрос, отображающий все поля таблицы «Товары» с сортировкой товаров по условию продажи "без рецепта" и по уменьшению цены. Результат выполнения запроса представлен на рисунке 3.6.

```
SELECT *
FROM Products
WHERE VacationConditions = 'Без рецепта'
ORDER BY Price DESC
```

Результаты		Сообщения				
	ProductID	VendoreCode	ProductName	Units	Price	VacationConditions
1	4	126	Бепантен	Тюбик	15,42	Без рецепта
2	6	128	Антигриппин	Упаковка	12,35	Без рецепта
3	1	123	Парацетамол	Упаковка	5,96	Без рецепта
4	5	127	Лизак	Упаковка	4,81	Без рецепта
5	2	124	Анальгин	Упаковка	1,24	Без рецепта

Рисунок 3.6 – Результат выполнения запроса с сортировкой данных

Был создан запрос, который выполняет группировку дат поставок товаров и сортирует их в порядке убывания. Это может быть полезно в том случае, когда необходимо проверить, в какие дни обычно осуществляются поставки товара. Результат выполнения запроса представлен на рисунке 3.7.

```
SELECT DeliveryDate [Даты поставок товара]
FROM Deliveries
GROUP BY DeliveryDate
ORDER BY [Даты поставок товара] DESC
```

Результаты		Сообщения					
	WaybillID	DeliveryDate	Counterparty	Amount	Price	WarehouseID	ProductID
1	1	2020-11-10	1	50	1,24	1	2
2	2	2020-11-10	3	45	15,42	2	4
3	3	2020-11-11	2	35	5,96	3	1
4	4	2020-11-11	4	25	32,64	4	3
5	5	2020-11-12	5	20	12,35	5	6
6	6	2020-11-13	6	65	4,81	6	5
7	7	2020-11-13	1	64	32,64	2	3
8	8	2020-11-13	2	40	5,96	4	1
9	9	2020-11-14	3	45	15,42	5	4
10	10	2020-11-14	2	30	1,24	1	2

Даты поставок товара	
1	2020-11-14
2	2020-11-13
3	2020-11-12
4	2020-11-11
5	2020-11-10

Рисунок 3.7 – Результат выполнения запроса с группировкой

Одним из интересных запросов является запрос, позволяющий сгруппировать данные из таблиц «Товары» и «Продажи» по названию товара и вывести суммарное количество и итоговую стоимость проданного товара. Затем производится сортировка по столбцу «Суммарное количество проданного товара» по убыванию, это необходимо для того, чтобы определить самый покупаемый товар. Результат выполнения запроса представлен на рисунке 3.8.

```

SELECT P.ProductName [Название проданного товара],
SUM(S.Amount) [Суммарное количество проданного товара],
SUM(S.Price * S.Amount) [Общая стоимость проданного товара]
FROM Sales S, Products P
WHERE S.ProductID = P.ProductID
GROUP BY P.ProductName
ORDER BY [Суммарное количество проданного товара]
DESC

```

	Название проданного товара	Суммарное количество проданного товара	Общая стоимость проданного товара
1	Лизак	2720	10466,56
2	Анальгин	370	367,04
3	Бепантен	310	3824,16
4	Зинерит	250	6528,00
5	Парацетамол	170	810,56
6	Антигриппин	160	1580,80

Рисунок 3.8 – Результат выполнения запроса с группировкой с расчетным полем

Был разработан запрос, результирующая таблица которого соответствует по структуре записям дочерней таблицы «Оптовые заказы», но коды в полях внешнего ключа заменены соответствующими им наименованиями из родительских таблиц «Товары» и «Контрагенты». Это облегчает понимание и чтение пользователю. В таблицу добавлен столбец с «Ценой» и расчетный столбец «Стоимость», выполнена сортировка по убыванию по столбцам «Стоимость» и «Дата Заказа». Результат выполнения запроса представлен на рисунке 3.9.

```

SELECT W.OrderID, W.OrderDate, W.Amount, P.Price, W.Amount *
P.Price AS Cost, P.ProductName, C.CounterpartyName
FROM WholesaleOrders W, Products P, Counterparties C
WHERE W.ProductID = P.ProductID AND W.CounterpartyID =
C.CounterpartyID
ORDER BY Cost DESC, OrderDate
DESC

```

	OrderID	OrderDate	Amount	Price	Cost	ProductName	CounterpartyName
1	6	2020-11-06	65	15,42	1002,30	Бепантен	ООО "Фармекс"
2	9	2020-11-08	25	32,64	816,00	Зинерит	ООО "ЧехМедПрепарат"
3	4	2020-11-04	15	15,42	231,30	Бепантен	ООО "Белфармация"
4	3	2020-11-03	45	4,81	216,45	Лизак	ООО "ЧехМедПрепарат"
5	8	2020-11-08	20	5,96	119,20	Парацетамол	ООО "Белфармация"
6	1	2020-11-01	20	5,96	119,20	Парацетамол	ООО "ФармВитекс"
7	2	2020-11-17	50	1,24	62,00	Анальгин	ООО "Фармекс"

Рисунок 3.9 – Результат выполнения запроса с заменой кодов

Был разработан запрос для создания представления на основе предыдущего запроса на выборку с заменой кодов.

```

CREATE VIEW FullWholesaleOrders AS
SELECT W.OrderID, W.OrderDate, W.Amount, P.ProductName,
C.CounterpartyName
FROM WholesaleOrders W, Products P, Counterparties C
WHERE W.ProductID = P.ProductID AND W.CounterpartyID =
C.CounterpartyID

```

На основе предыдущего представления было создано новое, где добавлен столбец с «Ценой» и расчетный столбец «Стоимость», по которому выполнена сортировка по убыванию. Результат выполнения запроса представлен на рисунке 3.10.

```

CREATE VIEW FullWholesaleOrdersCost AS
SELECT W.OrderID, W.OrderDate, W.Amount, P.Price, W.Amount *
P.Price AS Cost, P.ProductName, C.CounterpartyName
FROM WholesaleOrders W, Products P, Counterparties C
WHERE W.ProductID = P.ProductID AND
W.CounterpartyID = C.CounterpartyID
SELECT *
FROM FullWholesaleOrdersCost
ORDER BY Cost DESC

```

	OrderID	OrderDate	Amount	Price	Cost	ProductName	CounterpartyName
1	6	2020-11-06	65	15,42	1002,30	Бепантен	ООО "Фармекс"
2	9	2020-11-08	25	32,64	816,00	Зинерит	ООО "ЧехМедПрепарат"
3	4	2020-11-04	15	15,42	231,30	Бепантен	ООО "Белфармация"
4	3	2020-11-03	45	4,81	216,45	Лизак	ООО "ЧехМедПрепарат"
5	1	2020-11-01	20	5,96	119,20	Парацетамол	ООО "ФармВитекс"
6	8	2020-11-08	20	5,96	119,20	Парацетамол	ООО "Белфармация"
7	2	2020-11-17	50	1,24	62,00	Анальгин	ООО "Фармекс"

