**Содержание**

[**Введение** 5](#_Toc483572218)

[**1 Постановка задачи** 6](#_Toc483572219)

[**2 Теоретические основы проектирования баз данных** 8](#_Toc483572220)

[**3 Технология разработки приложения** 13](#_Toc483572221)

[**Заключение** 14](#_Toc483572222)

[**Список литературы** 15](#_Toc483572223)

[ПРИЛОЖЕНИЕ 1 16](#_Toc483572224)

[ПРИЛОЖЕНИЕ 2 21](#_Toc483572225)

[ПРИЛОЖЕНИЕ 3 22](#_Toc483572226)

**Введение**

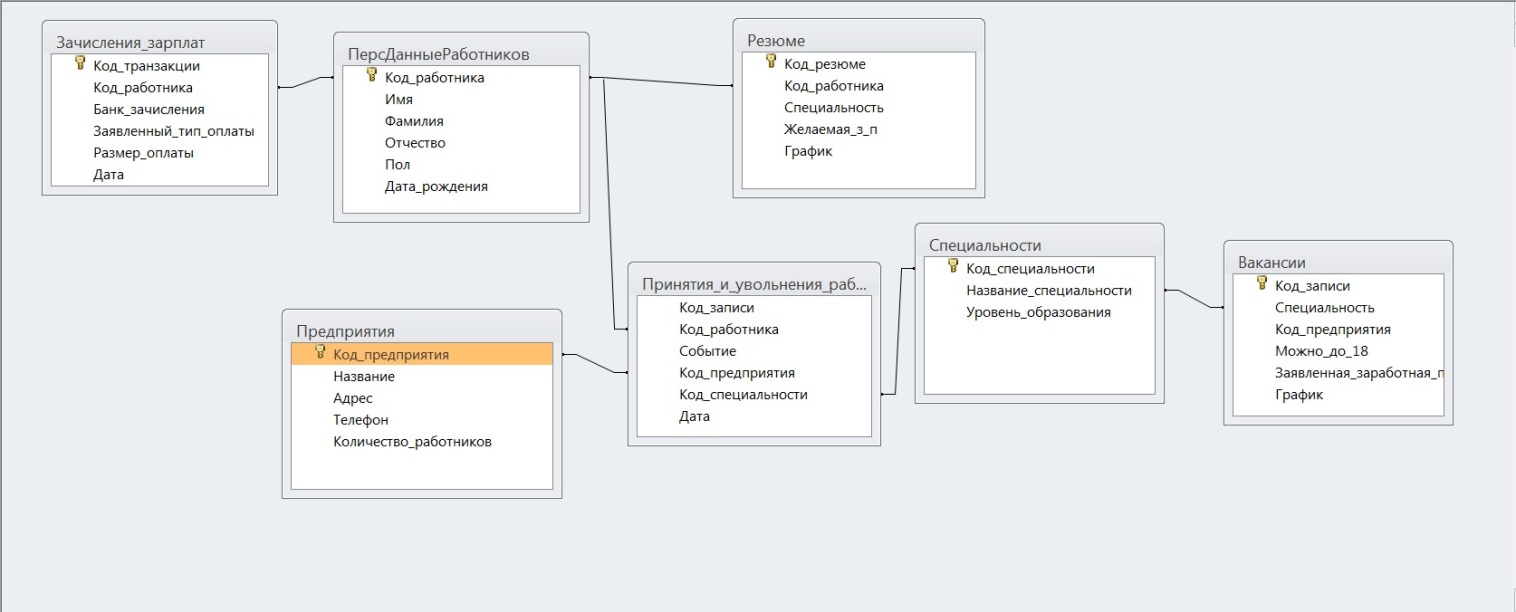
В современном мире базы данных задействованы повсюду: от маленьких магазинов и складов, до банковских счетов и учета в государственных органах. Любому программисту необходимо иметь навыки работы с базами данных, в частности уметь администрировать и поддерживать их в надлежащем состоянии. В процессе разработки данного проекта задействовано множество решений, связанных с созданием и администрированием баз данных, а так же наглядно показано применение всех необходимых программисту навыков для создания и администрирования баз данных. Данное приложение позволяет пользователю работать с таблицами базы данных “Кадры предприятия”, и имеет, множество разнообразных видов запросов, позволяет пользователю вносить изменения в базу данных в соответствии со своими задачами, не манипулируя данными таблиц напрямую, и тем самым значительно облегчая работу. Программа поддерживает режим пользователя, позволяющий взаимодействовать с данными таблиц исключительно непрямым путем, с помощью специализированных запросов, и режим администратора – дающий все возможности для манипулирования данными базы данных. Программа создана на языке программирования C#, в среде разработки Visual studio 2015.

