|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ | | | | | | | | | |
|  | | |  | | |  | | | |
| ФГБОУ ВО «Пермский государственный  национальный исследовательский университет» | | | | | | | | | |
|  | | |  | | | |  | | |
|  | | ОТЧЕТ  по лабораторной работе «Древовидное представление» | | | | | |  | |
|  | | |  | | | |  | | |
|  | Работу выполнил  студент гр. \_\_\_\_\_\_\_\_\_\_\_\_\_ | | |  | Проверил  \_\_\_\_\_\_\_\_\_\_\_\_\_кафедры МОВС  (доц., ст.преп., асс.)  ФИО \_\_\_\_\_\_\_\_\_\_\_\_  (подпись)  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_2018 | | | |  |
| Минин А.С. \_\_\_\_\_\_\_\_\_\_\_  (подпись)  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_2018 | | |
|  |  | | |  |  | | | |  |
| Пермь 2018 | | | | | | | | | |

1 Постановка задачи

Написать программу, которая на графическом интерфейсе пользователя строит древовидный список по данным из спроектированной ранее базы данных, состоящей из трёх таблиц, последовательно связанных связью типа 1:М, в котором элементы каждой из трёх уровней иерархии отражают иерархию записей, хранящихся в базе данных. Раскрытии вершины на самом верхнем уровне иерархии к имени вершины приписывается строка " – A/B", где А – число непосредственных дочерних вершин, у которых есть дочерние вершины, а В – число непосредственных дочерних вершин без своих дочерних.

Необходимо также отразить записи с верхних уровней иерархии, для которых нет соответствующих записей на нижних уровнях иерархии.

Использование метода ToString(), а также конструкции try/catch запрещено. На каждом уровне иерархии не должны объединяться различные сущности с одинаковым текстовым представлением (названием).

2 Проектирование базы данных

В предметной области «Производство автомобилей» выделены следующие сущности: «Автоконцерны», «Бренды», «Модели автомобилей».

Автоконцерны имеют название и год основания. Бренды имеют название и название автоконцерна, к которому они принадлежат. Модели машин имеют название и название бренда, который занимается их производством.

Автоконцерну может принадлежать несколько брендов, но каждый бренд принадлежит только одному автоконцерну. Между таблицами «Автоконцерны» и «Бренд» связь типа 1:М.

Бренду может принадлежать несколько моделей машин, но каждая модель машины принадлежит только одному бренду. Между таблицами «Бренд» и «Модели автомобилей» связь типа 1:М.

Каждый автоконцерн характеризуется параметрами:

1. идентификатор;
2. название;
3. год основания.

Название и год основания могут совпадать, поэтому для каждой записи вводится уникальный идентификатор.

Каждый бренд характеризуется параметрами:

1. идентификатор;
2. название;
3. идентификатор автоконцерна, к которому принадлежит бренд.

Название и идентификатор автоконцерна, к которому принадлежит могут совпадать, поэтому для каждой записи вводится уникальный идентификатор.

Каждая модель автомобиля характеризуется параметрами:

1. идентификатор;
2. название;
3. идентификатор бренда, производящего данную модель автомобиля.

Название и идентификатор бренда, производящего данную модель автомобиля могут совпадать, поэтому для каждой записи вводится уникальный идентификатор.

В связи с этим, можно сделать вывод, что для проектирования БД понадобится три таблицы с соответствующими связи и характеристиками.

В БД «Производители автомобилей» будут фигурировать три таблицы: «Автоконцерны», «Бренды», «Модели автомобилей». Структура таблица представлена в таблицах 1-3.

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Разрешить NULL |
| id | int | Нет |
| Name | nvarchar(50) | Нет |
| Year | int | Да |

Таблица 2.1 – Автоконцерны

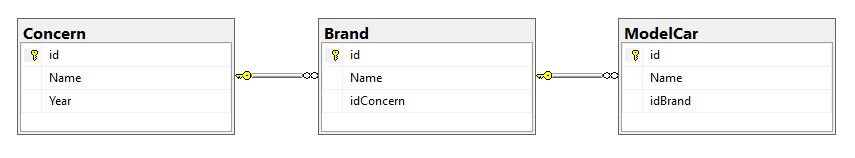
|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Разрешить NULL |
| id | int | Нет |
| Name | nvarchar(50) | Нет |
| idConcern | int | Нет |

Таблица 2.2 – Бренды

|  |  |  |
| --- | --- | --- |
| Имя столбца | Тип данных | Разрешить NULL |
| id | int | Нет |
| Name | nvarchar(50) | Нет |
| idBrand | int | Нет |

Таблица 2.3 – Модели автомобилей

Диаграмма связей таблиц «Автоконцерны», «Бренды», «Модели автомобилей» приведена на рисунке 2.1.

