Министерство науки и высшего образования
Российской Федерации
Братский педагогический колледж
федерального государственного бюджетного
образовательного учреждения высшего образования
«Братский государственный университет»

МДК 11.01.ТЕХНОЛОГИЯ РАЗРАБОТКИ И ЗАЩИТЫ БАЗ ДАННЫХ

методические рекомендации
по выполнению лабораторных занятий

для студентов 3 курса
специальности
09.02.07 Информационные системы и программирование

Автор: Л.Д. Разумова

Братск, 2020

МДК 11.01. Технология разработки и защиты баз данных. Методические рекомендации по выполнению лабораторных занятий. / Сост. Л.Д. Разумова.- Братск, 2020.- 58 с.

Методические рекомендации предназначены для студентов третьего курса специальности 09.02.07 Информационные системы и программирование при изучении дисциплины «Технология и защиты баз данных»

Печатается по решению научно-методического совета
Братского педагогического колледжа ФГБОУ ВО «БрГУ»
665709, г. Братск, ул. Макаренко 40

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Лабораторные занятия являются одним из основных видов учебных занятий, направленных на экспериментальное подтверждение теоретических положений и формирование учебных и профессиональных практических умений. Они составляют важную часть теоретической и профессиональной практической подготовки.

Лабораторные занятия направлены на экспериментальное подтверждение теоретических положений и формирование учебных и профессионально-значимых умений обучающихся.

Выполнение лабораторных занятий проводится с целью:

– формирования практических умений в соответствии с требованиями к уровню подготовки студентов, установленными рабочей программой дисциплины по конкретным разделам (темам);

– обобщение, систематизацию, углубление, закрепление полученных теоретических знаний;

– совершенствование умений применять полученные знания на практике, реализацию единства интеллектуальной и практической деятельности;

– развитие интеллектуальных умений у будущих специалистов: аналитических, проектировочных, конструктивных и др.;

– выработку при решении поставленных задач таких профессионально значимых качеств, как самостоятельность, ответственность, точность, творческая инициатива.

В части освоения основного вида профессиональной деятельности (ВПД): «Разработка и администрирование баз данных» и соответствующих профессиональных компетенций (ПК):

2.1. Разрабатывать объекты базы данных.

2.2. Реализовывать базу данных в конкретной системе управления базами данных (далее - СУБД).

2.3. Решать вопросы администрирования базы данных.

2.4. Реализовывать методы и технологии защиты информации в базах данных.

С целью овладения указанным видом профессиональной деятельности и соответствующими профессиональными компетенциями обучающийся в ходе освоения профессионального модуля должен:

иметь практический опыт:

– работы с объектами базы данных в конкретной системе управления базами данных;

– использования средств заполнения базы данных;

– использования стандартных методов защиты объектов базы данных;

уметь:

– создавать объекты баз данных в современных СУБД и управлять доступом к этим объектам;

– работать с современными case-средствами проектирования баз данных;

– формировать и настраивать схему базы данных;

– разрабатывать прикладные программы с использованием языка SQL;

– создавать хранимые процедуры и триггеры на базах данных;

– применять стандартные методы для защиты объектов базы данных;

знать:

– основные положения теории баз данных, хранилищ данных, баз знаний;

– основные принципы построения концептуальной, логической и физической модели данных;

– современные инструментальные средства разработки схемы базы данных;
– методы описания схем баз данных в современных СУБД;
– структуры данных СУБД, общий подход к организации представлений, таблиц, индексов и кластеров;
– методы организации целостности данных;
– способы контроля доступа к данным и управления привилегиями;
– основные методы и средства защиты данных в базах данных;
– модели и структуры информационных систем;
– основные типы сетевых топологий, приемы работы в компьютерных сетях;
– информационные ресурсы компьютерных сетей;
– технологии передачи и обмена данными в компьютерных сетях;
– основы разработки приложений баз данных.

Общие указания по выполнению работ на лабораторных занятиях

О проведении лабораторного занятия студентам сообщается заблаговременно: когда предстоит практическое занятие, какие вопросы нужно повторить, чтобы ее выполнить. Просматриваются задания, оговаривается объем и время выполнения. Критерии оценки сообщаются перед выполнением каждого лабораторного занятия.

Студенты получают распечатанный или электронный вариант задания с описанием этапов выполнения работы.

При выполнении лабораторного занятия студент придерживается следующего алгоритма:

1. Записать в тетради дату, тему и цель занятия.
2. Ознакомиться с правилами и условиями выполнения практического задания.
3. Повторить теоретические задания, необходимые для рациональной работы и других практических действий.
4. Выполнить работу по предложенному алгоритму действий.
5. Обобщить результаты работы, сформулировать выводы по работе.
6. Записать в тетрадь ответы на контрольные вопросы.

Задания студентами сдаются в электронном виде. Все выполненные работы нумеруются, в соответствии с номером практического задания, и сохраняются в отдельной папке.

ЛАБОРАТОРНОЕ ЗАНЯТИЕ № 1

РАБОТА С БАЗАМИ ДАННЫХ .ТЕХНОЛОГИЯ ADO.NET. СОЗДАНИЕ ФИЗИЧЕСКОЙ МОДЕЛИ БД СРЕДСТВАМИ СУБД MICROSOFT ACCESS

Цель -обеспечить студентов конкретным инструментарием для практической реализации методов проектирования приложений клиент-сервер.

Физическая модель данных зависит от выбранной СУБД.

В данном цикле лабораторных работ будут рассмотрено создание физической модели БД средствами СУБД Microsoft Access и сервера баз данных Microsoft SQL Server

Создание БД в СУБД Microsoft Access

MS Access является частью пакета MS Office и представляет собой СУБД. Запустите MS Access, используя команды меню «Пуск» -> «Программы» -> «Microsoft Office» -> «Access». Перед вами откроется основное окно программы. Выполните пункты меню «Файл» - «Создать», после чего из вариантов выберите «Новая база данных» и сохраните вашу БД на диск.

Открывшееся окно представляет собой новую БД.

Объекты БД в среде MS Access

MS Access поддерживает следующие объекты:

• Таблица – отношение в реляционной терминологии;

• Запрос – сохраненный текст (модель) запроса на языке SQL к таблицам БД;

• Формы – конструирование интерфейса пользователя;

• Отчеты – конструирование выходной информации БД;

• Страницы – страницы доступа через web;

• Макросы – группы макрокоманд;

• Модули – программные модули на языке VBA, которые могут быть использованы в запросах, формах, отчетах, макросах.

Создание таблиц в среде MS Access

СУБД MS Access поддерживает различные варианты создания таблиц. Рассмотрим наиболее часто используемый вариант создания – в режиме конструктора. Для этого среди объектов БД выберите «Таблицы», после чего выполните пункт «Создание таблицы в режиме конструктора».

В появившемся окне предлагается ввести имена атрибутов таблицы с указанием их типов. Типы атрибутов выбираются из выпадающего списка вариантов, а конкретные характеристики указываются на панели внизу (например, для текстового формата можно указать максимальный размер). Третий столбец на форме конструирования отношений – описание атрибутов. При создании сложных БД с большим числом отношений и атрибутов в них обязательно описывайте логику, которую вы закладываете в тот или иной атрибут.

После создания всех атрибутов не забудьте указать ключевое поле вашего отношения. Для этого выделите нужные поля и нажмите на иконку «ключевое поле» изображающую ключ на панели конструктора.

Создание схем данных в среде MS Access

СУБД MS Access реализует инструмент проектирования связей между таблицами (объектами) БД. Этот инструмент называется «Схема данных».

Для вызова этого инструмента можете воспользоваться соответствующей кнопкой на панели «База данных» (рис. 1

Рисунок 1 – Панель «База данных»

В рабочую область схемы можно добавлять и удалять таблицы, а также связи между ними. Для добавления связи между таблицами выберите необходимое поле (внешний ключ) и использую Drag'n'Drop протяните его до нужного поля связанной таблицы (потенциальный ключ). После чего, в появившемся окне свойств вновь созданной связи, можете выставить необходимые настройки (рис.2).

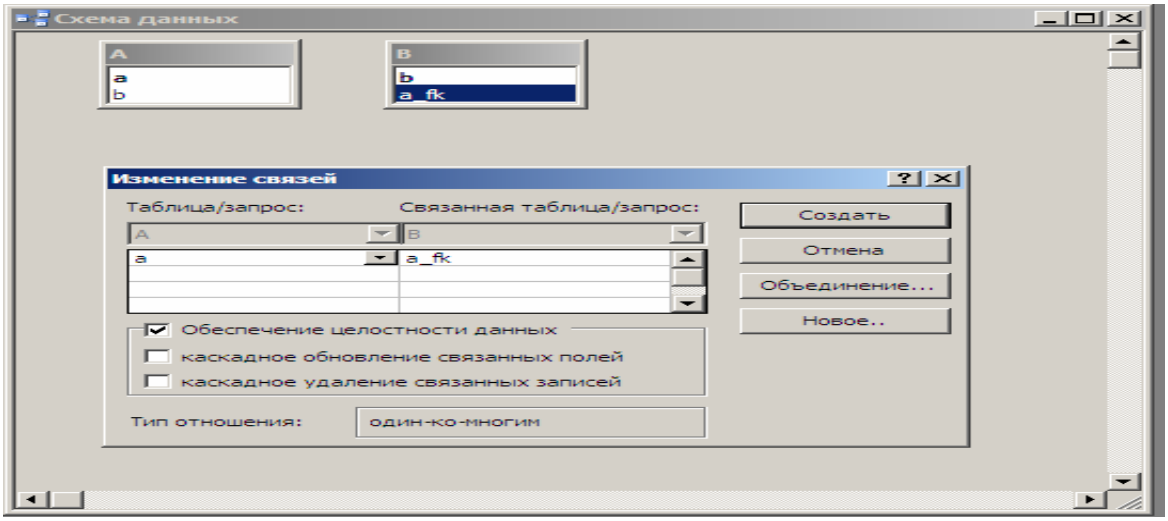


Рисунок 2 Редактирование связей в MS Access

После окончания редактирования не забудьте сохранить схему.

Задание на лабораторную работу

1. Создать новую базу данных и сохранить в личном каталоге на своем флэш-накопителе. (**ВАМ работать с этой базой до конца семестра).**
2. В БД определить три новые таблицы со структурой полей согласно номеру варианта задания.
3. Определить требуемые индексы для таблиц БД (первичный ключ и вторичные ключи). Установить связи между таблицами БД.
4. Внести в таблицы БД по 10-15 записей, содержание которых соответствует семантике и типу полей.

ЛАБОРАТОРНОЕ ЗАНЯТИЕ № 2

РАБОТА С БАЗАМИ ДАННЫХ .ТЕХНОЛОГИЯ ADO.NET. СОЗДАНИЕ БАЗЫ ДАННЫХ В СРЕДЕ MICROSOFT SQL SERVER

Цель - обеспечить студентов конкретным инструментарием для практической реализации методов проектирования приложений клиент-сервер.

Физическая модель данных зависит от выбранной СУБД.

В данном цикле лабораторных работ будут рассмотрено создание физической модели БД средствами СУБД Microsoft Access и сервера баз данных Microsoft SQL Server

Задание на лабораторную работу.

1. Изучить пример создания локальной базы данных Microsoft SQL Server в MS Visual Studio.(раздел 1)

Создать базу данных Microsoft SQL Server в MS Visual Studio в соответствии с полученным вариантом и сохранить в личном каталоге на своем флэш-накопителе. (**ВАМ работать с этой базой до конца семестра).** При создании база данных воспользоваться таблицей 1 «Соответствие типов данных Microsoft Access и Microsoft SQL». Как сохранить базу данных, смотри в разделе «**Перенос файла БД Microsoft SQL на другой компьютер»**

2. В БД определить три новые таблицы со структурой полей согласно номеру варианта задания.

3. Определить требуемые индексы для таблиц БД (первичный ключ и вторичные ключи).. Установить связи между таблицами БД.

4. Внести в таблицы БД по 10-15 записей, содержание которых соответствует семантике и типу полей.

Таблица 1. Соответствие типов данных Microsoft Access и Microsoft SQL

№	Тип данных Microsoft Access	Тип данных Microsoft SQL	Описание типа данных Microsoft SQL
1	Текстовый	nvarchar	Тип данных для хранения текста до 4000 символов
2	Поле MEMO	ntext	Тип данных для хранения символов в кодировке Unicode до 1 073 741 823 символов
3	Числовой	int	Численные значения (целые) в диапазоне от -2 147 483 648 до +2 147 483 647
4	Дата/время	smalldatetime	Дата и время от 1 января 1900 г. до 6 июня 2079 года с точностью до одной минуты
5	Денежный	money	Денежный тип данных, значения которого лежат в диапазоне от -922 337 203 685 477.5808 до +922 337 203 685 477.5807, с точностью до одной десятитысячной
6	Счетчик	int	См. пункт 3
7	Логический	bit	Переменная, способная принимать только два значения - 0 или 1
8	Поле объекта OLE	image	Переменная для хранения массива байтов от 0 до 2 147 483 647 байт
9	Гиперссылка	ntext	См. пункт 2
10	Мастер подстановок	nvarchar	См. пункт 1

Раздел 1. Пример создания базы данных Microsoft SQL Server в MS Visual Studio

В данной теме показано решение задачи создания базы данных типа SQL Server с помощью MS Visual Studio. Рассматриваются следующие вопросы:

- работа с окном Server Explorer в MS Visual Studio;
- создание локальной базы данных типа SQL Server Database;
- создание таблиц в базе данных;
- редактирование структур таблиц;
- связывание таблиц базы данных между собой;
- внесение данных в таблицы средствами MS Visual Studio.

Содержание

Условие задачи

Используя средства MS Visual Studio создать базу данных типа MS SQL Server с именем Education. База данных содержит две таблицы Student и Session. Таблицы между собой связаны по некоторыму полю.

Структура первой таблицы «Student».

Название поля	Тип поля	Объяснение
Num_book	Текстовый	Номер зачетной книжки
Name	Текстовый	Фамилия, имя, отчество студента
Group	Текстовый	Название группы, в которой учится студент
Year	Целое число	Год рождения

Структура второй таблицы “Session”.

Название поля	Тип поля	Объяснение
Num_book	Текстовый	Номер зачетной книжки
Mathematics	Целое число	Оценка по математике
Informatics	Целое число	Оценка из информатики
Philosophy	Целое число	Оценка из философии

Выполнение

1. Загрузить MS Visual Studio.

2. Активировать окно Server Explorer.(обозреватель серверов)

Для работы с базами данных корпорация Microsoft предлагает облегченный сервер баз данных Microsoft SQL Server.

Прежде всего, перед созданием базы данных, нужно активировать утилиту Server Explorer. Для этого, в MS Visual Studio нужно вызвать (рис. 1)

View -> Server Explorer

(в некоторых версиях –обозреватель серверов находится в вид-другие объекты)

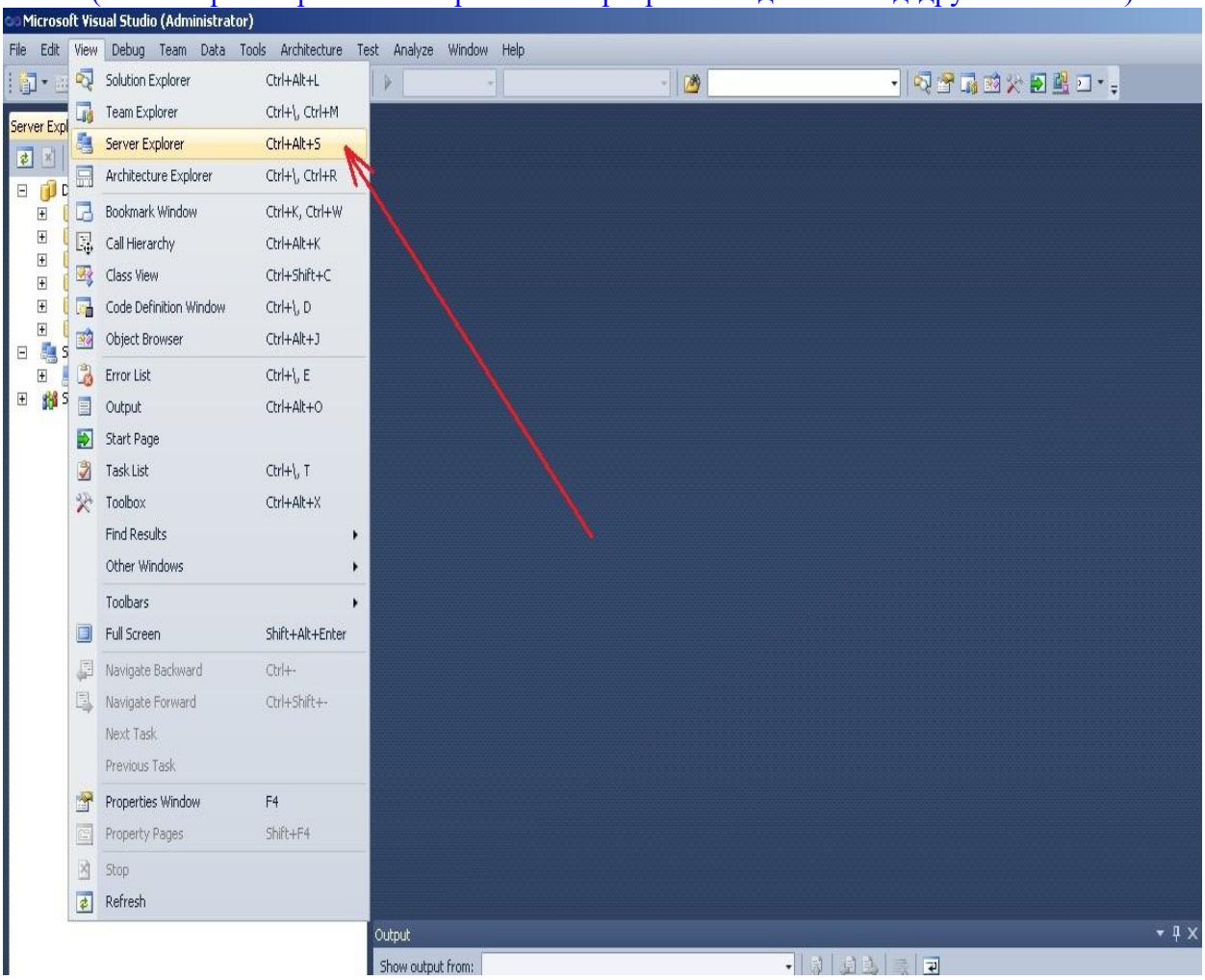


Рис. 1. Вызов Server Explorer

После вызова окно Server Explorer будет иметь приблизительный вид, как показано на рисунке 2.

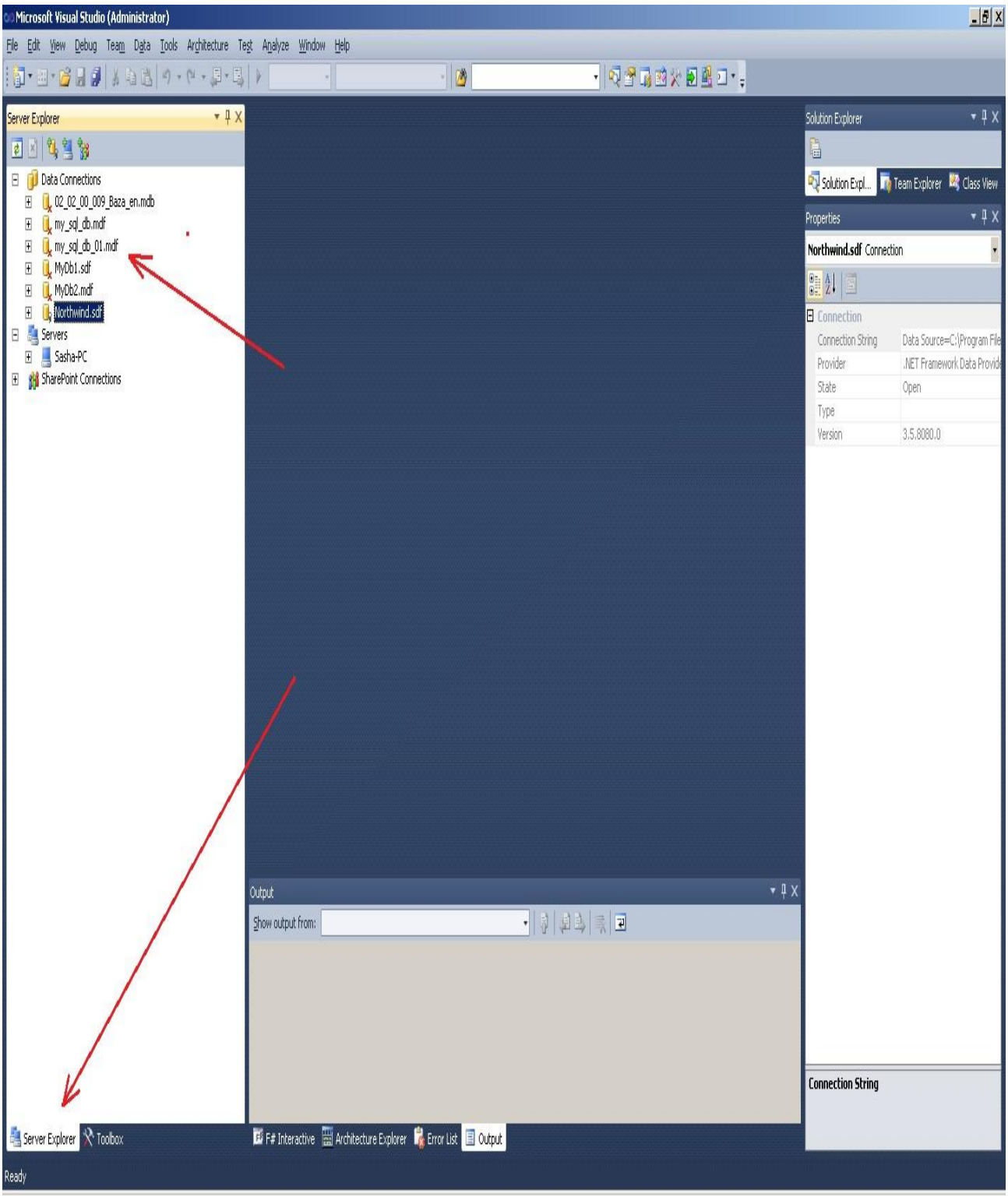


Рис. 2. Окно Server Explorer

3. Создание базы данных “Education”.

Чтобы создать новую базу данных, базирующуюся на поставщике данных Microsoft SQL Server, нужно кликнуть на узле Data Connections, а потом выбрать “Create New SQL Server Database …” (рис. 3).

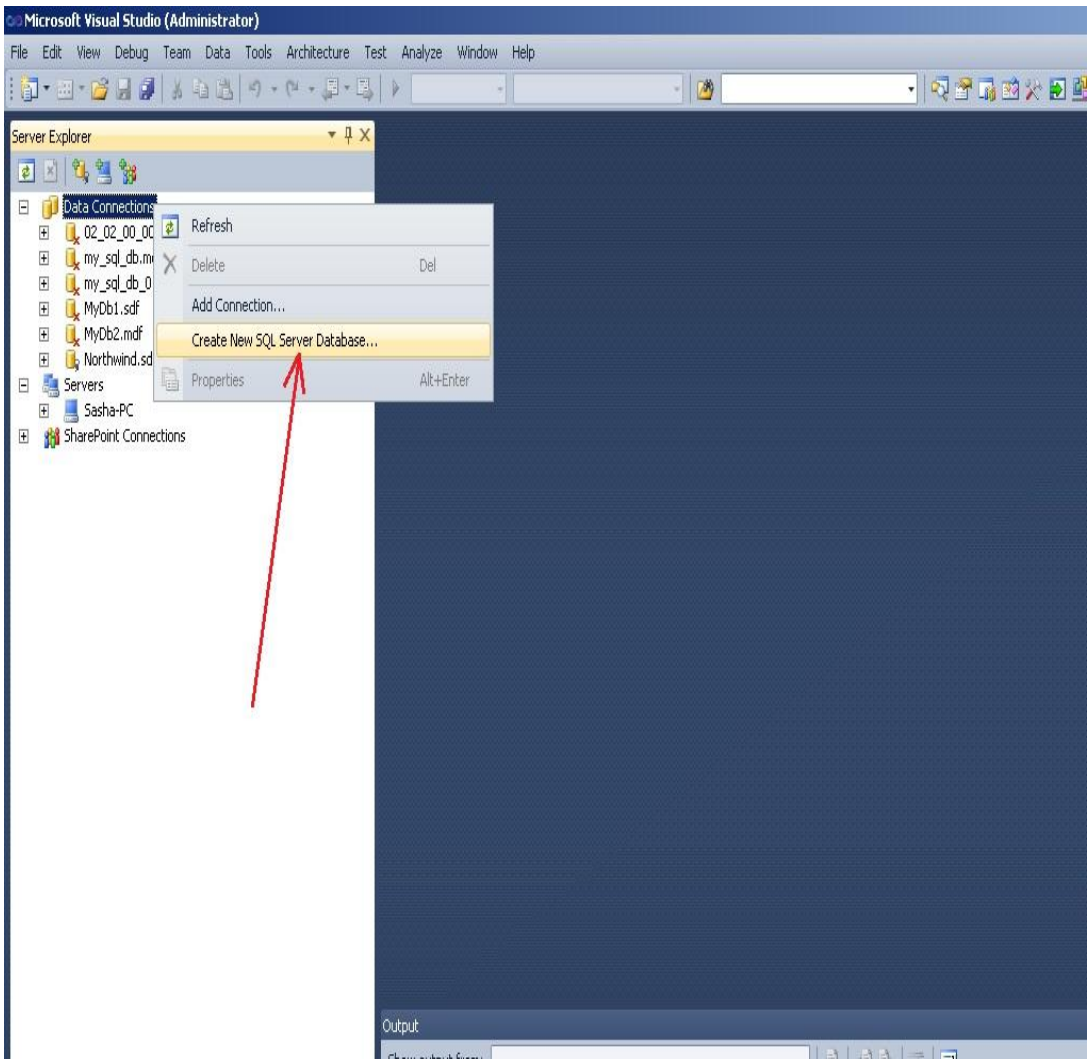


Рис. 3. Вызов команды создания базы данных SQL Server

В результате откроется окно «Create New SQL Server Database» (рис. 4).

