**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КОМИ**

**Государственное профессиональное образовательное учреждение**

**«Воркутинский арктический горно-политехнический колледж»**

**КУРСОВОЙ ПРОЕКТ**

По дисциплине МДК.05.02 Разработка кода информационных систем

**Разработка информационной системы «Продажа компьютерной периферии»**

Выполнила студентка гр. ИСП-21 /\_\_\_\_\_\_\_\_\_\_\_\_/Соколова Дана Романовна/

                                                                                    (подпись)                                          (Ф.И.О.)

**ОЦЕНКА:\_\_\_\_\_\_\_\_\_\_\_\_\_**

Дата:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ПРОВЕРИЛ**

Научный руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_/ Егоров Данил Павлович/

                                              (подпись)                                (Ф.И.О.)

Воркута

2024

**СОДЕРЖАНИЕ**

[Техническое задание 2](#_Toc160626985)

[ВВЕДЕНИЕ 9](#_Toc160626986)

[ГЛАВА 1. ВЫБОР ИНСТРУМЕНТАРИЯ 10](#_Toc160626987)

[Платформа .NET 11](#_Toc160626988)

[1.2 Язык программирования С# 11](#_Toc160626989)

[1.3 Windows Presentation Foundation (WPF) 11](#_Toc160626990)

[1.4 СУБД SQL server 11](#_Toc160626991)

[1.5 Microsoft SQL Server Management Studio 11](#_Toc160626992)

[1.6 Entity Framework 11](#_Toc160626993)

[ГЛАВА 2. ПРОЕТИРОВАНИЕ БАЗЫ ДАННЫХ 11](#_Toc160626994)

[2.1 Разработка диаграммы ERD 11](#_Toc160626995)

[2.2 Разработка базы данных 11](#_Toc160626996)

[ГЛАВА 3. РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ 11](#_Toc160626997)

[3.1 Разработка прототипа информационной системы 11](#_Toc160626998)

[3.2 Программирование информационной системы 11](#_Toc160626999)

[3.2.1 Разработка модуля «Авторизация» 11](#_Toc160627000)

[3.2.2 Разработка модуля ….. 11](#_Toc160627001)

[3.2.3 Разработка модуля ….. 11](#_Toc160627002)

[3.2.4 Разработка модуля «Отчет» 11](#_Toc160627003)

[ЗАКЛЮЧЕНИЕ 11](#_Toc160627004)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 11](#_Toc160627005)

[ЗАКЛЮЧЕНИЕ 12](#_Toc160627006)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 13](#_Toc160627007)

# 

# ВВЕДЕНИЕ

Актуальность выбранной темы в том, что в связи с ростом спроса на компьютерную технику, возможностью оптимизации работы магазина, улучшения коммуникации с клиентами и анализа данных для принятия решений о развитии бизнеса. Кроме того, интеграция с другими системами делает ее еще более удобной для использования.

 Информационная система помогает автоматизировать многие процессы, такие как управление запасами, обработка заказов, учет продаж и др. Это позволяет сократить время на выполнение задач и улучшить качество обслуживания клиентов.

Данная система подойдет для владельцев, сотрудников компьютерных магазинов и тех, кто хочет улучшить коммуникацию с клиентами, автоматизировать процессы и анализировать данные.

**Объект:** информационная система «Компьютерный магазин».

**Предмет:** анализ бизнес-процессов «Компьютерный магазин».

**Цель работы:** разработать информационную систему «Компьютерный магазин».

**Задачи:**

* выбор инструментария;
* проектирование базы данных;
* разработка информационной системы.

# ГЛАВА 1. ВЫБОР ИНСТРУМЕНТАРИЯ

# Платформа .NET

.NET — это платформа для программирования, разработанная компанией Microsoft. Изначально создавалась, чтобы писать программы под операционную систему Windows. Среда работает с несколькими языками программирования, умеет запускать и исполнять код, написанный на них. Разработчик на .NET может одновременно работать с несколькими языками программирования. Среди них — C#, F#, Visual Basic и Visual C++, а также более редкие языки типа J#. Эти языки разные, но благодаря возможностям .NET разработчик может использовать для них единое пространство имен, один и тот же набор технологий, библиотек и API.

.NET включает следующие компоненты:

* Среда выполнения — выполняет код приложения.
* Библиотеки — предоставляют служебные функции, такие как синтаксический анализ JSON.
* Компилятор — компилирует исходный код C# (и других языков) в исполняемый код (среда выполнения).
* Пакет SDK и другие средства — включение создания и мониторинга приложений с помощью современных рабочих процессов.
* Стеки приложений, такие как ASP.NET Core и Windows Forms, которые позволяют создавать приложения.

Среда выполнения, библиотеки и языки являются основными компонентами стека .NET. Компоненты более высокого уровня, такие как средства .NET и стеки приложений, такие как ASP.NET Core, создаются на основе этих основных компонентов. C# — это основной язык программирования для .NET, и большая часть .NET написана на C#.