Рисунок 3.10 – Результат выполнения запроса для создания представления

Был разработан запрос, позволяющий объединить таблицы «Оптовые заказы» и «Контрагенты», он необходим для удобного просмотра всех данных по каждому оптовому заказу. Результат выполнения запроса представлен на рисунке 3.11.

```

SELECT W.OrderID, W.OrderDate, W.ProductID, C.CounterpartyName,
C.Adress, C.FullName, C.Phone, C.Email
FROM WholesaleOrders W
FULL JOIN Counterparties C
ON W.CounterpartyID = C.CounterpartyID

```

	OrderID	OrderDate	ProductID	CounterpartyName	Adress	FullName	Phone	Email
1	1	2020-11-01	1	ООО "ФармВитекс"	г. Минск, ул. Тимирязева, 54	Иванов Петр Владимирович	+375291104294	FamViteks@mail.ru
2	3	2020-11-03	5	ООО "ЧехМедПрепарат"	г. Прага, ул. Чешская, 7	Августовская Ирина Владимировна	+75912364856	zechmedpreparat@mail.ru
3	4	2020-11-04	4	ООО "Белфармация"	г. Брест, ул. Солнечная, 65	Борисова Ангелина Дмитриевна	+375291234567	belfarmcia@mail.ru
4	6	2020-11-06	4	ООО "Фармекс"	г. Дрезден, ул. Несецкая, 45	Федоров Владимир Вечеславович	+789265894657	fameks@mail.ru
5	8	2020-11-08	1	ООО "Белфармация"	г. Брест, ул. Солнечная, 65	Борисова Ангелина Дмитриевна	+375291234567	belfarmcia@mail.ru
6	9	2020-11-08	3	ООО "ЧехМедПрепарат"	г. Прага, ул. Чешская, 7	Августовская Ирина Владимировна	+75912364856	zechmedpreparat@mail.ru
7	2	2020-11-17	2	ООО "Фармекс"	г. Дрезден, ул. Несецкая, 45	Федоров Владимир Вечеславович	+789265894657	fameks@mail.ru

Рисунок 3.11 – Результат выполнения запроса на объединение таблиц

Был создан запрос с двумя подзапросами и многострочным оператором сравнения, в результате выполнения которого выводятся не просто номера оптовых заказов из конкретной страны, в данном случае из Германии, но и всю информацию по оптовому заказу (дата, код продукта, количество, код контрагента). Результат выполнения запроса представлен на рисунке 3.12.

```
SELECT * FROM WholesaleOrders
        WHERE CounterpartyID IN
              (SELECT C1.CounterpartyID
               FROM Counterparties C1
               WHERE CountryID IN
                     (SELECT C.CountryID
                      FROM Countries C
                      WHERE C.CountryName =
'Германия'))
```

	OrderID	OrderDate	Amount	ProductID	CounterpartyID
1	2	2020-11-17	50	2	3
2	6	2020-11-06	65	4	3

Рисунок 3.12 – Результат выполнения запроса с двумя подзапросами

3.4 Разработка триггеров, пользовательских функций и хранимых процедур.

Была разработана хранимая процедура, которая будет выводить товары по условию продажи "По рецепту" и "Без рецепта". Результат выполнения хранимой процедуры представлен на рисунке 3.13.

```
CREATE PROCEDURE pr_recept
    @recept VARCHAR(20) = 'По рецепту'
AS
    SELECT *
        FROM Products
        WHERE VacationConditions = @recept
GO
EXEC pr_recept
EXEC pr_recept 'Без рецепта'
```

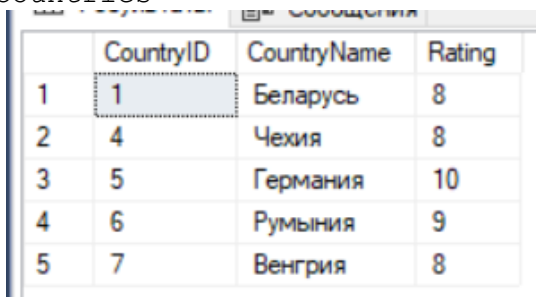
	ProductID	VendoreCode	ProductName	Units	Price	VacationConditions
1	3	125	Зинерит	Упаковка	32,64	По рецепту
1	1	123	Парацетамол	Упаковка	5,96	Без рецепта
2	2	124	Анальгин	Упаковка	1,24	Без рецепта
3	4	126	Бепантен	Тюбик	15,42	Без рецепта
4	5	127	Лизак	Упаковка	4,81	Без рецепта
5	6	128	Антигриппин	Упаковка	12,35	Без рецепта
6	7	759	Ибупрофен	Упаковка	4,1...	Без рецепта

Рисунок 3.13 – Результат выполнения хранимой процедуры "pr_recept"

Была разработана хранимая процедура, которая будет вставлять новые строки в таблицу «Страны». Результат выполнения хранимой процедуры представлен на рисунке 3.14.

```
CREATE PROCEDURE pr_CountryInsertion
    @CountryID INT,
    @CountryName VARCHAR(50),
    @Rating INT
AS
    INSERT INTO Countries
        VALUES (@CountryID, @CountryName, @Rating)
GO

EXEC pr_CountryInsertion '6', 'Румыния', '9'
EXEC pr_CountryInsertion '7', 'Венгрия', '8'
SELECT * FROM Countries
```



	CountryID	CountryName	Rating
1	1	Беларусь	8
2	4	Чехия	8
3	5	Германия	10
4	6	Румыния	9
5	7	Венгрия	8

Рисунок 3.14 – Результат выполнения хранимой процедуры "pr_CountryInsertion"

Была разработана хранимая процедура, которая будет осуществлять выбор ФИО и телефона контрагента по его названию. Результат выполнения хранимой процедуры представлен на рисунке 3.15.

```
DROP PROCEDURE pr_CounterpartyData
CREATE PROCEDURE pr_CounterpartyData
    @Name VARCHAR(50) OUTPUT,
    @FullName VARCHAR(50) OUTPUT,
    @Phone CHAR(15) OUTPUT
AS
    SELECT @FullName = FullName, @Phone = Phone
    FROM Counterparties
    WHERE CounterpartyName = @Name
GO
DECLARE @Name VARCHAR(50), @FullName VARCHAR(50), @Phone
CHAR(15)
SET @Name = 'ООО "Фармекс"'
EXEC pr_CounterpartyData @Name OUTPUT, @FullName
OUTPUT, @Phone OUTPUT
SELECT @Name AS [Наименование контрагента],
@FullName AS [ФИО контрагента], @Phone AS [Телефон]
GO
DECLARE @Name VARCHAR(50), @FullName VARCHAR(50), @Phone
CHAR(15)
SET @Name = 'ООО "ЧехМедПрепарат"'
EXEC pr_CounterpartyData @Name OUTPUT, @FullName
OUTPUT, @Phone OUTPUT
SELECT @Name AS [Наименование контрагента],
@FullName AS [ФИО контрагента], @Phone AS [Телефон]
GO
```

Результаты			
	Наименование контрагента	ФИО контрагента	Телефон
1	ООО "Фармекс"	Федоров Владимир Вечеславович	+789265894657

	Наименование контрагента	ФИО контрагента	Телефон
1	ООО "ЧехМедПрепарат"	Августовская Ирина Владимировна	+75912364856

