Проектирование и реализация автоматизированной системы для организации мероприятий

Весь код и всё выполненное можно найти в репозитории:

https://github.com/Ultimatereo/YaProfiPI

1. Введение

Автоматизированная система для организации мероприятий предназначена для упрощения процесса планирования и проведения различных событий (конференций, фестивалей, соревнований и др.). Данное решение предоставляет веб-интерфейс, где участники могут регистрироваться в системе и записываться на интересующие их мероприятия, а организаторы (администраторы) могут управлять списком мероприятий, площадками и данными участников. Система обеспечивает хранение всей необходимой информации в базе данных и включает функции регистрации новых пользователей, регистрации на мероприятия, ведения списка площадок с их вместимостью, а также мониторинга заполненности мероприятий. В результате использование данной системы позволяет автоматизировать рутинные задачи организации мероприятий, снизить вероятность ошибок и предоставить участникам удобный сервис для участия в событиях.

2. Инфологическая модель (ER-диаграмма)

Инфологическая модель отражает предметную область системы в виде ER-диаграммы (сущность-связь), показывая основные сущности, их атрибуты и связи между ними. Модель разработана в третьей нормальной форме, что означает отсутствие избыточности данных и минимизацию дублирования за счёт корректного разделения сущностей. На диаграмме выделены первичные ключи (помечены как PK) каждой сущности и внешние ключи (FK), определяющие отношения между таблицами. Так, одна площадка может быть назначена для многих мероприятий, каждое мероприятие может иметь нескольких участников, а каждый участник может зарегистрироваться на множество мероприятий – эту связь “многие ко многим” моделирует отдельная сущность регистрации на мероприятие.

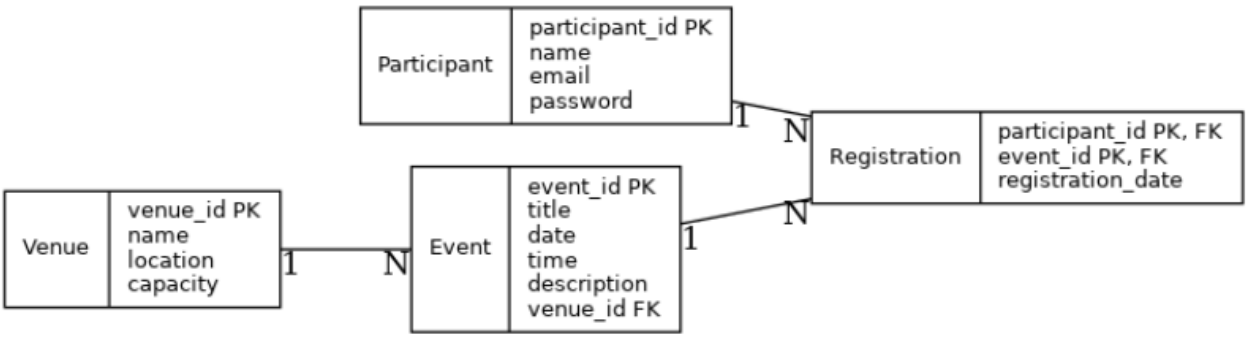

Рисунок 1: ER-диаграмма предметной области системы организации мероприятий.

На диаграмме представлены четыре сущности: **Participant** (участник), **Event** (мероприятие), **Venue** (площадка) и **Registration** (регистрация на мероприятие). У **Participant** хранится информация о пользователях системы (например, ФИО, email, пароль), у **Event** – данные о мероприятиях (название, дата, время, описание и ссылка на площадку проведения), у **Venue** – сведения о площадках/локациях (название, адрес/место и вместимость), а таблица **Registration** связывает участников и мероприятия, фиксируя факт регистрации участника на определённое событие. Первичные ключи уникально идентифицируют записи (например, participant_id для участников, event_id для мероприятий), а внешние ключи обеспечивают целостность ссылок (например, venue_id в мероприятии ссылается на площадку, а поля participant_id и event_id в регистрации ссылаются на участника и мероприятие соответственно). Связи на диаграмме показывают, что один участник может иметь много регистраций (участвовать во многих событиях), каждое событие может иметь много регистраций (множество участников), а у каждой регистрации есть ровно один связанный участник и одно мероприятие. Также каждое мероприятие проводится на одной площадке, при этом каждая площадка может использоваться для проведения многих разных мероприятий. Такая структура данных в 3НФ устраняет избыточность: информация о площадке хранится единожды в таблице **Venue**, информация об участнике – в **Participant**, а таблица **Registration** содержит только ссылки (IDs), связывающие мероприятия и участников, что исключает дублирование данных участника или площадки в записях о регистрации.

