МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Информационные технологии»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к проведению практических занятий
по дисциплине ««Учебная технологическая (проектно-технологическая) практика»»

Автор
Барашко Е.Н.

Ростов-на-Дону,

2020

ПРАКТИЧЕСКАЯ РАБОТА № 1
« Создание проекта и подключение к нему файла данных»

Цель: Изучить возможности проекты и подключать к ним файлы данных.

Общая информация

Microsoft SQL Server — система управления реляционными базами данных (СУРБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка.

Для лучшего понимания можно разделить SQL Server на два крупных модуля – Database Engine, отвечающий за управление базами данных, хостящимися на данном сервере; и Business Intelligence, обеспечивающий глубокий и всесторонний анализ хранящейся информации.

Database Engine технически реализован в виде системного сервиса, позволяющего добавлять, извлекать и изменять хранящуюся информацию. Именно он обеспечивает реализацию высокопроизводительных систем с использованием механизмов транзакций (OLTP) и аналитики (OLAP). Database Engine при этом является многомпонентным и состоит из следующих модулей:

- Storage Engine, или механизм хранения данных, обеспечивает физическую запись информации в файловые структуры;
- Подсистема безопасности (Security Subsystem) позволяет контролировать доступ пользователей к хранящейся информации;
- Программные интерфейсы (Programming Interfaces) соединяющие нити SQL Server, посредством которых возможно взаимодействие с сервером. Краеугольный камень программных интерфейсов – язык Transact-SQL;
- Service Broker позволяет организовать очередь сообщений внутренними средствами SQL Server. Это дает возможность работы с асинхронными запросами к базе данных;
- Агент SQL Server (SQL Server Agent) это механизм оперативного мониторинга состояния сервера;
- Система репликации (Replication) позволяет организовать распределенные информационные хранилища, состоящие из взаимодействующих удаленных баз данных;
- механизм отказоустойчивости (High Availability), позволяющий организовать бесперебойную работу с данными вне зависимости от нагрузки на сервер.

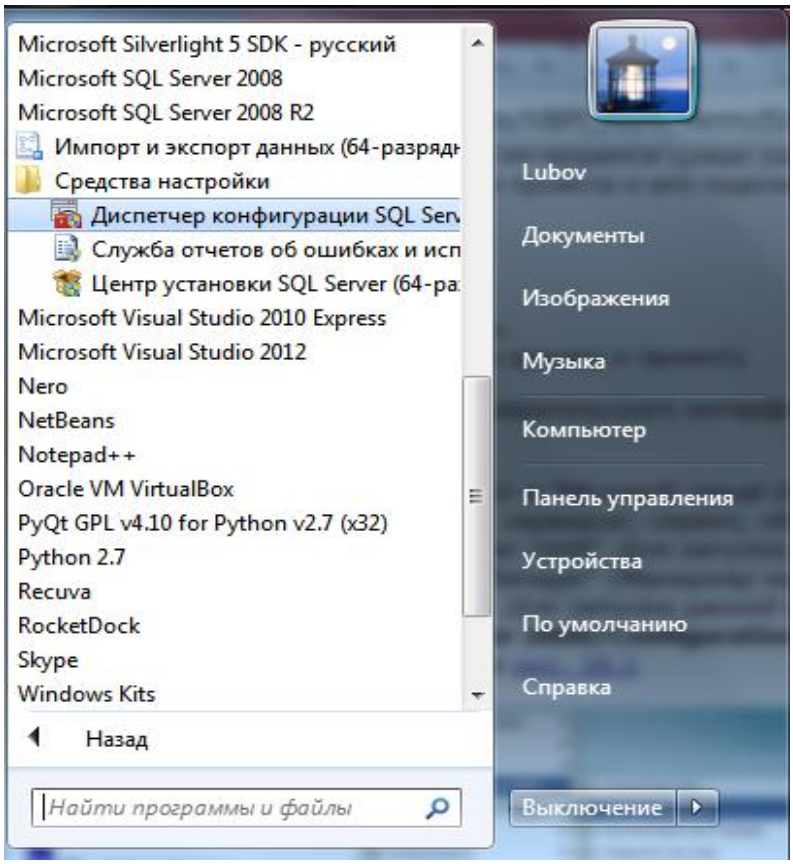
Business Intelligence (система поддержки принятия решений) - относительно новый компонент SQL Server, позволяющий консолидировать данные, полученные из разных источников и рассмотреть из под нужным углом. Состоит из следующих модулей:

- SQL Server Integration Services (SSIS) представляет собой гибкую расширяемую платформу интеграции данных. Сюда относятся средства, позволяющие организовать импорт и экспорт данных;
- Reporting Services (SSRS) – данный механизм позволяет рядовому пользователю получать необходимую ему информацию в нужном и понятном ему виде (отчеты, таблицы, диаграммы, и пр.);

- Analysis Services (SSAS) предназначен для того, чтобы сократить разрыв между потребностями в данных со стороны бизнеса и наличием ресурсов у IT подразделения компании. Для решения этой задачи используются два компонента SSAS – OLAP и Data Mining. Подробнее об этих механизмах мы поговорим в лекциях.

Порядок выполнения работы

1. Запустить «SQL Server Browser» (Обозреватель SQL серверов) «Пуск» пункт «Программы/ Microsoft SQL Server 2008 R2/ Средства настройки/ Диспетчер конфигурации Microsoft SQL Server»



2. В появившемся окне выбираем «Службы SQL Server» и в правой части выделите «Браузер SQL Server». Состояние должно быть «Работает». Для этого нужно на панели инструментов выбрать кнопку запуска.

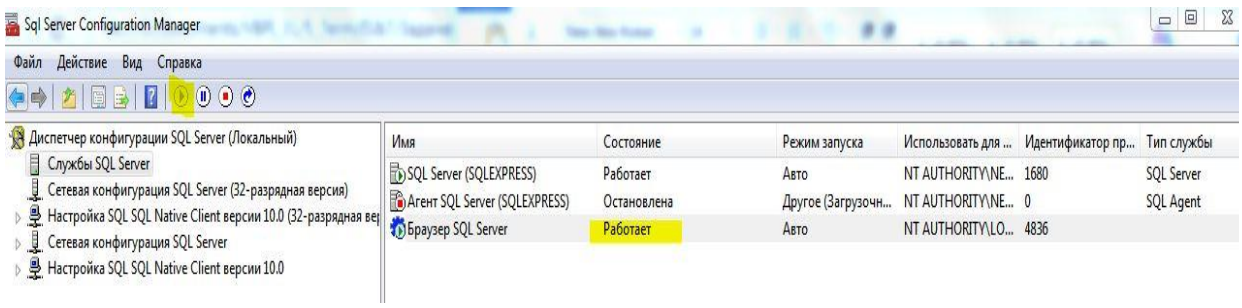


Рис. 2

3. Перейдите к созданию самого пользовательского интерфейса БД «Товары». Для этого создайте новый проект, запустив «Пуск /Все программы/ Microsoft Visual Studio 2012/ Visual Studio 2012»

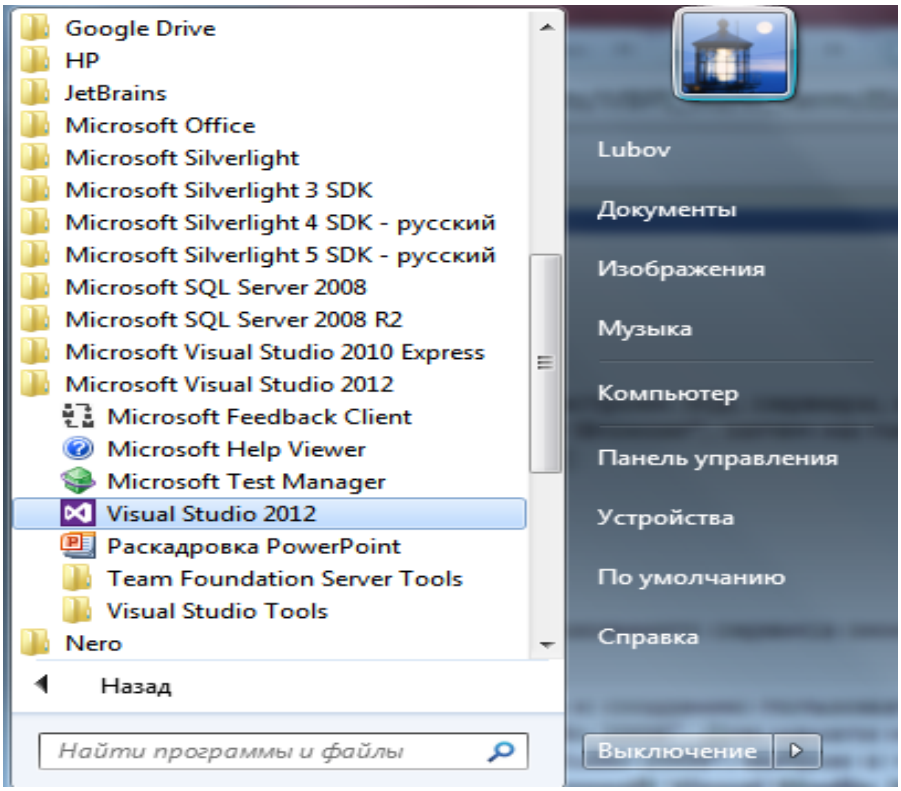


Рис. 3

4. Выполните команду «Файл» - «Создать» - «Проект»

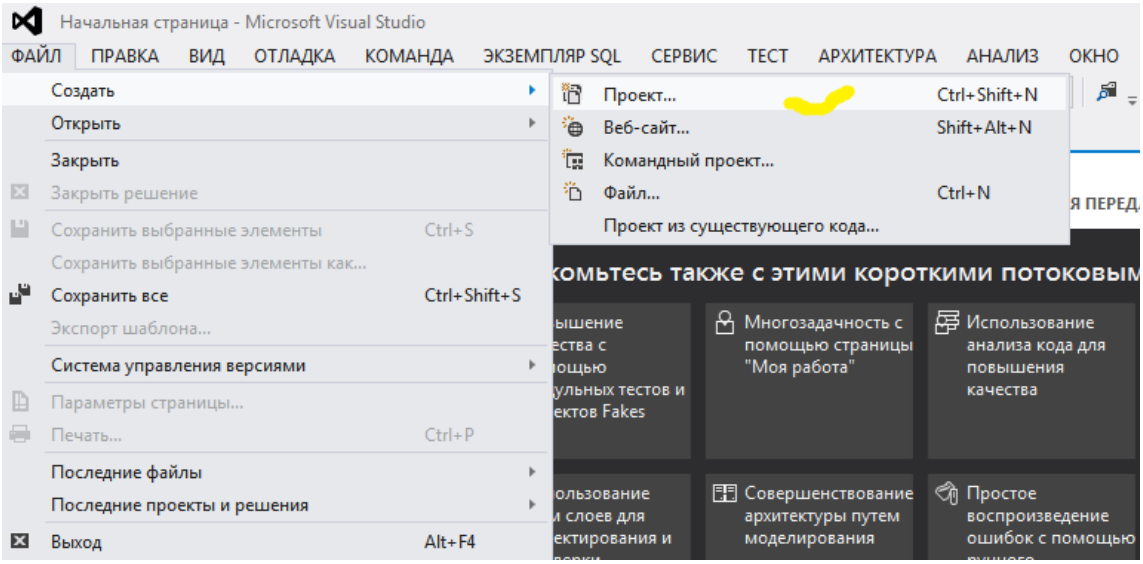


Рис. 4

5. В появившемся окне выберите язык C# и тип проекта Windows Form Application. Задайте название проекту «Goods» и укажите путь.

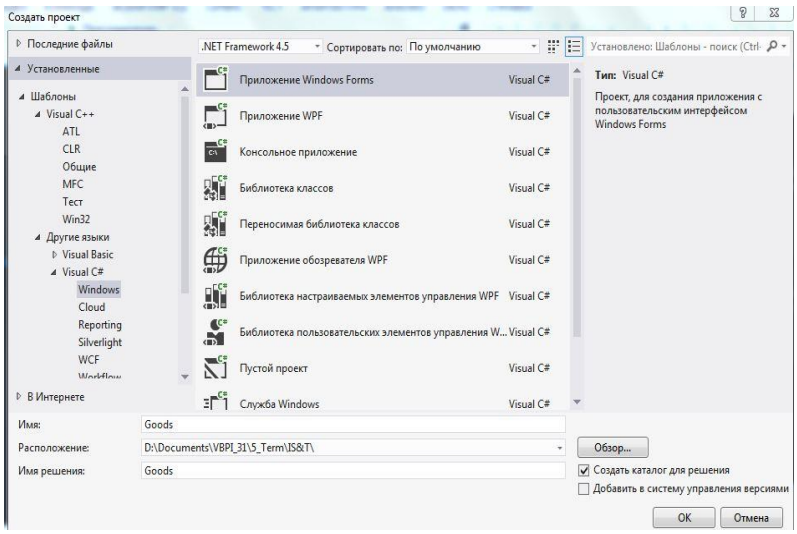


Рис. 5

Перед вами появится окно следующего вида:

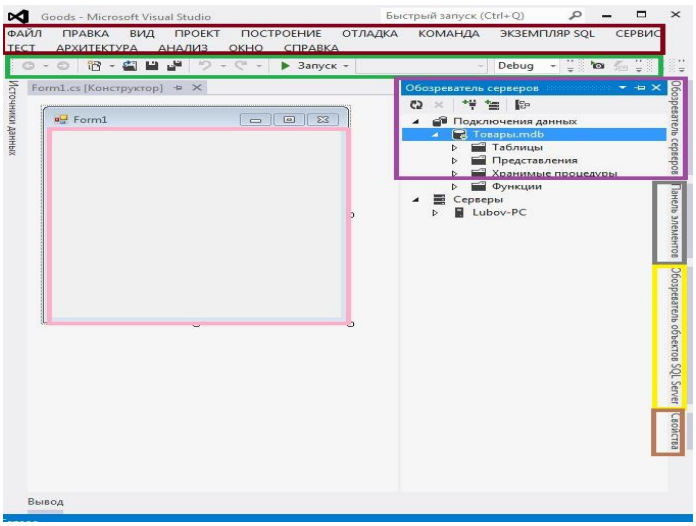


Рис. 6

Структура окна разработки приложения языка Microsoft C#»:

Оконное меню (бордовый цвет) - содержит полный набор команд для управления средой разработки;

Панель инструментов (зеленый) - содержит кнопки с наиболее часто используемыми командами среды разработки;

Панель элементов (серый) - содержит кнопки классов для создания различных объектов (Элементов управления);

Обозреватель проекта/Источники данных (фиолетовый) - в зависимости от активизированной в нижней части данной панели вкладки, отображает обозреватель проекта или источники данных, подключенные к проекту. Обозреватель проекта (Solution Explorer) отображает все файлы, входящие в проект и позволяет переключаться между ними. Источники данных это базы данных, службы или объекты из которых поступают данные в проект;

Панель свойств (коричневый) - отображает и позволяет изменять свойства выбранного объекта;

Рабочая область (розовый) - в зависимости от выбранной вкладки, расположенной в верхней части области, отображает область дизайна формы, код формы или стартовую страницу.

6. После успешного создания проекта Goods необходимо подключить к нему БД, созданную ранее. Для подключения файла БД в формате mdb (Access), требуется выполнить следующие команды.

Выберете вкладку «Источники данных» - «Подключиться к БД»

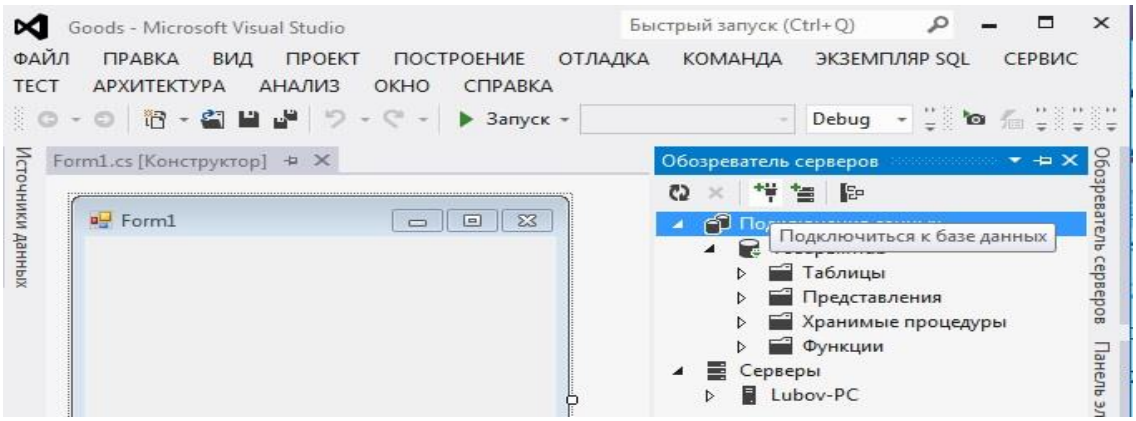


Рис. 7

7. Пропишите путь к БД

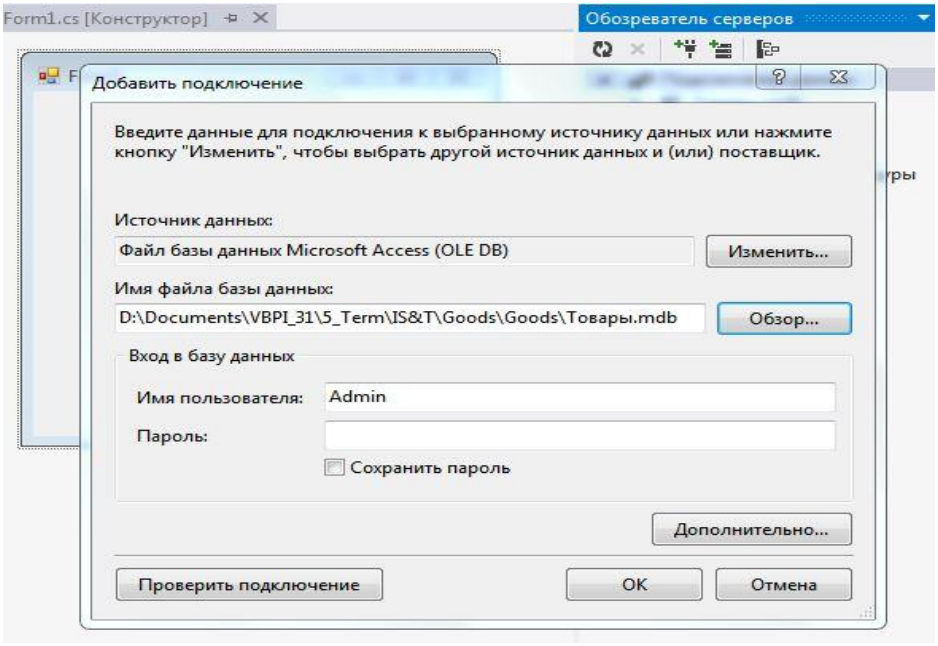


Рис. 8

…и БД отобразится в списке подключенных данных.

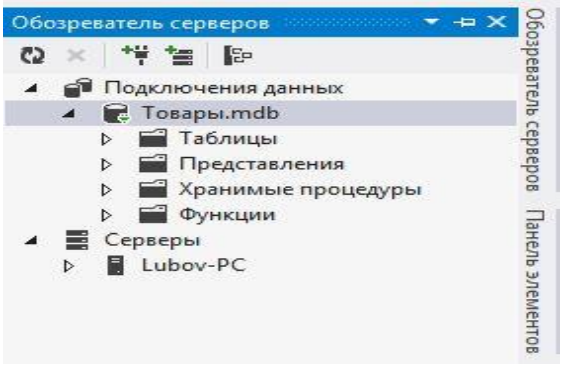


Рис. 9

Для просмотра таблицы БД выберите «Источники данных» - «Таблицы». Сохраните все изменения в проекте, нажав «Сохранить все».

8. Для подключения к БД, хранящейся на сервере, выполните следующие операции. Отключите текущую БД - Товары.mdb. Для этого выберете справа вкладку «Источники данных SQL» - «Подключения данных» - «Товары.mdb» - правая кнопка мыши – «Удалить…».