**1 Постановка задачи**

Цель работы: Создать приложение для работы с базой данных “Кадры предприятия”

Задачи:

* Изучить структуру исходной базы данных, и при необходимости внести изменения в ее структуру
* Создать в среде Visual studio формы, отражающие структуру приложения
* Изучить эргономические требования к интерфейсу
* Разработать адаптированный для пользователей интерфейс

Входные данные программы: исходный файл базы данных, содержащий необходимые табличные данные.

Выходная информация программы: файл приложения для работы с базой данных “Кадры предприятия”.

Требования к аппаратному обеспечению:

- Процессор с тактовой частотой не менее 1ггц.

- Оперативная память объемом не менее 256 мб.

- Видеокарта с памятью объемом не менее 64 мб.

- Свободное место на жестком диске размером не менее 500 мб.

- Периферийные устройства: клавиатура, мышь.

Требования к программному обеспечению:

- Операционная система Windows версии 7 или выше.

- Наличие установленного Microsoft .Net framework версии не ниже 4.5

**2 Теоретические основы проектирования баз данных**

**База данных** – это организованная структура, предназначенная для хранения информации. В современных базах данных хранятся не только данные, но и информация.

Это определение легко пояснить, если, например, рассмотреть базу данных крупного банка. В ней есть все необходимые сведения о клиентах, их адреса, кредитные истории, состояние расчетных счетов, финансовые операции и т.д. Доступ к этой базе данных имеется у достаточно большого количества сотрудников банка, но среди них вряд ли найдется такое лицо, которое имеет доступ ко всей базе полностью и при этом способно единолично вносить в нее произвольные изменения. Кроме данных, база содержит методы и средства, позволяющие каждому из сотрудников оперировать только с теми данными, которые входят в его компетенцию. В результате взаимодействия данных, содержащихся в базе, с методами, доступными конкретным сотрудникам, образуется информация, которую они потребляют и на основании которой в пределах собственной компетенции производят ввод и редактирование данных.

С понятием базы данных тесно связано понятие системы управления базой данных. Это комплекс программных средств, предназначенных для создания структуры новой базы, наполнение ее содержимым, редактирования содержимого и визуализации информации. Под визуализацией информации базы понимается отбор отображаемых данных в соответствии с заданным критерием, их упорядочение, оформление и последующая выдача на устройства вывода или передачи по каналам связи.

В мире существует множество систем управления базами данных. Несмотря на то, что они могут по-разному работать с разными объектами и предоставляют пользователю различные функции и средства, большинство СУБД опираются на единый устоявшийся комплекс основных понятий. Это дает возможность рассмотреть одну систему и обобщить ее понятия, приемы и методы на весь класс СУБД. В качестве такого учебного объекта можно взять СУБД Microsoft Access, входящую в пакет Microsoft Office.

**Состав СУБД**

Язык описания данных (ЯОД) – средства описания данных в БД и связей между ними. Средствами этого языка описывается структура БД, форматы записей, пароли, защищающие данные.

Язык манипулирования данными (ЯМД) – язык для выполнения операций над данными, позволяющий менять их строение.