3. Даталогическая модель (модель базы данных)

Даталогическая модель представляет реализацию инфологической модели в виде схемы базы данных. Ниже описаны таблицы базы данных, их столбцы с указанием типов данных, а также первичные и внешние ключи. Все ограничения целостности соблюдаются: определены первичные ключи для

уникальной идентификации записей и внешние ключи для поддержания ссылочной целостности между таблицами.

- **Таблица Participant (Участники):** хранит данные о зарегистрированных пользователях-участниках.
  Поля:
  - participant_id – **INT**, первичный ключ (PK), уникальный идентификатор участника (например, автоинкремент).
  - name – VARCHAR(100), имя и фамилия участника.
  - email – VARCHAR(100), email участника (уникальный для входа в систему).
  - password – VARCHAR(100), пароль (в зашифрованном виде) для аутентификации участника.
- **Таблица Event (Мероприятия):** содержит информацию о проводимых мероприятиях.
  Поля:
  - event_id – **INT**, первичный ключ, уникальный идентификатор мероприятия.
  - title – VARCHAR(200), название мероприятия.
  - date – DATE, дата проведения мероприятия.
  - time – TIME, время начала мероприятия.
  - description – TEXT, описание мероприятия (программа, детали и т.д.).
  - venue_id – **INT**, внешний ключ (FK) на таблицу Venue, указывает площадку, на которой проводится мероприятие.
    Связи: У каждого мероприятия одно связанное значение venue_id, ссылающееся на запись в таблице Venue. Таким образом, устанавливается связь многие-к-одному: многие мероприятия могут проходить на одной и той же площадке.
- **Таблица Venue (Площадки):** хранит сведения о площадках, где могут проводиться мероприятия.
  Поля:
  - venue_id – **INT**, первичный ключ, уникальный идентификатор площадки.
  - name – VARCHAR(150), название площадки (например, название зала или места).
  - location – VARCHAR(200), местоположение (адрес или описание расположения).
  - capacity – INT, вместимость площадки (максимальное количество участников).
    Связи: В таблице Event поле venue_id ссылается на venue_id из Venue. У одной площадки может быть несколько связанных мероприятий, но каждое мероприятие привязано к одной площадке.

- **Таблица Registration (Регистрация на мероприятия):** фиксирует факты регистрации участников на мероприятия, реализуя связь многие-ко-многим между Participant и Event.
  Поля:
  - participant_id – **INT**, часть составного первичного ключа и внешний ключ на Participant, идентификатор участника.
  - event_id – **INT**, часть составного первичного ключа и внешний ключ на Event, идентификатор мероприятия.
  - registration_date – DATETIME, дата и время регистрации участника на мероприятие.
    Связи и ограничения: Первичный ключ составной по двум полям: (participant_id, event_id), что не позволяет одному и тому же участнику зарегистрироваться более одного раза на одно мероприятие (уникальная пара). Поля participant_id и event_id являются внешними ключами, ссылаясь соответственно на таблицы Participant и Event. Это гарантирует, что каждая запись регистрации связывает существующего в системе участника с существующим мероприятием. При удалении записей из Participant или Event каскадно или с ограничением удаления должны учитываться связанные записи в Registration для сохранения целостности данных.

Каждая таблица использует подходящие типы данных для своих атрибутов (например, строки фиксированной длины для имен, текст для описаний, числовые типы для идентификаторов и вместимости). Данная структура данных обеспечивает отсутствие избыточности: информация о площадках и участниках не дублируется в таблице мероприятий или регистраций, а связывается через ключи. Таким образом, модель соответствует третьей нормальной форме, что упрощает поддержку целостности и актуальности данных.

4. Прототипы веб-страниц

Для наглядного представления функционирования системы были разработаны прототипы основных веб-страниц. Эти прототипы отражают предполагаемый интерфейс системы: страницы для пользователей-участников и для администратора. Ниже приведены ключевые экраны с описанием их элементов и функциональности.