* 1. **Язык программирования С#**

C# — объектно-ориентированный язык программирования общего назначения. Разработан в 1998—2001 годах группой инженеров компании Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота как язык разработки приложений для платформы Microsoft .NET Framework и .NET Core. Впоследствии был стандартизирован как ECMA-334 и ISO/IEC 23270.

С#‎ разрабатывался как язык программирования прикладного уровня для CLR и, как таковой, зависит, прежде всего, от возможностей самой CLR. Это касается, прежде всего, системы типов С#‎, которая отражает BCL. Присутствие или отсутствие тех или иных выразительных особенностей языка диктуется тем, может ли конкретная языковая особенность быть транслирована в соответствующие конструкции CLR. Так, с развитием CLR от версии 1.1 к 2.0 значительно обогатился и сам C#; подобного взаимодействия следует ожидать и в дальнейшем (однако, эта закономерность была нарушена с выходом C# 3.0, представляющего собой расширения языка, не опирающиеся на расширения платформы .NET). CLR предоставляет С#‎, как и всем другим .NET-ориентированным языкам, многие возможности, которых лишены «классические» языки программирования. Например, сборка мусора не реализована в самом C#‎, а производится CLR для программ, написанных на C#, точно так же, как это делается для программ на VB.NET, J# и др.

Переняв многое от своих предшественников — языков C++, Delphi, Модула, Smalltalk и, в особенности, Java — С#, 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 множественная реализация интерфейсов).С# популярен за счет своей «простоты». Простоты для современных программистов и больших команд разработчиков, чтобы те могли в сжатые сроки создавать функциональные и производительные приложения. Этому способствуют нетипичные конструкции языка и специфичный синтаксис, помогающий максимально органично реализовать намеченные функции.

Популярность языка – еще одно значимое преимущество. Большое количество поклонников C# способствуют его развитию. Также это благоприятно влияет на рост числа вакансий, связанных с разработкой на языке Microsoft. Программисты, хорошо знакомые с С#, востребованы в индустрии, несмотря на их большое и постоянно увеличивающееся количество.

Понятный синтаксис C# заметно упрощает не только разработку как таковую, но и другие важные аспекты совместной работы, например, чтение чужого кода. Это упрощает процесс рефакторинга и исправления ошибок при работе над приложениями в больших командах.

# Windows Presentation Foundation (WPF)

Технология WPF (Windows Presentation Foundation) является часть экосистемы платформы .NET и представляет собой подсистему для построения графических интерфейсов.

Если при создании традиционных приложений на основе WinForms за отрисовку элементов управления и графики отвечали такие части ОС Windows, как User32 и GDI+, то приложения WPF основаны на DirectX. В этом состоит ключевая особенность рендеринга графики в WPF: используя WPF, значительная часть работы по отрисовке графики, как простейших кнопочек, так и сложных 3D-моделей, ложиться на графический процессор на видеокарте, что также позволяет воспользоваться аппаратным ускорением графики.

Одной из важных особенностей является использование языка декларативной разметки интерфейса XAML, основанного на XML: вы можете создавать насыщенный графический интерфейс, используя или декларативное объявление интерфейса, или код на управляемых языках C#, VB.NET и F#, либо совмещать и то, и другое.

Первая версия - WPF 3.0 вышла вместе с .NET Framework 3.0 и операционной системой Windows Vista в 2006 году. И с тех пор платформа WPF является частью экосистемы .NET и развивается вместе с фреймворком .NET. Например, на сегодняшний день последней версией фреймворка .NET является .NET 8, и WPF полностью поддерживается этой версией фреймворка.

WPF существует в виде подмножества типов .NET, которые по большей части находятся в пространстве имен System.Windows. Если ранее вы создавали приложения с помощью .NET, используя управляемые технологии, такие как ASP.NET и Windows Forms, основные принципы программирования с помощью WPF должны быть вам знакомы: вы создаете экземпляры классов, задаете свойства, вызываете методы и обрабатываете события — все это с использованием своего любимого языка программирования .NET, например C# или Visual Basic.

WPF включает в себя дополнительные конструкции программирования, которые расширяют возможности свойств и событий: свойства зависимостей и перенаправленные события.

XAML — это язык разметки на основе XML, который служит для определения внешнего вида приложения в декларативной форме. Обычно он используется для создания окон, страниц и пользовательских элементов управления, а также их заполнения элементами управления, фигурами и графическими элементами.

# СУБД SQL server

Microsoft SQL Server — система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной 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. SQL Server используется для хранения и обработки данных в приложениях и веб-сайтах, он поддерживает различные языки программирования и интерфейсы. Центральным аспектом в MS SQL Server, как и в любой СУБД, является база данных. База данных представляет хранилище данных, организованных определенным способом. Нередко физически база данных представляет файл на жестком диске, хотя такое соответствие необязательно. Для хранения и администрирования баз данных применяются системы управления базами данных (database management system) или СУБД (DBMS).

# Microsoft SQL Server Management Studio

SQL Server Management Studio (SSMS) - это интегрированная среда для управления любой инфраструктурой SQL, от SQL Server до базы данных SQL Azure. SSMS предоставляет инструменты для настройки, мониторинга и администрирования экземпляров SQL Server и баз данных. Используйте SSMS для развертывания, мониторинга и обновления компонентов уровня данных, используемых вашими приложениями, а также для создания запросов и сценариев.

