Федеральное агентство железнодорожного транспорта

Омский государственный университет путей сообщения

Кафедра «Автоматика и системы управления»

К ЗАЩИТЕ ДОПУСТИТЬ

Доцент кафедры АиСУ

\_\_\_\_\_\_\_\_ Н.А.Тихонова

ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ

«Прокат автомобилей»

Пояснительная записка к курсовому проекту по дисциплине «Управление данными и Big Data»

ИНМВ.300000.000 ПЗ

|  |  |  |
| --- | --- | --- |
|  |  | Студент гр. 70ЦЭ  \_\_\_\_\_\_\_\_\_\_\_\_Б.С. Коломиец  «\_\_» \_\_\_\_\_\_\_\_2022 г.  Руководитель –  доцент кафедры АиСУ  \_\_\_\_\_\_\_\_\_\_\_\_Н.А. Тихонова  «\_\_» \_\_\_\_\_\_\_\_2022 г. |

Омск 2022

Задание

Необходимо создать базу данных для небольшой фирмы проката автомобилей. База должна предоставлять, и обеспечивать следующие возможности:

- Поиск подходящего для клиента варианта автомобиля по марке, году выпуска, цвету;

- Развёрнутый вывод информации о доходах с проката автомобилей в целом, и каждой конкретной единицы в частности;

- Внесение новых единиц автопарка в базу;

- Составление договора проката.

Так же в базе данных должны быть реализованы следующие запросы:

- Запросы с параметром (3 единицы), позволяющие найти клиенту автомобиль нужного цвета или марки, а также увидеть, находился ли данный автомобиль в пользовании в определённый период времени;

- Запросы на выборку (2 единицы), выводящие информацию по автомобилям после определённого года выпуска, марке;

- Запрос с вычисляемым полем, позволяющий определить общую сумму проката по конкретному автомобилю;

- Запросы с групповыми операциями (2 единицы), показывающие среднюю стоимость марки авто и количество прокатов по каждому клиенту;

- Запрос на обновление, позволяющий показать автомобили только после определённого года выпуска;

- Запрос на удаление, убирающий записи тех позиций, где стоимость ежедневного проката превышает определённую сумму;

- Запрос на создание таблицы (2 единицы), который создаёт новую таблицу с перечнем автомобилей до определённого года выпуска, а также создаёт копию существующей таблицы.

Реферат

УДК 621.398

Пояснительная записка к курсовому проекту содержит 39 страниц, 25 рисунков, 5 таблиц, 7 использованных источников.

Объектом курсового проекта является проектирование базы данных.

Цель курсового проекта – предоставить удобный и быстрый доступ к базе данных, содержащей информацию о работе небольшой фирмы проката автомобилей.

Результатом индивидуальной работы является база данных «Прокат автомобилей».

База данных разработана с использованием реляционной СУБД корпорации Microsoft – Access 2010.

Пояснительная записка выполнена в текстовом редакторе Microsoft Word 2010. ER-диаграмма выполнена в пакете Microsoft Visio 2010.

Содержание

[Введение 5](#_Toc91541214)

[1 Общие теоретические положения 7](#_Toc91541215)

[2 UML-диаграммы для описания бизнес-процессов 9](#_Toc91541216)

[3 Проектирование баз данных с использованием средств CASE 11](#_Toc91541217)

[4 Предметная область 13](#_Toc91541218)

[5 Безопасность данных 16](#_Toc91541219)

[6 Логическая модель базы данных 20](#_Toc91541220)

[6.1 Адаптация концептуальной модели БД 20](#_Toc91541221)

[6.2 Создание межтабличных связей 23](#_Toc91541222)

[7 Создание форм для наполнения таблиц данными 24](#_Toc91541223)

[8 Создание запросов 26](#_Toc91541224)

[8.1 Запрос на выборку 26](#_Toc91541225)

[8.2 Запрос на выборку с параметром 27](#_Toc91541226)

[8.3 Итоговые запросы 27](#_Toc91541227)

[8.4 Запрос на создание нового вычисляемого поля 28](#_Toc91541228)

[8.5 Запрос на удаление записей 29](#_Toc91541229)

[8.6 Запрос на обновление записей 30](#_Toc91541230)

[8.7 Перекрестный запрос 30](#_Toc91541231)

[8.8 Запрос на создание новой таблицы 31](#_Toc91541232)

[9 Создание отчетов 33](#_Toc91541233)

[10 Проектирование кнопочной формы 35](#_Toc91541234)

[Вывод 38](#_Toc91541235)

[Библиографический список 39](#_Toc91541236)

# Введение

Целью данного курсового проекта является проектирование базы данных заданной предметной области с помощью системы управления базами данных (СУБД) MS Access.

Программный комплекс MS Office, в состав которого входит СУБД MS Access, является самым распространенным пакетом для автоматизации работы в офисе. СУБД MS Access, представляет собой комплекс программных средств, предназначенных для создания структуры новой базы, наполнение её содержимым, редактирование содержимого, отбора отображаемых данных в соответствии с заданным критерием упорядочивания данных, оформление данных с последующей выдачей на устройство вывода или передачи информации по каналам связи.

База данных (БД) – это средство сбора и организации информации. В базах данных могут содержаться сведения о людях, продуктах, заказах и т. д. Многие базы данных изначально представляют собой список в текстовом процессоре или электронной таблице. По мере того, как список разрастается, в нем накапливаются излишние и противоречивые данные. В форме списка эти данные становится все труднее понять, а возможности поиска или извлечения подмножеств данных для просмотра весьма ограничены. Когда возникают подобные проблемы, полезно перенести информацию в базу данных, созданную с помощью системы управления базами данных, что позволит управлять данными, осуществлять хранение, поиск, извлечение, редактирование информации, хранимой в базе. Существуют иерархические, сетевые и реляционные базы данных.

MS Access – это интерактивная, реляционная СУБД для операционной системы Windows. Она предназначена для хранения и поиска данных, представления информации в удобном виде и автоматизации часто повторяющейся операции (ведение счетов, учета товара, планирование и т.д.) С помощью MS Access можно разрабатывать удобные формы ввода данных, а также осуществлять обработку данных и выдачу сложных отчетов. Объектом обработки MS Access является файл БД, имеющий произвольное имя и расширение .accdb. В этот файл входят основные объекты MS Access: таблицы, формы, запросы, отчеты, макросы, модули. В состав СУБД входят конструкторы таблиц, форм, запросов и отчетов. Эту систему можно рассматривать как среду разработки приложений. Используя макросы и модули для автоматизации решения задач можно создавать ориентированные на пользователя приложения. MS Access предоставляет пользователю механизмы работы с базами различных форматов. Мощность и доступность MS Access делает эту систему одной из лучших СУБД.

Базы данных обеспечивают хранение информации, отражающей основные параметры работы компании (предприятия или учреждения) и дополнительные сведения. Базами данных пользуются повсеместно во всех сферах жизнедеятельности людей, в том числе и в сфере проката автомобилей. Они служат для автоматизации работы, так как работа с информацией в бумажном виде затруднена невозможностью её быстрой обработки.

Целью курсового проекта является совершенствование навыков использования современных компьютерных технологий. В ходе выполнения курсового проекта необходимо разработать примерную базу данных для хранения информации фирмы проката автомобилей и возможности работать с данными базы. Для достижения поставленной цели необходимо решить следующие задачи:

* создать ряд таблиц и других основных элементов базы данных;
* построить базу данных таким образом, чтобы время обработки информации было минимальным;
* обеспечить наглядность представления данных;
* максимально облегчить ввод в базу данных новой информации;
* информация, содержащаяся в базе данных, должна быть взаимосвязана, обеспечивая тем самым удобство изменения данных, а также их целостность.

# 1 Общие теоретические положения

Система управления базами данных – это совокупность языковых и программных средств, которая осуществляет доступ к данным, позволяет их создавать, менять и удалять, обеспечивает безопасность данных и т.д. В общем СУБД – это система, позволяющая создавать базы данных и манипулировать сведениями из них. А осуществляет этот доступ к данным СУБД посредством специального языка – SQL. SQL – язык структурированных запросов, основной задачей которого является предоставление простого способа считывания и записи информации в базу данных. Сами по себе базы данных не представляли бы интереса, если бы не было систем управления базами данных (СУБД). Существуют иерархические, сетевые и реляционные базы данных.

Иерархическая модель данных – это [модель данных](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), где используется представление [базы данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) в виде  [древовидной](https://ru.wikipedia.org/wiki/%D0%94%D1%80%D0%B5%D0%B2%D0%BE%D0%B2%D0%B8%D0%B4%D0%BD%D0%B0%D1%8F_%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0)  (иерархической) структуры, состоящей из объектов ([данных](https://ru.wikipedia.org/wiki/%D0%94%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5)) различных уровней.

Между объектами существуют связи, каждый объект может включать в себя несколько объектов более низкого уровня. Такие объекты находятся в отношении предка (объект более близкий к корню) к потомку (объект более низкого уровня), при этом возможна ситуация, когда объект-предок не имеет потомков или имеет их несколько, тогда как у объекта-потомка обязательно только один предок. Объекты, имеющие общего предка, называются близнецами (в программировании применительно к структуре данных дерево устоялось название братья).

Строки таблицы называются записями. Все записи таблицы имеют одинаковую структуру – они состоят из полей (элементов данных), в которых хранятся атрибуты объекта. Каждое поле записи содержит одну характеристику объекта и представляет собой заданный тип данных (например, текстовая строка, число, дата). Для идентификации записей используется первичный ключ. Первичным ключом называется набор полей таблицы, комбинация значений которых однозначно определяет каждую запись в таблице.

Ключ – это столбец (может быть несколько столбцов), добавляемый к таблице, и позволяющий установить связь с записями в другой таблице. Существуют ключи двух типов: первичные и вторичные или внешние.

Первичный ключ – это одно или несколько полей (столбцов), комбинация значений которых однозначно определяет каждую запись в таблице. Первичный ключ не допускает значений Null и всегда должен иметь уникальный индекс. Первичный ключ используется для связывания таблицы с\внешними\ключами\в\других\таблицах.

Внешний (вторичный) ключ – это одно или несколько полей (столбцов) в таблице, содержащих ссылку на поле или поля первичного ключа в другой таблице. Внешний ключ определяет способ объединения таблиц [1].

Таблицы объединяются связями. Связь осуществляется путем сопоставления данных в ключевых столбцах; обычно это столбцы, имеющие в обеих таблицах одинаковые названия. В большинстве случаев сопоставляются первичный ключ одной таблицы, содержащий для каждой из строк уникальный идентификатор, и внешний ключ другой таблицы.   
 Существует три вида связей между таблицами. Вид создаваемой связи зависит от того, как заданы связанные столбцы.

