

Клиент-серверное приложение с базой данных MySQL

Введение. Об архитектурах серверных приложений

Часть 1. Когда БД уже есть. Связываем MySQL + C#

Часть 2. Создаём свою БД, подсоединяя её и используем

Часть 3. Некоторые примеры структуры базы данных и запросов к ней

Для справки – откуда грузить:

MySQL Community Server 8.0.20

Select Operating System:
Microsoft Windows

Looking for previous GA
versions?

Recommended Download:

скачать сервер MySQL:

<https://dev.mysql.com/downloads/mysql/>

MySQL Installer
for Windows



All MySQL Products. For All Windows Platforms.
In One Package.

Starting with MySQL 5.6 the MySQL Installer package replaces the standalone MSI packages.

Windows (x86, 32 & 64-bit), MySQL Installer MSI

[Go to Download Page >](#)

Other Downloads:

Windows (x86, 64-bit), ZIP Archive

8.0.20

187.5M

[Download](#)

(mysql-8.0.20-winx64.zip)

MD5: 1335fe593b055686823fd69c7ef035f5 | Signature

Windows (x86, 64-bit), ZIP Archive

8.0.20

411.8M

[Download](#)

Debug Binaries & Test Suite

(mysql-8.0.20-winx64-debug-test.zip)

MD5: 377ae6ee648c28455e7d9c948e77ba8f | Signature

MySQL Workbench 8.0.20

Select Operating System:

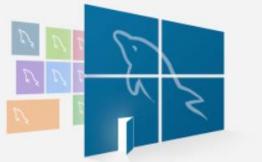
Microsoft Windows

скачать редактор Workbench:

<https://dev.mysql.com/downloads/workbench/>

Recommended Download:

MySQL Installer
for Windows



All MySQL Products. For All Windows Platforms.
In One Package.

Starting with MySQL 5.6 the MySQL Installer package replaces the standalone MSI packages.

Windows (x86, 32 & 64-bit), MySQL Installer MSI

[Go to Download Page >](#)

Other Downloads:

Windows (x86, 64-bit), MSI Installer

8.0.20

35.6M

[Download](#)

(mysql-workbench-community-8.0.20-winx64.msi)

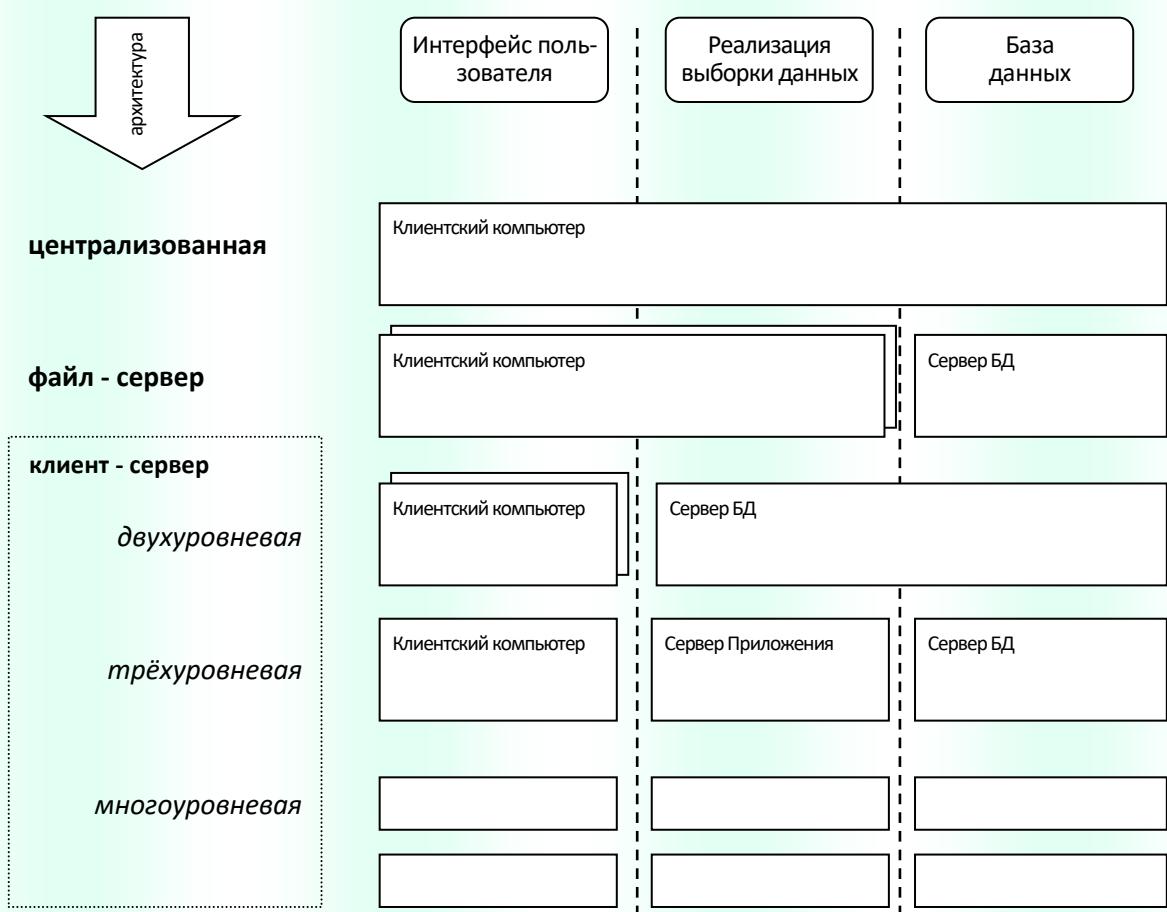
MD5: edd40cd63fa929127389f0440b930012 | Signature

Введение. Об архитектурах серверных приложений

Разновидности архитектур в общем виде:

- централизованная система;
- архитектура «файл-сервер»;
- архитектура «клиент-сервер»;
 - двухуровневая,
 - трёхуровневая,
 - многоуровневая.

Схема в помощь:



Поясняющий текст:

Централизованная система – приложение и БД на одном компьютере, например, связка Lazarus (Delphi) + БД Access.

Архитектура «файл-сервер» - на удалённом сервере хранится БД, а приложение на клиентском компьютере. Нагрузка на сеть повышена, так как во время работы на «клиента» загружается не выборка из БД, а сама БД.

Двухуровневая архитектура «клиент-сервер» - на удалённом сервере хранится БД и установлена СУБД, а на клиенте только интерфейс пользователя, через который формируются SQL-запросы - они отправляются через сеть к СУБД, она их обрабатывает и возвращает через сеть к клиенту только выборку из БД (например, C# + MySQL).

Трёхуровневая архитектура «клиент-сервер» - на одном удалённом сервере хранится БД, на клиенте реализован интерфейс пользователя, но нет непосредственной связи между этими уровнями - между ними есть промежуточное звено: сервер приложения, где и реализована бизнес-логика, именно там (тоже удалённо от клиента) формируются запросы к БД и принимаются выборки данных от БД (например, php + MySQL). Преимущества:

- функции обработки данных можно использовать в нескольких проектах - нет дублирования кода, экономия ресурсов;
- независимость сервера БД и сервера Приложения;
- можно дублировать ПО для одного проекта на нескольких серверах Приложений - возрастает надёжность и скорость работы при перегрузках.

Многоуровневая архитектура «клиент-сервер» - используется для крупных и территориально-распределённых предприятий. Есть несколько филиалов предприятия с локализованными трёхуровневыми архитектурами, объединёнными в общую систему. Ведутся полная БД для всего предприятия и копии БД, адаптированные под функционал филиалов. Репликация данных может проходить синхронно (на всех серверах одновременно) и асинхронно (периодически, по расписанию).

Замечательный вариант статьи про клиент-серверную архитектуру читайте тут:

<https://habr.com/ru/post/495698/>



Часть 1. Когда БД уже есть. Связываем MySQL + C#

Чтобы понять, как это работает и, особенно, чтобы понять почему что-то из написанного не работает, нужно всё это проделать своими руками...

Этап 1 – настройка среды Visual Studio.

Что нужно для связки MySQL + C#? Нужно чтобы в Visual Studio появилась возможность (по умолчанию нет её) подключать через «Обозреватель решений» ссылку на DLL-библиотеку, которая имеет классы для работы с MySQL. Возможны два варианта подключения. Саму библиотеку (`MySql.Data.dll`) можно заранее скачать из сети с учётом версии фреймворка (.NET) того компьютера, на котором собираетесь работать с БД (можно использовать версию библиотеки ниже, чем фреймворк на компьютере, но не выше), например, версии 4.0 и 4.5 можете взять по прямой ссылке:

<https://pcoding.ru/dll/MySQL/net40/MySql.Data.dll>

<https://pcoding.ru/dll/MySQL/net45/MySql.Data.dll>

или закачать самую последнюю версию, подходящую именно для вашей системы, используя возможности самого Visual Studio через NuGet:

- создайте обычное приложение с одной формой,
- в обозревателе решений нажмите правой клавишей мыши по опции «Ссылки»,
- в контекстном меню выберите «Управление пакетами NuGet»,
- в открывшемся слева окне перейдите на вкладку «Обзор»
- в поисковой строке наберите `mySql` (рис.1),
- после поиска выберите версию `MySql.data`,
- и нажмите «Установить» (рис.2).

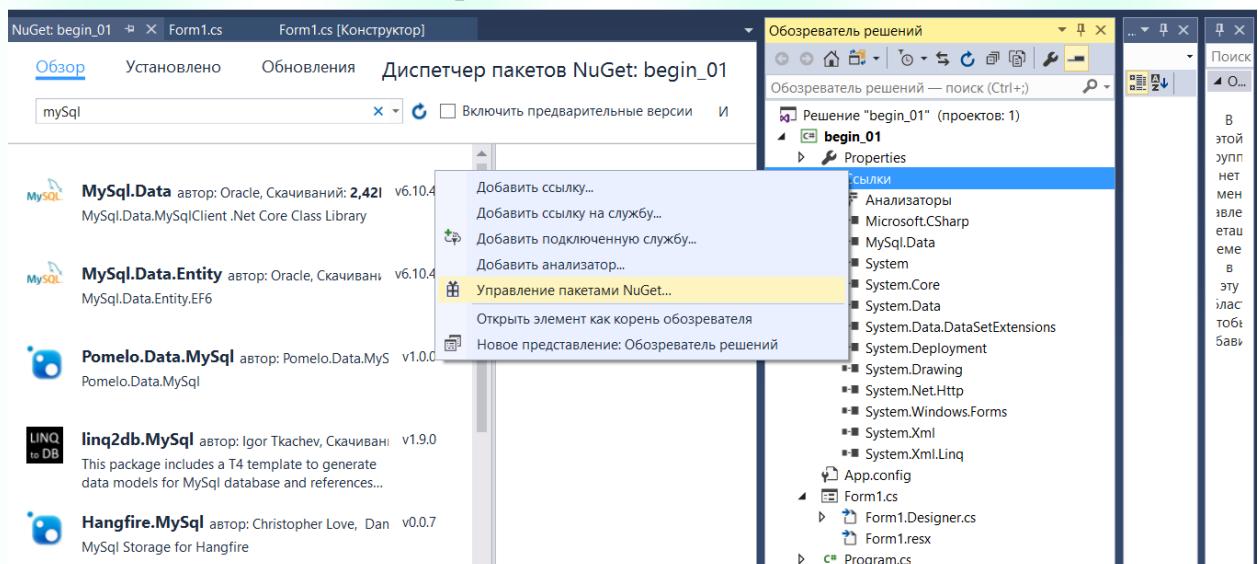


Рис.1.

