ЛАБОРАТОРНАЯ РАБОТА №1. РАЗРАБОТКА ПРИЛОЖЕНИЙ ДЛЯ БАЗ ДАННЫХ MS ACCESS С ПОМОЩЬЮ ТЕХНОЛОГИИ ADO.NET (MS VISUAL STUDIO C#).

Цель: изучить базовые понятия технологии ADO.NET, принципы разработки приложения для базы данных MS Access, получить практические навыки разработки приложения к базе данных на языке C# в MS Visual Studio

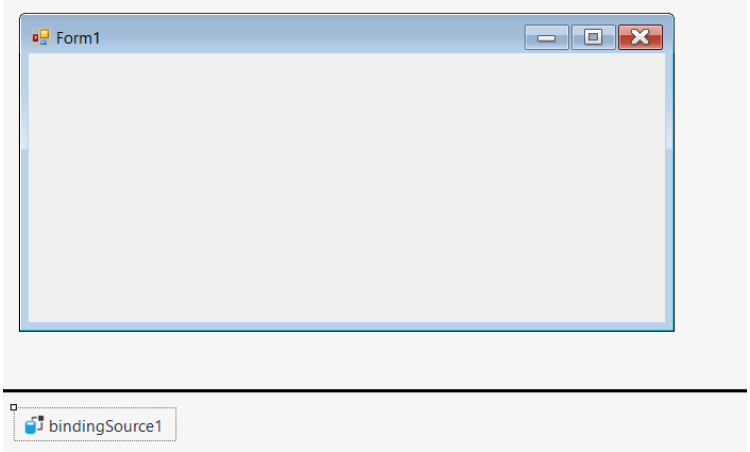
Содержание лабораторной работы:

1. Изучить теоретические сведения лабораторной работы.
2. Создать приложение Application Windows Forms C# (проект C#).
3. Соединить базу данных MS Access (макет по дисциплине «Модели данных») с созданной формой.
4. Спроектировать интерфейс приложения, содержащий: несколько взаимосвязанных форм с данными из базы данных с возможностью чтения, добавления, обновления, удаления данных, кнопку для перехода к другим формам, компоненты DataGridView, BindingNavigator, TextBox и другие компоненты (см. пункт 5).
5. Требования к интерфейсу:
    a. Форма 1, содержащая:
        i. Две связанные между собой таблицы из базы данных;
        ii. Подписи названий таблиц;
        iii. Панель навигации;
        iv. Компоненты типа Textbox для изменения данных;
        v. Кнопка для обновления информации в таблицах, кнопка сохранения;
        vi. Кнопка перехода к другой форме (форма 2, форма 3).
    b. Форма 2, содержащая любую информацию из базы данных (не использовать DataGridView).
    c. Форма 3, содержащая вывод результата запроса в DataGridView по кнопке.
6. Оформить и защитить отчет по лабораторной работе.
    a. Требования к отчету:
        i. Титульный лист;
        ii. Цель лабораторной работы;
        iii. Описание шагов выполнения лабораторной работы с принтскринами;
        iv. Выводы по лабораторной работе.
    b. Защита лабораторной работы (только при наличии печатного отчета):
        i. Любой вопрос по выполнению лабораторной работы;
        ii. Любой вопрос по отчету;
        iii. Любой вопрос из контрольных вопросов (стр.9).

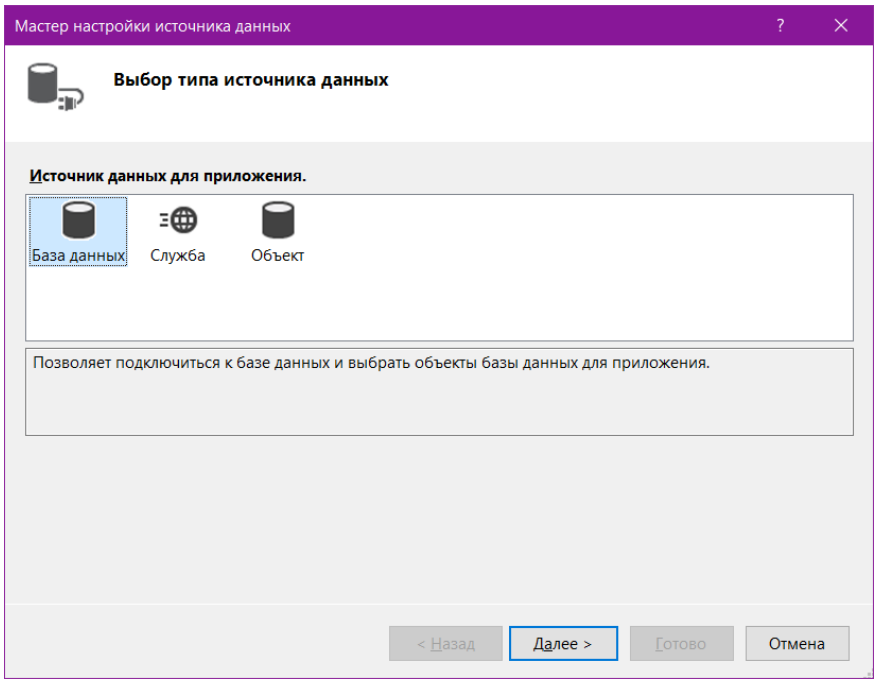
Краткий вспомогательный материал

Создадим простое приложение баз данных, которое выводит на экранную форму информацию из таблицы «Отдел» и связанную с текущей записью таблицы «Отдел» запись таблицы «Работники» из базы данных MS Access.

Привязку данных БД к форме осуществляет компонент «Binding Source». Перенесем его на форму. Компонент является не визуальным, поэтому он отображается на дополнительной панели. Основным свойством компонента является свойство DataSource, указывающее на источник данных. По умолчанию свойство является пустым, поэтому необходимо сформировать его значение.