Главная страница со списком мероприятий

Главная страница предназначена для отображения всем посетителям перечня предстоящих мероприятий. Список содержит название каждого мероприятия и дату проведения. Рядом с каждым событием доступна кнопка

«Регистрация», позволяющая заинтересованному пользователю перейти к оформлению участия в данном мероприятии. Таким образом, с главной страницы пользователь сразу видит анонсы событий и может выбрать нужное мероприятие для регистрации.

Форма регистрации участника

Страница «Регистрация участника» предоставляет форму для создания новой учетной записи в системе. Пользователь вводит необходимые данные: имя, адрес электронной почты и пароль. После заполнения формы и нажатия кнопки «Зарегистрироваться» новый пользователь сохраняется в системе, и ему предоставляется доступ к личному кабинету и возможности регистрации на мероприятия. Данный интерфейс упрощает процесс присоединения новых участников к системе, обеспечивая сбор необходимой информации для их идентификации и последующего входа.

Форма регистрации на мероприятие

Данная форма позволяет участнику записаться на определённое мероприятие. Если пользователь выбрал мероприятие на главной странице, оно будет отображено в поле выбора мероприятия. Участнику достаточно подтвердить свой выбор, нажав кнопку «Записаться». После этого система зарегистрирует данного участника как участника выбранного мероприятия (добавив запись в таблицу Registration). В случае, если у пользователя есть несколько вариантов событий, он может выбрать нужное мероприятие из выпадающего списка. Интерфейс простой и понятный: он выводит название выбранного события и предоставляет элемент управления (список или подтверждение) для подачи заявки на участие.

Личный кабинет участника

Личный кабинет доступен после входа пользователя в систему. На этой странице отображается персональная информация участника (например, имя и email) и список мероприятий, на которые он зарегистрирован. Пользователь может просматривать, на какие события он успешно записался, что помогает ему отслеживать свои текущие и предстоящие участия. В дальнейшем функционал личного кабинета может быть расширен возможностью редактирования личных данных или отказа от участия в мероприятии, но в базовом варианте он служит информативной панели для пользователя.

Панель администратора (мероприятия, площадки, участники)

Администраторская панель предназначена для пользователей-организаторов, отвечающих за наполнение и ведение системы. В верхней части интерфейса представлены разделы: «Мероприятия», «Площадки» и «Участники» – для перехода к соответствующим спискам. На изображении показан раздел **«Мероприятия»**, где администратор видит таблицу всех мероприятий с основными данными: названием, датой, площадкой и заполненностью (процент заполнения площадки участниками). Для каждого мероприятия доступны действия: редактирование (например, изменение деталей или количества мест) и удаление мероприятия. Ниже списка предусмотрена кнопка «+ Добавить мероприятие» для создания нового события. Аналогично, в разделе **«Площадки»** администратор управляет списком доступных мест проведения (добавляет новые площадки, указывает их вместимость и местоположение), а в разделе **«Участники»** – просматривает и при необходимости удаляет или блокирует учетные записи участников. Таким образом, админ-панель предоставляет полный набор инструментов для поддержки актуальности данных системы и контроля за ходом регистрации на мероприятия.

5. Алгоритм расчёта заполненности площадки

Заполненность площадки – это показатель, отражающий отношение числа зарегистрированных участников мероприятия к максимальной вместимости площадки, на которой оно проводится. Данный показатель важен для организаторов, чтобы понимать степень заполнения зала и, при необходимости, ограничивать дальнейшую регистрацию или менять зал. Формально заполненность можно определить как долю:

$$\text{Заполненность} = \frac{\text{Количество зарегистрированных пользователей}}{\text{Вместимость площадки}}$$

Часто результат выражается в процентах: например, если зарегистрировано 50 человек при вместимости зала 100, заполненность составляет 50%. Алгоритм расчёта может быть реализован как на стороне базы данных (через SQL-запрос с агрегатной функцией COUNT), так и на уровне бизнес-логики приложения.

