Министерство образования и науки РФ

Пермский национальный исследовательский политехнический университет

Электротехнический факультет

Кафедра Информационные технологии и автоматизированные системы

Курсовая работа по дисциплине

«Базы данных»

Вариант 6

Тема: «Разработка реляционной базы данных лодочная станция»

Выполнил: студент группы Ивт-22-2б

Ильина К.Н.

Проверил: ассистент кафедры ИТАС

Петренко А.А.

г. Пермь – 2024

Реферат

Отчет 24 с., 8 рис., 2 источн., 3 прил.

Ключевые слова: реляционная база данных, лодочная станция, предметная область, модель базы данных, система управления базами данных, приложение, разграничение прав доступа.

Цель работы: разработать реляционную базу данных «Лодочная станция», а также приложение к ней.

Задачи работы:

1. Провести анализ предметной области;
2. Построить 3 модели базы данных: концептуальную, логическую и физическую;
3. Разработать базу данных на основе физической модели;
4. Разработать приложение для работы с базой данных.

Содержание

[Термины и сокращения 4](#_wvrfo7kpooga)

[Введение 5](#_mveky4uyzjzi)

[1 Анализ предметной области 6](#_4aamq19cp3t6)

[1.1 Постановка задачи 6](#_mw2addimxg4y)

[1.2 Концептуальная модель базы данных 7](#_u2lfzsapfp5w)

[2 Моделирование базы данных 11](#_oekwc849r4p2)

[2.1 Логическая модель базы данных 11](#_by90rwpana3y)

[2.2 Физическая модель базы данных 12](#_po05u7paqn5p)

[3 Реализация базы данных 14](#_sza62unintww)

[3.1 Создание базы данных и таблиц 14](#_unqssjrg0c5g)

[3.2 Создание пользователей, настройка привилегий 18](#_rhwkugjzmm6c)

[4 Тестирование работы системы 20](#_meozr35zxaq4)

[Заключение 24](#_cjpl67azkghj)

[Список литературы 25](#_5p772svknavb)

# Термины и сокращения

Атрибут – это столбец таблицы.

База данных (БД) – это упорядоченный набор структурированной информации или данных.

Ограничения ссылочной целостности – это ограничение в работе с данными, при котором соблюдается соответствие каждого значения внешнего ключа со значением первичного ключа.

Реляционная база данных – это форма хранения и упорядочения информации в табличной форме.

Система управления базами данных (СУБД) – это совокупность программных средств, обеспечивающих работу с базами данных.

Сущности – это отношения в реляционной модели данных.

# Введение

В мире отдыха на воде, эффективное управление лодочной станцией играет ключевую роль в обеспечении безопасности и комфорта клиентов. Для этого необходима система учета и контроля, которая бы эффективно отслеживала имеющиеся плавсредства, их техническое состояние, а также информацию о клиентах и водоемах.

Целью данной курсовой работе рассматривается проектирование и реализация реляционной базы данных для лодочной станции, а также разработка системы ввода, просмотра и корректировки данных в базе.

Для достижения цели необходимо выполнить следующие задачи:

1. Изучение требований и особенностей управления лодочной станцией, определение основных сущностей и их атрибутов.
2. Создание структуры реляционной базы данных с учетом всех выявленных сущностей и связей между ними.
3. Выбор технологий и инструментов для реализации базы данных и приложения, обеспечивающего взаимодействие с ней.
4. Создание интерфейсов для управления данными в базе, включая формы ввода информации, отображение данных и возможности их редактирования.
5. Тестирование разработанной системы на соответствие требованиям, выявление и устранение возможных ошибок и недочетов.

В результате выполнения данных задач будет создана функциональная и эффективная система управления лодочной станцией, способная обеспечить надежный учет и безопасность использования плавсредств, а также оптимизировать процессы работы персонала и обслуживания клиентов.

# 1 Анализ предметной области

## 1.1 Постановка задачи

Постройте реляционную базу данных Лодочная станция. Разработайте систему ввода, просмотра и корректировки данных базы. Разработайте систему решения задач на основе информации базы.

Лодочная станция ведет учет имеющихся и выданных отдыхающим плавсредств: катеров, яхт, лодок, катамаранов, байдарок, водных лыж и т.п. Кроме того, проводится учет технического состояния средств, их текущий ремонт, списание и заказ новых. Создаваемая база данных должна предоставить информацию для решения перечисленных задач.

Клиент может использовать только одно плавсредство в пределах установленного срока. Выдача плавсредства оформляется только на одно физическое лицо. Плавсредства могут использоваться в одном из имеющихся и сообщающихся между собой водоемов: реке, озере или пруду. Перед плаванием клиент должен получить инструктаж одного из сотрудников станции.

О плавсредствах известно: название, номер, состояние, тип, наличие и список спасательных средств.

О клиентах известно: фамилия, адрес, паспортные данные, выданное плавсредство, водоем, где предполагается его использование, время выдачи и длительность использования.

О водоемах известно: название, опасность плавания для каждого типа плавсредств.

Об инструктаже известно: тип инструктажа, дата и время проведения, фамилия и должность проводившего инструктаж.

База данных должна позволять получать ответы на запросы:

1. Вывести список заказов и ответственных за них сотрудников
2. Вывести выданные плавсредства, срок использования которых заканчивается сегодня
3. Вывести плавсредства, у которых количество спасательного оборудования больше 3
4. Вывести выданные плавсредства, их пользователей и их техническое состояние
5. Вывести номер выданного плавсредства, его пользователя и водоем
6. Показать водоемы, где выдаются плавательные средства, вместе с их уровнем опасности:
7. Отобразить все свободные плавательные средства вместе с их статусом доступности и типом плавательного средства:
8. Показать плавательные средства, находящиеся в настоящее время на ремонте
9. Показать персонал, ответственный за каждое выданное плавательное средство
10. Вывести список всех заказов, сделанных определенным сотрудником

Определим границы предметной области. Лодочная станция выдает отдыхающему одно из имеющихся плавсредств в пределах одного срока использования. Нельзя использовать средство, которое находится сейчас в ремонте. Перед получением плавсредства клиент должен пройти инструктаж техники безопасности. Обязателен учет технического состояния каждого плавсредства, так же как и его тип. В зависимости от типа и водоема, выбранного клиентом, записывается уровень опасности использования данного плавстведсва на воде. Также производятся закупки новых плавсредств, проводимые персоналом.

## 1.2 Концептуальная модель базы данных

Для анализа предметной области составим концептуальную модель. Для начала определим информацию, которую нам необходимо хранить и обозначим ее как сущности:

1. Сущность вид состояния. Нужна для перечня технических состояний, которые могут быть присвоены плавательным средствам.
2. Сущность техническое состояние. Обозначает состояние каждого плавательного средства.
3. Сущность вид плав. средства. Обозначает возможные виды плавательных средств, которые есть на лодочной станции, а также используется для заказа нужных приспособлений
4. Сущность заказ. Содержит информацию о всех заказах, которые персонал станции делает.
5. Сущность ремонт. Несет историю ремонта всех плавательных средств станции.
6. Сущность плав. средство. Содержит перечень всех устройств, имеющихся на станции.
7. Сущность персонал. Перечислены все работники станции и их должности
8. Сущность клиент. Содержит информацию о всех клиентах, пользовавшихся услугами лодочной станции.
9. Сущность выдача плав. средств. Обозначает какие плавсредства в данный момент выданы каким клиентам.
10. Сущность водоем. Содержит перечень водоемов, на которых можно использовать плавательные средства станции.
11. Сущность опасность водоема. Служит для перечисления опасности для каждого типа плавательных средств станции на водоемах.
12. Сущность инструктаж. Перечисляет инструктажи, проводимые станцией.

Рассмотрим возможные связи между сущностями:

1. Каждый вид технического состояния может относится к нескольким тех. состояниям.
2. У каждого плавсредства обязательно указано его техническое состояние.
3. Плавсредство могло стоять на ремонте.
4. Плавсредство используется для выдачи плавсредств.
5. Клиенту выдается плавательное средство.
6. Для выдачи плавсредства обязателен инструктаж.
7. Каждое плавсредство относится к определенному виду
8. Для заказа плавсредств используется их вид.
9. Каждый заказ делает один из работников.
10. Персонал проводит инструктаж для выдачи плавсредства.
11. Водоем указывается при выдачи плавсредства.
12. Для каждого водоема обозначена опасность.
13. Опасность для водоема указана для каждого вида плавсредств.

На основе

На основе указанных сущностей и их связей построим ER-диаграмму в нотации Питера Чена(рис.1).