В настоящий момент список пуст, поэтому необходимо создать новый источник данных, выбрав команду «Add Project Data Source» для создания нового источника данных и соединения с ним. Появляется следующее окно диалога:

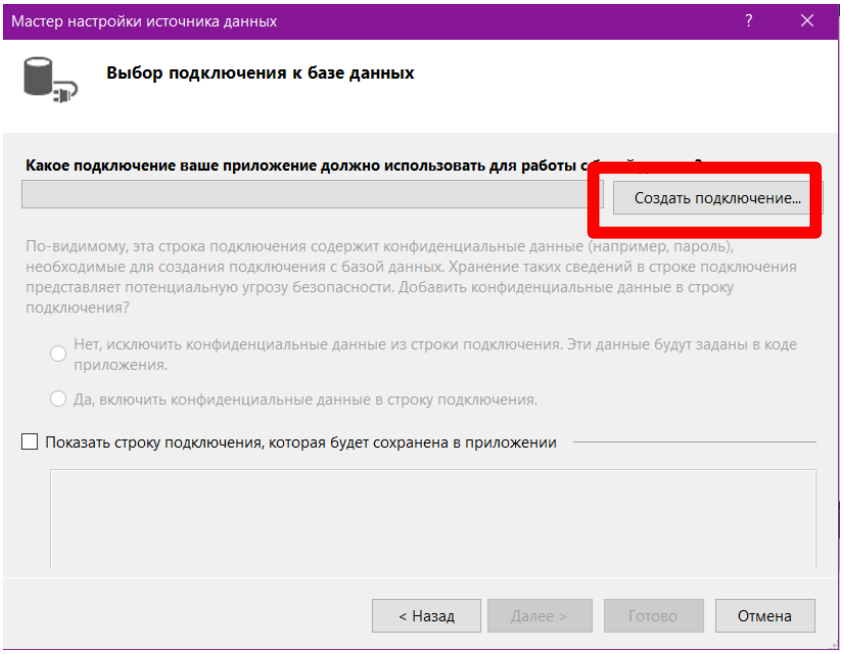


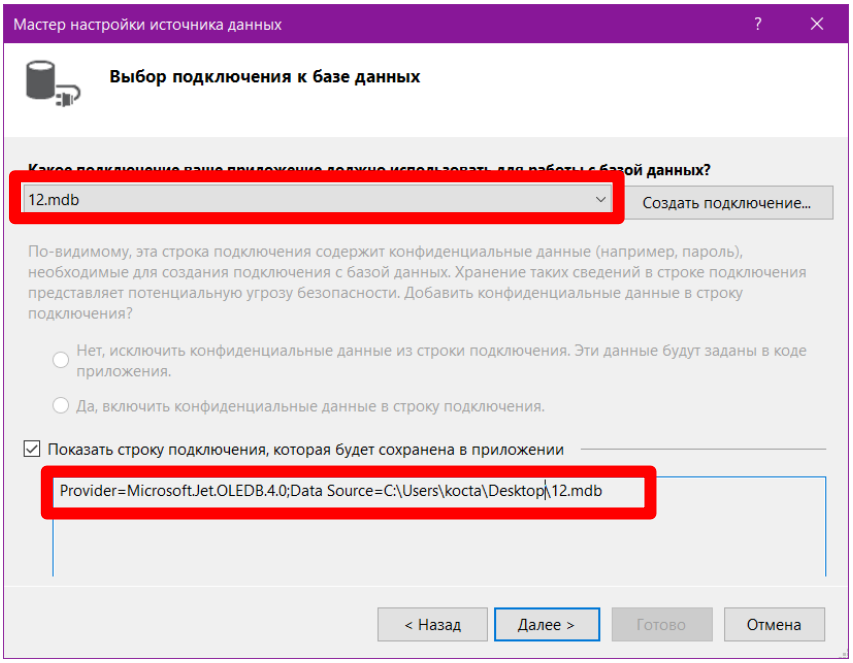
Данный диалог предоставляет следующий выбор источников данных:

- Database – База данных;
- Service – Служба, это некоторый сервис, предоставляющий данные. Чаще всего это Web-сервис;

- Object – Объект для выбора объекта, который будет генерировать данные и объекты для работы с ними.

В нашем случае необходимо выбрать пункт «База данных» («Database»). Появляется окно выбора соединения с данными.



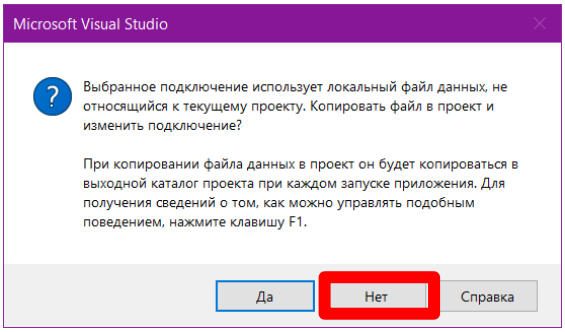


Целью данного диалога является создание строки соединения, в которой будут описаны параметры соединения для механизма ADO, такие как тип базы данных, ее местонахождение, имена пользователей, средства безопасности и пр.

