|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Министерство науки и Высшего образования РФ | | | | | | | | |  |
|  |  | | | | | | | | |  |
|  | | Пермский государственный национальный исследовательский университет | | | | | |  | | |
|  | | | |  | | |  | | | |
|  | | ОТЧЕТ  задание № 1 "Древовидное представление" по дисциплине "Учебная практика по базам данных" | | | | | |  | | |
|  | | | |  | | |  | | | |
|  | | | Работу выполнил  студент гр. ПМИ-1,2  Ковыляев Д.А.\_\_\_\_\_\_\_\_  (подпись)  «\_\_\_» \_\_\_\_\_\_\_\_2020 | |  | Проверил  старший преподаватель кафедры МОВС  Постаногов И.С. \_\_\_\_\_\_\_\_\_\_\_  (подпись)  «\_\_\_» \_\_\_\_\_\_\_\_2020 | | |  | |
|  | | | |  | | |  | | | |
|  | | | | Пермь 2020 | | |  | | | |

Содержание

[1 Постановка задачи. Задание № 1 «Древовидное представление» 3](#_Toc31782615)

[2 Определение идеи алгоритма, выбор методов решения и структур данных 4](#_Toc31782616)

[3 Тестирование 7](#_Toc31782617)

# Постановка задачи. Задание № 1 «Древовидное представление»

Написать программу, которая на графическом интерфейсе пользователя строит древовидный список по данным из спроектированной ранее базы данных в формате SQLite, 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. При 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.

Провести тестирование корректности работы программы. Удостовериться, что на каждом уровне иерархии не объединяются различные сущности с одинаковым текстовым представлением (названием).

Написать отчёт о проделанной работе.

**P.S.** Не забудьте также отразить записи с верхних уровней иерархии, для которых нет соответствующих записей на нижних уровнях иерархии.

**P.P.S.** Использование метода *ToString*() для приведения типов, а также конструкции *try/catch* запрещено.

# Определение идеи алгоритма, выбор методов решения и структур данных

Для решения задачи была выбрана система программирования Microsoft Visual Studio Community 2019 C#.

## Была составлена БД в формате SQLite, содержащая информацию о серверах. В БД есть 3 таблицы: сервера, игроки и предметы. Таблицы связаны связью типа 1:М соответственно. Диаграмму БД можно увидеть на рисунке Рисунок 1.

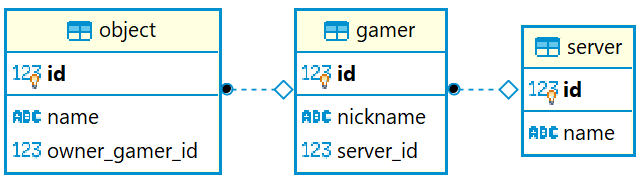


Рисунок 1 – Визуализация Server\_tree в DBeaver

У каждого сервера есть имя и идентификатор. Соответственно таблица server содержит столбцы name и id. name не может быть null.

У игрока есть идентификатор, никнейм и идентификатор сервера, на котором находится игрок. В таблице gamer это столбцы id, nickname и server\_id соответственно. nickname и server\_id не могут быть null.

У предмета есть идентификатор, название и идентификатор владельца. В таблице object это столбцы id, name и owner\_gamer соответственно. name и owner\_gamer не могут быть null.

Так как в таблицах object, gamer и server столбцы name, nickname и name соответственно являются типом varchar и не являются уникальными полями для соответствующих таблиц, то для каждой таблицы используется поле id в качестве первичного ключа.

## Построение дерева осуществляется при помощи элемента TreeView. Полученное приложение выглядит как представлено на рисунке Рисунок 2.

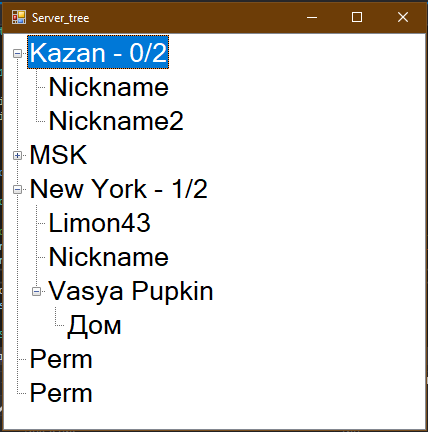


Рисунок 2 – Готовое приложение с treeView

## Для выполнения части задания (при 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) было решено создать Dictionary с ключами-ссылками на вершины иерархии – сервера и значениями специально написанного класса ServerInfo.