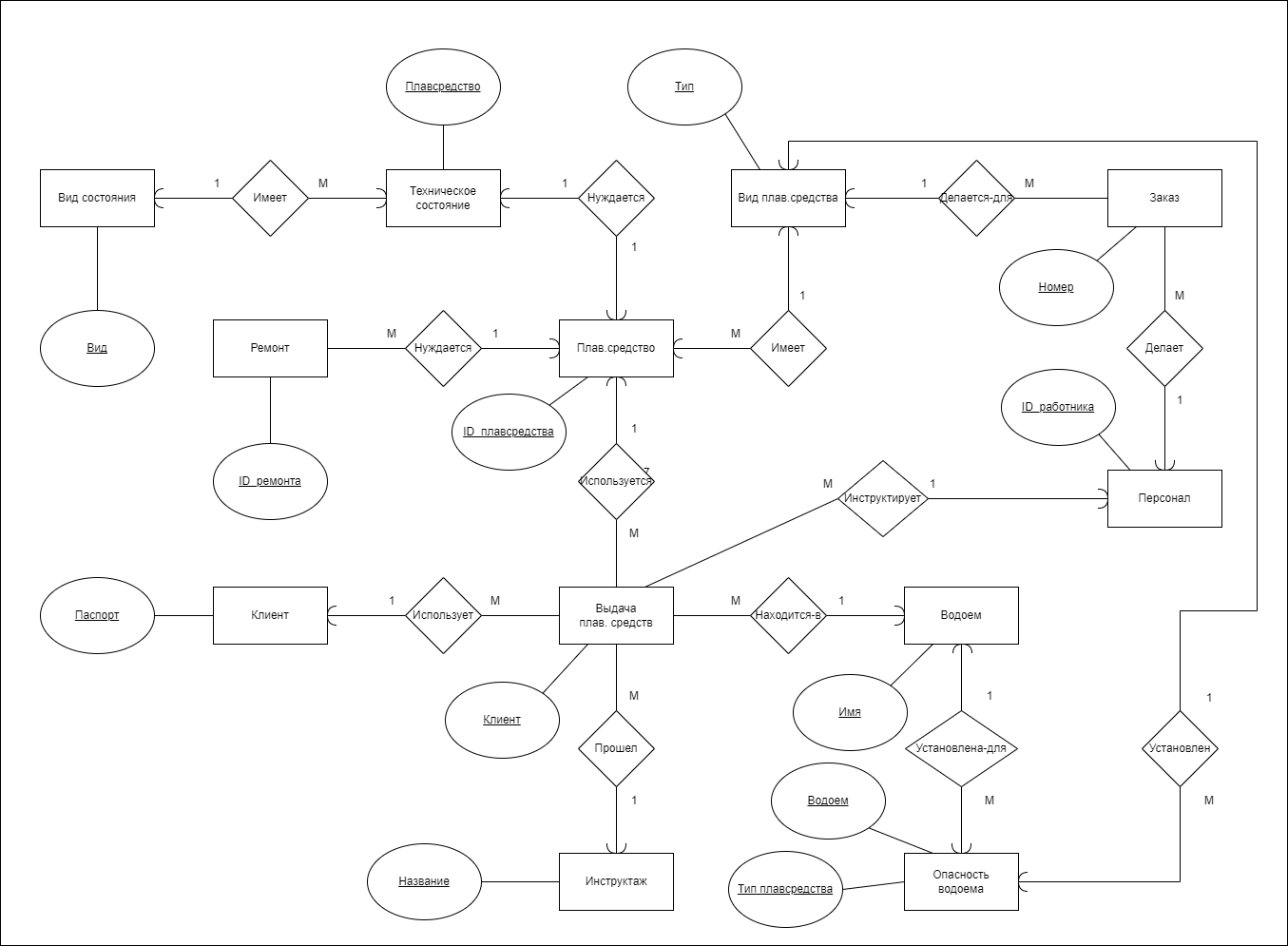


Рисунок 1 - Концептуальная модель базы данных

Основываясь на постановке задачи создана концептуальная модель для базы данных, обозначающая сущности с идентификаторами и связи между ними. Использованы связи: один-к-одному и многие-ко-многим. Также указано на диаграмме обязательна ли связь.

# 2 Моделирование базы данных

## 2.1 Логическая модель базы данных

Построим логическую модель на основе концептуальной, создается логическая модель данных, которая представляет собой абстрактное представление структуры базы данных без привязки к конкретной СУБД. Добавим все атрибуты сущностей и изменим нотацию на “Вороньи лапки”(рис.2). На модели показаны все атрибуты сущностей, первичные ключи обозначены подчеркиванием, одинаковые атрибуты свидетельствуют о связи между сущностями.

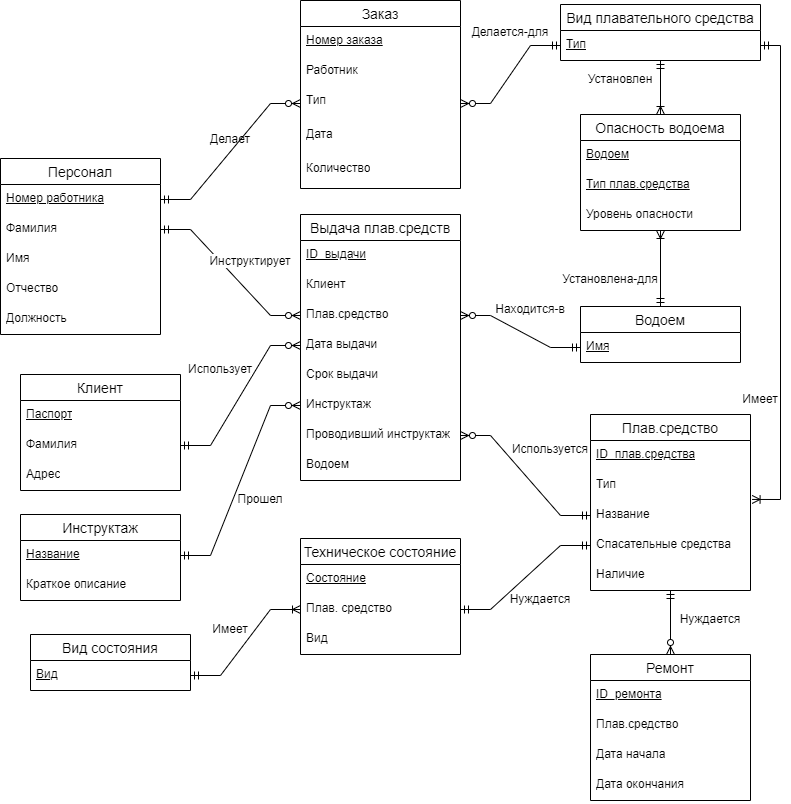


Рисунок 2 - Логическая модель базы данных

## 2.2 Физическая модель базы данных

Для реализации базы данных была выбрана СУБД MySQL, учитывая доступные ресурсы, а пользовательское приложение будет разработано посредством PHP, HTML и CSS. Также была построена физическая модель базы данных, основываясь на выбранной СУБД(рис.3). Процесс преобразования включает в себя нормализацию данных и определение индексов и ограничений целостности для базы данных. Например в таблице “Клиент” атрибут “адрес” был разбит на два других: “street” и “house”.

