Министерство образования и науки Украины

Харьковский национальный университет радиоэлектроники

Кафедра ПИ

61166, Харьков, пр. Ленина, 14

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОЙ РАБОТЕ

По дисциплине «Разработка библиотек программных компонентов»

Тема работы: «Программная система TravelQuestions»

Выполнил Лапошко С. К.

Ст. гр. ПИ-12-6

Руководитель:

Лещинская И. А.

Комиссия:

Зав. каф. ПИ, проф., Дудар З.В.

Доц. каф. ПИ Мазурова О.О.

Лещинская И. А.

Харьков, 2015

\_\_\_Харьковский национальный университет радиоэлектроники\_\_\_\_\_\_\_

Кафедра \_\_\_Программной инженерии\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дисциплина \_\_\_\_Разработка библиотек программныз компонентов \_\_\_

Специальность \_\_\_\_\_ Программной инженерии \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Курс \_\_\_3\_\_\_\_\_ Группа \_\_\_ПІ-12-6\_\_\_ Семестр\_\_\_\_ 6\_\_\_\_

**ЗАДАНИЕ**

на курсовую роботу студента

Лапошко Сергея Константиновича\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1 Тема работы: «Программная система TravelQuestions»

2 Срок сдачи законченной работы\_\_\_\_ 6.06 \_\_\_\_\_\_\_\_

3 Исходные данные для работы: постановка задачи, предметна область, методические указания к выполнению курсовой работы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

4 Содержание расчетно-пояснительной записки: вступление, анализ предметной области; постановка задачи; проектирование базы данных; описание программы; выводы; перечень ссылок.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5 Перечисление графического материала: концептуальна модель, ER-диаграмма, отношения в 1НФ, 2НФ, 3НФ, UML-диаграммы, скриншоты прикладной программы, примеры отчетов прикладной программы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6 Дата выдачи задания\_\_\_ 10.03.2015 \_\_\_ TODO

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| Номер | Название этапов курсовой работы | Срок выполнения этапов курсовой работы | Примечания |
| 1 | Анализ предметной области | 10.03 – 20.03 | Выполнено |
| 2 | Постановка задачи | 15.03 – 20.03 | Выполнено |
| 3 | Проектирование базы данных | 20.03 – 29.03 | Выполнено |
| 4 | Программная реализация | 29.03 – 30.04 | Выполнено |
| 5 | Тестирование программы и наполнение данными | 10.04 – 30.04 | Выполнено |
| 6 | Оформления пояснительной записки | 1.03 – 15.05 | Выполнено |
| 7 | Защита курсового проекту | 6.06 |  |

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель Лещинская И. А.

\_\_ \_\_\_\_\_\_\_\_\_ 2015 р.

РЕФЕРАТ

Пояснительная записка: 43 с., 26 рис., 3 табл., 10 источников.

Целью работы является разработка информационной системы для туристов, которая облегчит процессы хранения, использования и обмена информацией между пользователями.

Метод разработки – концептуальное, ER- и UML-моделирование предметной области, объектно-ориентированный подход к разработки программного приложения.

БАЗА ДАННЫХ, ОТНОШЕНИЕ, СУБД, ЗАПРОС, ТУРИЗМ, ПОХОД, ИНСТРУКТОР, ВОПРОС, ОТВЕТ

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 6

1 АНАЛИЗ И КОНЦЕПТУАЛЬНОЕ МОДЕЛИРОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 8

2 ПОСТАНОВКА ЗАДАЧИ 13

3 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 16

3.1 UML-моделирование предметной области 16

3.2 Построение ER-диаграммы 19

3.3 Построение схемы реляционной базы данных в третьей

нормальной форме 20

4 ОПИСАНИЕ ПРОГРАММЫ 26

4.1 Общие сведенья 26

4.2 Вызов и загрузка 26

4.3 Назначение и логическая структура 27

4.4 Описание физической модели базы данных 28

4.5 Описание программной реализации 31

4.6 Описание задачи автоматизации 40

ВЫВОДЫ 42

ПЕРЕЧЕНЬ ССЫЛОК 43

ВВЕДЕНИЕ

Областью применения данного курсового проекта является туристический сектор. Основными пользователями есть туристы и люди, которые просто любят путешествовать.

Проблема этой области состоит в том, что на данный момент некоторые люди боятся путушествовать из-за того, что много не знают о месте, куда они хотят поехать, не используя новые технологии, которые могут облегчить процессы обмена и хранения информации.

Использование базы данных позволит более удобно обмениваться информацией и хранить ее. База данных может хранить в себе информацию о клиентах, вопросах, ответах, горотдах, странах. При этом она сможет предоставить удобный доступ к этой информации. Поиск, фильтрацию и сортировку данных.

Программная система сможет обеспечить пользователя часто необходимой информацией. Для того, чтобы узнать, какие какую-то информацию о городе, не нужно будет просматривать все страницы об этом городе в Интернете, или обзванивать всех знакомых. Можно просто добавить вопрос на сайт и другие пользователи смогут ответить на этот вопрос. Также она может предложить ответить на существующий вопрос или просмотреть информацию о прошлох вопросах.

Хранение всех данных в одной базе обеспечит целостность информации. Если информация про один вопрос поменялась, не будет необходимости проверять все места, где информация о нем была скопирована. Достаточно будет один раз исправить эту информацию, а в остальных местах система автоматически изменит данные на более актуальные.

Когда создается маршрут, который проходит через несколько мест, можно выбирать места, которые ранее уже использовались в программе. Если информация о месте изменится, то программа скорректирует информацию о маршруте.

Из приведенной оценки состояния предметной области можно сделать вывод об актуальности разрабатываемой программы. Она существенно облегчит деятельность людей в сфере туристического бизнеса, выполняя за них большинство повседневных задач.

Таким образом целью данного курсового проекта есть спроектировать базу данных, выбрать и использовать СУБД для работы с данными, которые лежат в базе, а также разработать программную систему, которая позволит пользователю взаимодействовать с данными через удобный интерфейс программы. Еще целью является создание хорошо спроектированной системы, которая будет масштабируемой и безопасной.

В качестве СУБД была выбрана MySQL из-за ее быстродействия, наличия бесплатных лицензий и поддержки параллельных запросов. Для реализации информационной системы будет использован Spring Framework, AngularJS. Эта технология позволяет довольно просто создавать приложения с удобным и простым интерфейсом. В качестве интегрированной среды разработки была выбрана InteliJ IDEA из-за встроенной поддержки Java и MySQL.

1 АНАЛИЗ И КОНЦЕПТУЛЬАНОЕ ПРОЕКТИРОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

Исследуемая система представляет собой туристическую сектор, которая организовывает походы для туристов разных уровней. Клиент может отправлятся в какое-то путешествие. После этого туристу будет предложено заполнить заявление, с указанием информации о себе и документах, необходимых для регистрации в спасательных. У каждого похода есть свой инструктор, который поведет группу. Заранее известна дата начала похода, его длительность и стоимость для одного человека. Поход имеет свой маршрут, по которому идет группа.

В предметной области есть города, которые принадлежат странам. Впорос может относиться к какому-то городу, также он должен быть привязан к какому-то пользователю системы.

У одного вопроса может быть много ответов.

Каждый ответ должен обязательно иметь пользователя, который его написал. У города могут быть места, которые расположены в этом городе.

Ответ обязательно должен быть связан с каким-то вопросом.

Все вопросы могу иметь список голосов. У каждого голоса должен быть пользователь, который его сделал.

Все ответы могу иметь список голосов. У каждого голоса за ответ должен быть пользователь, который сделал этот голос. На основании длины этого списка формируется рейтинг вопроса или ответа. Ответы или вопросы с большим рейтингом имеют больший приоритет.