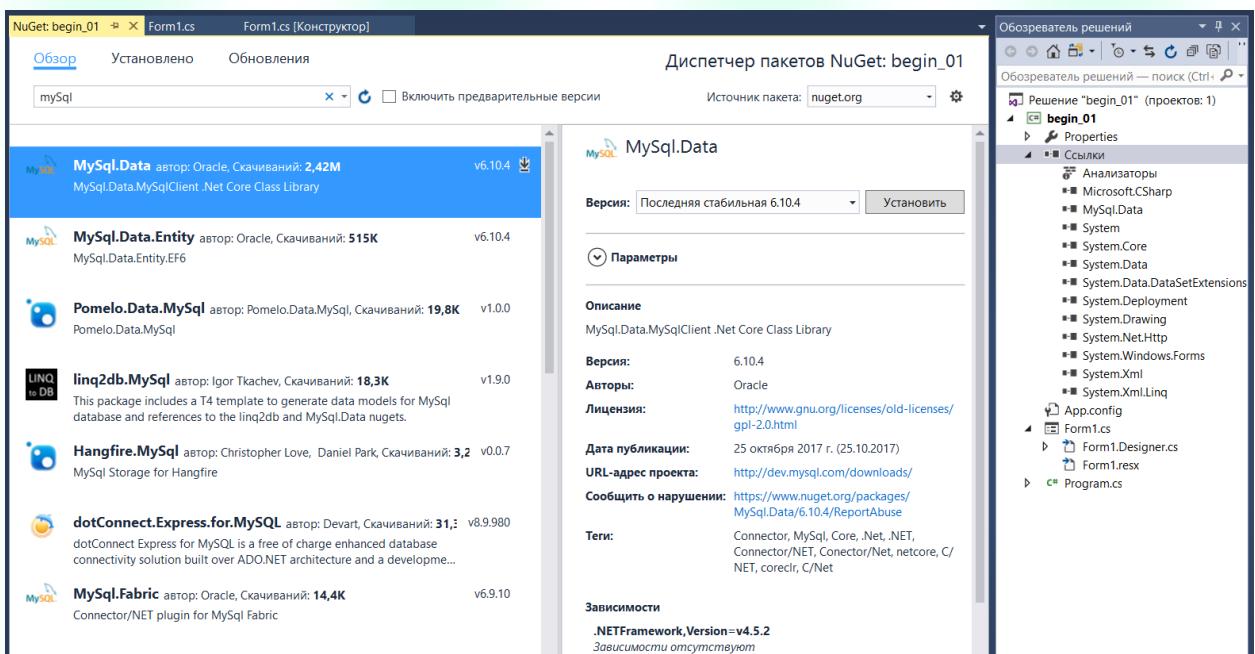


Рис.2.

Если у вас уже есть скачанная ранее библиотека (`MySql.Data.dll`), то можно просто её подключить:

- в «Обозревателе решений» правой клавишей мыши кликаете по опции «Ссылки»,
- выбираете «Добавить ссылку» (рис.3),
- далее жмёте клавишу «Обзор», выбираете путь и сам файл (рис.4), нажимаете «OK».

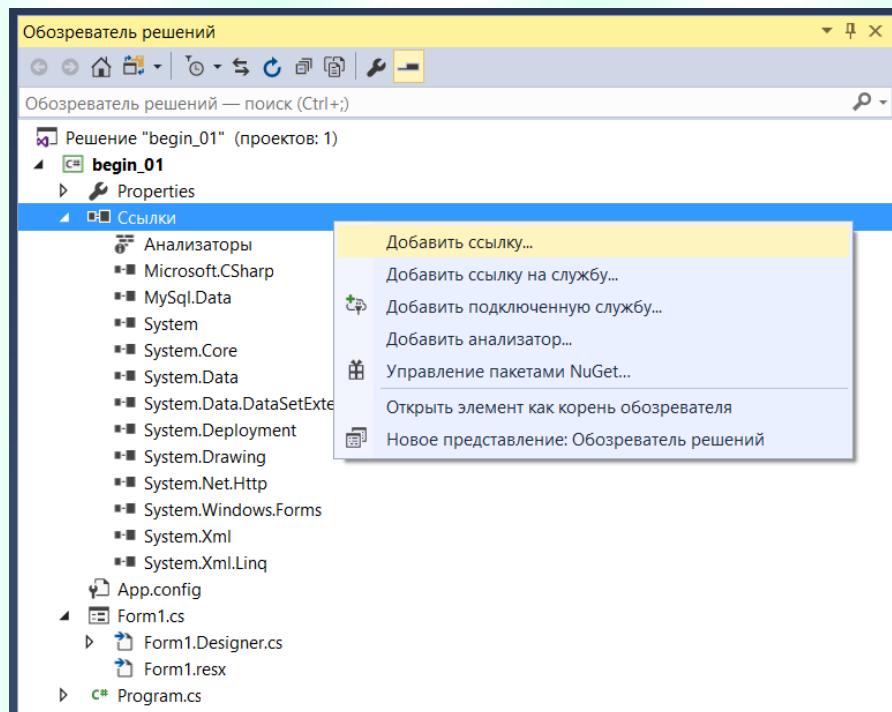


Рис.3

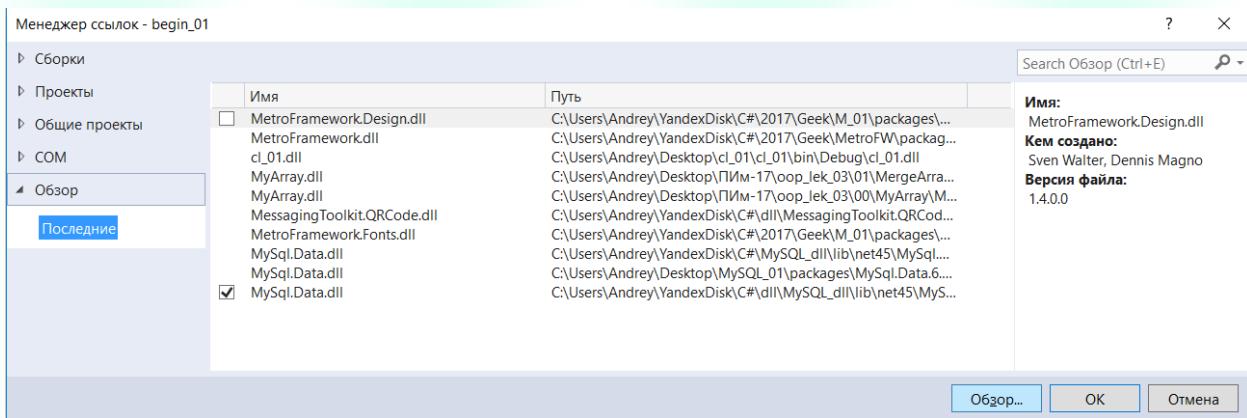


Рис.4.

Когда вы проделаете один из выше приведённых вариантов подключения библиотеки, то в «Обозревателе решений» вы увидите новую строчку – `MySQL.Data` (см. на рис.3), но для использования классов из этой библиотеки в тексте кода (в самом верху) каждого создаваемого модуля к формам вашего приложения ещё нужно будет добавить строчку:

```
using MySql.Data.MySqlClient;
```

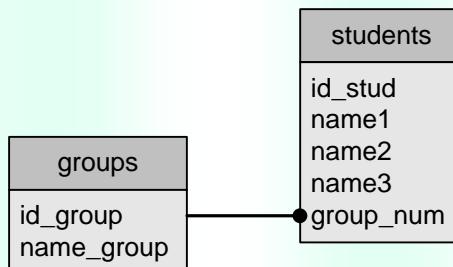
Этап первичной подготовки приложения на этом закончен.

Этап 2 – подсоединимся к БД, считаем данные и отобразим на форме.

После подключения библиотеки можно уже приступить работать с самой базой данных MySQL (если, конечно, её кто-то ранее создал). Для начала так и поступим. Для реализации этой задачи у меня есть специальная пробная база данных – она состоит из двух таблиц:

- учебные группы – `groups`,
- студенты – `students`,

связанных отношением «один-ко-многим» (`id_group` – `group_num`):



Чтобы подключить программу на C# к сетевой базе данных, нужны такие параметры:

- адрес (хостинг) БД в сети,
- идентификатор (имя) БД,
- имя пользователя,
- пароль пользователя

– кодировка (это необязательно).

Разместите на форме кнопку, сгенерируйте для неё обработчик события «клик мышкой» и в нём создайте объект подключения к БД с такими параметрами:

```
MySqlConnectionStringBuilder db;  
db = new MySqlConnectionStringBuilder();  
db.Server = "mysql195.1gb.ru"; // хостинг БД  
db.Database = "gb_psis"; // Имя БД  
db.UserID = "gb_psis"; // Имя пользователя БД  
db.Password = "ca8484adc89a"; // Пароль пользователя БД  
db.CharacterSet = "utf8"; // Кодировка Базы Данных
```

и сгенерируйте строку подключения:

```
MySqlConnection conn;  
conn = new MySqlConnection(db.ConnectionString);
```

Чтобы убедиться в корректности подключения, попытаемся открыть БД и уведомим пользователя о результатах.

```
try  
{  
    conn.Open();  
    MessageBox.Show("Подключение к БД установлено");  
}  
catch (Exception ex)  
{  
    MessageBox.Show("Проблемы с подключением к БД \n\r" + ex.ToString());  
}
```

Все эти строчки следует разместить в обработчике события клик мыши по клавише:

```
private void button1_Click(object sender, EventArgs e)
```

Апробируйте работу программы по подключению. Обратите внимание, что должно быть в наличии подключение к сети интернет и, после нажатия на клавишу Button1, подключение проходит не мгновенно, а занимает пару секунд. Если удалось установить подключение, то можно подумать уже о начале работы с данными.

Что же можно сделать для начала? Можно, например, вывести что-нибудь из таблицы groups – это будет просто список групп.

Пришло время уточнить, что некоторые переменные следует объявлять, как глобальные, чтобы они были доступны в каждой из функций модуля, поэтому две ранее объявленные переменные (db и conn) перенесите и две новые напишите в самом начале объявления класса формы, а процесс настройки параметров подключения БД перенесите в обработчик загрузки формы:

```

public partial class Form1 : Form
{
    MySqlConnectionStringBuilder db;
    MySqlConnection conn;
    MySqlCommand cmd;
    string sql;

    public Form1()
    {
        InitializeComponent();
    }

    private void Form1_Load(object sender, EventArgs e)
    {
        db = new MySqlConnectionStringBuilder();
        db.Server = "mysql95.1gb.ru"; // хостинг БД
        db.Database = "gb_psis"; // Имя БД
        db.UserID = "gb_psis"; // Имя пользователя БД
        db.Password = "ca8484adc89a"; // Пароль пользователя БД
        db.CharacterSet = "utf8"; // Кодировка Базы Данных
        conn = new MySqlConnection(db.ConnectionString);
    }
}

```

Далее создайте на форме вторую кнопку и сгенерируйте для неё обработчик (клик мышкой). Давайте пока, в качестве первой пробы, просто в текстовое поле (не забудьте установить MultiLine=True) выведем список групп:

```

private void button2_Click(object sender, EventArgs e)
{
    sql = "SELECT name_group FROM groups";
    cmd = new MySqlCommand(sql, conn);
    MySqlDataReader reader = cmd.ExecuteReader();
    textBox1.Clear();
    while (reader.Read())
    {
        textBox1.Text += reader[0].ToString() + Environment.NewLine;
    }
    reader.Close();
}

```

В этом обработчике сначала мы создаём строку с SQL-командой, затем формируем из неё объект cmd для трансляции команды в БД и, с помощью метода ExecuteReader запускаем её на исполнение, результаты возвращаем в переменную reader. Далее очищаем текстовое поле и в цикле читаем все записи, полученные по команде SELECT. В каждой записи может быть несколько полей (это как в одной строке таблицы), нумерация полей начинается с нуля. Какие именно поля, их количество и названия вы сами определили в запросе:

```
sql = "SELECT name_group FROM groups";
```

Итак, у нас есть только одно поле – `name_group`, к нему можно обратиться как по порядковому номеру, так и по имени:

```
textBox1.Text += reader["name_group"].ToString() + Environment.NewLine;
```

Апробируйте оба варианта.