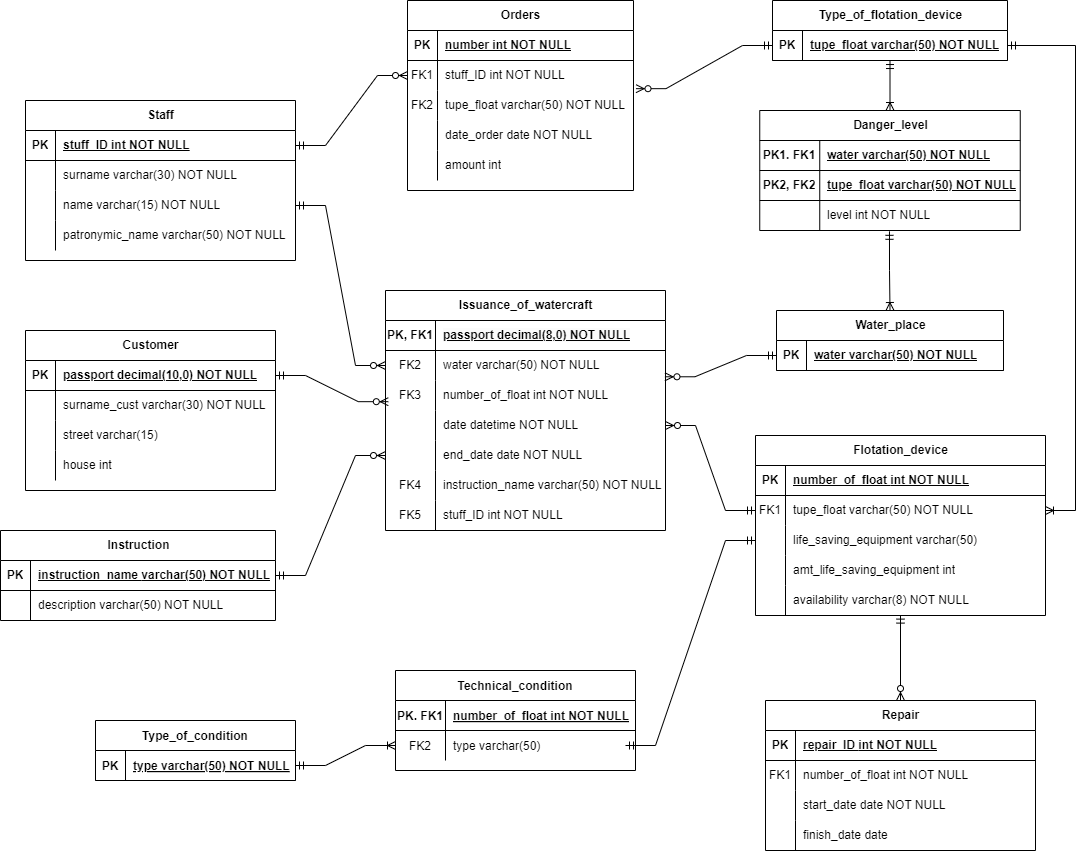


Рисунок 3 - Физическая модель базы данных

Получены логическая модель, в которой показаны ограничения данных, посредством связи сущностей, и физическая модель, которая показывает модель реализации базы данных в MySQL.

# 3 Реализация базы данных

## 3.1 Создание базы данных и таблиц

Создадим базу данных, основываясь на физической модели. Начнем с создания пустой базы данных в выбранной СУБД. Для этого можно воспользоваться SQL-скриптами, написав SQL запрос:

CREATE DATABASE BoatStation;

После создания базы данных следует определить структуру таблиц и их связи, основываясь на логической модели данных. Для этого также используются SQL-скрипты.

Для создания таблиц использовался запрос CREATE TABLE, например для таблицы “Customer”:

CREATE TABLE `Customer` (

`passport` decimal(8,0) NOT NULL,

`surname\_cust` varchar(30) NOT NULL,

`address` varchar(50) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

помимо самого запроса инициализируются используемой для данной таблицы подсистемы хранения и кодировку. Используется подсистема InnoDB, как наиболее широко используемый движок с поддержкой транзакций. Кодировка используется utf8mb4, это важно для корректного хранения и сортировки текстовых данных.

Таким образом была прописана каждая сущность, а также все ее атрибуты, базы данных в виде таблиц. Затем в каждой таблице прописывались индексы для первичных и внешних ключей, используя запрос:

ALTER TABLE `Danger\_level`

ADD PRIMARY KEY (`danger\_ID`),

ADD KEY `water` (`water`),

ADD KEY `tupe\_float` (`tupe\_float`);

так:

1. ALTER TABLE указывает на таблицу, для которой мы создаем индексы.
2. ADD PRIMARY KEY создает первичный ключ для указанного столбца.
3. Строка ADD KEY создает индекс, используемый на указанные в скобках столбцы.

Затем создавались ограничения для внешних ключей:

ALTER TABLE `Danger\_level`

ADD CONSTRAINT `danger\_level\_ibfk\_1` FOREIGN KEY (`water`) REFERENCES `Water\_place` (`water`) ON DELETE RESTRICT ON UPDATE RESTRICT,

ADD CONSTRAINT `danger\_level\_ibfk\_2` FOREIGN KEY (`tupe\_float`) REFERENCES `Type\_of\_flotation\_device` (`tupe\_float`) ON DELETE RESTRICT ON UPDATE RESTRICT;

где:

1. ADD CONSTRAINT создает ограничение с именем.
2. FOREIGN KEY обозначает внешний ключ.
3. REFERENCES указывает таблицу и индекс, на который ссылается внешний ключ.
4. ON DELETE RESTRICT ON UPDATE RESTRICT указывает, что в случае удаления или обновления ссылочных данных информация в таблице не изменится.

Так же, для удобства пользования, в некоторых таблица были созданы автоматические заполняемые столбцы с помощью автоматического заполнения и триггеров.

Автоматическое заполнение происходит с помощью модификации нужного столбца MODIFY и использованием AUTO\_INCREMENT:

ALTER TABLE `Staff`

MODIFY `stuff\_ID` int NOT NULL AUTO\_INCREMENT;

А так же триггер, для обозначения свободно ли сейчас плавсредство, то есть не указано ли оно в таблице Issuance\_of\_watercraft:

DELIMITER $$

CREATE TRIGGER `update\_availability` AFTER INSERT ON `Issuance\_of\_watercraft` FOR EACH ROW BEGIN

DECLARE count\_used INT;

SELECT COUNT(\*) INTO count\_used

FROM Issuance\_of\_watercraft

WHERE number\_of\_float = NEW.number\_of\_float;

IF count\_used > 0 THEN

UPDATE Flotation\_device

SET availability = 'занято'

WHERE number\_of\_float = NEW.number\_of\_float;

ELSE

UPDATE Flotation\_device

SET availability = 'свободно'

WHERE number\_of\_float = NEW.number\_of\_float;

END IF;

END

$$

DELIMITER ;

Этот код создает триггер update\_availability, используя:

1. DELIMITER $$ для изменения разделителя запросов. Используем $$ вместо стандартного “;”, чтобы обозначить конец тела триггера, так же в конце снова указывается для возвращения “;” как разделителя.
2. CREATE TRIGGER создает новый триггер с именем.
3. AFTER указывает когда будет срабатывать триггер, в нашем случае после выполнения запроса.
4. INSERT ON обозначает действие добавления информации в какой таблице будет вызывать триггер.
5. FOR EACH ROW указывает, что триггер будет выполняться для каждой вставленной строки.
6. BEGIN обозначает начало тела триггера.
7. DECLARE count\_used INT объявляет локальную переменную count\_used типа INT, которая будет использоваться для хранения количества использований плавсредства.
8. SELECT COUNT(\*) INTO подсчитывает количество записей в таблице Issuance\_of\_watercraft, которые используют плавсредство с тем же номером, что и новая вставленная запись, и сохраняет результат.
9. IF … THEN ... END IF;: Эта конструкция проверяет, было ли плавсредство уже использовано.
10. UPDATE … SET обновляет данные.

Далее заполним данные в таблицах, используя запрос INSERT INTO … VALUES, например для базы данных

INSERT INTO `Staff` (`surname`, `name`, `patronymic\_name`, `position`) VALUES

('Смирнов', 'Иван', 'Иванович', 'инструктор'),

('Иванов', 'Петр', 'Сергеевич', 'водитель'),

('Петров', 'Алексей', 'Александрович', 'инструктор'),

('Сидоров', 'Дмитрий', 'Иванович', 'техник'),

('Козлова', 'Мария', 'Петровна', 'инструктор');

Хотя таблица Staff содержит 5 атрибутов, один из них задается автоматически, так что мы задаем в скобках после названия таблицы столбцы, в которые будем добавлять информацию и после ключевого слова VALUES указываев через запятую в скобках информацию в том же порядке, в котором указаны столбцы таблицы.

## 3.2 Создание пользователей, настройка привилегий

Для обеспечения безопасного пользования базой данных создадим несколько пользователей и, при необходимости, ограничим действия, для безопасной работы базы данных.

Создадим 4 пользователей:

1. Администратор базы данных:
   1. Имя пользователя: admin
   2. Пароль: S\*dWwqw9\_Qw0V8Ux
   3. Привилегии:
      1. Полный доступ для всех таблиц базы данных
      2. GRANT OPTION
2. Менеджер лодочной станции:
   1. Имя пользователя: manager
   2. Пароль: IZDz9RMu8LZJD!ru
   3. Привилегии:
      1. SELECT, INSERT, UPDATE, DELETE, CREATE TEMPORARY TABLES, EXECUTE, CREATE VIEW, SHOW VIEW для всех таблиц базы данных "BoatStation"
3. Инструктор:
   1. Имя пользователя: instructor
   2. Пароль: 5F(ku3jJ2\_k4b]\*3
   3. Привилегии:
      1. SELECT для всей базы данных
      2. INSERT, DELETE для таблиц "Issuance\_of\_watercraft" и “flotation\_device”
      3. UPDATE для таблицы "Technical\_condition"
      4. INSERT для таблиц: "Danger\_level", “Customer”, “Repair”, “Orders”.