В выпадающем списке диалога находятся все создаваемые ранее соединения. Если необходимого соединения в списке нет, то следует использовать кнопку «Создать подключение» («New connection»). В следующем диалоге выбирается тип источника данных (в данном случае Microsoft Access), имя базы данных (в данном случае имя и местоположение файла базы данных), имя пользователя и пароль, используемые для

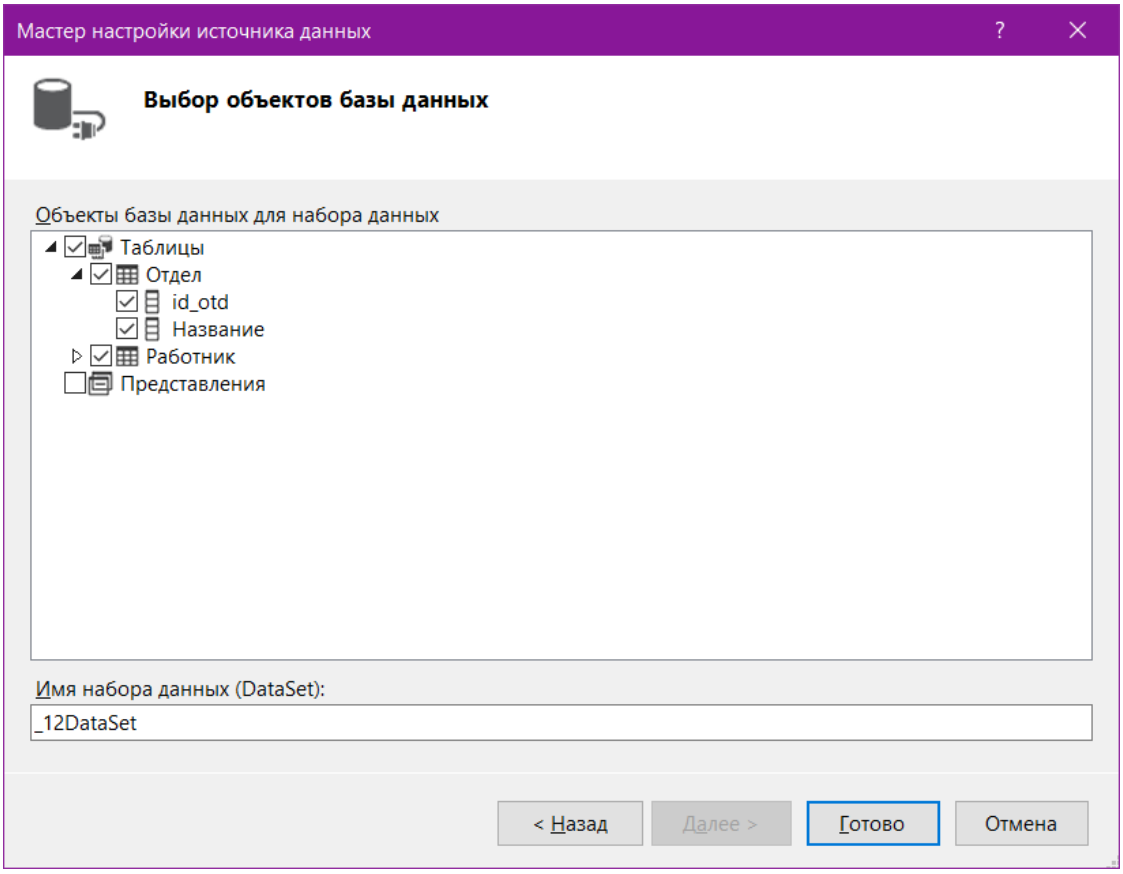
подключения к базе данных. Кнопка «Дополнительно» («Advanced») позволяет задать большое количество параметров, относящихся к различным деталям механизма ADO. Использование кнопки «Проверить подключение» («Test Connection») позволит убедиться в правильности введенных параметров и работоспособности соединения.

В данном моменте нажать кнопку **НЕТ.**



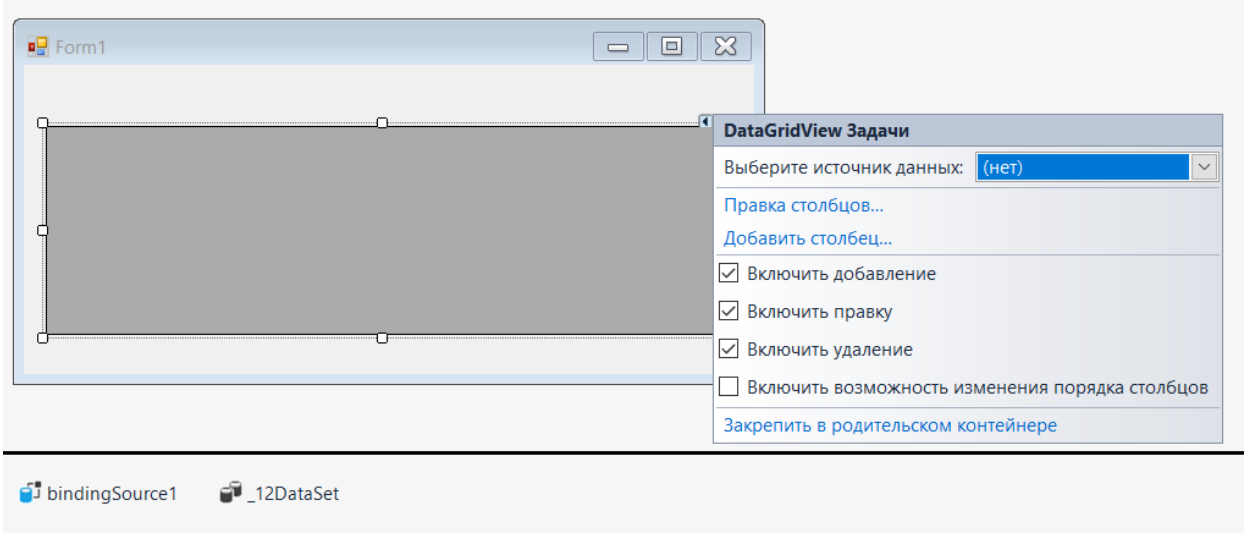
Следующий шаг диалога предлагает сохранить полученную строку соединения в файле настроек приложения. Рекомендуется принять данный выбор для упрощения последующего размещения и поддержки программного продукта.

