Оглавление

**Введение2**

**1 Аналитическая часть3**

1.1 Постановка задачи3

1.2 Анализ существующих решений3

1.3 Формализация данных4

1.4 Типы пользователей5

1.5 Описание существующих СУБД6

1.5.1 Основные функции СУБД6

1.5.2 Классификации СУБД7

1.5.3 Обзор наиболее популярных СУБД9

Выводы11

**Список литературы12**

**Введение**

Проблема поиска места для репетиций является актуальной для любого музыканта, тем более группы. В крупных городах есть достаточно много репетиционных баз (далее: реп. баз). Все они имеют разные цены и характеристики. Поэтому существует потребность в приложении, которое собирало бы воедино всю имеющуюся информацию о различных репетиционных базах, таким образом освобождая музыкантов от необходимости вручную искать и изучать каждую реп. базу, заходить на их сайты, звонить лично, чтобы забронировать репетицию и т. д.

Цель данной работы – реализовать базу данных, содержащую информацию о репетиционных базах. В приложении, работающем с этой БД, должна быть возможность для музыканта бронировать или отменять свои репетиции, а для владельца реп. базы - отслеживать записи на свою реп. базу.

Чтобы достигнуть поставленной цели, требуется решить следующие задачи:

* формализовать задание, определить необходимый функционал;
* провести анализ СУБД;
* описать структуру БД;
* создать и заполнить БД;
* реализовать интерфейс для доступа к БД;
* разработать ПО, которое позволит пользователю-музыканту бронировать и отменять свои репетиции, а владельцу отслеживать их.

**1 Аналитическая часть**

В этом разделе будет проанализирована поставленная задача, и рассмотрены различные способы её реализации.

**1.1 Постановка задачи**

Необходимо разработать программу для отображения информации о репетиционных базах с возможностью для музыканта бронировать или отменять свои репетиции, а для владельца реп. базы - отслеживать записи на свою реп. базу.

**1.2 Анализ существующих решений**

На сегодняшний день существует всего два подобных приложения:

* MUSbooking

Это наиболее известное и популярное приложение по бронированию творческих площадок.

Его основные преимущества:

* + возможность бронирования не только репетиционных баз, но и, например, студий звукозаписи, танцевальных залов и т. д.;
  + возможность бронирования творческих площадок в других городах России (не только в Москве).

Основной недостаток: нельзя посмотреть сразу весь список зарегистрированных реп. баз в текущем городе. Список доступных реп. баз можно посмотреть только после указания конкретного времени репетиции, что крайне неудобно в определённых ситуациях.

* TONESKY

Основное преимущество: возможность заранее посмотреть весь список зарегистрированных реп. баз.

Из недостатков:

* + нет поиска по реп. базам;
  + нет цены на превью репетиционной комнаты (т. е., чтобы посмотреть цену, надо зайти «вглубь» и посмотреть подробную информацию);
  + ориентировано только на Москву (не существенно в рамках моей задачи).

**1.3 Формализация данных**

База данных должна хранить информацию о:

* репетиционной базе и её комнатах;
* оборудовании и дополнительных услугах;
* пользователях (музыкантах и владельцах) и об их забронированных репетициях или реп. базах соответственно.

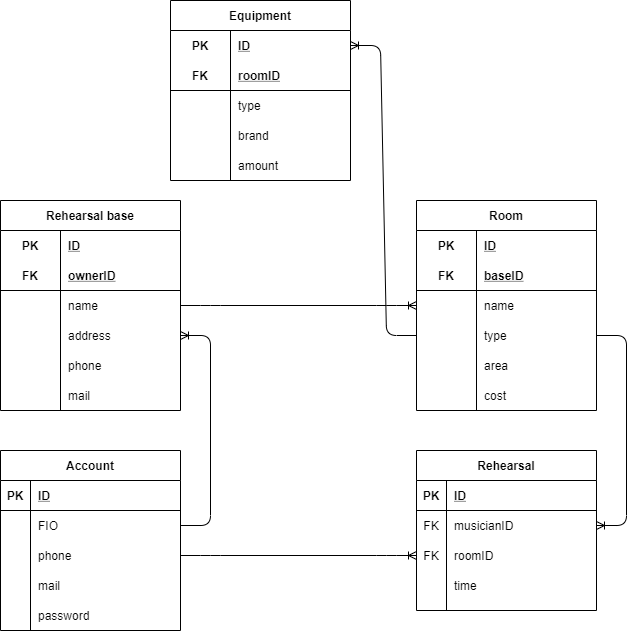


Рис. 1.1: ER диаграмма разрабатываемой базы данных

Таблица 1.1: категории и сведения о данных

|  |  |
| --- | --- |
| Категория | Сведения |
| Реп. база | Название, адрес, телефон, почта, кому принадлежит (аккаунт) |
| Комната | Название, тип (вокал/группа и т. д.), площадь, стоимость за 3 часа, к какой реп. базе относится |
| Оборудование в комнате | Тип (усилитель/ударные/микрофон и т. д.), бренд, количество, к какой комнате относится |
| Аккаунт | ФИО, телефон, почта, пароль |
| Репетиция | Время, какой музыкант (аккаунт), какая комната |

**1.4 Типы пользователей**

Из задачи ясно, что есть два типа пользователей: обычный музыкант и владелец реп. базы. Помимо этого, будет также выделена роль администратора приложения. Для всех троих будет нужна авторизация.

