Введение в ADO.NET

- Объектная модель ADO.NET
- Создание приложений для работы с базами данных
- Хранение и извлечение данных с помощью ADO.NET

Объектная модель

ADO.NET

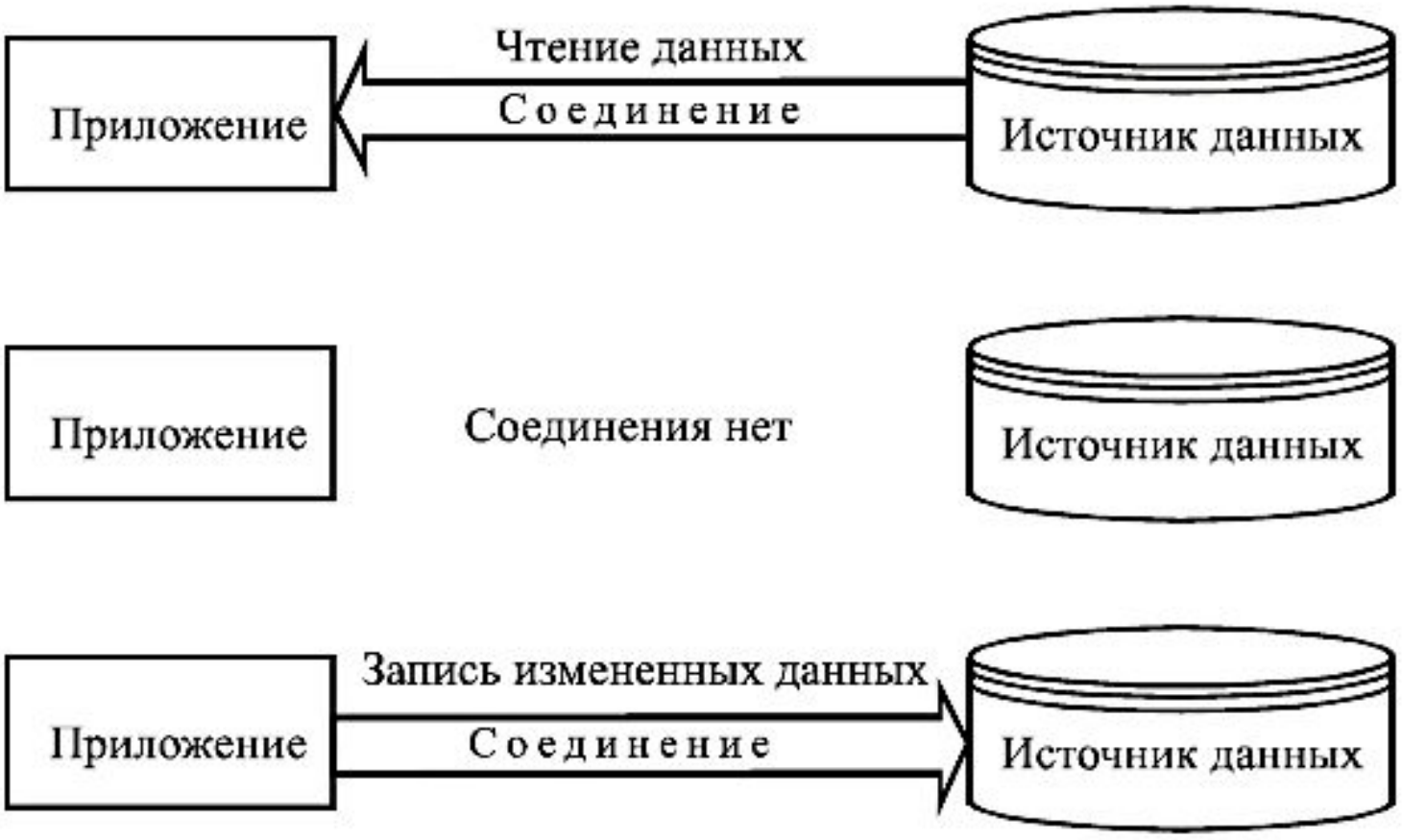
ADO.NET – это технология доступа к данным или библиотека пространства System.Data, где размещены используемые при доступе к данным классы, интерфейсы, делегаты и перечисления.