Последний шаг диалога – выбор тех таблиц или иных объектов базы данных, которые необходимы в данном источнике данных.



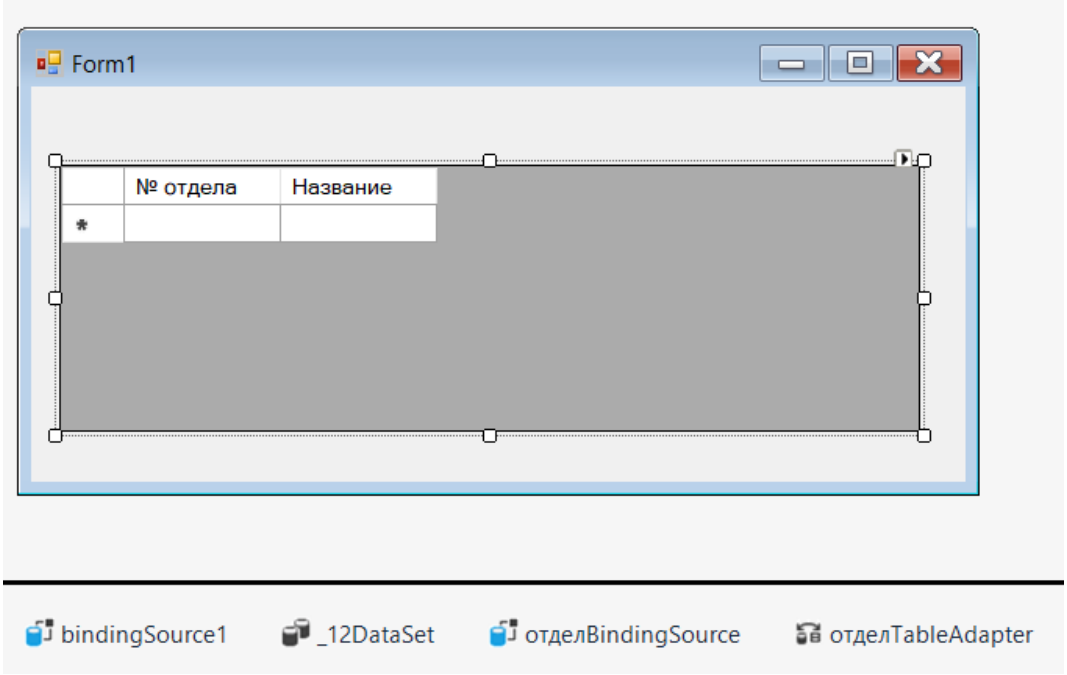
В данном окне выбраны таблицы «Отдел» и «Работник». На этом создание источника данных завершено. После нажатия кнопки «Готово» («Finish») рядом с компонентом BindingSource на форме появляется компонент DataSet.

Теперь данные, подключенные выше, необходимо отобразить на форме. Простейшим способом отображения данных является использование компонента DataGridView из группы компонентов Data.



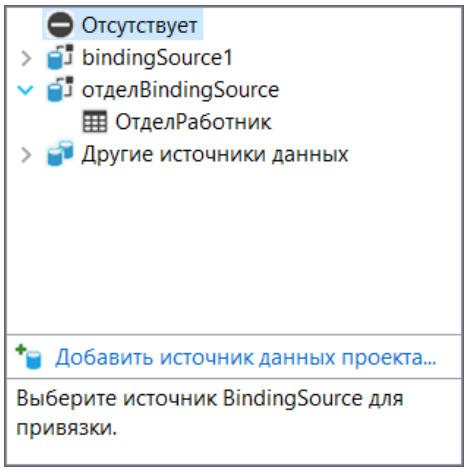
Сразу же возникает окно настройки компонента, которое определяет его возможности по редактированию данных: «Включить редактирование» («EnableAdding»), «Включить правку» («Enable Editing»), «Включить удаление» («Enable Deleting»); возможность изменения последовательности столбцов: «Включить возможность изменения порядка столбцов» («Enable Column Reordering»); а также возможность закрепления в контейнере-родителе.

Для того чтобы компонент мог отображать данные, необходимо выбрать источник данных в выпадающем списке. В данном случае мы выбрали в качестве источника данных таблицу «Отдел». Данный выбор изменяет экранную форму следующим образом



