Оглавление

[ВВЕДЕНИЕ 3](#_Toc473094324)

[1. Постановка на курсовую работу. 4](#_Toc473094325)

[2. Исследование предметной области 4](#_Toc473094326)

[3. Проектирование базы данных 4](#_Toc473094327)

[3.1. Концептуальная модель базы данных 4](#_Toc473094328)

[3.2. Логическая модель базы данных 6](#_Toc473094329)

[Первая нормальная форма (1НФ) 7](#_Toc473094330)

[Вторая нормальная форма (2НФ) 7](#_Toc473094331)

[Третья нормальная форма (3НФ) 8](#_Toc473094332)

[Четвёртая нормальна форма (4НФ) 8](#_Toc473094333)

[3.3. Выбор системы управления базы данных (СУБД) 10](#_Toc473094334)

[3.3.1. MySQL 10](#_Toc473094335)

[3.3.2. PostgreSQL 11](#_Toc473094336)

[3.4. Физическая модель базы данных 12](#_Toc473094337)

[4. Разработка программного обеспечения 12](#_Toc473094338)

[ЗАКЛЮЧЕНИЕ 14](#_Toc473094339)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ: 15](#_Toc473094340)

[ПРИЛОЖЕНИЕ А. Руководство пользователя 16](#_Toc473094341)

[ПРИЛОЖЕНИЕ Б. Листинги приложения 19](#_Toc473094342)

# 1. Постановка задачи на курсовую работу.

*Склад*. Имеется несколько складов. Для каждого склада известен владелец и название. На каждом складе хранятся товары. Одинаковые товары могут храниться на разных складах. Некоторые склады могут временно пустовать. Известна вместимость каждого склада в тоннах. Складов без владельцев не бывает.

Товар хранится на складе определенный период времени, по истечении которого списывается и увозится на полигон отходов. Во время хранения товар отпускается по заявкам магазинов.

О каждом товаре известно его наименование, уникальный номер-артикул.

Товары на склады привозятся на автомашинах. О каждой автомашине известна ее марка, грузоподъемность в тоннах и фамилия владельца. Машин без владельцев не бывает. Имеется информация о поступлениях, показывающая, какая машина, какой товар, на какой склад привозит, в каком количестве (в тоннах).

Товары в магазины отвозятся теми же автомашинами.

# 2. Исследование предметной области

Предметной областью называется фрагмент реальности, который описывается или моделируется с помощью БД и ее приложений. В предметной области выделяются информационные объекты – идентифицируемые объекты реального мира, процессы, системы, понятия и т.д., сведения о которых хранятся в БД. В данной курсовой работе разрабатывается база данных "Склады" для хранения и перевозки товаров.

В соответствии с предметной областью система строится с учётом следующих особенностей:

Каждая партия товара привозится автомашинами и хранится на складе, Во время хранения товар отпускается по магазинам теми же автомашинами и по истечению времени со склада увозится на полигон.

# 3. Проектирование базы данных

## 3.1. Концептуальная модель базы данных

Для создания концептуальной модели необходимо выделить сущности предметной области и атрибуты:

**Склад:** Идентификатор, Название, Вместимость, владелец, срок хранения.

**Машина:** Идентификатор, Марка, Грузоподъемность, Владелец.

**Товар:** Идентификатор, Название, Артикул, Количество товара.

**Магазин:** Идентификатор, Товар, Название, Машина.

**Полигон:** Идентификатор, Название склада.

**Завоз:** Идентификатор, Ид. Товара, Ид. Склада, Ид. Машины.

Определим связи между сущностями:

Склад хранит товары –один ко многим.

Машина перевозит на склады –многие ко многим.

Товар проходит учет многие ко многим

В магазин машина завозит в магазин товары – один ко многим

Концептуальная модель базы данных представлена на рис. 3.1.