Связь "один ко многим" – наиболее распространенный вид связи. При такой связи каждой строке таблицы А может соответствовать множество строк таблицы Б, однако каждой строке таблицы Б может соответствовать только одна строка таблицы А.

При установлении связи "многие ко многим" каждой строке таблицы А может соответствовать множество строк таблицы Б и наоборот. Такая связь создается при помощи третьей таблицы, называемой соединительной, первичный ключ которой состоит из внешних ключей, связанных с таблицами А и Б.

При установлении связи "один к одному" каждой строке таблицы А может соответствовать только одна строка таблицы Б и наоборот. Связь "один к одному" создается в том случае, когда оба связанные столбца являются первичными ключами или на них наложены ограничения уникальности.

Транзакция – группа последовательных операций с [базой данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), которая представляет собой логическую единицу работы с данными. Транзакция может быть выполнена либо целиком и успешно, соблюдая целостность данных и независимо от параллельно идущих других транзакций, либо не выполнена вообще, и тогда она не должна произвести никакого эффекта. Транзакции обрабатываются [транзакционными системами](https://ru.wikipedia.org/wiki/%D0%A2%D1%80%D0%B0%D0%BD%D0%B7%D0%B0%D0%BA%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0), в процессе работы которых создаётся [история транзакций](https://ru.wikipedia.org/wiki/%D0%96%D1%83%D1%80%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_%D0%B8%D0%B7%D0%BC%D0%B5%D0%BD%D0%B5%D0%BD%D0%B8%D0%B9).

При проектировании баз данных важную роль играет нормализация. Нормализация – это процесс организации данных в базе данных, включающий создание таблиц и установление отношений между ними в соответствии с правилами, которые обеспечивают защиту данных и делают базу данных более гибкой, устраняя избыточность и несогласованные зависимости [2].

# 2 UML-диаграммы для описания бизнес-процессов

Unified Modeling Language (UML) − унифицированный язык моделирования. Расшифруем: modeling подразумевает создание модели, описывающей объект. Unified (универсальный, единый) − подходит для широкого класса проектируемых программных систем, различных областей приложений, типов организаций, уровней компетентности, размеров проектов. UML описывает объект в едином заданном синтаксисе, поэтому, где бы вы не нарисовали диаграмму, ее правила будут понятны для всех, кто знаком с этим графическим языком − даже в другой стране.

Одна из задач UML − служить средством коммуникации внутри команды и при общении с заказчиком. Давайте рассмотрим возможные варианты использования диаграмм.

− Проектирование. UML-диаграммы помогут при моделировании архитектуры больших проектов, в которой можно собрать как крупные, так и более мелкие детали и нарисовать каркас (схему) приложения. По нему впоследствии будет строиться код.

− Реверс-инжиниринг − создание UML-модели из существующего кода приложения, обратное построение. Может применяться, например, на проектах поддержки, где есть написанный код, но документация неполная или отсутствует.

− Из моделей можно извлекать текстовую информацию и генерировать относительно удобочитаемые тексты − документировать. Текст и графика будут дополнять друг друга.

Как и любой другой язык, UML имеет собственные правила оформления моделей и синтаксис. С помощью графической нотации UML можно визуализировать систему, объединить все компоненты в единую структуру, уточнять и улучшать модель в процессе работы. На общем уровне графическая нотация UML содержит 4 основных типа элементов [3]:

− фигуры;

− линии;

− значки;

− надписи.

UML-нотация является де-факто отраслевым стандартом в области разработки программного обеспечения, ИТ-инфраструктуры и бизнес-систем.

В языке UML есть 12 типов диаграмм:

1. Диаграммы, представляющие статическую структуру приложения:

− Диаграмма классов

− Диаграмма компонентов

− Диаграмма композитной/составной структуры

− Диаграмма развёртывания

− Диаграмма объектов

− Диаграмма пакетов

− Диаграмма профилей (UML2.2)

2. Диаграммы поведения:

− Диаграмма деятельности

− Диаграмма состояний

− Диаграмма вариантов использования

3. Диаграммы взаимодействия:

− Диаграмма коммуникации (UML2.0) / Диаграмма кооперации (UML1.x)

− Диаграмма обзора взаимодействия (UML2.0)

− Диаграмма последовательности

− Диаграмма синхронизации (UML2.0)

В рамках нашего проекта фирмы проката автомобилей данные диаграммы будут выглядеть следующим образом (Рисунок 1).

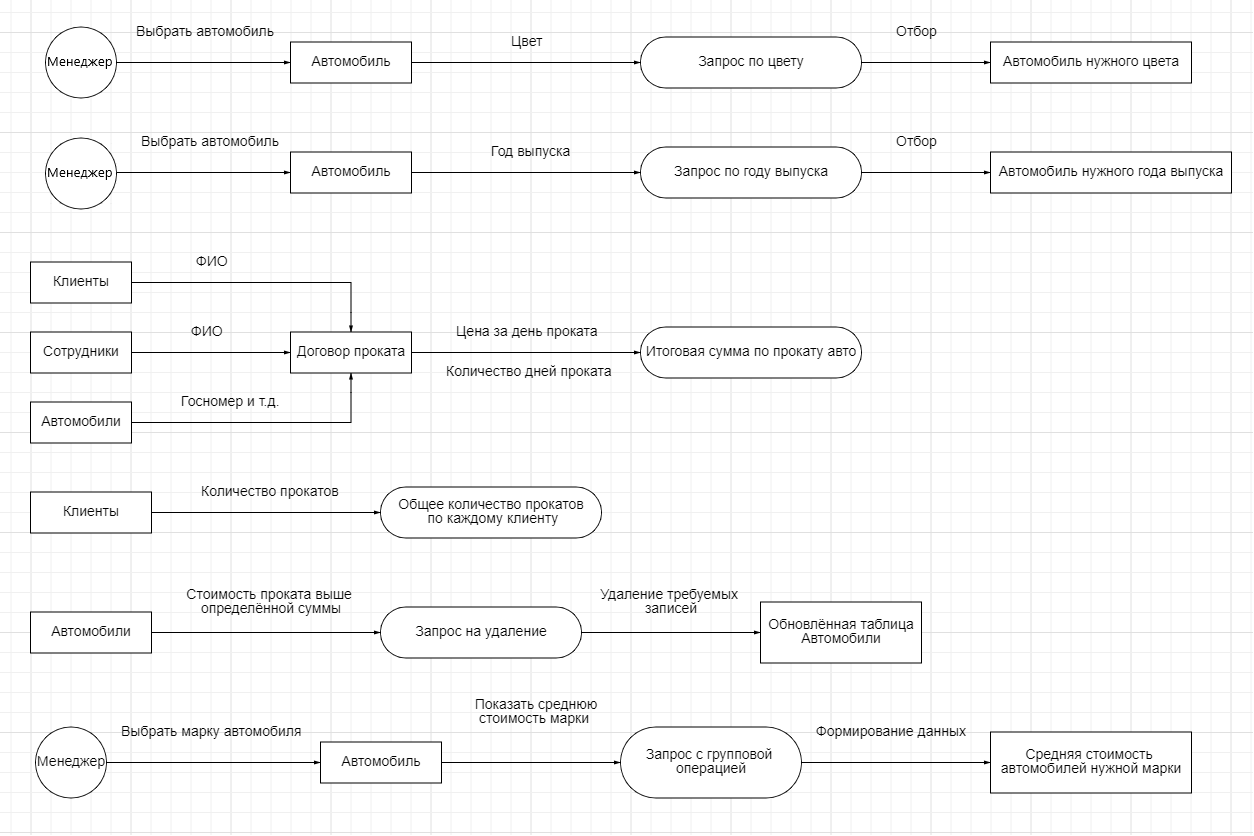


Рисунок 1 – UML-диаграмма бизнес-процессов фирмы проката автомобилей

# 3 Проектирование баз данных с использованием средств CASE

Тенденции развития современных информационных технологий приводят к постоянному возрастанию сложности систем баз данных. Опыт проектирования таких систем показывает, что это логически сложная, трудоемкая и длительная по времени работа, требующая высокой квалификации участвующих в ней специалистов. Начиная с 70-х и 80-х годов, при разработке информационных систем широко применяется структурная методология, предоставляющая в распоряжение разработчиков строгие формализованные методы описания систем и принимаемых технических решений.

Она основана на наглядной графической технике: для описания различного рода моделей используются схемы и диаграммы. Для автоматизации этой технологии в настоящее время используются программно-технологические средства специального класса – CASE-средства, реализующие CASE-технологию создания и сопровождения информационных систем. Термин CASE (Computer Aided Software Engineering) используется в настоящее время в весьма широком смысле. Первоначальное значение термина CASE, ограниченное вопросами автоматизации разработки только лишь программного обеспечения, в настоящее время приобрело новый смысл, охватывающий процесс разработки сложных автоматизированных систем в целом.

Под термином CASE-средства понимаются программные средства, поддерживающие процессы создания и сопровождения информационных систем, включая анализ и формулировку требований, проектирование приложений и баз данных, генерацию кода, тестирование, документирование, обеспечение качества, конфигурационное управление и управление проектом, а также другие процессы.

CASE-средства – это автоматизированные средства, основанные на CASE-технологиях, позволяющие автоматизировать отдельные этапы жизненного цикла программного обеспечения. Все современные CASE-средства могут быть классифицированы по типам и категориям. Классификация по типам отражает функциональную ориентацию на процессы жизненного цикла программного обеспечения. Классификация по категориям определяет степень интеграции по выполняемым функциям и включает отдельные локальные средства, решающие наиболее автономные задачи (по-английски tools), набор частично интегрированных средств, охватывающих большинство этапов жизненного цикла (toolkit) и полностью интегрированные средства, поддерживающие весь жизненный цикл информационных систем.

Классификация по типам включает следующие основные CASE-средства:

1. Средства анализа, предназначенные для построения и анализа моделей предметной области (Bpwin, Design/IDEF);

2. Средства анализа и проектирования, предназначенные для создания проектных спецификаций (CASE.Аналитик, Vantage Team Builder, Designer/2000, Silverrun, PRO-IV);

3. Средства проектирования баз данных, обеспечивающие моделирование данных и генерацию схем баз данных для наиболее распространенных СУБД (Silverrun, Vantage Team Builder, Designer/2000, ERwin, S-Designor);

4. Средства разработки приложений и генераторы кодов (Vantage Team Builder, Silverrun, PRO-IV);

5. Средства реинжениринга, обеспечивающие анализ программных кодов, схем баз данных и формирование на их основе различных моделей и проектных спецификаций. Средства анализа схем баз данных входят в состав: (Silverrun, Vantage Team Builder, Designer/2000, Erwin, S-Designor). Для анализа программных кодов используются такие средства, как Rational Rose и Object Team.