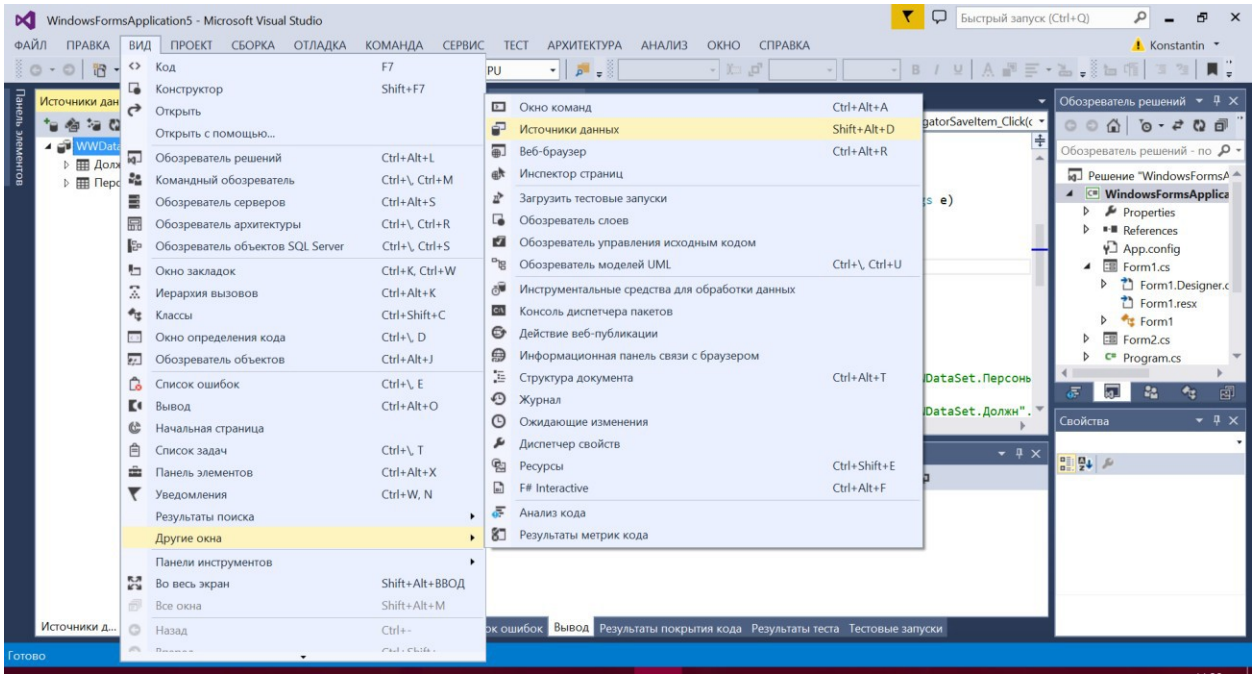
На рисунке видно, что появился еще один компонент BindingSource и компонент TableAdapter, работающий с таблицей «Отдел». Обратите внимание, что в design-time или

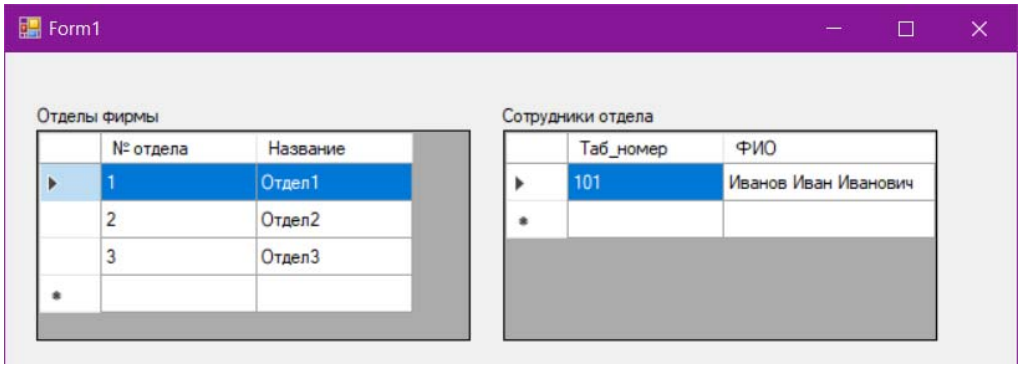
в процессе разработки данные из таблицы не отображаются. Теперь необходимо отобразить данные из связанной таблицы «Работники». Для этого разместим на форме еще один компонент DataGridView и в качестве источника данных выберем следующее:



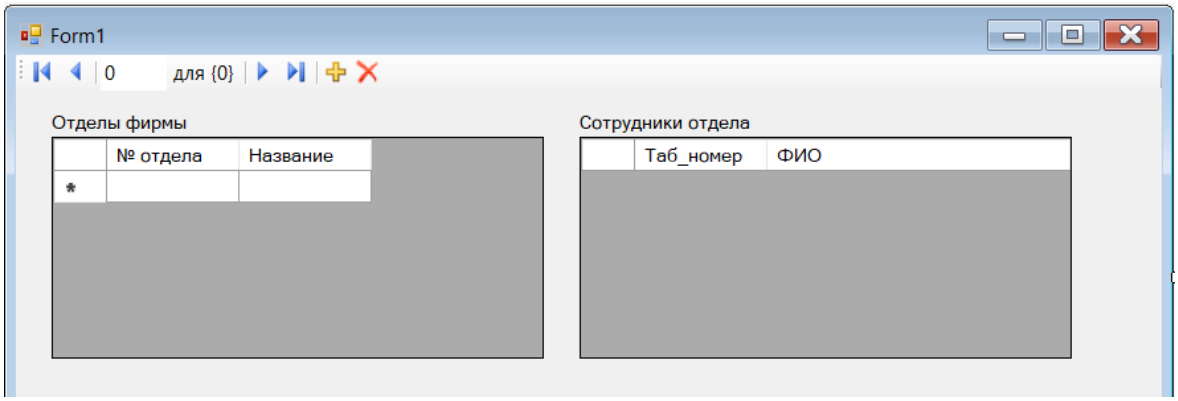
Здесь в качестве источника данных выступает не сама таблица «Работник», а связь (Binding Source) между таблицами «Отдел» и «Работник». Такой выбор гарантирует выбор из таблицы «Работник» только тех строк, которые связаны с текущей строкой в таблице «Отдел». Также такой выбор гарантирует правильность обновления и удаления связанных данных.

Также таблицы можно добавить с помощью окна «Источники данных», перетащив их на форму.



