

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

**«МИРЭА – Российский технологический университет» РТУ МИРЭА**

Институт информационных технологий Кафедра промышленной информатики (ПИ)

**ОТЧЕТ ПО ПРАКТИЧЕСКИМ РАБОТАМ**

**по дисциплине**

«Разработка баз данных»

**Студент группы:** ИВБО-02-19 Денисов К. Ю. 

**Преподаватель:** Инженер, ассистент Киселев Д. С.

Работа выполена « » 2021 г.

«Зачтено» « » 2021 г.

Москва 2021

**СОДЕРЖАНИЕ**

[ПЕРЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ](#_bookmark0) . . . . . . . . . . . . . . 3 [ВВЕДЕНИЕ](#_bookmark1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4

[1 Построение модели данных](#_bookmark2) . . . . . . . . . . . . . . . . . . . . . . . . 5

* 1. [Работа по методическим указаниям](#_bookmark3) . . . . . . . . . . . . . . . . . . 5
     1. [Описание предметной области](#_bookmark4) . . . . . . . . . . . . . . . . . . . . . 5 [1.1.2 Построение модели](#_bookmark5) . . . . . . . . . . . . . . . . . . . . . . . . . . . 7 [1.2 Индивидуальное задание](#_bookmark7) . . . . . . . . . . . . . . . . . . . . . . . . 7

[1.2.1 Описание предметной области](#_bookmark8) . . . . . . . . . . . . . . . . . . . . . 8 [1.2.2 Построение модели](#_bookmark9) . . . . . . . . . . . . . . . . . . . . . . . . . . . 9

1. [Создание логической и физической модели данных](#_bookmark11) 11
   1. [Работа по методическим указаниям](#_bookmark12) 11
      1. [Построение модели данных](#_bookmark13) 11
   2. [Индивидуальное задание](#_bookmark16) 13
      1. [Построение модели данных](#_bookmark17) 13
2. [Создание базы данных](#_bookmark20) 15
   1. [Работа по методическим указаниям](#_bookmark21) 15
   2. [Индивидуальное задание](#_bookmark30) 19
      1. [Использование MySQL CLI Client](#_bookmark33) 22
      2. [Использование MySQL Workbench](#_bookmark38) 25
3. [Создание веб–клиента](#_bookmark42) 27
   1. [Общие требования к приложению](#_bookmark43) 27
      1. [Требования к функционалу](#_bookmark44) 27
      2. [Требования к интерфейсу](#_bookmark45) 27
   2. [Реализация приложения](#_bookmark46) 28
      1. [Используемые технологии](#_bookmark47) 28
      2. [Возможные сценария использования веб-приложения](#_bookmark48) 29
      3. [Демонстрация работы веб-приложения](#_bookmark49) 29
         1. [Авторизация](#_bookmark50) 30
         2. [Отображение записей таблиц](#_bookmark52) 30
         3. [Добавление записей в в таблицы](#_bookmark55) 31
         4. [Выполнение произвольных запросов](#_bookmark57) 32

[ЗАКЛЮЧЕНИЕ](#_bookmark59) 34

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ](#_bookmark60) 35

**ПЕРЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ**

**AJAX** — Asynchronous Javascript and XML

**CLI** — Command line interface

**DDL** — Data Definition Language **DML** — Data Manipulation Language **CSS** — Cascading Style Sheets

**HTML** — HyperText Markup Language

**SQL** — Structured query language.

**БД** — База данных.

**ПО** — Программное обеспечение.

**СУБД** — Система управления базами данных.

**ВВЕДЕНИЕ**

Целью работы является построение логической и физической мо­ дели данных выбранной предметной области, освоение навыков работы с программным обеспечением ERwin Data Modeler, ER Assistant, MySQL Workbench, MySQL Server.

На заключительном этапе работы необходимо представить веб–прило­ жение или полнофункциональное приложение, организующее взаимодействие пользователя с созданной базой данных.

* + - * 1. **Построение модели данных**

В данном разделе описывается процесс создания проекта базы данных в среде ErAssistant *(ver. 2.10).*

**Работа по методическим указаниям**

Произведем разработку проекта базы данных по теме, утвержденной в рамках дисциплины «Разработка базы данных».

**Описание предметной области**

Компания занимается производством и продажей небольших статуэток, раскрашиваемых вручную. Компания имеет несколько производственных на­ правлений. Миниатюры изготавливаются из гипса, олова или алюминия.

Компания распространяет свои товары по трем каналам. Компания со­ держит пять собственных розничных магазинов. Помимо этого, компания вла­ деет сайтом, на котором ведется online-торговля, и осуществляет оптовые по­ ставки сторонним дистрибьюторам. Для анализа статистики, системой автома­ тизации производства, нужен интерактивный аналитический инструмент. По­ этому необходимо спроектировать и построить модель данных, которая станет хранилищем информации по производству.

В ходе производства изделий система автоматизации производства управляет всеми станками компании. Каждый станок реализует полный цикл производства изделий, включая:

* заполнение формы сырьем (гипсом, оловом или алюминием);
* затвердевание материала;
* удаление изделия из формы после затвердевания;
* при необходимости автоматизированная раскраска изделий (оловянные фигурки не раскрашиваются);
* сушку после покраски (при необходимости).

Покраска и сушка могут производиться за несколько этапов в зависи­ мости от сложности изделия. По мере готовности изделия проходят проверку, выполняемую оператором станка.

Оператор станка регистрируется в системе. В ходе этого процесса опе­ ратор сообщает системе автоматизации производства тип производимых изде­ лии и объем загруженного в машину сырья. Оператор также делает в системе запись при отбраковке изделий.

В ходе интервью необходимые для эффективного анализа статистики:

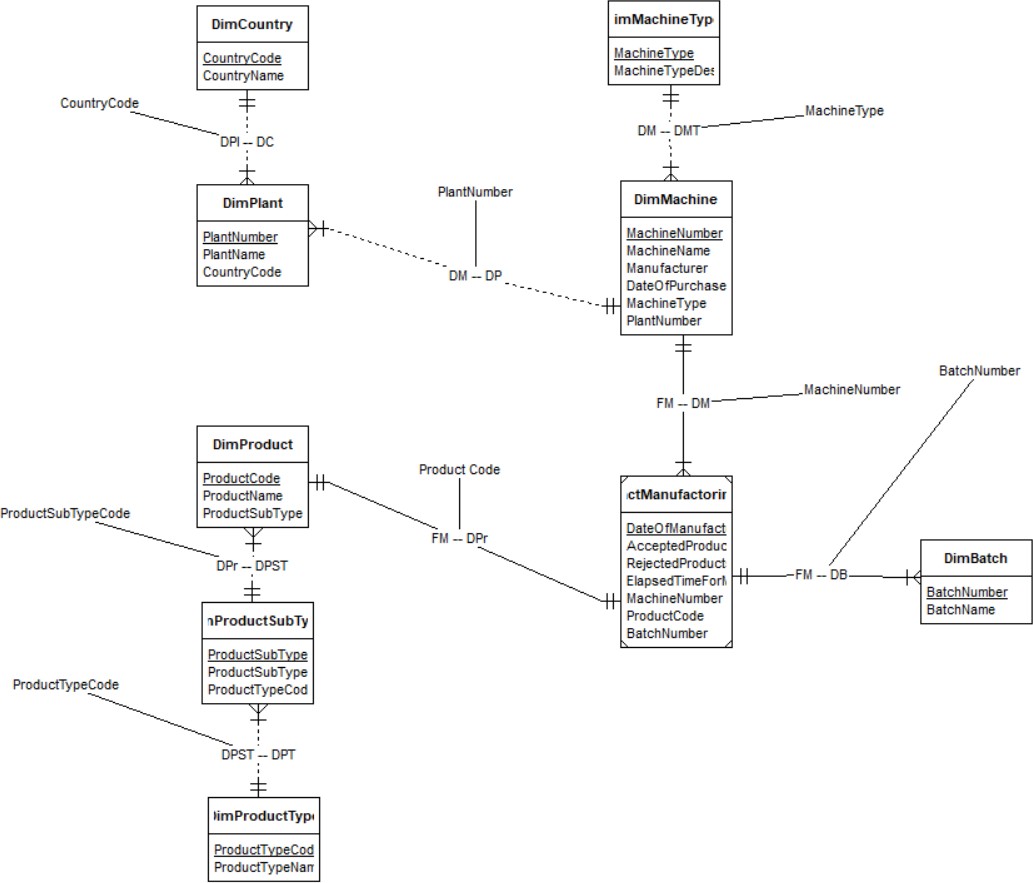
* число принятых изделий по объему сырья, видам изделий, машинам и
* время формовки и затвердевания по видам изделий, машинам и дням;
* время покраски и сушки по типам краски, видам изделий, машинам и
* сворачивание по подтипам изделий, которые сворачиваются по типам;
* сворачивание по типам машин, которые сворачиваются по материалам (гипс, олово или алюминий);
* сворачивание машин по фабрикам, которые сворачиваются по странам;
* сворачивание дней по месяцам, месяцев — по кварталам;
* возможность фильтрации информации по производителю и дате покуп­ ки машины.

Анализ файла-экспорта из системы автоматизации производства пока­ зал, что для каждого вида производимых изделий есть отдельная строка, в которой присутствует следующая информация:

* тип изделия;
* объем сырья;
* номер машины;
* личный номер оператора;
* время и дата начала производства (когда серия начата);
* время и дата окончания производства (когда серия закончена);
* флаг отбраковки.

**Построение модели**

По приведенному описанию предметной области построим ее модель в среде ErAssistant . Укажем линии связей, назначим им имена, укажем типы и кратность связей. В результате работы, модель примет вид, приведенный на Рисунке [1.1](#_bookmark6).



**Рисунок 1.1 — Модель данных Производства со связями**

**Индивидуальное задание**

Произведем разработку проекта базы данных по индивидуальной теме.

**Описание предметной области**

В качестве индивидуального задания была выбрана реализация моде­ ли информационной системы по хранению и анализу данных предприятия, занимающегося сборкой и поставкой спортивных велосипедов для конечных потребителей по индивидуальному заказу.

Предприятие располагает широким выборов компонентов и комплекту­ ющих для сборки велосипедов следующих типов:

* дорожный;
* горный;
* кросс-кантри;
* эндуро;
* прогулочный.

Предприятие может предложить сконфигурировать велосипед, отдельно выбрав каждый из предложенных компонентов:

* рама;
* вилка;
* руль;
* трансмиссия;
* колеса;
* тормозная система.