Для различных СУБД реализация этих уровней языков может быть различной. В одних случаях ЯОД и ЯМД требует составления пользователем программы полностью “вручную”, в других (что отражает современную тенденцию) в СУБД присутствуют средства визуальной (зримой, наглядной) разработки программ. Для этого в современных СУБД имеются редакторы экранных форм, отчетов. “Кирпичиками” (инструментами) таких редакторов являются поля различных видов (поля ввода, поля вывода, вычисляемые поля), процедуры обработки различных типов (формы ввода, таблицы, отчеты, запросы). На основании созданных пользователем объектов программы – генераторы формируют программный код на языке конкретной машины или на промежуточном языке.

**Структура простейшей базы данных**

Если даже в базе нет никаких данных (пустая база), то это все равно полноценная база данных. Этот факт имеет методическое значение. Хотя данных в базе и нет, но информация в ней все-таки есть – это структура базы, которая определяет методы занесения данных и хранения их в базе. Простейший «некомпьютерный» вариант базы данных – деловой ежедневник, в котором каждому календарному дню выделено по странице. Даже если в нем не записано ни строки, он не перестает быть ежедневником, поскольку имеет структуру, четко отличающую его от записных книжек, рабочих тетрадей и прочей писчебумажной продукции.

Базы данных могут содержать различные объекты. Основными объектами любой базы данных являются ее таблицы. Простейшая база данных имеет хотя бы одну таблицу. Соответственно, структура простейшей базы данных тождественно равна структуре ее таблицы.

Структуру двумерной таблицы образуют столбцы и строки. Их аналогами в простейшей базе данных являются поля и записи. Если записей в таблице пока нет, значит, ее структура образована только набором полей. Изменив состав полей базовой таблицы (или их свойства), мы изменяем структуру базы данных и, соответственно, получаем новую базу данных.

**Свойства полей базы данных**

Поля базы данных не просто определяют структуру базы – они еще определяют групповые свойства данных, записываемых в ячейки, принадлежащие каждому из полей. Ниже перечислены основные свойства полей таблиц баз данных на примере СУБД Microsoft Access.

Имя поля – определяет, как следует обращаться к данным этого поля при автоматических операциях с базой (по умолчанию имена полей используются в качестве заголовков столбцов таблиц).

Тип поля – определяет тип данных, которые могут содержаться в данном поле.

Размер поля – определяет предельную длину (в символах) данных, которые могут размещаться в данном поле.

Формат поля – определяет способ форматирования данных в ячейках, принадлежащих полю.

Маска ввода – определяет форму, в которой вводятся данные в поле (средство автоматизации ввода данных).

Подпись – определяет заголовок столбца таблицы для данного поля (если подпись не указана, то в качестве заголовка столбца используется свойство Имя поля).

Значение по умолчанию – то значение, которое вводится в ячейки поля автоматически (средство автоматизации ввода данных).

Условие на значение – ограничение, используемое для проверки правильности ввода данных (средство автоматизации ввода, которое используется, как правило, для данных, имеющих числовой тип, денежный тип или тип даты).

Сообщение об ошибке – текстовое сообщение, которое выдается автоматически при попытке ввода в поле ошибочных данных.

Обязательное поле – свойство, определяющее обязательность заполнения данного поля при наполнении базы.

Пустые строки – свойство, разрешающее ввод пустых строковых данных (от свойства Обязательное поле отличается тем, что относится не ко всем типам данных, а лишь к некоторым, например к текстовым).

Индексированное поле – если поле обладает этим свойством, то все операции, связанные с поиском или сортировкой записей по значению, хранящемуся в данном поле, существенно ускоряются. Кроме того, для индексированных полей можно сделать так, что значение в записях будут проверяться по этому полю на наличие повторов, что позволяет автоматически исключить дублирование данных.