Создана база данных На основе физической модели была реализована база данных в выбранной СУБД, MySQL. Были созданы таблицы, определены типы данных и связи между ними. Этот этап включал в себя создание скриптов для создания таблиц, индексов, внешних ключей и пользователей базы данных.

# 4 Тестирование работы системы

Тестирование приложения будет проводится через созданное приложение. Для разработки приложения был выбран язык программирования C#. Используя библиотеку using MySql.Data.MySqlClient; установлено соединение с базой данных, а в качестве интерфейса использовалась платформа Windows Forms(рис.4).

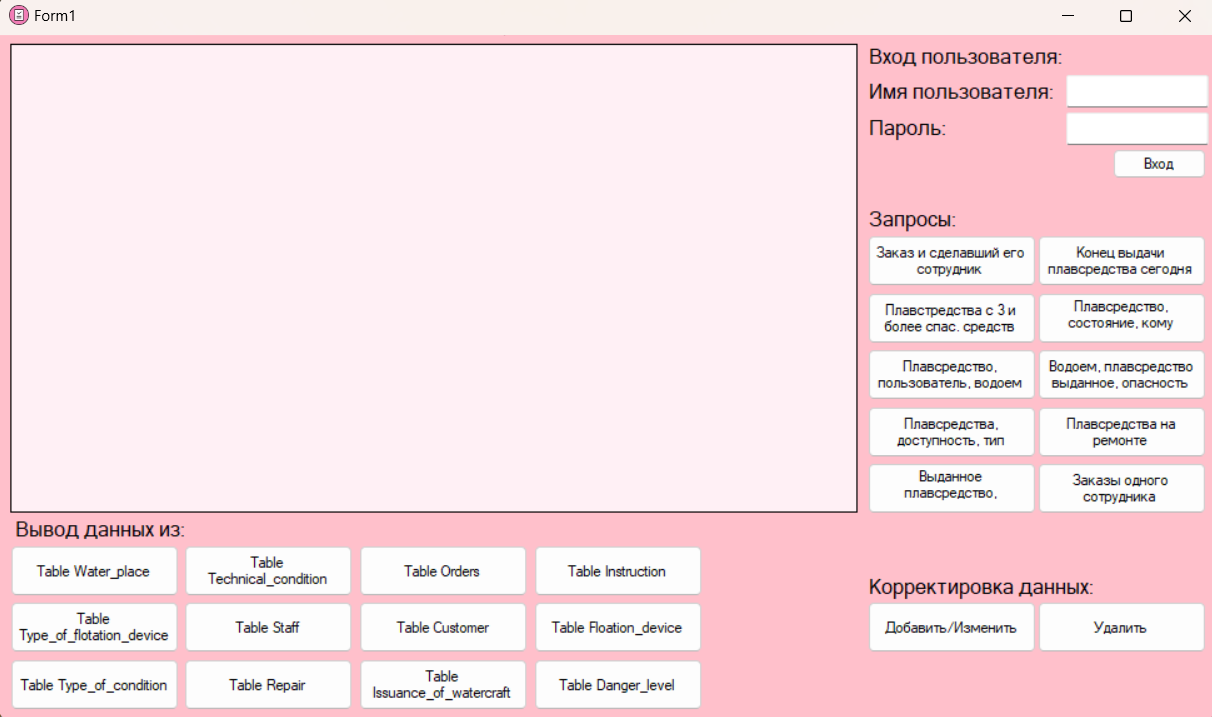


Рисунок 4 - Интерфейс приложения

Приложение состоит из окна, в котором будут выводиться таблицы, и кнопок, реализующих просмотр всех таблиц и запросов. Для добавления или изменения записи в таблицу нужно внести соответствующие корректировки в таблицу на экране и нажать кнопку “Добавить/Изменить”(рис.5).

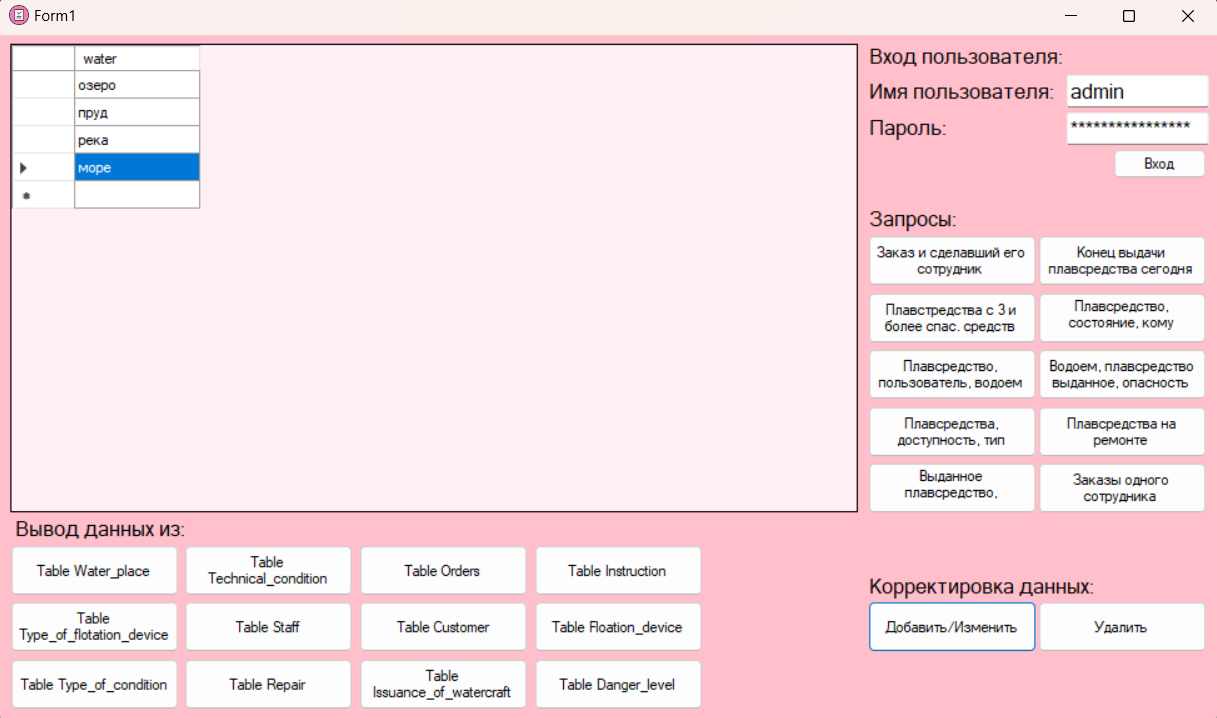


Рисунок 5 - Добавление в таблицу “Water\_рlace” данных

Для удаления данных из таблиц есть соответствующая кнопка “Удалить”(рис.6)