System.Data

System.Npgsql – провайдер от Postgresql

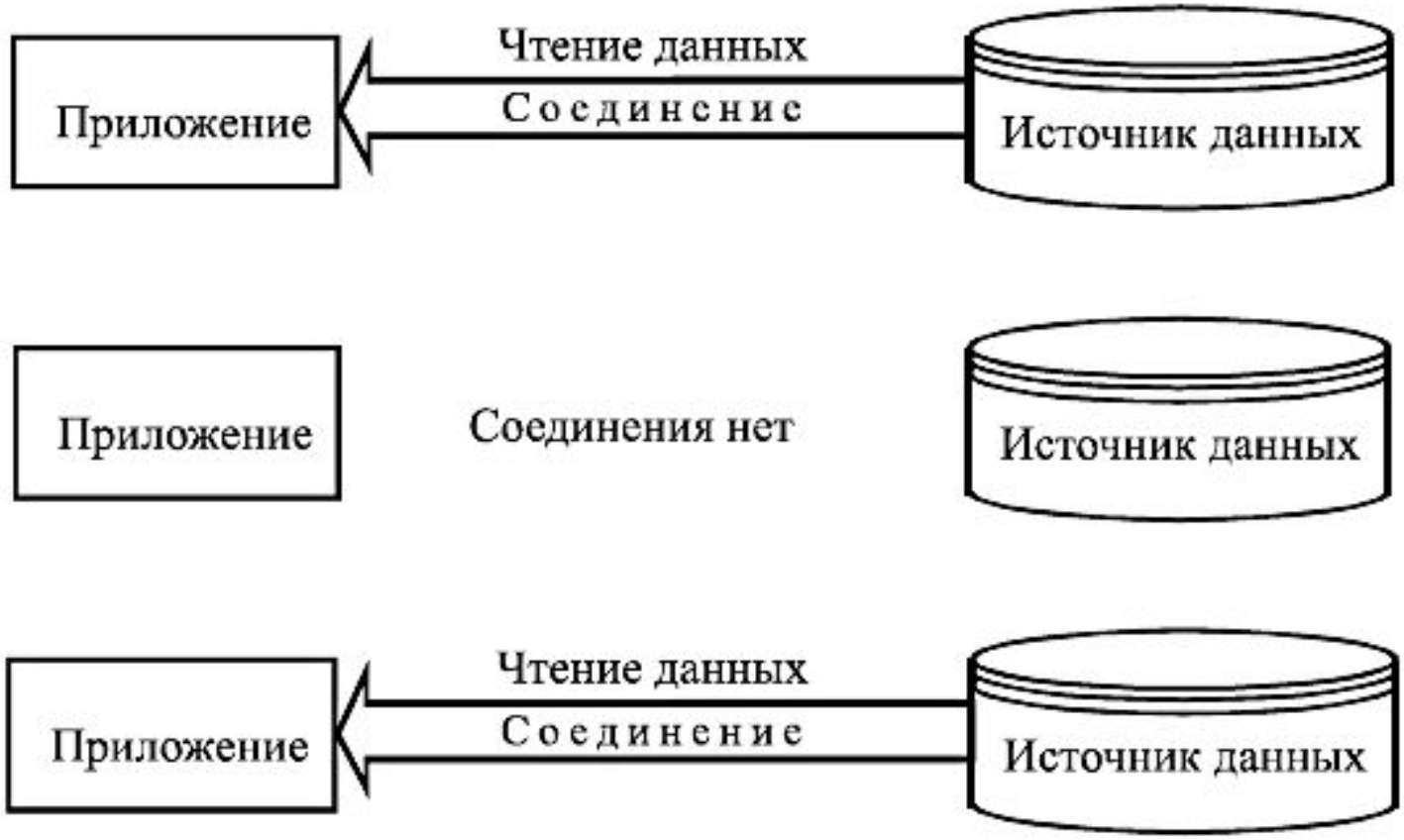
Объектная модель ADO.NET