Рисунок 3.15 – Результат выполнения хранимой процедуры "pr_CounterpartyData"

Была создана определяемая пользователем функция типа Scalar, которая возвращает название товара по его артикулу. Результат выполнения функции представлен на рисунке 3.16.

```
CREATE FUNCTION fn_ProductNameChoice
    (@Code INT)
RETURNS VARCHAR(50)
BEGIN
    DECLARE @Name VARCHAR(50)
    SELECT @Name = ProductName
        FROM Products
        WHERE VendoreCode = @Code
    RETURN @Name
END
GO

DECLARE @Code INT
SET @Code = '124'
SELECT @Code AS [Артикул],
    dbo.fn_ProductNameChoice(@Code) AS [Название товара]
```

Результаты		
	Артикул	Название товара
1	124	Анальгин

Рисунок 3.16 –3 Результат выполнения функции "fn_ProductNameChoice"

Была создана определяемая пользователем функция типа Inline Table-valued, которая возвращает список оптовых заказов по коду товара. Результат выполнения функции представлен на рисунке 3.17.

```
CREATE FUNCTION fn_SelectionWholesaleOrdersByProductIdNAMES
    (@Code INT)
RETURNS TABLE
AS RETURN
    SELECT W.OrderID, W.OrderDate, P.ProductName, W.Amount,
        P.Price, C.CounterpartyName
    FROM WholesaleOrders W, Products P, Counterparties C
    WHERE W.ProductID = P.ProductID AND W.CounterpartyID =
        C.CounterpartyID AND P.ProductID = @Code
GO
SELECT * FROM fn_SelectionWholesaleOrdersByProductIdNAMES ('1')
SELECT * FROM fn_SelectionWholesaleOrdersByProductIdNAMES ('4')
```


Результаты		Сообщения				
	OrderID	OrderDate	ProductName	Amount	Price	CounterpartyName
1	1	2021-01-01	Парацетамол	200	5,96	ООО "ФармВитекс"
2	8	2021-01-08	Парацетамол	200	5,96	ООО "Белфармация"

	OrderID	OrderDate	ProductName	Amount	Price	CounterpartyName
1	4	2021-01-04	Бепантен	150	15,42	ООО "Белфармация"
2	6	2021-01-06	Бепантен	650	15,42	ООО "Фармекс"

Рисунок 3.17 – Результат выполнения функции

Была создана определяемая пользователем функция типа Multi-statement table-valued, которая возвращает таблицу @SalesMainWarehouse с новым столбцом MainWarehouse, в котором определяется осуществилась ли продажа товара определенного контрагента с основного склада (1) или нет (0). Основным является склад с кодом 1. Результат выполнения функции представлен на рисунке 3.18.

```

CREATE FUNCTION fn_SalesFromMainWarehouse
    (@CounterpartyID INT)
RETURNS @SalesMainWarehouse TABLE (
    WaybillID int NOT NULL,
    SaleDate date NOT NULL,
    Counterparty varchar(50) NOT NULL,
    Amount int NOT NULL CHECK (Amount>=0),
    Price money NOT NULL CHECK (Price>=0),
    WarehouseName varchar(50) NOT NULL,
    ProductName varchar(50) NOT NULL,
    MainWarehouse BIT NULL)
BEGIN
    DECLARE @rowset TABLE (
        WaybillID int PRIMARY KEY,
        SaleDate date NOT NULL,
        Counterparty varchar(50) NOT NULL,
        Amount int NOT NULL CHECK (Amount>=0),
        Price money NOT NULL CHECK (Price>=0),
        WarehouseName varchar(50) NOT NULL,
        ProductName varchar(50) NOT NULL,
        MainWarehouse BIT DEFAULT 0 NULL)

    INSERT @rowset (WaybillID, SaleDate, Counterparty, Amount, Price,
WarehouseName, ProductName)
        SELECT S.WaybillID, S.SaleDate, S.Counterparty, S.Amount,
S.Price, W.WarehouseName, P.ProductName
            FROM Sales S, Warehouses W, Products P
            WHERE S.WarehouseID = W.WarehouseID AND
S.ProductID = P.ProductID AND Counterparty = @CounterpartyID
    UPDATE @rowset SET MainWarehouse = 1
        WHERE WarehouseName = 'Склад 123'
    INSERT @SalesMainWarehouse
        SELECT WaybillID, SaleDate, Counterparty, Amount, Price,
WarehouseName, ProductName, MainWarehouse
            FROM @rowset

    RETURN
END
GO
SELECT * FROM fn_SalesFromMainWarehouse('1')
SELECT * FROM fn_SalesFromMainWarehouse('5')

```

Результаты		Сообщения						
	WaybillID	SaleDate	Counterparty	Amount	Price	WarehouseName	ProductName	MainWarehouse
1	1	2021-01-15	1	170	4,768	Склад 123	Парацетамол	1
2	11	2021-01-22	1	80	3,848	Склад 123	Лизак	1
3	13	2021-01-23	1	710	3,848	Склад 234	Лизак	0
	WaybillID	SaleDate	Counterparty	Amount	Price	WarehouseName	ProductName	MainWarehouse
1	5	2021-01-16	5	80	3,848	Склад 567	Лизак	0
2	9	2021-01-19	5	210	12,336	Склад 345	Бепантен	0

Рисунок 3.18 –4 Результат выполнения функции "fn_SalesFromMainWarehouse"

Была разработана процедура с использованием курсора, которая подсчитывает суммарную стоимость всех товаров, проданных в течение указанного интервала времени. Задан интервал с 17 по 22 ноября 2020 года. Результат выполнения процедуры представлен на рисунке 3.19.

```

CREATE PROCEDURE pr_CostProductsInterval
@IntervalStart DATE,
@IntervalEnd DATE,
@Cost MONEY OUTPUT AS
IF @IntervalStart IS NULL
    SET @IntervalStart = CAST(GETDATE() - 100 AS DATE)
IF @IntervalEnd IS NULL
    SET @IntervalEnd = CAST(GETDATE() AS DATE)
SET @Cost = 0
DECLARE @OrderCost MONEY
DECLARE myCursor CURSOR LOCAL STATIC FOR
SELECT Sales.Amount * Sales.Price
    FROM Sales
        WHERE Sales.SaleDate BETWEEN @IntervalStart AND
@IntervalEnd
OPEN myCursor
FETCH FIRST FROM myCursor INTO @OrderCost
WHILE @@FETCH_STATUS = 0
BEGIN
    SET @Cost = @Cost + @OrderCost
    FETCH NEXT FROM myCursor INTO @OrderCost
END
CLOSE myCursor
DEALLOCATE myCursor
GO
DECLARE @BeginDate DATE, @EndDate DATE, @Cost MONEY
SET DATEFORMAT Dmy
SET @BeginDate = '17.11.2020'
SET @EndDate = '22.11.2020'
EXEC pr_CostProductsInterval @BeginDate, @EndDate, @Cost OUTPUT
SELECT @BeginDate [Начало интервала],
    @EndDate [Конец интервала],
    @Cost [Суммарная стоимость заказов]
GO