В исследуемой таблице (`groups`) есть ещё один столбец – `id_group`, он нужен для того, чтобы в связной таблице хранить не полное наименование групп, а только ссылки на них. Но сейчас вы можете попробовать немного доработать свою программу и вывести в текстовое поле оба столбца таблицы, для чего в запросе нужны изменения:

```
sql = "SELECT id_group, name_group FROM groups";
```

и в организации вывода, например, так:

```
while (reader.Read())
{
    textBox1.Text +=
        reader["id_group"].ToString() + '\t' +
        reader["name_group"].ToString() +
        Environment.NewLine;
}
```

Если, в дальнейшем, мы хотим организовать вывод студентов из определённой группы по выбору пользователя, то имеет смысл список групп загружать в подходящий для этой цели визуальный компонент, например, в `ComboBox` (добавьте его на форму). Перепишите обработчик второй клавиши следующим образом:

```
sql = "SELECT name_group FROM groups";
cmd = new MySqlCommand(sql, conn);
MySqlDataReader reader = cmd.ExecuteReader();
comboBox1.Items.Clear();
while (reader.Read())
{
    comboBox1.Items.Add(reader["name_group"].ToString());
}
reader.Close();
comboBox1.Text = "Список групп";
```

Теперь уже можно сделать обработчик выбора опции в выпадающем списке (`comboBox1`), который и будет в текстовое поле `textBox1` выводить список фамилий (`name1`) из выбранной группы (пока номер группы равен 1):

```
private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    sql = "SELECT name1 FROM students WHERE group = 1";
    cmd = new MySqlCommand(sql, conn);
    MySqlDataReader reader = cmd.ExecuteReader();
    textBox1.Clear();
    while (reader.Read())
    {
        textBox1.Text +=
            reader["name1"].ToString() + Environment.NewLine;
    }
    reader.Close();
}
```

Апробируйте работоспособность этого кода – вполне возможно, что в текстовое поле не будет загружено ни одной фамилии, так как пользователи

этой открытой БД (такие же студенты, как и вы) могли удалить все записи из неё. Проследить это обстоятельство совсем несложно. Если программа не зависит, работает корректно, но в текстовом поле нет записей, то внесите в программу следующие изменения – добавьте функцию получения количества записей в таблице и в сам обработчик добавьте несколько строчек:

```
int getCount(string nameTable)
{
    sql = "SELECT COUNT(*) FROM " + nameTable;
    cmd = new MySqlCommand(sql, conn);
    return Convert.ToInt32(cmd.ExecuteScalar().ToString());
}

private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    sql = "SELECT name1 FROM students WHERE group_num = 1";
    cmd = new MySqlCommand(sql, conn);
    MySqlDataReader reader = cmd.ExecuteReader();
    textBox1.Clear();
    // тут добавка
    textBox1.Text += "Список группы " +
        comboBox1.SelectedItem + ":" + Environment.NewLine;
    //
    while (reader.Read())
    {
        textBox1.Text +=
            reader["name1"].ToString() + Environment.NewLine;
    }
    reader.Close();
    // тут добавка:
    textBox1.Text +=
        " - - - " + Environment.NewLine +
        "Общее кол-во студентов во всех группах = " +
        getCount("students").ToString() + Environment.NewLine;
}
}
```

Я полагаю, что указанные изменения стоит реализовать у себя в программе даже если программа и так работала корректно и, не только потому, что вы сами можете в дальнейшем случайно удалить всех студентов из таблицы, но и для расширения спектра используемых методов работы.

Если всё же в группах есть студенты, то всё равно в таком виде этот код работает не совсем так, как мы задумывали, так как в запросе указано выбирать группу с номером 1 вне зависимости от выбора пользователя. Переобратим обработчик (я использую первый, более короткий вариант, но вы можете оставить более полный вариант), вернув сначала первым запросом номер группы по её имени, а уже затем список студентов выбранной группы:

```
private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    MySqlDataReader reader;

    sql = "SELECT id_group FROM groups WHERE name_group = " +
        "'" + comboBox1.SelectedItem + "'";
```

```

cmd = new MySqlCommand(sql, conn);
reader = cmd.ExecuteReader();
reader.Read();
int numId = Convert.ToInt32(reader[0]); // номер группы
reader.Close();

sql = "SELECT name1 FROM students WHERE group_num = " + numId.ToString();
cmd = new MySqlCommand(sql, conn);
reader = cmd.ExecuteReader();
textBox1.Clear();
while (reader.Read())
{
    textBox1.Text +=
        reader["name1"].ToString() + Environment.NewLine;
}
reader.Close();
}

```

Это, конечно, не самое лучшее решение, но оно сделано по аналогии с предыдущими решениями и интуитивно понятно, что достаточно для первой работоспособной программы. Возможные варианты доработки таковы:

- когда мы ранее загружали список групп, то нужно вместе с ними загружать и id групп, хранить их в какой-то структуре и, при необходимости, использовать (это сможете сделать сами);
- можно просто сделать вложенный запрос вместо двух запросов, как было в нашем последнем обработчике, и выглядит это так:

```

private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    MySqlDataReader reader;

    sql = "SELECT name1 FROM students WHERE group_num = " +
          "(SELECT id_group FROM groups WHERE name_group = " +
          "''' + comboBox1.SelectedItem + ''')";
    cmd = new MySqlCommand(sql, conn);
    reader = cmd.ExecuteReader();
    textBox1.Clear();
    while (reader.Read())
    {
        textBox1.Text +=
            reader["name1"].ToString() + Environment.NewLine;
    }
    reader.Close();
}

```

Может сложиться такая ситуация (в другой БД), при которой внутренний подзапрос вернёт не одно, а несколько значений, в этом случае необходимо внести изменения в наш запрос, заменив символ сравнения (“=”) на команду проверки вхождения в множество («IN»):

```

sql = "SELECT name1 FROM students WHERE group_num IN " +
      "(SELECT id_group FROM groups WHERE name_group = " +
      "''' + comboBox1.SelectedItem + ''')";

```

Проверьте, что и для нашей БД это работает корректно, ведь множество может состоять и из одного элемента.

Теперь я приведу пример всех уже готовых функций и скриншот экрана программы во время выполнения запроса, чтобы вы смогли сравнить результаты:

```
using System;
using System.Data;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

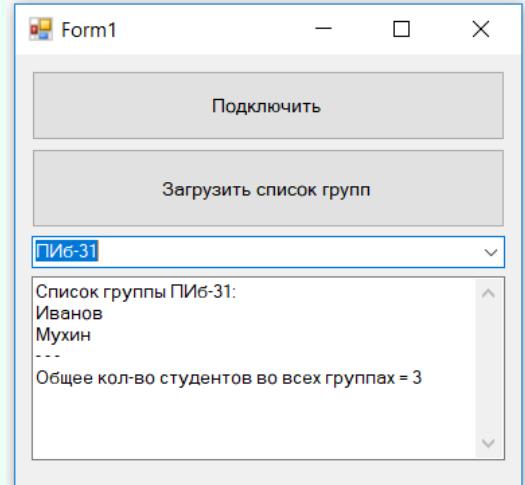
namespace beginMySQL
{
    public partial class Form1 : Form
    {
        MySqlConnectionStringBuilder db;
        MySqlConnection conn;
        MySqlCommand cmd;
        string sql;

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            db = new MySqlConnectionStringBuilder();
            db.Server = "mysql95.1gb.ru"; // хостинг БД
            db.Database = "gb_psis"; // Имя БД
            db.UserID = "gb_psis"; // Имя пользователя БД
            db.Password = "ca8484adc89a"; // Пароль пользователя БД
            db.CharacterSet = "utf8"; // Кодировка Базы Данных
            conn = new MySqlConnection(db.ConnectionString);
        }

        private void button1_Click(object sender, EventArgs e)
        {
            try
            {
                if (conn.State == ConnectionState.Closed)
                {
                    conn.Open();
                }
                MessageBox.Show("Подключение к БД установлено");
            }
            catch (Exception ex)
            {
                MessageBox.Show("Проблемы с подключением к БД \n\r" + ex.ToString());
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            sql = "SELECT name_group FROM groups";
            cmd = new MySqlCommand(sql, conn);
            MySqlDataReader reader = cmd.ExecuteReader();
            comboBox1.Items.Clear();
            while (reader.Read())
            {
                comboBox1.Items.Add(reader["name_group"].ToString());
            }
        }
    }
}
```



```

        reader.Close();
        comboBox1.Text = "Список групп";
    }

    int getCount(string nameTable)
    {
        sql = "SELECT COUNT(*) FROM " + nameTable;
        cmd = new MySqlCommand(sql, conn);
        return Convert.ToInt32(cmd.ExecuteScalar().ToString());
    }

    private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
    {
        MySqlDataReader reader;

        sql = "SELECT name1 FROM students WHERE group_num IN " +
              "(SELECT id_group FROM groups WHERE name_group = " +
              "'" + comboBox1.SelectedItem + "'')";
        cmd = new MySqlCommand(sql, conn);
        reader = cmd.ExecuteReader();
        textBox1.Clear();
        textBox1.Text += "Список группы " +
                         comboBox1.SelectedItem + ":" + Environment.NewLine;
        while (reader.Read())
        {
            textBox1.Text +=
                reader["name1"].ToString() + Environment.NewLine;
        }
        reader.Close();
        textBox1.Text +=
            " - - - " + Environment.NewLine +
            "Общее кол-во студентов во всех группах = " +
            getCount("students").ToString() + Environment.NewLine;
    }

    private void button3_Click(object sender, EventArgs e)
    {
        sql = "INSERT INTO students (name1, name2, name3, group_num) ";
        sql += "VALUES(NULL, '" + textBox1.Text + "')";
        cmd = new MySqlCommand(sql, conn);
        cmd.ExecuteNonQuery();
    }
}

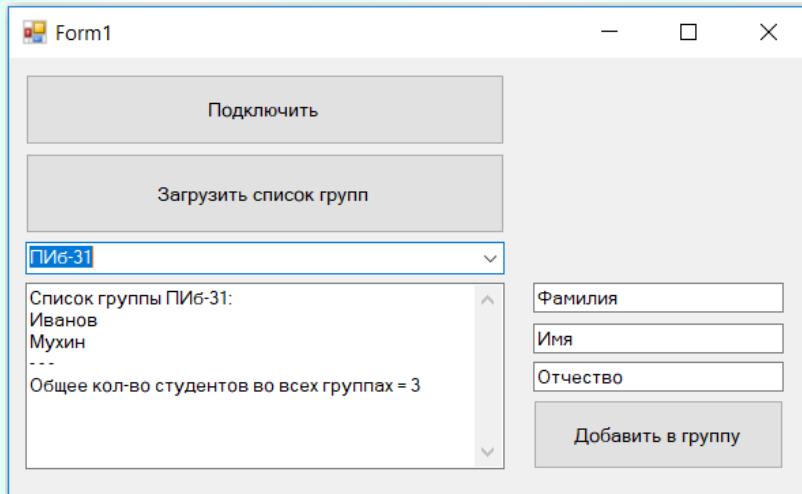
```

Этап 3 – добавляем записи в таблицу БД.

Дальше мы будем апробировать команды по *добавлению, изменению и удалению* строчек из таблицы «Студенты». Давайте сразу договоримся, что вы не трогаете таблицу с учебными группами (groups), так как эта БД используется не только вами (чтобы и остальные смогли воспользоваться этой методичкой). Итак, чтобы работать с таблицей, нужно понимать её структуру, количество и названия полей. Я приведу скриншот структуры и содержания таблицы students:

id_stud	name1	name2	name3	group_num
1	Иванов	Иван	Иванович	1
2	Сидоров	Сидор	Сидорович	4
3	Мухин	Олег	Николаевич	1

Доработайте форму для реализации функции добавления записи в таблицу:



Создайте функцию для определения номера текущей группы и запрос на добавление записи про студента в таблицу students:

```

int getNumGroup()
{
    MySqlDataReader reader;
    sql = "SELECT id_group FROM groups WHERE name_group = " +
          "" + comboBox1.SelectedItem + "";
    cmd = new MySqlCommand(sql, conn);
    reader = cmd.ExecuteReader();
    reader.Read();
    int numId = Convert.ToInt32(reader[0]); // номер группы
    reader.Close();
    return numId;
}

private void button3_Click(object sender, EventArgs e)
{
    sql = "INSERT INTO students (id_stud, name1, name2, name3, group_num) ";
    sql += "VALUES(NULL, " +
           "" + textBox2.Text + ", " + // Фамилия
           "" + textBox3.Text + ", " + // Имя
           "" + textBox4.Text + ", " + // Отчество
           getNumGroup().ToString() + ")"; // номер группы
    cmd = new MySqlCommand(sql, conn);
    cmd.ExecuteNonQuery();
}

```

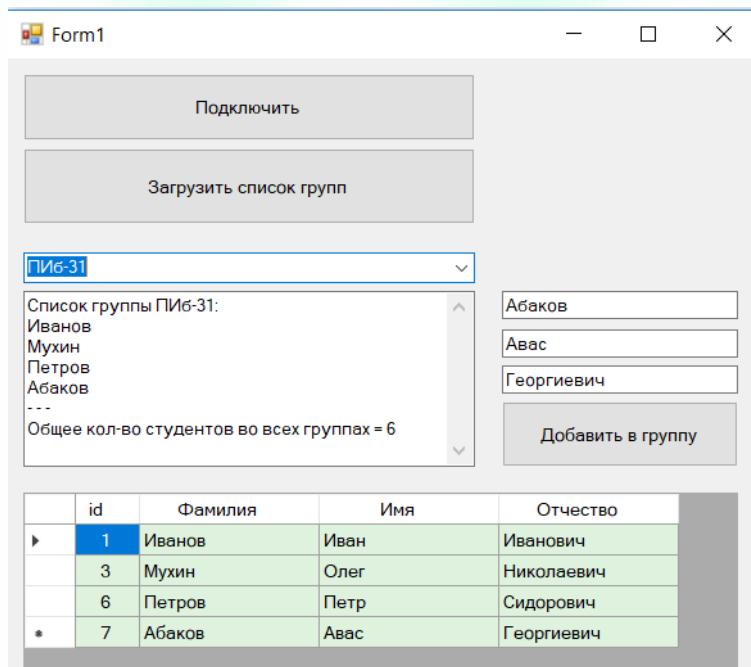
Добавьте несколько студентов в разные группы и проверьте полученный результат.