ServerInfo, как говорит название класса, хранит некоторую информацию о сервере: имя сервера, количество потомков (игроков) и количество потомков, у которых имеются потомки (предметы).

Класс ServerInfo имеет два метода. Первый увеличивает счётчик количества потомков на 1, второй увеличивает счётчик количество потомков, у которых имеются потомки.

## Получение информации из БД происходит с помощью SQL запроса, представленного ниже.

SELECT server.name as server\_name,

server.id as server\_id,

gamer.nickname as gamer\_name,

gamer.id as gamer\_id,

object.name as object\_name

FROM server

LEFT JOIN gamer ON server.id = gamer.server\_id

LEFT JOIN object ON gamer.id = object.owner\_gamer\_id

ORDER BY server\_name,

server.id,

gamer\_name,

gamer.id,

object\_name,

object.id

## Запрашиваемыми элементами являются:

Имена серверов и их идентификаторы;

Никнеймы игроков и их идентификаторы;

Имена объектов.

Использование LEFT JOIN при соединении таблиц в запросе позволяет выводить сервера, на которых отсутствуют игроки, а также игроков, у которых отсутствуют предметы.

Запрашиваемая информация сортируется по возрастанию, чтобы было возможно построить дерево в TreeView за один проход.

## Построение дерева осуществляется так, что на каждом этапе добавления очередной строчки из таблицы, полученной из БД, имеется позиция вставки элемента в TreeView, меняющаяся по следующим правилам:

* если появился уникальный сервер, то добавляем его в Dictionary и делаем текущим местом вставки вершин-игроков (делаем текущим сервером). Добавляем соответственно корневую вершину в TreeViev;
* если появился уникальный игрок, то добавляем новую вершину к текущему серверу и делаем игрока текущим;
* пришедший из таблицы предмет добавляем в дерево к вершине текущего игрока.

При добавлении вершин в дерево так же происходит увеличение счётчиков класса ServerInfo для соответствующих серверов.

## Обработка открытия вершины дерева сделана следующим образом: если открытая вершина является сервером, то меняем текст приписывая информацию из ServerInfo этого сервера.

Закрытие сервера-вершины меняет текст вершины на название сервера.

## Работа с пользователем ведётся через окно Server\_tree, имеющее в своей основе элемент TreeView.

# Тестирование

Проведя анализ возможных входных данных согласно критериям тестирования программы по типу «черного ящика», было составлено 10 тестов, для каждого из них ниже дано описание и результат прохождения данного теста программой.

Все тесты, кроме первого, тестируются на тестовой БД приведённой на таблицах 1, 2 и 3.

1. – server

|  |  |
| --- | --- |
| **id** | name |
| 1 | MSK |
| 2 | New York |
| 3 | Perm |
| 4 | Kazan |
| 5 | Perm |

1. – gamer

|  |  |  |
| --- | --- | --- |
| **id** | nickname | server\_id |
| 1 | olegase | 1 |
| 3 | olegase | 1 |
| 4 | Vasya Pupkin | 2 |
| 5 | Limon43 | 2 |
| 6 | Nickname | 4 |
| 7 | Nickname2 | 4 |
| 8 | Nickname | 2 |

1. – object

|  |  |  |
| --- | --- | --- |
| **id** | name | owner\_gamer\_id |
| 1 | Банка | 1 |
| 2 | Бутылка | 1 |
| 5 | Бочка | 3 |
| 6 | Бутылка | 1 |
| 7 | Банка | 3 |
| 8 | Дом | 4 |

На следующей таблице 4 приведены тесты, их описания и результаты.

