Оглавление

Тема 1. Создание структуры базы данных и импорт исходных данных....3
1.1. Проектирование структуры базы данных....3
1.2. Создание структуры базы данных в Microsoft SQL Server....7
1.3. Подготовка файлов формата csv к импорту в базу данных....12
1.4. Подготовка таблиц Microsoft Excel к импорту в базу данных....14
1.5. Импорт исходных данных из Microsoft Excel в базу данных....16

Тема 2. Импорт в базу данных из файлов JSON....19

Тема 3. Импорт в базу данных из графических файлов....27

Тема 4. Использование для поля таблицы только уникальных значений....32

Тема 5. Преобразование серверной базы данных в локальную базу данных и подключение ее к проекту....35
5.1. Открепление базы данных от SQL-сервера и подготовка файлов базы данных....35
5.2. Присоединение базы данных к проекту. Способ 1....37
5.3. Решение проблемы несовместимости версий базы данных. Присоединение базы данных к проекту. Способ 2....40

Тема 6. Модуль авторизации пользователей с различными правами....43
6.1. Подготовка и подключение базы данных, настройка таблицы....43
6.2. Формирование интерфейса формы для авторизации, создание и настройка форм для рабочих мест пользователей....45
6.3. Описание программного кода для авторизации....47

Тема 7. Модуль создания и проверки капчи....50

Тема 8. Модуль регистрации пользователей....54
8.1. Создание и подключение DLL, содержащей функцию для проверки пароля....54
8.2. Создание формы для регистрации пользователя....56

Тема 9. Модуль редактирования профиля пользователя....61
9.1. Формирование интерфейса формы для редактирования профиля пользователя....61
9.2. Описание программного кода обработчиков событий....63

Тема 10. Модуль нечеткого поиска с учетом расстояния Левенштейна....66
10.1. Формирование интерфейса формы для нечеткого поиска....66
10.2. Описание программного кода для нечеткого поиска....70

Тема 11. Модуль экспорта данных в Microsoft Excel....73
11.1. Создание шаблона для экспорта в Microsoft Excel....73
11.2. Подключение к приложению библиотеки для работы с Microsoft Excel....74
11.3. Экспорт в Excel всех записей....74
11.4. Экспорт в Excel выделенных записей....76
11.5. Работа с приложением на другом компьютере....77

Тема 12. Модуль экспорта данных в Microsoft Word....80
12.1. Создание шаблона для экспорта в Microsoft Word....80
12.2. Подключение к приложению библиотеки для работы с Microsoft Word....82
12.3. Экспорт в Word всех записей....82

12.4. Экспорт в Word выделенных записей....84
12.5. Работа с приложением на другом компьютере....86

Тема 13. Модуль экспорта данных в файлы формата CSV....88
13.1. Экспорт в файл формата CSV всех записей....88
13.2. Экспорт в файл формата CSV выделенных записей....89

Тема 14. Программирование специализированной СУБД «Агентство недвижимости»....91
14.1. Создание базы данных, импорт исходных данных, авторизация, регистрация и редактирование профиля....94
14.2. Формирование интерфейса и описание программного кода для рабочего места клиента....94

Тема 15. Программирование специализированной СУБД «Марафон»....97

Тема 16. Программирование специализированной СУБД «Швейная фабрика»....98
16.1. Создание базы данных, импорт исходных данных, авторизация, регистрация и редактирование профиля....101
16.2. Формирование интерфейса и описание программного кода для рабочего места заказчика....101
16.3. Формирование интерфейса и описание программного кода для добавления нового заказа....107
16.4. Нечеткий поиск по названиям тканей с учетом расстояния Левенштейна....117
16.5. Формирование интерфейса и описание программного кода для рабочего места кладовщика....117
16.6. Формирование интерфейса и описание программного кода для приема материалов на склад....121
16.7. Формирование интерфейса и описание программного кода для списания материалов 126
16.8. Формирование интерфейса и описание программного кода для рабочего места менеджера....132
16.9. Программирование экспорта заказов в файлы формата CSV и в Microsoft Word....135

Тема 1. Создание структуры базы данных и импорт исходных данных

1.1. Проектирование структуры базы данных

Рассмотрим проектирование структуры базы данных и импорт исходных данных на конкретном примере.

Ваша задача - разработать информационную систему для агентства недвижимости, которое помогает своим клиентам купить/продать объекты недвижимости (без аренды).

Для работы с информационной системой необходимо выполнить авторизацию. Пользователи могут быть одного из двух типов: клиенты и риелторы. В зависимости от того, кто выполняет авторизацию, необходимо открывать форму, содержащую визуальные объекты для выполнения действий либо клиента, либо риелтора.

Необходимо также предусмотреть регистрацию новых пользователей. Если регистрируется новый пользователь, то у него уровень доступа будет по умолчанию – клиент.

Исходные данные для пользователей находятся в файле «Пользователи.csv». Фотографии пользователей находятся в папках «Клиенты» и «Риелторы».

Клиенты приходят в агентство недвижимости либо с потребностью, либо с предложением. Потребность - желание клиента купить объект недвижимости, соответствующего указанным параметрам.

Предложение - желание клиента продать указанный объект недвижимости за указанную цену.

Сделка - факт осуществления продажи недвижимого имущества. В сделке участвуют две стороны: клиент-покупатель с потребностью и клиент-продавец с предложением.

Продажа объекта недвижимости

1) Клиент обращается в компанию с желанием продать объект недвижимости.
2) «Предложению» клиента назначается ответственный риэлтор. Он осуществляет поиск подходящих потребностей
3) Клиент выбирает потребность и заключается сделка.
4) Система рассчитывает размер комиссии для риелтора клиента-продавца и клиента-покупателя.

Покупка объекта недвижимости

1) Клиент обращается в компанию с желанием купить объект недвижимости.
2) «Потребности» клиента назначается ответственный риэлтор. Он осуществляет поиск подходящих предложений
3) Клиент выбирает предложение и заключается сделка.
4) Система рассчитывает размер комиссии для риелтора клиента-продавца и клиента-покупателя.

Для потребностей и предложений исходные данные находятся в файле «Потребности и предложения.xlsx».

На основании описания задания и анализа исходных данных создайте базу данных, сформируйте связи между таблицами, импортируйте исходные данные в таблицы базы данных. При необходимости выполните дополнительную обработку исходных данных перед импортом.

Исходные данные:

Файл «Пользователи.csv»:

```
id;fam;name;login;password;type
1;Носов;Григорий;gri72sha;2361895;0
2;Колова;Зинаида;kol.zina;zi19na85;0
3;Жгутова;Маргарита;margo;2370912;0
4;Борков;Генадий;borkov;760923127;0
5;Маркова;Елизавета;liZa;547806351;0
6;Тарасов;Анатолий;anatoliy;463863325;0
7;Жукова;Галина;galYa;3256835;0
8;Баранов;Марат;MARAT;436607235;0
9;Филипов;Макар;makar;23537458;0
10;Комарова;Даяна;dayana;421463907;1
11;Давыдов;Назар;naZZar;2145740865;1
12;Беляев;Платон;platon;421683244;1
13;Герасимова;Инна;iHHa;235678432;1
```

Данные в Microsoft Excel для таблицы «Потребности»:

	A	B	C	D	E	F	G	H	I	J	K
1	id клиента	id риелтора	Минимальная площадь	Максимальная площадь	Минимальный этаж	Максимальный этаж	Кол-во комнат минимум	Кол-во комнат максимум	Минимальная цена	Максимальная цена	Завершено
2	10	1	45	50	1	3	2	3	1 700 000р.	1900000руб	True
3	10	1	60	70	1	3	3	3	2 100 000р.	2300000 руб	False
4	10	2	36	42	2	5	1	1	1400000р	1 550 000р.	False
5	11	2	40	45кв.м	2	4	2	2	1750000 р	2000000 р	False
6	11	3	50кв.м	60	2	4	3	4	2 300 000р.	2550000рублей	False
7	11	4	35	40 кв.м.	3	4	1	2	1300000руб	1400 тыс.р.	False
8	11	5	55	65	2	4	3	4	2230000руб.	2 500 000р.	False
9	12	5	40 кв.м.	50	3	4	2	4	1900 тыс.р.	1950000рублей	True
10	12	6	41	45	4	5	2	3	1800 тыс.руб.	1990 тыс.руб.	True
11	13	7	41	50	3	5	2	2	1 500 000р.	1800000руб	False
12											

Данные в Microsoft Excel для таблицы «Предложения»:

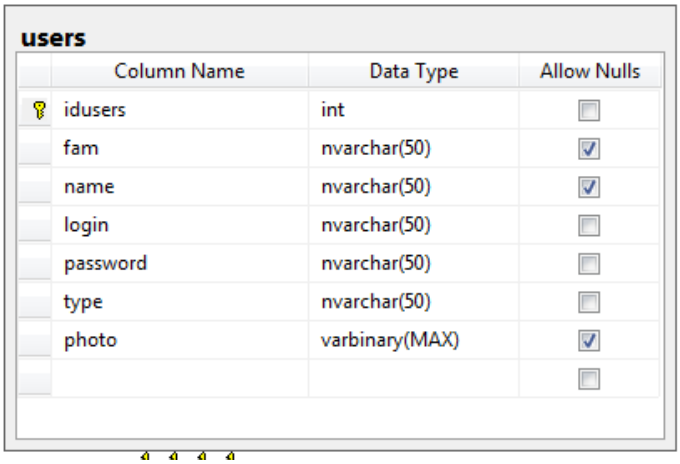
	A	B	C	D	E	F	G	H
1	id клиента	id риелтора	Адрес	Площадь	Этаж	Кол-во комнат	Цена	Завершено
2	12	6	ул. Ленина, д.4, кв.1	41 кв.м	2	2	1950тыс.ру	False
3	10	6	ул. Первомайская, д.10, кв.3	56	3	3	2200000р	False
4	10	7	ул. Наримановская, д.15, кв.4	35	1	1	1 400 000р.	False
5	11	7	ул. Вокзальная, д.21, кв.15	42	4	2	1890000руб	False
6	13	7	ул. Луговая, д.20, кв.1	60	1	3	2 300 000р.	True
7	13	8	ул. Коммунистическая, д.5, кв.3	33	2	1	1500000р	False
8	12	9	ул. Садовая, д.21, кв.10	40кв.м.	4	2	1900000	True
9	12	9	ул. Чкалова, д.12, кв.2	45	2	2	1950000	False
10	12	9	ул. Красная, д.5, кв.3	70	1	4	2500 тыс.р.	False

Нам нужно сформировать структуру базы данных на основе описания задания и исходных данных

Шаг 1. По заданию сказано, что для работы с информационной системой необходимо выполнить авторизацию. Пользователи могут быть одного из двух типов: клиенты и риелторы. Среди файлов с исходными данными у нас есть файл «Пользователи.csv». Файлы формата csv можно открыть в Microsoft Excel. Откройте его в Microsoft Excel. На экране появится следующая таблица:

	A	B	C	D	E	F
1	id	fam	name	login	password	type
2	1	Носов	Григорий	gri72sha	2361895	0
3	2	Колова	Зинаида	kol.zina	zi19na85	0
4	3	Жгутова	Маргарит	margo	2370912	0
5	4	Борков	Генадий	borkov	760923127	0
6	5	Маркова	Елизавета	liZa	547806351	0
7	6	Тарасов	Анатолий	anatoliy	463863325	0
8	7	Жукова	Галина	galYa	3256835	0
9	8	Баранов	Марат	MARAT	436607235	0
10	9	Филипов	Макар	makar	23537458	0
11	10	Комарова	Даяна	dayana	421463907	1
12	11	Давыдов	Назар	naZZar	2145740865	1
13	12	Беляев	Платон	platon	421683244	1
14	13	Герасимо	Инна	iHHa	235678432	1
15						

В этом файле указаны id, фамилии, имена, логины, пароли и типы пользователей. Кроме того, у нас есть папки с именами «Клиенты» и «Риелторы». В этих папках находятся фотографии пользователей. Таким образом можно сделать вывод, что одна из таблиц нашей базы данных будет таблица users следующей структуры:

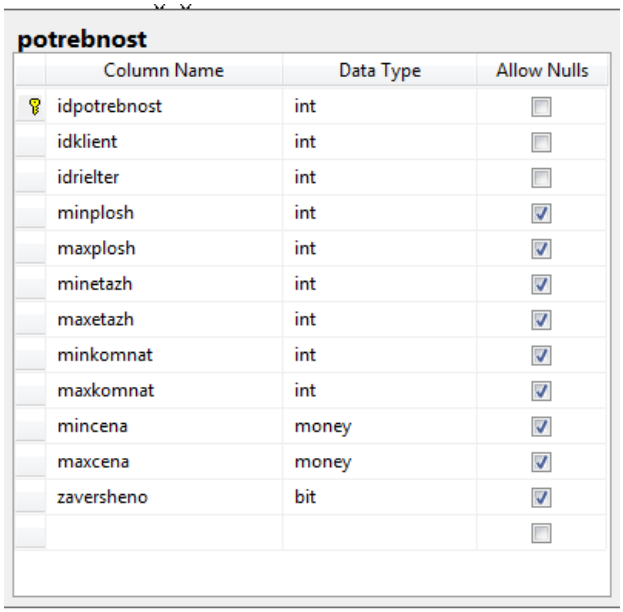


idusers - это первичный ключ. Для него будем использовать тип int. fam, name, login, password, type будут являться строками текста. Для строк текста рекомендуется использовать тип nvarchar. Если оставить тип данных по умолчанию - nchar, то поле будет иметь фиксированную длину, а вместо недостающих символов будут использоваться пробелы. Для хранения фотографий будем использовать тип varbinary(MAX). Этот тип подходит для хранения любых двоичных данных.

Поля idusers, login, password, type являются обязательными для заполнения. Они не могут быть пустыми.

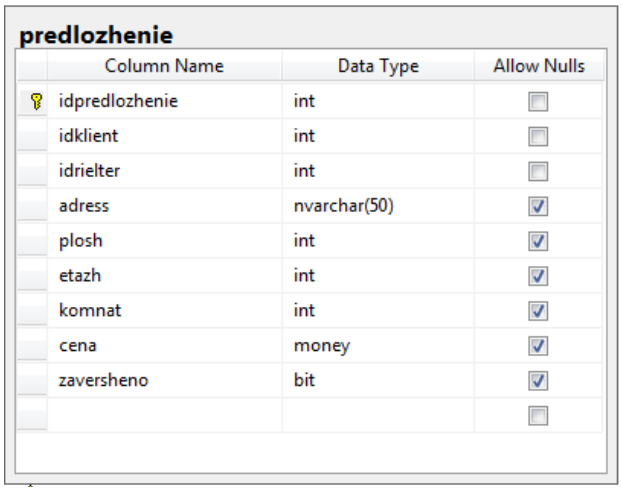
Шаг 2. По условию задачи сказано, что клиенты приходят в агентство недвижимости либо с потребностью, либо с предложением. Для потребностей и предложений исходные данные находятся в файле «Потребности и предложения.xlsx». Откройте этот файл в Microsoft Excel.

Исходя из содержимого таблиц приходим к выводу, что для нашей базы данных нужно использовать еще две таблицы:



potrebnost

Column Name	Data Type	Allow Nulls
idpotrebnost	int	☐
idklient	int	☐
idrielter	int	☐
minplosh	int	☑
maxplosh	int	☑
minetazh	int	☑
maxetazh	int	☑
minkomnat	int	☑
maxkomnat	int	☑
mincena	money	☑
maxcena	money	☑
zaversheno	bit	☑
		☐

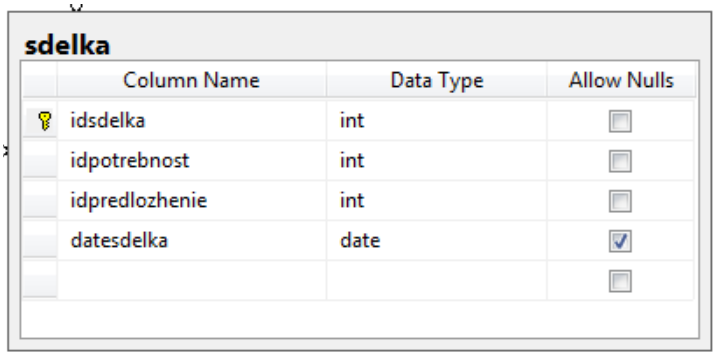


predlozhenie

Column Name	Data Type	Allow Nulls
idpredlozhenie	int	☐
idklient	int	☐
idrielter	int	☐
adress	nvarchar(50)	☑
plosh	int	☑
etazh	int	☑
komnat	int	☑
cena	money	☑
zaversheno	bit	☑
		☐

В каждой таблице должен быть первичный ключ, даже если он в исходных данных не указан. В нашем случае первичными ключами будут idpotrebnost и idpredlozhenie. Поля idklient и idrielter будут вторичными ключами. Они будут использоваться для связи таблицы users с таблицами potrebnost и predlozhenie. Тип данных для вторичных ключей должен быть таким же, как и для первичного ключа из другой таблицы - int. Минимальная и максимальная площадь, минимальный и максимальный этаж, минимальное и максимальное количество комнат являются целыми числами. Для минимальной и максимальной цены можно использовать тип money. Поле «завершено» должно хранить логическую величину - True или False. В базах данных для таких полей используется тип bit. Адрес является строкой текста nvarchar.

Шаг 3. В условии задачи сказано, что При выборе одного из подходящих предложений для совершения сделки и нажатия на кнопку «Совершить сделку», необходимо закрыть потребность и предложение, вычислить комиссию каждому риэлтору. А также сказано, что нужно выполнить экспорт в Excel данных по закрытым сделкам для данного риелтора и сделать вывод в отчет фамилии клиента-продавца, фамилии клиента-покупателя, даты совершения сделки. Отсюда делаем вывод, что для нашей базы данных нужна еще одна таблица sdelka, в которой хранилась бы информация о выполненных сделках:

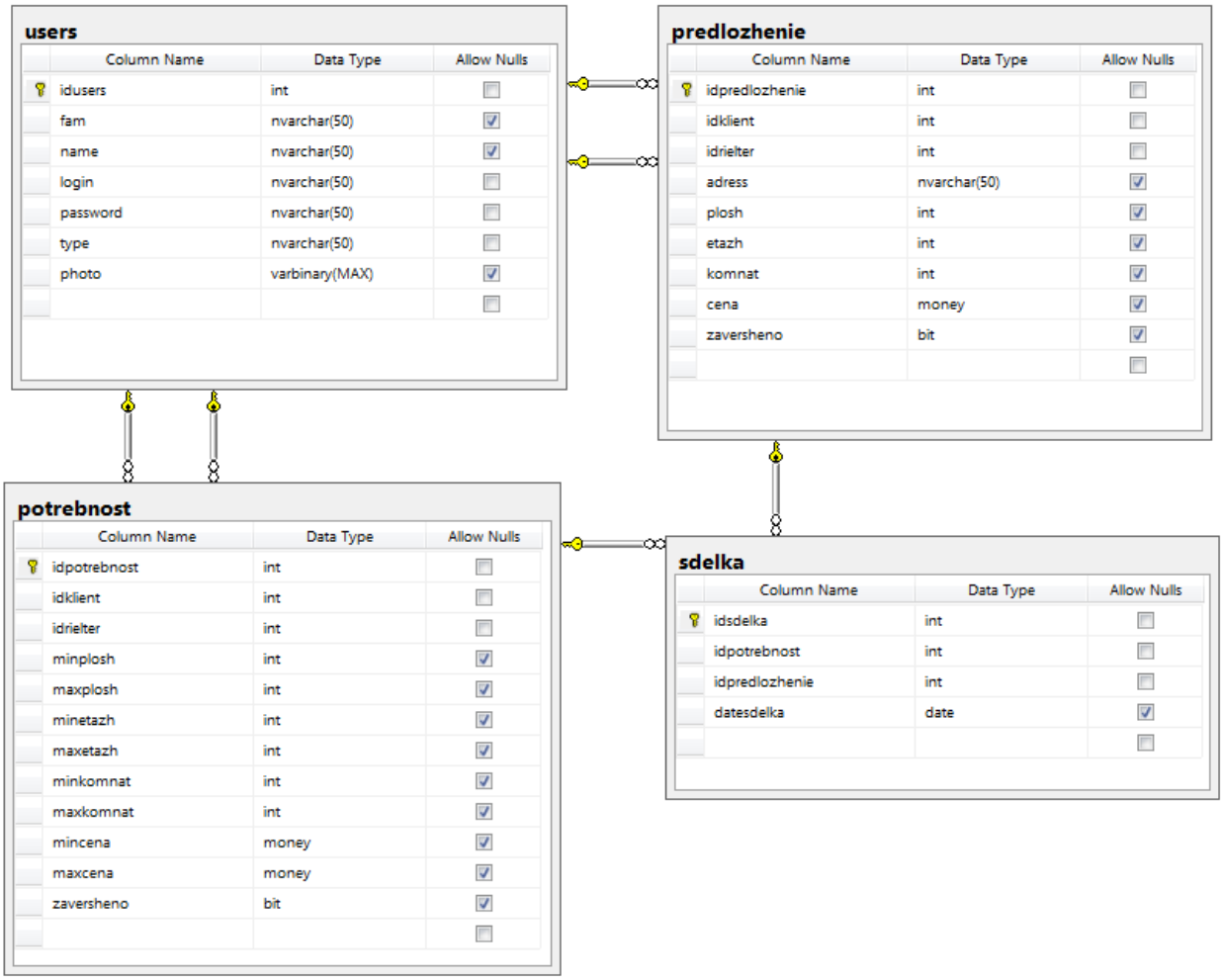


sdelka

Column Name	Data Type	Allow Nulls
idsdelka	int	☐
idpotrebnost	int	☐
idpredlozhenie	int	☐
datesdelka	date	☑
		☐

idsdelka - это первичный ключ для таблицы. В сделке участвует одна потребность и одно предложение, поэтому будем использовать два вторичных ключа - idpotrebnost и idpredlozhenie. Кроме того, нужно запоминать дату сделки.

Наша база данных должна иметь следующую структуру:



Между таблицами users и predlozhenie будут две связи: idusers - idklient и idusers - idrielter. Обе связи «один-ко-многим».

Между таблицами users и potrebnost также будут две связи: idusers - idklient и idusers - idrielter. Обе связи «один-ко-многим».

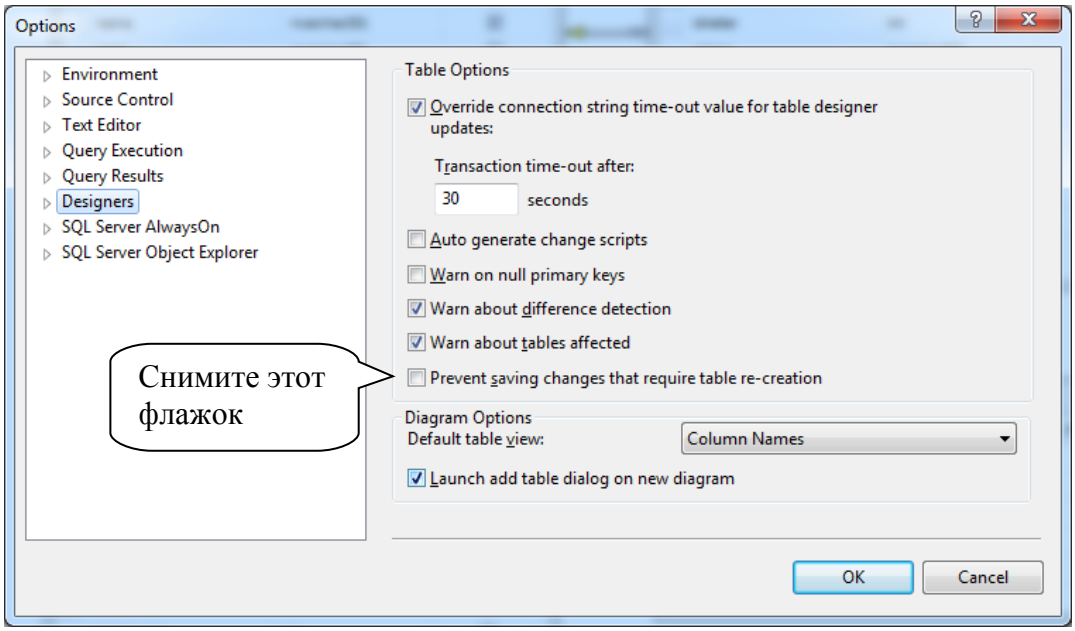
Между таблицами potrebnost и sdelka будет связь idpotrebnost - idpotrebnost. Тип отношения - «один-ко-многим».

Между таблицами predlozhenie и sdelka будет связь idpredlozhenie - idpredlozhenie. Тип отношения - «один-ко-многим».

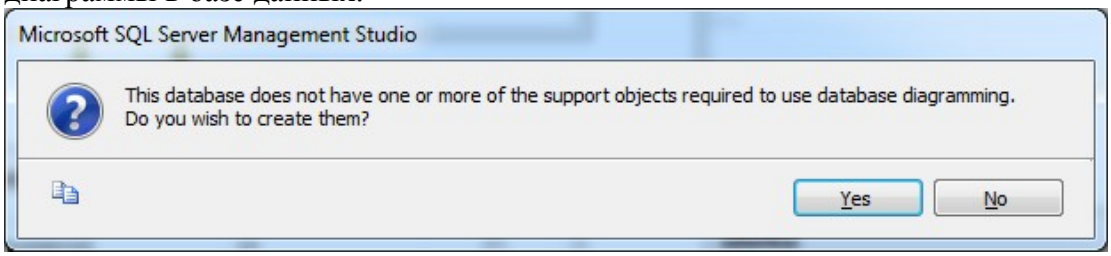
1.2. Создание структуры базы данных в Microsoft SQL Server

Шаг 1. Запустите Microsoft SQL Server. Либо создайте новую базу данных БДАгентство, либо авторизуйтесь с использованием логина и пароля, которые вам выделили для выполнения работы.

Шаг 2. В процессе создания структуры базы данных между таблицами нужно будет создавать связи. Возможно, после создания связей нужно будет внести изменения в структуру таблиц - добавить или удалить поля, изменить тип поля и т.п. При попытке сохранить изменения в базе данных чаще всего в этом случае будет появляться сообщение о невозможности сохранить изменения. Нужно изменить одну из настроек Microsoft SQL Server. Откройте меню Tools ► Options... ► Designer. Снимите флажок с кнопки «Prevent saving changes that require table re-creation».

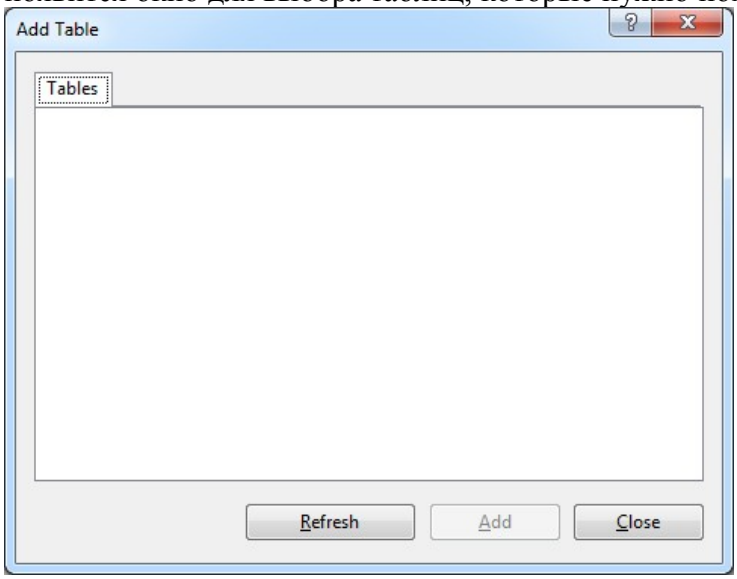


Шаг 3. В панели Object Explorer разверните базу данных БДАгенство. А в ней разверните узел Database Diagrams. На экране появится окно с предложением использовать диаграммы в базе данных.



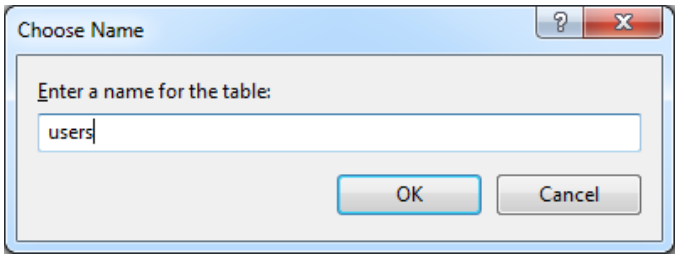
Нажмите на кнопку Yes.

Шаг 4. Создайте новую диаграмму. Для этого выполните щелчок правой кнопкой мыши на узле Database Diagrams и выберите пункт New Database Diagrams. На экране появится окно для выбора таблиц, которые нужно поместить в диаграмму.

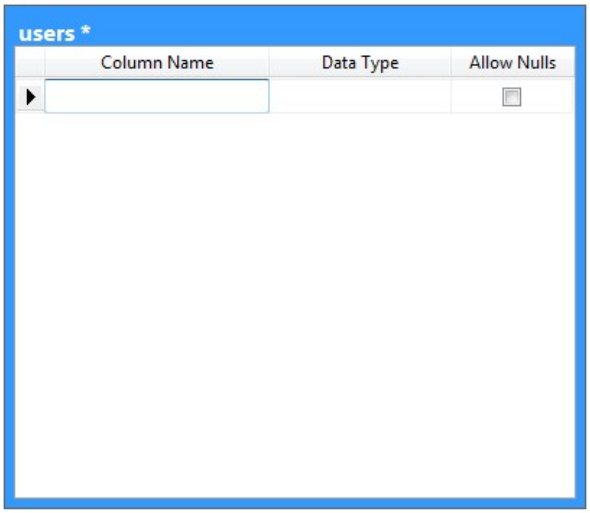


В нашей базе данных пока нет ни одной таблицы. Поэтому просто закройте это окно.

Шаг 5. Для добавления новой таблицы выполните щелчок правой кнопкой мыши на пустом месте диаграммы и выберите пункт меню New Table...

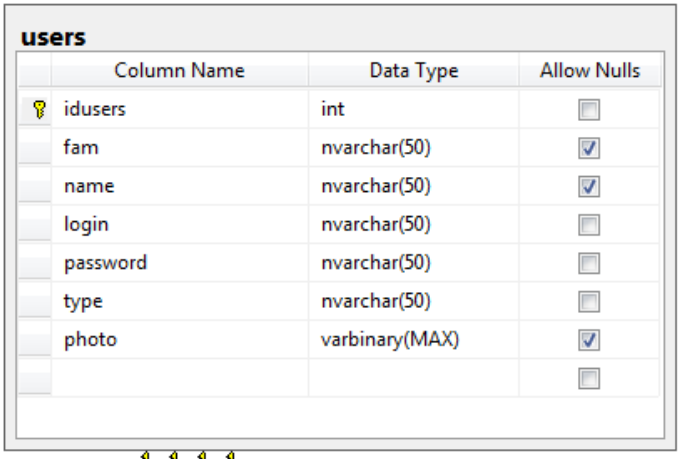


Укажите имя нашей первой таблицы users и нажмите кнопку OK. На экране появится пустое окно, в котором нужно будет описать структуру нашей будущей таблицы.

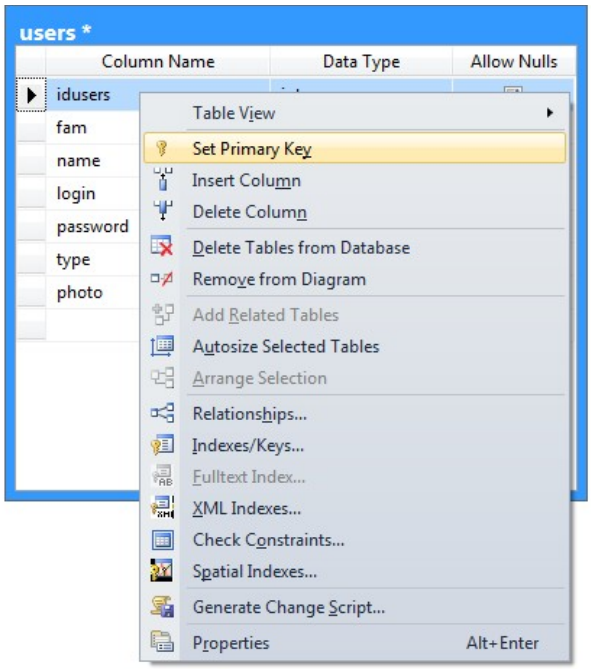


В столбце «Column Name» нужно указывать имена полей, в столбце «Data Type» следует выбрать тип поля. В столбце «Allow Nulls» устанавливается флажок, если в поле допускаются пустые значения или снимается флажок, если в поле пустые значения не допускаются.

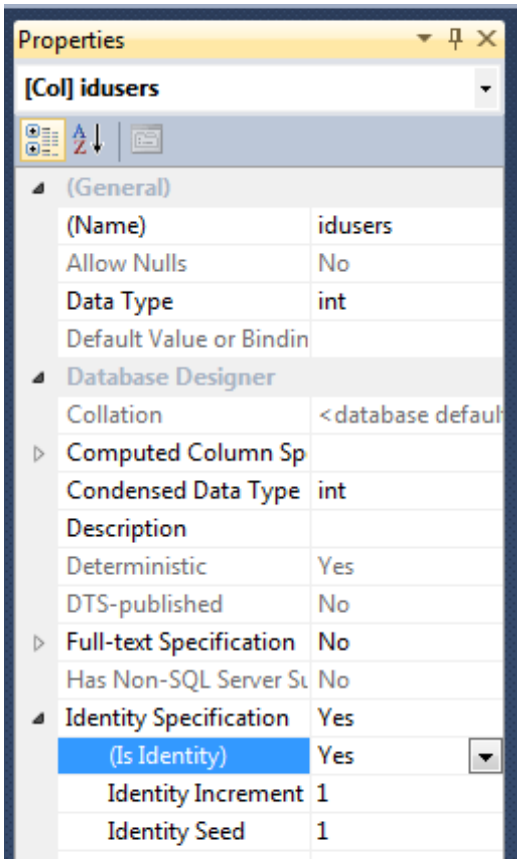
Создайте таблицу users следующей структуры:



Шаг 6. Поле idusers должно быть первичным ключом. Для этого выполните щелчок правой кнопкой мыши на поле idusers и выберите пункт меню «Set Primary Key».



Кроме того, поле, являющееся первичным ключом, должно автоматически заполняться значениями. Для этого выделите поле idusers и в панели Properties в свойстве Identity Specification ► (Is Identity) установите значение Yes.

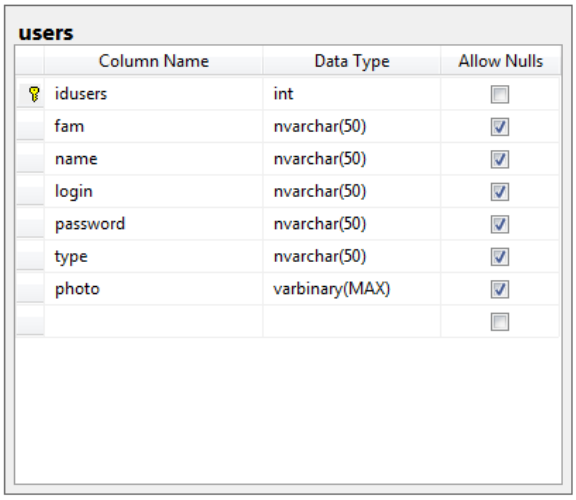


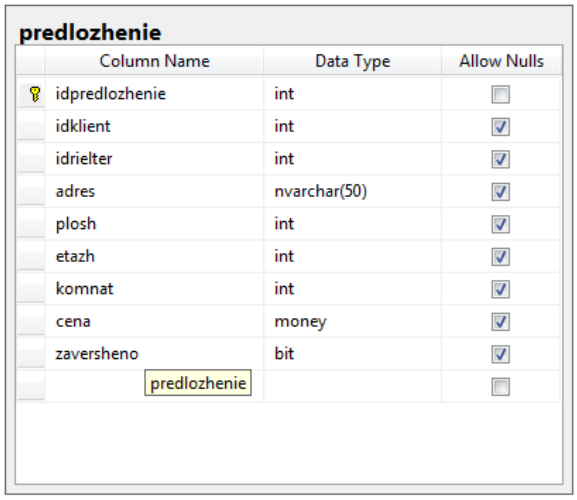
Теперь при добавлении записей в таблицу users в поле iduser будут автоматически записываться значения 1, 2, 3 и т.д.

Шаг 7. Точно так же, как мы создавали структуру таблицы uses, сделайте структуру остальных таблиц: potrebnost, predlozhenie и sdelka. Обязательно в каждой таблице сделайте первичный ключ и включите автоматическое заполнение значений первичного ключа.

Внимание! Для вторичных ключей в свойстве Identity Specification не нужно устанавливать значение Yes. Это делается только для первичных ключей.

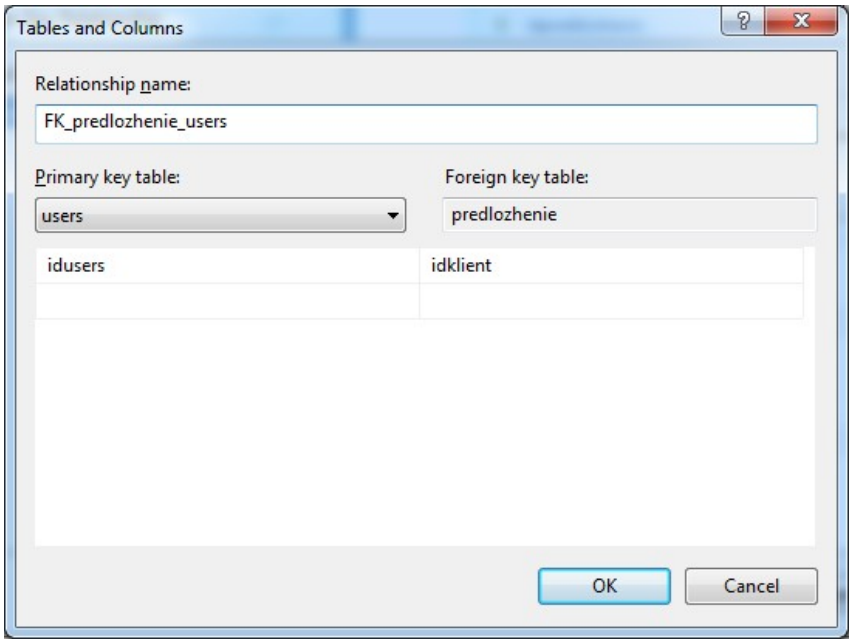
Шаг 8. Сделайте связи между таблицами. Рассмотрим, как это делается, на примере двух таблиц: users и predlozhenie.





Поле idusers таблицы users нужно связать с полями idklient и idrielter таблицы predlozhenie.

Установите курсор мыши на изображение ключа слева от имени поля idusers и отбуксируйте его на поле idklient в таблице predlozhenie. На экране появится окно:



В этом окне указывается имя связи, а также отображается информация о том, какое поле с каким полем будет связано. Мы видим, что поле idusers таблицы users будет связано с полем idklient таблицы predlozhenie. Проверьте еще раз, все ли верно указано в диалоговом окне. Если все верно, то нажимайте ОК. Между нашими таблицами появится связь.

Точно также сделайте вторую связь между полями idusers и idrielter.

Аналогичным образом сделайте остальные связи в нашей базе данных. Смотрите в п. 1.1. структуру нашей базы данных.

Шаг 9. Поле login в таблице users должно иметь только уникальные значения. Не может быть нескольких пользователей с одинаковыми логинами. О том, как настроить поле таблицы таким образом, чтобы оно хранило только уникальные значения, читайте в теме №4.

1.3. Подготовка файлов формата csv к импорту в базу данных

Файлы формата CSV это текстовые файлы, которые имеют следующую структуру:

```
поле1;поле2;поле2
значение_поля1;значение_поля2;значение_поля3
значение_поля1;значение_поля2;значение_поля3
значение_поля1;значение_поля2;значение_поля3
```

...

В первой строке перечисляются имена полей, разделенные символом ";". В остальных строках перечисляются значения полей для отдельных записей, также разделенные символом ";".

Файлы формата csv можно открыть в приложении Microsoft Excel. Откройте в Microsoft Excel файл «Пользователи.csv». Если с csv-файлом все в порядке, то откроется таблица примерно такого вида:

	A	B	C	D	E	F
1	id	fam	name	login	password	type
2	1	Носов	Григорий	gri72sha	2361895	0
3	2	Колова	Зинаида	kol.zina	zi19na85	0
4	3	Жгутова	Маргарит	margo	2370912	0
5	4	Борков	Генадий	borkov	760923127	0
6	5	Маркова	Елизавета	liZa	547806351	0
7	6	Тарасов	Анатолий	anatoliy	463863325	0
8	7	Жукова	Галина	galYa	3256835	0
9	8	Баранов	Марат	MARAT	436607235	0
10	9	Филипов	Макар	makar	23537458	0
11	10	Комарова	Даяна	dayana	421463907	1
12	11	Давыдов	Назар	naZZar	2145740865	1
13	12	Беляев	Платон	platon	421683244	1
14	13	Герасимо	Инна	iHHa	235678432	1
15						

Каждое поле занимает отдельный столбец таблицы. В первой строке таблицы перечисляются имена полей. В остальных строках перечисляются записи.

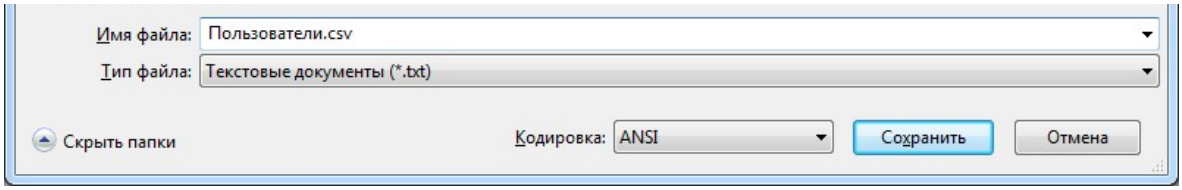
Однако, при открытии csv-файла в Microsoft Excel может появиться иная картина:

	A	B	C	D	E	F
1	id	fam	name	login	password	type
2	1	РќРѕСЃРѕРІ	Р“СЂРёРіРѕСЂРёР№	gri72sha	2361895	0
3	2	РљРѕР»РѕРІР°	Р—РёРЅР°РёРґР°	kol.zina	zi19na85	0
4	3	Р–РіСѓС‚РѕРІР°	РњР°СЂРіР°СЂРёС‚	margo	2370912	0
5	4	Р‘РѕСЂРєРѕРІ	Р“РµРЅР°РґРёР№	borkov	760923127	0
6	5	РњР°СЂРєРѕРІР°	Р•Р»РёР·Р°РІРµС‚Р°	liZa	547806351	0
7	6	РўР°СЂР°СЃРѕРІ	РђРЅР°С‚РѕР»РёР№	anatoliy	463863325	0
8	7	Р–СѓРєРѕРІР°	Р“Р°Р»РёРЅР°	galYa	3256835	0
9	8	Р‘Р°СЂР°РЅРѕРІ	РњР°СЂР°С‚	MARAT	436607235	0
10	9	РӀРёР»РёРїРѕРІ	РњР°РєР°СЂ	makar	23537458	0
11	10	РљРѕРјР°СЂРѕРІР°	Р”Р°СЏРЅР°	dayana	421463907	1
12	11	Р”Р°РІС‹РґРѕРІ	РќР°Р·Р°СЂ	naZZar	2145740865	1
13	12	Р‘РµР»СЏРµРІ	РџР»Р°С‚РѕРЅ	platon	421683244	1
14	13	Р“РµСЂР°СЃРёРјРѕРІР°	РРЅРЅР°	iHHa	235678432	1

Если вместо русских букв отображаются нечитаемые символы, то это означает, что в csv-файле применяется кодировка, которая не подходит для использования в Microsoft Excel. В этом случае сделайте следующее:

Шаг 1. Откройте файл «Пользователи.csv» в Блокноте.

Шаг 2. Выберите меню Файл ► Сохранить как...



В нижней части диалогового окна выберите кодировку ANSI, после чего сохраните файл.

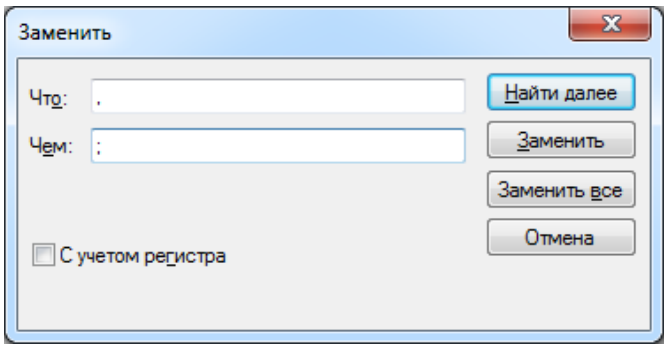
Предположим, вы открыли csv-файл в Microsoft Excel и на экране появилась следующая таблица:

	A	B	C	D	E	F
1	id	fam	name	login	password	type
2	1	Носов	Григорий	gri72sha	2361895	0
3	2	Колова,Зинаида	kol.zina	zi19na85	0	
4	3	Жгутова,Маргарита	margo	2370912	0	
5	4	Борков	Генадий	borkov	760923127	0
6	5	Маркова	Елизавета,liZa	547806351	0	
7	6	Тарасов	Анатолий	anatoliy	463863325	0
8	7	Жукова	Галина	galYa	3256835	0
9	8	Баранов	Марат	MARAT	436607235	0
10	9	Филипов	Макар	makar	23537458	0
11	10	Комарова	Даяна	dayana	421463907	1
12	11	Давыдов	Назар	naZZar	2145740865	1
13	12	Беляев	Платон	platon	421683244	1
14	13	Герасимова	Инна	iHHa	235678432	1

Обратите внимание - в 3 и 4 строках фамилии и имена находятся в одном столбце B, при этом они разделяются запятой. А в 6-й строке имя и логин точно так же находятся в одном столбце C, разделенные запятой. Это означает что в исходном csv-файле между некоторыми полями вместо символа «;» используется символ «,». В этом случае сделайте следующее:

Шаг 1. Откройте файл «Пользователи.csv» в Блокноте.

Шаг 2. Выберите меню Правка ► Заменить... В диалоговом окне укажите, что нужно заменить запятую на точку с запятой. Нажмите на кнопку «Заменить все».



Шаг 3. Заново откройте в Microsoft Excel измененный csv-файл, проверьте, чтобы в Microsoft Excel все отображалось без ошибок. Сохраните файл в формате xlsx.

1.4. Подготовка таблиц Microsoft Excel к импорту в базу данных

Откройте в Microsoft Excel файл «Потребности и предложения.xlsx». На экране появится следующая таблица, содержащая данные о потребностях:

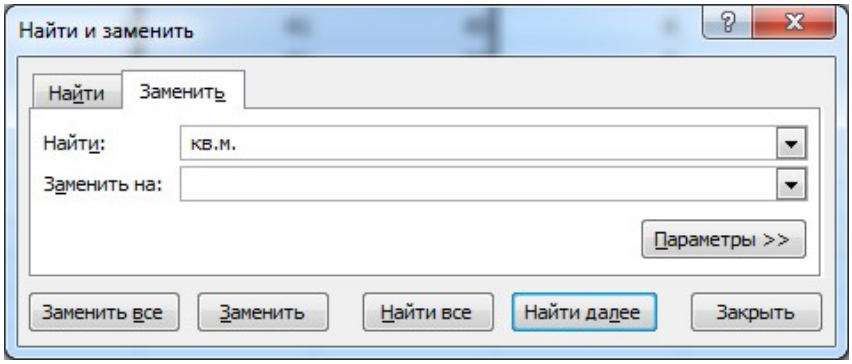
	A	B	C	D	E	F	G	H	I	J	K
1	id клиента	id риелтора	Минимальная площадь	Максимальная площадь	Минимальный этаж	Максимальный этаж	Кол-во комнат минимум	Кол-во комнат максимум	Минимальная цена	Максимальная цена	Завершено
2	10	1	45	50	1	3	2	3	1 700 000р.	1900000руб	True
3	10	1	60	70	1	3	3	3	2 100 000р.	2300000 руб	False
4	10	2	36	42	2	5	1	1	1400000р	1 550 000р.	False
5	11	2	40	45кв.м	2	4	2	2	1750000 р	2000000 р	False
6	11	3	50кв.м	60	2	4	3	4	2 300 000р.	2550000рублей	False
7	11	4	35	40 кв.м.	3	4	1	2	1300000руб	1400 тыс.р.	False
8	11	5	55	65	2	4	3	4	2230000руб.	2 500 000р.	False
9	12	5	40 кв.м.	50	3	4	2	4	1900 тыс.р.	1950000рублей	True
10	12	6	41	45	4	5	2	3	1800 тыс.руб.	1990 тыс.руб.	True
11	13	7	41	50	3	5	2	2	1 500 000р.	1800000руб	False
12											

В этой таблице видно, что для минимальной и максимальной площади используются целые числа, являющиеся квадратными метрами. Но в некоторых строках возле числа есть символы «кв.м». Причем обозначения используются двух видов: «кв.м» и « кв.м.». Нужно убрать эти обозначения и оставить только числа. В нашей таблице небольшое количество строк и убрать обозначения можно было бы вручную. Но что делать, если в таблице содержится несколько сотен или тысяч записей? В этом случае лучше всего воспользоваться операцией поиска и замены. Нужно заменить удаляемый текст на пустой текст. Причем замену нужно делать в порядке убывания символов в заменяемом тексте. Т.е. сначала нужно заменить « кв.м.» на пустой текст, а затем «кв.м» на пустой текст.

Шаг 1. Выделите два столбца таблицы с минимальной и максимальной площадью.

Шаг 2. Нажмите комбинацию клавиш Ctrl + H (либо откройте диалоговое окно для поиска и замены иным способом).

Шаг 3. В поле «Найти» введите строку « кв.м.», а в поле «Заменить на» не вводите ничего, оставьте пустой текст. Пробелы в поле «Заменить на» вводить не нужно. Пробелы не являются пустым текстом.



Нажмите на кнопку «Заменить все».

Шаг 4. Аналогичным образом замените на пустой текст строку «кв.м».

Обратите внимание на минимальную и максимальную цену. Цена является вещественным числом. Однако в нашей таблице для цен используется ряд обозначений: «рублей», «руб.», «руб», «р.», « р», «р», « тыс.руб.», « тыс.р.».

Обратите внимание, что для цены у нас используются величины разного порядка: рубли и тысячи рублей. В базе данных нам нужно хранить величины только одного порядка. Поэтому тысячи рублей нужно преобразовать в рубли. Это можно сделать путем умножения тысяч рублей на 1000, т.е. к цене справа нужно добавить три нуля.

Шаг 1. Выделите в таблице два столбца с минимальной и максимальной ценой.

Шаг 2. Выполните замену строки « тыс.руб.» на «000».

Шаг 3. Выполните замену строки « тыс.р.» на «000».

Еще раз обратите внимание на то, что выполнять замену нужно в порядке убывания количества символов заменяемого текста. А выделять столбцы нужно для того, чтобы вы были уверены, что в других столбцах, содержащих текстовые данные, не произойдет ошибочной замены фрагмента текста.

Далее у нас остаются обозначения: «рублей», «руб.», «руб», «р.», « р», «р». Выполните замену этих обозначений на пустой текст уже знакомым вам способом.

Обратите внимание на то, что если вы будете выполнять замену не в порядке убывания символов, то могут возникнуть ошибки замены. Например, выполним замену вначале «руб» на пустой текст, а затем «руб.» на пустой текст. После выполнения замены «руб» на пустой текст в тех строках таблицы, где было указано «руб.» после числа останется точка. А это приведет к ошибкам при импорте данных в SQL-сервер.

Обратите внимание, что после замен обозначений денежных величин на пустой текст в нашей таблице в некоторых строках осталось обозначение «р.», хотя мы выполняли замену «р.» на пустой текст.

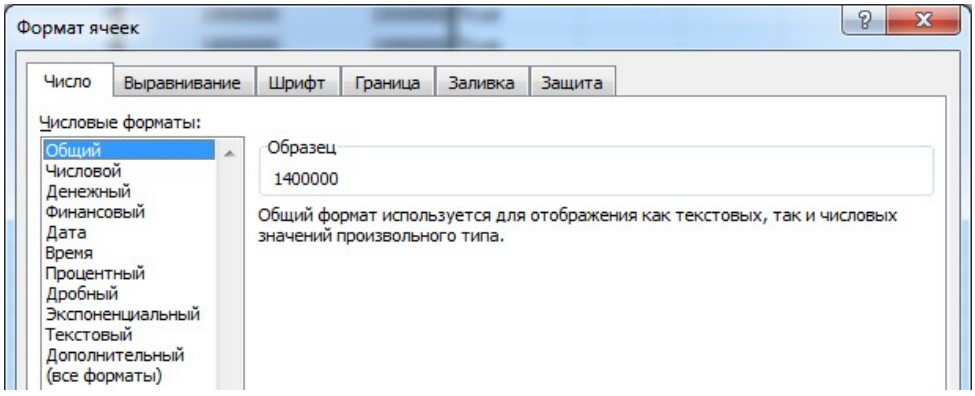
	A	B	C	D	E	F	G	H	I	J	K
1	id клиента	id риелтора	Минимальная площадь	Максимальная площадь	Минимальный этаж	Максимальный этаж	Кол-во комнат минимум	Кол-во комнат максимум	Минимальная цена	Максимальная цена	Завершено
2	10	1	45	50	1	3	2	3	1 700 000р.	1900000	True
3	10	1	60	70	1	3	3	3	2 100 000р.	2300000	False
4	10	2	36	42	2	5	1	1	1400000	1 550 000р.	False
5	11	2	40	45	2	4	2	2	1750000	2000000	False
6	11	3	50	60	2	4	3	4	2 300 000р.	2550000	False
7	11	4	35	40	3	4	1	2	1300000	1400000	False
8	11	5	55	65	2	4	3	4	2230000	2 500 000р.	False
9	12	5	40	50	3	4	2	4	1900000	1950000	True
10	12	6	41	45	4	5	2	3	1800000	1990000	True
11	13	7	41	50	3	5	2	2	1 500 000р.	1800000	False

Это произошло потому, что для ячеек таблицы задан денежный формат. Использование денежного формата автоматически добавляет к числу в ячейках таблицы «р.».

Шаг 1. Выделите в таблице два столбца с минимальной и максимальной ценой.

Шаг 2. Выполните щелчок правой кнопкой мыши на выделенной части таблицы и выберите пункт меню «Формат ячеек...».

Шаг 3. В списке форматов выберите Общий и нажмите кнопку ОК.



Теперь наши данные подготовлены к импорту в базу данных в SQL-сервер.

На вкладке «Предложения» у нас находится таблица, содержащая данные о предложениях:

	A	B	C	D	E	F	G	H
1	id клиента	id риелтора	Адрес	Площадь	Этаж	Кол-во комнат	Цена	Завершено
2	12	6	ул. Ленина, д.4, кв.1	41 кв.м	2	2	1950тыс.руб	False
3	10	6	ул. Первомайская, д.10, кв.3	56	3	3	2200000р	False
4	10	7	ул. Наримановская, д.15, кв.4	35	1	1	1 400 000р.	False
5	11	7	ул. Вокзальная, д.21, кв.15	42	4	2	1890000руб	False
6	13	7	ул. Луговая, д.20, кв.1	60	1	3	2 300 000р.	True
7	13	8	ул. Коммунистическая, д.5, кв.3	33	2	1	1500000р	False
8	12	9	ул. Садовая, д.21, кв.10	40кв.м.	4	2	1900000	True
9	12	9	ул. Чкалова, д.12, кв.2	45	2	2	1950000	False
10	12	9	ул. Красная, д.5, кв.3	70	1	4	2500 тыс.р.	False

Подготовьте эту таблицу к импорту в базу данных так же, как мы подготавливали таблицу «Потребности».