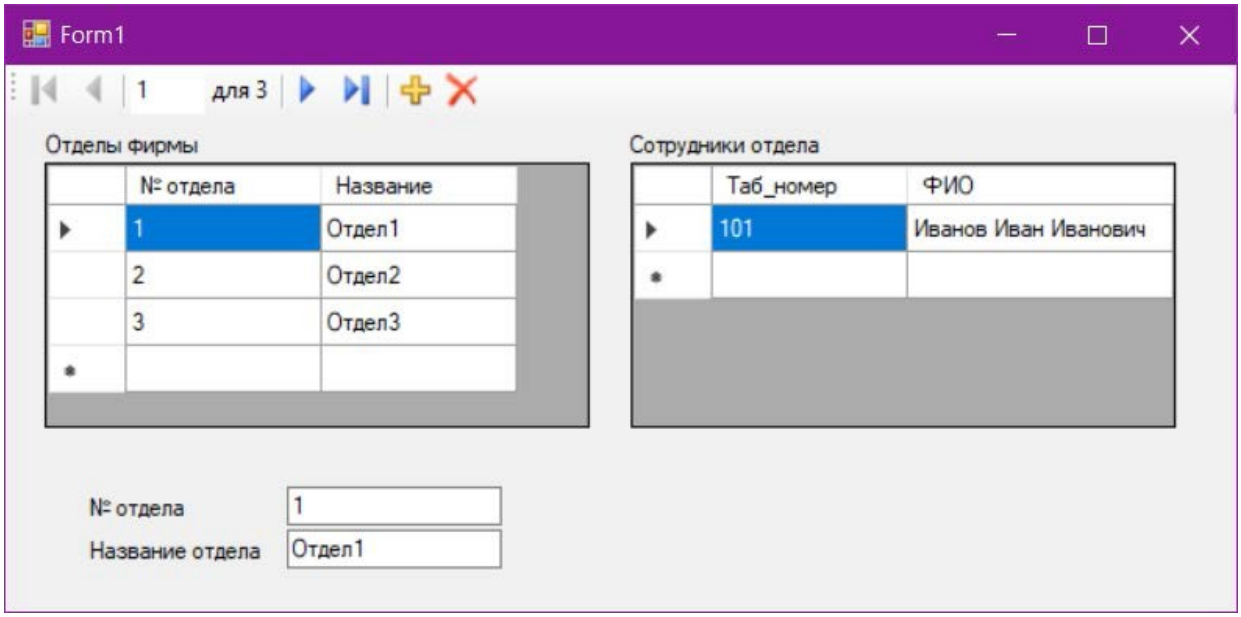


Перемещение по данным при помощи стрелочных клавиш является неудобным. Для упрощения навигации по данным существует компонент BindingNavigator.

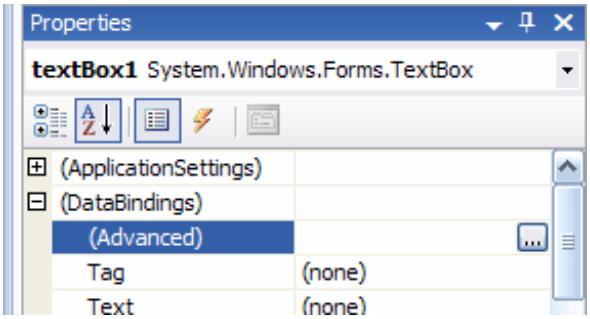


Данный компонент позволяет осуществлять навигацию между записями таблицы, добавлять и удалять строки. Возможности и внешний вид компонента можно настраивать, так как он представляет собой полосу меню ToolStripContainer. Таблица, по которой производится навигация, задается свойством BindingSource.

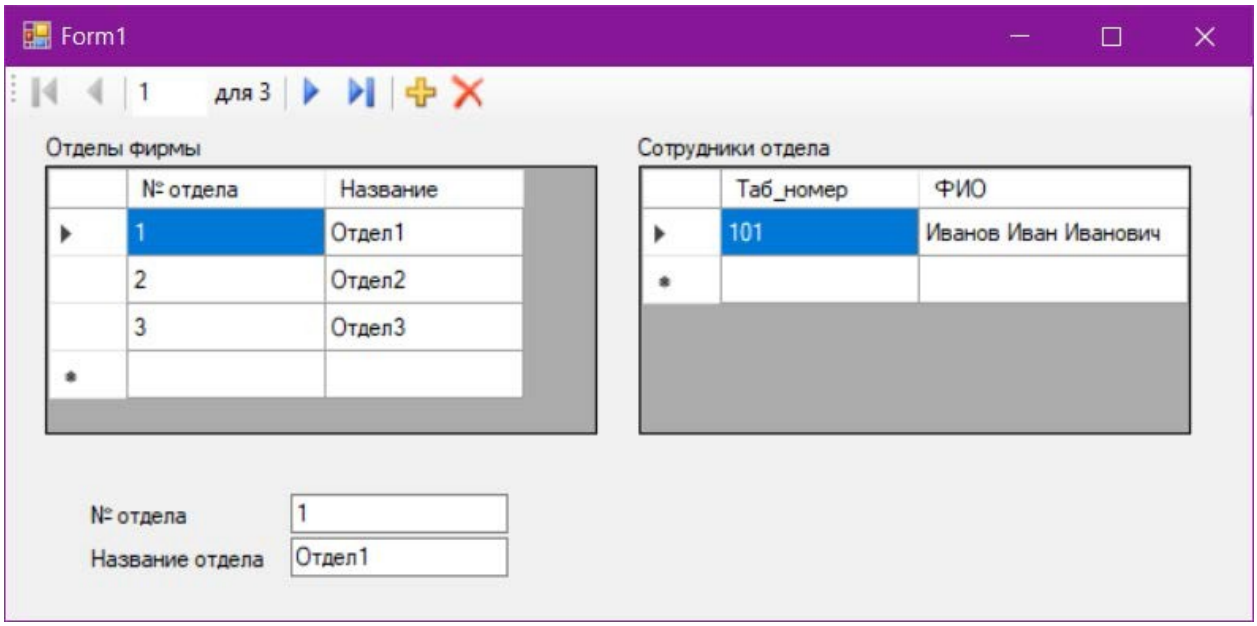
Редактирование данных в ячейках компонента DataGridView бывает неудобно, добавим компоненты TextBox, позволяющие отображать и редактировать данные в отдельных окошках. Для этого разместим на форме контейнер типа Panel, а на нем два компонента TextBox.