Таблица 4 – Тесты

| Таблица 4 - продолжение | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| № | Заголовок теста | Суть теста | Последовательность шагов тестирования | Входные данные | Ожидаемый результат | Фактический результат |
| 1 | БД без записей | Проверка отображения дерева, построенного по пустой БД | 1. Запустить программу 2. Проверить дерево | БД без записей | Дерево не отображается | Как ожидалось |
| 2 | Пустой сервер | Проверка отображения в дереве сервера без игроков | 1. Запустить программу 2. Проверить в дереве наличие пустого сервера | Сервер из тестовой БД, не имеющий игроков | Узел, представляющий сервер из входных данных не имеет потомков, не может быть раскрыт | Как ожидалось |
| 3 | Игрок без предметов | Проверка отображения в дереве игрока без предметов | 1. Запустить программу 2. Развернуть узел, в котором находятся игроки без предметов 3. Проверить наличие игрока без предметов | Сервер из тестовой БД с игроком, у которого нет предметов | Узел, представляющий игрока без предметов, не имеет потомков, не может быть раскрыт | Как ожидалось |
| 4 | Проверка приписываемой строки | Проверка приписывания строки к серверу, имеющему только игроков, не имеющих предметов | 1. Запустить программу 2. Развернуть узел, в котором находятся игроки без предметов 3. Проверить приписанную строку | Сервер из тестовой БД с игроками, у каждого из которых нет предметов | Параметр «А» будет равен нулю, а параметр «В» равен количеству игроков на открытом узле сервера без предметов | Как ожидалось |
| 5 | Проверка приписываемой строки | Проверка приписывания строки к серверу, имеющему только игроков с предметами | 1. Запустить программу 2. Развернуть узел, в котором находятся игроки с предметами 3. Проверить приписанную строку | Сервер из тестовой БД с игроками, у каждого из которых есть предмет | Параметр «В» будет равен нулю, а параметр «А» равен количеству игроков на открытом узле сервера с предметами | Как ожидалось |
|  | Проверка приписываемой строки | Проверка приписывания строки к серверу, на котором есть игроки с предметами и есть игроки без предметов | 1. Запустить программу 2. Развернуть узел, в котором находятся игроки с предметами и игроки без предметов 3. Проверить приписанную строку | Сервер из тестовой БД с игроками, у которых есть предметы и игроками, у которых нет предметов | Параметры «А» и «В» будут соответствовать количеству игроков с предметами и без соответственно | Как ожидалось |
| 6 | Серверы с одинаковыми названиями | Проверка отображения серверов, имеющих одинаковые названия | 1. Запустить программу 2. Проверить отобразились ли сервера с одинаковыми названиями | Пара серверов из тестовой БД с одинаковыми названиями | Оба сервера отобразятся | Как ожидалось |
| 7 | Игроки с одинаковыми никнеймами | Проверка отображения игроков, имеющих одинаковый никнейм | 1. Запустить программу 2. Открыть узел сервера, на котором находятся игроки с одинаковыми никнеймами 3. Проверить отобразились ли игроки с одинаковыми никнеймами | Сервер из тестовой БД, на котором находятся игроки с одинаковыми никнеймами | Все игроки с одинаковыми никнеймами отобразились успешно | Как ожидалось |
| 8 | Предметы с одинаковыми названиями | Проверка отображения предметов, имеющих одинаковые названия | 1. Запустить программу 2. Открыть узел сервера с игроком, имеющим искомые одинаковые предметы 3. Проверить отображение предметов | Сервер из тестовой БД, на котором находится игрок, имеющий несколько одинаковых предметов | Все предметы с одинаковыми названиями отобразились успешно | Как ожидалось |
| 9 | Одинаковые никнеймы, но разные сервера | Проверка правильного расположения в дереве игроков с одинаковыми никнеймами, но разными серверами | 1. Запустить программу 2. Открыть узлы серверов, имеющих игроков с одинаковыми именами 3. Проверить нахождение в дереве игроков | Сервера из тестовой БД, на каждом из которых находится игрок, имеющий такой же никнейм, как и игрок на другом сервере. | Все узлы игроков принадлежат своим серверам | Как ожидалось |
| 10 | Одинаковые наименования предметов, нов у разных пользователей | Проверка правильного расположения предметов с одинаковым наименованием, но разными владельцами | 1. Запустить программу 2. Открыть узлы серверов, имеющих игроков с одинаковыми наименованиями предметов 3. Проверить правильность распределения предметов | Сервер из тестовой БД, на котором находятся игроки, имеющие одинаковые по названию предметы | Предметы принадлежат своим владельцам | Как ожидалось |