1.5. Импорт исходных данных из Microsoft Excel в базу данных

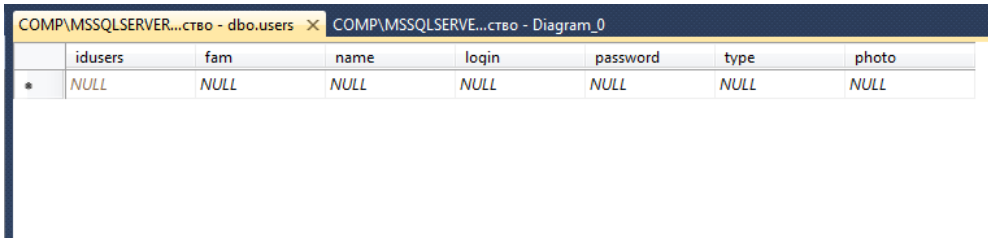
При выполнении импорта исходных данных в таблицы базы данных нужно использовать следующий принцип: вначале выполняется импорт в главную таблицу, а затем в подчиненную. У нас есть исходные данные для трех таблиц: пользователи, потребности и предложения. Главной таблицей здесь является таблица «Пользователи». Между таблицами «Пользователи» и «Потребности», а также между таблицами «Пользователи» и «Предложения» установлены отношения «один-ко-многим». Следовательно, вначале нужно выполнить импорт исходных данных в таблицу «Пользователи».

Если мы вначале выполним импорт данных в таблицы «Предложения» и «Потребности», то у нас в этих таблицах появятся записи, которые не связаны ни с одним пользователем из таблицы «Пользователи». SQL-сервер не разрешит импорт таких данных.

Шаг 1. Откройте в Microsoft Excel данные для таблицы «Пользователи».

	A	B	C	D	E	F
1	id	fam	name	login	password	type
2	1	Носов	Григорий	gri72sha	2361895	0
3	2	Колова	Зинаида	kol.zina	zi19na85	0
4	3	Жгутова	Маргарита	margo	2370912	0
5	4	Борков	Генадий	borkov	760923127	0
6	5	Маркова	Елизавета	liZa	547806351	0
7	6	Тарасов	Анатолий	anatoliy	463863325	0
8	7	Жукова	Галина	galYa	3256835	0
9	8	Баранов	Марат	MARAT	436607235	0
10	9	Филипов	Макар	makar	23537458	0
11	10	Комарова	Даяна	dayana	421463907	1
12	11	Давыдов	Назар	naZZar	2145740865	1
13	12	Беляев	Платон	platon	421683244	1
14	13	Герасимова	Инна	iHHa	235678432	1
15	14	Иванов	Петр	r	r	0
16	15	Петров	Степан	k	k	1
17						

Шаг 2. Откройте в SQL-сервере базу данных. В панели Object Explorer разверните узел Tables. Выполните щелчок правой кнопкой мыши на таблице users и выберите пункт «Edit Top 200 Rows». На экране должна открыться пустая таблица, в которую можно вводить данные.



Шаг 3. В таблице Microsoft Excel нужно расположить столбцы с данными в таком порядке, в каком они идут в базе данных SQL-сервера. Если каких-то столбцов не хватает, то добавьте их. Имена столбцов в Microsoft Excel и имена полей в таблице базы данных могут не совпадать. Главное, чтобы порядок следования столбцов в таблице и порядок следования полей в базе данных был одинаковым.

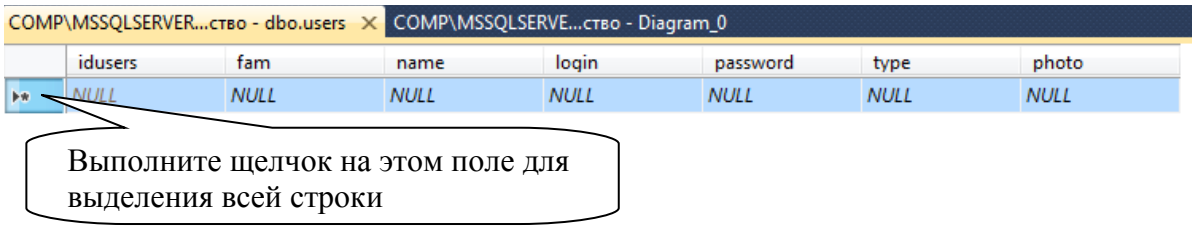
Добавьте в нашу таблицу столбец для фотографии пользователей.

	A	B	C	D	E	F	G
1	id	fam	name	login	password	type	Фотография
2	1	Носов	Григорий	gri72sha	2361895	0	
3	2	Колова	Зинаида	kol.zina	zi19na85	0	
4	3	Жгутова	Маргарита	margo	2370912	0	
5	4	Борков	Генадий	borkov	760923127	0	
6	5	Маркова	Елизавета	liZa	547806351	0	
7	6	Тарасов	Анатолий	anatoliy	463863325	0	
8	7	Жукова	Галина	galYa	3256835	0	
9	8	Баранов	Марат	MARAT	436607235	0	
10	9	Филипов	Макар	makar	23537458	0	
11	10	Комарова	Даяна	dayana	421463907	1	
12	11	Давыдов	Назар	naZZar	2145740865	1	
13	12	Беляев	Платон	platon	421683244	1	
14	13	Герасимова	Инна	iHHa	235678432	1	
15	14	Иванов	Петр	r	r	0	
16	15	Петров	Степан	k	k	1	
17							

Шаг 4. В таблице Microsoft Excel выделите все данные, включая пустые добавленные столбцы. Заголовки столбцов при этом выделять не нужно. Скопируйте выделенный фрагмент таблицы в буфер.

	A	B	C	D	E	F	G
1	id	fam	name	login	password	type	Фотография
2	1	Носов	Григорий	gri72sha	2361895	0	
3	2	Колова	Зинаида	kol.zina	zi19na85	0	
4	3	Жгутова	Маргарита	margo	2370912	0	
5	4	Борков	Генадий	borkov	760923127	0	
6	5	Маркова	Елизавета	liZa	547806351	0	
7	6	Тарасов	Анатолий	anatoliy	463863325	0	
8	7	Жукова	Галина	galYa	3256835	0	
9	8	Баранов	Марат	MARAT	436607235	0	
10	9	Филипов	Макар	makar	23537458	0	
11	10	Комарова	Даяна	dayana	421463907	1	
12	11	Давыдов	Назар	naZZar	2145740865	1	
13	12	Беляев	Платон	platon	421683244	1	
14	13	Герасимова	Инна	iHHa	235678432	1	
15	14	Иванов	Петр	r	r	0	
16	15	Петров	Степан	k	k	1	
17							

Шаг 5. Переключитесь в SQL-сервер. Выделите пустую строку в таблице users, выполнив щелчок на поле слева от этой строки.



Шаг 6. Вставьте содержимое буфера обмена с помощью комбинации клавиш Ctrl + V или любым иным способом.

Шаг 7. Внимательно посмотрите на импортированные данные. Убедитесь, что в каждом поле таблицы находятся правильные данные. Нет такого, что в поле photo будет находиться фамилия, а в поле type - логин пользователя.

Если вы видите, что данные импортировались неправильно, то удалите их и повторите процесс импорта.

Аналогичным образом выполните импорт исходных данных в таблицы potrebnost и predlozhenie

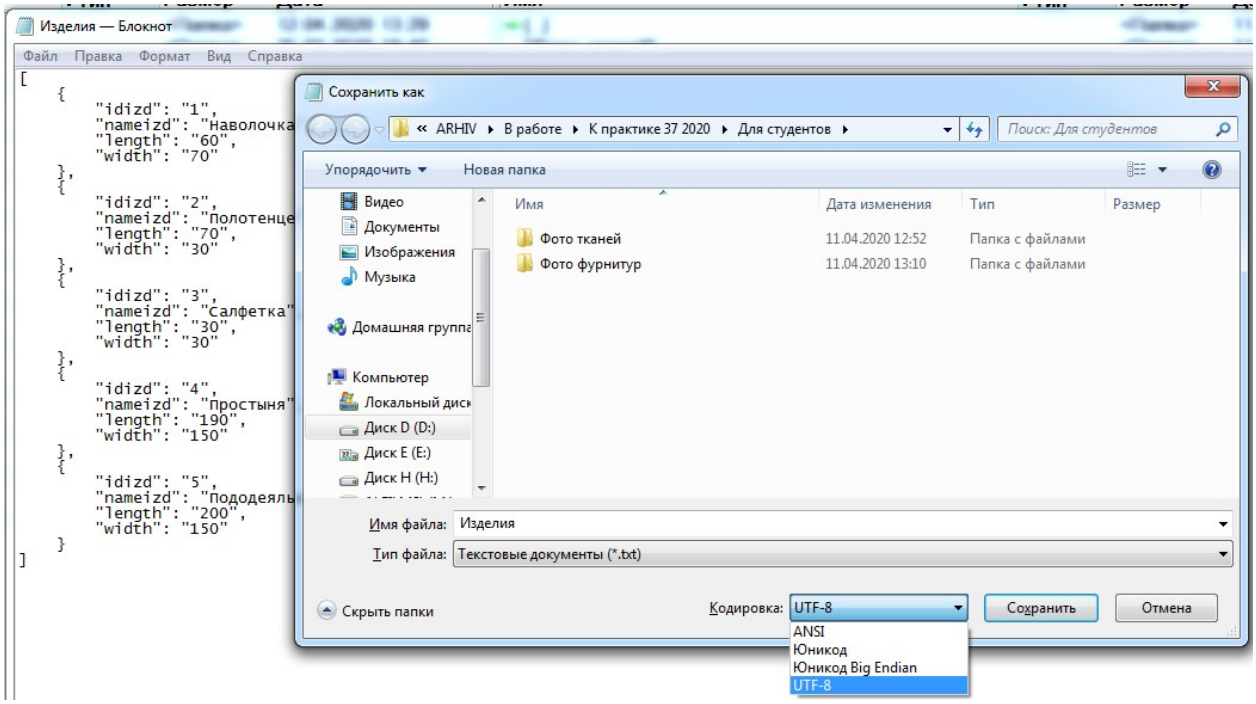
Тема 2. Импорт в базу данных из файлов JSON

Пусть у нас есть файл «Изделия.json» со следующим содержимым:

```
[
   {
      "idizd": "1",
      "nameizd": "Наволочка",
      "length": "60",
      "width": "70"
   },
   {
      "idizd": "2",
      "nameizd": "Полотенце",
      "length": "70",
      "width": "30"
   },
   {
      "idizd": "3",
      "nameizd": "Салфетка",
      "length": "30",
      "width": "30"
   },
   {
      "idizd": "4",
      "nameizd": "Простыня",
      "length": "190",
      "width": "150"
   },
   {
      "idizd": "5",
      "nameizd": "Пододеяльник",
      "length": "200",
      "width": "150"
   }
]
```

Нам необходимо в базе данных создать таблицу Izdeliya и импортировать в эту таблицу данные из JSON-файла.

Шаг 1. Мы будем для импорта данных использовать программу на языке программирования C#. В языке программирования C# для работы с текстом используется кодировка UTF8. Откройте JSON файл в блокноте и сохраните его в этой кодировке.

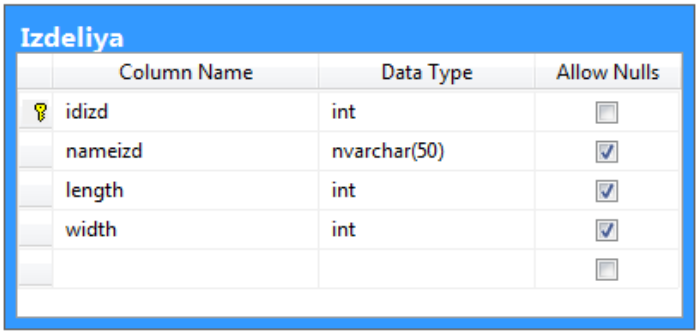


Шаг 2. В Microsoft SQL Server создайте новую базу данных или откройте существующую. Создайте в базе данных таблицу Izdeliya.

В JSON файле мы видим, что в таблице должно быть четыре поля:

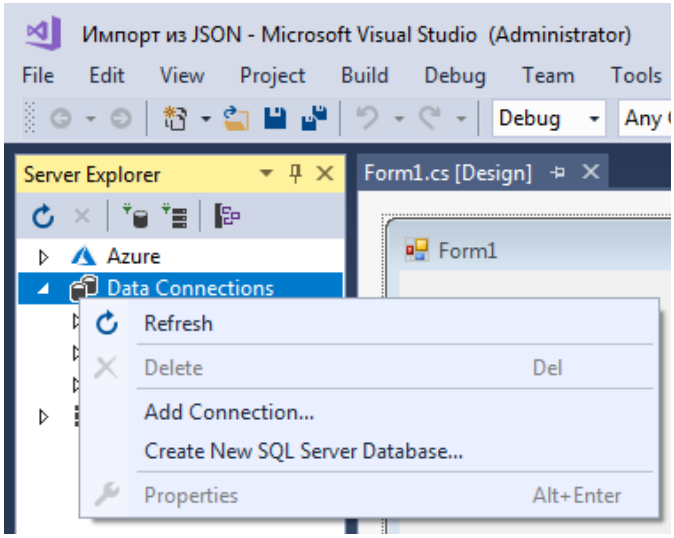
"idizd": "1",
"nameizd": "Наволочка",
"length": "60",
"width": "70"

idizd- это первичный ключ, тип int. Для автоматического заполнения поля значениями установите для этого поля в свойстве Identity Specification значение Yes. Название изделия - это строка текста, длина и ширина задаются в сантиметрах в виде целых чисел.

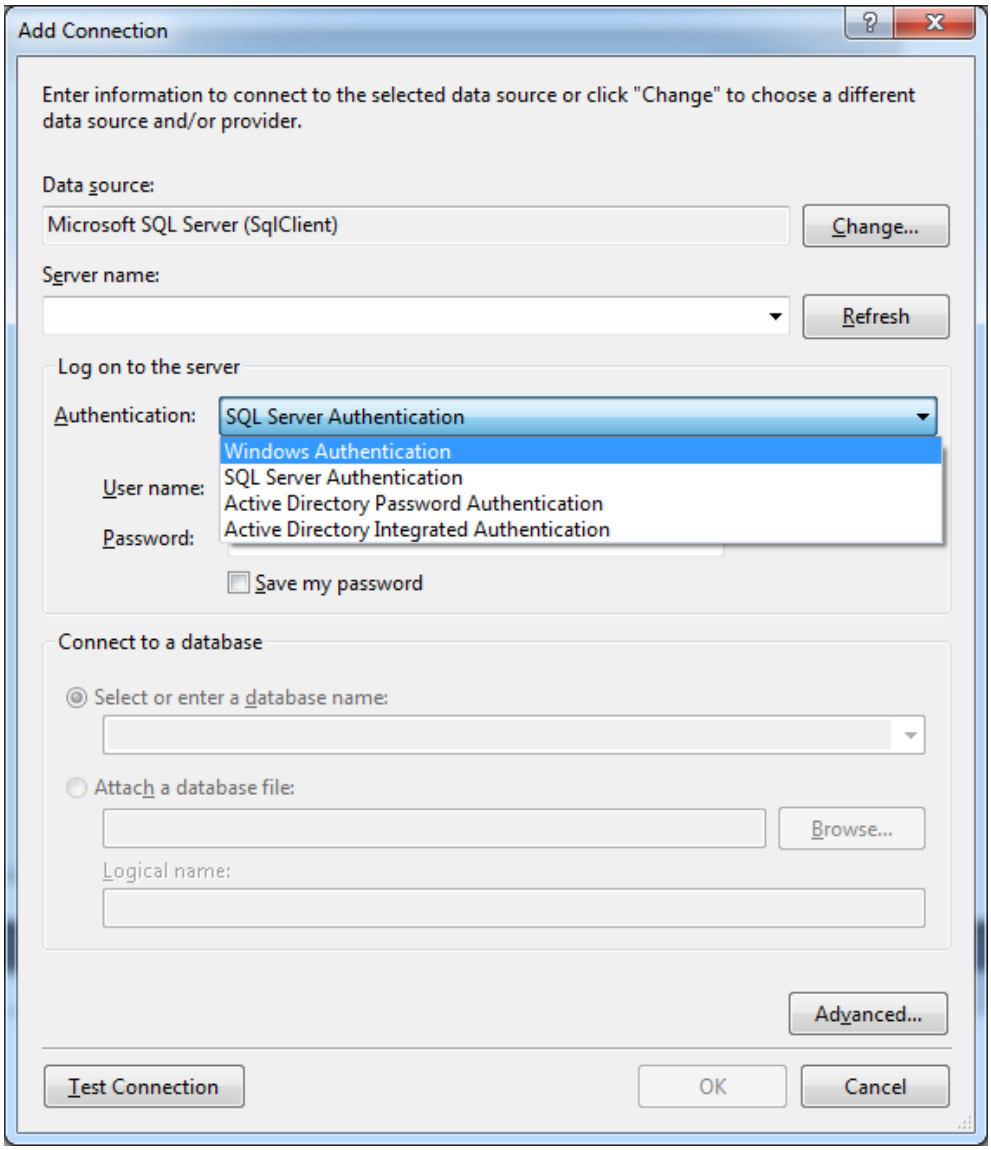


Создайте таблицу с указанными полями и сохраните изменения в базе данных.

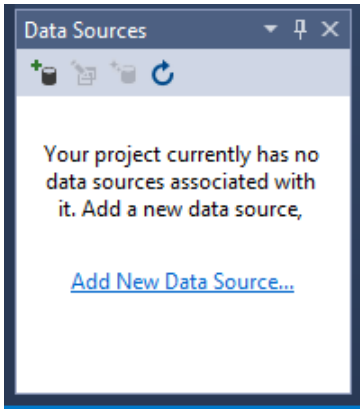
Шаг 3. Запустите Visual Studio. Создайте и сохраните новый проект Windows Forms. Для этого на панели Server Explorer выполните щелчок правой кнопкой мыши на узле Data Connections и выберите пункт Add Connection...



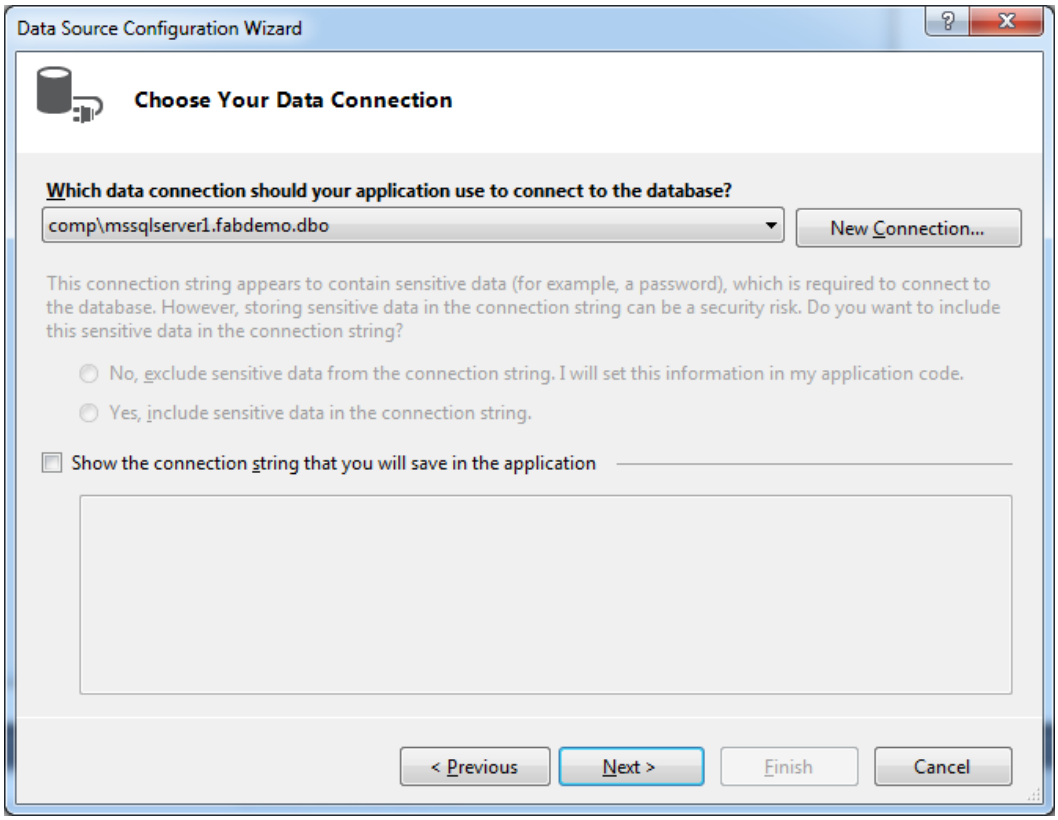
В открывшемся диалоговом окне выберите тип аутентификации. Если вы работаете с базой данных, которая хранится на вашем локальном компьютере, выберите «Windows Authentication». Если эта база данных хранится на SQL-сервере в локальной сети, выберите «SQL Server Authentication». Выберите из списка, или укажите имя сервера, введите при необходимости имя пользователя и пароль для доступа к серверу SQL. Выберите базу данных на сервере и нажмите OK.



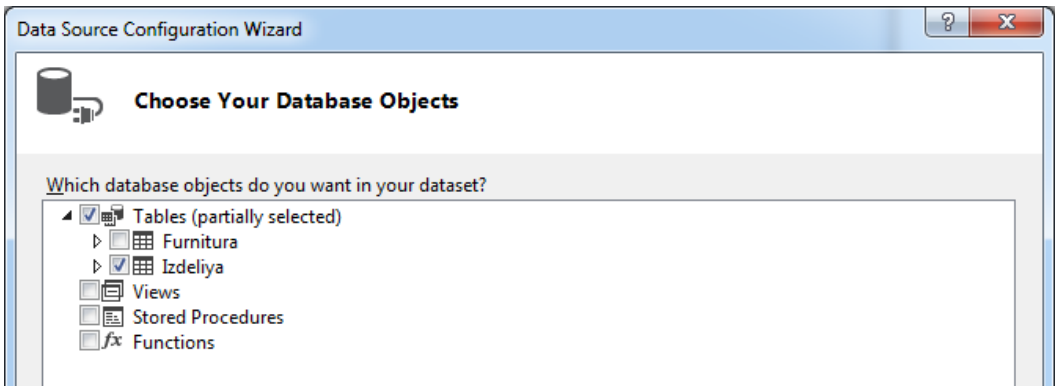
Шаг 4. В панели Data Source добавьте новый источник данных.



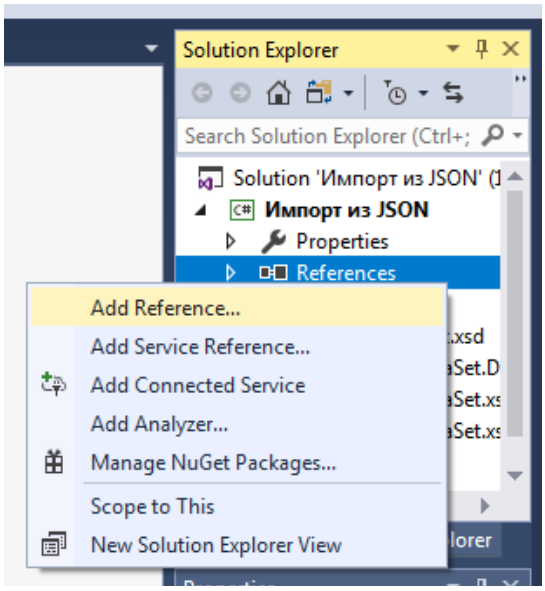
В процессе добавления источника данных выберите правильный Data Connections для подключения к вашей базе данных.



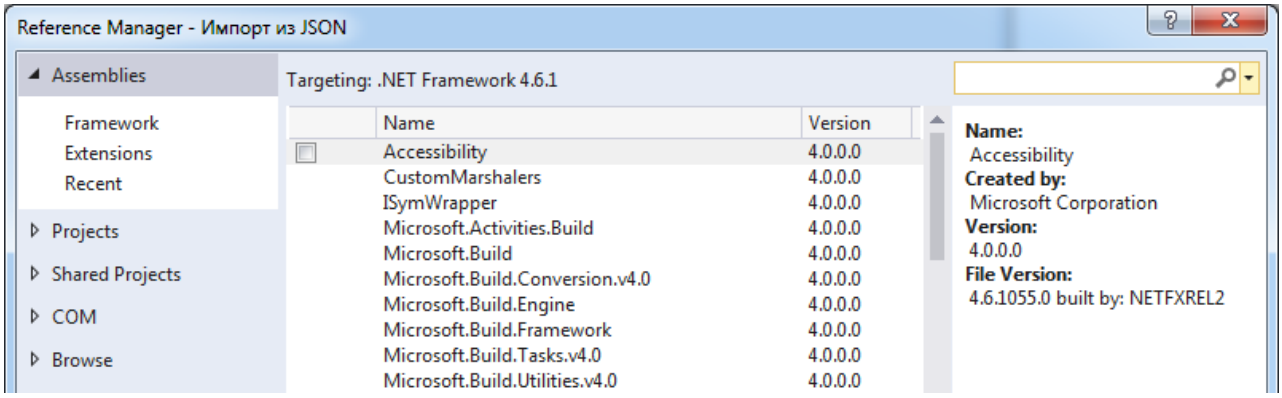
Обязательно выделите таблицу Izdeliya, чтобы она добавилась в источник данных.



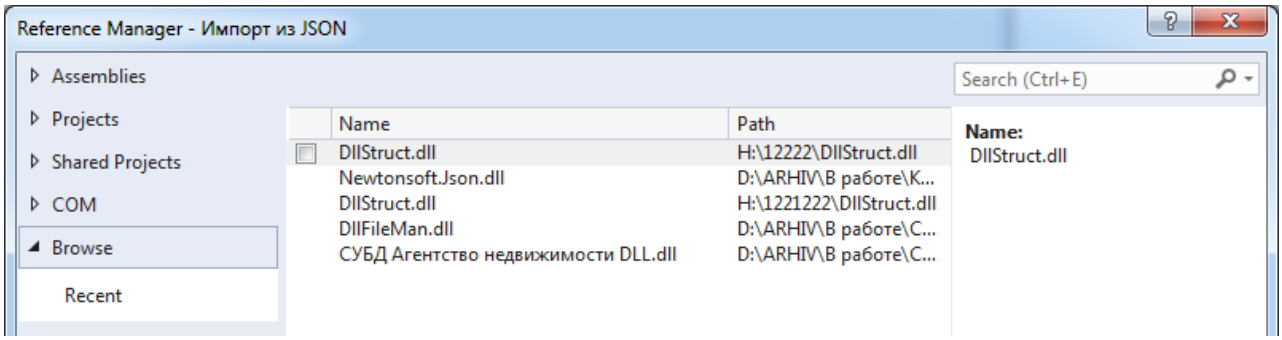
Шаг 5. Для работы с файлами JSON нам потребуется подключить библиотеку Newtonsoft.Json. Если в панели Solution Explorer щелкнуть правой кнопкой мыши на узле References и выбрать пункт Add Reference...



То в разделе Assemblies этой библиотеки может и не быть.



В таком случае нужно найти на компьютере DLL-файл с именем «Newtonsoft.Json» и подключить эту библиотеку как обычный DLL файл. Выполните поиск этого файла во вложенных каталогах по адресу C:\Program Files или C:\Program Files (x86). Затем в диалоговом окне «Reference Manager» выберите Browse и подключите этот файл к проекту.



В программном коде подключите два пространства имен: для работы с JSON и для работы с файлами:

```
using Newtonsoft.Json;
using System.IO;
```

Шаг 6. Скопируйте файл Изделия.json в подкаталог bin \ debug в папку с проектом.

Шаг 7. Давайте еще раз взглянем на структуру JSON-файла.

```
[
  {
    "idizd": "1",
    "nameizd": "Наволочка",
    "length": "60",
    "width": "70"
  },
  {
    "idizd": "2",
    "nameizd": "Полотенце",
    "length": "70",
    "width": "30"
  },
  {
    "idizd": "3",
    "nameizd": "Салфетка",
    "length": "30",
    "width": "30"
  },
]
```

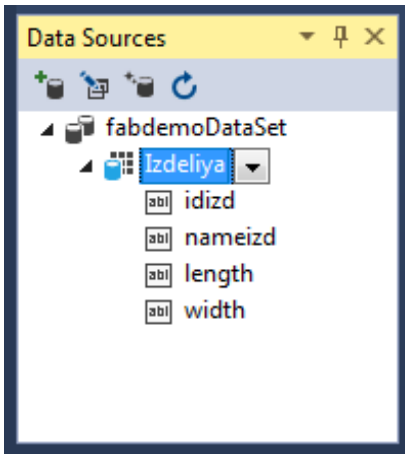
Мы видим, что этот файл состоит из одного объекта - списка, который заключен в квадратные скобки. Внутри этого списка мы видим несколько записей, заключенных в фигурные скобки. Каждая запись состоит из четырех полей.

В классе Form1 опишите класс ClassIzdeliya, который будет использоваться для считывания данных из JSON-файла. А также создайте список List, элементами которого будут экземпляры класса ClassIzdeliya.

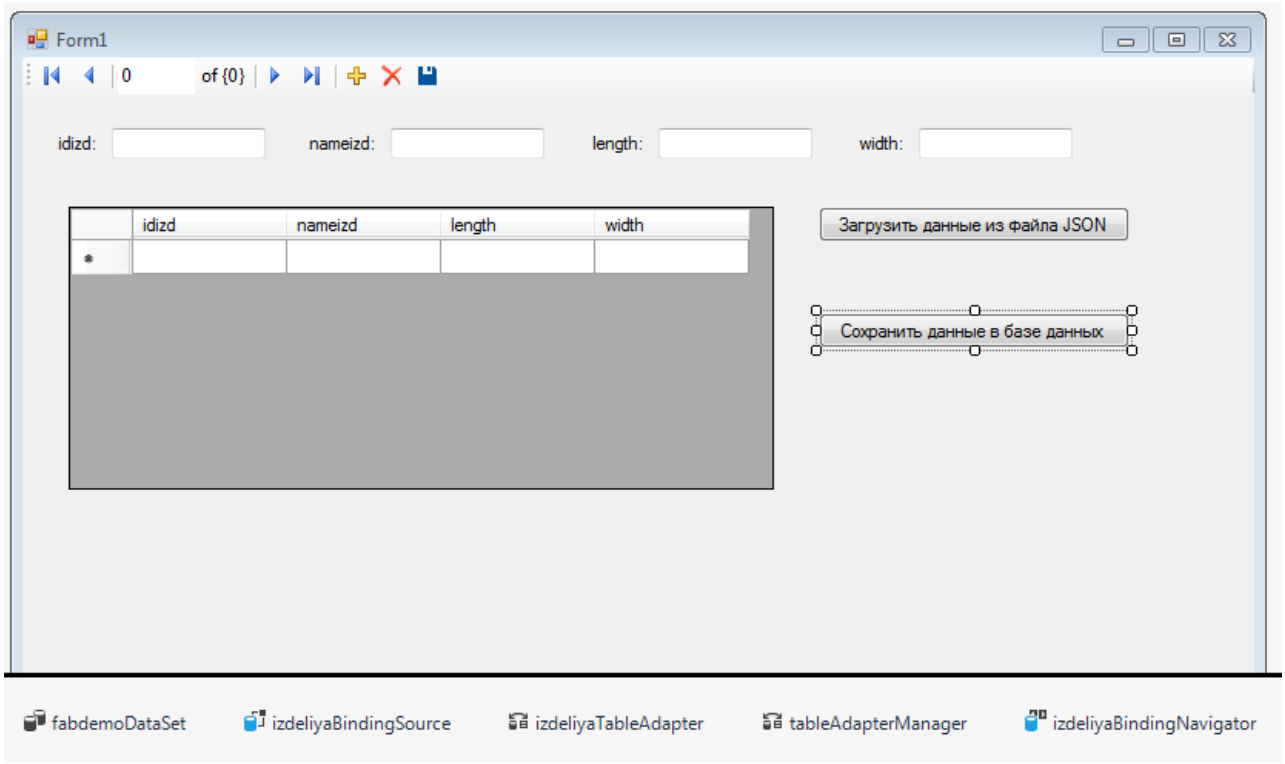
```
// класс для хранения одной записи из JSON-файла
public class ClassIzdeliya
{
    [JsonProperty("idizd")]
    public string ididizd { get; set; }
    [JsonProperty("nameizd")]
    public string nameizd { get; set; }
    [JsonProperty("length")]
    public string length { get; set; }
    [JsonProperty("width")]
    public string width { get; set; }
}
// список записей, прочитанных из JSON-файла
List<ClassIzdeliya> lst = new List<ClassIzdeliya>();
```

Обратите внимание, что поля класса должны быть доступны и для чтения и для записи. Кроме того, перед каждым полем должна быть директива JsonProperty, в которой должно быть указано имя поля в JSON-файле. По указанному имени компьютер будет определять какое поле из файла JSON в какое поле класса нужно поместить. Если вы допустите ошибки в именах полей в JsonProperty, невозможно будет прочитать данные из файла.

Шаг 8. В панели Data Sources разверните таблицу Izdeliya и отбуксируйте на форму с помощью мыши каждое поле таблицы Izdeliya по отдельности и всю таблицу Izdeliya целиком.



Должна получиться форма, имеющая следующий интерфейс:



Поставьте также на форму две кнопки: для загрузки данных из файла и для сохранения загруженных данных в таблице базы данных.

Шаг 9. Для кнопки «Загрузить данные» опишите программный код.

Вначале мы открываем JSON-файл как обычный текстовый файл. Затем с помощью метода ReadToEnd() мы считываем все содержимое JSON-файла, а с помощью метода JsonConvert.DeserializeObject() выполняем десериализацию. Содержимое JSON-файла преобразуется в список List, элементами которого являются экземпляры класса ClassIadeliya.

Затем мы выполняем перебор элементов этого списка. Для каждого элемента в таблицу базы данных добавляется новая запись, после чего в текстовые поля, связанные с базой данных, мы выводим информацию из i-го элемента списка lst.

Если программный код написан верно, то после его выполнения в таблице DataGridView на форме появятся записи для таблицы Izdeliya.

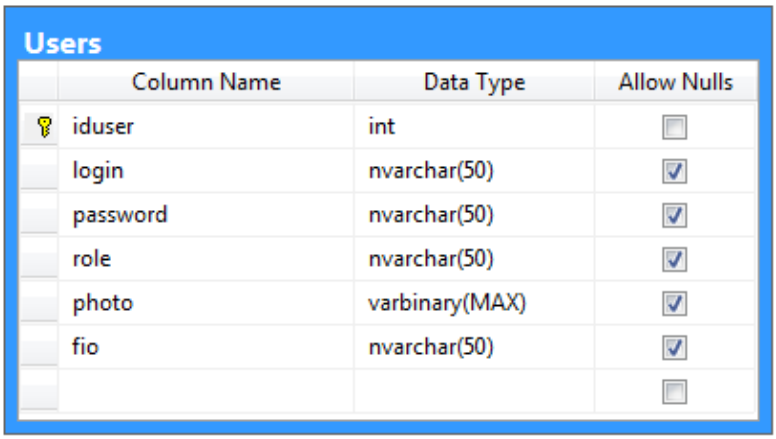
```
private void btnLoad_Click(object sender, EventArgs e)
{
    // открыть JSON файл для чтения
    StreamReader sr = new StreamReader(Application.StartupPath + "\\Изделия.json");
    // выполнить десериализацию
    // загрузить содержимое JSON файла в список lst
    lst = JsonConvert.DeserializeObject<List<ClassIzdeliya>>(sr.ReadToEnd());
    sr.Close();  // закрыть JSON-файл
    // выполнить перебор записей в списке
    for (int i = 0; i <= lst.Count - 1; i++)
    {
        // добавить новую запись в таблицу Изделия
        izdeliyaBindingSource.AddNew();
        // поместить в эту запись данные из i-го элемента списка
        ididizdTextBox.Text = lst[i].ididizd;
        nameizdTextBox.Text = lst[i].nameizd;
        lengthTextBox.Text = lst[i].length;
        widthTextBox.Text =  lst[i].width;
    }
}
```

Шаг 10. Для кнопки «Сохранить данные» опишите следующий код:

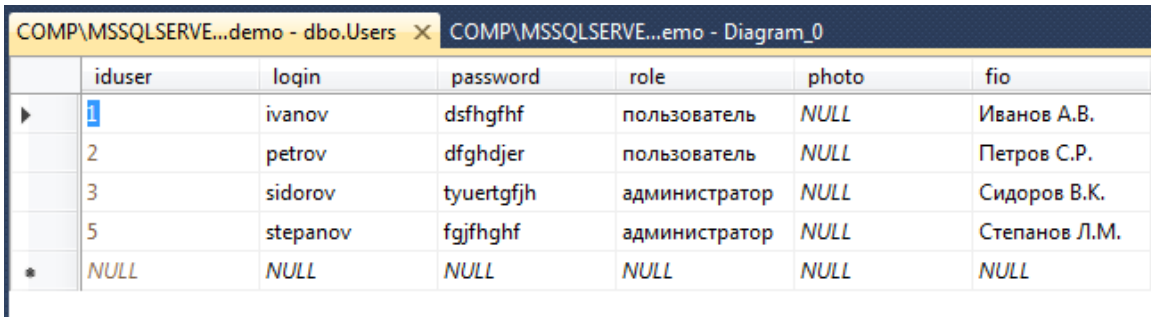
```
private void btnSave_Click(object sender, EventArgs e)
{
    // сохранить изменения в базе данных
    izdeliyaBindingSource.EndEdit();
    izdeliyaTableAdapter.Update(fabdemoDataSet.Izdeliya);
    MessageBox.Show("Изменения в базе данных сохранены.");
}
```

Тема 3. Импорт в базу данных из графических файлов

Пусть у нас имеется таблица Users, содержащая информацию о пользователях: логин, пароль, роль, фото, ФИО. Для хранения графических изображений в таблице должно быть поле типа **varbinary (MAX)**.



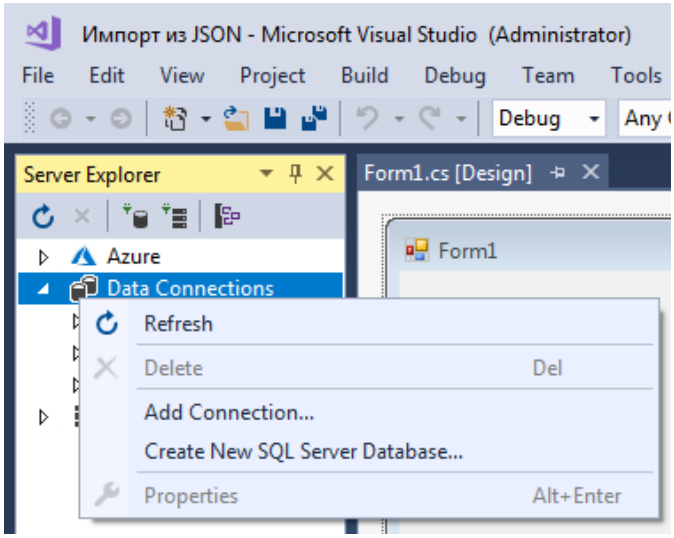
Пусть в этой таблице имеются следующие записи:



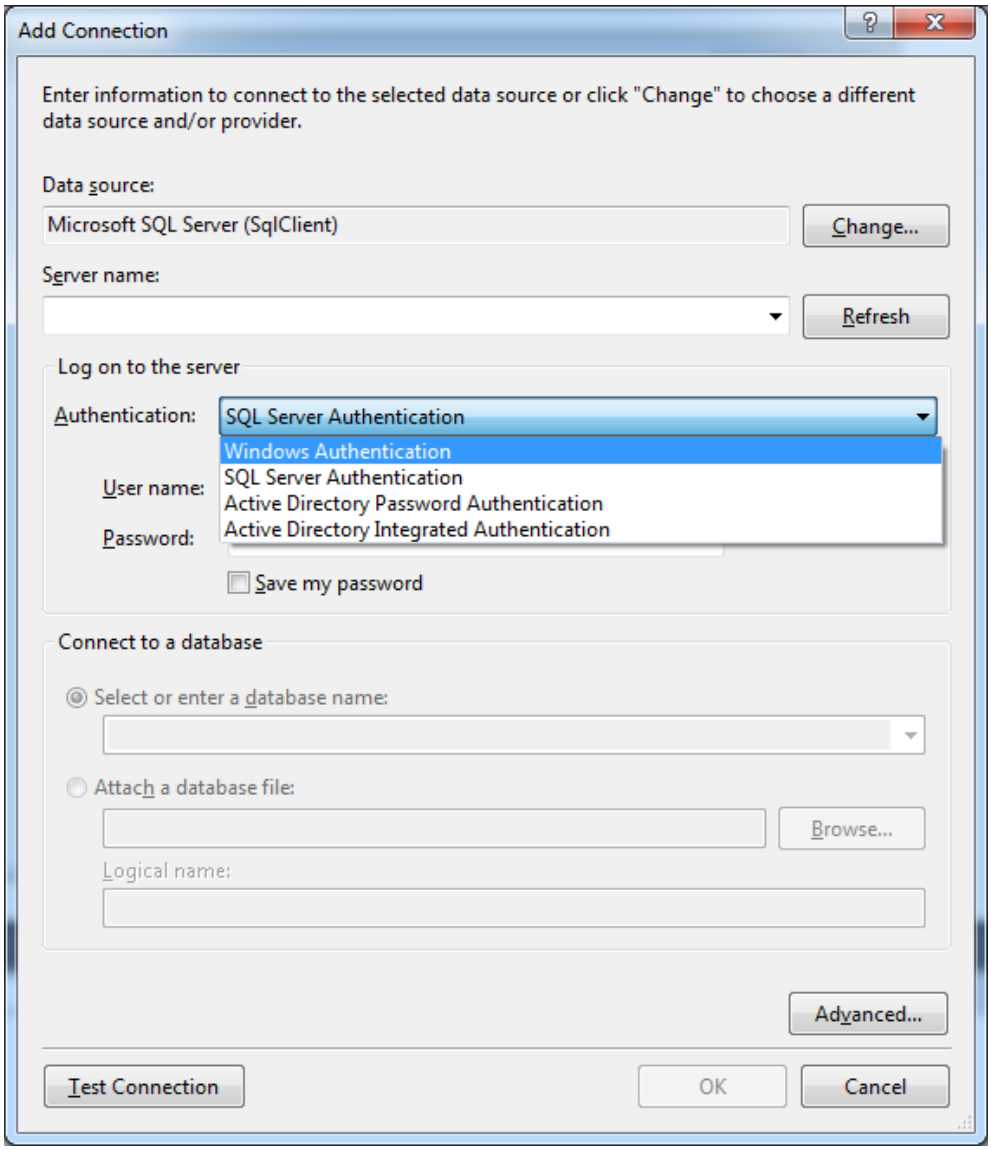
iduser	login	password	role	photo	fio
1	ivanov	dsfhgfhf	пользователь	NULL	Иванов А.В.
2	petrov	dfghdjer	пользователь	NULL	Петров С.Р.
3	sidorov	tyuertgfjh	администратор	NULL	Сидоров В.К.
5	stepanov	fgjfhghf	администратор	NULL	Степанов Л.М.
NULL	NULL	NULL	NULL	NULL	NULL

Для каждого пользователя у нас есть файлы, имена которых совпадают с логинами: ivanov.jpg, petrov.jpg, sidorov.jpg, stepanov.jpg. Нам нужно в поле photo загрузить графические изображения из этих файлов.

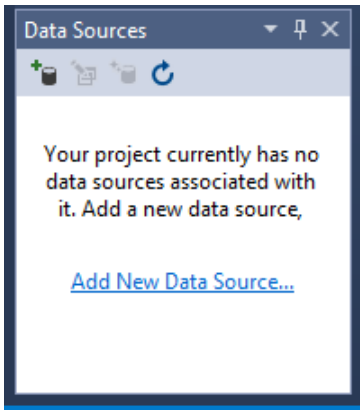
Шаг 1. Запустите Visual Studio. Создайте и сохраните новый проект Windows Forms. Для этого на панели Server Explorer выполните щелчок правой кнопкой мыши на узле Data Connections и выберите пункт Add Connection...



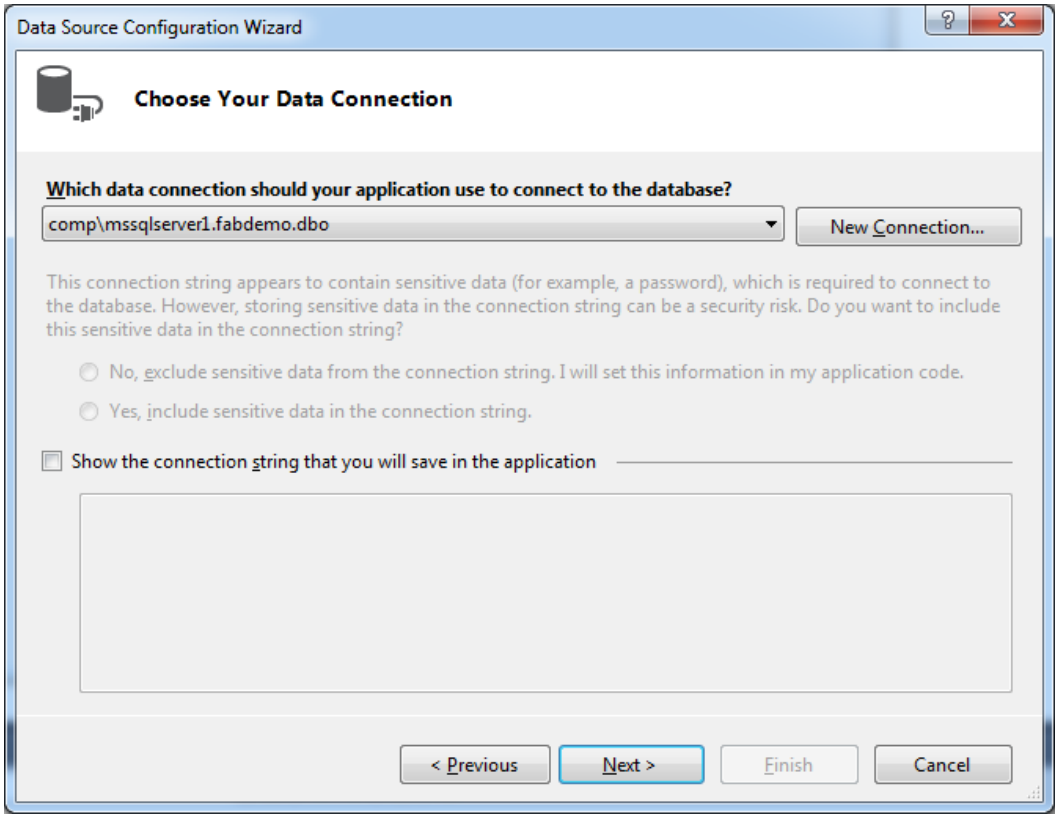
В открывшемся диалоговом окне выберите тип аутентификации. Если вы работаете с базой данных, которая хранится на вашем локальном компьютере, выберите «Windows Authentication». Если эта база данных хранится на SQL-сервере в локальной сети, выберите «SQL Server Authentication». Выберите из списка, или укажите имя сервера, введите при необходимости имя пользователя и пароль для доступа к серверу SQL. Выберите базу данных на сервере и нажмите OK.



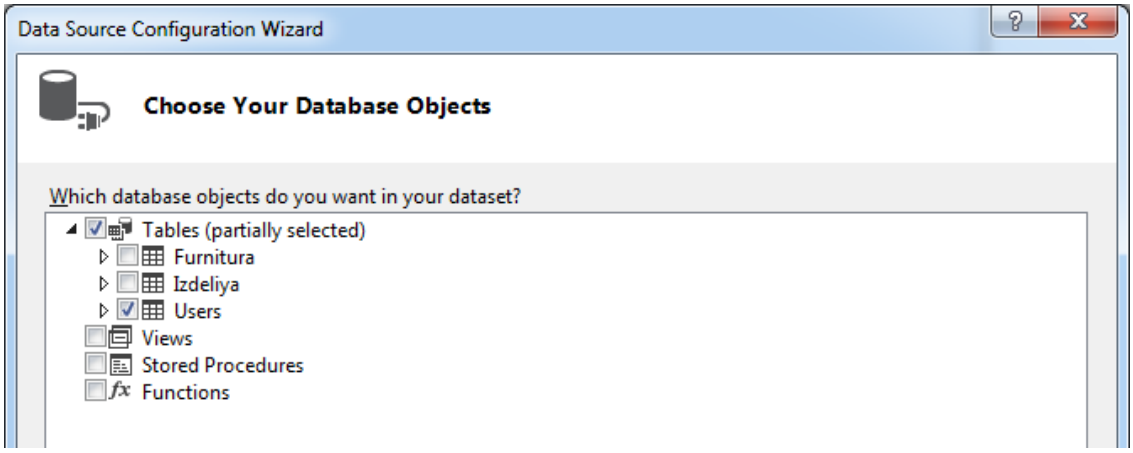
Шаг 4. В панели Data Source добавьте новый источник данных.



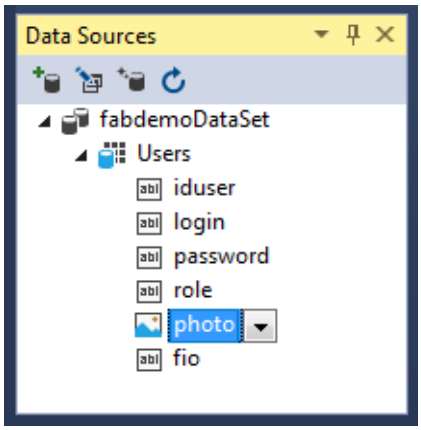
В процессе добавления источника данных выберите правильный Data Connections для подключения к вашей базе данных.



Обязательно выделите таблицу Users, чтобы она добавилась в источник данных.

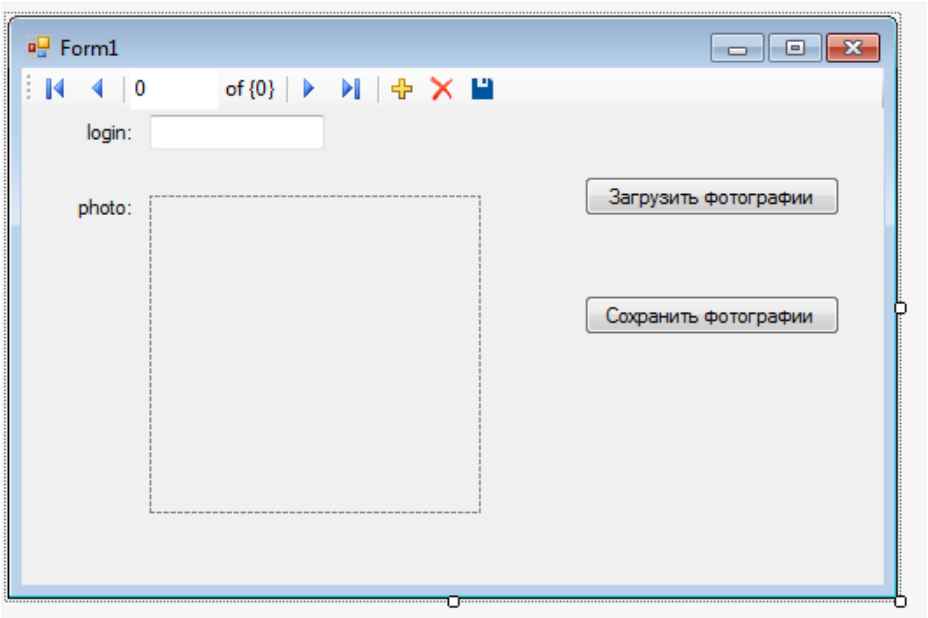


Шаг 2. В панели Data Sources разверните таблицу Users. Для поля photo откройте выпадающий список и выберите тип визуального объекта **PictureBox**.



Затем отбуксируйте на форму с помощью мыши из таблицы Users поле login и поле photo.

Должна получиться форма, имеющая следующий интерфейс:



Поставьте также на форму две кнопки: для загрузки фотографий из файла и для сохранения загруженных фотографий в таблице базы данных.

Поле login нам потребуется для того, чтобы из него можно было прочитать логин пользователя, который совпадает с именем файла с фотографией.

Шаг 3. Для PictureBox, в котором будет отображаться фотография, в свойстве SizeMode установите значение Zoom. Это нужно для того, чтобы большие изображение уменьшались в размерах и полностью отображались в PictureBox. Увеличьте также размер PictureBox по ширине и высоте.

Шаг 4. В корневом каталоге какого-либо диска создайте отдельную папку, в которую поместите фотографии пользователей.

Шаг 5. Опишите программный код для кнопки «Загрузить фотографии».

Вначале указатель устанавливается в начало таблицы Users.

Затем с помощью цикла for выполняется перебор записей в таблице.

С помощью метода Image.FromFile() выполняется загрузка графического изображения из файла в PictureBox. Имя файла состоит из трех частей: путь доступа к папке с изображениями, имя файла, которое берется из текстового поля loginTextBox, и расширения ".jpg". Т.к. PictureBox связан с полем таблицы базы данных, изображение из него попадает в базу данных.

После того, как изображение загружено, выполняем переход к следующей записи таблицы Users.

В таблице может быть много записей - несколько сотен или тысяч. В этом случае процесс загрузки изображений может оказаться слишком долгим. Чтобы компьютер не зависал при этом, используется команда Application.DoEvents(). Она после загрузки каждой фотографии на некоторое время передает управление операционной системе, чтобы другие приложения и сама операционная система могли выполнять свою работу.

