МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное ГОСУДАРСТВЕННОЕ бюджетное ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информатики

и вычислительной техники

Кафедра информационной

безопасности

Отчет к курсовому проекту

по дисциплине «Безопасность систем баз данных»

**Разработка базы данных кондитерской фирмы**

Вариант № 17

Выполнили: студенты группы БИ-31

Иконников К.М., Тюлькин А.А., Мокеев И.А.

Проверил: доцент кафедры

ИБ Сучков Д.С.

Йошкар-Ола

2019 г.

**СОДЕРЖАНИЕ**

**Введение**3

**1. Техническое задание4**

1.1 Требования к курсовой работе4

1.2 Требования к базе данных4

1.3 Требования к API (минимальное количество реализованных методов)4

**2. Порядок выполнения работы5**

2.1 Этапы разработки базы данных5-8

2.2 Этапы разработки API8

2.3 Скриншоты программы8-10

**3. Приложения11**

3.1 ER-диаграмма11

3.2 Ссылка на github.com11

**4. Вывод11**

**Введение:** в курсовой работе рассматривается создание базы данных, предназначенной для автоматизации работы кондитерской фирмы. База данных позволяет клиентам просматривать ассортимент, состав товара. Также администратору взаимодействовать с базой пользователей.

1. **Техническое задание**

***1.1 Требования к курсовой работе:***

* Получить структуру данных из файла, согласно варианту. Привести к 3й нормальной форме. Добавить недостающие таблицы.
* Составить ER-диаграмму
* Разработать API для базы данных на любом языке, выполняющемся на стороне сервера (php, ASP.NET, Java, python, node.js, etc)
* Взаимодействие должно осуществляться по клиент-серверной архитектуре, подключение с клиентской программы недопустимо
* Провести настройку пользователей базы данных для разграничения прав доступа, привести пример конфигурации
* Все документы и исходные коды для курсовой работы должны храниться под контролем системы контроля версий — git или mercurial (https://github.com/, https://bitbucket.org/)
* Во время сдачи курсового проекта необходимо предоставить отчет о проделанной работе в печатном виде (отчет)

***1.2 Требования к базе данных***

* Наличие не менее 7 таблиц, в том числе таблицы сессий и пользователей
* Структура таблицы должна содержать не менее 3-х полей, одно из которых ключевое
* Правомерное использование типов данных
* Обязательно использование триггеров и/или хранимых процедур
* Форма нормализации не менее 3NF
* Индексирование по полям поиска

**1.3 Требование к API (минимальное количество реализованных методов)**

* аутентификация пользователя (создание сессии);
* добавление/удаление/изменение данных в таблицах;
* выборка данных их ключевых таблиц по запросам;
* выборка данных из таблиц с объединением результатов.

1. **Порядок выполнения работы**
   1. Этапы разработки базы данных

Разработана база данных, содержащая 12 таблиц. В том числе таблицы сессий (*sessions*) и пользователей (*people*). Для авторизации пользователя используется таблица – *people,* содержащая информацию об аккаунте пользователя. Для хранения информации об ассортименте используются следующие таблицы:

*tovar* – название и цена товара;

*spec\_ingredients –* состав товара (ингредиенты);

Структуры, реализованных таблиц:

* Структура таблицы buyer

1. CREATE TABLE `buyer` (
2. `user\_id` varchar(45) NOT NULL,
3. `club\_card` varchar(50) NOT NULL
4. ) ENGINE=InnoDB DEFAULT CHARSET=utf8;

Индексы таблицы buyer

1. ALTER TABLE `buyer`
2. ADD PRIMARY KEY (`user\_id`);

Внешние ключи таблицы buyer

1. ALTER TABLE `buyer`
2. ADD CONSTRAINT `buyer\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `people` (`user\_id`) ON DELETE CASCADE;

