**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 «Автосалон»**

Выполнил студент гр. ИСП-21/\_\_\_­\_\_/Тельтевский Николай Владимирович/

(подпись) (Ф.И.О.)

**ОЦЕНКА:\_\_\_\_\_\_\_\_\_\_\_\_\_**

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

**ПРОВЕРИЛ**

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

(подпись) (Ф.И.О.)

Воркута

2024

**Содержание**

[ВВЕДЕНИЕ 3](#_Toc166506898)

[ГЛАВА 1. ВЫБОР ИНСТРУМЕНТАРИЯ 4](#_Toc166506899)

[**1.1 СУБД SQL server 4**](#_Toc166506900)

[**1.2 Microsoft SQL Server Management Studio 6**](#_Toc166506901)

[**1.3 Entity Framework 7**](#_Toc166506902)

[ГЛАВА 2. ПРОЕТИРОВАНИЕ БАЗЫ ДАННЫХ 9](#_Toc166506903)

[**2.1 Разработка диаграммы ERD 9**](#_Toc166506904)

[**2.2 Разработка базы данных. 11**](#_Toc166506905)

[ГЛАВА 3. РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ 17](#_Toc166506906)

[**3.1 Разработка прототипа информационной системы 17**](#_Toc166506907)

[**3.2 Программирование информационной системы 19**](#_Toc166506908)

[**3.2.1 Разработка модуля «Авторизация» 19**](#_Toc166506909)

[**3.2.2 Разработка модуля «Добавление» 19**](#_Toc166506910)

[**3.2.3 Разработка модуля «Удаление» 21**](#_Toc166506911)

[**3.2.4 Разработка модуля «Отчет» 21**](#_Toc166506912)

[ЗАКЛЮЧЕНИЕ 25](#_Toc166506913)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 26](#_Toc166506914)

# **ВВЕДЕНИЕ**

В современном мире автомобильная отрасль является одной из ключевых . Создание информационной системы для автосалона позволит улучшить управление клиентской базой, автоматизировать процессы продажи и обслуживания автомобилей, а также повысить уровень обслуживания клиентов. Это будет стратегически важно для автосалонов в условиях активной конкуренции и постоянно меняющихся технологий.

Данная информационная система поможет управлять информацией о доступных автомобилях, заказах, клиентах и других аспектах, связанных с продажей автомобилей. Она обеспечит эффективное управление бизнесом, улучшит сервис для клиентов и повысит общую производительность.

**Объект:** информационная система «Автосалон».

**Предмет:** автоматизация бизнес-процесса работы автосалона.

**Цель работы -**  разработать базу данных и информационную систему «Автосалон».

**Задачи:**

* выбрать инструментарий;
* спроектировать базу данных;
* сделать разграничение прав доступа базы данных;
* создать информационную систему.

# **ГЛАВА 1. ВЫБОР ИНСТРУМЕНТАРИЯ**

## **1.1 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.

СУБД SQL server используются для создания, размещения, хранения и управления реляционными (табличными) базами данных на специальных серверах или в облаке. Они работают через настольные приложения и web-сайты. К основным преимуществам их функционирования относятся:

* высокоскоростной доступ к данным, обеспечиваемый надежной клиент-серверной архитектурой СУБД;
* простота работы и администрирования, обусловленные понятной структурой языка программирования SQL;
* безопасность хранения информации в БД - благодаря возможности шифрования данных и резервного копирования;
* Транзакционная обработка данных. По каждому запросу от СУБД обрабатывается и сохраняется небольшое количество информации.

Специфика работы сервера базы данных SQL server заключается в транзакционной обработке данных. Это означает, что по каждому запросу от СУБД обрабатывается и сохраняется небольшое количество информации.

Применение SQL server позволяет автоматизировать решение различных бизнес-задач, поддерживать проведение аналитики данных в режиме онлайн, отслеживать направление ресурсов СУБД, управлять транзакциями (операциями по обработке данных).

Для реализации функций СУБД на сегодняшний день чаще всего используются следующие SQL-серверы:

MS SQL server - многопользовательский программный продукт, разработанный компанией Microsoft, обладающий высокой производительностью и отказоустойчивостью, тесно интегрированный с ОС Windows. Этот сервер поддерживает удаленные подключения, работает с многими популярными типами данных, дает возможность создавать триггеры и хранимые данные, имеет практичные и удобные утилиты для настройки;

Oracle Database server - СУБД, предназначенная для создания, консолидации и управления базами данных в облачной среде. Используя этот сервер, можно как автоматизировать обычные бизнес-операции, так и выполнять динамический многомерный анализ данных (OLAP), проводить операции с документами xml-формата и управлять разделенной и локальной информацией;

IBM DB2 - семейство СУБД для работы с реляционными базами данных, признанное самым производительным, имеющим высокие технические показатели и возможности масштабирования. SQL-серверы этой группы характеризуются мультиплатформенностью, способностью к мгновенному созданию резервных копий и восстановлению БД, реорганизации таблиц в онлайн-режиме, разбиению баз данных, определению пользователями новых типов данных;

MySQL - СУБД, разработанная и поддерживаемая компанией Oracle. В основном она используется локальными или удаленными клиентами, позволяя им работать с таблицами разных типов, поддерживающих полнотекстовый поиск или выполняющих транзакции на уровне отдельных записей;

PostgreSQL - СУБД с открытым исходным кодом, работающая с объектно-реляционными (поддерживающими пользовательские объекты) базами данных. Также PostgreSQL предназначена для создания, хранения и извлечения сложных структур данных. Она поддерживает самые различные типы данных (среди них - числовые, текстовые, булевы, денежные, бинарные данные, сетевые адреса, xml и другие).