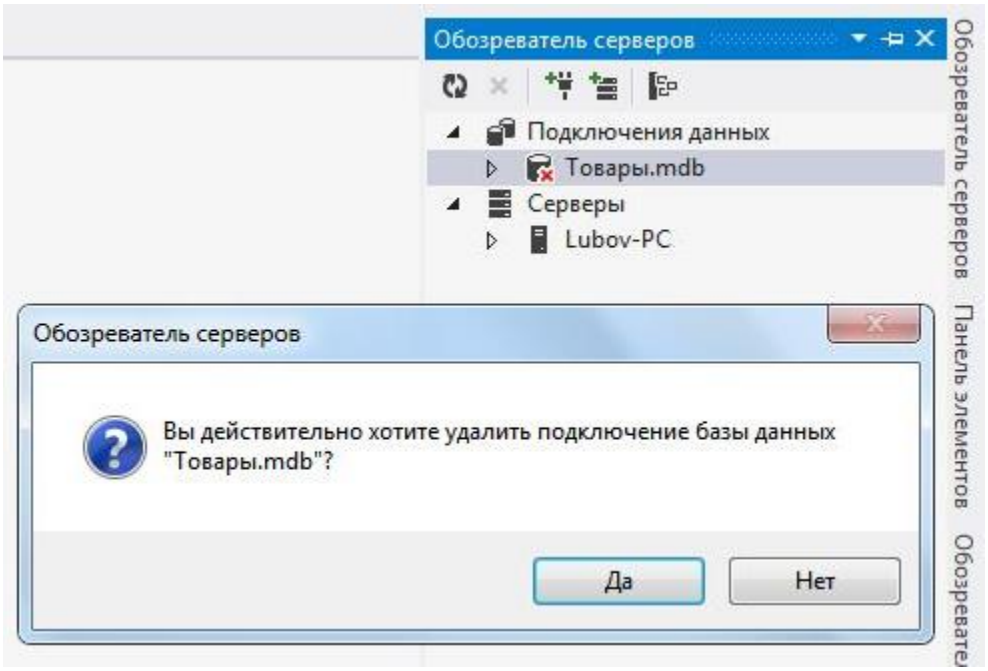


Рис. 10

9. Выберете в оконном меню «Сервис» - «Подключиться к базе данных…» - «Источник данных…» - «Изменить» и затем Microsoft SQL Server (SqlClient) – «Ок».

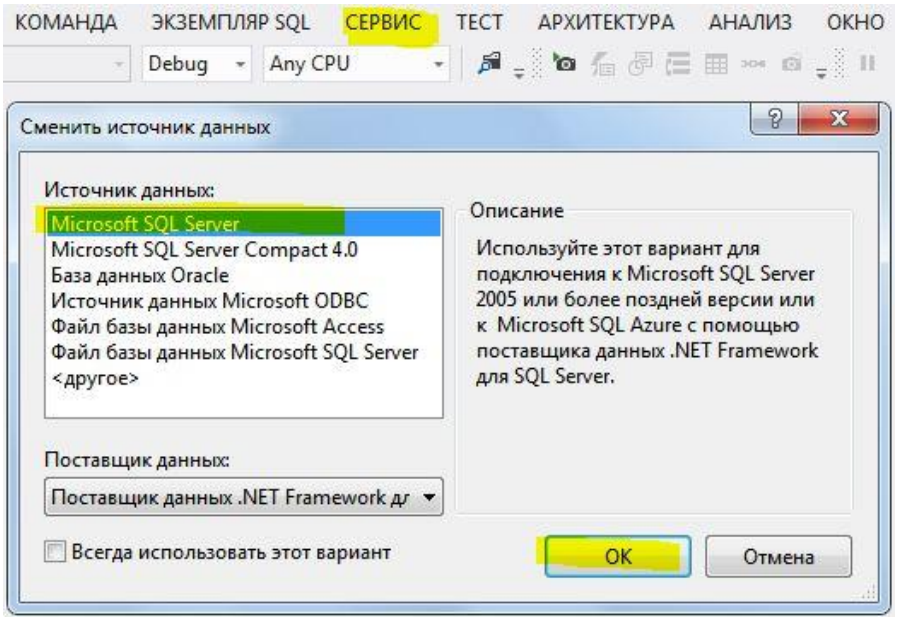


Рис. 11

10. Далее требуется выбрать имя сервера из выпадающего списка. Подключение к БД – выберите или введите имя БД – выбрать имя вашей БД из выпадающего списка. В итоге форма «Добавить подключение» примет вид:

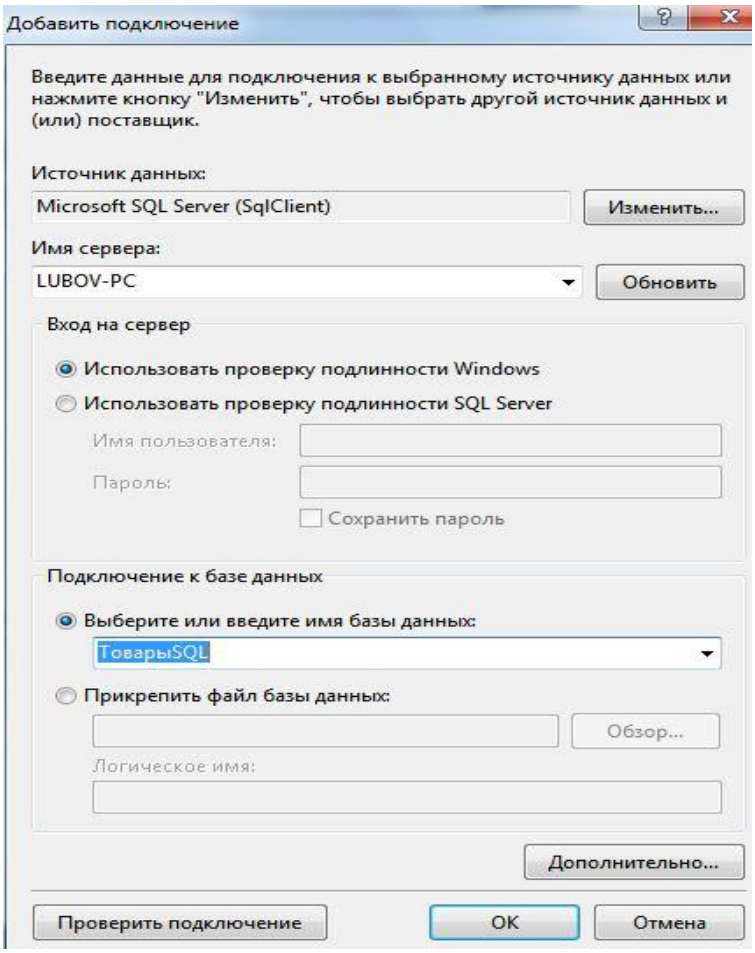


Рис. 12

11. В итоге в обозревателе серверов появилась БД.

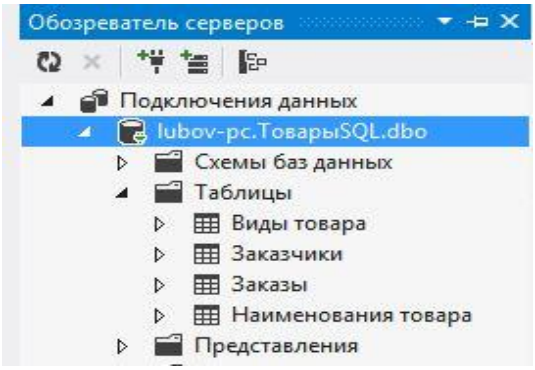


Рис. 13

12. Теперь сохраните сделанные изменения, нажав ctrl+shift+S или выбрав: «Файл» - «Сохранить все».

13. Для дальнейшей работы нам необходимо удалить все пробелы из названий таблиц и названий полей таблиц. Для этого нажимаем на клавишу "Изменить набор данных в конструкторе". Открывается новая панель. На этой панели мы изменяем каждую строку содержащую пробел. Не забываем сохранить.

Во всех последующих действиях названия таблиц и их полей вводить только слитно(даже если на скриншоте содержится "_". Например:"Наименование_заказа" писать как "Наименованиезаказа"

Вопросы

- На какие крупные модули делится SQL Server
- Database Engine – понятие, включаемые модули
- Что представляет собой система репликации
- Business Intelligence – понятие, включаемые модули
- Что представляет SQL Server Integration Services (SSIS)
- Дайте определение SSRS
- Где запускается браузер SQL Server
- Что представляет собой панель элементов
- Для чего используются оконное меню, панель инструментов, панель элементов
- Обозреватель проекта
- Как подключиться к БД, хранящейся на сервере
- Как подключиться к БД, с расширением .mdb

ПРАКТИЧЕСКАЯ РАБОТА №2
«СОЗДАНИЕ ИНТЕРФЕЙСА ПОЛЬЗОВАТЕЛЯ: ГЛАВНАЯ КНОПОЧНАЯ ФОРМА»

Цель:

- Рассмотреть интерфейс информационных систем;
- изучить порядок создания интерфейса пользователя;
- создать главную форму.

Общая информация

В системах, построенных по технологии клиент-сервер, существует **два вида** интерфейса:

- Интерфейс, реализуемый при помощи клиентского приложения;
- Web -интерфейс.

Интерфейс, реализуемый при помощи клиентского приложения - это компьютерная программа, устанавливаемая на клиентские компьютеры, предназначенная для работы с файлами данных через сеть. Основными элементами клиентских приложений являются формы (окно программы) и отчёты. Элементы управления на форме называется **объектами.** Каждый объект обладает своим набором свойств, событий и методов.

Свойства объекта - это его характеристики (высота, ширина и т.д.);
События объекта - это события операционных систем или события инициируемые пользователем, на которые может реагировать объект (нажатие кнопки);
Методы объекта - действия, которые можно производить с объектом в ходе выполнения программ.

В БД все объекты форм делятся на **два класса**:

- Объекты управления - объекты, осуществляющие управление БД (Например: Кнопка или Выпадающий список);

- Объекты для отображения информации - элементы, отображающие содержимое таблиц, запросов или фильтров, позволяющие добавлять изменять и удалять информацию, и проводить ее анализ.

Все формы в клиентском приложении делятся на три группы:

- **Формы для работы с данными** - формы, содержащие как объекты управления, так и объекты просмотра данных. Такие формы предназначены для отображения, изменения, удаления и анализа данных;
- **Кнопочные формы** - формы, содержащие только объекты управления, предназначаются для открытия всех других форм.

Замечание: Кнопочная форма, которая появляется первой после запуска программы, называется, главной кнопочной формой.

- Информационные и служебные формы - формы, содержащие только элементы управления, предназначены для отображения служебной информации (справки), несвязанной с таблицами, запросами и фильтрами, либо для выполнения служебных операций не связанных с данными (Например: форма с калькулятором).

Существует два вида дизайна форм:

- Ленточные формы - формы, выводящие информацию по одной записи;
- Табличные формы - формы выводящие информацию в виде таблицы.

Объекты связи используются только в клиентском интерфейсе. В web-интерфейса функции объекта связи выполняет сервер.

Основой web-интерфейса являются страницы (файл с расширением htm или html). Работа со страницами осуществляется с помощью браузера. Изначально страницы находятся на сервере, пользователь сначала загружает их на свой компьютер с сервера, а затем с помощью страниц пользователь работает с файлом данных.В web-интерфейсе отсутствуют отчёты, их роль выполняют сами страницы.

Порядок выполнения работы

Запустите «Microsoft Visual Studio 2012» и откройте созданный ранее проект «Goods», щелкнув по его значку в области «Последние проекты» стартовой страницы «Стартовая страница».

После появления стандартного окна среды разработки откройте форму «Form1.cs».

В рабочей области поместите на форму надпись (Label) и четыре кнопки (Button) как показано на рисунке 14.

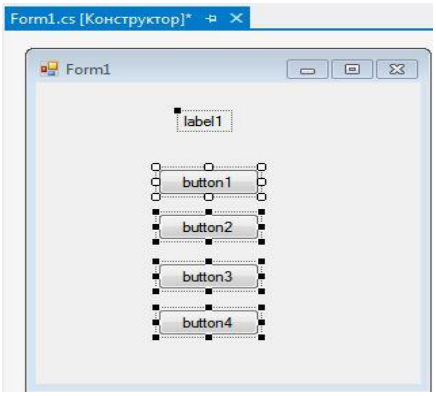


Рис.14

1. Для этого выберете на панели элементов Label,нажмите на него левой кнопкой мыши (ЛКМ) и, удерживая, перетащите на форму в требуемое место.
2. Аналогично поступите с кнопками.
3. После создания объектов перейдите к настройке их свойств. Начните с настройки свойств формы. Выделите форму, щелкнув ЛКМ в пустом месте формы.
4. На панели элементов задайте свойства формы как представлено ниже:
FormBorderStyle (Стиль границы формы): Fixed3D;
MaximizeBox (Кнопка развертывания формы во весь экран): False;
MinimizeBox (Кнопка свертывания формы на панель задач): False;
Text (Текст надписи в заголовке формы): База данных «Товары».
5. Теперь выделите на самой форме надпись, щелкнув ЛКМ, выберете на панели элементов следующие свойства для надписи:
AutoSize (Авторазмер): False;
Font (Шрифт): Microsoft Sans Serif, размер 14;
ForeColor (Цвет текста): HotTrack;
Text (Текст надписи): База данных «Товары»;
TextAlign (Выравнивание текста): MiddleCenter.
6. Для кнопок задайте свойство надписи (Text) как показано ниже.

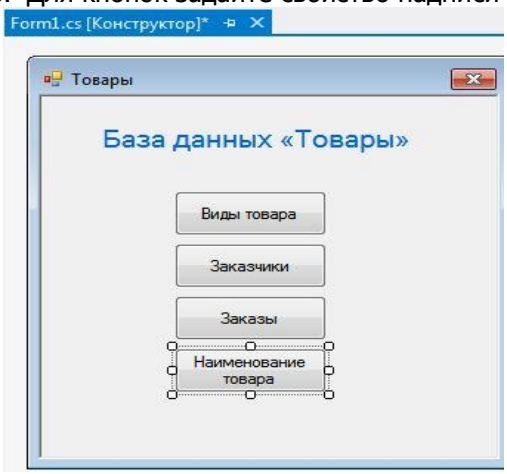


Рисунок 15

7. Сохраните выполненные изменения.
8. Если названия не помещаются на кнопке, то можете ее увеличить: щелкните ЛКМ, и потяните за необходимый квадрат (как на кнопке «Наименование товара», рис.15), расположенный по периметру кнопки.

Вопросы

- Сколько видов интерфейсов существует в системах, построенных по технологии клиент-сервер
- На какие классы делятся объекты форм
- Кнопочная форма – определение, особенности
- Какие виды дизайна форм вы знаете
- Что является основой web-интерфейса
- После подключения источника данных что отображается в окне «Источники данных»
- Как увеличить кнопку
- Как задать выравнивание текста по правому краю
- Как поменять свойства формы

ПРАКТИЧЕСКАЯ РАБОТА №3

«СОЗДАНИЕ ИНТЕРФЕЙСА ПОЛЬЗОВАТЕЛЯ ПРИ ПОМОЩИ ОКНА «ИСТОЧНИКИ ДАННЫХ»

Цель

- Научиться создавать интерфейс пользователя (ленточные формы для работы с данными);
- освоить подключение объектов к источнику данных при помощи окна свойств.

Общая информация

Visual C# 2012 позволяет создавать простой интерфейс БД, без помощи панели объектов и окна свойств, лишь используя окно «Источники данных». В окне «Источники данных» после подключения источника данных отображается все таблицы, запросы, фильтры данных и их поля. В Visual C# 2012 как и в ранних версиях можно перетаскивать источники данных, соответственно таблицы, запросы, фильтры прямо из окна «Источники данных» на форму. Главное отличие - при перетаскивании можно выбрать для каждого поля источника данных свой объект, который будет отображать его содержимое. Таким способом можно создавать только определённые объекты для отображения данных поля, и набор этих объектов зависит от типа данных поля.

Создание объектов для отображения данных перетаскиванием состоит из двух шагов:

- Для каждого поля таблицы, запроса, или фильтра выбирается объект, который будет отображать его содержимое. Для этого необходимо щелкнуть мышью по полю в окне «Источники данных». Рядом с именем поля появится кнопка со стрелкой. Щелкнув мышью по стрелке, вы увидите выпадающее меню с объектами, которые могут отображать информацию, содержащуюся в поле. Для полей стандартными объектами являются: TextBox, ComboBox, Label, LinkLabel, ListBox. Для полей типа данных DateTime возможно использования объекта DataTimePicker. Для полей логических типов данных возможно использование объекта CheckBox. В выпадающем меню с вариантами выбора объектов имеется пункт «Customize» (Настройки), который позволяет выбрать дополнительные допустимые объекты для отображения информации. Для отображения таблиц, запросов или фильтров целиком возможно два варианта отображения:
    - При помощи объекта DataGridView - информация из таблицы, запроса или фильтра отображается в виде таблицы;
    - DetalledView - отображение всех полей источника данных в TextBox по отдельности.
- после выбора объектов для отображения, необходимо поместить их на форму, перетаскивая мышью с панели «Источники данных» в необходимое место на форме. После перетаскивания с созданным объектом можно работать как и с обычным объектом Visual C#.

При помещении первого объекта на форму на ней автоматически создаются объекты для связей с файлом данных и объекты навигации по источникам данных (DataSet, BindingSource, TableAdapter, BindingNavigator). По умолчанию панель навигации располагается в верхней части формы. Эту панель можно прикрепить около различных краев формы. Для этого необходимо воспользоваться меню действий объектов. Это меню схоже с окном «Property Pages», но кроме основных свойств объектов оно содержит и действия, которые можно производить с объектами. Чтобы вызвать это меню, необходимо выделить объект. В его правом верхнем углу появится кнопка (квадратик со стрелочкой), при нажатии этой кнопки появляется выпадающее меню с настройками и действия с объектом. Например, чтобы поменять местоположение навигации панели - надо в этом меню выбрать настройку «Docking». При перетаскивании на форму полей источников данных автоматически создаются подписи к ним (Label).

Подключение объектов к источнику данных при помощи окна свойств

Visual C# позволяет подключать источники данных к объектам без использования перетаскивания, то есть вручную, с использованием панели свойств. Для этого на форму помещается объект, который

будет подключаться к источнику данных. Его выделяют, затем на панели свойств разворачивается группа свойств «DataBindings». Она содержит два свойства:

- **Text** - определяет таблицу, запрос или фильтр, из которого выводятся данные в объект;
- **Tag** - определяет поле, выбранного в свойстве Text источника данных, которое отображается в объекте.

На этом завершим рассмотрение простых ленточных форм для работы с данными.

Порядок выполнения работы

1. Создайте простую ленточную форму. Для этого откройте ранее созданный проект «Goods.sln».
2. Создайте ленточную форму, отображающую таблицу «Виды товара». Добавьте в проект новую пустую форму. Для этого в оконном меню выберите пункт «Проект/Добавить новую форму Windows».
3. В появившемся окне «Добавление нового элемента» В данном окне в разделе «Установленные» раскройте «Элементы Visual C#», затем выберете «Windows Forms» и после нажмите «Форма Windows Forms». Оставьте имя по умолчанию «Form2.cs» и нажмите «Добавить» (см.рис.16).

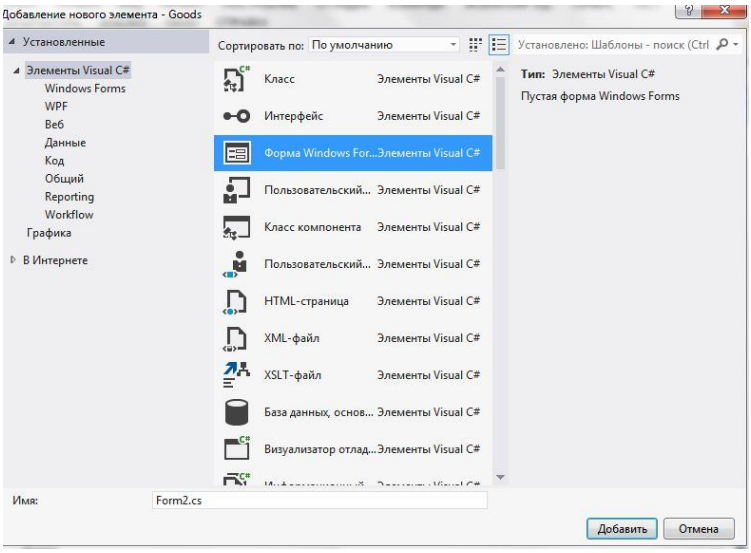


Рисунок 16

4. В верхней части новой формы создайте надпись (**Label**).
5. Настройте свойства формы и надписи. Выделите Выделите форму, щелкнув ЛКМ в пустом месте формы. На панели свойств задайте свойства формы следующим образом:
FormBorderStyle (Стиль границы формы): Fixed3D;
MaximizeBox (Развертывание формы во весь экран): False;
MinimizeBox (Свертывание формы на панель задач): False;
Text (Текст в заголовке формы): Таблица «Виды товара».

6. На форме выделите надпись, щелкнув по ней ЛКМ и на панели свойств, задайте свойства надписи как показано ниже:
AutoSize (Авторазмер): False;
Font (Шрифт): Microsoft Sans Serif, размер 14;
ForeColor (Цвет текста): HotTrack;
Text (Текст надписи): Таблица «Виды товара»;
TextAlign (Выравнивание текста): MiddleCenter.

7. После настройки всех вышеперечисленных свойств форма будет выглядеть как на рисунке 17.