Контроль над выполнением работ по сборке велосипеда проводится в виде учета всех операций по сборке, настройке и тестированию, проводимых на территории предприятия ее сотрудниками. При этом каждая запись содер­ жит следующую информацию о проведенных работах:

* внутренний номер изделия;
* время;
* этап работ;
* название цеха;
* имя мастера;
* статус;
* примечание.

На предприятии ведется учет всех компонентов велосипедов. В базе дан­ ных предприятия хранится информация о каждом компоненте, приобретенном у партнеров или изготовленном самостоятельно.

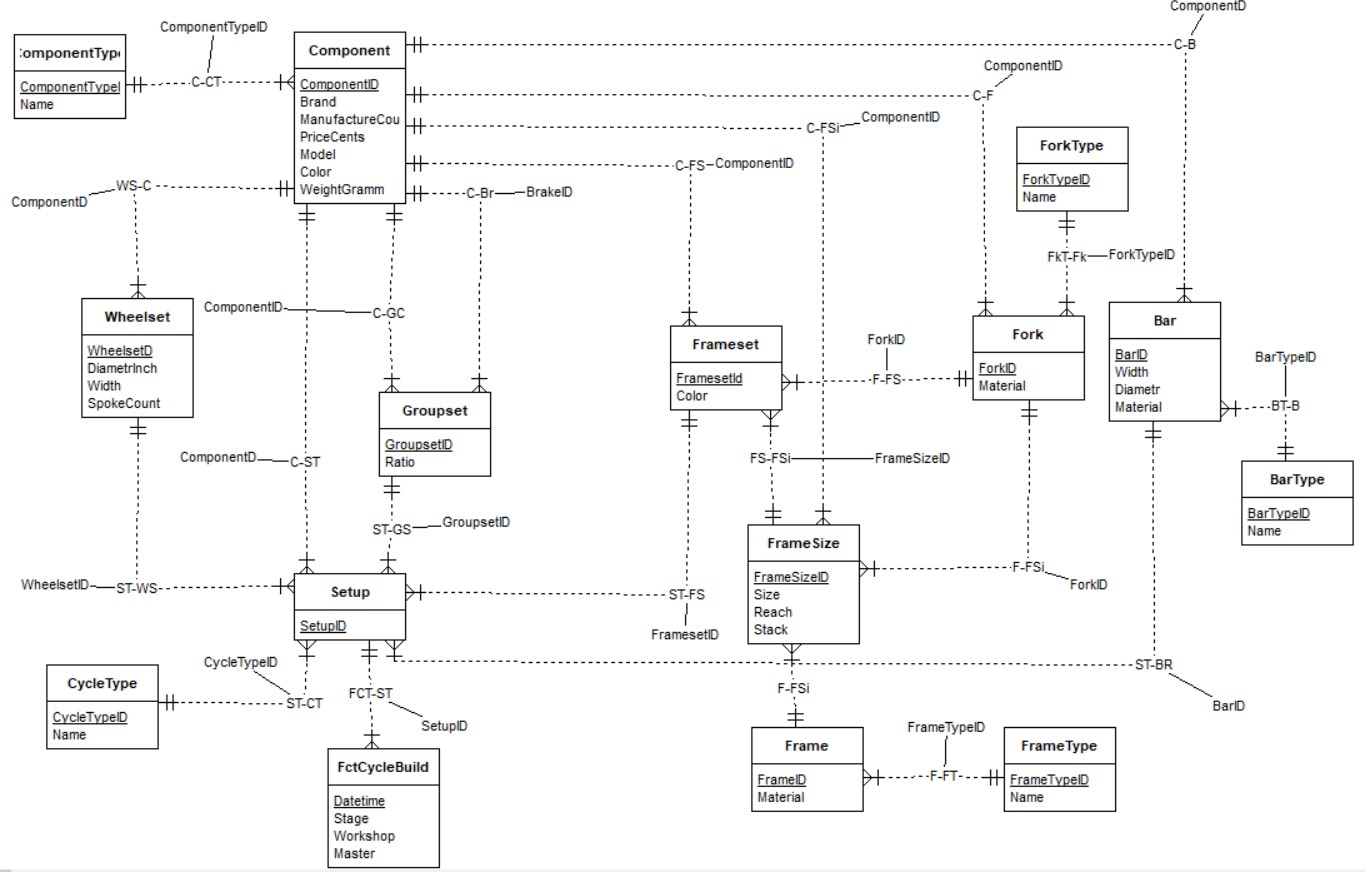
В независимости от типа компонента он обладает общей информацией о наименовании производителя, месте и времени изготовления, типе и рекомен­ дованной розничной цене. Также каждый компонент имеет особые сведения, присущие данному типу детали.

**Построение модели**

После приведения общих сведений о роде деятельности предприятия, факторизируем модель данных информационной системы предприятия в сре­ де ErAssistant (cм Рисунок [1.2](#_bookmark10)). Приложение ErAssistant позволяет пользова­ телю создавать, редактировать диаграммы сущностей и связей.

Укажем названия связей, их идентификаторы и кратность, исходя из вида отношений, выстроенных между сущностями.

На данном этапе выполнения работы мы реализовали проект базы дан­ ных будущего хранилища данных, планируемого к применению в компании, занимающейся сборкой и поставкой велосипедов по индивидуальному зака­ зу. Был проведен анализ переметной области, определен список сущностей, которые наиболее полно смогут описать сущности, участвующие в производ­ ственном процессе.



**Рисунок 1.2 — Модель данных вело-предприятия**

# Создание логической и физической модели данных

В данном разделе будет рассмотрено создание логической и физиче­ ской модели данных предложенных предметных областей в ПО ERwin Data Modeler.

**Работа по методическим указаниям**

Порядок построения модели данных в среде ERwin Data Modeler рас­ смотрим на примере автоматизированной информационной системы «Реали­ зация средств вычислительной техники», предназначенной для учета продаж настольных компьютеров по заказам клиентов.

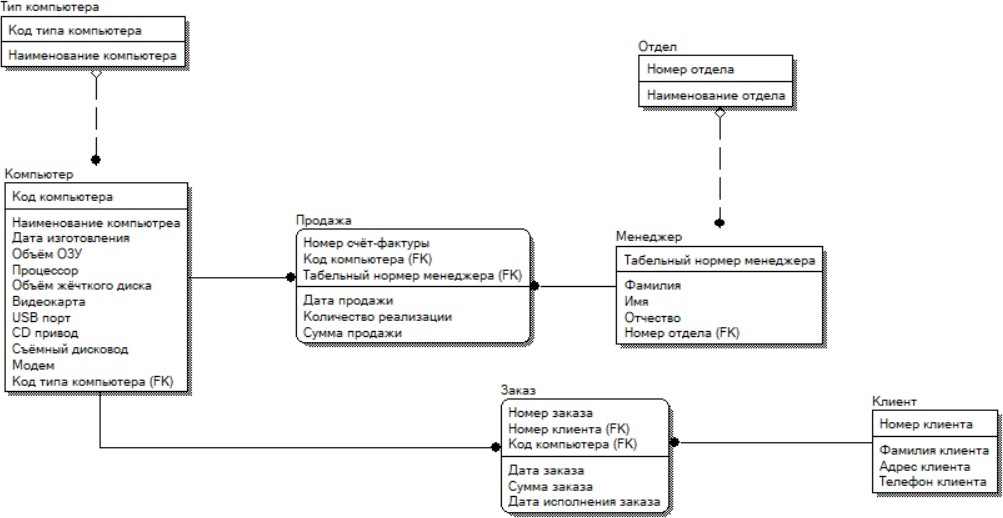
Создание модели данных начинается с разработки логической модели, которая должна представлять состав сущностей предметной области с переч­ нем атрибутов и отношений между ними.

**Построение модели данных**

Результат разработки логической модели данных системы «Реализация средств вычислительной техники», предназначенной для учета продаж на­ стольных компьютеров по заказам клиентов приведен на Рисунке [2.1](#_bookmark14).

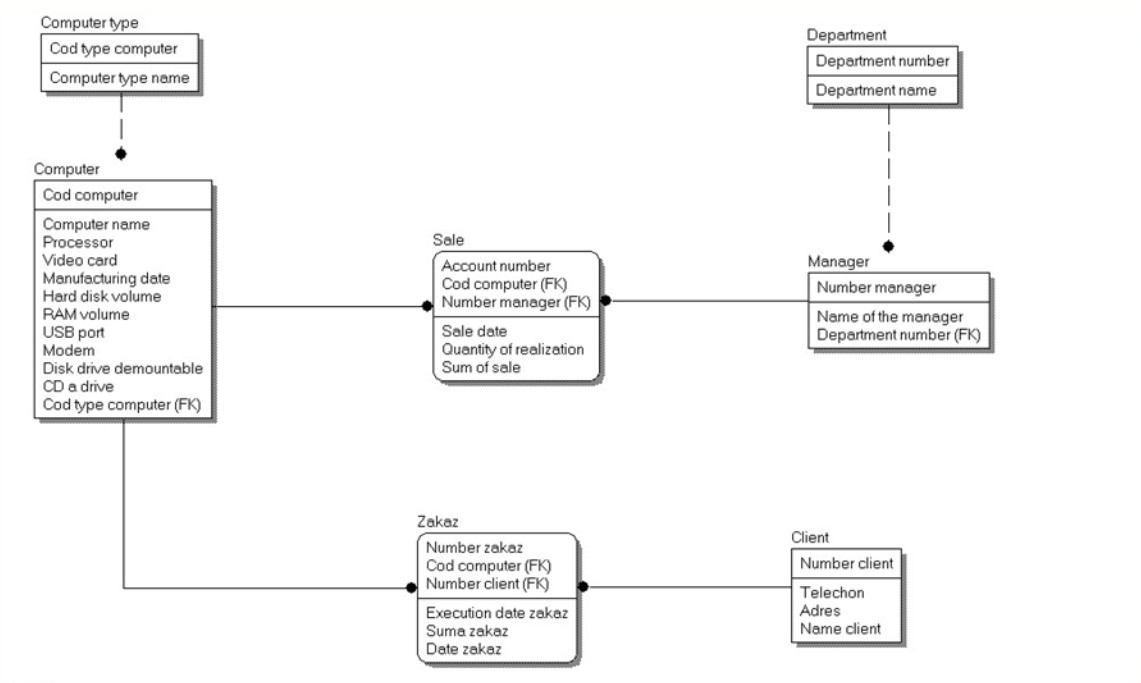
Для построения физической модели данных системы, следует опреде­ литься с СУБД, в которой будет реализована модель. При построении физиче­ ской модели данных следует учитывать формальную теория представления и обработки данных в конкретной системе управления базами данных (СУБД).