## **1.2 Microsoft SQL Server Management Studio**

Microsoft SQL Server Management Studio (SSMS) — это программное приложение, разработанное корпорацией Майкрософт, которое используется для настройки, управления и администрирования всех компонентов Microsoft SQL Server. Впервые запущенный в Microsoft SQL Server 2005, он является преемником Enterprise Manager в SQL 2000 или более ранних версиях. Инструмент включает в себя как редакторы скриптов, так и графические инструменты, которые работают с объектами и функциями сервера.

Центральной функцией SSMS является обозреватель объектов, который позволяет пользователю просматривать, выбирать и работать с любыми объектами на сервере. Он также поставлял отдельное экспресс-издание, которое можно было свободно загрузить; однако последние версии SSMS полностью поддерживают подключение к любому экземпляру SQL Server Express и управление им. Корпорация Майкрософт также включила обратную совместимость для более старых версий SQL Server, что позволило более новой версии SSMS подключаться к более старым версиям экземпляров SQL Server. Он также поставляется с Microsoft SQL Server Express 2012, или пользователи могут загрузить его отдельно.

В среде SQL Server Management Studio доступны следующие функции:

* Создание баз данных SQL Server, сценариев SQL и заданий агента SQL Server с помощью SQL Server;
* Подключение к экземплярам базы данных SQL — объекты базы данных SQL, такие как таблицы, индексы и схемы, можно создавать, изменять и удалять;
* Экземплярами базы данных SQL можно управлять и отслеживать.

Он также используется для создания запросов и скриптов, а также для развертывания, мониторинга и обновления компонентов уровня данных, используемых приложениями.

SQL Server Management Studio — это веб-программа и клиентская программа, которая позволяет управлять базами данных и хранилищами данных на локальном компьютере или в облаке. Он также позволяет управлять средой SQL Server путем создания, изменения и удаления объектов, администрирования баз данных и мониторинга среды SQL Server.

SSMS также предоставляет платформу, которая позволяет администраторам и разработчикам SQL использовать различные мощные редакторы скриптов. В двух словах, это упрощает обслуживание и администрирование SQL Server и связанных с ним экземпляров, чтобы вы могли сосредоточиться на своих приложениях.

SSMS позволяет администраторам выполнять следующие задачи:

* Подключение к базам данных, данным и экземплярам SQL Server;
* Конфигурацию экземпляров SQL Server можно просматривать и изменять;
* Управление подключениями и разрешениями в SQL Server;
* Создание новых баз данных, пользователей и ролей, а также выполнение других административных обязанностей;
* Доступ к журналам ошибок, системным базам данных и двоичным файлам экземпляра SQL Server.

## **1.3 Entity Framework**

Entity Framework — это набор технологий в ADO.NET, которые поддерживают разработку приложений программного обеспечения, ориентированных на данные. Архитекторам и разработчикам приложений, ориентированных на обработку данных, приходится учитывать необходимость достижения двух совершенно различных целей. Они должны моделировать сущности, связи и логику решаемых бизнес-задач, а также работать с ядрами СУБД, используемыми для сохранения и получения данных. Данные могут распределяться по нескольким системам хранения данных, в каждой из которых применяются свои протоколы, но даже в приложениях, работающих с одной системой хранения данных, необходимо поддерживать баланс между требованиями системы хранения данных и требованиями написания эффективного и удобного для обслуживания кода приложения.

Entity Framework позволяет разработчикам работать с данными в виде объектов и свойств, относящихся к домену, таких как клиенты и адреса клиентов, без необходимости беспокоиться о базовых таблицах и столбцах базы данных, где хранятся эти данные. Entity Framework дает разработчикам возможность работать с данными на более высоком уровне абстракции, создавать и сопровождать приложения, ориентированные на работу с данными, одновременно с этим сокращая объем кода по сравнению с традиционными приложениями. Так как Entity Framework является компонентом платформа .NET Framework, приложения Entity Framework могут выполняться на любом компьютере, на котором установлен платформа .NET Framework начиная с версии 3.5 с пакетом обновления 1 (SP1).

Entity Framework обеспечивает жизнь моделям, позволяя разработчикам запрашивать сущности и связи в модели предметной области (называемой концептуальной моделью в Entity Framework), опираясь на Entity Framework для перевода этих операций в команды источника данных. Это позволяет отказаться от применения в приложениях жестко заданных зависимостей от конкретного источника данных.

Платформа Entity Framework — это не просто еще одно средство объектно-реляционного сопоставления. Ее цель — предоставить приложениям возможность чтения и изменения данных, представленных в виде сущностей и связей в концептуальной модели. Entity Framework использует сведения в модели и файлах сопоставления для преобразования запросов объектов к типам сущностей, представленным в концептуальной модели, в запросы, относящиеся к источнику данных. Результаты запроса материализуются в объекты, которым управляет Entity Framework.

# **ГЛАВА 2. ПРОЕТИРОВАНИЕ БАЗЫ ДАННЫХ**

## **2.1 Разработка диаграммы ERD**

Описание:

ER-модель (от англ. Entity-Relationship model, модель «сущность — связь») — модель данных, позволяющая описывать концептуальные схемы предметной области.

ER-модель используется при высокоуровневом (концептуальном) проектировании баз данных. С её помощью можно выделить ключевые сущности и обозначить связи, которые могут устанавливаться между этими сущностями.