На этом можно считать, что первые, базовые трудности преодолены, так как вы уже научились связывать сетевую БД с программой, делать запросы по отбору данных из связанных таблиц и отображать полученные результаты на форме приложения.

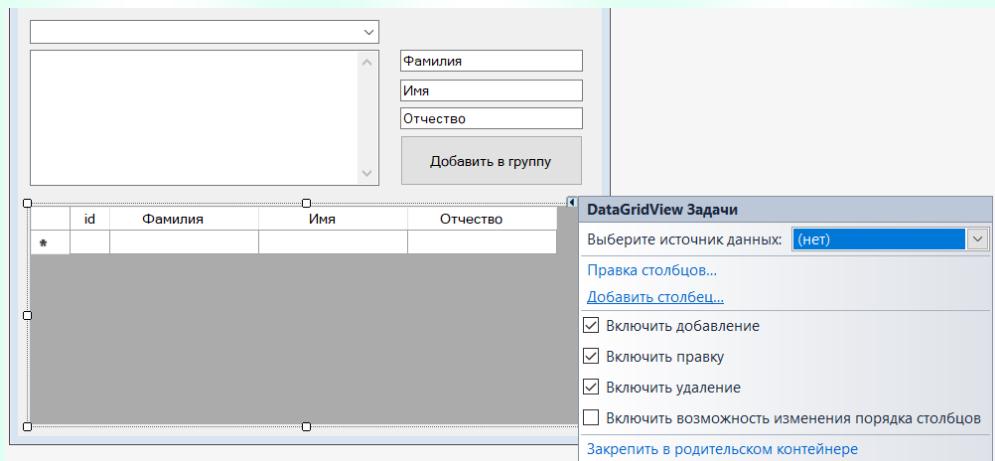
Этап 4 – удаляем и изменяем записи в БД.

Для реализации функций *изменения* и *удаления* записей в БД уже неудобно результаты отображать в обычном, хоть и многострочном, текстовом поле. Пришло время добавить на форму таблицу. Чтобы мне не переписывать полностью программу, я, для реализации функции отбора (SELECT), оставлю на форме многострочное текстовое поле, а для функций редактирования и удаления добавлю DataGridView. После освоения всех предложенных в данной методичке методов вы сами для себя будете выбирать приемлемые подходы.

Настройте внешний вид таблицы по примеру (*уточню, что на рисунке в таблице первая строка является фиксированной, а фиксированных столбцов нет совсем*):



Напомню, что во вновь размещённую таблицу dataGridView можно добавить и настроить столбцы через контекстное меню, вызываемое по нажатию на треугольник в правом верхнем углу таблицы:



Итак, приступим к кодированию функционала. Для начала добавим три кнопки: «Обновить таблицу», «Удалить запись» и «Обновить запись». Сгенерируйте обработчик клика мышки по клавише «Обновить таблицу» и добавьте туда одну строчку кода – вызов функции заполнения таблицы:

```
fillTable();
```

Это вызов функции по выводу списка студентов в таблицу на форме. Сама эта функция пока не описана в коде, но прямо сейчас займёмся этим. Функция должна выполнять две задачи:

- сначала загружаем данные студенческой группы в список,
- затем отображаем этот список в таблице.

Конечно, вы можете обойтись и без списка и сразу после чтения из БД данные размещать в таблице на форме, как мы ранее делали с многострочным текстовым полем. Однако хранить данные в специально созданной для этого структуре и использовать их по необходимости может оказаться удобным. Итак, добавьте описание *структуры* в части, где мы объявляли глобальные переменные:

```
struct tableStud
{
    public int tsId;
    public string tsName1, tsName2, tsName3;
}
```

Структура нужна нам для более удобного (структурированного) хранения записей из таблицы студенты. Далее опишите две функции (функция `getTable()` вызывается из функции `fillTable()`) – получить записи и отобразить их на форме в таблице:

```
List<tableStud> getTable()
{
    List<tableStud> tbStud = new List<tableStud>();
    tableStud tmp; // для хранения текущей считанной записи
    tbStud.Clear(); // очистим список
    MySqlDataReader reader; // объект для чтения записей
```

```

sql = "SELECT id_stud,name1,name2,name3 FROM students WHERE group_num IN " +
      "(SELECT id_group FROM groups WHERE name_group = " +
      """ + comboBox1.SelectedItem + ")";
cmd = new MySqlCommand(sql, conn);
reader = cmd.ExecuteReader();
while (reader.Read())
{
    tmp.tsId = Convert.ToInt32(reader["id_stud"].ToString());
    tmp.tsName1 = reader["name1"].ToString();
    tmp.tsName2 = reader["name2"].ToString();
    tmp.tsName3 = reader["name3"].ToString();
    tbStud.Add(tmp); // добавим текущую запись в список
}
reader.Close();
return tbStud; // вернём список записей из таблицы
}

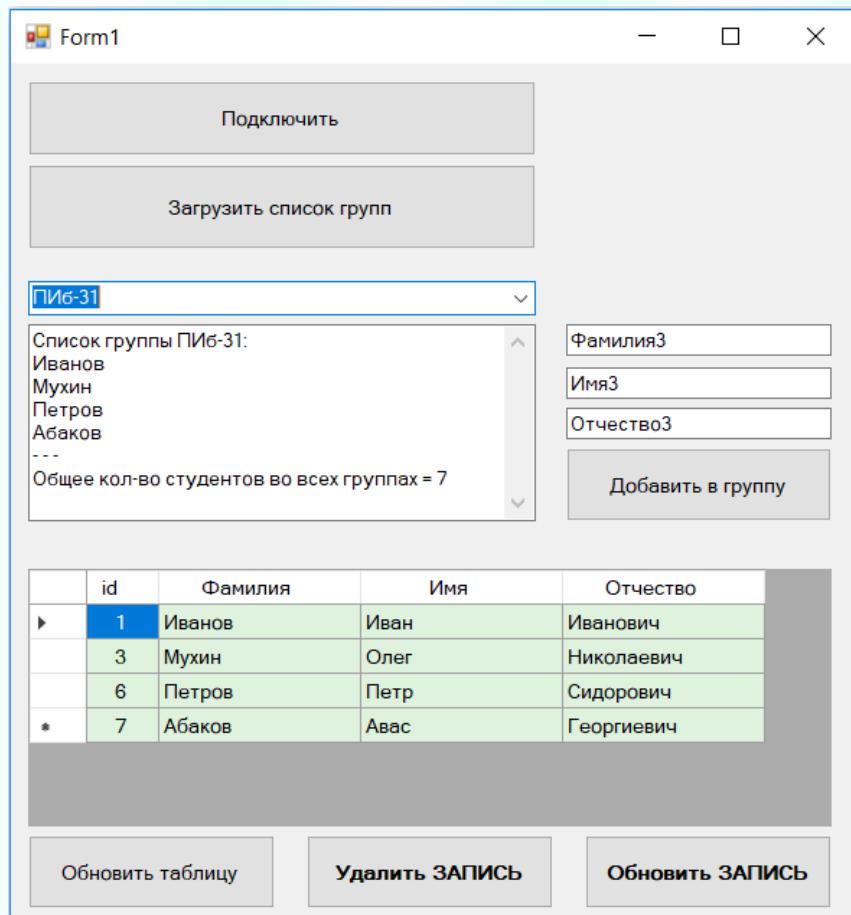
void fillTable() // данные из списка переносим в таблицу
{
    List tbStud = getTable(); // получим список студентов

    // начальные настройки для таблицы
    dataGridView1.Rows.Clear();
    dataGridView1.DefaultCellStyle.BackColor = Color.FromArgb(222, 242, 222);
    dataGridView1.Columns[0].Width = 50;
    dataGridView1.Columns[1].Width = 140;
    dataGridView1.Columns[2].Width = 140;
    dataGridView1.Columns[3].Width = 140;

    dataGridView1.RowCount = tbStud.Count;
    for (int i = 0; i < tbStud.Count; i++)
    {
        dataGridView1.Rows[i].Cells[0].Value = tbStud[i].tsId;
        dataGridView1.Rows[i].Cells[1].Value = tbStud[i].tsName1;
        dataGridView1.Rows[i].Cells[2].Value = tbStud[i].tsName2;
        dataGridView1.Rows[i].Cells[3].Value = tbStud[i].tsName3;
    }
}

```

Если всё было прописано корректно, то вы сможете получить примерно такой результат:



Теперь уже будет гораздо удобнее выполнять задачи по удалению или изменению записей. Например, можно сделать так: удалять запись, если был клик по ней средней клавишей мыши или просто удалять текущую запись по нажатию на специально установленную на форму клавишу удаления:

```
private void button5_Click(object sender, EventArgs e)
{
    int indRow = dataGridView1.CurrentRow.Index; // узнаём текущую строку
    int idStud = Convert.ToInt32(dataGridView1.Rows[indRow].Cells[0].Value);
    sql = "DELETE FROM students WHERE id_stud = '" + idStud.ToString() + "'";
    cmd = new MySqlCommand(sql, conn);
    cmd.ExecuteNonQuery();
}
```

В данном обработчике мы сначала узнаём номер текущей строки в таблице, затем по нему мы определяем идентификатор записи в таблице `students`, потом формируем SQL-команду на удаление и, собственно, исполняем её. Так как у нас достаточно простой интерфейс пользователя, то после удаления записи следует нажать клавишу «Обновить таблицу» для отображения текущих изменений. Однако, заметно удобнее будет, если вы добавите в последний обработчик в самом конце ещё и строчку:

```
fillTable();
```

В данном варианте сразу после удаления будет происходить обновление отображения данных в таблице.

Завершающая задача данной части изложения – это создание обработчика по внесению изменений в текущую запись. Для простоты описания сделаем это так: у пользователя есть возможность редактировать записи непосредственно в компоненте dataGridView, после редактирования пользователь должен нажать клавишу «Обновить запись» для внесения изменений в сетевую БД (курсор должен находиться в текущей строке таблицы):

```
private void button6_Click(object sender, EventArgs e)
{
    int indRow = dataGridView1.CurrentRow.Index; // узнаём текущую строку
    int idStud = Convert.ToInt32(dataGridView1.Rows[indRow].Cells[0].Value);
    string n1 = dataGridView1.Rows[indRow].Cells[1].Value.ToString();
    string n2 = dataGridView1.Rows[indRow].Cells[2].Value.ToString();
    string n3 = dataGridView1.Rows[indRow].Cells[3].Value.ToString();
    sql = "UPDATE students SET ";
    sql += "name1 = '" + n1 + "', name2 = '" + n2 + "', name3 = '" + n3 + "' ";
    sql += "WHERE id_stud = '" + idStud.ToString() + "'";
    cmd = new MySqlCommand(sql, conn);
    cmd.ExecuteNonQuery();
}
```

Этих знаний вполне достаточно чтобы продумать и разработать полноценное приложение по работе с сетевой базой данных. Осталось дело за мальм – [научиться самому создавать и размещать базы данных в сети.](#)

Часть 2. Создаём свою БД, подсоединяем её и используем

Что нужно для того, чтобы «создавать и размещать базы данных в сети»? Чтобы создавать (быстро и удобно) – нужен специализированный визуальный редактор (типа MySQL **Workbench**) или имеющаяся на всех хостингах панель для управления базами данных **phpMyAdmin**. В первом варианте после создания базы данных можно её использовать и без хостинга в интернете, а прямо у себя на компьютере, как локальную БД. Это может быть удобно для редактирования и тестирования программы, а строка подключения к локально размещённой БД выглядит крайне просто:

```
string connStr = "server=localhost;user=root;database=students;password=0000;";
```

Конечно, логин и пароль индивидуальны. Однако, потом, тем не менее, необходимо будет потратить время на поиск хостинга и загрузку (может быть и настройку) базы данных. По этой причине, для несложных задач имеет смысл сразу создавать структуру БД непосредственно в сети на хостинге через панель phpMyAdmin.