После исследования предметной области можно составить диаграмму взаимодействия (см. рис.1.1).

Из данной функциональной структуры можно сделать вывод, что в системе существует турист, который путешествует по миру. Также он может ходить в походы. Также у него могу быть вопросы. Вместе она представляют собой подсистему, которая взаимодействует между собой.

Необходимо выделить участников системы, которые принимают решения и их информационные нужды. Гости системы и зарегистрированные пользователи.

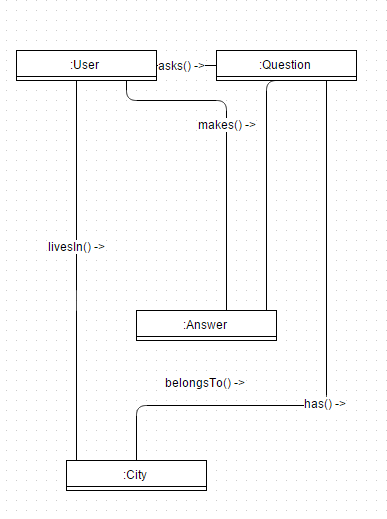


Рисунок 1.1 Функциональная структура исследуемой системы

Информационные нужды руководителя и администраторов:

* знать список вопросов;
* иметь информацию о городах;
* иметь информацию о местах;
* иметь возможность просмотреть список ответов для вопроса;
* знать дату вопроса и дату ответа;
* знать рейтинг вопроса и ответа;
* знать список вопросов для одного пользователя;

Далее будут рассмотрены объекты, которые должны храниться в базе данных и конкретная информация о них.

Клиент:

* фамилия;
* имя;
* пол;
* дата рождения (число.месяц.год);
* инфрмация о пользователе;
* фото;
* страна.

Город:

* название;
* фото;
* страна.

Страна:

* название;
* города.

Вопрос:

* заголов;
* текст;
* дата;
* автор.

После анализа предметной области были выявлены следующие ограничения целостности:

* клиент может быть идентифицирован по логину;
* поход может быть идентифицирован по дате начала и маршруту;
* вопрос и ответ имеют ссылку на клиента;
* ответ может иметь неограниченног количество голосов;
* вопрос может иметь неограниченное количество голосов.

Необходимо рассмотреть процессы потребления информации и существующие в системе документы. Информация поступает от клиента к программной системе в виде текста из формы с заполненной информацией об участнике вопросе[1]. Потом клиент может просматривать список все вопросов, своих вопросов, ответов на эти вопросы. Он может отвечать на другие вопросы. Таким образом пользователи обмениваются информацией между собой. Также они могут зайти на страницу друг другая, чтобы посмотреть список вопросов и информацию о другом пользователе.

2 ПОСТАНОВКА ЗАДАЧИ

В курсовом проекте необходимо спроектировать и реализовать программную систему. В этом разделе перечислены задачи, которые должна решать информационная система.

Система должна отображать данные страны, города, вопросы, ответы, клиентов и информации о них.

Система должна обеспечивать добавление нового вопроса, ответа, пользователя, страны.

Система должна предоставлять возможность просмотреть список вопросов во всем приложении, список вопросов по городам, список вопросов по одному пользователю или городу.

Система должна отображать список ответов к вопросу.

Система дожна предоставлять возможность поиска по ключевому слову среди списка всех вопросов.

Так же она должна предоставлять возможность отслеживать рейтинг вопросов и ответов.

В системе должна быть предусмотрена возможность зарегистрироваться.

Зарегистрированные пользователи должна иметь возможность войти в систему под своим логином.

После этого они должны иметь возможность проголосовать за какой-то вопрос или ответ.

Еще пользователи, которые вошли в систему, имеют возможность ответить на вопрос, на который они знают ответ.

Система должна отображать список ответов к вопросу.

Система дожна предоставлять возможность поиска по ключевому слову среди списка всех вопросов.

Зарегистрированные пользователи должна иметь возможность войти в систему под своим логином.

Система дожна предоставлять возможность поиска по ключевому слову среди списка всех вопросов.

В системе должна быть предусмотрена возможность зарегистрироваться.

Система должна предоставлять возможность просмотреть список вопросов во всем приложении, список вопросов по городам, список вопросов по одному пользователю или городу.

Система должна хранить информацию о пользователе. Ниже рассмотрен пример такой информации (см. таб. 2.1):

Таблица 2.1 – Пример данных о пользователе

|  |  |
| --- | --- |
| Имя | Иван |
| Фамилия | Иванов |
| Пол | М |
| Страна | Украина |
| Города | Харьков, Киев |
| Логин | Ivani |
| Пароль | 23423sdfk312 |
| Фото | http://localhost:8080/img/1.jpg |
| Обо мне | Родился в городе Харьков, проживаю сейчас в городе Киев. Знаю много об этих городах. Могу рассказать много интересного. Ответить на ваши вопросы касательно этих двух городов и Украины. |

Также система должна предоставлять возможность отсортировать ответы или вопросы по их рейтингу.

Система дожна предоставлять возможность поиска по ключевому слову среди списка всех вопросов. Зарегистрированные пользователи должна иметь возможность войти в систему под своим логином. Система дожна предоставлять возможность поиска по ключевому слову среди списка всех вопросов.

Требованием также есть создание трехуровневой масштабируемой системы.

В этом разделе были представлены задачи, которые требуют реализации. Также была выбрана задача автоматизации – регистрация пользователя по электронному заявлению.

3 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

3.1 UML – моделирование

На основе поставленных задач была построена диаграмма прецедентов (см. рис 3.1).

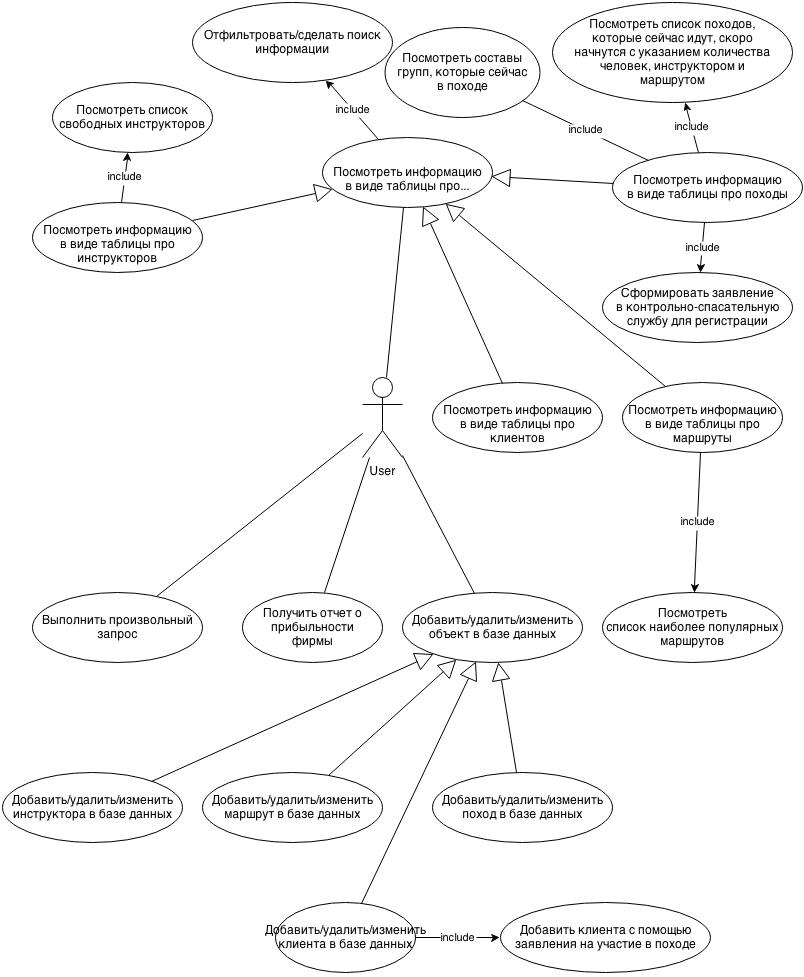


Рисунок 3.1 – Диаграмма прецедентов

Данная диаграмма отображает поставленные задачи. Как видно, в системе пользователи не разделяются, для системы существует только один вид пользователей. Из диаграммы мы узнаем, что пользователь может просмотреть информацию о сущности (следует отметить, что у этого элемента есть наследники, которые уточняют его и говорят о возможности посмотреть про конкретную сущность, например, про инструкторов и т.д.)[6]. С каждым элементом связан прецедент, который показывает возможность выполнить часто используемый запрос, подготовить заявление в контрольно-спасательную службу из окна просмотра походов, просмотреть список наиболее популярных маршрутов.