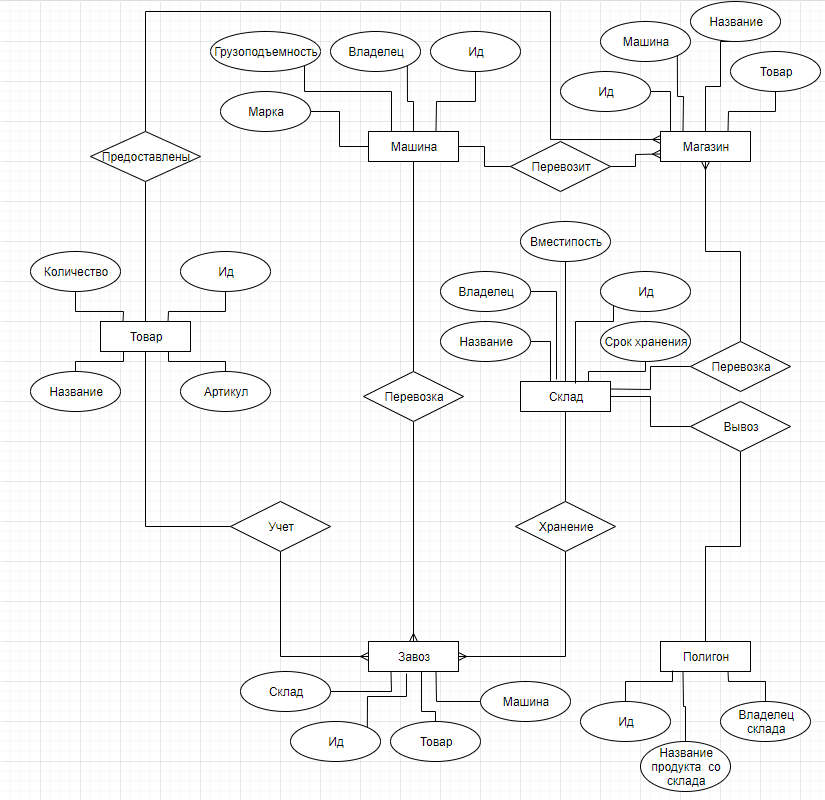


Рисунок 3.1 Концептуальная модель

## 3.2. Логическая модель базы данных

На основании концептуальной модели базы данных строится логическая модель, путём сопоставления каждой сущности и каждой связи, имеющей атрибуты, отношения (таблицы) базы данных. Связь типа 1-N (один ко многим) между отношениями реализуется через внешний ключ. Ключ вводится для того отношения, к которому осуществляется множественная связь. Внешнему ключу должен соответствовать первичный или уникальный ключ основного (родительского) отношения.

Для каждого отношения определяются первичный ключ и внешние ключи. В том случае, если базовое отношения не имеет потенциальных ключей, вводится суррогатный первичный ключ, который не несёт смысловой нагрузки и служит только для идентификации записи.

Отношения базы данных «Склады» приведены в таблицах 1-4

Потенциальными ключами отношения Товары являются атрибуты «Название». Название товара не может совпасть, так как эта сущность является каталожным объектом и для удобства связывания этой сущности с другими введём суррогатный ключ -идентификатор («ИД»)

Таблица 1 Схема отношения «Склады»

|  |  |  |
| --- | --- | --- |
| Ключ | Имя поля | Примечание |
| ИД | Id | Номер склада, первичный ключ |
| Название | Title | Содержит название продукта |
| Владелец | Owner | Содержит Имя владельца |
| Вместимость | Capacity | Содержит вместимость склада |
| Срок хранения | Shelf\_life | Содержит дату хранения |

Таблица 2 Схема отношения «Машина»

|  |  |  |
| --- | --- | --- |
| Ключ | Имя поля | Примечание |
| ИД | Id | Номер машины, первичный ключ |
| Марка | mark | Содержит марку машины |
| Владелец | owner | Содержит имя владельца |
| Грузоподъемность | Mweigth | Максимальный вес |

Таблица 3 Схема отношения «Товар»

|  |  |  |
| --- | --- | --- |
| Ключ | Имя поля | Примечание |
| ИД | Id | Номер товара, первичный ключ |
| Название | title | Содержит название продукта |
| Количиство | number | Содержит количество продукта |
| Артикул | vencode | Содержит уникальный номер продукта |

Таблица 4 Схема отношения «Магазин»

|  |  |  |
| --- | --- | --- |
| Ключ | Имя поля | Примечание |
| ИД | Id | Номер магазина, первичный ключ |
| Название склада | Id\_sclad | Содержит номер склада |
| Машина | Id\_car | Содержит номер машины |
| Товар | Id\_product | Содержит номер товара |

Таблица 5 Схема отношения «Полигон»

|  |  |  |
| --- | --- | --- |
| Ключ | Имя поля | Примечание |
| ИД | Id | Номер полигона, первичный ключ |
| Название склада | Id\_sclad | Содержит номер склада |

Таблица 6 Схема отношения «Завоз»

|  |  |  |
| --- | --- | --- |
| Ключ | Имя поля | Примечание |
| ИД | Id | Номер завоза, первичный ключ |
| Название склада | Id\_sclad | Содержит номер склада |
| Машина | Id\_car | Содержит номер машины |
| Товар | Id\_product | Содержит номер товара |

### Первая нормальная форма (1НФ)

Отношение приведено к первой нормальной форме, если все его атрибуты простые. В нашем случае данные атомарные.

### Вторая нормальная форма (2НФ)

Отношение находится во 2НФ, если оно приведено к 1НФ, и каждый неключевой атрибут функционально полностью зависит от составного первичного ключа. Таки образом, если отношение имеет простой первичный ключ, оно сразу находится во второй нормальной форме.