Поскольку в разных полях могут содержаться данные разного типа, то и свойства у полей могут различаться в зависимости от типа данных. Так, например, список вышеуказанных свойств полей относится в основном к полям текстового типа. Поля других типов могут иметь или не иметь эти свойства, но могут добавлять к ним и свои. Например, для данных, представляющих действительные числа, важным свойством является количество знаков после десятичной запятой. С другой стороны, для полей, используемых для хранения рисунков, звукозаписей, видео клипов и других объектов OLE, большинство вышеуказанных свойств не имеют смысла.

**Типы данных**

Таблицы баз данных, как правило, допускают работу с гораздо большим количеством разных типов данных. Так, например, базы данных Microsoft Access работают со следующими типами данных.

Текстовый – тип данных, используемый для хранения обычного неформатированного текста ограниченного размера (до 255 символов).

Числовой – тип данных для хранения действительных чисел.

Поле Мемо – специальный тип данных для хранения больших объемов текста (до 65 535 символов). Физически текст не хранится в поле. Он храниться в другом месте базы данных, а в поле храниться указатель на него, но для пользователя такое разделение заметно не всегда.

Дата/время – тип данных для хранения календарных дат и текущего времени.

Денежный - тип данных для хранения денежных сумм. Теоретически, для их записи можно было бы пользоваться и полями числового типа, но для денежных сумм есть некоторые особенности (например, связанные с правилами округления), которые делают более удобным использование специального типа данных, а не настройку числового типа.

Счетчик – специальный тип данных для уникальных (не повторяющихся в поле) натуральных чисел с автоматическим наращиванием. Естественное использование служит для порядковой нумерации записей.

Логический - тип для хранения логических данных (могут принимать только два значения, например Да или Нет).

Гиперссылка – специальное поле для хранения адресов URL Web-объектов Интернета. При щелчке на ссылке автоматически происходит запуск браузера и воспроизведение объекта в его окне.

Мастер подстановок – это не специальный тип данных. Это объект, настройкой которого можно автоматизировать ввод данных в поле так, чтобы не вводить их вручную, а выбирать их из раскрывающегося списка /1/.

**Безопасность баз данных**

Базы данных – это тоже файлы, но работа с ними отличается от работы с файлами других типов, создаваемых прочими приложениями. Выше мы видели, что всю работу по обслуживанию файловой структуры берет на себя операционная система. Для базы данных предъявляются особые требования с точки зрения безопасности, поэтому в них реализован другой подход к сохранению данных.

Базы данных – это особые структуры. Информация, которая в них содержится, очень часто имеет общественную ценность. Нередко с одной и той же базой работают тысячи людей по всей стране. От информации, которая содержится в некоторых базах, может зависеть благополучие множества людей. Поэтому целостность содержимого базы не может и не должна зависеть ни от конкретных действий некоего пользователя, забывшего сохранить файлы перед выключением компьютера, ни от перебоев в электросети.

Проблема безопасности баз данных решается путем того, что в СУБД  для сохранения информации используется двойной подход. В части операций, как обычно, участвует операционная система компьютера, но некоторые операции сохранения происходят в обход операционной системы.

**Режимы работы с базами данных**

Обычно с базами данных работают две категории пользователей. Первая категория – проектировщики. Их задача состоит в разработке структуры таблиц базы данных и согласование ее с заказчиком. Кроме таблиц проектировщики разрабатывают и другие объекты базы данных, предназначенные, с одной стороны, для автоматизации работы с базой, а с другой стороны – для ограничения функциональных возможностей работы с базой (если это необходимо из соображений безопасности). Проектировщики не наполняют базу конкретными данными, (заказчик может считать их конфиденциальными и не предоставлять посторонним лицам). Исключение составляет экспериментальное наполнение модельными данными на этапе отладки объектов базы /2/.

Вторая категория исполнителей, работающих с базами данных, - пользователи. Они получают исходную базу данных от проектировщиков и занимаются ее наполнением и обслуживанием. В общем случае пользователи не имеют средств доступа к управлению структурой базы – только к данным, да и то не ко всем, а к тем, работа с которыми предусмотрена на конкретном рабочем месте.