В окне (в поле «Server Name») указывается имя локального сервера, установленного на вашем компьютере. В нашем случае это имя “SQLEXPRESS”.

В поле «New database name:» указывается имя создаваемой базы данных. В нашем случае это имя Education.

Опцию Use Windows Autentification нужно оставить без изменений и нажать кнопку OK.

Рис. 4. Создание новой базы данных SQL Server с помощью MS Visual Studio

После выполненных действий, окно Server Explorer примет вид, как показано на рисунке 5. Как видно из рисунка 5, в список имеющихся баз данных добавлена база данных Education с именем

sasha-pc\sqlexpress.Education.dbo

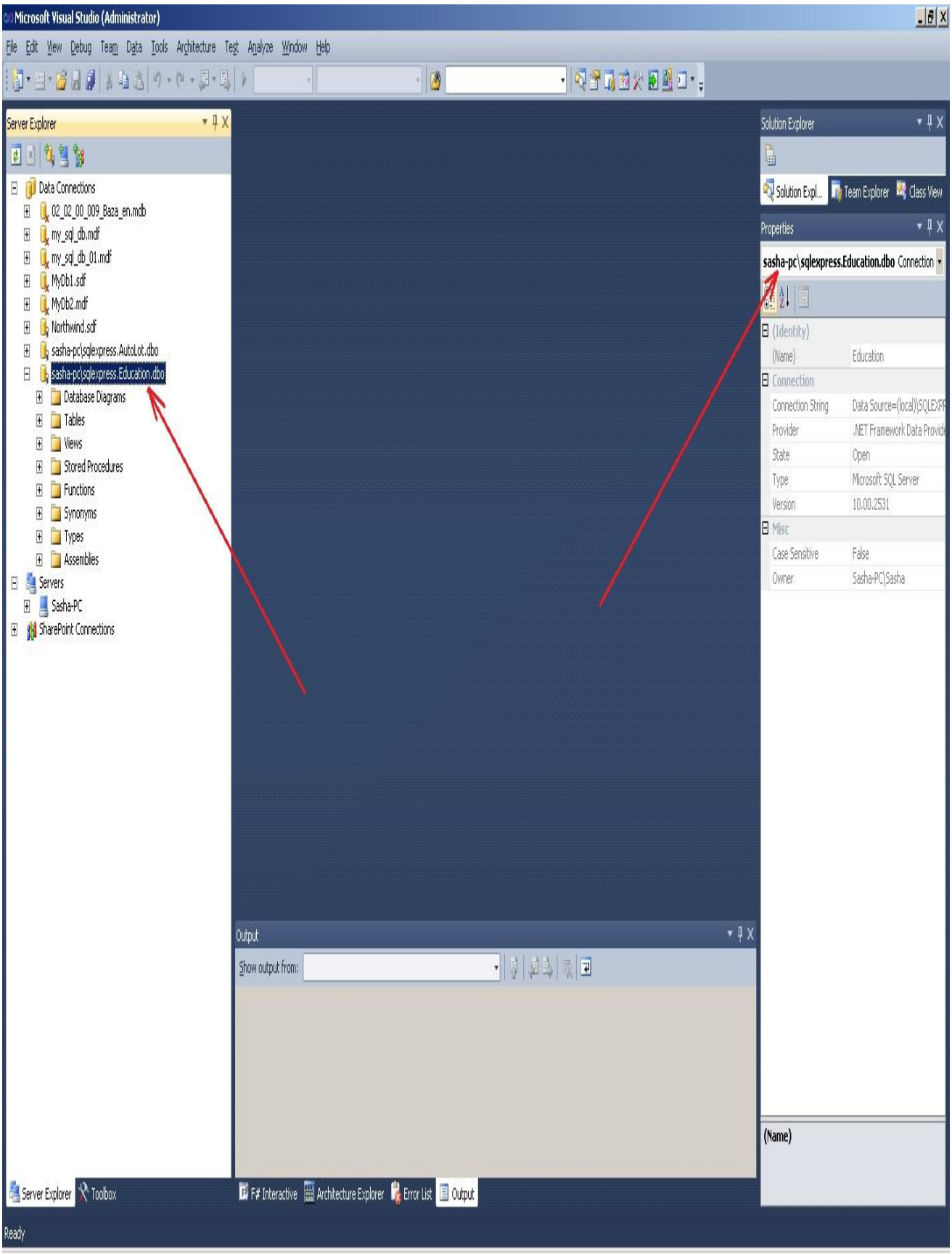


Рис. 5. Окно Server Explorer после добавления базы данных Education

4. Объекты базы данных Education.

Если развернуть базу данных Education (знак «+»), то можно увидеть список из следующих основных объектов:

– Database Diagrams – диаграммы базы данных. Диаграммы показывают связи между таблицами базы данных, отношения между полями разных таблиц и т.п.;

– Tables – таблицы, в которых помещаются данные базы данных;

– Views – представления. Отличие между представлениями и таблицами состоит в том, что таблицы баз данных содержат данные, а представления данных не содержат их, а содержимое выбирается из других таблиц или представлений;

– Stored procedures – хранимые процедуры. Они представляют собою группу связанных операторов на языке SQL, что обеспечивает дополнительную гибкость при работе с базой данных.

5. Создание таблицы Student.

На данный момент база данных Education абсолютно пустая и не содержит никаких объектов (таблиц, сохраненных процедур, представлений и т.д.).

Чтобы создать таблицу, нужно вызвать контекстное меню (клик правой кнопкой мышки) и выбрать команду "Add New Table" (рисунок 6).

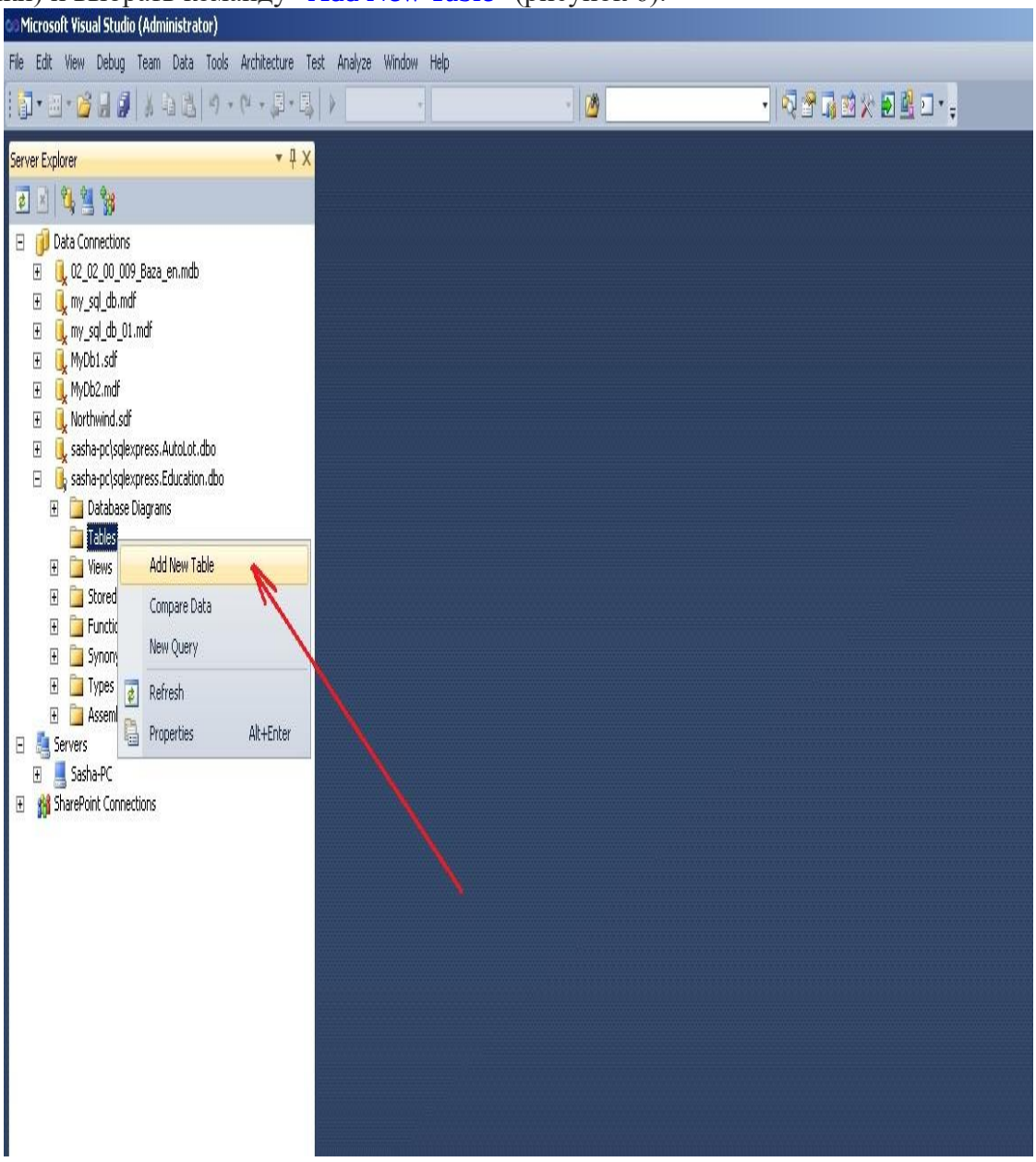


Рис. 6. Команда добавления новой таблицы

Существует и другой вариант добавления таблицы базы данных с помощью команд меню Data:

Data -> Add New -> Table

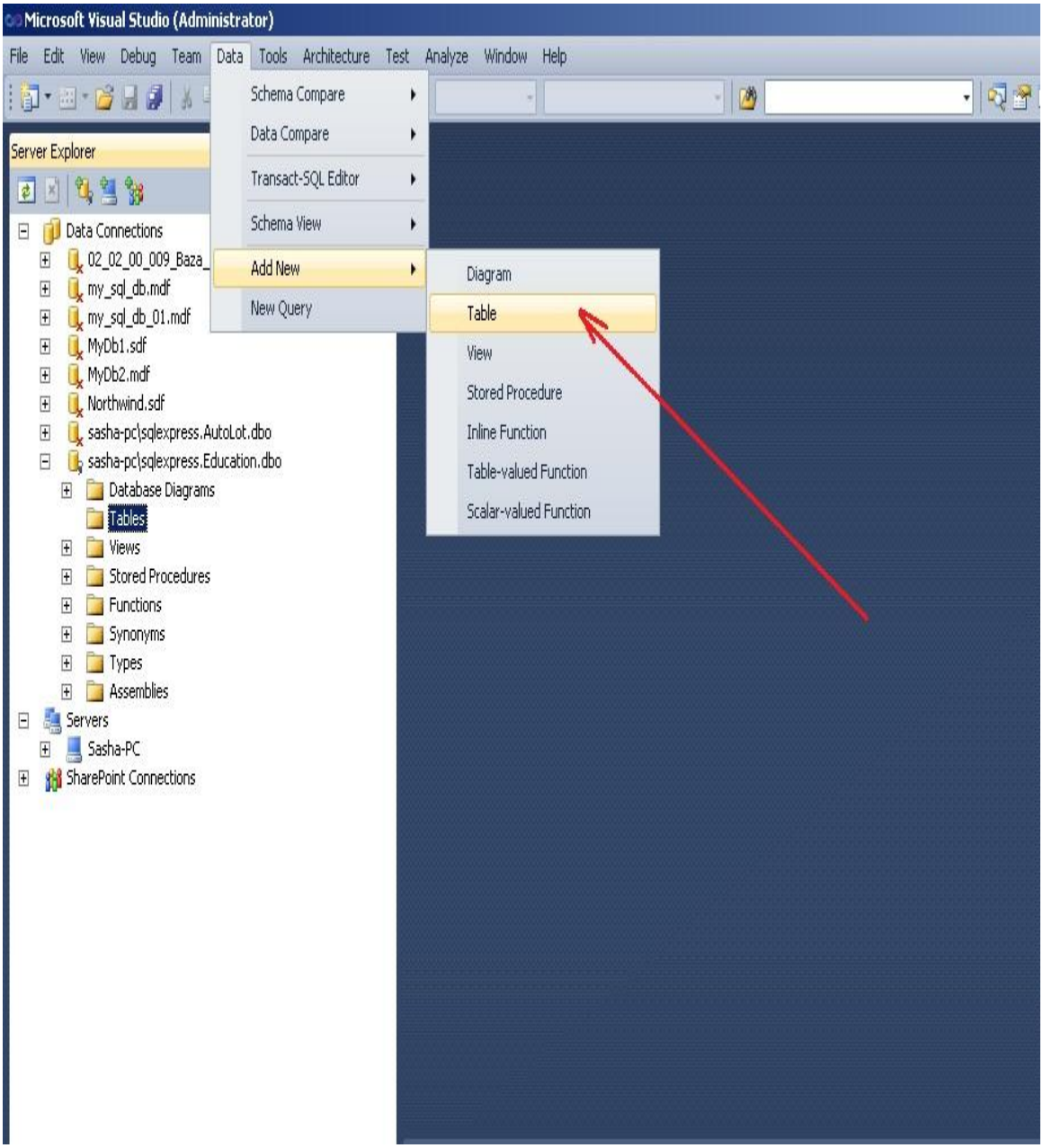


Рис. 7. Альтернативный вариант добавления новой таблицы

В результате откроется окно добавления таблицы, которое содержит три столбца (рисунок 8). В первом столбце “Column Name” нужно ввести название соответствующего поля таблицы базы данных. Во втором столбце “Data Type” нужно ввести тип данных этого поля. В третьем столбце “ Allow Nulls”указывается опция о возможности отсутствия данных в поле.

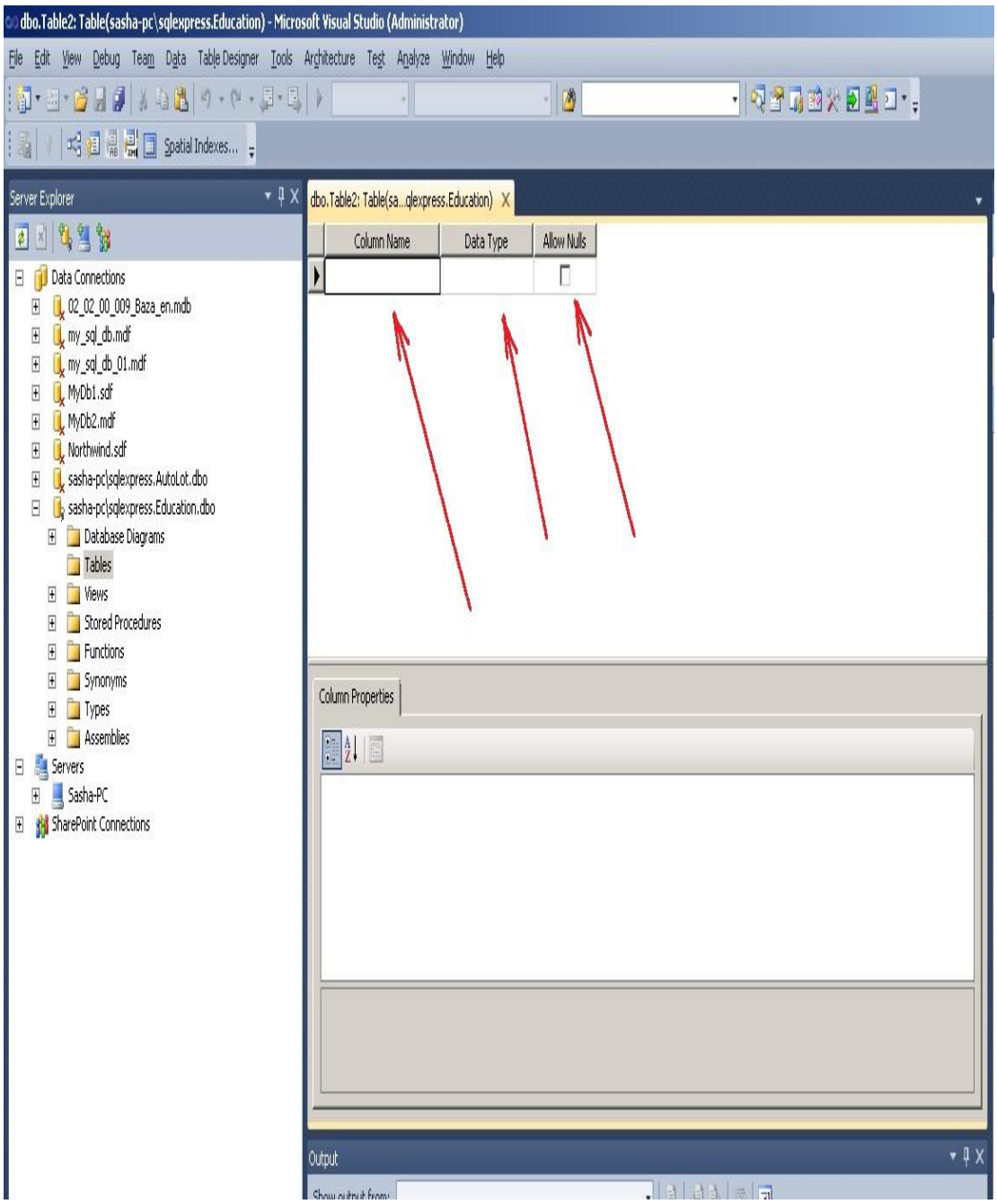


Рис. 8. Окно создания новой таблицы

С помощью редактора таблиц нужно сформировать таблицу Student как изображено на рисунке 9. Имя таблицы нужно задать при ее закрытии.

В редакторе таблиц можно задавать свойства полей в окне Column Properties. Для того, чтобы задать длину строки (nvchar) в символах, в окне Column Properties есть свойство Length. По умолчанию значения этого свойства равно 10.

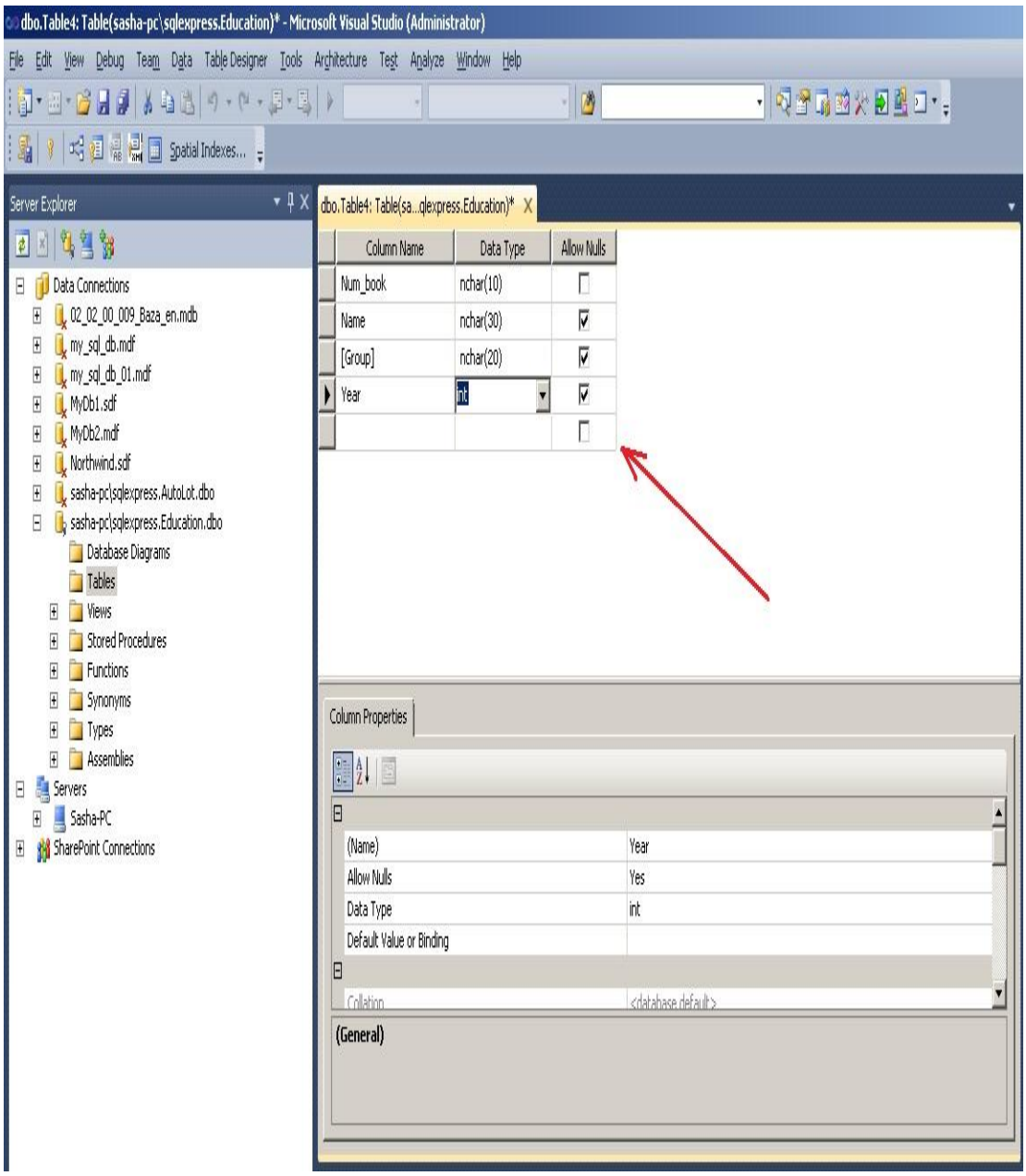


Рис. 9. Таблица Student

Следующим шагом нужно задать ключевое поле. Это осуществляется вызовом команды “Set Primary Key” из контекстного меню поля Num_book. С помощью ключевого поля будут установлены связи между таблицами. В нашем случае ключевым полем есть номер зачетной книжки.