Главным инструментом SQL Server Management Studio является Object Explorer, который позволяет пользователю просматривать, извлекать объекты сервера, а также полностью ими управлять.

Также есть SQL Server Management Studio Express для Express версии сервера, которая является бесплатной. Однако в ней нет поддержки ряда компонентов (Analysis Services, Integration Services, Notification Services, Reporting Services) и SQL Server 2005 Mobile Edition.

# Entity Framework

Entity Framework — объектно-ориентированная технология доступа к данным, является object-relational mapping (ORM) решением для .NET Framework от Microsoft. Предоставляет возможность взаимодействия с объектами как посредством LINQ в виде LINQ to Entities, так и с использованием Entity SQL. Для облегчения построения web-решений используется как ADO.NET Data Services (Astoria), так и связка из Windows Communication Foundation и Windows Presentation Foundation, позволяющая строить многоуровневые приложения, реализуя один из шаблонов проектирования MVC, MVP или MVVM. Первая версия Entity Framework - 1.0 вышла еще в 2008 году и представляла очень ограниченную функциональность, базовую поддержку ORM (object-relational mapping - отображения данных на реальные объекты) и один единственный подход к взаимодействию с бд - Database First. С выходом версии 4.0 в 2010 году многое изменилось - с этого времени Entity Framework стал рекомендуемой технологией для доступа к данным, а в сам фреймворк были введены новые возможности взаимодействия с бд - подходы Model First и Code First.

Дополнительные улучшения функционала последовали с выходом версии 5.0 в 2012 году. И наконец, в 2013 году был выпущен Entity Framework 6.0, обладающий возможностью асинхронного доступа к данным.

Центральной концепцией Entity Framework является понятие сущности или entity. Сущность представляет набор данных, ассоциированных с определенным объектом. Поэтому данная технология предполагает работу не с таблицами, а с объектами и их наборами.

Любая сущность, как и любой объект из реального мира, обладает рядом свойств. Например, если сущность описывает человека, то мы можем выделить такие свойства, как имя, фамилия, рост, возраст, вес. Свойства необязательно представляют простые данные типа int, но и могут представлять более комплексные структуры данных. И у каждой сущности может быть одно или несколько свойств, которые будут отличать эту сущность от других и будут уникально определять эту сущность. Подобные свойства называют ключами.

# ГЛАВА 2. ПРОЕТИРОВАНИЕ БАЗЫ ДАННЫХ

# 2.1 Разработка диаграммы ERD

**Описание:**

Схема «сущность-связь» (также ERD или ER-диаграмма) — это разновидность блок-схемы, где показано, как разные «сущности» (люди, объекты, концепции и так далее) связаны между собой внутри системы, ER-диаграммы чаще всего применяются для проектирования и отладки реляционных баз данных в сфере образования, исследования и разработки программного обеспечения и информационных систем для бизнеса. ERдиаграммы (или ER-модели) полагаются на стандартный набор символов, включая прямоугольники, ромбы, овалы и соединительные лини для отображения сущностей, их атрибутов и связей. Эти диаграммы устроены по тому же принципу, что и грамматические структуры: сущности выполняют роль существительных, а связи - глаголов.

B ER-моделях и моделях данных обычно выделяют до трех уровней детализации:

* **Концептуальная модель данных** - схема наивысшего уровня с минимальным количеством подробностей. Достоинство этого подхода заключается в возможности отобразить общую структуру модели и всю архитектуру системы. Менее масштабные системы могут обойтись и без этой модели. В этом случае можно сразу переходить к логической модели.
* **Логическая модель** **данных** содержит более подробную информацию, нежели концептуальная модель. На этом уровне определяются более подробные операционные и транзакционные сущности. Логическая модель не зависит от технологии, в которой она будет применяться.
* **Физическая модель данных**: на основе каждой логической модели данных можно составить одну или две физических модели. В последних должно присутствовать достаточно технических подробностей для составления и внедрения самой базы данных.

**Элементы:**

Для того чтобы построить ER-диаграмму, можно использовать разные нотации:

* Нотация IDEF1X. Ее относят к фундаментальным, но на практике давно не используют, потому что есть более удобные варианты.
* Нотация Чена. Классическая нотация, которая состоит из простых символов – прямоугольников, овалов и линий. Из-за этого нотацию часто используют для концептуальных моделей, которые презентуют заказчику.
* Нотация Мартина. Ее еще называют «воронья лапка» (от англ. Crow's Foot). Она компактнее нотации Чена, поэтому ее используют для построения ER-моделей логического уровня, когда нужно описать в модели все атрибуты сущностей.

В нотациях Чена и Мартина есть одинаковые элементы: сущности, атрибуты и связи. Но эти элементы диаграмм обозначают разными символами.

Элементы ER-диаграммы в нотации Чена соединяют линиями. Если линия соединяет две сущности, сверху обозначают тип связи:

● 1:1 — «один-к-одному»;

● 1:N — «один-ко-многим»;

● M:N — «многие-ко-многим».