Теперь необходимо осуществить привязку компонентов TextBox к соответствующим полям таблицы. Для этого используем свойство из группы DataBindings – Advanced.



Данный диалог позволяет осуществить не только привязку данных, но также задать событие, в рамках которого будет проводиться обновление данных, а также форматирование данных при их выводе. Для верхнего компонента TextBox в выпадающем списке Binding выберем источником данных «отделBindingSource» и поле источника – «id_отдела». Разработанное приложение в работе выглядит следующим образом:

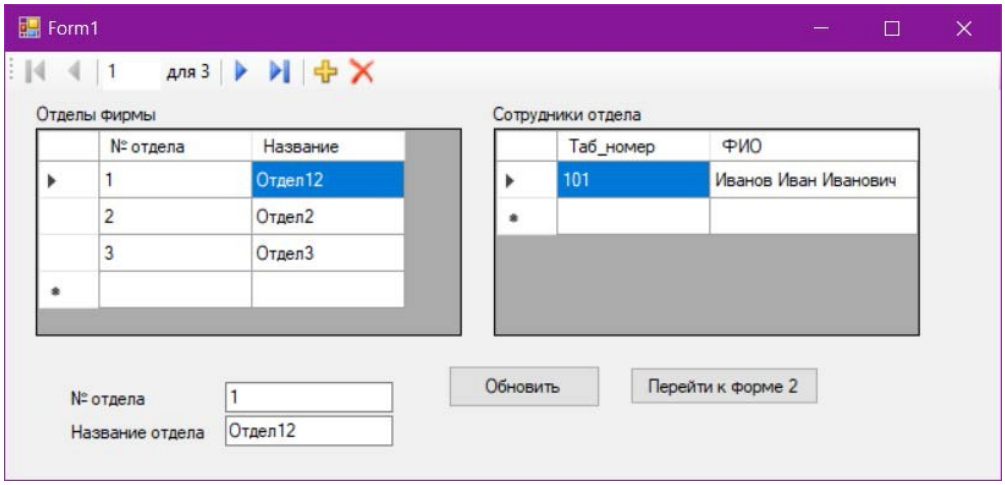


Однако при внесении изменений все новые данные остаются только на форме. В базе данных они не сохраняются, и при повторном вызове приложения, будут отсутствовать. Это происходит потому, что данные были загружены в объект DataSet, который представляет собой копию таблицы в памяти. Все действия выполняются с этой копией. Для того чтобы изменения отобразились в базе данных, необходимо выполнить метод Update класса TableAdapter. Таким образом, в разрабатываемом приложении необходимо разместить кнопку «Обновить» и записать в обработчик события Click следующий программный код:

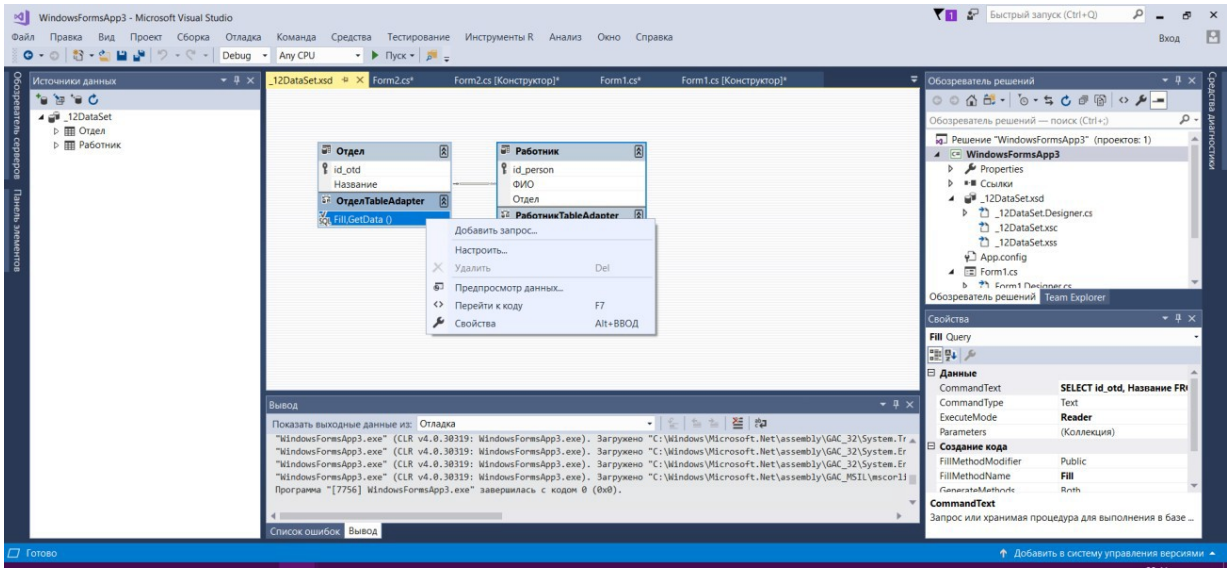
```
название_таблицы_TableAdapter.Update(название_DataSet);
this.Validate();
this.название_BindingSource.EndEdit();
this.tableAdapterManager.UpdateAll(this.название_DataSet);
```