Соответственно СУБД имеет два режима работы: проектировочный  и пользовательский. Первый режим предназначен для создания или изменения структуры базы и создание ее объектов. Во втором режиме происходит использование ранее подготовленных объектов для наполнения базы или получения данных из нее.

**3 Технология разработки приложения**

Курсовая работа содержит множество программных решений, реализованных в языке программирования C#. Данный язык был выбран как оптимальный вариант для быстрого проектирования легких приложений для Windows в силу его общедоступности, “дружелюбного” синтаксиса, множества библиотек, позволяющих создавать проекты под практически любые задачи. В силу схожести синтаксиса данного языка с языками серии “Си” и Java, знание данного языка в дальнейшем позволит с легкостью при необходимости перейти на программирование на данных платформах.

Подключение к базе данных, находящейся в файле проекта Access, реализуется с помощью стандартной библиотеки “System.Data.OleDb”.

Все обращения к базе данных так или иначе используют данную библиотеку. В частности, получение таблицы из SQL-запроса производится следующим образом:

В следующей строке создается экземпляр класса, содержащего в себе данные, позволяющие сформировать запрос.

OleDbCommand command1 = new OleDbCommand();

Далее задается текст запроса.

command1.Commandtext = “Ваш\_SQL\_запрос”;

Далее создается новый экземпляр класса, реализующий всю работу с базой данных.

OleDbDataAdapter ODDA = new OldDbDataAdapter(command1);

Далее создается пустая таблица, в которую будут помещены данные из таблицы, и метод “Fill” экземпляра класса “ OleDbDataAdapter ” заполняет пустую таблицу данными, возвращенными SQL-запросом.

DataTable table1 = new DataTable();

ODDA.Fill(Table1);

Все формы в приложении создаются при помощи общедоступной библиотеки C# Windows Forms. Данная библиотека позволяет создавать формы любой сложности, и без особых усилий связывать события элементов форм с соответствующими методами.

**Заключение**

Разработанное в данной курсовой работе приложение является полностью законченным программным продуктом, и может быть использовано как самостоятельно, так и как компонент другой информационной системы.

Поставленная в курсовой работе цель достигнута. В ходе выполнения работы были изучены приемы программной работы с языком программирования C# и языком структурированных запросов SQL. Освоена технология создания таблиц исходной базы данных в СУБД Microsoft Access.

В процессе выполнения курсовой работы было освоено применение функциональной декомпозиции, методов, классов, библиотек, и иных программных конструкций в объектно-ориентированном программировании. Изучены особенности различных сред разработки программного обеспечения. Приобретены навыки работы с программными продуктами, предназначенными для систематической обработки данных.

Освоены следующие профессиональные компетенции:

|  |  |
| --- | --- |
| ПК 1 | Разрабатывать объекты базы данных. |
| ПК 2 | Реализовывать базу данных в конкретной СУБД. |
| ПК 3 | Решать вопросы администрирования базы данных. |
| ПК 4 | Реализовывать методы и технологии защиты информации  в базах данных. |

**Список литературы**

1. Самоучитель Access 2010 Бекаревич Ю.Б., 2013 г.
2. Учебник “.NET Программирование для профессионалов” Эндрю Кровчик, Винод Кумар, 2015г.
3. Форум для начинающих программистов http://www.cyberforum.ru/

ПРИЛОЖЕНИЕ 1

Код класса, созданного мною для упрощения работы с базой данных, в частности: составления SQL-запроса по входным параметрам, реализация подключения к базе данных, а так же упрощенная работа с запросами.