Очевидно, что в нынешнем состоянии наши таблицы уже находятся во второй нормальной форме.

### Третья нормальная форма (3НФ)

Отношение находится в 3НФ, если оно находится во 2НФ и каждый неключевой атрибут не транзитивно зависит от первичного ключа.

В наших отношениях нет атрибутов, зависящих не от первичного ключа, поэтому наши отношения находятся в 3НФ.

### Четвёртая нормальна форма (4НФ)

Отношение находится в 4НФ, если оно находится в 3НФ и в нём отсутствуют нетривиальные многозначные зависимости.

В наших отношениях нет вычисляемых полей и нет многозначных зависимостей в одном отношении, поэтому наши отношения находятся в четвёртой нормальной форме.

На основе выше написанного создадим логическую модель базы данных, приведеную на рис. 2.

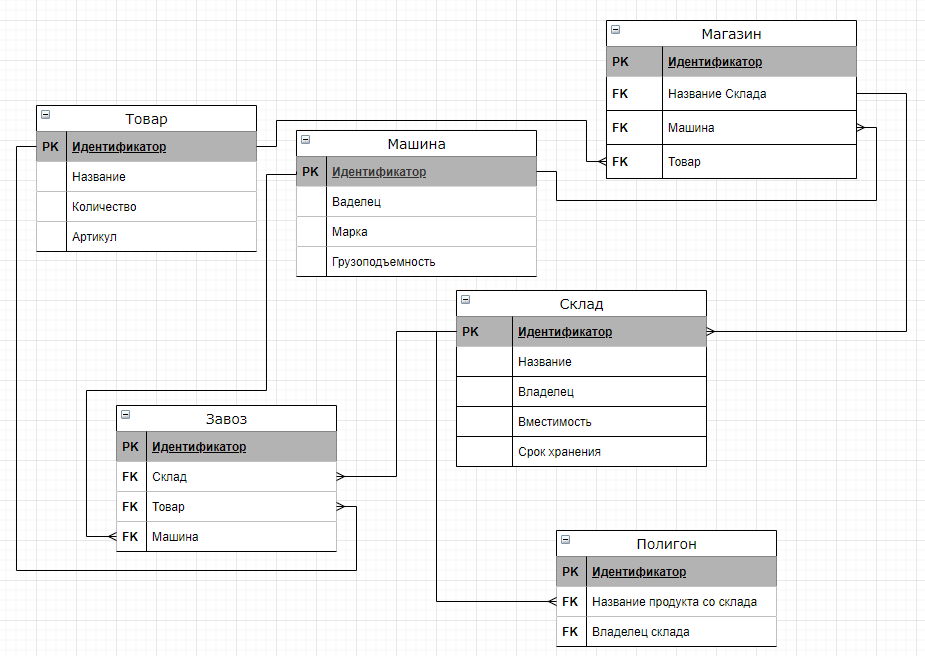


Рисунок 2 Логическая модель базы данных

## 3.3. Выбор системы управления базы данных (СУБД)

Базы данных - это специально разработанное хранилище для различных типов данных. Каждая база данных, имеет определённую модель (реляционная, документно-ориентированная), которая обеспечивает удобный доступ к данным. Системы управления базами данных (СУБД) - специальные приложения (или библиотеки) для управления базами данных различных размеров и форм.

### 3.3.1. MySQL

Клиентская программа MySQL представляет собой утилиту командной строки. Эта программа подключается к серверу по сети. Команды, выполняемые сервером, обычно связаны с чтением и записью данных на жестком диске.

Клиентские программы могут работать не только в режиме командной строки. Есть и графические клиенты, например MySQL GUI, PhpMyAdmin и др.

Программное обеспечение MySQL - это ПО с открытым кодом. ПО с открытым кодом означает, что применять и модифицировать его может любой желающий. Такое ПО можно получать поInternet и использовать бесплатно. При этом каждый пользователь может изучить исходный код и изменить его в соответствии со своими потребностями.

**Преимущества MySQL:**

* Простота в работе - установить MySQL довольно просто. Дополнительные приложения, например, GUI, позволяет довольно легко работать с БД
* Богатый функционал - MySQL поддерживает большинство функционала SQL.
* Безопасность - большое количество функций, обеспечивающих безопасность, которые поддерживается по умолчанию
* Масштабируемость - MySQL легко работает с большими объемами данных и легко масштабируется
* Скорость - упрощение некоторых стандартов позволяет MySQL значительно увеличить производительность.

**Недостатки MySQL:**

* Известные ограничения - по задумке в MySQL заложены некоторые ограничения функционала, которые иногда необходимы в особо требовательных приложениях.
* Проблемы с надежностью - из-за некоторых способов обработки данных MySQL (связи, транзакции, аудиты) иногда уступает другим СУБД по надежности.
* Медленная разработка - Хотя MySQL технически открытое ПО, существуют жалобы на процесс разработки.