* Структура таблицы ingredients

1. CREATE TABLE `ingredients` (
2. `tovar\_id` **int**(11) DEFAULT NULL,
3. `id\_ing` **int**(11) DEFAULT NULL,
4. `ing\_count` **int**(11) DEFAULT NULL
5. ) ENGINE=InnoDB DEFAULT CHARSET=utf8;

* Структура таблицы people

1. CREATE TABLE `people` (
2. `user\_id` varchar(45) NOT NULL,
3. `username` varchar(300) DEFAULT NULL,
4. `email` varchar(300) DEFAULT NULL,
5. `password` varchar(300) DEFAULT NULL,
6. `fio` varchar(300) DEFAULT NULL,
7. `address` text DEFAULT NULL,
8. `registration` datetime DEFAULT NULL,
9. `role` varchar(20) NOT NULL
10. ) ENGINE=InnoDB DEFAULT CHARSET=utf8;

Индексы таблицы people

1. ALTER TABLE `people`
2. ADD PRIMARY KEY (`user\_id`),
3. ADD UNIQUE KEY `username\_password\_index` (`username`,`password`),
4. ADD KEY `fio\_index` (`fio`);

* Структура таблицы people\_info

1. CREATE TABLE `people\_info` (
2. `user\_id` varchar(45) NOT NULL,
3. `birthday\_date` date DEFAULT NULL,
4. `about` varchar(300) DEFAULT NULL,
5. `phone\_number` varchar(50) DEFAULT NULL,
6. `company` varchar(300) DEFAULT NULL
7. ) ENGINE=InnoDB DEFAULT CHARSET=utf8;

Индексы таблицы people\_info

1. ALTER TABLE `people\_info\_two`
2. ADD PRIMARY KEY (`user\_id`);

Внешние ключи таблицы people\_info

1. ALTER TABLE `people\_info`
2. ADD CONSTRAINT `people\_info\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `people` (`user\_id`) ON DELETE CASCADE;

* Структура таблице people\_info\_two

1. CREATE TABLE `people\_info\_two` (
2. `user\_id` varchar(45) NOT NULL,
3. `card\_number` varchar(300) DEFAULT NULL,
4. `amount` **float** DEFAULT NULL
5. ) ENGINE=InnoDB DEFAULT CHARSET=utf8;

Индексы таблицы people\_info\_two

1. ALTER TABLE `people\_info\_two`
2. ADD PRIMARY KEY (`user\_id`);

Внешние ключи таблицы people\_info\_two

1. ALTER TABLE `people\_info\_two`
2. ADD CONSTRAINT `people\_info\_two\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `people` (`user\_id`) ON DELETE CASCADE;

* Структура таблицы provider

1. CREATE TABLE `provider` (
2. `user\_id` varchar(45) NOT NULL,
3. `rating` **int**(11) DEFAULT NULL
4. ) ENGINE=InnoDB DEFAULT CHARSET=utf8;

Индексы таблицы provider

1. ALTER TABLE `provider`
2. ADD PRIMARY KEY (`user\_id`);

Внешние ключи таблицы provider

1. ALTER TABLE `provider`
2. ADD CONSTRAINT `provider\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `people` (`user\_id`) ON DELETE CASCADE;

* Структура таблицы sessions

1. CREATE TABLE `sessions` (
2. `user\_id` varchar(45) NOT NULL,
3. `token` varchar(100) DEFAULT NULL,
4. `date` datetime DEFAULT NULL
5. ) ENGINE=InnoDB DEFAULT CHARSET=utf8;

Индексы таблицы sessions

1. ALTER TABLE `sessions`
2. ADD PRIMARY KEY (`user\_id`);

Внешние ключи таблицы sessions

1. ALTER TABLE `provider`
2. ADD CONSTRAINT `provider\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `people` (`user\_id`) ON DELETE CASCADE;

* Структура таблицы sklad

1. CREATE TABLE `sklad` (
2. `sklad\_id` **int**(11) NOT NULL,
3. `sklad\_name` varchar(100) DEFAULT NULL,
4. `tovar\_count` **int**(11) DEFAULT NULL
5. ) ENGINE=InnoDB DEFAULT CHARSET=utf8;