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

using System.Windows.Forms;

using System.Security.Cryptography;

using System.Data.OleDb;

using System.IO;

namespace Kursovaya\_Voroshilov\_A\_I

{

class Worker4Admin

{

public bool AsAdmin = false;

OleDbConnection Coursach\_Connection = new OleDbConnection();

internal Worker4Admin()

{

try

{

Coursach\_Connection.ConnectionString = @"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" + Directory.GetCurrentDirectory() + @"\Copy\_of\_cursach.mdb" + ";Persist Security Info=False;";

Coursach\_Connection.Open();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

internal Worker4Admin(bool AsAdmin, string path)

{

try

{

Coursach\_Connection.ConnectionString = @"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" + path + ";Persist Security Info=False;";

Coursach\_Connection.Open();

}

catch (Exception ex)

{

throw new Exception();

}

this.AsAdmin = AsAdmin;

}

internal Worker4Admin(bool AsAdmin, string path, string password)

{

try

{

Coursach\_Connection.ConnectionString = @"Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" + path + ";Jet OLEDB:Database Password=" + password +";";

Coursach\_Connection.Open();

}

catch (Exception ex)

{

throw new Exception();

}

this.AsAdmin = AsAdmin;

}

public string ReturnChastZaprosaFromArguments(string StolbecName, string type /\*value|region|perechisleniye\*/, string Logic /\*=|<>|~\*/, string StolbecIntOrString, bool SetAnd, string[] Values, string[] between)

{

string Zapros = " ";

if (SetAnd)

{

Zapros = " and ";

}

switch (type)

{

case ("value"):

{

#region ValueCode

if (StolbecIntOrString == "string")

{

Zapros = Zapros + StolbecName + Logic + "'" + Values[0] + "'";

}

else if (StolbecIntOrString == "int")

{

Zapros = Zapros + StolbecName + Logic + Values[0];

}

else if (StolbecIntOrString == "date")

{

Values[0] = Values[0].Replace('.', '-');

Zapros = Zapros + StolbecName + Logic + "#" + Values[0] + "#";

}

break;

#endregion ValueCode

}

case ("region"):

{

#region RegionCode

if (StolbecIntOrString == "date")

{

if (Logic == "=")

{

between[0] = "#" + between[0].Replace('.', '-') + "#";

between[1] = "#" + between[1].Replace('.', '-') + "#";

Zapros = Zapros + StolbecName + " BETWEEN " + between[0] + " AND " + between[1];

}

else if (Logic == "<>")

{

between[0] = "#" + between[0].Replace('.', '-') + "#";

between[1] = "#" + between[1].Replace('.', '-') + "#";

Zapros = Zapros + StolbecName + " NOT BETWEEN " + between[0] + " AND " + between[1];

}

}

else if (StolbecIntOrString == "int")

{

if (Logic == "=")

{

Zapros = Zapros + StolbecName + " BETWEEN " + between[0] + " AND " + between[1];

}

else if (Logic == "<>")

{

Zapros = Zapros + StolbecName + " NOT BETWEEN " + between[0] + " AND " + between[1];

}

}

#endregion RegionCode

break;

}

case ("perechisleniye"):

{

#region PerechisleniyeCode

if (StolbecIntOrString == "string")

{

if (Logic == "=")

{

Zapros = Zapros + StolbecName + " in (";

}

else if (Logic == "<>")

{

Zapros = Zapros + StolbecName + " not in (";

}

bool UjeWas = false;

foreach (string s in Values)

{

if (!UjeWas)

{

Zapros = Zapros + "'" + s + "'";

UjeWas = true;

}

else

{

Zapros = Zapros + ", " + "'" + s + "'";

}

}

Zapros = Zapros + ")";

}

else if (StolbecIntOrString == "int")

{

if (Logic == "=")

{

Zapros = Zapros + StolbecName + " in (";

}

else if (Logic == "<>")

{

Zapros = Zapros + StolbecName + " not in (";

}

bool UjeWas = false;

foreach (string s in Values)

{

if (!UjeWas)

{

Zapros = Zapros + s;

UjeWas = true;

}

else

{

Zapros = Zapros + ", " + s;

}

}

Zapros = Zapros + ")";

}

else if (StolbecIntOrString == "date")

{

if (Logic == "=")

{

Zapros = Zapros + StolbecName + " in (";

}

else if (Logic == "<>")

{

Zapros = Zapros + StolbecName + " not in (";

}

bool UjeWas = false;

foreach (string s in Values)