В нотации Мартина сущность также вписывают в прямоугольник, а атрибуты и связи обозначают по-другому:

● атрибуты перечисляют прямо под сущностью,

● связи рисуют разными соединительными линиями (рис. 2.1).

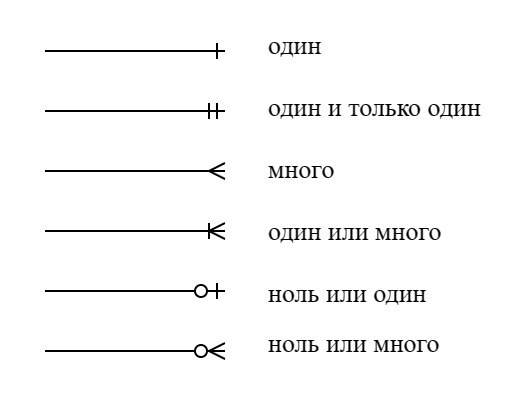


Рисунок 2.1.1 Виды соединительных линий нотации Мартина

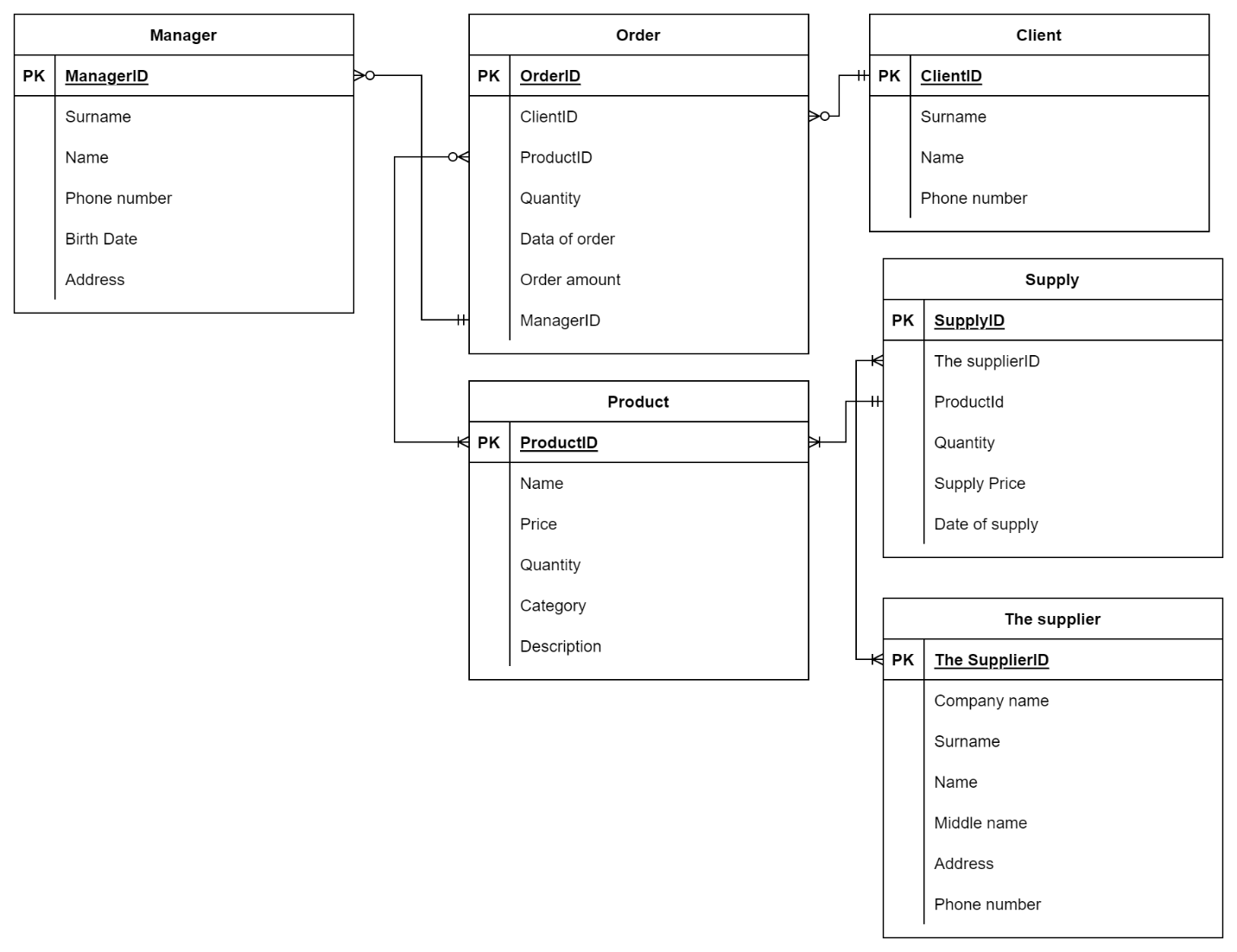


Рисунок 2.1.2 Диаграмма «Сущность-связь» (Entity-Relationship diagram)

На рис. 2.1.2 показана ER-диаграмма информационной системы «Компьютерный магазин». В данной диаграмме используются такие сущности как: Менеджер (Manager), Клиент (Client), Заказ (Order), Продукт (Product), Поставка (Supply), Поставщик (The supplier). Каждая из сущностей имеет свои атрибуты.