В данной практической работе в качестве СУБД выбрана MySQL.



**Рисунок 2.1 — Логическая модель данных системы «Реализация средств вычислительной техники»**

Приступим к построению физической модели данных системы «Реали­ зация средств вычислительной техники». Результат работы можно видеть на Рисунке [2.2](#_bookmark15).



**Рисунок 2.2 — Физическая модель данных системы «Реализация средств вычислительной техники»**

**Индивидуальное задание**

Приступим к построению логической модели данных системы «Велоси­ педное предприятие». В соответствии с моделью, реализованной в ходе пер­ вой практической работы, добавим в рабочую область следующие сущности:

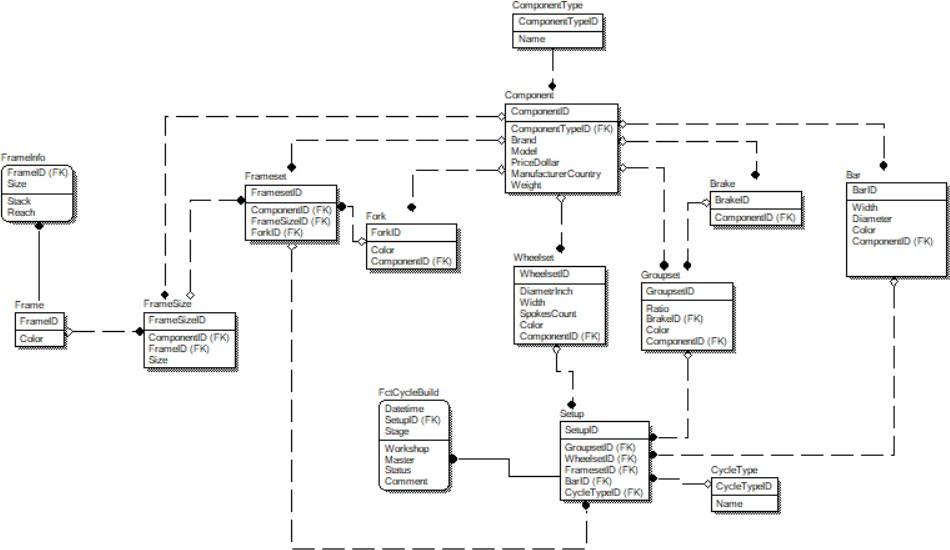
* Component;
* FrameInfo;
* Frame;
* Frameset;
* FrameSize;
* Fork;
* ComponentType;
* Wheelset;
* Groupset;
* Brake;
* FctCycleBuild;
* CycleType;
* Bar;
* Setup.

Добавим связи между сущностями в соответствии с ранее построенной моде­ лью. Логическая модель системы «Велосипедное предприятие» приведена на Рисунке [2.3](#_bookmark18).

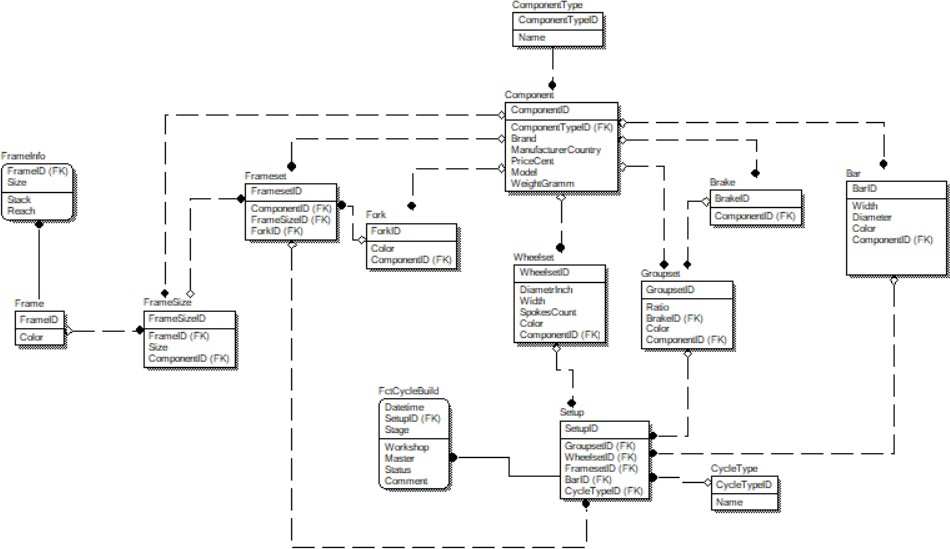
**Построение модели данных**

После уточнения типов данных, выбранных в соответствии с предмет­ ной областью и спецификой СУБД MySQL. Физическая модель системы «Ве­ лосипедное предприятие» приведена на Рисунке [2.4](#_bookmark19).

После реализации физической и логической модели можно приступать к реализации модели данной системы в СУБД MySQL.



**Рисунок 2.3 — Логическая модель данных системы «Велосипедное предприятие»**



**Рисунок 2.4 — Физическая модель данных системы «Велосипедное предприятие»**

# Создание базы данных

**Работа по методическим указаниям**

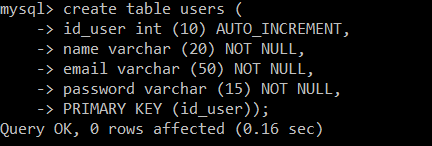
Создадим базу данных forum, которая хранит в себе сведения о пользо­ вателях форумах и размещенных ими темах.

Помимо суперпользователя root, был создан пользователь denilai, под которым производятся все манипуляции с данными.

Создадим базу данных forum с помощью команды CREATE DATABASE

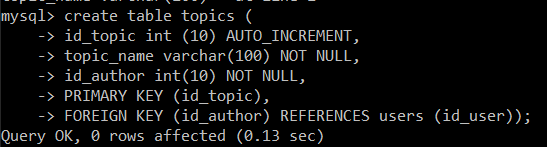
forum;

Создадим таблицу users (см. Рисунок [3.1](#_bookmark22)):



**Рисунок 3.1 — Создание базы данных users**

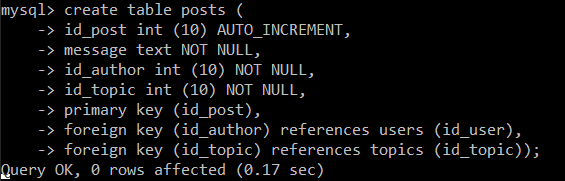
Создадим таблицу topics (см. Рисунок [3.2](#_bookmark23)):



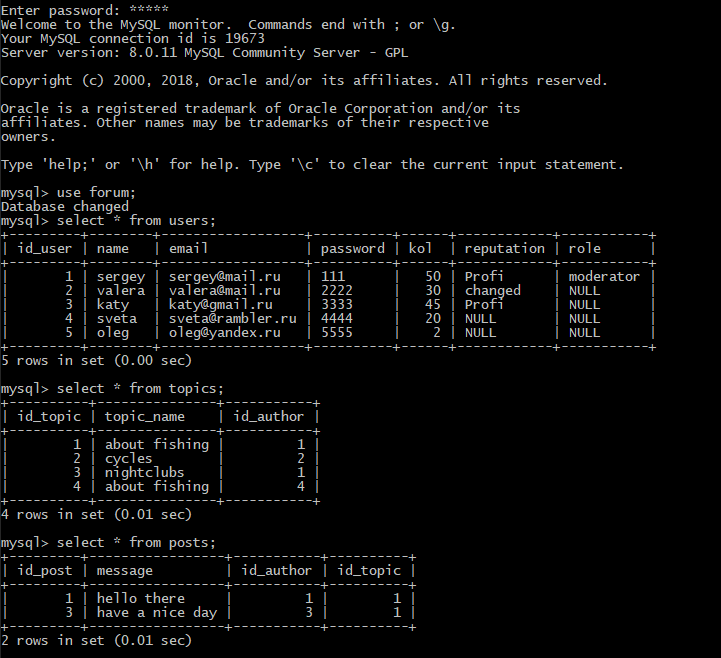
**Рисунок 3.2 — Создание базы данных topics**

Создадим таблицу posts (см. Рисунок [3.3](#_bookmark24)):

После создания таблиц, заполним их данными о пользователях форума, о темах и размещенных публикациях. Выполним операцию выборки данных без условия, чтобы увидеть все записи, занесенные в таблицы с помощью команды SELECT \* FROM <table-name> (см. Рисунок [3.4](#_bookmark25)).



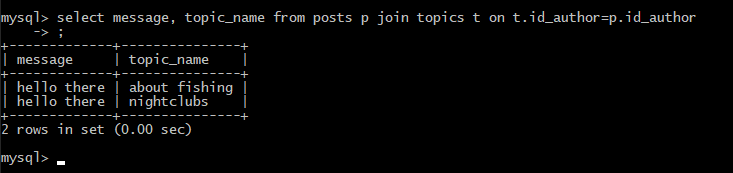
**Рисунок 3.3 — Создание базы данных posts**



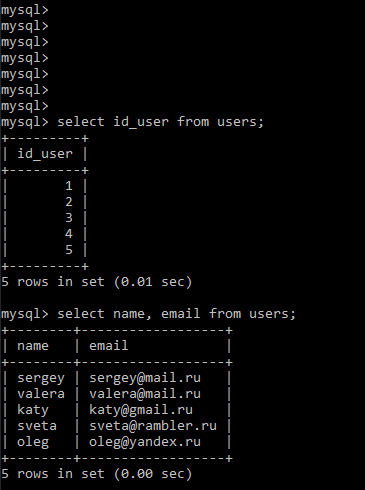
**Рисунок 3.4 — Операция выборки из всех таблиц**

Выполним запрос SELECT mesage, topic\_name FROM posts p JOIN topics t ON t.id\_author = p.id\_author; для объединения данных из таблиц topics и posts по ключу id\_author и получения полной информации о сообщении и названию темы, в которой оно было размещено (см. Рисунок [3.5](#_bookmark26)).

Выполним запрос выборки данных, явно указав поля отношения. Для этого перечислим имена полей через запятую после зарезервированного слова SELECT (см. Рисунок [3.6](#_bookmark27)).