Рисунок 2.1 – Диаграмма БД

3 Общее описание

Для проектирования БД использовалось ПО MS SQL Server Management Studio. Приложение Windows Form для взаимодействия с БД будет разрабатывать на языке C# в среде Visual Studio 2012.

На форме расположен элемент управления TreeView, который отображает данные из базы данных в древовидной структуре. Внешний вид готового приложения представлен на рисунке 3.1.

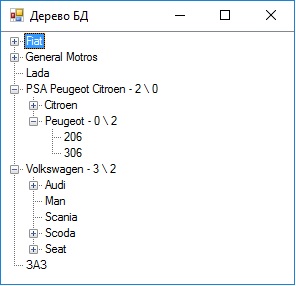


Рисунок 3.1 – Форма приложения

4 Описание алгоритма

1. Подключаем приложение к спроектированной базе данных и создаем запрос.
2. Просматриваем содержимое базы данных, пока не дойдем до последнего элемента.
   1. При каждой итерации мы получаем название автоконцерна и добавляем его на первый уровень дерева.
   2. Если автоконцерну принадлежит бренд, то мы получаем название бренда и добавляем бренд на второй уровень дерева соответствующему автоконцерну.
      1. Если при этом бренд занимается производством определенных моделей машин, то мы получаем название модели машины и добавляем на третий уровень дерева соответствующему бренду.
3. Если узел не имеет дочерних элементов, то мы не выводим численных характеристик. При раскрытии узла, просматриваем список дочерних вершин у узла, у которых смотрим количество таких же дочерних вершин. Если есть – увеличиваем счетчик «узлов с дочерними вершина» на 1, если нет – увеличиваем счетчик «узлов без дочерних вершин» на 1. Формируем строку вывода.
4. При сворачивании вершины удаляем все числовые характеристики, оставляя одно название.

5 Описание SQL – запроса

Рассмотрим SQL-запрос, который необходим для решения поставленной задачи:

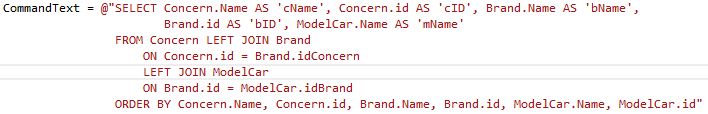


Рисунок 5.1 SQL – запрос

В блоке SELECT мы указываем, что надо получить следующие значения столбцов:

* название концерна;
* идентификатор концерна;
* название бренда;
* ндентификатор бренда;
* название модели.

С помощью AS присваиваются «уникальные» названия столбцам, которые в дальнейшем будут использоваться в коде для извлечения данных.

Таблицы концернов и брендов соединяются с помощью оператора LEFT JOIN (в результирующую таблицу войдут все элементы таблицы концернов, а отсутствующие значения в таблице брендов будут NULL). При это устанавливается соответствие, что идентификатор концерна равен идентификатору концерна в таблице брендов. Полученная таблица аналогичным образом соединяется с помощью LEFT JOIN с таблицей моделей машин, при этом устанавливается значение, что идентификатор бренда равен идентификатору бренда в таблице моделей машин.

С помощью ORDER BY таблицы сортируются по идентификаторам и названиям (в алфавитном порядке).

Далее подключаемся к БД, делаем запрос, извлекаем нужную информацию и наполняем контейнер древовидного представления.

6 Особенности реализации на C#

1. Формируем строку подключения:

SqlConnectionStringBuilder – средство для создания строки подключения.

Устанавливаем свойства:

* DataSource – задает сетевой адрес экземпляра SQL Server, с которым устанавливается соединение;
* IntegratedSecurity – определяет способ проверки подлинности. При true - используются учетные данные текущей учетной записи Windows;
* InitialCatalog – имя БД для подключения.

Свойство ConnectionString вернет строку подлюкчения, которую сохраняем.

