Министерство образования и науки Самарской области

ГБПОУ «ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ КОЛЛЕДЖ»

**КУРСОВОЙ ПРОЕКТ**

**Разработка удаленной базы данных для автоматизации работы логистики ООО «Самарская алюминиевая компания»**

ПМ.11 РАЗРАБОТКА, АДМИНИСТРИРОВАНИЕ И ЗАЩИТА БАЗ ДАННЫХ

09.02.07 Информационные системы и программирование

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Обучающийся | | | | | |  |  |  | А.В. Третьяков |
|  | | | | | |  | *подпись* |  | *И.О. Фамилия* |
|  |  |  | 20 | 19 |  |  |  |  |  |
|  | | | | | |  |  |  |  |
| Оценка выполнения и защиты курсового проекта | | | | | | | | |  |
|  | | | | | | | | |  |
|  | | | | | | | | |  |
| Руководитель курсового проекта | | | | | |  |  |  | Е.В.Третьякова |
|  | | | | | |  | *подпись* |  | *И.О. Фамилия* |
|  |  |  | 20 | 19 |  |  |  |  |  |
|  | | | | | |  |  |  |  |

г. Самара, 2019

**ГБПОУ «ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ КОЛЛЕДЖ»**

**КАЛЕНДАРНЫЙ ПЛАН  
выполнения курсового проекта**

ПМ.11 РАЗРАБОТКА, АДМИНИСТРИРОВАНИЕ И ЗАЩИТА БАЗ ДАННЫХ

09.02.07 Информационные системы и программирование

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Обучающемуся | | 3 | курса | Исп-333 | группы | Третьяков А.В. |
|  | |  |  |  |  | ФИО |
| По теме |  | | | | | |
|  | | | | | | |

| **№**  **этапа**  **работы** | **Содержание этапов работы** | **Плановый срок выполнения этапа** | **Отметка о выполнении**  **этапа (подпись руководителя)** |
| --- | --- | --- | --- |
|  | Написано введение курсового проекта. | 25.11.19 |  |
|  | Проведен анализ: объекта, процесса, предметной области как объекта информатизации; информационного обеспечения объекта информатизации. | 26.11.19 |  |
|  | Описаны: организационная и функциональные схемы объектов информатизации; категории пользователей и потоков данных объекта информатизации. | 26.11.19 |  |
|  | Проведен анализ функциональных требований программного продукта и входных-выходных данных. | 27.11.19 |  |
|  | Описана структура данных и проведена ее оптимизация. Разработана схема данных. | 27.11.19 |  |
|  | Описано клиент-серверное взаимодействие. | 27.11.19 |  |
|  | Проведен анализ и выбор программного обеспечения для разработки программы и СУБД. | 27.11.19 |  |
|  | Разработана удаленная база данных для программного продукта. | 28.11.19 |  |
|  | Разработаны и реализованы запросы к БД и функциональные требования к программному продукту. | 2.12.19 |  |
|  | Разработан интерфейс программы, проведена интеграция всех модулей программы и СУБД. | 2.12.19 |  |
|  | Проведено тестирование и отладка программного продукта. Программа готова к эксплуатации. | 3.12.19 |  |
|  | Написано заключение курсового проекта. | 3.12.19 |  |
|  | Подготовлена презентация и речь для защиты. | 3.12.19 |  |

|  |  |  |
| --- | --- | --- |
| Студент |  | А.В. Третьяков |
| Руководитель |  | Е.В. Третьякова |

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc26361802)

[ГЛАВА 1 АНАЛИТИЧЕСКАЯ ЧАСТЬ 6](#_Toc26361803)

[1.1 Анализ объекта, процесса, предметной области 6](#_Toc26361804)

[1.2 Анализ информационного обеспечения объекта информатизации 6](#_Toc26361805)

[1.3 Организационная и функциональная схемы объектовинф орматизации 7](#_Toc26361806)

[1.4 Описание категорий пользователей и потоков данных 7](#_Toc26361807)

[ГЛАВА 2 ПРЕДПРОЕКТНАЯ ЧАСТЬ 8](#_Toc26361808)

[2.1 Анализ функциональных требований 8](#_Toc26361809)

[2.2Анализ входных и выходных данных 8](#_Toc26361810)

[2.3Анализ структуры данных 8](#_Toc26361811)

[2.4 Разработка и оптимизация схемы данных 9](#_Toc26361812)

[2.5 Описание клиент-серверного взаимодействия 9](#_Toc26361813)

[2.6 Анализ и обоснование выбора программного обеспечения 10](#_Toc26361814)

[2.7Анализ выбора СУБД 10](#_Toc26361815)

[2.8 Разработка тестового сценария и методики испытаний программного продукта 10](#_Toc26361816)

[ГЛАВА 3 ПРОЕКТНАЯ ЧАСТЬ 12](#_Toc26361817)

[3.1Разработка интерфейса программного продукта 12](#_Toc26361818)

[3.2 Разработка удаленной БД 15](#_Toc26361819)

[3.3Реализация функциональных требований 17](#_Toc26361820)

[3.4 Интеграция программных модулей 19](#_Toc26361821)

[3.5 Тестирование и отладка программного продукта 20](#_Toc26361822)

[ЗАКЛЮЧЕНИЕ 21](#_Toc26361823)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 22](#_Toc26361824)

[ПРИЛОЖЕНИЕ А Структурная схема программного продукта 23](#_Toc26361825)

[ПРИЛОЖЕНИЕ Б Структурная схема базы данных 24](#_Toc26361826)

[ПРИЛОЖЕНИЕ В Листинг кода программного продукта 26](#_Toc26361827)

[ПРИЛОЖЕНИЕ Г Листинг кода запросов в базе данных 35](#_Toc26361828)

# ВВЕДЕНИЕ

**ООО «Самарская алюминиевая компания»** — Компания которая занимается оптовой торговлей алюминиевыми полуфабрикатами.

**Актуальность курсового проекта** — В современном мире, за последнее время объемы информации, хранящейся на электронных носителях увеличились, а значит остро встала проблема ее передачи. Для этого можно использовать много способов, от передачи USB устройств из рук в руки, до электронной почты, но не один из этих способов не является максимально практичным. Передавать устройства из рук в руки неудобно и требует времени, не говоря о том, что может понадобиться передать их в другой город и даже в другую страну. Почта же кажется идеальным вариантом, но на почте могут копиться тонны, не относящейся к работе информации, поэтому я и решил создать данную программу.

**Цель курсового проекта:** Разработка удаленной базы данных для отдела логистики ООО «Самарская алюминиевая компания», а так-же программы для управления этой БД.

**Задачи:**

* Создать базу данных;
* произвести нормализацию структуры базы данных;
* произвести перенос данных из имеющейся базы в MYSQL в созданную удаленную базу данных;
* произвести правки в программном продукте для реализации работы с удаленной базой данных;
* произвести разработку интерфейса, работающего с удаленной базой данных;
* произвести тестирование и отладку работы модулей программного продукта.

**Объект исследования**: Управление логистикой в компании ООО «Самарская алюминиевая компания».

**Предмет исследования:** удаленная база данных и программный продукт для автоматизации управление логистикой в компании ООО «Самарская алюминиевая компания».