Индексы таблицы sklad

1. ALTER TABLE `sklad`
2. ADD PRIMARY KEY (`sklad\_id`);

AUTO\_INCREMENT таблицы sklad

1. ALTER TABLE `sklad`
2. MODIFY `sklad\_id` **int**(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=4;

* Структура таблицы spec\_ingredients

1. CREATE TABLE `spec\_ingredients` (
2. `id\_ing` **int**(11) NOT NULL,
3. `name\_ing` varchar(100) DEFAULT NULL,
4. `user\_id` varchar(45) DEFAULT NULL
5. ) ENGINE=InnoDB DEFAULT CHARSET=utf8;

Индексы таблицы spec\_ingredients

1. ALTER TABLE `spec\_ingredients`
2. ADD PRIMARY KEY (`id\_ing`),
3. ADD KEY `user\_id` (`user\_id`);

AUTO\_INCREMENT таблицы spec\_ingredients

1. ALTER TABLE `spec\_ingredients`
2. MODIFY `id\_ing` **int**(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=15;

* Структура таблицы staff

1. CREATE TABLE `staff` (
2. `user\_id` varchar(45) NOT NULL,
3. `post` varchar(50) DEFAULT NULL,
4. `salary` **int**(11) DEFAULT NULL
5. ) ENGINE=InnoDB DEFAULT CHARSET=utf8;

Индексы таблицы staff

1. ALTER TABLE `staff`
2. ADD PRIMARY KEY (`user\_id`);

Внешние ключи таблицы staff

1. ALTER TABLE `staff`
2. ADD CONSTRAINT `staff\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `people` (`user\_id`) ON DELETE CASCADE;

* Структура таблицы tovar

1. CREATE TABLE `tovar` (
2. `tovar\_id` **int**(11) NOT NULL,
3. `name\_tovar` varchar(100) DEFAULT NULL,
4. `massa\_tovar` **int**(11) DEFAULT NULL,
5. `price\_tovar` **int**(11) DEFAULT NULL
6. ) ENGINE=InnoDB DEFAULT CHARSET=utf8;

Индексы таблицы tovar

1. ALTER TABLE `tovar`
2. ADD PRIMARY KEY (`tovar\_id`);

AUTO\_INCREMENT таблицы spec\_ingredients

1. ALTER TABLE `tovar`
2. MODIFY `tovar\_id` **int**(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=17;

Внешние ключи таблицы spec\_ingredients

1. ALTER TABLE `spec\_ingredients`
2. ADD CONSTRAINT `spec\_ingredients\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `provider` (`user\_id`) ON DELETE CASCADE;

* Структура таблицы tovar\_on\_sklad

1. CREATE TABLE `tovar\_on\_sklad` (
2. `tovar\_id` **int**(11) NOT NULL,
3. `sklad\_id` **int**(11) DEFAULT NULL,
4. `count` **int**(11) DEFAULT NULL );

Используемые триггеры:

* *add\_tovar\_on\_sklad –* триггер отвечающий за добавление кол-ва товара на склад из таблицы tovar\_on\_sklad ;

1. BEGIN
2. UPDATE sklad set tovar\_count = tovar\_count - OLD.count;
3. END

* *drop\_tovar\_on\_sklad –* триггер отвечающий за изменения кол-ва товара на складе при удалении товара из таблицы tovar\_on\_sklad;

1. BEGIN
2. UPDATE sklad set tovar\_count = tovar\_count - OLD.count;
3. END

* *drop\_ing –* триггер отвечающий за удаление ингредиентов при удалении товара из таблицы tovar;

1. BEGIN
2. DELETE FROM ingredients WHERE tovar\_id = OLD.tovar\_id;
3. END

Проведена настройка пользователей базы данных для разграничения прав доступа:

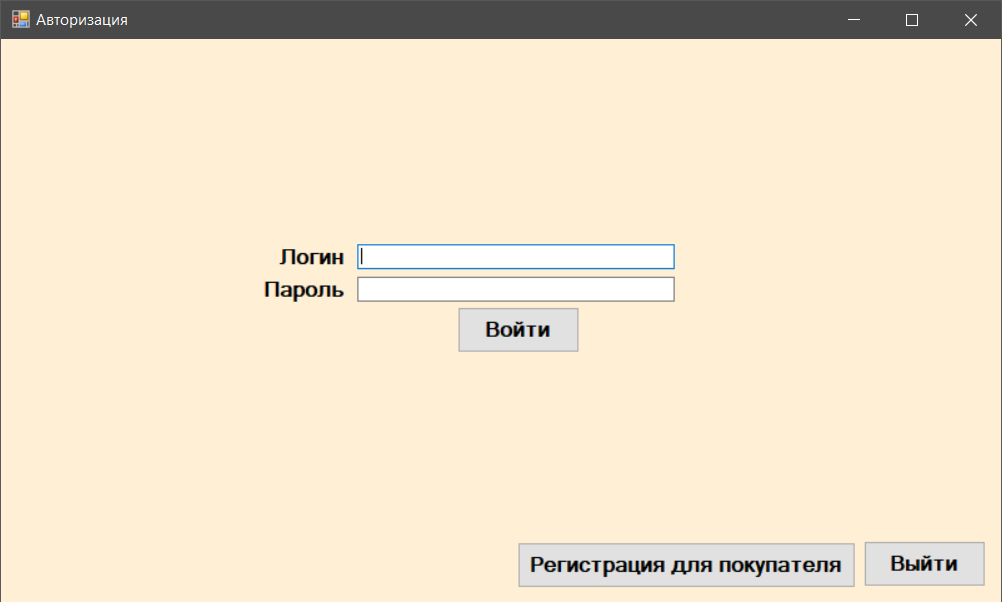


*Рис 1. Пользователи базы данных*

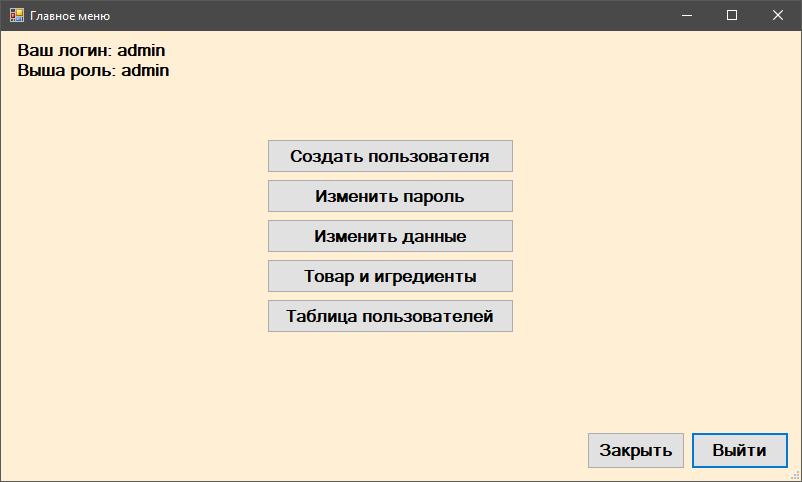
2.2 Этапы разработки API

Было разработано API для аутентификации пользователя и взаимодействия с базой. API было написано на языке С# .