В рамках нашей работы для последующего формирования необходимых диаграмм остановим свой выбор на программе ERwin. ERwin (Logic Works) - CASE-инструмент для создания концептуальных и логических схем баз данных. Механизм работы программного обеспечения основан на методологии системного представления данных IDEF1X, хотя теперь он также поддерживает и другие методологии, включая пространственное моделирование. Программа позволяет строить диаграммы, удобные для чтения и анализа, отображающие потоки данных и управление ими в любой системе (не обязательно в целом информационной). Условно это можно представить как взаимодействие сущностей и связей, где сущности – это различные процессы в системе, а связи – это потоки данных, информации, документов, распоряжений и прочего. Каждая сущность может быть декомпозирована и быть представлена, в свою очередь, как совокупность более мелких и быстрых процессов и обмена информацией, требуемого для их реализации. ERwin находит наиболее актуальное применение в сфере бизнес-аналитики, для детального изучения работы предприятия и оптимизации его деятельности, но в целом может быть применен практически во всех сферах жизни для многих целей – вплоть до менеджмента личного времени [4].

# 4 Предметная область

База данных «Прокат автомобилей» создана для автоматизации работы фирмы, предоставляющей услуги выдачи автомобилей разных марок в аренду посуточно.

Персонал фирмы состоит из трёх сотрудников с фиксированной заработной платой.

База данных должна выполнять следующие основные функции:

* Поиск подходящего для клиента варианта автомобиля по марке, году выпуска, цвету;
* Развёрнутый вывод информации о доходах с проката автомобилей в целом, и каждой конкретной единицы в частности;
* Внесение новых единиц автопарка в базу;
* Составление договора проката.

Сотрудники обладают полным доступом к базе данных.

База данных будет применяться всякий раз, когда появится новый клиент с перечнем своих запросов на автомобиль. В базу данных будет вводиться следующая информация:

* Марка автомобиля
* Год выпуска
* Цвет
* Стоимость одного дня проката и т.п.

Такие объекты базы данных как автомобили, клиенты, сотрудники имеют собственный код. При создании нового договора проката ему также присваивается код.

В базе данных будут использоваться следующие запросы:

- Запросы с параметром (3 единицы), позволяющие найти клиенту автомобиль нужного цвета или марки, а также увидеть, находился ли данный автомобиль в пользовании в определённый период времени;

- Запросы на выборку (2 единицы), выводящие информацию по автомобилям после определённого года выпуска, марке;

- Запрос с вычисляемым полем, позволяющий определить общую сумму проката по конкретному автомобилю;

- Запросы с групповыми операциями (2 единицы), показывающие среднюю стоимость марки авто и количество прокатов по каждому клиенту;

- Запрос на обновление, позволяющий показать автомобили только после определённого года выпуска;

- Запрос на удаление, убирающий записи тех позиций, где стоимость ежедневного проката превышает определённую сумму;

- Запрос на создание таблицы (2 единицы), который создаёт новую таблицу с перечнем автомобилей до определённого года выпуска, а также создаёт копию существующей таблицы.

Выходной информацией является договор проката, в котором будут отражены:

- Номер договора

- ФИО клиента

- Госномер автомобиля

- Дата начала сдачи в прокат

- Количество дней в аренде

- ФИО сотрудника проката

- Дата формирования договора

Концептуальное проектирование – сбор, анализ и редактирование требований к данным. Для этого осуществляются следующие мероприятия:

- обследование предметной области, изучение ее информационной структуры;

- выявление всех фрагментов, каждый из которых характеризуется пользовательским представлением, информационными объектами, связями между ними и процессами;

- моделирование и интеграция всех представлений.

Результат данного этапа – концептуальная модель, инвариантная к структуре Базы данных, часто представляется в виде модели «сущность-связь».

Концептуальное проектирование начинается с анализа предметной области, включает анализ концептуальных требований и информационных потребностей, выявление информационных объектов и связей между ними, построение концептуальной модели (схемы) данных.

Главными элементами концептуальной модели данных являются объекты и отношения. Объекты представляют собой любой конкретный (реальный) объект в рассматриваемой области.

Исходя из спецификации требования, определим основные типы сущностей. Сущностью называется некоторая принятая в конкретной постановке задачи абстракция реального мира, процесса или явления, о котором необходимо хранить информацию в системе. В качестве синонима термина «сущность» используется также термин «информационный объект».

Объекты в каждый момент времени характеризуются определенным состоянием, которое описывается набором свойств и отношений (или связей) с другими объектами.

Характеристика, описывающая какое-либо свойство сущности, которое можно сформулировать и записать, называется атрибутом. Атрибут, который однозначно определяет сущность, называется идентификатором [5].

Каждый объект предметной области характеризуется некоторым наборов атрибутов, отображающим свойства объекта. Атрибуты используются для определения того, какая информация должна быть собрана об объекте. Примерами атрибутов для объекта «Автомобили» служат Госномер, Марка, Цвет, Год выпуска, Страховая стоимость, Стоимость одного дня проката.

В рассматриваемой предметной области можно выделить следующие сущности:

- Договор проката: содержит информацию о номере договора, ФИО клиента, госномере автомобиля, ФИО сотрудника, дате формирования и т.д.;

- Клиенты: содержит информацию о ФИО клиента и данных паспорта клиента;

- Автомобили: содержит информацию о госномере автомобиля, его марке, цвете, годе выпуска и т.д.;

- Сотрудники: содержит информацию о ФИО сотрудника фирмы.

Определим связи между сущностями (Таблица 1).

Таблица 1 – Связи между сущностями

|  |  |  |
| --- | --- | --- |
| Название связи | Тип | Связи между сущностями |
| Ввод клиента | один-ко-многим | КодКлиента-Клиент |
| Ввод автомобиля | один-ко-многим | Госномер-Автомобиль |
| Ввод сотрудника | один-ко-многим | КодСотрудника-Сотрудник |

Исходя из этого, концептуальная модель будет выглядеть следующим образом (Рисунок 2).

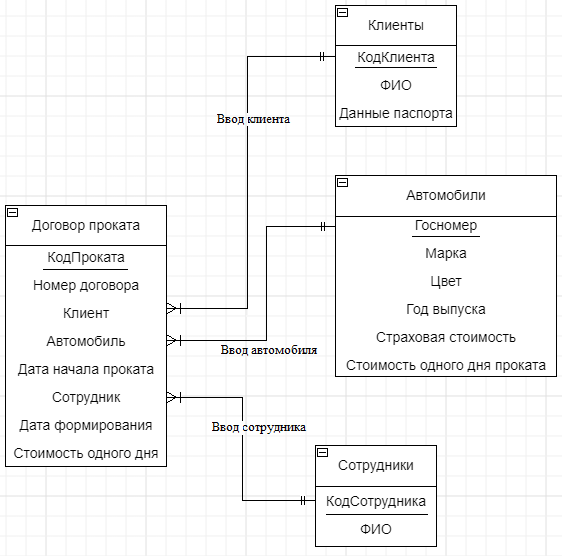


Рисунок 2 – Концептуальная модель

При разработке концептуальной модели использовались средства анализа, такие как ERwin. Нотация выполнена в нотации Мартина.

# 5 Безопасность данных

Понятие «безопасность баз данных» включает в себя целый ряд инструментов, средств контроля и мер, направленных на обеспечение и поддержку конфиденциальности, целостности и доступности баз данных. Основное внимание в данной статье уделяется вопросам конфиденциальности, поскольку большинство инцидентов безопасности связано с утечкой конфиденциальных данных.

Безопасность баз данных должна включать в себя следующие элементы:

− Данные, хранящиеся в базе данных

− Система управления базами данных (СУБД)

− Любые связанные приложения

− Физический сервер базы данных и/или виртуальный сервер базы данных и используемое оборудование

− Вычислительные ресурсы и/или сетевая инфраструктура, предназначенная для доступа к базе данных

Безопасность баз данных — сложная и комплексная инициатива, охватывающая все аспекты технологий и методов в области информационной безопасности. Кроме того, требования к безопасности баз данных и удобству использования часто противоречат друг другу.

Помимо этого, также существуют утечки данных. Под утечкой данных понимают неспособность обеспечить конфиденциальность данных в базе данных. Степень серьезности последствий утечки данных для предприятия зависит от целого ряда событий или факторов:

− утечка интеллектуальной собственности: интеллектуальная собственность — коммерческая тайна, изобретения, запатентованные методики — может иметь крайне важное значение для сохранения конкурентного преимущества на рынке. В случае кражи или раскрытия интеллектуальной собственности будет трудно или даже невозможно сохранить или восстановить конкурентные преимущества.

− ущерб для репутации бренда: если клиенты и партнеры не доверяют вам в вопросах защиты данных, это неизбежно скажется на объеме продаж ваших продуктов или услуг (или желании сотрудничать с вашей компанией).

− бесперебойная работа (или невозможность ее обеспечения): некоторые компании не могут продолжать работу, пока не будет устранен инцидент утечки данных.

− штрафы или санкции за несоблюдение нормативных требований: несоблюдение глобальных требований, в частности закона сарбейнза-оксли или стандарта payment card industry data security standard (PCI DSS), отраслевых законов в области конфиденциальности данных, например HIPAA, или региональных норм, например общеевропейского регламента о защите персональных данных (GDPR), может повлечь за собой катастрофические финансовые последствия. В наиболее серьезных случаях штрафы могут превышать несколько миллионов долларов за каждое нарушение.

− расходы на устранение последствий и уведомление клиентов: помимо расходов на информирование клиентов об инциденте, организация вынуждена оплачивать услуги по расследованию инцидентов, кризисному управлению, приоретизации, восстановлению затронутых систем и пр.

Необходимо, помимо всего прочего, учитывать возможные угрозы и проблемы, которые могут возникнуть в процессе эксплуатации БД. Ниже приводятся наиболее распространенные виды и причины атак на систему безопасности баз данных.

Внутренняя угроза — это угроза безопасности, исходящая от любого из трех возможных источников, обладающих привилегированным доступом к базе данных:

− Внутренний злоумышленник, целенаправленно совершающий противоправные действия для нанесения ущерба

− Внутренний злоумышленник, совершающий ошибки по небрежности, в результате чего база данных становится уязвимой для атак

− Проникший злоумышленник — стороннее лицо, которое каким-либо образом завладело идентификационными данными, например с помощью фишинга или получения доступа к самой базе учетных данных

Внутренние угрозы относятся к основным причинам нарушений в сфере безопасности баз данных и часто являются результатом чрезмерного количества привилегированных учетных записей.