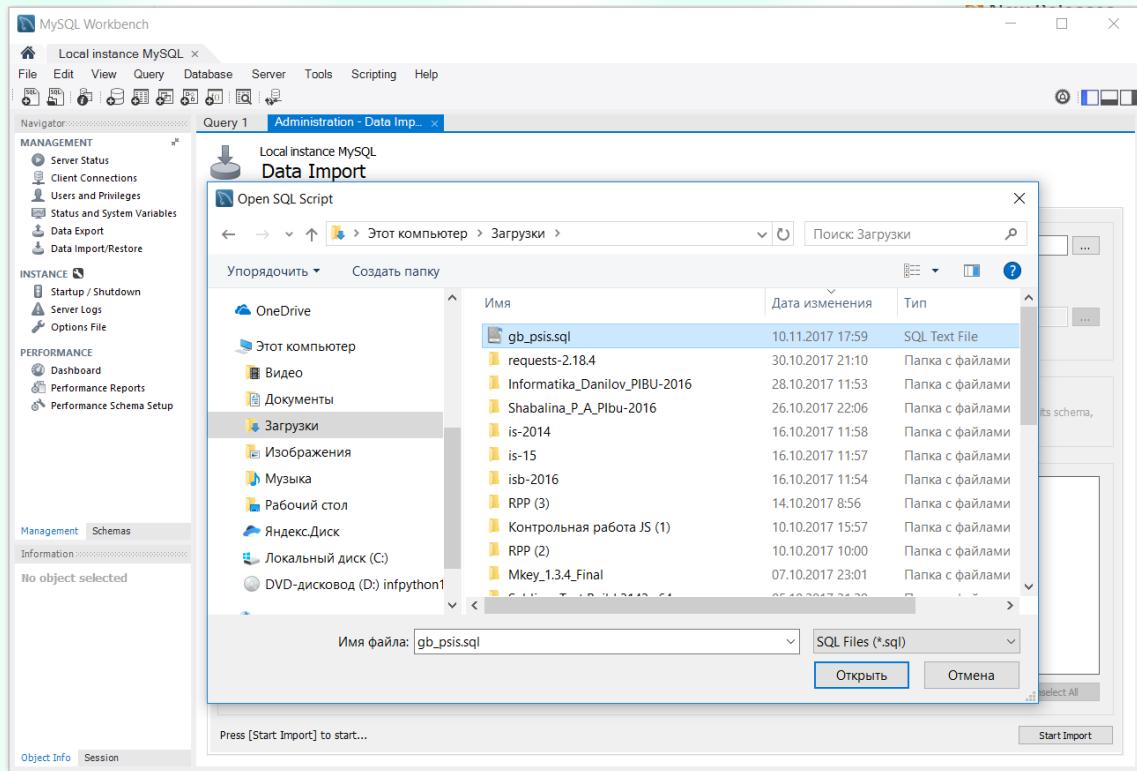
Прежде чем перейдем к практике приведу примеры названных программных продуктов и некоторые их возможности в общем виде.

Обращаю ваше внимание, что в этой части я не буду затрагивать и описывать подробности работы с Workbench, так как работа непосредственно в phpMyAdmin может сэкономить время на разработку при небольшой БД.

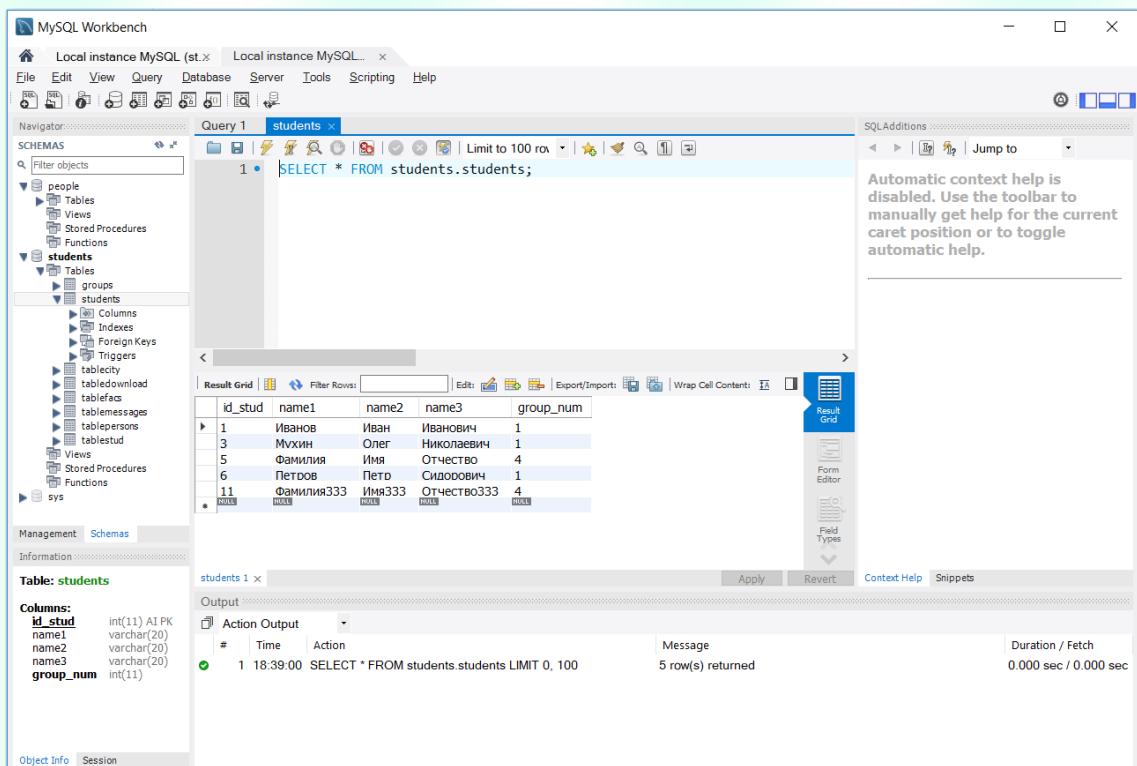
Итак, вот отсюда можно скачать бесплатную версию MySQL Workbench: <https://dev.mysql.com/downloads/workbench/>

The screenshot shows the MySQL Downloads page. At the top, there's a navigation bar with links for MySQL.COM, DOWNLOADS (which is underlined), DOCUMENTATION, and DEVELOPER ZONE. Below this is a secondary navigation bar with links for Enterprise, Community (which is highlighted in blue), Yum Repository, APT Repository, SUSE Repository, Windows, and Archives. On the left side, there's a sidebar with a list of MySQL products: MySQL on Windows, MySQL Yum Repository, MySQL APT Repository, MySQL SUSE Repository, MySQL Community Server, MySQL Cluster, MySQL Router, MySQL Utilities, MySQL Shell, MySQL Workbench (which is also highlighted in blue), MySQL Connectors, and Other Downloads. The main content area is titled 'Download MySQL Workbench'. It describes MySQL Workbench as a tool for DBAs and developers. It lists the following features: Database Design & Modeling, SQL Development, Database Administration, and Database Migration. It also mentions that the Community (OSS) Edition is available under the GPL. Below this, it says 'MySQL Workbench Windows Prerequisites' and lists the required libraries: Microsoft .NET Framework 4.5 and Visual C++ Redistributable for Visual Studio 2015.

Вот так можно в эту программу импортировать чужую БД, например, созданную в сети через phpMyAdmin (файл `gb_psis.sql` был как раз сохранён с моего хостинга – эта и есть та база, что мы использовали в первой части):



И после импорта с ней уже можно поработать (обратите внимание – в левой части есть проводник по БД и там видны названия таблиц – `groups` и `students`):



А вот как эта же БД выглядела во время её разработки непосредственно на хостинге через панель phpMyAdmin (обратите внимание, что в верхней части как раз виден хостинг (и конкретный сервер `mysql95.1gb.ru` для хранения БД) и имя БД, что мы использовали в первой части нашей работы):

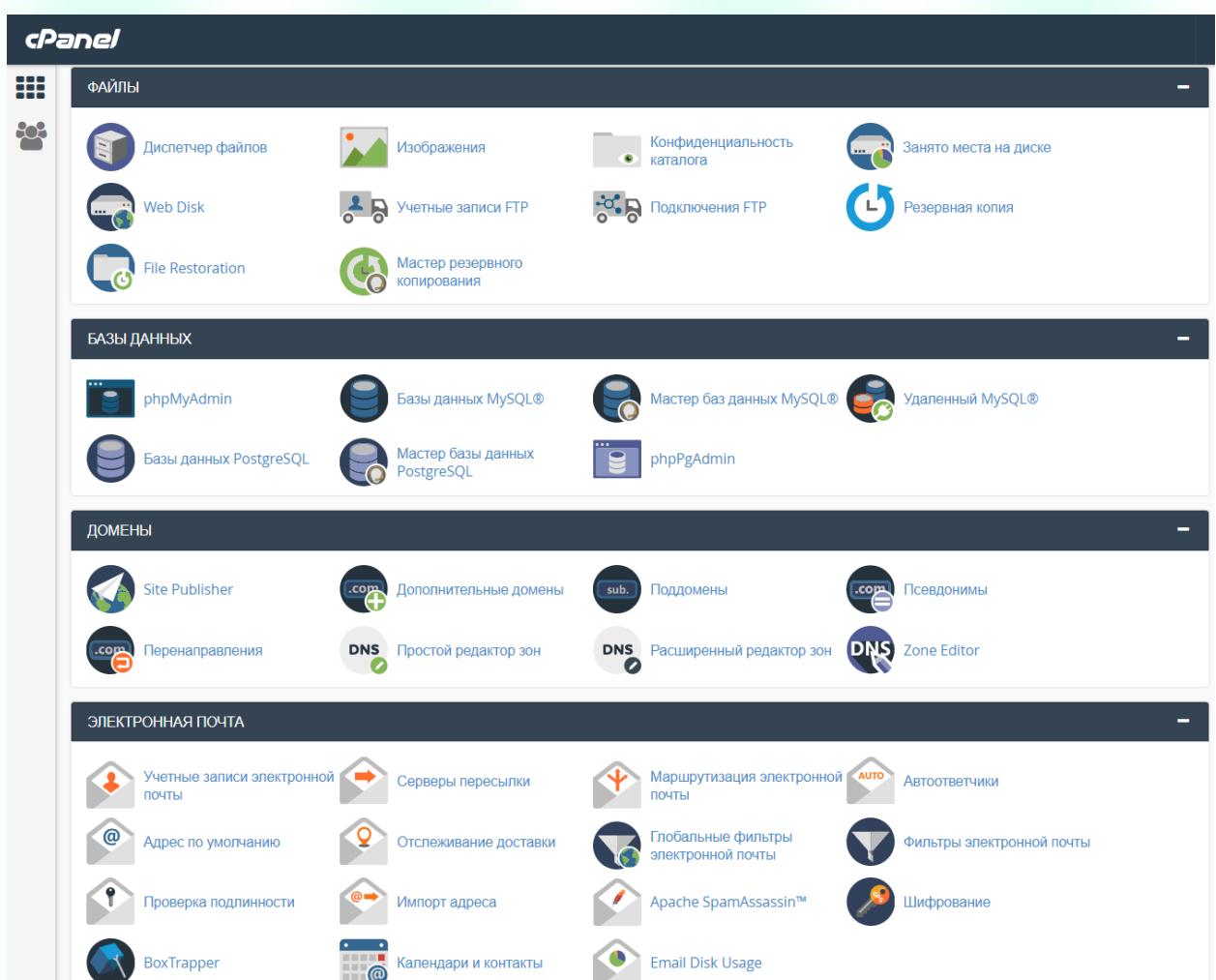
А так можно просматривать и редактировать содержимое таблиц:

На этом первичный обзор возможностей закончен, теперь перейдём к практике. Создадим БД, две таблицы в ней и свяжем некоторые поля из разных таблиц. Ну, а как соединять БД с программой на C# и работать с таблицами и полями вы уже изучили в первой части...

Выбор инструмента

Когда вы зарегистрировались (*про выбор хостинга читайте ниже в «лирическом отступлении»*) на каком-то хостинге с базами данных, то вам предоставляется возможность кроме размещения собственно сайта (html + css + js + php) добавлять базы данных, как правило: MySQL, PostgreSQL или Microsoft SQL Server.