Курсовой проект основан на **гипотезе,** согласно которой автоматизируемый процесс на рассматриваемом предприятии будет максимально эффективен если:

* Разработаны таблицы базы данных в соответствии с проведенным анализом организационной и функциональной схемы объектов информатизации.
* Разработана схема данных в соответствии с категориями пользователей и потоками входных и выходных данных.
* Проведен анализ и выбор программного обеспечения для реализации СУБД.
* Разработана база данных в выбранной системе управления удаленными базами данных.
* Реализован интерфейс и процесс взаимодействия программного продукта с удаленной базой данных.
* Написано приложение для администрирования данных.
* Разработана форма аутентификации для обеспечения защиты информации в базе данных.

То в таком случае:

1. Программный продукт сможет значительно облегчить обмен информацией, как в одном офисе, так и в компании в целом.
2. Программный продукт сможет упростить управление логистическими функциями.

# ГЛАВА 1 АНАЛИТИЧЕСКАЯ ЧАСТЬ

## 1.1 Анализ объекта, процесса, предметной области

 ООО «Самарская алюминиевая компания» - компания с большим опытом на рынке, имеющая контракты с многими гос. Организациями однако не имеющая автоматизированной системы для работы в спектре логистики.

## 1.2 Анализ информационного обеспечения объекта информатизации

Все сотрудники «Сам. Ал. Ко.» используют такие программные продукты, как:

* Операционная система Windows 10;
* 1C Предприятие.
* Google chrome
* Microsoft Exel
* Open office
* Microsoft Onedrive
* Microsoft Outlook

А так-же такое аппаратное обеспечение как: Стандартный ПК, Принтер, Ксерокс

## 1.3 Организационная и функциональная схемы объекто-информатизации

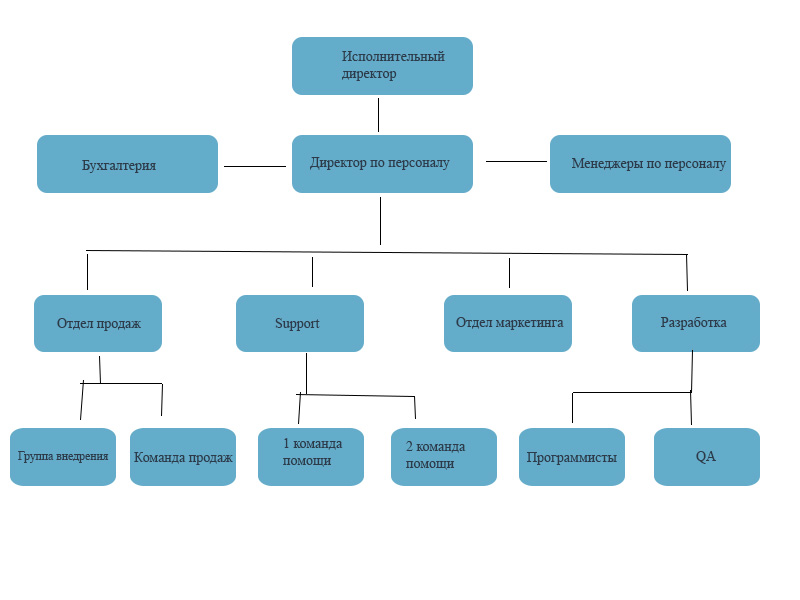


Рисунок 1 – Организационная схема \*\*\*.

## 1.4 Описание категорий пользователей и потоков данных

Сотрудники отдела логистики.

# ГЛАВА 2 ПРЕДПРОЕКТНАЯ ЧАСТЬ

## 2.1 Анализ функциональных требований

Программный продукт должен выполнять следующие функции/соответствовать следующим функциональным требованиям:

* Иметь базу данных со всей необходимой информацией
* Иметь удобный интерфейс для просмотра и изменения базы данных

## 2.2Анализ входных и выходных данных

Входными данными программного продукта являются:

* Информация о водителях
* Информация о складах
* Информация о товарах перевозки
* Информация о машинах

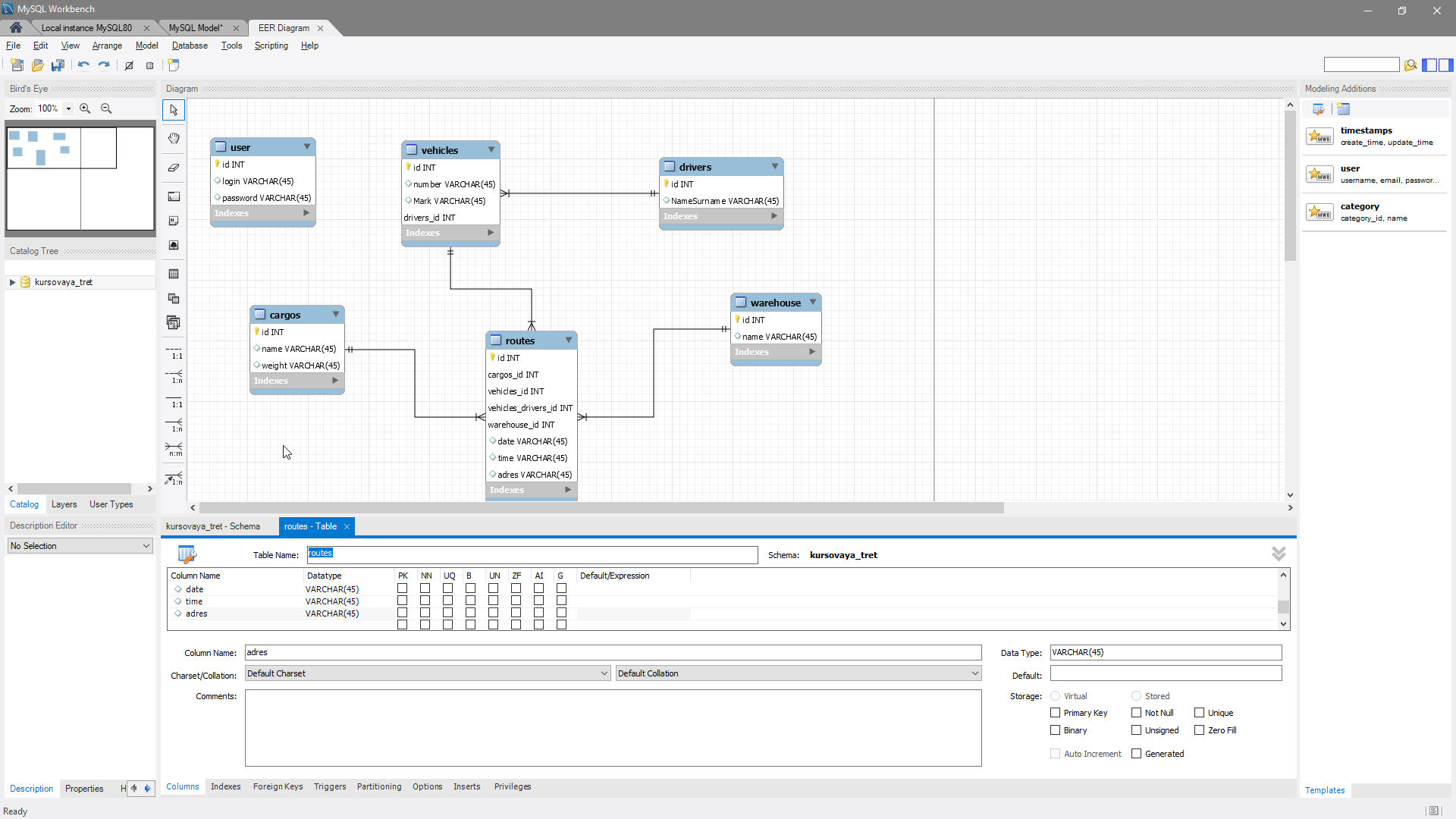
Выходными данными программного продукта являются:

* Составленная информация о маршрутах

## 2.3Анализ структуры данных

База данных имеет таблицы с товарами, водителями, складами и машинами в так же пользователями, все они за исключением последней объединяются в таблицу с маршрутами, где назначается время, дата и адрес доставки. Таблица с пользователями требуется для авторизации в программе.