Во время проектирования баз данных происходит преобразование схемы, созданной на основе ER-модели, в конкретную схему базы данных на основе выбранной модели данных (реляционной, объектной, сетевой или др.).

ER-модель представляет собой формальную конструкцию, которая сама по себе не предписывает никаких графических средств её визуализации. В качестве стандартной графической нотации, с помощью которой можно визуализировать ER-модель, была предложена диаграмма «сущность-связь».

Понятия «ER-модель» и «ER-диаграмма» часто не различают, хотя для визуализации ER-моделей могут быть использованы и другие графические нотации, либо визуализация может вообще не применяться (например, использоваться текстовое описание).

Элементы:

Сущность — представляет объект или сущность, которая будет храниться в базе данных как таблица. Обычно изображается в виде прямоугольника с названием сущности внутри.

Атрибут — представляет информацию о сущности. Он хранится в виде столбца в таблице базы данных. Обычно изображается как овал, присоединенный к сущности линией.

Связь — показывает связь между двумя сущностями. Она может быть однонаправленной или двунаправленной и иметь атрибуты, определяющие характер связи. Обычно изображается линией, соединяющей две сущности, с названием связи и типом связи.

Кардинальность — определяет количество связанных сущностей в отношении. Например, один ко многим, многие ко многим и т.д. Он обычно отображается в виде числовых значений рядом с линиями связей.

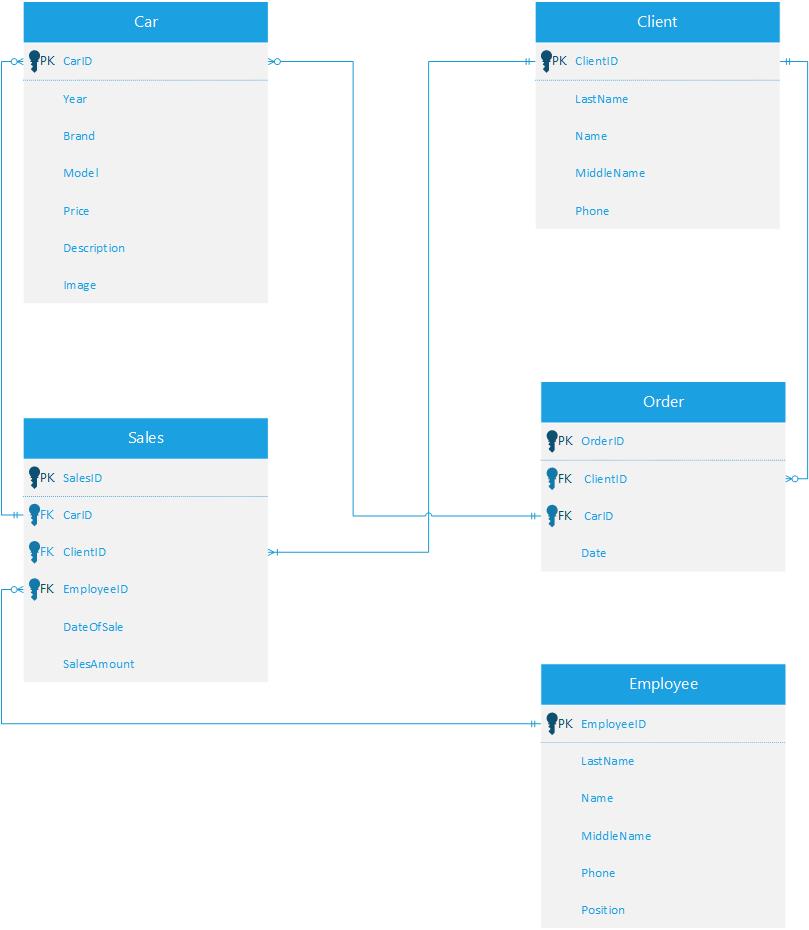
Ключ — определяет уникальный идентификатор сущности. Обычно обозначается подчеркнутым атрибутом.

Рисунок 2.1. Диаграмма ERD

В данной диаграмме используются такие сущности как Клиент(Client), Заказ(Order), Продажи(Sales), Автомобиль(Car), Сотрудник(Employee).

Сущность Клиент(Client) имеет следующие атрибуты: Client\_ID(Уникальный ключ), last\_name(Фамилия), name(Имя), phone(Номер телефона).

Сущность Автомобиль(Car) имеет следующие атрибуты: Car\_ID(Уникальный ключ), year(Год), brand(Бренд), model(Модель), price(Цена), description(Описание), image(Изображение).

## **2.2 Разработка базы данных.**

SQL Server Management Studio – это набор административных средства для управления компонентами, относящимися к SQL Server. Эта интегрированная среда позволяет пользователям выполнять разнообразные задачи, например резервное копирование данных, редактирование запросов и автоматизацию общих функций в одном интерфейсе.

Среда SQL Server Management Studio включает в себя следующие средства:

Редактор кода – богатый возможностями редактор скриптов для написания и редактирования скриптов. Среда SQL Server Management Studio предоставляет четыре версии редактора кода: редактор запросов ядра СУБД для скриптов Transact-SQL, редактор запросов многомерных выражений, редактор запросов расширения интеллектуального анализа данных и редактор запросов XML/A.

Обозреватель объектов для размещения, изменения, создания скрипта или выполнения объектов, принадлежащих экземплярам SQL Server.

Обозреватель шаблонов для размещения и написания сценариев шаблонов.

Обозреватель решений для организации и хранения связанных скриптов как части проекта.