**Рисунок 3.5 — Запрос объединения**

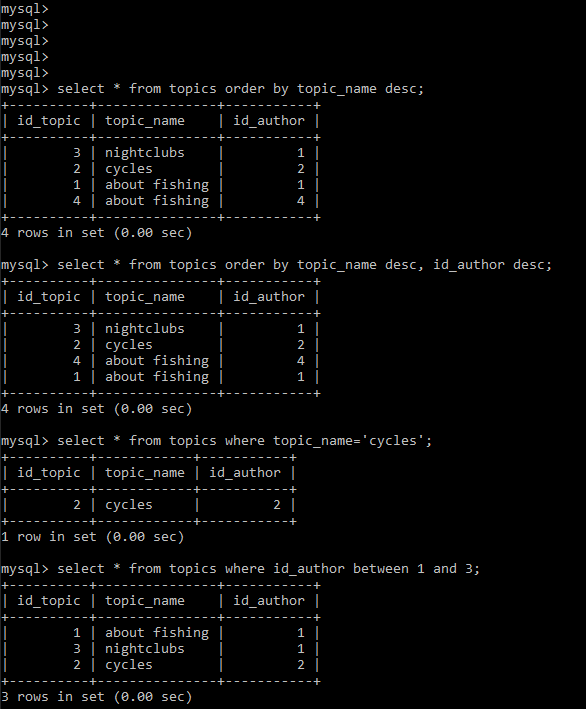


**Рисунок 3.6 — Операция выборки с указанием полей**

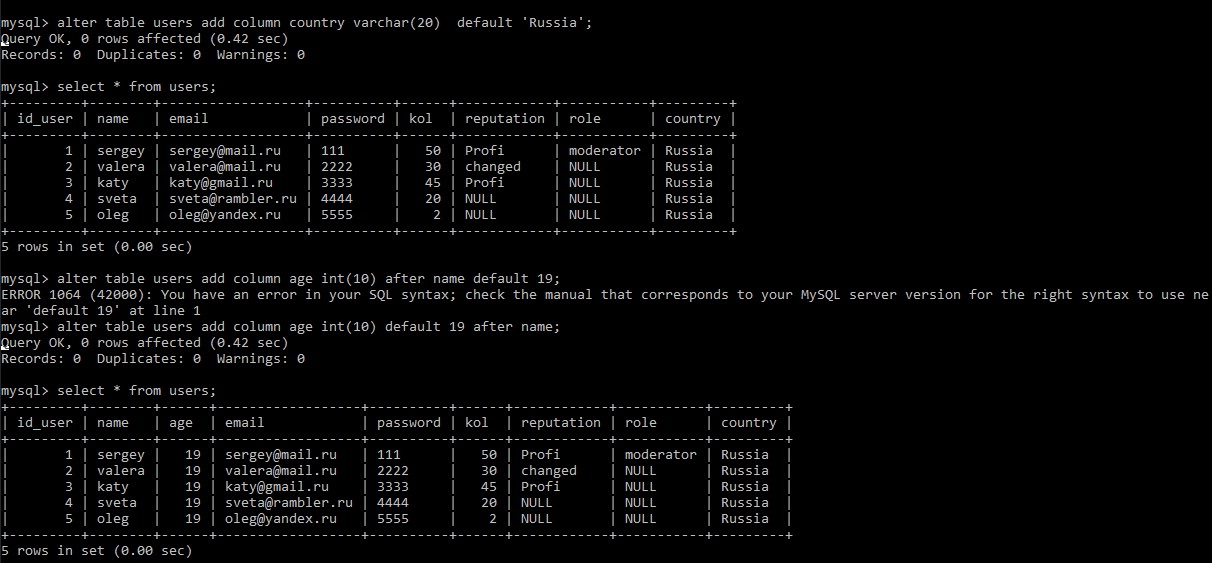
Выполним более сложные запросы выборки, отсортировав записи в таб­ лице topics по убыванию значения поля topic\_name и id\_author, а также опишем условие сравнения значения поля id\_author в сецкции WHERE(см. Ри­ сунок [3.7](#_bookmark28)).

Выполним операции по модификации таблицы — добавим в таблицу users поле country типа varchar (20) со значением по умолчанию "Russia а также добавим в эту же таблицу поле age int(10) со значением по умолча­ нию 19.

Выведем все записи из таблицы users, обнаружим, что столбцы были вставлены успешно (см. Рисунок [3.8](#_bookmark29)).



**Рисунок 3.7 — Сложные запросы выборки с сортировкой и условием**



**Рисунок 3.8 — Добавление полей в таблицу** users

В ходе данной практической работы были рассмотрены операторы DDL и DML диалекта MySQL. С помощью данных операторов была создана база учебная база данных forum, содержащая о пользователях форумах и разме­ щенных ими темах.

**Индивидуальное задание**

Продолжим работу над созданием модели данных велосипедного пред­ приятия. Создадим базу данных cycle, сущностями которой будут таблицы, описания которых были проработаны в прошлых практических работах.

Описание проектируемых отношений базы данных cycle приведены в таблицах [3.1](#_bookmark31)–[3.15](#_bookmark32).

*Таблица 3.1 — Описание таблицы Bar*

|  |  |
| --- | --- |
| **Имя** | **Тип** |
| Width | INTEGER NOT NULL |
| Diameter | INTEGER NOT NULL |
| BarID | INTEGER NOT NULL AUTO\_INCREMENT |
| Color | VARCHAR(20) NULL DEFAULT ’black’ |

*Таблица 3.2 — Описание таблицы Brake*

|  |  |
| --- | --- |
| **Имя** | **Тип** |
| BrakeID | INT NOT NULL AUTO\_INCREMENT |
| ComponentID | INTEGER NULL |

*Таблица 3.3 — Описание таблицы Component*

|  |  |
| --- | --- |
| **Имя** | **Тип** |
| ComponentID | INTEGER NOT NULL AUTO\_INCREMENT |
| Brand | VARCHAR(20) NOT NULL |
| ManufacturerCountry | VARCHAR(20) NOT NULL |
| PriceCent | INTEGER NOT NULL |
| Model | VARCHAR(20) NOT NULL |
| ComponentTypeID | INTEGER NOT NULL |
| WeightGramm | INTEGER NOT NULL |

*Таблица 3.4 — Описание таблицы ComponentType*

|  |  |
| --- | --- |
| **Имя** | **Тип** |
| ComponentTypeID | INTEGER NOT NULL AUTO\_INCREMENT |
| Name | VARCHAR(20) |

*Таблица 3.5 — Описание таблицы CycleType*

|  |  |
| --- | --- |
| **Имя** | **Тип** |
| CycleTypeID | INTEGER NOT NULL AUTO\_INCREMENT |
| Name | VARCHAR(20) NOT NULL |

*Таблица 3.6 — Описание таблицы FctCycleBuild*

|  |  |
| --- | --- |
| **Имя** | **Тип** |
| SetupID | INTEGER NOT NULL |
| Datetime | DATE NOT NULL |
| Stage | INTEGER NOT NULL |
| Workshop | varchar(20) NOT NULL DEFAULT ’main’ |
| Master | VARCHAR(20) NOT NULL |
| Status | VARCHAR(20) NOT NULL |
| Comment | VARCHAR(20) NOT NULL |

*Таблица 3.7 — Описание таблицы Fork*

|  |  |
| --- | --- |
| **Имя** | **Тип** |
| ForkID | INTEGER NOT NULL AUTO\_INCREMENT |
| Color | VARCHAR(20) NULL DEFAULT ’black’ |
| ComponentID | INTEGER NULL |

*Таблица 3.8 — Описание таблицы Frame*

|  |  |
| --- | --- |
| **Имя** | **Тип** |
| FrameID | INTEGER NOT NULL AUTO\_INCREMENT |
| Color | VARCHAR(20) NULL DEFAULT ’black’ |

*Таблица 3.9 — Описание таблицы FrameInfo*

|  |  |
| --- | --- |
| **Имя** | **Тип** |
| FrameID | INTEGER NOT NULL |
| Size | INTEGER NOT NULL |
| Stack | INTEGER NOT NULL |
| Reach | INTEGER NOT NULL |

*Таблица 3.10 — Описание таблицы Frameset*

|  |  |
| --- | --- |
| **Имя** | **Тип** |
| FrameSizeID | INTEGER NOT NULL AUTO\_INCREMENT |
| ForkID | INTEGER NOT NULL |
| FramesetID | INTEGER NOT NULL |
| ComponentID | INTEGER NOT NULL |

*Таблица 3.11 — Описание таблицы FrameSize*

|  |  |
| --- | --- |
| **Имя** | **Тип** |
| FrameID | INTEGER NOT NULL |
| Size | INTEGER NOT NULL |
| FrameSizeID | INTEGER NOT NULL AUTO\_INCREMENT |
| ComponentID | INTEGER NOT NULL |

*Таблица 3.12 — Описание таблицы Groupset*

|  |  |
| --- | --- |
| **Имя** | **Тип** |
| Ratio | INTEGER NOT NULL |
| BrakeID | CHAR(18) NOT NULL |
| GroupsetID | INTEGER NOT NULL AUTO\_INCREMENT |
| Color | VARCHAR(20) NOT NULL |
| ComponentID | INTEGER NOT NULL |

*Таблица 3.13 — Описание таблицы Setup*

|  |  |
| --- | --- |
| **Имя** | **Тип** |
| GroupsetID | INTEGER NOT NULL |
| WheelsetID | INTEGER NOT NULL |
| FramesetID | INTEGER NOT NULL |
| BarID | INTEGER NOT NULL |
| CycleTypeID | INTEGER NOT NULL |
| SetupID | INTEGER NOT NULL AUTO\_INCREMENT |

*Таблица 3.14 — Описание таблицы Wheelset*

|  |  |
| --- | --- |
| **Имя** | **Тип** |
| DiametrInch | INTEGER NOT NULL |
| Width | INTEGER NOT NULL |
| SpokesCount | INTEGER NULL |
| WheelsetID | INTEGER NOT NULL AUTO\_INCREMENT |
| Color | VARCHAR(20) NOT NULL |
| ComponentID | INTEGER NOT NULL |