### 3.3.2. PostgreSQL

За более чем 20-летнюю историю своего развития PostgreSQL из академической разработки превратилась в полноценную СУБД корпоративного уровня, составляющую реальную альтернативу коммерческим базам. Лицензия PostgreSQL разрешает ее неограниченное использование, модификацию кода, а также включение в состав других продуктов, в том числе закрытых и коммерческих.

**Достоинства PostgreSQL:**

* Открытое ПО соответствующее стандарту SQL - PostgreSQL - бесплатное ПО с открытым исходным кодом. Эта СУБД является очень мощной системой.
* Большое сообщество - существует довольно большое сообщество, в котором вы запросто найдёте ответы на свои вопросы
* Большое количество дополнений - несмотря на огромное количество встроенных функций, существует очень много дополнений, позволяющих разрабатывать данные для этой СУБД и управлять ими.
* Расширения - существует возможность расширения функционала за счет сохранения своих процедур.
* Объектность - PostgreSQL это не только реляционная СУБД, но также и объектно-ориентированная с поддержкой наследования и много другого система.

**Недостатки PostgreSQL:**

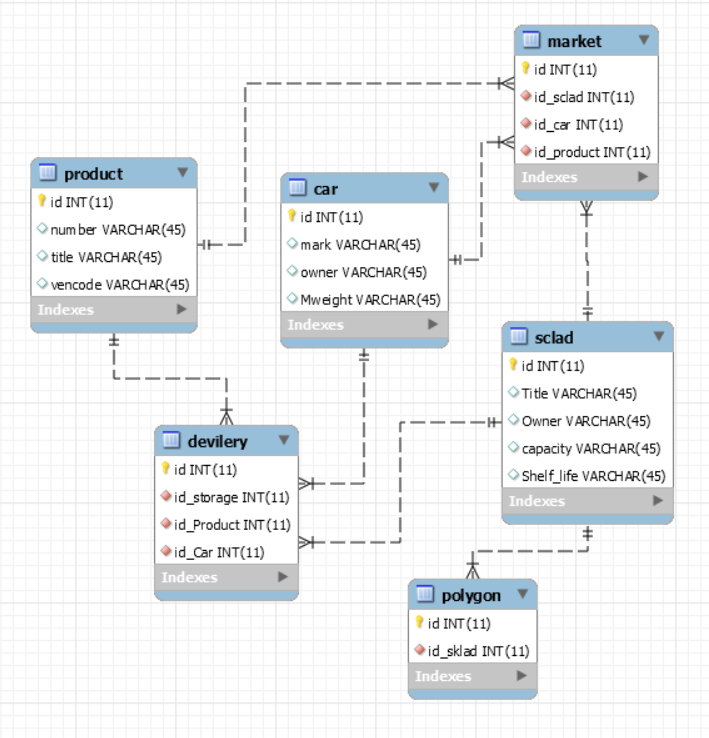
* Производительность - при простых операциях чтения PostgreSQL может значительно замедлить сервер и быть медленнее своих конкурентов, таких как MySQL
* Популярность - по своей природе, популярностью эта СУБД похвастаться не может, хотя и присутствует довольно большое сообщество.
* Хостинг - в силу выше перечисленных факторов иногда довольно сложно найти хостинг с поддержкой этой СУБД.

Нами были рассмотрены самые используемые СУБД со своими плюсами и минусами. В список рассматриваемых СУБД были внесены эти базы также по причине их бесплатного использования. Два эти продукта противопоставлены друг другу по критерию скорость – безопасность.

Учитывая наши реальные данные, приходим к выводу, что потеря скорости работы из-за более высокой надежности является лишней в нашем продукте. Поэтому разрабатываться база данных будет в MYSQL.

## 3.4. Физическая модель базы данных

Физическая модель – это логическая модель базы данных, выраженная в терминах языка описания данных конкретной СУБД, в нашем случае MySQL. (см. рис. 3)**Рисунок 3 Физическая модель базы данных**



# ****4. Разработка программного обеспечения****

Для работы с разработанной моделью базы данных требуется программное обеспечение (клиентское приложение), упрощающее взаимодействие с базой данных обычных пользователей и представляющее данные в удобном для восприятия виде, а также предоставляющие инструменты для удобной манипуляции данными.

Для реализации клиентского приложения выбран Java язык за ее многоплатформенность без переделывания самого кода под каждую операционную систему. В процессе создания приложения активно использовалась программная оболочка компании Oracle – Workbench. Она поставляется также свободно как и сама база MySQL и инициализируется из одного дистрибутива.