1. Создаем подключение:

SqlConnection – класс, предоставляющий открытое подключение к БД SQL Server. Создаем объект типа SqlConnection и инициализируем, передав в качестве аргумента строку подключения. Открываем подключение.

1. Формируем запрос к БД:

SqlCommand – класс, определяющий запрос к БД.

Устанавливаем свойства:

* Connection – задает объект SqlConnection;
* CommandText – текст запроса к БД.

Пункт 3 обернет в пункт 2 с помощью конструкции using().

1. Извлекаем данные из БД:

SqlDataReader – класс, реализующий чтение потока строк последовательного доступа из БД.

Чтобы выполнить запрос к БД, необходимо воспользоваться методом ExecuteReader() из класса SqlCommand. Выполнение данного метода вернет объект типа SqlDataReader, который будет использоваться для извлечения информации из БД.

Создаем цикл, который выполняется пока не дойдем до конца БД. Переход к следующей записи БД выполняется с помощью метода Read() из класса SqlDataReader.

Для получения значения конкретного элемента необходимо обратиться к объекту SqlDataReader по индексу, передав в качестве индекса имя столбца типа String.

Если объект в БД отсутствует, то при чтении он имеет тип DBNull, поэтому для проверки того, что получаемое значение отлично от NULL используется конструкция вида:

if (!(reader["название столбца"] is DBNull)) {}

1. Представление данных:

Для древовидного представления в приложении Windows Form будет использоваться элемент TreeView.

1. Вычисление кол-ва вершин:

Используются встроенные методы элемента TreeView.

* BeforeExpand – срабатывает при разворачивании узла дерева.
* BeforeCollapse – срабатывает при сворачивании узла дерева.

7 Тестирование программы

Рассмотрим следующий набор тестов:

1. у автоконцерна много брендов;
2. у автоконцерна один бренд;
3. у автоконцерна нет брендов;
4. у бренда много моделей;
5. у бренда одна модель;
6. у бренда нет моделей;
7. два автоконцерна имеют одинаковое название и год основания;
8. два бренда имеют одинаковое название, но принадлежат разным автоконцернам;
9. два бренда имеют модели автомобилей с одинаковым названием.

Описание данных для тестирования:

Данные автоконцернов представлены в таблице 7.1.

|  |  |  |
| --- | --- | --- |
| id | Name | Год основания |
| 1 | General Motors | 1908 |
| 2 | Fiat | 1899 |
| 3 | Fiat | 1899 |

Таблица 7.1 – Данные о автоконцернах

Автоконцерн с id = 1 добавлен для тестирования случая, когда автоконцерн имеет много брендов.

Автоконцерн с id = 2 добавлен для тестирования случая, когда автоконцерн имеет один бренд.

Автоконцерн с id = 3 добавлен для тестирования случая, когда автоконцерн не имеет брендов.

Автоконцерны с id = 2 и с id = 3 также проверяют случай, когда два автоконцерна имеют одинаковое название и год основания.

Данные брендах представлены в таблице 7.2.

|  |  |  |
| --- | --- | --- |
| id | Name | idConcern |
| 1 | Chevrolet | 1 |
| 2 | GMC | 1 |
| 3 | GMC | 2 |

Таблица 7.2 – Данные о брендах

Бренд с id = 1 добавлен для тестирования случая, когда бренд имеет много моделей автомобилей.

Бренд с id = 2 добавлен для тестирования случая, когда бренд имеет одну модель автомобиля.

Бренд с id = 3 добавлен для тестирования случая, когда бренд не имеет моделей автомобилей.

Бренды с id = 2 и с id = 3 также проверяют случай, когда два бренда имеют одинаковое название, но принадлежат разным автоконцернам.

Данные моделей автомобилей представлены в таблице 7.3.

|  |  |  |
| --- | --- | --- |
| id | Name | idBrand |
| 1 | Aveo | 1 |
| 2 | Cobalt | 1 |
| 3 | Canyon | 2 |

Таблица 7.3 – Данные о моделях автомобилей

Тестирование на заданном наборе данных прошло успешно. Результат работы программы показан на рисунке 7.1.

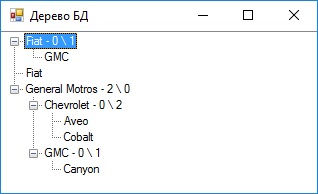


Рисунок 7.1 Результат работы программы