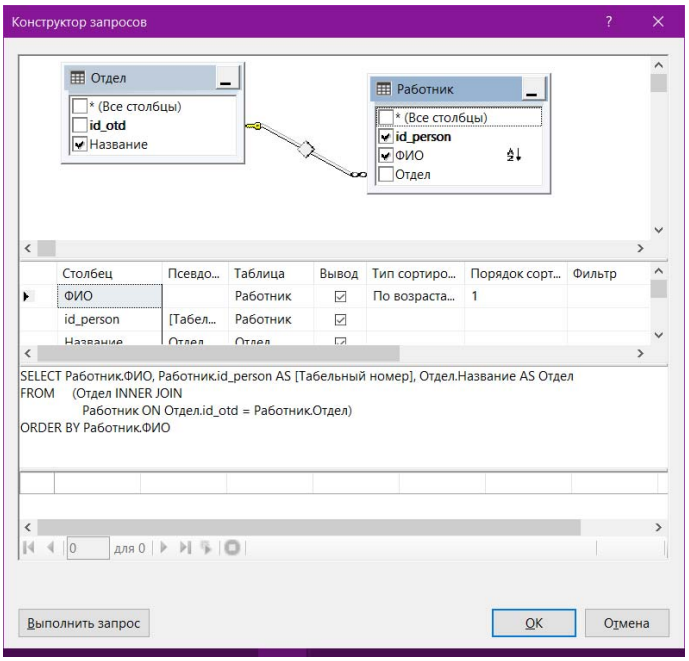
// если появляется надпись, что tableAdapterManager - нет в пространстве имен, перетащите любую таблицу из источника данных на форму (потом можно удалить)

Данный код обновляет информацию в таблицах, предоставляемых источником данных. Отметим, что данный метод является перегруженным, и его варианты позволяют обновлять как отдельную строку таблицы, так и группу строк.



Далее создаем вторую форму с DataGridView, которая будет содержать отображение запроса к таблицам. Переход к форме должен осуществляться по кнопке. Запрос можно предварительно составить/проверить в конструкторе запросов.






```
string mybdpath = "путь к БД";
string conStr = "строка подключения" + mybdpath;
OleDbConnection connection = new
OleDbConnection(conStr); OleDbDataAdapter adapter = new
OleDbDataAdapter(); connection.Open();
OleDbCommand command = new OleDbCommand("SQL_запрос",
          connection);      // добавляем текст SQL
          запроса
connection.Close();
adapter.SelectCommand =
command; DataSet dataSet = new
DataSet();
adapter.Fill(dataSet);
название dataGridView.DataSource =
dataSet.Tables[0]; adapter.Update(dataSet);
// для работы нужно подключить пространство имен System.Data.OleDb;
// строку подключения можно найти, задав поиск по проекту
«connectionString="Provider»
```

Обзор объектов ADO .NET

Источник данных DataSet

Основным объектом ADO является источник данных, представленный объектом DataSet. DataSet состоит из объектов типа DataTable и объектов DataRelation. В коде к ним можно обращаться как к свойствам объекта DataSet, то есть, используя точечную нотацию. Свойство Tables возвращает объект типа DataTableCollection, который содержит все объекты DataTable используемой базы данных.

Таблицы и поля (объекты DataTable и DataColumn)

Объекты DataTable используются для представления одной из таблиц базы данных в DataSet. В свою очередь, DataTable составляется из объектов DataColumn.

DataColumn – это блок для создания схемы DataTable. Каждый объект DataColumn имеет свойство DataType, которое определяет тип данных, содержащихся в каждом объекте DataColumn. Например, можно ограничить тип данных до целых, строковых и десятичных чисел. Поскольку данные, содержащиеся в DataTable, обычно переносятся обратно в исходный источник данных, необходимо согласовывать тип данных с источником.

Объекты DataRelation

Объект DataSet имеет также свойство Relations, возвращающее коллекцию DataRelationCollection, которая в свою очередь состоит из объектов DataRelation. Каждый объект DataRelation выражает отношение между двумя таблицами (сами таблицы связаны по какому-либо полю (столбцу)). Следовательно, эта связь осуществляется через объект DataColumn.

Строки (объект DataRow)

Коллекция Rows объекта DataTable возвращает набор строк (записей) заданной таблицы. Эта коллекция используется для изучения результатов запроса к базе данных. Мы можем обращаться к записям таблицы как к элементам простого массива.

DataAdapter

DataSet – это специализированный объект, содержащий образ базы данных. Для осуществления взаимодействия между DataSet и собственно источником данных используется объект типа DataAdapter. Само название этого объекта – адаптер, преобразователь, – указывает на его природу. DataAdapter содержит метод Fill() для обновления данных из базы и заполнения DataSet.

Объекты DBConnection и DBCommand

Объект DBConnection осуществляет связь с источником данных. Эта связь может быть одновременно использована несколькими командными объектами. Объект DBCommand позволяет послать базе данных команду (как правило, команду SQL или хранимую процедуру). Объекты DBConnection и DBCommand иногда создаются неявно в момент создания объекта DataSet, но их также можно создавать явным образом.

Контрольные вопросы

1. Как передаются данные из БД MS Access в приложение? Расскажите о работе компонента «Binding Source».
2. Какие существуют параметры соединения для механизма ADO?
3. Как осуществляется отображение данных на форме?
4. Какой компонент реализует навигацию? Принцип работы.
5. Опишите, как и с помощью каких компонентов и их свойств вы создали экранную форму?
6. Как осуществляется привязка данных, обновление данных?
7. Опишите основные объекты ADO.NET, их структуру и функции.
8. Какие объекты могут быть источниками данных для приложения?
9. Что такое строка соединения, какие параметры она описывает?
10. Что такое и для чего используется DataGridView?