## 2.4 Разработка и оптимизация схемы данных



## 2.5 Описание клиент-серверного взаимодействия

Из программы в базу данных задаются запросы о просмотре или вводе данных из БД. Исходя из запроса БД либо выдает данные, которые выводятся программный продукт либо вносит данные введенные в программе в необходимую таблицу.

## 

## 2.6 Анализ и обоснование выбора программного обеспечения

Для написания программного продукта был выбран язык C# так как является оптимальным выбором для написания ПО с пользовательским интерфейсом и программное обеспечение VisualStudio для удобного взаимодействия с выше-приведенным языком.

## 2.7Анализ выбора СУБД

Для разработки базы данных был выбран СУБД MYSQLWORKBENCH, поскольку mysql является одним из самых распространенных типов БД со множеством ПО для работы с ним.

## 2.8 Разработка тестового сценария и методики испытаний программного продукта

Тестовый сценарий, представленный в таблице 1, предназначен для проверки корректности работы программного продукта.

*Таблица 1*

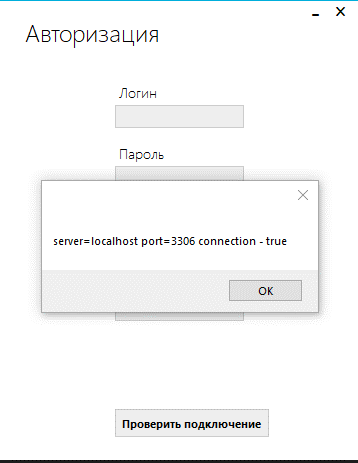
Тестовый сценарий программного продукта

| **Действия пользователя** | **Ожидаемый результат** |
| --- | --- |
| Ввод данных подключения к БД | Появляется сообщение о сохранении данных подключения |
| Проверка отображения данных из БД | Данные выводятся в datagridview |
| Проверка ввода данных в БД | Данные появляются в БД |
| Проверка корректной работы интерфейса | Ошибок в работе нет |

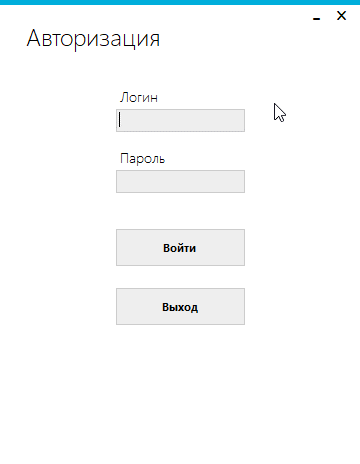
# ГЛАВА 3 ПРОЕКТНАЯ ЧАСТЬ

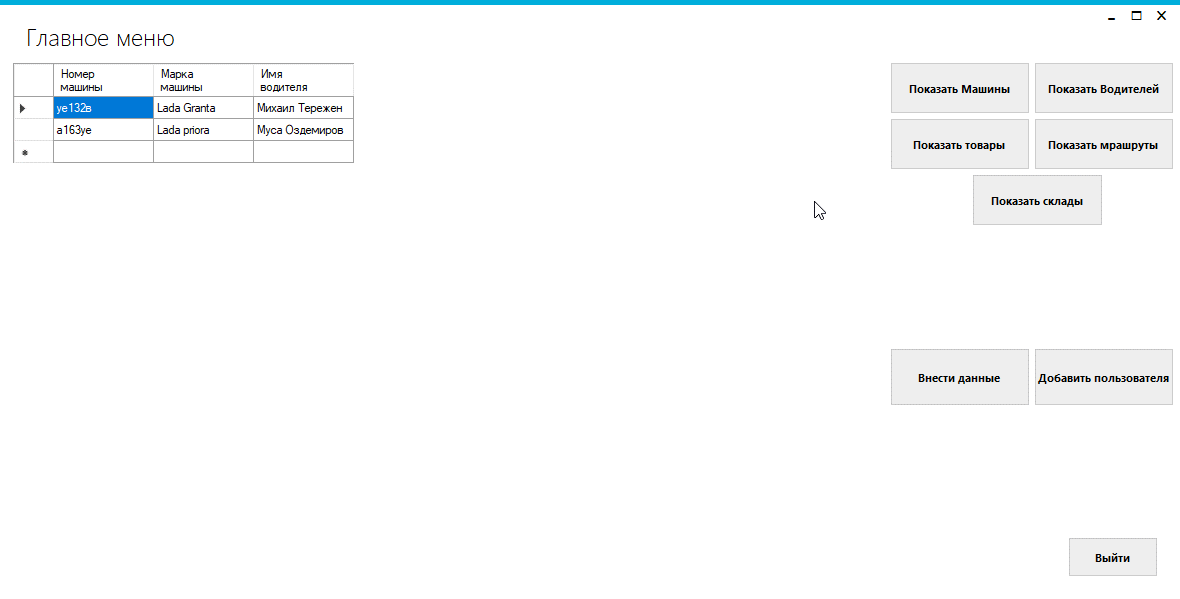
## 3.1Разработка интерфейса программного продукта

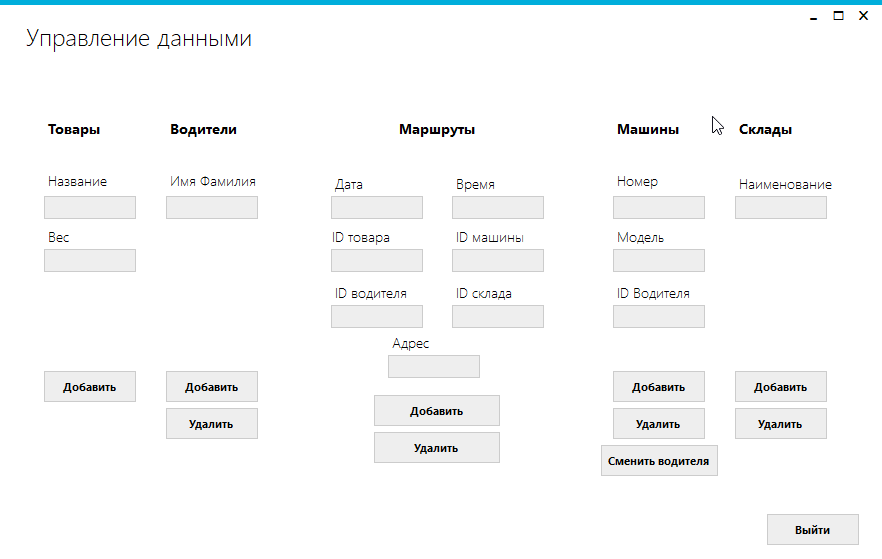
Прверка подключения



**Окно авторизации**



**Окно отображения данных**

**Окно ввода данных**

## 3.2 Разработка удаленной БД

БД была создана в СУБД MYSQLWORKBENCH данные для нее были взяты из EXEL файлов отдела логистики, так же были сделаны связи между таблицами для их синхронизации на рисунке приведенном ниже показывается структура данных в таблице отображаемая в MYSQLWORKBENCH