Ошибки, вызванные человеческим фактором.

Случайности, ненадежные пароли, предоставление паролей другим лицам и другие непродуманные или недальновидные действия пользователей по-прежнему составляют около половины (49%) всех зарегистрированных случаев утечки данных.

Использование уязвимостей ПО баз данных.

Поиск уязвимостей в разнообразных программных продуктах, включая ПО для управления базами данных, и проведение преднамеренных атак с их использованием — источник заработка для злоумышленников. Все крупные производители коммерческого ПО для баз данных и открытые платформы управления базами данных регулярно выпускают исправления безопасности для устранения обнаруженных уязвимостей, однако несвоевременная установка таких исправлений повышает вероятность атак.

Атаки путем внедрения кода SQL/NoSQL.

Этот вид угроз, характерных для баз данных, заключается во внедрении произвольных строк SQL или другого кода в запросы к базе данных, обрабатываемые веб-приложениями или заголовками HTTP. Организации, которые не придерживаются рекомендаций по созданию защищенного кода веб-приложений и не выполняют регулярное тестирование на наличие уязвимостей, могут стать жертвами таких атак.

Атаки переполнения буфера.

Переполнение буфера происходит, когда процесс пытается записать в блок памяти фиксированной длины данные, размер которых превышает максимально допустимое значение. Для проведения атак злоумышленники могут использовать дополнительные данные, хранящиеся в соседних адресах памяти.

Атаки типа «отказ в обслуживании» (DoS/DDoS).

Во время атаки типа «отказ в обслуживании» (DoS) злоумышленник обрушивает на целевой сервер — в данном случае сервер базы данных — такое количество запросов, что сервер не в состоянии выполнять запросы реальных пользователей. В большинстве случаев это приводит к нестабильной работе или отказу сервера.

При распределенной атаке типа «отказ в обслуживании» (DDoS) поток запросов поступает одновременно с нескольких серверов, из-за чего прервать атаку становится еще труднее.

При оценке уровня безопасности базы данных с целью определения приоритетных задач следует обратить внимание на следующие вопросы:

− Физическая безопасность: вне зависимости от расположения сервера базы данных, в локальном или облачном ЦОД, это должно быть защищенное, кондиционируемое помещение.

− Административный контроль и управление доступом к сети: минимально допустимое количество пользователей должно обладать доступом к базе данных; пользователям должны быть предоставлены минимально необходимые права доступа в соответствии с выполняемыми задачами. Доступ к сети также должен быть ограничен до минимально необходимого уровня разрешений.

− Безопасность устройств и учетных записей конечных пользователей: крайне важно обеспечить постоянный контроль доступа к базе данных и действий с данными. Решения для мониторинга данных позволяют получать уведомления о нетипичных или подозрительных действиях с данными. Все пользовательские устройства, подключающиеся к сети базы данных, должны быть физически защищены (и доступны только разрешенным пользователям), в том числе с использованием средств контроля безопасности.

− Шифрование: все данные, включая данные в базе данных и идентификационные данные, должны быть защищены с помощью передовых алгоритмов шифрования – в процессе передачи и во время хранения. Необходимо придерживаться рекомендаций по управлению ключами шифрования.

− Безопасность ПО базы данных: следует всегда использовать новейшую версию ПО управления базами данных и устанавливать все исправления по мере их выхода.

− Безопасность веб-сервера/сервера приложений: любой сервер приложений или веб-сервер, взаимодействующий с базой данных, может оказаться каналом проведения атаки, поэтому следует обеспечить непрерывное тестирование безопасности и передовые методы управления.

− Безопасность резервных копий: все резервные копии, реплики или образы базы данных должны находиться под не менее строгим контролем безопасности, чем сама база данных.

− Аудит: необходимо регистрировать все операции входа в операционную систему и на сервер базы данных, а также действия с конфиденциальными данными. Необходимо обеспечить проведение регулярного аудита безопасности базы данных.

Соответственно, для нашей базы данных в рамках рассматриваемого курсового проекта, мы обеспечим безопасность следующим образом: каждый пользователь, который имеет доступ к базе данных, вначале должен проходить процедуру авторизации в системе Windows. После того, как пользователь зайдёт под своей учётной записью, ему необходимо будет ввести пароль от базы данных при её открытии, чтобы пользователь, не знающий пароля, не смог зайти в БД и менять данные. Таким способом защиты мы отсеем потенциально опасные и нежелательные элементы, обезопасив искомую БД. Пароль установим следующий: 123. Дополнительно мы будем производить резервное копирование базы данных по внутреннему расписанию, чтобы сохранять целостность и доступность важных данных.

# 6 Логическая модель базы данных

Перед созданием базы данных изучается предложенная предметная область («Прокат автомобилей»), состав рассматриваемых данных, которые будут храниться в БД, их организация. Далее определяется, из каких таблиц будет состоять база данных, какова их структура, какие данные будут помещены в каждую таблицу, тип данных в таблицах, как и по каким полям таблицы базы данных будут связаны, чтобы иметь возможность работать с этими данными. Разрабатываются формы для эффективного ввода данных в таблицы и возможности работать с данными - просматривать, редактировать, вводить новые данные. Проектируются запросы для получения информации из таблиц базы данных по заданному критерию, отчеты, позволяющие выводить информацию на бумажный носитель из таблиц и по результатам работы запросов.

Проектирование базы данных и основных ее объектов осуществлялось поэтапно. Для создания объектов использовались различные режимы: ручной – с использованием Конструктора, автоматизированный – с использованием программ – мастеров, автоматический. Этапы проектирования описаны ниже.

## 6.1 Адаптация концептуальной модели БД

Таблицы – это основные объекты БД, они хранят все данные базы, структуру базы (поля, их типы, свойства). Структуру таблиц БД образуют поля и записи. Структура таблицы определяет методы занесения данных и хранение их в базе.

В рассматриваемой базе данных создано 4 таблицы. Для создания таблиц используется режим Конструктор (вкладка Создание – группа Таблицы – режим Конструктор таблиц). При создании таблицы в соответствующие столбцы бланка таблицы вводятся имена полей, задается тип данных и свойства (размер поля, маска ввода, подпись, формат поля, число десятичных знаков и др.). Для связи с другими таблицами задается ключевое поле. Первичный ключ однозначно идентифицирует (определяет) запись в таблице. Внешний ключ **—** это поле, не являющееся первичным в данной таблице, оно лишь дублирует поле, которое служит первичным ключом в другой таблице и используется для связи этих таблиц.

При создании таблиц использовался тип данных Мастер подстановок, что позволит при вводе данных в таблицу выбирать значения из раскрывающегося списка, а не вводить вручную одни и те же значения данных этого поля.

Для редактирования структуры таблиц используется режим Конструктора (вкладка Главная – Режимы – Режим).

На рисунках 3–6 в режиме Конструктора представлены созданные таблицы, а в таблицах 2-5 определена структура соответствующих таблиц.

В начале начнём с таблицы «Автомобили» и рассмотрим её структуру: названия полей, типы данных, размеры данных.

Таблица 2 – Структура таблицы «Автомобили»

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Размер/Комментарий** |
| Госномер | Текстовый | Ключ |
| Марка | Текстовый | 20 |
| Цвет | Текстовый | 20 |
| Год выпуска | Числовой | Целое |
| Страховая стоимость | Денежный | Денежный |
| Стоимость одного дня проката | Денежный | Денежный |

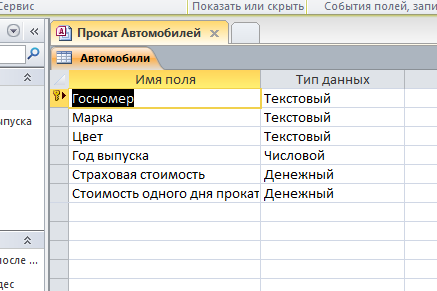


Рисунок 3 – таблица «Автомобили»

Теперь перейдём к таблице «Клиенты». Рассмотрим её структуру, используя аналогичные параметры.

Таблица 3 – Структура таблицы «Клиенты»

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Размер/Комментарий** |
| КодКлиента | Счётчик | Ключ |
| ФИО | Текстовый | 45 |
| Данные паспорта | Текстовый | 10 |

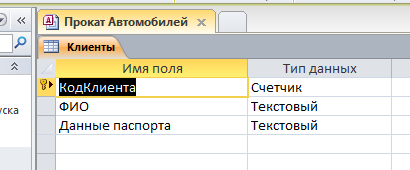


Рисунок 4 – таблица «Клиенты»

На данном этапе перейдём к таблице «Сотрудники». Рассмотрим её структуру, используя аналогичные параметры.

Таблица 4 – Структура таблицы «Сотрудники»

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Размер/Комментарий** |
| КодСотрудника | Счётчик | Ключ |
| ФИО | Текстовый | 45 |

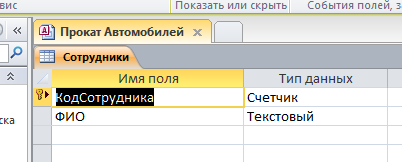


Рисунок 5 – таблица «Сотрудники»

После перейдём к таблице «Договор Проката». Рассмотрим её структуру, используя аналогичные параметры.

Таблица 5 – Структура таблицы «Договор Проката»

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Размер/Комментарий** |
| КодПроката | Счётчик | Ключ |
| Номер договора | Текстовый | 64 |
| Клиент | Числовой | Длинное целое |
| Автомобиль | Текстовый | 9 |
| Дата начала проката | Дата/Время | Краткий формат даты |
| Количество дней проката | Числовой | Байт |
| Сотрудник | Числовой | Длинное целое |
| Дата формирования | Дата/Время | Краткий формат даты |
| Стоимость одного дня | Денежный | Денежный |

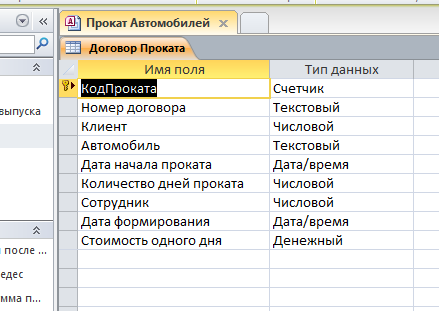


Рисунок 6 – таблица «Договор Проката»

## 6.2 Создание межтабличных связей

Связи позволяют черпать данные из разных таблиц, обеспечивают целостность данных.

Связи могут быть установлены:

* вручную (методом наложения ключевого поля одной таблицы на аналогичное поле другой);
* автоматически - при использовании типа данных Мастер подстановки.