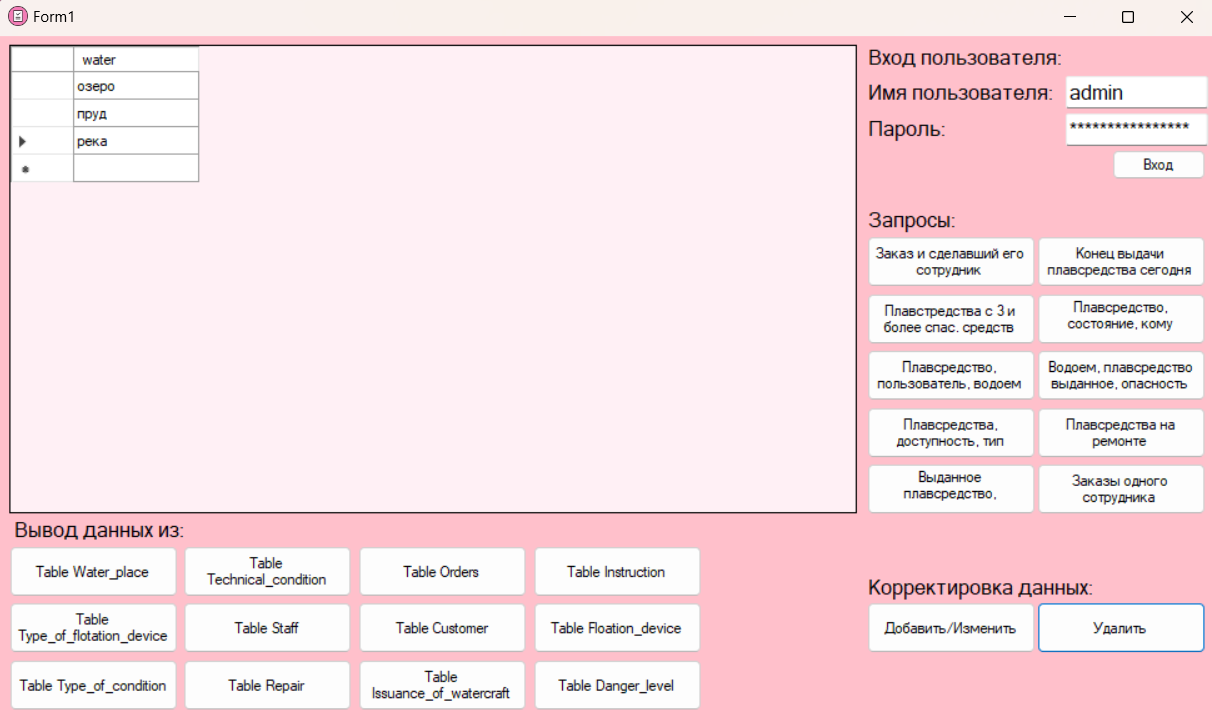


Рисунок 6 - Удаление из таблицы “Water\_рlace” данных

Если у пользователя будет недостаточно прав доступа на изменение данных, то измененные данные не сохранятся в базе, хотя будут отображаться на экране до обновления таблицы.(рис.7, 8)

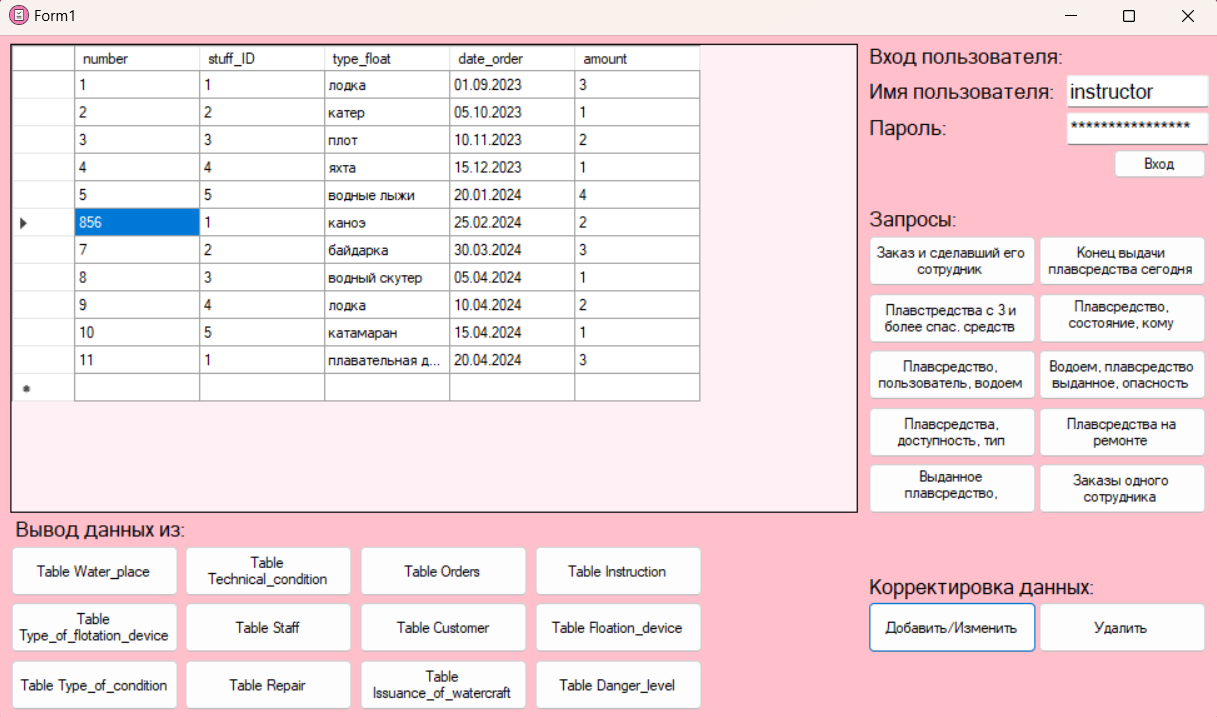


Рисунок 7 - Изменение без прав доступа до обновления

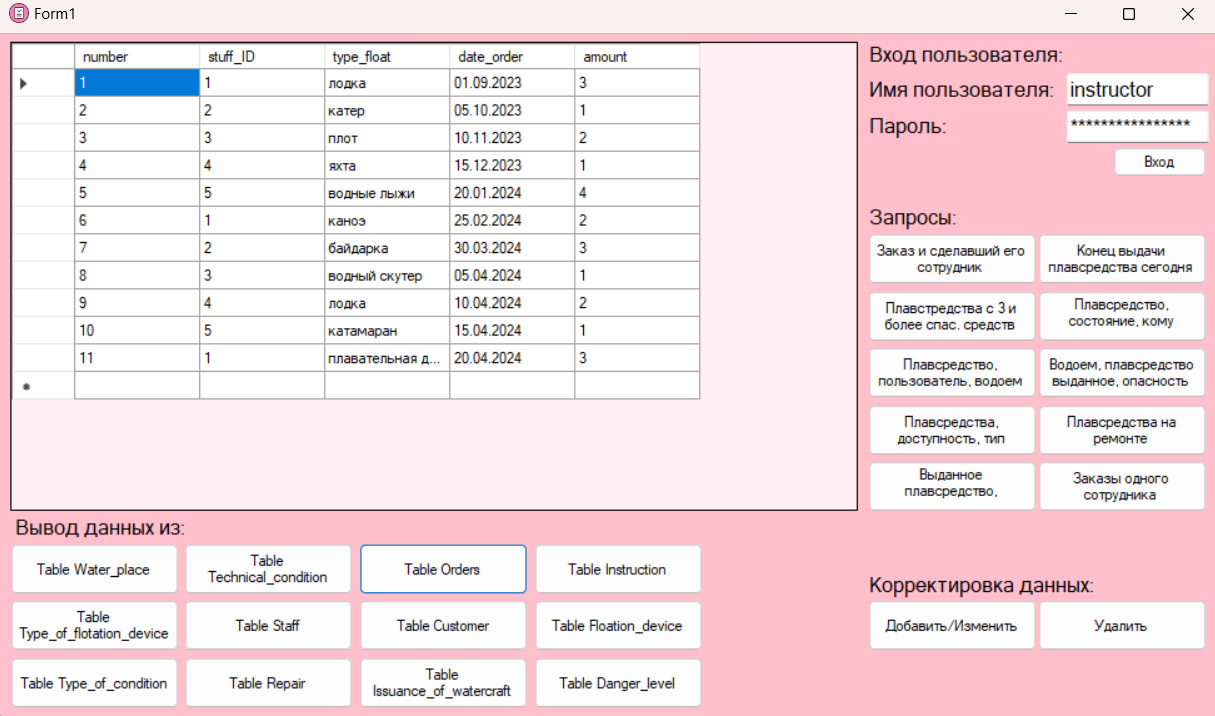


Рисунок 8 - Изменение без прав доступа после обновления

Разработано и протестировано приложение для базы данных. Тестирование показало, что все функции работают корректно.

# 

# Заключение

В ходе курсовой работы была разработана база данных “Лодочная станция” и приложение для работы с ней.

Для достижения цели были построены 3 модели базы данных: концептуальная, логическая, физическая. Прописаны SQL-скрипты для создания базы данных, создания таблиц и их заполнения, а также разработано приложения с использованием языка C# и интерфейсом Windows Forms.

Разработанная база данных и приложение к ней удовлетворяет предъявленным требованиям, а также реализуют санкционированный доступ к данным из таблиц.

Перспектива дальнейшей разработки заключается в добавлении функционала для пользователей приложения, также возможна разработка более сильной защиты приложения от взлома и несанкционированного доступа к данным таблиц.

# Список литературы

1. Зотин, А. Г. Разработка приложений баз данных : учебное пособие : в 2 частях / А. Г. Зотин, А. И. Пахирка. — Красноярск : СибГУ им. академика М. Ф. Решетнёва, 2023 — Часть 1 : Основы разработки клиент-серверных приложений — 2023. — 108 с. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/400592> (дата обращения: 10.05.2024). — Режим доступа: для авториз. пользователей.
2. Кожевникова, П. В. PHP и MySQL : учебное пособие / П. В. Кожевникова. — Ухта : УГТУ, 2020. — 51 с. — Текст : электронный // Лань : электронно-библиотечная система. — URL: <https://e.lanbook.com/book/209591> (дата обращения: 10.05.2024). — Режим доступа: для авториз. пользователей.