# 2.2 Разработка базы данных

SQL Server Management Studio (SSMS) - это интегрированная среда для управления любой инфраструктурой SQL, от SQL Server до базы данных SQL Azure. SSMS предоставляет инструменты для настройки, мониторинга и администрирования экземпляров SQL Server и баз данных. Используйте SSMS для развертывания, мониторинга и обновления компонентов уровня данных, используемых вашими приложениями, а также для создания запросов и сценариев.

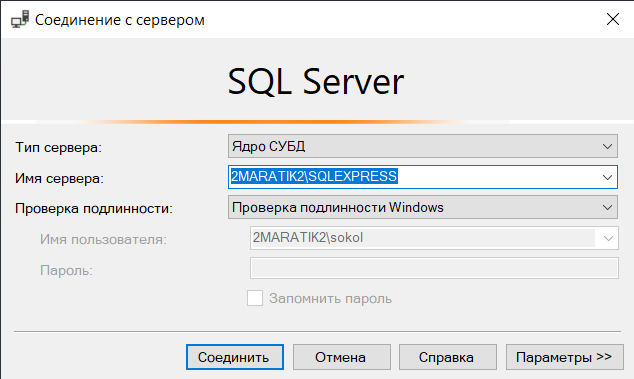


Рисунок 2.2.1 Соединение с сервером

Рисунок 2.2.1 Окно подключения к локальному серверу

После чего у нас станет доступна таблица Object Explorer в которой будут показаны файлы и базы данных на локальном сервере. Создадим базу данных, нажимаем ПКМ по папке «Базы данных» и выбираем «Создать базу данных».

# 

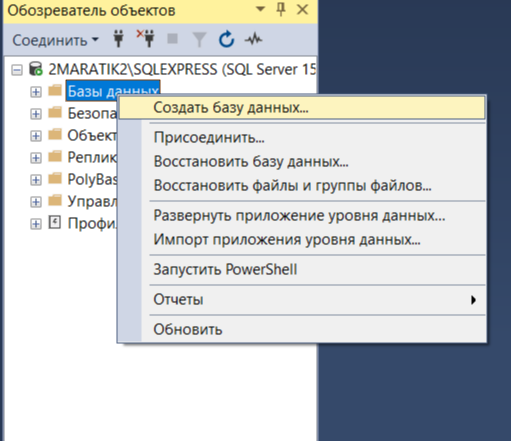


Рисунок 2.2.2 Создание базы данных

Далее нам выводиться окно, в котором мы даём название для базы данных и нажимаем на кнопку ОК. В Object Explorer появиться созданная база данных, развернём её, нажмём ПКМ по папке «Диаграмма баз данных» и нажмем на «Создать диаграмму базы данных».

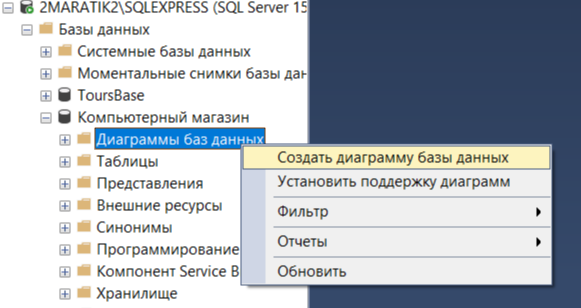


Рисунок 2.2.3 Создаем диаграмму базы данных

После чего нам выведется поле для создания диаграммы базы данных, на котором мы можем создавать таблицы с название столбцов и их типом данных.