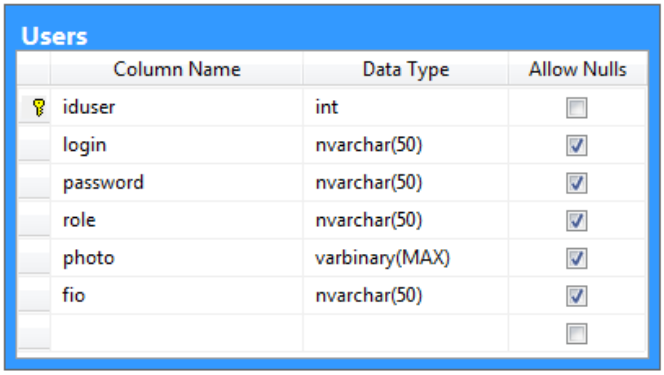
```
private void btnLoad_Click(object sender, EventArgs e)
{
    // поставить указатель в начало таблицы
    usersBindingSource.Position = 0;
    // перебор записей в таблице
    for (int i = 0; i <= usersBindingSource.Count - 1; i++)
    {
        try  // на тот случай, если изображение загрузить невозможно
        {
            // загрузить изображение из файла
            photoPictureBox.Image =
                Image.FromFile(@"C:\Фото пользователей\" + loginTextBox.Text.Trim()
                + ".jpg");
        }
        catch { }
        usersBindingSource.MoveNext(); // перейти к следующей записи

        Application.DoEvents(); // передать управление ОС
    }
    MessageBox.Show("Изображения загружены.");
}
```

Шаг 6. Для кнопки «Сохранить фотографии» опишите следующий код:

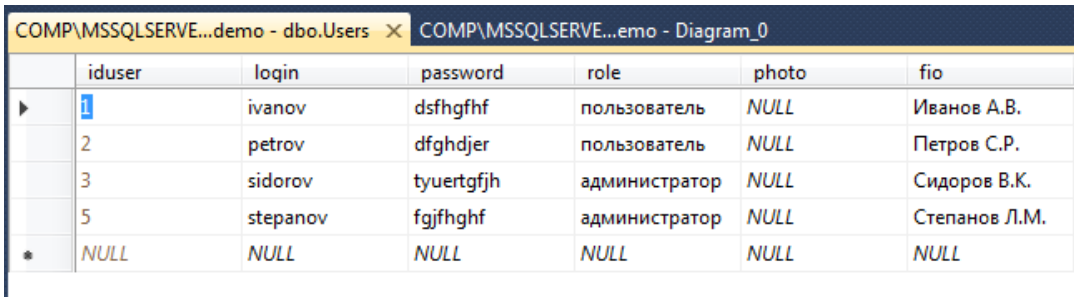
```
private void btnSave_Click(object sender, EventArgs e)
{
    // сохранить изменения в базе данных
    usersBindingSource.EndEdit();
    usersTableAdapter.Update(fabdemoDataSet.Users);
    MessageBox.Show("Изображения сохранены.");
}
```

Тема 4. Использование для поля таблицы только уникальных значений

Пусть у нас имеется таблица Users, содержащая информацию о пользователях: логин, пароль, роль, фото, ФИО.

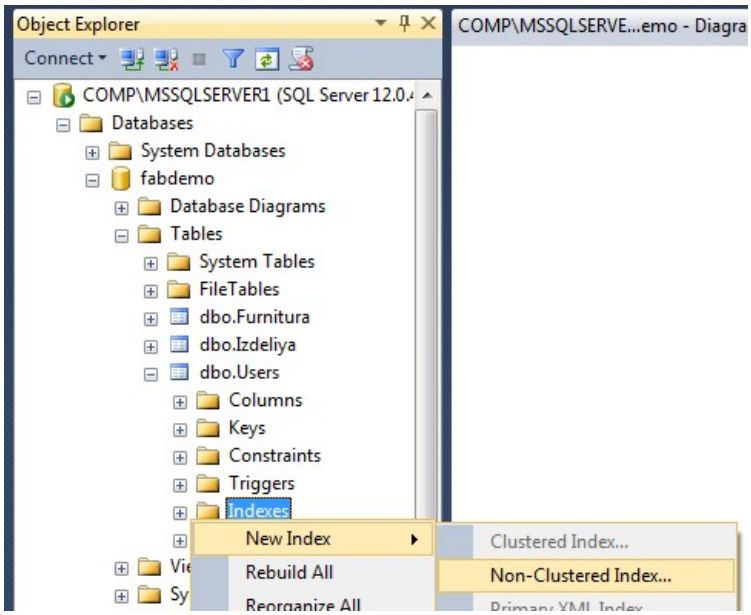


Пусть в этой таблице имеются следующие записи:

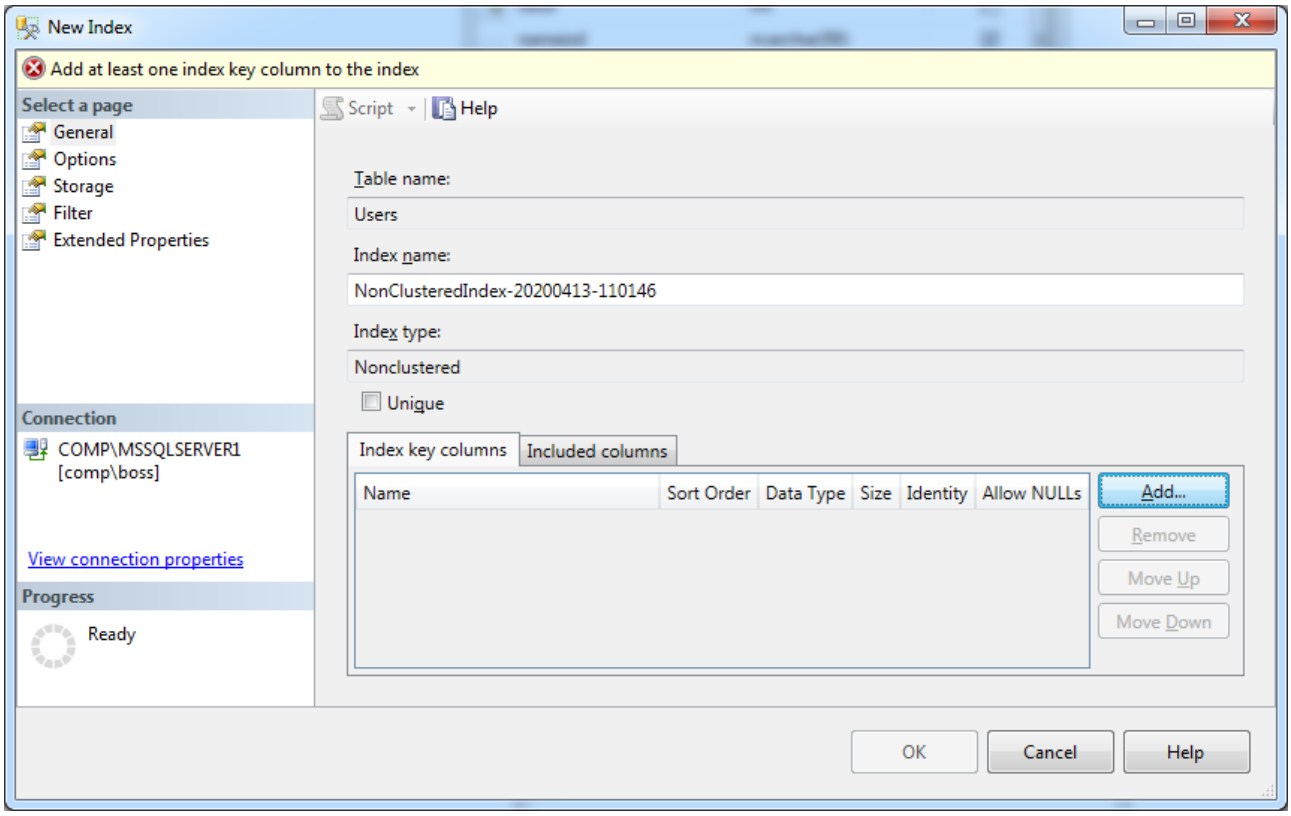


Логины пользователй должны быть уникальными. Не может быть нескольких пользователей с одинаковыми логинами. Рассмотрим, как это сделать средствами Microsoft SQL Server.

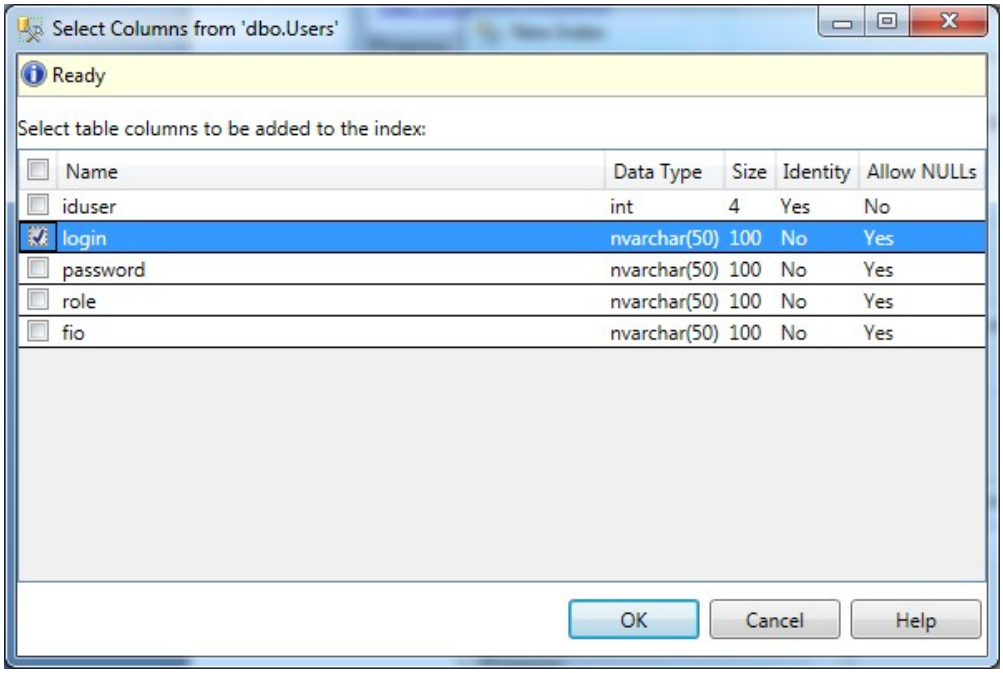
Шаг 1. В панели Object Explorer разверните вашу базу данных, разверните таблицу Users, щелкните правой кнопкой мыши на узле Indexes и выберите пункт New Index ► Non-Clustered Index...



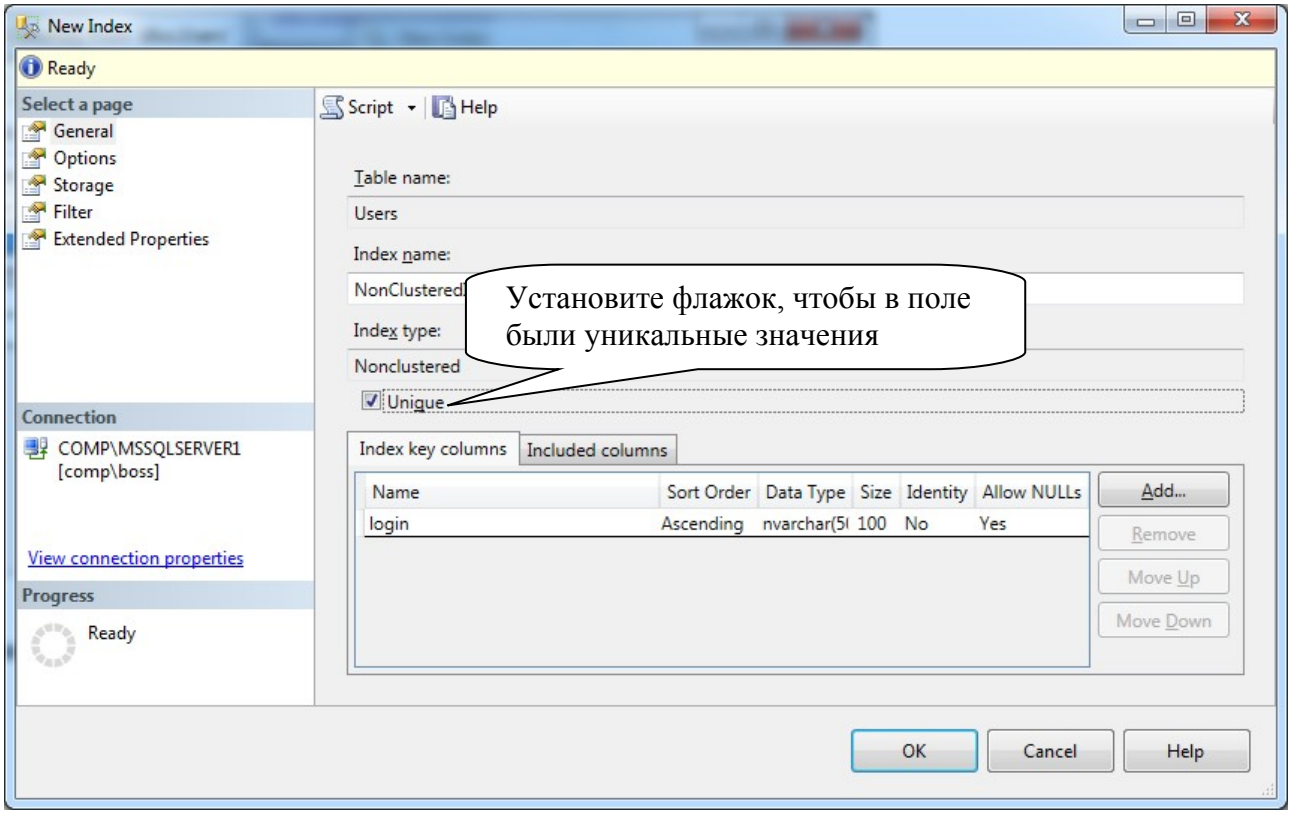
Шаг 2. В диалоговом окне New Index нажмите на кнопку Add.



Среди полей таблицы выделите поле login и нажмите на кнопку OK.



Шаг 3. В диалоговом окне New Index установите флажок Unique и нажмите OK.



Тема 5. Преобразование серверной базы данных в локальную базу данных и подключение ее к проекту

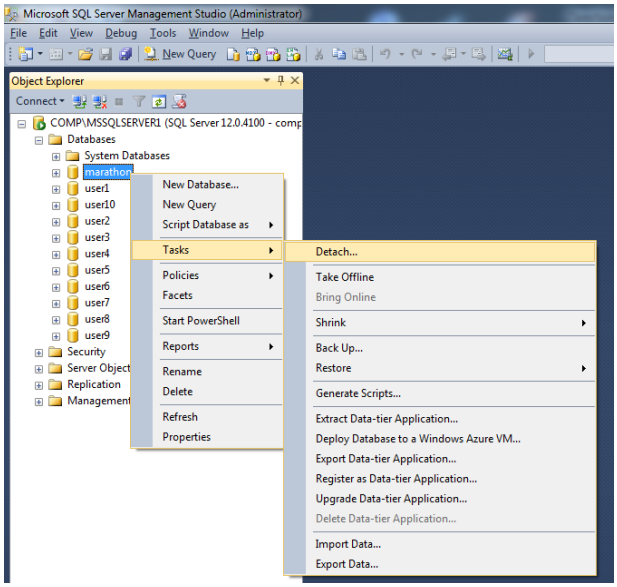
Пусть имеется проект, разработанный в Visual Studio, в котором используется база данных, расположенная на сервере SQL. В этом случае для работы проекта нужен постоянный доступ к SQL- серверу, на котором должна находится требуемая база данных.

Рассмотрим, как открепить базу данных от SQL-сервера и присоединить ее к проекту таким образом, чтобы база данных располагалась в папке с exe-файлом проекта. При этом должна быть возможность скопировать проект на любой другой компьютер. Если на другом компьютере будет установлен SQL Server LocalDB, то проект должен запускаться и работать со скопированной базой данных.

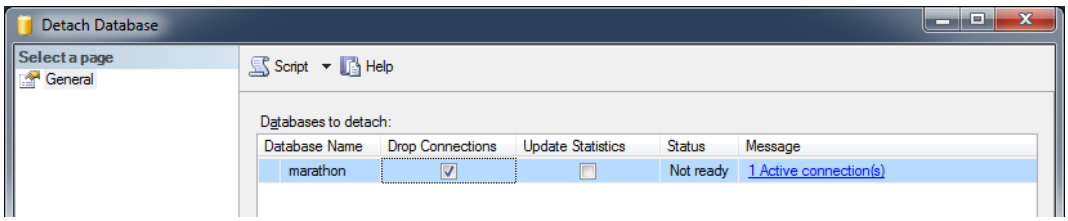
5.1. Открепление базы данных от SQL-сервера и подготовка файлов базы данных

Шаг 1. Запустите Microsoft SQL Server Management Studio, выполните авторизацию и открепите базу данных от SQL-сервера.

Выберите:



В появившемся окне установите флажок «Drop Connections», чтобы закрыть открытые подключения к базе данных, и нажмите кнопку ОК.



Шаг 2. Откройте папку, в которой SQL-сервер хранит базы данных. Обычно эта папка находится по следующему пути доступа:
c:\Program Files\Microsoft SQL Server\MSSQL<номер версии>.<название экземпляра SQL-сервера>\MSSQL\DATA

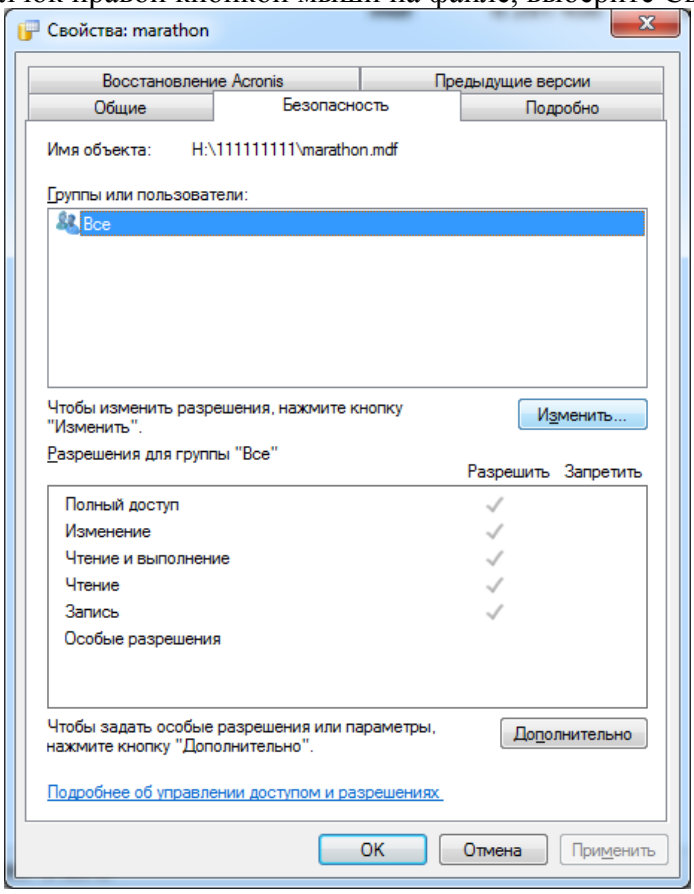
Если вы не можете найти папку с базами данных SQL-сервера, попробуйте выполнить поиск файлов на компьютере. Укажите для поиска имя базы данных, под которым она была в SQL-сервере.

К базе данных относятся два файла. Один имеет расширение **mdf** и содержит саму базу данных. Другой имеет расширение **ldf** и содержит информацию о фиксации транзакций и изменений базы данных.

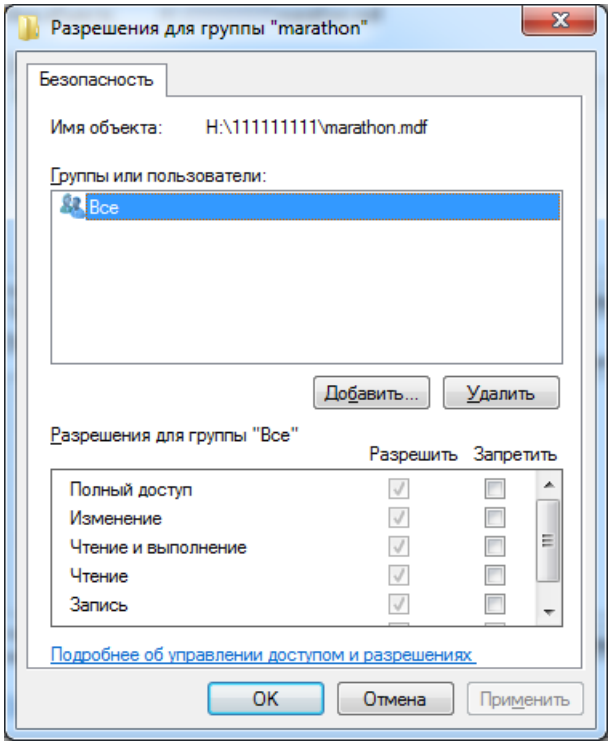
Например, **marathon.mdf** и **marathon_log.ldf**.
Скопируйте их в отдельный каталог.

Шаг 3. Установите **поочередно для обоих файлов** пользователя под именем «Все» и задайте для него полный доступ к файлам.

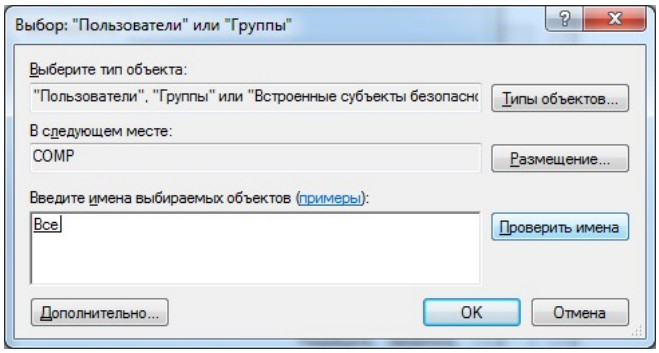
Выполните щелчок правой кнопкой мыши на файле, выберите Свойства.



На вкладке Безопасность нажмите кнопку Изменить.



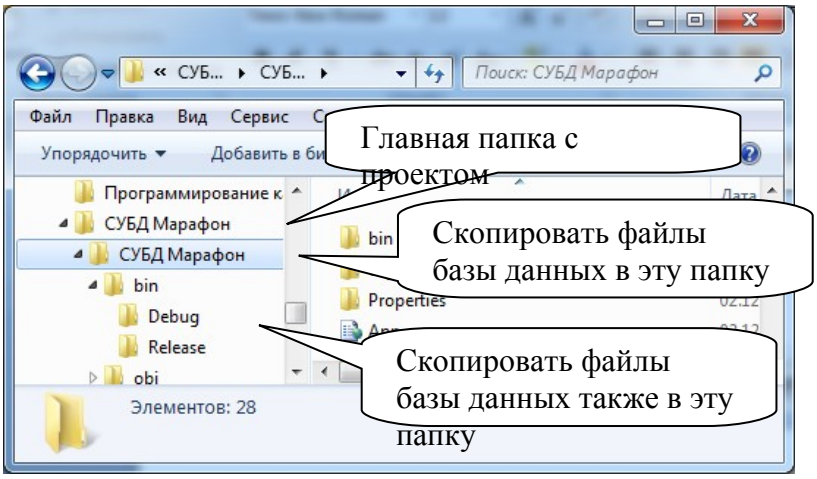
Если есть пользователь с именем «Все», то разрешите ему полный доступ. Если такого пользователя нет, нажмите на кнопку Добавить, введите имя пользователя «Все», нажмите кнопку Проверить и кнопку ОК. Разрешите этому пользователю полный доступ.



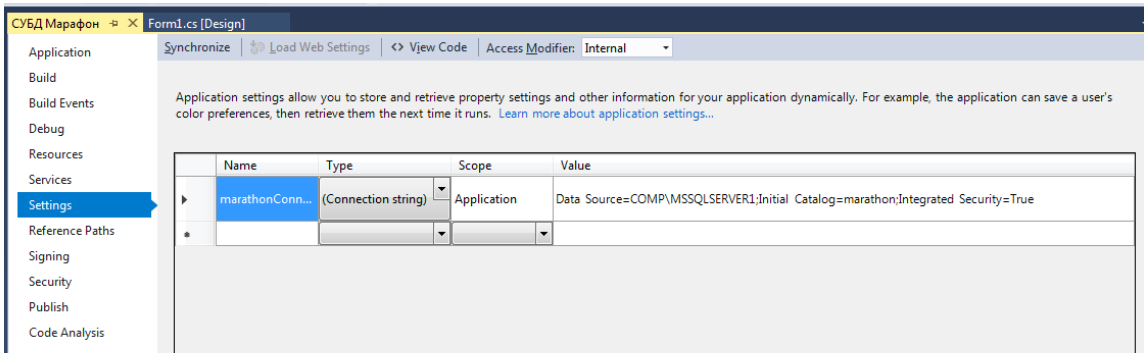
Внимание! Эти действия необходимо выполнить для обоих файлов, относящихся к базе данных.

5.2. Присоединение базы данных к проекту. Способ 1.

Шаг 1. Скопируйте оба файла базы данных в папку с проектом. Их нужно будет скопировать в два места: 1) в папке с проектом будет одноименный каталог, содержащий файлы с расширением cs, скопировать туда; 2) в папке с проектом будет подкаталог bin►Debug, в котором будет находиться exe-файл приложения, скопировать туда.



Шаг 2. Откройте ваш проект в Visual Studio. Выберите меню Project ► <Название проекта> Properties...



Выберите пункт Settings. В нем будет отображаться строка подключения к базе данных (Connection String). Замените ее на следующую строку:

Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|
DataDirectory|**marathon.mdf**;Integrated Security=True;Connect Timeout=30

Внимание! Вместо имени файла **marathon.mdf** поставьте имя **mdf** файла своей базы данных. Не допускайте лишних пробелов слева или справа от имени файла.

Шаг 3. Если вы в программном коде используете работу с базами данных с помощью SQL-запросов, то на главной форме у вас должна быть переменная txtcon, в которой находится строка подключения. Т.к. подключение к базе данных изменилось, эту переменную нужно будет обновить.

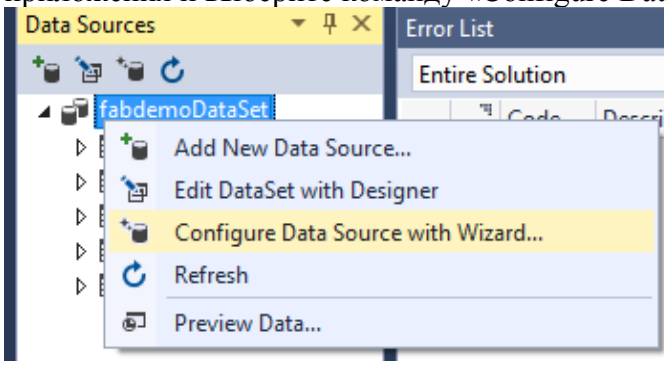
Переключитесь в программный код главной формы. После конструктора формы найдите строковую переменную txtcon и присвойте ей значение пустого текста.

```
public Form1()
{
    InitializeComponent();
}

// строка подключения к базе данных
public static string txtcon = @"";
```

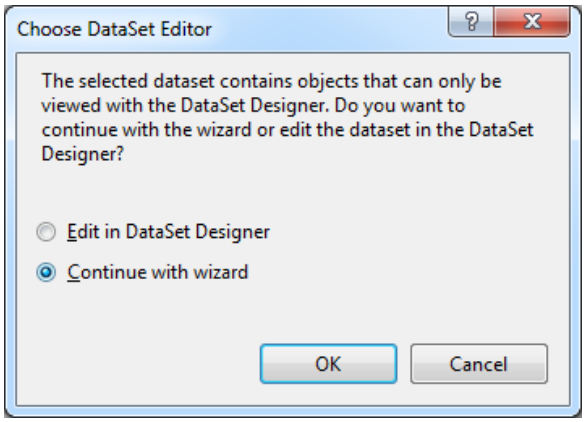
public означает, что к этой переменной у нас есть глобальный доступ

static означает, что это статическая переменная, и к ней можно обращаться прямо из класса формы (Form1.txtcon).

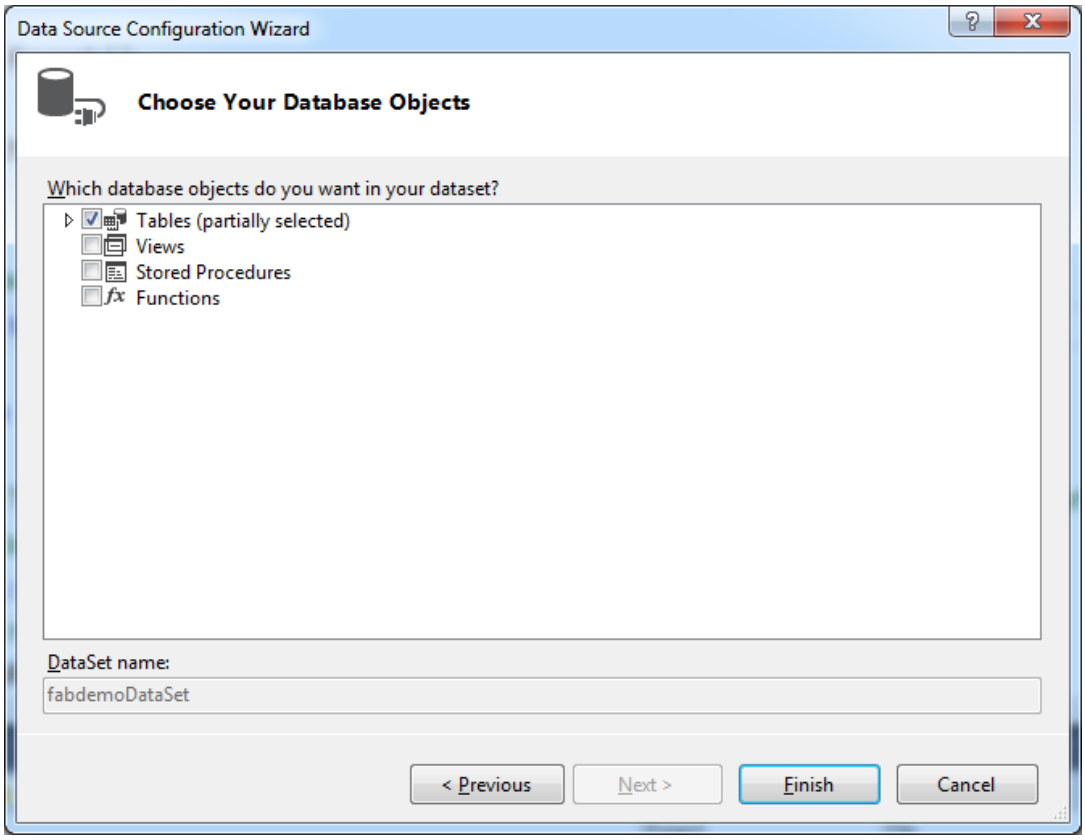
В панели Data Sources щелкните правой кнопкой на источнике данных для нашего приложения и выберите команду «Configure Data Source with Wizard...».



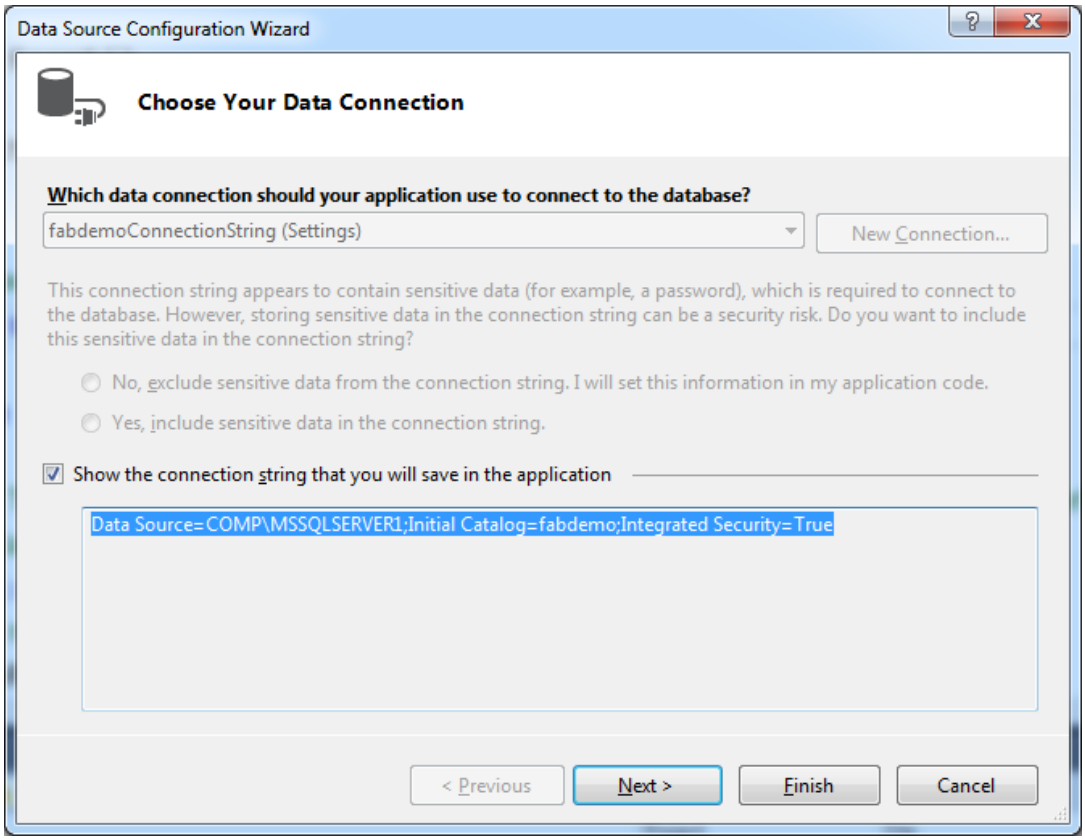
Выберите пункт «Continue with Wizard».



В диалоговом окне конфигурации нажмите на кнопку Previous.



Установите флажок «Show the connection string» и скопируйте в буфер строку подключения к базе данных.



Затем вставьте текст строки подключения в переменную txtcon внутри кавычек.

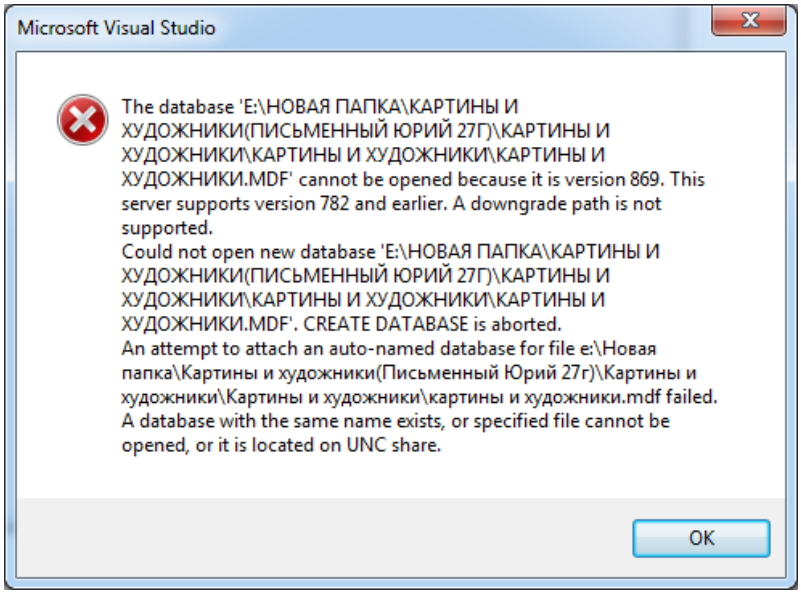
```
// строка подключения к базе данных
public static string txtcon = @"Data Source=COMP\MSSQLSERVER1;Initial
  Catalog=fabdemo;Integrated Security=True";
```

Внимание! Сначала в программном коде нужно поставить пустые кавычки, и только потом вставлять в них текст строки подключения. Если сделать наоборот, то редактор Visual Studio добавит в строку подключения лишние пробелы и она будет уже испорчена.

Шаг 4. Сохраните изменения своего проекта. Запустите его и проверьте взаимодействие приложения с базой данных.

5.3. Решение проблемы несовместимости версий базы данных. Присоединение базы данных к проекту. Способ 2.

Внимание! Если версия SQL сервера более новая, чем та, которая поддерживается вашей версией Visual Studio, то при попытке работать с локальной базой данных вы получите сообщение примерно такого содержания:



В этом сообщении говорится, что файл базы данных создан с использованием версии 869, а Visual Studio может работать только с версией не более 782.

Рассмотрим версии SQL Server, соответствующие определенным номерам:
782 - SQL Server 2014
852 - SQL Server 2016
869 - SQL Server 2017
895 - SQL Server 2019.

В случае подобной ошибки сделайте следующее:

1) найдите в интернете, скачайте и установите SqlLocalDB от той версии SQL-сервера, в которой сделана база данных;

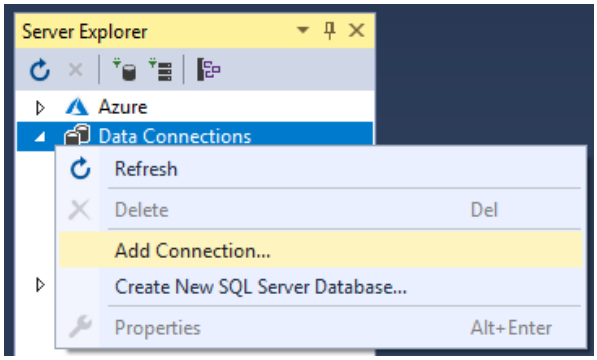
2) откройте консоль (Win+R, команда cmd) и последовательно выполните следующие команды:
sqllocaldb stop MSSQLLocalDB
sqllocaldb delete MSSQLLocalDB
sqllocaldb create MSSQLLocalDB
sqllocaldb start MSSQLLocalDB

Этими командами сначала останавливается работа MSSQLLocalDB, затем он удаляется, затем создается заново и запускается. MSSQLLocalDB используется для работы с локальными базами данных.

Предположим, вы заменили строку подключения Connection String так, как было описано в шаге 5. Но проект все равно не запускается и доступа к базе данных нет. В этом случае используйте другой способ подключения файла базы данных к проекту.

Шаг 1. В панели Server Explorer щелкните правой кнопкой мыши на узле Data Connections и выберите пункт Add Connections. Подключите файл базы данных к проекту.

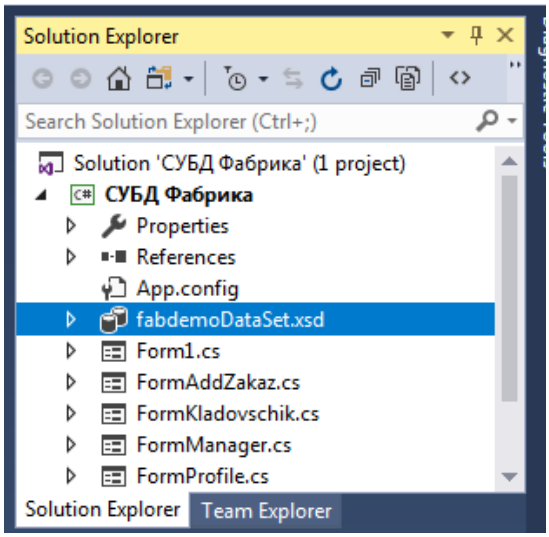
Внимание! База данных должна быть отсоединена от SQL-сервера. Если базу данных не отсоединить, то у вас не будет доступа к данным.



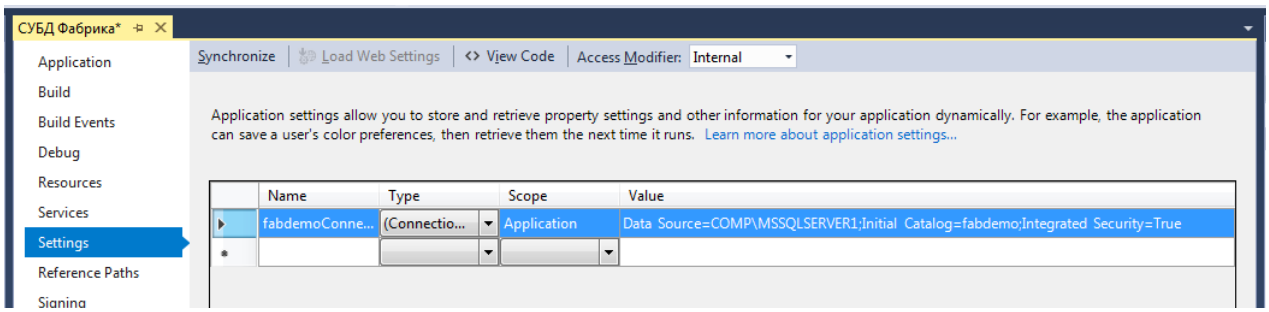
Шаг 2. Если у вас в источнике данных Data Sources есть вычисляемые поля в каких-либо таблицах, то скопируйте в отдельный текстовый документ имена этих полей, имена таблиц, в которых они находятся, и выражения для вычисления значений этих полей из свойства Expression.

Шаг 3. Закройте в окне Visual Studio все вкладки, на которых у вас открыты формы проекта.

Шаг 4. В панели Solution Explorer найдите источник данных - файл с расширением xsd. Удалите его.



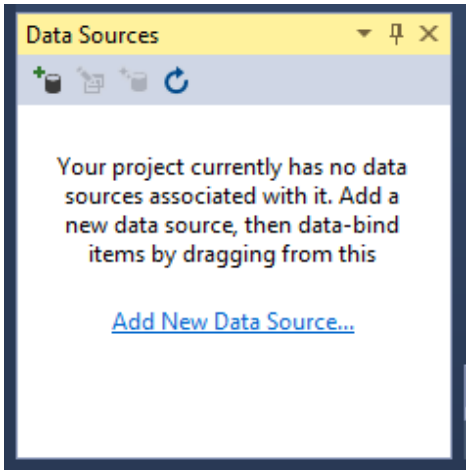
Шаг 5. Откройте меню Project▶Properties.



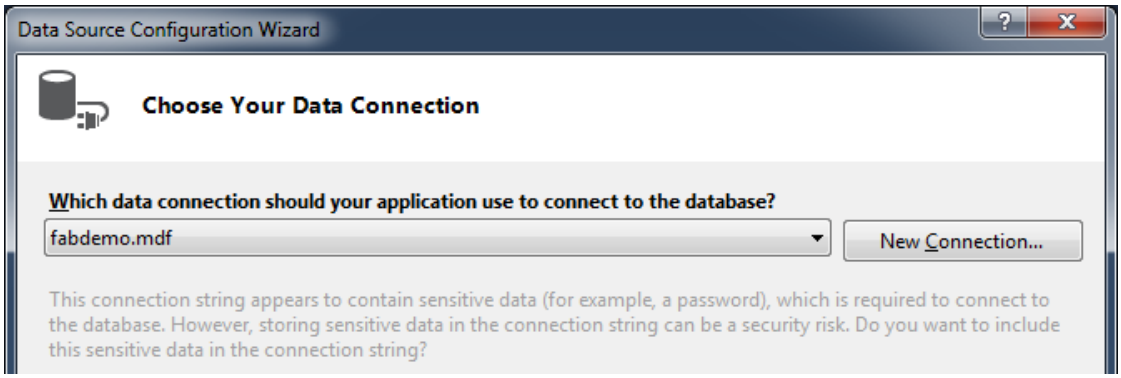
В разделе Settings выделите и удалите строку подключения к вашей прежней базе данных.

Сохраните изменения в проекте.

Шаг 6. На панели Data Sources нажмите Add New Data Sources и добавьте к проекту новый источник данных. В процессе добавления выберите файл вашей базы данных и подтвердите копирование базы данных в папку с проектом.



Внимание! В процессе добавления нового источника данных выбирайте именно файл базы данных с расширением mdf, а не подключение к SQL-серверу.



Шаг 7. Откройте источник данных Data Sources в режиме конструктора. Добавьте вычисляемые поля в те таблицы, в которых они должны быть. Используйте для вычисляемых полей те же самые имена и те же самые выражения Expression, которые у вас были изначально.

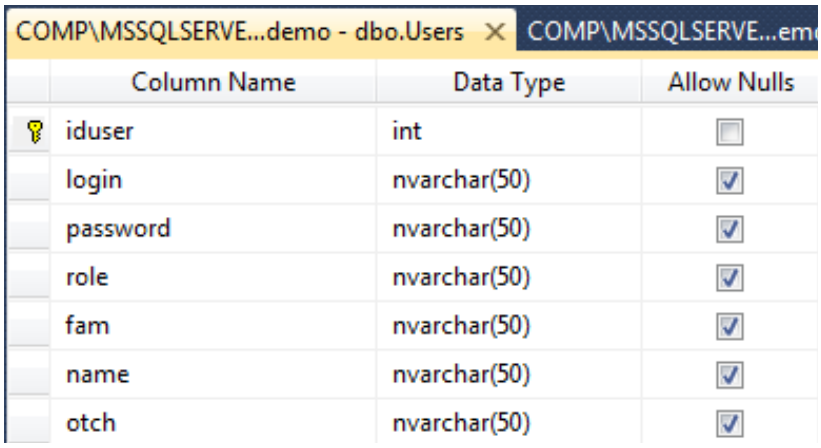
Для первичных ключей в каждой таблице базы данных в свойствах AutoIncrementSeed и AutoIncrementStep установите значение равное 1.

Шаг 8. Сохраните изменения, запустите проект и проверьте работу с базой данных.

Тема 6. Модуль авторизации пользователей с различными правами

6.1. Подготовка и подключение базы данных, настройка таблицы

Пусть у нас имеется таблица Users со следующими полями:

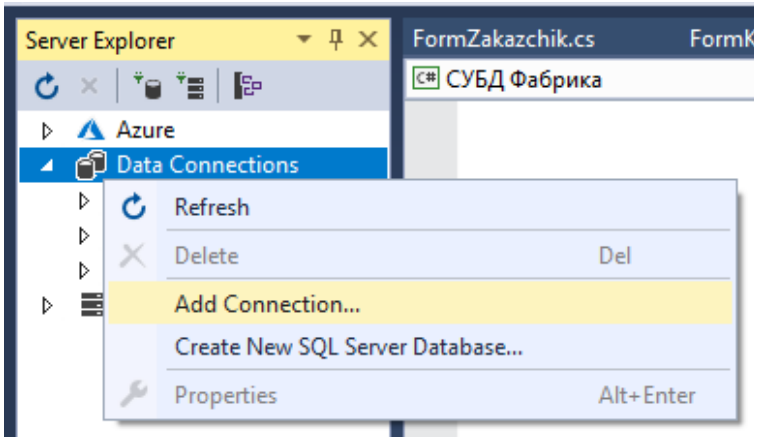


Поле login будет использоваться для хранения логинов пользователей. Значения этого поля должны быть уникальными. О том, как разрешить в поле сохранять только уникальные значения, читайте в теме №3.

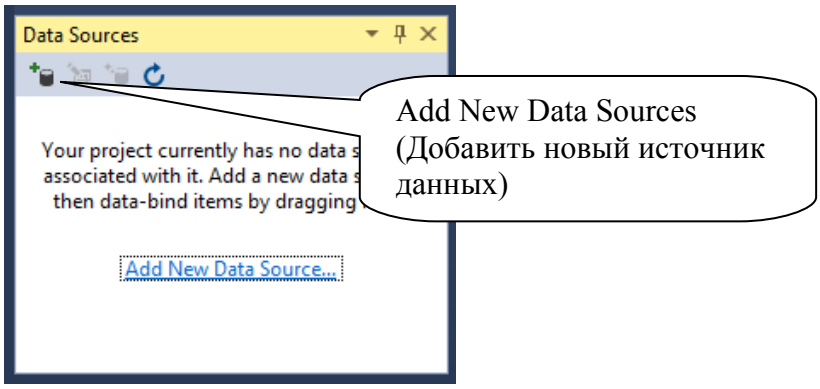
В поле role будет указываться роль пользователя. В нашем примере в этом поле будет указано: заказчик, менеджер или кладовщик.

Нам необходимо создать форму для авторизации пользователей. При успешной авторизации должна открываться форма - рабочее место соответствующего типа пользователя (заказчика, менеджера или кладовщика). При этом на форме рабочего места соответствующего типа пользователя должны отображаться фамилия, имя и отчество авторизовавшегося пользователя.

Шаг 1. Создайте новый проект. Подключите базу данных, содержащую таблицу Users, к проекту. Для этого воспользуйтесь панелью Server Explorer. Щелкните правой кнопкой на узле Data Connections и выберите пункт Add Connections...

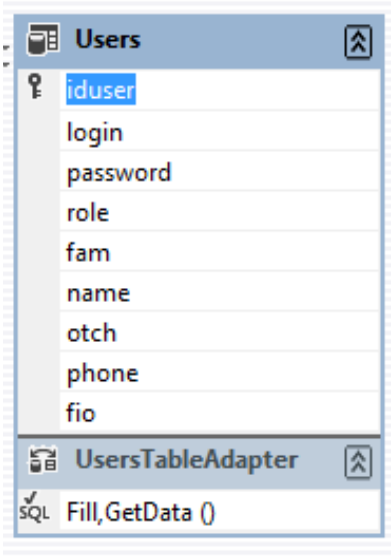


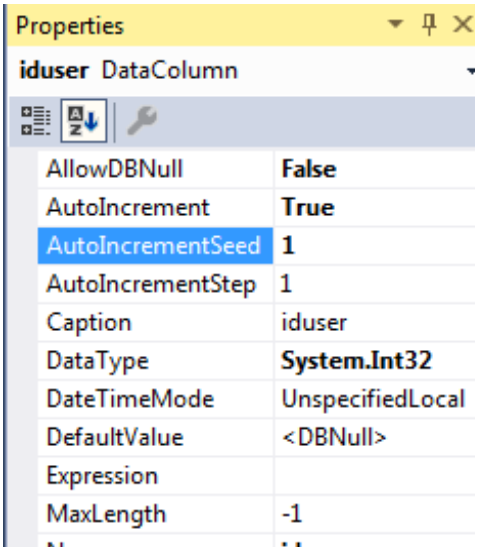
Шаг 2. В панели «Data Sources» нажмите кнопку «Add New Data Sources», или на ссылку с тем же названием.



Запустится мастер добавления новых источников данных. Добавьте Data Sources для нашей базы данных. Поместите в источник данных таблицу Users.

Шаг 3. Откройте Data Sources в режиме редактирования.

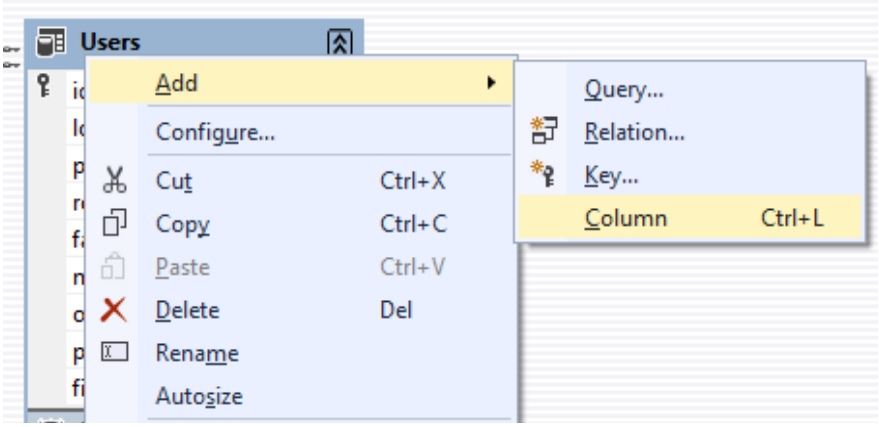




Для поля iduser в свойствах AutoIncrementSeed и AutoIncrementStep установите значение 1. Это начальное значение счетичка и шаг изменения счетчика.

Шаг 4. После успешной авторизации пользователя нам нужно выводить на форму фамилию, имя и отчество авторизовавшегося пользователя. Чтобы это можно было сделать, мы создадим вычисляемое поле, в котором будет храниться фамилия, имя и отчество пользователя.

Щелкните правой кнопкой мыши на заголовке таблицы Users, выберите пункт Add ► Column.

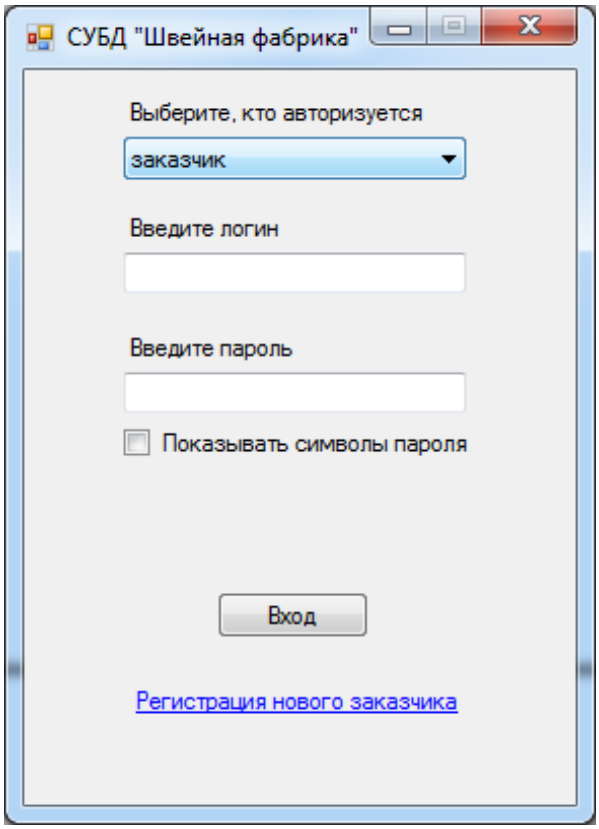


В таблицу Users добавится новое поле. В свойстве Name для него укажите fio, а в свойстве Expression напишите выражение: fam + ' ' + name + ' '+ otch

Сохраните изменения DataSources.

6.2. Формирование интерфейса формы для авторизации, создание и настройка форм для рабочих мест пользователей

Шаг 1. Интерфейс формы для авторизации может выглядеть следующим образом:



Для формы задайте свойства:
Text - СУБД "Швейная фабрика" (заголовок формы)
StartPosition - CenterScreen (для отображения формы по центру экрана)
MaximizeBox - False (заблокировать кнопку разворачивания формы до максимума)
FormBorderStyle - Fixed3D или FixedSingle (не разрешать изменять размеры формы)

Для выбора типа авторизующегося пользователя используйте ComboBox.
Name - cbxRole
Items - заказчик, менеджер, кладовщик (элементы списка)
DropDownStyle - DropDownList (не разрешать вводить в ComboBox текст с клавиатуры)

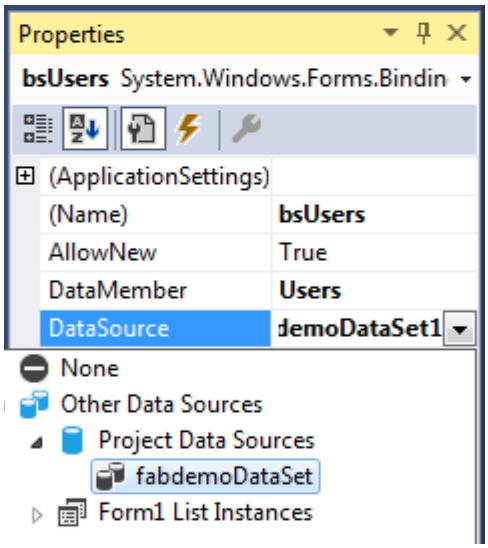
Для ввода логина и пароля используйте TextBox, дайте им имена tbxLogin, tbxPass. При вводе с клавиатуры символов пароля они не должны быть видны. Поэтому для tbxPass в свойстве UseSystemPasswordChar установите значение True.

Для отображения или скрытия символов пароля используйте CheckBox. Дайте ему имя cbxShowPass.

Для входа используйте кнопку Button. Дайте ей имя btnLogin.

Для регистрации нового заказчика будем использовать метку Label. Дайте ей имя lblReg. В свойствах ForeColor и Font измените цвет текста и задайте подчеркивание для текста.