Схема связи данных

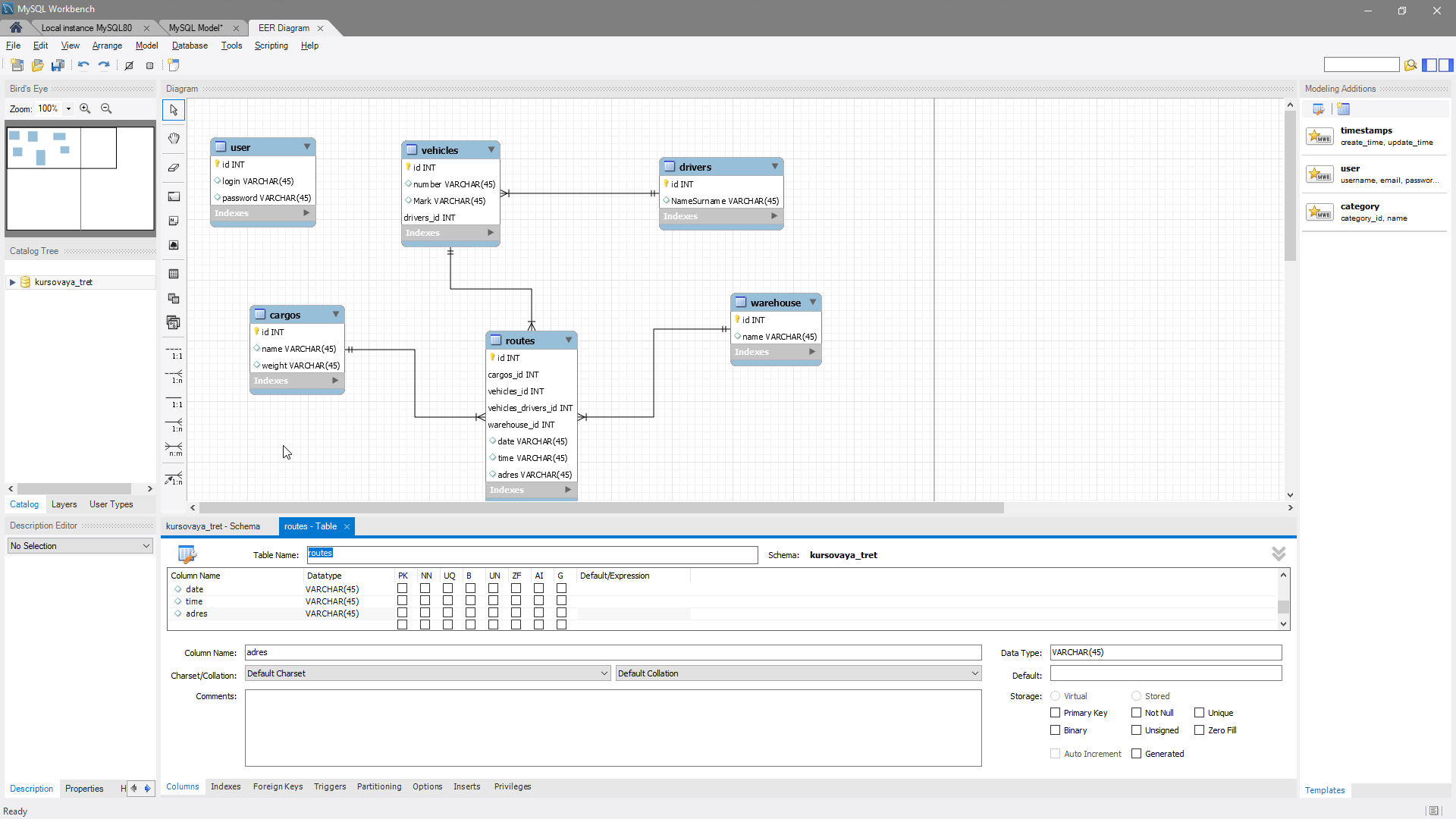


Таблица со складами

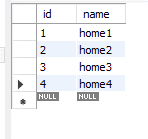


Таблица с машинами

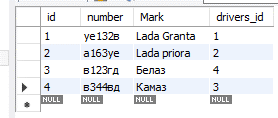


Таблица с пользователями

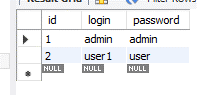


Таблица с водителями

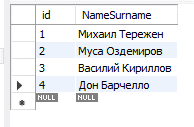


Таблица с товарами

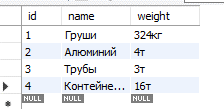
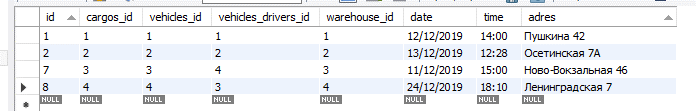