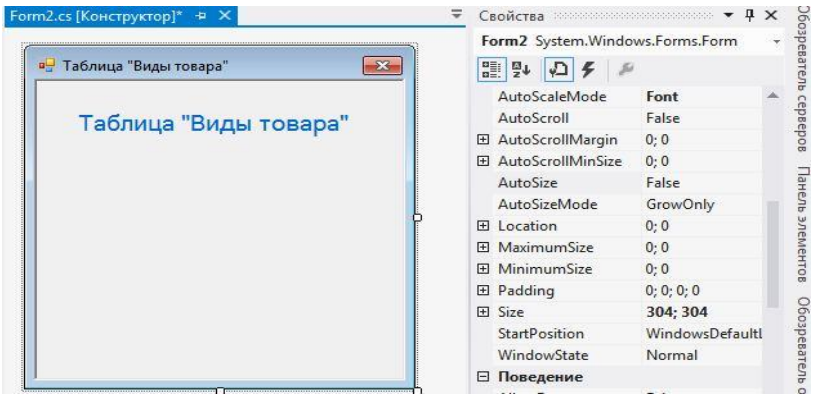


Рисунок 17

8. Теперь поместите на форму поля таблицы «Виды товара». Сначала откройте панель «Источники данных», щелкнув по ее вкладке в правой части окна среды разработки.
9. На панели «Источники данных» выберете «Подключения данных», затем требуемое подключение и отобразите таблицу «Виды товара» вместе с полями. Результат представлен на рисунке 18.

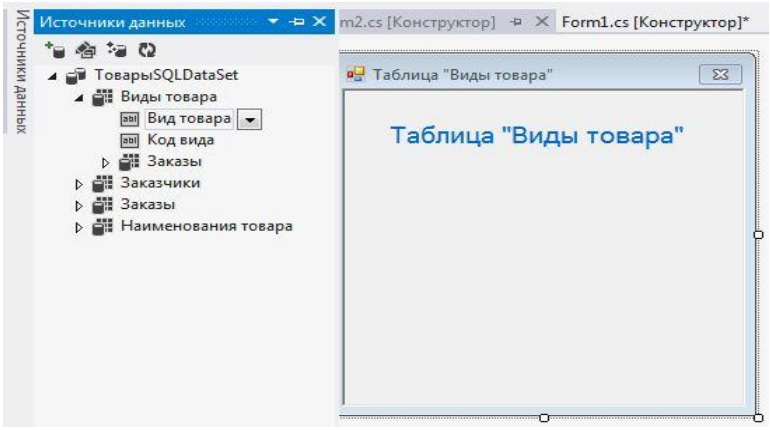


Рисунок 18

10. **Примечание:** под полями таблицы, T1 на том же уровне иерархии, может располагаться еще одна таблица T2, называемая подтаблицей. Подтаблица показывает, что она является вторичной по отношению к таблице T1.
11. **Примечание:** при выделении какого либо поля таблицы, оно будет отображаться в виде выпадающего списка, позволяющего выбирать объект, отображающий содержимое выделенного поля.
12. Поместите на созданную ранее форму поля таблицы. Для этого их необходимо перетащить из панели «Источники данных» на форму. Из таблицы «Виды товара» перетащите мышью на форму поля: «Вид товара» и «Код вида». Форма примет вид, представленный на рисунке 19.

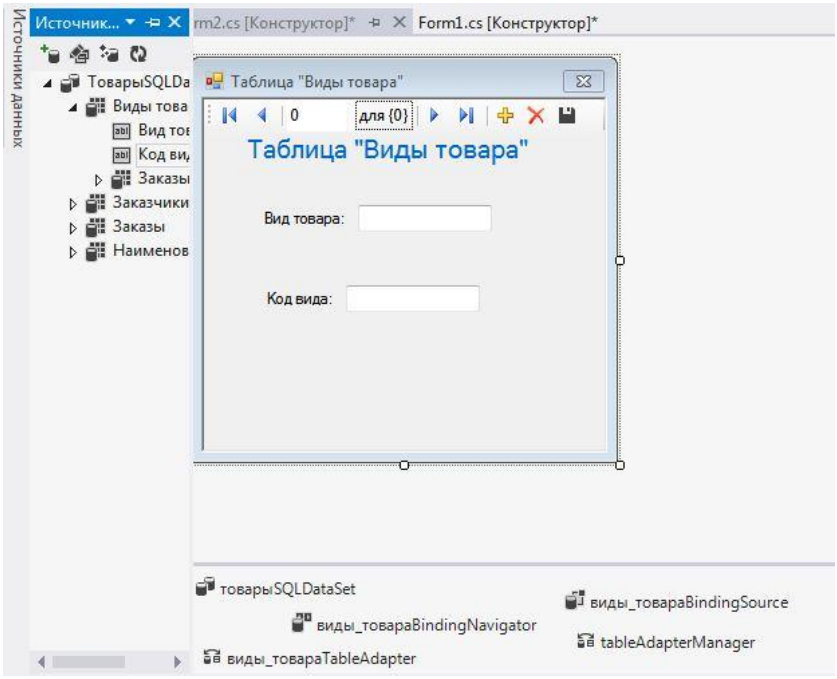


Рисунок 19

13. **Примечание:** поле «Код вида» обычно не помещается на форму, т.к. оно является первичным полем связи и заполняется автоматически. Конечный пользователь не должен видеть такие поля.

14. После перетаскивания полей с панели «Источники данных» на форму в верхней части формы появилась навигационная панель, а в нижней части рабочей области среды разработки появились пять невидимых объектов. Эти объекты предназначены для связи нашей формы с таблицей «Виды товара», расположенной на сервере. Функции этих объектов:

- товарыSQLDataSet (Набор данных Student) - обеспечивает подключение формы к конкретной БД на сервере (в нашем случае это БД Students);
- видытовараBindingSource (Источник связи для таблицы"Специальности") - обеспечивает подключение к конкретной таблице (в нашем случае к таблице специальности), а также позволяет управлять таблицей;
- ВидытовараTableAdapter (Адаптер таблиц для таблицы"Специальности") - обеспечивает передачу данных с формы в таблицу и наоборот.
- ВидытоваратableAdapterManager (Менеджер адаптера таблиц) - управляет работой объета СпециальностиTableAdapter ;
- ВидытовараBindingNavigator (Панель управления таблицей"Специальности") - голубая панель с кнопками управления таблицей, расположенная в верхней части формы

15. Проверьте работоспособность формы. Подключите ее к главной кнопочной форме, а затем запустите проект и откройте форму «Виды товара» при помощи кнопки на главной кнопочной форме.

16. Для выполнения п.15, отобразите главную кнопочную форму в рабочей области среды разработки, щелкнув по вкладке Form1.cs[Конструктор]. Для подключения новой формы «Виды товара» к главной кнопочной форме дважды щелкните ЛКМ по кнопке «Таблица «Виды товара»», расположенной на главной кнопочной форме. В появившемся окне кода формы в процедуре "Button1_Click" наберите команду, предназначенную для открытия формы «Таблица «Виды товара»» (Form2), как это показано на рисунке 20.

17. Запустите проект, нажав «Запуск» на верхней панели. Выберете «Виды товара» и в появившемся окне осуществите навигацию по представленным товарам.

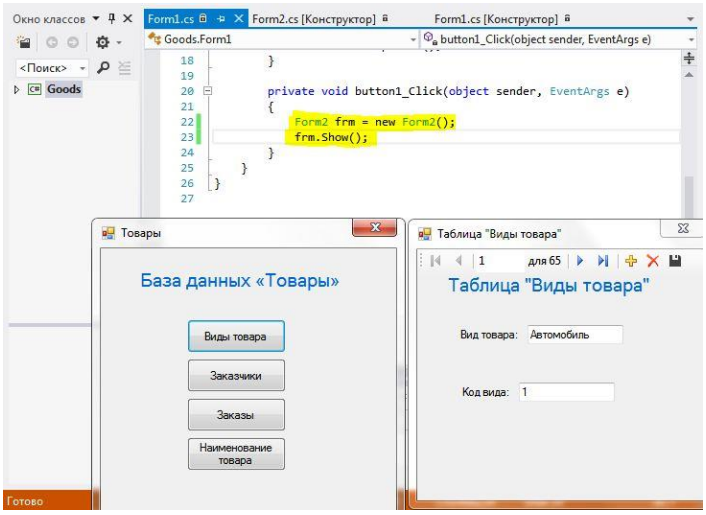


Рисунок 20

18. Для возвращения в среду разработки закройте форму таблицы «Виды товара» и главную кнопочную форму.
19. Аналогичным образом создайте формы для отображения содержимого таблиц «Заказы»(Form3), «Заказчики»(Form4), «Наименование товара» (Form5). Продемонстрируйте результат преподавателю.

Вопросы

- Опишите алгоритм создания объектов для отображения данных путем перетаскивания
- Назовите стандартные объекты для полей
- Как представляет информацию объект DataGridView
- Что происходит при помещении первого объекта на форму
- Перечислите объекты навигации по источникам данных
- Что создается автоматически при перетаскивании на форму полей источников данных
- Расскажите, как можно подключить источники данных к объектам вручную
- За что отвечает SQLDataSet

ПРАКТИЧЕСКАЯ РАБОТА №4

«СТАНДАРТНЫЕ ОБЪЕКТЫ ДЛЯ ОТОБРАЖЕНИЯ ДАННЫХ. ПРОГРАММНОЕ УПРАВЛЕНИЕ ИНФОРМАЦИОННОЙ СИСТЕМОЙ»

Цель

- Изучить стандартные объекты для отображения данных;
- рассмотреть программное управление информационной системой;
- научиться создавать сложные ленточные формы для работы с данными

Общая информация

Способ создания объектов для отображения данных описанный ранее, позволяет создавать только ограниченный набор объектов. Однако, Visual C# 2012 позволяет подключать источники данных практически к любому объекту, который может быть создан на форме. Это можно сделать при помощи перетаскивания поля источника данных из окна "Data Sources" на объект на форме.

Операция состоит из двух шагов:

При помощи панели объектов (слева) на форме создается какой-то объект. Объекты несвязанные с источником данных называют несвязанными объектами;
Вновь созданный объект связывается с источником данных. Для этого на объект нужно перетащить поле таблицы запроса или фильтра из окна "Data Sources".
Замечание: При перетаскивании поля из окна "Data Sources" необходимо учитывать его тип данных. Объект на форме должен поддерживать тип данных подключаемого к нему поля.

Замечание: В случае подключения объекта к источнику данных, способом, описанным выше, подпись к объекту не создаётся автоматически и её надо создавать вручную с помощью объекта Label.

Наиболее часто в БД используются следующие **объекты для отображения информации**:

Текстовое поле (TextBox)
Надпись (Label)
Надпись со ссылкой (LinkLabel)
Календарь (DataPicker)
Переключатель (CheckBox)
Таблица (DataGridView)
Список (ListBox)
Выпадающий список (ComboBox)
Текстовое поле с маской ввода (MaskedTextBox)
TextBox - отображает текст и числовые поля, это наиболее часто употребляемый объект для отображения данных. Его можно создавать либо перетаскиванием из окна "Data Sources", либо подключить вручную. Создание этого объекта, перетаскиванием возможно почти у полей любых типов данных.

Label - полностью аналогичен объекту TextBox, но не позволяет изменить данные. Этот объект используется для отображения заблокированных неизменяемых полей.

LinkLabel - специальный объект для отображения ссылок на адреса в Интернете. Его используют для отображения текстовых полей, если в них хранятся адреса Интернета или какой-то компьютерной сети.

DataTimePicker - специальный объект, предназначенный для отображения полей типа данных "Дата/Время" в виде календаря.

CheckBox - объект используется для отображения логических полей, может быть создан перетаскиванием только для логических полей.

DataGridView - объект, отображающий источник данных (таблицу, запрос или фильтр) в виде таблицы.

ListBox - список отображающий значения полей и позволяющий выбирать значения полей из списка. Более того, пункты списка можно задавать, используя другой источник данных.

ComboBox - объект подобный объекту ListBox, однако информация отображается не в списке, а выпадающем списке.

MaskedTextBox - нестандартный объект, предназначенный для отображения и ввода информации по заранее заданному шаблону (маске). Этот объект может быть создан только при помощи панели объектов и его подключение осуществляется либо перетаскиванием на него поля из окна "Data Sources", либо заданием его свойств вручную. По своим свойствам он ничем не отличается от объекта TextBox. Единственное дополнительное свойство у этого объекта это свойство Mask. Для этого нужно щелкнуть по кнопке действий объекта в верхнем правом углу объекта. Затем в списке действий выбрать пункт "Edit Mask". В появившемся окне выбрать шаблон ввода, то есть маску (Mask).

Замечание: Тип данных отображаемой информации должен совпадать с типом данных маски.

Замечание: Объекты ListBox и ComboBox могут использоваться для заполнения полей с кодами, то есть списки заполняются информацией из одной таблицы, а при выборе пункта списка его код подставляется в другую таблицу. Для этого на форму располагают не подключенный ListBox или ComboBox, затем открываем его меню действий. В меню действий в указанной последовательности выполняют следующие шаги:

Установить галочку "Use Data Bound Items",
В выпадающем списке "Data Source" выбрать пункт "Other Data Source" и там выбрать нужную таблицу.
В выпадающем списке "Display Member" и указываем поле, которое отображается в списке.
В выпадающем списке "Value Member" указываем поле, которое подставляем при выборе пункта списка.
В выпадающем списке "Selected Value" указываем поле, куда подставляется выбранное в "Value Member" значение.
Программное управление информационной системой

В Visual Studio 2012 добавлять, удалять записи и перемещаться по ним можно как используя объект Navigator, так и используя обычные кнопки. Рассмотрим создание кнопок для управления записями. В Visual Studio 2012 отсутствует объект "RecordSet" все операции с записями осуществляются с использованием объекта "BindingSource".

В Visual C# 2012 для добавления новой записи из таблицы "Товары" используется команда вида ТоварыBindingSource.AddNew. Вместо метода AddNew можно использовать методы:

MoveNext (перейти к следующей);
MoveFirst (Перейти к первой);
Move Previous (Перейти к предыдущей);
Move Last (Перейти к последней);
Delete (Удалить запись).
У объекта BindingSource имеется свойство Filter. В свойстве Filter задаётся строка, определяющая условие отбора записей в динамических фильтрах, выполняемых на стороне клиента. Данная строка имеет следующий синтаксис:

<Поле1><Оператор1><Выражение1>
[AND|OR <Поле2><Оператор2><Выражение2>…]
Здесь:

<Поле1>, <Поле2>... - поля на которые накладываются условия;
<Оператор1>, <Опрераторы2> - операторы сравнения, участвующие в условиях;

<Выражение1>, <Выражение2> - выражения с которыми сравниваются поля. Под выражениями понимаются, константы, переменные, формулы, функции и свойства объектов
Пример: Из таблицы "Товары" необходимо отобразить товар, у которого значение поля КодВидаТовара равно "12".

ТоварыBindingSource.Filter = " КодВидаТовара = '12'"
Обычно при формировании запроса при помощи свойства Filter задания условий отбора используют либо списки ListBox, либо выпадающие списки ComboBox.

Замечание: Если мы используем ComboBox для создания динамического фильтра, то в меню действий параметры "Value Member" и "Selected Value" настраивать не надо.

Пример: Имеется таблица "Товары", которая отображается на форме в DataGridView. Необходимо на форме поместить ComboBox с фамилиями студентов. При выборе КодВидаТовара и нажатием на кнопку отобразить данные только по выбранному товару.

В этом случае в меню действий ComboBox в параметре "Data Source" указываем "Other Data Source/Товары". Затем в "Display Member" выбираем КодВидаТовара. В коде кнопки прописываем следующую команду:

ТоварыBindingSource.Filter = "КодВидаТовара='" & ComboBox1.Text & "'";

После нажатия кнопки в DataGridView отображаются данные по студенту, выбранному в выпадающем списке ComboBox1.

Порядок выполнения работы

1. Модернизируем форму для ранее созданной таблицы "Заказы". Сначала продублируем кнопки панели навигации, расположенной в верхней части формы. Откройте проект "Goods" и отобразите форму таблицы "Заказы" (Form3). В нижней части формы расположите семь кнопок, как это показано на рис.21.

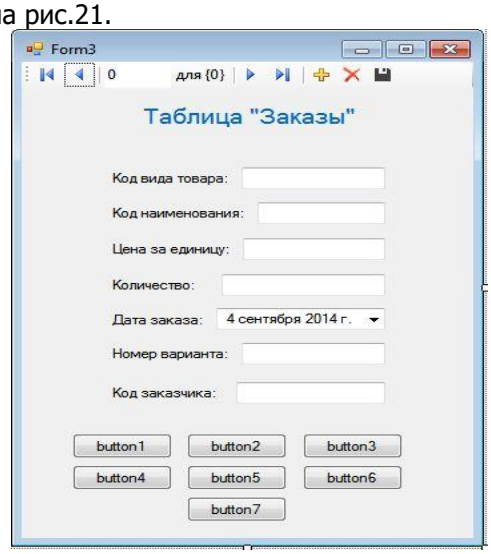


Рисунок 21

2. В качестве надписей на созданных кнопках (Свойство "Text") задайте как: "Первая", "Предыдущая", "Добавить", "Последняя", "Следующая", "Удалить" и "Сохранить".

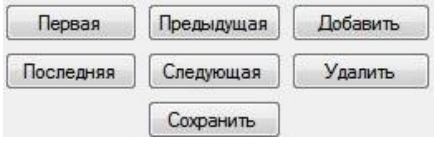


Рисунок 22

3. Дважды щелкните ЛКМ по кнопке "**Первая**" и в появившемся окне кода формы "Form3" в процедуре "Button1_Click" наберите команду для перехода к первой записи "ТоварыBindingSource.MoveFirst()":

```
private void button1_Click(object sender, EventArgs e)
    {
       this.заказыBindingSource.MoveFirst();
    }
```

4. Аналогично, пропишите код в процедурах остальных кнопок(ЛКМ х 2, в появившейся процедуре для требуемой кнопки наберите код):

```
 private void button4_Click(object sender, EventArgs e)
        {
           this.заказыBindingSource.MoveLast();
        } (Последняя);
private void button2_Click(object sender, EventArgs e)
        {
            this.заказыBindingSource.MovePrevious();
        } (Предыдущая);
private void button5_Click(object sender, EventArgs e)
        {
            this.заказыBindingSource.MoveNext();
        } (Следующая);
private void button3_Click(object sender, EventArgs e)
        {
            this.заказыBindingSource.AddNew();
        } (Добавить);
private void button6_Click(object sender, EventArgs e)
        {
            this.заказыBindingSource.RemoveCurrent();
        } (Удалить);
private void button7_Click(object sender, EventArgs e)
        {
             try
{this.Validate();              this.заказыBindingSource.EndEdit();
this.tableTableAdapter.UpdateAll(this.товарыSQLDataSet);
              MessageBox.Show("Update successful");
           }
           catch (System.Exception ex)
           {
               MessageBox.Show("Update failed");
           }
        } (Сохранить).
```

Замечание: Рассмотрим последнюю процедуру более подробно. Она содержит следующие команды:
this.Validate();- проверяет введенные в поля данные на соответствие типам данных полей;
this.заказыBindingSource.EndEdit();- закрывает подключение с сервером;
this.tableAdapterManager.UpdateAll(this.товарыSQLDataSet);- обновляет данные на сервере.

5. Отобразите Form3 через главную форму нашего приложения(Form1), аналогично предыдущей лабораторной работе.
6. Запустите проект и проверьте работоспособность созданной формы и кнопок.
7. Измените объекты, отображающие поля для более удобного ввода информации. Удалите некоторые текстовые поля ввода(TextBox), как представленно на рисунке 23:

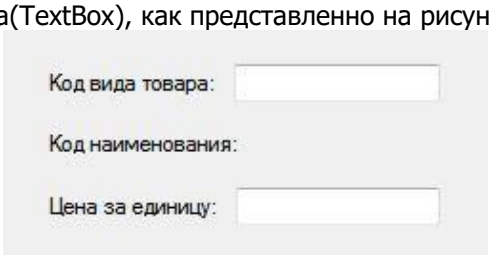


Рисунок 23

8. Для отображения полей "Телефон", "Паспортные данные" и "Номер зачетки" будем использовать текстовые поля ввода по маске (MaskedTextBox).

9. Создайте данные объекты при помощи панели объектов (Toolbox), а затем подключать их к соответствующим полям вручную. Для создания текстовых полей ввода по маске на панели объектов используется MaskedTextBox.
10. Теперь создайте удаленные поля, используя текстовые поля ввода по маске.
11. У созданных объектов настройте маски ввода. Начните с объекта, отображающего «Код вида товара».
12. На форме выделите соответствующее полю «Код вида товара» текстовое поле ввода по маске. Для задания маски в меню действий с объектом выберите пункт «Установка маски…».