Для создания связей между таблицами выбирается вкладка Работа с базами данных – группа Отношения - кнопка Схема данных, после чего появляется окно «Схема данных». В случае автоматической связи таблиц в окне «Схема данных» отображены имена таблиц, связанных между собой и списки полей этих таблиц. Для добавления в связь других таблиц в группе Связи выбирается кнопка «Отобразить таблицу», после чего появляется окно «Добавление таблицы», где выбираются таблицы для связи. В окне «Схема данных» становятся видны списки полей этих таблиц. Для установки связи ключевое поле нужной таблицы «перетаскивается» на аналогичное поле другой. Появляется диалоговое окно «Изменение связей», в котором указаны связываемые поля, тип отношения (один-к-одному или один-ко-многим). Для обеспечения целостности данных, каскадного обновления связанных полей и удаления связанных записей включаются соответствующие переключатели. Образовавшаяся связь отображается после установки связей. Контекстное меню линии связи позволит связь редактировать, удалить.

Получившиеся в итоге межтабличные связи рассматриваются также и как логическая модель данных (Рисунок 7).

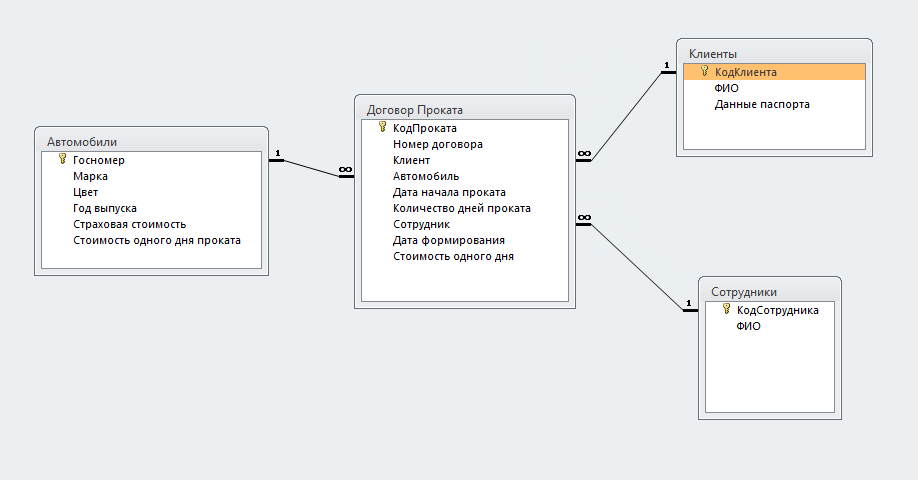


Рисунок 7 – Логическая модель данных

# 7 Создание форм для наполнения таблиц данными

Формы – это специальные средства для ввода данных без доступа к самим таблицам и их отображение с использованием специальных средств оформления. Основной смысл форм – предоставить пользователю возможность заполнять только те поля, которые ему заполнять положено. В форме можно размещать специальные элементы управления – кнопки, раскрывающиеся списки, переключатели и др.

Форма может быть создана автоматически (вкладка Создание – группа Формы – кнопка Формы или Другие формы). Наиболее удобным и эффективным способом создания форм является использование Мастера форм. При создании формы с помощью Мастера форм, она может быть создана на основе нескольких таблиц или запросов, что позволяет выбирать поля из нескольких таблиц или запросов.

Для создания формы с помощью Мастера форм выбирается вкладка Создание – группа Формы - Мастер форм (появляется окно Создание форм). Далее следует выбрать в качестве источника данных нужную таблицу (или таблицы), затем нужные поля. На следующем шаге выбирается внешний вид формы (в столбец, ленточный и др.). На последнем шаге работы с мастером форм указывается имя формы.

В режиме Конструктора форму можно редактировать, используя область заголовка, область данных и область примечаний.

Некоторые из созданных форм представлены ниже как в режиме Формы (Рисунок 8), так и в режиме Конструктора (Рисунок 9).

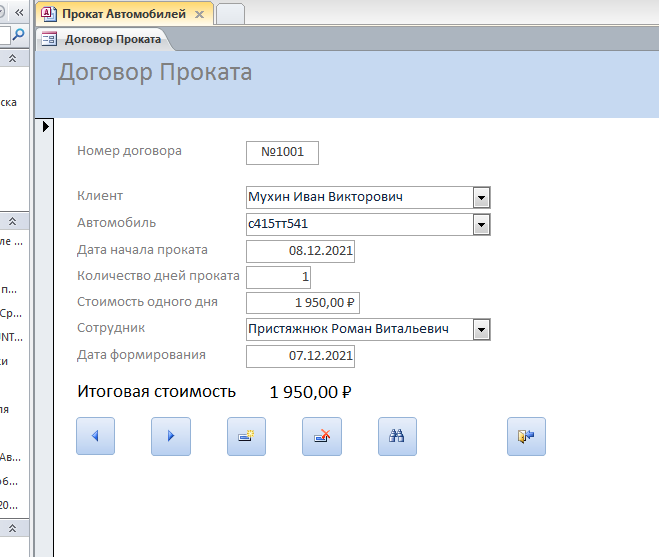


Рисунок 8 – форма «Договор Проката» в режиме Формы

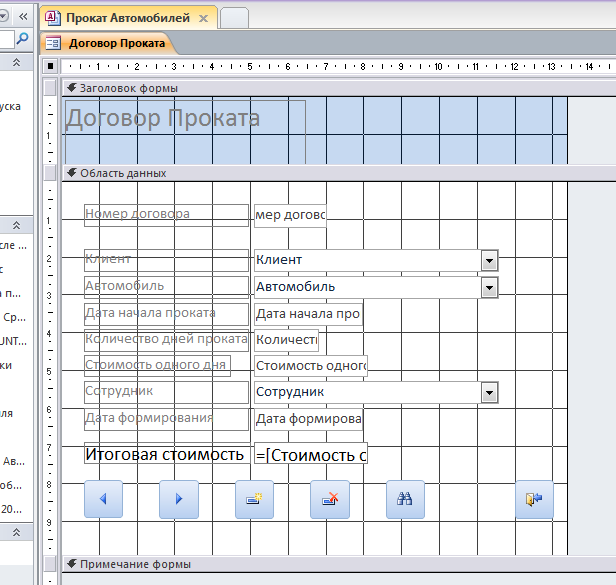


Рисунок 9 – форма «Договор Проката» в режиме Конструктора

При создании формы для ввода данных в таблицы, связанные отношением один-ко-многим, можно создавать так называемые сложные формы (формы с подчиненной формой или связанной формой). Форма создается с помощью Мастера форм. На этапе задания вида представления данных по умолчанию помечен переключатель Подчиненные формы (можно выбрать Связанные формы). Далее выбирается внешний вид подчиненной формы (например, ленточный). На последнем шаге создания формы задаются имена форм (формы и подчиненной ей формы). Форма с подчиненной формой представлена на Рисунке 10.

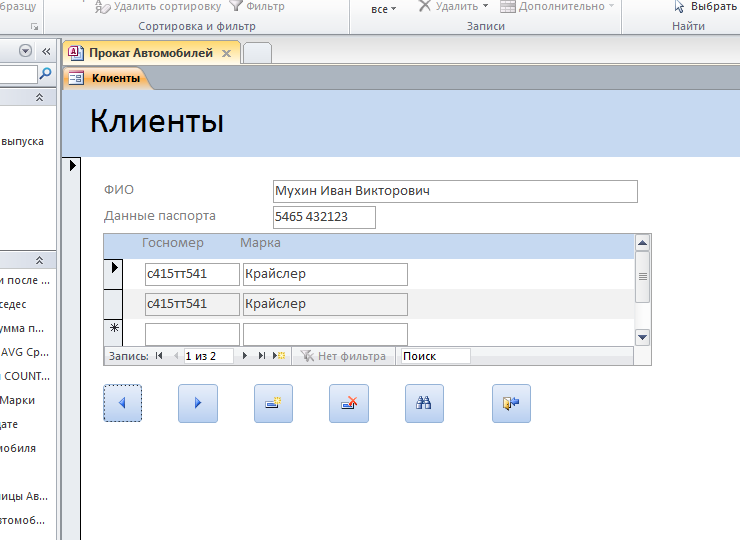


Рисунок 10 – форма с Подчиненной формой

# 8 Создание запросов

Запросы – это специальные средства для отбора и анализа данных, служат для извлечения данных из таблиц, предоставления их в удобном для пользователя виде. С помощью запросов выполняются операции по отбору данных, сортировке, фильтрации. Запросы позволяют преобразовать данные по заданному алгоритму, создавать новые таблицы, выполнять автоматическое наполнение таблиц данными, импортируемыми из других источников, выполнять простейшие вычисления в таблицах и т.д. Особенность запросов состоит в том, что они черпают данные из базовых таблиц, создавая на их основе временную результирующую таблицу.

Запросы создаются и редактируются в режиме Конструктор. Для этого выбирается вкладка Создание – Запросы – Конструктор запросов. В окне «Добавление таблицы» выбираются нужные для запроса таблицы, затем нужные поля.

В курсовом проекте были созданы различные типы запросов, которые рассмотрены ниже.

## 8.1 Запрос на выборку

Запрос позволяет получить информацию согласно конкретному критерию отбора.

Пример: вывести данные по конкретной марке – Мерседес.

Созданный запрос в режиме Конструктора, а также результаты его работы представлены на Рисунке 11 и 12.