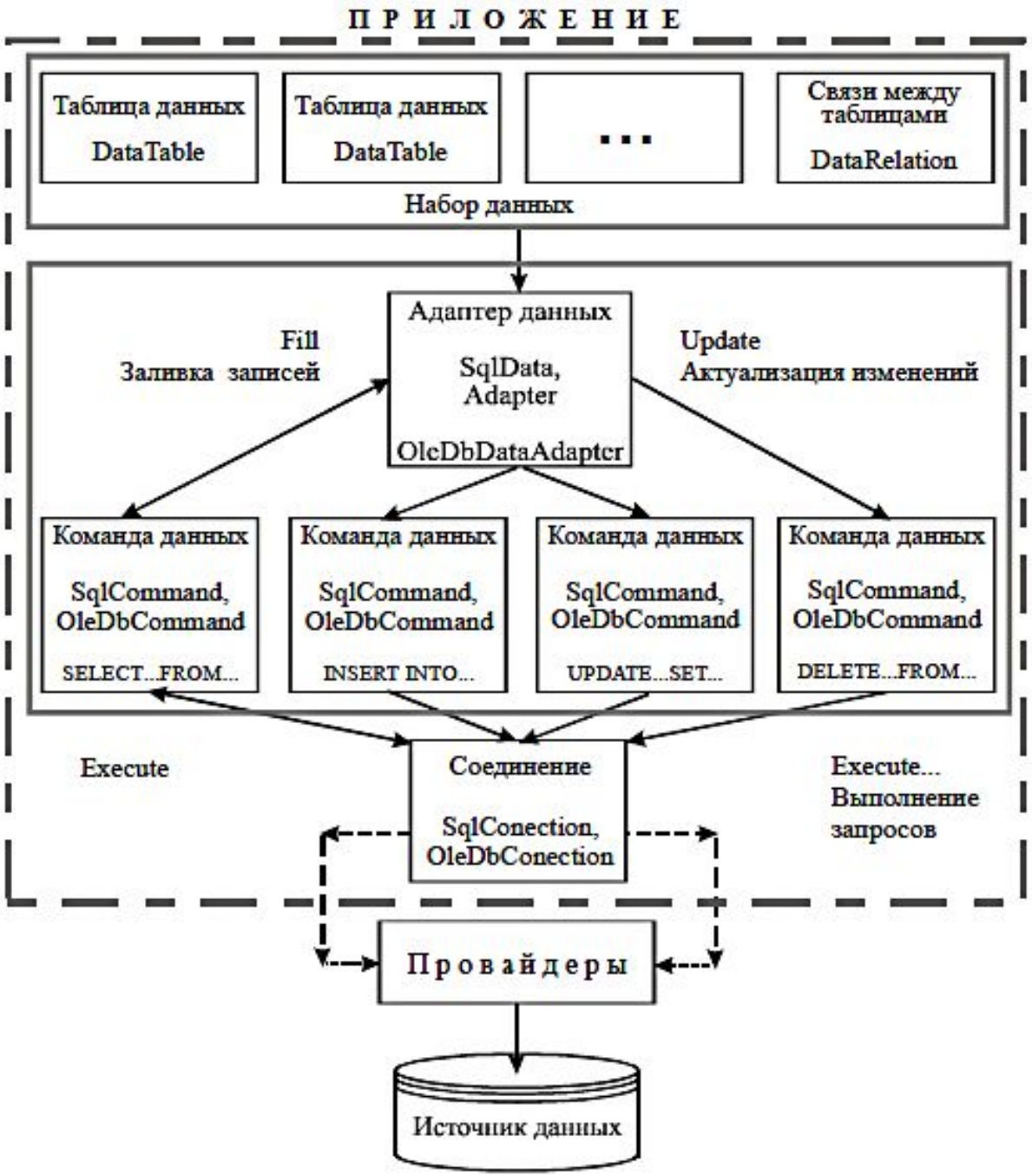
1. Отсоединенный режим с двухсторонним обменом данными.