```

	Начало интервала	Конец интервала	Суммарная стоимость заказов
1	2020-11-17	2020-11-22	1345,73

Рисунок 3.19 – Результат выполнения процедуры с курсором

Был разработан триггер, запрещающий модификацию данных в столбце «WarehouseName» таблицы «Warehouses». Результат работы триггера представлен на рисунке 3.20 [3].

```
CREATE TRIGGER tr_NoModifyWarehouseName
ON Warehouses
FOR UPDATE AS
IF UPDATE (WarehouseName)
BEGIN
PRINT 'Обновление столбца WarehouseName запрещено'
ROLLBACK TRAN
END
GO
UPDATE Warehouses
SET WarehouseName = 'Новыйсклад'
WHERE WarehouseID = 4
GO
```

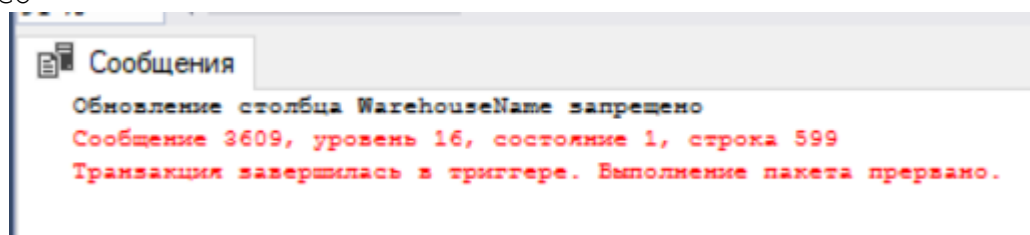


Рисунок 3.20 –5 Результат работы триггера "tr_NoModifyWarehouseName"

Был разработан триггер, который будет изменять цену товара (надбавка 0,38) при любой операции INSERT или UPDATE над таблицей «Products». Результат работы триггера представлен на рисунке 3.21.

```
CREATE TRIGGER tr_ProductsInsertUpdate
ON Products
AFTER INSERT,UPDATE
AS
UPDATE Products
SET Price = Price + Price * 0.38
WHERE ProductId = (SELECT ProductId FROM inserted)
INSERT INTO Products
VALUES
('7', '759', 'Ибупрофен', 'Упаковка', '2.98', 'Без рецепта')
SELECT * FROM Products
```

Результаты		Сообщения				
	ProductID	VendoreCode	ProductName	Units	Price	VacationConditions
1	1	123	Парацетамол	Упаковка	5,96	Без рецепта
2	2	124	Анальгин	Упаковка	1,24	Без рецепта
3	3	125	Зинерит	Упаковка	32,64	По рецепту
4	4	126	Бепантен	Тюбик	15,42	Без рецепта
5	5	127	Лизак	Упаковка	4,81	Без рецепта
6	6	128	Антигриппин	Упаковка	12,35	Без рецепта
7	7	759	Ибупрофен	Упаковка	4,1124	Без рецепта

Рисунок 3.21 – Результат работы триггера "tr_ProductsInsertUpdate"

Был разработан триггер, который будет сохранять изменения столбца «Rating» в таблице «Countries» в таблицу «CountriesRatingUpdate». Результат работы триггера представлен на рисунке 3.22.

```
CREATE TABLE CountriesRatingUpdate (
    CountryID INT NULL,
    UserName CHAR(16) NULL,
    Date DATETIME NULL,
    RatingOld FLOAT NULL,
    RatingNew FLOAT NULL
);
CREATE TRIGGER tr_ModifyCountryRating
ON Countries AFTER UPDATE
AS IF UPDATE(Rating)
BEGIN
    DECLARE @ratingOld INT
    DECLARE @ratingtNew INT
    DECLARE @CountryID INT
    SELECT @ratingOld = (SELECT Rating FROM deleted)
    SELECT @ratingtNew = (SELECT Rating FROM inserted)
    SELECT @CountryID = (SELECT CountryID FROM deleted)
    INSERT INTO CountriesRatingUpdate VALUES
        (@CountryID, USER_NAME(), GETDATE(), @ratingOld,
@ratingtNew)
END
UPDATE Countries
SET Rating = 8
WHERE CountryID = '4';
SELECT * FROM Countries
SELECT * FROM CountriesRatingUpdate
```



	CountryID	UserName	Date	RatingOld	RatingNew
1	4	dbo	2021-05-11 23:25:53.187	9	8

Рисунок 3.22 – Результат работы триггера "tr_ModifyCountryRating"

Был разработан триггер с использованием курсора для таблицы «Sales», который срабатывает при любом изменении цены продажи и корректирует значения добавленных столбцов «Cost» и «CostUSD» в тех строках, которые соответствуют продаже с изменившейся ценой. Результат работы триггера представлен на рисунке 3.23.

```
ALTER TABLE Sales ADD Cost MONEY NULL
ALTER TABLE Sales ADD CostUSD MONEY NULL
SELECT * FROM Sales
CREATE TRIGGER tr_SalesCost
ON Sales
FOR UPDATE AS
    IF UPDATE(Price)
BEGIN
    DECLARE @WaybillID INT, @Price MONEY, @PriceUSD MONEY;
    DECLARE myCursor CURSOR LOCAL STATIC FOR
        SELECT inserted.WaybillID, inserted.Price, inserted.Price /
2.56
        FROM inserted
OPEN myCursor;
```

```

FETCH FIRST FROM myCursor INTO @WaybillID, @Price, @PriceUSD;
WHILE @@FETCH_STATUS = 0
BEGIN
    UPDATE Sales
    SET Cost = Amount * @Price,
        CostUSD = Amount * @PriceUSD
    WHERE WaybillID = @WaybillID;
    FETCH NEXT FROM myCursor INTO @WaybillID, @Price, @PriceUSD;
END;
CLOSE myCursor
DEALLOCATE myCursor
END
GO
UPDATE Sales
SET Price = Price * 0.80
SELECT * FROM Sales

```

Результаты		Сообщения							
	WaybillID	SaleDate	Counterparty	Amount	Price	WarehouseID	ProductID	Cost	CostUSD
1	1	2020-11-15	1	17	4,768	1	1	81,056	31,6625
2	2	2020-11-15	2	12	0,992	2	2	11,904	4,65
3	3	2020-11-16	3	20	26,112	3	3	522,24	204,00
4	4	2020-11-16	4	10	12,336	4	4	123,36	48,188
5	5	2020-11-16	5	8	3,848	5	5	30,784	12,0248
6	6	2020-11-17	6	16	9,88	6	6	158,08	61,7504
7	7	2020-11-17	3	25	0,992	1	2	24,80	9,6875
8	8	2020-11-18	4	5	26,112	2	3	130,56	51,00
9	9	2020-11-19	5	21	12,336	3	4	259,056	101,1948
10	10	2020-11-20	6	30	3,848	4	5	115,44	45,093
11	11	2020-11-22	1	8	3,848	1	5	30,784	12,0248
12	11	2020-11-22	3	45	3,848	4	5	173,16	67,6395
13	12	2020-11-22	2	12	3,848	2	5	46,176	18,0372
14	12	2020-11-22	4	36	3,848	3	5	138,528	54,1116
15	13	2020-11-23	1	71	3,848	2	5	273,208	106,7201
16	13	2020-11-23	3	7	3,848	3	5	26,936	10,5217
17	14	2020-11-24	2	34	3,848	1	5	130,832	51,1054
18	14	2020-11-24	4	21	3,848	4	5	80,808	31,5651

Рисунок 3.23 – Результат работы триггера "tr_SalesCost"

3.5 Администрирование базы данных.