6. Описание SQL-запросов для формирования отчетов

Ниже приведены два примера SQL-запросов, обеспечивающих вывод необходимой информации для двух форм системы:

Запрос списка мероприятий

Задача запроса:
Получить список предстоящих мероприятий с отображением ключевой информации:

- Название и описание мероприятия
- Дата проведения
- Данные о площадке (название, адрес)
- Количество свободных мест (рассчитано на основе вместимости площадки и числа уже зарегистрированных участников)
- Возрастной ценз

Пример запроса:

```
SELECT
    e.event_id,
    e.event_name,
    e.description,
    e.event_date,
    v.venue_name,
    v.address,
    (v.capacity - COUNT(r.participant_id)) AS free_seats,
    e.age_limit
FROM events e
JOIN venues v ON e.venue_id = v.venue_id
LEFT JOIN registrations r ON e.event_id = r.event_id
WHERE e.event_date >= CURRENT_DATE
GROUP BY
    e.event_id,
    e.event_name,
    e.description,
    e.event_date,
    v.venue_name,
    v.address,
    v.capacity,
    e.age_limit
ORDER BY e.event_date ASC;
```

Пояснение:

- **Соединение таблиц:** Таблица events объединяется с таблицей venues по venue_id, что позволяет получить сведения о площадке, где проводится мероприятие.

- **Подсчёт регистраций:** LEFT JOIN с таблицей registrations обеспечивает подсчёт участников. Функция COUNT(r.participant_id) возвращает число зарегистрированных на конкретное мероприятие.
- **Расчёт свободных мест:** Вычисляется как разница между вместимостью площадки (v.capacity) и количеством зарегистрированных участников.
- **Фильтрация:** Условие WHERE e.event_date >= CURRENT_DATE гарантирует выборку только предстоящих мероприятий.
- **Группировка и сортировка:** Группировка по идентификаторам и сортировка по дате позволяет корректно агрегировать данные и выводить мероприятия в хронологическом порядке.

Запрос для регистрации на мероприятие

Задача запроса:
Вставить новую запись регистрации в таблицу registrations, но только если для данного мероприятия ещё доступны места. Предотвращается ситуация, когда количество участников превышает вместимость площадки.

Пример запроса с условием:

```
INSERT INTO registrations (event_id, participant_id, registration_date)
SELECT
   :event_id,
   :participant_id,
   CURRENT_TIMESTAMP
WHERE (
   SELECT COUNT(*)
   FROM registrations
   WHERE event_id = :event_id
) < (
   SELECT v.capacity
   FROM venues v
   JOIN events e ON v.venue_id = e.venue_id
   WHERE e.event_id = :event_id
);
```

Пояснение:

- **Использование параметров:** В запросе применяются параметры :event_id и :participant_id, которые подставляются при выполнении запроса.
- **Подзапрос для подсчёта регистраций:** Первый подзапрос определяет текущее число записей в таблице registrations для выбранного мероприятия.

- **Подзапрос для получения вместимости:** Второй подзапрос выполняет соединение таблиц venues и events, возвращая вместимость площадки для данного мероприятия.
- **Условие вставки:** Оператор WHERE сравнивает текущий подсчитанный показатель с вместимостью. Если количество зарегистрированных участников меньше вместимости, условие истинно, и вставка выполняется. Иначе запись не добавляется, что предотвращает регистрацию сверх лимита.

Замечание: Подобное условное добавление записи можно также реализовать через триггеры или посредством логики приложения, но приведённый запрос демонстрирует возможность встроенной проверки на стороне СУБД.

7. Заключение

В ходе проекта было спроектировано и реализовано решение для автоматизации организации мероприятий. Разработана инфологическая модель предметной области в виде ER-диаграммы, удовлетворяющая требованиям третьей нормальной формы, что обеспечивает целостность и отсутствие избыточности данных. На основе инфологической модели создана даталогическая модель – структура реляционной базы данных с четко определёнными таблицами, полями и связями (первичными и внешними ключами). Для наглядности предложены прототипы веб-интерфейса, демонстрирующие работу системы с точки зрения участника (регистрация в системе, запись на мероприятия, личный кабинет) и администратора (управление мероприятиями, площадками, участниками). Алгоритм расчёта заполненности площадки был описан и проиллюстрирован примерами, а также предложены SQL-запросы для ключевых операций – получения списка мероприятий и регистрации участников. Предложенная система позволяет значительно упростить процесс управления событиями: участники могут легко находить и бронировать интересующие их мероприятия, а организаторы имеют все необходимые инструменты для контроля и ведения событийного процесса.