На разных хостингах интерфейс добавления новой БД устроен по-разному, поэтому описывать что-то конкретное тут не имеет смысла, просто найдите пункт меню или кнопку «Работать с базами данных» или «Добавить базу данных». Весь этот «интерфейс пользователя» обычно называют клиентской панелью – CPanel. В некоторых случаях такая панель может занимать 2-3 экрана браузера и предоставлять очень широкий функционал (тут поместились только треть функционала):



Не редкость, когда клиентской панели, как таковой, нет вообще:

Центр управления аккаунтом

быстрый поиск:

- [Зарегистрировать/привязать домен и создать сайт](#)
- С помощью этого мастера можно также создать домен 3-го уровня или получить сайт с именем в домене 1gb.ru.
- [Зарегистрировать/привязать домен к существующему сайту](#)
- [Зарегистрировать домен \(парковка домена\)](#)
- [Создать ресурс другого типа](#)
- [Обновить контактную информацию](#)

Тариф и период: Услуги хостинга (12 мес.) [изменить](#) [посмотреть параметры текущего тарифа](#)

Окончание периода: 2018.07.17

Баланс хостинга: 0 Р.

оплата услуг, подтвердить оплату Сбербанком

вопросы оплаты, отчетность

покупка дополнительных услуг

Договор 2443833/17Н: режим оферты [подписать/посмотреть договор](#)
договор с ЗАО "Ин-Соль", партнер RU-CENTER 330/NIC-REG
[заказать доставку договора почтой](#)

Лирическое отступление.

Чем меньше наворотов в интерфейсе, тем дешевле вам обойдется содержание вашего хостинга. На непродолжительный срок, например, для тренировок (от 10 дней до месяца) можно найти и бесплатный вариант. Простенькие (платные) варианты от 30 руб./мес. – зависит от срока оплаты (если оплачивать сразу за пару лет, то скидки могут быть существенными). Как бы там ни было, надо иметь ввиду, что то, что дороже, то не всегда будет лучше. Но при покупке недорогого хостинга можно встретиться с ограничением на количество создаваемых баз данных, на объем хранимой информации, на возможность доступа к БД с любого IP. Ограничение по IP означает, что ваша программа на C# будет иметь доступ к БД только с компьютера с таким IP, который вы сами указали в клиентской панели. Иногда это удобно – например, вы хотите, чтобы студенты могли проходить тестирование только из учебной аудитории, а не у себя дома и, конечно, это обеспечивает безопасность БД (никто со стороны не проникнет). Зачастую это создаёт ненужную проблему, так как вы не можете просто так перенести программу в другое место, так как доступ к БД будет прекращён, пока вы лично его не разрешите (введя IP нового места работы). Этую проблему можно преодолеть, купив тариф подороже или просто у другого хостинг-провайдера. Например, на 1GB.ru даже на базовых тарифах нет ограничений ни на количество баз данных ни на IP-адреса, а на shneider-host.ru присутствуют оба эти ограничения на начальном тарифе. Однако интерфейс пользователя, наполненность клиентской панели и отзывчивость службы поддержки на высоте именно на shneider-host.ru.

Внимание! Существуют хостинги баз данных без поддержки сайта, то есть БД MySQL вы там можете создать, хранить и работать с ней, а html, css и js там не размещают – попробуйте поискать стоимость услуг и сравнить дополнительные опции таких вариантов. Например, для тренировок очень удобны сервисы с размещением базы без оплаты (хоть и на ограниченный период)

[https://www.freysqldatabase.com/](https://www.freysqlatabase.com/) или <https://www.freemysqlhosting.net/>,

так как там довольно легко пройти регистрацию, сразу же получить доступ к **phpMyAdmin** и приступить к созданию БД, причем первые 10 дней бесплатно с поддержкой всего функционала, включая и динамический доступ по IP. Впоследствии, вы можете просто оплатить этот сервис для дальнейшей работы на нём или поискать, что-нибудь более функциональное или приемлемое по цене.

Теперь собственно создание.

Нажмите найденное (там, где что-то про Базы данных) и следуйте, как говорится, инструкциям. На некоторых сервисах всё достаточно удобно, понятно и логично устроено, но так не везде – просто нужно быть внимательнее. Не с первой так со второй попытки всё получится, например, тут всё уж очень аскетично:

имя	пользователь	тип	активна	действия
gb_mytest Москва, РТКомм/СТЕК	gb_mytest	mySQL 5.7.* x64 / l34 / aux1 / NEW_PASS	да	удалить новый пароль управлять базой
gb_psis Москва, РТКомм/СТЕК	gb_psis	mySQL 5.7.* x64 / l34 / aux2 / NEW_PASS	да	удалить новый пароль управлять базой
gb_x_pcoding Москва, РТКомм/СТЕК	gb_x_pcoding	mySQL 5.7.* x64 / l34 / aux1 / NEW_PASS	да	удалить новый пароль управлять базой изменить все mySQL пароли

Когда новая БД будет создана, то вы должны себе скопировать адрес сервера для её хранения, имя БД, логин пользователя и пароль. Напомню, как мы настраивали свою программу на подключение к БД в первой части:

```
db.Server = "mysql95.1gb.ru"; // хостинг БД
db.Database = "gb_psis"; // Имя БД
db.UserID = "gb_psis"; // Имя пользователя БД
db.Password = "ca8484adc89a"; // Пароль пользователя БД
```

Все параметры, кроме адреса сервера хостинга, вы можете настраивать сами во время создания новой БД. Отмечу, что, для корректной работы с кириллицей, важно правильно настроить кодировку – обратите внимание, что для корректной работы с C# в phpMyAdmin следует выбирать кодировку utf8_unicode_ci:

mysql95.1gb.ru » gb_psis » students

Обзор Структура SQL Поиск Вставить Экспорт Импорт Операции Три

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно	Действие
1	id_stud	int(11)			Нет	Hem	AUTO_INCREMENT	
2	name1	varchar(20)	utf8_unicode_ci		Нет	Hem		
3	name2	varchar(20)	utf8_unicode_ci		Нет	Hem		
4	name3	varchar(20)	utf8_unicode_ci		Нет	Hem		
5	group_num	int(11)			Нет	Hem		

↑ Отметить все / Снять выделение С отмеченными:

Версия для печати Связи Анализ структуры таблицы

Добавить 1 поле(я) В конец таблицы В начало таблицы После id_stud

+ Индексы

Информация

Используемое пространство	Статистика строк
Данные 16 КБ	Формат динамический
Индекс 32 КБ	Сравнение utf8_unicode_ci
Всего 48 КБ	Следующий автоматический индекс 14
	Создание Окт 28 2017 г., 07:38
	Последнее обновление Ноя 10 2017 г., 17:48

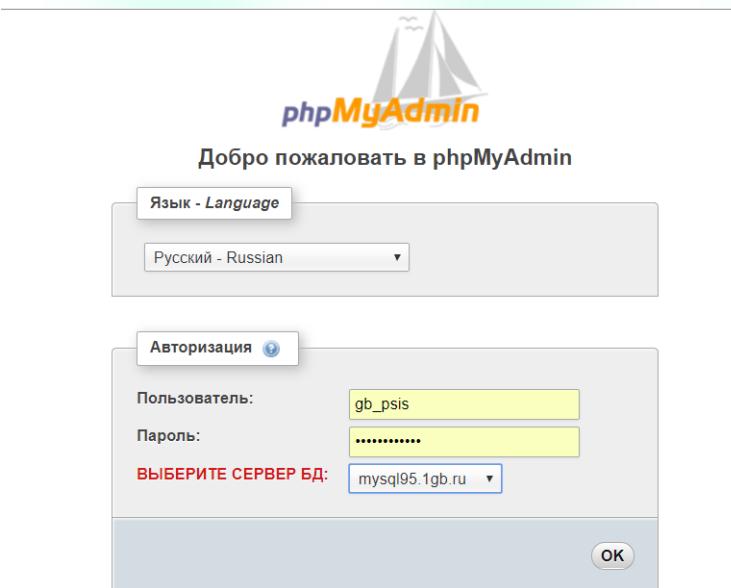
Причём выбор кодировки возможен как для всей БД, так и для отдельный полей индивидуально (в том числе и после создания):

Структура

Имя	Тип	Длина/значения	По умолчанию	Сравнение	Атрибуты
name1	VARCHAR	20	Нет	utf8_unicode_ci	

В принципе, сам инструмент phpMyAdmin можно скачать (<https://www.phpmyadmin.net/>) и установить на своём компьютере и работать локально (без подключения к сети).

Итак, вы зарегистрировались на хостинге, создали там базу данных MySQL, и собираетесь работать с данными. Вам будет предоставлена ссылка для входа в phpMyAdmin, перейдя по ней, вы встретитесь с окном авторизации:



Пройдите авторизацию с данными установленными ранее, и вы перейдёте к окну редактирования вашей базы, где уже можно будет добавлять таблицы, редактировать их содержимое, связывать поля и строить запросы:

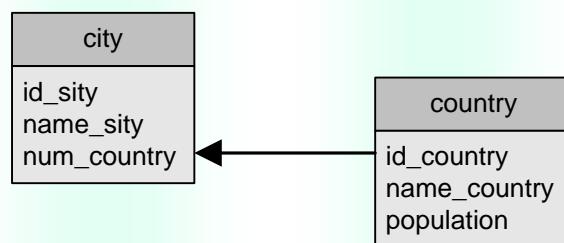
A screenshot of the phpMyAdmin main interface. The top navigation bar includes tabs for "Базы данных" (Databases), "SQL", "Состояние" (Status), "Экспорт" (Export), "Импорт" (Import), "Настройки" (Settings), "Синхронизировать" (Sync), "Переменные" (Variables), "Кодировки" (Charsets), and "Типы таблиц" (Table Types). The left sidebar shows databases "gb_psis" and "information_schema". The main content area has several tabs: "Основные настройки" (General Settings) showing "Текущий сервер" (Current Server) as "mysql95.1gb.ru" and "Сопоставление кодировки соединения с MySQL" (MySQL connection encoding mapping) as "utf8_unicode_ci"; "Настройки внешнего вида" (Visual Settings) showing "Язык" (Language) as "Русский - Russian" and "Тема" (Theme) as "pmahomme"; "Сервер баз данных" (Database Server) listing server details like "Сервер: mysql95.1gb.ru via TCP/IP", "Программа: MySQL", "Версия программы: 5.7.13-log - Source distribution", "Версия протокола: 10", "Пользователь: gb_psis@10.0.2.26", and "Кодировка сервера: UTF-8 Unicode (utf8)"; "Веб-сервер" (Web Server) listing "Apache/2.2.29 (Unix) PHP/5.3.28", "Версия клиента базы данных: libmysql - 5.5.30", and "PHP расширение: mysqli"; and "phpMyAdmin" tab listing links like "Информация о версии: 3.5.8.1", "Документация", "Вики", "Официальная страница phpMyAdmin", "Пожертвования", "Получение помощи", and "Список изменений".

Не забудьте сразу установить кодировку, соответствующую кодировке C#.

При первом запуске в левой колонке вы увидите доступ к виртуальной базе information_schema (которая содержит метаданные – информацию о других базах данных, которые поддерживает сервер MySQL – это несколько таблиц только для чтения, более подробно можно почитать тут: <http://www.rldp.ru/mysql/mysqlpro/schema.htm>) и к созданной вами, кликните по ней и приступите к созданию таблиц (кнопка слева «Создать таблицу»):

The screenshot shows the phpMyAdmin interface for the 'gb_psis' database. The left sidebar shows the database structure with 'groups' and 'students' listed. The main area displays the 'Структура' (Structure) tab for the 'gb_psis' database. It lists two tables: 'groups' and 'students'. The 'groups' table has 2 rows and 16 KB size. The 'students' table has 7 rows and 48 KB size. Below the table list, there is a 'Создать таблицу' (Create Table) dialog box with fields for 'Имя:' (Name:) and 'Количество столбцов:' (Number of columns:).