*Таблица 3.15 — Описание таблицы Log*

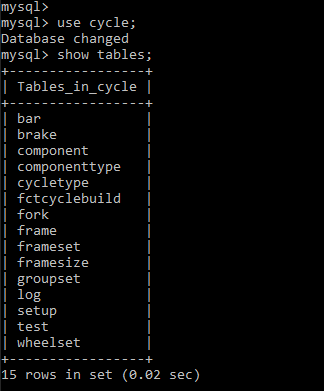
|  |  |
| --- | --- |
| **Имя** | **Тип** |
| ID | INTEGER NOT NULL AUTO\_INCREMENT |
| msg | VARCHAR100 NOT NULL |
| row\_id | INTEGER NOT NULL |

**Использование MySQL CLI Client**

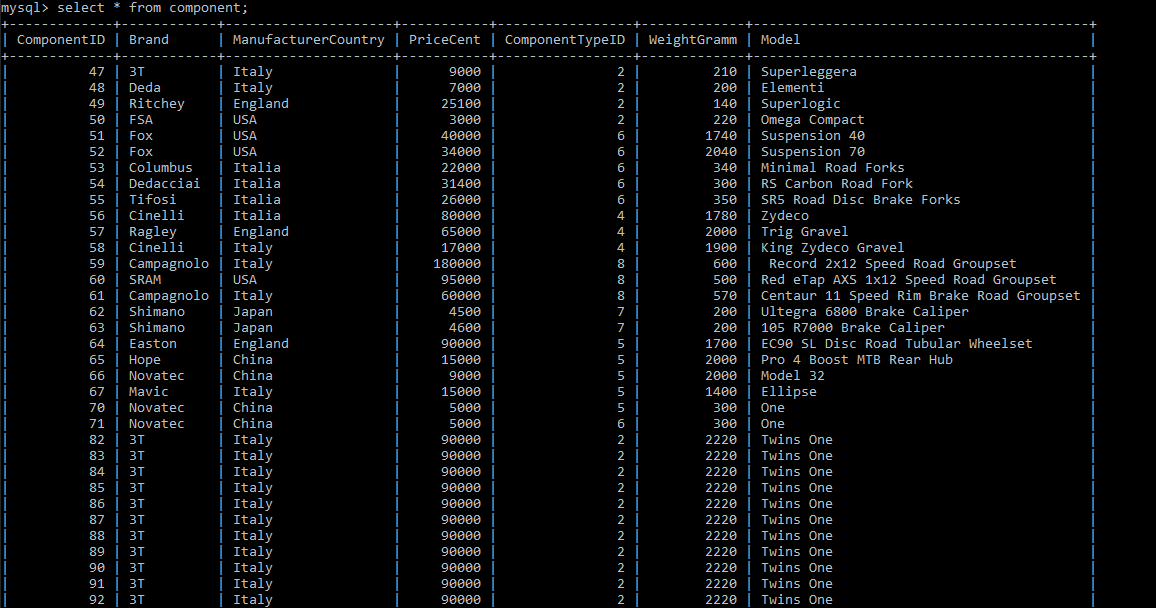
Создадим данные таблицы с помощью MySQL CLI Client — клиента, предоставляющего доступ к СУБД через интерфейс командой строки, исполь­ зовав ключевое слово CREATE TABLE. После создания таблиц выполним ко­ манду SHOW TABLES, выбрав базу данных cycle (см. Рисунок [3.9](#_bookmark34)).

Выведем записи из таблицы Component (см. Рисунок [3.10](#_bookmark35)).

Выполним более сложные запросы выборки — объединим таб­ лицы с помощью оператора INNER JOIN Component и fork по полю

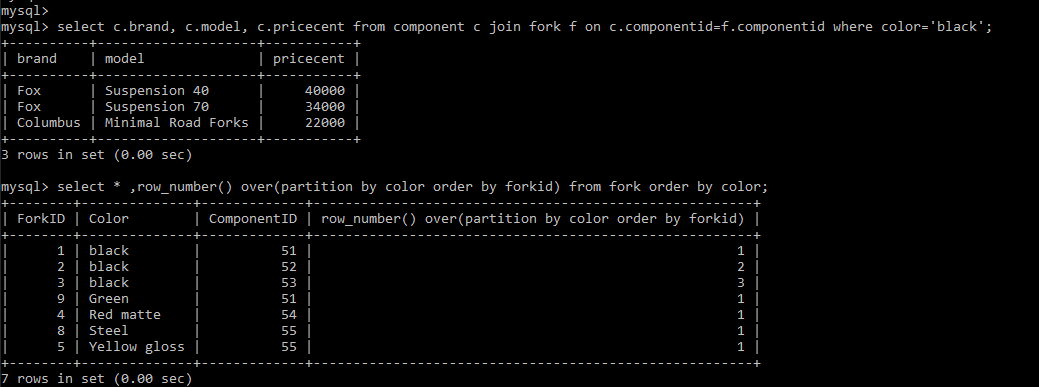


**Рисунок 3.9 — таблицы базы данных** cycle



**Рисунок 3.10 — Вывод записей из таблицы Components**

component\_id, а также воспользуемся функцией ROW\_NUMBER() в сочетании с оконной функцией OVER(), пронумеровав компоненты из таблицы Fork од­ ного цвета по возрастанию значения поля color (см. Рисунок [3.11](#_bookmark36)).



**Рисунок 3.11 — Сложные запросы выборки. База данных cycle**

Создадим триггеры delete\_component и add\_component на добавление и удаление записи в таблице Component. При срабатываении данного триггера, будет добавляться запись в таблицу log, информирующая о совершении мани­ пуляций с данными (см. Листинг [3.1](#_bookmark37)). Приведем объявление данного триггера на диалекте MySQL:

*Листинг 3.1 — Триггер*

DELIMITER $$

drop trigger delete\_component ; $$

create trigger ‘ delete\_component ‘ after delete on component for each row begin

insert into log ( msg , row\_id ) values ( concat (’ delete component ’, old . Brand ,’ ’, old . Model ), old . Component ID );

end ; $$

create trigger ‘ add\_component ‘ after insert on component for each row begin

insert into log ( msg , row\_id ) values ( concat (’ insert component ’, new . Brand ,’ ’, new . Model ), new . Component ID );

end ; $$ DELIMITER $$

1

2

3

4

5

6

7

8

9

10

11

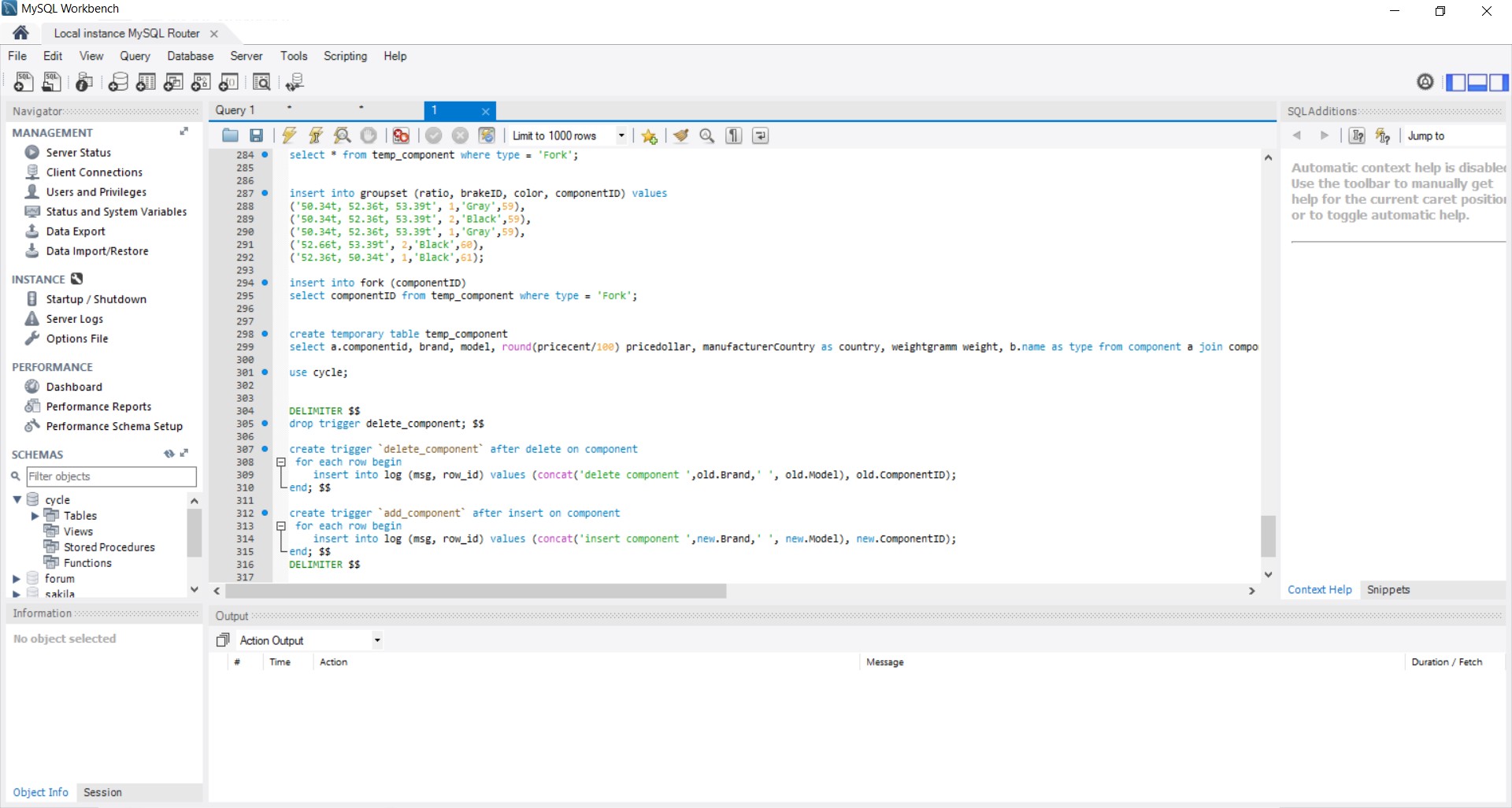
12

13

**Использование MySQL Workbench**

Выполним операции по изменению и просмотру данных, занесенных в базу данных cycle, с помощью инструмента для визуального проектирования баз данных MySQL Workbench.

Для этого подключимся к локально развернутому на машине MySQL Server, указав порт, имя пользователя и пароль. После этого мы получим до­ ступ к пользовательскому интерфейсу программы, представляющему собой две области — область выполнения запросов и область отображения результа­ тов (см. Рисунок [3.12](#_bookmark39)).