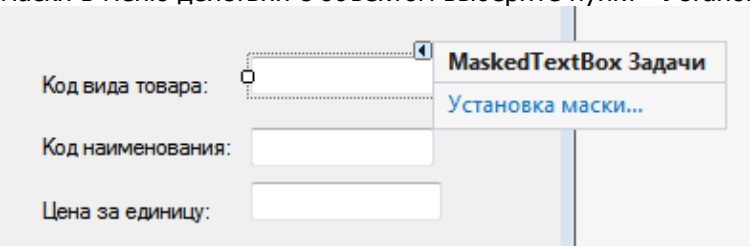


Рисунок 24

13. После выбора пункта «Установка маски…» на экране появится окно задания маски «Маска ввода».

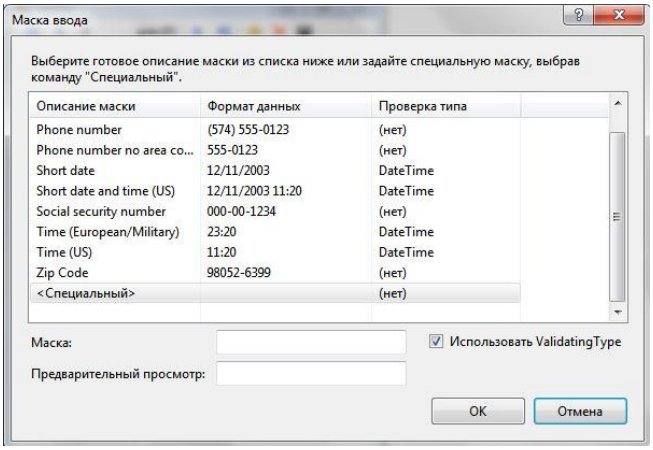


Рисунок 25

14. В окне «Маска ввода» выберите маску "Numeric (5-digits)" (Числовое (5-цифр)) и нажмите "Ok".
15. Для текстового поля ввода по маске для поля «Цена за единицу» задайте маску как показано на рис. 26.
16. Используя такую маску, вы ограничиваете диапазон цен 4-х значным числом (например, 2 345-50 рублей). Данная маска удобна для создания полей с паспортными данными. В поле «Маска» знак "0" обозначает цифру;а в поле «Предварительный просмотр» отображается вид текстового поля ввода по маске на форме.

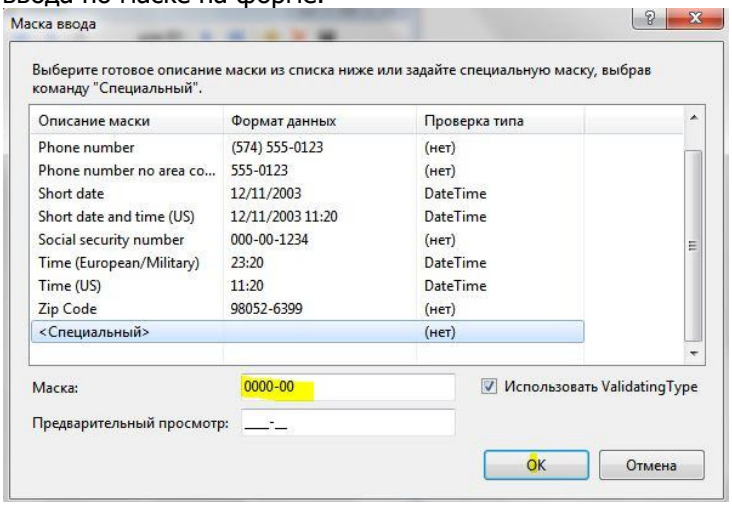


Рисунок 26

17. Нажмите «Ок».

18. Теперь необходимо подключить созданные текстовые поля ввода по маске к соответствующим полям. Для этого с панели "Источники данных" перетащите поле «Код вида товара» на текстовое поле ввода по маске, расположенное справа от надписи «Код вида товара». Проделайте такую же операцию с полем «Цена за единицу», перетащив его на соответствующие ему текстовое поле ввода по маске.
19. Запустите проект и сделайте выводы по полученным результатам.
20. Создайте форму4: «Информация о заказчиках».
21. Сразу добавьте кнопку «Информация о заказчиках» на главную форму и пропишите код, отображающий форму 4.
22. Отобразите все поля данной таблицы.
23. Добавьте кнопки, аналогичные используемым на форме 3.
24. Поля «Паспортные данные» и «Телефон» представьте, используя MaskedTextBox.

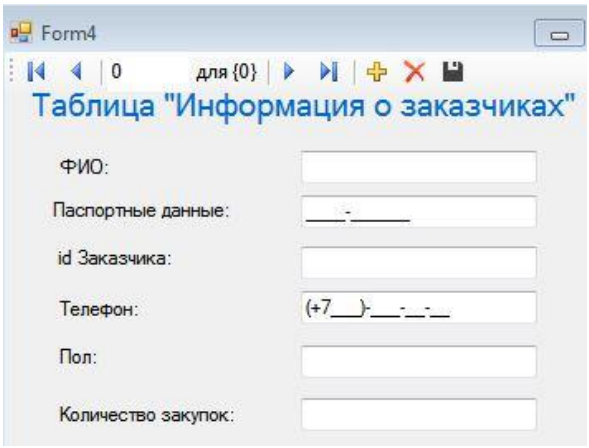


Рисунок 27

25. Отобразите поле "Пол" в виде выпадающего списка (Объект ComboBox). Для этого, на панели "Источники данных" нажмите кнопку, расположенную справа от поля "Пол" и в выпадающем списке выберите объект для отображения данного поля как "ComboBox".
26. Выделите выпадающий список, отображающий поле "Пол". На панели свойств (Properties) и нажмите кнопку в свойства – «Изменить элементы…»

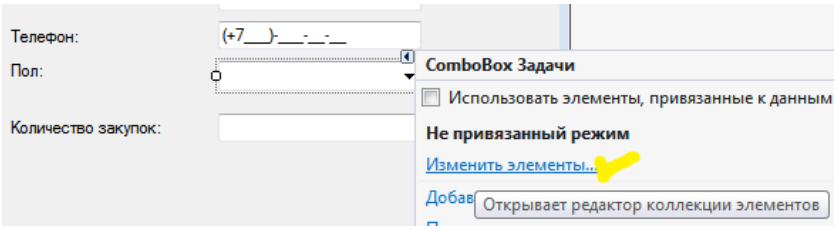


Рисунок 28

27. Введите пол «Мужской», «Женский».

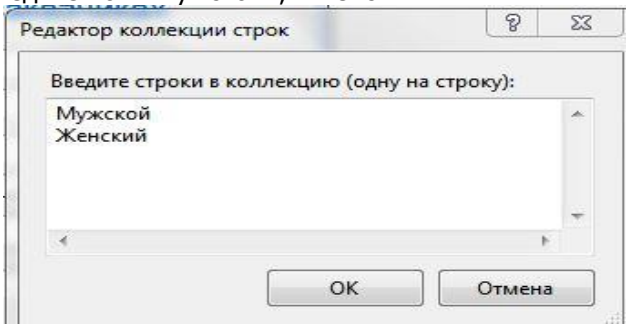


Рисунок 29

28. Проверьте работоспособность созданной формы.
29. Реализуйте вычислимые поля. Для этого отобразите поле «Цена за единицу» (Источники данных – таблица «Заказы»).

30. Добавьте на форму TextBox, Label и кнопку.

Рисунок 30

31. Теперь дважды щелкните ЛКМ по кнопке "Вычислить" и в появившемся коде процедуры "Button1_Click" наберите код, представленный на рис. 31, вычисляющий цену с учетом НДС, равный 18%.

```
private void button1_Click(object sender, EventArgs e)
{
    double ant, bnt;
    ant = Convert.ToDouble(цена_за_единицуTextBox.Text);
    bnt = (118 * ant) / 100;
    textBox1.Text = bnt.ToString();
}
```

Рисунок 31

32. Запустите проект и проверьте верность расчетов, выполняющихся в форме4.

Задание на самостоятельное изучение(Form3)

1. Выведите поле «Номер варианта», используя поля "Курс" числовой счетчик (задать объект NumericUpDown, расположен в «Источники данных»; перетащить на форму3).
2. Отобразите вместо поля «Код наименования» поле «Наименование». При этом сам выпадающий список будет заполнен наименованиями из таблицы "Наименования товара" и при выборе наименования ее код будет автоматически подставляться в поле "Код наименования" таблицы "Заказы".
3. Поместите справа от надписи "Код наименования", неподключенный ни к каким полям выпадающий список. Для создания выпадающего списка на панели объектов воспользуйтесь кнопкой ComboBox.
4. После создания выпадающего списка подключите его к полю " Код наименования " из таблицы "Заказы" и настроим заполнение списка значениями поля "Наименование товара".
5. Для этого выделите вновь созданный выпадающий список, отобразите меню действий и в меню действий включите опцию "Use data bound items" (Использовать связанные с данными элементы списка)
6. В панели действий под опцией "Use data bound items" расположены следующие параметры:

- **Data Source** (Источник данных) - определяет таблицу или запрос из которого заполняется список;
- **Display Member** (Член отображения) - определяет поле значениями которого заполняется список;
- **Value Member** (Член значений) - определяет значения какого поля подставляются в связанное с выпадающим списком поле;
- **Selected Value** (Выбранное значение) - определяет связанное с выпадающим списком поле.

7. Задайте вышеперечисленные параметры следующим образом:

- Параметр "DataSource" как "Other Data Sources\Project Data Sources\StudentsDataSet\Товары";
- Параметр "DataMember" задайте как "Наименование товара" ;
- Параметр "Value Member" задайте как "Код наименования";
- Параметр "Selected Value» задайте как "ЗаказыBindingSource\Код наименования".

Проверьте верность выполненной работы, запустив проект.

Вопросы

- Для чего требуется панель объектов
- Что может быть использовано вместо метода AddNew. Для чего используется этот метод
- Опишите процесс создания MaskedTextBox
- Для каких полей может быть создан CheckBox
- Как отобразить ссылки на адреса в интернете
- Перечислите способы подключения источников данных
- Каковы основные преимущества и недостатки этих способов
- Как можно использовать ListBox и ComboBox

- Как отобразить студента, у которого значение поля ФИО равно "Петров"(приведите пример кода одной строкой)
- Объясните, каким образом происходит сохранение в данной лабораторной работе при нажатии на кнопку «Сохранить»
- Как реализовать работу с вычисляемым полем, опишите поэтапно
- Какая процедура проверяет введенные в поля данные на соответствие типам данных полей

ПРАКТИЧЕСКАЯ РАБОТА №5

«DATAGRIDVIEW. СОЗДАНИЕ ТАБЛИЧНЫХ ФОРМ»

Цель

- Изучить объекты для отображения табличной информации DataGridView
- Рассмотреть настройка свойств столбцов в DataGridView
- Научиться создавать табличные формы

Общая информация

Объект **DataGridView** предназначен для отображения всей информации из таблиц, запросов или фильтров на форме в виде таблицы. Этот объект может быть создан как вручную (с последующим его подключением), так и перетаскиванием всего источника данных из окна "Data Sources". Однако наиболее часто его создают перетаскиванием всей таблицы, запроса или фильтра из окна "Data Sources" на форму.

При перетаскивании этого объекта на форму, как и в случае с другими объектами появляется панель навигации. Она выполняет функции: перемещение по записям, добавление, удаление и сохранение записей. После создания объекта DataGridViewможно настраивать как свойства всего объекта, так и свойства отдельных столбцов. Начнём с настройки свойств всего объекта. Настройка данных свойств осуществляется в основном через меню действий. Возможны следующие настройки:
Chose Data Source - источник данных, отображаемый в таблице;
Enable Adding - добавлять записи;
Enable Deleting - разрешается пользователям удалять записи;
Enable Editing - разрешается пользователям изменять значения полей таблицы;
Enable Column Reordering - разрешается пользователям изменять порядок столбцов, просто перетаскивая их мышью.

Также в меню действий возможны следующее действия с таблицей:
Dock in parent container - вписать объект в форму;
Preview Data - появляется окно с предварительным просмотром таблицы;
Add Query - добавляет SQL - запрос, который выполняется на стороне клиента;
Add Column - добавление нового столбца в таблицу;
Edit Columns - настройка свойств отдельных столбцов таблицы.

Если в меню действий выбрать пункт "Edit Columns", то появляется окно, где можно добавлять, удалять и редактировать столбцы. Для этого в списке столбцов левой части окна выбираем столбец, а в правой - настраиваем его свойства. Наиболее часто настраиваются следующие свойства:
Name - имя столбца;
AutoSizeMode - подгонка ширины столбца по его содержимому;
ColumnType - определяет внешний вид ячеек столбца (какой объект для отображения информации находится в ячейках столбца);
DataPropertyName - имя, отображающего в столбце поля;
Frozen - фиксация столбца (столбец не передвигается при прокручивании таблицы);
HeaderText - текст заголовка столбца;
Width - ширина поля;
MaxInputLength - максимально вводимая длина текста;
MinimumWidth - минимальная ширина столбца;
ReadOnly - блокировка столбца для редактирования данных;
Resizable - разрешает менять ширину столбца;

SortMode - сортировка данных в таблице по этому столбцу;
ToolTipText - всплывающая подсказка для столбца;
Visible - делает столбец невидимым.

Порядок выполнения работы

1. Запустите «Microsoft Visual Studio 2012» и откройте созданный ранее проект «Goods», щелкнув по его значку в области «Последние проекты» стартовой страницы «Стартовая страница». После появления стандартного окна среды разработки откройте форму «Form1.cs».
2. Добавьте кнопку «Табличная форма».
3. Добавьте в проект новую форму.
4. Поместите на нее:
   - четыре надписи (**Label**),
   - пять кнопок (**Button**),
   - выпадающий список (**ComboBox**),
   - текстовое поле ввода (**TextBox**),
   - группирующую рамку (**GroupBox**),
   - список (**ListBox**),
   - два переключателя (**RadioButton**).

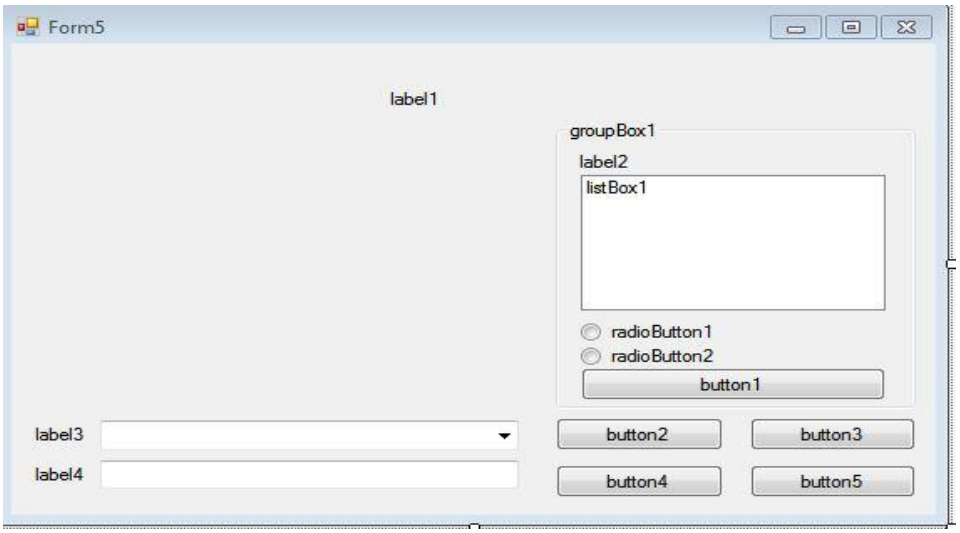
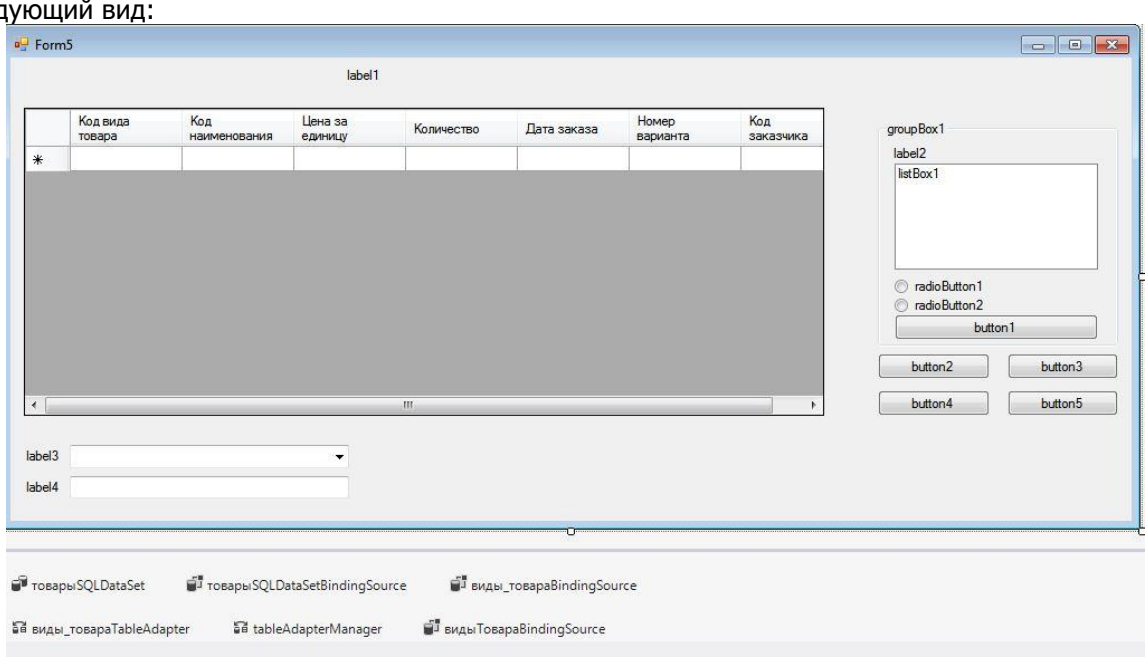


Рисунок1

5. Добавим на форму таблицу для отображения данных (DataGridView) из таблицы "Goods".
6. Перетащите таблицу "Заказы" из панели "Источники данных" на форму. Форма примет следующий вид:



7. Обратите внимание на то, что на форме появилась таблица для отображения данных, подключенная к таблице "заказы". Также появились объекты связи и панель навигации.

8. Теперь перейдем к настройке свойств объектов. Начнем с настройки свойств формы. Задайте **свойства формы** следующим образом:

FormBorderStyle (Стиль границы формы): Fixed3D;
MaximizeBox (Кнопка развертывания формы во весь экран): False;
MinimizeBox (Кнопка свертывания формы на панель задач): False;
Text (Текст надписи в заголовке формы): Таблица "Заказы" (Табличный вид).

Задайте свойства надписей (Label1, Label2, Label3 и Label4) как:

AutoSize (Авторазмер): False;
Text (Текст надписи): "Таблица "Студенты" (Табличный вид)", "Поле для сортировки", "Дата:" и "Критерий" (Соответственно для Label1, Label2, Label3 и Label4).
Для надписи Label1 задайте:

Font (Шрифт): Microsoft Sans Serif, размер 14;
ForeColor (Цвет текста): Темно синий;
TextAlign (Выравнивание текста): MiddleCenter.
Задайте надписи на кнопках как: "Сортировать", "Фильтровать", "Показать все", "Найти" и "Закрыть" (Соответственно для кнопок Button1, Button2, Button3, Button4 и Button5). Для того чтобы нельзя было произвести сортировку не выбрав поля изначально заблокируем кнопку "Сортировать" (Button1).

У группирующей рамки задайте заголовок (Свойство Text) равным "Сортировка". У переключателей (Объекты RadioButton1 и RadioButton2) задайте надписи как "Сортировка по возрастанию" и "Сортировка по убыванию", а у переключателя "Сортировка по возрастанию" (RadioButton1) задайте свойство Checked (Включен) равное True (Истина).