В моём случае уже видны две таблицы, так как я структуру базы данных создавал ранее. Давайте создадим две новые таблицы «Страны» (с полями идентификатор, название страны, население страны) и «Города» (с полями идентификатор, название города, ссылка на название страны) и свяжем их по полям:



Создадим новую таблицу:

The screenshot shows the 'Создать таблицу' (Create Table) dialog for the 'country' table. The 'Имя таблицы:' (Table Name:) field is set to 'country'. The 'Структура' (Structure) tab shows the following columns:

Имя	Тип	Длина/значения	По умолчанию	Сравнение	Атрибуты	Null	Индекс	A_I	Комментарии
id_country	INT		Нет			<input checked="" type="checkbox"/>	PRIMARY	<input checked="" type="checkbox"/>	
name_country	VARCHAR	20	Нет	utf8_unicode_ci		<input checked="" type="checkbox"/>	---	<input checked="" type="checkbox"/>	
population	INT		Как определен			<input checked="" type="checkbox"/>	---	<input checked="" type="checkbox"/>	
			0						

Below the structure, there are fields for 'Комментарии к таблице:' (Table comments), 'Тип таблицы:' (Table type: InnoDB), 'Сравнение:' (Collation: utf8_unicode_ci), and buttons for 'Сохранить' (Save) and 'Отмена' (Cancel).

Таблица должна быть типа InnoDB, чтобы была возможность поддерживать связи между таблицами базы данных. Обратите также внимание, что поле id_country следует установить как Primary и добавить параметр AUTO_INCREMENT (чтобы индекс сам увеличивался).

Возможные настройки параметра Индекс:

primary – первичный ключ;

unique – уникальный идентификатор (допускает NULL-значения);

fulltext – полнотекстовой индекс (поиск по всему тексту);

index – простой индекс.

Сохраните таблицу (клавиша «Сохранить») и снова кликните в левом столбце по ней – откроется режим просмотра структуры таблицы:

The screenshot shows the phpMyAdmin interface for the 'country' table in the 'gb_psis' database. The SQL tab contains the query: `SELECT * FROM country LIMIT 0, 30`. The results pane shows three columns: id_country, name_country, and population. The 'id_country' column is defined as int(11), primary key, AUTO_INCREMENT. The 'name_country' column is defined as varchar(20). The 'population' column is defined as int(11). Below the table, there are tabs for 'Информация' (Information) and 'Статистика строк' (Row statistics). The 'Информация' tab displays the following details:

Используемое пространство	Статистика строк
Данные 16 КБ	Формат динамический
Индекс 0 Байт	Сравнение utf8_unicode_ci
Всего 16 КБ	Следующий автоматический индекс 1
	Создание Ноя 11 2017 г., 07:18

Поля в таблице можно удалять, добавлять и изменять их настройки. Если вас что-то не устраивает в настройках полей, воспользуйтесь соответствующими клавишами в правой части таблицы. Если вы хотите добавить поля к таблице, то выберите соответствующее действие непосредственно под таблицей. Если вы хотите добавить данные вручную, то нажмите на клавишу «Вставить» в верхнем меню:

Наберите необходимые данные (обратите внимание, что номер `id_country` мы не пишем сами, так как поле настроено так, что будет самостоятельно увеличиваться) и нажмите OK для добавления записи. В таблицу будет добавлена одна строка и вам будет показан выполненный запрос:

На данной вкладке (SQL) вы можете самостоятельно создавать и запускать на исполнение запросы к вашей БД. Это может пригодиться, когда вы будете писать программу на C# – прежде чем добавлять запрос в код и испытывать его в рамках программы, можно его апробировать непосредственно в панели phpMyAdmin.

После добавления данных становится активной клавиша «Обзор», перейдите по ней и вы сможете просмотреть содержимое таблицы и, при необходимости, изменить данные:

The screenshot shows the phpMyAdmin interface for a MySQL database named 'gb_psis'. The left sidebar lists tables: 'country', 'groups', and 'students'. The main area shows the 'country' table structure with columns: id_country, name_country, and population. A query results grid displays one row: id_country 1, name_country 'Россия', and population 146000000. Below the table, there are two sets of 'Show' buttons.

Аналогичным образом создадим и вторую таблицу:

The screenshot shows the 'Create Table' dialog in phpMyAdmin. The table name is 'city'. It contains four columns: 'id_city' (INT), 'name_city' (VARCHAR(20)), 'num_country' (INT), and an unnamed fourth column (INT). The 'id_city' column is set as the primary key (PRIMARY). The table uses InnoDB engine and utf8_unicode_ci collation. The 'OK' button is highlighted.

И добавим в неё одну запись:

The screenshot shows the 'Insert' dialog for the 'city' table. It has three fields: 'id_city' (set to null), 'name_city' (set to 'Пермь'), and 'num_country' (set to 1). The 'OK' button is highlighted.

Обратите внимание, что пока таблицы не связаны, можно вручную проставить номер num_country (это ссылка на запись про страну).

Так как уже есть две таблицы, то между ними можно создать связь, для чего необходимо сначала добавить в индекс связываемые поля таблиц. Выберите пункт «Структура», выделите необходимое для индексирования поле (num_country) и нажмите на клавишу «Индекс» в нижней части справа:

Screenshot of phpMyAdmin showing the structure of the 'city' table. The 'num_country' field is selected for indexing.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно	Действие						
1	id_city	int(11)			Нет	Нем	AUTO_INCREMENT							
2	name_city	varchar(20)	utf8_unicode_ci		Нет	Нем								
3	num_country	int(11)			Нет	Нем								

Версия для печати Связи Анализ структуры таблицы Индекс

+ Добавить [1] поле(я) В конец таблицы В начало таблицы После id_city OK

+ Индексы

В ответ выполнится запрос, который, кстати, можно было бы и вручную набрать во вкладке SQL и самостоятельно запустить на исполнение:

Screenshot of phpMyAdmin showing the successful execution of the ALTER TABLE command to add an index on the num_country column.

SQL-запрос был успешно выполнен

```
ALTER TABLE `city` ADD INDEX (`num_country`);
```

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно	Действие						
1	id_city	int(11)			Нет	Нем	AUTO_INCREMENT							
2	name_city	varchar(20)	utf8_unicode_ci		Нет	Нем								
3	num_country	int(11)			Нет	Нем								

Аналогичные действия проделайте для поля id_country:

Screenshot of phpMyAdmin showing the successful execution of the ALTER TABLE command to add an index on the id_country column.

SQL-запрос был успешно выполнен

```
ALTER TABLE `country` ADD INDEX (`id_country`);
```

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно	Действие						
1	id_country	int(11)			Нет	Нем	AUTO_INCREMENT							
2	name_country	varchar(20)	utf8_unicode_ci		Нет	Нем								
3	population	int(11)			Нет	0								

После назначения индексируемых полей можно установить между ними связь, для чего выбираем таблицу city, выбираем режим структуры и кликаем на клавишу «Связи» в нижней части:

Screenshot of phpMyAdmin showing the 'city' table structure with the 'num_country' field selected for indexing.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно
1	id_city	int(11)			Нет	Нем	AUTO_INCREMENT
2	name_city	varchar(20)	utf8_unicode_ci		Нет	Нем	
3	num_country	int(11)			Нет	Нем	

Версия для печати Связи Анализ структуры таблицы

В следующем окне выбираем соединение поля num_country с полем name_country:

Столбец	Ограничение внешнего ключа (INNODB)
'id_city'	
'name_city'	Индекс не определен!
'num_country'	'gb_psis`.`city`.`id_city' 'gb_psis`.`city`.`num_country' 'gb_psis`.`country`.`id_country' 'gb_psis`.`country`.`name_country' 'gb_psis`.`groups`.`id_group' 'gb_psis`.`students`.`id_stud' 'gb_psis`.`students`.`group_num'

В следующем окне определите дополнительные действия для таблицы:

Столбец	Ограничение внешнего ключа (INNODB)
'id_city'	
'name_city'	Индекс не определен!
'num_country'	'gb_psis`.`country`.`name_country' ON DELETE RESTRICT ON UPDATE RESTRICT

Выражения ON DELETE и ON UPDATE внешних ключей используются для указания действий, которые будут выполняться при удалении строк родительской таблицы (ON DELETE) или изменении родительского ключа (ON UPDATE). В данном случае приложению будет запрещено удалять или изменять родительский ключ, если существуют ссылающиеся на него дочерние ключи. Нажмите сохранить и получите следующий ответ:

SQL-запрос был успешно выполнен

```
ALTER TABLE `city` ADD FOREIGN KEY (`num_country`) REFERENCES `gb_psis`.`country`(`id_country`);
```

Связи

Столбец	Ограничение внешнего ключа (INNODB)
'id_city'	
'name'	Индекс не определен!
'num_country'	'gb_psis`.`country`.`id_country' ON DELETE RESTRICT ON UPDATE RESTRICT

После установки связей в поле num_country уже нельзя будет вручную заносить данные, их нужно будет выбирать из выпадающего списка, формирующегося из поля id_country, на которое мы и установили ссылку:

Столбец	Тип	Функция	Null	Значение
id_city	int(11)			
name	varchar(20)			Кунгур
num_country	int(11)			1

Итак, таблицы созданы, связи между соответствующими полями таблиц наложены, теперь уже можно приступить к разработке приложения по работе с базой данных. Основываясь на опыте, полученном в первой части, попробуйте самостоятельно сделать запросы по добавлению и удалению записей, по работе со связанными таблицами, по изменению записей.

Прежде чем писать запросы в коде программы можно провести тесты на самой БД в панели phpMyAdmin. Предположим у меня вот такое содержимое таблиц:

	id_country	name_country	population		id_city	name_sity	num_country
	1	Россия	146000000		1	Пермь	1
					2	Кунгур	1

Выделите таблицу city и перейдите во вкладку SQL, там будет стоять запрос по умолчанию на выборку всего содержимого из таблицы с ограничением «первые 30»:

```
SELECT *
FROM `city`
WHERE 1
LIMIT 0 , 30
```

Давайте доработаем его и выведем только отсортированные по алфавиту названия городов:

```
SELECT `name_sity` FROM `city` ORDER BY `name_sity`
```

Для того чтобы не писать названия полей вручную, пользуйтесь таблицей «Столбцы» в правой части вкладки SQL (выделить столбец и нажать клавишу переноса «<<>>»):



Ряд клавиш под полем редактирования самого запроса предоставляют нам шаблоны написания запросов, для пробы нажмите на клавишу INSERT:



Вам будет предоставлен шаблон на добавление записи:

```
INSERT INTO `city`(`id_city`, `name_sity`, `num_country`)
VALUES ([value-1], [value-2], [value-3])
```

где поля value вы можете заполнить сами и запустить уже готовый запрос на исполнение. Напомню, что поле id_city у нас с автоинкрементом, поэтому туда номер не следует писать вручную (он будет добавляться автоматически), а шаблон можно исправить так:

```
INSERT INTO `city`(`id_city`, `name_sity`, `num_country`)
VALUES (null, 'Москва', 1)
```

или так:

```
INSERT INTO `city`(`name_sity`, `num_country`) VALUES ('Ижевск', 1)
```

Попробуйте работу запросов и убедитесь в изменении записей в таблицах. Следует отметить, что базы данных чувствительны к использованию правильных символов в качестве апострофов. Для примера приведу пару скриншотов, которые демонстрируют использование корректных символов:



```
1 DELETE FROM `city` WHERE `name_city` = 'Рязань'
```

SELECT * | SELECT | INSERT | UPDATE | DELETE | Очистить

Ну, и, наконец, можно испытать запрос, адресованный на извлечение данных из связанных таблиц. Пусть нужно вывести данные о названиях городов и их странах. Если мы оформим простой запрос:

```
SELECT name_sity, num_country FROM city
```

то получим неудовлетворительный результат:

	← ↑ →	▼	name_sity	num_country
<input type="checkbox"/>			Пермь	1
<input type="checkbox"/>			Кунгур	1
<input type="checkbox"/>			Москва	1
<input type="checkbox"/>			Ижевск	1
<input type="checkbox"/>			Рязань	1
<input type="checkbox"/>			Лондон	3

Очевидно, что вместо порядковых номеров в последнем столбце хотелось бы видеть названия самих стран. Исправьте SQL-запрос так:

```
SELECT city.name_sity, country.name_country FROM country  
INNER JOIN city ON country.id_country = city.num_country
```

и получите приемлемый вариант:

name_sity	name_country
Пермь	Россия
Кунгур	Россия
Москва	Россия
Ижевск	Россия
Рязань	Россия
Лондон	Великобритания