По условию задания на данную работу стояла задача максимально быстро пройти таможенный досмотр. Для удобства пользования и информативности в приложение были введены дополнительные функциональные особенности, связанные с добавлением данных в базы, их обновлением.

Для упрощения выполняемой задачи удаление реализовано только из сводной таблицы таможенного досмотра. Приложение возможно в дальнейшем расширять, так как программа строилась модульным способом.

# ЗАКЛЮЧЕНИЕ

В курсовом проекте была разработана база данных для расчета эффективного расположения партий груза по таможенным пунктам, с учетом их пропускной способности

В работе продемонстрированы этапы разработки базы данных, переход от событий и свойств реального мира к абстрактной модели информационной системы. На основе концептуальной модели построена логическая модель – графическое представление базы данных, показывающее, что хранится в базе данных, но не отвечающее на вопрос как. Также отражён этап выбора СУБД и построения физической модели базы данных с учётом особенностей СУБД.

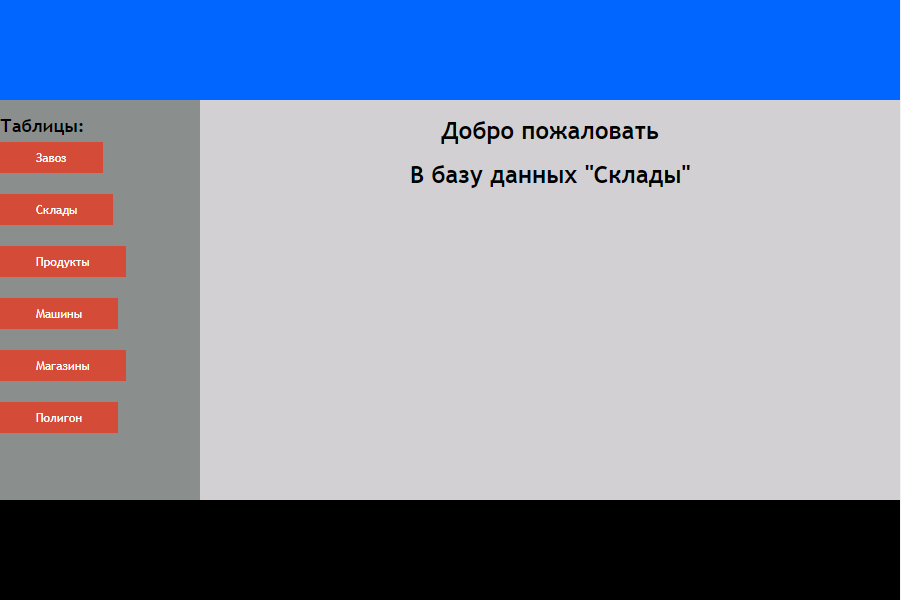
Для комфортной работы приложение дополнено функциональностью, не требуемым заданием. Разработанное приложение нельзя назвать идеальным программным продуктом, выполняющим свою функцию по причине, что реализация общения с пользователем минимальна.

# СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ:

1. Виктор Гольцман, MySQL 5.0, - СПб: Питер, 2009
2. Дмитрий Котеров, Алексей Костарев,PHP 5, 2-е издание, - СПб: БХВ-Петербург, 2008
3. Карпов И.П., Методические указания к курсовому проектированию по курсу «Базы данных», - М., 2010
4. http://www.cyberforum.ru/
5. http://javadevblog.com
6. http://info.javarush.ru/
7. https://www.mysql.com
8. http://www.mysql.ru

# ПРИЛОЖЕНИЕ А. Руководство пользователя

**Начальное окно приветствия**



Окно приложения разделено на две части: слева окно, выводящее таблицы с помощью соответствующих кнопок. Справа панель взаимодействия с программой, где будут выводится таблицы.