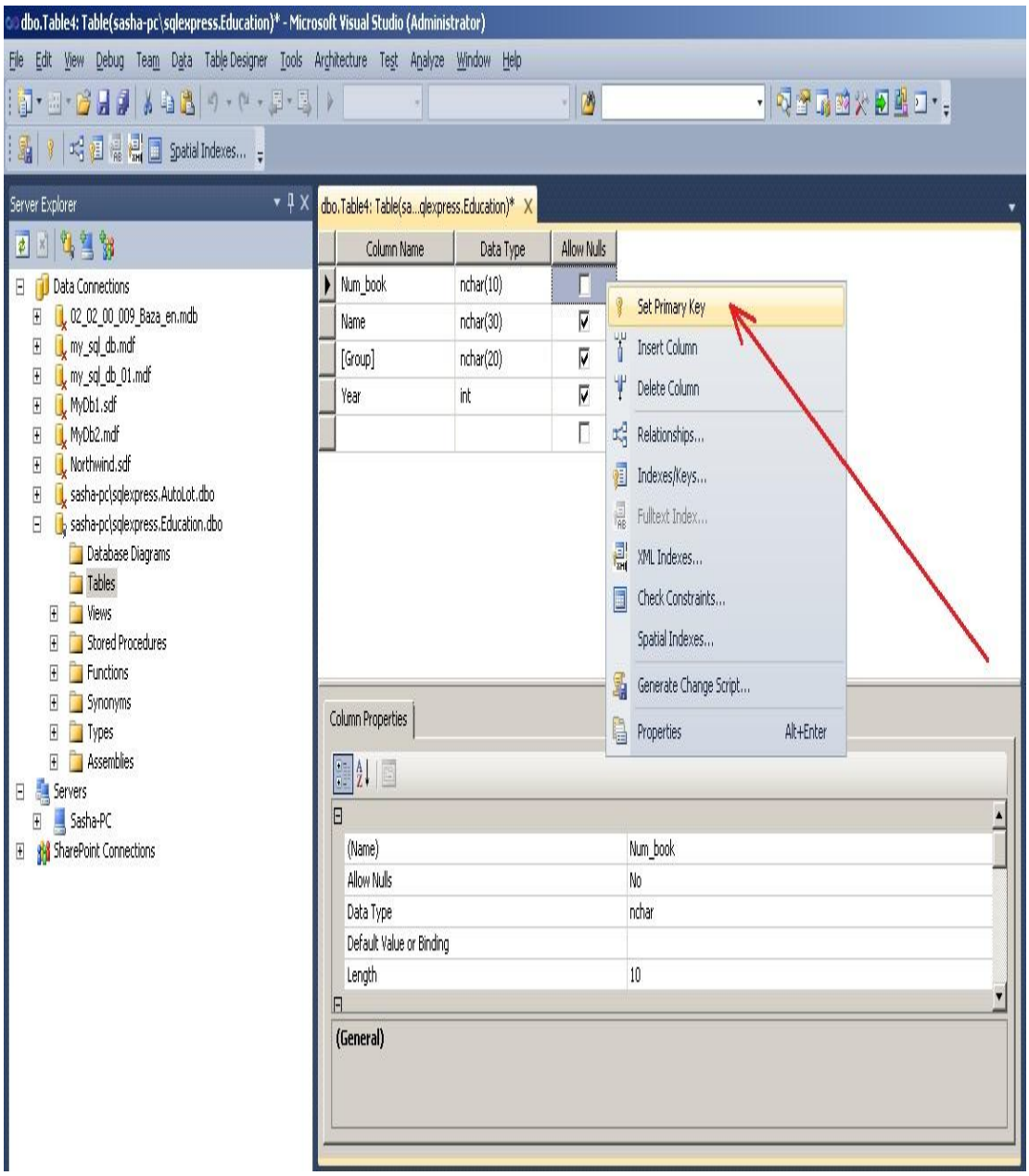


Рис. 10. Задание ключевого поля

После установки первичного ключа окно таблицы будет иметь вид как изображено на рисунке 11.

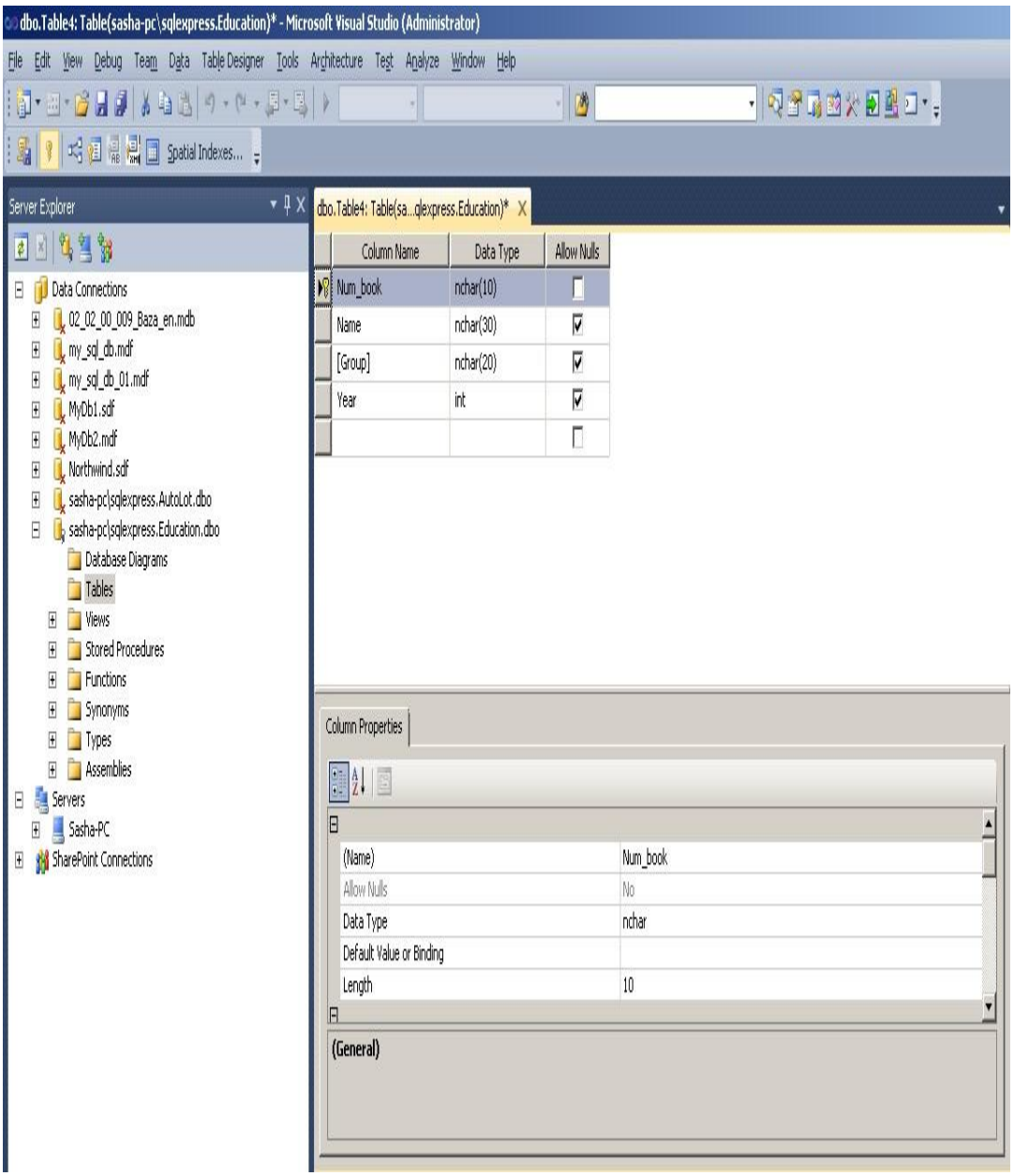


Рис. 11. Таблица Student после окончательного формирования

Теперь можно закрыть таблицу. В окне сохранения таблицы нужно задать ее имя – Student (рис. 12).

Рис. 12. Ввод имени таблицы Student

6. Создание таблицы Session.

По образцу создания таблицы Student создается таблица Session.

На рисунке 13 изображен вид таблицы Session после окончательного формирования. Первичный ключ (Primary Key) устанавливается в поле Num_book. Имя таблицы задается Session.

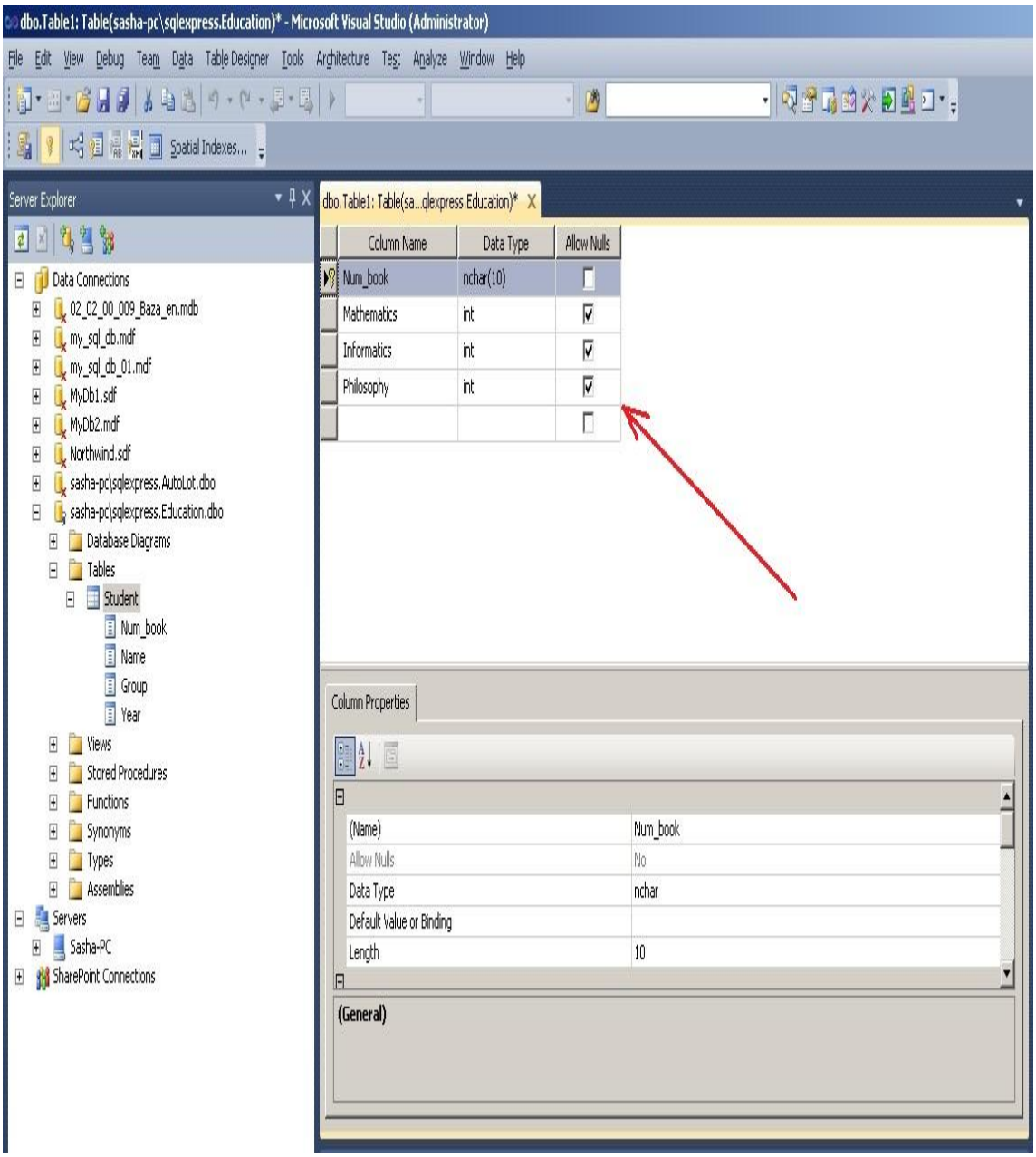


Рис. 13. Таблица Session

После выполненных действий, в окне Server Explorer будут отображаться две таблицы Student и Session.

Таким образом, в базу данных можно добавлять любое количество таблиц.

7. Редактирование структуры таблиц.

Бывают случаи, когда нужно изменить структуру таблицы базы данных.

Для того, чтобы вносить изменения в таблицы базы данных в MS Visual Studio, сначала нужно снять опцию “Prevent Saving changes that require table re-creation” как показано на рисунке 14. Иначе, MS Visual Studioбудет блокировать внесения изменений в ранее созданную таблицу. Окно Options, показанное на рисунке 14 вызывается из меню Tools в такой последовательности:

Tools -> Options -> Database Tools -> Table and Database Designers

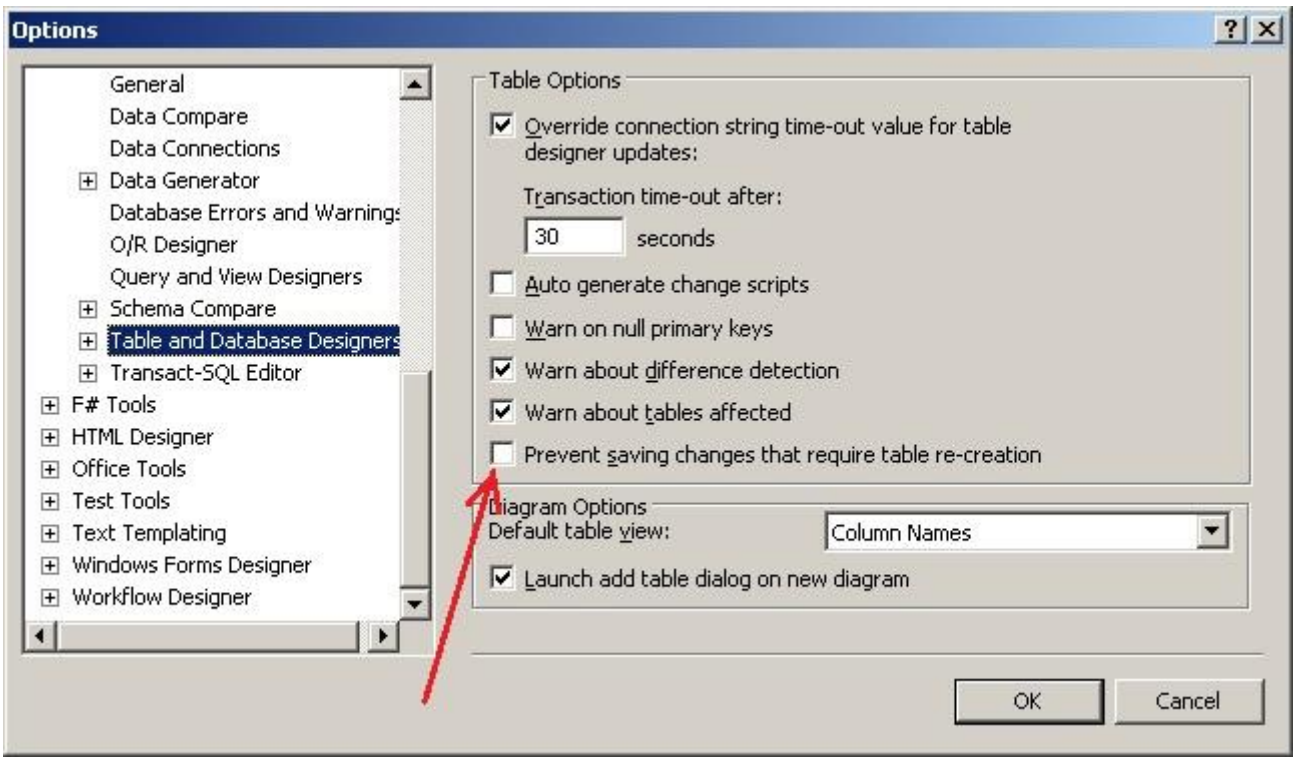


Рис. 14. Опция “Prevent Saving changes that require table re-creation”

После настройки можно изменять структуру таблицы. Для этого используется команда “Open Table Definition” (рисунок 15) из контекстного меню, которая вызывается для выбранной таблицы (правый клик мышкой).

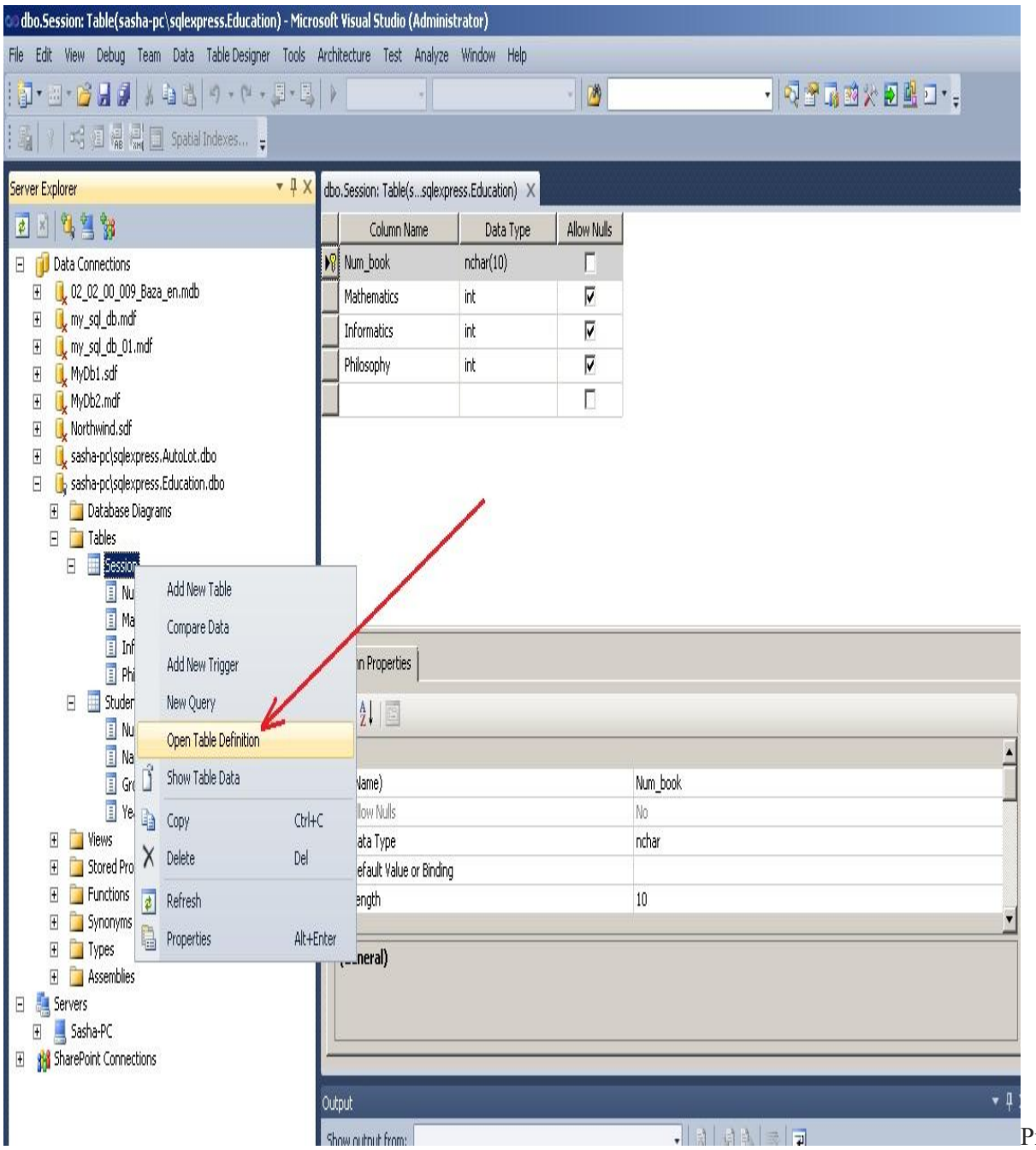


Рис. 15. Вызов команды “Open Table Definition”

Также эта команда размещается в меню Data:
Data -> Open Table Definition
Предварительно таблицу нужно выделить.

8. Установление связей между таблицами.

В соответствии с условием задачи, таблицы связаны между собою по полю Num_book.

Чтобы создать связь между таблицами, сначала нужно (рисунок 16):

– выделить объект Database Diagram;
– выбрать команду Add New Diagram из контекстного меню (или из меню Data).

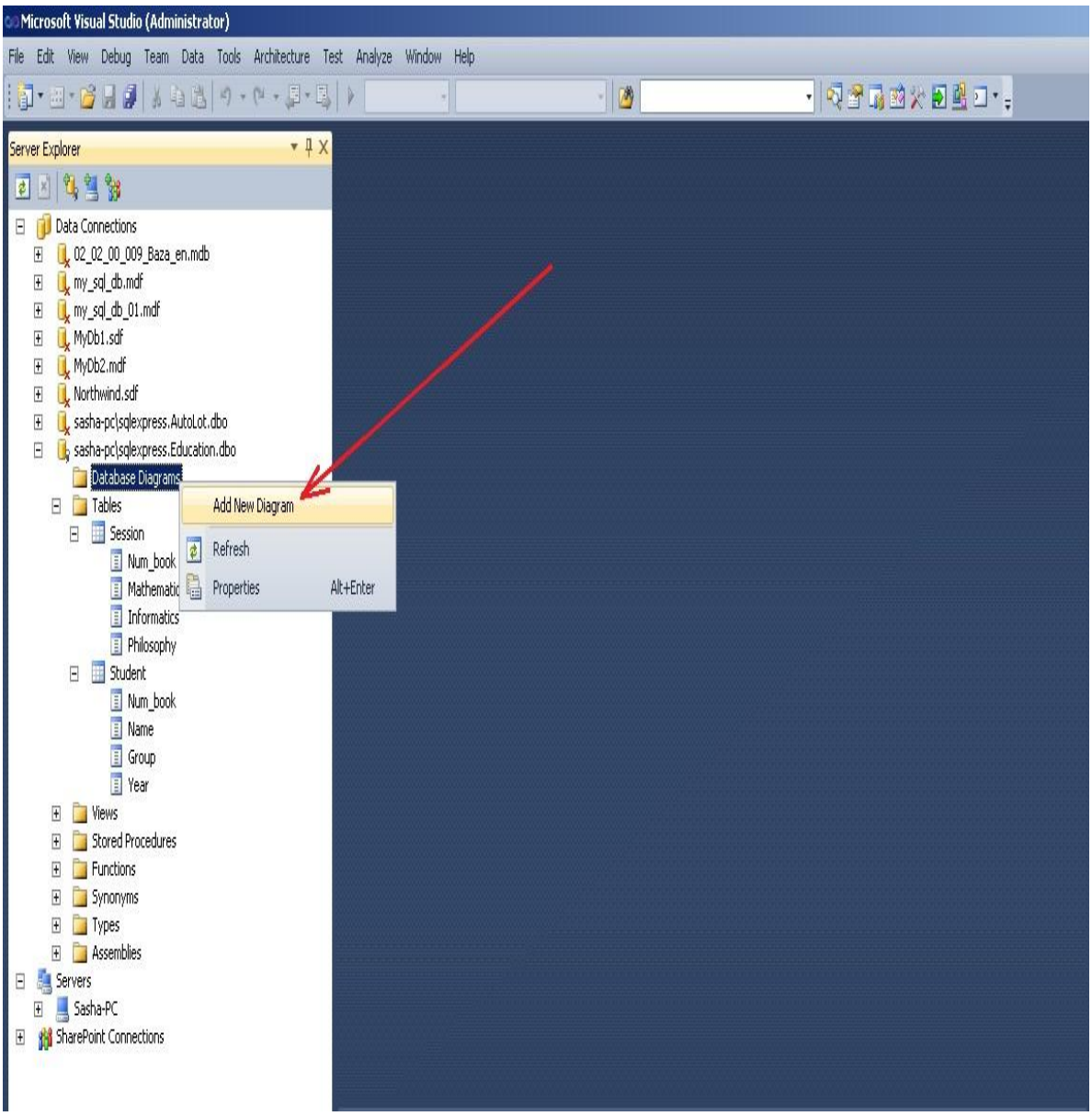


Рис. 16. Вызов команды добавления новой диаграммы

В результате откроется окно добавления новой диаграммы Add Table (рисунок 17). В этом окне нужно выбрать последовательно две таблицы Session и Student и нажать кнопку Add.