9. Настройте таблицу «DataGridView Задачи» - «Правка столбцов» и удалите «Код вида товара» и «Код наименования»:

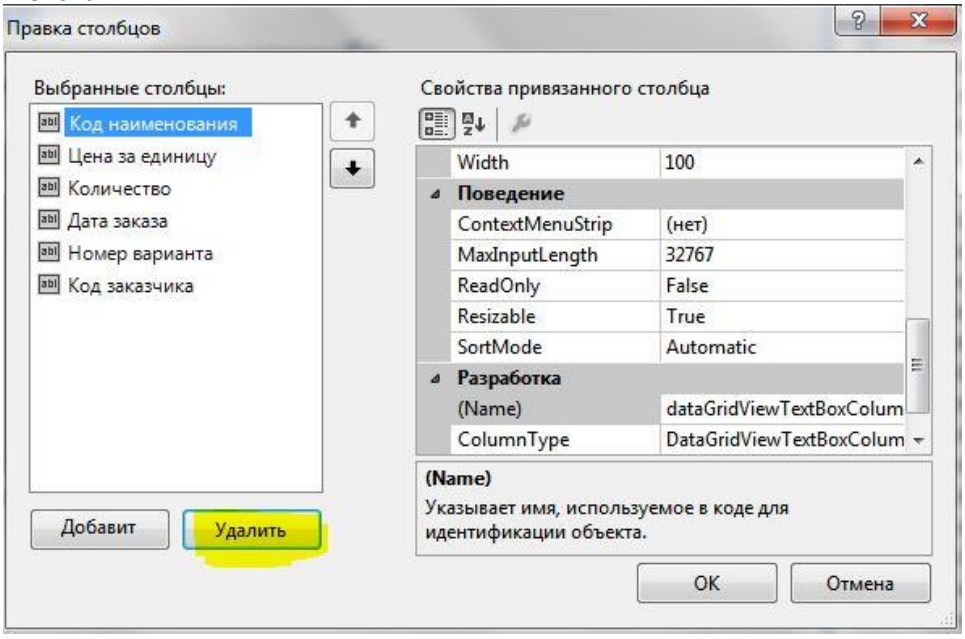


Рисунок3

10. Настроим заполнение выпадающего списка именами студентов из таблицы студенты. Отобразите меню действий выпадающего списка. Включите опцию "Use Data Bound Items". Установите параметр "Data Source" равным "Other Data Sources\Project Data Sources\StudentsDataSet\Студенты", а параметр "Display Member" равным "Дата". Остальные параметры оставьте без изменений

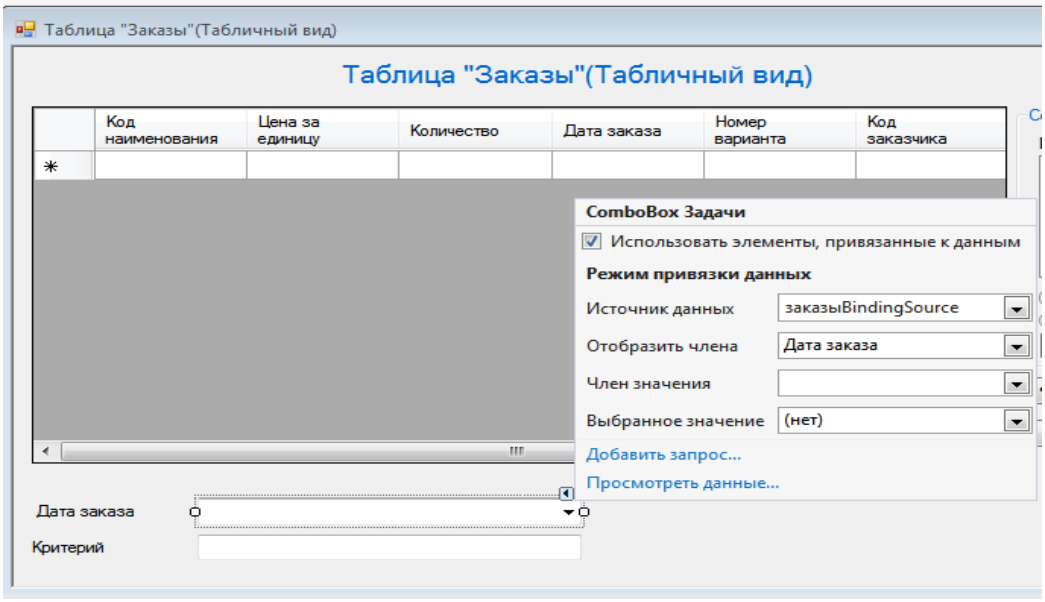


Рисунок4

11. Заполните ListBox значениями: «Цена за единицу», «Количество», «Дата заказа», «Номер варианта», «Код заказчика» как показано на рисунке 5:

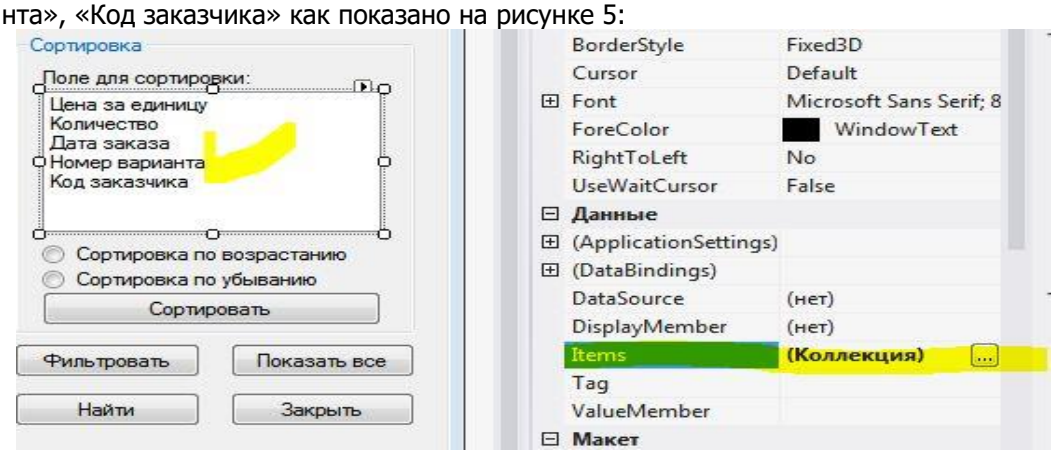


Рисунок 5

12. Для перехода к написанию кода, убедитесь, что ваша форма имеет вид, аналогичный рисунку 6:

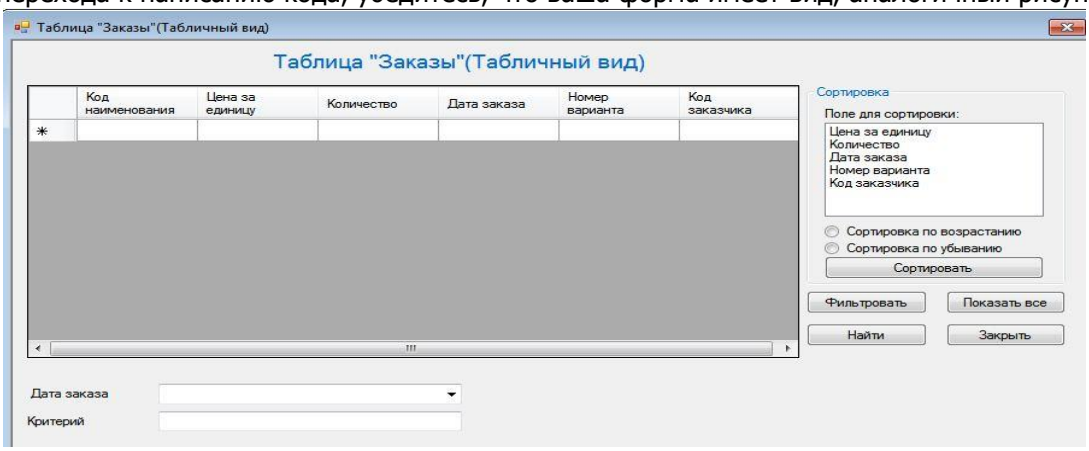


Рисунок 6

Работу с кодом начнем с написания кода для разблокирования кнопки "Сортировать", при выборе пункта списка (ListBox1). Для создания процедуры события дважды щелкните ЛКМ по списку. Появится процедура обработки события, происходящего при выборе пункта списка (ListBox1_SelectedIndexChanged). В процедуре наберите команду разблокировки кнопки "Сортировать" (Button1):

Button1.Enabled = true;

13. Теперь перейдем к созданию кода сортирующего нашу таблицу в зависимости от выбранного поля и порядка сортировки при нажатии кнопки "Сортировать". Дважды щелкните ЛКМ по кнопке "Сортировать". Появится процедура "Button1_Click", выполняемая при щелчке ЛКМ по кнопке. В процедуре наберите код:

```
private void button1_Click(System.Object sender, System.EventArgs e)
        {
            System.Windows.Forms.DataGridViewColumn Col =
default(System.Windows.Forms.DataGridViewColumn);
            switch (listBox1.SelectedIndex)
            {
                case 0:
                    Col = dataGridViewTextBoxColumn3;
                    break;
                case 1:
                    Col = dataGridViewTextBoxColumn4;
                    break;
                case 2:
                    Col = dataGridViewTextBoxColumn5;
                    break;
                case 3:
                    Col = dataGridViewTextBoxColumn6;
                    break;
                case 4:
                    Col = dataGridViewTextBoxColumn7;
                    break;
            }
            if (radioButton1.Checked)
            {
                заказы1DataGridView.Sort(Col,
System.ComponentModel.ListSortDirection.Ascending);
            }
            else
            {
                заказы1DataGridView.Sort(Col,
System.ComponentModel.ListSortDirection.Descending);
            }
                 }
```

Рассмотрим код более подробно:

Команда Col создает переменную Col для хранения имени выбранного столбца таблицы;
Затем следует блок switch, присваивающий в переменную Col имя выбранного столбца таблицы в зависимости от номера выбранного пункта списка (ListBox1.SelectedIndex). Если выбран первый пункт списка, то в переменную Col записывается столбец DataGridViewTextBoxColumn2, если второй, то - DataGridViewTextBoxColumn3 и так далее. Хотелось бы отметить тот факт, что нумерация пунктов списка начинается с нуля, а нумерация столбцов с единицы. Первый столбец «дата заказа» носит имя DataGridViewTextBoxColumn2, так как имя DataGridViewTextBoxColumn1 имеет столбец заголовков строк;
Блок If… выполняет следующую операцию: если включен переключатель "Сортировка по возрастанию" (RadioButton1), то отсортировать таблицу по полю заданному в переменной Col по возрастанию, иначе – по убыванию.

14. Рассмотрим код обработчика события нажатия кнопки "Фильтровать" (Button2). Дважды щелкните по кнопке "Фильтровать" и в процедуре обработки события "Button2_Click" наберите код:

```
заказыBindingSource.Filter = "Дата_заказа='" + ComboBox1.Text
                                    + "'";
```

15. Теперь перейдем к кнопке "Показать все", отменяющей фильтрацию записей. Дважды щелкните по вышеперечисленной кнопке. Появится процедура Button3_Click. В появившейся процедуре наберите команду, которая сбрасывает настройки фильтра(если присвоить свойству "Filter" значение пустой строки (""), то его действие будет отменено):

```
заказыBindingSource.Filter = "";
```

16. Запишите в обработчик нажатия кнопки «Закрыть» код:

```
this.Close();
```

17. Проверьте, как работает фильтрация и сортировка записей в таблице, нажимая на соответствующие кнопки. После проверки работы формы для возвращения в среду разработки просто закройте все формы.

Код для кнопки найти.

```
                private void button4_Click(object sender, EventArgs e)
        {
            int i = 0;
            int j = 0;

            for (i = 0; i < заказы1DataGridView.ColumnCount; i++)
            {
                for (j = 0; j < заказы1DataGridView.RowCount; j++)
                {
                    заказы1DataGridView.Rows[j].Cells[i].Style.BackColor = Color.White;
                    заказы1DataGridView.Rows[j].Cells[i].Style.ForeColor = Color.Black;
                }
            }
            for (i = 0; i < заказы1DataGridView.ColumnCount; i++)
            {
                for (j = 0; j < заказы1DataGridView.RowCount; j++)
                {
                    var value = заказы1DataGridView.Rows[j].Cells[i].Value;
                    if (value != null)
                    {
                        string baseStr = value.ToString();

                        if (baseStr.IndexOf(textBox1.Text) > -1)
                        {
                            заказы1DataGridView.Rows[j].Cells[i].Style.BackColor =
Color.Aqua;

                            заказы1DataGridView.Rows[j].Cells[i].Style.ForeColor =
Color.Blue;
                        }
                     }
                }
            }

        }
```

Вопросы

1. За что отвечает **Enable Deleting**
2. Как удалить ненужное поле
3. Как создать процедуру события
4. Какую конструкцию необходимо использовать для выбора необходимого столбца сортировки
5. С чего начинается нумерация столбцов
6. Для чего используется привязка данных в **ComboBox**
7. Как вписать объект в форму
8. Какими способами можно добавлять на форму **DataGridView**

ПРАКТИЧЕСКАЯ РАБОТА №6

«ОСНОВНЫЕ КОМПОНЕНТЫ MICROSOFT SQL SERVER 2008»

Цель

- Изучить систему основных компонентов Microsoft SQL Server 2008
- Понять процесс создания файла данных
- Освоить управление базами данных при помощи команд языка T-SQL

Общая информация

Все компоненты Microsoft SQL Server 2008 запускаются из меню "Пуск \ Программы \ Microsoft SQL Server 2008. В Microsoft SQL Server 2008 входят следующие компоненты:

Deployment Wizard - мастер по выводу информации хранимой на сервере;
SQL Server Installation Center - центр установки SQL Server 2008;
Reporting Services Configuration Manager - менеджер службы настройки отчётов;
SQL Server Configuration Manager - менеджер настройки сервера;
SQL Server Error and Usage Reporting - служба протоколирования работы сервера и служба отчётов об ошибках;
Microsoft Samples Overview - ссылка на сайт корпорации Microsoft, где можно просмотреть примеры работы с сервером;
SQL Server Books Online - полная справочная система по Microsoft SQL Server 2008. Она содержит справки, как по программированию, так и по администрированию сервера;
SQL Server Tutorials - учебники по работе с сервером;
Data Profile Viewer - просмотр профилей по работе с данными;
Execute Package Utility - инструменты по сжатию данных;
Database Engine Tuning Advisor - мастер настройки ядра базы данных;
SQL Server Profiler - настройка профилей по работе с данными;
Import and Export Data - импорт и экспорт данных;
SQL Server Business Intelligence Development Studio - интегрированная среда разработки Business Intelligence Development Studio;
SQL Server Management Studio - графическая оболочка для управления сервером и разработки баз данных.

Создание файла данных

Новую БД можно создать, используя стандартные команды языка T-SQL. Для создания новой БД необходимо сделать активную БД "Master". Это можно сделать либо выбором ее из выпадающего списка БД на панели инструментов, либо набором команды USE Master на вкладке нового запроса.

Все команды языка T-SQL набираются на вкладке нового запроса (SQLQuery). В Microsoft SQL Server БД состоит из двух частей:

- Файл данных - файл, имеющий расширение mdf и где находятся все таблицы и запросы;
- Файл журнала транзакций - файл, имеющий расширение ldf, содержит журнал, где фиксируются все действия с БД. Данный файл предназначен для восстановления БД в случае её выхода из строя.

Для создания нового файла данных используется команда CREATE DATABASE, которая имеет следующий синтаксис:

CREATE DATABASE <Имя БД>
ON (Name=<Логическое имя>,
FileName=<Имя файла>
[Size=<Нач.размер>,]
[Maxsize=<Макс.размер>,]
[FileGrowth=<Шаг>])
[LOG ON
(Name=<Логическое имя>,
FileName=<Имя файла>
[Size=<Нач.размер>,]
[Maxsize=<Макс.размер >,]
[FileGrowth=<Шаг>])

Имя БД - имя создаваемой БД
Логическое имя - определяет логическое имя файла данных БД, по которому происходит обращение к файлу данных.
Имя файла - определяет полный путь к файлу данных.
Нач.размер - начальный размер файла данных в Мб.
Макс.размер - максимальный размер файла данных в Мб.
Шаг - шаг увеличения файла данных, либо в Мб либо в %.
Параметры в разделе LOG ON аналогичны параметрам в разделе CREATE DATABASE. Однако они определяют параметры журнала транзакций.

В языке запросов T-SQL с БД возможны следующие действия:
Отображение сведений о БД: EXEC sp_helpdb <Имя БД> ;
Изменение параметров БД: EXEC sp_dboption <Имя БД>, <Параметр>, <Значение> ;
Добавления новых файлов, удаление файлов и переименования файлов, входящих в БД:

ALTER DATABASE <Имя БД>
ADD FILE (<Параметры>)|
REMOVE FILE <Логическое имя файла>|
MODIFY FILE (<Параметры>)

где, раздел ADD FILE - добавляет файл, REMOVE FILE - удаляет, а раздел MODIFY FILE - изменяет параметры файла;
Сжатие всей БД: DBCC SHRINKDATABASE <Имя БД>;
Сжатие конкретного файла БД: DBCC SHRINKFILE <Логическое имя файла>;
Переименование БД: EXEC SP_RENAMEDB <Имя БД>,<Новое имя БД>;
Удаление БД: DROP DATABASE <Имя БД>.

Замечание: Вышеперечисленные команды используют следующие параметры:
<Имя БД> - имя БД с которой производится действие;
<Параметр> - изменяемый параметр;
<Значение> - новое значение изменяемого параметра;
<Параметры> - параметры файла БД, аналогичные параметрам, используемым в команде *CREATE DATABASE*;
<Логическое имя файла> - логическое имя файла, входящего в БД;
<Новое имя БД> - новое имя БД.

Ход выполнения работы

1. Откройте среду SQL Server Manager Studio

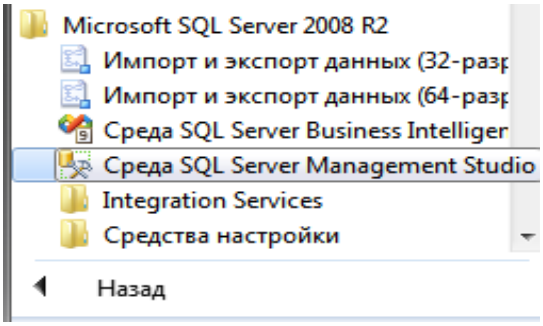


Рисунок 12

2. После запуска среды разработки появится окно подключения к серверу "Connect to Server"

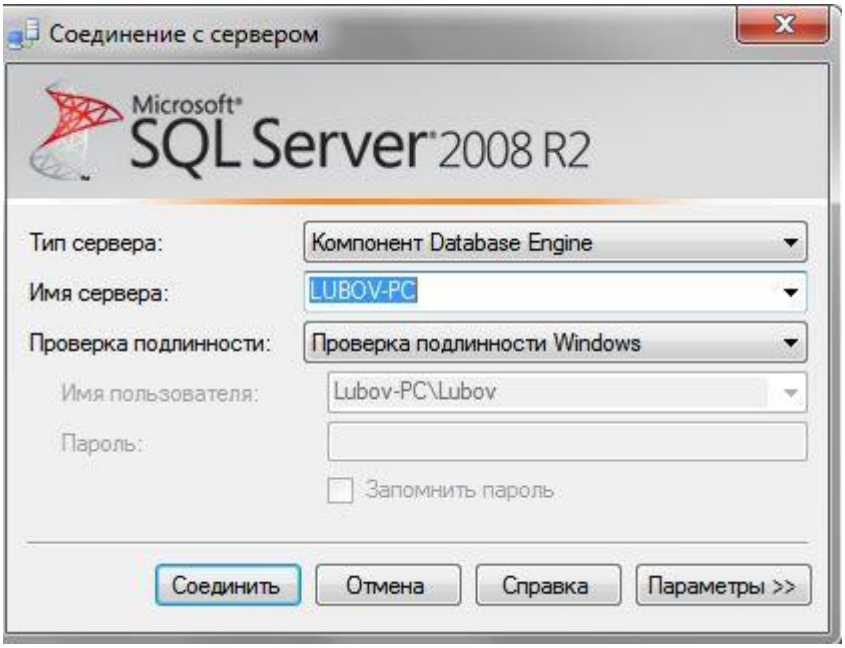


Рисунок 13

Если при установке "Microsoft SQL Server 2008" был задан логин и пароль подключения к серверу, то перед нажатием кнопки "Connect", в выпадающем списке "Authentication" нужно выбрать "SQL Server Authentication", а затем необходимо ввести заданные при установке логин и пароль.

3. После нажатия кнопки "Connect" появится окно среды разработки "SQL Server Management Studio"

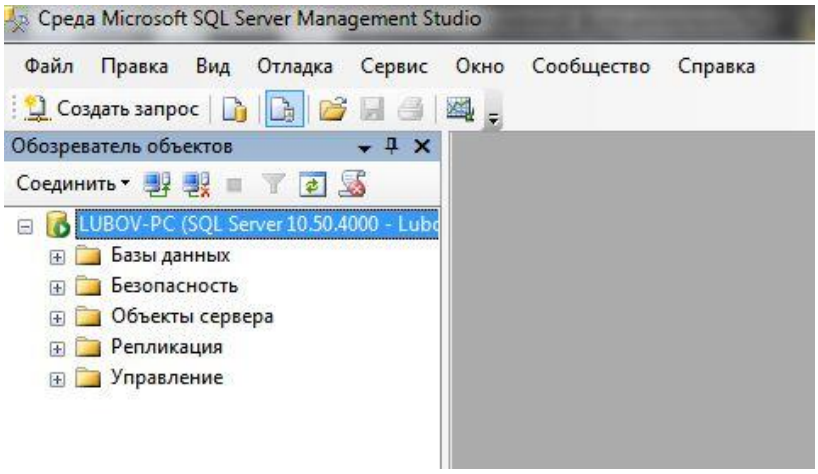


Рисунок 14

Оконное меню(Файл, Правка и т.д.) - содержит полный набор команд для управления сервером и выполнения различных операций.

Панель инструментов (Создать запрос и т.д.) - содержит кнопки для выполнения наиболее часто производимых операций. Внешний вид данной панели зависит от выполняемой операции.