{

if (!UjeWas)

{

Zapros = Zapros + "#" + s.Replace('.', '-') + "#";

UjeWas = true;

}

else

{

Zapros = Zapros + ", " + "#" + s.Replace('.', '-') + "#";

}

}

Zapros = Zapros + ")";

}

break;

#endregion PerechisleniyeCode

}

}

return Zapros;

}

public DataTable GetFullTableFromZapros(string TableName)

{

OleDbCommand TekCommand = new OleDbCommand();

TekCommand.CommandText = "select \* from " + TableName;

TekCommand.Connection = Coursach\_Connection;

OleDbDataAdapter DatAdapt = new OleDbDataAdapter(TekCommand);

DataTable TekTable = new DataTable();

DatAdapt.Fill(TekTable);

return TekTable;

}

public DataTable GetTableFromZapros(string Zapros)

{

OleDbCommand TekCommand = new OleDbCommand();

TekCommand.CommandText = Zapros;

TekCommand.Connection = Coursach\_Connection;

OleDbDataAdapter DatAdapt = new OleDbDataAdapter(TekCommand);

DataTable TekTable = new DataTable();

DatAdapt.Fill(TekTable);

return TekTable;

}

public int SetZapros(string Zapros)

{

OleDbCommand TekCommand = new OleDbCommand();

TekCommand.CommandText = Zapros;

TekCommand.Connection = Coursach\_Connection;

return TekCommand.ExecuteNonQuery();

}

public OleDbConnection TekOleDbConnection

{

get

{

return Coursach\_Connection;

}

set

{

Coursach\_Connection = value;

}

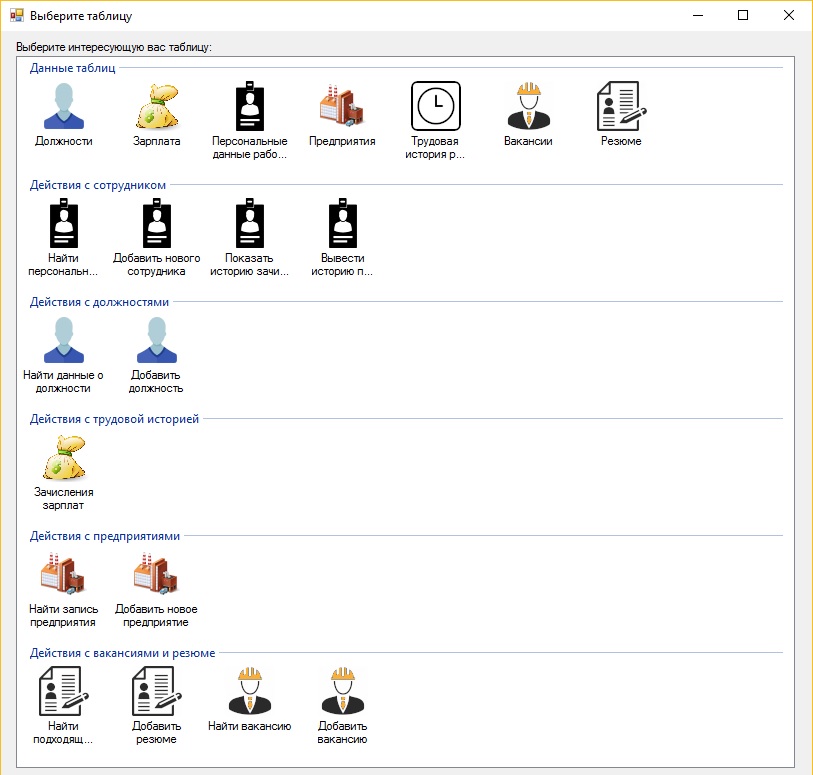
}

}

}

ПРИЛОЖЕНИЕ 2

Форма главного меню приложения



ПРИЛОЖЕНИЕ 3

Форма, дающая доступ а данным таблицы “Персональные данные работников”