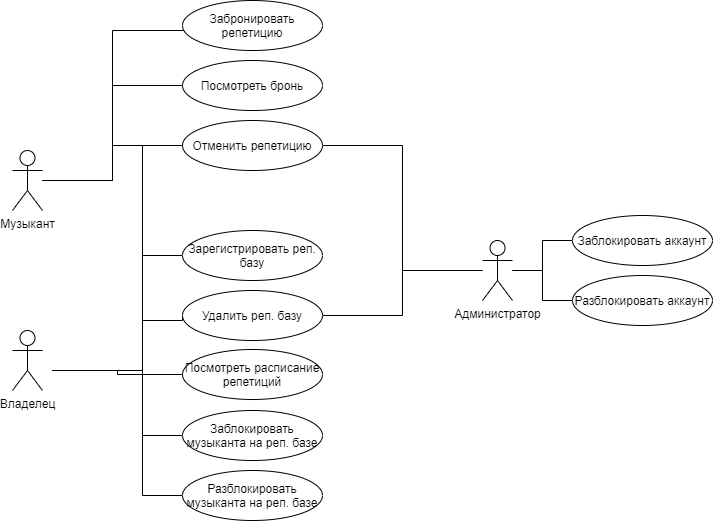


Рис. 1.2: Use-case диаграмма приложения

Таблица 1.2: типы пользователей и их полномочия

|  |  |
| --- | --- |
| Тип пользователя | Полномочия |
| Музыкант | Бронирование репетиций, отмена репетиций, просмотр брони |
| Владелец | Добавление реп. базы, удаление своей реп. базы, просмотр записей на репетиции, отмена репетиций, блокировка/разблокировка музыканта на своей реп. базе |
| Администратор | Удаление реп. базы, отмена репетиций, блокировка/разблокировка аккаунта |

**1.5 Описание существующих СУБД**

Система управления базами данных (СУБД) – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием [баз данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) [1].

**1.5.1 Основные функции СУБД**

Основными функциями СУБД являются:

* управление данными на внешней памяти;
* управление данными в оперативной памяти с использованием дискового кэша;
* журнализация изменений, резервное копирование и восстановление базы данных после сбоев;
* поддержка языков БД.

**1.5.2 Классификации СУБД**

По модели данных:

* дореляционные;

**Иерархические.** Это модель данных, где используется представление базы данных в виде древовидной (иерархической) структуры, состоящей из объектов (данных) различных уровней.

**Сетевые.** Это логическая модель данных, являющаяся расширением иерархического подхода, строгая математическая теория, описывающая структурный аспект, аспект целостности и аспект обработки данных в сетевых базах данных.

Разница между иерархической моделью данных и сетевой состоит в том, что в иерархических структурах запись-потомок должна иметь в точности одного предка, а в сетевой структуре данных у потомка может иметься любое число предков.

* реляционные.

Реляционная модель данных является совокупностью данных и состоит из набора двумерных таблиц. При табличной организации отсутствует иерархия элементов. Таблицы состоят из строк – записей и столбцов – полей. На пересечении строк и столбцов находятся конкретные значения. Для каждого поля определяется множество его значений. За счет возможности просмотра строк и столбцов в любом порядке достигается гибкость выбора подмножества элементов.

Реляционная модель является удобной и наиболее широко используемой формой представления данных.

Модель данных — это абстрактное, самодостаточное, логическое определение объектов, операторов и прочих элементов, в совокупности составляющих абстрактную машину доступа к данным, с которой взаимодействует пользователь. Эти объекты позволяют моделировать структуру данных, а операторы — поведение данных [2].

По степени распределённости:

* локальные (все части локальной СУБД размещаются на одном компьютере);
* распределённые (части СУБД могут размещаться не только на одном, но на двух и более компьютерах).

По способу доступа к БД:

* файл-серверные;

В файл-серверных СУБД файлы данных располагаются централизованно на файл-сервере. СУБД располагается на каждом клиентском компьютере (рабочей станции). Доступ СУБД к данным осуществляется через локальную сеть. Синхронизация чтений и обновлений осуществляется посредством файловых блокировок.

Преимуществом этой архитектуры является низкая нагрузка на процессор файлового сервера.