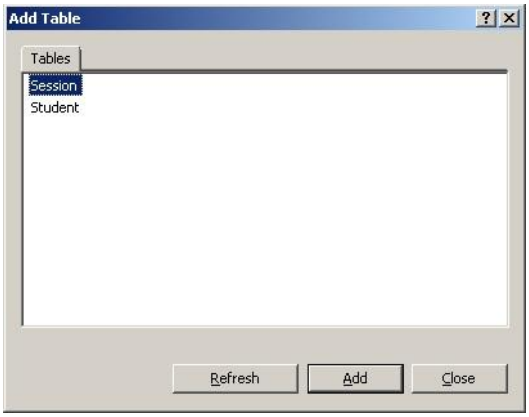


Рис. 17. Окно добавления таблиц к диаграмме

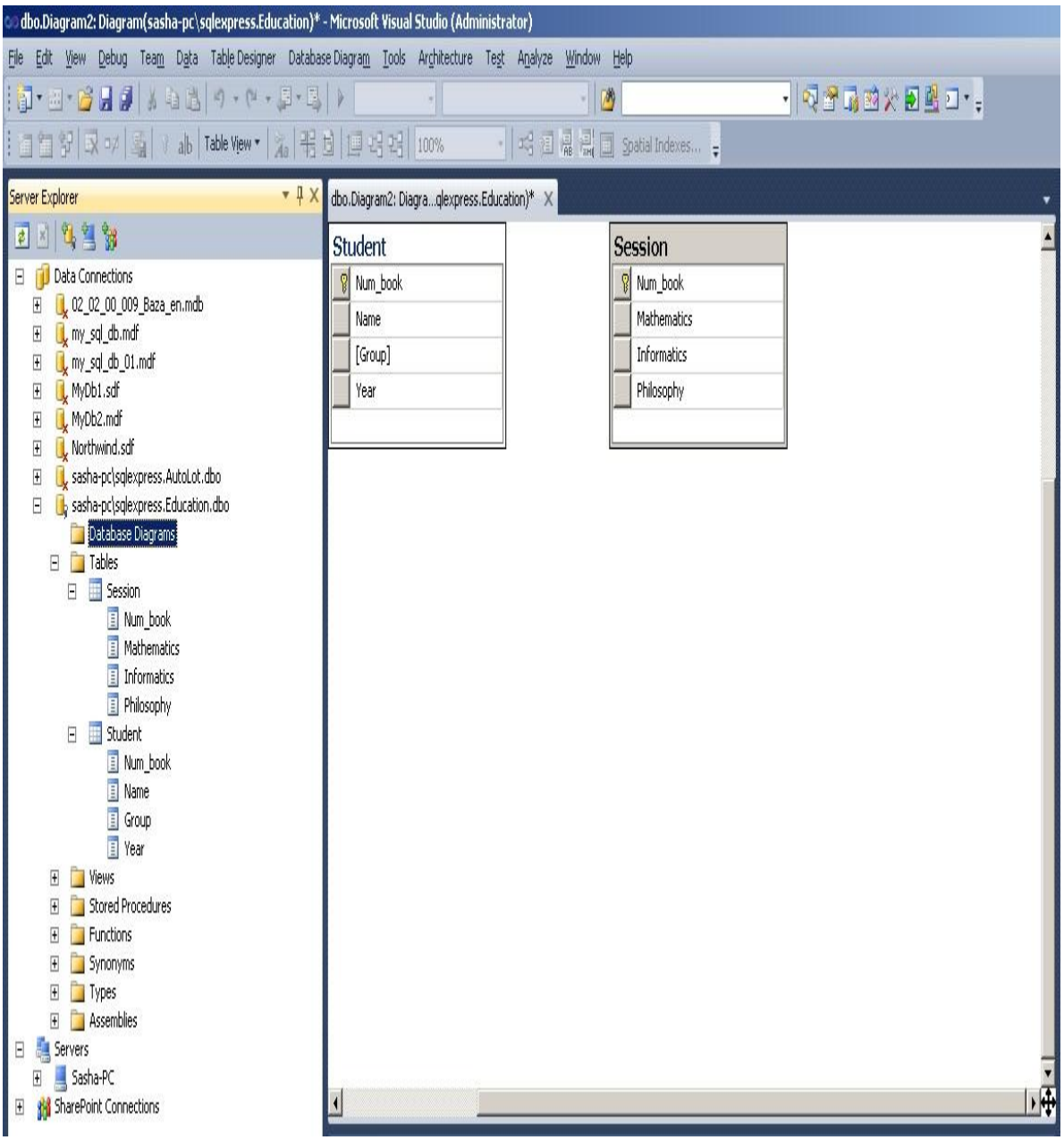


Рис. 18. Таблицы Student и Session после добавления их к диаграмме

Чтобы начать устанавливать отношение между таблицами, надо сделать клик на поле Num_book таблицы Student, а потом (не отпуская кнопку мышки) перетянуть его на поле Num_book таблицы Session.

В результате последовательно откроются два окна: Tables and Columns (рис. 19) и Foreign Key Relationship(рис. 20), в которых нужно оставить все как есть и подтвердить свой выбор на OK.

В окне Tables and Columns задается название отношения (FK_Session_Student) и названия родительской (Student) и дочерней таблиц.

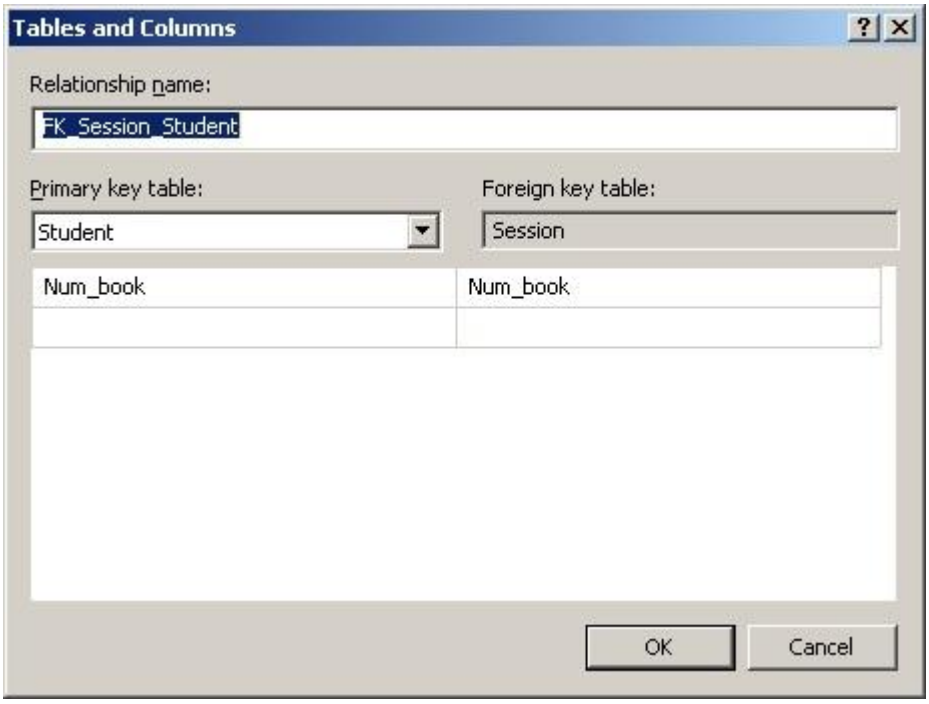


Рис. 19. Окно Tables and Columns

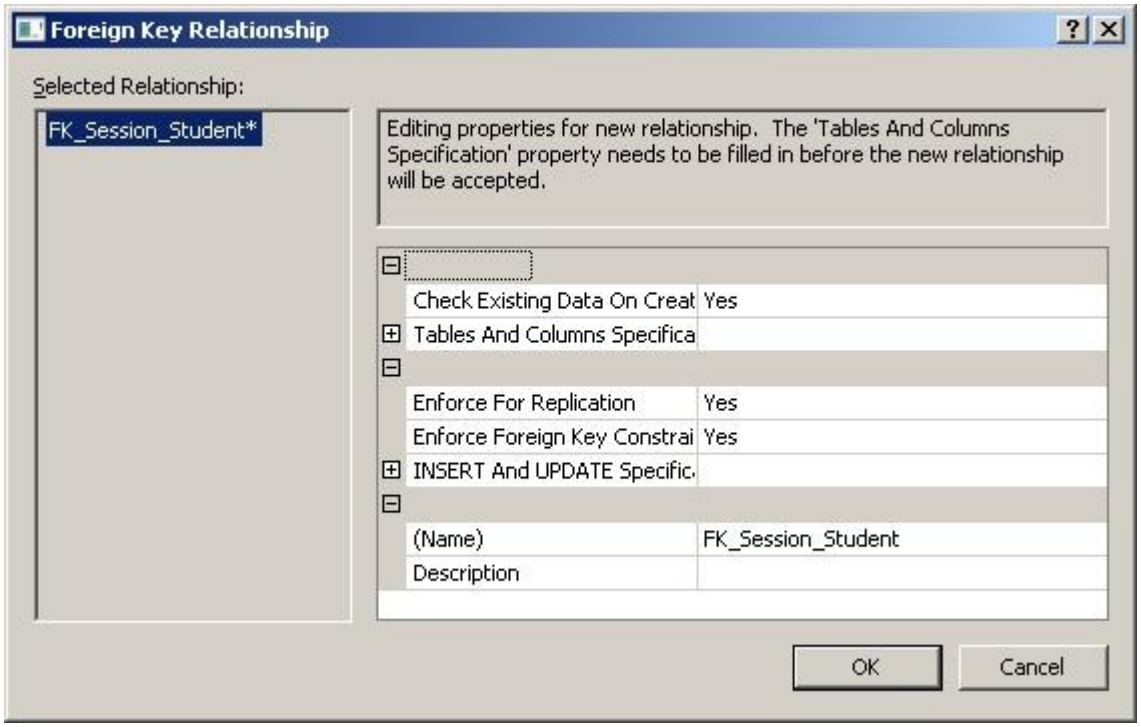


Рис. 20. Окно настройки свойств отношения

После выполненных действий будет установлено отношение между таблицами (рисунок 21).

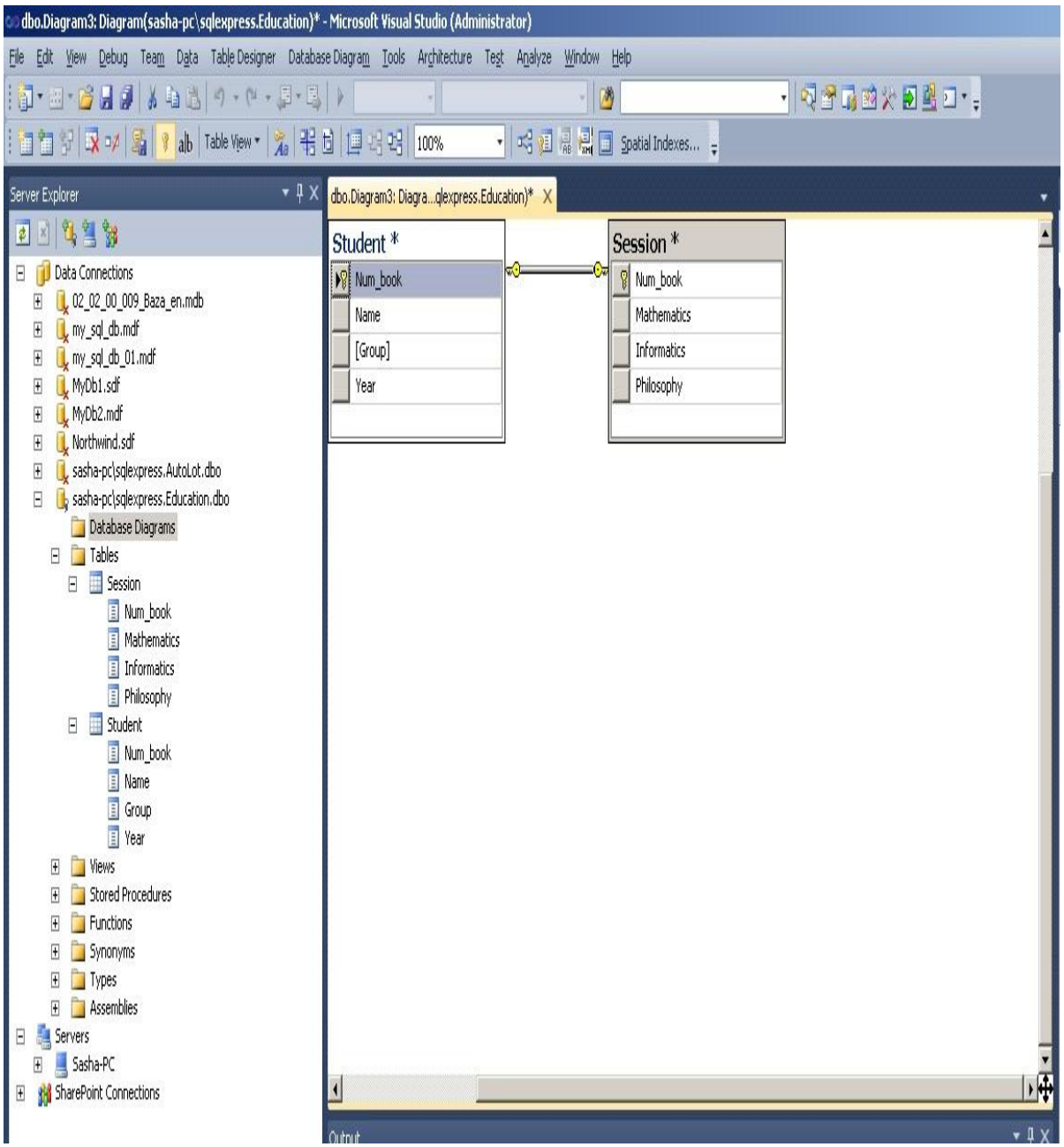


Рис. 21. Отношение между таблицами Student и Session

Сохранение диаграммы осуществляется точно также как и сохранение таблицы. Имя диаграммы нужно выбрать на свое усмотрение (например Diagram1).

После задания имени диаграммы откроется окно Save, в котором нужно подтвердить свой выбор (рисунок 22).

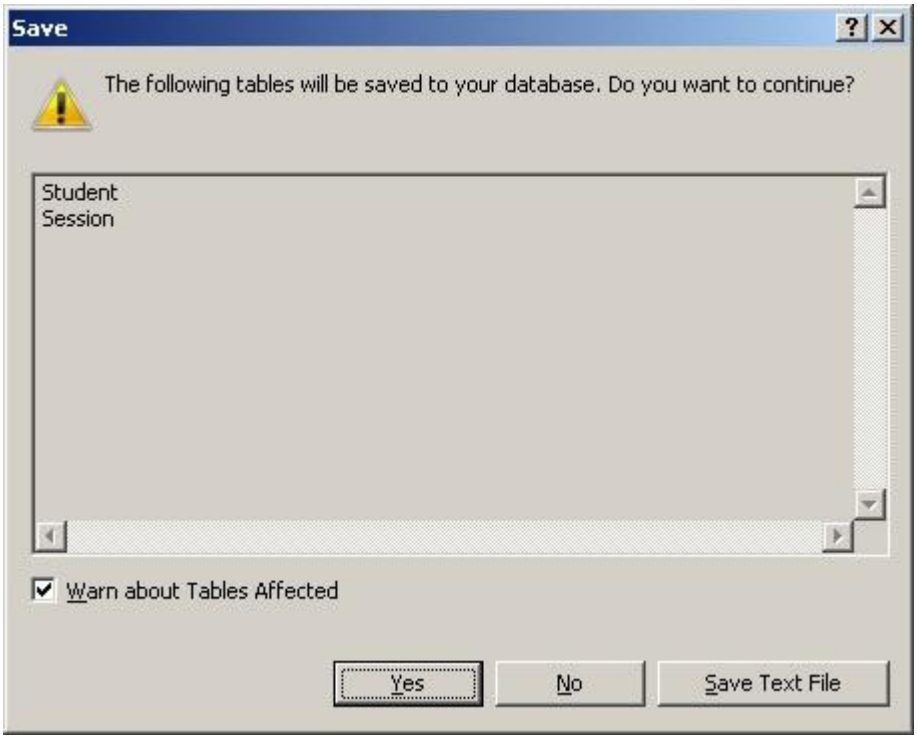


Рис. 22. Подтверждение сохранения изменений в таблицах

9. Внесение данных в таблицы.

Система Microsoft Visual Studio разрешает непосредственно вносить данные в таблицы базы данных.

В нашем случае, при установлении связи (рис. 19) первичной (Primary Key Table) избрана таблица Student.Поэтому, сначала нужно вносить данные в ячейки именно этой таблицы. Если попробовать сначала внести данные в таблицу Session, то система заблокирует такой ввод с выводом соответствующего сообщения.

Чтобы вызвать режим ввода данных в таблицу Student, нужно вызвать команду Show Table Data из контекстного меню (клик правой кнопкой мышки) или с меню Data (рис. 23).

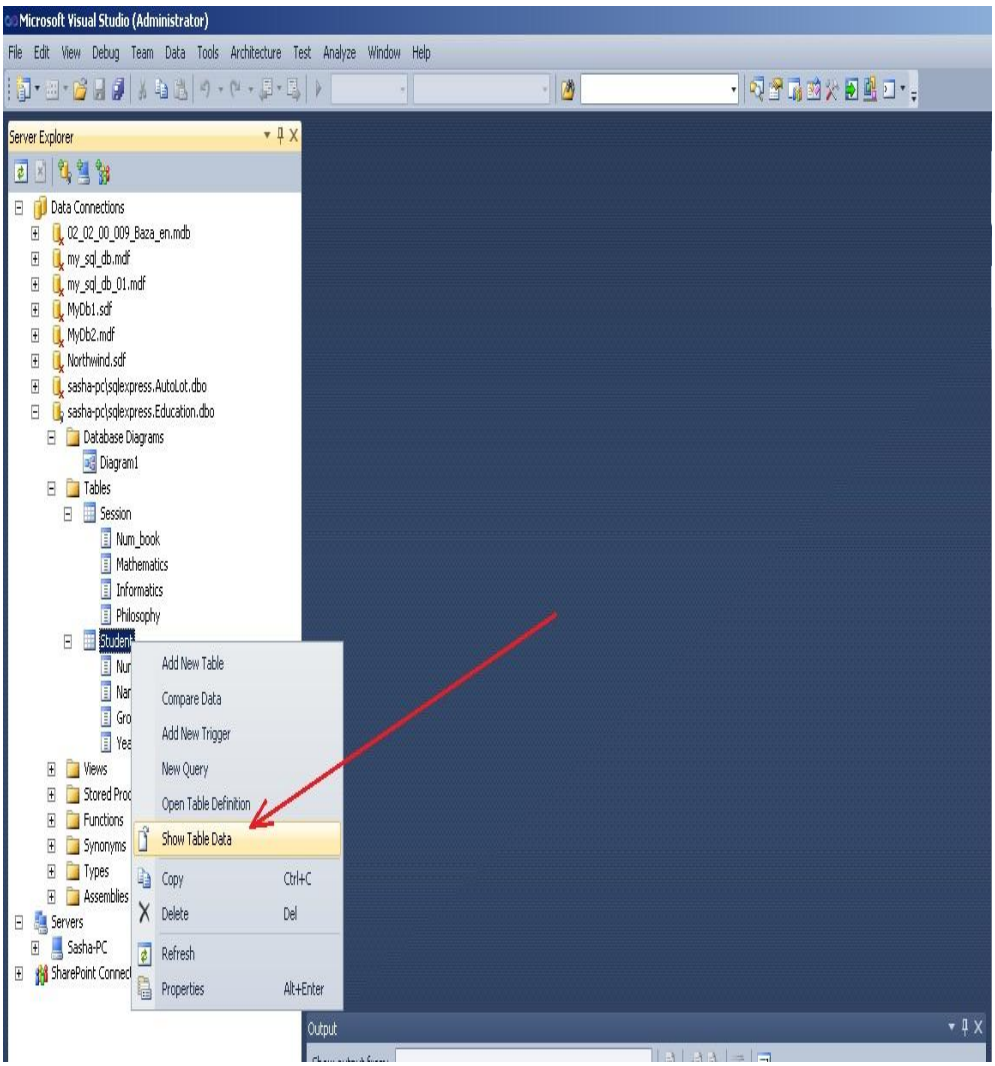


Рис. 23. Команда Show Table Data

Откроется окно, в котором нужно ввести входные данные (рис. 24).

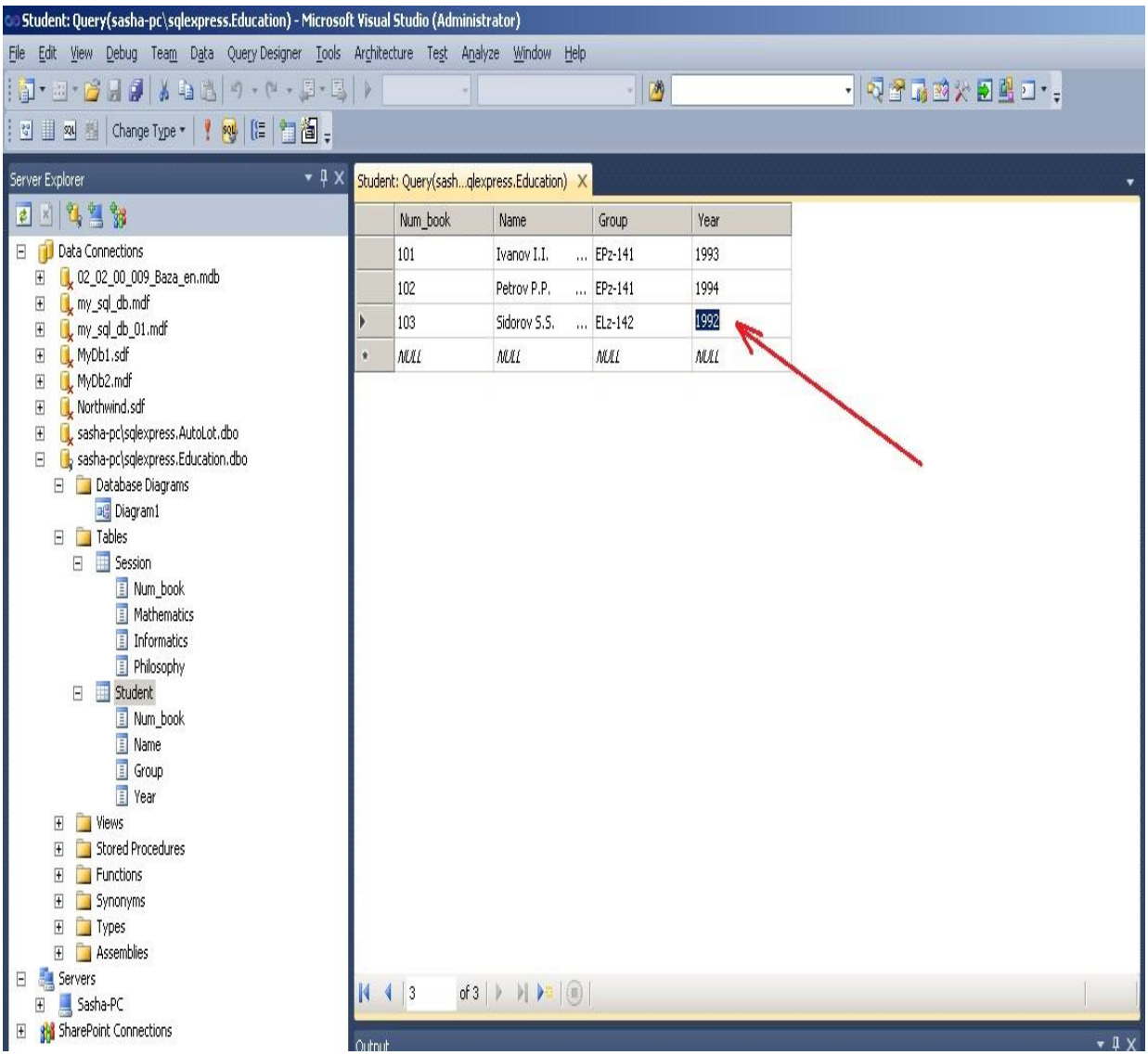


Рис. 24. Ввод данных в таблице Student

После внесения данных в таблицу Student нужно внести данные в таблицу Session.

При внесении данных в поле Num_book таблицы Session нужно вводить точно такие же значения, которые введены в поле Num_book таблицы Student (поскольку эти поля связаны между собой).

Например, если в поле Num_book таблицы Student введены значения “101”, “102”, “103” (см. рис. 24), то следует вводить именно эти значения в поле Num_book таблицы Session. Если попробовать ввести другое значение, система выдаст приблизительно следующее окно (рис. 25).

Рис. 25. Сообщение об ошибке ввода данных связанных таблиц Student и Session

Таблица Session с введенными данными изображена на рисунке 26.