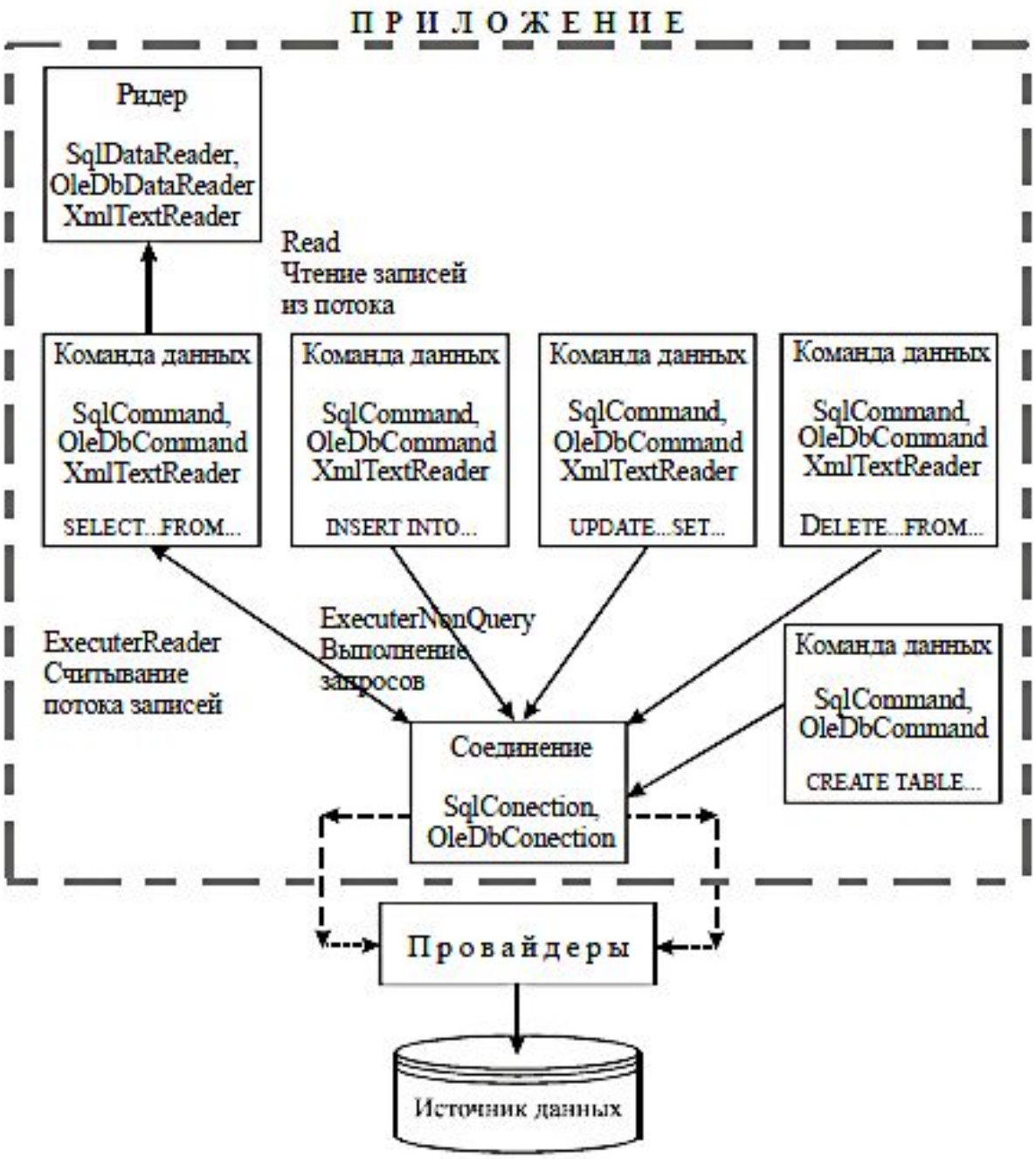


Объектная модель ADO.NET

2. Режим однонаправленного использования данных «только-на-чтение»




ПРИЛОЖЕНИЕ
Таблица данных
DataTable
Таблица данных
DataTable
...
Связи между таблицами
DataRelation
Набор данных
Адаптер данных
SqlData, Adapter
OleDbDataAdapter
Fill
Заливка записей
Update
Актуализация изменений
Команда данных
SqlCommand, OleDbCommand
SELECT...FROM...
Команда данных
SqlCommand, OleDbCommand
INSERT INTO...
Команда данных
SqlCommand, OleDbCommand
UPDATE...SET...
Команда данных
SqlCommand, OleDbCommand
DELETE...FROM...
Execute
Соединение
SqlConection, OleDbConection
Execute...
Выполнение запросов
Провайдеры
Источник данных


ПРИЛОЖЕНИЕ
Ридер
SqlDataReader,
OleDbDataReader
XmlTextReader
Read
Чтение записей
из потока
Команда данных
SqlCommand,
OleDbCommand
XmlTextReader
SELECT...FROM...
Команда данных
SqlCommand,
OleDbCommand
XmlTextReader
INSERT INTO...
Команда данных
SqlCommand,
OleDbCommand
XmlTextReader
UPDATE...SET...
Команда данных
SqlCommand,
OleDbCommand
XmlTextReader
DELETE...FROM...
ExecuterReader
Считывание
потока записей
ExecuterNonQuery
Выполнение
запросов
Команда данных
SqlCommand,
OleDbCommand
CREATE TABLE...
Соединение
SqlConection,
OleDbConection
Провайдеры
Источник данных

Объектная модель

ADO.NET

DataSet – элемент для хранения данных

- **Tables – таблицы данных (DataTable)**
- **Relations – отношения между таблицами из DataSet (DataRelation)**

Объектная модель

ADO.NET

DataTable – таблицы данных
Columns – столбцы (DataColumn)
Rows – строки (DataRow)

System.Data.DataView – реализуют просмотры данных, отсортированные некоторым образом

Создание приложений баз данных

Организация работы с базой данных:

1. Установка соединения с базой данных.
2. Заполнение объекта DataSet.
3. Считывание из DataSet.

Создание приложений баз данных

Установка соединения с базой данных:

```
//строка для подключения к базе postgres
string  connStr = "server=имя_сервера;
user=имя_пользователя; password=пароль;
database=имя_бд";

//создаем объект подключения
NpgsqlConnection  connection = new
NpgsqlConnection(connStr);
```

Создание приложений баз данных

Заполнение набора данных:

```
//SQL команда:
string  command = "SELECT  * FROM  table_name;";

//создаем экземпляр адаптера:
NpgsqlDataAdapter adapter =
    new NpgsqlDataAdapter(command, connection);
//создаем объект DataSet
DataSet  dataSet = new DataSet();

connection.Open();     //открываем соединение
adapter.Fill(ds, "tabl_name");  //заполняем DataSet
connection.Close();     //закрываем соединение
```