Кроме просмотра информации также существуют прецеденты, которые указывают на возможность удаления или редактирования информации. По аналогии с просмотром информации существуют наследники, которые конкретизируют сущность, над которой выполняется операция. Также существует элемент Use Case, который указывает на возможность добавления клиента с помощью заполненного заявления клиента.

Кроме перечисленных функций также следует отметить элемент «Выполнить произвольный запрос», который указывает на возможность вручную ввести интересующий запрос и получить по нему результат. Этот элемент не связан с другими, потому что его функциональность не зависит от других прецедентов.

Есть элемент «Получить отчет о прибыльности фирмы», который подразумевает собой возможность просмотреть или сохранить документ, который будет наполнен информацией о походах и данными о том, сколько какой поход принес денег и сколько было потрачено на дополнительные расходы и зарплату руководителю похода.

Было решено построить Activity diagram для задачи автоматизации (добавлению клиента по таблице –заявлению) (рис 3.2).

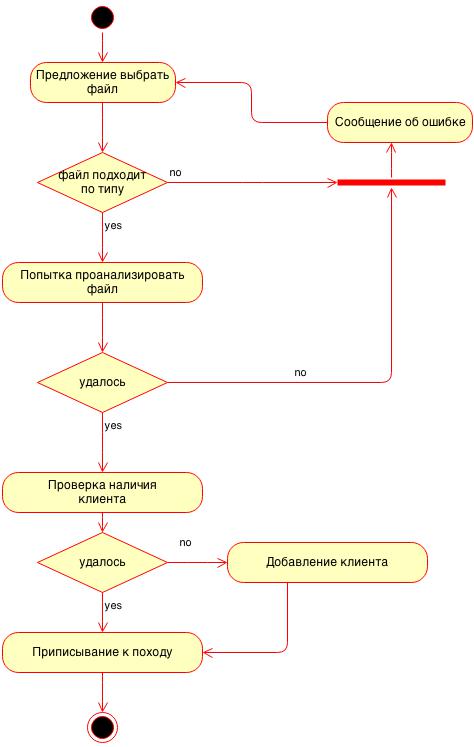


Рисунок 3.2 – Activity diagram

На этой диаграмме мы видим последовательно выполняемую задачу автоматизации. На первом этапе необходимо выбрать файл, в котором содержится информация о будущем клиенте. Файл представляет собой таблицу в Excel’е, в которой есть два столбца с названием данных и значением парами. Например, «Фамилия» – «Иванов». После эта информация анализируется и на ее основе происходит добавление нового участника.

3.2 Построение ER-диаграммы

Одним из этапов проектирования базы данных является построение ER-диаграммы [3]. На основе концептуального анализа была создана данная диаграмма (рис 3.3).

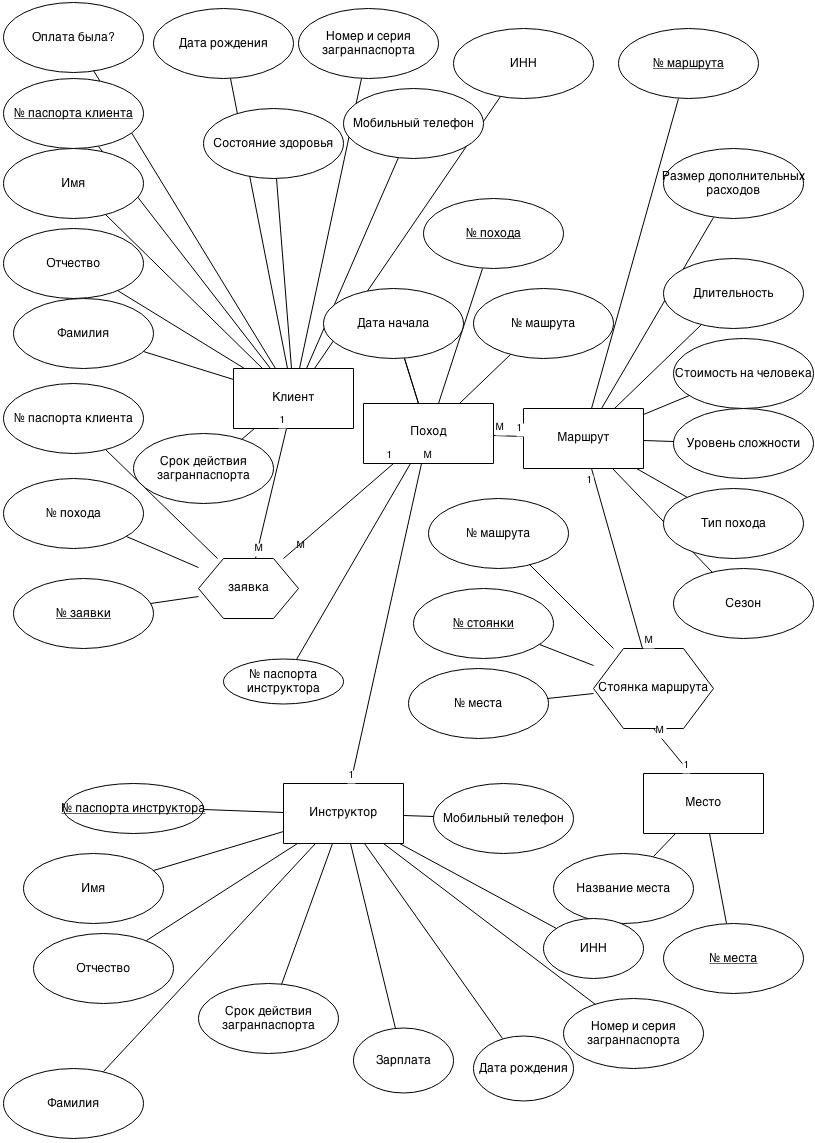


Рисунок 3.3 – ER-диаграмма

Выше представлена полная диаграмма с приведением всех атрибутов, которые будут у сущностей.

На данной диаграмме можно наблюдать связь многие ко многим между походом и клиентом, которая реализована с помощью дополнительной сущности «заявка». Ключом сущности «Клиент» есть № паспорта клиента, «Поход» - № похода, «заявка» - № заявки, кроме этого заявка содержит первичные ключи похода и клиента для связки.

Существует связь между инструктором и походом один ко многим, которая реализуется с помощью внесения в поход номера паспорта инструктора.

Также на диаграмме присутствует связь один ко многим (поход и маршрут с помощью поля «№ маршрута» у похода) и многие ко многим (маршрут и место через дополнительную сущность стоянка маршрута).