Окно свойств для отображения текущих свойств выбранных объектов.

Среда SQL Server Management Studio обеспечивает эффективность рабочих процессов, предоставляя:

Отключенный доступ. Можно писать и изменять скрипты, не соединяясь с экземпляром SQL Server.

Создание сценариев из любого диалогового окна. Можно создать скрипт и любого диалогового окна, а также читать, изменять, сохранять и многократно использовать скрипты после создания.

Немодальные диалоговые окна. При обращении к диалоговому окну интерфейса можно посмотреть другие ресурсы в среде SQL Server Management Studio, не закрывая диалоговое окно.

Разработка базы данных начинается с ранее разработанной ERD диаграммой. Откроем Microsoft SQL Server Management Studio и подключаемся к нашему локальному серверу.

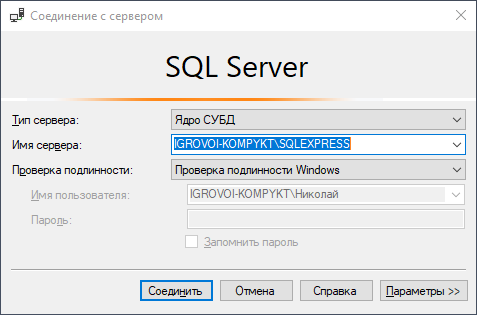


Рисунок 2.2. Окно подключения к локальному серверу

После чего у нас станет доступна таблица «Обозреватель объектов», 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».

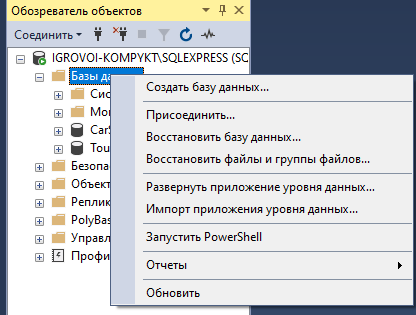


Рисунок 2.3. Создание базы данных

Далее нам выводиться окно, в котором мы даём название для базы данных и нажимаем на кнопку OK. В «Обозреватель объектов» появиться созданная база данных, развернем её нажмём ПКМ по папке «Диаграммы баз данных» и нажмём на «Создать диаграмму базы данных».

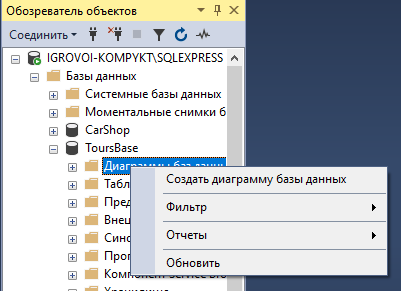


Рисунок 2.4. Создание диаграммы

После чего нам выведется поле для создания диаграммы базы данных, на котором мы можем создавать таблицы с названием столбцов и их типов 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: Клиент (Client), Продавец (Employees), Лекарственные препараты (Medicines), Заказ (Sales) и Типы лекарственных препаратов (MedicineTypes). После чего заполним таблицы необходимым полями и типами данных из ERD таблицы.

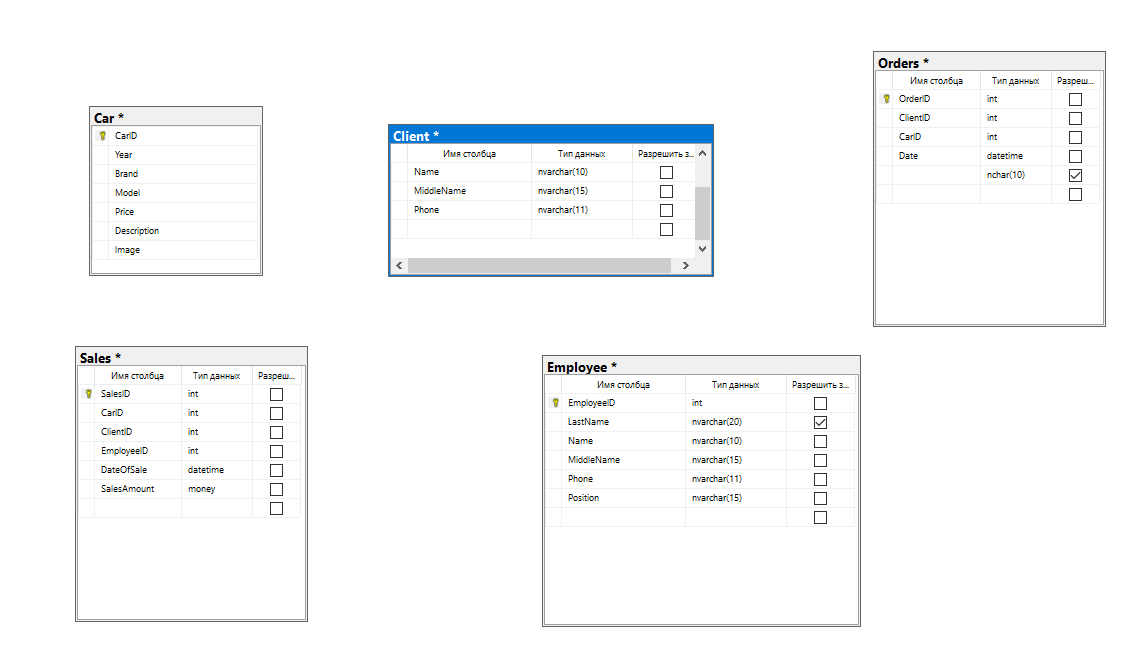


Рисунок 2.5. Все таблицы

Далее создаём связи как в ERD сущностях и после сохраняем таблицы нажав сочетание клавиш «Ctrl + S» и в высветившимся окне пишем название диаграммы.