Панель "Обозреватель объектов" - обозреватель объектов. Обозреватель объектов - это панель с древовидной структурой, отображающая все объекты сервера, а также позволяющая производить различные операции, как с самим сервером, так и с БД. Обозреватель объектов является основным инструментом для разработки БД. В обозревателе объектов сами объекты находятся в папках. Чтобы открыть папку необходимо щелкнуть по знаку "+" слева от изображения папки.

Рабочая область(серый цвет) - В рабочей области производятся все действия с БД, а также отображается ее содержимое.

4. Создайте запрос, представленный на рисунке 15.

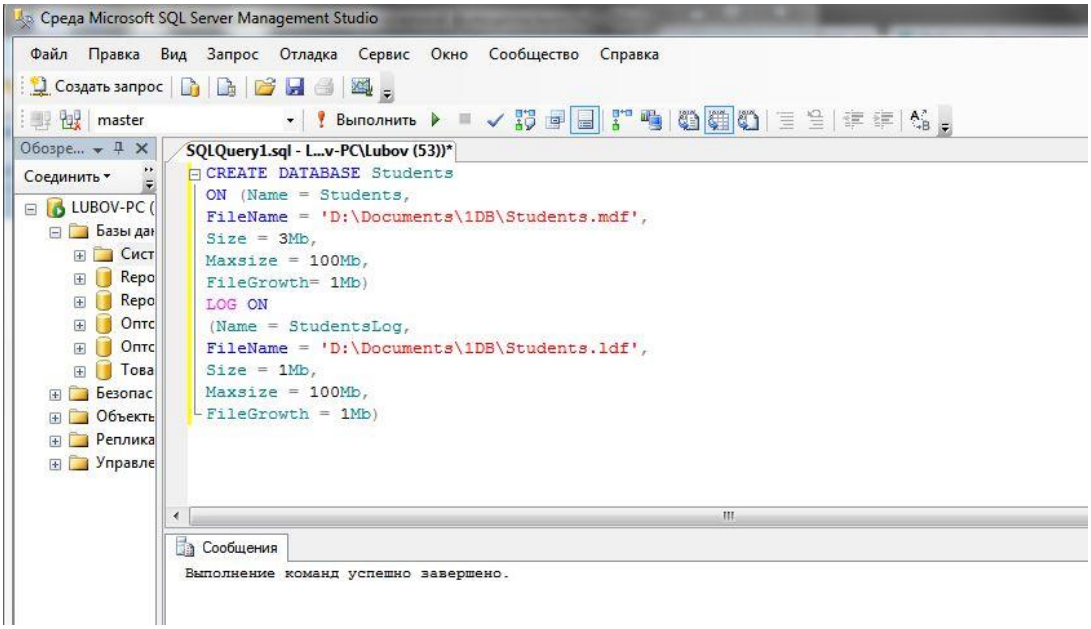


Рисунок 15

5. Просмотрите успешность выполнения данной команды, зайдите по указанному вами пути.

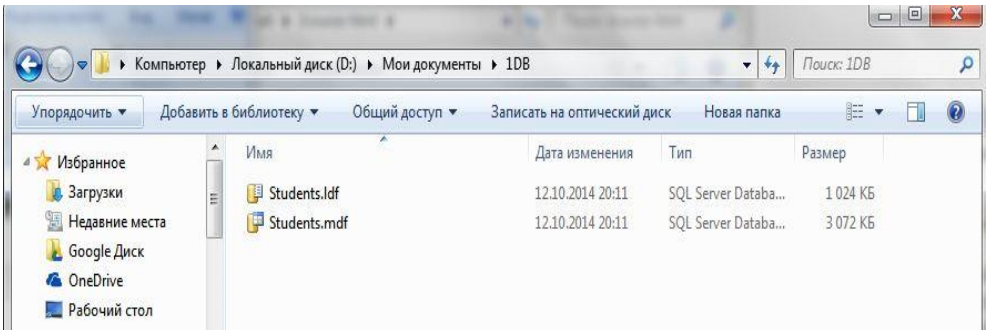


Рисунок 16

6. Перейдем непосредственно к созданию файла данных. Для этого в обозревателе объектов щелкните ПКМ на папке «Базы данных» и в появившемся меню выберите пункт «Создать БД». Появится окно настроек параметров файла данных новой БД. В левой части окна настроек имеется список «Выбор настроек». Этот список позволяет переключаться между группами настроек.

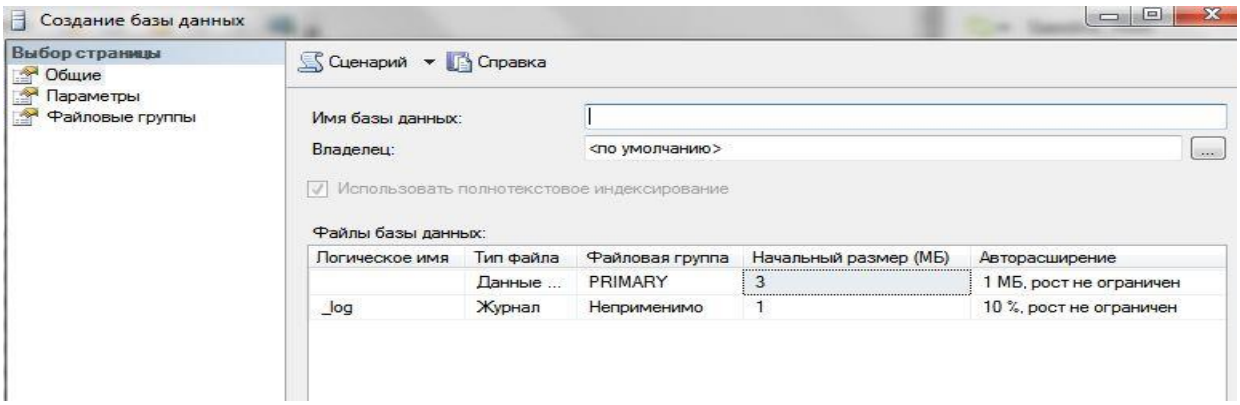


Рисунок 17

7. Для начала настроим основные настройки «Общие».

   Рассмотрим их более подробно. В верхней части окна расположено два параметра: «Имя БД» и «Владелец». Задайте параметр «Имя БД» равным «Student». Владельца оставьте без изменений.
   Под вышеприведенными параметрами в виде таблицы располагаются настройки файла данных и журнала транзакций. Таблица имеет следующие столбцы:
   Логическое имя - логическое имя файла данных и журнала транзакций. По этим именам будет происходить обращение к вышеприведенным файлам в БД. Можно заметить, что файл данных имеет то же имя что и БД, а имя файла журнала транзакций составлено из имени БД и суффикса "_log".
   Тип файла - тип файла. Этот параметр показывает, является ли файл файлом данных или журналом транзакций.
   Файловая группа - группа файлов, показывает к какой группе файлов относится файл. Группы файлов настраиваются в группе настроек.
   Начальный размер (MB) - начальный размер файла данных и журнала транзакций в мегабайтах.
   Авторасширение - автоувеличение размера файла. Как только файл заполняется информацией его размер автоматически увеличивается на величину, указанную в параметре «Авторасширение». Увеличение можно задавать как в мегабайтах так и в процентах. Здесь же можно задать максимальный размер файлов. Для изменения этого параметра надо нажать кнопку "…". В нашем случае размер файлов не ограничен. Файл данных увеличивается на 1 мегабайт, а файл журнала транзакций на 10%.
   Путь - путь к папке, где хранятся файлы. Для изменения этого параметра также надо нажать кнопку "…".Имя файла по умолчанию имена файлов аналогичны логическим именам. Однако файл данных имеет расширение "mdf", а файл журнала транзакций - расширение "ldf".В нашем случае мы оставим все основные настройки без изменений.

8. Теперь перейдем к другим второстепенным настройкам файла данных. Для доступа к этим настройкам необходимо щелкнуть мышью по пункту «Параметры» в списке «Выбор страницы». Появится приведенное ниже окно. Параметры задайте аналогичные.

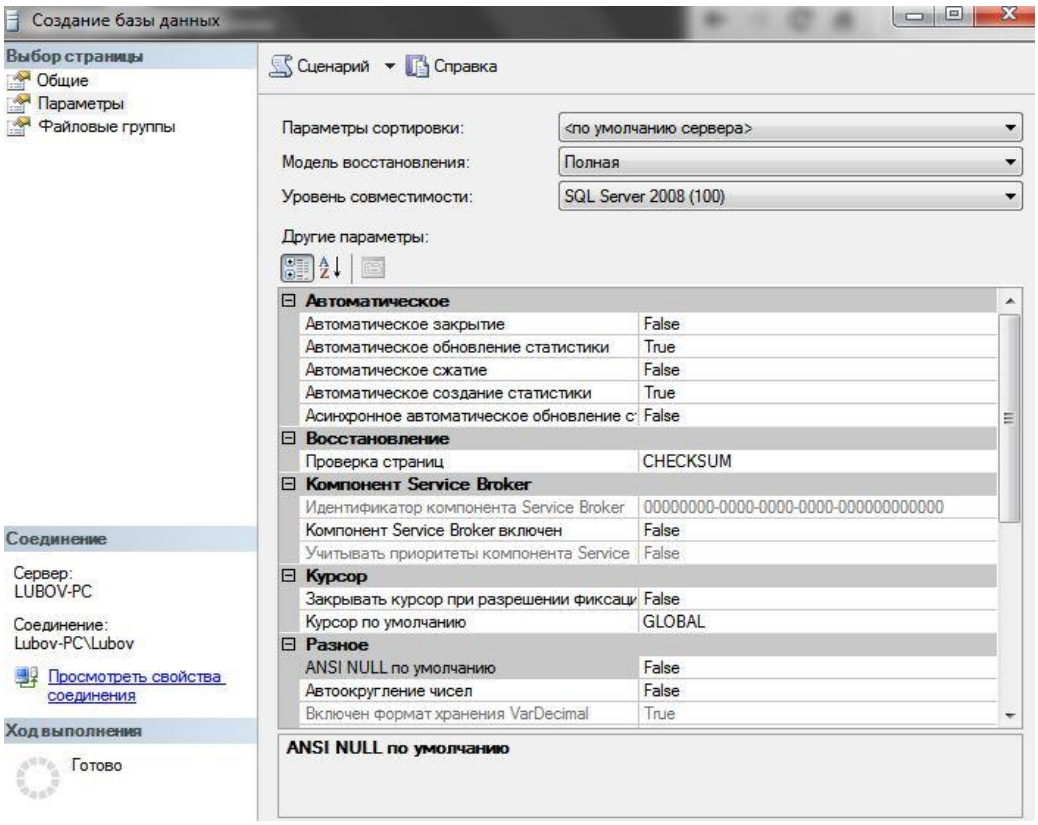


Рисунок 18

Параметры сортировки - этот параметр отвечает за обработку текстовых строк, их сравнение, текстовый поиск и т.д. Рекомендуется оставить его как "<server default>". При этом данный параметр будет равен значению, заданному на вкладке "Collation", при установке сервера.
Модель восстановления - модель восстановления. Данный параметр отвечает за информацию, предназначенную для восстановления БД, хранящуюся в файле транзакций. Чем полнее модель восстановления, тем больше вероятность восстановления данных при сбое системы или ошибках пользователей, но и больше размер файла журнала транзакций. При наличии места на диске, рекомендуется оставить этот параметр в значении "Full".
Уровень совместимости - уровень совместимости, определяет совместимость файла данных с более ранними версиями сервера. Если планируется перенос данных на другую, более раннюю версию сервера, то ее необходимо указать в этом параметре.
Другие параметры - второстепенные параметры. Данные параметры являются необязательными для изменения.

9. Откройте «Файловые группы». Группы файлов представлены в таблице «Строки» в правой части окна. Данная таблица имеет следующие столбцы:
   Имя - имя группы файлов.
   Файлы - количество файлов входящих в группу.
   Только для чтения - файлы в группе будут только для чтения: их можно только просматривать, но нельзя изменять.
   По умолчанию - все новые файлы данных будут входить в эту группу.
   В рассматриваемой БД нет необходимости добавлять новые группы файлов. Поэтому оставим эту группу настроек без изменений.

10. На этом мы заканчиваем настройку свойств наших файлов. Для принятия всех настроек и создание файла данных и журнала транзакций нашей БД в окне «Новая БД» нажмем кнопку «Ok».
    Произойдет возврат в окно среды разработки «SQL Server Management Studio». На панели обозревателя объектов в папке «Базы данных» появится новая БД «Student».

11. Для переименования БД необходимо в обозревателе объектов щелкнуть по ней ПКМ и в появившемся меню выбрать пункт «Переименовать». Аналогично для удаления, обновления.

12. Для создания таблицы «Студент»:

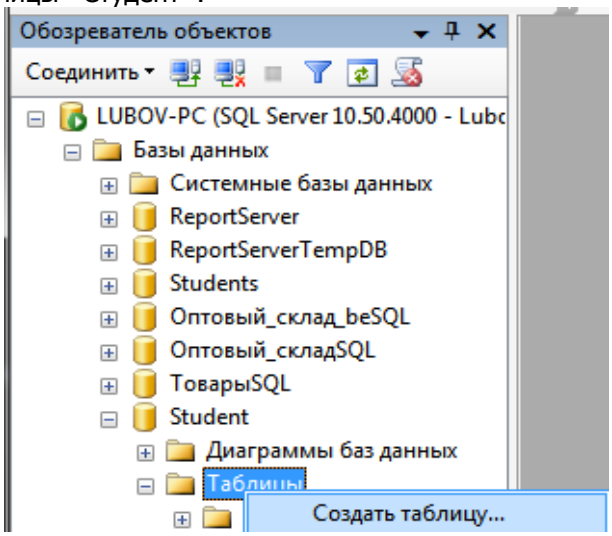
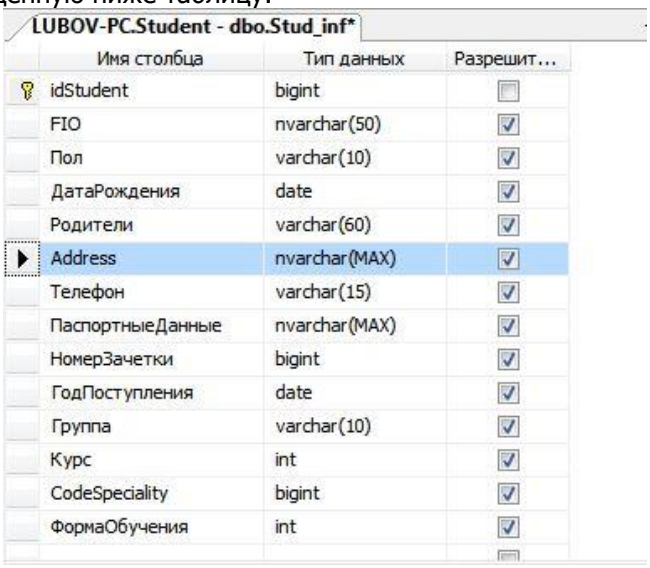


Рисунок 19

13. Создайте приведенную ниже таблицу:



Имя столбца	Тип данных	Разрешит...
idStudent	bigint	☐
FIO	nvarchar(50)	☑
Пол	varchar(10)	☑
ДатаРождения	date	☑
Родители	varchar(60)	☑
Address	nvarchar(MAX)	☑
Телефон	varchar(15)	☑
ПаспортныеДанные	nvarchar(MAX)	☑
НомерЗачетки	bigint	☑
ГодПоступления	date	☑
Группа	varchar(10)	☑
Курс	int	☑
CodeSpeciality	bigint	☑
ФормаОбучения	int	☑

Рисунок 20

Рассматривая поля новой таблицы можно прийти к следующим выводам:

Поле "Код студента" - это первичное поле для связи с таблицей оценки. Следовательно, данное поле необходимо сделать числовым счетчиком и ключевым (см. создание таблицы "Специальности" выше);
Поля "ФИО", "Пол", "Родители", "Адрес", "Телефон", "Паспортные данные" и "Группа" являются текстовыми полями различной длины (для задания длины выделенного текстового поля необходимо в таблице свойств выделенного поля установить свойство Length равное максимальному количеству знаков текста вводимого в поле);
Поля "Дата рождения" и "Дата поступления" предназначены для хранения дат. Поэтому они имеют тип данных "date";
Поле "Очная форма обучения" является логическим полем. В "Microsoft SQL Server 2008" такие поля должны иметь тип данных "bit";

Поля "Номер зачетки" и "Курс" являются целочисленными. Единственным отличием является размер полей. Поле "Номер зачетки" предназначено для хранения целых чисел в диапазоне -263…+263 (тип данных "bigint"). Поле "Курс " предназначено для хранения целых чисел в диапазоне 0…255 (тип данных "tinyint");
Поле "Код специальности" - это поле связи с таблицей "Специальности". Однако, данное поле связи является вторичным, поэтому его можно сделать просто целочисленным, то есть, "bigint".

14. Создайте аналогичным образом таблицу «Оценки»

Код студента	int
Дата экзамена1	date
Код предмета1	bigint
Оценка1	tinyint
Дата экзамена2	date
Код предмета2	bigint
Оценка2	tinyint
Дата экзамена3	date
Код предмета3	bigint
Оценка3	tinyint
Средний балл	real

15. Создайте аналогичным образом таблицу «Предметы»

КодПредмета	bigint
НаименованиеПредмета	Varchar(30)
ОписаниеПредмета	Varchar(MAX)

16. -//- таблицу «Специальности»

КодСпециальности	bigint
НаименованиеСпециальности	Varchar(30)
ОписаниеСпециальности	Varchar(MAX)

- Для отображения названия поля с пробелами, указывайте название в [] :
[Код специальности]

17. В конце выполнения данной лабораторной работы у вас должна получиться БД с 4 таблицами.

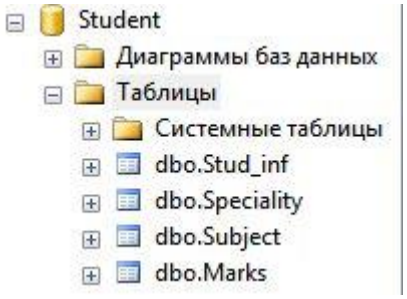


Рисунок 21

Вопросы

1. Как можно создать БД, перечислите способы
2. Что находится в «Обозревателе объектов»
3. С помощью какой команды происходит сжатие всей БД
4. Для чего используется «Логическое имя»
5. Перечислите основные настройки группы «Общие»
6. Что такое «Файловая группа»
7. Как создать новую таблицу
8. Какие типы данных вы узнали в ходе выполнения этой ЛР

ПРАКТИЧЕСКАЯ РАБОТА №7
«ЗАПОЛНЕНИЕ БД ДАННЫМИ»

Цель

- ❖ Изучить процесс заполнения БД необходимой для дальнейшей работы информацией
- ❖ Познакомиться с нормализацией таблиц

Общая информация

При создании и заполнении таблиц информационной системы необходимо следовать 3 правилам:

- В таблицах не должно быть повторяющихся групп записей. Это достигается введением индексных полей, то есть сортировкой записей;
- В таблице не должно быть полей с одинаковыми именами. Это достигается разбиением одной таблицы на несколько, с последующим связыванием их запросом;
- Не должно быть правил при заполнении таблиц, это достигается хаотичностью заполнения таблиц базы данных.

Информационная система, которая удовлетворяет этим условиям, называется нормализованной информационной системой или базой данных.

Нормализация — это процесс организации данных в базе данных, включающий создание таблиц и установление отношений между ними в соответствии с правилами, которые обеспечивают защиту данных и делают базу данных более гибкой, устраняя избыточность и несогласованные зависимости.

Избыточность данных приводит к непродуктивному расходованию свободного места на диске и затрудняет обслуживание баз данных. Например, если данные, хранящиеся в нескольких местах, потребуется изменить, в них придется внести одни и те же изменения во всех этих местах. Изменение адреса клиента гораздо легче реализовать, если в базе данных эти сведения хранятся только в таблице Customers и нигде больше.

Что такое «несогласованные зависимости»? Пользователь, которому нужно узнать, например, адрес определенного клиента, вполне обоснованно будет искать его в таблице Customers (клиенты), но искать в ней сведения о зарплате сотрудника, который работает с этим клиентом, не имеет смысла. Зарплата сотрудника связана с сотрудником (зависит от него), поэтому эти сведения следует хранить в таблице Employees (сотрудники). Несогласованные зависимости могут затруднять доступ к данным, так как путь к данным при этом может отсутствовать или быть неправильным.

Существует несколько правил нормализации баз данных. Каждое правило называется «нормальной формой». Если выполняется первое правило, говорят, что база данных представлена в «первой нормальной форме». Если выполняются три первых правила, считается, что база данных представлена в «третьей нормальной форме». Есть и другие уровни нормализации, однако для большинства приложений достаточно нормализовать базы данных до третьей нормальной формы.