3.3 Построение схемы реляционной базы данных в третьей нормальной форме

Из концептуального анализа предметной области получили набор атрибутов, который следует нормализировать.

На рисунке 3.4 изображено отношение в первой нормальной форме [4]. Таблица в целях наглядности разделена на две, хотя является целой. Все атрибуты неделимы, нет таких атрибутов, как, например, ФИО, существует три поля, хранящие имя, фамилию и отчество. Также нет повторяющихся строк, существуют уникальные ключи. В качестве уникального ключа выступает пара {ЗЗаявка№, ССтоянка№}. По этим двум значениям можно определить значения всех остальных атрибутов.

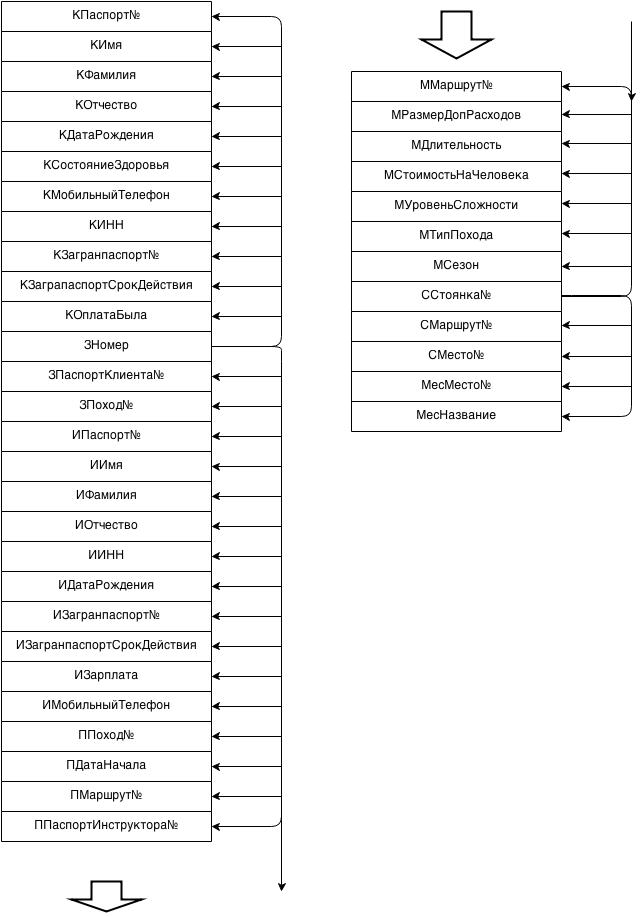


Рисунок 3.4 – Отношение в первой нормальной форме

Данное отношение не находится во второй нормальной форме, потому что есть неключевые атрибуты, которые зависят от частей составного ключа. Теперь перейдем к построению отношений во второй нормальной форме (рис 3.5)

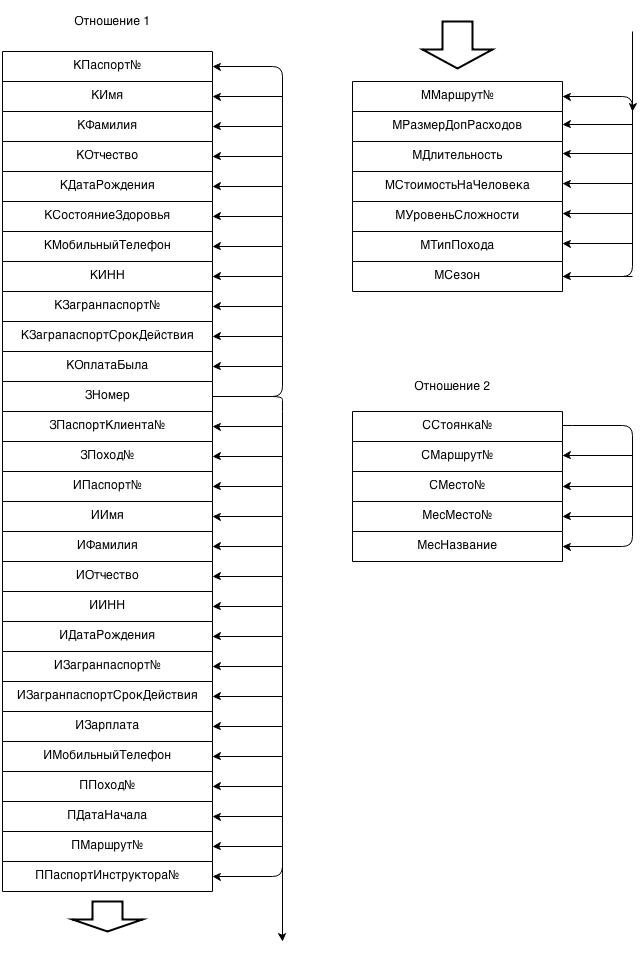


Рисунок 3.5 – Отношения во второй нормальной форме

Мы получили вторую нормальную форму, при этом мы разбили первоначальное отношение на два [5, 6]. Отношения находятся во второй нормальной форме, потому что невозможно выделить меньшее подмножество атрибутов в ключе, от которого бы зависела часть неключевых атрибутов, первичные ключи простые.

Теперь необходимо привести отношения к третей нормальной форме. Третья нормальная форма требует, чтобы неключевые атрибуты не находились в транзитивной функциональной зависимости от потенциального ключа. Это условие не выполняется. К примеру, по паспорту клиента мы можем определить его имя, фамилию, отчество и остальные данные о нем. Все такие зависимости были выделены и на их основе были составлены отношения в третьей нормальной форме (рис. 3.6).

Зависимости:

{КПаспорт№} = {КИмя, КФамилия, КОтчество, КДатаРождения, КСостояниеЗдоровья, КМобильныйТелефон, КИНН, КЗагранпаспорт№, КЗаграпаспортСрокДействия, КОплатаБыла}

{ЗНомер} = {ЗПаспортКлиента№, ЗПоход№}

{ИПаспорт№} = {ИИмя, ИФамилия, ИОтчество, ИИНН, ИДатаРождения, ИЗагранПаспорт№, ИЗагранПаспортСрокДействия, ИЗарплата, ИМобильныйТелефон}

{ППоход№} = {ПДатаНачала, ПМаршрут№, ППаспортИнструктора№}

{ММаршрут№} = {МРазмерДопРасходов, МДлительность, МСтоимостьНаЧеловека, МУровеньСложности, МТипПохода, МСезон}

{ССтоянка№} = {СМаршрут№, СМесто№}

{МесМесто№} = {МесНазвание}

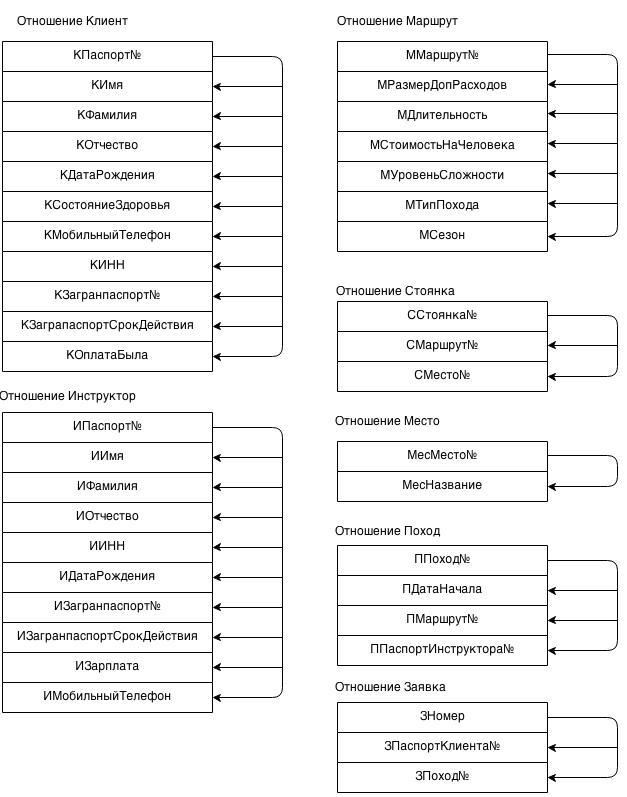


Рисунок 3.6 – Отношения в третьей нормальной форме

По итогами нормализации была получена схема базы данных, представленная на рисунке 3.7.