Созданные таблицы можно увидеть в «Обозреватель объектов» кликнув на название базы данных и перейдя в папку «Таблицы». Зажав уникальный ключ, тянем к полю в нужной таблице и создается связь.

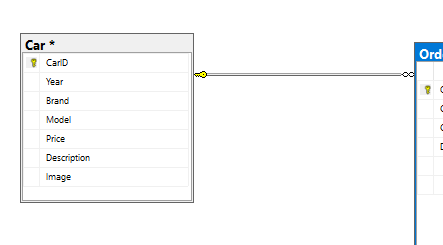


Рисунок 2.6. Создание связи

После создания связей наша таблица полностью готова.

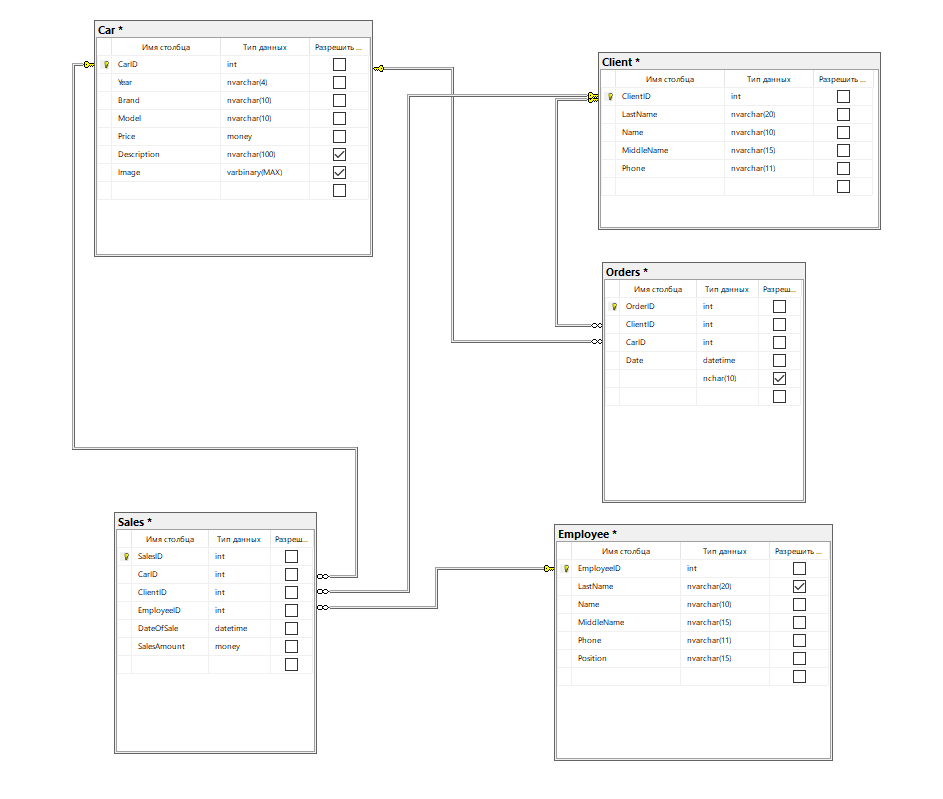


Рисунок 2.7. База данных

В данной работе используется 5 таблиц базы данных. Client содержит Уникальный ключ, Имя, Фамилия и Номер телефона. Employees содержит Уникальный ключ, Имя, Фамилия, Номер телефона, Должность. Car содержит Уникальный ключ, Марка, Модель, Год выпуска, Цена, Описание. Sales содержит Уникальный ключ, Уникальный ключ клиента, Уникальный ключ машины, Уникальный код продавца, Дату и Сумму продажи . Orders содержит, Уникальный ключ клиента, Уникальный ключ машины и Дату.