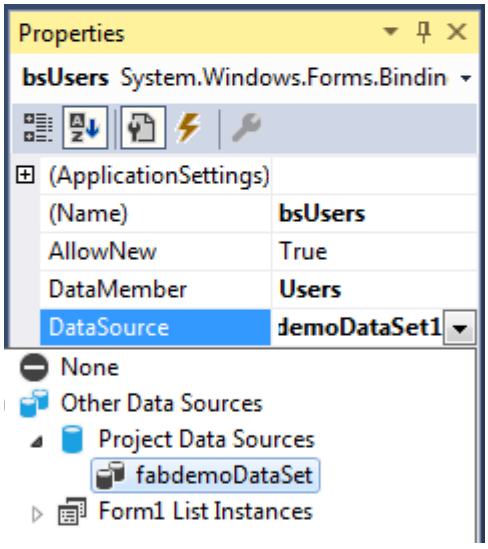
Шаг 2. Добавьте на главную форму DataSet и BindingSources. Для BindingSources задайте имя bsUsers и свяжите его с таблицей Users. В свойстве DataSources выберите источник данных, а в свойстве DataMember выберите таблицу Users.



Шаг 3. Добавьте к проекту три дополнительных формы. Дайте им имена: FormZakazchik, FormManager, FormKladovschik. Это будут формы - рабочие места пользователей соответствующих типов.

Для добавления новой формы к проекту используйте меню Project ► Add Windows Form...

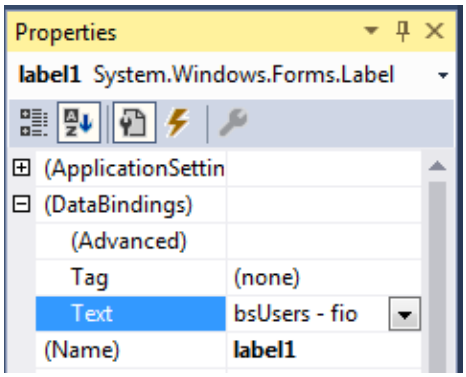
Шаг 4. На каждую из этих форм FormZakazchik, FormManager, FormKladovschik добавьте DataSet и BindingSources. Для BindingSources задайте имя bsUsers и свяжите его с таблицей Users. В свойстве DataSources выберите источник данных, а в свойстве DataMember выберите таблицу Users.



К bsUsers нужно будет обращаться с главной формы. Поэтому на каждой из трех форм - рабочих мест пользователей для bsUsers в свойстве Modifiers установите значение Public.

Шаг 5. В верхний левый угол каждой формы FormZakazchik, FormManager, FormKladovschik установите метку Label. Свяжите ее с полем fio таблицы Users. Поле fio - это вычисляемое поле, в котором у нас хранится фамилия, имя и отчество пользователя.

Чтобы связать Label с полем базы данных, выделите Label и в свойстве Data Bindings ► Text выберите поле fio из bsUsers.



6.3. Описание программного кода для авторизации

Шаг 1. В обработчике события Load для главной формы добавьте команду, которая при запуске приложения выбирает по умолчанию нулевой элемент списка типов пользователей:

```
// по умолчанию выбрать в списке заказчика
cbxRole.SelectedIndex = 0;
```

Шаг 2. Чтобы скрыть или показать символы пароля, в обработчике события CheckedChanged для cbxShowPass укажите следующий код:

```
private void cbxShowPass_CheckedChanged(object sender, EventArgs e)
{
    // включить / выключить показ пароля
    tbxPass.UseSystemPasswordChar = !cbxShowPass.Checked;
}
```

Шаг 3. Опишите программный код для кнопки «Вход».

```
private void btnLogin_Click(object sender, EventArgs e)
{
    // роль пользователя (считывается из ComboBox)
    string role = cbxRole.SelectedItem.ToString();
    // фильтр для таблицы Пользователи
    string txtfilter = String.Format("login = '{0}' and password = '{1}' and
      role = '{2}'", tbxLogin.Text, tbxPass.Text, role);
    // фильтр для рабочего места пользователя
    string filterworkplace = String.Format("login = '{0}'", tbxLogin.Text);

    bsUsers.Filter = txtfilter;  // включить фильтр

    if (bsUsers.Count == 0)
    {
        MessageBox.Show(String.Format("Нет пользователя '{0}' с указанным
          логином и паролем!", role));
        return;
    }
```

В переменную role мы записываем тип авторизующегося пользователя (заказчик, менеджер или кладовщик). Затем в переменную txtfilter мы записываем фильтр для поиска пользователя в таблице Users. Это фильтр будет выглядеть следующим образом:

login = 'логин из tbxLogin' and password = 'пароль из tbxPass' and role = 'тип пользователя из ComboBox'

Этот фильтр записывается в свойство Filter для bsUsers. Если в таблице Users есть пользователь с таким логином, паролем и типом, то в результате фильтрации будет отобрана одна запись. Если такого пользователя нет, то количество отфильтрованных записей будет равно нулю.

Если пользователя с указанным логином и паролем не найдено, то выводится сообщение и происходит выход из процедуры.

В переменную filterworkplace записывается фильтр для передачи на рабочее место пользователя. Здесь для фильтрации пользователей используется только логин.

Если же в базе данных был найден пользователь с указанным логином и паролем, то выполняется дальнейший программный код:

```
tbxLogin.Clear(); tbxPass.Clear(); // очистить поля
this.Visible = false;  // скрыть форму для авторизации

if (role == "заказчик")
{
    // создать форму для заказчика
    FormZakazchik frm = new FormZakazchik();
    frm.bsUsers.Filter = filterworkplace; // передать на форму фильтр
    frm.ShowDialog();
}

if (role == "менеджер")
{
    // создать форму для менеджера
    FormManager frm = new FormManager();
    frm.bsUsers.Filter = filterworkplace; // передать на форму фильтр
    frm.ShowDialog();
}
if (role == "кладовщик")
{
    // создать форму для кладовщика
    FormKladovschik frm = new FormKladovschik();
    frm.bsUsers.Filter = filterworkplace; // передать на форму фильтр
    frm.ShowDialog();
}
```

Здесь вначале очищаются текстовые поля для ввода логина и пароля. Это нужно для того, чтобы пользователь, который закроет свое рабочее место и вернется на форму для авторизации был вынужден вводить логин и пароль для авторизации заново.

Перед тем, как откроется рабочее место пользователя, форма для авторизации скрывается.

Затем выполняется проверка типа пользователя, который авторизуется. В зависимости от того, пользователь какого типа авторизуется, создается форма FormZakazchik, FormManager или FormKladovschik. На форму пользователя передается фильтр для поиска авторизовавшегося пользователя в базе данных. Этот фильтр записывается в свойство Filter для bsUsers на форме пользователя. Далее открывается форма рабочего места пользователя.

После того, как пользователь закроет форму, нужно сделать видимой форму для авторизации, установить курсор в поле для ввода логина и загрузить обновленные записи из таблицы Users. Пользователи будут иметь возможность изменять свой профиль. Следовательно, у пользователя может измениться пароль. Поэтому нужна загрузка обновленнх записей из базы данных.

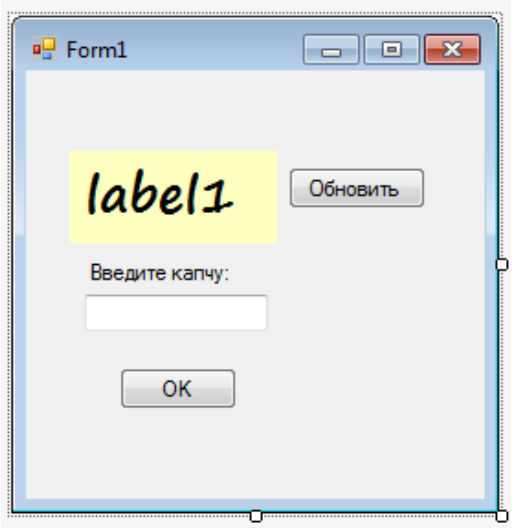
```
        this.Visible = true;  // показать форму для авторизации
        tbxLogin.Focus(); // поставить курсор
        // загрузить обновленные записи из таблицы Пользователи
        this.usersTableAdapter.Fill(this.fabdemoDataSet1.Users);

    }
```

Шаг 4. Запустите приложение и попробуйте авторизоваться под учетными записями пользователей различных типов.

Тема 7. Модуль создания и проверки капчи

Рассмотрим пример создания и проверки капчи. Капча должна состоять из четырех символов, определяемых случайным образом. Символы могут быть большими буквами латинского алфавита, либо цифрами. При этом обязательно должна быть хотя бы одна буква и хотя бы одна цифра. Капча должна отображаться нестандартным шрифтом, затрудняющим распознавание капчи. Поверх капчи нужно вывести несколько линий случайного цвета со случайными координатами.

Шаг 1. Сформируйте интерфейс формы следующим образом:



Для отображения капчи поставьте метку Label. Задайте для нее следующие свойства:
Name - lblCapcha
AutoSize - False (чтобы можно было менять размеры)
BackColor - фоновый цвет, отличный от фонового цвета формы
Font - шрифт (выберите нестандартный, но разборчивый шрифт, например, Segoe Print)

Для кнопок Button Обновить и OK задайте имена btnRefresh и btnOK. Для текстового поля, в которое пользователь вводит капчу, задайте имя tbxCapcha.

Шаг 2. В классе Form1 опишите программный код функции GetCapcha (). С помощью этой функции мы будем создавать капчу и возвращать ее в качестве результата.

```
string GetCapcha()
{
    string t1 = "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789";
    int cntAlpha, cntNum;
    string capcha;
    Random rnd = new Random();

    do
    {
        // капча, счетчики букв и цифр
        capcha = ""; cntAlpha = 0; cntNum = 0;
```

В переменной t1 записаны символы, которые используются для создания капчи. cntAlpha - это счетчик букв в капче, cntNum - счетчик цифр. В переменной capcha у нас будут записываться символы капчи.

```
        // в капче должно быть 4 символа
        for (int i = 1; i <= 4; i++)
        {
            // взять символ из строки с символами для капчи
            char sim = t1[rnd.Next(t1.Length)];
            // если символ - цифра, увеличить счетчик цифр
            if (char.IsDigit(sim)) cntNum++;
            // если символ - буква, увеличить счетчик букв
            if (char.IsLetter(sim)) cntAlpha++;
            capcha += sim;  // добавить символ к капче
        }
        // повторять до тех пор, пока в капче не будет хотя бы одной цифры
        // и хотя бы одной буквы
    } while (cntAlpha == 0 || cntNum == 0);

    return capcha;  // возвратить результат из функции
}
```

Так как по условию в капче должно быть 4 символа, будем использовать цикл for. В цикле for мы из строки t1 берем случайный символ. Если этот символ является цифрой, то увеличиваем счетчик цифр, если этот символ - буква, то увеличиваем счетчик букв. Этот символ добавляется к строке с капчей.

По условию задачи сказано, что в капче должна быть хотя бы одна буква и хотя бы одна цифра. Мы будем формировать капчу до тех пор, пока один из счетчиков cntAlpha или cntNum остается равным нулю.

После того, как капча сформирована, возвратим ее в качестве результата.

Шаг 3. В обработчике события Load вызовите функцию GetCapcha(), чтобы при запуске приложения символы капчи отобразились на форме.

```
private void Form1_Load(object sender, EventArgs e)
{
    // вывести капчу в метку на форме
    lblCapcha.Text = GetCapcha();
}
```

Для кнопки Обновить также поставьте команду вызова функции GetCapcha().

```
private void btnRefresh_Click(object sender, EventArgs e)
{
    // вывести капчу в метку на форме
    lblCapcha.Text = GetCapcha();
}
```

Шаг 4. По условию нам нужно поверх капчи вывести несколько линий случайного цвета со случайными координатами. Когда операционная система выводит изображение метки Label на форму, наступает событие Paint. Если нужно что-то изобразить на метке дополнительно, то программный код для этого записывается в обработчике события Paint.

```
private void lblCapcha_Paint(object sender, PaintEventArgs e)
{
    // массив цветов
    Color[] colors = {Color.Green, Color.Black, Color.Yellow, Color.White};
    Random rnd = new Random();
    // та, метка, которая сейчас изображается на форме
    // (для которой наступило событие Paint)
    Label lbl = (sender as Label);
    // изобразить от 5 до 10 линий
    for (int i = 1; i <= rnd.Next(6) + 5; i++)
    {
        // координаты начала и конца отрезка
        int x1 = rnd.Next(lbl.Width);
        int y1 = rnd.Next(lbl.Height);
        int x2 = rnd.Next(lbl.Width);
        int y2 = rnd.Next(lbl.Height);
        // случайный цвет из массива цветов
        Color col = colors[rnd.Next(colors.Length)];
        // нарисовать линию на метке Label
        e.Graphics.DrawLine(new Pen(col), x1, y1, x2, y2);
    }
}
```

Вначале мы создаем массив, содержащий различные цвета из перечисления Color. Затем создается датчик случайных чисел. sender - это объект, который вызвал событие, т.е. наша метка. Но так как sender имеет тип Object, нужно выполнить приведение к типу Label. Нужно сообщить компьютеру, что sender является именно меткой. В переменную lbl мы записываем ту метку Label, в которой будем рисовать линии. Количество линий, которые мы будем рисовать, определяется случайным образом. Мы будем рисовать от 5 до 10 линий. В цикле for с помощью датчика случайных чисел мы определяем координаты начала и конца отрезка. Координата по оси OX берется из диапазона от 0 до ширины метки. Координата по оси OY берется из диапазона от 0 до высоты метки. В переменную col записывается один из цветов из массива colors. Цвет выбирается случайным образом.

С помощью e.Graphics.DrawLine() изображается линия на метке Label. Первый параметр - цвет карандаша, которым рисуется линия, далее идут координаты начала и конца отрезка.

Шаг 5. По нажатию на кнопку OK нам нужно проверить символы капчи, которые ввел пользователь.

```
private void btnOK_Click(object sender, EventArgs e)
{
    // если символы, введенные пользователем, не совпадают с символами капчи
    if (tbxCapcha.Text != lblCapcha.Text)
    {
        MessageBox.Show("Символы капчи введены неверно.\nПопробуйте еще раз.");
        // обновить капчу на форме
        lblCapcha.Text = GetCapcha();
        return;  // выйти из процедуры
    }

    MessageBox.Show("OK");  // если капча введена верно, вывести сообщение
}
```

Правильные символы капчи хранятся в свойстве Text метки lblCapcha. Если они не совпадают с символами, указанными пользователем, то нужно вывести сообщение, обновить капчу и выйти из процедуры.

Если же символы, указанные пользователем оказались верными, то выполняется дальнейший программный код в обработчике Click для кнопки OK. В нашем примере это вывод сообщения OK. Однако здесь может быть и другой программный код. Например, открытие какой-либо другой формы и т.п.

Тема 8. Модуль регистрации пользователей

Пусть в нашей базе данных имеется таблица Users со следующими полями:

COMP\MSSQLSERVE...demo - dbo.Users | COMP\MSSQLSERVE...emo

Column Name	Data Type	Allow Nulls
iduser	int	☐
login	nvarchar(50)	☑
password	nvarchar(50)	☑
role	nvarchar(50)	☑
fam	nvarchar(50)	☑
name	nvarchar(50)	☑
otch	nvarchar(50)	☑
phone	nvarchar(50)	☑
photo	varbinary(MAX)	☑

Для хранения графических изображений в таблице должно быть поле типа **varbinary (MAX)**. Поле login должно хранить уникальные значения без повторов. О том, как настроить поле для хранения уникальных значений, читайте в теме №3 этого учебного пособия.

Необходимо создать форму и описать программный код для регистрации пользователей. В процессе регистрации предусмотреть загрузку фотографии пользователя из файла.

Логин пользователя должен быть уникальным. Пароль должен отвечать следующим требованиям безопасности:

- длина пароля – минимум 6 символов;
- обязательно хотя бы одна буква и хотя бы одна цифра.

Если введенный пользователем пароль не соответствует требованиям безопасности, то вывести на экран соответствующее сообщение и не допускать регистрацию такого пользователя.

Проверку допустимости пароля реализуйте с помощью подключаемой DLL-библиотеки.

Пароль должен быть скрыт и не отображаться на экране. Однако предусмотрите также возможность скрыть или показать пароль при необходимости.

При регистрации нового пользователя нужно вводить ФИО, логин, пароль и фотографию.

8.1. Создание и подключение DLL, содержащей функцию для проверки пароля

Шаг 1. Создайте новый проект. Выберите для него тип «Class Library (.NET Framework)».

```
namespace ClassLibrary1
{
    public class Class1
    {
    }
}
```

Переименуйте пространство имен и класс. Дайте им имена DLLPassword и ClassCheckPassword соответственно.

```
namespace DLLPassword
{
    public class ClassCheckPassword
    {
    }
}
```

Шаг 2. В классе ClassCheckPassword опишите программный код функции CheckPass(). В эту функцию будет передаваться строка текста, содержащая пароль, а возвращаться логическое значение: True, если пароль соответствует требованиям или False, если пароль не соответствует требованиям.

```
public class ClassCheckPassword
{
    public static bool CheckPass(string passw)
    {
        if (passw.Length < 6)  // минимум 6 символов
            return false;

        // счетчики цифр и букв
        int cnt_num = 0, cnt_letter = 0;
        // перебрать символы в пароле
        for (int i = 0; i <= passw.Length - 1; i++)
        {
            if (char.IsNumber(passw[i])) cnt_num++;  // сосчитать цифры
            if (char.IsLetter(passw[i])) cnt_letter++;  // сосчитать буквы
        }

        // должна быть хотя бы одна буква и хотя бы одна цифра
        if (cnt_letter == 0 || cnt_num == 0)
            return false;

        // если все проверки пройдены, то вернуть True
        return true;
    }
}
```

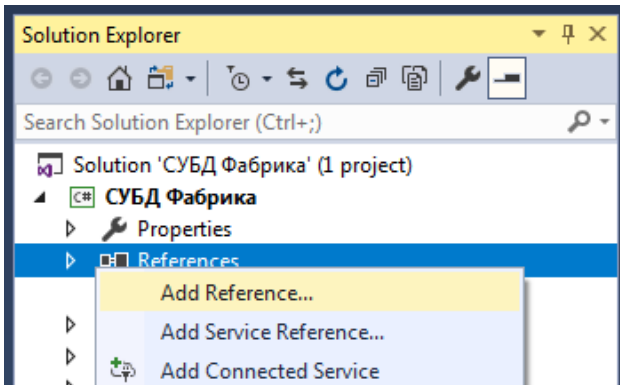
Вначале выполняется проверка длины пароля. Если пароль короче 6 символов, то возвращается значение False и происходит выход из функции. Затем выполняется подсчет количества цифр и количества букв. Для этого в цикле for выполняется перебор символов пароля от 0 до последнего. Если i-й символ цифра, увеличивается счетчик цифр, если буква - увеличивается счетчик букв. Чтобы проверить, является ли i-й символ цифрой, используется функция char.IsNumber(), для проверки, является ли символ буквой, используется функция char.IsLetter(). Если хотя бы один из счетчиков (цифр и букв) остался равным нулю, пароль не соответствует требованиям, необходимо вернуть значение False и выйти из функции.

В том случае, если пароль соответствует требованиям, все проверки будут пройдены и программный код будет выполняться до конца функции. В этом случае из функции возвратиться значение True.

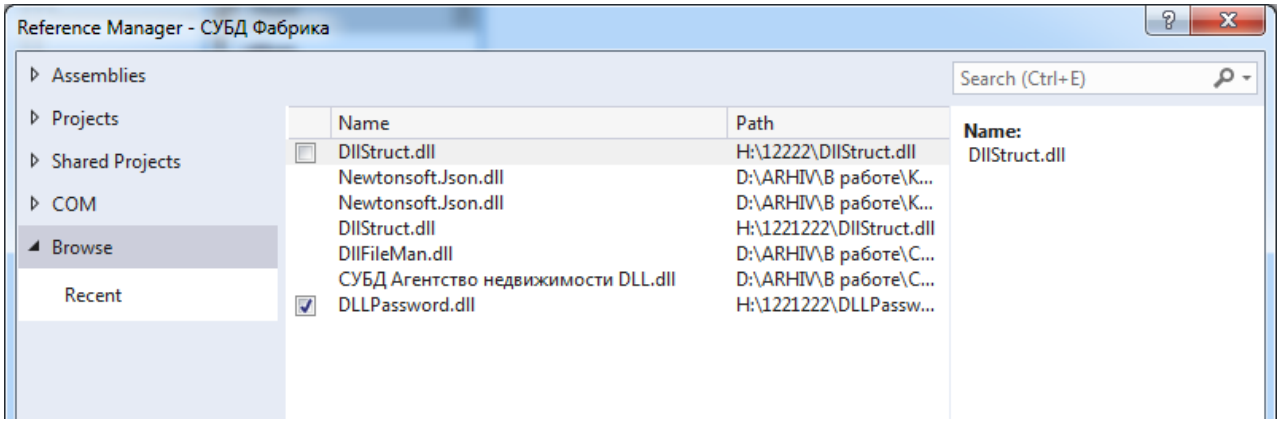
Шаг 3. Создайте dll-файл. Для этого выберите меню Build►Rebuild Solution.

Шаг 4. Откройте проект, в котором будет выполняться регистрация пользователей. К этому проекту нужно будет подключить библиотеку DLL. В панели Solution Explorer

найдите ваше приложение, щелкните правой кнопкой мыши на узле References и выберите пункт Add Reference...



Выберите пункт Browse и с помощью кнопки Browse найдите и подключите библиотеку DLL под именем DLLPassword.dll. Нажмите OK.



8.2. Создание формы для регистрации пользователя

Шаг 1. Добавьте к проекту новую форму. Дайте ей имя FormRegistration.

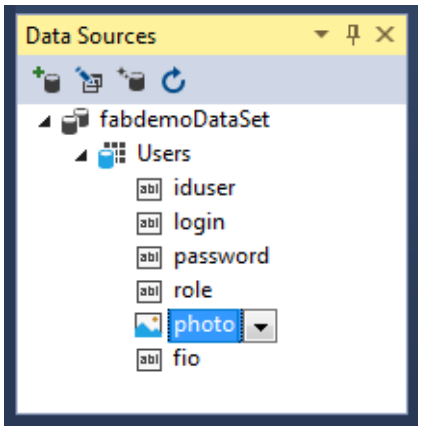
Для формы задайте свойства:

Text - Регистрация нового пользователя (заголовок формы)
StartPosition - CenterParent (отображать форму по центру родительской формы)
MaximizeBox - False (заблокировать кнопку разворачивания формы до максимума)
MinimizeBox - False (заблокировать кнопку сворачивания формы до минимума)
FormBorderStyle - FixedDialog (диалоговое окно без кнопки системного меню, не разрешать изменять размеры формы)

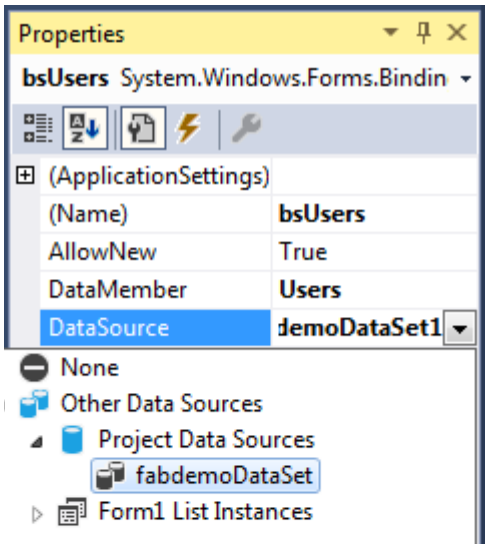
Шаг 2. Откройте программный код формы FormRegistration. Подключите пространство имен для вашей библиотеки DLL.

```
using DLLPassword;
```

Шаг 3. В панели Data Sources разверните таблицу Users. Для поля photo откройте выпадающий список и выберите тип визуального объекта **PictureBox**. Если в Data Sources у вас нет таблицы Users, то откройте Data Sources в режиме конструктора и отбуксируйте из базы данных в конструктор таблицу Users.



Шаг 4. Добавьте на главную форму DataSet и BindingSources. Для BindingSources задайте имя bsUsers и свяжите его с таблицей Users. В свойстве DataSources выберите источник данных, а в свойстве DataMember выберите таблицу Users.



Шаг 5. Из Data Sources отбуксируйте на форму поля fio, login, password, photo. Для PictureBox, который будет использоваться для загрузки фотографии, в свойстве SizeMode установите значение Zoom. Это нужно для того, чтобы размеры загруженной фотографии изменялись в соответствии с размерами PictureBox. Для текстового поля, в которое пользователь будет вводить пароль, в свойстве UseSystemPasswordChar установите значение True. Это нужно для того, чтобы отображались точки вместо символов пароля, которые вводит пользователь.

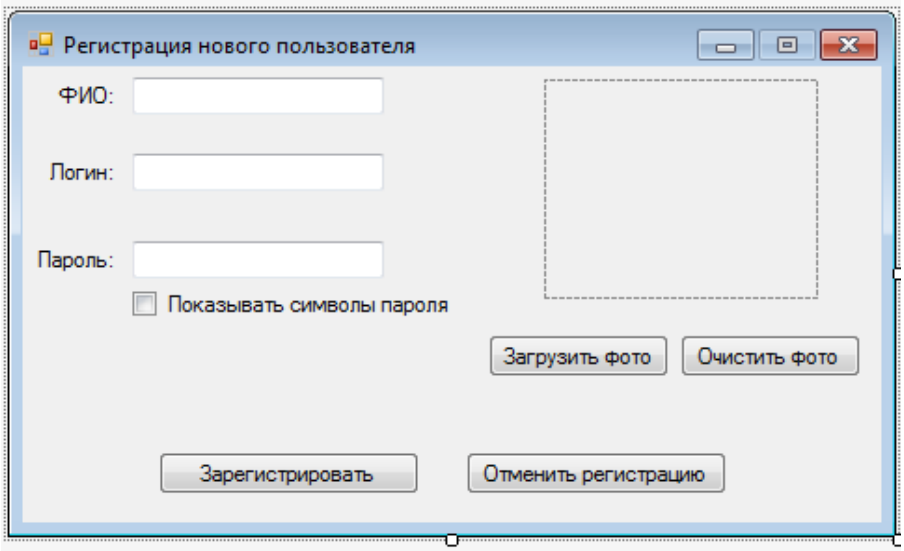
Шаг 6. Сформируйте интерфейс формы для регистрации (см. на следующей странице).

Для CheckBox «Показывать символы пароля» дайте имя cbxShowPass. Для кнопок «Загрузить фото» и «Очистить фото» дайте имена btnLoadPhoto и btnClearPhoto.

Для кнопок «Зарегистрировать» и «Отменить регистрацию» дайте имена btnReg и btnCancel.

Кнопка «Отменить регистрацию» должна закрывать форму. Поэтому в свойстве DialogResult установите для этой кнопки значение Cancel.

Кнопка «Зарегистрировать» должна быть кнопкой по умолчанию, которая срабатывает по нажатию на Enter, а кнопка «Отменить регистрацию» должна быть кнопкой отмены, которая срабатывает по нажатию на Escape. Для формы FormRegistration в свойстве AcceptButton установите значение btnReg (кнопка по умолчанию), а в свойстве CancelButton установите значение btnCancel (кнопка отмены).



Шаг 7. Чтобы скрыть или показать символы пароля, в обработчике события CheckedChanged для cbxShowPass укажите следующий код:

```
private void cbxShowPass_CheckedChanged(object sender, EventArgs e)
{
    // включить / выключить показ пароля
    passwordTextBox.UseSystemPasswordChar = !cbxShowPass.Checked;
}
```

Шаг 8. Для загрузки фотографии добавьте на форму регистрации объект OpenFileDialog. Очистите у этого диалогового окна свойство FileName.

Шаг 9. Для кнопки «Загрузить фото» опишите следующий код:

```
private void btnLoadPhoto_Click(object sender, EventArgs e)
{
    // если пользователь выбрал файл с фотографией, ...
    if (openFileDialog1.ShowDialog() == DialogResult.OK)
        // загрузить фотографию в PictureBox
        photoPictureBox.Image = Image.FromFile(openFileDialog1.FileName);
}
```

Так как PictureBox связан с полем photo из таблицы Users, загруженное изображение будет сохраняться в базе данных.

Шаг 10. Для кнопки «Очистить фото» опишите следующий код:

```
private void btnClearPhoto_Click(object sender, EventArgs e)
{
    // диалоговое окно с вопросом
    DialogResult rez = MessageBox.Show("Очистить фотографию?", "Внимание!",
      MessageBoxButtons.YesNo, MessageBoxIcon.Question);
    // если подтверждается очистка, ...
    if (rez == DialogResult.Yes)
        // то очистить фотографию в PictureBox
        photoPictureBox.Image = null;
}
```

Здесь выводится диалоговое окно с вопросом и кнопками «Да» и «Нет». Если пользователь соглашается с очисткой, то изображение в PictureBox очищается.

Шаг 11. В обработчике события Load для FormRegistrations напишите команду для добавления новой записи в таблицу Users.

```
// добавить нового пользователя в таблицу Users
bsUsers.AddNew();
```

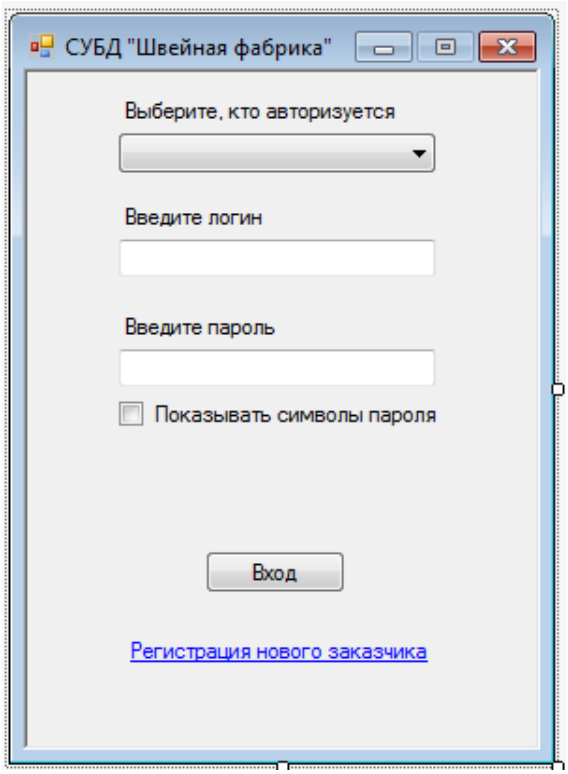
Шаг 12. Для кнопки «Зарегистировать» укажите следующий код:

```
private void btnReg_Click(object sender, EventArgs e)
{
    // если пароль, указанный пользователем, не соответствует требованиям
    if (!ClassCheckPassword.CheckPass(passwordTextBox.Text))
    {
        MessageBox.Show("Пароль не соответствует требованиям.");
        return;  // выйти из процедуры
    }
    try
    {
        // сохранить изменения в таблице Users
        bsUsers.EndEdit();
        usersTableAdapter.Update(fabdemoDataSet1.Users);
        MessageBox.Show("Регистрация нового пользователя успешно
          завершена.");
        this.Close();
    }
    catch
    {
        MessageBox.Show(@"Пользователь с таким логином уже существует.
другой логин.");
    }
}
```

Вначале здесь выполняется проверка пароля с помощью функции CheckPass() из библиотеки DLL. Если проверка успешно пройдена, то выполняется сохранение изменений в базе данных. Если сохранить изменения не удается, это означает, что в таблице Users уже есть пользователь с таким логином.

Шаг 13. Вернитесь на главную форму. На главной форме должна быть кнопка Button или ссылка, или пункт меню, по нажатию на которую должна открываться форма для регистрации. В нашем примере используется метка Label (см. изображение на следующей странице).

В обработчике Click нужно описать программный код, позволяющий открыть форму для регистрации, а после ее закрытия загрузить из базы данных обновленные записи таблицы Users.

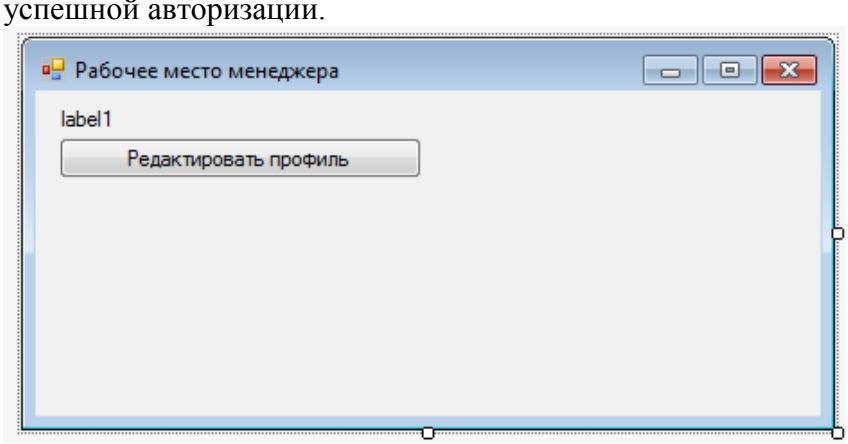



```
private void lblReg_Click(object sender, EventArgs e)
{
    FormRegistration frm = new FormRegistration();
    // открыть форму для регистрации
    frm.ShowDialog();
    // после регистрации нового пользователя загрузить
    // обновленные записи из таблицы Users
    this.usersTableAdapter.Fill(this.fabdemoDataSet1.Users);
}
```

Шаг 14. Запустите приложение и выполните проверку регистрации нового пользователя.

Тема 9. Модуль редактирования профиля пользователя

Пусть у нас имеется рабочее место пользователя, которое открывается после успешной авторизации.



Нам нужно выполнить просмотр и редактирование профиля текущего пользователя. Нужно предусмотреть возможность изменения ФИО, пароля, фотографии. В целях безопасности не отображать пароль пользователя на экране. Разрешать сохранение изменений в профиле только в том случае, если пользователь укажет старый пароль и новый пароль.

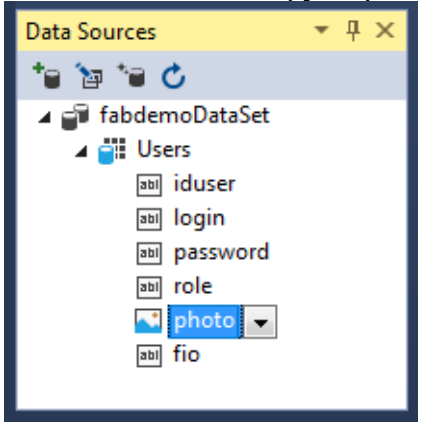
9.1. Формирование интерфейса формы для редактирования профиля пользователя

Шаг 1. Добавьте к проекту новую форму. Назовите ее FormProfileManager. Задайте для формы следующие свойства:
Text - Профиль пользователя (заголовок формы)
StartPosition - CenterParent (отображать форму по центру родительской формы)
MaximizeBox - False (заблокировать кнопку разворачивания формы до максимума)
MinimizeBox - False (заблокировать кнопку сворачивания формы до минимума)
FormBorderStyle - FixedDialog (диалоговое окно без кнопки системного меню, не разрешать изменять размеры формы)

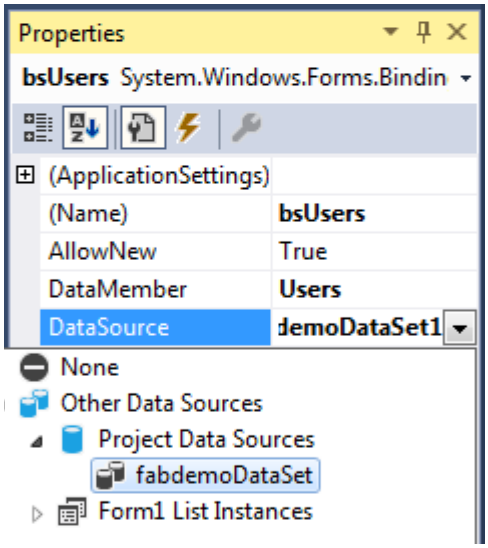
Шаг 2. Откройте программный код формы FormProfile. Подключите пространство имен для вашей библиотеки DLL. Создание этой библиотеки рассматривалось в теме №7 «Модуль регистрации пользователей».

```
using DLLPassword;
```

Шаг 3. В панели Data Sources разверните таблицу Users. Для поля photo откройте выпадающий список и выберите тип визуального объекта **PictureBox**. Если в Data Sources у вас нет таблицы Users, то откройте Data Sources в режиме конструктора и отбуксируйте из базы данных в конструктор таблицу Users.



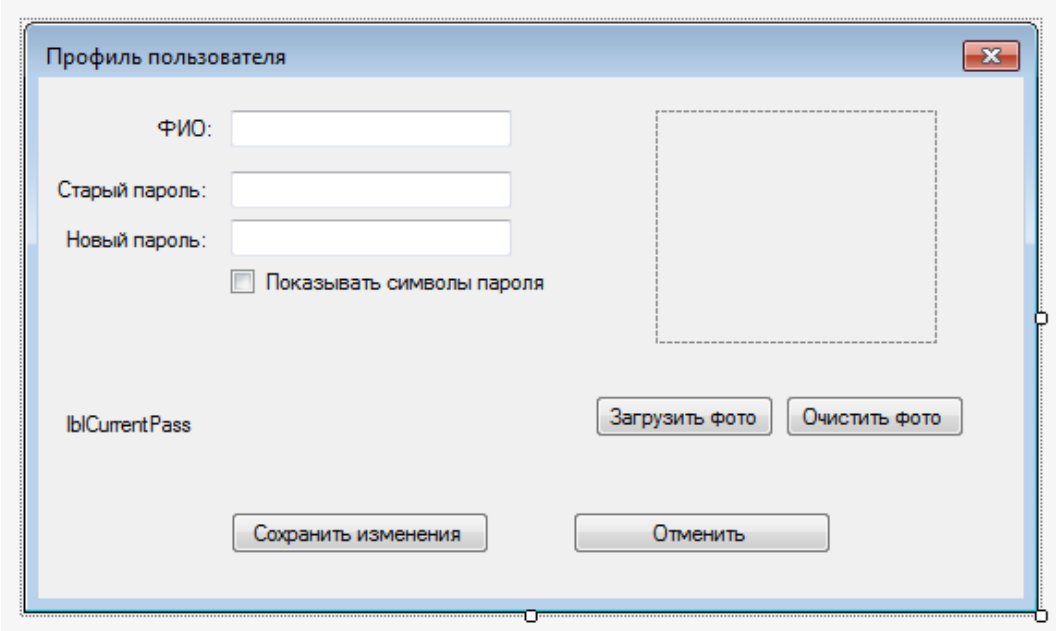
Шаг 4. Добавьте на главную форму DataSet и BindingSources. Для BindingSources задайте имя bsUsers и свяжите его с таблицей Users. В свойстве DataSources выберите источник данных, а в свойстве DataMember выберите таблицу Users.



К bsUsers нам нужно будет обращаться с формы - рабочего места пользователя. Поэтому для bsUsers нужно открыть глобальный доступ. В свойстве Modifiers установите значение Public.

Шаг 5. Из Data Sources отбуксируйте на форму поля fio, и photo. Для PictureBox, который будет использоваться для загрузки фотографии, в свойстве SizeMode установите значение Zoom. Это нужно для того, чтобы размеры загруженной фотографии изменялись в соответствии с размерами PictureBox.

Шаг 6. Сформируйте следующий интерфейс формы для редактирования профиля пользователя:



Текстовое поле для ФИО и PictureBox для фото мы буксировали из Data Sources. Эти визуальные объекты связаны с базой данных Текстовые поля для ввода старого и нового паролей должны быть обычными текстовыми полями, не связанными с базой данных. Дайте им имена tbxOldPass и tbxNewPass. В свойстве UseSystemPasswordChar установите значение True, чтобы вместо символов пароля отображались точки.

Для CheckBox «Показывать символы пароля» дайте имя cbxShowPass. Для кнопок «Загрузить фото» и «Очистить фото» дайте имена btnLoadPhoto и btnClearPhoto.

Для кнопок «Зарегистрировать» и «Отменить регистрацию» дайте имена btnReg и btnCancel.

Кнопка «Отменить регистрацию» должна закрывать форму. Поэтому в свойстве DialogResult установите для этой кнопки значение Cancel.

Кнопка «Зарегистрировать» должна быть кнопкой по умолчанию, которая срабатывает по нажатию на Enter, а кнопка «Отменить регистрацию» должна быть кнопкой отмены, которая срабатывает по нажатию на Escape. Для формы FormRegistration в свойстве AcceptButton установите значение btnReg (кнопка по умолчанию), а в свойстве CancelButton установите значение btnCancel (кнопка отмены).

Метка lblCurrentPass должна быть связана с полем password таблицы Users. В этой метке будет храниться текущий пароль пользователя. В свойстве (DataBindins)►Text для метки установите bsUsers - password. Однако, сама метка должна быть невидимой. Ее содержимое будет использоваться только в программном коде. Для этой метки сделайте цвет текста и цвет фона (свойства ForeColor и BackColor) одинаковыми. Если вы попытаетесь в свойстве Visible для нее установите значение False, то такой прием будет работать не всегда. Во многих случаях метки, имеющие в свойстве Visible значение False, не могут хранить данные из таблиц базы данных.

9.2. Описание программного кода обработчиков событий

Шаг 1. Чтобы скрыть или показать символы пароля, в обработчике события CheckedChanged для cbxShowPass укажите следующий код:

```
private void cbxShowPass_CheckedChanged(object sender, EventArgs e)
{
    // включить / выключить показ пароля
    tbxOldPass.UseSystemPasswordChar = !cbxShowPass.Checked;
    tbxNewPass.UseSystemPasswordChar = !cbxShowPass.Checked;
}
```

Обратите внимание, у нас два текстовых поля для ввода пароля. Следовательно, нужно скрывать или показывать пароль для обоих текстовых полей.

Шаг 2. Для загрузки фотографии добавьте на форму регистрации объект OpenFileDialog. Очистите у этого диалогового окна свойство FileName.

Шаг 3. Для кнопки «Загрузить фото» опишите следующий код:

```
private void btnLoadPhoto_Click(object sender, EventArgs e)
{
    // если пользователь выбрал файл с фотографией, ...
    if (openFileDialog1.ShowDialog() == DialogResult.OK)
        // загрузить фотографию в PictureBox
        photoPictureBox.Image = Image.FromFile(openFileDialog1.FileName);
}
```

Так как PictureBox связан с полем photo из таблицы Users, загруженное изображение будет сохраняться в базе данных.

Шаг 4. Для кнопки «Очистить фото» опишите следующий код:

```
private void btnClearPhoto_Click(object sender, EventArgs e)
{
    // диалоговое окно с вопросом
    DialogResult rez = MessageBox.Show("Очистить фотографию?", "Внимание!",
      MessageBoxButtons.YesNo, MessageBoxIcon.Question);
    // если подтверждается очистка, ...
    if (rez == DialogResult.Yes)
        // то очистить фотографию в PictureBox
        photoPictureBox.Image = null;
}
```

Здесь выводится диалоговое окно с вопросом и кнопками «Да» и «Нет». Если пользователь соглашается с очисткой, то изображение в PictureBox очищается.

Шаг 5. Для кнопки «Сохранить изменения» опишите следующий код:

```
private void btnSave_Click(object sender, EventArgs e)
{
    // если старый пароль не соответствует паролю в базе данных
    if (tbxOldPass.Text != lblCurrenrPass.Text)
    {
        MessageBox.Show("Вы ввели неправильный старый пароль.");
        return;  // выйти из процедуры
    }
    // если пароль, указанный пользователем, не соответствует требованиям
    if (!ClassCheckPassword.CheckPass(tbxNewPass.Text))
    {
        MessageBox.Show("Новый пароль не соответствует требованиям.");
        return;  // выйти из процедуры
    }

    // записать новый пароль в метку lblCurrentPass
    // т.к. lblCurrentPass связана с базой данных, пароль будет
    // сохранен в таблицу Users
    lblCurrenrPass.Text = tbxNewPass.Text;
    // сохранить изменения в таблице Users
    bsUsers.EndEdit();
    usersTableAdapter.Update(fabdemoDataSet1.Users);
    this.Close();
}
```

Вначале выполняется проверка, соответствует ли старый пароль, указанный пользователем, текущему паролю, который храниться в базе данных. Текущий пароль пользователя находится в метке lblCurrentPass. Эта метка связана с полем password таблицы Users. Мы сделали ее невидимой, чтобы пользователь не видел своего текущего пароля.

Затем с помощью функции CheckPass из библиотеки DLL новый пароль проверяется на соответствия требованиям. См. требования к паролю в теме «Регистрация пользователя».

Если все проверки пройдены успешно, то новый пароль записывается в метку lblCurrentPass. А так как эта метка связана с полем password таблицы Users, пароль попадает в базу данных.

Далее сохраняются изменения в таблице Users и текущая форма закрывается.

Шаг 6. Вернитесь на форму FormManager - рабочее место менеджера. Для кнопки «Редактировать профиль» опишите следующий код:

```
private void btnProfile_Click(object sender, EventArgs e)
{
    FormProfile frm = new FormProfile();
    // передать на форму для профиля фильтр для пользователя
    // с формы-рабочего места пользователя
    frm.bsUsers.Filter = this.bsUsers.Filter;
    // открыть форму для редактирования профиля пользователя
    frm.ShowDialog();
    // загрузить из базы данных обновленные записи в таблице Users
    this.usersTableAdapter.Fill(this.fabdemoDataSet1.Users);
}
```

Вначале здесь создается форма для редактирования профиля пользователя. Затем на форму для редактирования профиля передается фильтр с формы рабочего места пользователя. Это нужно для того, чтобы при открытии формы с профилем мы видели не первого пользователя в списке, а того пользователя, который авторизовался. Далее форма для редактирования профиля открывается. После того, как форма для редактирования профиля закроется, из таблицы Users загружаются обновленные записи.

Шаг 7. Запустите приложение, авторизуйтесь и попробуйте редактировать профиль пользователя.

Тема 10. Модуль нечеткого поиска с учетом расстояния Левенштейна

Расстояние Левенштейна - это минимальное количество операций вставки одного символа, удаления одного символа и замены одного символа на другой, необходимых для превращения одной строки в другую. Например:

Расстояние между одинаковыми строками равно 0

Расстояние между строками «строка» и «собака» равно 3, замены: «т» на «о», «р» на «б» и «о» на «а».

Расстояние между строками «строка» и «вафля» равно 6, необходимо заменить все 5 символов и удалить еще 1 лишний.

Поэтому при нечетком поиске с использованием слова «строка» в результирующую выборку должны попасть слова «строка», «собака», но не слово «вафля».

Пусть у нас имеется таблица Users, содержащая данные пользователей. Рассмотрим, как сделать нечеткий поиск с учетом расстояния Левенштейна по фамилиям или именам пользователей.

COMP\MSSQLSERVE...demo - dbo.Users

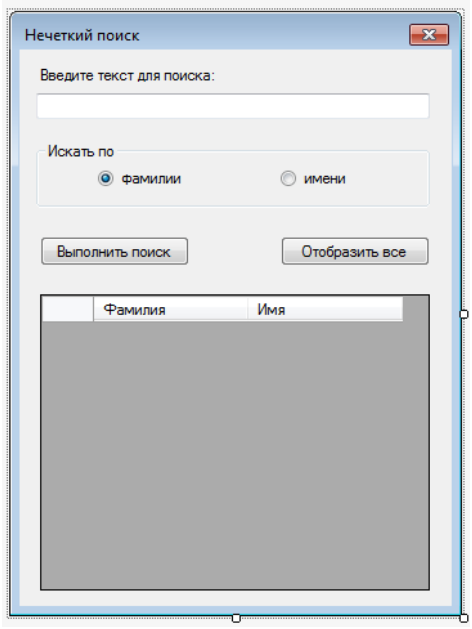
iduser	loqin	password	role	fam	name	otch	phone
16	z	z	заказчик	Иванов	Иван	Ильич	+7(915) 158-98-...
17	m	m	менеджер	Петров	Андрей	Егорович1	+7 (905) 543-45...
18	k	k	кладовщик	Андреев	Семен	Степанович	+7(910) 324-65-...
19	stepanova	sgf432	заказчик	Степанова	Ирина	Петровна	+7(952) 543-67-...
20	ilyina	3245bghfx	менеджер	Ильина	Анна	Петровна	+7(905) 678-65-...
21	egorov	sdgf435	кладовщик	Егоров	Петр	Андреевич	+7(910) 122-54-...
22	vasilyeva	gfbet345	заказчик	Васильева	Елена	Алексеевна	+7(915) 677-99-...
24	z1	1A!zzz	менеджер	Яколвев	Егор	Петрович	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Если вы реализуете поиск с учетом расстояния Левенштейна в уже существующем проекте, убедитесь, что база данных, содержащая таблицу Users, к проекту подключена и источник данных Data Sources настроен.

Если вы создаете новый проект, то подключите к нему базу данных с таблицей Users и настройте Data Sources.

10.1. Формирование интерфейса формы для нечеткого поиска

Шаг 1. Сформируйте форму для поиска следующим образом:



С базой данных мы будем работать с помощью SQL-запросов. Поэтому на форму не нужно добавлять DataSet и BindingSources.

Задайте для формы следующие свойства:

Text - Нечеткий поиск (заголовок формы)
StartPosition - CenterParent (отображать форму по центру родительской формы)
MaximizeBox - False (заблокировать кнопку разворачивания формы до максимума)
MinimizeBox - False (заблокировать кнопку сворачивания формы до минимума)
FormBorderStyle - FixedDialog (диалоговое окно без кнопки системного меню, не разрешать изменять размеры формы)

Текстовому полю для поиска дайте имя tbxFind.

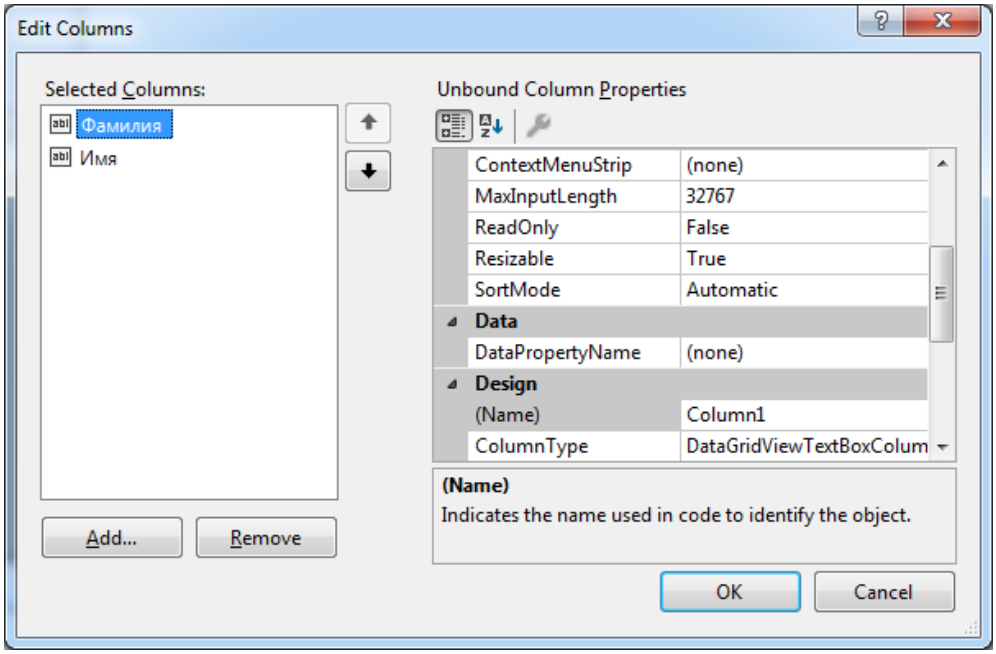
Кнопкам для выбора поля, по которому будет выполняться поиск, дайте имена rbtFam и rbtName.

Кнопкам Button для выполнения поиска и отображения всех записей дайте имена btnFind и btnShowAll.

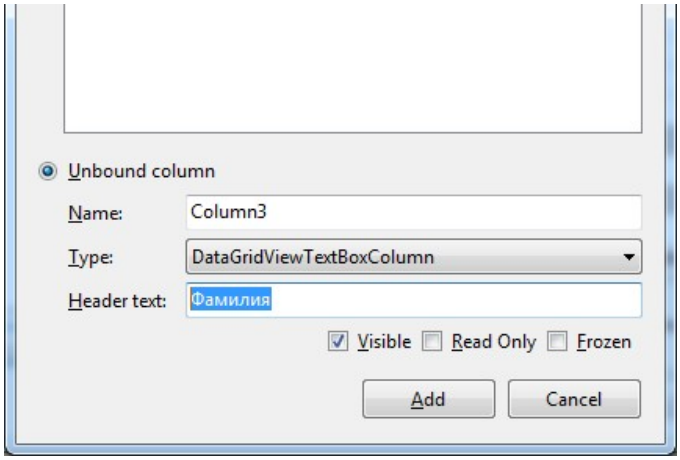
Для DataGridView дайте имя dgvUsers. Этот DataGridView не должен быть связан с какой-либо таблицей базы данных. Он будет заполняться вручную.

Шаг 2. Настройка DataGridView.

Щелкните правой кнопкой на DataGridView и выберите пункт Edit Columns...



С помощью кнопки Add добавьте в DataGridView два столбца, непривязанные к базе данных.



Для каждого столбца в свойстве Header Text напишите заголовок.

Увеличьте ширину каждого столбца до 120 пикселей. В свойстве Width для столбцов укажите значение 120.

После того, как закончите формирование столбцов, задайте для DataGridView следующие свойства:
AllowUserToAddRows - False (чтобы пользователь не мог добавлять строки в таблицу)
AllowUserToDeleteRows - False (чтобы пользователь не мог удалять строки из таблицы)
ReadOnly - True (чтобы пользователь не мог редактировать ячейки таблицы)
SelectionMode - FullRowSelect (чтобы при выделении в таблице выделялась сразу вся строка)

Шаг 3. Далее нам нужно работать с базой данных с помощью SQL-запросов. На главной форме приложения у вас уже должна быть переменная txtcon, содержащая строку подключения к базе данных. Если такой переменной нет, то создайте ее следующим образом.

Сделаем на главной странице строковую переменную, в которой будет храниться строка подключения. Эта переменная будет с глобальным доступом и к ней можно будет обратиться с любой формы.

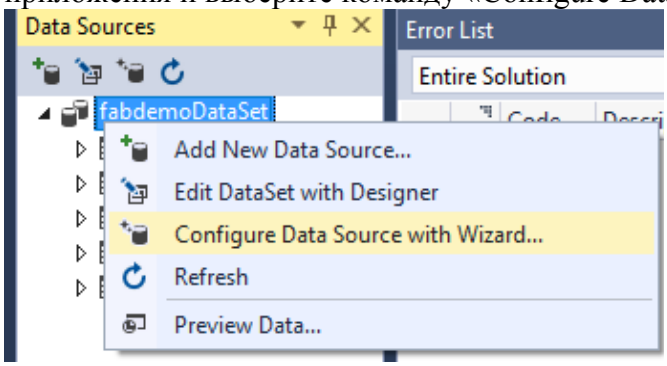
Переключитесь в программный код главной формы (форма для авторизации). После конструктора формы объявите строковую переменную txtcon и присвойте ей значение пустого текста.

```
public Form1()
{
    InitializeComponent();
}

// строка подключения к базе данных
public static string txtcon = @"";
```

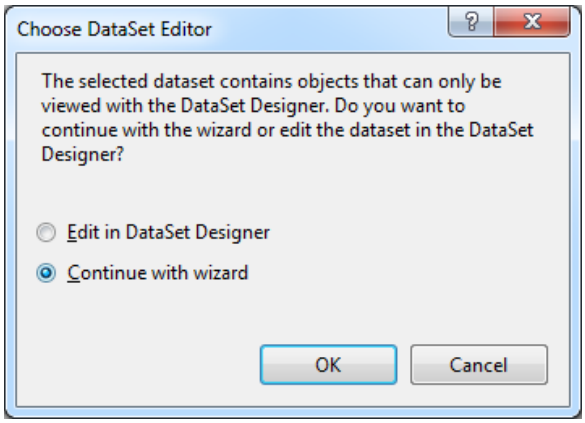
public означает, что к этой переменной у нас есть глобальный доступ

static означает, что это статическая переменная, и к ней можно обращаться прямо из класса формы (Form1.txtcon).