**Форма каждой таблицы**

**Таблица завоз:**

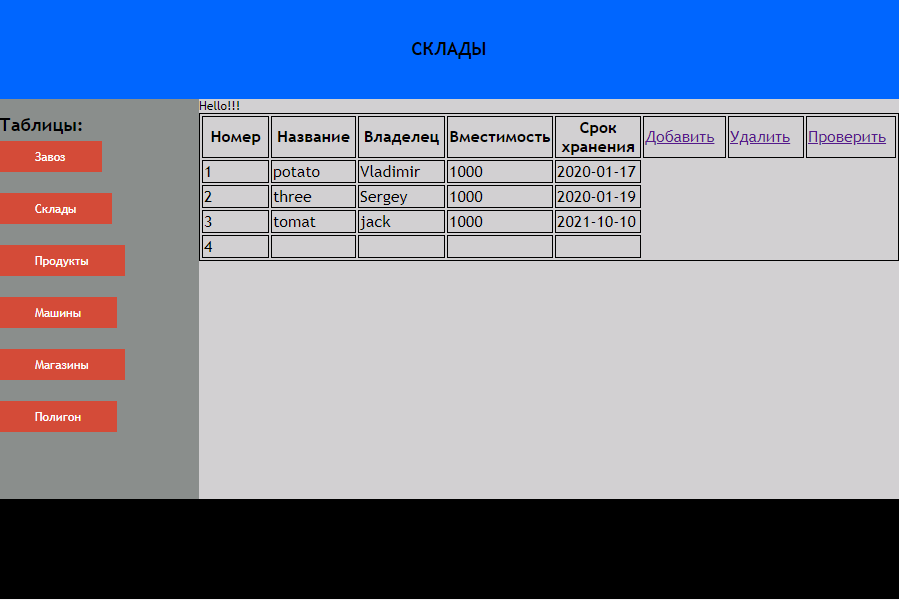


Окно в приложении необходима для просмотра параметров учета товаров, на каком складе хранится и какой водитель привез этот товар. Значение кнопок (добавить, удалить) понятно из названий самих кнопок.

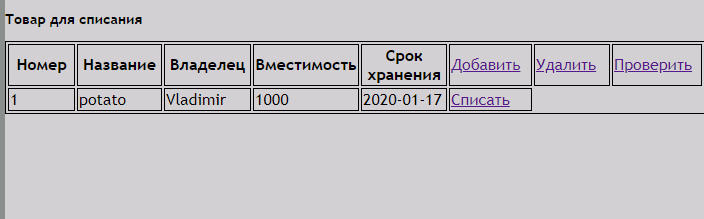
Кнопка “добавить” - добавляет новую запись для учета завоза.

Кнопка “удалить” – удаляет запись

**Таблица склады:**

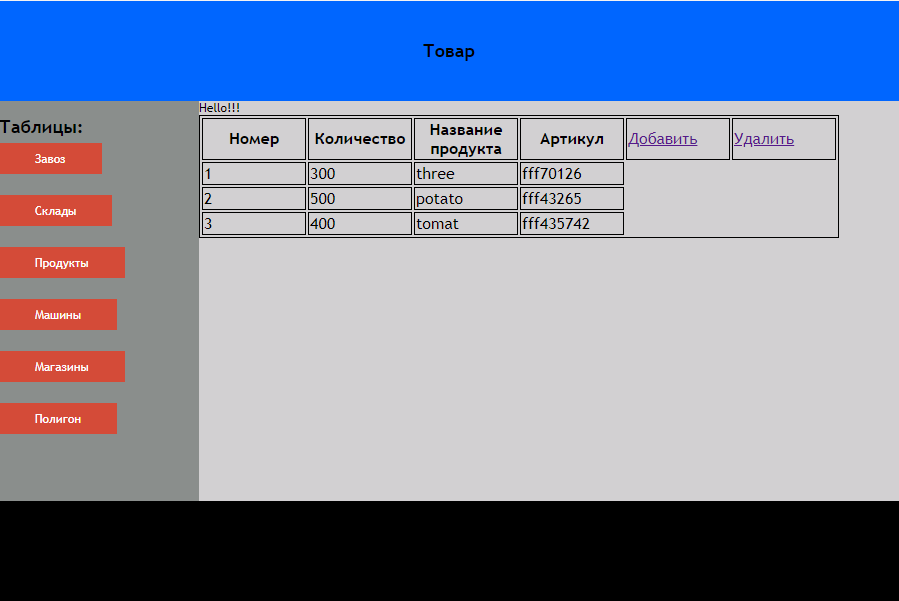


Окно в приложении необходима для просмотра на каком складе хранится название товара, кто владелец склада, его вместимость и срок хранения по кроме кнопок “Добавить” и ”Удалить” имеется кнопка “Проверить” она предназначена для того что бы имелась возможность проверить срок хранения товара. При ее нажатии выводится новая таблица с истекшим сроком хранения товара, если оно таковое имеется.



При нажатии кнопки ”Списать” срок хранения сравнивается с локальным временем на сервере, после чего название продукта и имя владельца переносится в таблицу “Полигон”.