Приложение А. SQL-скрипт для создания базы данных

-- Структура таблицы `Customer`

--

CREATE TABLE `Customer` (

`passport` decimal(10,0) NOT NULL,

`surname\_cust` varchar(30) NOT NULL,

`Street` varchar(15) NOT NULL,

`House` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Дамп данных таблицы `Customer`

--

INSERT INTO `Customer` (`passport`, `surname\_cust`, `Street`, `House`) VALUES

('1234432199', 'Соловьев', 'Ленина', 56),

('1234567842', 'Иванов', 'Парковый', 20),

('2345678940', 'Петров', 'Уральская', 7),

('3456789054', 'Сидоров', 'Борчанинова', 52),

('4567890128', 'Козлов', 'Боровая', 96),

('5678901273', 'Новикова', 'Екатерининская', 1),

('6789012370', 'Семенов', 'Попова', 83),

('7890123405', 'Михайлов', 'Окулова', 13),

('8901234582', 'Кузнецов', 'Луначарского', 52),

('9012345645', 'Ковалев', 'Беляева', 37);

-- --------------------------------------------------------

--

-- Структура таблицы `Danger\_level`

--

CREATE TABLE `Danger\_level` (

`water` varchar(50) NOT NULL,

`type\_float` varchar(50) NOT NULL,

`level` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Дамп данных таблицы `Danger\_level`

--

INSERT INTO `Danger\_level` (`water`, `type\_float`, `level`) VALUES

('озеро', 'байдарка', 1),

('озеро', 'водные лыжи', 3),

('озеро', 'водный скутер', 3),

('озеро', 'каноэ', 1),

('озеро', 'катамаран', 2),

('озеро', 'катер', 3),

('озеро', 'лодка', 2),

('озеро', 'плавательная доска', 2),

('озеро', 'плот', 2),

('озеро', 'яхта', 1),

('пруд', 'байдарка', 1),

('пруд', 'водные лыжи', 3),

('пруд', 'водный скутер', 3),

('пруд', 'каноэ', 1),

('пруд', 'катамаран', 2),

('пруд', 'катер', 3),

('пруд', 'лодка', 2),

('пруд', 'плавательная доска', 2),

('пруд', 'плот', 2),

('пруд', 'яхта', 1),

('река', 'байдарка', 2),

('река', 'водные лыжи', 5),

('река', 'водный скутер', 5),

('река', 'каноэ', 3),

('река', 'катамаран', 3),

('река', 'катер', 3),

('река', 'лодка', 3),

('река', 'плавательная доска', 4),

('река', 'плот', 5),

('река', 'яхта', 5);

-- --------------------------------------------------------

--

-- Структура таблицы `Flotation\_device`

--

CREATE TABLE `Flotation\_device` (

`number\_of\_float` int(11) NOT NULL,

`type\_float` varchar(50) NOT NULL,

`life\_safing\_equipment` varchar(50) NOT NULL,

`amt\_life\_safing\_equipment` int(11) NOT NULL,

`availability` varchar(8) NOT NULL DEFAULT 'свободно'

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Дамп данных таблицы `Flotation\_device`

--

INSERT INTO `Flotation\_device` (`number\_of\_float`, `type\_float`, `life\_safing\_equipment`, `amt\_life\_safing\_equipment`, `availability`) VALUES

(1, 'байдарка', 'Спасательный жилет', 4, 'свободно'),

(2, 'водные лыжи', 'Спасательный круг', 3, 'занято'),

(3, 'водный скутер', 'Спасательное кольцо', 2, 'занято'),

(4, 'каноэ', 'Спасательный жилет', 1, 'свободно'),

(5, 'катамаран', 'Спасательный жилет', 2, 'занято'),

(6, 'катер', 'Спасательный жилет', 3, 'занято'),

(7, 'лодка', 'Спасательный жилет', 2, 'занято'),

(9, 'плот', 'Спасательный жилет', 1, 'занято'),

(10, 'яхта', 'Спасательный жилет', 2, 'занято'),

(11, 'байдарка', 'Спасательный жилет', 4, 'занято'),

(12, 'водные лыжи', 'Спасательный круг', 3, 'свободно'),

(14, 'каноэ', 'Спасательный жилет', 2, 'занято'),

(15, 'катамаран', 'Спасательный жилет', 1, 'свободно'),

(16, 'катер', 'Спасательный жилет', 4, 'занято'),

(17, 'лодка', 'Спасательный жилет', 2, 'занято'),

(18, 'плавательная доска', 'Спасательное кольцо', 3, 'свободно'),

(19, 'плот', 'Спасательный жилет', 4, 'занято'),

(20, 'яхта', 'Спасательный жилет', 1, 'занято');

--

-- Триггеры `Flotation\_device`

--

DELIMITER $$

CREATE TRIGGER `add\_technical\_condition` AFTER INSERT ON `Flotation\_device` FOR EACH ROW BEGIN

INSERT INTO Technical\_condition (number\_of\_float, type)

VALUES (NEW.number\_of\_float, 'отличное');

END

$$

DELIMITER ;

-- --------------------------------------------------------

--

-- Структура таблицы `Instruction`

--

CREATE TABLE `Instruction` (

`instruction\_name` varchar(50) NOT NULL,

`description` varchar(50) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Дамп данных таблицы `Instruction`

--

INSERT INTO `Instruction` (`instruction\_name`, `description`) VALUES

('Безопасное плавание', 'Основные правила безопасности на воде'),

('Использование спасательного круга', 'Правильное применение спасательного круга'),

('Правила использования спасательного жилета', 'Инструкция по применению спасательного жилета'),

('Управление лодкой', 'Основы управления лодкой');

-- --------------------------------------------------------

--

-- Структура таблицы `Issuance\_of\_watercraft`

--

CREATE TABLE `Issuance\_of\_watercraft` (

`passport` decimal(10,0) NOT NULL,

`water` varchar(50) NOT NULL,

`number\_of\_float` int(11) NOT NULL,

`date` datetime NOT NULL,

`end\_date` date NOT NULL,

`instruction\_name` varchar(50) NOT NULL,

`stuff\_ID` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Дамп данных таблицы `Issuance\_of\_watercraft`

--

INSERT INTO `Issuance\_of\_watercraft` (`passport`, `water`, `number\_of\_float`, `date`, `end\_date`, `instruction\_name`, `stuff\_ID`) VALUES

('1234432199', 'река', 3, '2024-05-11 09:00:00', '2024-05-18', 'Управление лодкой', 1),

('1234567842', 'пруд', 2, '2024-05-11 18:00:00', '2024-05-15', 'Безопасное плавание', 5),

('2345678940', 'пруд', 11, '2024-05-11 10:00:00', '2024-05-16', 'Безопасное плавание', 2),

('3456789054', 'озеро', 7, '2024-05-11 11:00:00', '2024-05-15', 'Использование спасательного круга', 3),

('4567890128', 'река', 6, '2024-05-11 12:00:00', '2024-05-17', 'Правила использования спасательного жилета', 4),

('5678901273', 'озеро', 16, '2024-05-11 23:00:00', '2024-05-17', 'Использование спасательного круга', 5),

('6789012370', 'пруд', 17, '2024-05-11 14:00:00', '2024-05-19', 'Безопасное плавание', 1),

('7890123405', 'озеро', 10, '2024-05-11 15:00:00', '2024-05-18', 'Использование спасательного круга', 2),

('8901234582', 'река', 14, '2024-05-11 16:00:00', '2024-05-15', 'Правила использования спасательного жилета', 3),

('9012345645', 'озеро', 20, '2024-05-11 17:00:00', '2024-05-16', 'Управление лодкой', 4);

--

-- Триггеры `Issuance\_of\_watercraft`

--

DELIMITER $$

CREATE TRIGGER `prevent\_issuance\_duplicate` BEFORE INSERT ON `Issuance\_of\_watercraft` FOR EACH ROW BEGIN

DECLARE repair\_count INT;

SELECT COUNT(\*) INTO repair\_count

FROM Repair

WHERE number\_of\_float = NEW.number\_of\_float

AND finish\_date IS NULL;

IF repair\_count > 0 THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Добавление плавсредства невозможно, так как оно находится в ремонте.';