# **ГЛАВА 3. РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ**

## **3.1 Разработка прототипа информационной системы**

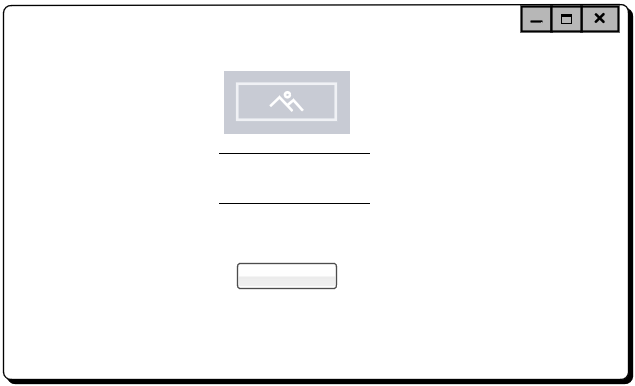


Рисунок 3.1. Окно входа

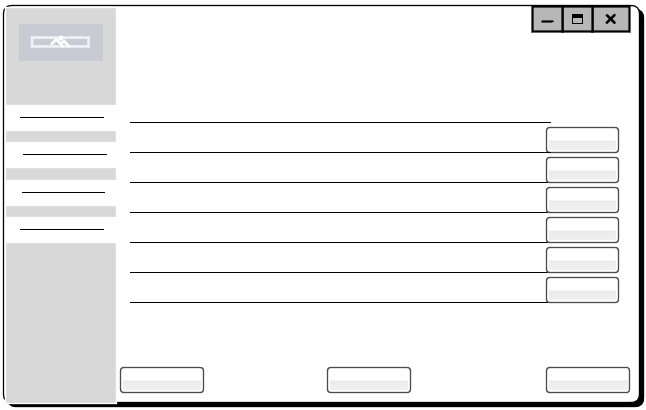


Рисунок 3.2. Окно сотрудника № 1

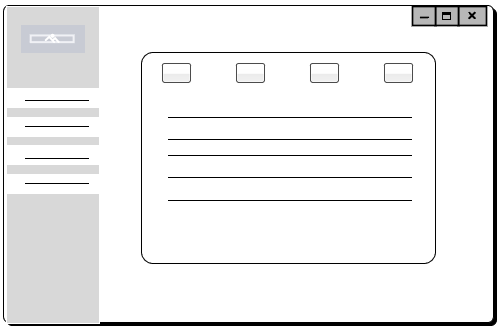


Рисунок 3.3. Окно сотрудника №2

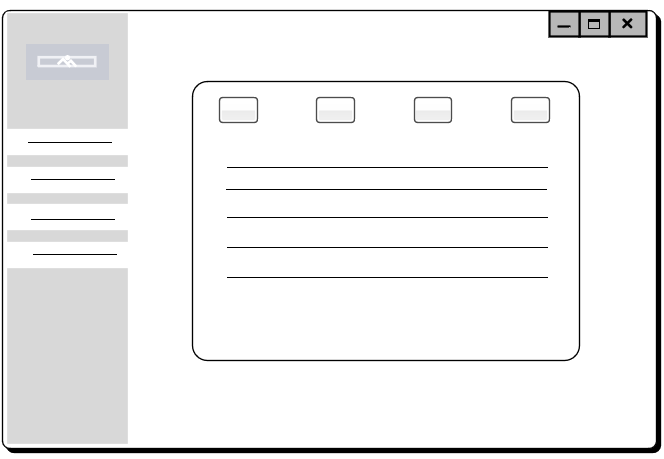


Рисунок 3.4. Окно сотрудника №3

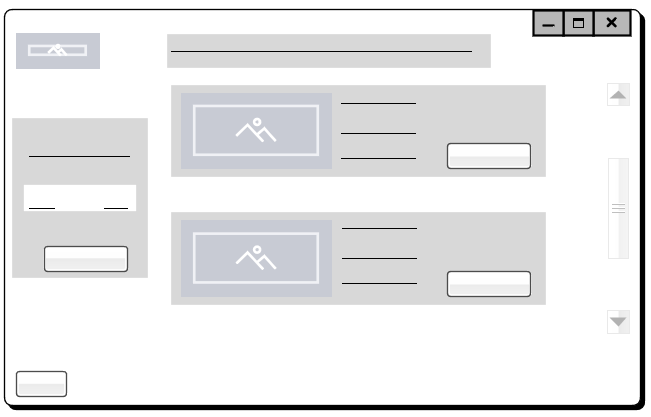


Рисунок 3.5. Окно клиента

## **3.2 Программирование информационной системы**

### **3.2.1 Разработка модуля «Авторизация»**

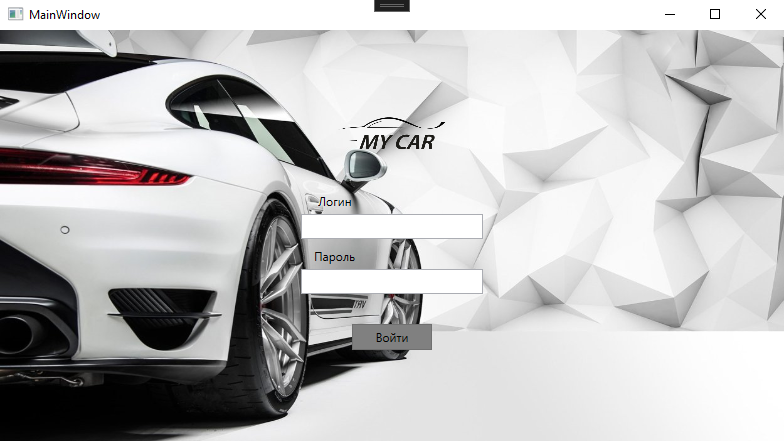
****

Рисунок 3.6. Авторизация

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void BtnLog\_Click(object sender, RoutedEventArgs e)

{

if (Login.Text == "Admin" && Password.Password == "admin")

{

Window1 AdminWindow = new Window1();

AdminWindow.Show();

}

else if (Login.Text == "User" && Password.Password == "user")

{

Window2 UserWindow = new Window2();

UserWindow.Show();

}

else

{

MessageBox.Show("Пароль или логин введен не верно!");