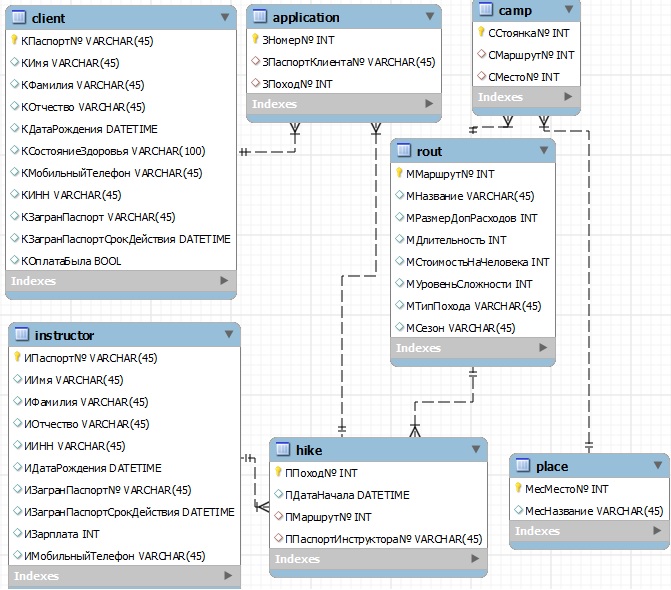


Рисунок 3.7 – Схема базы данных

4 ОПИСАНИЕ ПРОГРАММЫ

4.1 Общие сведенья

Разрабатываемый программный продукт создавался в интегрированной среде разработки Visual Studio 2012 Professional Edition. В качестве СУБД была выбрана MySQL. Ее основные преимущества:

* лучшая скорость обработки данных на объеме до 500000 записей;
* наличие бесплатных открытых лицензий;
* простота использования, поддержка многих платформ;
* поддержка нескольких одновременных запросов;
* наличие большого количества встроенных функций.

Также эта СУБД поддерживается Visual Studio, благодаря чему разработка стала более легкой и автоматизированной [7].

Требования к оборудованию:

* процессор с тактовой частотой 1 ГГц или выше;
* ОЗУ объемом 512 мегабайт или больше;
* 2 мегабайта свободно пространства на жестком диске;
* установленный .NET Framework 4.0.

4.2 Вызов и загрузка

Для того чтобы запустить программу необходимо открыть папку с программой и запустить исполняемый файл tourism.exe. После этого откроется главное окно программы. Информационная система сразу готова к использованию. Пользователю не нужно создавать псевдоним базы данных. Система сама подключается к ней тогда, когда ей это необходимо. При этом информационная система подразумевает, что база данных находится на localhost. Подключение осуществляется с помощью имени пользователя и пароля. Эта процедура выполняется неявно при необходимости получить какие-то данные из базы.

4.3 Назначение и логическая структура

В информационной системе были реализованы все функции, перечисленные в постановке задачи. Ниже приведен этот список и краткое описание модулей, которые отвечают за выполнение функции.

Отображение данных о клиентах, инструкторах, походах и маршрутах реализовано в MainForm.

Изменение и удаление информации о сущностях реализовано в соответствующих формах, таких как EditClientForm, EditRoutForm и т. д.

Для удаления данных предусмотрена кнопка в главном окне программы. Она удаляет информацию из базы данных о выбранной в таблице сущности.

Для поиска и фильтрации информации возле каждой таблицы отображения информации о сущностях расположены поля для ввода информации для поиска и фильтрации и соответствующие кнопки.

Функции узнать список туристов, которые сейчас находятся в походе, список наиболее популярных маршрутов, список походов, которые сейчас идут реализованы в модуле MainForm.

Можно узнать список инструкторов, которые можно привязать к походу можно. Эта функция реализована в модуле EditHikeForm.

Возможность формирования собственного запроса реализована в модуле QueryEditorForm.

Формирование отчета о прибыльности фирмы (см. табл. 2.1) реализовано путем создания самого отчета в файле ReportProfit.rdlc и размещения его на форме ProfitsForm.

Система также может сформировать заявку в контрольно-спасательную службу. С точки зрения программной реализации эта заявка представляет собой отчет, котором перечислена необходимая информация. Поэтому был создан отчет ReportApp.rdlc и размещен на форме CSSapplicationForm.

В информационной системе также реализована функция добавления клиента и привязки его к походу посредством указания excel файла с необходимой информацией (см. табл. 2.2). Разбор файла был реализован в модуле ExcelToDataTable [8], а само добавление в MainForm.

4.4 Описание физической модели базы данных

В качестве СУБД была выбрана MySQL, поэтому описание физической модели представлено в виде скриншотов из этой программы с описанием.

Структура таблицы «Клиент» приведена на рисунке 4.1:

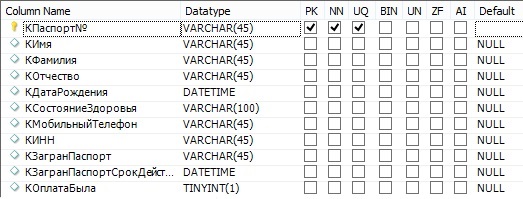


Рисунок 4.1 – Структура таблицы «Клиент»

На приведенном рисунке перечислены поля таблицы «Клиент». В качестве первичного ключа обозначено поле «КПаспорт№», на которое наложено ограничение не пустое и уникальное. Внешних ключей у этой таблицы нету, дополнительных ограничений наложено не было.

Структура таблицы «Инструктор» приведена на рисунке 4.2:

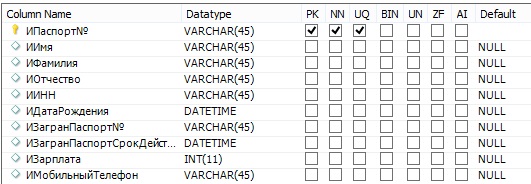


Рисунок 4.2 – Структура таблицы «Инструктор»

На приведенном рисунке перечислены поля из таблицы «Инструктор». В качестве первичного ключа обозначено поле «ИПаспорт№», аналогично таблице «Клиент». Других ограничений нет.

Таблица «Маршруты» приведена на рисунке 4.3:

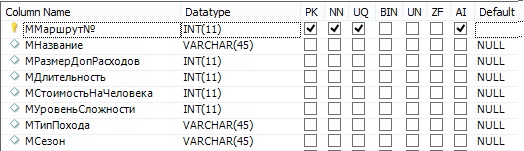


Рисунок 4.3 – Структура таблицы «Маршрут»

На рисунке выше приведена структура таблицы «Маршрут». В качестве первичного ключа выбрано поле «ММаршрут№», в котором помимо обычных ограничений также отмечено «AI», что означает, что поле автоматически растет. Это позволяет сделать целочисленный тип поля. Других ограничений на таблицу не наложено.