Было создано 2 роли (основная и дополнительная) и 4 пользователя базы данных [12].

Первая основная роль – это «MainUser», она предназначена для работников аптеки. Внутри данной роли было создано два пользователя – это “Главный аптекарь” и “Аптекарь” (рис. 3.24).

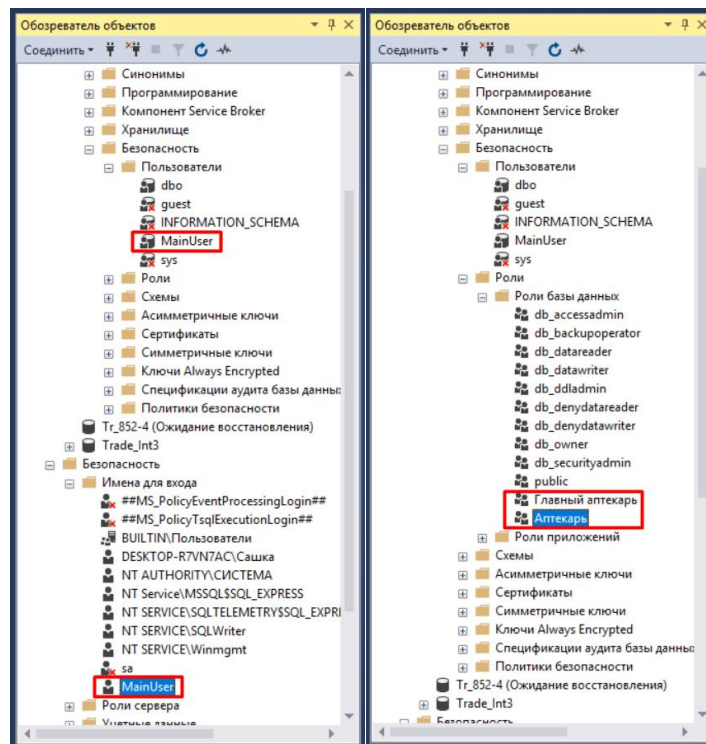


Рисунок 3.24 – Результат создания роли "MainUser" и пользователей в ней

Вторая дополнительная роль – это «AdditionalUser», она предназначена для клиентов аптеки. Внутри данной роли было создано два пользователя – это «Постоянный покупатель» и «Покупатель» (рис. 3.25).

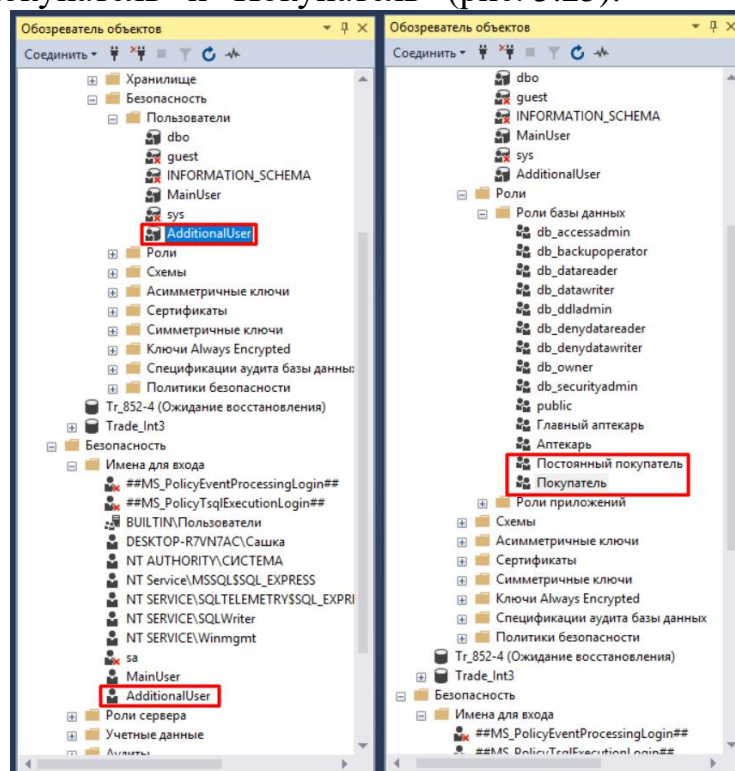


Рисунок 3.25 – Результат создания роли "AdditionalUser" и пользователей в ней

Следующим шагом было назначение полномочий на объекты базы данных созданным пользователям [8].


```

GRANT SELECT, INSERT, UPDATE, DELETE ON Products TO [Главный
аптекарь] WITH GRANT OPTION
GRANT SELECT, INSERT, UPDATE, DELETE ON WholesaleOrders TO
[Главный аптекарь] WITH GRANT OPTION
GRANT SELECT, INSERT, UPDATE, DELETE ON Sales TO [Главный
аптекарь] WITH GRANT OPTION
GRANT SELECT, INSERT, UPDATE, DELETE ON Deliveries TO [Главный
аптекарь] WITH GRANT OPTION
GRANT SELECT, INSERT, UPDATE, DELETE ON Counterparties TO
[Главный аптекарь] WITH GRANT OPTION
GRANT SELECT, INSERT, UPDATE, DELETE ON Countries TO [Главный
аптекарь] WITH GRANT OPTION
GRANT SELECT, INSERT, UPDATE, DELETE ON Warehouses TO [Главный
аптекарь] WITH GRANT OPTION
GRANT SELECT, INSERT, UPDATE, DELETE ON CountriesRatingUpdate TO
[Главный аптекарь] WITH GRANT OPTION
GRANT SELECT, INSERT, UPDATE ON Products TO Аптекарь
GRANT SELECT ON WholesaleOrders TO Аптекарь
GRANT SELECT, INSERT, UPDATE ON Sales TO Аптекарь
GRANT SELECT, INSERT, UPDATE ON Deliveries TO Аптекарь
GRANT SELECT ON Counterparties TO Аптекарь
GRANT SELECT ON Countries TO Аптекарь
GRANT SELECT ON Warehouses TO Аптекарь
GRANT SELECT ON CountriesRatingUpdate TO Аптекарь
GRANT SELECT ON Products TO [Постоянный покупатель]
GRANT SELECT ON Products TO Покупатель

```

На рисунке 3.26 представлены полномочия пользователя «Главный аптекарь».