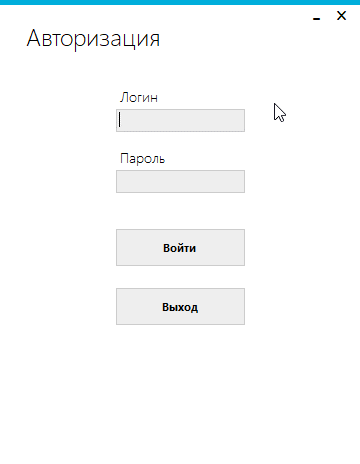
Таблица с маршрутами

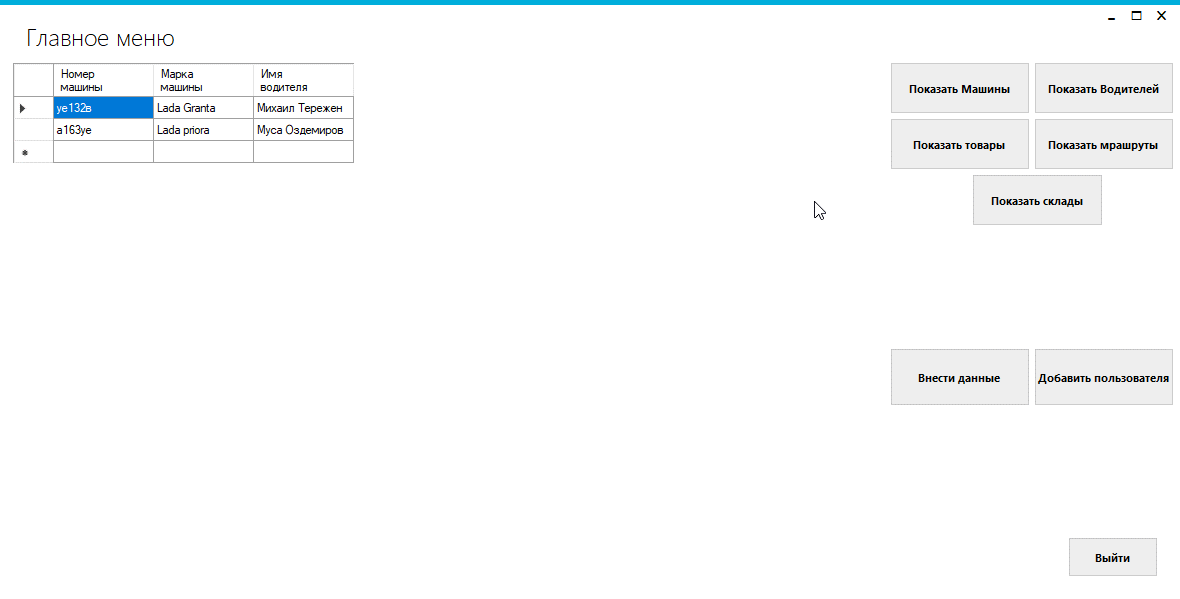


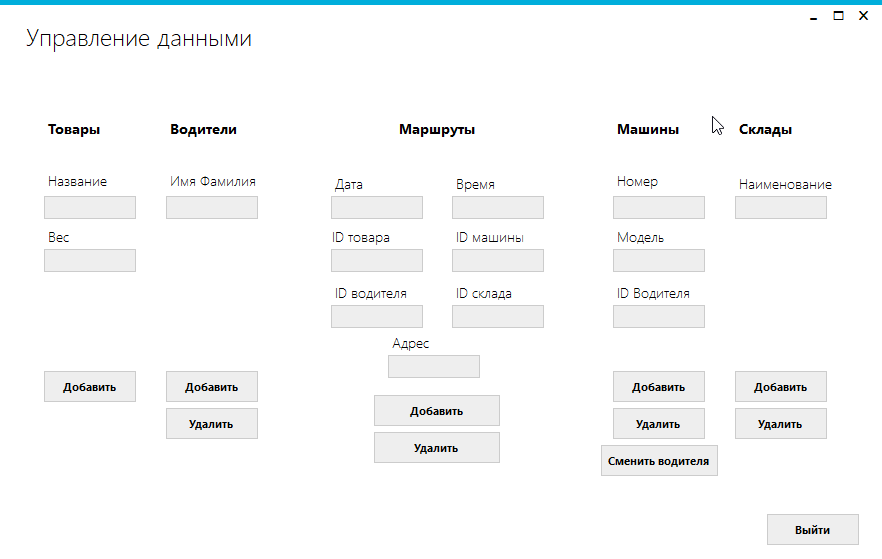
## 3.3Реализация функциональных требований

При создании программы вначале был создан класс для подключения к БД (см рис. 1). Далее было создано окно авторизации которое берет информацию о пользователях из соответствующей таблицы в БД (см рис. 2). После создано окно отображения данных с кнопками которые запускают функции вывода данных из различных таблиц (см рис. 3). Далее было реализован интерфейс для ввода данных через запрос в базу данных с данными введенными в текстовые поля (см рис. 4).









## 3.4 Интеграция программных модулей

Программа и база данных подключены друг к другу посредством класса подключения описанным в вышестоящем пункте, а так же по средством запросов из программы в БД (пример рис.1 и рис.2).





## 3.5 Тестирование и отладка программного продукта

В процессе тестирования была осуществлена проверка работы запросов к БД а так же корректной работы интерфейса программы. При проверки БД выдавала ошибку в связи некорректным заполнением связанных столбцов таблиц. В самой же программе возникали ошибки неработающих кнопок из-за непрописанных функций. Однако после тестирования все ошибки были исправлены.

# ЗАКЛЮЧЕНИЕ

В процессе работы над курсовым проектом все поставленные задачи и цель были достигнуты.

Выполнена цель курсового проекта:

Разработка удаленной базы данных для отдела логистики ООО «Самарская алюминиевая компания», а так-же программы для управления этой БД.

Выполнены поставленные задачи:

* Создать базу данных;
* произвести нормализацию структуры базы данных;
* произвести перенос данных из имеющейся базы в MYSQL в созданную удаленную базу данных;
* произвести правки в программном продукте для реализации работы с удаленной базой данных;
* произвести разработку интерфейса, работающего с удаленной базой данных;
* произвести тестирование и отладку работы модулей программного продукта.

Таким образом, гипотеза, на которой был основан курсовой проект, согласно которой автоматизируемый процесс на рассматриваемом предприятии будет максимально эффективен, подтвердилась.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

**Нормативные материалы**

1. ГОСТ Р 7.0.13-2011Система стандартов по информации, библиотечному и издательскому делу. Карточки для каталогов и картотек, макет аннотированной карточки в издании. Общие требования и издательское оформление. – М.: Изд-во стандартов, 2005. –12 с.
2. ГОСТ Р 51904-2002. Программное обеспечение встроенных систем. Общие требования к разработке и документированию. – М.: Изд-во стандартов, 2005. –71 с.

**Научные, технические и учебно-методические издания**

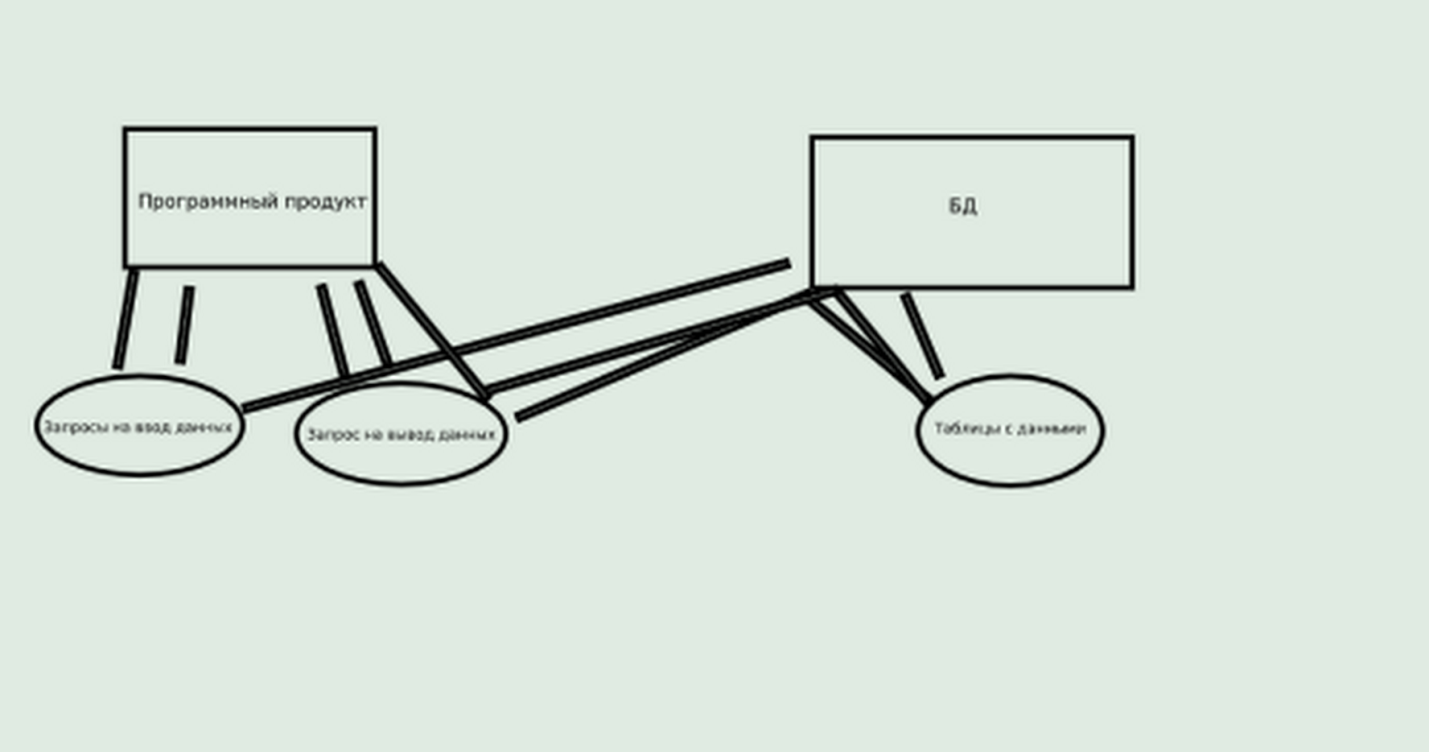
1. АртеменкоЮ.Н. MySQL. Справочник по языку. – М.: Издательский дом "Вильямс", 2015. – 432 с.
2. ЛиннБейли, Майкл Моррисон. Изучаем PHP и MySQL. – М.: Эксмо, 2017. – 786 с.

