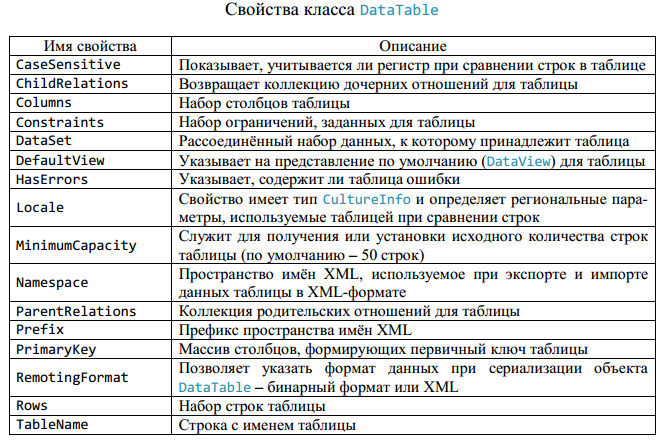
Последним элементом архитектуры ADO.NET является *рассоединённый набор данных* (*data set*). Набор данных содержит информационный фрагмент источника данных. Для описания набора данных используются классы из пространства имён System. Data. Эти классы универсальны и не зависят от поставщика данных. Главным компонентом набора является класс DataSet, агрегирующий объекты остальных классов. Класс DataTable служит для описания таблиц. Элементами класса DataTable являются коллекции объектов DataColumn  
(колонки таблицы), DataRow (строки таблицы) и Constraint (ограничения на значения элементов таблицы). Класс DataRelation описывает связи между таблицами.

Технология ADO.NET предоставляет возможность работы с рассоединённым набором данных. Такой набор реализуется объектом класса DataSet. Это реляционная структура, которая хранится в памяти. DataSet содержит коллекцию таблиц (объекты класса DataTable) и связей между таблицами (объекты класса DataRelation). Отдельная таблица хранит набор столбцов (объекты класса DataColumn), строк (объекты класса DataRow) и ограничений (объекты классов, унаследованных от Constraint). Столбцы и ограничения описывают структуру таблицы, а строки содержат данные таблицы.

Технически отдельные компоненты DataSet хранятся в специализированных коллекциях. Например, DataSet содержит коллекции Tables и Relations. Таблица имеет коллекции Columns (для колонок), Rows (для строк), Constraints (для ограничений), ParentRelations и ChildRelations (для связей таблицы). Любая подобная коллекция обладает сходным набором свойств и методов. Коллекции имеют перегруженные индексаторы для обращения к элементу по номеру или по имени, методы добавления, поиска и удаления элементов. Методы добавления перегружены и обеспечивают как добавление существующего объекта, так и автоматическое создание соответствующего объекта перед помещением в коллекцию.

Опишем состав класса DataTable. Этот класс является сериализуемым и поддерживает сохранение данных в формате XML. Класс DataTable реализует интерфейс IListSource, который позволяет объекту возвратить список своих данных (это необходимо для связывания с визуальными элементами управления). DataTable имеет конструктор без параметров и конструктор, принимающий в качестве аргумента строковое имя таблицы.

Наиболее важные свойства класса DataTable приведены в табл. 7.1.



Введём понятие схемы рассоединённого набора данных DataSet. Будем считать, что это совокупность следующих элементов:

– имена таблиц;  
– типы и имена отдельных столбцов таблицы;   
– ограничения на столбцы таблицы: уникальность, отсутствие пустых  
значений, первичные и внешние ключи;  
– связи между таблицами.

Правильная схема обеспечивает контроль целостности данных в приложении. Схема может быть определена вручную, путём создания и настройки свойств столбцов, таблиц, связей. Некоторые (но, к сожалению, не все) элементы схемы создаются автоматически при загрузке данных из базы в пустой DataSet. Кроме этого, схемы можно импортировать в виде XSD-файлов. Структура любой таблицы описывается свойствами её столбцов. Столбец таблицы представлен объектом класса DataColumn.

Создадим схему рассоединённого набора данных, взяв за основу схему таблиц Book и Category. Для этого сконструируем объект DataSet, а затем добавим таблицы в его коллекцию Tables:

var dataSet = new DataSet("Library");  
dataSet.Tables.Add(book);  
dataSet.Tables.Add(category);

Добавим связь между таблицами Book и Category. Связь представлена объектом класса DataRelation. При создании объекта DataRelation следует указать его имя, чтобы объект удалось найти в наборе; кроме этого, необходимо указать родительские и дочерние столбцы, на которых основано отношение. Чтобы упростить создание связей, класс DataRelation предоставляет конструкторы, принимающие как отдельные объекты DataColumn, так и массивы таких объектов.

Каждый поставщик данных содержит класс, описывающий адаптер данных. Адаптер является своеобразным мостом между источником данных и рассоединённым набором. Он позволяет считывать информацию из источника в набор и производит обратную операцию. В принципе подобные действия вполне осуществимы при помощи команд и объектов *DataReader*. Использование адаптера данных – это более унифицированный подход.  
Любой адаптер реализует интерфейс IDataAdapter и наследуется от класса DataAdapter.

Заметим, что вызов метода Fill() не нарушает состояние соединения с источником данных для команды выборки. Если соединение было открыто до вызова Fill(), оно останется открытым и после вызова. Если соединение было не установлено, метод Fill() откроет соединение, произведёт выборку данных и закроет соединение. Так же ведут себя и остальные методы адаптера, работающие с источником данных. Поведение адаптера при заполнении DataSet зависит от настроек адаптера и наличия схемы в объекте DataSet. Если при помощи адаптера заполняется пустой рассоединённый набор, то адаптер строит в DataSet *минимальную схему*, используя имена и тип столбцов из источника данных и стандартные имена для