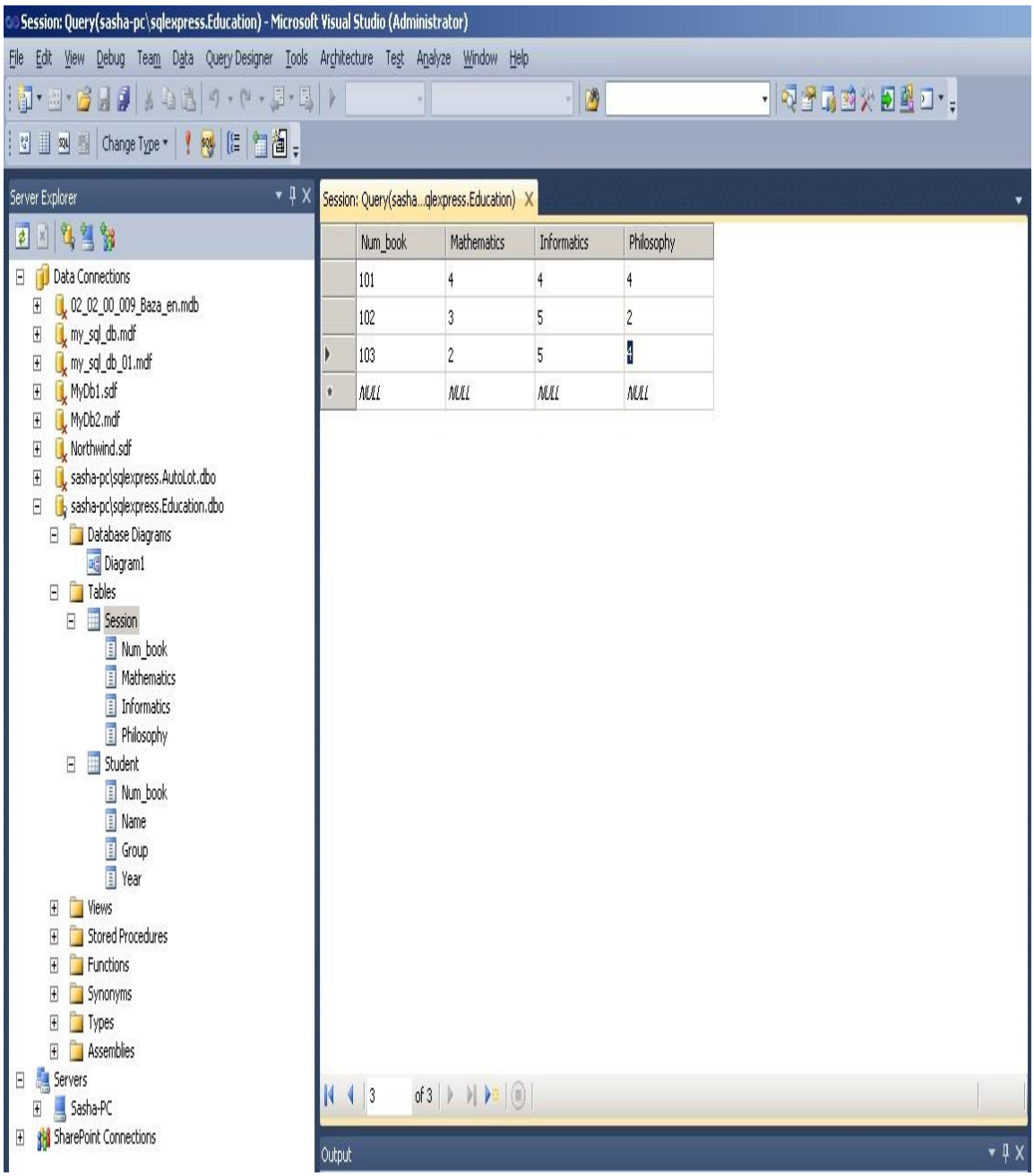


Рис. 26. Таблица Session с введенными данными

Итак, база данных создана. Ввод и обработку данных в таблицах можно реализовать программным путем.

Раздел 2. Перенос файла БД Microsoft SQL на другой компьютер (Обращаю Ваше внимание-это общая технология, в конкретной версии Visul Studio могут быть вариации)

В большинстве случаев необходимо разрабатывать приложения, использующие в качестве базы данных Microsoft SQL Server. Наиболее рациональным решением является разработка базы данных в формате Microsoft SQL на рабочем компьютере с установленной локальной версией Microsoft SQL Server.

При сдаче проекта заказчику возникает необходимость переноса базы данных с локального компьютера. Для переноса на другой компьютер нам потребуется скопировать два файла – саму базу данных имя вашей базы.mdf и файл отчетов о транзакциях имя вашей базы .ldf. Однако непосредственное копирование данных файлов невозможно, так как данные файлы используются сервером баз данных. Для того чтобы сделать файлы доступными для копирования, базу данных необходимо отсоединить от сервера через диалоговое окно «Отсоединение базы данных». Подтверждаем отсоединение, нажимая кнопку «ОК», – и база отсоединена. Теперь нужные файлы доступны для копирования.

Для присоединения базы данных на другом компьютере запускаем Visual Studio, выделяем ветку «Базы данных» и в контекстном меню выбираем «Присоединить».

В появившемся окне указываем расположение файла базы данных Имя вашей базы.mdf – файл отчетов присоединится автоматически – и нажимаем «ОК». Присоединившаяся база данных немедленно отображается в папке «Базы данных». Следует отметить, что после присоединения БД может потребоваться настройка пользователей БД и прав доступа.

ЛАБОРАТОРНОЕ ЗАНЯТИЕ № 3
РАБОТА С БАЗАМИ ДАННЫХ. ТЕХНОЛОГИЯ ADO.NET. СОЗДАНИЕ ХРАНИМЫХ ПРОЦЕДУР В MICROSOFT VISUAL STUDIO.

Цель - обеспечить студентов конкретным инструментарием для практической реализации методов проектирования приложений клиент-сервер.

Задание на лабораторную работу.

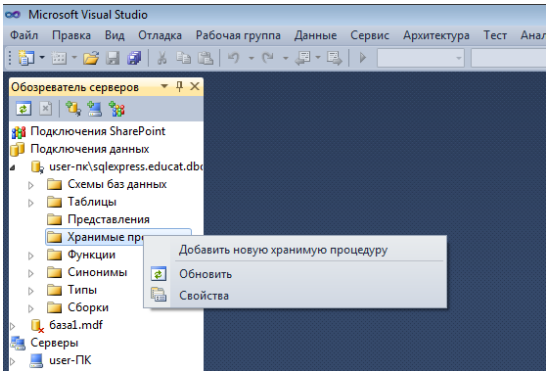
1. Создать хранимые процедуры, обеспечивающие возможность выполнения запросов к базе данных, согласно полученному варианту(варианты задания получить у преподавателя).
2. Отчет представить в письменной форме. В отчете представить скриншоты хранимых процедур и результатов запуска процедур в виде таблицы

Описание процедуры	
SQL-конструкция для создания	**Команда для извлечения**
Результат запуска	

Теоретическая часть

Среда Visual Studio предоставляет интерфейс для создания хранимых процедур в базе данных при наличии подключения к ней.

Запустим Visual Studio (даже нет необходимости создавать какой-либо проект), перейдем на вкладку «Обозреватель баз данных» («Server Explorer»), раскрываем подключение к базе данных, затем на узле «Хранимые процедуры» («Stored Procedures») щелкаем правой кнопкой и выбираем пункт «Добавить новую хранимую процедуру» («New Stored Procedure»)



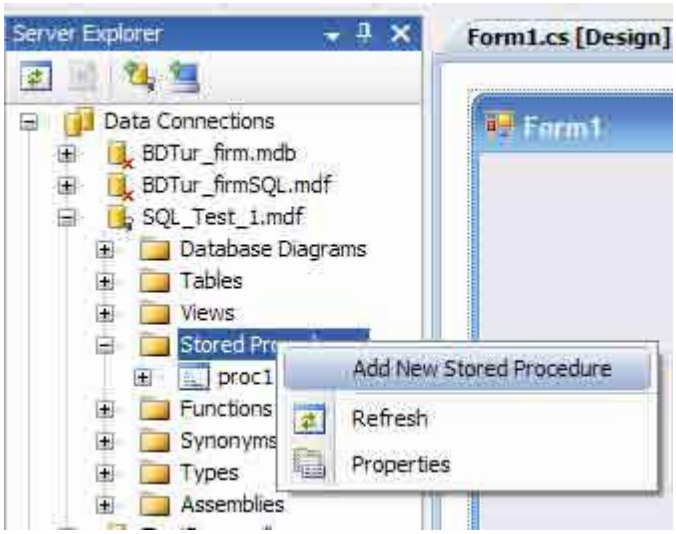


Рис. 1. Создание новой процедуры в окне «Server Explorer»

Появляется шаблон структуры, сгенерированный мастером:

```
CREATE PROCEDURE dbo.StoredProcedure1
/*
(
@parameter1 int = 5,
@parameter2 datatype OUTPUT
)
*/
AS
/* SET NOCOUNT ON */
RETURN
```

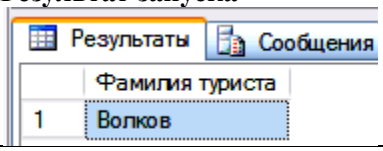
Для того чтобы приступить к редактированию, достаточно убрать знаки комментариев «/*», «*/».

Команда NOCOUNT со значением ON отключает выдачу сообщений о количестве

строк таблицы, получающейся в качестве запроса. Необходимость данного оператора заключается в том, что при использовании более чем одного оператора (SELECT, INSERT, UPDATE или DELETE) в начале запроса надо поставить команду SET NOCOUNT ON, а перед последним оператором SELECT – команду SET NOCOUNT OFF.

Например, хранимую процедуру proc_po1 (см. таблицу 2)

Таблица 2

<table>
<tr><td colspan="2">Описание
Извлечение фамилии туриста по заданному коду</td></tr>
<tr><td>SQL-конструкция для создания</td><td>Команда для извлечения</td></tr>
<tr><td>create proc proc_po1
@TouristID int,
@LastName nvarchar(60) output as
select @LastName = Фамилия
from Туристы
where [Код туриста] = @TouristID</td><td>declare @LastName nvarchar(60)
exec proc_po1 '4',
@LastName output
select @LastName as
'Фамилия туриста'</td></tr>
<tr><td colspan="2">Результат запуска
 
</td></tr>
</table>

можно переписать следующим образом:

```
CREATE PROCEDURE dbo.proc_vs1
(
@TouristID int,
@LastName nvarchar(60) OUTPUT
)
AS
SET NOCOUNT ON
SELECT @LastName = Фамилия
FROM Туристы
WHERE ([Код туриста] = @TouristID)
RETURN
```

После завершения редактирования SQL-конструкция будет обведена синей рамкой. Щелкнув правой кнопкой в этой области и выбрав пункт меню

«Разработать блок SQL» («Design SQL Block»), можно перейти к построителю запросов (Query Builder) (рис. 2).

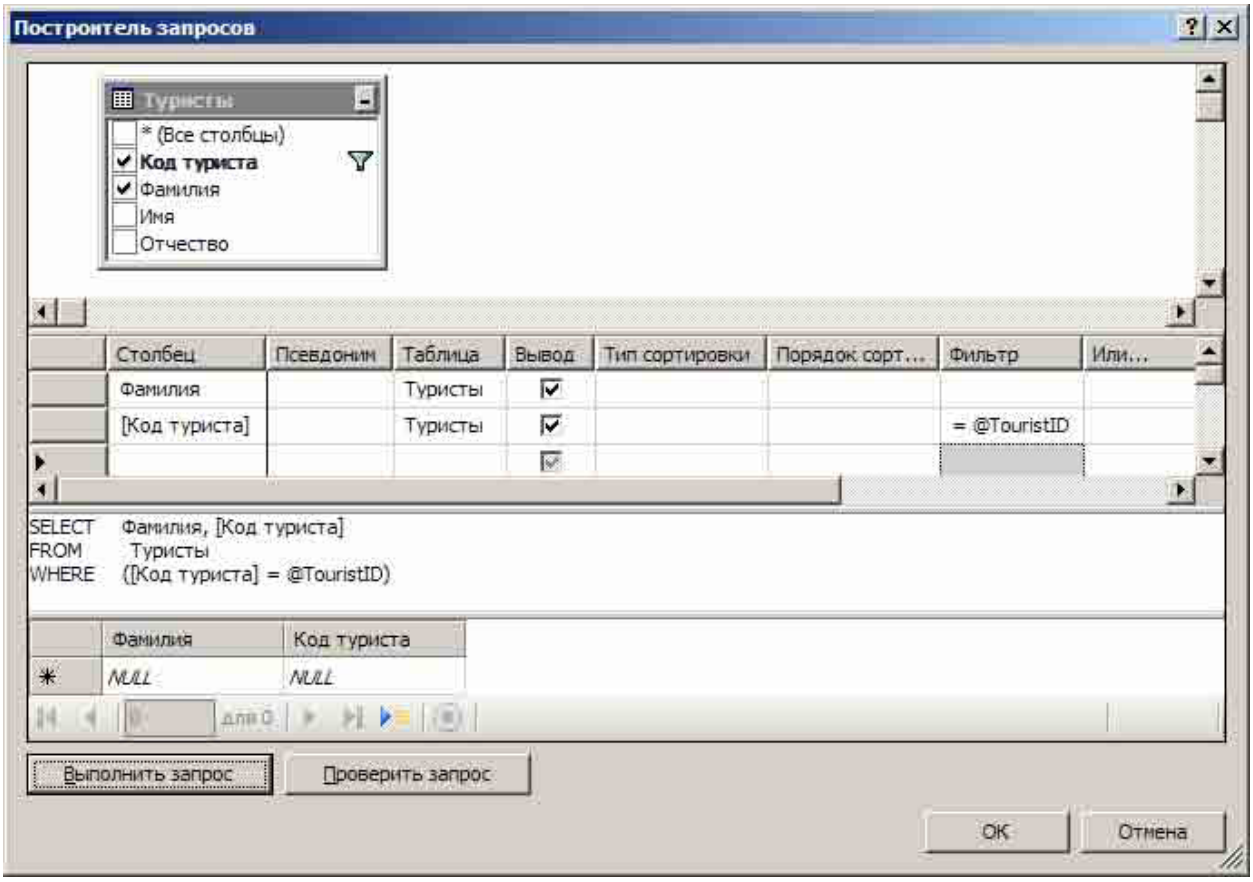


Рис. 2. Построитель запросов (Query Builder)

Для выполнения процедуры необходимо в выпадающем меню процедуры, представленном на рисунке 3, выбрать пункт Execute.

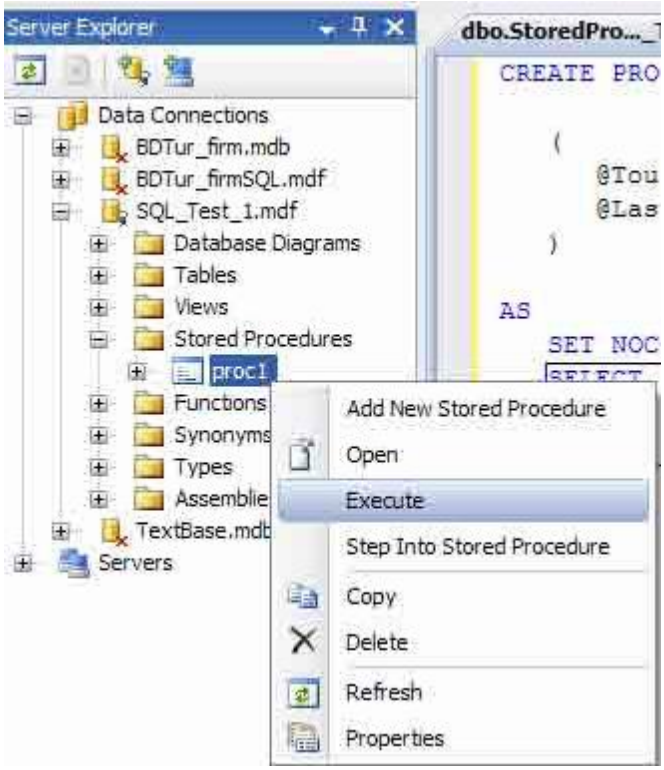


Рис. 3. Выполнение хранимой процедуры

При попытке выполнить хранимую процедуру в окно сообщений будет выведена

информация о том, что данная процедура требует значение параметра. Для записи процедуры в базу данных достаточно ее сохранить. Происходит синхронизация с базой данных, и процедура "proc_vs1" появляется в списке. Двойной щелчок открывает ее для редактирования, причем заголовок имеет следующий вид:

ALTER PROCEDURE dbo.proc_vs1

Оператор ALTER позволяет производить действия (редактирование) с уже имеющимся объектом базы данных.

ЛАБОРАТОРНОЕ ЗАНЯТИЕ № 4
РАБОТА С БАЗАМИ ДАННЫХ. ТЕХНОЛОГИЯ ADO.NET. РАЗРАБОТКА КЛИЕНТСКОЙ ЧАСТИ БАЗЫ ДАННЫХ В ИНСТРУМЕНТАЛЬНОЙ ОБОЛОЧКЕ

Цель - обеспечить студентов конкретным инструментарием для практической реализации методов проектирования приложений клиент-сервер.

Задания к лабораторной работе

1. Соберите так, как описано в п. 1.1, простейшее приложение для работы с **базой данных вашего варианта**
2. Каково назначение объектов классов DataSet, DataAdapter, DataTable, DataRelation?
3. Нарисуйте схему связей классов DataSet, DataTable, DataRelation; DataTable и DataRelation; DataColumn и DataTable; DataRow и DataTable.
4. Соберите в соответствии с описанием п. 1.4 «ручной» вариант приложения для работы с базой данных Microsoft Access.
5. Соберите в соответствии с описанием п. 1.4 «ручной» вариант приложения для работы с базой данных Microsoft SQL
6. Для чего нужен компонент BindingNavigator и как его использовать?
7. В чем различие визуальных и не визуальных компонент? Приведите пример их использования.
8. Как связываются объекты DataGridView и BindingNavigator с базой данных?
9. Как связываются текстовые поля TextBox с полями базы данных?
10. Для чего в приложениях использованы строки ConnectionString и CommandText?
11. **Отчет представить в письменном виде. Задание 1- в виде скриншотов, задания 4-5 в виде скриншотов и листингами кодов, остальные задания -это ответы на вопросы.**

Теоретическая часть

1. СОЗДАНИЕ ПРИЛОЖЕНИЙ БАЗ ДАННЫХ

Реализация клиентского приложения с помощью проекта на языке C#

В среде Microsoft Visual Studio работа по созданию клиентского приложения начинается с построения проекта для выбранного языка программирования, который выбирается в окне, открывающемся после исполнения команды среды File/New/Project и создания нового проекта Windows Application

1.1. Пример простейшего приложения баз данных

Создадим простое приложение баз данных, которое выводит на экранную форму информацию из таблицы «Туристы» и связанную с текущей записью таблицы «Туристы»

запись таблицы «Информация о туристах» из базы данныхю

Для этого создадим пустое Windows-приложение1. Внешний вид среды разработки приведен на рисунке 1.

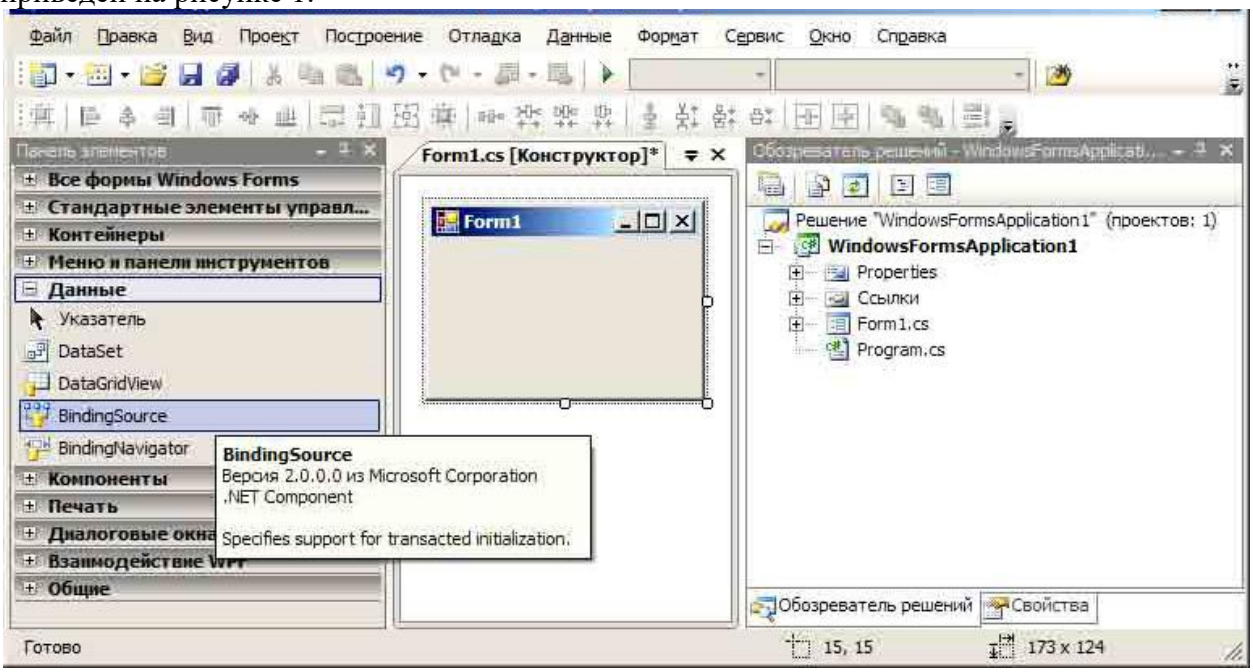


Рис. 1. Пустое приложение

На рисунке 1 выделена группа компонентов «Данные» («Data»), которая содержит компоненты для доступа к данным и манипулирования ими.

Привязку данных БД к форме осуществляет компонент «Binding Source».

Перенесем его на форму. После размещения его на форме среда разработки принимает следующий вид (рис. 2).

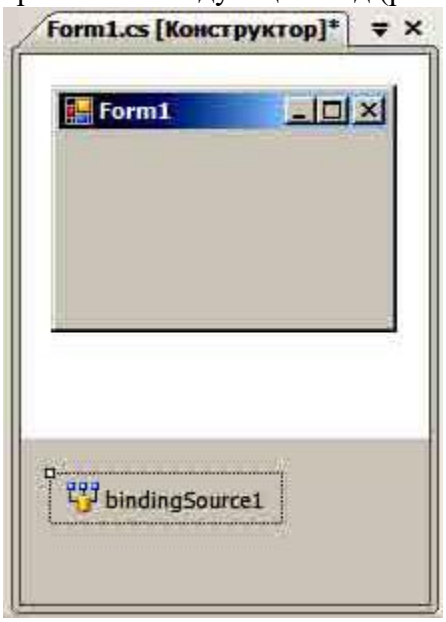


Рис. 2. Компонент Binding Source на форме

Компонент является не визуальным, поэтому он отображается на дополнительной панели. Основным свойством компонента является свойство DataSource, указывающее на источник данных. По умолчанию свойство является пустым, поэтому необходимо сформировать его значение. При выборе данного свойства в окне свойств появляется

следующее окно (рис. 3).

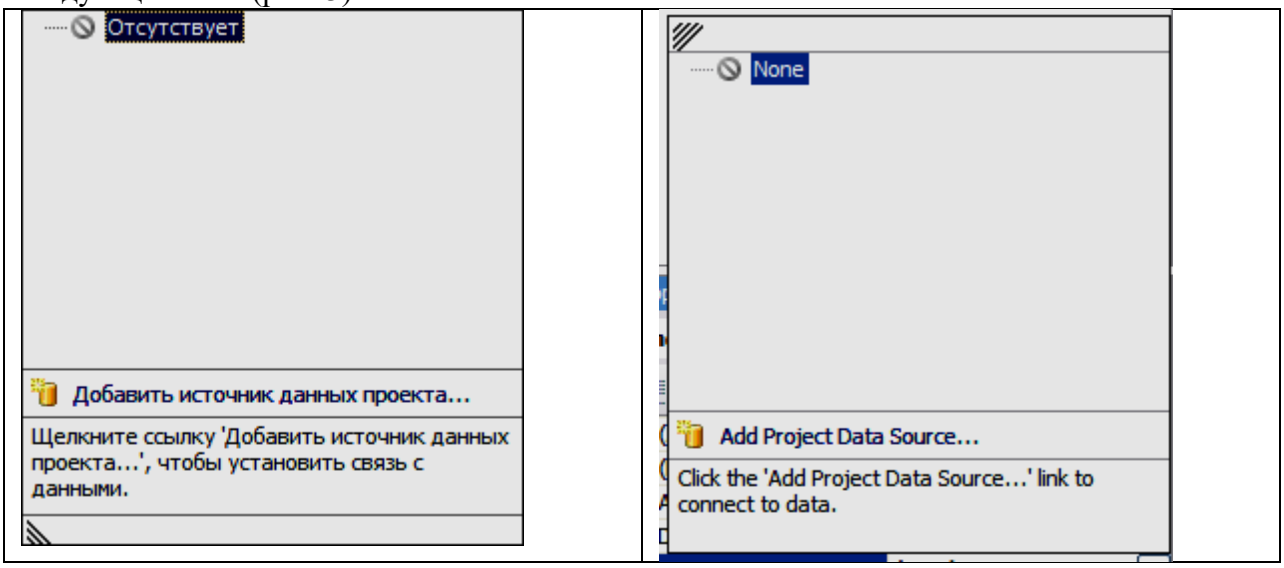


Рис. 3. Список источников данных. Компонент «Binding Source»

В настоящий момент список пуст, поэтому необходимо создать новый источник данных, выбрав команду «Add Project Data Source» для создания нового источника данных и соединения с ним. Появляется следующее окно диалога (рис. 4).