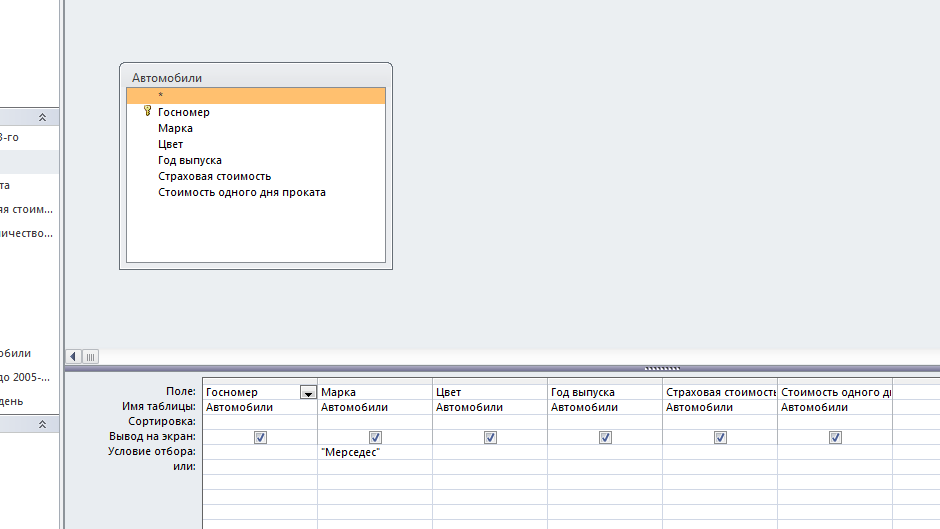


Рисунок 11 – Запрос на выборку

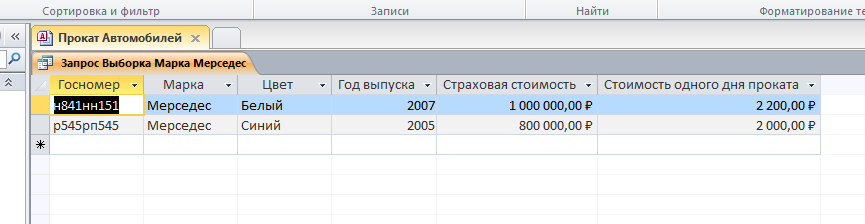


Рисунок 12 – Результат выполнения запроса на выборку

## 8.2 Запрос на выборку с параметром

Запрос позволяет пользователю многократно запускать запрос, причем при запуске можно каждый раз вводить новый критерий отбора данных. Критерий отбора вводится в специальное окно для ввода параметров отбора, которое появляется после запуска запроса.

Запрос создается аналогично запросу на выборку. Строка «Условие отбора» для нужного поля имеет вид: [текст]. В квадратных скобках записывается произвольный текст, поясняющий критерий отбора (что вводить).

Пример: создать запрос на отображение всех автомобилей определённого цвета.

Строка Условие отбора для нужных полей имеет вид: [Введите цвет автомобиля].

При запуске такого запроса появляется диалоговое окно с полем, где следует ввести критерий отбора. Условий отбора может быть несколько (Рисунок 13).

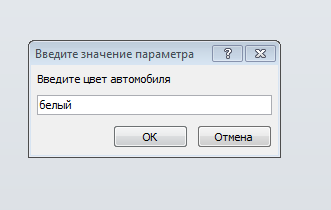


Рисунок 13 – Критерий отбора

Результат запроса представлен на Рисунке 14.

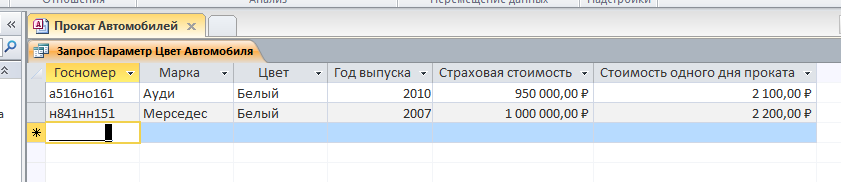


Рисунок 14 – Результат запроса с параметром

## 8.3 Итоговые запросы

Запросы позволяют производить математические вычисления по заданному полю с выдачей результатов.

Пример: определить общее количество прокатов по каждому клиенту.

После выбора таблиц и полей используется кнопка Итоги (Работа с запросами – группа «Показать или скрыть» – Итоги), в нижней части бланка запроса появляется новая строка Групповая операция. В этой строке в поле Количество из раскрывающегося списка выбирается нужная функция (Count).

Созданный запрос в режиме Конструкторf представлен на Рисунке 15.

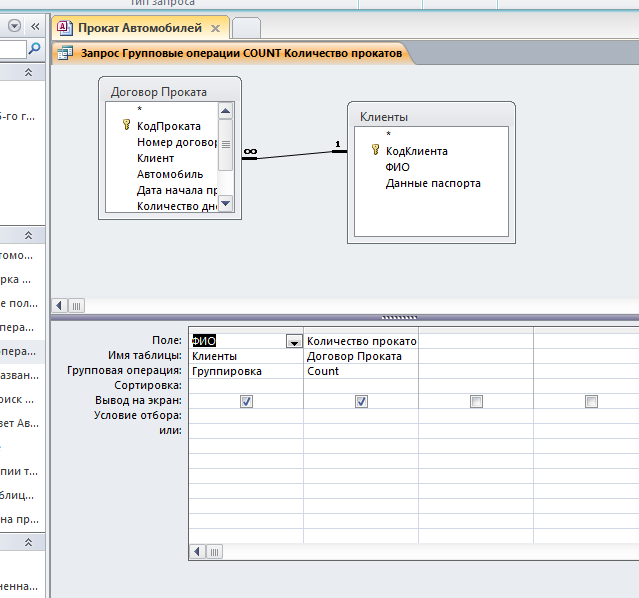


Рисунок 15 – Итоговый запрос в режиме Конструктора

## 8.4 Запрос на создание нового вычисляемого поля

Запросы позволяют создавать новые поля, выполняющие специальные вычисления в таблице на основании данных других полей. После выбора полей, в новом поле вводится вычисляемое выражение.

Пример: создать новое поле «Сумма проката» с помощью запроса.

Для создания запроса выбрать таблицы, затем поля из них. В новом поле ввести выражение (Рисунок 16).



Рисунок 16 – Выражение вычисляемого поля

При создании запроса формирование вычисляемого выражения осуществлялось с использованием Построителя выражений (кнопка Построитель в группе Настройка запроса), с помощью которого можно выбирать «графически» поля из таблиц базы данных, функции.

Созданный запрос в режиме Конструктор представлен на Рисунке 17.

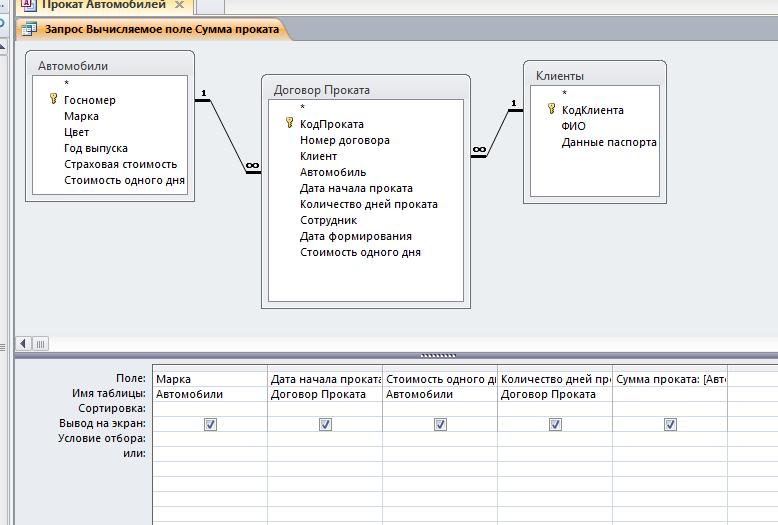


Рисунок 17 – Запрос на создание вычисляемого поля в режиме Конструктора

## 8.5 Запрос на удаление записей

Запрос позволяет удалить записи из таблиц. Такой запрос может быть построен либо на основании запроса на выборку, либо на основании запроса с параметром.

Пример: удалить записи автомобилей стоимость проката которых выше определённой суммы в день.

После выбора таблицы и нужного поля в строке Условие отбора ввести строку вида: >[Введите стоимость]. Запрос построен на основании запроса на выборку с параметром. Далее следует изменить тип запроса - Удаление (группа Тип запроса), после чего появляется новая строка Удаление с параметром Условие.

Созданный запрос в режиме Конструктор представлен на Рисунке 18.

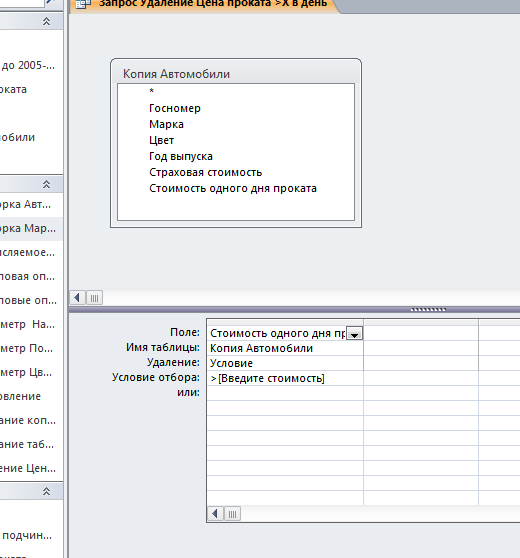


Рисунок 18 – Запрос на удаление в режиме Конструктора

## 8.6 Запрос на обновление записей

Запрос позволяет обновить информацию в группе записей. Такой запрос может быть построен либо на основании запроса на выборку, либо на основании запроса с параметром.

Пример: стоимость дня проката автомобилей после 2005-го года выпуска включительно увеличивается на 15%.

Выбирается таблица, где находится поле для замены. Для построения запроса из таблицы выбрать поля Стоимость одного дня проката. Изменить тип запроса – Обновление (группа Тип запроса). В появившейся новой строке Обновление для поля Стоимость одного дня проката ввести выражение: [Стоимость одного дня проката] + [Стоимость одного дня проката] \*0,15. В поле года выпуска поставить условие отбора: >=2005.

Созданный запрос в режиме Конструктор представлен на Рисунке 19.

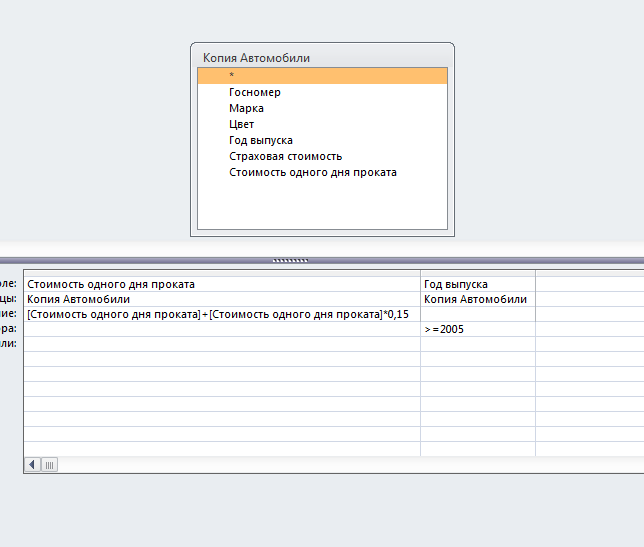


Рисунок 19 – Запрос на обновление в режиме Конструктора

## 8.7 Перекрестный запрос

Запрос позволяет получить сводную информацию из полей разных таблиц, отображает итоговые значения, получаемые из заданных полей (сумма, количество значений, среднее значение). Поля (от одного до трех) используются для заголовков строк с левой стороны запроса, заголовков столбцов (одно поле) и значений внутри перекрестной таблицы (одно поле). Результатом работы запроса всегда является таблица. При формировании запроса можно использовать вычисляемые поля.