Недостатки: потенциально высокая загрузка локальной сети; затруднённость или невозможность централизованного управления; затруднённость или невозможность обеспечения таких важных характеристик, как высокая надёжность, высокая доступность и высокая безопасность. Применяются чаще всего в локальных приложениях, которые используют функции управления БД; в системах с низкой интенсивностью обработки данных и низкими пиковыми нагрузками на БД.

На данный момент файл-серверная технология считается устаревшей, а её использование в крупных информационных системах — недостатком [3].

Пример: Microsoft Access.

* клиент-серверные;

Клиент-серверная СУБД располагается на сервере вместе с БД и осуществляет доступ к БД непосредственно, в монопольном режиме. Все клиентские запросы на обработку данных обрабатываются клиент-серверной СУБД централизованно.

Недостаток клиент-серверных СУБД состоит в повышенных требованиях к серверу.

Достоинства: потенциально более низкая загрузка локальной сети; удобство централизованного управления; удобство обеспечения таких важных характеристик, как высокая надёжность, высокая доступность и высокая безопасность.

Примеры: Oracle Database, MS SQL Server, PostgreSQL, MySQL.

* встраиваемые.

СУБД, которая может поставляться как составная часть некоторого программного продукта, не требуя процедуры самостоятельной установки. Встраиваемая СУБД предназначена для локального хранения данных своего приложения и не рассчитана на коллективное использование в сети.

Физически встраиваемая СУБД чаще всего реализована в виде подключаемой библиотеки. Доступ к данным со стороны приложения может происходить через SQL либо через специальные программные интерфейсы.

Примеры: SQLite, Microsoft SQL Server Compact.

**1.5.3 Обзор наиболее популярных СУБД**

В данном подразделе будут рассмотрены популярные СУБД, которые могут быть использованы для реализации хранения данных в разрабатываемом программном продукте.

1. **PostgreSQL**

Свободная объектно-реляционная система управления базами данных (СУБД).

Сильными сторонами PostgreSQL считаются:

* высокопроизводительные и надёжные механизмы транзакций и репликации;
* расширяемая система встроенных языков программирования: в стандартной поставке поддерживаются PL/pgSQL, PL/Perl, PL/Python и PL/Tcl; дополнительно можно использовать PL/Java, PL/PHP, PL/Py, PL/R, PL/Ruby, PL/Scheme, PL/sh и PL/V8, а также имеется поддержка загрузки модулей расширения на языке C [4];
* наследование;
* возможность индексирования геометрических (в частности, географических) объектов и наличие базирующегося на ней расширения PostGIS;
* встроенная поддержка слабоструктурированных данных в формате JSON с возможностью их индексации;
* расширяемость (возможность создавать новые типы данных, типы индексов, языки программирования, модули расширения, подключать любые внешние источники данных).

1. **Microsoft SQL Server**

Система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка.

СУБД поставляется в различных редакциях, ориентированных на различные сценарии разработки и развертывания приложений (а также отличающиеся ценой): Enterprise, Standard, Web, Business intelligence, Workgroup, Express.

1. **Oracle Database**

Объектно-реляционная система управления базами данных компании Oracle.

СУБД поставляется в шести различных редакциях, ориентированных на различные сценарии разработки и развертывания приложений (а также отличающиеся ценой):

* Enterprise Edition;
* Standard Edition – не может устанавливаться на системы, имеющие более 4 процессорных разъёмов;
* Standard Edition One – не может устанавливаться на системы, имеющие более 2 процессорных разъёмов; не поддерживает кластеризацию (RAC);
* Personal Edition – один пользователь;
* Lite – для мобильных и встраиваемых устройств;
* Express Edition (XE) – бесплатная редакция, используемая оперативная память — 1 ГБ, а также используется только 1 процессор, максимальный объём базы данных — 11 ГБ (для 10g — 4ГБ), из них от 0,5 до 0,9 ГБ используются словарём данных, внутренними схемами и временным дисковым пространством. В 18c используется 2 процессора, оперативная память — 2 ГБ, максимальный объём базы данных — 12 ГБ.

**Выводы**

В этом разделе была проанализирована поставленная задача. А также были формализованы данные и типы пользователей и рассмотрены разные типы СУБД. Для данной работы выбрана реляционная, клиент-серверная СУБД PostgreSQL, так как она является свободно распространяемой.

**Список литературы**

1. ISO/IEC TR 10032:2003 Information technology – Reference model of data management.
2. Дейт К. Дж. Введение в системы баз данных. – 8-е изд. – М.: «Вильямс», 2006.
3. Еленев Д.В. и др. Автоматизация системы управления национальным исследовательским университетом и мониторинга его деятельности // Программные продукты и системы, №3, 2012.
4. PostgreSQL: Documentation: 11: Procedural Languages. [Электронный ресурс]. Режим доступа: <https://www.postgresql.org/docs/11/xplang.html>, свободный (дата обращения: 28.04.2022).