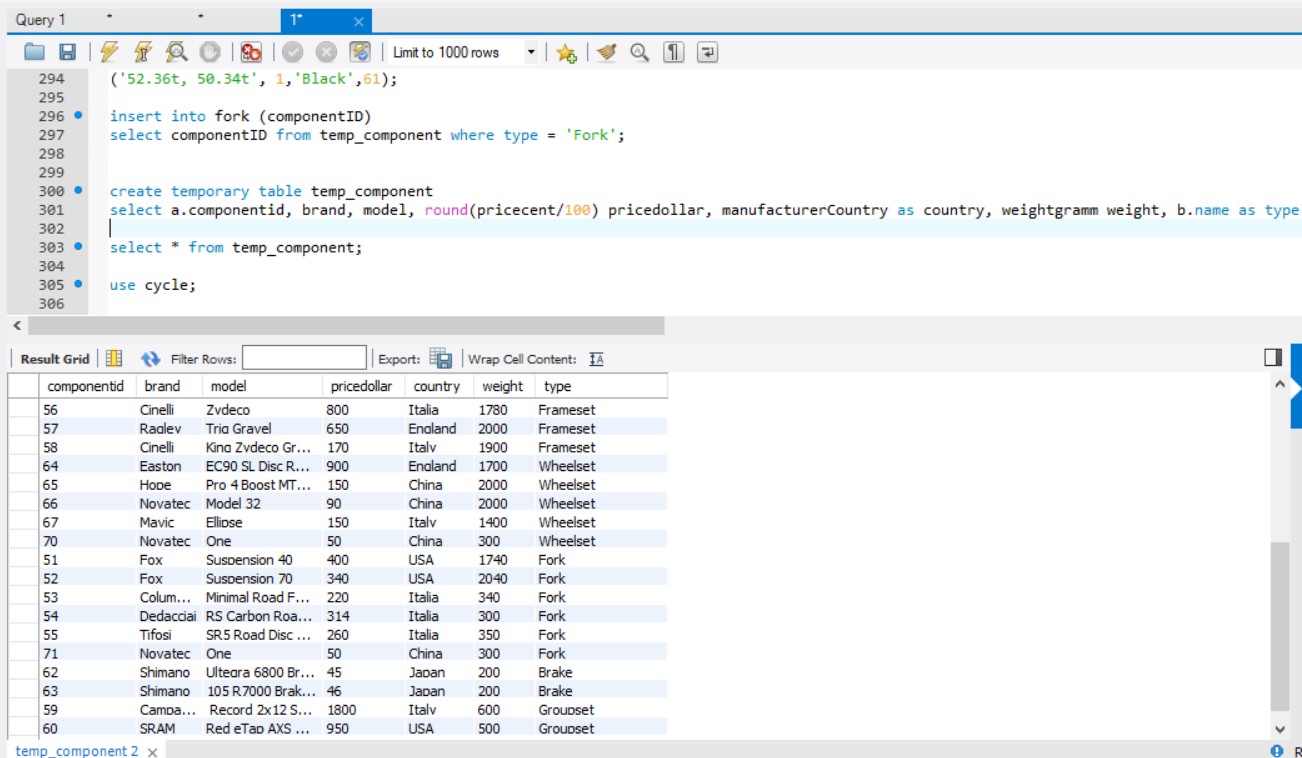


**Рисунок 3.12 — Интерфейс MySQL Workbench**

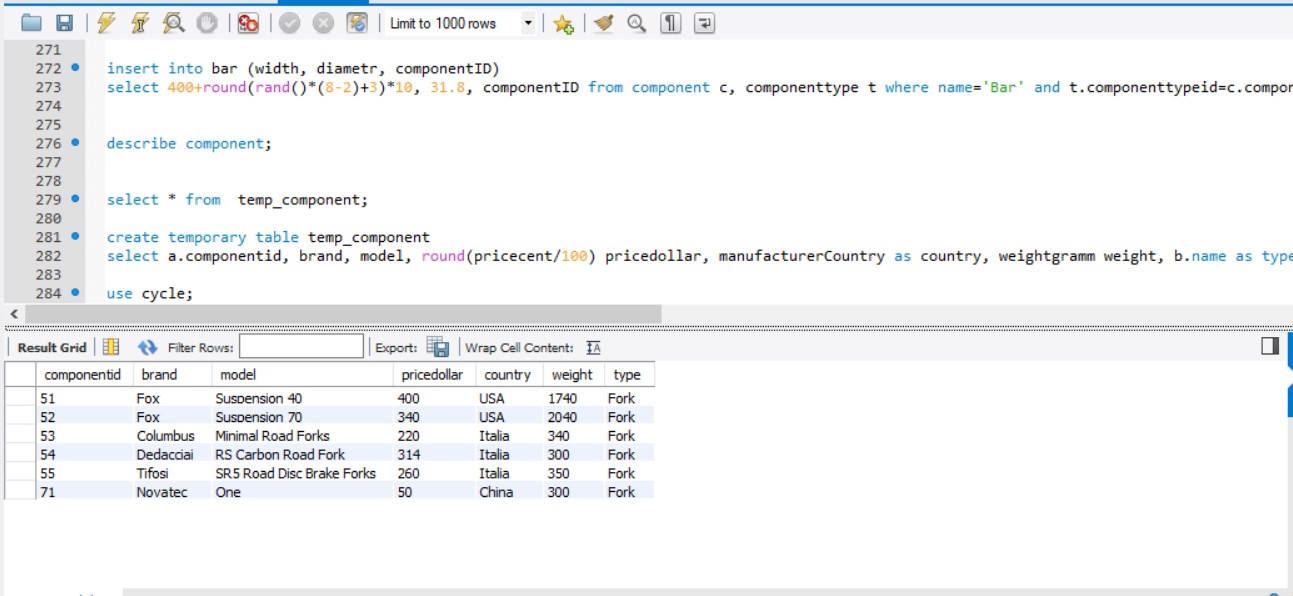
Создадим временную таблицу, в котором объединим сведения о компо­ нентах (см. Рисунок [3.13](#_bookmark40)).

Выполним еще один запрос — ограничим вывод только компонентами типа «Fork» (см. Рисунок [3.14](#_bookmark41)).

Данный визуальный инструмент позволяет удобно организовать работу с базой данных, сохранять SQL–скрипты, параметры подключения и настройки.



**Рисунок 3.13 — Просмотр временной таблицы компонентов**



**Рисунок 3.14 — Просмотр временной таблицы компонентов. Компоненты «Fork»**

# Создание веб–клиента

В данном разделе будет рассмотрен процесс реализации веб–приложе­ ния, с помощью которого будет производится взаимодействие конечного поль­ зователя с созданной в рамках предыдущих работ базой данных.

**Общие требования к приложению**

Приведем технические требования, предъявляемые к разрабатываемому приложению.

**Требования к функционалу**

К функционалу приложения предъявляются следующие требования:

* регистрация и авторизация пользователя в системе;
* добавление, изменение, удаление, обновление информации по теме;
* фильтрация списков по соответствующим признакам;
* просмотр информации по запросу.

**Требования к интерфейсу**

Интерфейс системы должен поддерживать русский язык.

Интерфейс системы должен быть спроектирован с учетом ролевой мо­ дели и уровней доступа пользователей.

Интерфейс системы должен обеспечивать наглядное, интуитивно понят­ ное представление структуры размещенной информации, быстрый и логич­ ный переход к соответствующим разделам.

Навигационные элементы интерфейса должны обеспечивать понимание пользователем их смысла и обеспечивать навигацию по всем доступным поль­ зователю разделам и отображать соответствующую информацию.

Интерфейс системы должен позволять решать задачи пользователя наи­ более быстрым, простым и удобным из возможных способов.

Дизайн и удобство интерфейса должны быть на уровне ожиданий со­ временного пользователя и восприниматься им как комфортная, удобная и приятная рабочая среда.

**Реализация приложения**

На основании функциональных требований, указанных в задании, бы­ ло реализовано веб приложение, позволяющие взаимодействовать с данными, хранящимися в базе данных cycle, созданной в ходе выполнения предыдущих практических работ данного курса.

Интерфейс системы спроектирован с учетом ролевой модели и уровней доступа пользователей и подразумевает регистрацию пользователя в СУБД MySQL.

Интерфейс поддерживает русский язык и позволяет пользователю удоб­ но просматривать сведения, содержащиеся в таблицах выбранной базы дан­ ных.

Акцент в приложении сделан на содержании, дизайн интерфейса прост и лаконичен, чтобы облегчить и упростить взаимодействие пользователя с системой.

**Используемые технологии**

В качестве набора программных технологий был выбран LAMP–стэк, объединяющий в себе следующие компоненты:

* Linux;
* Apache;
* MySQL;
* PHP.

Программно-аппаратная часть сервиса (бэкенд) реализована на языке PHP в парадигме объектно-ориентированного программирования, а также с использованием языка MySQL для работы с базой данных.

Структура гипертекстового документа описана средствами HTML и

CSS.

Клиентская сторона пользовательского интерфейса системы (фронтэнд)

основана на подходе AJAX (Asynchronous Javascript and XML). В качестве языка программирования был использован Java Script.

Запросы веб-сервиса обрабатываются сервером Apache.

**Возможные сценария использования веб-приложения**

Реализуемое веб–приложение может быть использовано для просмотра таблиц, добавления данных в таблицы базы данных под управлением СУБД MySQL, а также для написания произвольных DML и DDL запросов на диа­ лекте MySQL.