На рисунке 4.4 приведена структура таблицы «Поход»:

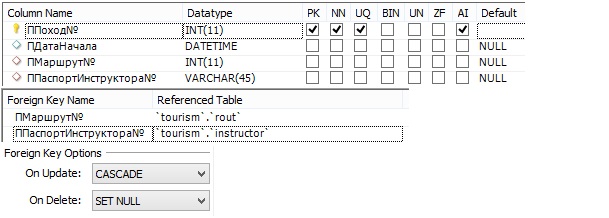


Рисунок 4.4 – Структура таблицы «Поход»

На приведенном рисунке описана структура таблицы «Поход». В качестве первичного ключа выбрано поле «ППоход№» с опцией автоматического инкрементирования. В этой таблице есть внешние ключи, которые ссылаются на таблицы «Маршрут» и «Инструктор». У внешних ключей выбраны опции: при удалении инструктора или маршрута выставить соответствующие поле в NULL, при обновлении – изменить.

На рисунке 4.5 приведена схема таблицы «Заявка»:

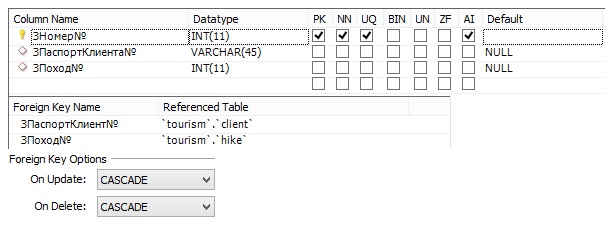


Рисунок 4.5 – Структура таблицы «Заявка»

В качестве первичного ключа выбрано поле «ЗНомер№». С внешними ключами выбраны опции каскадирования. При удалении клиента или похода, удаляется и сама заявка.

На рисунке 4.6 приведена схема таблицы «Стоянка»:

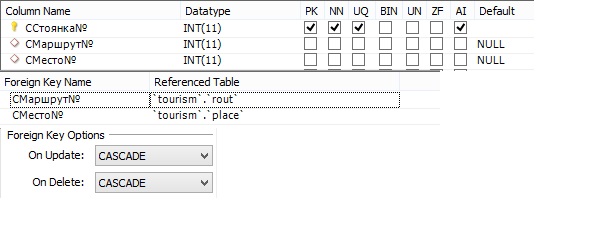


Рисунок 4.6 – Структура таблицы «Стоянка»

Дополнительные опции этой таблицы схожи с таблицей «Заявка».

На рисунке 4.7 представлена структура таблицы «Место»:

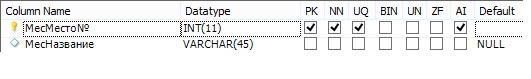


Рисунок 4.7 – Структура таблицы «Место»

4.5 Описание программной реализации

В этом разделе будут приведены экранные формы, которые реализуют перечисленные ранее функции.

Отображение данных о клиентах, инструкторах, походах и маршрутах реализовано в модуле MainForm. Экранная форма приведена на рисунке 4.8:

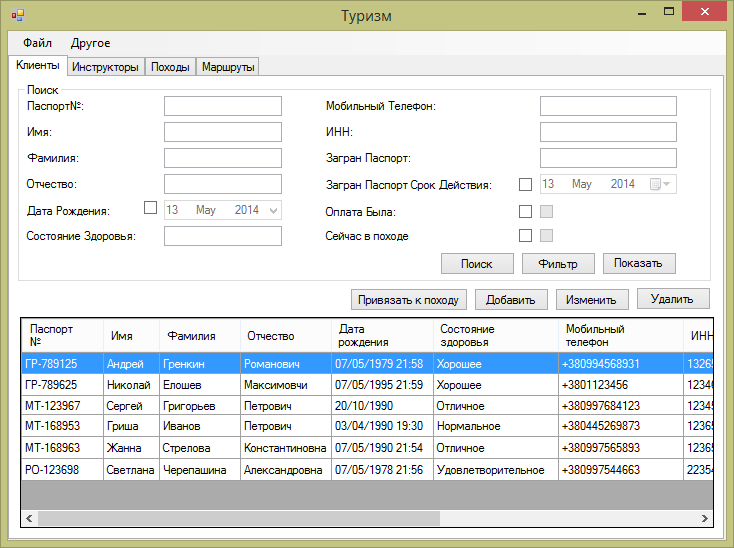


Рисунок 4.8 – Отображение данных о клиентах

Аналогично реализовано отображения информации об инструкторах, походах и маршрутах во вкладках главной формы.

Экранные формы изменения и удаления информации о сущностях приведены на рисунке 4.9. Также эти формы используются для добавления новых сущностей в базу данных:

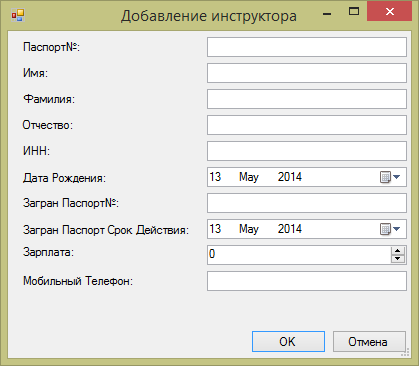


Рисунок 4.9 – Форма редактирования сущности

Следует обратить внимание на добавление походов, форма представлена на рисунке 4.10:

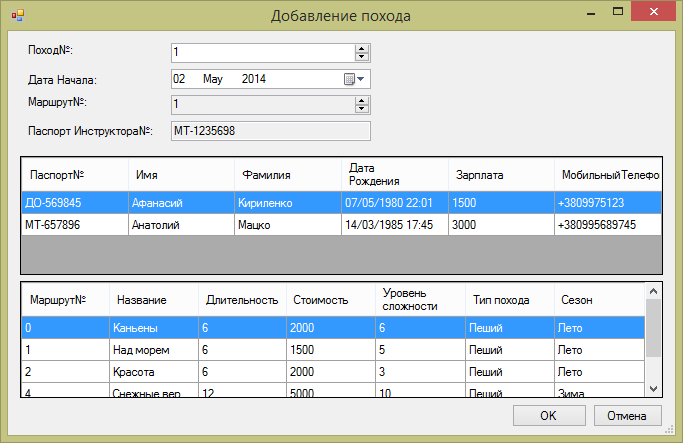


Рисунок 4.10 – Форма редактирования походов

В этой форме нужно внести информацию о походу, но для предотвращения ввода неправильных паспорта инструктора и номера похода, программа сама заполняет эти поля, а пользователи нужно выбрать лишь строчку в таблице с информацией об инструкторе или маршруте.

Еще одна интересная форма эта добавление и редактирование маршрута. Она представлена на рисунке 4.11:

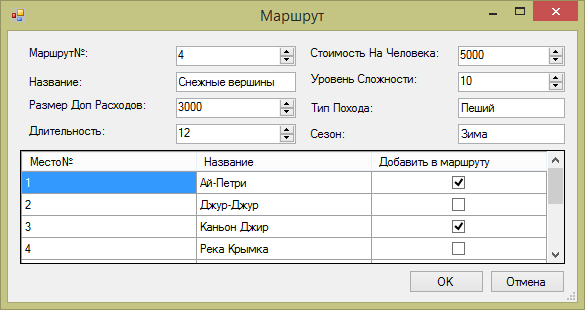


Рисунок 4.11 – Форма редактирования маршрутов

На приведенной выше форме пользователю необходимо выбрать места, через которые будет проходить маршрут и программа автоматически включит их в маршрут, создаст соответствующие стоянки. При редактировании маршрута можно снимать галочка, чтобы маршрут не проходил через данное место.

Для удаления данных предусмотрена кнопка в главном окне программы «Удалить» (см. рис. 4.8). При этом нужно выбрать в таблицу соответствующий элемент.