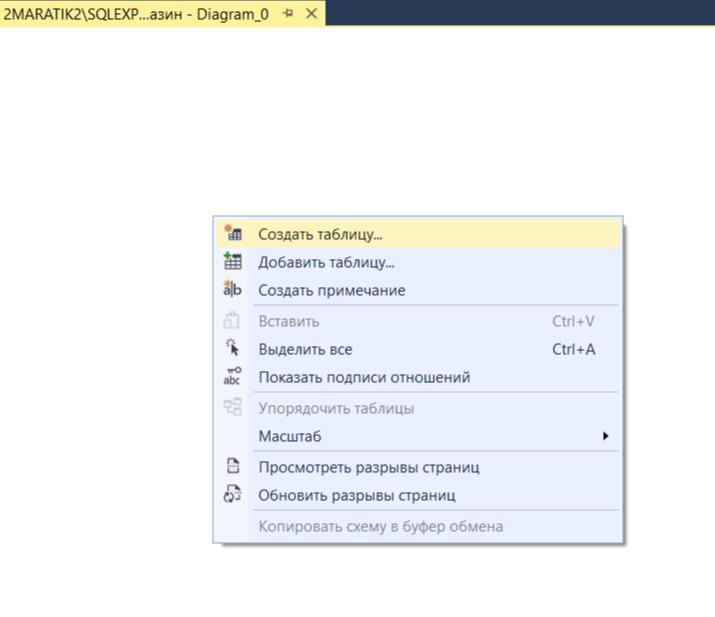


Рисунок 2.2.4 Создаем таблицы в диаграмме базы данных

Для создания таблицы данных необходимо нажать ПКМ в любом месте диаграммы, выбираем «Создать таблицу» и даём название для нашей таблицы. Для информационной системы необходимо создать пять таблиц: Менеджер (Manager), Клиент (Client), Заказ (Order), Продукт (Product), Поставка (Supply), Поставщик (The supplier). После чего заполним таблицы необходимыми полями и типами данных из ERD таблицы.