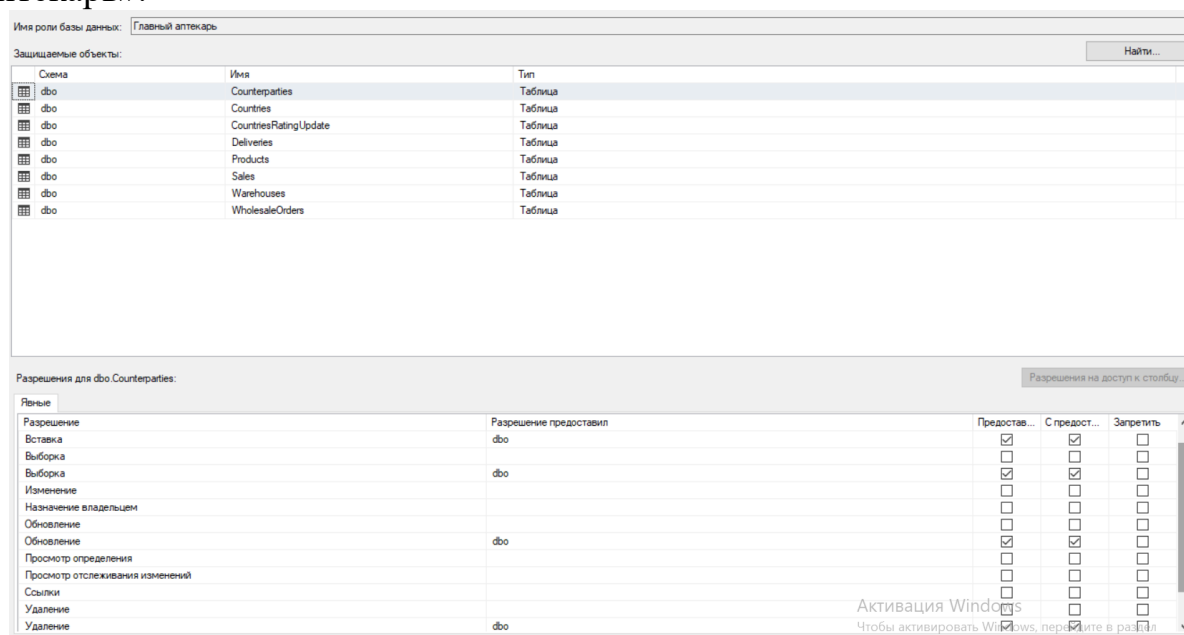


Рисунок 3.26 – Полномочия пользователя "Главный аптекарь"

3.6 Разработка интерфейса приложения.

На рисунке 3.27 представлена первая форма программы – форма «Авторизация». В программе предусмотрена защита на несанкционированный доступ посредством пароля [14].

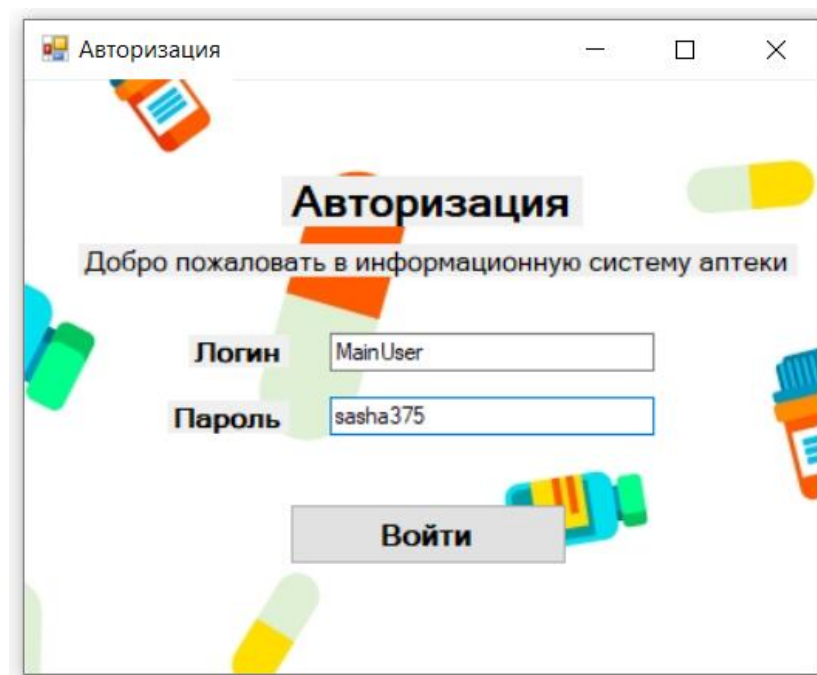


Рисунок 3.27 – Форма «Авторизация»

Листинг формы «Авторизация»:

```
namespace DB_Pharmacy_Sevryuk
{public partial class LoginForm : Form
    {public LoginForm()
        {InitializeComponent();}
        private void productsBindingNavigatorSaveItem_Click(object
sender, EventArgs e)
        {this.Validate();
        this.productsBindingSource.EndEdit();
        this.tableAdapterManager.UpdateAll(this.dB_Pharmacy_SevryukDataSe
t);}

        private void button1_Click(object sender, EventArgs e)
        {
            if (textBox1.Text == "MainUser" && textBox2.Text == "sasha375")
            {
                MainForm s = new MainForm();
                s.Show();
                this.Hide();
            }
            else
            {
                textBox1.Text = "";
                textBox2.Text = "";
                MessageBox.Show("Неправильный логин или
пароль!");
            }
        }
    }
}
```

При вводе логина и пароля с нее осуществляется переход на главную форму программы, а именно «Основную форму» (рис. 3.28). На основной форме расположены кнопки меню для перехода на остальные формы с таблицами БД [1].

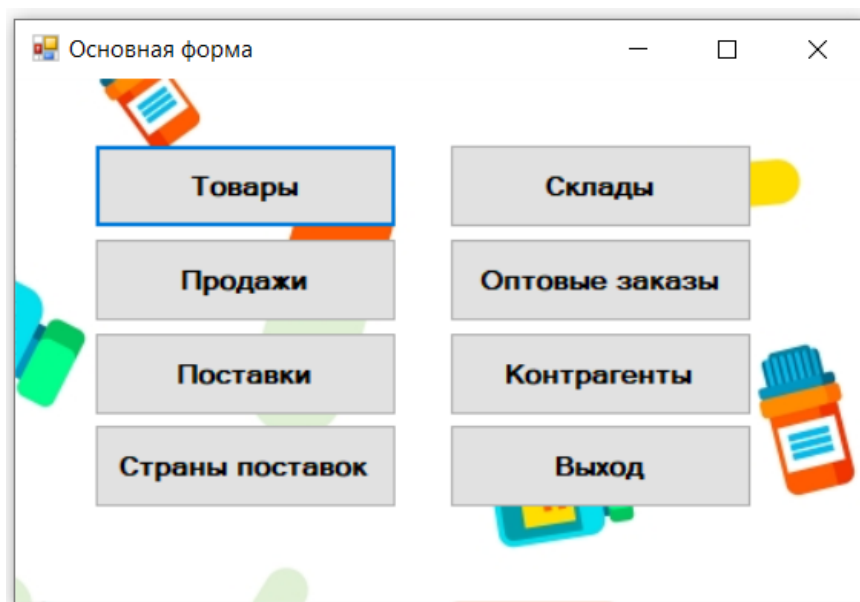


Рисунок 3.286 – Основная форма с кнопками меню

Листинг «Основной формы»:

```
namespace DB_Pharmacy_Sevryuk
{public partial class MainForm : Form
    {public MainForm()
        {InitializeComponent();}
        private void button1_Click(object sender, EventArgs e)
        {ProductsForm f = new ProductsForm();
            f.Show();}
        private void button2_Click(object sender, EventArgs e)
        {SalesForm f = new SalesForm();
            f.Show();}
        private void button3_Click(object sender, EventArgs e)
        {DeliveriesForm f = new DeliveriesForm();
            f.Show();}
        private void button7_Click(object sender, EventArgs e)
        {CountriesForm f = new CountriesForm();
            f.Show();}
        private void button4_Click(object sender, EventArgs e)
        {WarehousesForm f = new WarehousesForm();
            f.Show();}
        private void button5_Click(object sender, EventArgs e)
        {WholesaleOrdersForm f = new WholesaleOrdersForm();
            f.Show();}
        private void button6_Click(object sender, EventArgs e)
        {CounterpartiesForm f = new CounterpartiesForm();
            f.Show();}
        private void button8_Click(object sender, EventArgs e)
        {Application.Exit();}
    }
}
```