Пример: создать перекрёстный запрос с информацией по доходу с клиента за всё время.

После выбора нужных таблиц и полей следует изменить тип запроса - Перекрестный (группа Тип запроса). Далее выбрать поля для заголовков строк (2 поля) для заголовков столбцов (1 поле) и одно поле для значений.

Созданный запрос в режиме Конструктор (Рисунок 20), а также результаты его работы (Рисунок 21) представлены ниже.

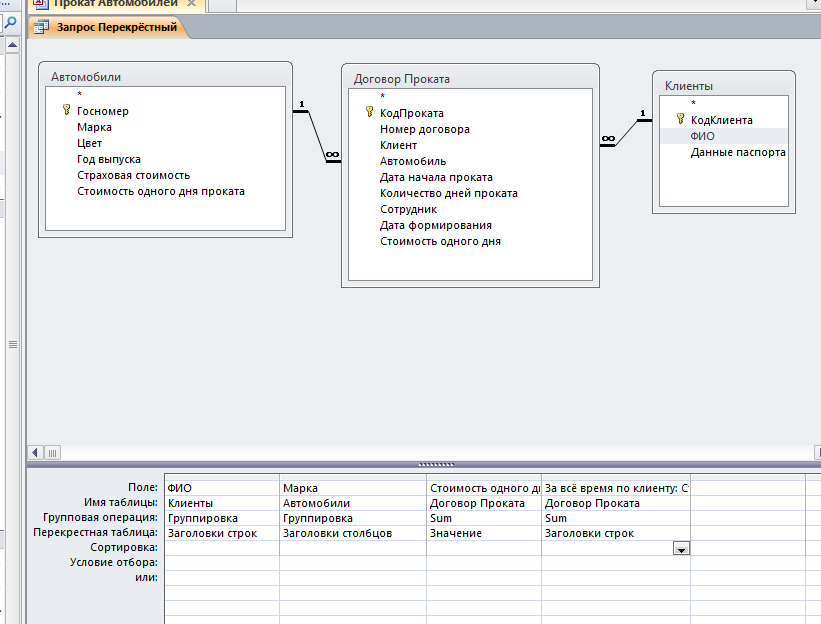


Рисунок 20 – Перекрестный запрос в режиме Конструктора

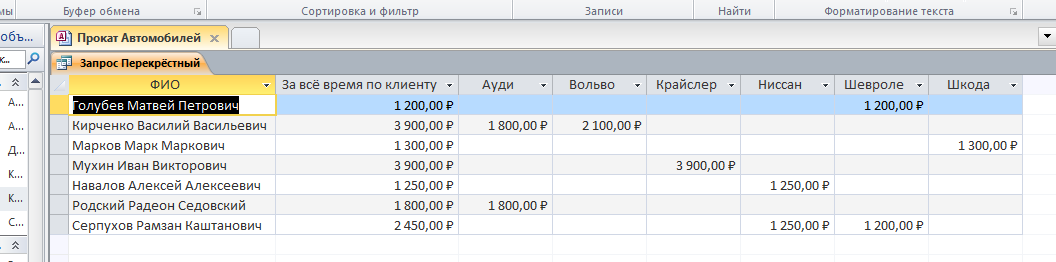


Рисунок 21 – Перекрестный запрос в режиме Таблицы

## 8.8 Запрос на создание новой таблицы

Запрос создает новую таблицу для базы данных на основании данных, собираемых вместе согласно определенному критерию, из существующих таблиц базы.

Пример: создать новую таблицу с данными по автомобилям до 2005-го года выпуска.

Запрос создается подобно запросу на выборку. После выбора таблиц и полей необходимо изменить тип запроса – Создание таблицы (группа Тип запроса). В диалоговом окне «Создание таблицы» указать имя новой таблицы и режим ее создания в текущей базе данных.

После запуска запроса и после подтверждения добавления записей новая таблица будет создана. Запрос представлен на Рисунке 22.

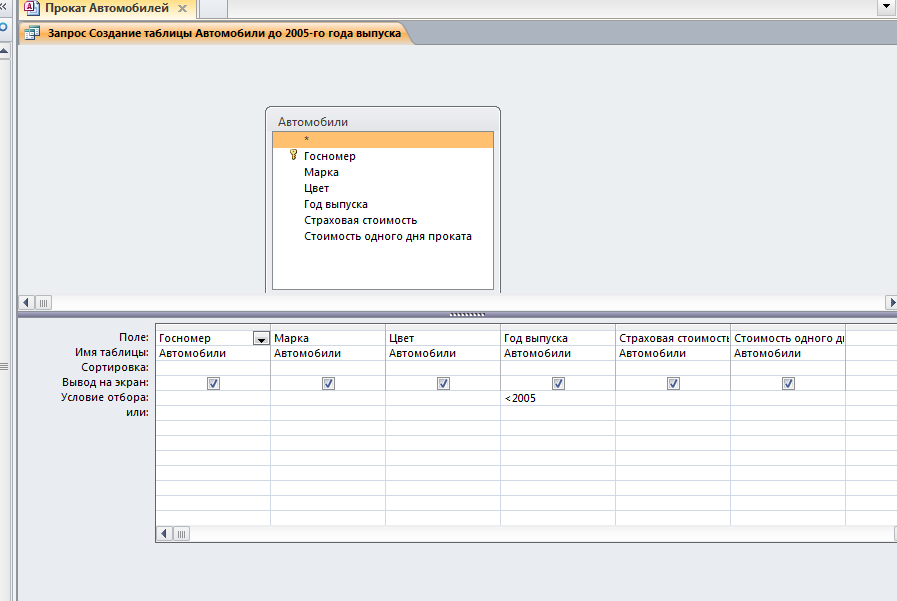


Рисунок 22 – Запрос на создание новой таблицы

# 9 Создание отчетов

Отчет создается для вывода данных на печать. Структура отчета может быть сформирована на основе таблиц и на основе запросов. В отчетах можно группировать информацию по нескольким уровням по любому выбранному полю, выполнять сортировку по нескольким полям, выполнять вычисления по заданному полю.

Для автоматического создания отчета выбирается вкладка Создание – группа Отчеты – Отчет. В режиме Конструктор отчет можно редактировать, используя область заголовка отчета, раздел верхних и нижних колонтитулов, область данных и область примечаний отчета. В отчете для вывода текущей даты, времени и подсчета числа страниц отчета используются функции Date(), Time() и Page. В отчет можно помещать вычисления. Для этого используется элемент управления типа «Поле», выбираемый на панели элементов. Элемент «Поле» должен размещаться в области «Примечание отчета». Внутри поля вводится формула для вычисления вида: = имя функции ([имя поля]). Созданный отчет обычно представлен в режиме предварительного просмотра.

В работе были созданы отчеты по основным запросам (на выборку, на выборку с параметром, итоговый, запрос на создание нового поля, создание новой таблицы) и отчеты по основным таблицам базы данных.

Далее представлен отчёт по прокату в режиме Предварительного просмотра (Рисунок 23) и режиме Конструктора (Рисунок 24).

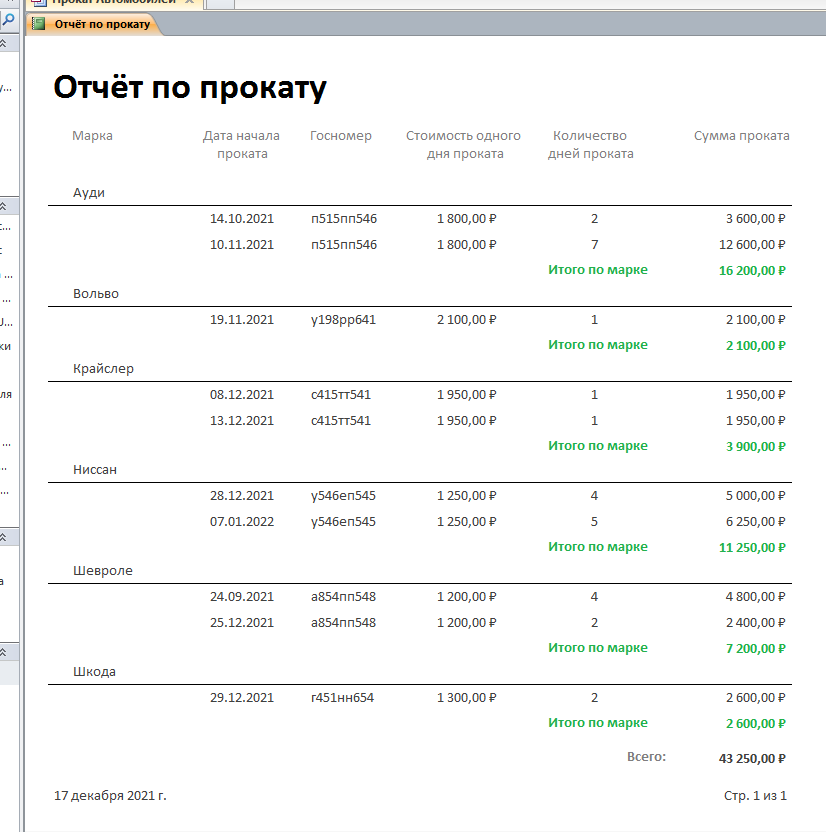


Рисунок 23 – Отчёт по прокату в режиме предварительного просмотра

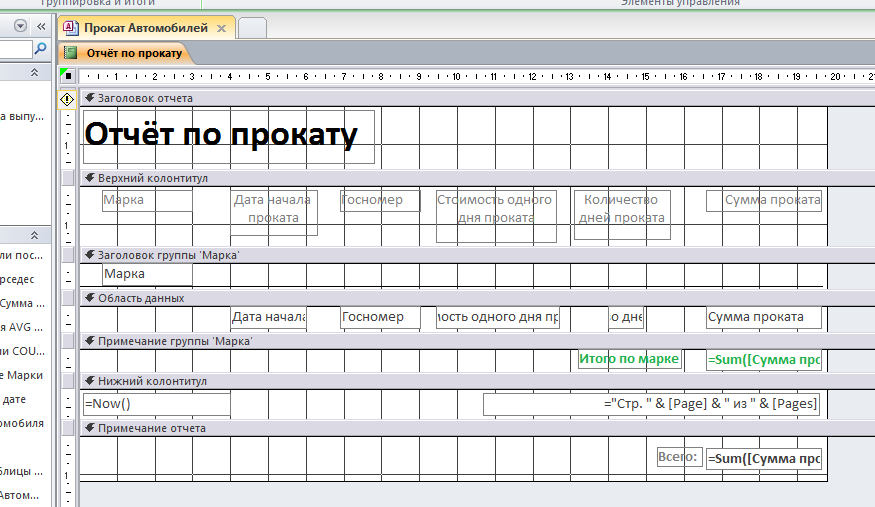


Рисунок 24 – Отчёт по прокату в режиме Конструктора

Отчет может быть создан с помощью Мастера отчетов, в этом случае он может быть создан на основе нескольких таблиц или запросов и соответственно в нем могут быть поля из нескольких таблиц или запросов. Для создания отчета выбирается вкладка Создание – группа Отчеты – Мастер отчетов. В появившемся окне «Создание отчетов» выбираются из таблицы поля, которые войдут в будущий отчет. Затем в поле «Таблицы и запросы» можно выбрать другую таблицу, поля в ней. Далее последовательно при необходимости выбираются уровни группировки данных по заданному полю, порядок сортировки записей, вид макета для отчета, ориентация. На последнем шаге работы с Мастером отчетов задается имя отчета.