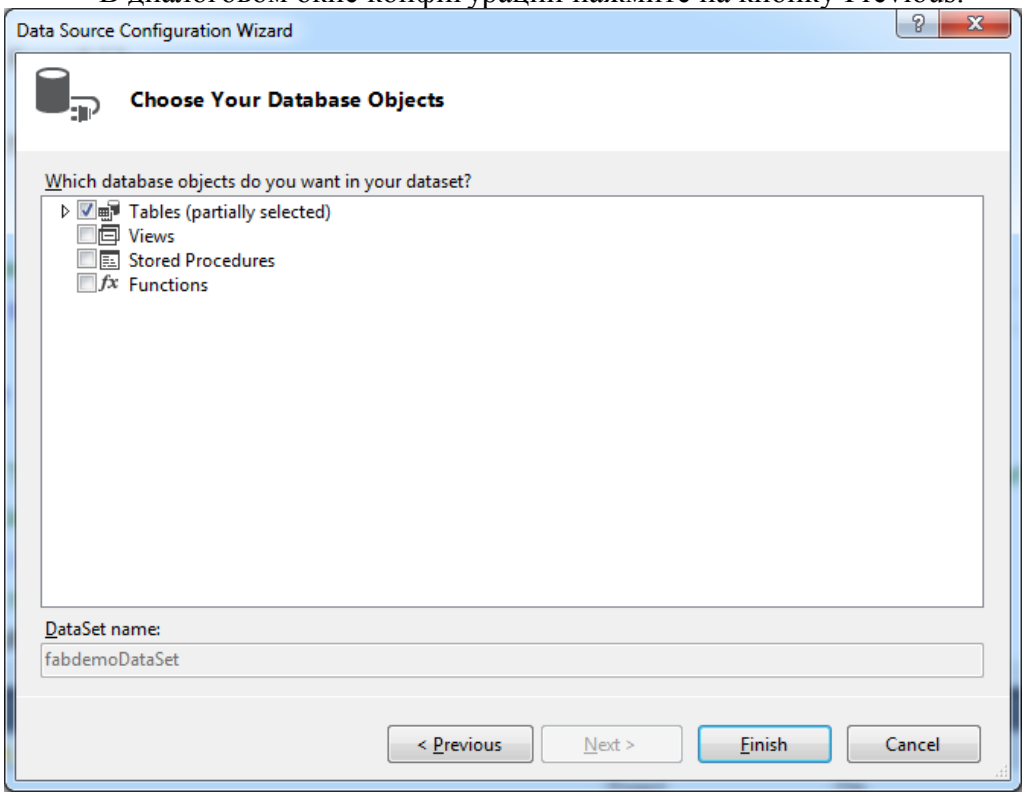
В панели Data Sources щелкните правой кнопкой на источнике данных для нашего приложения и выберите команду «Configure Data Source with Wizard...».



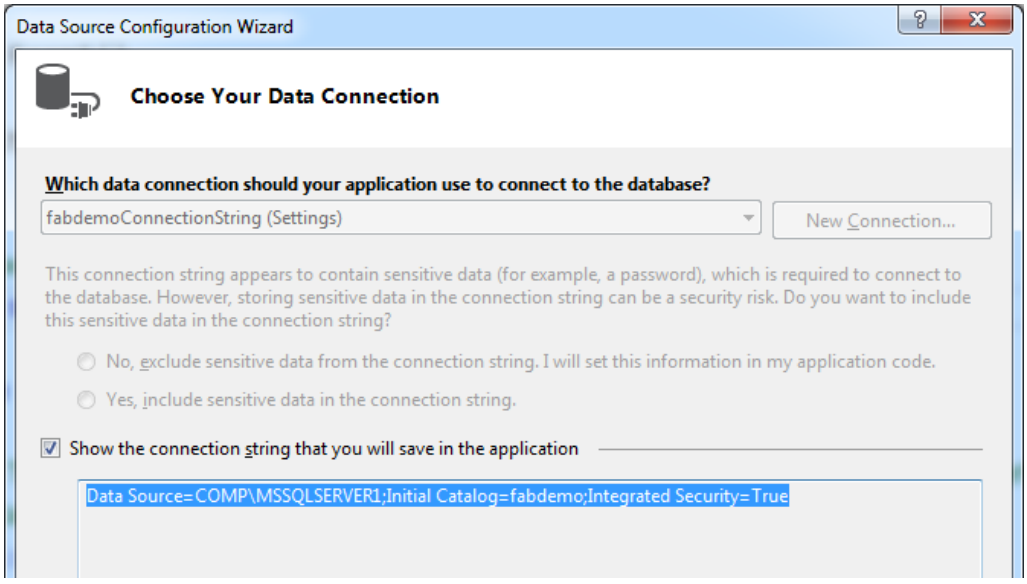
Выберите пункт «Continue with Wizard».



В диалоговом окне конфигурации нажмите на кнопку Previous.



Установите флажок «Show the connection string» и скопируйте в буфер строку подключения к базе данных.



Затем вставьте текст строки подключения в переменную txtcon внутри кавычек.

```
// строка подключения к базе данных
public static string txtcon = @"Data Source=COMP\MSSQLSERVER1;Initial
  Catalog=fabdemo;Integrated Security=True";
```

Внимание! Сначала в программном коде нужно поставить пустые кавычки, и только потом вставлять в них текст строки подключения. Если сделать наоборот, то редактор Visual Studio добавит в строку подключения лишние пробелы и она будет уже испорчена.

10.2. Описание программного кода для нечеткого поиска

Шаг 1. Подключите пространство имен для работы базами данных с помощью SQL-запросов.

```
using System.Data.SqlClient;  // для работы с базами данных
```

Шаг 2. Создадим функцию с именем Levenshtein(), в которую будут передаваться две строки, а в качестве результата возвращаться расстояние Левенштейна.

Переключитесь в программный код. После конструктора формы добавьте программный код функции.

```
int levenshtein(string t1, string t2)
{
    // разница в кол-ве символов в проверяемых строках
    int lev = Math.Abs(t1.Length - t2.Length);
    // минимальная длина из длин двух строк
    int minlen = Math.Min(t1.Length, t2.Length);
    // перебор символов от нулевого и до последнего более короткой строки
    for (int i = 0; i <= minlen - 1; i++)
        // если символы не совпали, сосчитать их
        if (t1[i] != t2[i]) lev++;
    // возвратить расстояние Левенштейна для строк t1 и t2
    return lev;
}
```

По определению расстояние Левенштейна - это минимальное количество операций вставки одного символа, удаления одного символа и замены одного символа на другой, необходимых для превращения одной строки в другую.

Вначале мы найдем разницу между длиной строк. Это будет количество символов, необходимых для вставки или удаления, чтобы превратить одну строку в другую.

Затем определим, какая из строк более короткая. Найдем минимальную длину из длин двух строк.

Затем будем выполнять перебор всех символов более короткой строки от нулевого до последнего. Если i-е символы в одной и другой строке не совпадают, то увеличиваем счетчик расстояния Левенштейна.

Полученное расстояние Левенштейна возвращаем в качестве значения функции.

Шаг 3. Создадим структуру для хранения фамилии и имени одного пользователя и список List, элементами которого будет являться данная структура. Этот программный код, как и функция Levenshtein() также должен быть внутри класса формы после описания конструктора формы.

```
struct user
{
    public string fam, name;
}
// список с фамилиями и именами клиентов
List<user> lst = new List<user>();
```

Шаг 4. Когда наша форма отображается на экране, в DataGridView должны быть видны фамилии и имена пользователей. Кроме того, эти же фамилии и имена должны быть прочитаны из базы данных в список List.

В обработчике события Load опишите следующий код:

```
private void Form1_Load(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection(Form1.txtcon);
    con.Open(); // открыть подключение к базе данных

    // SQL-запрос для извлечения из БД списка нужных пользователей
    string sql = "select fam, name from users";
    SqlCommand query1 = new SqlCommand(sql, con);
    // выполнить запрос и сохранить результат запроса в rez
    SqlDataReader rez = query1.ExecuteReader();
```

Вначале здесь создается и открывается подключение к базе данных SqlConnection. В строку sql записывается SQL-запрос, извлекающий из таблицы Users фамилии и имена пользователей. Затем создается объект SqlCommand и наш запрос выполняется. Результат запроса записывается в переменную rez. Результатом является виртуальная таблица, содержащая фамилии и имена пользователей.

```
    // если есть строки в результате запроса
    if (rez.HasRows)
    {
        // прочитать очередную строку из результата запроса
        // выполнять цикл до тех пор, пока не достигли последней записи в
        // таблице с результатами запроса
        while (rez.Read())
        {
            user u1;
            // сформировать новый элемент для списка пользователей
            u1.fam = rez["fam"].ToString(); u1.name = rez["name"].ToString();
            // добавить в список очередного пользователя
            lst.Add(u1);
            // вывести данные очередного пользователя в таблицу
            dgvUsers.Rows.Add(u1.fam, u1.name);
        }
    }
    con.Close();  // закрыть подключение к базе данных
}
```

Результат запроса может быть пустым, если в таблице Users не будет ни одного пользователя. Свойство HasRows будет содержать значение True, если в результате запроса есть отобранные записи или False, если ни одна запись из базы данных не была выбрана.

В цикле while мы используем команду rez.Read(). Эта команда делает два действия:
1) считывает из виртуальной таблицы с результатами запроса очередную запись;
2) возвращает значение True, если удалось прочитать очередную запись из результата запроса, или значение False, если записи закончились и читать больше нечего.

В цикле мы создаем переменную типа user. В поля fam и name этой переменной записываем фамилию и имя из переменной rez (результат запроса). Далее добавляем эту переменную в список List и выводим фамилию с именем в DataGridView на форму.

Таким образом, у нас получается две таблицы с фамилиями и именами. Одна таблица видимая, располагается в DataGridView. Другая - невидимая, хранится в списке List.

Шаг 5. Для кнопки «Выполнить поиск» опишем следующий программный код:

```
private void btnFind_Click(object sender, EventArgs e)
{
    dgvUsers.Rows.Clear(); // очистить таблицу
    // перебор пользователей в списке
    for (int i = 0; i <= lst.Count - 1; i++)
    {
        // сравниваем фамилию или имя пользователя с текстом для поиска
        // если пользователь выделил кнопку rbtFam, то сравниваем фамилию
        // если пользователь выделил кнопку rbtName, то сравниваем имя
        if (levenshtein(lst[i].fam, tbxFind.Text) <= 3 && rbtFam.Checked ||
            levenshtein(lst[i].name, tbxFind.Text) <= 3 && rbtName.Checked)
            // если расстояние Левенштейна не превышает 3, то
            // вывести в DataGridView такого пользователя
            dgvUsers.Rows.Add(lst[i].fam, lst[i].name);
    }
}
```

Вначале очищаем DataGridView от старых записей. Затем с помощью цикла for выполняем перебор элементов списка List.

Если пользователь выполняет поиск по фамилии (выделил кнопку rbtFam), то вызываем функцию Levenshtein(), в которую передаем фамилию i-го пользователя и символы для поиска из текстового поля tbxFind. Если пользователь выполняет поиск по имени (выделил кнопку rbtName), то вызываем функцию Levenshtein(), в которую передаем имя i-го пользователя и символы для поиска из текстового поля tbxFind. В нашем примере расстояние Левенштейна не должно превышать 3, т.е. имя или фамилия пользователя должны отличаться от символов для поиска не более чем на 3 символа. Если этому критерию пользователь соответствует, то он выводится в DataGridView.

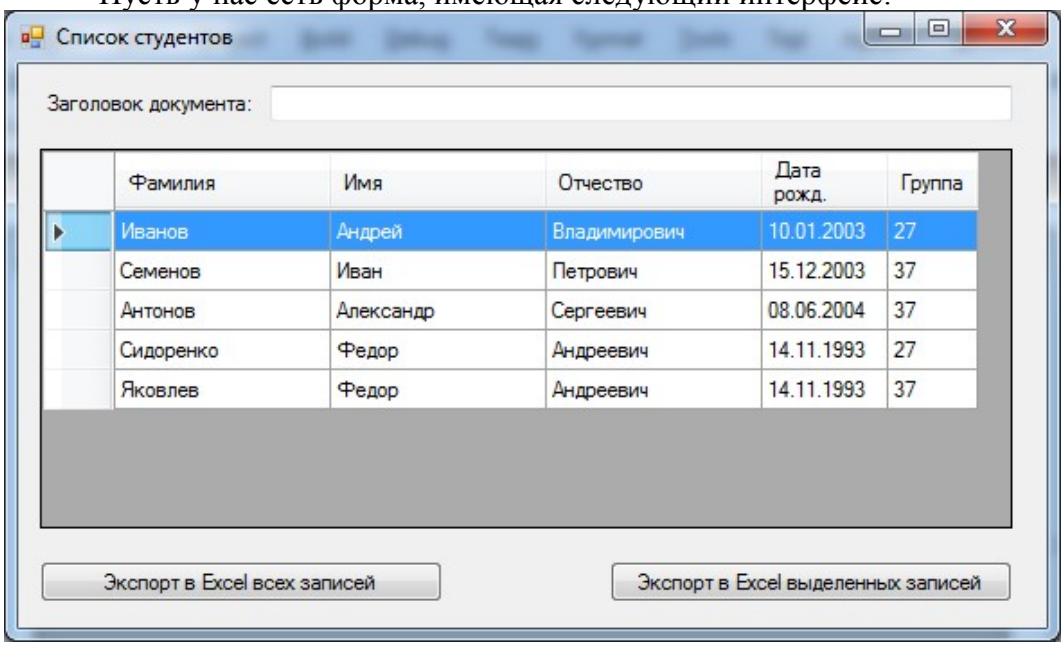
Шаг 6. Для кнопки «Отобразить все» опишите следующий код:

```
private void btnShowAll_Click(object sender, EventArgs e)
{
    dgvUsers.Rows.Clear(); // очистить таблицу
    // перебор пользователей в списке
    for (int i = 0; i <= lst.Count - 1; i++)
        // вывести пользователя в DataGridView
        dgvUsers.Rows.Add(lst[i].fam, lst[i].name);
}
```

Вначале очищаем DataGridView от старых записей. Затем с помощью цикла for выполняем перебор элементов списка List. Фамилию и имя каждого i-го пользователя выводим в DataGridView безо всякой проверки каких-либо условий.

Тема 11. Модуль экспорта данных в Microsoft Excel

Пусть у нас есть форма, имеющая следующий интерфейс:



DataGridView имеет имя dgvStud, текстовое поле для заголовка документа - tbxHead, кнопки для экспорта - btnToExcelAll и btnToExcelSelect. DataGridView отображает данные, взятые из таблицы базы данных, либо добавленные вручную из программного кода.

Необходимо по нажатию на кнопку «Экспорт в Excel всех записей» сформировать документ Excel, в который вывести заголовок документа и все данные из таблицы. По нажатию на кнопку «Экспорт в Excel выделенных записей» нужно делать то же самое, но не для всех записей, а только для выделенных.

11.1. Создание шаблона для экспорта в Microsoft Excel

В текстовом редакторе Microsoft Word нужно создать документ - заготовку, в которую мы будем выводить данные из нашего приложения.

Шаг 1. В Microsoft Excel создайте новый документ следующего вида:

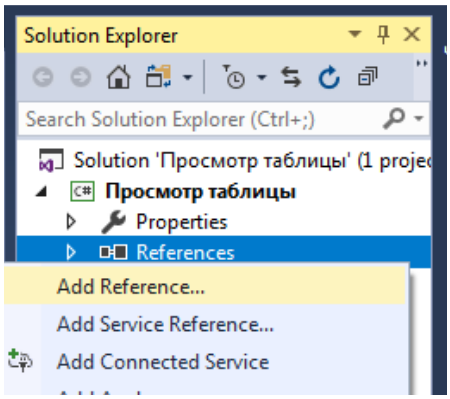
	A	B	C	D	E	F
1	Место для заголовка					
2	№ п/п	Фамилия	Имя	Отчество	Дата рождения	Группа
3						
4						
5						
6						

Ячейки в диапазоне A1:F1 нужно объединить. В это место будет выводиться заголовок нашего документа. Для заголовков столбцов (диапазон A2:F2) нужно задать выравнивание по центру по верхнему краю ячейки. Сделайте также перенос по словам и измените границы и заливку.

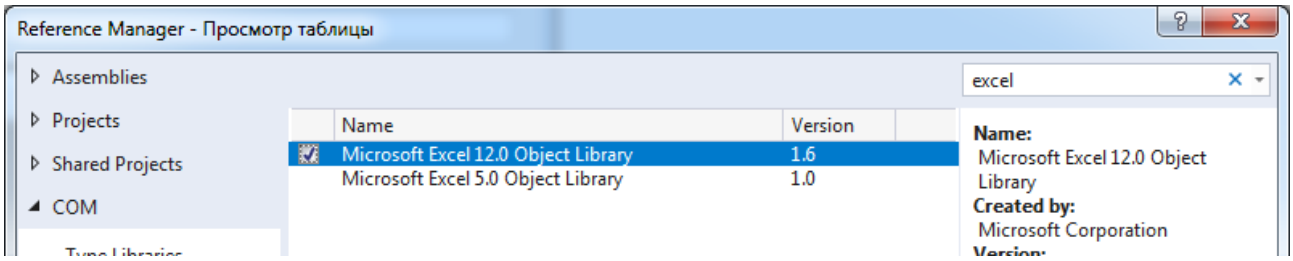
Шаг 2. Сохраните документ в папку с exe-файлом приложения, в подкаталог bin \ debug.

11.2. Подключение к приложению библиотеки для работы с Microsoft Excel

Шаг 1. В панели Solution Explorer щелкните правой кнопкой мыши на узле References и выберите пункт Add Reference...



В разделе COM выполните поиск библиотеки с именем Excel. Если будут найдены библиотеки нескольких версий, выберите версию с наибольшим номером. Выделите ее и нажмите ОК.



Шаг 2. Переключитесь в программный код и подключите пространство имен для работы с Microsoft Excel.

```
// подключить для работы с Microsoft Excel
using Excel = Microsoft.Office.Interop.Excel;
```

11.3. Экспорт в Excel всех записей

Для кнопки «Экспорт в Excel всех записей» опишите следующий код:

```
private void btnToExcelAll_Click(object sender, EventArgs e)
{
    // создать объект для связи с приложением Excel
    Excel.Application exapp = new Excel.Application();
    // сделать приложение Excel видимым
    exapp.Visible = true;
    // открыть в Excel шаблон, находящийся в папке с exe-файлом
    //в режиме только для чтения
    exapp.Workbooks.Open(Application.StartupPath + "\\Студенты.xlsx", Type.Missing, true);
    // создать переменную list1 для связи с первым листом рабочей книги
    Excel.Worksheet list1 = exapp.Worksheets.get_Item(1);

    list1.get_Range("A1").Value = tbxHead.Text;  // вывести в Excel заголовок
```

Здесь вначале создается объект Excel.Application. При создании этого объекта запускается приложение Microsoft Excel. Но на экране это окно этого приложения пока не отображается. Далее приложение Microsoft Excel делается видимым. С помощью метода Workbooks.Open() в Microsoft. Excel открывается шаблон для будущей таблицы. Первым параметром указывается имя файла. В нашем случае это файл «Студенты.xlsx», который

находится в папке с exe-файлом приложения. Третий параметр - true - обозначает, что файл нужно открыть в режиме «только для чтения». Это нужно для того, чтобы пользователь не испортил шаблон, нажав в Microsoft Excel на кнопку Сохранить.

Документ Microsoft Excel называется рабочей книгой. Книга состоит из отдельных листов. Чтобы выводить данные в Microsoft Excel, нужно объявить переменную типа Excel.Worksheet и связать ее с одним из листов рабочей книги. Номер листа, с которым связывается переменная, задается в методе get_Item(). В нашем примере создается переменная list1, связанная с первым листом рабочей книги.

Чтобы вывести данные в какую-либо ячейку Excel, используется следующая команда:

```
list1.get_Range("координаты_ячейки").Value = значение;
```

В ячейку A1 мы выводим заголовок документа из текстового поля tbxHead.

```
    int rowexcel = 3;  // в Excel начинать выводить с 3 строки
    // перебор записей в DataGridView
    for (int i = 0; i <= dgvStud.RowCount - 1; i++)
    {
        // вывести в столбец A порядковый номер
        // апостроф перед номером устанавливает для ячейки Excel строковый тип
        // после номера добавляется точка
        list1.get_Range("A" + rowexcel).Value = "'" + (i + 1) + ".";
        // в столбцы B - F вывести данные из DataGridView
        list1.get_Range("B" + rowexcel).Value = dgvStud.Rows[i].Cells[0].Value;
        list1.get_Range("C" + rowexcel).Value = dgvStud.Rows[i].Cells[1].Value;
        list1.get_Range("D" + rowexcel).Value = dgvStud.Rows[i].Cells[2].Value;
        list1.get_Range("E" + rowexcel).Value =
            Convert.ToDateTime(dgvStud.Rows[i].Cells[3].Value).ToShortDateString();
        list1.get_Range("F" + rowexcel).Value = dgvStud.Rows[i].Cells[4].Value;
        // диапазон для выделения всех ячеек заполненой строки в Excel
        string range = String.Format("A{0}:F{1}", rowexcel, rowexcel);
        // установить для заполненой строки границы между ячеками сплошно линией
        list1.get_Range(range).Borders.LineStyle = Excel.XlLineStyle.xlContinuous;

        rowexcel++;  // перейти к следующей строке в Excel
    }
}
```

В таблице Excel записи будут выводиться начиная с третьей строки. Мы будем использовать переменную rowexcel для счетчика строк в таблице Excel.

В цикле for выполняется перебор всех записей в DataGridView. В столбец A нужно выводить порядковый номер, начиная от единицы. Переменная i, являющаяся счетчиком строк в DataGridView, изменяет свои значения начиная от нуля. В Excel будем выводить значение на единицу большее. Символ апострофа перед содержимым ячейки указывает, что в ячейке Excel будут находиться текстовые данные. После порядкового номера добавляется точка. Если не поставить апостроф, то Excel будет рассматривать порядковый номер как вещественное число, а точку как разделитель целой и дробной части числа. А так как дробной части у порядкового номера нет, Excel удалит эту точку.

Далее в столбцы от B до F выводятся данные из i-й строки DataGridView. Только дату рождения студента мы вначале преобразуем к типу DateTime(), а затем используем метод ToShortDateString(). Это нужно для того, чтобы получить дату в коротком формате: дд.мм.гггг, где дд -день, мм - месяц, гггг - год. Без использования этого метода вместе с датой будет выводиться еще и время.

После того, как очередная строка в Excel мы заполнили данными, нужно установить для этой строки границы между ячейками сплошной линией.

Далее увеличиваем значение rowexcel для перехода к новой строке.

Запустите приложение. В текстовое поле введите заголовок «Студенты ГАПОУ НППК» и нажмите на кнопку «Экспорт в Excel всех записей». В Microsoft Excel должен сформироваться следующий документ:

	A	B	C	D	E	F
1	Студенты ГАПОУ НППК					
2	№ п/п	Фамилия	Имя	Отчество	Дата рождения	Группа
3	1.	Иванов	Андрей	Владимирович	10.01.2003	27
4	2.	Семенов	Иван	Петрович	15.12.2003	37
5	3.	Антонов	Александр	Сергеевич	08.06.2004	37
6	4.	Сидоренко	Федор	Андреевич	14.11.1993	27
7	5.	Яковлев	Федор	Андреевич	14.11.1993	37
8						
9						

11.4. Экспорт в Excel выделенных записей

Шаг 1. Чтобы можно было выделять несколько записей в DataGridView, установите для dgvStud следующие свойства:

MultiSelect - True (разрешить множественное выделение).

SelectionMode - FullRowSelect (выделять в таблице целиком всю строку).

Чтобы выделить в DataGridView несколько строк, выделяйте их, удерживая нажатой клавишу Ctrl.

Шаг 2. Для кнопки «Экспорт в Excel выделенных записей» опишите следующий код:

```
private void btnToExcelSelect_Click(object sender, EventArgs e)
{
    if (dgvStud.SelectedRows.Count == 0)
    {
        MessageBox.Show("Нет выделенных строк в таблице.");
        return;
    }

    // создать объект для связи с приложением Excel
    Excel.Application exapp = new Excel.Application();
    // сделать приложение Excel видимым
    exapp.Visible = true;
    // открыть в Excel шаблон, находящийся в папке с exe-файлом
    //в режиме только для чтения
    exapp.Workbooks.Open(Application.StartupPath + "\\Студенты.xlsx", Type.Missing, true);
    // создать переменную list1 для связи с первым листом рабочей книги
    Excel.Worksheet list1 = exapp.Worksheets.get_Item(1);

    list1.get_Range("A1").Value = tbxHead.Text;  // вывести в Excel заголовок
```

Вначале следует проверить, есть ли в DataGridView выделенные строки. Если пользователь не выделил ни одной строки, то экспортировать в Microsoft Excel будет нечего. В этом случае нужно вывести текстовое сообщение и выйти из процедуры.

Дальнейший программный код пока без изменений. Он точно такой же, как и для экспорта в Microsoft Excel всех записей.

```
    int rowexcel = 3;  // в Excel начинать выводить с 3 строки
    // перебор выделенных записей в DataGridView
    for (int i = 0; i <= dgvStud.SelectedRows.Count - 1; i++)
    {
        // вывести в столбец A порядковый номер
        // апостроф перед номером устанавливает для ячейки Excel строковый тип
        // после номера добавляется точка
        list1.get_Range("A" + rowexcel).Value = "'" + (i + 1) + ".";
        // в столбцы B - F вывести данные из i-й выделенной строки DataGridView
        list1.get_Range("B" + rowexcel).Value = dgvStud.SelectedRows[i].Cells[0].Value;
        list1.get_Range("C" + rowexcel).Value = dgvStud.SelectedRows[i].Cells[1].Value;
        list1.get_Range("D" + rowexcel).Value = dgvStud.SelectedRows[i].Cells[2].Value;
        list1.get_Range("E" + rowexcel).Value =
            Convert.ToDateTime(dgvStud.SelectedRows[i].Cells[3].Value)
            .ToShortDateString();
        list1.get_Range("F" + rowexcel).Value = dgvStud.SelectedRows[i].Cells[4].Value;
        // диапазон для выделения всех ячеек заполненой строки в Excel
        string range = String.Format("A{0}:F{1}", rowexcel, rowexcel);
        // установить для заполненой строки границы между ячеками сплошно линией
        list1.get_Range(range).Borders.LineStyle = Excel.XlLineStyle.xlContinuous;

        rowexcel++;  // перейти к следующей строке в Excel
    }
}
```

В цикле for мы будем выполнять перебор не всех строк DataGridView, а только выделенных. Выделенные строки находятся в свойстве SelectedRows. Фамилии, имена и прочие данные студентов мы также будем брать из очередной i-й выделенной строки таблицы DataGridView.

Шаг 3. Запустите приложение, выделите несколько строк в DataGridView. В текстовое поле введите заголовок «Студенты ГАПОУ НППК» и нажмите на кнопку «Экспорт в Word выделенных записей». У вас должен сформироваться примерно такой документ:

	A	B	C	D	E	F
1	Студенты ГАПОУ НППК					
2	№ п/п	Фамилия	Имя	Отчество	Дата рождения	Группа
3	1.	Яковлев	Федор	Андреевич	14.11.1993	37
4	2.	Антонов	Александр	Сергеевич	08.06.2004	37
5	3.	Иванов	Андрей	Владимирович	10.01.2003	27
6						
7						

11.5. Работа с приложением на другом компьютере

Предположим, что вы начинали работу с приложением, использующим экспорт в Microsoft Excel, на одном компьютере, а затем скопировали его на другой компьютер. Рассмотрим, какие при этом могут возникнуть проблемы.

Для того, чтобы можно было выполнять экспорт в Microsoft Excel, на компьютере должен быть установлен Microsoft Office. При этом версия Microsoft Office должна быть не ниже 2007. При использовании более старых версий: 2003, 2000 и т.п. могут быть проблемы с экспортом.

Если вы на другом компьютере запускаете exe-файл, уже откомпилированный проект, то для успешного экспорта в Microsoft Excel достаточно, чтобы на компьютере был установлен Microsoft Office.

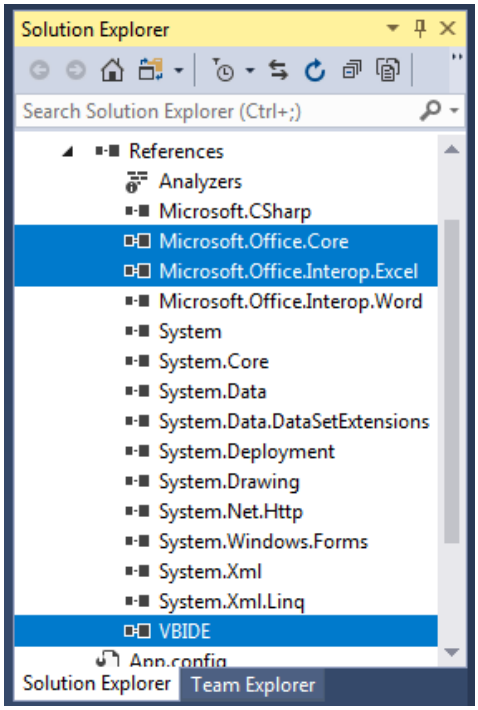
Если вы на другом компьютере открываете проект в Visual Studio, чтобы продолжить его редактировать, то возможны два варианта:

1) На новом компьютере установлена та же самая версия Microsoft Office, что и на старом. В этом случае нет никаких проблем. Вы открываете свой проект и продолжаете его редактировать.

2) На новом компьютере установлена другая версия Microsoft Office. В этом случае при попытке запустить проект из Visual Studio вы получите сообщение об ошибке. Ошибка появляется из-за того, что Visual Studio не может найти библиотеку для работы с Microsoft Excel, которая была на старом компьютере и которой нет на новом компьютере.

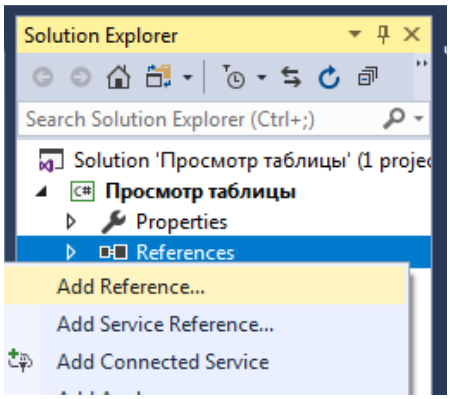
Если версия Microsoft Office на новом компьютере другая, то сделайте следующее:

Шаг 1. В панели Solution Explorer разверните узел References.

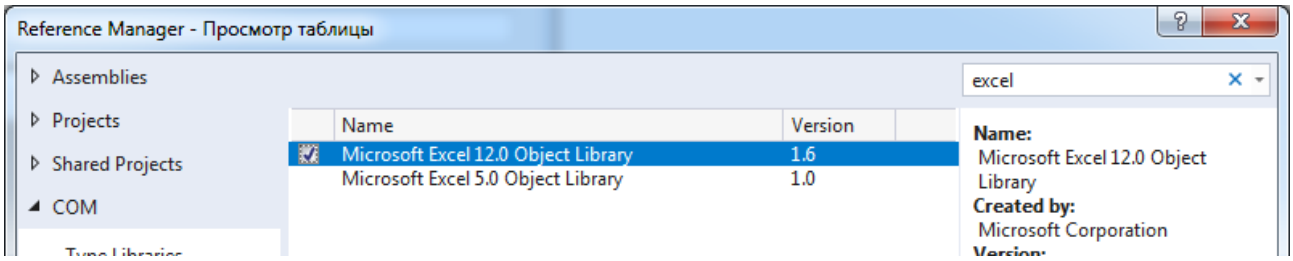


Удалите библиотеки с именами: Microsoft.Office.Core, Microsoft.Office.Interop.Excel, VBIDE.

Шаг 2. В панели Solution Explorer щелкните правой кнопкой мыши на узле References и выберите пункт Add Reference...

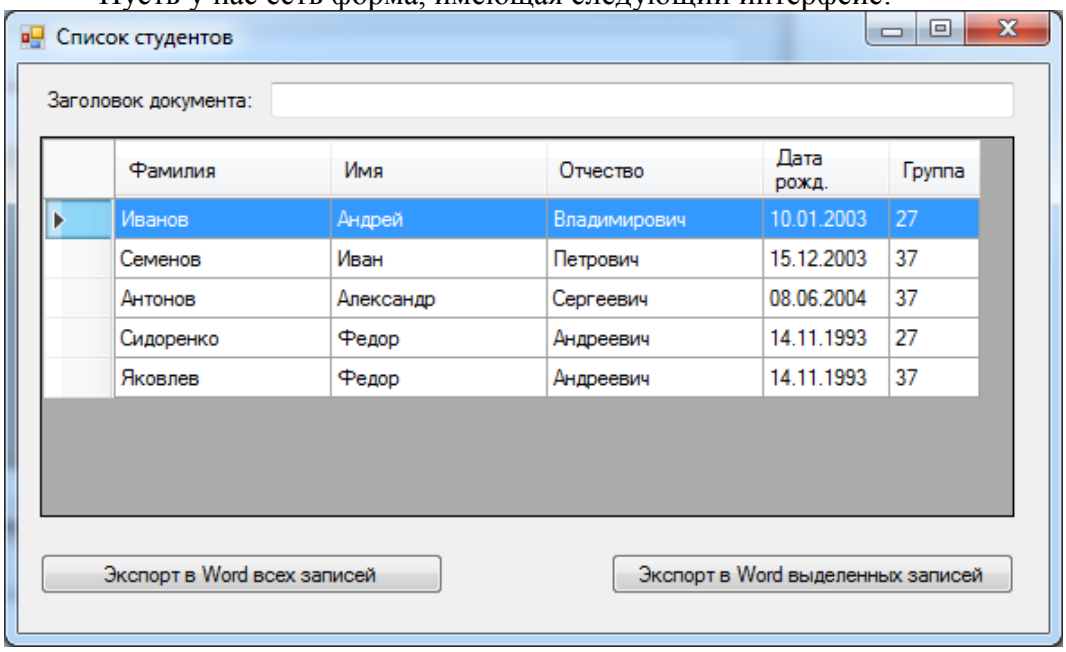


В разделе COM выполните поиск библиотеки с именем Excel. Если будут найдены библиотеки нескольких версий, выберите версию с наибольшим номером. Выделите ее и нажмите ОК.



Тема 12. Модуль экспорта данных в Microsoft Word

Пусть у нас есть форма, имеющая следующий интерфейс:



DataGridView имеет имя dgvStud, текстовое поле для заголовка документа - tbxHead, кнопки для экспорта - btnToWordAll и btnToWordSelect. DataGridView отображает данные, взятые из таблицы базы данных, либо добавленные вручную из программного кода.

Необходимо по нажатию на кнопку «Экспорт в Word всех записей» сформировать документ Word, в который вывести заголовок документа и все данные из таблицы. По нажатию на кнопку «Экспорт в Word выделенных записей» нужно делать то же самое, но не для всех записей, а только для выделенных.

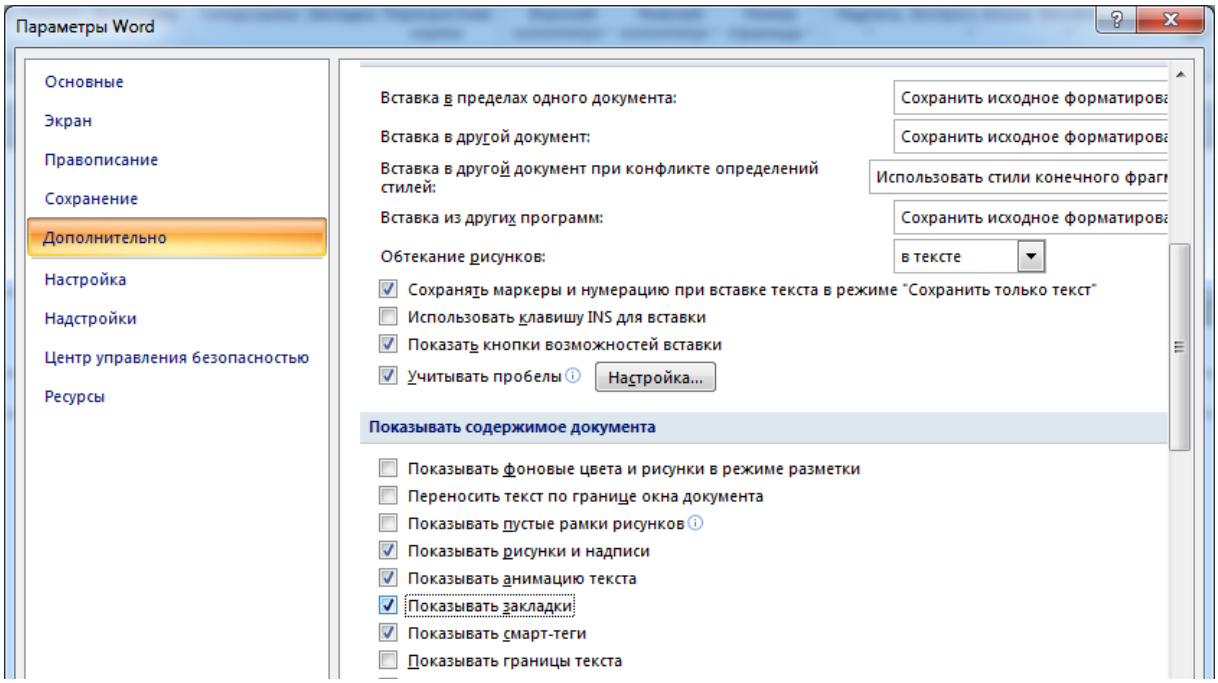
12.1. Создание шаблона для экспорта в Microsoft Word

В текстовом редакторе Microsoft Word нужно создать документ - заготовку, в которую мы будем выводить данные из нашего приложения.

Шаг 1. В редакторе Microsoft Word создайте новый текстовый документ. Добавьте в него таблицу.

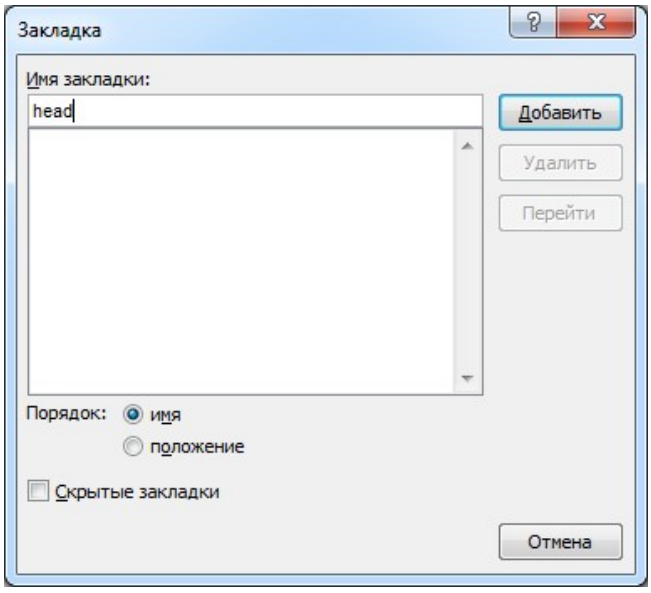
№ п/п	Фамилия	Имя	Отчество	Дата рождения	Группа

Шаг 2. Чтобы выводить данные в текстовый документ, нужно отметить те места в документе, к которым будем обращаться из программного кода. Для этого используются закладки. По умолчанию в настройках Microsoft Word отображение закладок отключено. На вкладке Файл откройте параметры Word. В параметрах откройте Дополнительно, и в разделе «Показывать содержимое документа» установите флажок «Показывать закладки».



Шаг 3. Добавьте закладки в шаблон нашего текстового документа.

Сдвиньте таблицу на две строки вниз. Поставьте курсор в пустую строку перед таблицей. На вкладке Вставка выберите Закладка. Введите имя закладки head и нажмите Добавить.



Поставьте курсор в пустую ячейку таблицы для порядкового номера. На вкладке Вставка выберите Закладка. Введите имя закладки num и нажмите Добавить.

Аналогичным образом добавьте закладки в пустые ячейки таблицы для всех остальных полей: fam, name, otch, bday, group.

В текстовом документе должны появиться пометки в тех местах, куда были добавлены закладки.

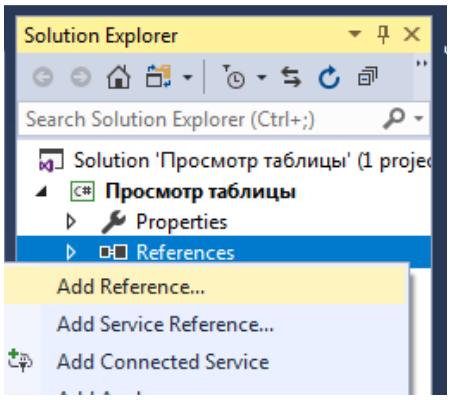
№ п/п	Фамилия	Имя	Отчество	Дата рождения	Группа

Теперь в настройках Word отображение закладок можно выключить.

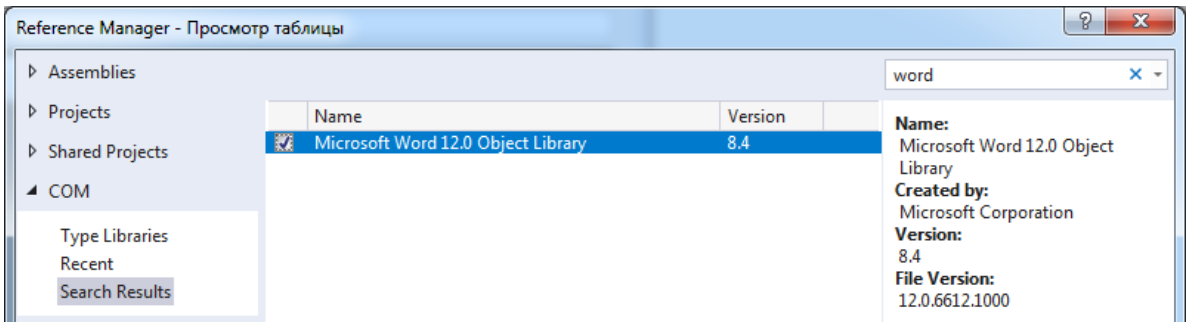
Шаг 4. Сохраните текстовый документ с закладками в папку с exe-файлом приложения, в подкаталог bin \ debug.

12.2. Подключение к приложению библиотеки для работы с Microsoft Word

Шаг 1. В панели Solution Explorer щелкните правой кнопкой мыши на узле References и выберите пункт Add Reference...



В разделе COM выполните поиск библиотеки с именем Word. Если будут найдены библиотеки нескольких версий, выберите версию с наибольшим номером. Выделите ее и нажмите ОК.



Шаг 2. Переключитесь в программный код и подключите пространство имен для работы с Microsoft Word.

```
// подключить для работы с Microsoft Word
using Word = Microsoft.Office.Interop.Word;
```

12.3. Экспорт в Word всех записей

Для кнопки «Экспорт в Word всех записей» опишите следующий код:

```
private void btnToWordAll_Click(object sender, EventArgs e)
{
    // запустить приложение Word
    Word.Application app = new Word.Application();
    // открыть в нем шаблон
    Word.Document doc = app.Documents.Open(Application.StartupPath + "\\Студенты.docx",
      Type.Missing, true);
    // сделать видимым Word
    app.Visible = true;

    // wBookmarks содержит все закладки в документе, открытом в Word
    Word.Bookmarks wBookmarks = doc.Bookmarks;
```

Здесь вначале создается объект Word.Application. При создании этого объекта запускается приложение Microsoft Word. Но на экране это окно этого приложения пока не отображается. С помощью метода Documents.Open() в редакторе Microsoft.Word открывается текстовый документ. Первым параметром указывается имя файла. В нашем случае это файл «Студенты.docx», который находится в папке с exe-файлом приложения. Третий параметр - true - обозначает, что файл нужно открыть в режиме «только для чтения». Это нужно для того, чтобы пользователь не испортил шаблон, нажав в Microsoft Word на кнопку Сохранить.

Далее приложение Microsoft Word делается видимым. Нам нужно выводить данные в закладки. Все закладки текстового документа находятся в doc.Bookmarks, где doc - это объект, соответствующий открытому документу в Microsoft Word. Мы сохраним все закладки документа в переменной wBookmarks.

```
wBookmarks["head"].Range.Text = tbxHead.Text;  // заголовок
// переменные, в которых будут накапливаться данные для вывода в Word
string txtnum = "", fam = "", name = "",
    otch = "", bday = "", group = "";
int num = 1;  // счетчик для нумерации записей в Word
```

Чтобы вывести какой-то текст в закладку с определенным именем, используется следующая форма записи:

```
wBookmarks["имя_закладки"].Range.Text = "текст для вывода в документ";
```

В нашем случае мы в закладку с именем head выводим заголовок документа из текстового поля на форме tbxHead.

Затем мы объявляем шесть строковых переменных. В них мы будем накапливать текст для вывода в шесть столбцов нашей таблицы в Microsoft.Word: txtnum - порядковый номер, fam, name, otch - фамилия, имя и отчество, bday - дата рождения, group - номер группы. Переменная num будет использоваться для счетчика порядковых номеров. Для каждого нового студента ее значение будет увеличиваться на один.

```
// перебор всех строк в DataGridView
for (int i = 0; i <= dgvStud.RowCount - 1; i++)
{
    // добавить к содержимому переменных порядковый номер
    // очередного студента, его фамилию, имя, отчество,
    // дату рождения и номер группы
    // после информации о каждом студенте делается перевод
    // курсора на новую строку
    txtnum += num + ".\n"; num++;
    fam += dgvStud.Rows[i].Cells[0].Value + "\n";
    name += dgvStud.Rows[i].Cells[1].Value + "\n";
    otch += dgvStud.Rows[i].Cells[2].Value + "\n";
    bday += Convert.ToDateTime(dgvStud.Rows[i].Cells[3].Value).ToShortDateString() + "\n";
    group += dgvStud.Rows[i].Cells[4].Value + "\n";
}
```

Дальше у нас идет цикл for, с помощью которого мы выполняем перебор строк в DataGridView. Для каждой i-й строки мы выполняем следующие действия:

1) К содержимому переменной txtnum мы добавляем порядковый номер очередного студента, за которым ставим точку и символ перевода курсора на новую строку. Счетчик порядковых номеров студентов увеличивается на 1.

2) К содержимому переменной fam добавляется фамилия очередного студента, после которой ставится символ перевода курсора на новую строку. Фамилия студента берется из i-й строки из столбца с номером 0 таблицы DataGridView.

3) То же самое делается для всех остальных полей: имя, отчество, дата рождения, номер группы. Только дату рождения студента мы вначале преобразуем к типу DateTime(), а затем используем метод ToShortDateString(). Это нужно для того, чтобы получить дату в

коротком формате: дд.мм.гггг, где дд -день, мм - месяц, гггг - год. Без использования этого метода вместе с датой будет выводиться еще и время.

После выполнения этого фрагмента программного кода в наших переменных будет записано следующее:

txtnum:	fam:	name:
1.	Иванов	Андрей
2.	Семенов	Иван
3.	Антонов	Александр
4.	Сидоренко	Федор
5.	Яковлев	Федор

и т.д.

Т.е. в этих переменных будет находится текст, готовый для вывода в столбцы таблицы в Microsoft Word.

```
    // выводить данные в таблицу Word по закладкам
    // при выводе убирается символ перевода курсора на новую строку
    // последнего студента
    wBookmarks["num"].Range.Text = txtnum.Substring(0, txtnum.Length - 1);
    wBookmarks["fam"].Range.Text = fam.Substring(0, fam.Length - 1);
    wBookmarks["name"].Range.Text = name.Substring(0, name.Length - 1);
    wBookmarks["otch"].Range.Text = otch.Substring(0, otch.Length - 1);
    wBookmarks["bday"].Range.Text = bday.Substring(0, bday.Length - 1);
    wBookmarks["group"].Range.Text = group.Substring(0, group.Length - 1);
}
```

Будем выводить содержимое этих переменных в закладки документа в Microsoft Word. Учитывайте, что конце текста каждой из переменных будет находится символ перевода курсора на новую строку. Он в конце текста является лишним. Чтобы избавиться от последнего символа, будем использовать метод Substring(), с помощью которого будем брать все символы от нулевого до предпоследнего.

Запустите приложение. В текстовое поле введите заголовок «Студенты ГАПОУ НППК» и нажмите на кнопку «Экспорт в Word всех записей». В редакторе Word должен сформироваться следующий документ:

Студенты ГАПОУ НППК

№ п/п	Фамилия	Имя	Отчество	Дата рождения	Группа
1.	Иванов	Андрей	Владимирович	10.01.2003	27
2.	Семенов	Иван	Петрович	15.12.2003	37
3.	Антонов	Александр	Сергеевич	08.06.2004	37
4.	Сидоренко	Федор	Андреевич	14.11.1993	27
5.	Яковлев	Федор	Андреевич	14.11.1993	37

12.4. Экспорт в Word выделенных записей

Шаг 1. Чтобы можно было выделять несколько записей в DataGridView, установите для dgvStud следующие свойства:

MultiSelect - True (разрешить множественное выделение).

SelectionMode - FullRowSelect (выделять в таблице целиком всю строку).

Чтобы выделить в DataGridView несколько строк, выделяйте их, удерживая нажатой клавишу Ctrl.

Шаг 2. Для кнопки «Экспорт в Word выделенных записей» опишите следующий код:

```
private void btnToWordSelect_Click(object sender, EventArgs e)
{
    if (dgvStud.SelectedRows.Count == 0)
    {
        MessageBox.Show("Нет выделенных строк в таблице.");
        return;
    }

    // запустить приложение Word
    Word.Application app = new Word.Application();
    // открыть в нем шаблон
    Word.Document doc = app.Documents.Open(Application.StartupPath + "\\Студенты.docx",
      Type.Missing, true);
    // сделать видимым Word
    app.Visible = true;

    // wBookmarks содержит все закладки в документе, открытом в Word
    Word.Bookmarks wBookmarks = doc.Bookmarks;

    wBookmarks["head"].Range.Text = tbxHead.Text;  // заголовок
    // переменные, в которых будут накапливаться данные для вывода в Word
    string txtnum = "", fam = "", name = "",
        otch = "", bday = "", group = "";
    int num = 1;  // счетчик для нумерации записей в Word
```

Вначале следует проверить, есть ли в DataGridView выделенные строки. Если пользователь не выделил ни одной строки, то экспортировать в Microsoft Word будет нечего. В этом случае нужно вывести текстовое сообщение и выйти из процедуры.

Дальнейший программный код пока без изменений. Он точно такой же, как и для экспорта в Microsoft Word всех записей.

```
    // перебор всех выделенных строк в DataGridView
    for (int i = 0; i <= dgvStud.SelectedRows.Count - 1; i++)
    {
        // добавить к содержимому переменных порядковый номер
        // очередного студента, его фамилию, имя, отчество,
        // дату рождения и номер группы
        // после информации о каждом студенте делается перевод
        // курсора на новую строку
        txtnum += num + ".\n"; num++;
        fam += dgvStud.SelectedRows[i].Cells[0].Value + "\n";
        name += dgvStud.SelectedRows[i].Cells[1].Value + "\n";
        otch += dgvStud.SelectedRows[i].Cells[2].Value + "\n";
        bday += Convert.ToDateTime(dgvStud.SelectedRows[i].Cells[3].Value).
            ToShortDateString() + "\n";
        group += dgvStud.SelectedRows[i].Cells[4].Value + "\n";
    }
```

В цикле for мы будем выполнять перебор не всех строк DataGridView, а только выделенных. Выделенные строки находятся в свойстве SelectedRows. Фамилии, имена и прочие данные студентов мы также будем брать из очередной i-й выделенной строки таблицы.

```
    wBookmarks["num"].Range.Text = txtnum.Substring(0, txtnum.Length - 1);
    wBookmarks["fam"].Range.Text = fam.Substring(0, fam.Length - 1);
    wBookmarks["name"].Range.Text = name.Substring(0, name.Length - 1);
    wBookmarks["otch"].Range.Text = otch.Substring(0, otch.Length - 1);
    wBookmarks["bday"].Range.Text = bday.Substring(0, bday.Length - 1);
    wBookmarks["group"].Range.Text = group.Substring(0, group.Length - 1);
}
```

Программный код для вывода значений строковых переменных в закладки остается без именений.

Шаг 3. Запустите приложение, выделите несколько строк в DataGridView. В текстовое поле введите заголовок «Студенты ГАПОУ НППК» и нажмите на кнопку «Экспорт в Word выделенных записей». У вас должен сформироваться примерно такой документ:

Студенты ГАПОУ НППК

№ п/п	Фамилия	Имя	Отчество	Дата рождения	Группа
1.	Яковлев	Федор	Андреевич	14.11.1993	37
2.	Антонов	Александр	Сергеевич	08.06.2004	37
3.	Иванов	Андрей	Владимирович	10.01.2003	27

12.5. Работа с приложением на другом компьютере

Предположим, что вы начинали работу с приложением, использующим экспорт в Microsoft Word, на одном компьютере, а затем скопировали его на другой компьютер. Рассмотрим, какие при этом могут возникнуть проблемы.

Для того, чтобы можно было выполнять экспорт в Microsoft Word, на компьютере должен быть установлен Microsoft Office. При этом версия Microsoft Office должна быть не ниже 2007. При использовании более старых версий: 2003, 2000 и т.п. могут быть проблемы с экспортом.

Если вы на другом компьютере запускаете exe-файл, уже откомпилированный проект, то для успешного экспорта в Microsoft Word достаточно, чтобы на компьютере был установлен Microsoft Office.

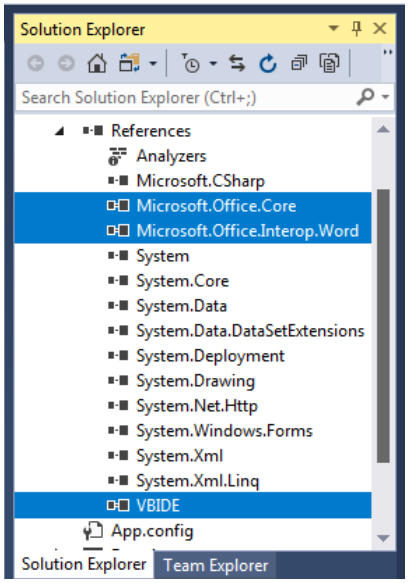
Если вы на другом компьютере открываете проект в Visual Studio, чтобы продолжить его редактировать, то возможны два варианта:

1) На новом компьютере установлена та же самая версия Microsoft Office, что и на старом. В этом случае нет никаких проблем. Вы открываете свой проект и продолжаете его редактировать.

2) На новом компьютере установлена другая версия Microsoft Office. В этом случае при попытке запустить проект из Visual Studio вы получите сообщение об ошибке. Ошибка появляется из-за того, что Visual Studio не может найти библиотеку для работы с Microsoft Word, которая была на старом компьютере и которой нет на новом компьютере.