**Таблица Товары:**



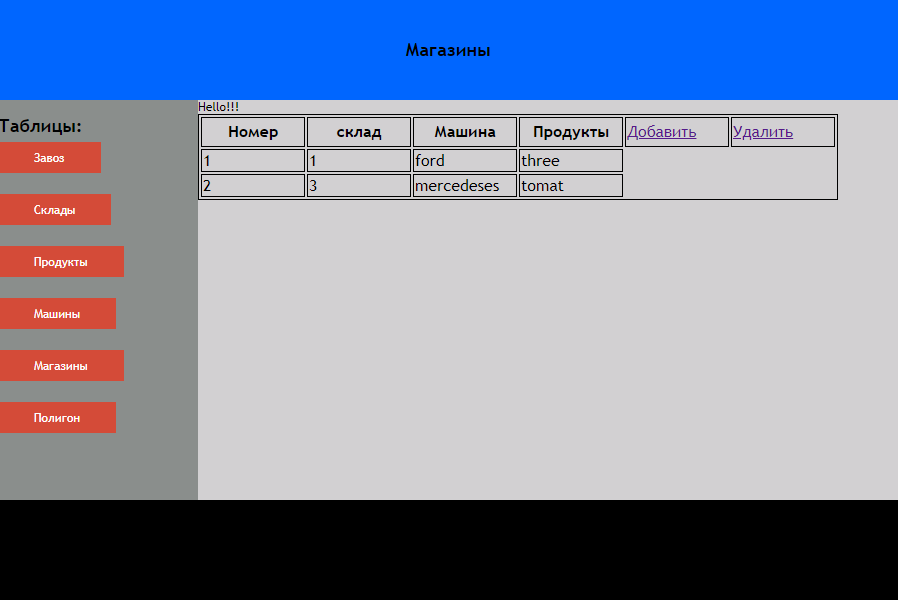
В этом окне выведены все товары для учета товаров, их количество в кг, название продукта, их уникальный номер – артикул.

**Таблица Машины:**



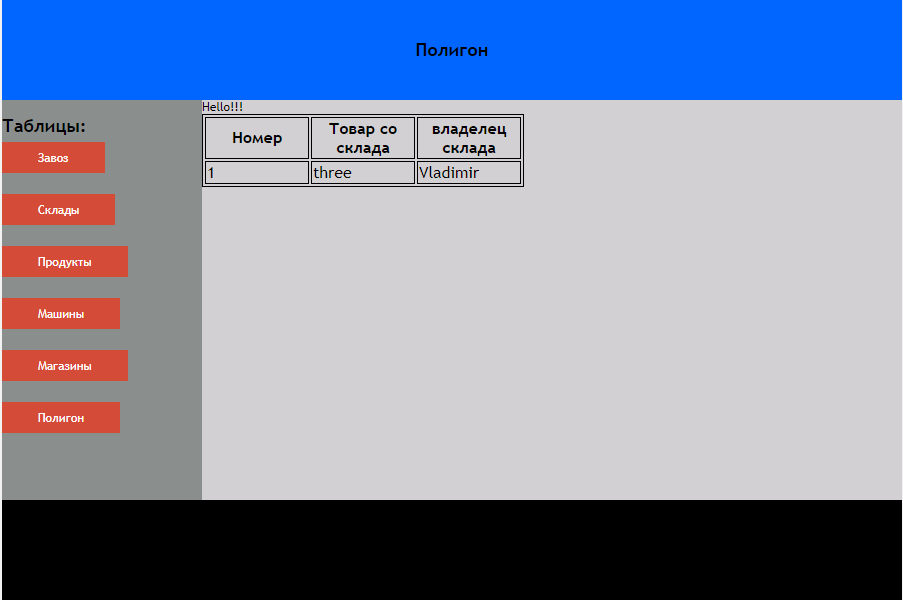
В этом окне выведены все Машины для перевозки грузов, марка, владелец и грукзоподъемность мешины в тоннах.

**Таблица магазины:**



В этом окне приведены номера магазинов, из какого склада и какая машина привезла продукты.

**Таблица магазины:**



В этом окне приведены товар со склада и владелец склада, это окно отличается тем что для добавления в эту таблицу есть кнопка в таблице “Склады”, по истечению времени со склада увозится на полигон.

# ПРИЛОЖЕНИЕ Б. Листинги приложения

Здесь представлен код функций, выполняемые приложением

**// код начального окна**

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<title>Пример блочной верстки</title>

<link rel="stylesheet" type="text/css" href="style.css" />

</head>

<body>

<div id="container">

<div id="header">

<h2> </h2>

</div>

<div id="menu">

<h2>Таблицы:</h2>

<p><a href="deliv.php" id ="knopka">Завоз</a></p>

<br>

<p><a href="sklad.php" id ="knopka">Склады</a></p>

<br>

<p><a href="product.php".html" id ="knopka">Продукты</a></p>

<br>

<p><a href="car.php".html" id ="knopka">Машины</a></p>

<br>

<p><a href="market.php".html" id ="knopka">Магазины</a></p>

<br>

<p><a href="poligon.php".html" id ="knopka">Полигон</a></p>

<br>

</div>

<div id="content">

<h1 align="center">Добро пожаловать</h1>

<h1 align="center">В базу данных "Склады"</h1>

</div>

<div id="clear">

</div>

</div>

</body>

</html>

**// Подключение к базе**

$con\_str=mysql\_connect('localhost', 'root', '', 'bd\_sklad');

if(mysql\_connect('localhost','root')){

echo "Hello!!!";