На форме «Товары» расположена таблица «Товары» и связанные с ней таблицы «Продажи», «Поставки», «Оптовые заказы» (рис. 3.29). Также на форме есть возможность выполнить поиск товаров по наименованию [13].

Товары

Код товара	Артикул	Наименование	Единицы измерения	Цена	Условия отпуска
4	126	Бепантен	Тюбик	15,4200	Без рецепта

Поиск по наименованию товара

Бепантен

Поиск

Продажи

Код Накладной	Дата Продажи	Контрагент	Количество	Цена	Код Склада	Код Товара	Стоимость	Стоимость в долларах
4	16.01.2021	4	100	12,3360	4	4	123,3600	48,1880
9	19.01.2021	5	210	12,3360	3	4	259,0560	101,1948

Поставки

Код накладной	Дата поставки	Контрагент	Количество	Цена	Код склада	Код товара
2	10.11.2020	3	45	15,4200	2	4
9	14.11.2020	3	45	15,4200	5	4

Оптовые заказы

Код заказа	Дата заказа	Количество	Код товара	Код контрагента
4	04.01.2021	150	4	2
6	06.01.2021	650	4	3

Рисунок 3.29 –7 Форма "Товары"

На формах «Продажи» и «Поставки» расположены таблицы с соответствующими названиями (рис. 3.30) [15].

Код накладной	Дата продажи	Контрагент	Количество	Цена	Код склада	Код товара	Стоимость	Стоимость в долларах
1	15.01.2021	1	170	4,7600	1	1	81,0560	31,6625
2	15.01.2021	2	120	0,9000	2	2	11,9040	4,6500
3	16.01.2021	3	200	26,1120	3	3	522,2400	204,0000
4	16.01.2021	4	100	12,3360	4	4	123,3600	48,1880
5	16.01.2021	5	80	3,9400	5	5	30,7640	12,0240
6	17.01.2021	6	160	9,8800	6	6	158,0800	61,7604
7	17.01.2021	3	250	0,9000	1	2	24,8000	9,6075
8	18.01.2021	4	50	26,1120	2	3	130,5600	51,0000
9	19.01.2021	5	210	12,3360	3	4	259,0560	101,1948
10	20.01.2021	6	300	3,9400	4	5	118,4400	45,0930
11	22.01.2021	1	80	3,9400	1	5	30,7640	12,0240
12	22.01.2021	3	450	3,9400	4	5	177,3000	67,6395
13	22.01.2021	2	120	3,9400	2	5	46,1760	18,0372
14	22.01.2021	4	360	3,9400	3	5	139,5200	54,1116
15	23.01.2021	1	710	3,9400	2	5	279,2800	106,7201
16	23.01.2021	3	70	3,9400	3	5	26,3600	10,2017
17	24.01.2021	2	340	3,9400	1	5	132,9200	51,1054
18	24.01.2021	4	210	3,9400	4	5	80,8000	31,5651

Код накладной	Дата поставки	Контрагент	Количество	Цена	Код склада	Код товара
1	10.11.2020	1	50	1,2400	1	2
2	10.11.2020	3	45	15,4200	2	4
3	11.11.2020	2	35	5,9600	3	1
4	11.11.2020	4	25	32,6400	4	3
5	12.11.2020	5	20	12,3500	5	6
6	13.11.2020	6	65	4,8100	6	5
7	13.11.2020	1	64	32,6400	2	3
8	13.11.2020	2	40	5,9600	4	1
9	14.11.2020	3	45	15,4200	5	4
10	14.11.2020	2	30	1,2400	1	2

Рисунок 3.30 – Формы "Продажи" и "Поставки"

На форме «Страны» расположена таблица «Страны» и связанная с ней таблица «Контрагенты» (рис. 3.31). Также на форме есть возможность выполнить поиск страны по названию [13].

Код страны	Название	Рейтинг
4	Чехия	8

Поиск по стране

Чехия

Поиск

Код контрагента	Наименование	Адрес	Телефон	Электронный адрес	ФИО контрагента	Код страны
6	ООО "ЧехМедП..."	г. Прага, ул. Че...	+75912364856	zechmedpreparat...	Августовская И...	4

Рисунок 3.31 –8 Форма "Страны"

На форме «Склады» расположена таблица «Склады» и связанные с ней таблицы «Поставки» и «Продажи» (рис. 3.32). Также на форме есть возможность выполнить поиск склада по наименованию [14].

Код склада	Наименование	Расположение	ФИО	Отв	Лица	Телефон
2	Склад 234	г. Брест, ул. По...	Цветкова Светл...			+375448963571

Поиск по наименованию склада

Склад 234

Поиск

Код накладной	Дата поставки	Контрагент	Количество	Цена	Код склада	Код товара
2	10.11.2020	3	45	15,4200	2	4
7	13.11.2020	1	64	32,6400	2	3

Код накладной	Дата продажи	Контрагент	Количество	Цена	Код склада	Код товара	Стоимость	Стоимость в долларах
2	15.01.2021	2	120	0,9920	2	2	11,9040	4,6500
8	18.01.2021	4	50	26,1120	2	3	130,5600	51,0000
12	22.01.2021	2	120	3,8480	2	5	46,1760	18,0372
13	23.01.2021	1	710	3,8480	2	5	273,2080	106,7201

Рисунок 3.32 – Форма "Склады"

На форме «Оптовые заказы» расположена таблица с соответствующим названием (рис. 3.33). Также на форме есть возможность выполнить поиск заказа по наименованию контрагента и по наименованию товара [5].