END IF;

END

$$

DELIMITER ;

DELIMITER $$

CREATE TRIGGER `update\_availability` AFTER INSERT ON `Issuance\_of\_watercraft` FOR EACH ROW BEGIN

DECLARE count\_used INT;

SELECT COUNT(\*) INTO count\_used

FROM Issuance\_of\_watercraft

WHERE number\_of\_float = NEW.number\_of\_float;

IF count\_used > 0 THEN

UPDATE Flotation\_device

SET availability = 'занято'

WHERE number\_of\_float = NEW.number\_of\_float;

ELSE

UPDATE Flotation\_device

SET availability = 'свободно'

WHERE number\_of\_float = NEW.number\_of\_float;

END IF;

END

$$

DELIMITER ;

-- --------------------------------------------------------

--

-- Структура таблицы `Orders`

--

CREATE TABLE `Orders` (

`number` int(11) NOT NULL,

`stuff\_ID` int(11) NOT NULL,

`type\_float` varchar(50) NOT NULL,

`date\_order` date NOT NULL,

`amount` int(11) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Дамп данных таблицы `Orders`

--

INSERT INTO `Orders` (`number`, `stuff\_ID`, `type\_float`, `date\_order`, `amount`) VALUES

(1, 1, 'лодка', '2023-09-01', 3),

(2, 2, 'катер', '2023-10-05', 1),

(3, 3, 'плот', '2023-11-10', 2),

(4, 4, 'яхта', '2023-12-15', 1),

(5, 5, 'водные лыжи', '2024-01-20', 4),

(6, 1, 'каноэ', '2024-02-25', 2),

(7, 2, 'байдарка', '2024-03-30', 3),

(8, 3, 'водный скутер', '2024-04-05', 1),

(9, 4, 'лодка', '2024-04-10', 2),

(10, 5, 'катамаран', '2024-04-15', 1),

(11, 1, 'плавательная доска', '2024-04-20', 3);

-- --------------------------------------------------------

--

-- Структура таблицы `Repair`

--

CREATE TABLE `Repair` (

`repair\_ID` int(11) NOT NULL,

`number\_of\_float` int(11) NOT NULL,

`start\_date` date NOT NULL,

`finish\_date` date DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Дамп данных таблицы `Repair`

--

INSERT INTO `Repair` (`repair\_ID`, `number\_of\_float`, `start\_date`, `finish\_date`) VALUES

(1, 1, '2023-09-01', '2023-09-05'),

(2, 2, '2023-10-05', '2023-10-10'),

(3, 3, '2023-11-10', '2023-11-15'),

(4, 4, '2023-12-15', '2023-12-25'),

(5, 5, '2024-01-20', '2024-01-25'),

(6, 1, '2024-02-25', '2024-03-01'),

(7, 2, '2024-03-30', NULL),

(8, 3, '2024-04-10', '2024-04-15'),

(9, 4, '2024-04-20', NULL),

(10, 5, '2024-05-01', '2024-05-05');

-- --------------------------------------------------------

--

-- Структура таблицы `Staff`

--

CREATE TABLE `Staff` (

`stuff\_ID` int(11) NOT NULL,

`surname` varchar(30) NOT NULL,

`name` varchar(15) NOT NULL,

`patronymic\_name` varchar(50) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Дамп данных таблицы `Staff`

--

INSERT INTO `Staff` (`stuff\_ID`, `surname`, `name`, `patronymic\_name`) VALUES

(1, 'Смирнов', 'Иван', 'Иванович'),

(2, 'Иванов', 'Петр', 'Сергеевич'),

(3, 'Петров', 'Алексей', 'Александрович'),

(4, 'Сидоров', 'Дмитрий', 'Иванович'),

(5, 'Козлова', 'Мария', 'Петровна');

-- --------------------------------------------------------

--

-- Структура таблицы `Technical\_condition`

--

CREATE TABLE `Technical\_condition` (

`number\_of\_float` int(11) NOT NULL,

`type` varchar(50) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Дамп данных таблицы `Technical\_condition`

--

INSERT INTO `Technical\_condition` (`number\_of\_float`, `type`) VALUES

(3, 'отличное'),

(12, 'отличное'),

(16, 'отличное'),

(20, 'отличное'),

(4, 'плохое'),

(6, 'плохое'),

(10, 'плохое'),

(15, 'плохое'),

(19, 'плохое'),

(2, 'среднее'),

(7, 'среднее'),

(9, 'среднее'),

(14, 'среднее'),

(18, 'среднее'),

(1, 'хорошее'),

(5, 'хорошее'),

(11, 'хорошее'),

(17, 'хорошее');

-- --------------------------------------------------------

--

-- Структура таблицы `Type\_of\_condition`

--

CREATE TABLE `Type\_of\_condition` (

`type` varchar(50) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Дамп данных таблицы `Type\_of\_condition`

--

INSERT INTO `Type\_of\_condition` (`type`) VALUES

('отличное'),

('плохое'),

('среднее'),

('хорошее');

-- --------------------------------------------------------

--

-- Структура таблицы `Type\_of\_flotation\_device`

--

CREATE TABLE `Type\_of\_flotation\_device` (

`type\_float` varchar(50) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Дамп данных таблицы `Type\_of\_flotation\_device`

--

INSERT INTO `Type\_of\_flotation\_device` (`type\_float`) VALUES

('байдарка'),

('водные лыжи'),

('водный скутер'),

('каноэ'),

('катамаран'),

('катер'),

('лодка'),

('плавательная доска'),

('плот'),

('яхта');

-- --------------------------------------------------------

--

-- Структура таблицы `Water\_place`

--

CREATE TABLE `Water\_place` (

`water` varchar(50) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Дамп данных таблицы `Water\_place`

--

INSERT INTO `Water\_place` (`water`) VALUES

('озеро'),

('пруд'),

('река');

--

-- Индексы сохранённых таблиц

--

--

-- Индексы таблицы `Customer`

--

ALTER TABLE `Customer`

ADD PRIMARY KEY (`passport`);

--

-- Индексы таблицы `Danger\_level`

--

ALTER TABLE `Danger\_level`

ADD PRIMARY KEY (`water`,`type\_float`),

ADD KEY `water` (`water`),

ADD KEY `tupe\_float` (`type\_float`);

--

-- Индексы таблицы `Flotation\_device`

--

ALTER TABLE `Flotation\_device`

ADD PRIMARY KEY (`number\_of\_float`),

ADD KEY `tupe\_float` (`type\_float`);

--

-- Индексы таблицы `Instruction`

--

ALTER TABLE `Instruction`

ADD PRIMARY KEY (`instruction\_name`);

--

-- Индексы таблицы `Issuance\_of\_watercraft`

--

ALTER TABLE `Issuance\_of\_watercraft`

ADD PRIMARY KEY (`passport`),

ADD UNIQUE KEY `number\_of\_float` (`number\_of\_float`) USING BTREE,

ADD UNIQUE KEY `passport` (`passport`) USING BTREE,

ADD KEY `water` (`water`),

ADD KEY `instruction\_name` (`instruction\_name`),

ADD KEY `stuff\_ID` (`stuff\_ID`);

--

-- Индексы таблицы `Orders`

--

ALTER TABLE `Orders`

ADD PRIMARY KEY (`number`),

ADD KEY `stuff\_ID` (`stuff\_ID`),

ADD KEY `tupe\_float` (`type\_float`);

--

-- Индексы таблицы `Repair`

--

ALTER TABLE `Repair`

ADD PRIMARY KEY (`repair\_ID`),

ADD KEY `number\_of\_float` (`number\_of\_float`);

--

-- Индексы таблицы `Staff`

--

ALTER TABLE `Staff`

ADD PRIMARY KEY (`stuff\_ID`);

--

-- Индексы таблицы `Technical\_condition`

--

ALTER TABLE `Technical\_condition`

ADD PRIMARY KEY (`number\_of\_float`),

ADD UNIQUE KEY `number\_of\_float` (`number\_of\_float`) USING BTREE,

ADD KEY `type` (`type`);

--

-- Индексы таблицы `Type\_of\_condition`

--

ALTER TABLE `Type\_of\_condition`

ADD PRIMARY KEY (`type`);

--

-- Индексы таблицы `Type\_of\_flotation\_device`

--

ALTER TABLE `Type\_of\_flotation\_device`

ADD PRIMARY KEY (`type\_float`);

--

-- Индексы таблицы `Water\_place`

--

ALTER TABLE `Water\_place`

ADD PRIMARY KEY (`water`);