}

**3.2.2 Разработка модуля «Добавление»**

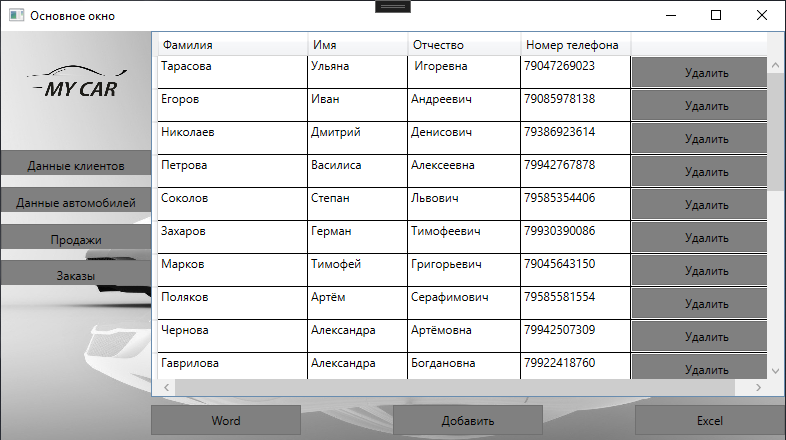


Рисунок 3.7. Добавление № 1

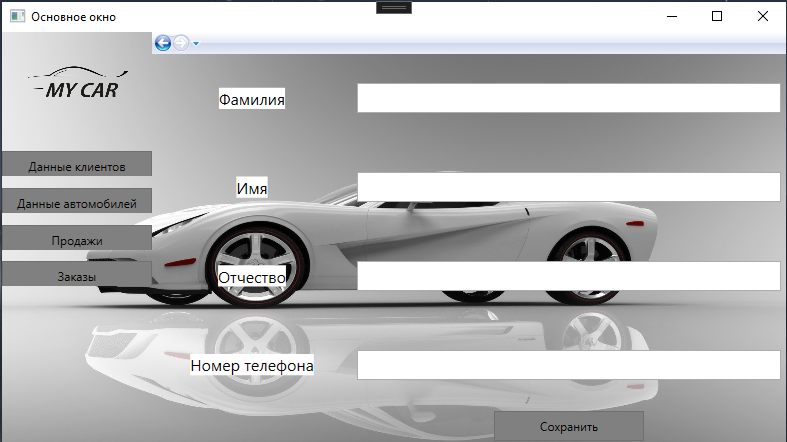


Рисунок 3.8. Добавление № 2

private void BtnSave\_Click(object sender, RoutedEventArgs e)

{

if (string.IsNullOrEmpty(LastName.Text) || string.IsNullOrEmpty(Name.Text) || string.IsNullOrEmpty(MiddleName.Text)

|| string.IsNullOrEmpty(Phone.Text))

{

MessageBox.Show("Заполните все поля!", "Уведомление");

return;

}

var query = $" insert into Client (LastName, Name, MiddleName, Phone) values( N'{LastName.Text}', N'{Name.Text}', N'{MiddleName.Text}', N'{Phone.Text}')";

if (dbc.Execute(query) != null)

{

MessageBox.Show("Успешно!", "Уведомление");

DB.sqlConnection.Close();

}

DB.sqlConnection.Close();

### }**3.2.3 Разработка модуля «Удаление»**

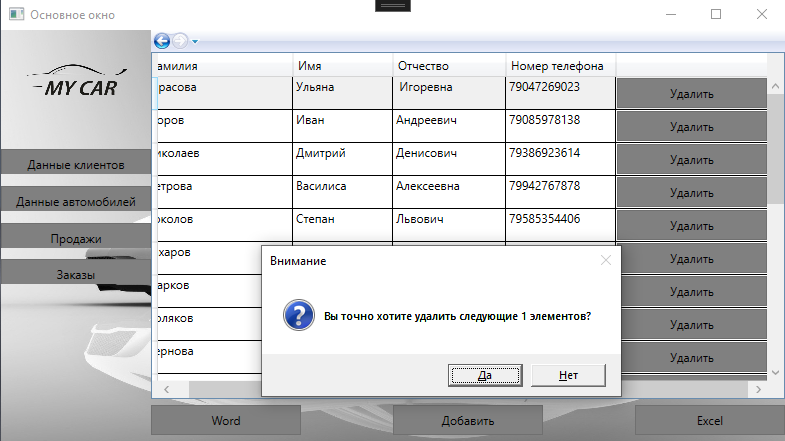


Рисунок 3.9. Удаление

var CarForRemoving = DGridCar.SelectedItems.Cast<Car>().ToList();

if (MessageBox.Show($"Вы точно хотите удалить следующие {CarForRemoving.Count()} элементов?", "Внимание",

MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

try

{

CarShopEntities.GetContext().Car.RemoveRange(CarForRemoving);

CarShopEntities.GetContext().SaveChanges();

MessageBox.Show("Данные удалены!");

DGridCar.ItemsSource = CarShopEntities.GetContext().Car.ToList();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