Для поиска и фильтрации информации возле каждой таблицы отображения информации о сущностях расположены поля для ввода информации для поиска и фильтрации и соответствующие кнопки (см. рис. 4.8). Пользователю необходимо ввести желаемые поля для поиска и нажать кнопку поиска или фильтрации. При этом некоторые поля могут быть неактивными (ввод даты или числа). Для этого нужно сначала их активировать, поставив галочку рядом с названием поля.

Список туристов, которые сейчас находятся в походе можно просмотреть с помощью фильтрации, отметив галочку «Сейчас в походе» (см. рис. 4.8).

Форма со списком наиболее популярных маршрутов представлена на рисунке 4.12:

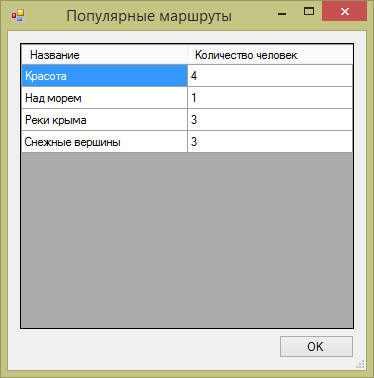


Рисунок 4.12 – Форма популярных маршрутов

Форма со списком походов, которые сейчас идут представлена на рисунке 4.13:

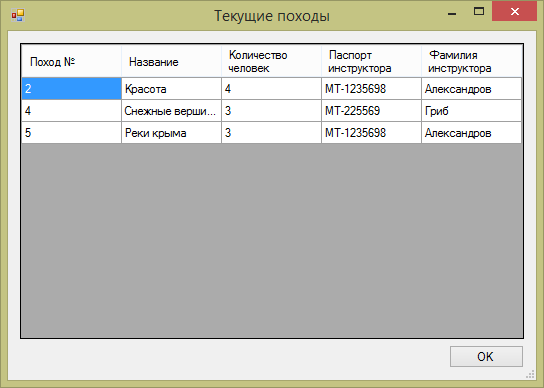


Рисунок 4.12 – Форма походов, которые сейчас идут

Возможность формирования собственного запроса реализована в форме QueryEditorForm, представленной на рисунке 4.13:

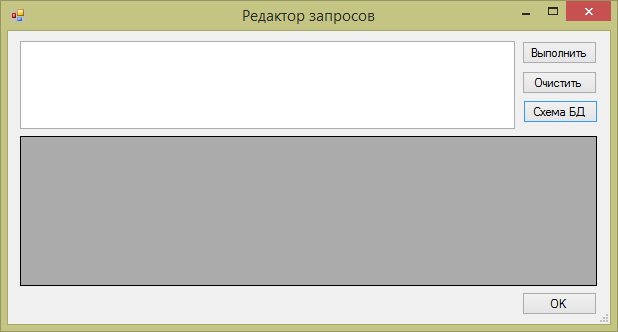


Рисунок 4.12 – Форма редактора собственных запросов

На этой форме можно ввести запрос, выполнить его, получить результат в DataGridView. Также можно просмотреть схему базы данных, нажав на кнопку «Схема БД».

Форма с отчетом о прибыльности фирмы представлена на рисунке 4.13:



Рисунок 4.13 – Форма с отчетом о прибыльности

Форма с заявкой в контрольно-спасательную службу представлена на рисунке 4.14:

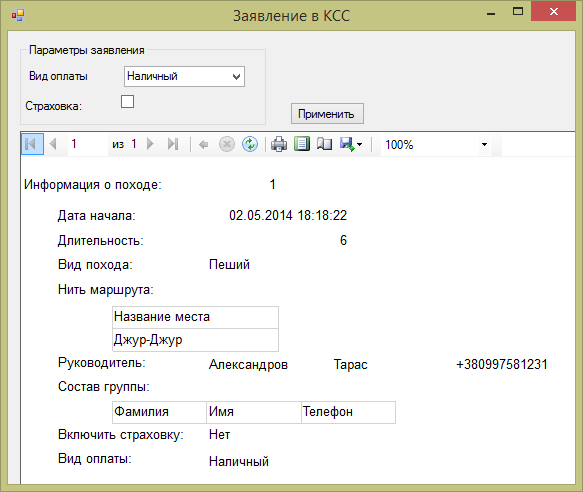


Рисунок 4.14 – Форма с заявкой в контрольно-спасательную службу

На этой форме можно выбрать способ оплаты и необходимость страховки, после этого следует нажать кнопку применить, чтобы изменить заявку.

Также в системе реализована функция добавления клиент и привязки его к походу посредством передачи в программу excel файла с необходимой информацией (см. табл. 2.2). Для того, чтобы воспользоваться этой функцией необходимо открыть форму добавления клиента, которая представлена на рисунке 4.15:

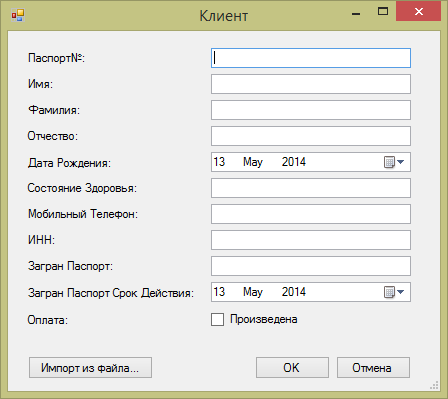


Рисунок 4.15 – Форма добавления клиента

На этой форме необходимо нажать на кнопку импорт из файла. После этого откроется диалоговое окно выбора файлов, где необходимо выбрать excel файл с информацией. Подробнее работа этой функции описана в подразделе 4.6 описание задачи автоматизации.

Далее будут приведены тексты наиболее интересных запросов, которые реализует созданная информационная система.

Запрос, который возвращает всех свободных инструкторов:

SELECT instructor.`ИПаспорт№`, instructor.ИИмя, instructor.ИФамилия, instructor.ИОтчество, instructor.ИИНН, instructor.ИДатаРождения, instructor.`ИЗагранПаспорт№`, instructor.ИЗагранПаспортСрокДействия, instructor.ИЗарплата, instructor.ИМобильныйТелефон

FROM instructor

WHERE (instructor.`ИПаспорт№` NOT IN

(SELECT hike.`ППаспортИнструктора№`

FROM hike, rout

WHERE (hike.`ПМаршрут№` = rout.`ММаршрут№`) AND (adddate(hike.ПДатаНачала, interval rout.МДлительность DAY) > SYSDATE()) AND hike.ПДатаНачала < SYSDATE()))

Запрос, который возвращает наиболее популярные маршруты:

select

`rout`.`МНазвание` AS `МНазвание`,

count(`application`.`ЗНомер№`) AS `count(``ЗНомер№``)`

from

((`application` join `hike`) join `rout`)

where

((`rout`.`ММаршрут№` = `hike`.`ПМаршрут№`)

and (`application`.`ЗПоход№` = `hike`.`ППоход№`))

group by `rout`.`МНазвание`

order by `count(``ЗНомер№``)`

Запрос, который возвращает клиентов, которые сейчас в походе:

SELECT `КПаспорт№`, КИмя, КФамилия, КОтчество, КДатаРождения, КСостояниеЗдоровья, КМобильныйТелефон, КИНН, КЗагранПаспорт, КЗагранПаспортСрокДействия, КОплатаБыла

FROM client

WHERE (`КПаспорт№` IN (SELECT `ЗПаспортКлиента№`FROM application))