# 10 Проектирование кнопочной формы

С помощью кнопочной формы можно получить быстрый доступ к созданным ранее формам и отчетам.

Для создания новой кнопочной формы или изменения существующей служит Диспетчер кнопочных форм.

Перед созданием кнопочной формы разрабатывается ее эскиз: определяется количество страниц кнопочной формы, элементы (или кнопки), которыми будут наполнены эти страницы, команды (действия), закрепленные за этими кнопками, подписи к кнопкам и т.д. По умолчанию создаваемая кнопочная форма уже имеет одну главную страницу (ее имя по умолчанию «Главная кнопочная форма»). Эта страница (форма) не содержит кнопок (элементов). Проектировщик базы данных сам определяет количество страниц (количество форм) в кнопочной форме, которые можно будет вызвать с главной страницы путем нажатия на соответствующие кнопки.

Созданная кнопочная форма состоит их трех страниц. Первая (главная) страница существует по умолчанию и обеспечивает переход к другим страницам. Созданы новые две страницы. Страница «Формы» (вторая страница) содержит кнопки, дающие доступ к созданным ранее формам для просмотра и редактирования записей таблиц базы данных, а также кнопки только для добавления записей в таблицы базы данных. Страница «Отчеты» (третья страница) содержит кнопки, дающие доступ к созданным ранее отчетам. При создании кнопочной формы были выполнены следующие этапы:

- 1 этап. Создание страниц кнопочной формы.

Для этого вызывается Диспетчер кнопочных форм и с помощью кнопки «Создать» в поле «Имя страницы кнопочной формы» вводится имя первой страницы (Формы). Аналогично создается и вторая страница («Отчёты»).

- 2 этап. Оформление элементами главной кнопочной формы.

В главной кнопочной форме предусмотрены четыре кнопки, с помощью которых можно будет перейти к другим страницам кнопочной формы, отредактировать кнопочную форму, выйти из приложения. Для этого выбрать главную страницу и кнопку «Изменить», при этом появится окно «Изменение страницы кнопочной формы», где выбрать кнопку «Создать» (появится окно «Изменения элемента кнопочной формы»). В соответствующих полях (путем ввода или выбора из раскрывающегося списка) вводится текст подписи к кнопке, выбирается из списка нужное действие (команда, кнопочная форма, форма, отчет). Например, для создания кнопки, осуществляющей переход к странице «Формы», поля в окне «Изменения элемента кнопочной формы» могут иметь вид:

Текст: Работа с формами

Команда: Переход к кнопочной форме

Кнопочная форма: Формы

Согласно описанной выше технологии, создается кнопка «Отчёты». Кнопка «Редактирование кнопочной формы» будет содержать только два поля: текст подписи к кнопке и выбранную из списка команду «Конструктор приложения». Кнопка «Выход из БД» содержит тоже два поля: текст подписи к кнопке и выбранную из списка команду «Выйти из приложения». После создания кнопок выбирается кнопка «Закрыть» и осуществляется переход назад к страницам кнопочной формы.

- 3 этап. Оформление страницы «Формы».

Для создания кнопок, дающих доступ к нужным формам для просмотра, редактирования записей или только добавления записей выбирается страницу «Формы», кнопка «Изменить», кнопка «Создать» и в соответствующих полях следует ввести или выбрать из списка:

Текст: <любой текст надписи к кнопке>

Команда: «Открыть форму для изменения»

Форма: <выбрать имя формы из списка>

Подобным образом создаются кнопки, дающие доступ к другим формам. Для создания кнопки, позволяющей только добавлять записи в таблицы базы данных без просмотра существующих записей, вместо команды «Открыть форму для изменения» выбирается команда «Открыть форму для добавления». Для перехода со страницы «Формы» назад к главной (первой) странице создается кнопка «Переход на главную страницу», где в качестве команды из списка выбирается команда «Перейти к кнопочной форме» и в поле «Кнопочная форма» выбирается страница «Главная кнопочная форма». Оформление страницы заканчивается нажатием на кнопку «Закрыть».

- 4 этап. Работа со страницей «Отчеты».

Кнопки создаются по аналогии с созданием кнопок для страницы «Формы». В качестве команды выбирается из списка команда «Открыть отчет», в поле «Отчёт» выбирается отчёт из списка существующих. Текст надписи к кнопке разъясняет ее назначение в соответствии с содержимым отчёта. Для перехода со страницы «Отчеты» к главной странице создается кнопка «Переход на главную страницу».

Создание кнопочной формы (работа с диспетчером кнопочных форм) заканчивается кнопкой «Закрыть». Среди объектов «Формы» появляется новая форма Кнопочная форма. В режиме Конструктор кнопочную форму можно редактировать, используя инструменты группы Элементы управления. Кнопочную форму можно запускать и работать со страницами, просматривая данные таблиц базы данных через формы и отчеты, созданные на основании запросов или табличных данных.

- 5 этап. Автоматический запуск кнопочной формы.

Для автоматического запуска кнопочной формы при открытии базы данных выполняются команды Файл - Параметры – Текущая база данных и в поле Форма просмотра вместо параметра «Отсутствует» задается параметр Кнопочная форма. После задания установки на автоматический запуск кнопочной формы при открытии базы данных появляется Главная кнопочная форма. Путем выбора нужных кнопок на этой странице осуществляется переход на другие страницы для получения быстрого доступа к нужным формам и отчётам. Кнопочная форма представлена на Рисунке 25.

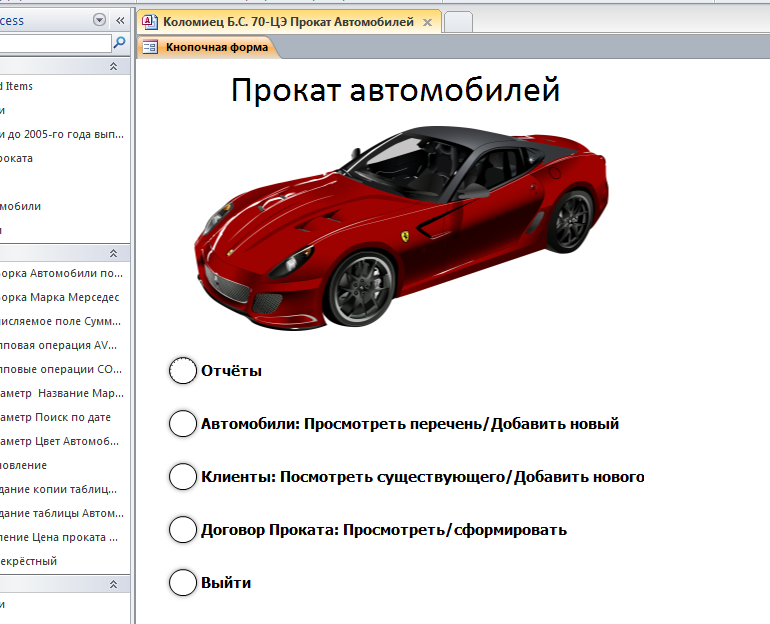


Рисунок 25 – Главная страница кнопочной формы

# Вывод

При помощи СУБД MSAccess удалось поэтапно реализовать поставленную в данной работе задачу, а также решить комплекс задач, вытекающих из неё. Так же в соответствие с предметной областью была построена концептуальная модель. База данных «Прокат автомобилей» состоит из всех необходимых элементов полноценной БД, а потому ее можно рассматривать как единое и отлаженное приложение, соответствующее всем предъявляемым требованиям. Созданная база данных обеспечивает хранение информации, отражающей основные параметры работы фирмы проката авто, а также дополнительные сведения о деятельности.

С помощью программы MicrosoftAccess были созданы необходимые объекты базы данных: таблицы, запросы, формы, отчёты, а также главная кнопочная форма. Использование созданной базы данных позволит упростить работу с большими массивами данных для получения информации о деятельности агентства проката автомобилей. Созданные запросы позволят быстро находить нужную информацию согласно заданному условию поиска. Пользователь имеет возможность добавлять новые, изменять старые записи.

Таблицы базы данных взаимосвязаны, что позволяет обеспечить её целостность и возможность обновления. Важным средством, облегчающим работу с MSAccess для начинающих проектировщиков, являются мастера – специальные программные надстройки, предназначенные для создания объектов базы данных в режиме последовательного диалога. Для опытных и продвинутых пользователей существуют возможности более гибкого управления ресурсами и возможностями объектов СУБД в режиме конструктора.

В результате выполнения курсового проекта, на основании задачи, поставленной в работе, разработана база данных, которая может быть использована на практике. Как и всякое другое программное приложение, она может быть дополнена и отредактирована в процессе эксплуатации.

# Библиографический список

1. Гагарина, Л.Г. Разработка и эксплуатация автоматизированных информационных систем: учеб. ИД Форум: Инфра – М, 2007.
2. Белянин М.В. Создание баз данных в Access 2007. – М.: НТ Пресс, 2008.
3. Буч Г., Якобсон А., Рамбо Дж. UML. Классика CS / С. Орлов. – 2-е изд. – СПб.: Питер, 2006.
4. Национальная библиотека им. Н.Э. Баумана: Erwin Data Modeler [Эл. ресурс]. – Режим доступа: https://ru.bmstu.wiki/Erwin\_Data\_Modeler
5. Сеннов А. Access 2010. Учебный курс. - СПб: Питер, 2010.
6. Информатика. Базовый курс: Учебник для вузов. 3-е изд. Стандарт третьего поколения. / Под ред. С.В. Симонович. – СПб: Питер, 2013.
7. Информатика и информационные технологии: учеб.пособие. / Под ред. Ю.Д. Романовой. – М: Эксмо, 2014.