**Ресурсы сети Интернет**

1. <https://msdn.microsoft.com/ru-ru/>
2. <http://php.net/manual/ru/>
3. <https://metanit.com/sharp/tutorial/>
4. https://metanit.com/sql/mysql/

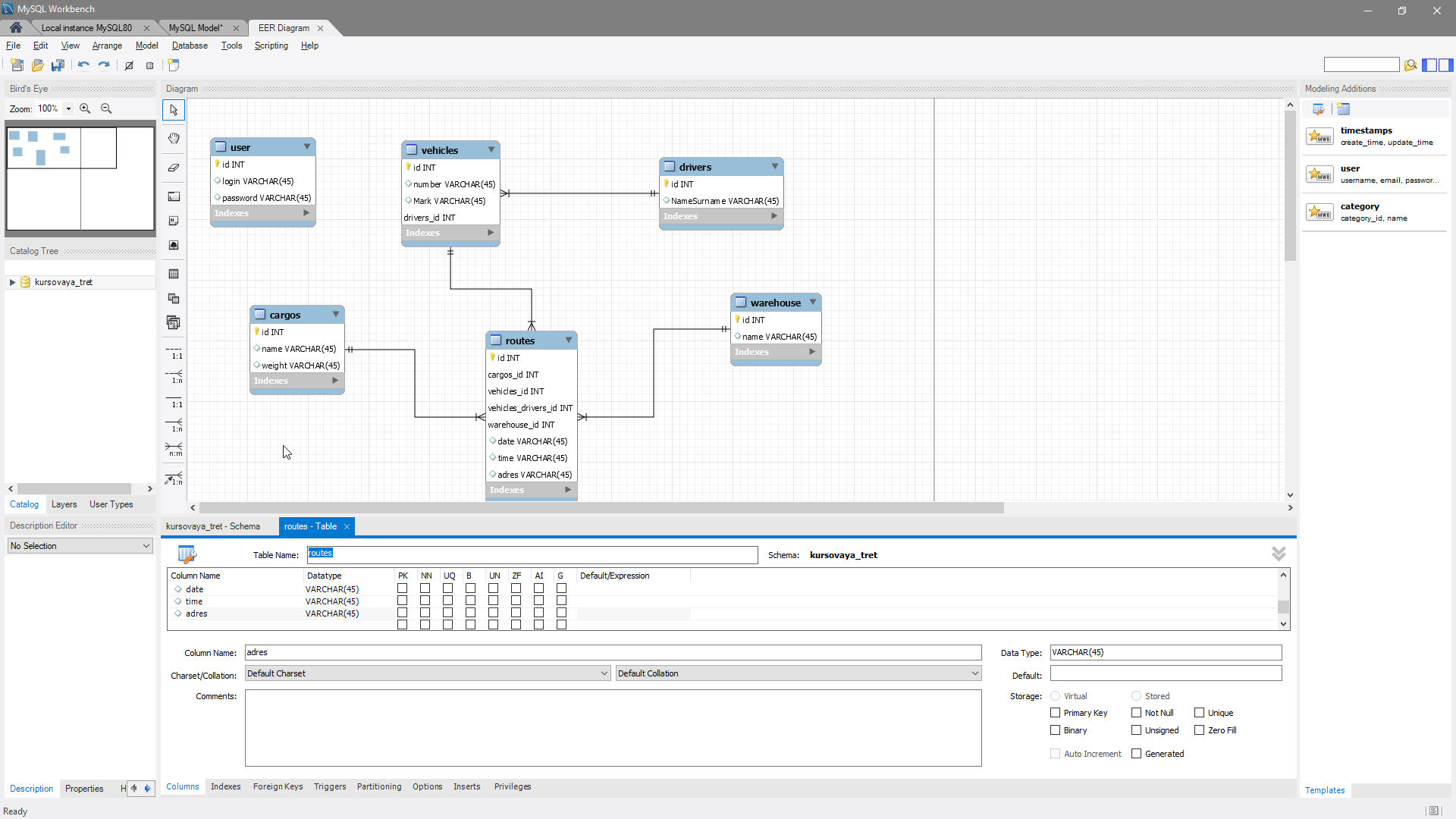
# ПРИЛОЖЕНИЕ А Структурная схема программного продукта

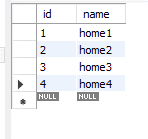
**Структурная схема программного продукта**

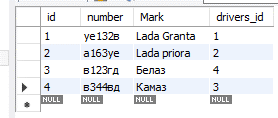


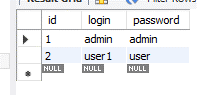
# ПРИЛОЖЕНИЕ Б Структурная схема базы данных

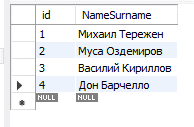
**Структурная схема базы данных**

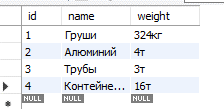


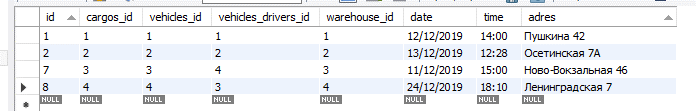












# ПРИЛОЖЕНИЕ В Листинг кода программного продукта

**Листинг кода программного продукта**

1)Подключение БД

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace tretjakov

{

class DB

{

MySqlConnection connection = new MySqlConnection("server=localhost;port=3306;username=root;password=Liquidchair0146;database=kursovaya\_tret;CharSet=utf8");

public void openConnection()

{

if (connection.State == System.Data.ConnectionState.Closed)

connection.Open();