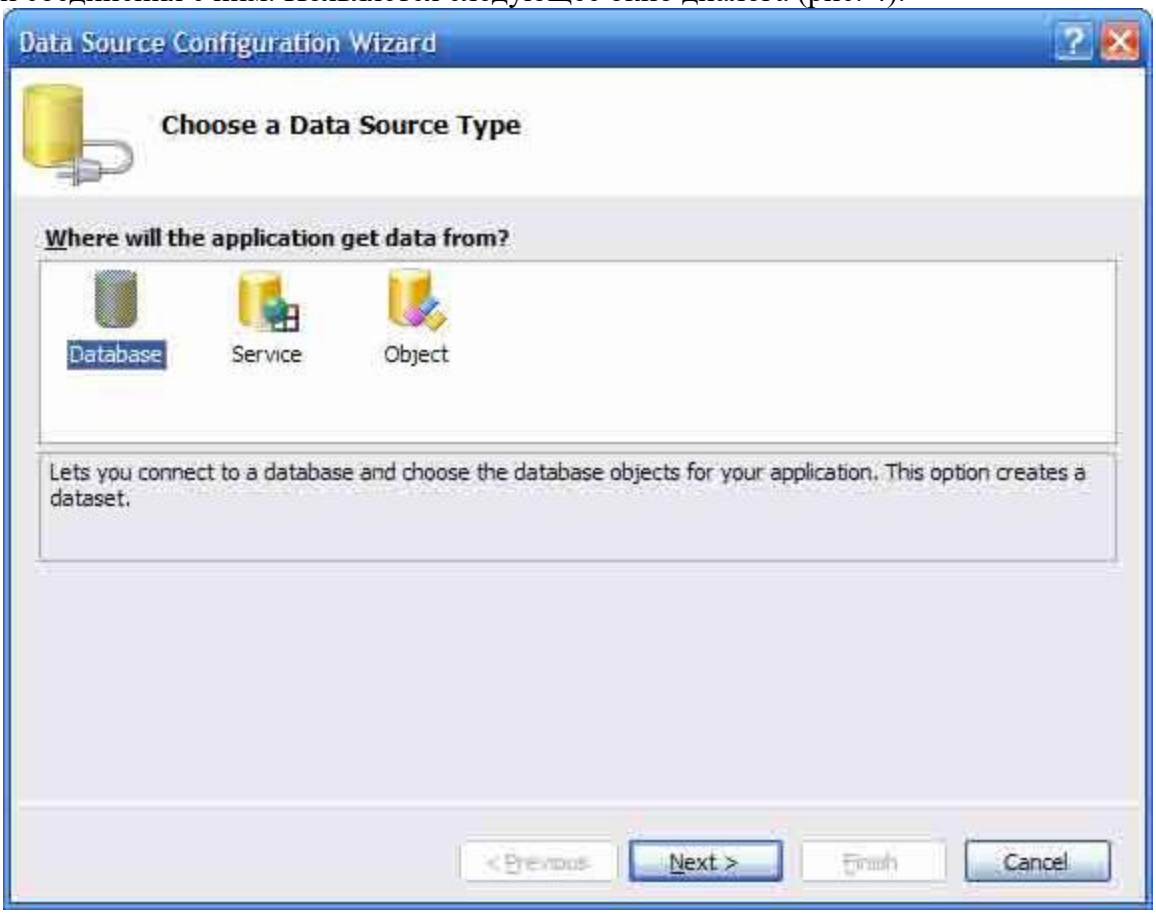


Рис. 4. Список источников данных

Данный диалог предоставляет следующий выбор источников данных:

• Database – База данных;

• Service – Служба, это некоторый сервис, предоставляющий данные. Чаще

всего это Web-сервис;

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

В нашем случае необходимо выбрать пункт «База данных» («Database»).

Появляется окно выбора соединения с данными (рис. 5).

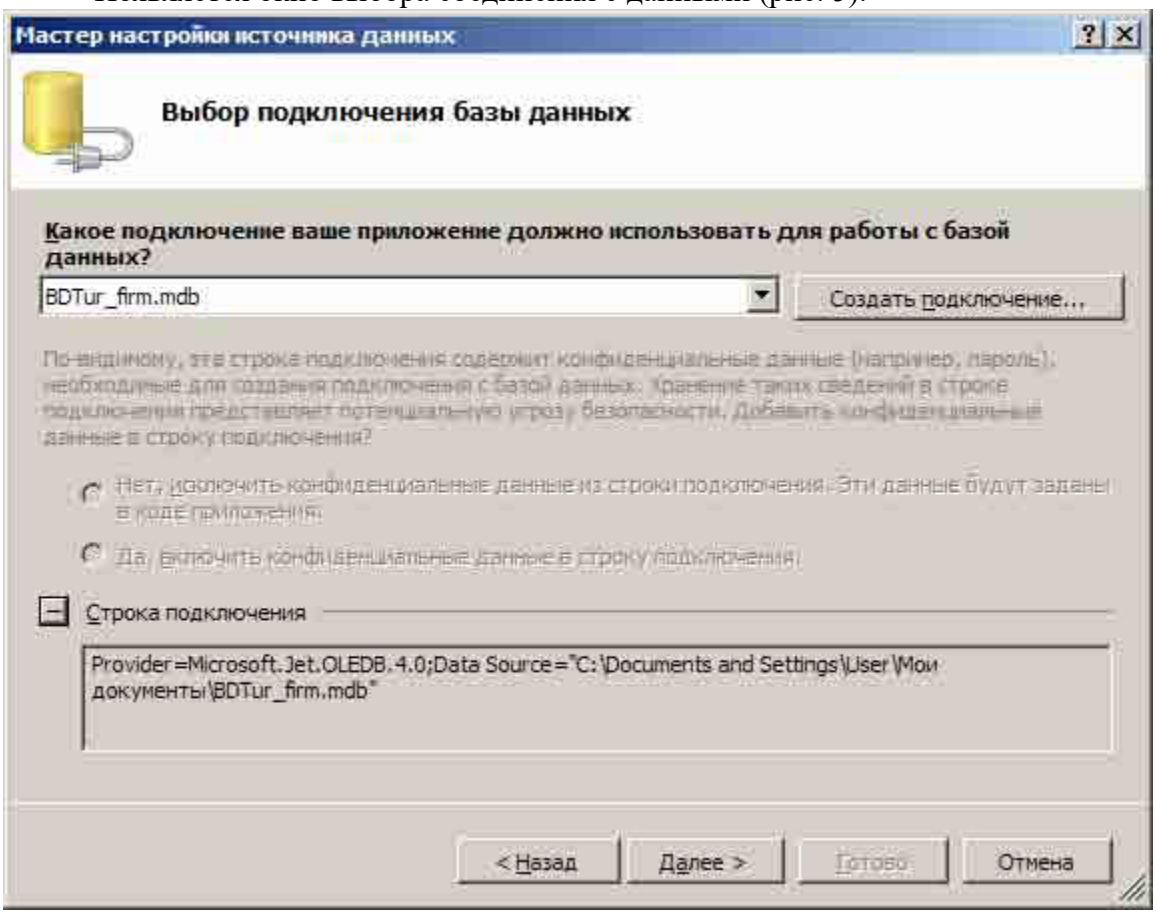


Рис. 5. Выбор соединения с данными

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

В выпадающем списке диалога находятся все создаваемые ранее соединения. Если необходимого соединения в списке нет, то следует использовать кнопку «Создать подключение» («New connection»). Нажатие кнопки приводит к появлению следующего диалога (рис. 6).

В данном диалоге выбирается тип источника данных (в данном случае Microsoft Access), имя базы данных (в данном случае имя и местоположение файла базы данных), имя пользователя и пароль, используемые для подключения к базе данных. Кнопка «Дополнительно» («Advanced») позволяет задать большое количество параметров, относящихся к различным деталям механизма ADO. Использование кнопки «Проверить подключение» («Test Connection») позволит убедиться в правильности введенных параметров и работоспособности соединения.

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

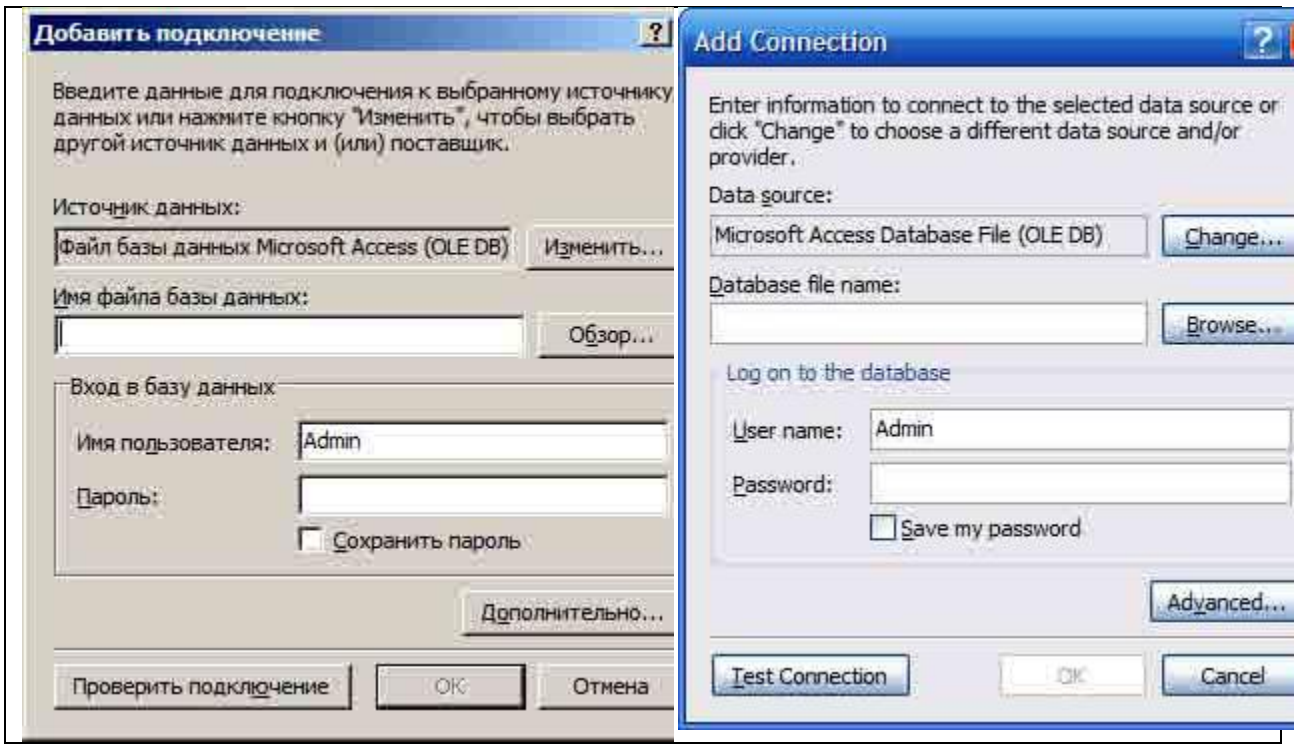


Рис. 6. Создание нового соединения

Последний шаг диалога – выбор тех таблиц или иных объектов базы данных, которые необходимы в данном источнике данных. Окно выбора представлено на рисунке 7.

Рис. 7. Выбор необходимых таблиц

В данном окне выбраны таблицы «Туристы» и «Информация о туристах».

Поскольку иных объектов, кроме таблиц, в базе данных не было создано, на рисунке 7 отображаются только таблицы. На этом создание источника данных завершено. После нажатия кнопки «Готово» («Finish») рядом с компонентом BindingSource на форме появляется компонент DataSet.

Теперь данные, подключенные выше, необходимо отобразить на форме.

Простейшим способом отображения данных является использование компонента DataGridView из группы компонентов Data. Компонент является визуальным и на форме выглядит следующим образом (рис. 8).

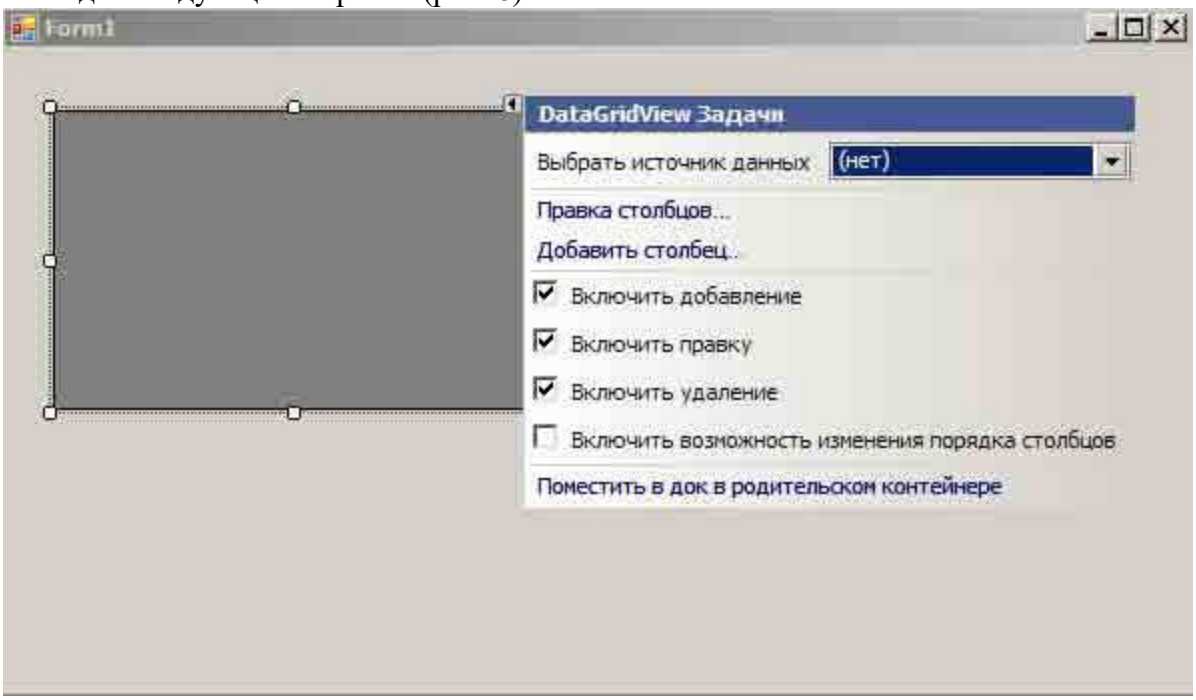


Рис. 8. Компонент DataGridView

Сразу же возникает окно настройки компонента, которое определяет его возможности по редактированию данных: «Включить редактирование» («Enable Adding»), «Включить правку» («Enable Editing»), «Включить удаление» («Enable Deleting»); возможность изменения последовательности столбцов:

«Включить возможность изменения порядка столбцов» («Enable Column Reordering»); а также возможность закрепления в контейнере-родителе.

Для того чтобы компонент мог отображать данные, необходимо выбрать источник данных в выпадающем списке. Выбор выпадающего списка приводит к появлению следующего диалога (рис. 9).

Рис. 9. Выбор источника данных для DataGridView

В данном случае мы выбрали в качестве источника данных таблицу «Туристы». Данный выбор изменяет экранную форму следующим образом (рис. 10).

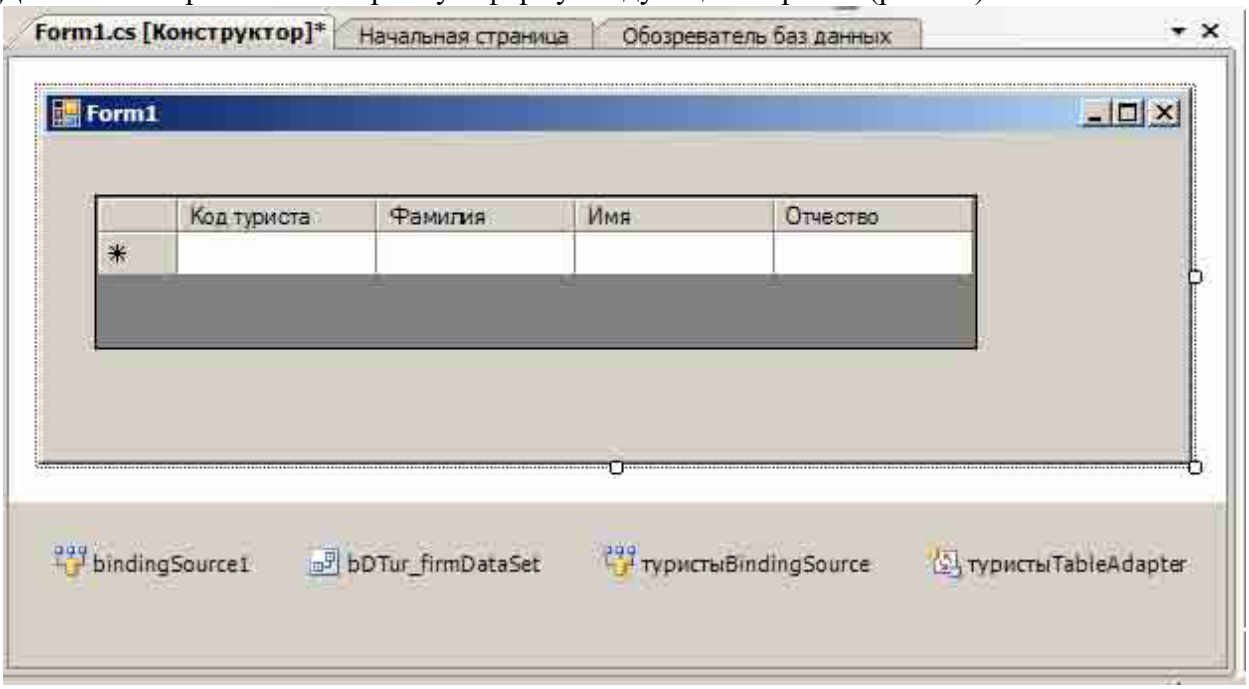


Рис. 10. Компонент DataGridView отображает структуру таблицы

На рисунке видно, что появился еще один компонент BindingSource и компонент TableAdapter, работающий с таблицей «Туристы». Обратите внимание, что в design-time или в процессе разработки данные из таблицы не отображаются.

Теперь необходимо отобразить данные из связанной таблицы «Информация о туристах». Для этого разместим на форме еще один компонент DataGridView и в качестве источника данных выберем следующее (рис. 11).

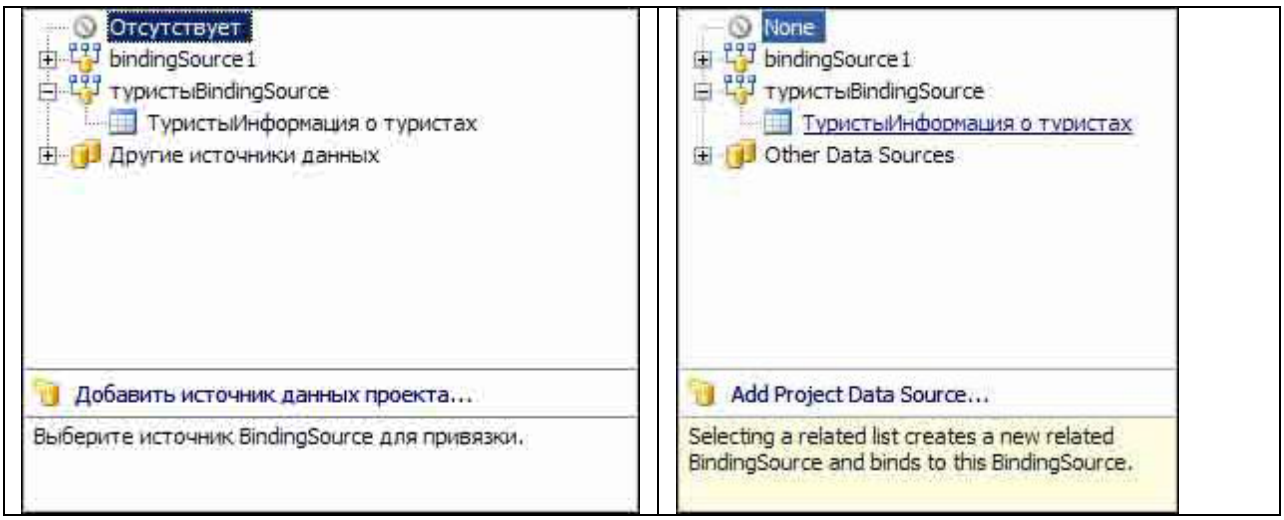


Рис. 11. Выбор источника данных для второго DataGridView

Здесь в качестве источника данных выступает не сама таблица «Информация о туристах», а связь (Binding Source) между таблицами «Туристы» и «Информация о туристах». Такой выбор гарантирует выбор из таблицы «Информация о туристах» только тех строк, которые связаны с текущей строкой в таблице «Туристы». Также такой выбор гарантирует правильность обновления и удаления связанных данных. Работа полученного приложения показана на рисунке 12.

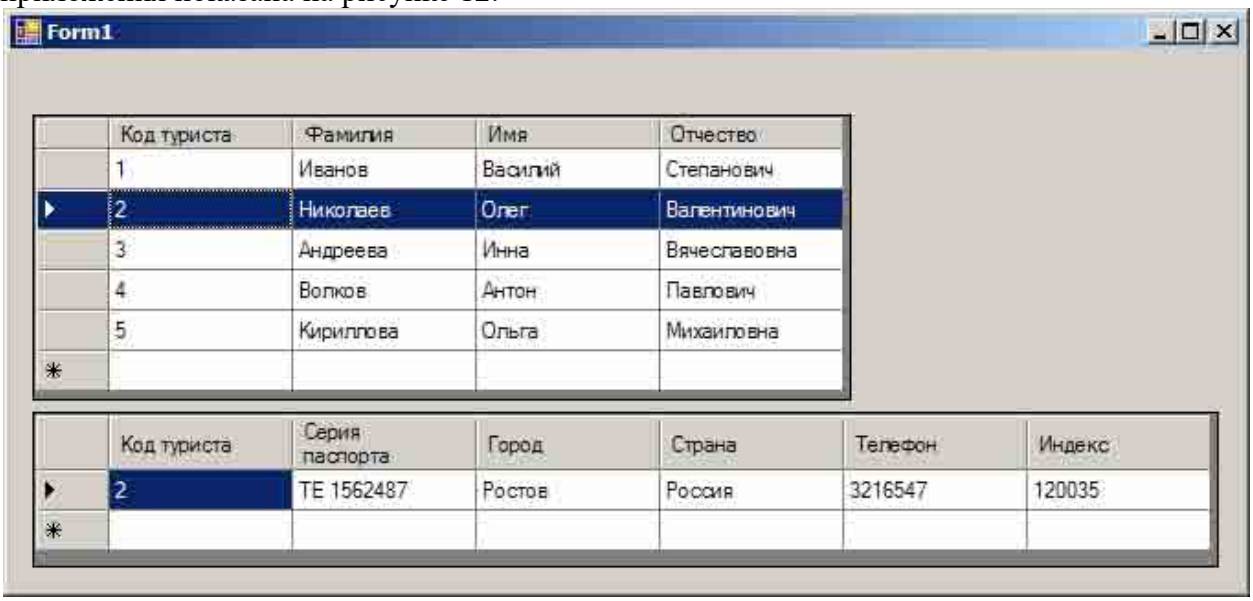


Рис. 12. Приложение базы данных в работе

Перемещение по данным при помощи стрелочных клавиш является неудобным. Для упрощения навигации по данным существует компонент BindingNavigator. Поместим его на форме (рис. 13).

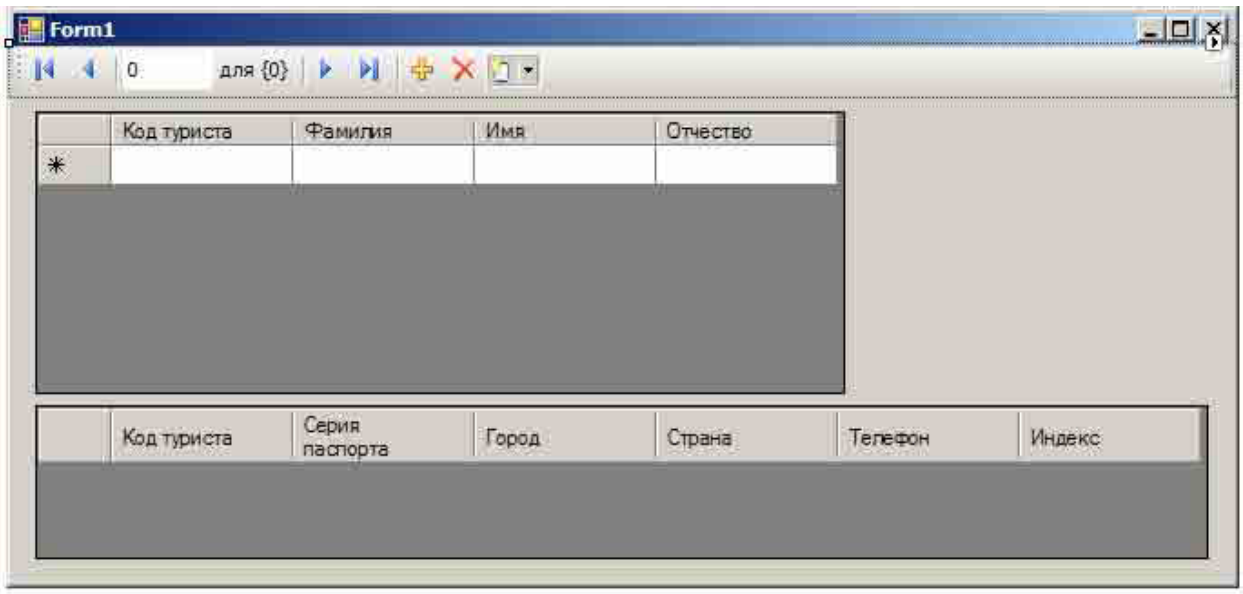


Рис. 13. Компонент BindingNavigator на форме

Данный компонент позволяет осуществлять навигацию между записями таблицы, добавлять и удалять строки таблицы. Возможности и внешний вид компонента можно настраивать, так как он представляет собой полосу меню ToolStripContainer.
Свойством, определяющим таблицу, по которой производится навигация, является свойство BindingSource. Установим значение этого свойства равным «туристыBindingSource». В работе компонент выглядит следующим образом
(рис. 14).