### }**3.2.4 Разработка модуля «Отчет»**

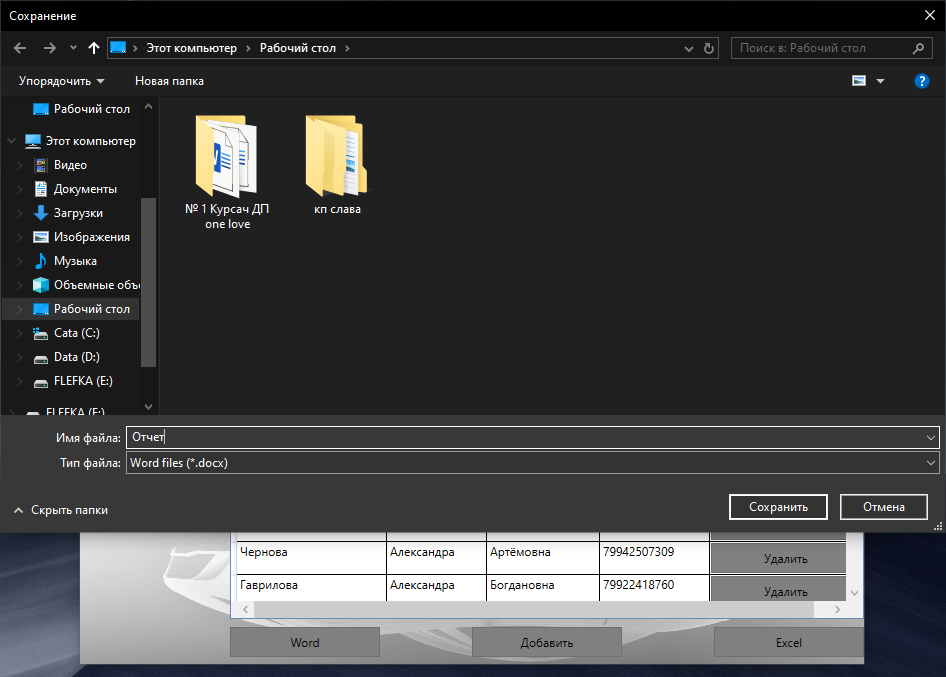
****

Рисунок 3.9.2 Создание отчета

private void WordExport\_Click(object sender, RoutedEventArgs e)

{

Word.Application word = new Word.Application();

word.Visible = false; // Установка свойства Visible в false

Word.Document document = word.Documents.Add();

Word.Paragraph para = document.Content.Paragraphs.Add();

para.Range.Text = "Отчет";

Word.Table table = document.Tables.Add(para.Range, DGridCar.Items.Count + 1, DGridCar.Columns.Count);

table.Borders.Enable = 1;

for (int j = 0; j < DGridCar.Columns.Count; j++)

{

if (DGridCar.Columns[j].Header != null)

{

table.Cell(1, j + 1).Range.Text = DGridCar.Columns[j].Header.ToString();

}

}

for (int i = 0; i < DGridCar.Items.Count; i++)

{

for (int j = 0; j < DGridCar.Columns.Count; j++)

{

TextBlock b = DGridCar.Columns[j].GetCellContent(DGridCar.Items[i]) as TextBlock;

if (b != null)

{

table.Cell(i + 2, j + 1).Range.Text = b.Text;

}

}

}

// Запрос у пользователя пути и названия файла

SaveFileDialog saveFileDialog = new SaveFileDialog();

saveFileDialog.Filter = "Word files (\*.docx)|\*.docx|All files (\*.\*)|\*.\*";

if (saveFileDialog.ShowDialog() == true)

{

string filePath = saveFileDialog.FileName;

document.SaveAs2(filePath); // Сохранение файла по указанному пути

document.Close(); // Закрытие документа

word.Quit(); // Закрытие приложения Word

}

}

private void ExcelExport\_Click(object sender, RoutedEventArgs e)

{

Excel.Application excel = new Excel.Application();

excel.Visible = false;

Workbook workbook = excel.Workbooks.Add();

Worksheet worksheet = workbook.ActiveSheet;

if (DGridCar != null)

{

for (int j = 0; j < DGridCar.Columns.Count; j++)

{

if (DGridCar.Columns[j].Header != null)

{

worksheet.Cells[2, j + 1] = DGridCar.Columns[j].Header.ToString();

}

}

for (int i = 0; i < DGridCar.Items.Count; i++)

{

for (int j = 0; j < DGridCar.Columns.Count; j++)

{

TextBlock b = DGridCar.Columns[j].GetCellContent(DGridCar.Items[i]) as TextBlock;

if (b != null)

{

worksheet.Cells[i + 2, j + 1] = b.Text;

}

}

}

}

else

{

MessageBox.Show("datagridcar не инициализирован или равен null.");

worksheet.Cells[1, 1] = "Отчет";

}

SaveFileDialog saveFileDialog = new SaveFileDialog();

saveFileDialog.Filter = "Excel files (\*.xlsx)|\*.xlsx|All files (\*.\*)|\*.\* ";

if (saveFileDialog.ShowDialog() == true)

{

string filePath = saveFileDialog.FileName;

workbook.SaveAs(filePath);

workbook.Close();

excel.Quit();

}

}

# **ЗАКЛЮЧЕНИЕ**

Таким образом, создание информационной системы для автосалона является стратегически важным шагом для улучшения управления клиентской базой, автоматизации процессов и повышения уровня обслуживания клиентов. Разработка базы данных и информационной системы «Автосалон» позволит эффективно управлять информацией о доступных автомобилях, заказах и клиентах, улучшит сервис для клиентов и общую производительность бизнеса. Внедрение такой системы будет способствовать конкурентоспособности автосалона в условиях быстро меняющегося рынка и технологий.

При выполнении данной работы, а именно информационная система «Автосалон», была разработана диаграмма Entity Relationship Diagram, была спроектирована база данных в приложении Microsoft SQL Server Management Studio, был разработан интерфейс с функционалом автоматизированной информационной системы с помощью C# WPF.

Ссылка на репозиторий: <https://github.com/NA4ER/KyrsovoyProject>

# **СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

Электронные ресурсы

1. .NET для начинающих. Что такое среда .NET и как она работает? // Режим доступа: <https://habr.com/ru/sandbox/13849/>
2. MS SQL Server и T-SQL // Режим доступа: <https://metanit.com/sql/sqlserver/2.1.php>
3. WPF и C# | Полное руководство // Режим доступа: <https://metanit.com/sharp/wpf/?ysclid=lvwp90h2zf169419289>
4. Документация по языку C# // Режим доступа: <https://learn.microsoft.com/ru-ru/dotnet/csharp/>
5. Что такое Windows Presentation Foundation - WPF .NET // Режим доступа: <https://learn.microsoft.com/ru-ru/dotnet/desktop/wpf/overview/?view=netdesktop-8.0>