Создание приложений баз данных

Работа с БД без использования DataSet:

```
string commStr = "UPDATE * FROM … ";
//создаем экземпляр SqlCommand:
NpgSqlCommand command =
        new SqlCommand(commStr, connection);

connection.Open();
//выполняем SQL-команду:
int  ch = command.ExecuteNonQuery();
connection.Close();      //закрываем соединение
```

Работа с объектом DataGridView

Заполнение элемента таблицы

```
for (int i = 0; i < myDataSet.Tables["tabl"].Rows.Count; i++)
{
   dataGridView1.Rows.Add();
   dataGridView1.Rows[i].Cells["Exam_id"].Value =
         myDataSet.Tables["tabl"].Rows[i]["exam_id"].ToString();
}
```

dataGridView1.Rows.**Add()** – метод, добавляющий пустую строку в таблицу

dataGridView1.Rows[i] – обращаемся к i-ой строке таблицы

dataGridView1.Rows[i].Cells["Exam_id"] .**Value** – значение в ячейке таблицы на пересечении i-ой строки и столбца «Exam_id»

Работа с объектом DataGridView

Элемент управления **DataGridView** позволяет отображать и редактировать табличные данные из различных типов источников данных.

Можно сделать привязку с помощью объекта **DataSource** (источник данных).

```
dataGridView1.DataSource = ds.Tables["Student"];
```

Основные события объекта DataGridView

CellValidating – Проверка корректности ввода значения ячейки и запрет ввода недопустимого значения.

DataError – Реакция на необработанную ошибку при работе с данными.

CellValueChanged – Установка признака изменения данных, управление доступностью функций, связанных с наличием несохранённых изменений.

Leave – Отмена редактирования ячейки, если её значение некорректно, и вывод сообщения об ошибке.

CellMouseClick – Вызов контекстного меню редактируемой таблицы.

CellMouseDoubleClick – Вызов формы выбора значения ячейки, например цвета.

Пример работы с БД Postgresql

```
NpgsqlConnection conn = new NpgsqlConnection("Server=127.0.0.1;Port=5432;User Id=postgres;Password=secret;Database=university;Encoding=unicode");

NpgsqlCommand command = new NpgsqlCommand("select * from student", conn);
try
   {
      NpgsqlDataReader dr = command.ExecuteReader();
      while (dr.Read())
      {
         for (int i = 0; i < dr.FieldCount; i++)
         {
            Console.Write("{0} \t", dr[i]);
         }
         Console.WriteLine();
      }
   }
catch { }
```

Объекты: DataTable, DataRow, DataColumn

DataTable – представляет одну таблицу с данными в памяти (из DataSet).
Сборка: System.Data (в System.Data.dll)

Конструкторы:
DataTable() – инициализирует новый экземпляр класса
DataTable(“Tabl_name”) – инициализирует новый экземпляр класса с указанным именем таблицы.

Объекты: DataTable, DataRow, DataColumn

DataTable позволяет «укорачивать» код:

```
DataSet  ds;
string  zn =
ds.Tables["tbl_nm"].Rows[row_index]["col_nm"].ToString()

DataTable  dt = ds.Tables["tbl_nm"];
string  zn = dt.Rows[row_index]["col_nm"].ToString()
```

Объекты: DataTable, DataRow, DataColumn

Свойства DataTable:

Columns – коллекция столбцов, принадлежащих данной таблице.
DataSet – класс DataSet, к которому принадлежит данная таблица.
PrimaryKey – возвращает или задает массив столбцов, которые являются столбцами первичного ключа для таблицы данных.
Rows – получает коллекцию строк, принадлежащих данной таблице.
TableName – возвращает или задает имя DataTable.

Объекты: DataTable, DataRow, DataColumn

Методы DataTable:

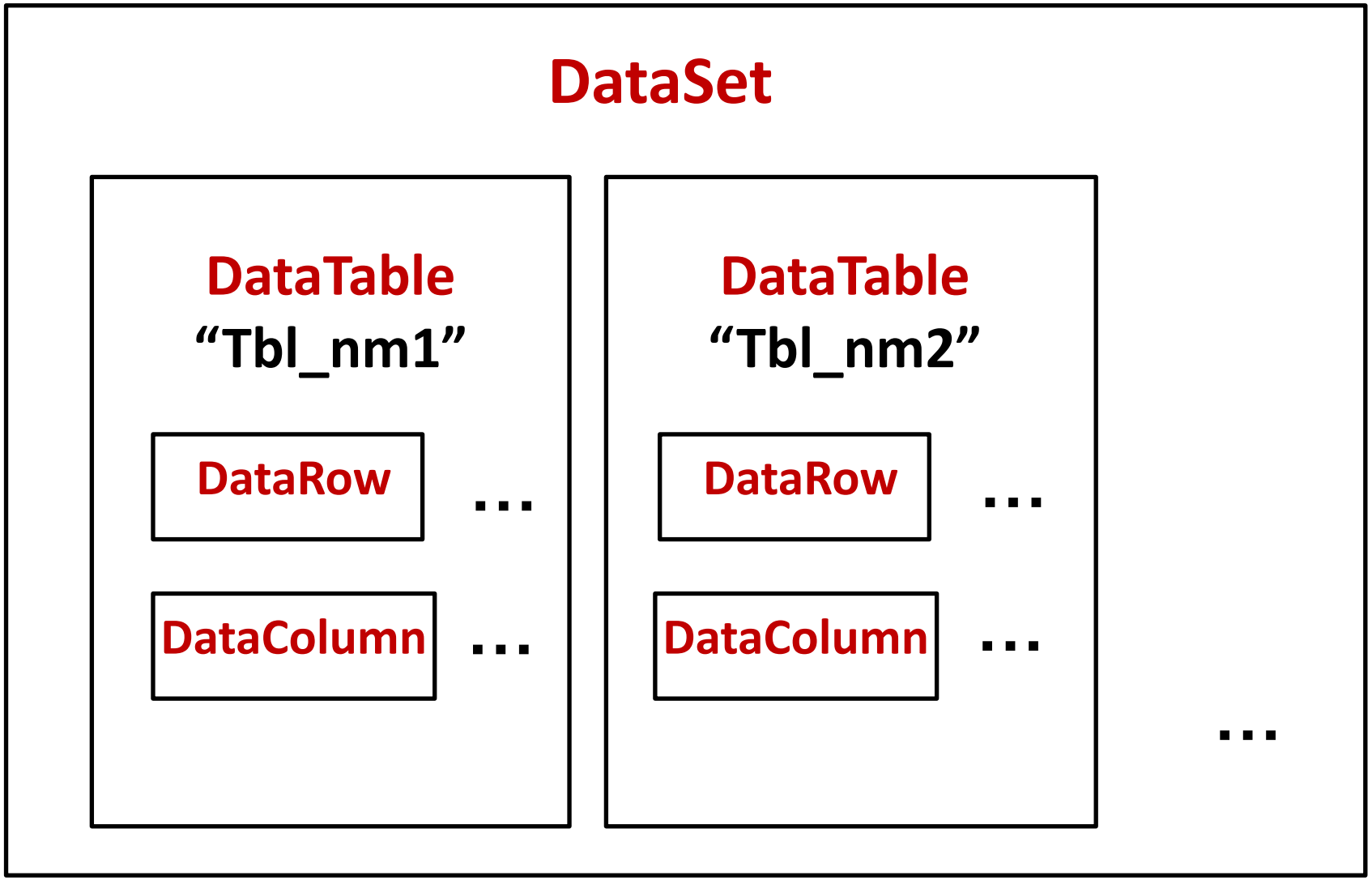
AcceptChanges – фиксирует все изменения, внесенные в таблицу после последнего вызова метода AcceptChanges.
BeginInit – начинает инициализацию класса DataTable, используемого в форме или другим компонентом. Инициализация происходит во время выполнения.
Clear() – очищает DataTable от всех данных.
Copy() – копирует структуру и данные для DataTable.
EndInit – завершает инициализацию объекта DataTable, используемого в форме или другим

Объекты: DataTable, DataRow, DataColumn

DataColumn – Получает коллекцию столбцов, принадлежащих данной таблице.

DataRow – Получает коллекцию строк, принадлежащих данной таблице.

Объекты: DataTable, DataRow, DataColumn



Объекты: DataTable, DataRow, DataColumn

```
//выводим данные таблицы в строку
private void  PrintValues(DataTable  table)
{
   foreach(DataRow  row  in  table.Rows)
   {
      foreach(DataColumn  column  in  table.Columns)
      {
         Label1.Text += row[column];
      }
   }
}
```

Объекты: DataTable, DataRow, DataColumn

```
//добавляем строку в таблицу DataTable
private void AddRow(DataSet  dataSet)
{
    DataTable  table;
    table = dataSet.Tables[“Student”];
    //с помощью метода NewRow создаем объект DataRow
    DataRow  newRow = table.NewRow();

    // Присваиваем значение:
    newRow[“Student_id”] = “163”;
    newRow[“Surname”] = “Кузьмин”;

    // Добавляем строку в таблицу:
    table.Rows.Add(newRow);
}
```

Метод DataTable.Select

DataTable.**Select**(условие фильтра) – возвращает массив всех объектов DataRow, отвечающих условиям фильтра.