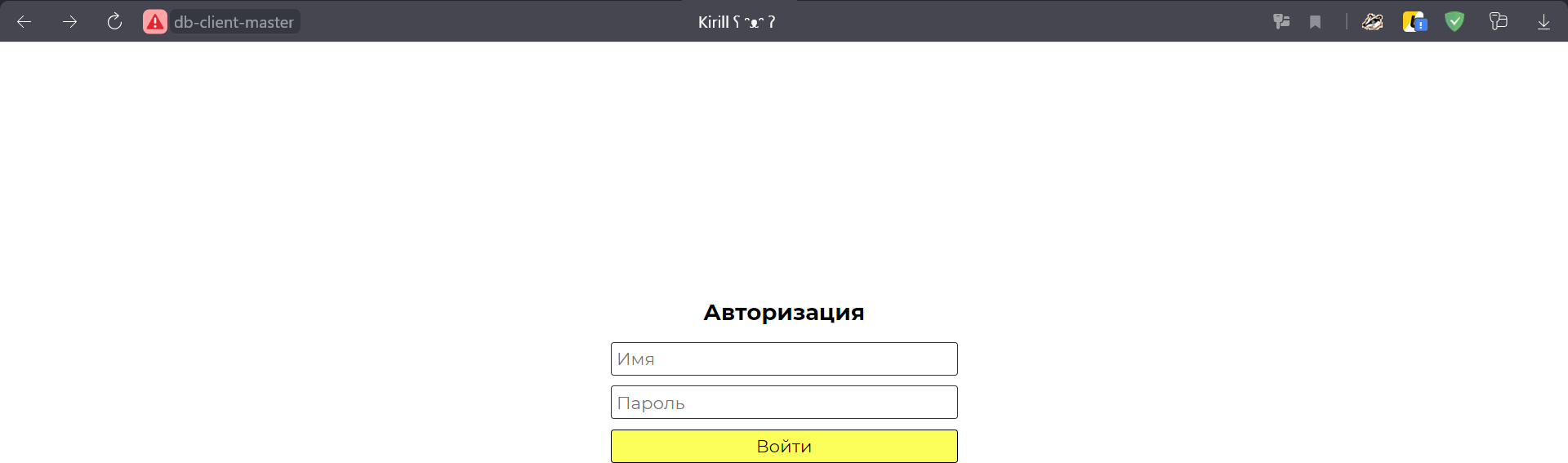
Веб–приложение объединяет в себе часть возможностей командой стро­ ки СУБД MySQL и веб-клиента для просмотра таблиц.

**Демонстрация работы веб-приложения**

Продемонстрируем сценарий использования веб–приложения, созданно­ го в рамках данной практической работы.

**Авторизация**

При первом входе на сайт пользователю необходимо авторизовать­ ся — выбрать пользователя СУБД MySQL, который имеет права на изменение и просмотр данных в выбранной базе данных. Форма авторизации приведена на Рисунке [4.1](#_bookmark51).

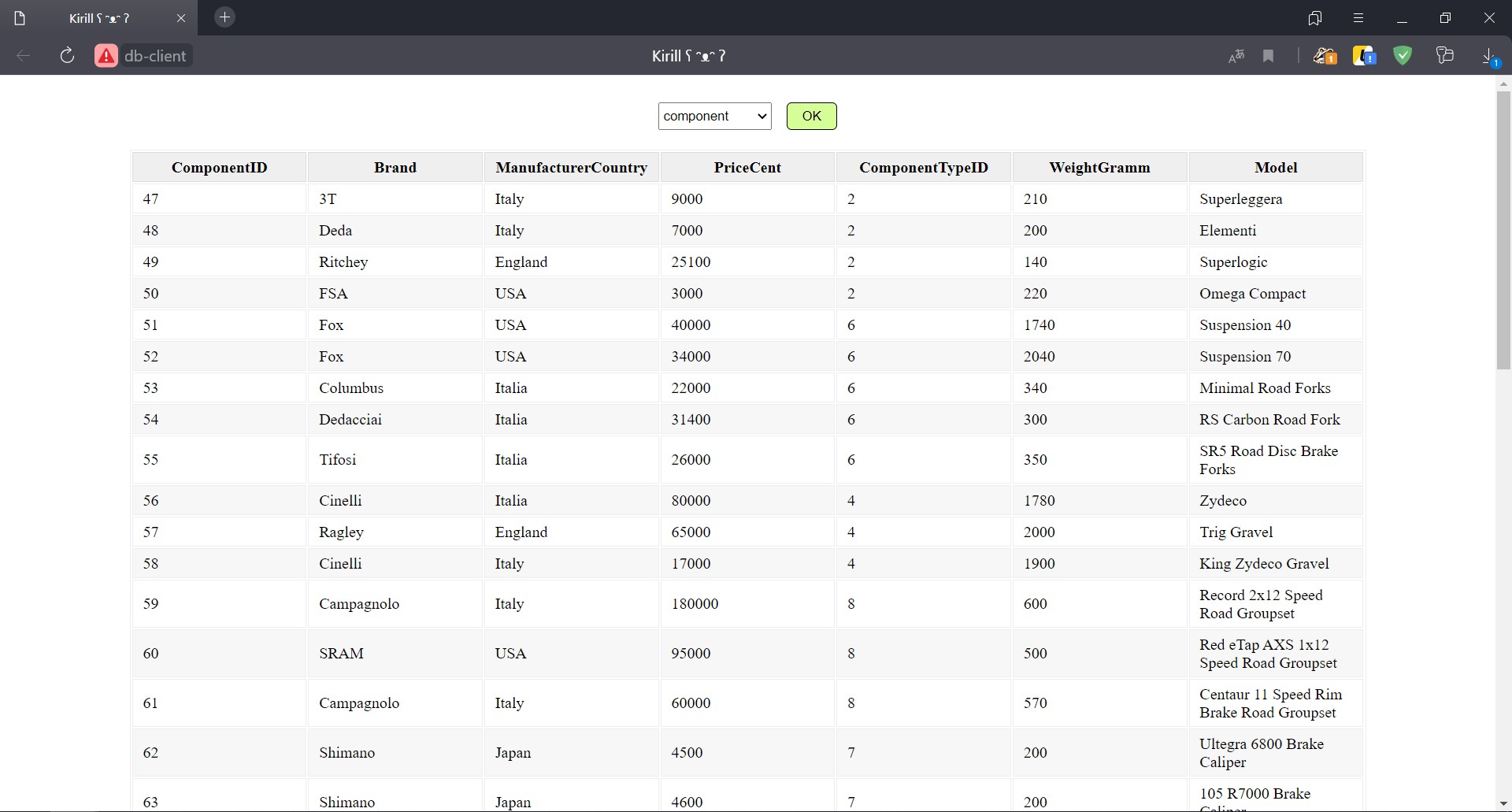


**Рисунок 4.1 — Авторизация пользователя**

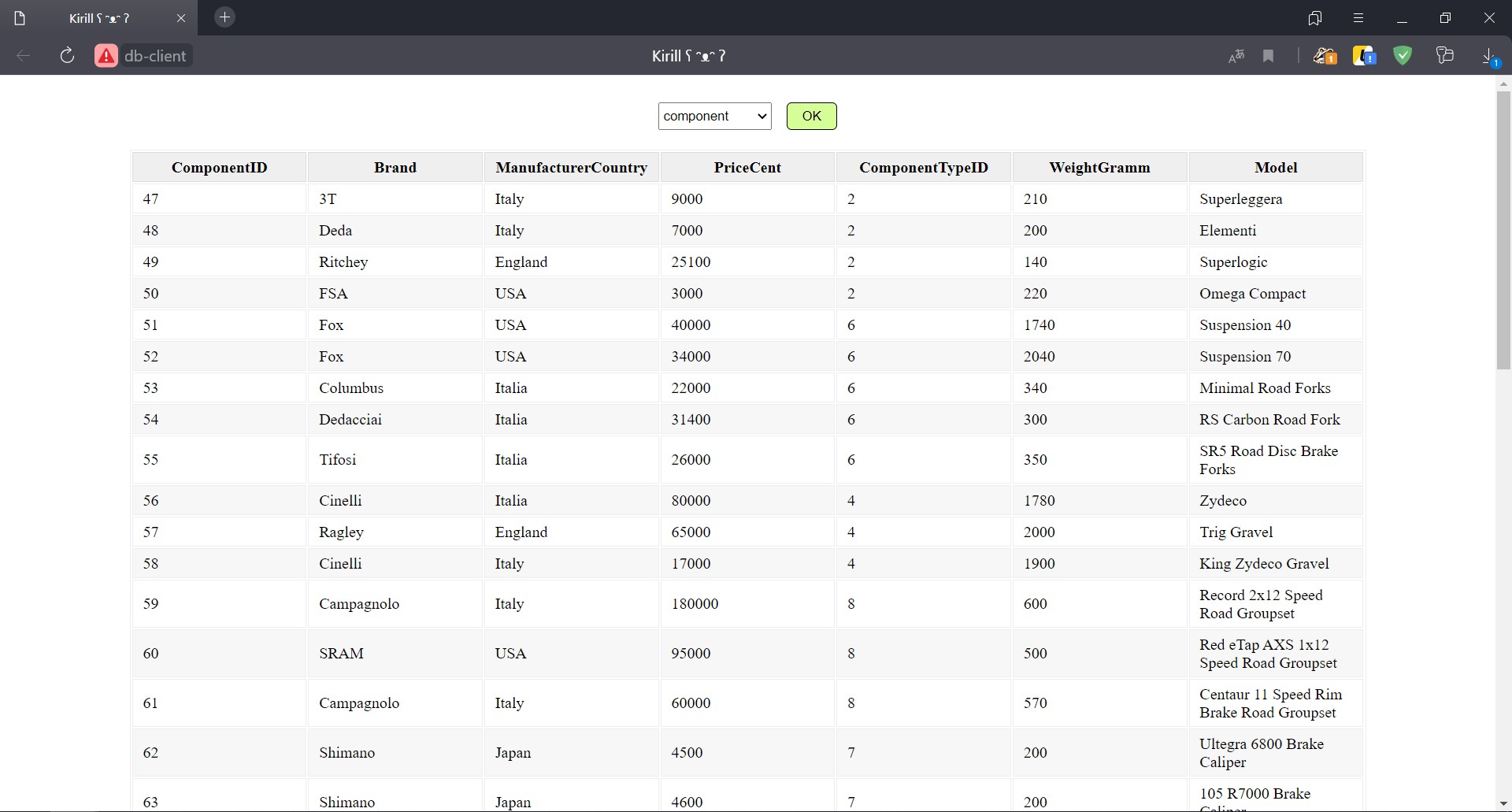
В данном примере будет выбран пользователь denilai, который имеет привилегии супер-пользователя в данной базе данных и может вносить из­ менения в базу даннных, просматривать все записи в ней, создавать новые постоянные и временные таблицы, процедуры и макросы.

**Отображение записей таблиц**

После успешной авторизации пользователю будет предложено выбрать таблицу в базе данных, записи в которой будут отображены на экране. По­ ля таблиц могут содержать кириллические символы — они будут корректно отображаться веб-приложением (см. Рисунки [4.2](#_bookmark53), [4.3](#_bookmark54)).