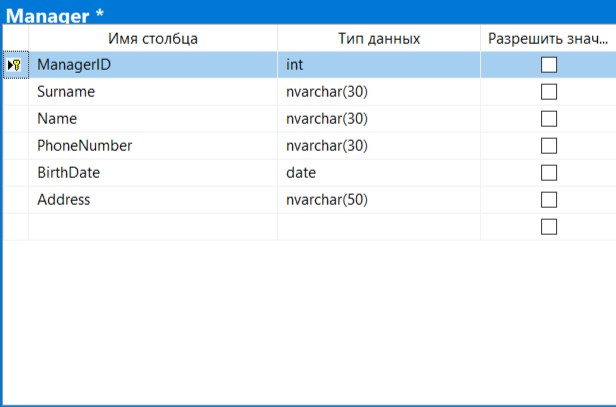


Рисунок 2.2.5 Задаем имена столбцов и указываем типы данных в таблице

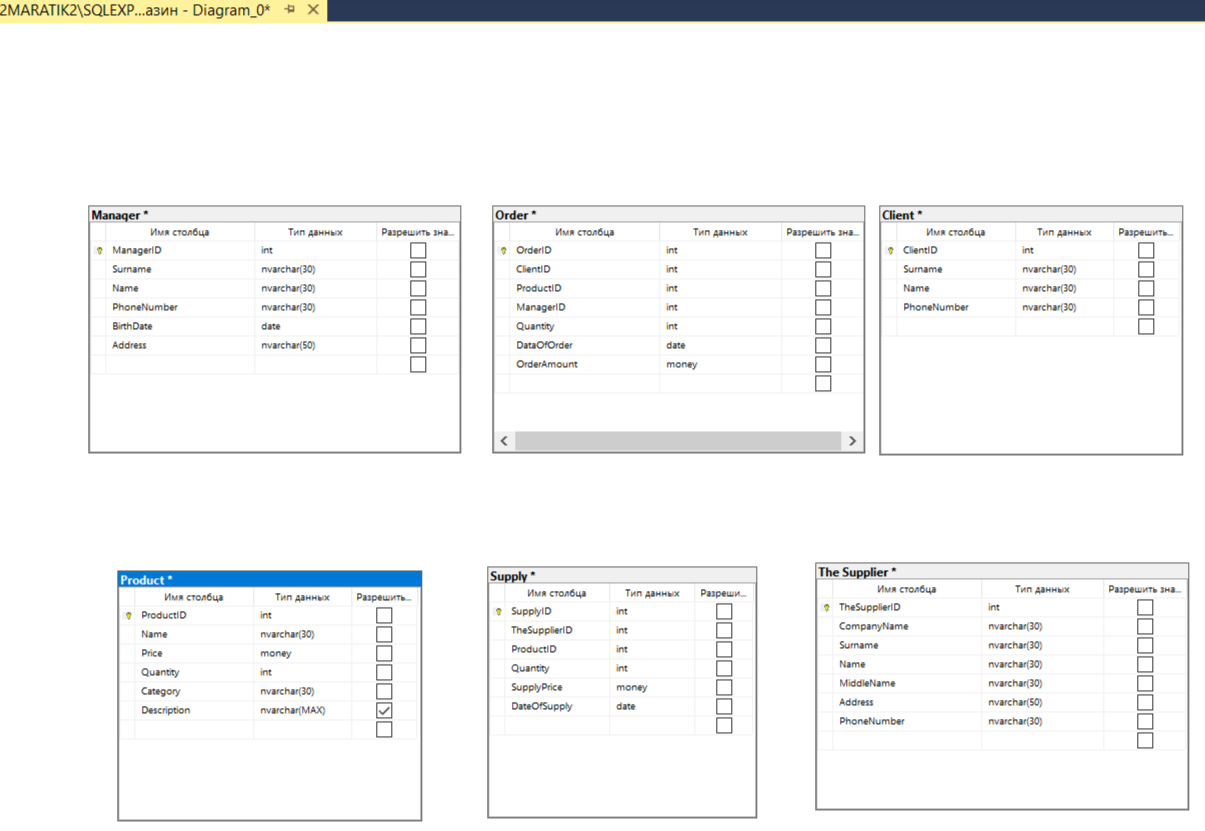


Рисунок 2.2.6 Таблицы диаграммы БД

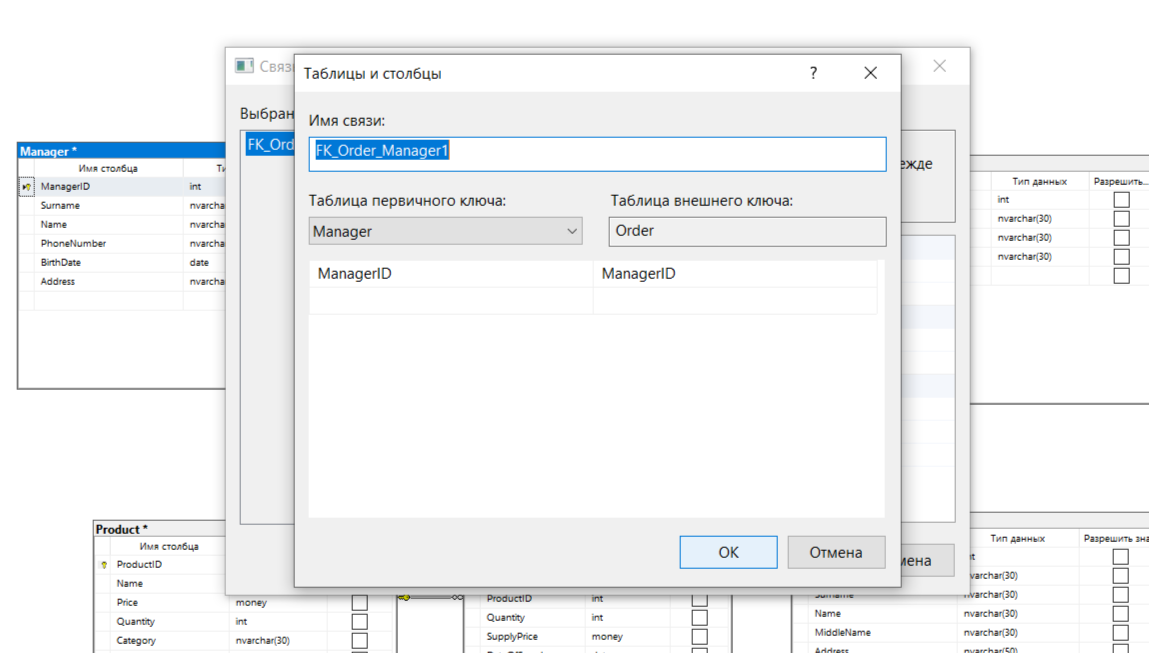


Рисунок 2.2.7 Создаем связи между атрибутами.

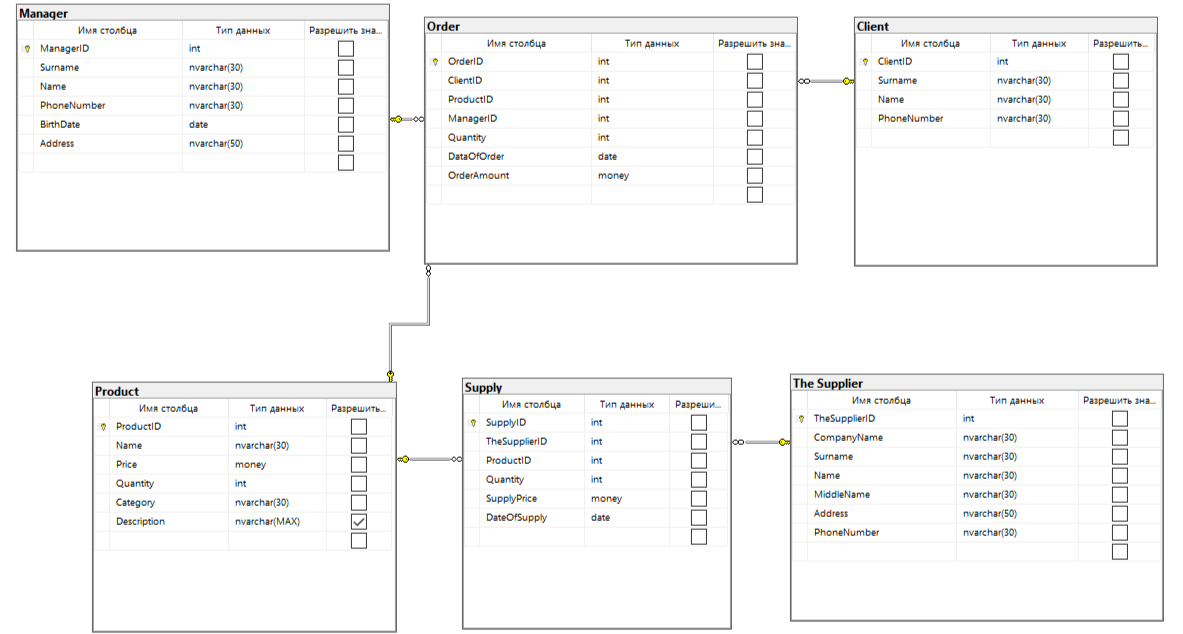


Рисунок 2.2.8 Готовые таблицы диаграммы БД

После создания связей наша таблица полностью готова.