}

public void closeConnection()

{

if (connection.State == System.Data.ConnectionState.Open)

connection.Close();

}

public MySqlConnection getConnection()

{

return connection;

}

}

}

2)Авторизация

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using MetroFramework.Components;

using MetroFramework.Forms;

using MetroFramework.Fonts;

using MySql.Data.MySqlClient;

namespace tretjakov

{

public partial class Autorize : MetroForm

{

public Autorize()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

}

private void metroButton1\_Click(object sender, EventArgs e)

{

String login = metroTextBox1.Text;

String password = metroTextBox2.Text;

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM `user` WHERE `login` = @uL AND `password` = @uP", db.getConnection());

command.Parameters.Add("@uL", MySqlDbType.VarChar).Value = login;

command.Parameters.Add("@uP", MySqlDbType.VarChar).Value = password;

adapter.SelectCommand = command;

adapter.Fill(table);

if(table.Rows.Count > 0)

{

Main main = new Main();

main.Show();

}

else

{

MessageBox.Show("Пользователь не найден", "Ошибка");

}

}

private void metroButton2\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void metroButton3\_Click(object sender, EventArgs e)

{

MessageBox.Show("server=localhost port=3306 connection - true");

}

}

}

3)Окно вывода данных

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using MetroFramework.Components;

using MetroFramework.Forms;

using MetroFramework.Fonts;

using MySql.Data.MySqlClient;

namespace tretjakov

{

public partial class Main : MetroForm

{

public Main()

{

InitializeComponent();

}

private void Main\_Load(object sender, EventArgs e)

{

}

private void SetBorderAndGridlineStyles()

{

this.dataGridView1.GridColor = Color.BlueViolet;

this.dataGridView1.CellBorderStyle =

DataGridViewCellBorderStyle.None;

this.dataGridView1.RowHeadersBorderStyle =

DataGridViewHeaderBorderStyle.Single;

this.dataGridView1.ColumnHeadersBorderStyle =

DataGridViewHeaderBorderStyle.Single;

}

private void metroButton1\_Click(object sender, EventArgs e)

{

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT vehicles.number AS 'Номер машины', vehicles.mark AS 'Марка машины', drivers.NameSurname AS 'Имя водителя' FROM `vehicles` JOIN `drivers` ON drivers.id = vehicles.drivers\_id", db.getConnection());

adapter.SelectCommand = command;

adapter.Fill(table);

dataGridView1.DataSource = table;

}

private void metroButton7\_Click(object sender, EventArgs e)

{

Autorize aut = new Autorize();

aut.Show();

this.Close();

}

private void metroButton3\_Click(object sender, EventArgs e)

{

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM `drivers`", db.getConnection());

adapter.SelectCommand = command;

adapter.Fill(table);

dataGridView1.DataSource = table;

}

private void metroButton2\_Click(object sender, EventArgs e)

{

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM `cargos`", db.getConnection());

adapter.SelectCommand = command;

adapter.Fill(table);

dataGridView1.DataSource = table;

}

private void metroButton4\_Click(object sender, EventArgs e)

{

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT routes.date AS 'Дата', routes.time AS 'Время', routes.adres AS 'Адрес доставки', drivers.NameSurname AS 'Имя Водителя', CONCAT(vehicles.mark, ' ', vehicles.number) AS 'Машина', cargos.name AS 'Товар', cargos.weight AS 'Вес товара', warehouse.name AS 'Склад' FROM `routes` JOIN `drivers` ON(drivers.id = vehicles\_drivers\_id)JOIN `vehicles` ON(vehicles.id = vehicles\_id)JOIN `cargos` ON(cargos.id = cargos\_id)JOIN `warehouse` ON(warehouse.id = warehouse\_id); ", db.getConnection()); ;

adapter.SelectCommand = command;

adapter.Fill(table);

dataGridView1.DataSource = table;

}

private void metroButton9\_Click(object sender, EventArgs e)

{

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM `warehouse`", db.getConnection());

adapter.SelectCommand = command;

adapter.Fill(table);

dataGridView1.DataSource = table;

}

private void metroButton8\_Click(object sender, EventArgs e)

{

adduser au = new adduser();

au.Show();

}

private void metroButton6\_Click(object sender, EventArgs e)

{

datachange dc = new datachange();

dc.Show();

}

}

}

4)Окно ввода данных

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using MetroFramework.Components;

using MetroFramework.Forms;

using MetroFramework.Fonts;

using MySql.Data.MySqlClient;

namespace tretjakov

{

public partial class datachange : MetroForm

{

public datachange()

{

InitializeComponent();

}

private void datachange\_Load(object sender, EventArgs e)

{

}

private void metroButton1\_Click(object sender, EventArgs e)

{

String name = metroTextBox1.Text;

String weight = metroTextBox2.Text;

DB db = new DB();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("insert into `cargos` (name, weight) values (@uL, @uP)", db.getConnection());

command.Parameters.Add("@uL", MySqlDbType.VarChar).Value = name;

command.Parameters.Add("@uP", MySqlDbType.VarChar).Value = weight;

db.openConnection();

if (command.ExecuteNonQuery() == 1)

MessageBox.Show("Товар добавлен");

else

MessageBox.Show("Ошибка");

db.closeConnection();

}

private void metroButton6\_Click(object sender, EventArgs e)

{

String name = metroTextBox4.Text;

DB db = new DB();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("insert into `drivers` (NameSurname) values (@uL)", db.getConnection());

command.Parameters.Add("@uL", MySqlDbType.VarChar).Value = name;

db.openConnection();

if (command.ExecuteNonQuery() == 1)

MessageBox.Show("Водитель добавлен");

else

MessageBox.Show("Ошибка");

db.closeConnection();

}

private void metroButton5\_Click(object sender, EventArgs e)

{

String name = metroTextBox4.Text;

DB db = new DB();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("DELETE FROM `drivers` WHERE NameSurname = @uL", db.getConnection());

command.Parameters.Add("@uL", MySqlDbType.VarChar).Value = name;

db.openConnection();

if (command.ExecuteNonQuery() == 1)

MessageBox.Show("Водитель Удален");

else

MessageBox.Show("Ошибка");

db.closeConnection();

}

private void metroButton12\_Click(object sender, EventArgs e)

{

String name = metroTextBox7.Text;

DB db = new DB();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("insert into `warehouse` (name) values (@uL)", db.getConnection());

command.Parameters.Add("@uL", MySqlDbType.VarChar).Value = name;

db.openConnection();

if (command.ExecuteNonQuery() == 1)

MessageBox.Show("Склад добавлен");

else

MessageBox.Show("Ошибка");

db.closeConnection();

}

private void metroButton11\_Click(object sender, EventArgs e)

{

String name = metroTextBox7.Text;

DB db = new DB();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("DELETE FROM `warehouse` WHERE name = @uL", db.getConnection());

command.Parameters.Add("@uL", MySqlDbType.VarChar).Value = name;

db.openConnection();

if (command.ExecuteNonQuery() == 1)

MessageBox.Show("Водитель Удален");

else

MessageBox.Show("Ошибка");

db.closeConnection();

}

private void metroButton9\_Click(object sender, EventArgs e)

{

String number = metroTextBox3.Text;

String model = metroTextBox5.Text;