Код заказа	Дата заказа	Количество	Наименование товара	Наименование контрагента
2	17.01.2021	500	Анальгин	ООО "Фармекс"
6	06.01.2021	650	Белантен	ООО "Фармекс"

Код заказа	Дата заказа	Количество	Наименование товара	Наименование контрагента
4	04.01.2021	150	Белантен	ООО "Белфарм..."
6	06.01.2021	650	Белантен	ООО "Фармекс"

Поиск по наименованию контрагента

Поиск по наименованию товара

Поиск

Поиск

Рисунок 3.33 – Форма "Оптовые заказы"

Листинг формы «Оптовые заказы»:

```
namespace DB_Pharmacy_Sevryuk
{
    public partial class WholesaleOrdersForm : Form
    {
        public WholesaleOrdersForm()
        {
            InitializeComponent();
        }
        private void button1_Click(object sender, EventArgs e)
        {
            if (textBox1.Text == "")
            {
                fullWholesaleOrdersBindingSource.RemoveFilter();
            }
            else
            {
                fullWholesaleOrdersBindingSource.Filter =
                "CounterpartyName LIKE '%" + textBox1.Text + "%'";
            }
        }
        private void button2_Click(object sender, EventArgs e)
        {
            if (textBox2.Text == "")
            {
                fullWholesaleOrdersBindingSource.RemoveFilter();
            }
            else
            {
                fullWholesaleOrdersBindingSource.Filter =
                "ProductName LIKE '%" + textBox2.Text + "%'";
            }
        }
    }
}
```

На форме «Контрагенты» расположена таблица «Контрагенты» и связанная с ней таблица «Оптовые заказы» (рис. 3.34). Также на форме есть возможность выполнить поиск контрагента по наименованию [15].

Код контрагента	Наименование	Адрес	Телефон	Электронная почта	ФИО контрагента	Код страны
3	ООО "Фармекс"	г. Дрезден, ул. ...	+789265894657	fatmeks@mail.ru	Федоров Влади...	5

Поиск по наименованию контрагента: ООО "Фармекс" Поиск

Код заказа	Дата заказа	Количество	Код товара	Код контрагента
2	17.01.2021	500	2	3
6	06.01.2021	650	4	3

Рисунок 3.34 – Форма "Контрагенты"

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной курсовой работы была достигнута поставленная нами цель, а именно закрепление, углубление, систематизация теоретических знаний, полученных в процессе изучения дисциплины «Системы баз данных», а также развитие практических навыков разработки автоматизированной информационной системы с помощью MS SQL Server и платформы .NET (язык C#). Также, были выполнены все поставленные задачи.

Проанализировав предметную область аптеки, мной были выделены две группы пользователей – сотрудники и клиенты аптеки. Был проведен комплексный анализ функционирования аптеки, в результате которого выявлен ряд недостатков и проблем. На основании этих недостатков в работе аптечного предприятия были определены следующие процессы, нуждающиеся в оптимизации: процессы сбора, обработки, хранения и анализа информации о продажах, поставках, контрагентах и местах хранения товаров, реализуемых в аптеке. Была выбрана СУБД, наиболее подходящая для решения поставленных задач.

Также в результате проведенного анализа было выделено семь объектов – оптовые заказы, товары, продажи, поставки, контрагенты, страны, места хранения.

Была создана концептуальная (инфологическая) модель данных в нотации Питера Чена исходя из представлений пользователей о предметной области. Концептуальная модель данных на основе реляционной модели данных была преобразована в логическую модель, не зависимую от особенностей используемой в дальнейшем СУБД для физической реализации базы данных. Была проведена нормализация и проверка логической модели данных на предмет возможности выполнения всех предусмотренных транзакций. Затем полученная в ERWin Data Modeler ER-диаграмма была выражена в терминах выбранной в первой главе СУБД, а именно MS SQL Server.

Были разработаны запросы для создания БД, таблиц (с учетом ограничений целостности данных, ограничений на значения, значений по умолчанию и т.п.) и заполнения ими данными, запросы манипулирования данными, на выборку данных и для создания представлений. Были разработаны триггеры, пользовательские функции и хранимые процедуры.

Таким образом, были автоматизированы процессы поиска, добавления, редактирования, сохранения информации о реализуемых товарах аптеки, продажах, поставках, контрагентах и местах хранения товаров. Что позволит аптеке минимизировать затраты времени, материальных и трудовых ресурсов

в ходе своей деятельности и упростить процесс обработки информации, что даст ощутимый экономический эффект. Разработанная информационная система соответствует поставленным задачам и является актуальной.

Планируется доработка данной информационной системы для возможности функционирования в многопользовательском режиме в архитектуре клиент-сервер. А также в дальнейшем планируется разработка мобильного приложения, которое будет синхронизировано с разработанным десктопным приложением. Рассматривается возможность внедрения разработанной информационной системы на предприятии аптечной сети.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Абрамян, М.Э. Visual C# на примерах / М.Э. Абрамян. - СПб: БХВ-Петербург, 2012. - 496 с.
2. Бен-Ган, И. Microsoft SQL Server 2012. Основы T-SQL. - Пер. с англ. / И. Бен-Ган. - Москва: ЭКСМО, 2015. - 400 с.
3. Бондарь, А. Г. Microsoft SQL Server 2012 / А.Г. Бондарь. - СПб.: БХВ-Петербург, 2013. - 608 с.
4. Голицына, О.Л. Базы данных: Учебное пособие / О.Л. Голицына, Н.В. Максимов, И.И. Попов. - М.: Форум, 2012. - 400 с.
5. Голицына, О.Л. Основы алгоритмизации и программирования: Учебное пособие / О.Л. Голицына, И.И. Попов. - М.: Форум: Инфра-М, 2015. - 125 с.
6. Горшков, Е.А. Обзор и анализ инструментальных средств обеспечения кадровой деятельности / Е.А. Горшков, В.Н. Саганова // Современные тенденции технических наук: материалы II Междунар. науч. конф. [Электронный ресурс]. - 2013. - Режим доступа: <https://moluch.ru/conf/tech/archive/74/3890/>. - Дата доступа: 24.05.2021.
7. Гриневич, Е.Г. Автоматизация проектирования БД. Создание Базы данных в СУБД MS SQL SERVER / Е.Г. Гриневич, И.Г. Орешко, Ю.Н. Силкович. - Минск, 2017. - 142 с.
8. Грофф, Дж.Р. SQL. Полное руководство. - Пер. с англ. / Дж.Р. Грофф, П.Н. Вайнберг, Э.Дж. Оппель. - Москва: Вильямс, 2015. - 952 с.
9. Емельянова, Н.З. Основы построения автоматизированных информационных систем: Учебное пособие / Н.З. Емельянова, Т.Л. Партыка, И.И. Попов. - М.: «Форум»: ИНФРА, 2017. - 416с.
10. Карпова, И.П. Базы данных: Учебное пособие / И.П. Карпова. - СПб.: Питер, 2013. - 240 с.
11. Коннолли, Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика. - Пер. с англ. / Т. Коннолли, К. Бегг. - Москва: Диалектика, 2017. - 1440 с.
12. Куликов, С.С. Работа с MySQL, SQL Server и Oracle в примерах. 10. Практическое пособие / С.С. Куликов - Минск, УП «Бэфф», 2016. - 556 с.
13. Лабор, В.В. C# Создание приложений для Windows / В.В. Лабор. - Мн.: Харвест, 2013. - 383 с.
14. Петцольд Ч. Программирование для Microsoft Windows на C#. Том 1 / Ч. Петцольд. - М.: Русская редакция, 2012. - 576 с.
15. Петцольд Ч. Программирование с использованием Microsoft Windows Forms / Ч. Петцольд. - СПб: Питер, 2012. - 432 с.