**Рисунок 4.2 — Отображение записей таблицы Component**

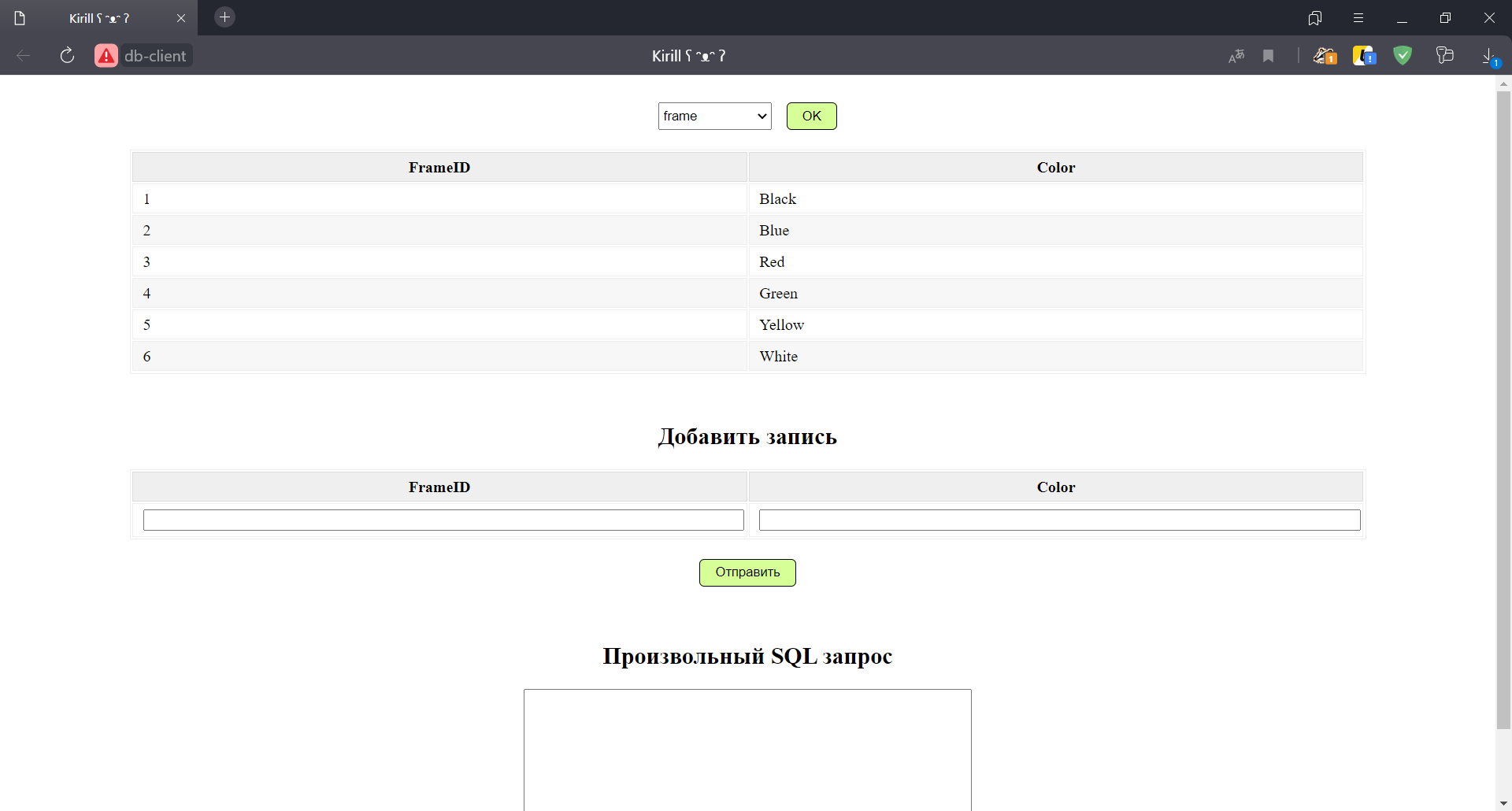


**Рисунок 4.3 — Отображение записей таблицы Log**

**Добавление записей в в таблицы**

Ниже, под выведенными записями таблицы расположены поля для до­ бавления значений в выбранную базу данных. В случае ввода корректных

данных, после нажатия кнопки «OK», поля будут немедленно добавлены в таблицу и отобразятся на экране (см. Рисунок [4.4](#_bookmark56)).



**Рисунок 4.4 — Добавление записей в таблицу Frame**

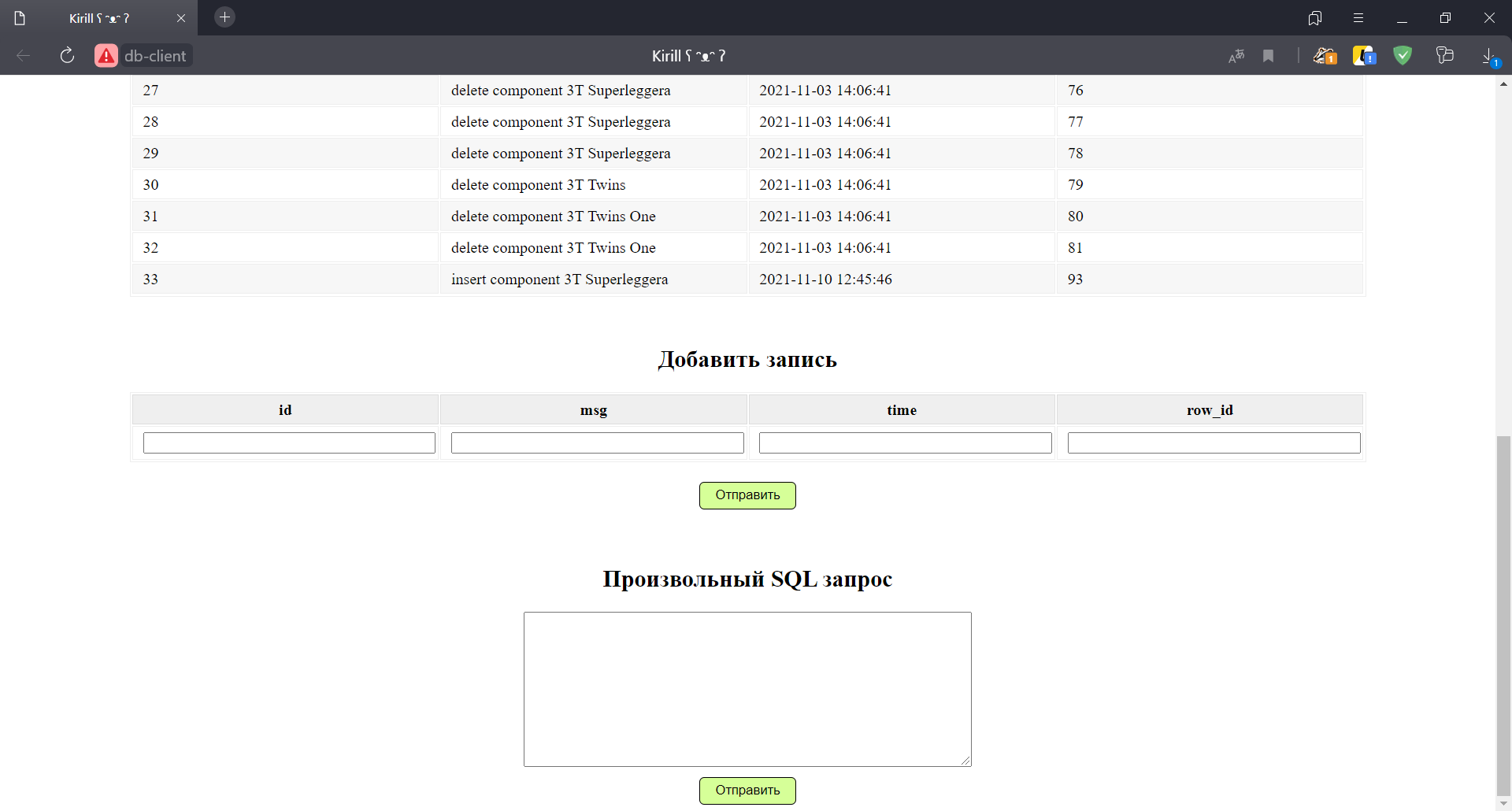
**Выполнение произвольных запросов**

В самом низу веб–страницы расположено текстовое поле для выполне­ ния SQL–запросов. Запросы могут относиться не только к выбранной таблице или базе данных, а к любой таблице, права на взаимодействие с которой имеет выбранный пользователь.

Текст, введенный пользователем, передается в виде строки на исполне­ ние MySQL серверу. В случае ввода корректного запроса, он будет выполнен, о чем будет сообщено пользователю.

С помощью данного поля также можно производить операции по уда­ лению и вставке записей в таблицы, отображаемые на экране (см. Рисунок [4.5](#_bookmark58)).

С помощью данного веб–клиента пользователь может просматривать за­ писи, хранящиеся в базе данных, отмечать продвижение деталей по произ­ водственным цехам, отражать текущий статус сборки велосипеда, проверять



**Рисунок 4.5 — Выполнение произвольных запросов**

наличие тех или иных компонентов на складе, получать полную информацию о них.

# ЗАКЛЮЧЕНИЕ

В ходе выполнения данных практических работ были получены знания по написанию запросов на диалекте MySQL, отработаны навыки работы в средах ER Assistant и ERwin Data Modeler, навыки работы с интерфейсом командной строки MySQL CLI Client, а также с инструментом для визуального проектирования баз данных MySQL Workbench.

Полученные знания были применены на практике для построения про­ ектов баз данных, физические и логические модели баз данных. В послед­ ствии данные базы данных были реализованы в СУБД MySQL. Также было создано пользовательское веб – приложение, представляющее удобный интер­ фейс для взаимодействия с информационной системой и базой данных.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Converse, Tim*. PHP5 and MySQL bible / Tim Converse, Joyce Park, Clark Morgan. — John Wiley & Sons, 2004. — Т. 147.
2. *Балдин, Евгений Михайлович*. Компьютерная типография LaTeX / Ев­ гений Михайлович Балдин. — Evgeny Baldin, 2008.
3. *Дюбуа, Поль*. MySQL: Полн. и исчерпывающее руководство по применению и администрированию баз данных MySQL 4, а также программированию приложений / Поль Дюбуа. — Издательский дом Вильямс, 2004.
4. *Юртанова, Екатерина Михайловна*. Разработка Web-приложений с использованием языка PHP / Екатерина Михайловна Юртанова // *Учебный эксперимент в образовании*.