В данной работе используется 6 таблиц базы данных. Manager содержит Уникальный ключ, Фамилию, Имя, Номер телефона, Дату рождения и адрес. Client содержит Уникальный ключ, Имя, Фамилию, Номер телефона. Order содержит в себе следующие атрибуты: Уникальный ключ, Код клиента, Код продукта, Код менеджера, Количество, Дату и Сумму заказа. Product содержит Уникальный ключ, Наименование, Цену, Количество, Категорию, Описание. Supply содержит Уникальный ключ, Код поставщика, Код продукта, Количество, Цену поставки и Дату поставки. The Supplier содержит Уникальный ключ, Название компании, Фамилия, Имя, Сокращенное имя, Адрес и Номер телефона.

# ГЛАВА 3. РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ

# 3.1 Разработка прототипа информационной системы

Проволочная диаграмма представляет собой наглядный детализированный макет пользовательского интерфейса – своеобразный рабочий чертеж, описывающий функциональные возможности и содержимое приложения. Такие проволочные шаблоны для веб-сайтов и мобильных приложений идеально подходят для создания нестрогих проектных эскизов, демонстрирующих идеи и позволяющих достичь единого мнения в команде и сформировать основу для разработки высококачественной каркасной модели.

Для каждого шаблона можно выбрать пустую схему для быстрого начала или пустой схемы. Каждый шаблон имеет пять элементов (контейнеры, элементы управления, мультимедиа, текст и навигация) с большим количеством фигур. Контейнеры предоставляют предварительно созданные группы фигур, представляющих различные диалоговые окно и карточки, что помогает быстро создать проволочное представление диаграммы. Проволочная фигура также включает интеллектуальные фигуры, например ползунок диапазона или ползунок хода выполнения, которые можно настроить с помощью пунктов управления и действий в меню, щелкнув правой кнопкой мыши, чтобы изменить состояние, значения или визуальные стили. Эти шаблоны являются универсальными и могут использоваться для разных браузеров и мобильных устройств.

# 

Рисунок 3.1.1 Разработка формы окна «Авторизация»

На рисунке 3.1.1 показано окно «Авторизация», включающее в себя поле для логина и пароля, кнопку «Войти». После правильного ввода логина и пароля, мы попадаем на главное окно.

# 

Рисунок 3.1.2 Разработка формы окна «Главное окно»

На рисунке 3.1.2 показана главная страница информационной системы, которая является связующим звеном всех последующих страниц. Она включает в себя кнопки переходов на разные страницы и кнопку перехода к авторизации.

# 

Рисунок 3.1.3 Разработка формы окон «Клиенты, продажа, товары»

На рисунке 3.1.3 показано окно форм «Клиенты, продажи, товарф», включающее в себя ряд полей, содержащих различные данные; кнопки переходов на разные страницы и кнопку перехода к авторизации; кнопки добавления, изменения и удаления.

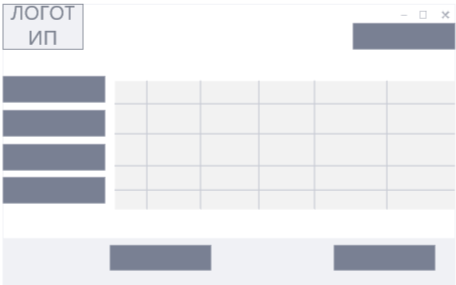


Рисунок 3.1.4 Разработка формы окна «Отчеты»

На рисунке показано окно «Отчеты», включающее в себя ряд полей, содержащих данные о продуктах(товарах) (номер продукта, название, цену, категорию, описание и количество); кнопки переходов на разные страницы и кнопку перехода к авторизации; кнопки сохранения в Word и Excel.

# 3.2 Программирование информационной системы

# 3.2.1 Разработка модуля «Авторизация»

# 3.2.2 Разработка модуля …..

# 3.2.3 Разработка модуля …..

# 3.2.4 Разработка модуля «Отчет»

# ЗАКЛЮЧЕНИЕ

При разработке курсового проекта, а именно создание автоматизированной информационной системы «Компьютерный магазин», была спроектирована диаграмма Entity-Relationship diagram. Автоматизированная информационная система была разработана для сотрудников (администратор системы, менеджеры и работник склада). С ее помощью можно быстро добавлять, изменять и удалять данные в необходимых отчетах; просматривать наличие товаров; отслеживать заказы клиентов.

В конечном итоге данный программный продукт упрощает работу персонала, улучшает обслуживание клиентов и помогает автоматизировать продажи, повысить точность учета и контроля за товарами.

В данной курсовой работе я определила нужный инструментарий, спроектировала базу данных «Компьютерный магазин» в интегрированной среде SSMS (SQL Server Management Studio) и разработала прототип и интерфейс информационной системы с помощью языка C# и API-интерфейса WPF.

Ссылка на репозиторий: <https://gitea.com/burichevaaaaaa/KP.git>

# СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