String dID = metroTextBox6.Text;

DB db = new DB();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("insert into `vehicles` (number, Mark, drivers\_id) values (@uN, @uM, @dID)", db.getConnection());

command.Parameters.Add("@uN", MySqlDbType.VarChar).Value = number;

command.Parameters.Add("@uM", MySqlDbType.VarChar).Value = model;

command.Parameters.Add("@dID", MySqlDbType.VarChar).Value = dID;

db.openConnection();

if (command.ExecuteNonQuery() == 1)

MessageBox.Show("Машина добавлена");

else

MessageBox.Show("Ошибка");

db.closeConnection();

}

private void metroButton8\_Click(object sender, EventArgs e)

{

String number = metroTextBox3.Text;

String model = metroTextBox5.Text;

String dID = metroTextBox6.Text;

DB db = new DB();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("DELETE FROM `vehicles` WHERE number = @uN", db.getConnection());

command.Parameters.Add("@uN", MySqlDbType.VarChar).Value = number;

command.Parameters.Add("@uM", MySqlDbType.VarChar).Value = model;

command.Parameters.Add("@dID", MySqlDbType.VarChar).Value = dID;

db.openConnection();

if (command.ExecuteNonQuery() == 1)

MessageBox.Show("Машина удалена");

else

MessageBox.Show("Ошибка");

db.closeConnection();

}

private void metroButton2\_Click(object sender, EventArgs e)

{

String number = metroTextBox3.Text;

String model = metroTextBox5.Text;

String dID = metroTextBox6.Text;

DB db = new DB();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("UPDATE `vehicles` SET drivers\_id = @dID WHERE number = @uN", db.getConnection());

command.Parameters.Add("@uN", MySqlDbType.VarChar).Value = number;

command.Parameters.Add("@uM", MySqlDbType.VarChar).Value = model;

command.Parameters.Add("@dID", MySqlDbType.VarChar).Value = dID;

db.openConnection();

if (command.ExecuteNonQuery() == 1)

MessageBox.Show("Водитель изменен");

else

MessageBox.Show("Ошибка");

db.closeConnection();

}

private void metroButton15\_Click(object sender, EventArgs e)

{

String dat = metroTextBox10.Text;

String tim = metroTextBox13.Text;

String dID = metroTextBox8.Text;

String wID = metroTextBox11.Text;

String cID = metroTextBox9.Text;

String vID = metroTextBox12.Text;

String adr = metroTextBox14.Text;

DB db = new DB();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("insert into `routes` (cargos\_id, vehicles\_id, vehicles\_drivers\_id, warehouse\_id, date, time, adres) values (@cID, @vID, @dID, @wID, @dT, @tI, @aD)", db.getConnection());

command.Parameters.Add("@cID", MySqlDbType.VarChar).Value = cID;

command.Parameters.Add("@vID", MySqlDbType.VarChar).Value = vID;

command.Parameters.Add("@dID", MySqlDbType.VarChar).Value = dID;

command.Parameters.Add("@wID", MySqlDbType.VarChar).Value = wID;

command.Parameters.Add("@dT", MySqlDbType.VarChar).Value = dat;

command.Parameters.Add("@tI", MySqlDbType.VarChar).Value = tim;

command.Parameters.Add("@aD", MySqlDbType.VarChar).Value = adr;

db.openConnection();

if (command.ExecuteNonQuery() == 1)

MessageBox.Show("Маршрут добавлен");

else

MessageBox.Show("Ошибка");

db.closeConnection();

}

private void metroButton14\_Click(object sender, EventArgs e)

{

String dat = metroTextBox10.Text;

String tim = metroTextBox13.Text;

String dID = metroTextBox8.Text;

String wID = metroTextBox11.Text;

String cID = metroTextBox9.Text;

String vID = metroTextBox12.Text;

String adr = metroTextBox14.Text;

DB db = new DB();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("DELETE FROM `routes` WHERE date = @dT AND time = @tI AND vehicles\_id = @vID", db.getConnection());

command.Parameters.Add("@cID", MySqlDbType.VarChar).Value = cID;

command.Parameters.Add("@vID", MySqlDbType.VarChar).Value = vID;

command.Parameters.Add("@dID", MySqlDbType.VarChar).Value = dID;

command.Parameters.Add("@wID", MySqlDbType.VarChar).Value = wID;

command.Parameters.Add("@dT", MySqlDbType.VarChar).Value = dat;

command.Parameters.Add("@tI", MySqlDbType.VarChar).Value = tim;

command.Parameters.Add("@aD", MySqlDbType.VarChar).Value = adr;

db.openConnection();

if (command.ExecuteNonQuery() == 1)

MessageBox.Show("Маршрут удален");

else

MessageBox.Show("Ошибка");

db.closeConnection();

}

private void metroButton3\_Click(object sender, EventArgs e)

{

this.Close();

}

}

}

# ПРИЛОЖЕНИЕ Г Листинг кода запросов в базе данных

**Листинг кода запросов в базе данных**

MySqlCommand command = new MySqlCommand("DELETE FROM `routes` WHERE date = @dT AND time = @tI AND vehicles\_id = @vID", db.getConnection());

command.Parameters.Add("@cID", MySqlDbType.VarChar).Value = cID;

command.Parameters.Add("@vID", MySqlDbType.VarChar).Value = vID;

command.Parameters.Add("@dID", MySqlDbType.VarChar).Value = dID;

command.Parameters.Add("@wID", MySqlDbType.VarChar).Value = wID;

command.Parameters.Add("@dT", MySqlDbType.VarChar).Value = dat;

command.Parameters.Add("@tI", MySqlDbType.VarChar).Value = tim;

command.Parameters.Add("@aD", MySqlDbType.VarChar).Value = adr;

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("insert into `routes` (cargos\_id, vehicles\_id, vehicles\_drivers\_id, warehouse\_id, date, time, adres) values (@cID, @vID, @dID, @wID, @dT, @tI, @aD)", db.getConnection());

command.Parameters.Add("@cID", MySqlDbType.VarChar).Value = cID;

command.Parameters.Add("@vID", MySqlDbType.VarChar).Value = vID;

command.Parameters.Add("@dID", MySqlDbType.VarChar).Value = dID;

command.Parameters.Add("@wID", MySqlDbType.VarChar).Value = wID;

command.Parameters.Add("@dT", MySqlDbType.VarChar).Value = dat;

command.Parameters.Add("@tI", MySqlDbType.VarChar).Value = tim;

command.Parameters.Add("@aD", MySqlDbType.VarChar).Value = adr;

MySqlCommand command = new MySqlCommand("UPDATE `vehicles` SET drivers\_id = @dID WHERE number = @uN", db.getConnection());

command.Parameters.Add("@uN", MySqlDbType.VarChar).Value = number;

command.Parameters.Add("@uM", MySqlDbType.VarChar).Value = model;

command.Parameters.Add("@dID", MySqlDbType.VarChar).Value = dID;

MySqlCommand command = new MySqlCommand("SELECT \* FROM `user` WHERE `login` = @uL AND `password` = @uP", db.getConnection());

command.Parameters.Add("@uL", MySqlDbType.VarChar).Value = login;

command.Parameters.Add("@uP", MySqlDbType.VarChar).Value = password;