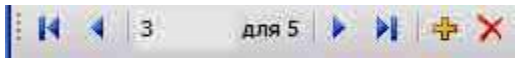


Рис. 14. Компонент BindingNavigator в работе

Редактирование данных в ячейках компонента DataGridView при соответствующих настройках возможно, но неудобно и не рационально. В частности, трудно проверять введенные значения на ошибки. Поэтому для таблицы «Туристы» сделаем экранную форму, позволяющую отображать данные в компонентах TextBox и редактировать их. Для этого разместим на форме контейнер типа Panel, а на нем три компонента TextBox следующим образом (рис. 15).

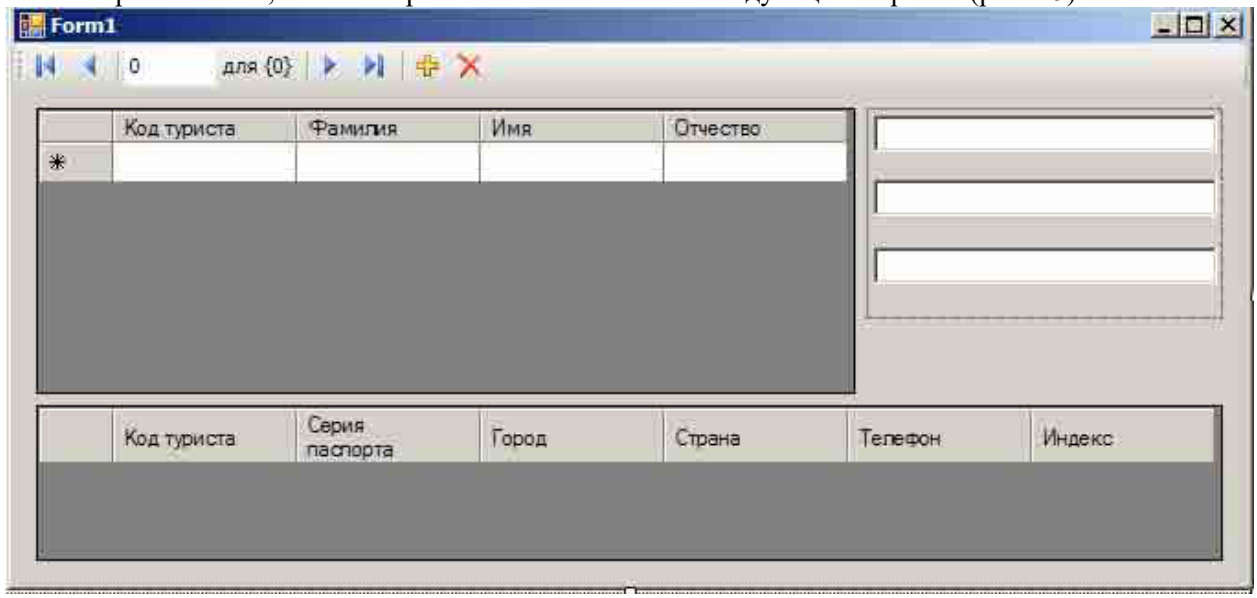


Рис. 15. Экранная панель для редактирования записей таблицы «Туристы»

Теперь необходимо осуществить привязку компонентов TextBox к соответствующим полям таблицы «Туристы». Для этого используем свойство из группы DataBindings – Advanced, показанное на рисунке 16.

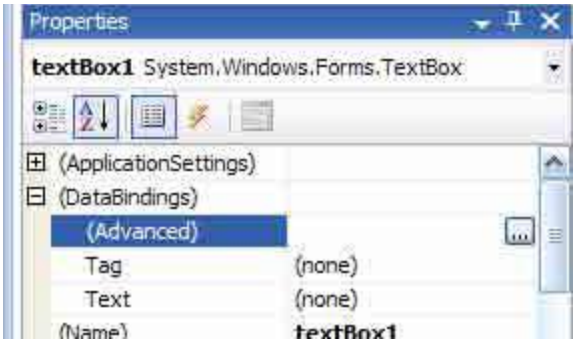


Рис. 16. Свойство «DataBindings – Advanced»

Выбор данного свойства приводит к появлению диалога, показанного на рисунке 17. Данный диалог позволяет осуществить не только привязку данных, но также задать событие, в рамках которого будет проводиться обновление данных, а также форматирование данных при их выводе.

Для верхнего компонента TextBox в выпадающем списке Binding выберем источником данных «туристыBindingSource» и поле источника – «Фамилия».

Для среднего и нижнего компонентов TextBox выберем тот же источник данных и поля «Имя» и «Отчество» соответственно.

Разработанное приложение в работе выглядит следующим образом (рис. 18).

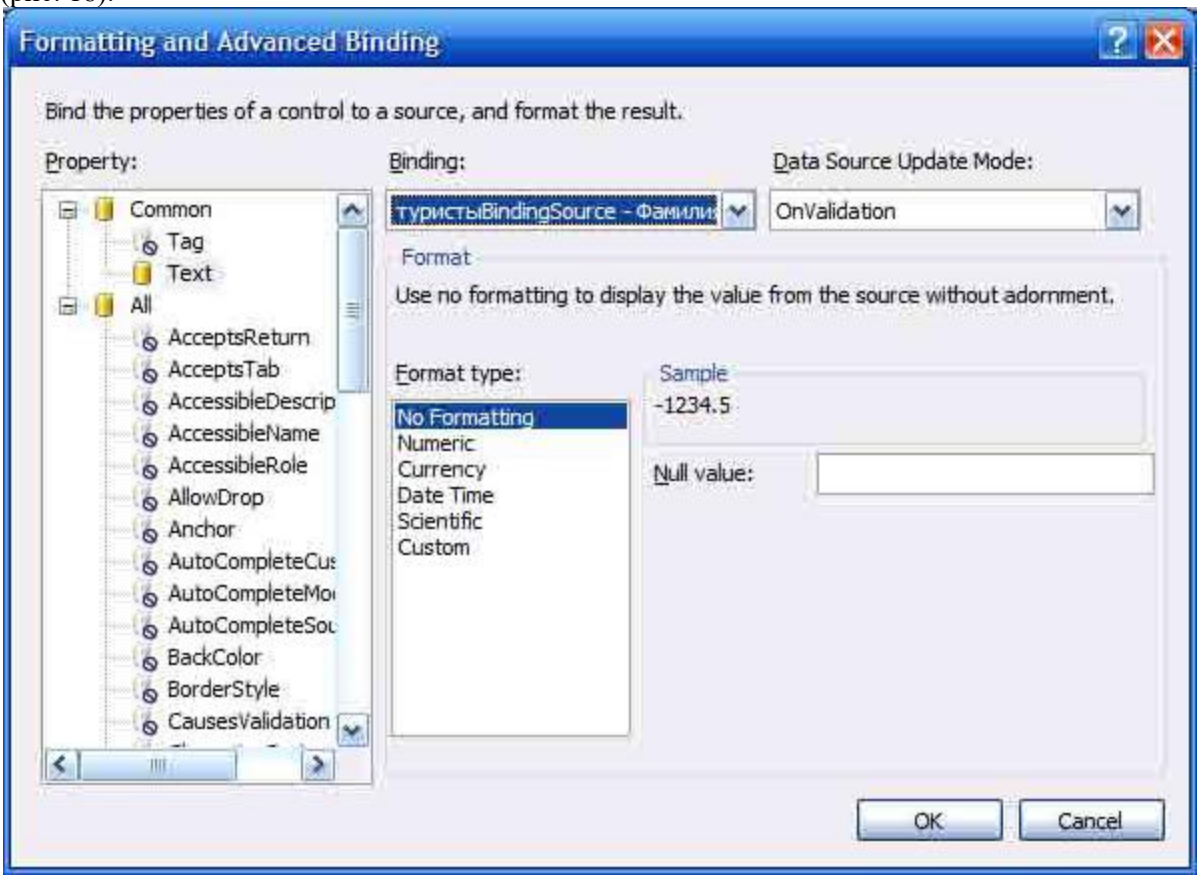


Рис. 17. Окно диалога для свойства «DataBindings – Advanced»

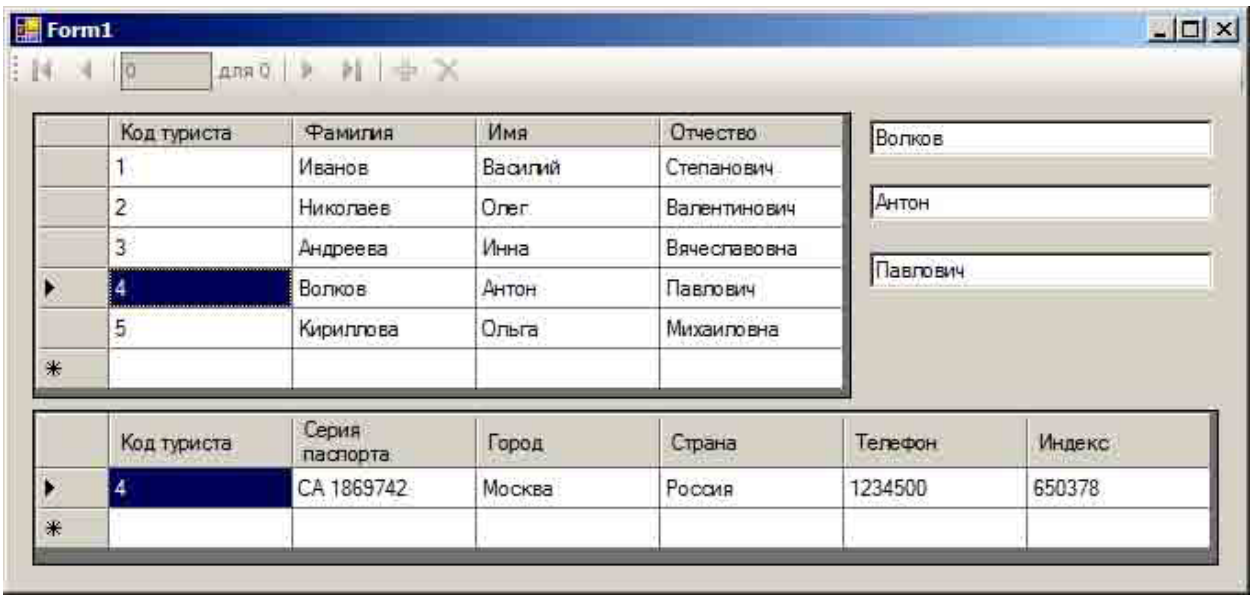


Рис. 18. Привязка данных к визуальным компонентам

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

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

1.4. Создание приложения БД «вручную»

Выше было рассмотрено создание приложений для работы с базами данных с использованием различных компонентов визуальной среды. Однако разрабатывать такие приложения можно также без использования визуальной среды.

Пример создания приложения БД «вручную»

Создадим в параллельно два приложения для работы с созданными ранее базами данных, аналогичные приложению, рассмотренному выше (см. п. 1.1).
Работа будет проводиться, соответственно, со следующими таблицами:
• Microsoft Access – BDTur_firm.mdb ;
• Microsoft SQL – BDTur_firmSQL.mdf .

Начнем с запуска Visual Studio и создания нового проекта Windows Application.

Размещаем на создавшейся форме элемент управления DataGridView, свойству Dock которого устанавливаем значение Fill. Переходим в код формы и подключаем соответствующие пространства имен:
• для MS Access – using System.Data.OleDb;
• для MS SQL – using System.Data.SqlClient;
В любом случае необходимо подключить пространство имен System.Data.
В конструкторе формы после InitializeComponent создаем объект DataAdapter.

В приложении, работающем с MS Access, соответствующий код будет выглядеть следующим образом:

```
public Form1()
{
//
// Required for Windows Form Designer support
//
InitializeComponent();
OleDbDataAdapter dataAdapter =
new OleDbDataAdapter(CommandText, ConnectionString);
}
```

А в приложении, работающем с MS SQL:

```
public Form1()
{
//
// Required for Windows Form Designer support
//
InitializeComponent();
SqlDataAdapter dataAdapter =
new SqlDataAdapter(CommandText, ConnectionString);
}
```

Как видно из приведенного кода, фрагменты отличаются только названиями объектов.
В качестве параметров DataAdapter передаются CommandText и ConnectionString. Переменная типа string CommandText представляет собой обычный SQL-запрос на выборку из таблицы «Туристы», а переменная типа ConnectionString – это так называемая строка подключения, в которой указываются расположение базы данных, ее название, параметры авторизации и пр.

Воспользуемся следующими строками подключения и командами:

```
// MS Access
CommandText: "SELECT [Код туриста], Фамилия, Имя, Отчество FROM Туристы";
ConnectionString: "Provider=Microsoft.Jet.OLEDB.4.0;Data Source="D:\ВМИ\For ADO\
BDTur_firm.mdb"

// MS SQL:
CommandText: "SELECT CustomerID, CompanyName, ContactName, ContactTitle, Address,
City, Region, PostalCode, Country, Phone, Fax FROM Customers"
ConnectionString: “Data Source=.\SQLEXPRESS;AttachDbFilename="D:\ВМИ\For ADO\
BDTur_firmSQL.mdf";Integrated Security=True;Connect Timeout=30;User Instance=True”
```

Обратите внимание на названия переменных CommandText и ConnectionString. Когда создается объект DataAdapter, в качестве параметров можно передать названия строк, таких как cmdText и conString, или даже cmt и cns –совершенно равноправно, не забыв, конечно же, назвать также эти переменные в классе Form1. Но сама среда Visual Studio генерирует эти строки именно с такими названиями – CommandText и ConnectionString, поэтому такое название переменных облегчает поддержку и сопровождение разработанного программного продукта.

Продолжим создание программы. Дальнейший код одинаков для обоих вариантов.

Создаем объект DataSet:

```
DataSet ds = new DataSet();
```

Заполняем таблицу «Туристы» объекта ds данными из базы:

```
dataAdapter.Fill(ds, "Туристы");
```

Связываем источник данных объекта dataGridView1 с таблицей «Туристы»
объекта ds:
dataGrid1.DataSource = ds.Tables["Туристы"].DefaultView;

Теперь запустим созданное приложение. Если все сделано правильно, то на экранной форме отобразится содержимое таблицы «Туристы».
Теперь изменим код следующим образом:
dataAdapter.Fill(ds, "Туристы2");
dataGridView1.DataSource = ds.Tables["Туристы2"].DefaultView;

Таблицы «Туристы2» в БД нет, однако код по-прежнему работает. Это связано с тем, что таблица, которую мы называем «Туристы», при вызове метода Fill объекта dataAdapter может быть названа как угодно – ее содержимое будет представлять собой извлекаемую таблицу из базы данных. При указании источника данных (DataSource) для объекта dataGridView1 мы ссылаемся именно на таблицу «Туристы», которая была создана при вызове метода Fill. Таким образом, при заполнении таблиц их можно называть произвольным образом. Однако для реальных проектов рекомендуется использовать настоящие названия таблиц, чтобы избежать путаницы и трудностей в сопровождении.

ЛАБОРАТОРНОЕ ЗАНЯТИЕ №5.
СРЕДСТВА АДМИНИСТРИРОВАНИЯ SQL SERVER

1. Средства администрирования SQL Server.
2. Утилита Enterprise Manager.
3. Утилита Query Analyzer.
4. Утилита Distributed Transaction Coordinator (DTC).
5. Утилита Data Transformation Services (DTS).
6. Система оперативной помощи Books Online.

Цель работы:

- приобрести знания и сформировать умения
- в администрировании SQL Server;
- в использовании утилит для управления сервером;
- в настройке параметров сервера.

Задание:

1. Настройте **параметры** работы утилиты **Enterprise Manager:**

- установите **интервал проверки** текущего состояния **SQL Server** – 20 секунд, **SQL Server Agent** – 30 секунд, **SQL Mail** – 15 секунд, **MS DistributedTransaction Coordinator** – 60 секунд;
- настройте **хранение информации** о зарегистрированных экземплярах SQL Server в **локальной** системе **реестра** Windows, сделайте ее доступной **другим** локальным и удаленным **пользователям**.

Опишите, как Вы это сделали. Укажите **назначение служб**, интервал проверки по которым Вы настраивали.

2. Задайте **дополнительные опции** SQL Server:

- укажите **время ожидания подключения** к SQL Server **равное 10 минутам**;
- **время ожидания выполнения запроса** к удаленному серверу должно выполняться **бесконечно долго;**
- настройте **Enterprise Manager** таким образом, чтобы при открытии утилиты **дерево**

объектов выводилось в том же виде, как и на момент закрытия утилиты.

Опишите, как Вы это сделали. Укажите назначение **дополнительных опций**.

3. Укажите, что регистрируется в журналах **SQL Server Logs**. Укажите как открыть **текущий журнал**. Приведите пример **содержания** журнала.

Объясните **назначение** представленных сообщений.

4. Используя утилиту **Data Transformation Services (DTS)**, экспортируйте записи таблицы **Employees** базы данных **Northwind** в **Txt-файл**.
Приведите **структуру** таблицы Employee и экспортированные данные, представьте результатный **Txt-файл**.

5. Создайте свою базу данных, используя утилиту **Enterprise Manager**. Присвойте ей **произвольное имя**. Опишите, как Вы это сделали.

6. Скопируйте таблицу **Products** базы данных **Northwind** с помощью утилиту **Data Transformation Services (DTS)** в таблицу **MyProduct** Вашей базы данных. Выберите только десять первых записей, используя запрос (**Query**) для спецификации данных. Опишите, как Вы это сделали.

7. Скопируйте **всю** базу данных **Northwind** в новую базу данных под именем **Northwind_xxxx**, где xxxx – номер Вашего студенческого билета. Воспользуйтесь для этого утилитой **Data Transformation Services (DTS).** Опишите, как Вы это сделали.

8. Почему **представления** базы данных **Northwind** превратились в **таблицы** Вашей базы данных? Приведите список этих представлений. Чем отличаются представления базы данных **Northwind** от одноименных таблиц Вашей базы данных? В чем их недостаток?

9. Для базы данных **Northwind** сгенерируйте **ER-диаграмму** (**диаграмму сущность-связь**) с помощью мастера **Create Database Diagram Wizard**. Используйте таблицы **Categories, Products, Order Details, Suppliers, Orders, Shippers, Customers** и **Employees.** Укажите, как Вы это сделали. Опишите созданные связи между таблицами. Какие для этого используются **внешние ключи**, по каким столбцам они построены?

10. Запустите утилиту **Query Analyzer**. Приведите перечень функций для работы с **датами и временем**. Воспользуйтесь системой оперативной помощи **Books Online**, укажите назначение и приведите примеры использования функций **DateAdd** и **DatePart**.

ЛАБОРАТОРНОЕ ЗАНЯТИЕ №6.
УПРАВЛЕНИЕ СЕРВЕРАМИ MS SQL SERVER

1. Управление серверами.
2. Регистрация серверов.
3. Учетные записи серверов и серверные роли.
4. Запуск, остановка и приостановка серверов.

Цель работы:
приобрести знания и сформировать умения

- в управлении серверами;
- регистрации серверов;
- об учетных записях серверов и серверных ролях;

– в запуске, остановке и приостановке серверов.

Задание:

1. Удалите регистрационную информацию о сервере **(LOCAL)**. Опишите, как Вы это сделали.

2. Создайте новую регистрационную информацию о Вашем сервере с помощью **Register SQL Server Wizard** или диалогового **окна SQL Server Properties**. Выберите любой из доступных серверов. Опишите, как Вы это сделали.

3. Создайте новую серверную группу **Server Group.** Присвойте ей имя Вашей студенческой группы **(например, DLI-201)**. Выберите для нее уровень **Sub-group of …** SQL Server Group. Опишите, как Вы это сделали.

4. Откройте окно редактирования свойств **регистрации Edit SQL Server Registration properties.** Присвойте **регистрационную запись сервера**Вашей серверной группе из **3 пункта** задания. Отключите опции **“Отображать состояние сервера на консоли”**, **“Показывать системные базы данных и системные объекты”.** Сохраните настройки. Опишите, как Вы это сделали. Что изменилось в настройках сервера?

5. Приостановите работу Вашего сервера, возобновите его работу, остановите Ваш сервер и запустите его снова с помощью **контекстного меню** Enterprise Manager. Опишите, как Вы это сделали.

6. Проделайте те же действия (**см. пункт 5**) с помощью утилиты **Service Manager.** Запустите службу **SQL Server Agent**. Настройте автоматический запуск **этой службы** при запуске **операционной системы**. Опишите, как Вы это сделали.

7. Остановите сервер с помощью инструкции языка Transact-SQL **SHUTDOWN.** Опишите, как Вы это сделали. Для чего применяется необязательное предложение **WITH NOWAIT.** Каковы преимущества и недостатки применения этой опции. Запустите сервер. Как Вы это сделали?

8. Укажите данные об **основных свойствах** SQL Server:
- наименование продукта;
- операционная система;
- версия продукта;
- язык;
- платформа;
- объем оперативной памяти;
- количество процессоров;
- корневая директория сервера;
- сопоставление сервера.

Укажите, какие **службы SQL Server** запускаются автоматически при старте операционной системы. Какие службы **не запускаются** автоматически?

9. Установите в **свойствах SQL Server** метод **аутентификации** на основе **учетных записей Windows**. Настройте регистрацию в журнале **ВСЕХ**попыток подключения к серверу. **Запретите** другим серверам **удаленно подключаться** к Вашему серверу. **Установите**, чтобы для **целостности распределенных запросов** использовалась служба **Distributed Transaction Coordinator**. Опишите, как Вы это сделали.

10. Установите в **свойствах SQL Server русский язык** для сообщений сервера по умолчанию. Опишите, как Вы это сделали. Определите, как интерпретируются **двухзначные номера** (от 0 до 99) для **представления года** в настройках сервера.

ЛАБОРАТОРНОЕ ЗАНЯТИЕ №7.
ХРАНИМЫЕ ПРОЦЕДУРЫ. ИХ НАЗНАЧЕНИЕ И ПРИМЕНЕНИЕ

1. Хранимые процедуры. Назначение и применение хранимых процедур.
2. Системные хранимые процедуры.
3. Создание, изменение и удаление хранимых процедур.
4. Входные и выходные параметры хранимых процедур.

Цель работы:
приобрести знания и сформировать умения

- в разработке хранимых процедур;
- в использовании системных хранимых процедур;
- в применении входных и выходных параметров для передачи значений в процедуры и для получения результатов их вычислений;
- в создании, изменении и удалении хранимых процедур.

Задание:

1. Создайте таблицу **Customer** в Вашей базе данных. Структура таблицы представлена ниже:

№№ п/п	Наименование столбца	Тип	Длина	Допустимо Null	Значение по умолчанию	Ограничение на значение
1.	CustId	int		Нет		> 0
2.	Name	char	30	Нет		<>’ ‘
3.	City	char	30	Нет	‘ ‘	
4.	State	char	2	Нет	‘ ‘	
5.	Country	char	30	Нет	‘ ‘	
6.	PhoneVoice	char	15	Нет	‘ ‘	
7.	PhoneFax	char	15	Нет	‘ ‘	
8.	Status	char	1	Нет	‘ ‘	
9.	Discount	decimal	7,3		0	>=0
10.	CreditLimit	money		Да	Null	Null или >=0
11.	EntryDateTime	datetime		Нет	Текущее дата-время	

Приведите команды языка DDL создания таблицы **Customer** с указанными характеристиками.

2. Заполните Вашу таблицу **Customer** с помощью инструкций **INSERT** языка DML следующими данными о клиентах:

CustId	Name	City	Discount
001	Smith Mfg.	Portland	0.500
002	Bolt Co.	Eugene	0.200
003	Ajax Inc.	Albany	NULL
004	Adopt	Portland	0.000
005	Bell Bldg.	Eugene	0.100
006	Floored	Seattle	0.000

007	Alpine Inc.	Seattle	0.100
008	Telex Co.	Albany	0.000
009	Nautilus	Portland	0.500
010	Adopt Mfg.	Seattle	0.000
011	Seaworthy	Albany	0.100
012	AA Products	Portland	0.100
013	Wood Bros.	Eugene	0.100

Остальные поля заполните произвольными значениями, а поле **Status** заполните **значением '0'** для **четных идентификаторов (CustId)** и '**1'** для **нечетных идентификаторов (CustId).**
Приведите тексты инструкций **INSERT** языка DML.