Если вы зарегистрировали для себя хостинг баз данных MySQL и смогли подготовить базу данных с указанными таблицами, то, после изучения способов работы средствами C# с базой данных, вы смогли бы подготовить следующее небольшое приложение по работе с этой базой данных и с запросом к связанным таблицам:

```
using System;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace MySQLQuery
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        MySqlConnection conn;

        private void Form1_Load(object sender, EventArgs e)
        {
            MySqlConnectionStringBuilder db;
            db = new MySqlConnectionStringBuilder();
            db.Server = "mysql95.1gb.ru"; // хостинг БД
            db.Database = "gb_psis"; // Имя БД
            db.UserID = "gb_psis"; // Имя пользователя БД
            db.Password = "ca8484adc89a"; // Пароль пользователя БД
            db.CharacterSet = "utf8"; // Кодировка Базы Данных

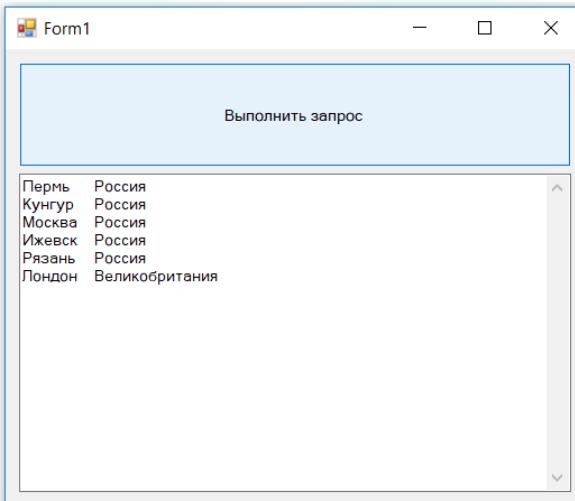
            conn = new MySqlConnection(db.ConnectionString);
            try
            {
                conn.Open();
                MessageBox.Show("Подключение к БД установлено");
                btnSQL.Enabled = true;
            }
            catch (Exception ex)
            {
                MessageBox.Show("Проблемы с подключением к БД \n\r" + ex.ToString());
            }
        }

        private void btnSQL_Click(object sender, EventArgs e)
        {
            string sql;
            sql = "SELECT city.name_sity, country.name_country FROM country ";
            sql += "INNER JOIN city ON country.id_country = city.num_country";
            MySqlCommand cmd = new MySqlCommand(sql, conn);
            MySqlDataReader reader = cmd.ExecuteReader();
            textBox1.Clear();
            while (reader.Read())
            {
                textBox1.Text += reader[0].ToString() + '\t' +
                    reader[1].ToString() + Environment.NewLine;
            }
            reader.Close();
        }
    }
}
```

Данные, приведённые в строке соединения действующие, специально для проведения испытаний обучающимися. Не злоупотребляйте этим, не пишите вредоносных программ и не удаляйте таблицы и записи в них – это нужно для последующих поколений студентов...

И не забывайте про подключение библиотеки для работы с БД MySQL (`MySql.Data.MySqlClient`).

Дизайн программы и результаты её работы таковы:



В программе всего две процедуры:

- `Form1_Load` – обеспечивает автоматическое подключение к БД с выдачей информационного сообщения о результатах подключения;
- `btnSQL_Click` – собственно реализует SQL-запрос с выдачей результатов в многострочное текстовое поле.

Этого материала вполне достаточно, чтобы понимать порядок разработки приложения по работе с сетевой базой данных, осознавать встречающиеся затруднения и искать способы их преодоления самостоятельно в сети интернет.

Часть 3. Некоторые примеры структуры базы данных и запросов к ней

Предположим в парикмахерской для учёта работ, производимых мастерами, ведётся БД, в которой есть три таблицы («Клиенты» – Clients, «Мастера» – Masters, «Работы» – Services) – базовая таблица Clients с индексами:

+ Параметры					
		← T →	↓	id_client	name_client
<input type="checkbox"/>				1	Иванов
<input type="checkbox"/>				2	Петров
<input type="checkbox"/>				3	Сидоров

и две таблицы, где хранится информация о содержимом этих индексов:

The screenshot shows the phpMyAdmin interface with three tables listed on the left: services, masters, and clients. The clients table is currently selected. It has two indexes: num_service and num_master. The services table has one index: id_service. The masters table has one index: id_master.

Обсудим, как правильно связать эти таблицы и, какой нужно сделать запрос, чтобы результаты выводились в приемлемом для человека виде, например, так:

Иванов	МастерВ	стрижка
Петров	МастерГ	стрижка
Сидоров	МастерА	покраска

Начнём с назначения индексных полей. В базовой таблице это сразу два поля – num_service и num_master, так как именно там будут храниться ссылки, поэтому в phpMyAdmin выберите таблицу Clients, перейдите на вкладку Структура, выделите «галочку» для данных столбцов и нажмите клавишу «Индекс» в правом нижнем углу:

The screenshot shows the 'Structure' tab for the clients table in phpMyAdmin. In the 'Indexes' section, two checkboxes are checked for the columns num_service and num_master. The 'Action' button is visible at the bottom right.

Аналогичные действия осуществите и с оставшимися двумя таблицами – Services для поля id_services:

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно	Действие
<input checked="" type="checkbox"/>	1 id_service	int(11)			Нет	Нем	AUTO_INCREMENT	
<input type="checkbox"/>	2 name_service	int(11)			Нет	Нем		

Отметить все / Снять выделение С отмеченными:

Версия для печати Связи Анализ структуры таблицы Индекс

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно	Действие
<input checked="" type="checkbox"/>	1 id_master	int(11)			Нет	Нем	AUTO_INCREMENT	
<input type="checkbox"/>	2 name_master	varchar(20)	cp1251_general_ci		Нет	Нем		

Отметить все / Снять выделение С отмеченными:

Версия для печати Связи Анализ структуры таблицы Индекс

и Masters для поля id_masters:

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Дополнительно	Действие
<input checked="" type="checkbox"/>	1 id_master	int(11)			Нет	Нем	AUTO_INCREMENT	
<input type="checkbox"/>	2 name_master	varchar(20)	cp1251_general_ci		Нет	Нем		

Отметить все / Снять выделение С отмеченными:

Версия для печати Связи Анализ структуры таблицы Индекс

Далее выделяем таблицу Clients, нажимаем в нижней части «Связи», указываем индексные столбцы, назначаем связи и нажимаем «Сохранить»:

The screenshot shows the 'clients' table in the 'gb_psis' database. The 'Связи' (Relationships) tab is active. It displays four foreign key definitions:

- Столбец: id_client, Ограничение внешнего ключа (INNODB): name_client Индекс не определен!
- Столбец: num_service, Ограничение внешнего ключа (INNODB): num_service 'gb_psis'.servieces.id_service' ON DELETE RESTRICT ON UPDATE RESTRICT
- Столбец: num_master, Ограничение внешнего ключа (INNODB): num_master 'gb_psis'.masters.id_master' ON DELETE RESTRICT ON UPDATE RESTRICT

Below the relationships, the SQL tab shows the executed commands:

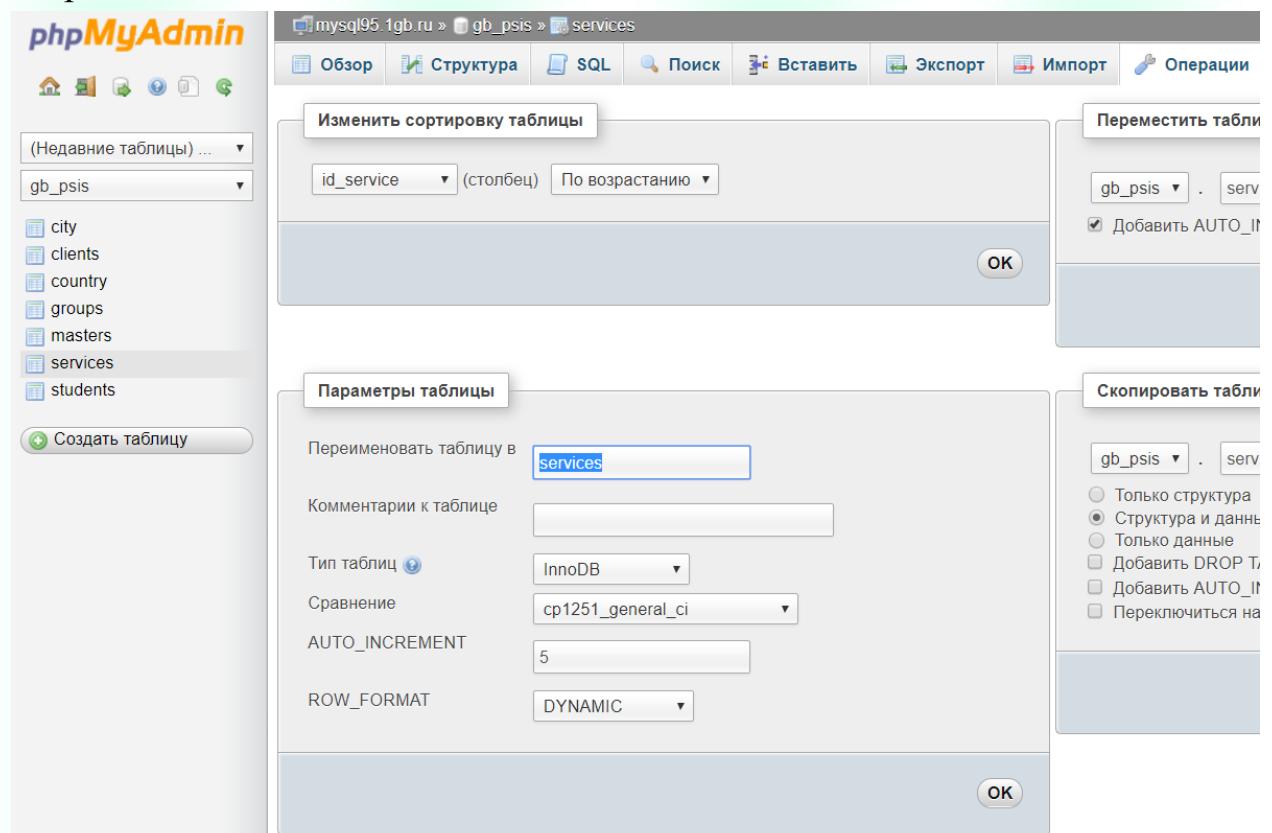
```

ALTER TABLE `clients` ADD FOREIGN KEY ( `num_service` ) REFERENCES `gb_psis`.`servieces` (
`id_service`
) ON DELETE RESTRICT ON UPDATE RESTRICT;
ALTER TABLE `clients` ADD FOREIGN KEY ( `num_master` ) REFERENCES `gb_psis`.`masters` (
`id_master`
) ON DELETE RESTRICT ON UPDATE RESTRICT;

```

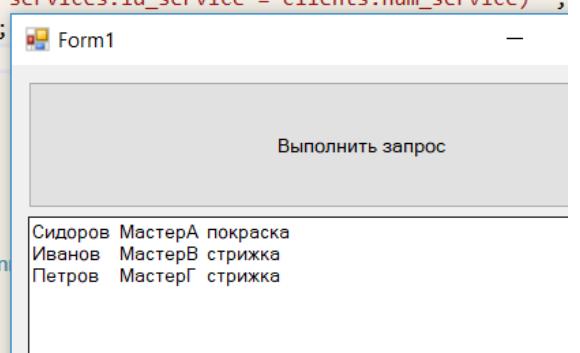
The bottom section of the interface shows the 'Связи' (Relationships) tab again, identical to the one above.

Обратите внимание, что при наборе названия таблицы Services я допустил ошибку – SerEces. В этом нет никакой опасности для дальнейшей работы, при переименовании таблиц связи остаются. Для переименования таблицы – выделите таблицу, перейдите на вкладку «Операции», в разделе «Параметры таблицы» внесите новой имя и нажмите «OK»:



Ну и наконец, рассмотрим сложный запрос с заменой индексов настоящими названиями:

```
private void button1_Click(object sender, EventArgs e)
{
    string sql;
    sql = "SELECT clients.name_client, masters.name_master, services.name_service FROM masters ";
    sql += "INNER JOIN (services INNER JOIN clients ON services.id_service = clients.num_service) ";
    sql += "ON masters.id_master = clients.num_master";
    MySqlCommand cmd = new MySqlCommand(sql, conn);
    MySqlDataReader reader = cmd.ExecuteReader();
    textBox1.Clear();
    while (reader.Read())
    {
        textBox1.Text += reader[0].ToString() + '\t' +
            reader[1].ToString() + '\t' +
            reader[2].ToString() + Environment.NewLine;
    }
    reader.Close();
}
```



Обратите внимание, что в таком запросе приходится дважды применять оператор INNER JOIN, который возвращает пересечение двух множеств по указанным критериям.

Дорогу осилит идущий, удачи...

Версия от 19.11.2017 14:00