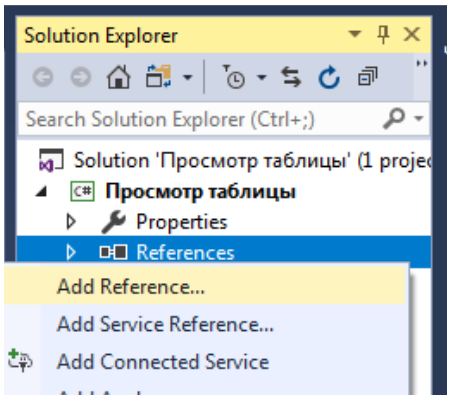
Если версия Microsoft Office на новом компьютере другая, то сделайте следующее:

Шаг 1. В панели Solution Explorer разверните узел References.

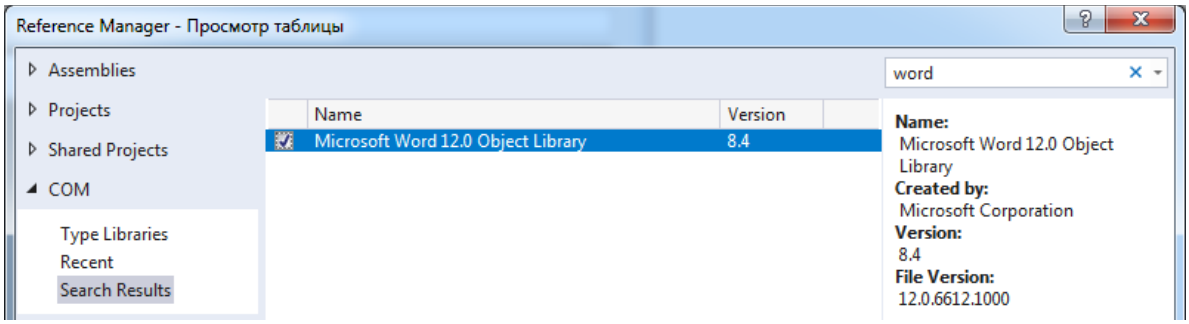


Удалите библиотеки с именами: Microsoft.Office.Core, Microsoft.Office.Interop.Word, VBIDE.

Шаг 2. В панели Solution Explorer щелкните правой кнопкой мыши на узле References и выберите пункт Add Reference...

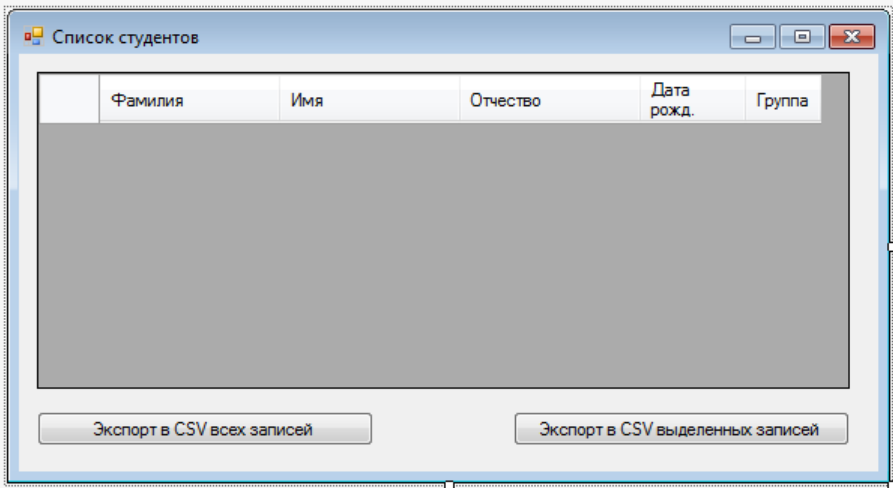


В разделе COM выполните поиск библиотеки с именем Word. Если будут найдены библиотеки нескольких версий, выберите версию с наибольшим номером. Выделите ее и нажмите ОК.



Тема 13. Модуль экспорта данных в файлы формата CSV

Пусть у нас есть форма, имеющая следующий интерфейс:



DataGridView имеет имя dgvStud, текстовое поле для заголовка документа - tbxHead, кнопки для экспорта - btnToCSVAll и btnToCSVSelect. DataGridView отображает данные, взятые из таблицы базы данных, либо добавленные вручную из программного кода.

Необходимо по нажатию на кнопку «Экспорт в CSV всех записей» сформировать файл формата CSV, в который вывести заголовок документа и все данные из таблицы. По нажатию на кнопку «Экспорт в CSV выделенных записей» нужно делать то же самое, но не для всех записей, а только для выделенных.

Так как нам нужно будет указывать имя файла и выбирать место для сохранения файла, добавьте к проекту SaveFileDialog. Файлы формата CSV имеют расширение CSV. Для объекта SaveFileDialog в свойстве DefaultExt укажите расширение по умолчанию csv. При сохранении файла это расширение будет автоматически добавляться к имени файла.

Файлы формата CSV это текстовые файлы, которые имеют следующую структуру:

```
поле1;поле2;поле2
значение_поля1;значение_поля2;значение_поля3
значение_поля1;значение_поля2;значение_поля3
значение_поля1;значение_поля2;значение_поля3
...
```

В первой строке перечисляются имена полей, разделенные символом ";". В остальных строках перечисляются значения полей для отдельных записей, также разделенные символом ";".

Для нашей задачи CSV файл должен выглядеть следующим образом:

```
fam;name;otch;bday;group
Иванов;Андрей;Владимирович;10.01.2003;27
Семенов;Иван;Петрович;15.12.2003;37
...
```

Файлы формата CSV можно открывать в приложении Microsoft Excel. Но при этом следует учитывать, что CSV-файл должен иметь кодировку ANSI (Windows 1251). Если для файла будет использоваться другая кодовая таблица, то в Microsoft Excel текст будет отображаться нечитаемыми символами.

13.1. Экспорт в файл формата CSV всех записей

Подключите пространство имен для работы с файлами:

```
// подключить для работы с файлами
using System.IO;
```

Для кнопки «Экспорт в CSV всех записей» опишите следующий программный код.

```
private void btnToCSVAll_Click(object sender, EventArgs e)
{
    if (saveFileDialog1.ShowDialog() == DialogResult.OK)
    {
        string t;
        // открыть файл для записи с использованием кодировки ANSI (Windows 1251)
        StreamWriter f = new StreamWriter(
            new FileStream(saveFileDialog1.FileName, FileMode.Create),
              Encoding.GetEncoding(1251));
        t = "fam;name;otch;bday;group";
        f.WriteLine(t);  // запись в файл названий полей
```

Вначале открывается диалоговое окно SaveFileDialog. Если пользователь указал имя файла и нажал на кнопку Сохранить, то начинается процесс сохранения данных в файл формата CSV.

С помощью StreamWriter файл, имя которого пользователь указал в диалоговом окне, открывается в режиме для записи. Метод GetEngoding(1251) задает для создаваемого файла кодировку ANSI (Windows 1251).

Первой строкой мы записываем в файл названия полей.

```
        // перебор всех строк в DataGridView с данными студентов
        for (int i = 0; i <= dgvStud.RowCount - 1; i++)
        {
            t = "";
            t += dgvStud.Rows[i].Cells[0].Value + ";";
            t += dgvStud.Rows[i].Cells[1].Value + ";";
            t += dgvStud.Rows[i].Cells[2].Value + ";";
            t += Convert.ToDateTime(dgvStud.Rows[i].Cells[3].Value)
                .ToShortDateString() + ";";
            t += dgvStud.Rows[i].Cells[4].Value;
            // запись в файл данных очередного студента
            f.WriteLine(t);
        }
        f.Close();
    }
}
```

Затем с помощью цикла for выполняется перебор всех строк в DataGridView. Для каждой i-й строки мы берем содержимое столбцов от нулевого до четвертого. Записываем в переменную t содержимое ячеек, отделяя одно значение от другого символом ";". Дату рождения студента мы вначале преобразуем к типу DateTime(), а затем используем метод ToShortDateString(). Это нужно для того, чтобы получить дату в коротком формате: дд.мм.гггг, где дд -день, мм - месяц, гггг - год. Без использования этого метода вместе с датой будет выводиться еще и время. Затем сформированную строку выводим в файл.

По завершении перебора строк в DataGridView файл закрывается.

13.2. Экспорт в файл формата CSV выделенных записей

Подключите пространство имен для работы с файлами:

```
// подключить для работы с файлами
using System.IO;
```

Для кнопки «Экспорт в CSV выделенных записей» опишите следующий программный код.

```
private void btnToCSVSelect_Click(object sender, EventArgs e)
{
    if (dgvStud.SelectedRows.Count == 0)
    {
        MessageBox.Show("Нет выделенных строк в таблице.");
        return;
    }

    if (saveFileDialog1.ShowDialog() == DialogResult.OK)
    {
        string t;
        // открыть файл для записи с использованием кодировки ANSI (Windows 1251)
        StreamWriter f = new StreamWriter(
            new FileStream(saveFileDialog1.FileName, FileMode.Create),
              Encoding.GetEncoding(1251));
        t = "fam;name;otch;bday;group";
        f.WriteLine(t);  // запись в файл названий полей
```

Вначале следует проверить, есть ли в DataGridView выделенные строки. Если пользователь не выделил ни одной строки, то экспортировать в файл формата CSV будет нечего. В этом случае нужно вывести текстовое сообщение и выйти из процедуры.

Дальнейший программный код пока без изменений. Он точно такой же, как и для экспорта в файл формата CSV всех записей.

```
        // перебор всех выделенных строк в DataGridView с данными студентов
        for (int i = 0; i <= dgvStud.SelectedRows.Count - 1; i++)
        {
            t = "";
            t += dgvStud.SelectedRows[i].Cells[0].Value + ";";
            t += dgvStud.SelectedRows[i].Cells[1].Value + ";";
            t += dgvStud.SelectedRows[i].Cells[2].Value + ";";
            t += Convert.ToDateTime(dgvStud.SelectedRows[i].Cells[3].Value)
                .ToShortDateString() + ";";
            t += dgvStud.SelectedRows[i].Cells[4].Value;
            // запись в файл данных очередного выделенного студента
            f.WriteLine(t);
        }
        f.Close();
    }
}
```

В цикле for мы будем выполнять перебор не всех строк DataGridView, а только выделенных. Выделенные строки находятся в свойстве SelectedRows. Фамилии, имена и прочие данные студентов мы также будем брать из очередной i-й выделенной строки таблицы.

Тема 14. Программирование специализированной СУБД «Агентство недвижимости»

1. Создание базы данных, импорт исходных данных.

Ваша задача - разработать информационную систему для агентства недвижимости, которое помогает своим клиентам купить/продать объекты недвижимости (без аренды).

Для работы с информационной системой необходимо выполнить авторизацию. Пользователи могут быть одного из двух типов: клиенты и риелторы. В зависимости от того, кто выполняет авторизацию, необходимо открывать форму, содержащую визуальные объекты для выполнения действий либо клиента, либо риелтора.

Необходимо также предусмотреть регистрацию новых пользователей. Если регистрируется новый пользователь, то у него уровень доступа будет по умолчанию – клиент.

Исходные данные для пользователей находятся в файле «Пользователи.csv». Фотографии пользователей находятся в папках «Клиенты» и «Риелторы».

Клиенты приходят в агентство недвижимости либо с потребностью, либо с предложением. Потребность - желание клиента купить объект недвижимости, соответствующего указанным параметрам.

Предложение - желание клиента продать указанный объект недвижимости за указанную цену.

Сделка - факт осуществления продажи недвижимого имущества. В сделке участвуют две стороны: клиент-покупатель с потребностью и клиент-продавец с предложением.

Продажа объекта недвижимости

1) Клиент обращается в компанию с желанием продать объект недвижимости.
2) «Предложению» клиента назначается ответственный риэлтор. Он осуществляет поиск подходящих потребностей
3) Клиент выбирает потребность и заключается сделка.
4) Система рассчитывает размер комиссии для риелтора клиента-продавца и клиента-покупателя.

Покупка объекта недвижимости

1) Клиент обращается в компанию с желанием купить объект недвижимости.
2) «Потребности» клиента назначается ответственный риэлтор. Он осуществляет поиск подходящих предложений
3) Клиент выбирает предложение и заключается сделка.
4) Система рассчитывает размер комиссии для риелтора клиента-продавца и клиента-покупателя.

Для потребностей и предложений исходные данные находятся в файле «Потребности и предложения.xlsx».

На основании описания задания и анализа исходных данных создайте базу данных, сформируйте связи между таблицами, импортируйте исходные данные в таблицы базы данных. При необходимости выполните дополнительную обработку исходных данных перед импортом.

Для разрабатываемого приложения также необходимо обеспечить хранение следующей информации:

- Максимальное количество клиентов, которых может обслуживать один риелтор одновременно.
- Процент от сделки риелтору клиента с предложением.
- Процент от сделки риелтору клиента с потребностью.

Эти данные являются общими для всех риелторов.

2. Регистрация и авторизация пользователей.

Регистрация нового пользователя.

При регистрации нового пользователя нужно вводить фамилию, имя, логин, пароль. Логин пользователя должен быть уникальным. Пароль должен отвечать следующим требованиям безопасности:

- длина пароля – от 6 до 18 символов;
- не должно быть 3 и более подряд идущих одинаковых символа;
- должен содержать хотя бы одну цифру и хотя бы одну букву;
- должен содержать минимум 1 символ из набора: * & { } | +.

Если введенный пользователем пароль не соответствует требованиям безопасности, то вывести на экран соответствующее сообщение и не допускать регистрацию такого пользователя.

Проверку допустимости пароля реализуйте с помощью подключаемой DLL-библиотеки.

Зарегистрированный пользователь по умолчанию является клиентом.

Авторизация пользователя.

Пользователи могут быть одного из двух типов: клиенты и риелторы. В зависимости от того, кто выполняет авторизацию, необходимо открывать форму, содержащую визуальные объекты для выполнения действий либо клиента, либо риелтора.

Помимо логина и пароля пользователь должен верно ввести капчу, которая представляет собой набор из 4 символов, сгенерированных случайным образом. Капча должна включать хотя бы одну латинскую букву и хотя бы одну цифру. Капча должна быть выведена особым шрифтом, который сложно распознать роботам, например Curlz MT.

3. Клиенты агентства недвижимости.

Для клиентов необходимо реализовать выполнение следующих операций:

1. Просмотр и редактирование профиля текущего пользователя.

Предусмотреть возможность изменения фамилии, имени, пароля, фотографии. В целях безопасности не отображать пароль пользователя на экране. Разрешать смену пароля только в том случае, если пользователь укажет старый пароль и новый пароль. Если пароль не изменяется, то выполнять сохранение без запроса пароля.

2. Просмотр списка клиентов.

Выведите список клиентов на экран без возможности изменения. Реализуйте быстрый поиск по первым буквам фамилии.

Реализуйте также нечёткий поиск клиентов по фамилии и имени. Для нечеткого поиска используйте расстояние Левенштейна.

Расстояние Левенштейна - это минимальное количество операций вставки одного символа, удаления одного символа и замены одного символа на другой, необходимых для превращения одной строки в другую. Например:

Расстояние между одинаковыми строками равно 0

Расстояние между строками «строка» и «собака» равно 3, замены: «т» на «о», «р» на «б» и «о» на «а».

Расстояние между строками «строка» и «вафля» равно 6, необходимо заменить все 5 символов и удалить еще 1 лишний.

Поэтому при нечетком поиске с использованием слова «строка» в результирующую выборку должны попасть слова «строка», «собака», но не слово «вафля».

Используйте для поиска расстояние Левенштейна не превышающее 3.

3. Добавление потребности.

Реализуйте добавление потребности клиента. При создании потребности предусмотрите назначение ответственного риелтора. Реализуйте выбор риелтора из выпадающего списка. Учитывайте, что для риелтора существует максимальное количество одновременно обслуживаемых клиентов.

4. Добавление предложения.

Реализуйте добавление предложения клиента. При создании предложения предусмотрите назначение ответственного риелтора. Реализуйте выбор риелтора из выпадающего списка. Учитывайте, что для риелтора существует максимальное количество одновременно обслуживаемых клиентов.

5. Просмотр потребностей и предложений текущего клиента.

Реализуйте просмотр списка потребностей и предложений авторизовавшегося клиента. Предусмотрите возможность отображения всех потребностей и предложений, либо только незавершенных.

4. Риелторы агентства недвижимости.

Для риелторов необходимо реализовать выполнение следующих операций:

1. Просмотр и редактирование профиля текущего пользователя.

Предусмотреть возможность изменения фамилии, имени, пароля, фотографии. В целях безопасности не отображать пароль пользователя на экране. Разрешать смену пароля только в том случае, если пользователь укажет старый пароль и новый пароль. Если пароль не изменяется, то выполнять сохранение без запроса пароля.

2. Просмотр списка риелторов.

Выведите список риелторов на экран без возможности изменения. Реализуйте быстрый поиск по первым буквам фамилии.

Реализуйте также нечёткий поиск риелторов по фамилии и имени. Для нечеткого поиска используйте расстояние Левенштейна. (См. описание расстояния Левенштейна в п. 3. Клиенты агентства недвижимости).

Расстояние Левенштейна реализуйте с помощью подключаемой DLL-библиотеки. Используйте для поиска расстояние Левенштейна не превышающее 3.

4. Потребности клиентов риелтора.

Реализуйте отображения списка потребностей, для которых данный риелтор назначен ответственным. Для каждой потребности предусмотрите вывод списка предложений, подходящих для совершения сделки.

Предложение считается подходящим для совершения сделки, если площадь, этаж, количество комнат и цена объекта недвижимости укладывается в диапазон между минимальным и максимальным значениями.

5. Совершение сделки.

При выборе одного из подходящих предложений для совершения сделки и нажатия на кнопку «Совершить сделку», необходимо закрыть потребность и предложение, вычислить комиссию каждому риэлтору. Сумма комиссии добавляется к цене объекта недвижимости. На экран необходимо вывести сумму, которую должен заплатить клиент-покупатель, сумму

комиссии для риелтора клиента-продавца и сумму комиссии для риелтора клиента-покупателя.

6. Создание отчетов.

Выполните экспорт в Excel данных по закрытым сделкам для данного риелтора. Сделайте вывод в отчет фамилии клиента-продавца, фамилии клиента-покупателя, даты совершения сделки.

Реализуйте вывод в отчет данных по сделкам, совершенным за определенный период (начальная и конечная дата).

14.1. Создание базы данных, импорт исходных данных, авторизация, регистрация и редактирование профиля

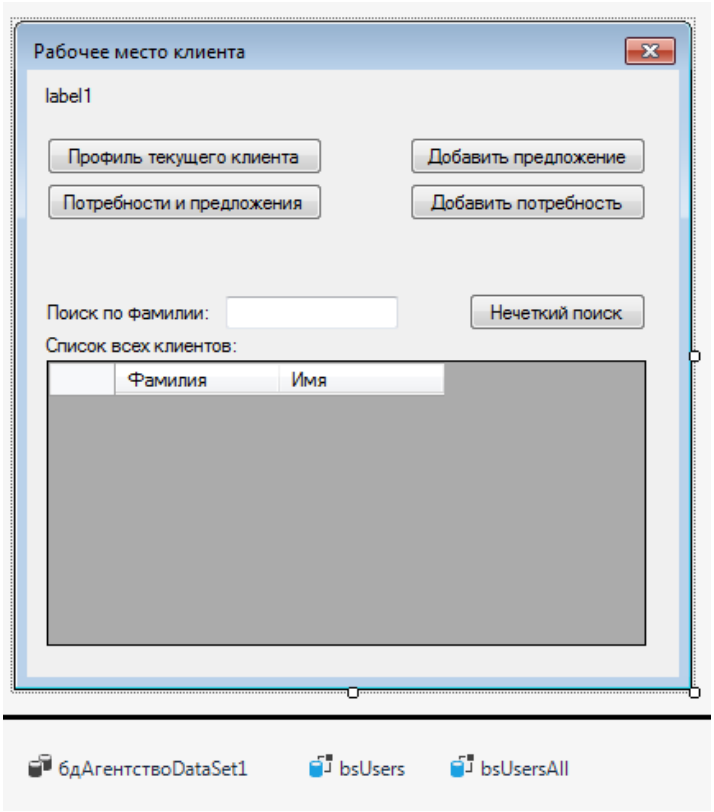
Для создания базы данных, импорта исходных данных, формирования интерфейса и описания программного кода для авторизации, регистрации пользователей и редактирования профиля пользователей читайте материал, изложенный в предыдущих темах.

14.2. Формирование интерфейса и описание программного кода для рабочего места клиента

Шаг 1. Если в вашем проекте еще нет формы для рабочего места клинета, то добавьте к проекту новую форму. Дайте ей имя FormKlient. Установите для формы следующие значения свойств:

Text - Рабочее место клиента (заголовок формы)
StartPosition - CenterParent (отображать форму по центру родительской формы)
MaximizeBox - False (заблокировать кнопку разворачивания формы до максимума)
MinimizeBox - False (заблокировать кнопку сворачивания формы до минимума)
FormBorderStyle - FixedDialog (диалоговое окно без кнопки системного меню, не разрешать изменять размеры формы)

Шаг 2. Сформируйте следующий интерфейс формы:



Если вы сделали авторизацию пользователей с различными правами так, как описано в теме №6, то на вашей форме уже будут находится объекты: DataSet, bsUsers (связанный с таблицей Users), label1 (связанный с полем fio из таблицы Users).

У кого нет этих объектов, добавьте их на форму и настройте так, как описано в теме №6.

Для кнопок «Профиль текущего клиента», «Потребности и предложения», «Добавить потребность», «Добавить предложение», «Нечеткий поиск» используйте объекты Button. Дайте им имена btnProfilKlient, btnPotrebPredlKlient, btnAddPredl, btnAddPotrebn, и btnFindLeven.

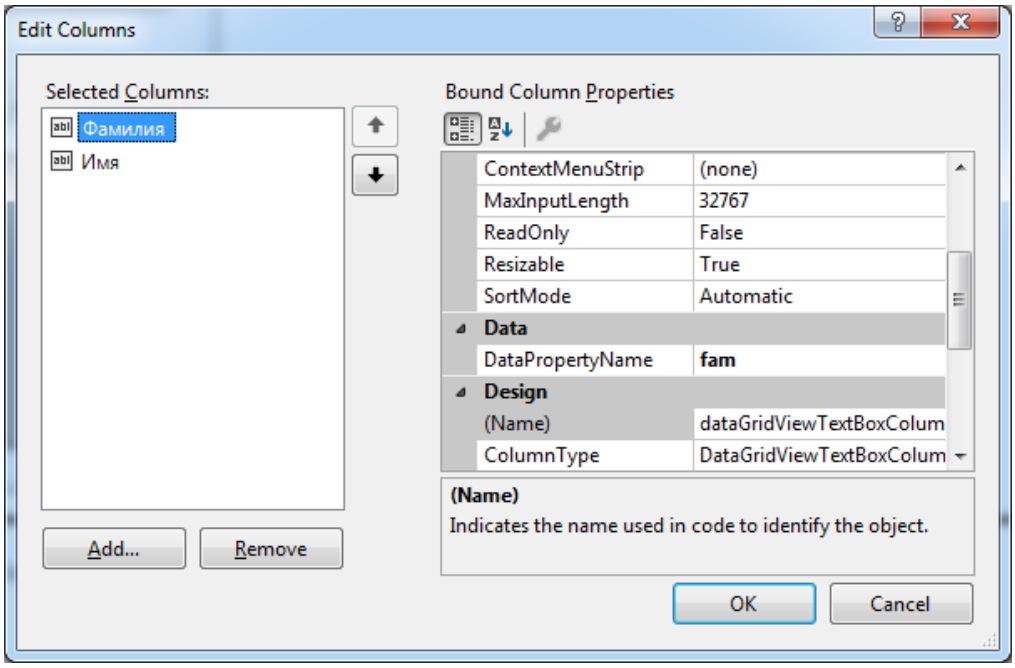
Для текстового поля поиска по фамилии дайте имя tbxFamFind.

Шаг 3. Настройка BindingSources и DataGridView для отображения списка клиентов.

bsUsersAll свяжите с таблицей Users базы данных. Это делается с помощью свойств DataSource и DataMember. В таблице Users у нас хранятся все пользователи - и клиенты и риелтеры. Нам нужен только список клиентов. Тип пользователя определяется с помощью поля Type в таблице Users. Если в этом поле установлено значение 1, то это клиент, если установлено значение 0, то пользователь является риелтером. В свойстве Filter для bsUsersAll напишите фильтр: «type = 1». Тем самым мы отфильтруем пользователей-клиентов.

Свяжите DataGridView с bsUsersAll. Это можно сделать с помощью свойства DataSource или с помощью кнопки со стрелкой в правом верхнем углу DataGridView.

Щелкните правой кнопкой на DataGridView и выберите пункт Edit Columns...



С помощью кнопки Remove удалите лишние столбцы. Для столбцов с фамилиями и именами в свойстве Header Text напишите заголовок.

После того, как закончите формирование столбцов, задайте для DataGridView следующие свойства:
AllowUserToAddRows - False (чтобы пользователь не мог добавлять строки в таблицу)
AllowUserToDeleteRows - False (чтобы пользователь не мог удалять строки из таблицы)
ReadOnly - True (чтобы пользователь не мог редактировать ячейки таблицы)
SelectionMode - FullRowSelect (чтобы при выделении в таблице выделялась сразу вся строка)

Шаг 4. При вводе символов в текстовое поле tbxFamFind нужно отфильтровывать тех студентов, чьи фамилии начинаются с указанных символов. Для этого в обработчике события TextChanged укажите следующий код:

```
private void tbxFamFind_TextChanged(object sender, EventArgs e)
{
    // поиск по первым буквам фамилии
    bsUsersAll.Filter = String.Format("type = 1 and fam LIKE '{0}%'", tbxFamFind.Text);
}
```

В условии для фильтрации сказано: тип пользователя равен 1 (клиент) и фамилия пользователя начинается с символов, введенных в текстовое поле tbxFamFind.

Тема 15. Программирование специализированной СУБД «Марафон»

Тема 16. Программирование специализированной СУБД «Швейная фабрика»

1. Создание базы данных, импорт исходных данных.

Ваша задача - разработать информационную систему «Швейная фабрика» для клиентов и работников фабрики. Информационная система должна хранить информацию о используемых материалах (ткани, фурнитура), пользователях информационной системы.

Для работы с информационной системой необходимо выполнить авторизацию. Пользователи могут быть одного из трех типов: заказчики, менеджеры и кладовщики. В зависимости от того, кто выполняет авторизацию, необходимо открывать форму, содержащую визуальные объекты для выполнения действий соответствующего пользователя.

Для пользователей-заказчиков информационная система должна предоставлять возможность сформировать заказ на изготовление изделий с возможностью выбора материалов, используемых в производстве изделия. Для пользователей-кладовщиков информационная система должна предоставлять возможность управлять материалами на складе (просмотр, прием и списание). Для пользователей-менеджеров информационная система должна предоставлять возможность создавать и редактировать изделия, которые будет выбирать заказчик.

Необходимо также предусмотреть регистрацию новых пользователей-заказчиков.

Исходные данные для пользователей находятся в файле «Пользователи.csv». Данные о тканях находятся в файле «Ткани.xlsx», данные о фурнитуре – в файле «Фурнитура.sql», данные об изделиях – в файле «Изделия.json». Фотографии тканей и фурнитур находятся в архиве «Фото.zip».

На основании описания задания и анализа исходных данных создайте базу данных, сформируйте связи между таблицами, импортируйте исходные данные в таблицы базы данных. При необходимости выполните дополнительную обработку исходных данных перед импортом.

2. Регистрация и авторизация пользователей.

Регистрация нового пользователя.

Логин пользователя должен быть уникальным. Пароль должен отвечать следующим требованиям безопасности:

- длина пароля – минимум 6 символов;
- обязательно и строчные и прописные символы;
- цифр должно быть не более половины от всех символов пароля;
- должен содержать минимум 1 символ из набора: ! @ # $ % ^.

Если введенный пользователем пароль не соответствует требованиям безопасности, то вывести на экран соответствующее сообщение и не допускать регистрацию такого пользователя.

Проверку допустимости пароля реализуйте с помощью подключаемой DLL-библиотеки.

При регистрации новых пользователей предусмотрите ввод пароля еще раз для подтверждения. Пароль должен быть скрыт и не отображаться на экране. Однако предусмотрите также возможность скрыть или показать пароль при необходимости.

При регистрации нового заказчика нужно вводить фамилию, имя, отчество, логин, пароль и номер телефона.

Номер телефона должен вводиться по шаблону +X(XXX) XXX-XX-XX, где X – десятичная цифра

Авторизация пользователя.

Пользователи могут быть одного из трех типов: заказчики, менеджеры и кладовщики. В зависимости от того, кто выполняет авторизацию, необходимо открывать форму, содержащую визуальные объекты для выполнения действий соответствующего пользователя.

Пароль должен быть скрыт и не отображаться на экране. Однако предусмотрите также возможность скрыть или показать пароль при необходимости.

При авторизации необходимо предусмотреть ввод капчи. Капча должна состоять из четырех символов, определяемых случайным образом. Символы могут быть большими буквами латинского алфавита, либо цифрами. При этом обязательно должна быть хотя бы одна буква и хотя бы одна цифра. Капча должна отображаться нестандартным шрифтом, затрудняющим распознавание капчи. Поверх капчи нужно вывести несколько линий случайного цвета со случайными координатами.

3. Заказчик.

Для пользователей-заказчиков необходимо реализовать выполнение следующих операций:

6. Отображение фамилии, имени, отчества авторизовавшегося заказчика.

7. Просмотр и редактирование профиля текущего пользователя.

   Предусмотреть возможность изменения фамилии, имени, отчества, телефона, пароля. В целях безопасности не отображать пароль пользователя на экране. Разрешать сохранение изменений в профиле только в том случае, если пользователь укажет старый пароль и новый пароль.

8. Просмотр списка тканей, фурнитур, изделий (которые может изготовить фабрика), заказов, которые создавал ранее данный заказчик.
   Эти данные выводите на экран без возможности редактирования.

9. Создание нового заказа.
   При создании нового заказа пользователь выбирает изделие, ткань для этого изделия, фурнитуру и количество фурнитуры, указывает количество изделий, которые нужно изготовить.
   Исходя из длины ширины ткани, которая нужна для изготовления изделия, необходимо вычислить, сколько метров ткани потребуется для изготовления данного количества изделий. При этом раскрой ткани должен выполняться наиболее экономным образом. Остатки ткани выбрасываются, обратно на склад не возвращаются.
   При формировании заказа автоматически выбирается наименее загруженный менеджер (тот, у которого меньше всего заказов).

10. Поиск.

Предусмотреть возможность поиска в списке тканей по названию ткани с учетом расстояния Левенштейна.

Расстояние Левенштейна - это минимальное количество операций вставки одного символа, удаления одного символа и замены одного символа на другой, необходимых для превращения одной строки в другую. Например:

Расстояние между одинаковыми строками равно 0

Расстояние между строками «строка» и «собака» равно 3, замены: «т» на «о», «р» на «б» и «о» на «а».

Расстояние между строками «строка» и «вафля» равно 6, необходимо заменить все 5 символов и удалить еще 1 лишний.

Поэтому при нечетком поиске с использованием слова «строка» в результирующую выборку должны попасть слова «строка», «собака», но не слово «вафля». Используйте для поиска расстояние Левенштейна не превышающее 3.

4. Кладовщики.

Для кладовщиков необходимо реализовать выполнение следующих операций:

1. Отображение фамилии, имени, отчества авторизовавшегося кладовщика.

2. Просмотр и редактирование профиля текущего пользователя.
   Предусмотреть возможность изменения фамилии, имени, отчества, номера телефона, пароля. В целях безопасности не отображать пароль пользователя на экране. Разрешать сохранение изменений в профиле только в том случае, если пользователь укажет старый пароль и новый пароль.

3. Просмотр списка тканей, фурнитур, заказов (всех, для всех заказчиков).
   Эти данные выводите на экран без возможности редактирования.

4. Прием на склад тканей и фурнитур.
   Предусмотреть возможность приема сразу нескольких тканей, или фурнитур. Разработайте документ для отражения факта поступления материалов от поставщиков. В одном документе может быть отражен факт поступления большого количества разных материалов от одного поставщика. В документе пользователь должен иметь возможность указать закупаемые материалы, их количества, название, цвет, фото, ширину, длину.

   После принятия документа к учёту он не может быть изменен. Для работы с документом разработайте отдельную форму.

5. Списание тканей и фурнитур.
   Реализуйте возможность списания тканей и фурнитур, находящихся на складе. Пользователь должен иметь возможность выбрать материалы на складе и сформировать документ, в котором указываются данные списываемых материалов, их количества, площадь тканей. Этот документ нужно выводить в виде таблицы в Microsoft Excel.

5. Менеджеры.

Для менеджеров необходимо реализовать выполнение следующих операций:

1. Отображение фамилии, имени, отчества авторизовавшегося менеджера.

2. Просмотр и редактирование профиля текущего пользователя.
   Предусмотреть возможность изменения фамилии, имени, отчества, пароля. В целях безопасности не отображать пароль пользователя на экране. Разрешать сохранение изменений в профиле только в том случае, если пользователь укажет старый пароль и новый пароль.

3. Просмотр списка тканей, фурнитур, заказов (только тех заказов, обработкой которых занимается данный менеджер).
   Эти данные выводите на экран без возможности редактирования.

4. Создание и редактирование изделий, которые может изготавливать фабрика.

5. Экспорт данных о заказах в Microsoft Word и файл формата csv.
   Менеджер должен иметь возможность выводить данные о всех, или некоторых своих заказах в документ Microsoft Word, либо в файл формата csv (по своему выбору).

16.1. Создание базы данных, импорт исходных данных, авторизация, регистрация и редактирование профиля

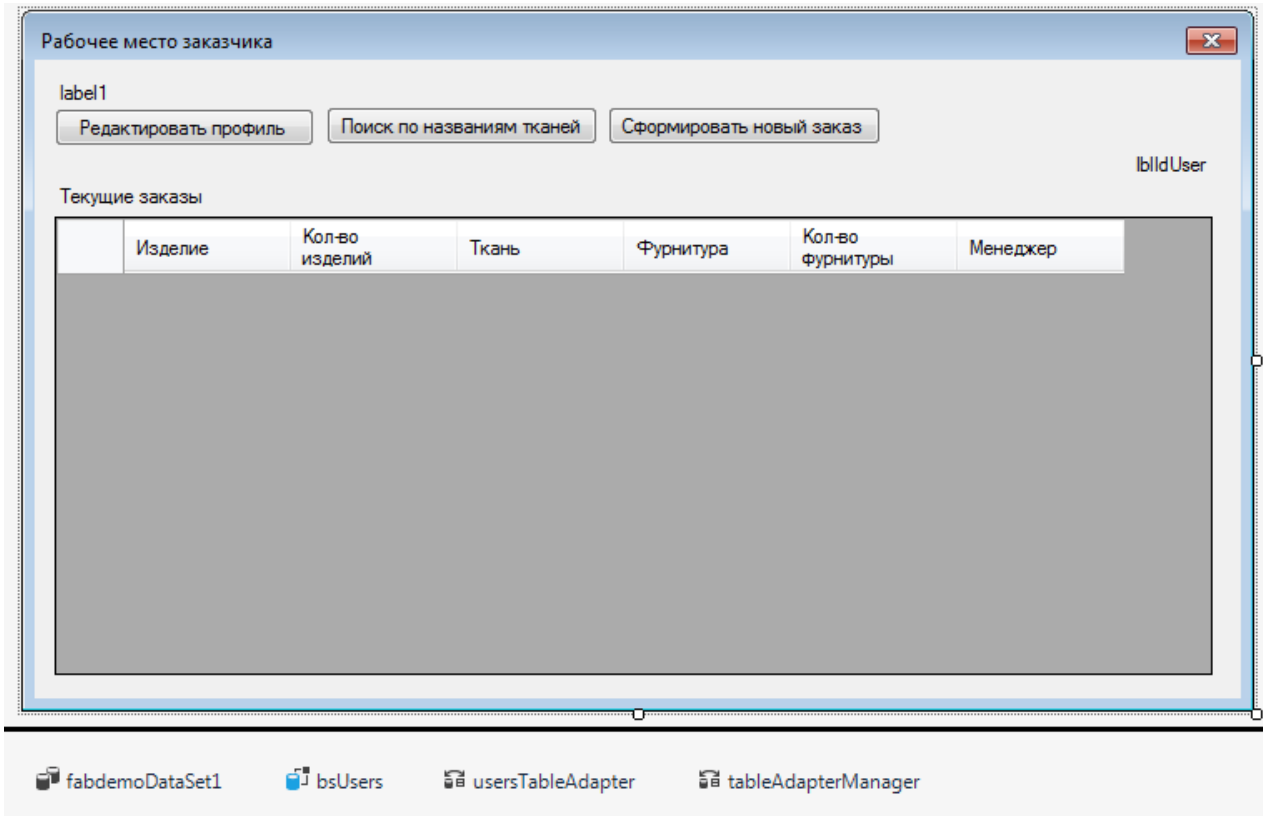
Для создания базы данных, импорта исходных данных, формирования интерфейса и описания программного кода для авторизации, регистрации пользователей и редактирования профиля пользователей читайте материал, изложенный в предыдущих темах.

16.2. Формирование интерфейса и описание программного кода для рабочего места заказчика

Шаг 1. Если в вашем проекте еще нет формы для рабочего места заказчика, то добавьте к проекту новую форму. Дайте ей имя FormZakazchik. Установите для формы следующие значения свойств:

Text - Рабочее место заказчика (заголовок формы)
StartPosition - CenterParent (отображать форму по центру родительской формы)
MaximizeBox - False (заблокировать кнопку разворачивания формы до максимума)
MinimizeBox - False (заблокировать кнопку сворачивания формы до минимума)
FormBorderStyle - FixedDialog (диалоговое окно без кнопки системного меню, не разрешать изменять размеры формы)

Шаг 2. Сформируйте следующий интерфейс формы:



Если вы сделали авторизацию пользователей с различными правами так, как описано в теме №6, то на вашей форме уже будут находится объекты: DataSet, bsUsers (связанный с таблицей Users), label1 (связанный с полем fio из таблицы Users).

У кого нет этих объектов, добавьте их на форму и настройте так, как описано в теме №6.

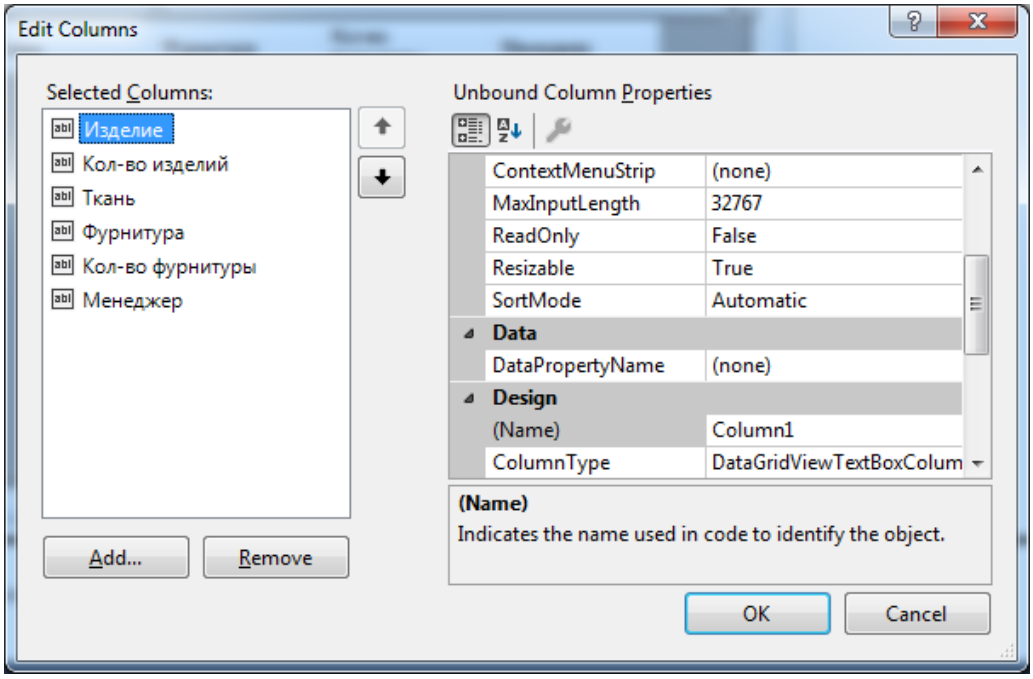
Для кнопок «Редактировать профиль», «Поиск по названиям тканей», «Сформировать новый заказ» используйте объекты Button. Дайте им имена btnProfile, btnLevenshtein и btnAddZakaz.

Для DataGridView дайте имя dgvZakaz. Но не связывайте его с какой-либо таблицей из базы данных. Мы будем заполнять его вручную из программного кода.

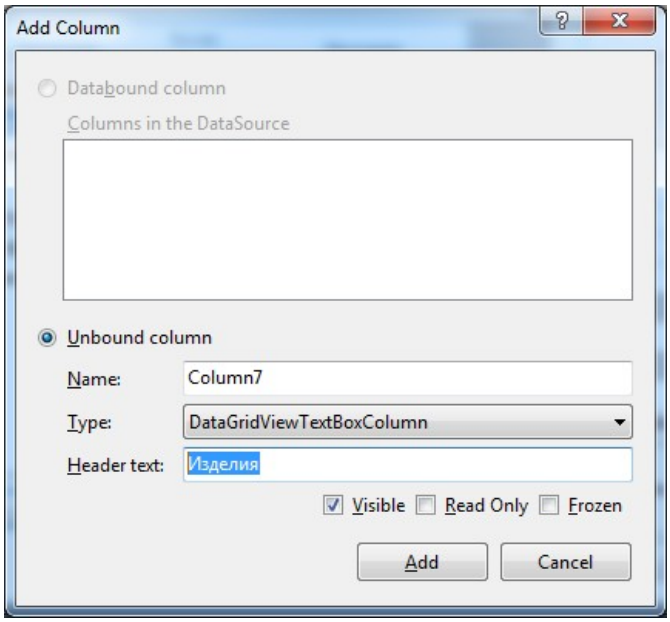
В метке lblIdUser у нас будет храниться id заказчика, для которого нужно отобразить список существующих заказов и сформировать новый заказ. Свяжите эту метку с полем iduser таблицы Users. В свойстве (DataBindings)►Text установите значение bsUsers - iduser. Однако, сама метка должна быть невидимой. Ее содержимое будет использоваться только в программном коде. Для этой метки сделайте цвет текста и цвет фона (свойства ForeColor и BackColor) одинаковыми. Если вы попытаетесь в свойстве Visible для нее установите значение False, то такой прием будет работать не всегда. Во многих случаях метки, имеющие в свойстве Visible значение False, не могут хранить данные из таблиц базы данных.

Шаг 3. Настройка DataGridView.

Щелкните правой кнопкой на DataGridView и выберите пункт Edit Columns...



С помощью кнопки Add добавьте в DataGridView пять столбцов, непривязанных к базе данных.



Для каждого столбца в свойстве Header Text напишите заголовок.

После того, как закончите формирование столбцов, задайте для DataGridView следующие свойства:
AllowUserToAddRows - False (чтобы пользователь не мог добавлять строки в таблицу)
AllowUserToDeleteRows - False (чтобы пользователь не мог удалять строки из таблицы)
ReadOnly - True (чтобы пользователь не мог редактировать ячейки таблицы)
SelectionMode - FullRowSelect (чтобы при выделении в таблице выделялась сразу вся строка)

Шаг 4. Далее нам нужно работать с базой данных с помощью SQL-запросов. Сделаем на главной форме строковую переменную, в которой будет храниться строка подключения. Эта переменная будет с глобальным доступом и к ней можно будет обратиться с любой формы.

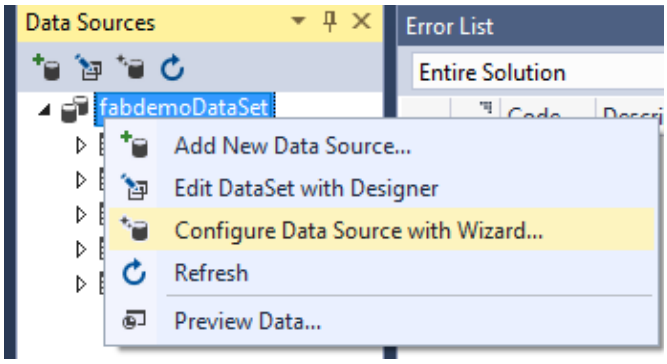
Переключитесь в программный код главной формы (форма для авторизации). После конструктора формы объявите строковую переменную txtcon и присвойте ей значение пустого текста.

```
public Form1()
{
    InitializeComponent();
}

// строка подключения к базе данных
public static string txtcon = @"";
```

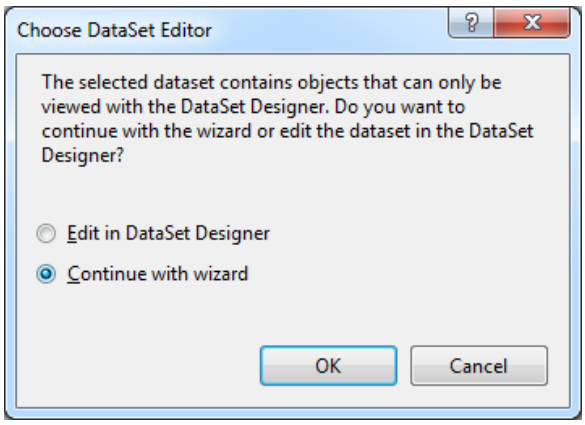
public означает, что к этой переменной у нас есть глобальный доступ

static означает, что это статическая переменная, и к ней можно обращаться прямо из класса формы (Form1.txtcon).

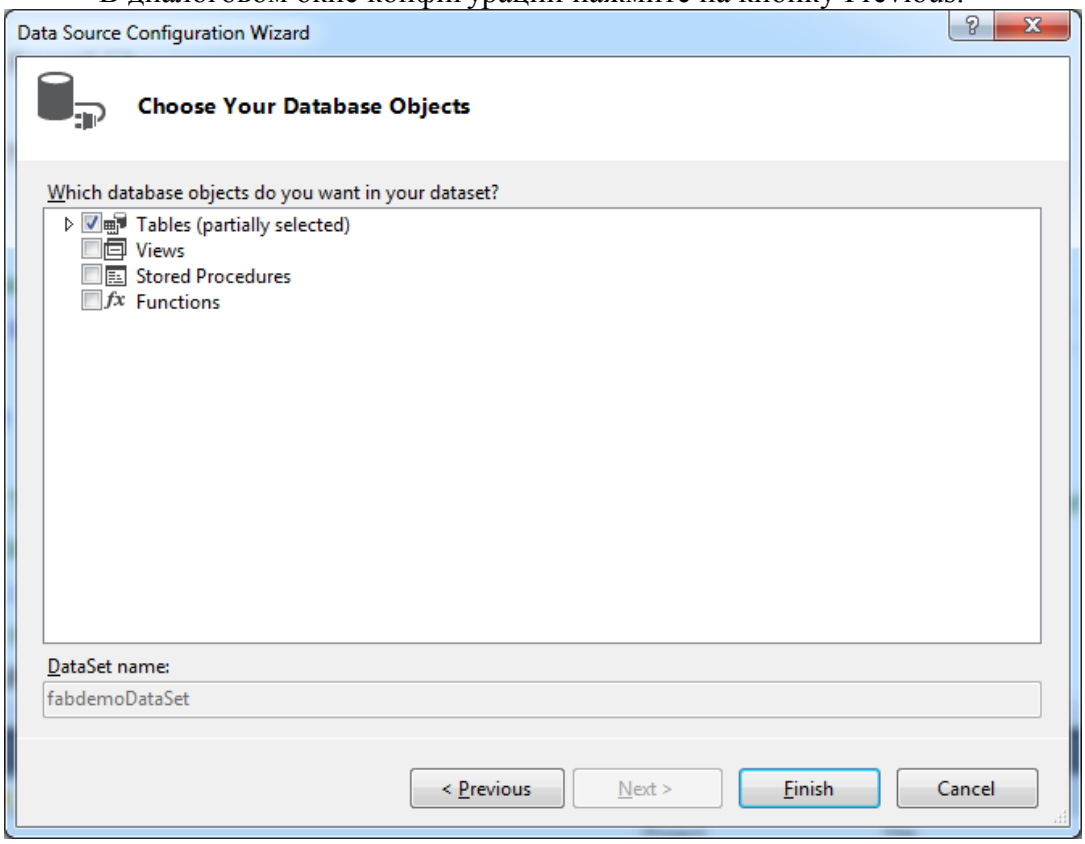
В панели Data Sources щелкните правой кнопкой на источнике данных для нашего приложения и выберите команду «Configure Data Source with Wizard...».



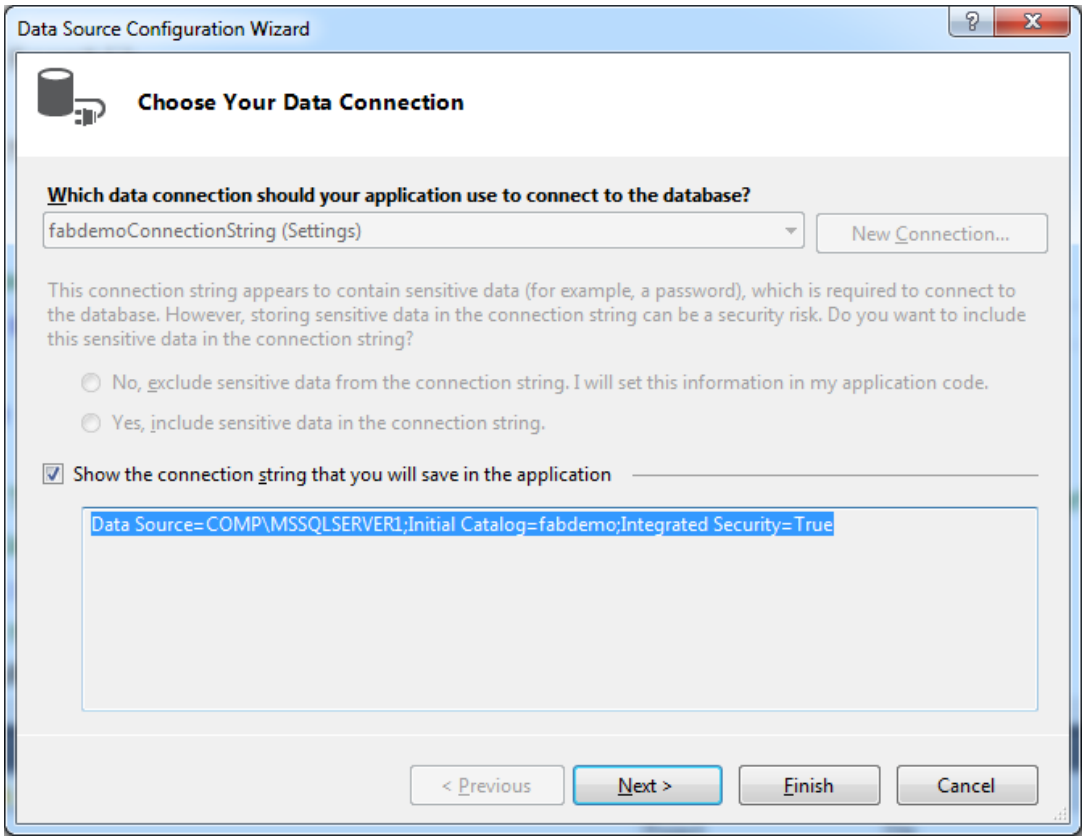
Выберите пункт «Continue with Wizard».



В диалоговом окне конфигурации нажмите на кнопку Previous.



Установите флажок «Show the connection string» и скопируйте в буфер строку подключения к базе данных.



Затем вставьте текст строки подключения в переменную txtcon внутри кавычек.

```
// строка подключения к базе данных
public static string txtcon = @"Data Source=COMP\MSSQLSERVER1;Initial
  Catalog=fabdemo;Integrated Security=True";
```

Внимание! Сначала в программном коде нужно поставить пустые кавычки, и только потом вставлять в них текст строки подключения. Если сделать наоборот, то редактор Visual Studio добавит в строку подключения лишние пробелы и она будет уже испорчена.

Шаг 5. Переключитесь в программный код. Нам нужно заполнять таблицу с заказами для текущего заказчика.

После конструктора формы опишите программный код процедуры для заполнения таблицы с данными о текущих заказах для авторизовавшегося заказчика.

```
        void FillListZakaz()
        {
            SqlConnection con = new SqlConnection(Form1.txtcon);
            con.Open();
            string txtquery =
@"select Izdeliya.nameizd as izd, Zakaz.countizd as countizd,
Tkani.nametkan as tkan, Furnitura.namefur as fur, Zakaz.countfur as countfur,
Users.fam as manager
from Zakaz, Users, Tkani, Furnitura, Izdeliya
where Users.iduser = Zakaz.idmanager and Tkani.idtkan = Zakaz.idtkan
and Furnitura.idfur = Zakaz.idfur and Izdeliya.idizd = Zakaz.idizd
and Zakaz.idzakazchik = " + lblIdUser.Text;

            SqlCommand query1 = new SqlCommand(txtquery, con);
            // выполнить запрос
            SqlDataReader sqlrez = query1.ExecuteReader();
```

Вначале создается и открывается подключение к базе данных. Обратите внимание, что строка подключения txtcon берется с главной формы Form1. Затем в переменную txtquery записывается SQL-запрос для выборки из базы данных. Рассмотрим его подробнее.

Для dgvZakaz нам нужны следующие сведения: наименование изделия, количество изделий, наименование ткани, наименование фурнитуры, количество фурнитуры, менеджер, обслуживающий заказ.

Информация о заказах пользователей хранится в таблице Zakaz. Но в этой таблице вместо конкретных наименований и фамилий хранятся только коды. Например:

COMP\MSSQLSERVE...demo - dbo.Zakaz | COMP\MSSQLSERVE...demo - dbo.Users

idzakaz	idizd	idzakazchik	idmanager	idtkan	idfur	countfur	countizd
1	1	16	17	10	12	2	1
2	2	16	17	11	11	3	2
3	5	16	20	9	10	1	1
4	2	16	17	4	10	1	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Нам нужно, используя коды, получить конкретные данных из других таблиц базы данных. С помощью

```
select  Izdeliya.nameizd as izd,
        Zakaz.countizd as countizd,
        Tkani.nametkan as tkan,
        Furnitura.namefur as fur,
        Zakaz.countfur as countfur,
        Users.fam as manager
from Zakaz, Users, Tkani, Furnitura, Izdeliya»
```

мы выбираем названия изделий из таблицы Izdeliya (даем этому полю имя izd), количество заказанных изделий из таблицы Zakaz (даем этому полю имя countizd), названия тканей из таблицы Tkani (даем этому полю имя tkan), название фурнитуры из таблицы Furnitura (даем этому полю имя fur), количество фурнитуры из таблицы Zakaz (даем этому полю имя countfur), фамилию менеджера из таблицы Users (даем этому полю имя fam).