Как и в случае со многими другими формальными правилами и спецификациями, обеспечить полное соответствие реальным ситуациям не всегда возможно. Как правило, для выполнения нормализации приходится создавать дополнительные таблицы, и некоторые клиенты считают это нежелательным. Собираясь нарушить одно из первых трех правил нормализации, убедитесь в том, что в приложении учтены все связанные с этим проблемы, такие как избыточность данных и несогласованные зависимости.

Первая нормальная форма

Устраните повторяющиеся группы в отдельных таблицах.
Создайте отдельную таблицу для каждого набора связанных данных.
Идентифицируйте каждый набор связанных данных с помощью первичного ключа.

Не используйте несколько полей в одной таблице для хранения похожих данных. Например, для слежения за товаром, который закупается у двух разных поставщиков, можно создать запись с полями, определяющими код первого поставщика и код второго поставщика. Что произойдет при добавлении третьего поставщика? Добавление третьего поля нежелательно, так как для этого нужно изменять программу и таблицу, поэтому данный способ плохо адаптируется к динамическому изменению числа поставщиков. Вместо этого можно поместить все сведения о поставщиках в отдельную таблицу Vendors (поставщики) и связать товары с поставщиками с помощью кодов товаров или поставщиков с товарами с помощью кодов поставщиков.

Вторая нормальная форма

Создайте отдельные таблицы для наборов значений, относящихся к нескольким записям. Свяжите эти таблицы с помощью внешнего ключа. Записи могут зависеть только от первичного ключа таблицы (составного ключа, если необходимо). Возьмем для примера адрес клиента в системе бухгалтерского учета. Этот адрес необходим не только таблице Customers, но и таблицам Orders, Shipping, Invoices, Accounts Receivable и Collections. Вместо того чтобы хранить адрес клиента как отдельный элемент в каждой из этих таблиц, храните его в одном месте: или в таблице Customers, или в отдельной таблице Addresses.

Третья нормальная форма

Устраните поля, не зависящие от ключа. Значения, входящие в запись и не являющиеся частью ключа этой записи, не принадлежат таблице. Если содержимое группы полей может относиться более чем к одной записи в таблице, подумайте о том, не поместить ли эти поля в отдельную таблицу.
Например, в таблицу Employee Recruitment (наем сотрудников) можно включить адрес кандидата и название университета, в котором он получил образование. Однако для организации групповой почтовой рассылки необходим полный список университетов. Если сведения об университетах будут храниться в таблице Candidates, составить список университетов при отсутствии кандидатов не получится. Таким образом, создайте вместо этого отдельную таблицу Universities и свяжите ее с таблицей Candidates при помощи ключа — кода университета.

Ход выполнения работы

1. Откройте ранее созданную БД «Student».
2. Выберете «Таблицы» -«Изменить первые 200 срок»

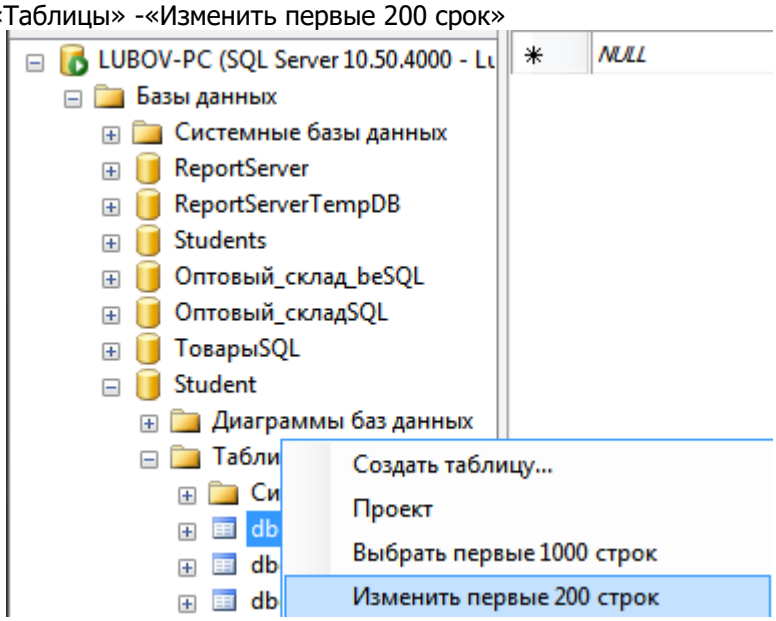


Рисунок 22

3. Заполните таблицы, поместив **в каждую(!)** из них по 35 записей, аналогично приведенным ниже.
4. «Специальности» Так как поле "Код специальности" является первичным полем связи и ключевым числовым счетчиком, то оно заполняется автоматически (заполнять его не нужно).

LUBOV-PC.Student - dbo.Speciality

idSpeciality	NameSpeciality	AboutSpeciality
1	Математика	NULL
2	Экономика	NULL
3	Химия	Программа бакалавриата 2-го поколения ГОС
4	Нейробиология...	Срок обучения 4 года, программа бакалавриата ФГОС
5	Физика	Срок обучения 5,5 лет
6	Переводоведе...	Срок обучения 5 лет
NULL	NULL	NULL

Рисунок 23

5. «Предмет»

LUBOV-PC.Student - dbo.Subject

idSubject	NameSubject	AboutSubject
1	История	NULL
2	Физика	NULL
3	РиСПСиИТ	Разработка и стандартизация программных средств и информационных технологий
4	МИР	Мировые информационные ресурсы
5	ТСиСА	Теория систем и системный анализ
NULL	NULL	NULL

Рисунок 24

6. «Студент» Для заполнения дат в качестве разделителя можно использовать знак ".". Даты можно заполнять в формате "день.месяц.год". Поле "Код специальности" является вторичным полем связи (для связи с таблицей "Специальности"). Следовательно, значения этого поля необходимо заполнять значениями поля "Код специальности" таблицы "Специальности". В нашем случая это значения от 1 до 35. Если у Вас коды специальностей в таблице "Специальности" имеют другие значения, то внесите их в данную таблицу.

LUBOV-PC.Student - dbo.Stud_inf

idСту...	ФИО	Пол	ДатаРожд...	Родители	Адрес	Телефон	Паспорт...	НомерЗа...	ДатаПосту...	Гру...	Курс	КодСпеци...	ОчнаяФорма
1	Агрыз...	женский	1993-01-20	Агрызков...	Ростов-на-...	89903422...	2324 546...	11334	2012-08-20	ВБПИ41	4	230700	True
2	Волоб...	мужской	1993-03-12	Волобуева...	Светлогра...	99806531...	6006 345...	11323	2012-08-20	ВБПИ41	4	230700	True
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 25

7. «Оценки». Поля с датами заполняются, как и в таблице "Студенты" (см. выше). Поля "Код предмета 1", "Код предмета 2" и "Код предмета 3" являются вторичными полями связи с таблицей "Предметы". Поэтому они должны быть заполнены значениями поля "Код предмета из этой таблицы", то есть значениями от 1 до 35.

LUBOV-PC.Student - dbo.Marks

idStudent	DateOfExam1	idSubject1	mark1	DateOfExam2	idSubject2	mark2	DateOfExam3	idSubject3	mark3	AVGMARK
1	2013-01-20	1	5	2013-01-21	2	5	2013-01-22	3	5	NULL
2	2013-01-20	1	4	2013-01-21	2	4	2013-01-22	3	4	NULL
3	2013-01-20	1	3	2013-01-21	2	4	2013-01-22	3	4	NULL
4	2013-01-20	1	5	2013-01-21	2	5	2013-01-22	3	4	NULL
5	2013-01-20	1	4	2013-01-21	2	3	2013-01-22	3	4	NULL
6	2013-01-20	1	4	2013-01-21	2	3	2013-01-22	3	3	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 25

8. По окончании заполнения, закройте окно заполнения таблиц. На этом мы заканчиваем создание и заполнение таблиц нашей БД "Students".

Вопросы

1. Дайте определение понятию «нормализация таблиц»
2. Что такое нормальная форма
3. Перечислите признаки 1НФ
4. Для чего служит 2НФ
5. Как осуществляется связь между таблицами
6. Как заполнить таблицу данными

ПРАКТИЧЕСКАЯ РАБОТА №8

«СОЗДАНИЕ ЗАПРОСОВ И ФИЛЬТРОВ. ВЫЧИСЛЕНИЕ ПРИ ПОМОЩИ ОПЕРАТОРА SELECT. ВСТРОЕННЫЕ ФУНКЦИИ»

Цель

- Изучить создание запросов и фильтров
- Понять процесс выполнения вычислений при помощи оператора SELECT.
- Познакомиться со встроенными функциями

Общая информация

Запросы предназначены для связи одной или нескольких таблиц, также они могут осуществлять отбор отдельных полей из таблицы и производить фильтрацию данных согласно условию, наложенному на одно или несколько полей, такие запросы называют фильтрами. Для реализации запросов используют специальный язык запросов SQL (Structured Query Language).

В ИС Запросы могут находиться как на стороне клиентского приложения, так и на стороне сервера. Если запрос хранится на стороне клиента, то он прописывается внутри объекта связи. В этом случае клиентское приложение не зависит от файла данных. Файл данных содержит только таблицы, поэтому, мы легко можем модифицировать клиентское приложение, не затрагивая файл данных, но в этом случае запрос передается серверу через сеть, что может вызвать проблемы с безопасностью. Если запрос хранится или выполняется на сервере, то сам запрос выступает в качестве компонента БД, вся передача информации происходит внутри файла данных, т.е. внутри самого сервера, клиентскому приложению только передаются результаты выполнения запроса. В этом случае обеспечивается высокая защита данных, но в случае изменения запроса придется менять сам файл данных.

Все запросы делятся на:

- статические;
- динамические

Структура статических запросов неизменна в ходе работы с программой, а динамические запросы могут меняться в зависимости от ситуации. Обычно динамические запросы могут быть реализованы только при помощи запросов, выполняющихся на стороне клиента. Если необходимо реализовать динамические запросы, которые выполняются на стороне сервера, то в этом случае необходимо использовать хранимые процедуры.

Хранимые процедуры - SQL запрос, хранимый на стороне сервера и этот запрос имеет параметры, которые подставляются внутрь SQL кода. При вызове хранимой процедуры необходимо передавать в нее значения параметра.

В основном запрос или хранимая процедура либо реализует связь между таблицами, либо осуществляет фильтрацию данных, некоторые SQL запросы также могут производить вычисления. В случае связей между таблицами одна таблица всегда выступает первичной, а другая вторичной, связь происходит при помощи полей связи. При связи сопоставляются записи с одинаковыми

значениями полей связи. Первичная таблица всегда заполняется первой, а ее поле связи заполняется автоматически (тип данных - счетчик). Вторичная таблица всегда заполняется после заполнения первичной таблицы, значения ее поля связи подставляется из значений поля связи первичной таблицы. Поля связи должны иметь одинаковый тип данных.

Существует **четыре вида связи** между таблицами:

- **Одна к одной** - одному полю в первичной таблице соответствует одно поле во вторичной таблице;
- **Одна ко многим** - одному полю в первичной таблице соответствует несколько полей во вторичной таблице;
- **Многие к одной** - нескольким полям в первичной таблице соответствует одно поле во вторичной таблице;
- **Многие ко многим** - одному полю в первичной таблице соответствует несколько полей во вторичной таблице и наоборот.

Чтобы создать запрос необходимо сделать активной БД для которой создается запрос, затем в рабочей области редактора запросов создать запрос с помощью команды SELECT, имеющей следующий синтаксис:

```
SELECT [ALL|DISTINCT]
[TOP|PERCENT n]
<Список полей>
[INTO <Имя новой таблицы>]
[FROM  <Имя таблицы >]
[WHERE <Условие>]
[GROUP BY <Поле>]
[ORDER BY <Поле > [ASC|DESC]]
[COMPUTE AVG|COUNT|MAX|MIN|SUM(<Выражение>)]
```

Здесь параметры ALL|DISTINCT показывают, какие записи обрабатываются: ALL обрабатывает все записи, DISTINCT только уникальные, удаляются повторения записей. TOP n определяет какое количество записей обрабатывают, если указан PERCENT, то n указывает процент от общего числа записей. <Список полей> - здесь указываются отображаемые поля из таблиц через запятую. Раздел INTO если присутствует, то на основе результатов запроса создается новая таблица. Параметр INTO это имя новой таблицы. Раздел FROM: указываются таблицы и запросы, через запятую, которые участвуют в новом запросе.

Ход выполнения работы

1. Откройте БД, созданную ранее.
2. Все запросы БД хранятся в «Представлениях».
3. Создайте запрос **«Запрос Студенты+Специальности»**, связывающий таблицы **«Студенты»** и **«Специальности»** по полю связи **«Код специальности»**. Для создания нового запроса необходимо в обозревателе объектов в БД **«Students»** щелкнуть **ПКМ** по папке **«Представления»**:

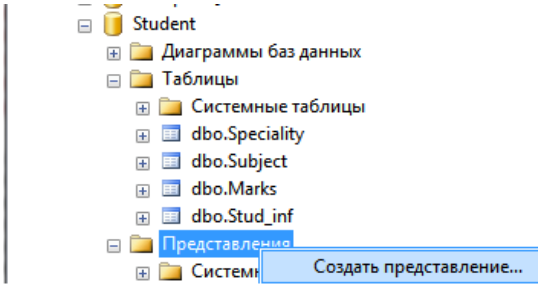


Рисунок 26

4. В появившемся окне выберете необходимые таблицы («Студенты», «Специальности»).

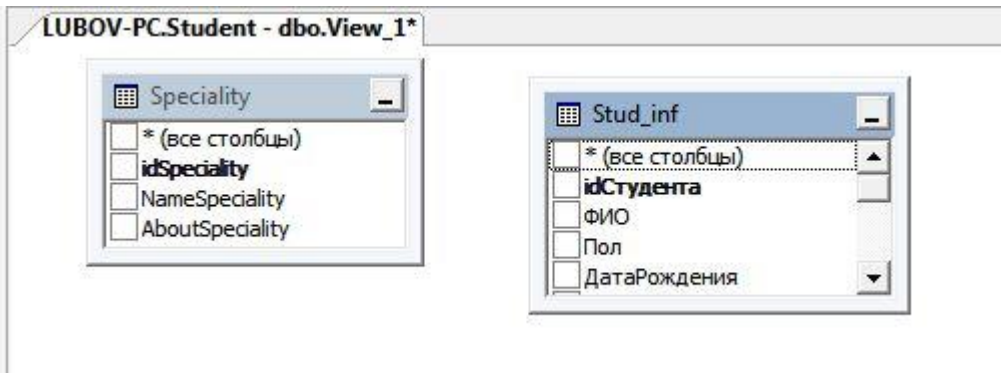


Рисунок 27

5. Задайте связь для этих таблиц: перетащите поде idSpeciality на КодСпециальности. Если необходимо удалить связь, то для этого необходимо щелкнуть по ней ПКМ и в появившемся меню выбрать пункт «Удалить».
6. Выберете данные как показано на рисунке 28.

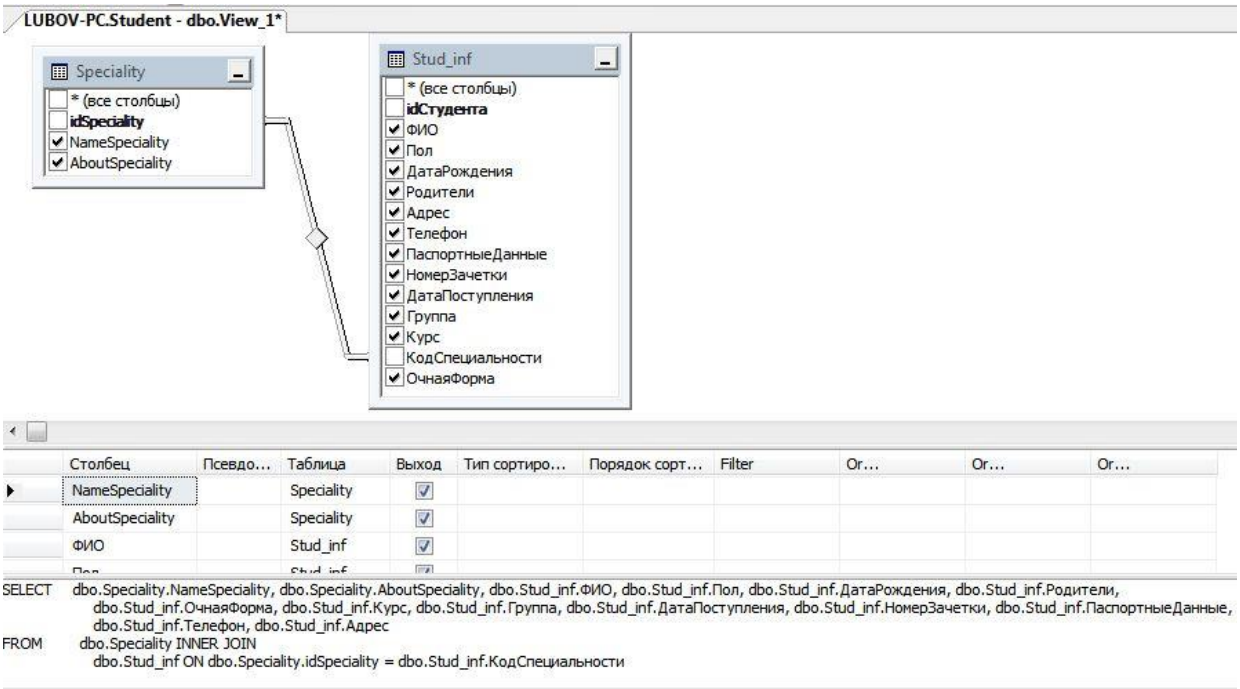


Рисунок 28

7. Если необходимо сделать поле невидимым при выполнении запроса, то нужно убрать галочку, расположенную слева от имени поля на схеме данных. Для этого просто щелкните мышью по галочке.
8. Если необходимо отобразить все поля таблицы, то необходимо установить галочку слева от пункта "* (Все столбцы), принадлежащего соответствующей таблице на схеме данных.
9. Проверьте работоспособность запроса. Выполните SQL-запрос. Для этого щелкните ПКМ в любом месте окна конструктора запросов и в появившемся меню выберите пункт «Выполнить SQL». Результат выполнения запроса появиться в виде таблицы в области результата
10. Если запрос выполняется правильно, то необходимо сохранить. Для сохранения запроса закройте окно конструктора запросов, щелкнув мышью по кнопке закрытия, расположенной в верхнем правом углу окна конструктора (над схемой данных). Появится окно с вопросом о сохранении запроса и выбором имени запроса.
11. Теперь данный запрос можно просмотреть в обозревателе объектов «Представления».
12. Проверим работоспособность созданного запроса вне конструктора запросов. Запустим вновь созданный запрос без использования конструктора запросов. Для выполнения уже сохраненного запроса необходимо щелкнуть ПКМ по запросу и в появившемся меню

выбрать пункт «Отобразить первые 1000 записей». Выполните эту операцию для созданного запроса. Результат представлен ниже.

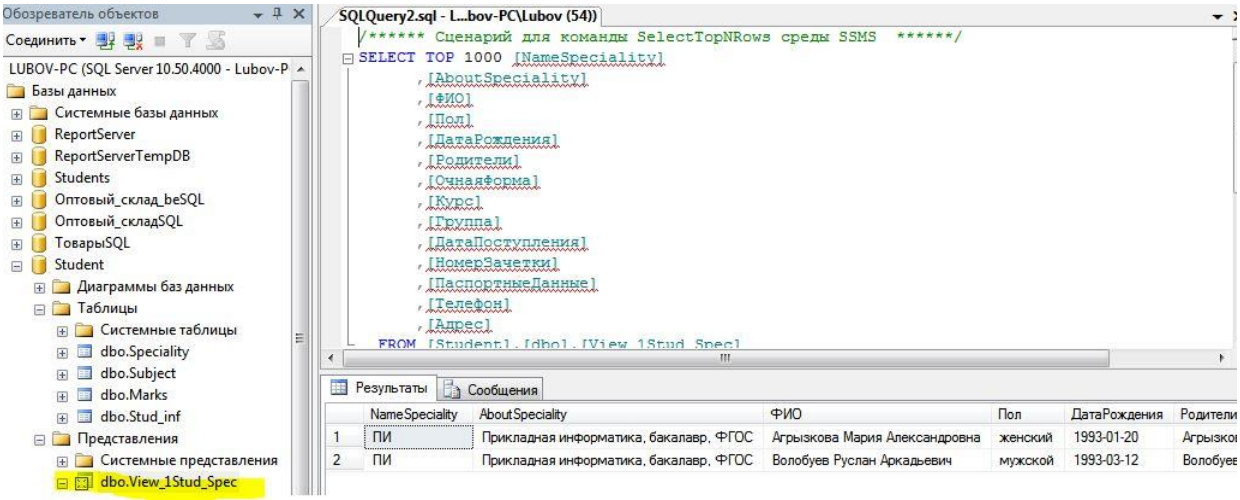


Рисунок 29

13. Создайте самостоятельно следующие запросы:

    - «Запрос Студенты+Оценки» выводящий успеваемость студентов по дате;
    - «Запрос Студенты+Средний Балл», выводящий средний балл каждого студента;
    - «Запрос Оценки+Предметы», выводящий средний балл по каждому из предметов.