}

mysql\_select\_db('bd\_sklad',$con\_str);

**// создание таблицы и вывод информации с базы данных**

**// подключение для склада**

$result=mysql\_query("SELECT \* FROM `sclad` WHERE 1");

**// подключение для машины**

$result=mysql\_query("SELECT \* FROM `car` ");

**// подключение для магазина и совмещение таблиц для вывода**

$result=mysql\_query("SELECT market.id, market.id\_sclad, car.mark, product.title FROM `market`\n"

. "inner join car on market.id\_car = car.id\n"

. "inner join product on market.id\_product = product.id");

**// подключение для полигона**

$result=mysql\_query("SELECT \* FROM `polygon`");

**// подключение для продукта**

$result=mysql\_query("SELECT \* FROM `product` ");

**// подключение для завоза и совмещение таблиц для вывода**

$result=mysql\_query("SELECT devilery.id, devilery.id\_storage, product.title, car.owner FROM `devilery`\n"

. "inner join car on devilery.id\_Car = car.id\n"

. "inner join product on devilery.id\_Product = product.id LIMIT 0, 30 ");

**//Создание таблицы и внесение данных**

echo "<table border=1px>";

echo"<tr><th>Номер</th>

<th>Название</th>

<th>Владелец</th>

<th>Вместимость</th>

<th>Срок хранения</th><td><a href=Add.php>Добавить</a></td><td><a href=del.php>Удалить</a></td><td><a href=Read.php>Проверить</a></td></tr>";

while($row = mysql\_fetch\_array($result)){

$ids=$row['id'];

$storage=$row['Title'];

$product=$row['Owner'];

$car=$row['capacity'];

$life=$row['Shelf\_life'];

echo

"<tr><td>",$ids,

"</td><td>",$storage,

"</td><td>",$product,

"</td><td>",$car,

"</td><td>",$life,

"</td></tr>";

}

echo "</table>";

**// форма для добавления**

<form method="POST" action="add2.php">

<input name="id" type="text" placeholder="id"/>

<input name="sklad" type="text" placeholder="id\_storage"/>

<input name="product" type="text" placeholder="id\_product"/>

<input name="car" type="text" placeholder="id\_car"/>

<input type="submit" value="Отправить"/>

</form> }

<?php

if (isset($\_POST['id']) && isset($\_POST['sklad']) && isset($\_POST['product']) && isset($\_POST['car'])){

// Переменные с формы

$id = $\_POST['id'];

$sklad = $\_POST['sklad'];

$product = $\_POST['product'];

$car = $\_POST['car'];

$con\_str=mysql\_connect('localhost', 'root', '', 'bd\_sklad');

if(mysql\_connect('localhost','root')){

echo "Hello!!!";

}

mysql\_select\_db('bd\_sklad',$con\_str);

$result=mysql\_query("INSERT INTO `devilery` (`id`, `id\_storage`, `id\_Product`, `id\_Car`) VALUES ('$id', '$sklad', '$product', '$car');");

if ($result == true){

echo "Информация занесена в базу данных";

}else{

echo "Информация не занесена в базу данных";

}

**// форма для удаления**

<form method="POST" action="del2.php">

<input name="id" type="text" placeholder="id"/>

<input type="submit" value="Отправить"/>

</form>

if (isset($\_POST['id'])){

// Переменные с формы

$id = $\_POST['id'];

$con\_str=mysql\_connect('localhost', 'root', '', 'bd\_sklad');

if(mysql\_connect('localhost','root')){

echo "Hello!!!";

mysql\_select\_db('bd\_sklad',$con\_str);

$result=mysql\_query("DELETE FROM devilery WHERE id = '$id';");

**// Списание в полигон**

$result=mysql\_query("SELECT \* FROM `sclad`WHERE '2020-01-19' > `Shelf\_life`");

echo "<h3>Товар для списания</h3>";

echo "<table border=1px>";

echo"<tr><th>Номер</th>

<th>Название</th>

<th>Владелец</th>

<th>Вместимость</th>

<th>Срок хранения</th><td><a href=Add.php>Добавить</a></td><td><a href=del.php>Удалить</a></td><td><a href=Read.php>Проверить</a></td></tr>";

while($row = mysql\_fetch\_array($result)){

$ids=$row['id'];

$storage=$row['Title'];

$product=$row['Owner'];

$car=$row['capacity'];

$life=$row['Shelf\_life'];

$result=mysql\_query("INSERT INTO `polygon` (`id`, `sclad\_title`, `sclad\_owner`) VALUES ('$ids', '$sklad', '$product');");

$result=mysql\_query("DELETE FROM devilery WHERE id = '$ids';");