С помощью ряда условий мы связываем эти таблицы между собой:

```
where Users.iduser = Zakaz.idmanager
  and Tkani.idtkan = Zakaz.idtkan
   and Furnitura.idfur = Zakaz.idfur
   and Izdeliya.idizd = Zakaz.idizd
```

Таблицы Users и Zakaz связываются по полям iduser и idmanager, таблицы Tkani и Zakaz по полю idtkan, таблицы Furnitura и Zakaz по полю idfur, таблицы Izdeliya и Zakaz по полю idizd.

Последнее условие "Zakaz.idzakazchik = " + lblIdUser.Text означает, что нужно выбрать из базы данных не все заказы, а только те, которые относятся к авторизовавшемуся заказчику. Код авторизовавшегося заказчика находится в метке lblIdUser. Эта метка у нас связана с полем iduser таблицы Users.

```
            dgvZakaz.Rows.Clear(); // очистить от старых строк
            // перебирать результаты запроса
            while (sqlrez.Read())
            {
                dgvZakaz.Rows.Add(sqlrez["izd"], sqlrez["countizd"],
                    sqlrez["tkan"], sqlrez["fur"], sqlrez["countfur"],
                    sqlrez["manager"]);
            }
            con.Close();
        }
```

Далее dgvZakaz очишается от старых записей, в цикле while выполняется перебор строк в результате запроса. В dgvZakaz выводятся поля из результата запроса.

Шаг 6. Поставьте вызов этой процедуры в обработчик события Load, чтобы таблица с заказами заполнялась автоматически при открытии формы с рабочим местом заказчика.

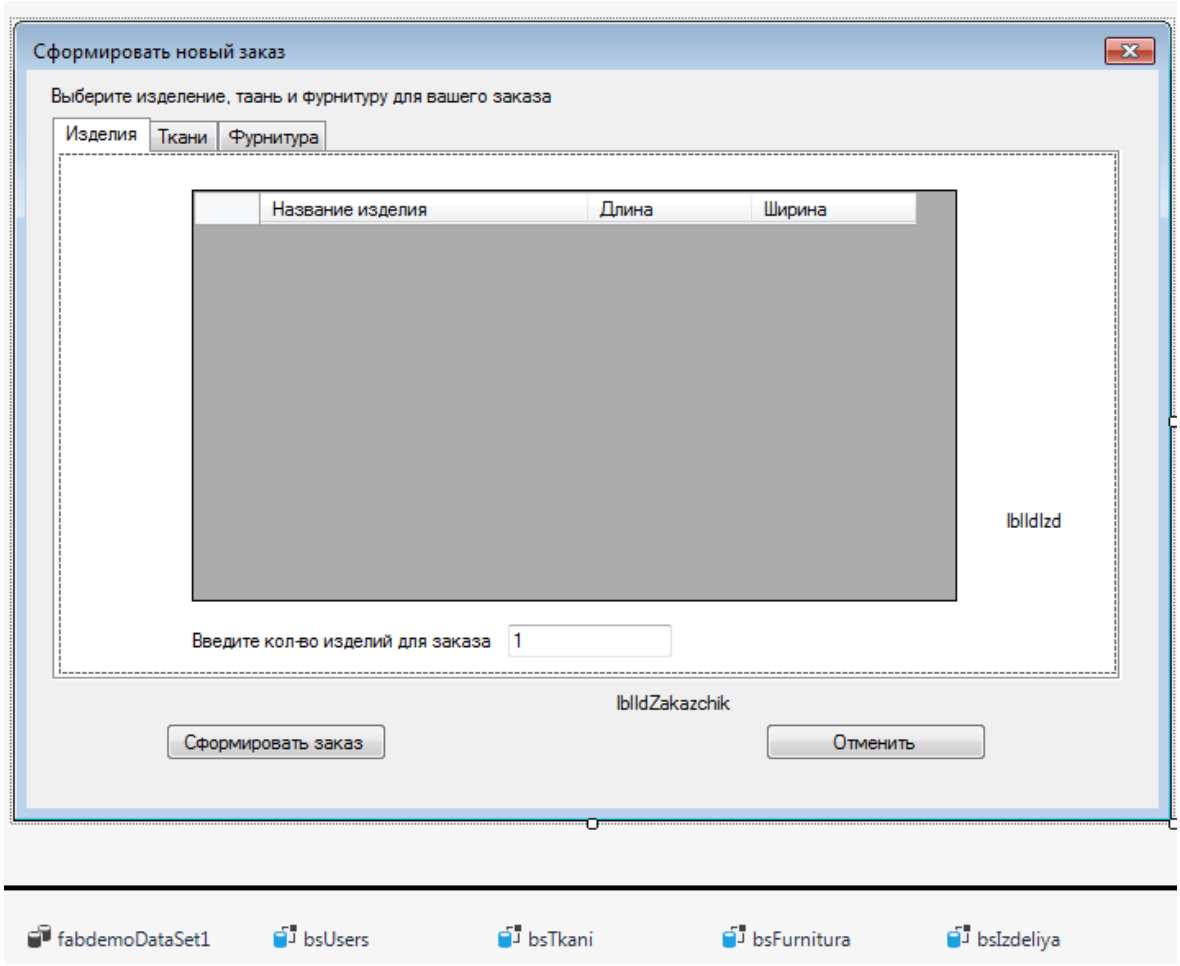
```
FillListZakaz();  // сформировать список с заказами
```

16.3. Формирование интерфейса и описание программного кода для добавления нового заказа

Шаг 1. Добавьте к проекту новую форму. Назовите ее FormAddZakaz. Задайте для этой формы следующие свойства:

Text - Сформировать новый заказ (заголовок формы)
StartPosition - CenterParent (отображать форму по центру родительской формы)
MaximizeBox - False (заблокировать кнопку разворачивания формы до максимума)
MinimizeBox - False (заблокировать кнопку сворачивания формы до минимума)
FormBorderStyle - FixedDialog (диалоговое окно без кнопки системного меню, не разрешать изменять размеры формы)

Шаг 2. Сформируйте следующий интерфейс формы:



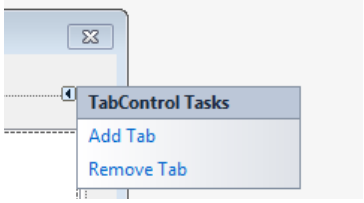
Добавьте на форму DataSet и четыре BindingSource. Дайте для BindingSource имена: bsUsers, bsTkani, bsFurnitura и bsIzdeliya. Свяжите их с соответствующими таблицами базы данных. Для этого используйте свойства DataSource и DataMember.

Для bsTkani в свойстве Filter установите length > 0, для bsFurnitura в свойстве Filter установите countfur > 0. Если длина рулона ткани равна нулю, или количество фурнитур равно нулю, значит их нет не складе и их нельзя использовать для добавления заказа.

Для кнопок «Сформировать заказ» и «Отменить» используйте объекты Button. Дайте им имена btnOK и btnCancel. Для кнопки btnCancel в свойстве DialogResult установите значение Cancel, чтобы по щелчку на кнопку форма закрывалась.

В метке lblIdZakazchik у нас будет храниться id заказчика, для которого нужно формировать новый заказ. Свяжите эту метку с полем iduser таблицы Users. В свойстве (DataBindings)►Text установите значение bsUsers - iduser. Однако, сама метка должна быть невидимой. Ее содержимое будет использоваться только в программном коде. Для этой метки сделайте цвет текста и цвет фона (свойства ForeColor и BackColor) одинаковыми. Если вы попытаетесь в свойстве Visible для нее установите значение False, то такой прием будет работать не всегда. Во многих случаях метки, имеющие в свойстве Visible значение False, не могут хранить данные из таблиц базы данных.

Для формирования заказа нам нужно выбирать изделие, ткань и фурнитуру. Поставьте на форму компонент TabControl, который позволяет разместить визуальные объекты на отдельных вкладках. Щелкните мышью на кнопку со стрелкой в правом верхнем углу TabControl и выберите пункт Add Tab для добавления дополнительных вкладок.



В свойстве Text каждой вкладки измените заголовок на Изделия, Ткани и Фурнитура.

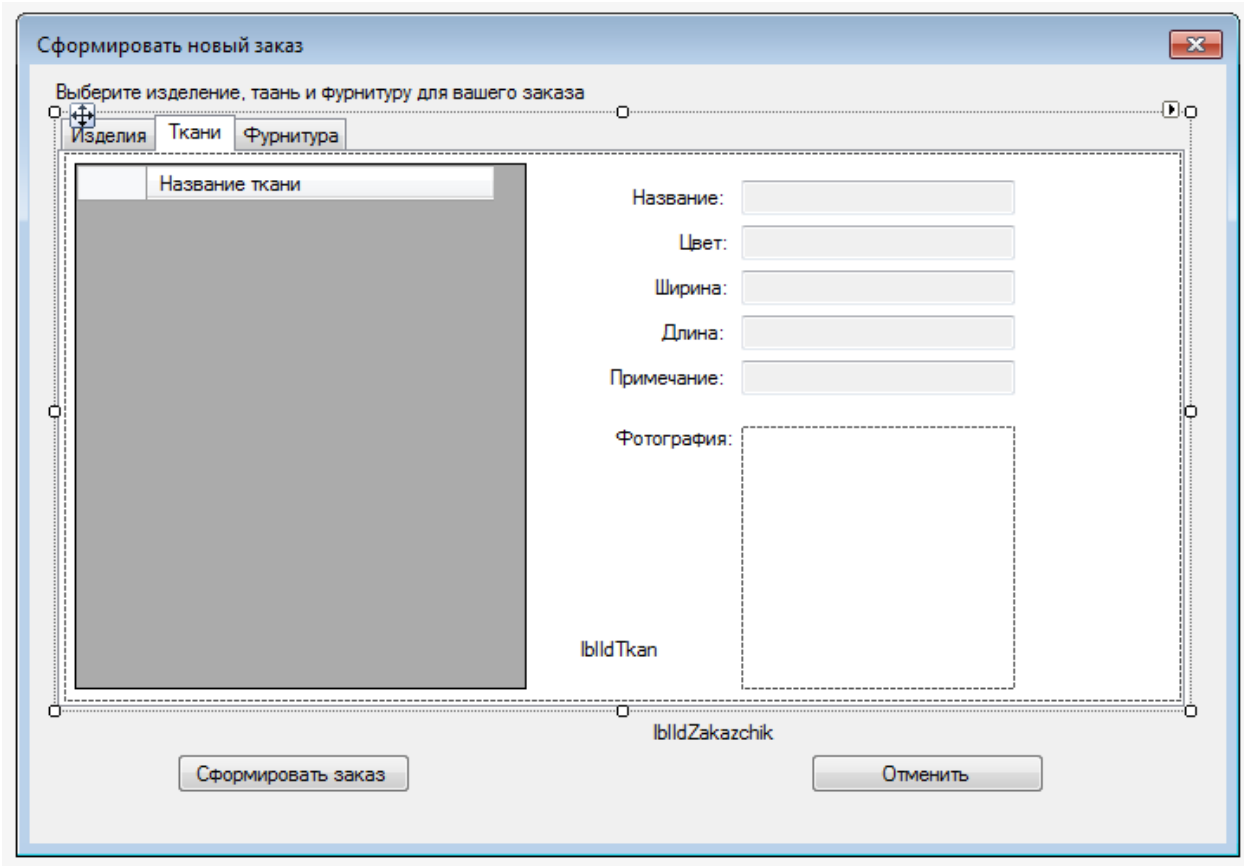
Шаг 3. На вкладку Изделия добавьте DataGridView. Свяжите его с таблицей Izdeliya (свойство DataSource для DataGridView). Выполните настройку столбцов для таблицы. Щелкните правой кнопкой на DataGridView и выберите пункт Edit Columns... Удалите столбец idizd, а на остальных столбцах измените заголовки (свойство HeaderText). Для названия изделия увеличьте ширину до 200 пикселей.

После того, как вы выполните настройку столбцов, для DataGridView измените свойства:
AllowUserToAddRows - False (чтобы пользователь не мог добавлять строки в таблицу)
AllowUserToDeleteRows - False (чтобы пользователь не мог удалять строки из таблицы)
ReadOnly - True (чтобы пользователь не мог редактировать ячейки таблицы)
SelectionMode - FullRowSelect (чтобы при выделении в таблице выделялась сразу вся строка)

На вкладку Изделия добавьте метку Label и переименуйте ее в lblIdIzd. В метке lblIdIzd у нас будет храниться id изделия, которое используется для формирования нового заказа. Свяжите эту метку с полем idizd таблицы Izdeliya. В свойстве (DataBindings)►Text установите значение bsIzdeliya - idizd. Однако, сама метка должна быть невидимой. Ее содержимое будет использоваться только в программном коде. Для этой метки сделайте цвет текста и цвет фона (свойства ForeColor и BackColor) одинаковыми. Если вы попытаетесь в свойстве Visible для нее установите значение False, то такой прием будет работать не всегда. Во многих случаях метки, имеющие в свойстве Visible значение False, не могут хранить данные из таблиц базы данных.

На вкладку Изделия добавьте также TextBox для ввода количества изделий. Дайте ему имя tbxCountIzd.

Шаг 4. Переключитесь на вкладку Ткани. Сформируем для нее следующий интерфейс.

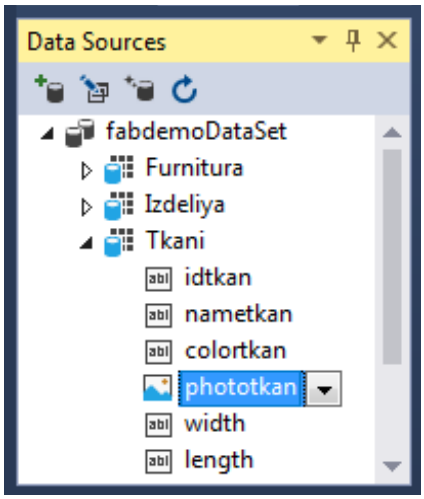


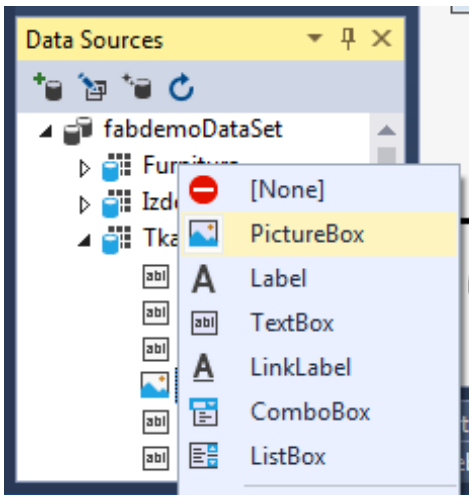
На вкладку Ткани добавьте DataGridView. Свяжите его с таблицей Tkani (свойство DataSource для DataGridView). Выполните настройку столбцов для таблицы. Щелкните правой кнопкой на DataGridView и выберите пункт Edit Columns... Удалите все столбцы кроме nametkan. Для столбца «Название ткани» измените заголовок (свойство HeaderText). Увеличьте также ширину столбца до 200 пикселей.

После того, как вы выполните настройку столбцов, для DataGridView измените свойства:
AllowUserToAddRows - False (чтобы пользователь не мог добавлять строки в таблицу)
AllowUserToDeleteRows - False (чтобы пользователь не мог удалять строки из таблицы)
ReadOnly - True (чтобы пользователь не мог редактировать ячейки таблицы)
SelectionMode - FullRowSelect (чтобы при выделении в таблице выделялась сразу вся строка)

На вкладку Ткани добавьте метку Label и переименуйте ее в lblIdTkan. В метке lblIdTkan у нас будет храниться id ткани, которая используется для формирования нового заказа. Свяжите эту метку с полем idtkan таблицы Tkani. В свойстве (DataBindings)►Text установите значение bsTkani - idtkan. Однако, сама метка должна быть невидимой. Ее содержимое будет использоваться только в программном коде. Для этой метки сделайте цвет текста и цвет фона (свойства ForeColor и BackColor) одинаковыми. Если вы попытаетесь в свойстве Visible для нее установите значение False, то такой прием будет работать не всегда. Во многих случаях метки, имеющие в свойстве Visible значение False, не могут хранить данные из таблиц базы данных.

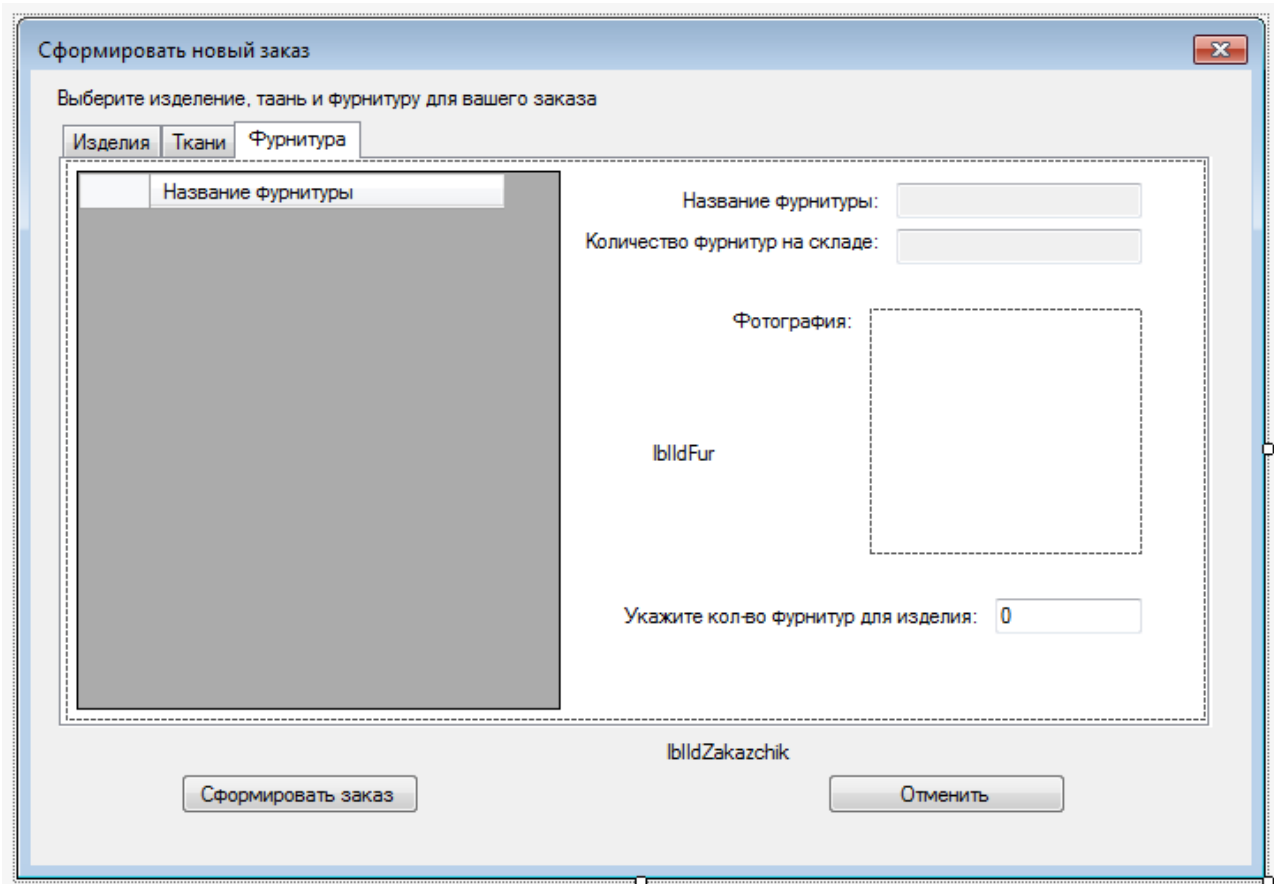
В панели Data Sources разверните таблицу Tkani, выделите поле phototkan и выберите для него тип визуального объекта PictureBox.





Затем отбуксируйте на вкладку поля: nametkan, colortkan, width, length, description, phototkan. В метках Label напишите названия полей по-русски. Для текстовых полей с названием, цветом, шириной, длиной и примечанием в свойстве ReadOnly установите значение True, чтобы пользователь не имел возможности редактировать содержимое этих полей. Для фотографии ткани в свойстве SizeMode установите значение Zoom, чтобы фотография ткани была вписана в PictureBox с соблюдением пропорций.

Шаг 5. Переключитесь на вкладку Фурнитура. Сформируем для нее следующий интерфейс.

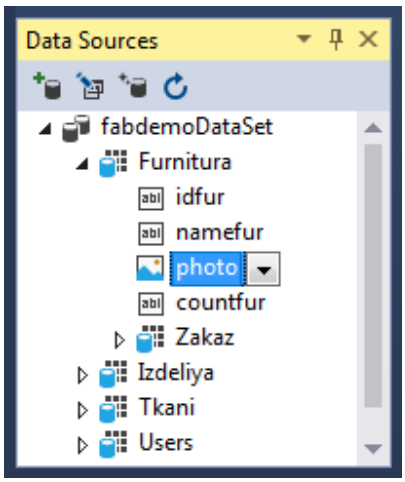


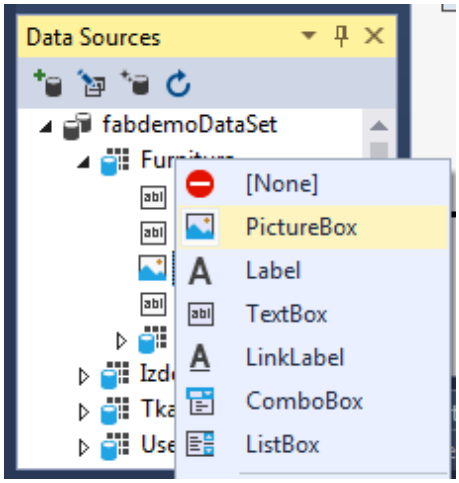
На вкладку Фурнитура добавьте DataGridView. Свяжите его с таблицей Furnitura (свойство DataSource для DataGridView). Выполните настройку столбцов для таблицы. Щелкните правой кнопкой на DataGridView и выберите пункт Edit Columns... Удалите все столбцы кроме namefur. Для столбца «Название фурнитуры» измените заголовок (свойство HeaderText). Увеличьте также ширину столбца до 200 пикселей.

После того, как вы выполните настройку столбцов, для DataGridView измените свойства:
AllowUserToAddRows - False (чтобы пользователь не мог добавлять строки в таблицу)
AllowUserToDeleteRows - False (чтобы пользователь не мог удалять строки из таблицы)
ReadOnly - True (чтобы пользователь не мог редактировать ячейки таблицы)
SelectionMode - FullRowSelect (чтобы при выделении в таблице выделялась сразу вся строка)

На вкладку Фурнитура добавьте метку Label и переименуйте ее в lblIdFur. В метке lblIdFur у нас будет храниться id фурнитуры, которая используется для формирования нового заказа. Свяжите эту метку с полем idfur таблицы Furnitura. В свойстве (DataBindings)►Text установите значение bsFurnitura - idfur. Однако, сама метка должна быть невидимой. Ее содержимое будет использоваться только в программном коде. Для этой метки сделайте цвет текста и цвет фона (свойства ForeColor и BackColor) одинаковыми. Если вы попытаетесь в свойстве Visible для нее установите значение False, то такой прием будет работать не всегда. Во многих случаях метки, имеющие в свойстве Visible значение False, не могут хранить данные из таблиц базы данных.

В панели Data Sources разверните таблицу Furnitura, выделите поле photofur и выберите для него тип визуального объекта PictureBox.





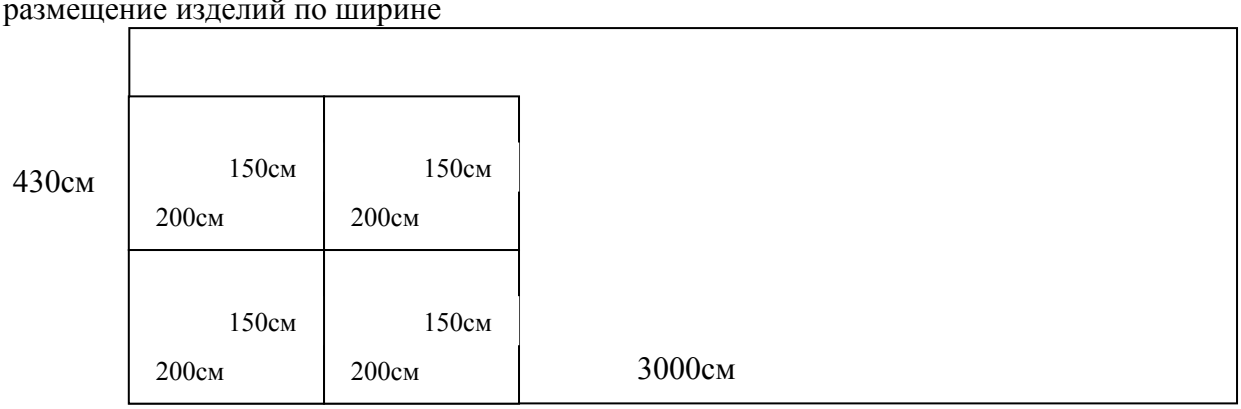
Затем отбуксируйте на вкладку поля: namefur, countfur, photo. В метках Label напишите названия полей по-русски. Для текстовых полей с названием и цветом в свойстве ReadOnly установите значение True, чтобы пользователь не имел возможности редактировать содержимое этих полей. Для фотографии ткани в свойстве SizeMode установите значение Zoom, чтобы фотография ткани была вписана в PictureBox с соблюдением пропорций.

На вкладку Фурнитура поставьте текстовое поле для ввода количества фурнитур, необходимых для изделия. Дайте этому полю имя tbxCountFur.

Шаг 6. Создадим функцию CutTkan(), в которую передаются следующие параметры: ширина рулона ткани, ширина и длина ткани для изделия, количество изделий. Функция должна возвратить результат - минимальную длину ткани, которую нужно отрезать от рулона, чтобы можно было изготовить указанное количество изделий.

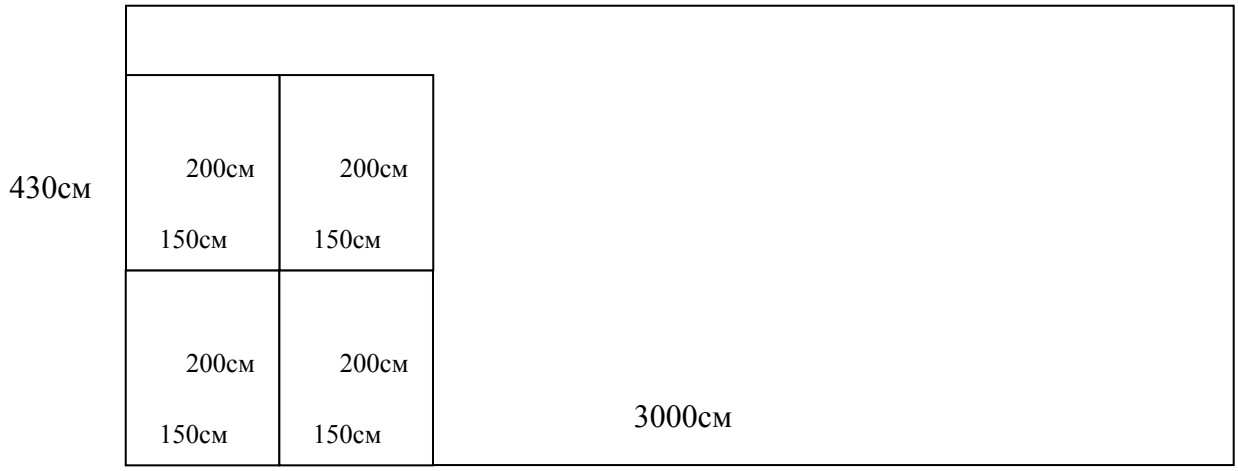
Рассмотрим, как определяется необходимая длина ткани на конкретном примере. Пусть у нас имеется рулон ткани шириной 430 и длиной 3000 см. Для изготовления одного изделия нам нужен кусок ткани 150 на 200см. У нас есть два варианта раскроя тканей. В одном случае мы можем вырезать куски ткани из рулона, размещая изделия на ткани по ширине. В другом случае мы можем вырезать куски ткани из рулона, размещая изделия на ткани по длине.

1) размещение изделий по ширине



В этом случае у нас будет два ряда по два изделия в каждом ряду. А от рулона ткани нужно будет отрезать кусок длиной 400см.

2) размещение изделий по длине



В этом случае у нас будет два ряда по два изделия в каждом ряду. А от рулона ткани нужно будет отрезать кусок длиной 300см.

Во втором случае от рулона нужно отрезать меньшее количество ткани.

Нам нужно описать такой программный код функции CutTkan(), в котором рассматривались бы оба варианта раскроя ткани, и выбирался бы тот вариант, в котором длина куска ткани, который нужно отрезать от рулона, была бы минимальной.

Программный код функции поместите после конструктора формы FormAddZakaz.

```
int CutTkan(int widthtkan, int widthizd, int lengthizd, int countizd)
{
    // ЕСЛИ УКЛАДЫВАТЬ ИЗДЕЛИЯ НА ТКАНЬ В РУЛОНЕ ПО ШИРИНЕ ИЗДЕЛИЯ
    // кол-во изделий, умещающихся в 1 ряд на рулоне ткани
    int izdinrow1 = widthtkan / widthizd;
    // кол-во рядов, в каждом из которых izdinrow изделий
    int row1 = countizd / izdinrow1;
    // если перемножить кол-во изделий в 1 ряду на кол-во рядов и полученное
    // кол-во будет меньше кол-ва необходимых изделий, это значит, что нужно
    // использовать еще один ряд на рулоне ткани (неполный ряд)
    if (izdinrow1 * row1 < countizd) row1++;
    // длина ткани, которую нужно отрезать от рулона, если укладывать
    // изделия по ширине
    int cuttkan1 = row1 * lengthizd;
```

В функцию передается ширина рулона ткани, ширина и длина ткани, необходимой для изготовления одного изделия, количество изделий.

Вначале рассмотрим вариант, когда изделия располагаются на ткани по ширине.

Разделив ширину рулона ткани на ширину одного изделия нацело, мы узнаем, сколько изделий умещается в один ряд. Разделив количество изделий, которые нужно изготовить, на количество изделий в одном ряду, мы узнаем, сколько полных рядов у нас должно быть. Может так получиться, что последний ряд будет неполным. В таком случае к количеству рядов нужно добавить еще один.

Умножив количество рядов на длину изделия, мы узнаем, сколько от рулона ткани нужно отрезать материала.

Далее рассматривается случай, когда изделия располагаются на ткани по длине.

```
    // ЕСЛИ УКЛАДЫВАТЬ ИЗДЕЛИЯ НА ТКАНЬ В РУЛОНЕ ПО ДЛИНЕ ИЗДЕЛИЯ
    // кол-во изделий, умещающихся в 1 ряд на рулоне ткани
    int izdinrow2 = widthtkan / lengthizd;
    // кол-во рядов, в каждом из которых izdinrow изделий
    int row2 = countizd / izdinrow2;
    // если перемножить кол-во изделий в 1 ряду на кол-во рядов и полученное
    // кол-во будет меньше кол-ва необходимых изделий, это значит, что нужно
    // использовать еще один ряд на рулоне ткани (неполный ряд)
    if (izdinrow2 * row2 < countizd) row2++;
    // длина ткани, которую нужно отрезать от рулона, если укладывать
    // изделия по длине
    int cuttkan2 = row2 * widthizd;

    // возвратить наименьшее кол-во ткани, которую нужно использовать для
    // изготовления countizd изделий
    if (cuttkan1 < cuttkan2) return cuttkan1;
    else return cuttkan2;
}
```

Разделив ширину рулона ткани на длину одного изделия нацело, мы узнаем, сколько изделий умещается в один ряд. Разделив количество изделий, которые нужно изготовить, на количество изделий в одном ряду, мы узнаем, сколько полных рядов у нас должно быть. Может так получиться, что последний ряд будет неполным. В таком случае к количеству рядов нужно добавить еще один.

Умножив количество рядов на ширину изделия, мы узнаем, сколько от рулона ткани нужно отрезать материала.

В переменных cuttkan1 и cuttkan2 у нас находятся два числа - длины куска ткани, которую нужно отрезать от рулона. В качестве результата мы будем возвращать минимальное значение.

Шаг 7. Для кнопки «Сформировать заказ» опишите следующий программный код:

```
private void btnOK_Click(object sender, EventArgs e)
{
    // сейчас все новые заказы обслуживает менеджер с кодом 17.
    // подумайте, какой программный код сюда поставить, чтобы в переменную
    // idmanager записывался код менеджера, у которого меньше всего заказов,
    // может быть, воспользоваться каким-либо SQL-запросом?
    int idmanager = 17;
```

Здесь в переменную idmanager записывается код менеджера, равный 17. Т.е. все заказы будет обслуживать менеджер с этим кодом. Посмотрите в свою базу данных в таблицу Users. Если у вас нет менеджера с кодом 17, то запишите в переменную idmanager код любого другого существующего в базе данных менеджера. Иначе при работе приложения будет возникать ошибка.

Затем подумайте, какой программный код поставить в это место, чтобы в переменную idmanager записывался код менеджера, у которого меньше всего заказов. Например, имеется таблица Zakaz со следующими записями:

COMP\MSSQLSERVE...demo - dbo.Zakaz

idzakaz	idizd	idzakazchik	idmanager	idtkan	idfur	countfur	countizd
1	1	16	17	10	12	2	1
2	2	16	17	11	11	3	2
3	5	16	20	9	10	1	1
4	2	16	17	4	10	1	1
5	3	16	17	4	7	2	1
6	3	16	17	16	1	0	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

По этим записям видно, что менеджер с кодом 17 обслуживает 5 заказов, а менеджер с кодом 20 имеет 1 заказ. Но в таблице Users у нас есть еще менеджер, который не обслуживает ни одного заказа:

COMP\MSSQLSERVE...demo - dbo.Users | COMP\MSSQLSERVE...demo - dbo.Zakaz

iduser	loqin	password	role	fam	name	otch	phone	photo
16	z	z	заказчик	Иванов	Иван	Ильич	+7(915) 158-98-...	NULL
17	m	m	менеджер	Петров	Андрей	Егорович1	+7 (905) 543-45...	<Binary data>
18	k	k	кладовщик	Андреев	Семен	Степанович	+7(910) 324-65-...	NULL
19	stepanova	sgf432	заказчик	Степанова	Ирина	Петровна	+7(952) 543-67-...	NULL
20	ilyina	3245bghfx	менеджер	Ильина	Анна	Петровна	+7(905) 678-65-...	NULL
21	egorov	sdgf435	кладовщик	Егоров	Петр	Андреевич	+7(910) 122-54-...	NULL
22	vasilyeva	gfbet345	заказчик	Васильева	Елена	Алексеевна	+7(915) 677-99-...	NULL
24	z1	1A!zzz	менеджер	Яковлев	Егор	Петрович	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Это менеджер Яковлев с кодом 24. Значит, новый заказ надо поручить обслуживать менеджеру с кодом 24.

Вам предлагается разработать SQL-запрос, результатом которого будет код менеджера у которого меньше всего записей в таблице Zakaz, либо количество заказов вообще равно нулю. Этот SQL-запрос нужно будет выполнить из программного кода приложения и записать полученный результат в переменную idmanager.

```
// количество фурнитур на складе
int countFurSklad = Convert.ToInt32(countfurTextBox.Text);
// уменьшить количество фурнитур на складе на то количество,
// которое указал пользователь
countFurSklad -= Convert.ToInt32(tbxCountFur.Text)
    * Convert.ToInt32(tbxCountIzd.Text);
// вывести новое количество фурнитур в текстовое поле, связанное
// с базой данных
countfurTextBox.Text = countFurSklad.ToString();
// сохранить изменения в таблице Furnitura
bsFurnitura.EndEdit();
furnituraTableAdapter.Update(fabdemoDataSet1.Furnitura);
```

Дальше в нашем программном коде в переменную countFurSklad считывается из текстового поля на форме количество фурнитур на складе. Это количество уменьшается на количество, необходимое для изготовления изделий (количество фурнитур на одно изделие умножается на количество изделий). Измененное количество фурнитур выводится в текстовое поле, связанное с базой данных, после чего сохраняются изменения в таблице Furnitura.

В обработчик события Click для кнопки «Сформировать заказ» дальнейший программный код будет следующим:

```
// записать в переменные ширину и длину рулона ткани,
// ширину и длину ткани для изделия, количество изделий
int widthtkan = Convert.ToInt32(widthTextBox.Text);
int lengthtkan = Convert.ToInt32(lengthTextBox.Text);
int widthizd =
    Convert.ToInt32(izdeliyaDataGridView.CurrentRow.Cells[2].Value);
int lengthizd =
    Convert.ToInt32(izdeliyaDataGridView.CurrentRow.Cells[1].Value);
int countizd = Convert.ToInt32(tbxCountIzd.Text);
```

В переменные widthtkan и lengthtkan записываются ширина и длина рулона ткани на складе. Эти данные берутся из текстовых полей на вкладке Ткани.

В переменные widthizd и lengthizd записываются ширина и длина ткани, необходимые для изготовления выбранного изделия. Эти данные берутся из DataGridView на вкладке Изделия. Свойство CurrentRow обозначает текущую выделенную строку в DataGridView, Cells[2] - столбец с шириной, Cells[1] - столбец с длиной изделия.

Количество изделий считывается из текстового поля tbxCountIzd на вкладке Изделия.

```
// отрезать от рулона ткани кусок такой длины, которая нужна для изготовления
// countizd изделий
lengthtkan = lengthtkan - CutTkan(widthtkan, widthizd, lengthizd, countizd);
// записать в поле с длиной рулона ткани новое значение
lengthTextBox.Text = lengthtkan.ToString();
// сохранить изменения таблицы Tkani в базе данных
bsTkani.EndEdit();
tkaniTableAdapter.Update(fabdemoDataSet1.Tkani);
```

Затем вызывается функция CutTkan(). Она возвращает длину куска ткани, который нужно отрезать от рулона ткани. Длина рулона ткани уменьшается на значение функции CutTkan(), новое значение длины рулона записывается в текстовое поле, связанное с базой данных. Затем выполняется сохранение в базе данных изменений в таблице Tkani.

Затем в программном коде нужно выполнить добавление новой записи в таблицу Zakaz:

```
SqlConnection con = new SqlConnection(Form1.txtcon);
try
{
    con.Open();
    // SQL-запрос для добавления нового заказа
    string t = String.Format("insert into Zakaz (idizd, idzakazchik,
      idmanager, idtkan, idfur, countfur, countizd) " +
        "values ({0}, {1}, {2}, {3}, {4}, {5}, {6})",
        lblIdIzd.Text, lblIdZakazchik.Text, idmanager, lblIdTkan.Text,
         lblIdFur.Text, tbxCountFur.Text, tbxCountIzd.Text);
    SqlCommand query1 = new SqlCommand(t, con);
    // выполнить запрос
    query1.ExecuteNonQuery();
    con.Close();
    MessageBox.Show("Заказ сформирован.", "Внимание!",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
    this.Close();  // закрыть форму
}
catch
{
    MessageBox.Show("Ошибка добавления нового заказа.", "Внимание!",
        MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
}
```

В таблицу Zakaz добавляются поля: idizd, idzakazchik, idmanager, idtkan, idfur, countfur, countizd. Рассмотрим подробнее, что записывается в каждое из этих полей:

Имя поля	Что в него записывается
idizd	lblIdIzd.Text - код изделия. Он находится на невидимой метке Label на вкладке Изделия. Этот код берется из таблицы Izdeliya.
idzakazchik	lblIdZakazchik.Text - код заказчика. Он находится на невидимой метке Label

	lblIdZakazchik. Этот код берется из таблицы Users.
idmanager	Код менеджера, который хранится в переменной idmanager.
idtkan	lblIdTkan.Text - код выбранной ткани. Он находится на невидимой метке Label на вкладке Ткани. Этот код берется из таблицы Tkani.
idfur	lblIdFur.Text - код выбранной фурнитуры. Он находится на невидимой метке Label на вкладке Фурнитуры. Этот код берется из таблицы Furnitura.
countfur	tbxCountFur.Text - количество фурнитур, необходимых для изготовления изделия. Это количество берется из текстового поля на вкладке Фурнитура.
countizd	tbxCountIzd.Text - количество изделий в заказе. Это количество берется из текстового поля на вкладке Изделия.

На тот случай, если в ходе выполнения SQL-запроса возникнет ошибка, в программном коде предусмотрена конструкция try...catch.

Шаг 8. Откройте форму FormZakazchik. Для кнопки «Сформировать новый заказ» опишите следующий код:

```
private void btnAddZakaz_Click(object sender, EventArgs e)
{
    FormAddZakaz frm = new FormAddZakaz();
    // передать на форму для добавления заказа фильтр текущего пользователя
    frm.bsUsers.Filter = this.bsUsers.Filter;
    frm.ShowDialog();
    FillListZakaz();  // обновить список с заказами
}
```

Здесь вначале создается форма для добавления нового заказа. Затем на форму для добавления заказа передается фильтр авторизовавшегося заказчика. Далее форма для добавления заказа открывается. После того, как пользователь добавил новый заказ, нужно вызвать процедуру FillListZakaz(), которая обновит список с заказами для текущего пользователя.

16.4. Нечеткий поиск по названиям тканей с учетом расстояния Левенштейна

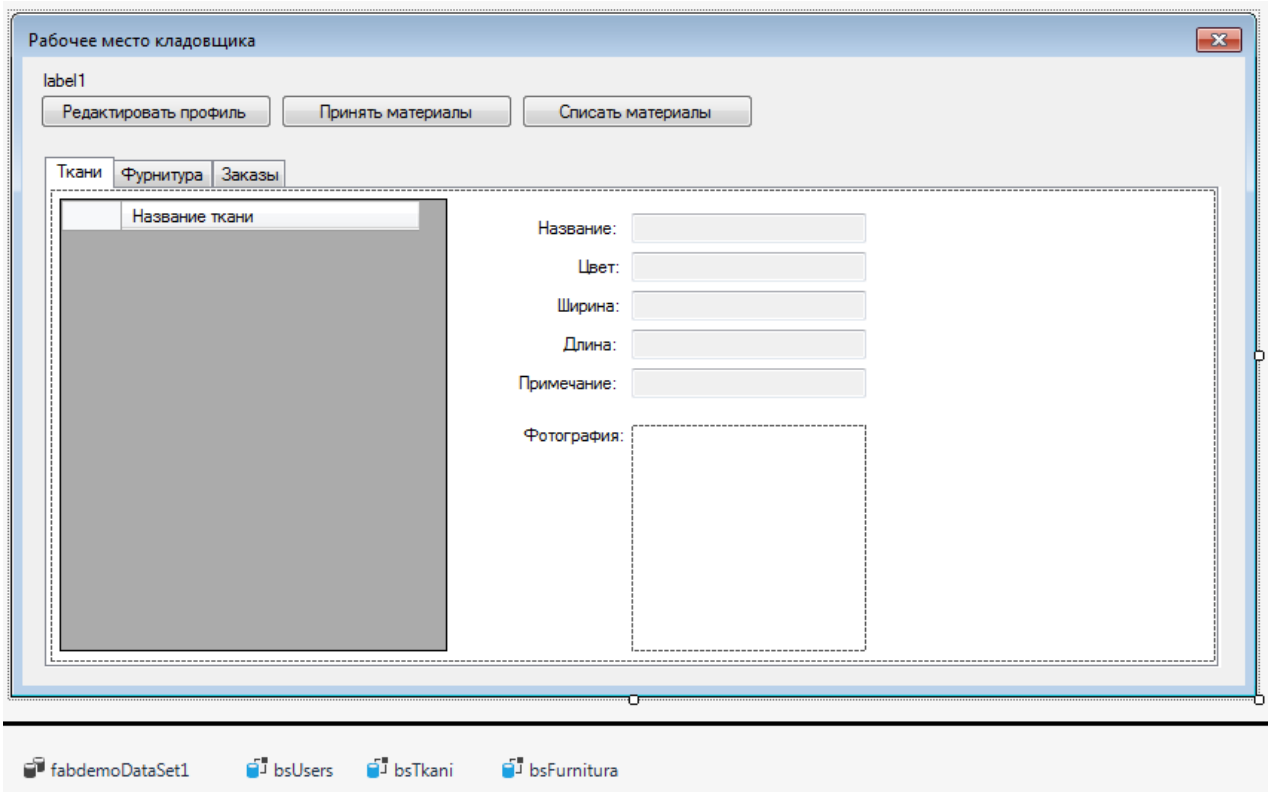
Как сформировать интерфейс и описать программный код для нечеткого поиска с учетом расстояния Левенштейна, рассмотрено в отдельной теме этого учебного пособия. Добавьте в СУБД «Швейная фабрика» нечеткий поиск по названиям тканей самостоятельно.

16.5. Формирование интерфейса и описание программного кода для рабочего места кладовщика

Шаг 1. Если в вашем проекте еще нет формы для рабочего места заказчика, то добавьте к проекту новую форму. Дайте ей имя FormKladovschik. Установите для формы следующие значения свойств:

Text - Рабочее место кладовщика (заголовок формы)
StartPosition - CenterParent (отображать форму по центру родительской формы)
MaximizeBox - False (заблокировать кнопку разворачивания формы до максимума)
MinimizeBox - False (заблокировать кнопку сворачивания формы до минимума)
FormBorderStyle - FixedDialog (диалоговое окно без кнопки системного меню, не разрешать изменять размеры формы)

Шаг 2. Сформируем следующий интерфейс формы:

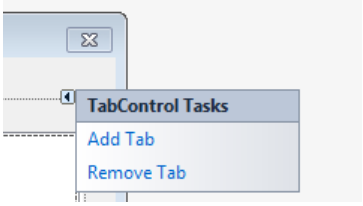


Для кнопок «Принять материалы» и «Списать материалы» используйте Button. Дайте им имена btnGetMaterial, btnSpisMaterial.

Добавьте на форму два BindingSource. Дайте им имена bsTkani и bsFurnitura. Свяжите эти объекты с таблицами Tkani и Furnitura. Для этого в свойстве DataSources выберите источник данных вашей базы данных, а в свойстве DataMember выберите соответствующую таблицу.

Для bsTkani в свойстве Filter установите length > 0, для bsFurnitura в свойстве Filter установите countfur > 0. Если длина рулона ткани равна нулю, или количество фурнитур равно нулю, значит их нет не складе и их нельзя использовать для добавления заказа.

Поставьте на форму компонент TabControl, который позволяет разместить визуальные объекты на отдельных вкладках. Щелкните мышью на кнопку со стрелкой в правом верхнем углу TabControl и выберите пункт Add Tab для добавления дополнительных вкладок.

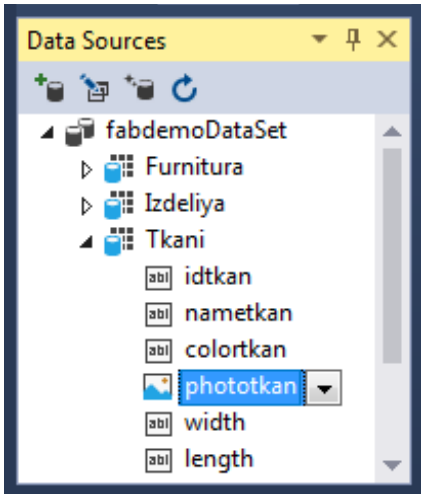


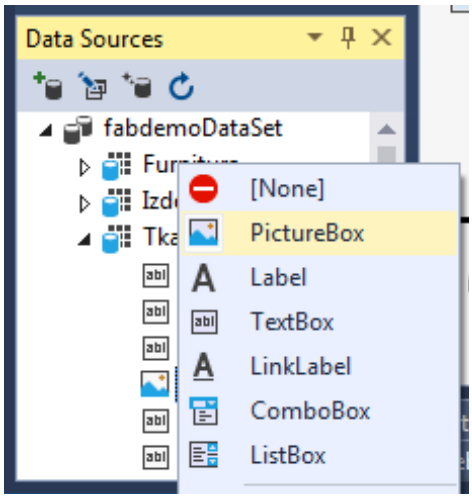
В свойстве Text каждой вкладки измените заголовок на Ткани, Фурнитура и Заказы.

Шаг 3. На вкладку Ткани добавьте DataGridView. Свяжите его с таблицей Tkani (свойство DataSource для DataGridView). Выполните настройку столбцов для таблицы. Щелкните правой кнопкой на DataGridView и выберите пункт Edit Columns... Удалите все столбцы, кроме nametkan. Для названия ткани измените заголовок (свойство HeaderText) и увеличьте ширину до 200 пикселей.

После того, как вы выполните настройку столбцов, для DataGridView измените свойства:
AllowUserToAddRows - False (чтобы пользователь не мог добавлять строки в таблицу)
AllowUserToDeleteRows - False (чтобы пользователь не мог удалять строки из таблицы)
ReadOnly - True (чтобы пользователь не мог редактировать ячейки таблицы)
SelectionMode - FullRowSelect (чтобы при выделении в таблице выделялась сразу вся строка)

В панели Data Sources разверните таблицу Tkani, выделите поле phototkan и выберите для него тип визуального объекта PictureBox.



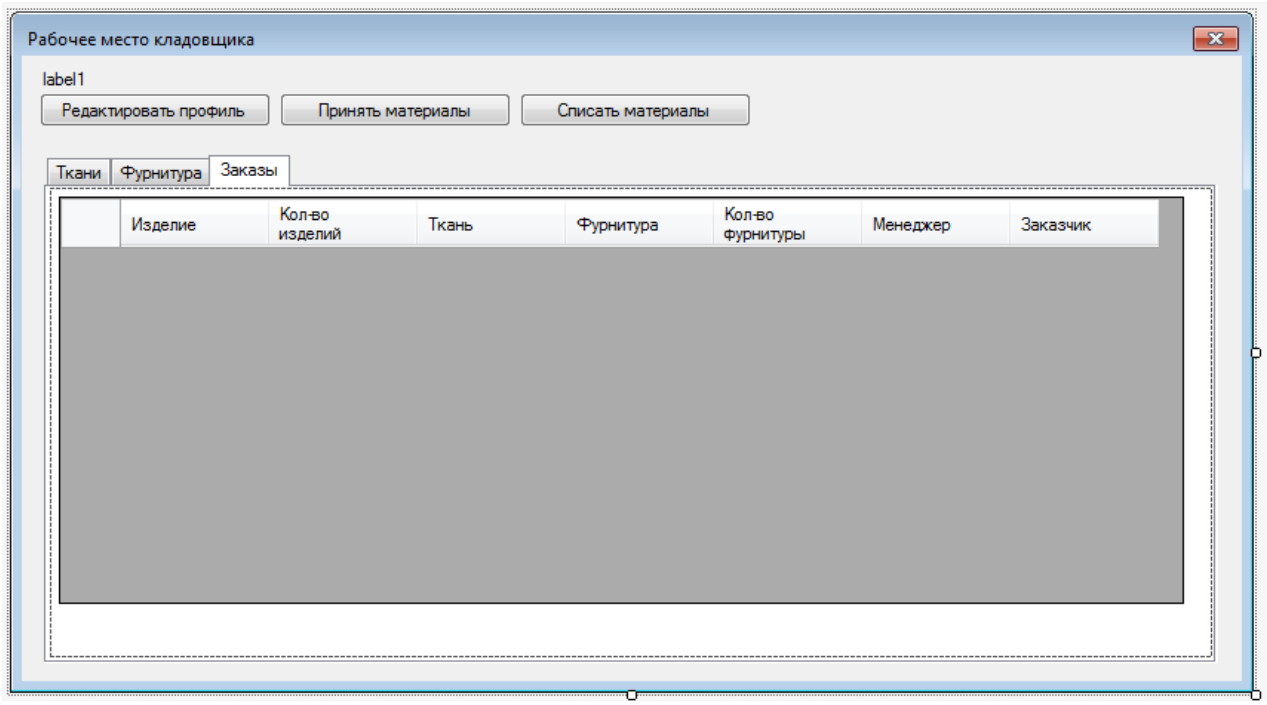


Затем отбуксируйте на вкладку поля: nametkan, colortkan, width, length, description, phototkan. В метках Label напишите названия полей по-русски. Для текстовых полей с названием, цветом, шириной, длиной и примечанием в свойстве ReadOnly установите значение True, чтобы пользователь не имел возможности редактировать содержимое этих полей. Для фотографии ткани в свойстве SizeMode установите значение Zoom, чтобы фотография ткани была вписана в PictureBox с соблюдением пропорций.

Шаг 4. Вкладка Фурнитура создается аналогично. Сделайте эту вкладку самостоятельно.

Шаг 5. Ранее мы рассматривали формирование интерфейса и описание программного кода для рабочего места заказчика. На форме с рабочим местом заказчика отображается таблица с заказами. В этой таблицу указываются названия изделий, количество изделий в заказе, название фурнитуры, фамилия менеджера, обслуживающего заказ.

На вкладке Заказы у нас должна располагаться почти такая же таблица:



В ней добавляется только одно дополнительное поле - Заказчик. Как формировать интерфейс и описывать программный код для заполнения этой таблицы, читайте в п. «Формирование интерфейса и описание программного кода для рабочего места заказчика». Некоторые отличия будут в SQL-запросе. Рассмотрим его подробнее.

```
            SqlConnection con = new SqlConnection(Form1.txtcon);
            con.Open();
            string txtquery =
@"select Izdeliya.nameizd as izd, Zakaz.countizd as countizd,
Tkani.nametkan as tkan, Furnitura.namefur as fur, Zakaz.countfur as countfur,
UsersMan.fam as manager, UsersZak.fam as zakazchik
from Zakaz, Users as UsersMan, Users as UsersZak, Tkani, Furnitura, Izdeliya
where UsersMan.iduser = Zakaz.idmanager and UsersZak.iduser = Zakaz.idzakazchik and
  Tkani.idtkan = Zakaz.idtkan
and Furnitura.idfur = Zakaz.idfur and Izdeliya.idizd = Zakaz.idizd";
```

В запросе указано:

`from Zakaz, Users as UsersMan, Users as UsersZak, Tkani, Furnitura, Izdeliya`

Данные извлекаются из таблиц: Заказы, Пользователи, Ткани, Фурнитура и Изделия. Но таблица Пользователи связана с таблицей Изделия двумя связями. Пользователь может быть заказчиком, который сформировал заказ, либо менеджером, который обрабатывает заказ. Поэтому таблица Users в нашем запросе используется под двумя именами: UsersMan - пользователи-менеджеры и UsersZak - пользователи-заказчики.

С помощью

```
select  Izdeliya.nameizd as izd,
        Zakaz.countizd as countizd,
        Tkani.nametkan as tkan,
        Furnitura.namefur as fur,
        Zakaz.countfur as countfur,
        UsersMan.fam as manager,
        UsersZak.fam as zakazchik
```

мы выбираем названия изделий из таблицы Izdeliya (даем этому полю имя izd), количество заказанных изделий из таблицы Zakaz (даем этому полю имя countizd), названия тканей из таблицы Tkani (даем этому полю имя tkan), название фурнитуры из таблицы Furnitura (даем этому полю имя fur), количество фурнитуры из таблицы Zakaz (даем этому полю имя countfur), фамилию менеджера из таблицы UsersMan (даем этому полю имя manager), фамилию заказчика из таблицы UsesrZak (даем этому полю имя zakazchik).

С помощью ряда условий мы связываем эти таблицы между собой:

```
where UsersMan.iduser = Zakaz.idmanager
  and UsersZak.iduser = Zakaz.idzakazchik
  and Tkani.idtkan = Zakaz.idtkan
  and Furnitura.idfur = Zakaz.idfur
  and Izdeliya.idizd = Zakaz.idizd
```

Таблицы UsersMan и Zakaz связываются по полям iduser и idmanager, таблицы UsersZak и Zakaz связываются по полям iduser и idzakazchik, таблицы Tkani и Zakaz по полю idtkan, таблицы Furnitura и Zakaz по полю idfur, таблицы Izdeliya и Zakaz по полю idizd.