14. При самостоятельном создании запросов, выводящих среднее значение, воспользуйтесь агрегатной функцией AVG.
15. Проверьте работоспособность созданных запросов.
16. На основе «Запрос Студенты+Специальности» создадим фильтры, отображающие студентов отдельных специальностей.
17. Так как он будет основан на запросе "Запрос Студенты+Специальности", то в окне "Добавить таблицу" перейдите на вкладку "Представления" и добавьте в новый запрос "Запрос Студенты+Специальности". Затем закройте окно "Добавить таблицу".

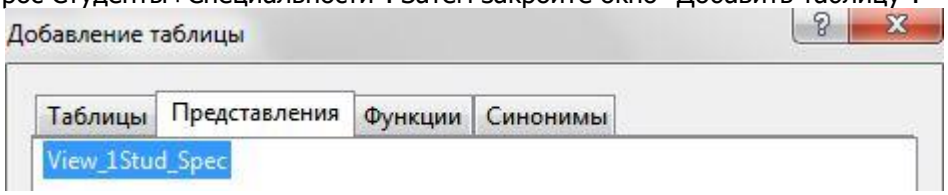


Рисунок 30

18. В появившемся окне конструктора запросов определите в качестве отображаемых полей все поля запроса "Запрос Студенты+Специальности". Для отображения всех полей запроса, в данном случае, мы не можем использовать пункт "* (All Columns)" (Все поля). Так как в этом случае мы не можем устанавливать критерий отбора записей в фильтре, а также невозможно установить сортировку записей.
19. Теперь установим критерий отбора записей в фильтре. Пусть наш фильтр отображает только студентов имеющих специальность "ММ". Для определения условия отбора записей в таблице отображаемых полей в строке, соответствующей полю, на которое накладывается условие, в столбце "Filter", необходимо задать условие. В нашем случае условие накладывается на поле "Наименование специальности". Следовательно, в строке "Наименование специальности", в столбце "Filter" нужно задать следующее условие отбора "='ПИ'".
20. В заключение настроим сортировку записей в фильтре. Пусть при выполнении фильтра сначала происходит сортировка записей по возрастанию по полю "Очная форма обучения", а затем по убыванию по полю "Курс". Для установки сортировки записей по возрастанию, в таблице определяемых полей, в строке для поля "Очная форма обучения", в столбце (Тип сортировки), задайте "(По возрастанию).

21. В строке для поля "Курс" - задайте (По убыванию).
22. Для определения порядка сортировки для поля "Очная форма обучения" в столбце "Sort Order" (Порядок сортировки) поставьте 1, а для поля "Курс" поставьте 2. то есть, при выполнении запроса записи сначала сортируются по полю "Очная форма обучения", а затем по полю "Курс".
23. После установки сортировки записей в фильтре проверим его работоспособность, выполнив его.
24. Результат выполнения фильтра должен выглядеть как на рисунке 31.

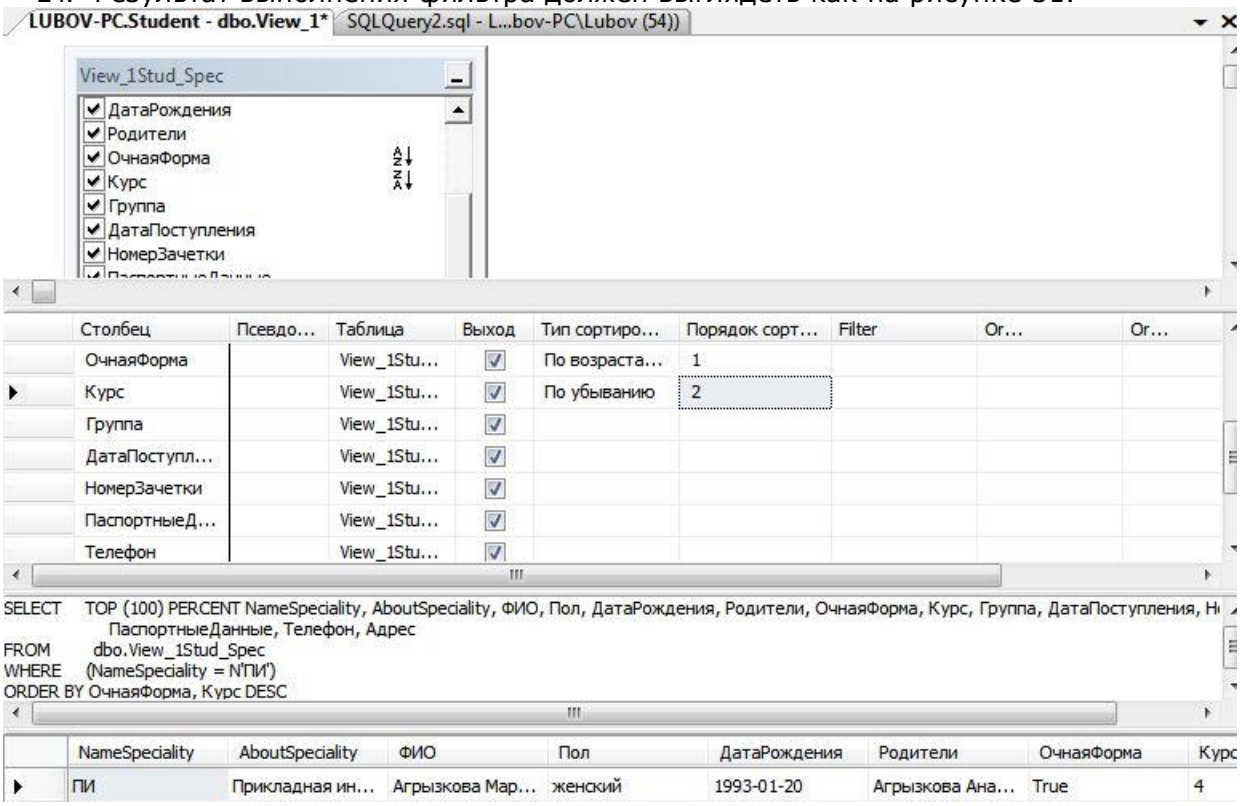


Рисунок 31

25. Закройте окно конструктора запросов. В качестве имени нового фильтра задайте **«Фильтр ПИ»** и нажмите «**Ok».**
26. Самостоятельно создайте фильтры для отображения других специальностей. Данные фильтры создаются аналогично фильтру "Фильтр ПИ" (смотри выше). Единственным отличием является условие отбора, накладываемое на поле "Наименование специальности", оно должно быть не "='ММ'", а, например, "='ПИ'", "='СТ'", "='МО'" или "='БУ'".
27. При сохранении фильтров задаем их имена соответственно их условиям отбора, то есть "Фильтр ПИ", "Фильтр СТ", "Фильтр МО" или "Фильтр БУ".
28. Проверьте созданные фильтры на работоспособность.
29. Создайте фильтры «Фильтр очная форма обучения», «Дата поступления 2013 год», «Фильтр старшекурсники(3,4,5курсы)».
30. Проверьте работоспособность созданных фильтров.

Вопросы

1. Назовите виды связей между таблицами
2. Какие виды запросов вы знаете
3. Что такое «хранимая процедура»
4. Дайте определение виду связи «Один-ко-многим»
5. Напишите полный синтаксис SQL-запроса
6. Как отсортировать поле по возрастанию
7. Опишите алгоритм создания фильтра на основе запроса-представления

ПРАКТИЧЕСКАЯ РАБОТА №9
«СОЗДАНИЕ ДИАГРАММ»

Цель

- Изучить создание диаграмм

Общая информация

Для создания новых диаграмм баз данных можно использовать обозреватель объектов. Диаграммы базы данных графически изображают структуру баз данных. При помощи диаграмм баз данных можно создавать и изменять таблицы, столбцы, связи и ключи. Кроме того, можно изменять индексы и ограничения. Для любой базы данных можно создать любое необходимое количество диаграмм; каждая из таблиц базы данных может использоваться в любом количестве диаграмм. Таким образом, для визуализации различных частей базы данных или для акцентирования различных аспектов ее конструирования можно создавать различные диаграммы. Например, можно создать большую диаграмму, в которой будут отображаться все таблицы и столбцы, а также меньшую диаграмму, в которой будут отображаться все таблицы, но не будет столбцов.

Внутри диаграммы базы данных каждая таблица имеет три отдельных элемента: строка заголовка, список выбора строк и набор столбцов свойств.

Строка заголовка В строке заголовка отображается имя таблицы
Если таблица была изменена, но еще не сохранена, то после имени таблицы появляется звездочка (*), показывающая наличие несохраненных изменений. Дополнительные сведения о сохранении измененных таблиц и диаграмм см. в разделе Работа с диаграммами баз данных (визуальные инструменты для баз данных)
Список выбора строк Чтобы выбрать столбец базы данных в таблице, щелкните список выбора строк. Если столбец является первичным ключом таблицы, то в этом списке отображается символ ключа. Дополнительные сведения о первичных ключах см. в разделе Ограничения первичных и внешних ключей.
Столбцы свойств Набор столбцов свойств виден не во всех представлениях таблицы. Таблицу можно просмотреть в любом из пяти различных представлений, позволяющих подобрать подходящий размер и размещение элементов диаграммы.

Внутри диаграммы базы данных у каждой связи есть три отдельных элемента: конечные точки, стиль линии и связанные таблицы.

Конечные точки Конечные точки линии показывают вид связи: «один к одному» или «один ко многим». Если на одной конечной точке связи находится ключ, а на другой — цифра восемь, то это связь «один ко многим». Если у связи по одному ключу на каждой конечной точке, то это связь «один к одному».
Стиль линии Разновидность линии (не ее конечные точки) показывает, проверяет ли СУБД ссылочную целостность для связи при добавлении новых данных в таблицу, связанную с помощью внешнего ключа. Если связь нарисована в виде сплошной линии, это значит, что СУБД проверяет ссылочную целостность для связи при добавлении или изменении строк в таблице, связанной с помощью внешнего ключа. Если линия пунктирная, это значит, что СУБД не проверяет ссылочную целостность для связи при добавлении или изменении строк в таблице, связанной с помощью внешнего ключа.
Связанные таблицы Линия связи показывает, что две таблицы связаны с помощью внешнего ключа. Для связи «один ко многим» таблица, связанная с помощью внешнего ключа, — это таблица около цифры 8 на линии. Если обе конечные точки линии присоединены к одной таблице, это означает возвратную связь.

Если в диаграмме содержатся удаленные из базы данных таблицы и столбцы, то при сохранении диаграммы в базе данных будут повторно созданы только их определения. При этом процессе не восстанавливаются данные, которые существовали в этих объектах до их удаления.

Ход выполнения лабораторной работы

1. В рабочей БД выбрать «Диаграммы БД» - «Создать диаграмму БД» -выберете все таблицы

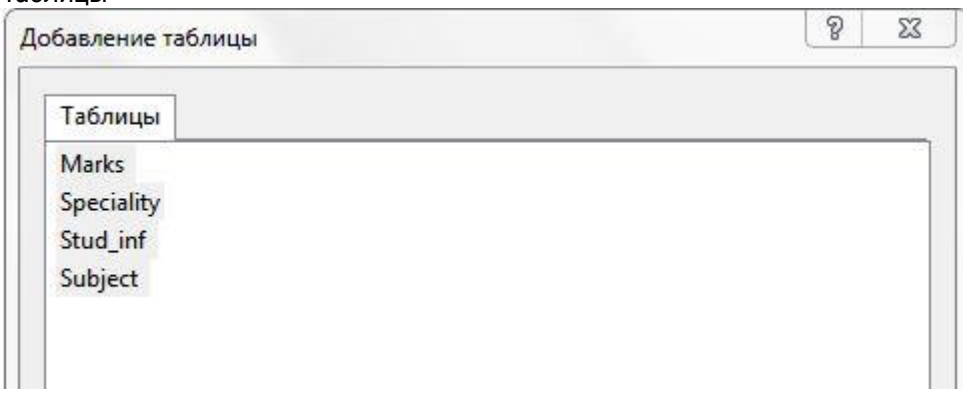


Рисунок 32

2. В появившемся окне удостоверьтесь в наличии всех таблиц:

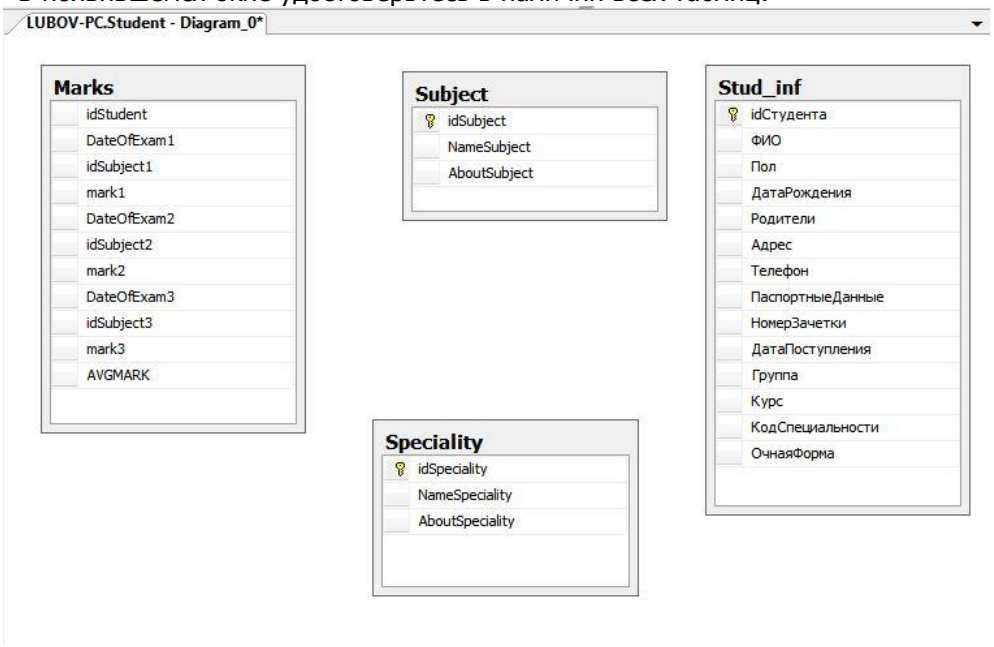


Рисунок 33

3. Теперь необходимо определить связи между таблицами. Перетащите поле "Код специальности" из таблицы "Специальности" на такое же поле в таблице "Студенты". Появится окно создания связи между таблицами "Tables and Columns". Согласитесь с окном, представленным на рисунке 34.

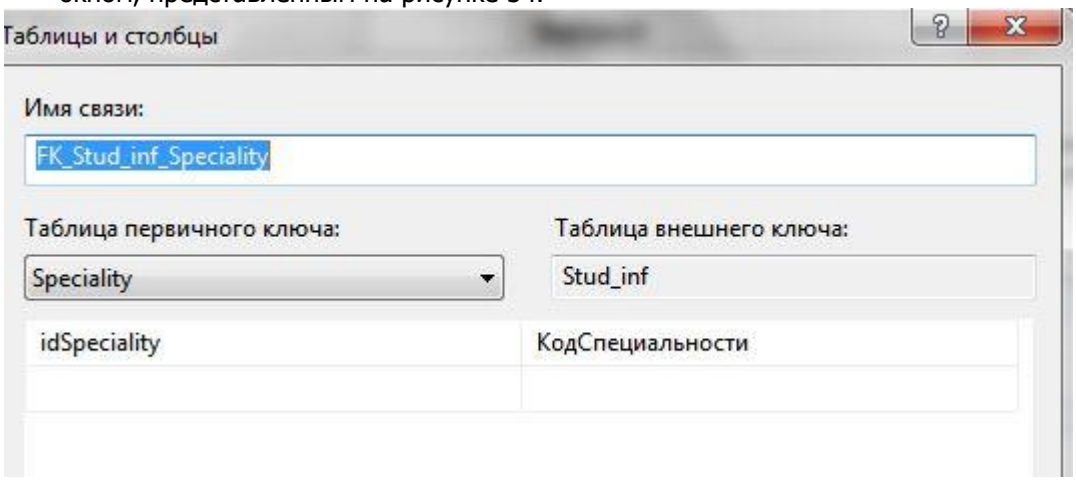


Рисунок 34

4. Появится окно настройки свойств связи "Foreign Key Relationship". Оставьте свойства связи без изменений и в окне свойств связи нажмите кнопку "Ok".
5. В диаграмме между таблицами "Студенты" и "Специальности" появится связь в виде ломанной линии.
6. Аналогичным образом создайте связь таблицы "Студенты" с таблицей "Оценки", перетащив поле "Код студента" из таблицы "Студенты" на одноименное поле в таблице "Оценки".
7. Затем, свяжите таблицы "Предметы" и "Оценки", перетащив поле "Код предмета" из таблицы "Предметы" на поля "Код предмета 1", "Код предмета 2" и "Код предмета 3" таблицы "Оценки".
8. После приведенных выше действий диаграмма примет вид:

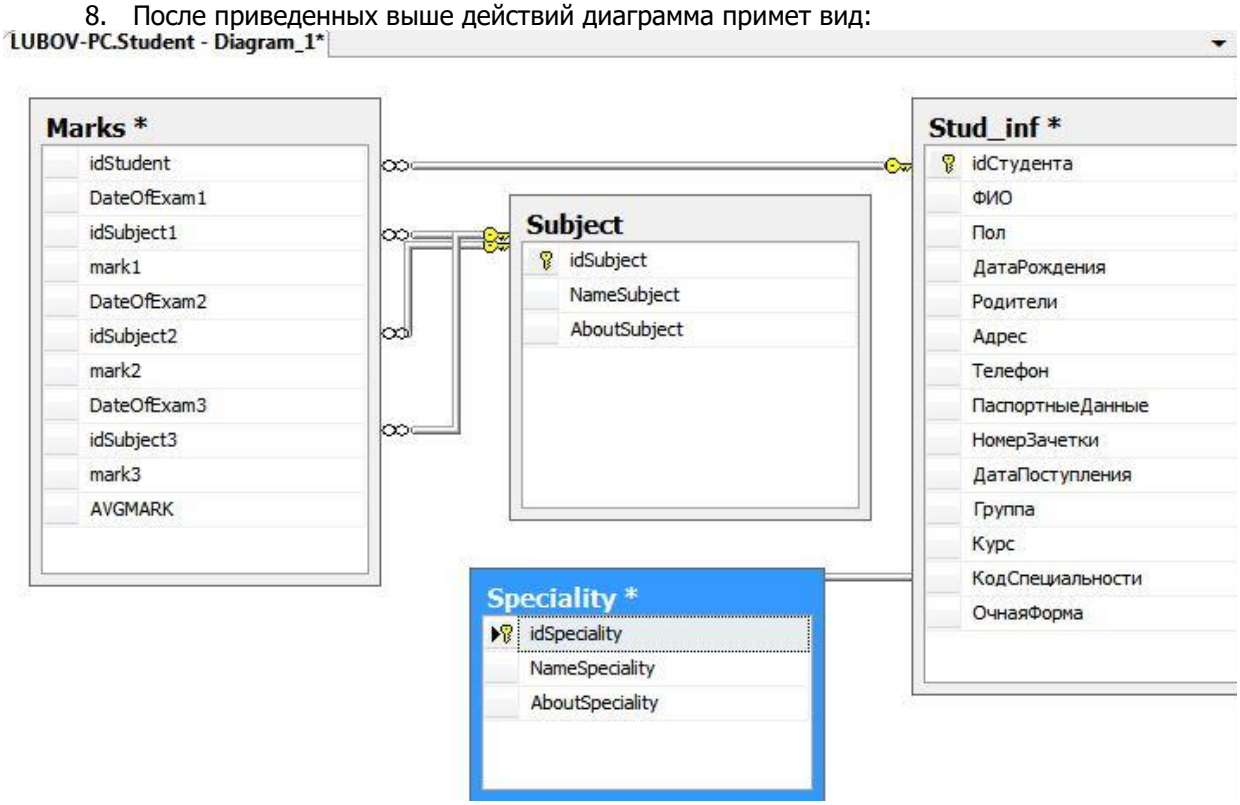


Рисунок 35

9. Закройте окно с диаграммой
10. Появится окно определения имени новой диаграммы. В окне определения имени, задайте имя диаграммы как "Diagram_1" и нажмите кнопку "Ok".
11. Появится окно с запросом сохранения таблиц, входящих в диаграмму. В данном окне необходимо нажать кнопку "Yes".

Вопросы

1. Что используется для создания диаграмм
2. Какие элементы имеет каждая таблица внутри диаграммы
3. Как добавить таблицу в диаграмму
4. Как добавить или изменить связи между таблицами
5. Какой объект Microcosft Officce Access напоминают вам диаграммы
6. В каких представлениях можно просмотреть таблицу в диаграмме
7. Какие типы связей вы использовали при создании диаграммы и почему