При создании фильтра используются те же правила, которые применяются при записи предикатов в SQL-запросах.
Ключевое слово Where не пишут! Значение условия всегда заключают в одинарные кавычки.

```
DataTable  dt = ds.Tables[0];  //новый экземпляр DataTable
string  filtr = " City = 'Москва' ";  //условие фильтра
DataRow[] dr;
```

Метод DataTable.Select

```
//Выбрать из таблицы записи о студентах с именем Иван
private void GetRowsByFilter()
{
   //создаем экземпляр таблицы Student из DataSet
   DataTable  tbl = ds.Tables["Student"];

   //применяем фильтр к строкам таблицы
   DataRow[]  dr = tbl.Select("name = 'Иван'");

   //выводим строки
   for(int i = 0; i < dr.Length; i ++)
       Label1.Text += dr[i]["surname"].ToString() + "  " +
                          dr[i]["name"].ToString() + ", " ;
}
```

Метод

DataTable.Select

Перегрузки:

Select() – Получает массив всех объектов DataRow.

Select(filter) – Получает массив всех объектов DataRow, отвечающих условиям фильтра.

Select(filter, sort) – Получает массив всех объектов DataRow, отвечающих условиям фильтра, согласно указанному порядку сортировки.

Пример работы с набором данных

Вывести данные о студентах, получающих стипендию.

Решение:

```
protected void Page_Load(object sender, EventArgs e)
{
    //выполняем только при первой загрузке страницы
    if (Page.IsPostBack)
    {
        string comm = "select * from student";
        string connstr = @"server='имя_сервера'; uid='логин'; pwd='пароль'; database=имя_бд";
        SqlConnection conn = new SqlConnection(connstr);
        //наводим мосты
        SqlDataAdapter adapt = new SqlDataAdapter(comm, conn);
        //создаем датасет
        DataSet ds = new DataSet();
        //заполняем датасет
        adapt.Fill(ds);

        //сохраняем данные в кеш
        Cache["ds_stud"] = ds;
    }
}
```

Пример работы с набором данных

```
protected void Button1_Click(object sender, EventArgs e)
{
    //заносим данные в экземпляр датасет из кеша
    DataSet ds = (DataSet)Cache["ds_stud"];

    DataTable dt = ds.Tables["students"]; //создаем экземпляр таблицы датасет
    DataRow[] dr = dt.Select("stipend > '0'"); //применяем фильтр по стипендии

    //выводим на экран фамилии, имена и размер стипендии студентов
    foreach (DataRow row in dr)
        Label1.Text += row["surname"] + " " + row["name"] + " " + row["stipend"] + "<br />";
}
```

Лабораторная работа 7

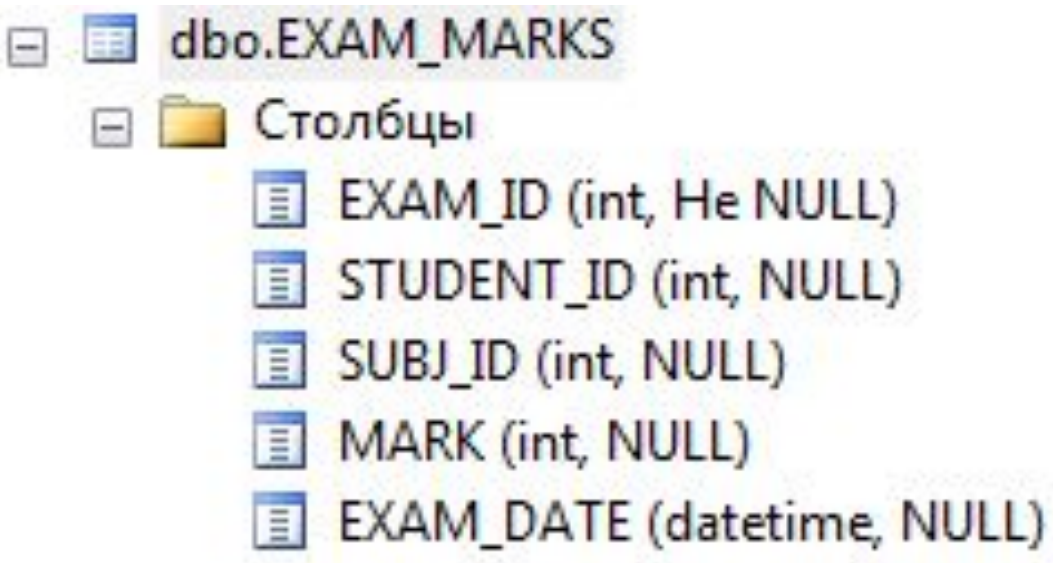
1. Из таблицы Lectures необходимо отобразить данные о преподавателях по городу, в котором они проживают. Выбор города предоставлять пользователям.
2. При выборе фамилии студента из списка отобразить текстовых полях следующую информацию по этому студенту:
   - идентификационный номер,
   - курс,
   - место проживания,
   - дата рождения.
3. Создать пользовательскую форму для добавления студентов (в таблицу Student).