Запрос, который отображает прибыльность фирмы:

select

`rout`.`МНазвание` AS `Название`,

`hike`.`ПДатаНачала` AS `ПДатаНачала`,

count(`application`.`ЗНомер№`) AS `Количество человек`,

(count(`application`.`ЗНомер№`) \* `rout`.`МСтоимостьНаЧеловека`) AS `Доход`,

`instructor`.`ИЗарплата` AS `Зарплата инструктора`,

`rout`.`МРазмерДопРасходов` AS `Дополнительные расходы`,

(((count(`application`.`ЗНомер№`) \* `rout`.`МСтоимостьНаЧеловека`) - `instructor`.`ИЗарплата`) - `rout`.`МРазмерДопРасходов`) AS `Прибыль`

from

(((`hike` `application`) `rout`) join `instructor`)

where

((`hike`.`ПМаршрут№` = `rout`.`ММаршрут№`) and (`hike`.`ППоход№` = `application`.`ЗПоход№`) (`hike`.`ППаспортИнструктора№` = `instructor`.`ИПаспорт№`))

group by `hike`.`ППоход№`

Запрос который представляет собой полностью таблицу прибыльности фирмы вместе со строчкой «Итого»:

select

`profits`.`Название` AS `Название`,

`profits`.`ПДатаНачала` AS `ПДатаНачала`,

`profits`.`Количество человек` AS `Количество человек`,

`profits`.`Доход` AS `Доход`,

`profits`.`Зарплата инструктора` AS `Зарплата инструктора`,

`profits`.`Дополнительные расходы` AS `Дополнительные расходы`,

`profits`.`Прибыль` AS `Прибыль`

from `profits`

union

select

'Итого:' AS `Итого:`,

'' AS ``,

count(`profits`.`Количество человек`) AS `Count(``Количество человек``)`,

sum(`profits`.`Доход`) AS `sum(Доход)`,

sum(`profits`.`Зарплата инструктора`) AS `sum(``Зарплата инструктора``)`,

sum(`profits`.`Дополнительные расходы`) AS `sum(``Дополнительные расходы``)`,

sum(`profits`.`Прибыль`) AS `sum(Прибыль)`

from

`profits`

4.6 Описание задачи автоматизации

В качестве задачи автоматизации была реализована функция добавления клиента и привязки его к походу с помощью файла с описанной ранее структурой (см. табл. 2.2). Эта функция выполняется при нажатии на кнопку «Импорт из файла…» на форме «EditClientForm». Весь программный код помещен в обработчике нажатия этой кнопки. Скриншот обработчика приведен на рисунке 4.16.

Как можно заметить сначала программа обращается в созданный класс cExcel [9, 10], к методу ExcelFileToDataTable, который преобразует файл в формат, с которым может работать программа. Он рассматривает файл как базу данных, в которой листы являются таблицами, первая строчка каждой таблицы – название полей таблицы, остальное – данные.

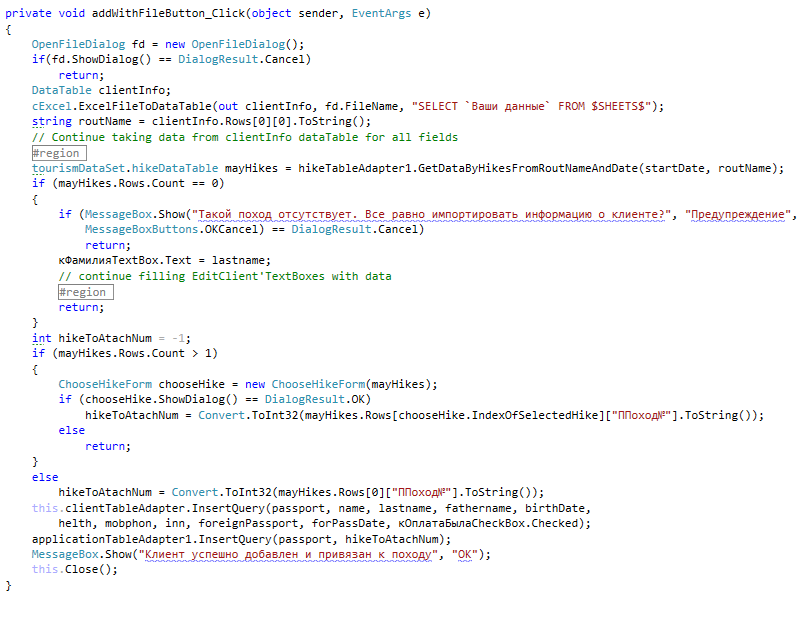


Рисунок 4.15 – Обработчик нажатия кнопки «Импорт из файла…»

После этого из полученного DataTable мы переносим данные в локальные переменные для большего удобства. Далее начинаем анализировать полученные данные. Заполняем mayHikes походами, которые подходят под описание. Если количество таких походов равно нулю, предлагаем пользователю просто импортировать данные про клиента и заполнить ими форму. В случае когда количество превышает 1, пользователю предлагается выбрать один из походов, который подходит, потом обработчик продолжает выполняться. В случае когда подходящий поход всего один программа просто создает нового клиента, создает заявку с соответствующими ссылками на поход и клиента. После сообщения про успешное выполнение операции обработчик завершает свою работу.

ВЫВОДЫ

В данном курсовом проекте была спроектирована и создана база данных и информационная система для удобного взаимодействия пользователей. Также была освоена работа с выбранной СУБД - MySQL.

Был произведен концептуальный анализ предметной области, который использовался далее для создания базы данных и информационной системы.

База данных была разработана для туристических фирм и хранит в себе информацию об инструкторах, клиентах, походах и маршрутах, местах, через которые проходят маршруты. В ней реализованы связи один ко многим и многие ко многим через вспомогательные сущности. К примеру, связь между клиентом и походом осуществляется через сущность «заявка», которая хранит в себе внешние ключи на записи из таблиц «Клиент» и «Поход».

Во время проектирования базы данных были также изучены первые три нормальные формы и применены на практике. База данных приведена к третьей нормальной форме, что является компромиссом между быстродействием и качеством структуры базы данных.

Во время выбора СУБД были изучены основные преимущества самых распространённых СУБД, таких как Oracle, MySQL, Microsoft SQL Server. Была выбрана вторая.

Во время создания программы было изучено взаимодействие приложения на технологии .NET и СУБД MySQL. Изучены такие элементы, как TableAdapter, DataSet и другие, которые помогали использовать информацию из базы данных, изменять ее и отображать на экране.

Были также приобретены практические навыки использования SQL при доступе к базе данных. Написаны структурно сложные запросы, которые получают структурированную информацию из базы данных.

Были выполнены все требования, которые были перечислены в разделе постановки задачи.

ПЕРЕЧЕНЬ ССЫЛОК

1. http://www.outdoorukraine.com/
2. Эндрю Троельсен. C# и платформа .NET. Библиотека программиста. – СПб.: Питер, 2005. – 796 с.
3. Дейт К.Дж. Введение в системы баз данных [Текст] / К.Дж. Дейт. - 8-е изд. – М.:Издательский дом „Вильямс”, 2005.- 1328 с.
4. Грабер М. SQL. [Текст] / М.Грабер.- К.: Изд-во “ЛОРИ”,2003.–644 с.
5. Гарсия-Молина Г. Системы баз данных. Полный курс. [Текст]: Пер. с англ./ Г.Гарсия-Молина, Дж.Ульман, Дж.Уидом. – М.: Издательский дом „Вильямс”, 2003. – 1088 с.
6. Кренке Д. Теория и практика построения баз данных. [Текст] / Д.Кренке - 8-е изд. – СПб.: Питер, 2003. – 800 с.
7. <http://strong-planet.ru/database.html>
8. Герберт Шилдт "C# 4.0. Полное руководство".
9. <http://msdn.microsoft.com>
10. http://stackoverflow.com/