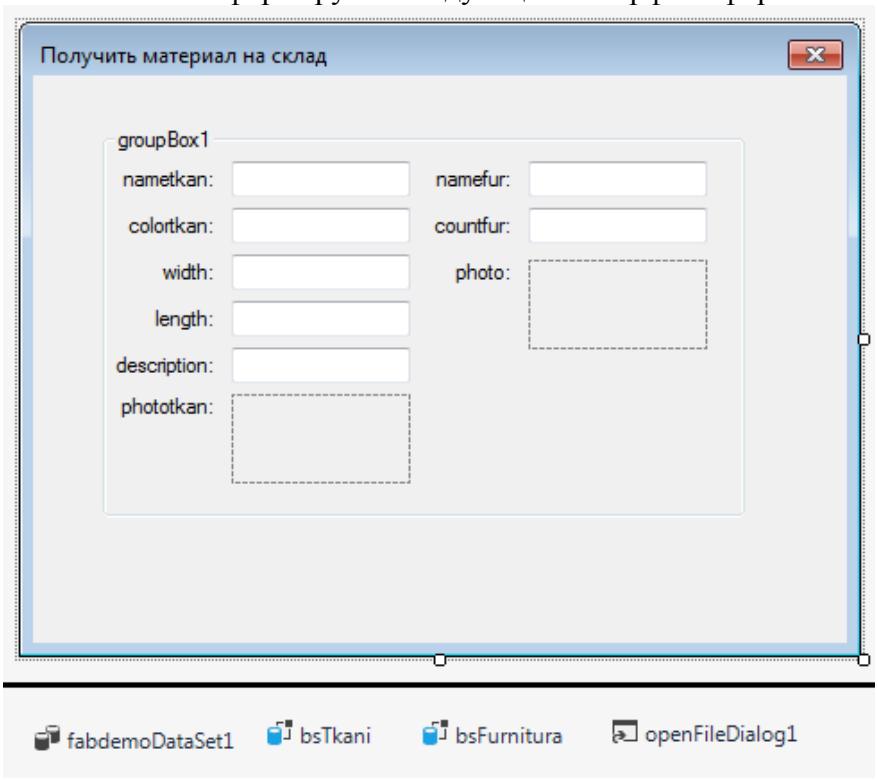
Для рабочего места заказчика у нас было еще одно условие "Zakaz.idzakazchik = " + lblIdUser.Text, чтобы были выбраны не все заказаы, к только те, которые создал заказчик с данным id. Но для рабочего места кладовщика этого не требуется.

Самостоятельно сформируйте интерфейс и опишите программный код для вкладки Заказы.

16.6. Формирование интерфейса и описание программного кода для приема материалов на склад

Шаг 1. Добавьте к проекту новую форму. Дайте ей имя FormGetMaterial. Установите для формы следующие значения свойств:
Text - Получить материал на склад (заголовок формы)
StartPosition - CenterParent (отображать форму по центру родительской формы)
MaximizeBox - False (заблокировать кнопку разворачивания формы до максимума)
MinimizeBox - False (заблокировать кнопку сворачивания формы до минимума)
FormBorderStyle - FixedDialog (диалоговое окно без кнопки системного меню, не разрешать изменять размеры формы)

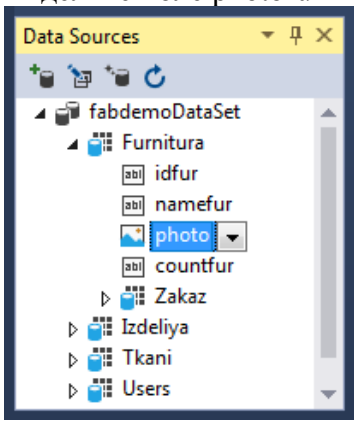
Шаг 2. Сформируйте следующий интерфейс формы:

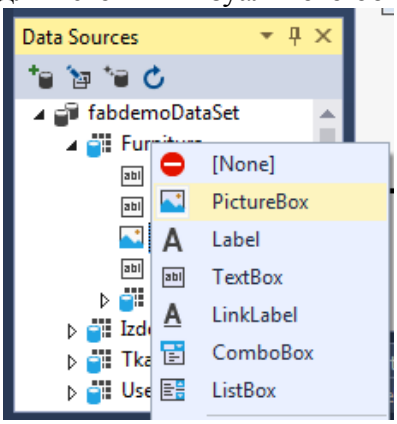


Добавьте на форму DataSet и два BindingSource. Дайте для BindingSource имена: bsUsers и bsTkani. Свяжите их с соответствующими таблицами базы данных. Для этого используйте свойства DataSource и DataMember.

Добавьте также OpenFileDialog для загрузки фотографий тканей и фурнитур.

Поставьте на форму GroupBox. В панели Data Sources разверните таблицу Furnitura, выделите поле photofur и выберите для него тип визуального объекта PictureBox.



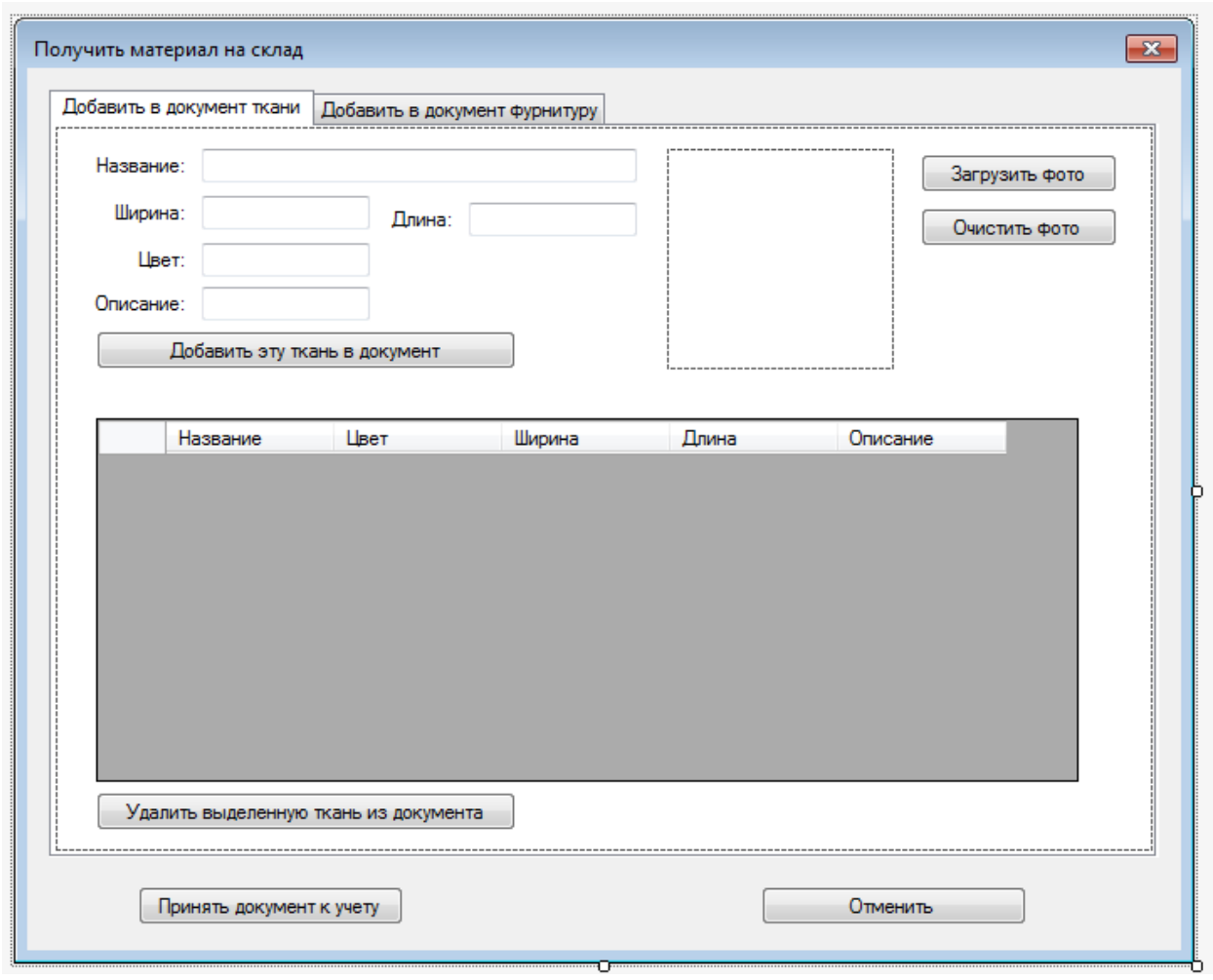


Затем отбуксируйте в GroupBox поля: namefur, countfur, photo.

Сделайте то же самое для полей таблицы Tkani.

Шаг 3. Эти текстовые поля и PictureBox будут нам нужны для того, чтобы внести в базу данных ткани и фурнитуры, которые требуется принять на склад. Но для пользователя они не должны быть видны на форме. Сдвиньте GroupBox со всем содержимым за пределы формы. Например, так: увеличьте ширину формы, сдвиньте GroupBox в правую часть формы, а затем уменьшите размеры формы.

Шаг 4. Для получения материалов на склад сделаем интерфейс формы следующего вида:



Поставьте на форму TabControl. Измените заголовки вкладок: «Добавить в документ ткани», «Добавить в документ фурнитуру».

В нижней части формы поставьте кнопки Button. Дайте кнопкам имена: «Принять документ к учету» - btnOK, «Отменить» - btnCancel. В свойстве DialogResult для кнопки отмены установите значение Cancel, чтобы по нажатию на кнопку форма закрывалась.

Шаг 5. Настройка вкладки для добавления в документ ткани.

Все визуальные объекты, которые находятся на вкладке «Добавить в документ ткани», не должны быть связаны с полями из таблиц базы данных. Они размещаются на форме и настраиваются вручную. Добавьте на эту вкладку следующие объекты:

Тип объекта	Имя объекта	Описание
TextBox	tbxNameTkan	Название ткани
TextBox	tbxWidthTkan	Ширина ткани

TextBox	tbxLengthTkan	Длина ткани
TextBox	tbxColorTkan	Цвет ткани
TextBox	tbxDescriptionTkan	Описание ткани
PictureBox	pbxPhotoTkan	Фотография ткани
DataGridView	dgvTkani	Таблица со списком тканей для приема на склад
Button	btnLoadPhotoTkan	Кнопка для загрузки фотографии ткани
Button	btnClearPhotoTkan	Кнопка для очистки фотографии ткани
Button	btnAddTkan	Кнопка для добавления ткани в список
Button	btnDelTkan	Кнопка для удаления ткани из списка

Для pbxPhotoTkan в свойстве SizeMode установите значение Zoom, чтобы фотография была вписана в размеры PictureBox с соблюдением пропорций.

Для DataGridView столбцы создаются вручную и заполняются из программного кода. Щелкните правой кнопкой на DataGridView, выберите пункт Edit Columns. С помощью кнопки Add добавьте пять столбцов, в свойстве HeaderText измените заголовки столбцов.

После того, как закончите формирование столбцов, задайте для DataGridView следующие свойства:
AllowUserToAddRows - False (чтобы пользователь не мог добавлять строки в таблицу)
AllowUserToDeleteRows - False (чтобы пользователь не мог удалять строки из таблицы)
ReadOnly - True (чтобы пользователь не мог редактировать ячейки таблицы)
SelectionMode - FullRowSelect (чтобы при выделении в таблице выделялась сразу вся строка)
MultiSelect - False (чтобы можно было выделить только один элемент в таблице)

Шаг 6. Интерфейс вкладки «Добавить в документ фурнитуру» сформируйте самостоятельно таким же образом, как и вкладку «Добавить в документ ткань».

Шаг 7. Переключитесь в программный код формы. При приеме тканей или фурнитур, они не сразу попадают в базу данных. Сначала создается список добавляемых тканей и список добавляемых фурнитур (на соответствующих вкладках). И только по нажатию на кнопку «Принять документ к учету» ткани и фурнитуры попадают в базу данных. В классе формы после конструктора опишите структуру для хранения данных об одной ткани и создайте список таких структур.

```
// структура для хранения данных об одной ткани
struct tkan
{
    public string nametkan, colortkan, widthtkan, lengthtkan, description;
    public Image phototkan;
}
// список для хранения данных о добавляемых тканях
List<tkan> lstTkani = new List<tkan>();
```

Шаг 8. По нажатию на кнопку «Загрузить фото» нужно открыть диалоговое окно для выбора файла и загрузить в PictureBox фотографию ткани из файла.

```
private void btnLoadPhotoTkan_Click(object sender, EventArgs e)
{
    // если пользователь выбрал файл с фотографией, ...
    if (openFileDialog1.ShowDialog() == DialogResult.OK)
        // загрузить фотографию в PictureBox
        pbxPhotoTkan.Image = Image.FromFile(openFileDialog1.FileName);
}
```

Шаг 9. По нажатию на кнопку «Очистить фото» нужно вывести диалоговое окно для подтверждения очистки фотографии и очистить фотографию, если пользователь это подтверждает.

```
private void btnClearPhotoTkan_Click(object sender, EventArgs e)
{
    // диалоговое окно с вопросом
    DialogResult rez = MessageBox.Show("Очистить фотографию?", "Внимание!",
      MessageBoxButtons.YesNo, MessageBoxIcon.Question);
    // если подтверждается очистка, ...
    if (rez == DialogResult.Yes)
        // то очистить фотографию в PictureBox
        pbxPhotoTkan.Image = null;
}
```

Шаг 10. По нажатию на кнопку «Добавить эту ткань в документ» нужно в список lstTkani добавить новый элемент с данными, прочитанными из текстовых полей и PictureBox на форме.

```
private void btnAddTkan_Click(object sender, EventArgs e)
{
    tkan tk1;  // создать переменную для добавляемой ткани
    // прочитать в переменную данные из текстовых полей
    tk1.nametkan = tbxNameTkan.Text;
    tk1.colortkan = tbxColorTkan.Text;
    tk1.widthtkan = tbxWidthTkan.Text;
    tk1.lengthtkan = tbxLengthTkan.Text;
    tk1.description = tbxDescriptionTkan.Text;
    // взять фотографию ткани из PictureBox
    tk1.phototkan = pbxPhotoTkan.Image;
    lstTkani.Add(tk1);  // добавить новую ткань в lstTkani
    // добавить новую ткань в DataGridView
    dgvTkani.Rows.Add(tk1.nametkan, tk1.colortkan, tk1.widthtkan,
      tk1.lengthtkan, tk1.description);
}
```

Здесь вначале создается переменная tk1. Затем в поля этой переменной считываются данные из текстовых полей и PictureBox. Далее эта переменная добавляется в список lstTkani, а данные выводятся в DataGridView.

Шаг 11. По нажатию на кнопку «Удалить выделенную ткань из документа» нужно удалить выделенную строку из DataGridView и элемент с таким же номером в списке lstTkani.

```
private void btnDelTkan_Click(object sender, EventArgs e)
{
    // номер строки, выделенной в DataGridView с тканями
    int numtkan = dgvTkani.CurrentRow.Index;
    // если пользователь выделил в списке какую-то ткань
    if (numtkan >= 0)
    {
        // удалить выделенную ткань из обоих списков
        lstTkani.RemoveAt(numtkan);
        dgvTkani.Rows.RemoveAt(numtkan);
    }
}
```

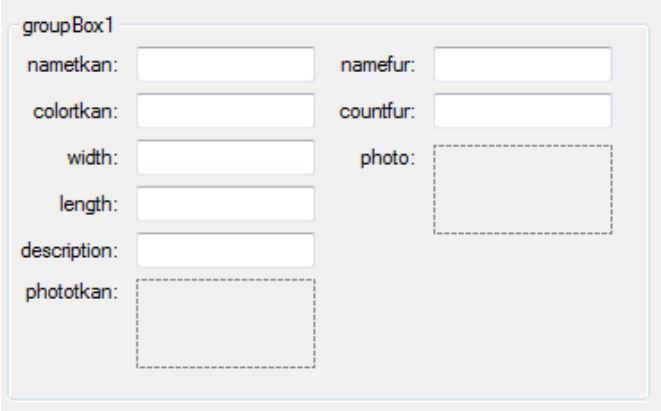
Сначала в переменную numtkan записывается номер выделенной строки из DataGridView. Затем выполняется проверка. Если пользователь действительно выделил какую-то строку в таблице, то удалить элемент с этим номером из lstTkani и строку с этим номером из DataGridView.

Шаг 12. Самостоятельно опишите программный код, необходимый для вкладки «Добавить в документ фурнитуру».

Шаг 13. По нажатию на кнопку «Принять документ к учету» нужно внести в базу данных сведения о тканях и фурнитурах, которые принимаются на склад. Рассмотрим как это делается на примере тканей.

```
private void btnOK_Click(object sender, EventArgs e)
{
    // выполнить перебор всех тканей в добавляемом документе
    foreach(tkan tk1 in lstTkani)
    {
        // для каждой ткани в таблицу Tkani добавить новую запись
        bsTkani.AddNew();
        // в текстовые поля и PictureBox, связанные с таблицей Tkani,
        // вывести данные очередной ткани из списка lstTkani
        nametkanTextBox.Text = tk1.nametkan;
        colortkanTextBox.Text = tk1.colortkan;
        widthTextBox.Text = tk1.widthtkan;
        lengthTextBox.Text = tk1.lengthtkan;
        phototkanPictureBox.Image = tk1.phototkan;
    }
    // сохранить изменения в таблице Tkani
    bsTkani.EndEdit();
    tkaniTableAdapter.Update(fabdemoDataSet1.Tkani);
```

Вспомните, что мы начинали формирование интерфейса формы с того, что добавили на форму GroupBox, в который из DataSources вытащили поля таблиц Tkani и Furnitura. А затем отодвинули этот GroupBox за край формы.



Эти текстовые поля и PictureBox связаны с полями из базы данных. Следовательно, любые данные, которые в них записываются, попадают в таблицы Tkani и Furnitura.

В нашем программном коде мы с помощью цикла foreach выполняем перебор элементов списка lstTkani. Для каждой ткани из списка мы добавляем в таблицу Tkani новую запись, затем в текстовые поля и PictureBox выводим данные очередной ткани из списка.

По завершении цикла мы сохраняем изменения в таблице Tkani.

Для добавления в базу данных фурнитур, которые мы вводили на вкладке «Добавить в документ фурнитуру», программный код опишите самостоятельно.

Внимание! По нажатию на кнопку «Принять документ к учету» в базу данных нужно добавлять и ткани и фурнитуры. Для добавления в базу данных фурнитур не нужна отдельная кнопка.

Не забудьте, что после добавления в базу данных тканей и фурнитур нужно закрыть форму.

Шаг 14. Вернитесь на форму FormKladovschik. Для кнопки «Принять материалы» опишите следующий код.

```
private void btnGetMaterial_Click(object sender, EventArgs e)
{
    // создать форму для добавления материалов на склад
    FormGetMaterial frm = new FormGetMaterial();
    frm.ShowDialog();  // открыть форму для добавления материалов
    // после добавления материалов загрузить из базы данных
    // обновленные записи
    this.furnituraTableAdapter.Fill(this.fabdemoDataSet1.Furnitura);
    this.tkaniTableAdapter.Fill(this.fabdemoDataSet1.Tkani);
}
```

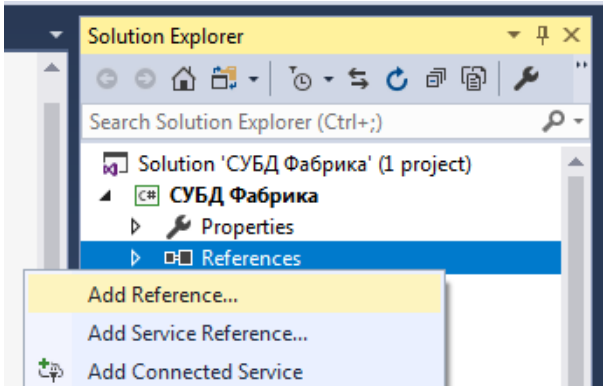
Вначале здесь создается и открывается форма для получения материалов на склад. После того, как форма для получения материалов закроется, нужно загрузить базы данных обновленные записи таблиц Tkani и Furnitura.

16.7. Формирование интерфейса и описание программного кода для списания материалов

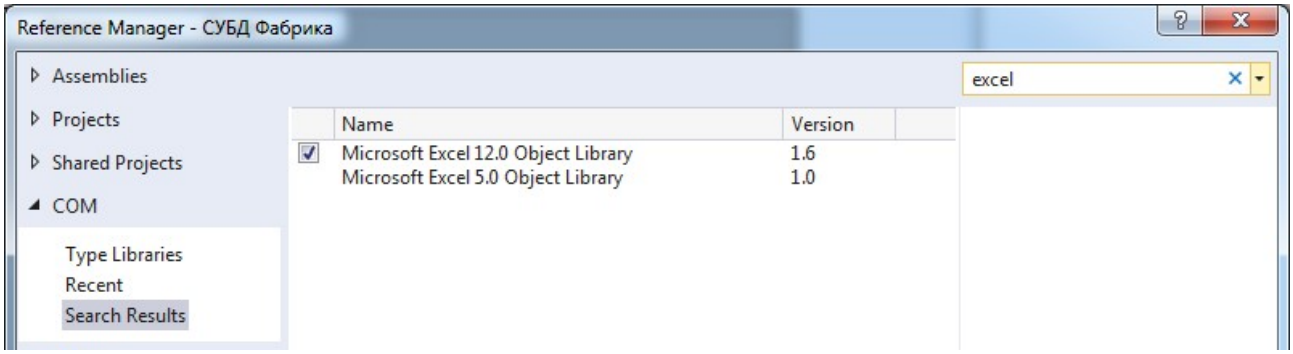
Шаг 1. Добавьте к проекту новую форму. Дайте ей имя FormSpisMaterial. Установите для формы следующие значения свойств:
Text - Списание материалов (заголовок формы)
StartPosition - CenterParent (отображать форму по центру родительской формы)
MaximizeBox - False (заблокировать кнопку разворачивания формы до максимума)
MinimizeBox - False (заблокировать кнопку сворачивания формы до минимума)
FormBorderStyle - FixedDialog (диалоговое окно без кнопки системного меню, не разрешать изменять размеры формы)

Шаг 2. В процессе списания нужно будет формировать документ в виде таблицы в Microsoft Excel. Нам нужно подключить библиотеку для работы с Microsoft Excel.

В панели Solution Explorer щелкните правой кнопкой на узле References и выберите пункт Add Reference...



Выберите раздел COM и выполните поиск библиотеки Excel. Если будет найдено несколько версий библиотеки Microsoft Excel, то подключайте ту, которая имеет больший номер.

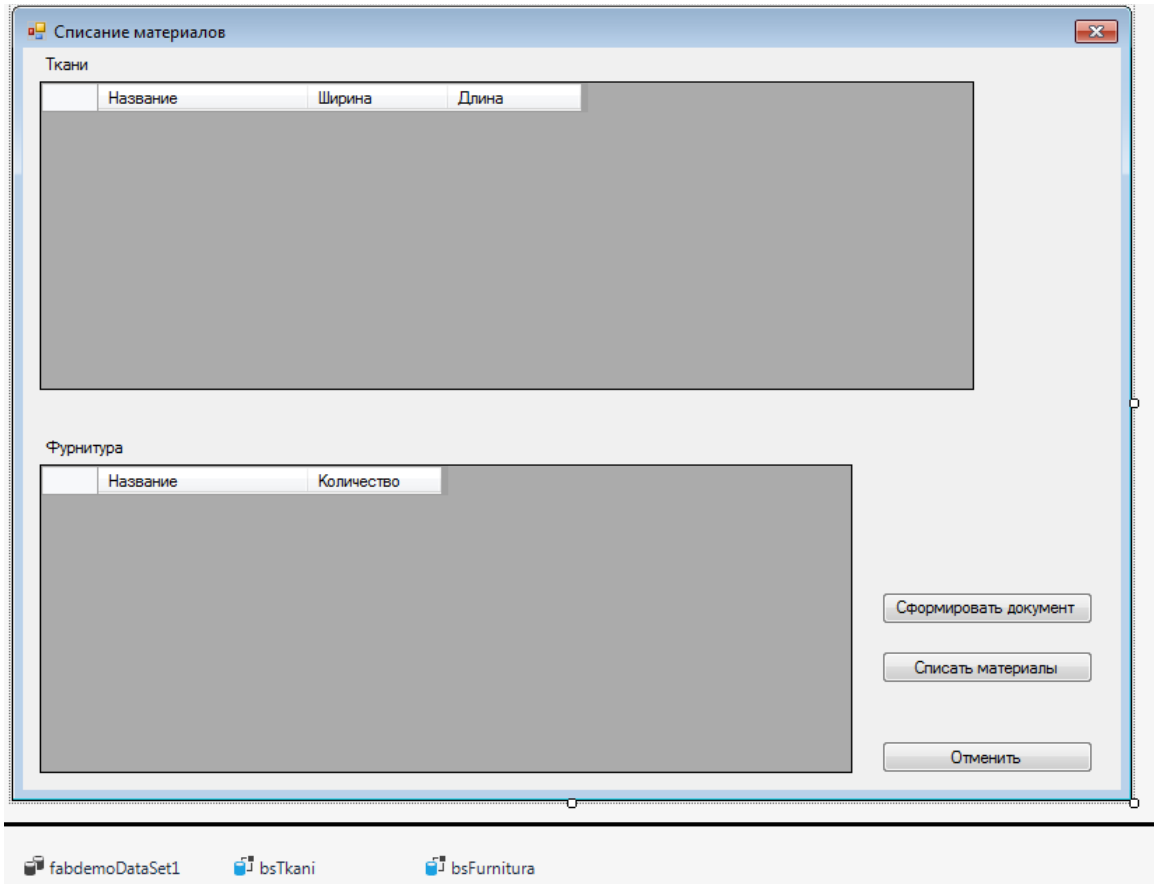


В программном коде подключите пространство имен для работы с этой библиотекой:

```
using Excel = Microsoft.Office.Interop.Excel;
```

Здесь мы не только подключили пространство имен, но и дали ему имя Excel. Это нужно для того, чтобы в программном коде не писать длинный путь доступа к какому-либо объекту из пространства имен.

Шаг 2. Сформируйте следующий интерфейс формы:



Добавьте на форму DataSet и два BindingSource. Дайте для BindingSource имена: bsUsers и bsTkani. Свяжите их с соответствующими таблицами базы данных. Для этого используйте свойства DataSource и DataMember.

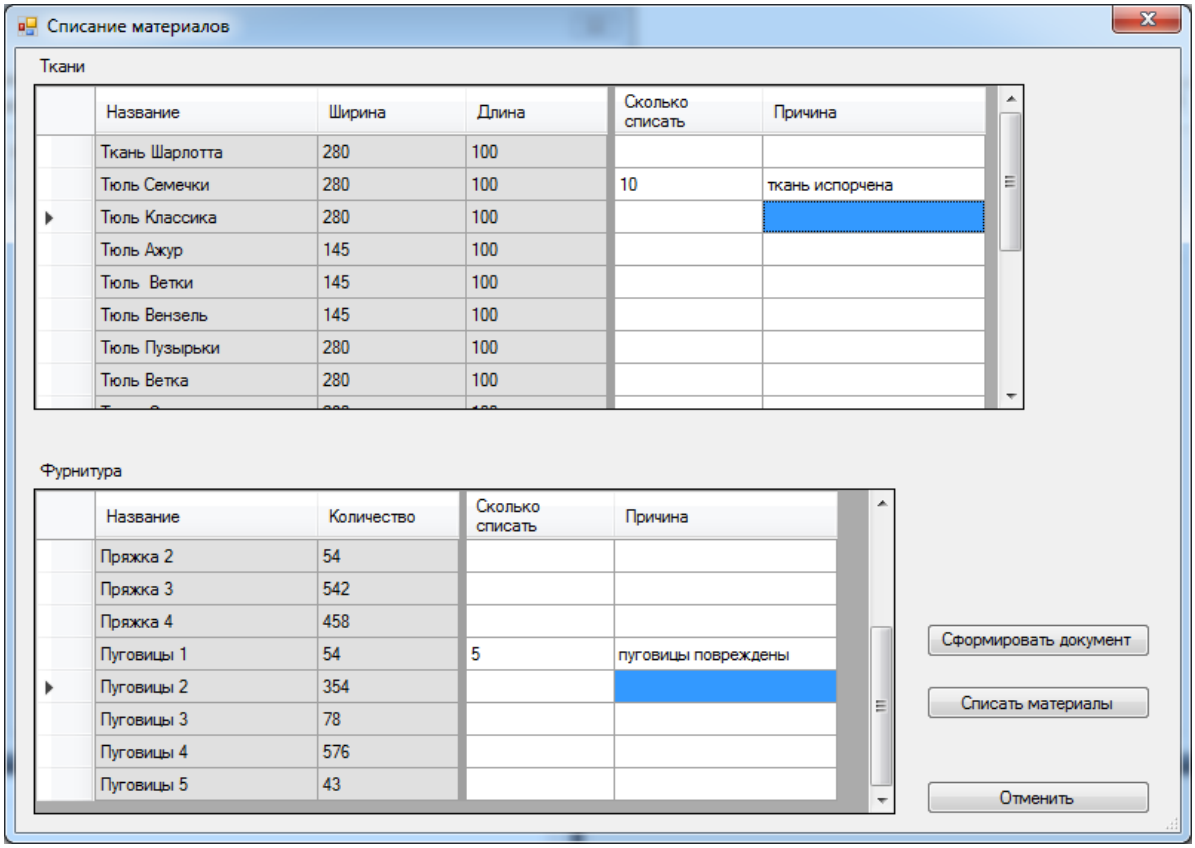
Для bsTkani в свойстве Filter установите length > 0, для bsFurnitura в свойстве Filter установите countfur > 0. Если длина рулона ткани равна нулю, или количество фурнитур равно нулю, значит их нет не складе и их нельзя использовать для списания.

Поставьте на форму два DataGridView и свяжите их с таблицами Tkani и Furnitura. Дайти им имена dgvTkan и dgvFurnitura. Щелчком правой кнопкой мыши на DataGridView откройте редактор столбцов (Edit Columns...) и настройте столбцы так, как указано на изображении формы для списания.

Для DataGridView задайте следующие свойства:
AllowUserToAddRows - False (чтобы пользователь не мог добавлять строки в таблицу)
AllowUserToDeleteRows - False (чтобы пользователь не мог удалять строки из таблицы)

Для кнопок Button дайте имена: btnToExcel - сформировать документ, btnSpis - списать материалы, btnCancel - отменить. Для кнопки отмены в свойстве DialogResult установите значение Cancel, чтобы по нажатию на нее форма закрывалась.

Шаг 3. Нужно будет выполнить дополнительную настройку DataGridView.



В столбцы Название, Ширина, Длина, Количество пользователь не может вносить изменения, поэтому в свойстве ReadOnly для каждого из установите значение True.

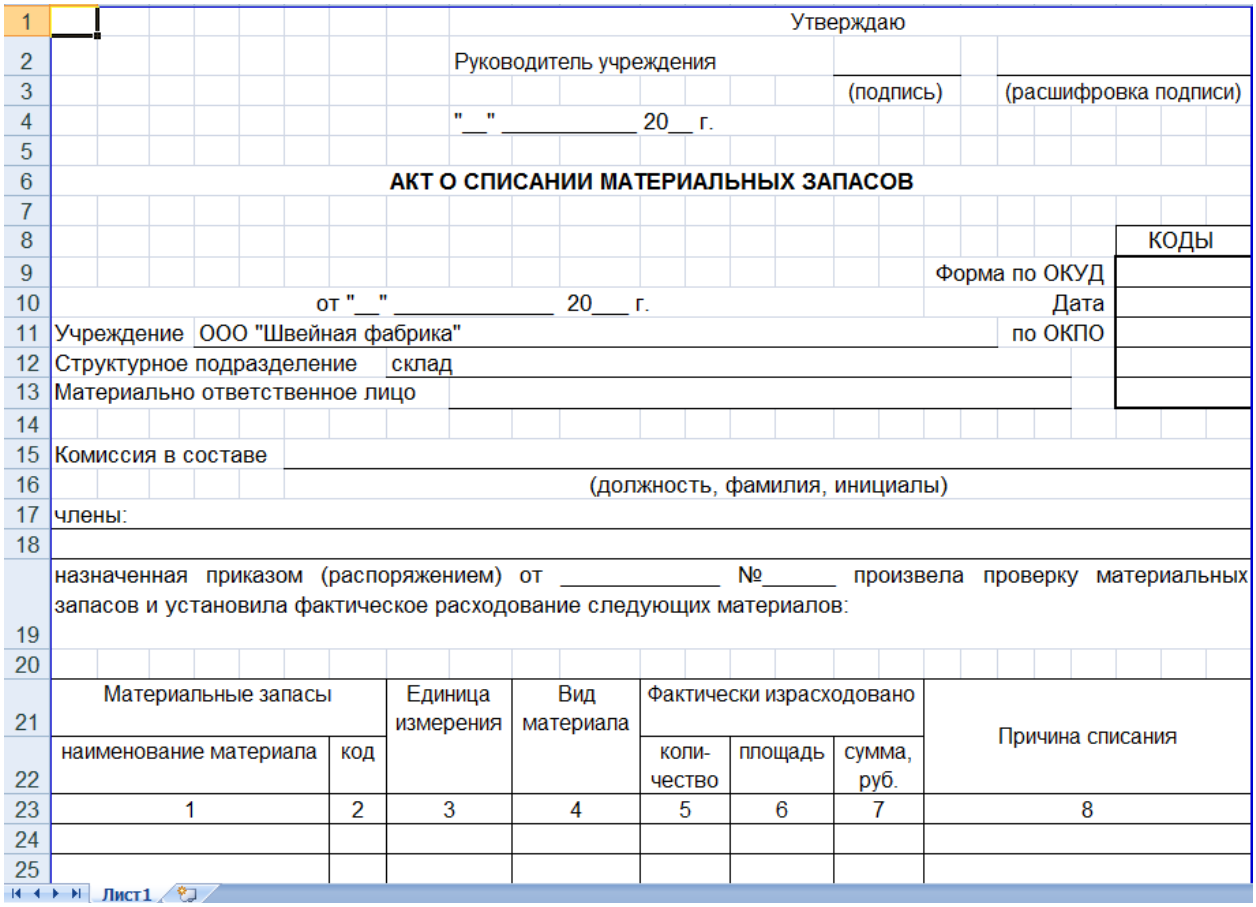
Измените фоновый цвет этих столбцов с помощью свойства DefaultCellStyle►Back Color в редакторе столбцов (Edit Columns).

После столбцов Длина и Количество поставьте разделительную линию шириной в 5 пикселей (свойство DividerWidth в редакторе столбцов).

В редакторе столбцов с помощью кнопки Add добавьте в DataGridView два столбца, не привязанные к базе данных (Сколько списать и Причина). Эти столбцы должны быть доступны для редактирования.

Шаг 4. Для формирования документа для списания нам потребуется шаблон. Пусть шаблон имеет имя «Списание.xlsx» и располагается в папке bin \ debug нашего проекта.

Внимание! Шаблон не нужно создавать самим. Он прилагается к материалам задания для программирования СУБД «Швейная фабрика».



Утверждаю

Руководитель учреждения _____ _____

(подпись) (расшифровка подписи)

"__" _____ 20__ г.

АКТ О СПИСАНИИ МАТЕРИАЛЬНЫХ ЗАПАСОВ

		КОДЫ
	Форма по ОКУД	
от "__" _____ 20___ г.	Дата	
Учреждение ООО "Швейная фабрика"	по ОКПО	
Структурное подразделение склад		
Материально ответственное лицо _____		

Комиссия в составе _____

(должность, фамилия, инициалы)

члены: _____

назначенная приказом (распоряжением) от _____ №_____ произвела проверку материальных запасов и установила фактическое расходование следующих материалов:

Материальные запасы		Единица измерения	Вид материала	Фактически израсходовано			Причина списания
наименование материала	код			коли-чество	площадь	сумма, руб.	
1	2	3	4	5	6	7	8

Лист1

Шаг 5. Рассмотрим программный код для кнопки «Сформировать документ».

```
private void btnToExcel_Click(object sender, EventArgs e)
{
    // создать объект для связи с приложением Excel
    Excel.Application exapp = new Excel.Application();
    // сделать приложение Excel видимым
    exapp.Visible = true;
    // открыть в Excel шаблон, находящийся в папке с exe-файлом в режиме только для чтения
    exapp.Workbooks.Open(Application.StartupPath + "\\Списание.xlsx", Type.Missing, true);
    // создать переменную list1 для связи с первым листом рабочей книги
    Excel.Worksheet list1 = (exapp.Worksheets.get_Item(1));
```

Здесь вначале создается объект для работы с приложением Microsoft Excel. При этом запускается само приложение Microsoft Excel, но на экране пока не отображается. Затем это приложение становится видимым и в нем открывается шаблон, находящийся в папке с exe-файлом нашего приложения. Параметр Type.Missing обозначает пустой параметр. Параметр true обозначает то, что файл должен открыться в режиме только для чтения. Наш документ будет находиться на первом листе рабочей книги. Поэтому мы создаем переменную list1 для работы с первым листом.

Дальше нам нужно выполнять перебор тканей в DataGridView и выводить те ткани, которые списываются, в таблицу Excel. Следует учитывать, что в DataGridView нумерация строк начинается от нуля, а в таблице Excel данные нужно выводить, начиная с 24 строки. Следовательно, нам потребуется отдельный счетчик для строк в таблице Excel. Будем для счетчика использовать переменную rowexcel.

```
int rowexcel = 24;  // в Excel начинать выводить с 24 строки
// перебор тканей в DataGridView
for (int i = 0; i <= dgvTkan.RowCount - 1; i++)
{
    int countspis = 0;
    try
    {
        // сколько списать i-й ткани
        countspis = Convert.ToInt32(dgvTkan.Rows[i].Cells[3].Value);
    }
    catch { }
    // если человек указал 0, или не указал ничего,
    // перейти к следующей ткани
    if (countspis == 0) continue;
```

В цикле for будем выполнять перебор строк в DataGridView. Для списываемых тканей пользователь указывает количество списываемой ткани в столбце №3. Если пользователь в 3-м столбце указал нулевое значение, или не указал ничего, то с помощью команды continue мы переходим к следующей итерации цикла (переходим к обработке следующей ткани в DataGridView).

```
    // если человек ввел ненулевое кол-во для списания,
    // добавить i-ю ткань в документ
    list1.get_Range("A" + rowexcel).Value = dgvTkan.Rows[i].Cells[0].Value;
    list1.get_Range("H" + rowexcel).Value = "см";
    list1.get_Range("J" + rowexcel).Value = "ткань";
    // количество ткани для списания
    list1.get_Range("L" + rowexcel).Value = countspis;
    int widthtkan = Convert.ToInt32(dgvTkan.Rows[i].Cells[1].Value);
    // площадь ткани для списания
    list1.get_Range("N" + rowexcel).Value = countspis * widthtkan;
    // причина списания
    list1.get_Range("R" + rowexcel).Value = dgvTkan.Rows[i].Cells[4].Value;
    rowexcel++;  // перейти в Excel к следующей строке
}
```

Если же пользователь указал какое-то количество ткани для списания, то выполняется дальнейший программный код. В нем с помощью get_Range().Value мы обращается к содержимому ячейки таблицы Excel по указанным координатам. Строка у нас будет равна переменной rowexcel. После вывода данных очередной списываемой ткани увеличиваем значение rowselect.

Самостоятельно опишите программный код, который выводит в таблицу Excel информацию о списываемой фурнитуре. Обратите внимание, что списываемые ткани и фурнитуры должны выводиться в один и тот же документ по нажатию на кнопку «Сформировать документ». Например:

Материальные запасы		Единица измерения	Вид материала	Фактически израсходовано			Причина списания
наименование материала	код			коли-чество	площадь	сумма, руб.	
1	2	3	4	5	6	7	8
Тюль Классика		см	ткань	10	2800		испорчена
Тюль Ажур		см	ткань	15	2175		испорчена
Пряжка 3		шт	фурнитура	4			сломаны
Пряжка 4		шт	фурнитура	3			сломаны

Шаг 6. По нажатию на кнопку «Списать материалы» нужно внести изменения в базу данных.

```
private void btnSpis_Click(object sender, EventArgs e)
{
    DialogResult rez = MessageBox.Show("Вы уверены, что нужно списать указанное
      количество материалов?", "Внимание!", MessageBoxButtons.YesNo,
      MessageBoxIcon.Question);
    // если пользователь отказался списать, то выйти из процедуры
    if (rez == DialogResult.No) return;

    // перебор тканей в DataGridView
    for (int i = 0; i <= dgvTkan.RowCount - 1; i++)
    {
        int countspis = 0;
        try
        {
            // сколько списать i-й ткани
            countspis = Convert.ToInt32(dgvTkan.Rows[i].Cells[3].Value);
        }
        catch { }
        // если человек указал 0, или не указал ничего,
        // перейти к следующей ткани
        if (countspis == 0) continue;
```

Вначале нужно получить у пользователя подтверждение того, что он действительно будет выполнять списание материалов.

```
// перебор тканей в DataGridView
for (int i = 0; i <= dgvTkan.RowCount - 1; i++)
{
    int countspis = 0;
    try
    {
        // сколько списать i-й ткани
        countspis = Convert.ToInt32(dgvTkan.Rows[i].Cells[3].Value);
    }
    catch { }
    // если человек указал 0, или не указал ничего,
    // перейти к следующей ткани
    if (countspis == 0) continue;

    // если человек ввел ненулевое кол-во для списания,
    // списать ткань
    int lengthtkan = Convert.ToInt32(dgvTkan.Rows[i].Cells[2].Value);
    // уменьшить кол-во ткани на складе на кол-во списываемой ткани
    lengthtkan -= countspis;
    // записать в столбец с длиной рулона ткани на складе новое значение
    dgvTkan.Rows[i].Cells[2].Value = lengthtkan;
}
```

Затем в цикле for будем выполнять перебор строк в DataGridView. Для списываемых тканей пользователь указывает количество списываемой ткани в столбце №3. Если пользователь в 3-м столбце указал нулевое значение, или не указал ничего, то с помощью команды continue мы переходим к следующей итерации цикла (переходим к обработке следующей ткани в DataGridView).

Количество ткани на складе находится в столбце №2, количество ткани для списания находится в столбце №3. Если пользователь указал ненулевое количество ткани для списания, то нужно уменьшить количество ткани на складе.

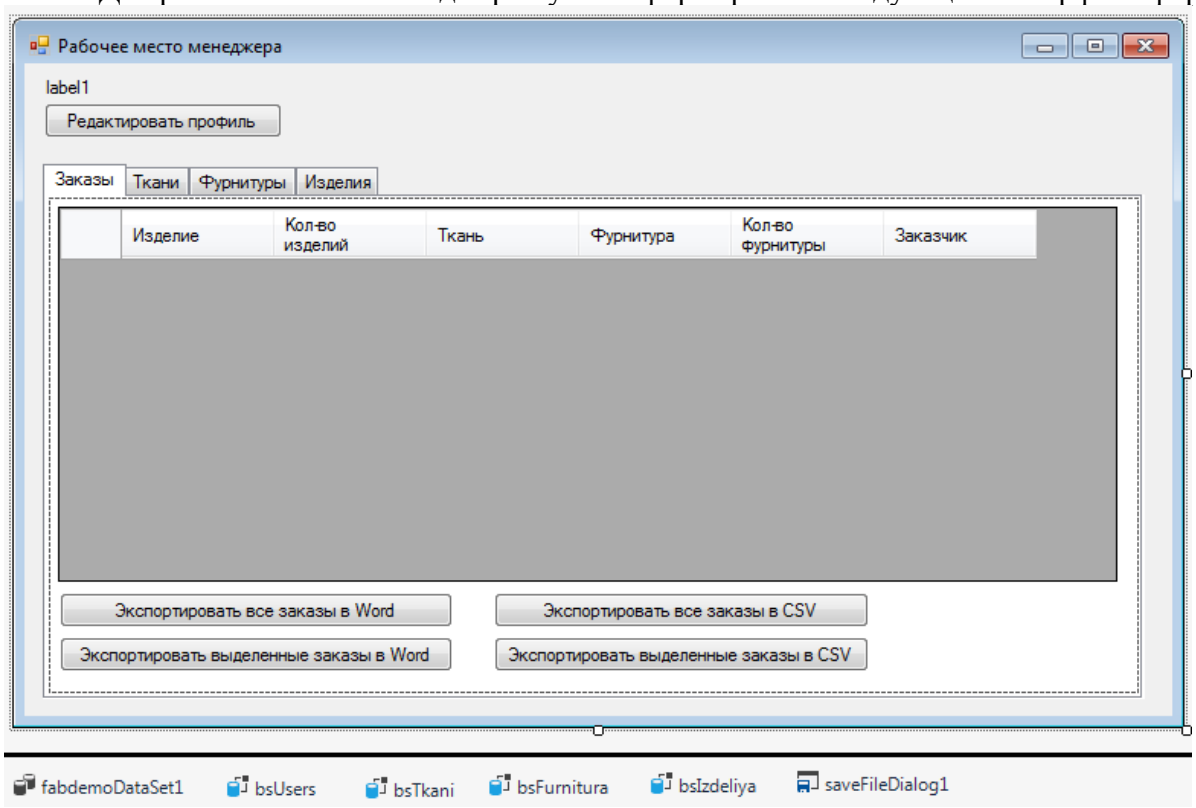
По завершении перебора списка тканей нужно сохранить изменения в базе данных.

```
// после списания сохранить изменения в базе данных
bsTkani.EndEdit();
tkaniTableAdapter.Update(fabdemoDataSet1.Tkani);
```

Для списания фурнитуры напишите программный код самостоятельно. Не забудьте закрыть форму после окончания процесса списания.

16.8. Формирование интерфейса и описание программного кода для рабочего места менеджера

Для рабочего места менеджера нужно сформировать следующий интерфейс формы:

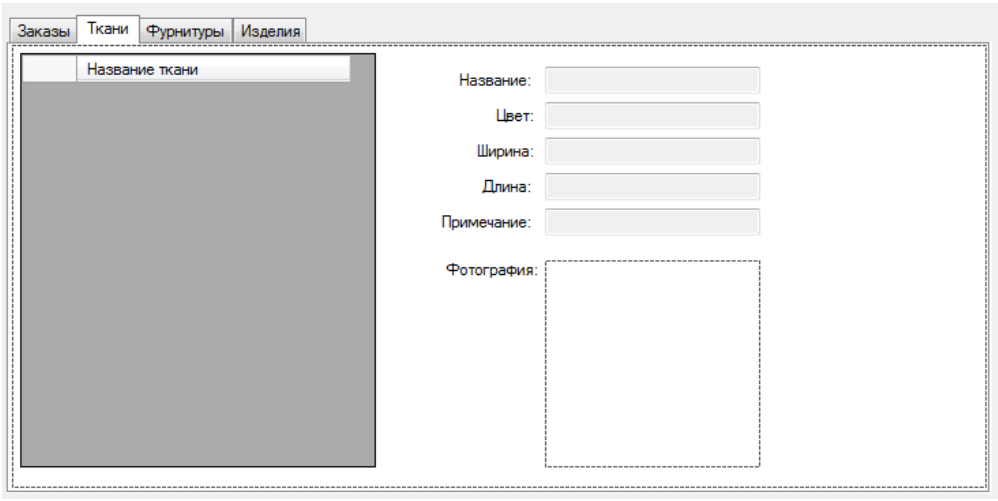


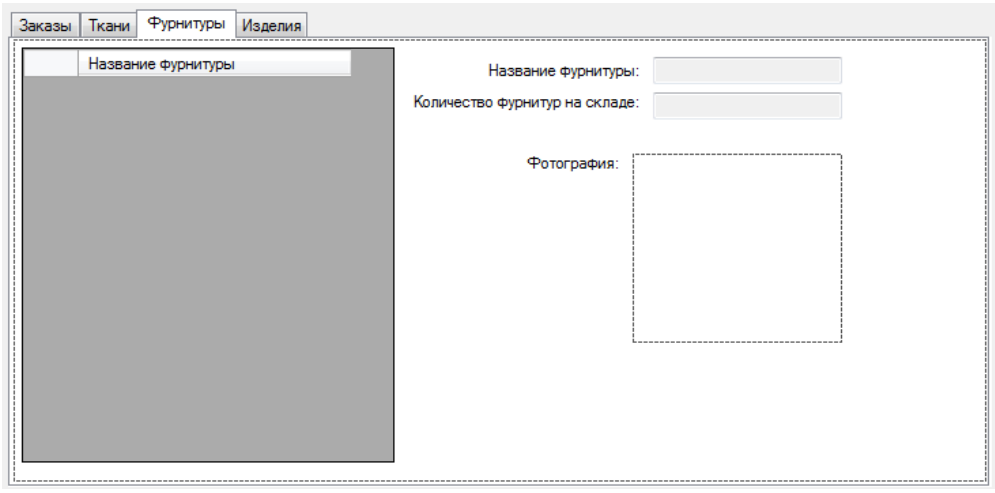
Для вкладок используйте объект TabControl.

Шаг 1. Добавьте к проекту объект DataSet. Добавьте четыре BindingSources. Свяжите их с таблицами Users, Tkani, Furnitura, Izdeliya.

Шаг 2. На вкладке Заказы нужно сформировать DataGridView, в котором будут отображаться заказы, обрабатываемые авторизовавшимся менеджером. Когда мы программировали рабочее место заказчика, то мы использовали программный код, который формирует похожую таблицу. Посмотрите, как это было реализовано для заказчика и сделайте аналогично.

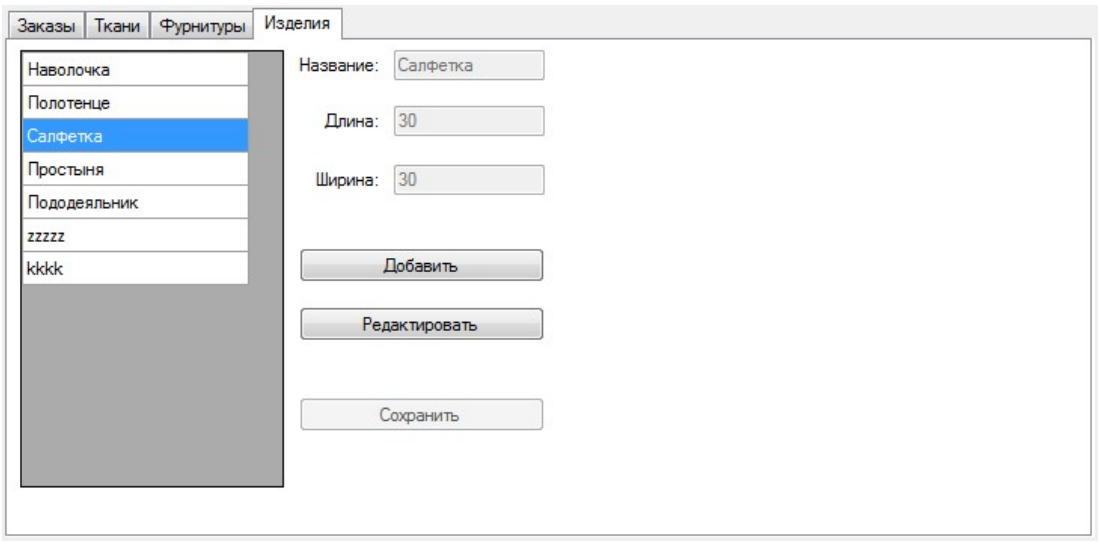
Шаг 3. На вкладках Ткани и Фурнитуры у нас должен отображаться список тканей и фурнитур без возможности редактирования. При выделении в списке какой-либо ткани или фурнитуры должны отображаться подробные сведения о тканях и фурнитурах. Также без возможности редактирования.





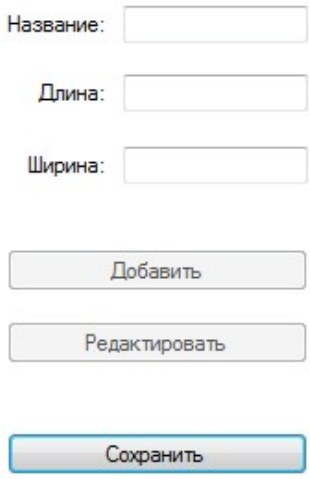
Как сделать отображение тканей и фурнитур в таком виде, мы рассматривали при формировании интерфейса формы для добавления нового заказа и при формировании интерфейса формы рабочего места кладовщика.

Шаг 4. На вкладке Изделия у нас должен отображаться список изделий, которые изготавливает швейная фабрика, без возможности редактирования. При выделении какого-либо изделия в списке должны отображаться подробные сведения об этом изделии, также без возможности редактирования.



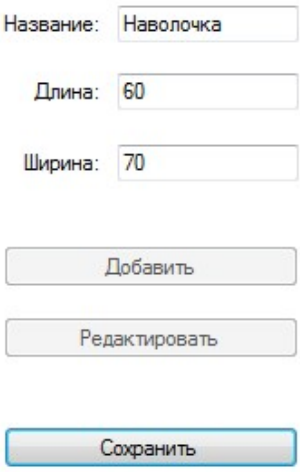
Интерфейс этой вкладки делается так же, как и интерфейс вкладок для тканей и фурнитур. Но для изделий нужно предусмотреть возможность добавлять и редактировать записи.

На начало работы должны быть доступны кнопки Добавить и Редактировать. Кнопка Сохранить должна быть заблокирована.

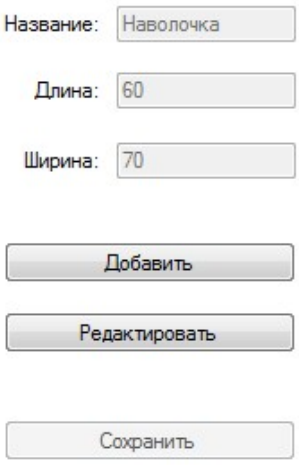


По нажатию на кнопку Добавить в таблицу Izdeliya должна добавиться новая запись. При этом нужно заблокировать кнопки Добавить и Редактировать. Заблокировать DataGridView, разблокировать кнопку Сохранить и разблокировать доступ к текстовым полям.

Для добавления новой записи в таблицу Izdeliya используйте команду: bsIzdeliya.AddNew();



По нажатию на кнопку Редактировать нужно заблокировать кнопки Добавить и Редактировать. Заблокировать DataGridView, разблокировать кнопку Сохранить и разблокировать доступ к текстовым полям. Новую запись при этом добавлять в таблицу не нужно.



По нажатию на кнопку Сохранить нужно заблокировать кнопку Сохранить, заблокировать доступ к текстовым полям, разблокировать кнопки Добавить и Редактировать, разблокировать DataGridView.

Для сохранения изменений в базе данных используйте команды:

```
bsIzdeliya.EndEdit();
izdeliyaTableAdapter.Update(fabdemoDataSet1.Izdeliya)
;
```

Самостоятельно опишите программный код для выполнения перечисленных действий.

16.9. Программирование экспорта заказов в файлы формата CSV и в Microsoft Word

Информация о том, как реализовать экспорт данных в файлы формата CSV и в приложение Microsoft Word описана в темах «Модуль экспорта данных в файлы формата CSV» и «Модуль экспорта данных в Microsoft Word». Самостоятельно опишите программный код для выполнения экспорта заказов.