3. **Создайте** хранимую процедуру под именем **ListCustomer**, которая принимает в качестве **входного параметра** значение **минимальной скидки** с типом данных **DECIMAL(5,3)** и возвращает список клиентов со скидкой **большей или равной минимальной**. Приведите **текст хранимой процедуры**, **команду ее вызова** со значением входного параметра**, равным 0.15**, и **результат ее выполнения**.

4. **Измените** хранимую процедуру **ListCustomer** с помощью инструкции **ALTER.** Установите тип данных **входного параметра DEC(7,3).** Ограничьте список клиентов со скидкой **большей или равной минимальной** наличием у них **статуса (Status) равным '1'**. Приведите **текст изменения** хранимой процедуры, **команду ее вызова** со значением входного параметра**, равным 0.1**, и **результат ее выполнения**.

5. Переименуйте Вашу хранимую процедуру **ListCustomer** в хранимую процедуру **ListCustomerWithDiscount** с помощью системной хранимой процедуры **sp_rename.** Текст правильного вызова системной хранимой процедуры **sp_rename** посмотрите в системе помощи **Books Online.** Приведите текст инструкции для переименования**.**

6. Откомпилируйте Вашу хранимую процедуру **ListCustomerWithDiscount** с помощью системной хранимой процедуры **sp_recompile.** Текст правильного вызова системной хранимой процедуры **sp_recompile** посмотрите в системе помощи **Books Online.** Приведите текст инструкции для компиляции**.** Поясните**, когда произойдет компиляция** Вашей хранимой процедуры. Что на самом деле делает системная хранимая процедура **sp_recompile?**

7. Получите сведения о Вашей хранимой процедуре **ListCustomerWithDiscount** с помощью системных хранимых процедур:

- **sp_help**;
- **sp_helptext**;
- **sp_depends**.

Текст правильного вызова системных хранимых процедур посмотрите в системе помощи **Books Online.** Приведите тексты инструкций и результаты их выполнения.

8. **Измените** Вашу хранимую процедуру **ListCustomerWithDiscount** с помощью инструкции **ALTER.** Задайте **значение по умолчанию** для входного параметра **"минимальная скидка"** равную **0.001**. Приведите **текст изменения** хранимой процедуры, **команду ее вызова** без значения входного параметра и **результат ее выполнения**.

9. Напишите **хранимую процедуру GetCustDiscount**, имеющую **один входной и один выходной параметр**. Процедура должна возвращать значение **скидки (Discount)** из таблицы **Customer** для клиента с заданным **идентификационным номером (CustId)**. Приведите **текст** хранимой процедуры, **команду ее вызова** с конкретным значением входного

параметра и **результат ее выполнения в выходном параметре**.

10. Удалите хранимую процедуру **GetCustDiscount** с помощью команды языка DDL. Приведите текст инструкции для удаления.

Отчет.
В отчете **обязательно** должны присутствовать **выводы** по работе.

ЛАБОРАТОРНОЕ ЗАНЯТИЕ №8.
СИСТЕМА БЕЗОПАСНОСТИ SQL SERVER

1. Система безопасности SQL Server.
2. Основы системы безопасности.
3. Управление пользователями.
4. Управление доступом к базам данных.

Цель работы:
приобрести знания и сформировать умения

- в создании пользователей;
- во включении пользователей в фиксированные серверные роли;
- во включении пользователей в фиксированные роли баз данных;
- в назначении разрешений пользователям на доступ к объектам баз данных и выполнение хранимых процедур.

Задание:

1. Создайте пользователя под именем **sa_xxxx**, где **xxxx** – номер Вашего студенческого билета. Установите для него аутентификацию на уровне SQLServer. Задайте для него пароль и **запишите**, чтобы не забыть. Установите в качестве базы данных по умолчанию **Вашу базу данных** (см. Лабораторная №1, пункт 5). Выберите язык базы данных – **русский**. Укажите, что Ваш пользователь **sa_xxxx** является **системным администратором**. Запишите в отчете, какие действия может выполнять пользователь **sa_xxxx.** Сохраните данные о пользователе. Подтвердите пароль для пользователя. Опишите, как Вы это сделали.

2. Отключитесь от сервера. Отредактируйте свойства регистрационной записи Вашего сервера. Установите подключение с использованием аутентификации на уровне SQL Server. Установите - всегда запрашивать учетную запись и пароль при подключении. Подключитесь к серверу под именем Вашего пользователя **sa_xxxx.** Опишите, как Вы это сделали.
Что означает **sa** в имени Вашего пользователя?

3. Создайте пользователя под именем **User_xxxx**, где **xxxx** – номер Вашего студенческого билета. Установите для него аутентификацию на уровне SQLServer. Задайте для него пароль и **запишите**, чтобы не забыть. Установите в качестве базы данных по умолчанию **Northwind**. Выберите язык базы данных – **английский**. Сохраните данные о пользователе. Подтвердите пароль для пользователя. Опишите, как Вы это сделали.

4. Разрешите Вашему пользователю **User_xxxx** доступ к базе данных **Northwind**. Разрешите ему **считывать** данные из этой базы данных. Разрешите Вашему пользователю **User_xxxx** доступ к базе данных **Northwind_xxxx**. Разрешите ему **считывать** и **записывать** данные в эту базу данных.
Разрешите Вашему пользователю **User_xxxx** доступ к Вашей **собственной** базе данных (см. Лабораторная №1, пункт 5). Назначьте его **собственником**этой базы данных.
Запишите в отчете, какие действия может выполнять Ваш пользователь **User_xxxx.** Сохраните данные о пользователе. Опишите, как Вы это сделали.

5. Отключитесь от сервера. Подключитесь к серверу под именем Вашего пользователя **User_xxxx.** Откройте утилиту **Query Analyzer.** Сделайте текущей базу данных **Northwind.** Выведите информацию о служащих из таблицы **Employees:** Номер служащего, Фамилия служащего, Имя служащего, Год рождения служащего, Возраст служащего (на текущий момент времени). Измените **Год рождения** служащего **Andrew Fuller** на **1982** с помощью операторов языка DML. Опишите, как Вы это сделали. Приведите текст инструкций и результаты их выполнения. Все ли операции выполнились успешно? Объясните, почему.

6. Сделайте текущей базу данных **Northwind_xxxx.** Выведите информацию о клиентах из таблицы **Customers:** Идентификатор клиента, Наименование компании, Город, Страна, Телефон. Измените **Телефон компании Great Lakes Food Market** на номер (**820) 999-7667** с помощью операторов языка DML. Опишите, как Вы это сделали. Приведите текст инструкций и результаты их выполнения. Все ли операции выполнились успешно? Объясните, почему.

7. Создайте в базе данных **Northwind_xxxx** таблицу **Продаж (Sales).** Структура таблицы представлена ниже:

№№ п/п	Наименование столбца	Тип	Длина	Допустимо Null	Значение по умол-чанию	Ограни-чение на значение
1.	OrderId (№ заказа)	int		Нет		> 0
2.	CustId (№ клиента)	int		Нет		> 0
3.	SellerId (№ продавца)	int		Нет		> 0
4.	SaleDate (Дата продажи)	datetime		Нет	Текущее дата-время	
5.	SaleTotal (Сумма заказа)	Money		Да	Null	Null или >=0

Приведите команды языка DDL для создания таблицы **Sales** с указанными характеристиками. Опишите, как Вы это сделали. Приведите текст инструкций и результаты их выполнения. Все ли операции выполнились успешно? Объясните, почему.

8. Сделайте текущей **Вашу собственную базу данных** (см. Лабораторная №1, пункт 5). Создайте в **этой базе данных** таблицу **Продаж (Sales).**

Структура таблицы представлена в пункте 7. Приведите команды языка DDL для создания таблицы **Sales** с указанными характеристиками. Опишите, как Вы это сделали. Приведите текст инструкций и результаты их выполнения. Все ли операции выполнились успешно? Объясните, почему.

9. Отключитесь от сервера. Подключитесь к серверу под именем Вашего пользователя **sa_xxxx.** В **Enterprise Manager** выберите базу данных**Northwind_xxxx,** создайте в ней роль под именем **MyRole.** Установите тип роли – **стандартная роль**. Добавьте к этой роли Вашего пользователя **User_xxxx.** Сохраните роль. Опишите, как Вы это сделали.

10. Откройте для редактирования роль **MyRole**. Назначьте ей **разрешения**. Для таблиц **Categories, Products, Order Details, Suppliers, Orders, Shippers, Customers** и **Employees** н азначьте **разрешения на чтение данных**. Для таблиц **Products, Order Details, Orders, Customers** назначьте также **разрешения на изменение данных**. Для **всех хранимых процедур** назначьте **разрешения на выполнение**. Укажите в отчете **имена** этих **хранимых процедур**. Проверьте **объекты**, которые Вы назначили для

этой **роли**, переключив **режим просмотра**. Сохраните изменения. Опишите, как Вы это сделали.

ЛАБОРАТОРНОЕ ЗАНЯТИЕ №9
РЕЗЕРВНОЕ КОПИРОВАНИЕ И ВОССТАНОВЛЕНИЕ ДАННЫХ В MS SQL SERVER

1. Резервное копирование баз данных в MS SQL Server.
2. Восстановление данных в базах данных MS SQL Server.
3. Создание плана обслуживания базы данных с помощью мастера обслуживания **Database Maintenance Plan Wizard.**

Цель работы:
приобрести знания и сформировать умения

- в резервном копировании объектов баз данных;
- в выборе устройств резервного копирования;
- в проверке целостности баз данных;
- в создании плана обслуживания баз данных;
- в восстановлении поврежденной базы данных.

Задание:

1. Создайте резервную копию базы данных **Northwind_xxxx,** где **xxxx** – номер Вашего студенческого билета**.** Укажите имя резервной копии **MyNorthwindBackup**. Настройте **полное копирование** базы данных. Укажите, **куда** будет сохраняться база данных, укажите **имя файла**. Выберите **“переписать существующий архив”**. Опишите, как Вы это сделали.

2. Удалите из базы данных **Northwind_xxxx** с помощью команд языка DDL таблицы **Categories, Products, Order Details, Suppliers, Orders, Shippers, Customers** и **Employees.** Приведите тексты инструкций. Опишите, как Вы это сделали.

3. Восстановите содержимое базы данных **Northwind_xxxx** с помощью резервной копии **MyNorthwindBackup.** Опишите, как Вы это сделали.
Убедитесь в наличии таблиц **Categories, Products, Order Details, Suppliers, Orders, Shippers, Customers** и **Employees.** Укажите это в Вашем отчете.

4. Создайте копию базы данных **Northwind_xxxx** под именем **New_Northwind_xxxx** с помощью резервной копии **MyNorthwindBackup.** Опишите, как Вы это сделали.

5. Сделайте текущей базу данных **Northwind_xxxx.** Вызовите утилиту **Query Analyzer.** Удалите из таблицы **Customers** данные о клиентах с помощью команд языка **DML**. Приведите тексты инструкций. Опишите, как Вы это сделали.

6. Создайте дифференциальную резервную копию базы данных **Northwind_xxxx.** Укажите имя резервной копии **MyNorthwindBackup**. Настройте **дифференциальное копирование** базы данных. Укажите, **куда** будет сохраняться база данных, укажите **имя файла**. Выберите **“переписать существующий архив”**. Опишите, как Вы это сделали.

7. Удалите базу данных **New_Northwind_xxxx** с помощью команд языка **DDL.** Приведите тексты инструкций. Опишите, как Вы это сделали.

8. Восстановите содержимое базы данных **Northwind_xxxx** из полной копии **MyNorthwindBackup**. Убедитесь в наличии записей о клиентах в таблице **Customers**. Опишите, как Вы это сделали.

9. Создайте копию базы данных **Northwind_xxxx** под именем **New_Northwind_xxxx** с помощью полной резервной копии **MyNorthwindBackup.** Затем восстановите в базу данных **New_Northwind_xxxx** дифференциальную копию **MyNorthwindBackup**.

Опишите, как Вы это сделали. Укажите в отчете, есть ли в таблице **Customers** данные о клиентах.

10. Создайте план обслуживания базы данных **Northwind_xxxx** с помощью мастера обслуживания **Database Maintenance Plan Wizard.**

- Выберите, **удалить неиспользованное пространство** из файлов базы данных. Установите, **выполнять ежемесячно** в первое воскресенье каждого месяца, в 02:00:00. Выполнять до 31 декабря текущего года.
- Установите**, проверять целостность** базы данных.
- Укажите**,** создавать **страховую копию** на диске, проверять **целостность** страховой копии. Установите**, выполнять ежедневно** в 03:00:00. Выполнять до 28 декабря текущего года. Установите**, удалять файлы старше** 1 недели. Укажите расширение для файла страховой копии **.BAC.**
- Присвойте наименование файлу обслуживания базы данных: **Maintenance Plan Northwind_xxxx.**

Опишите, как Вы это сделали.

Отчет.

В отчете **обязательно** должны присутствовать **выводы** по работе.

ЛАБОРАТОРНОЕ ЗАНЯТИЕ №10.
ИНСТРУКЦИИ GRANT, REVOK, DENY В СИСТЕМЕ БЕЗОПАСНОСТИ. ИСПОЛЬЗОВАНИЕ ХРАНИМЫХ ПРОЦЕДУР В СИСТЕМЕ БЕЗОПАСНОСТИ

1. Система безопасности SQL Server.
2. Использование хранимых процедур.
3. Инструкции GRANT, REVOKE и DENY.

Цель работы:

приобрести знания и сформировать умения

- в использовании хранимых процедур в системе безопасности;
- в применении инструкции GRANT для предоставления прав доступа к объектам баз данных;
- в применении инструкции GRANT для предоставления прав на выполнение хранимых процедур баз данных;
- в применении инструкции REVOKE для отмены прав доступа к объектам баз данных и выполнение хранимых процедур баз данных;
- в использовании инструкции DENY для запрета предоставления прав доступа к объектам баз данных и исполнения хранимых процедур путем наследования прав.

Задание:

1. Определите имеющиеся на Вашем сервере **учетные записи пользователей** с помощью запроса к системной таблице **syslogins** базы данных **master.** Укажите учетные имена пользователей, базы данных по умолчанию, язык, и их фиксированные серверные роли. Воспользуйтесь системой оперативной помощи **Books OnLine**. Укажите текст запроса и результат его выполнения.

2. С помощью системной хранимой процедуры **sp_addlogin** создайте новую учетную запись пользователя с именем **sysadmin_xxxx,** где **xxxx** – номер Вашего студенческого билета.

Укажите для нее пароль по Вашему усмотрению, базу данных по умолчанию - **master**, язык - **русский**. Воспользуйтесь системой оперативной помощи **Books OnLine**. Укажите текст вызова этой процедуры.

3. С помощью системной хранимой процедуры **sp_addsrvrolemember** добавьте Вашу учетную запись **sysadmin_xxxx** к системной роли **sysadmin.**

Воспользуйтесь системой оперативной помощи **Books OnLine**. Укажите текст вызова этой процедуры.

4. Определите имеющиеся в Вашей базе **Northwind_xxxx пользователей Windows, пользователей MS SQL Server и роли MS SQL Server** с помощью запроса к системной таблице **sysusers. Укажите имена пользователей** или **ролей**, **имеют ли они доступ к базе данных, являются ли они пользователями SQL Server**, являются ли они **фиксированными ролями базы данных**. Воспользуйтесь системой оперативной помощи **Books OnLine**. Укажите текст запроса и результат его выполнения.

5. С помощью системной хранимой процедуры **sp_addlogin создайте новую учетную запись** пользователя с именем **user2_xxxx,** где **xxxx** – номер Вашего студенческого билета. Укажите для нее пароль по Вашему усмотрению, базу данных по умолчанию – **Northwind_xxxx**, язык - **русский**.

С помощью системной хранимой процедуры **sp_grantdbaccess** предоставьте этому пользователю **доступ к базе данных Northwind_xxxx.** Воспользуйтесь системой оперативной помощи **Books OnLine**. Укажите тексты вызовов обеих системных хранимых процедур.

6. С помощью системной хранимой процедуры **sp_addrolemember** предоставьте Вашему пользователю **user2_xxxx** возможность считывать данные из базы данных **Northwind_xxxx.** Для этого сделайте его **членом фиксированной роли** базы данных **db_datareader**. Воспользуйтесь системой оперативной помощи **Books OnLine**. Укажите текст вызова этой процедуры.

7. С помощью системной хранимой процедуры **sp_addrolemember** предоставьте Вашему пользователю **user2_xxxx** возможность изменять данные в базе данных **Northwind_xxxx.** Для этого сделайте его **членом фиксированной роли** базы данных **db_datawriter**. Воспользуйтесь системой оперативной помощи **Books OnLine**. Укажите текст вызова этой процедуры.

8. С помощью системной хранимой процедуры **sp_addrole** создайте новую пользовательскую роль **user_role.** Задайте для пользовательской роли **user_role** с помощью операторов **GRANT** разрешения на чтение записей в таблицах **Categories, Products, Order Details, Suppliers, Orders, Shippers,** а также разрешения на чтение, добавление, изменение и удаление записей в таблицах **Customers** и **Employees.** Воспользуйтесь системой оперативной помощи**Books OnLine**. Укажите тексты вызовов процедур и инструкций.

9. С помощью системной хранимой процедуры **sp_addlogin создайте новую учетную запись** пользователя с именем **user3_xxxx,** где **xxxx** – номер Вашего студенческого билета. Укажите для нее пароль по Вашему усмотрению, базу данных по умолчанию – **Northwind_xxxx**, язык - **русский**.

С помощью оператора **DENY** запретите пользователю **user3_xxxx** чтение записей в таблице **Categories.** С помощью системной хранимой процедуры **sp_addrolemember** сделайте Вашего пользователя **user3_xxxx членом** пользовательской роли **user_role.** Воспользуйтесь системой оперативной помощи**Books OnLine**. Укажите тексты вызовов процедур и инструкций.

10. Проверьте, может ли пользователь **user3_xxxx** считывать записи из таблиц **Categories** и **Products.** Проверьте, может ли он считывать, добавлять, изменять и удалять записи в таблицах **Customers** и **Employees**. Укажите, как Вы это сделали. Какие получились результаты?

Отмените разрешение роли **user_role** на чтение записей в таблице **Products** с помощью оператора **REVOKE.** Что изменилось для пользователя**user3_xxxx**? Как Вы это определили?

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Основные источники:

1. Агальцов В.П. Базы данных. Распределенные и удаленные базы данных: учебник. – М.: ФОРУМ: ИНФРА-М, 2017. -271с.
2. Баженова И.Ю. Основы проектирования приложений баз данных / И.Ю. Баженова. - 2-е изд., испр. - Москва: Национальный Открытый Университет «ИНТУИТ», 2016. - 238 с.: ил. - Библиогр. в кн. - ISBN 5-94774-539-9; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=428933.
3. Базы данных в высокопроизводительных информационных системах: учебное пособие / авт.-сост. Е.И. Николаев; Министерство образования и науки РФ, Федеральное государственное автономное образовательное учреждение высшего образования «Северо-Кавказский федеральный университет». - Ставрополь: СКФУ, 2016. - 163 с.: ил. - Библиогр.: с.161.; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=466799.
4. Кузнецов С. Введение в реляционные базы данных / С. Кузнецов. - 2-е изд., исправ. - Москва: Национальный Открытый Университет «ИНТУИТ», 2016. - 248 с.: ил. - (Основы информационных технологий). - Библиогр. в кн.; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=429088.

Дополнительные источники:

1. Баженова И.Ю. SQL и процедурно-ориентированные языки / И.Ю. Баженова. - 2-е изд., испр. - Москва: Национальный Открытый Университет «ИНТУИТ», 2016. - 167 с.: ил. - Библиогр. в кн. - ISBN 5-94774-539-9; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=428934.
2. Гуров В.В. Архитектура и организация ЭВМ / В.В. Гуров, В.О. Чуканов. - 2-е изд., испр. - Москва: Национальный Открытый Университет «ИНТУИТ», 2016. - 184 с.: ил., схем. - (Основы информационных технологий). - Библиогр. в кн. - ISBN 5-9556-0040-X; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=429021.
3. Ковалев Д.В. Информационная безопасность : учебное пособие / Д.В. Ковалев, Е.А. Богданова; Министерство образования и науки РФ, Южный федеральный университет. - Ростов-на-Дону: Издательство Южного федерального университета, 2016. - 74 с.: схем., табл., ил. - Библиогр. в кн. - ISBN 978-5-9275-2364-1; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=493175.
4. Кузнецов С. Введение в модель данных SQL: курс / С. Кузнецов. - 2-е изд., исправ. - Москва: Национальный Открытый Университет «ИНТУИТ», 2016. - 351 с.: ил. - (Основы информационных технологий). - Библиогр. в кн. - ISBN 5-9556-00028-0; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=429087.
5. Кумскова И.А. Базы данных: учебник для СПО / И. А. Кумскова.- М.: КНОРУС, 2016.-488 с.
6. Мартишин С.А. Проектирование и реализация баз данных в СУБД MySQL с использование MySQL Workbench: учебное пособие. – М.: ФОРУМ: ИНФРА-М, 2017. – 160 с.
7. Мельников В.П., Схиртладзе А.Г. Методы и средства хранения и защиты компьютерной информации: Учебник. – Старый Оскол: ТНТ, 2016. - 400 с.
8. Семенов Ю.А. Алгоритмы телекоммуникационных сетей: учебное пособие в 3-х ч. Часть 1. - М.: Интернет- Университет Информационных технологий; БИНОМ Лаборатория знаний 2016. - 511с.
9. Семенов Ю.А. Алгоритмы телекоммуникационных сетей: учебное пособие в 3-х ч. Часть 2 -М.: Интернет- Университет Информационных технологий; БИНОМ Лаборатория знаний 2016. - 829с.

10. Семенов Ю.А. Алгоритмы телекоммуникационных сетей: учебное пособие в 3-х ч. Часть 3 -М.: Интернет- Университет Информационных технологий; БИНОМ Лаборатория знаний 2016. - 637 с.
11. Системы управления базами данных: лабораторный практикум / сост. Д.Л. Осипов, М.Г. Огур; Министерство образования и науки РФ, Федеральное государственное автономное образовательное учреждение высшего образования «Северо-Кавказский федеральный университет». - Ставрополь: СКФУ, 2017. - 148 с. - Библиогр. в кн.; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=483760.

Периодические издания:

1. Компоненты и технологии. ООО Издательство «Файнстрит»;
2. Проблемы информатики. Издательство «Федеральное государственное бюджетное учреждение науки Институт вычислительной математики и математической геофизики Сибирского отделения Российской академии наук»;
3. Проблемы информационной безопасности. Компьютерные системы. Издательство «Федеральное государственное автономное образовательное учреждение высшего образования Санкт-Петербургский политехнический университет Петра Великого»;
4. Linux Format: главное в мире Linux / ред. К. Степанов - Санкт-Петербург: Мезон.Ру; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=238521;
5. Системный администратор: ежемесячный журнал / изд. ООО «Синдикат 13»; гл. ред. Г. Положевец - Москва: Синдикат 13, - ISSN 1813-5579; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=430336;
6. Информационно-управляющие системы: научный журнал / гл. ред. М.Б. Сергеев; изд. Санкт-Петербургский государственный университет аэрокосмического приборостроения; учред. ООО «Информационно-управляющие системы» - Санкт-Петербург: Санкт-Петербургский государственный университет аэрокосмического приборостроения - ISSN 1684-8853; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=494277;
7. Прикладная информатика : научно-практический журнал / гл. ред. А.А. Емельянов - Москва : Университет «Синергия» - ISSN 1993-8314; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=495388;
8. Прикладная информатика: Университет «Синергия»;
9. Компоненты и технологии: Медиа КиТ.

Интернет-ресурсы:

1. Компьютерные книги. Режим доступа: [http://computers.plib.ru/programming/Books.VBasic6/index.html 09.04.2020].
2. On-line библиотека свободно доступных материалов по информационным технологиям. Режим доступа: [http://digitland.ru 09.04.2020].
3. Открытые системы. Режим доступа: [http://www.osp.ru 09.04.2020];
4. ComputerBild. Режим доступа: [http://www.computerbild.ru/ 09.04.2020];
5. Мир ПК. Режим доступа: [http://www.pcworld.ru/ 09.04.2020];
6. Мобильные компьютеры. Режим доступа: http://www.mconline.ru/ 09.04.2020];
7. Компьютерра. Режим доступа: [http://www.computerra.ru/ 09.04.2020].
8. Базы данных. В 2-х кн. Кн. 2. Распределенные и удаленные базы данных. Режим доступа: [http://znanium.com/catalog.php?bookin./ 09.04.2020].