Описание базы данных University

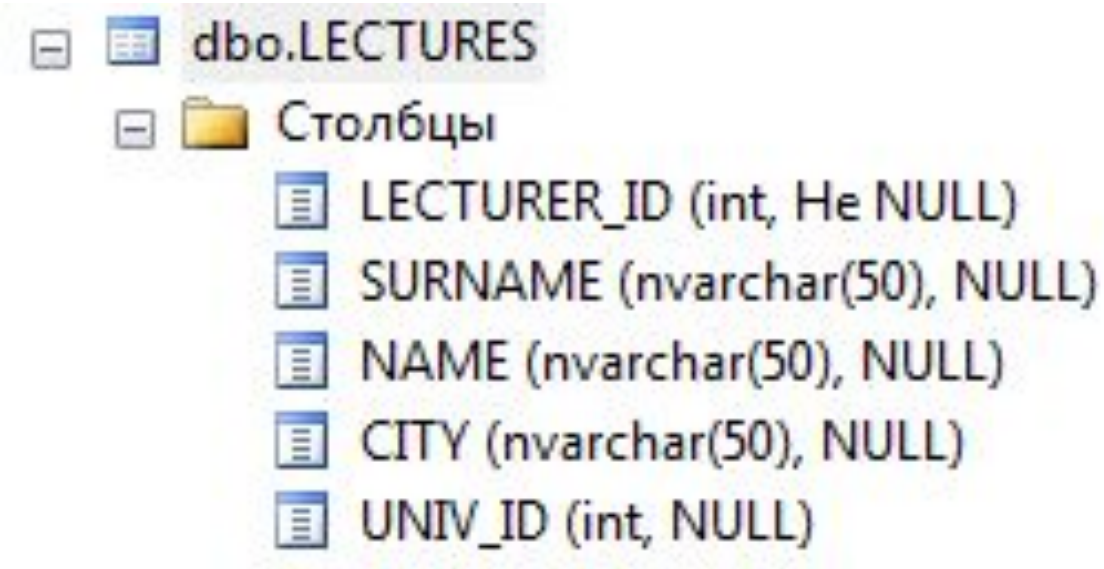
Список таблиц

EXAM_MARKS – экзаменационные оценки
LECTURES – сведения о преподавателях
STUDENT – сведения о студентах
SUBJECT – сведения о предметах
UNIVERSITY – сведения об университетах

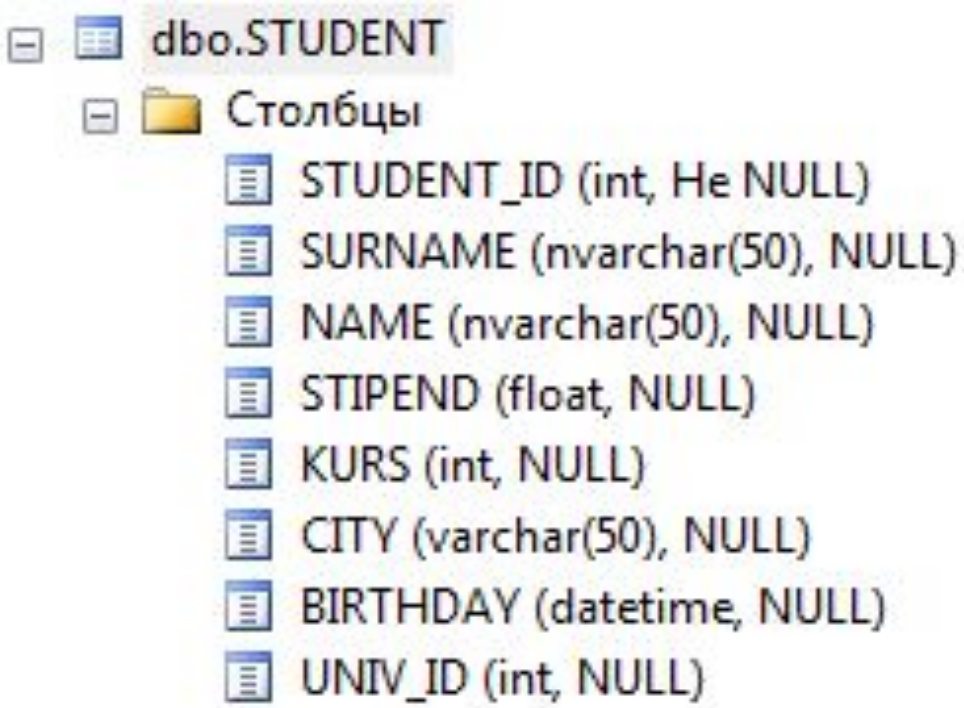
Описание базы данных University



Описание базы данных University



Описание базы данных University



Описание базы данных

University