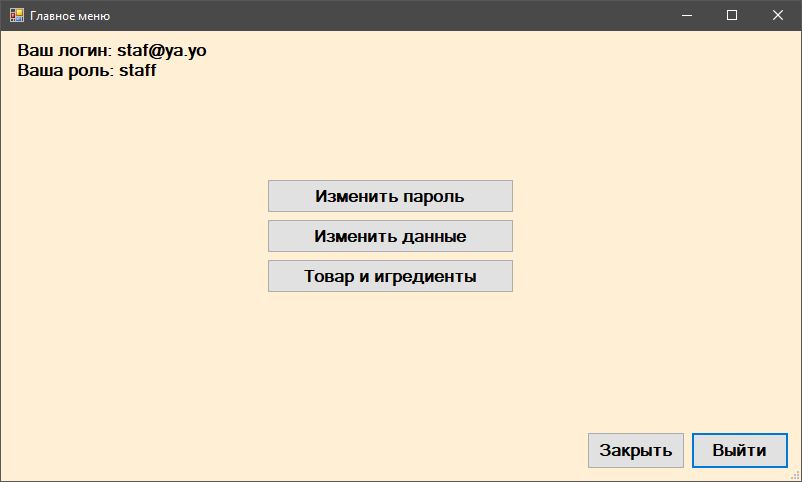
2.3 Скриншоты



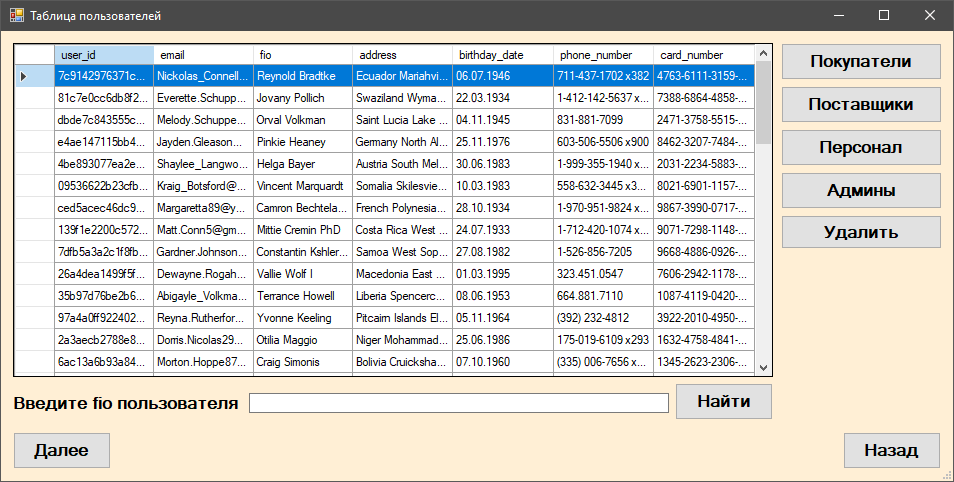
*Рис 2. Авторизаиця*



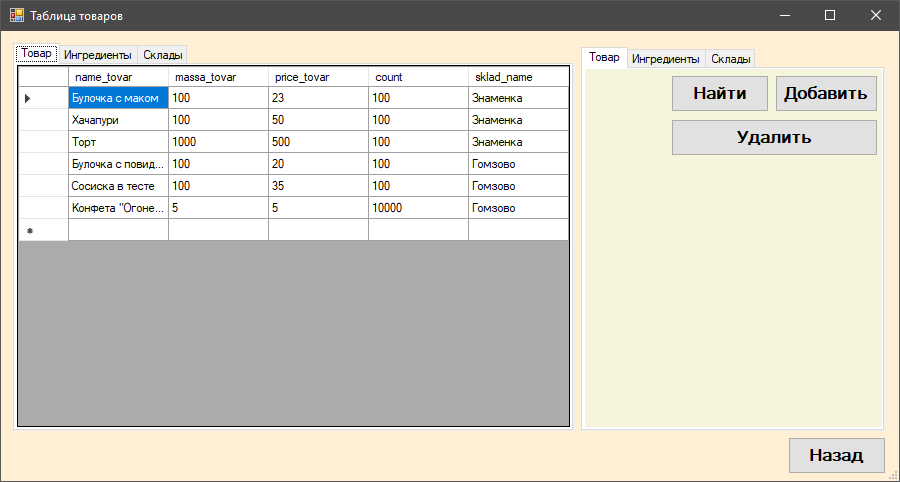
*Рис 3. Главное меню(администратор)*



*Рис 4. Главное меню(юзер)*



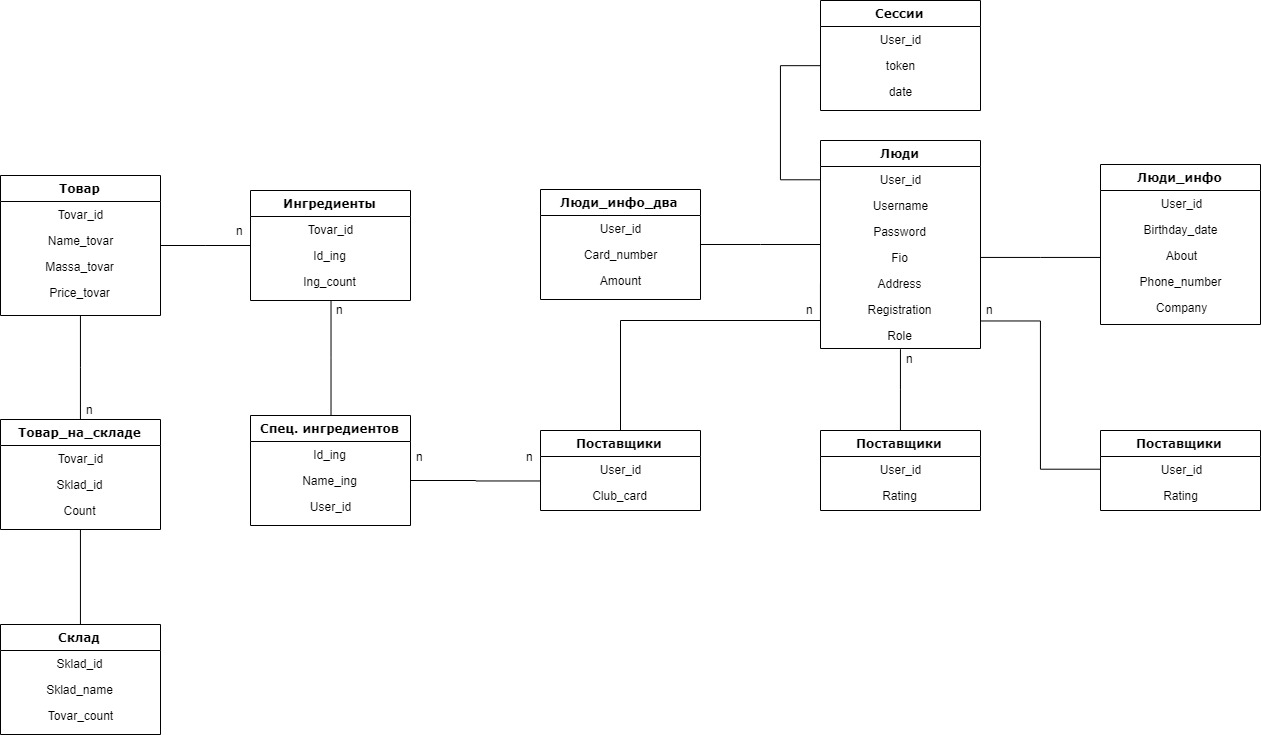
*Рис 5. Таблица пользователей доступная админу*



*Рис 6. Взаимодействие с товаром*

**Приложения**

1. ER-диаграмма



База данных находится в первой нормальной форме, т.к. в любом допустимом значении отношения каждый его кортеж содержит только одно значение для каждого из атрибутов. Т.к. база данных находится в 1НФ, то она также находится во второй нормальной форме, потому что каждый неключевой атрибут неприводимо (функционально полно) зависит от её потенциального ключа. Наличие 2НФ и отсутствие зависимости неключевых атрибутов от ключевых доказывает, что база данных находится в третьей нормальной форме.

1. Исходные коды и документы:

<https://github.com/kiryosha/DS-course-work>

**Вывод:** во время выполнения курсового проекта были изучены методы работы с базами данных, способы управления. Таким образом, в ходе работы была разработана база данных, а также API, для автоматизации работы кондитерской фирмы.