--

-- AUTO\_INCREMENT для сохранённых таблиц

--

--

-- AUTO\_INCREMENT для таблицы `Flotation\_device`

--

ALTER TABLE `Flotation\_device`

MODIFY `number\_of\_float` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=21;

--

-- AUTO\_INCREMENT для таблицы `Orders`

--

ALTER TABLE `Orders`

MODIFY `number` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=12;

--

-- AUTO\_INCREMENT для таблицы `Repair`

--

ALTER TABLE `Repair`

MODIFY `repair\_ID` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=11;

--

-- AUTO\_INCREMENT для таблицы `Staff`

--

ALTER TABLE `Staff`

MODIFY `stuff\_ID` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=6;

--

-- Ограничения внешнего ключа сохраненных таблиц

--

--

-- Ограничения внешнего ключа таблицы `Danger\_level`

--

ALTER TABLE `Danger\_level`

ADD CONSTRAINT `danger\_level\_ibfk\_1` FOREIGN KEY (`water`) REFERENCES `Water\_place` (`water`) ON DELETE CASCADE ON UPDATE CASCADE,

ADD CONSTRAINT `danger\_level\_ibfk\_2` FOREIGN KEY (`type\_float`) REFERENCES `Type\_of\_flotation\_device` (`type\_float`) ON DELETE CASCADE ON UPDATE CASCADE;

--

-- Ограничения внешнего ключа таблицы `Flotation\_device`

--

ALTER TABLE `Flotation\_device`

ADD CONSTRAINT `flotation\_device\_ibfk\_1` FOREIGN KEY (`type\_float`) REFERENCES `Type\_of\_flotation\_device` (`type\_float`) ON DELETE NO ACTION ON UPDATE NO ACTION;

--

-- Ограничения внешнего ключа таблицы `Issuance\_of\_watercraft`

--

ALTER TABLE `Issuance\_of\_watercraft`

ADD CONSTRAINT `issuance\_of\_watercraft\_ibfk\_2` FOREIGN KEY (`water`) REFERENCES `Water\_place` (`water`) ON DELETE NO ACTION ON UPDATE NO ACTION,

ADD CONSTRAINT `issuance\_of\_watercraft\_ibfk\_3` FOREIGN KEY (`number\_of\_float`) REFERENCES `Flotation\_device` (`number\_of\_float`) ON DELETE NO ACTION ON UPDATE CASCADE,

ADD CONSTRAINT `issuance\_of\_watercraft\_ibfk\_4` FOREIGN KEY (`instruction\_name`) REFERENCES `Instruction` (`instruction\_name`) ON DELETE NO ACTION ON UPDATE CASCADE,

ADD CONSTRAINT `issuance\_of\_watercraft\_ibfk\_5` FOREIGN KEY (`stuff\_ID`) REFERENCES `Staff` (`stuff\_ID`) ON DELETE NO ACTION ON UPDATE CASCADE,

ADD CONSTRAINT `issuance\_of\_watercraft\_ibfk\_6` FOREIGN KEY (`passport`) REFERENCES `Customer` (`passport`) ON DELETE NO ACTION ON UPDATE CASCADE;

--

-- Ограничения внешнего ключа таблицы `Orders`

--

ALTER TABLE `Orders`

ADD CONSTRAINT `orders\_ibfk\_1` FOREIGN KEY (`stuff\_ID`) REFERENCES `Staff` (`stuff\_ID`) ON DELETE NO ACTION ON UPDATE CASCADE,

ADD CONSTRAINT `orders\_ibfk\_2` FOREIGN KEY (`type\_float`) REFERENCES `Type\_of\_flotation\_device` (`type\_float`) ON DELETE NO ACTION ON UPDATE CASCADE;

--

-- Ограничения внешнего ключа таблицы `Repair`

--

ALTER TABLE `Repair`

ADD CONSTRAINT `repair\_ibfk\_1` FOREIGN KEY (`number\_of\_float`) REFERENCES `Flotation\_device` (`number\_of\_float`) ON DELETE NO ACTION ON UPDATE NO ACTION;

--

-- Ограничения внешнего ключа таблицы `Technical\_condition`

--

ALTER TABLE `Technical\_condition`

ADD CONSTRAINT `technical\_condition\_ibfk\_1` FOREIGN KEY (`number\_of\_float`) REFERENCES `Flotation\_device` (`number\_of\_float`) ON DELETE CASCADE ON UPDATE CASCADE,

ADD CONSTRAINT `technical\_condition\_ibfk\_2` FOREIGN KEY (`type`) REFERENCES `Type\_of\_condition` (`type`) ON DELETE CASCADE ON UPDATE CASCADE;

COMMIT;

Приложение Б. Контрольные запросы

1. Вывести список заказов и ответственных за них сотрудников

SELECT o.number, o.date\_order, o.amount, s.surname, s.name, s.patronymic\_name FROM Orders o JOIN Staff s ON o.stuff\_ID = s.stuff\_ID;

1. Вывести выданные плавсредства, срок использования которых заканчивается сегодня

SELECT \*

FROM Issuance\_of\_watercraft

WHERE end\_date = CURDATE();

1. Вывести плавсредства, у которых количество спасательного оборудования больше 3

SELECT \*

FROM Flotation\_device

WHERE lifesaving\_equipment > 2;

1. Вывести выданные плавсредства, их пользователей и их техническое состояние

SELECT i.water, i.date, i.end\_date, c.surname\_cust, c.address, t.type AS technical\_condition

FROM Issuance\_of\_watercraft i

JOIN Customer c ON i.passport = c.passport

JOIN Technical\_condition t ON i.number\_of\_float = t.number\_of\_float;

1. Вывести номер выданного плавсредства, его пользователя и водоем

SELECT i.number\_of\_float, c.surname\_cust, c.address, p.name AS pond\_name

FROM Issuance\_of\_watercraft i

JOIN Customer c ON i.passport = c.passport

JOIN Pond p ON i.pond\_ID = p.pond\_ID;

1. Показать водоемы, где выдаются плавательные средства, вместе с их уровнем опасности:

SELECT p.name AS pond\_name, p.danger\_level

FROM Pond p

JOIN Issuance\_of\_watercraft i ON p.pond\_ID = i.pond\_ID;

1. Отобразить все свободные плавательные средства вместе с их статусом доступности и типом плавательного средства:

SELECT f.tupe\_float, t.availability, t.type AS flotation\_device\_type

FROM Flotation\_device f

JOIN Availability t ON f.availability = t.availability

WHERE t.availability = 'free';

1. Показать плавательные средства, находящиеся в настоящее время на ремонте

SELECT f.tupe\_float, t.type AS flotation\_device\_type

FROM Flotation\_device f

JOIN Technical\_condition t ON f.number\_of\_float = t.number\_of\_float

WHERE t.type = 'repair';

1. Показать персонал, ответственный за каждое выданное плавательное средство

SELECT s.surname, s.name, s.patronymic\_name, i.number\_of\_float

FROM Staff s

JOIN Issuance\_of\_watercraft i ON s.stuff\_ID = i.stuff\_ID;

1. Вывести список всех заказов, сделанных определенным сотрудником

SELECT o.type\_float, o.date\_order, o.amount

FROM Orders o

JOIN Staff s ON o.stuff\_ID = s.stuff\_ID

WHERE s.stuff\_ID = 1;

Приложение В. Права доступа пользователей

1. Администратор базы данных

GRANT ALL PRIVILEGES ON \*.\* TO 'admin'@'%' WITH GRANT OPTION;

1. Менеджер

GRANT USAGE ON \*.\* TO 'manager'@'%';

GRANT SELECT, INSERT, UPDATE, DELETE, CREATE TEMPORARY TABLES, EXECUTE, CREATE VIEW, SHOW VIEW ON `boatstation`.\* TO 'manager'@'%';

1. Инструктор

GRANT USAGE ON \*.\* TO 'instructor'@'%';

GRANT SELECT ON `boatstation`.\* TO 'instructor'@'%';

GRANT INSERT, DELETE ON `boatstation`.`issuance\_of\_watercraft` TO 'instructor'@'%';

GRANT INSERT, DELETE ON `boatstation`.`flotation\_device` TO 'instructor'@'%';

GRANT UPDATE ON `boatstation`.`technical\_condition` TO 'instructor'@'%';

GRANT INSERT ON `boatstation`.`danger\_level` TO 'instructor'@'%';

GRANT INSERT ON `boatstation`.`customer` TO 'instructor'@'%';

GRANT INSERT, UPDATE ON `boatstation`.`repair` TO 'instructor'@'%';

GRANT INSERT ON `boatstation`.`orders` TO 'instructor'@'%';