Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей
Кафедра информатики
Дисциплина: Модели данных и системы управления базами данных

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовой работе
на тему

ПРОГРАММНОЕ СРЕДСТВО ДЛЯ ОНЛАЙН-РЕГИСТРАЦИИ НА МЕРОПРИЯТИЯ И УПРАВЛЕНИЯ ИМИ

БГУИР КП 1-40 04 01 011 ПЗ

Студент В. В. Андрейчук

Руководитель А. В. Давыдчик

ВВЕДЕНИЕ

В современном цифровом обществе использование информационных технологий для организации и управления различными мероприятиями стало неотъемлемой частью эффективного планирования. Компании, образовательные учреждения и частные лица всё чаще стремятся организовать и автоматизировать процессы, связанные с регистрацией участников, сбором данных и управлением событиями, что повышает эффективность и снижает затраты на ручной труд. В данном контексте создание программного средства для онлайн-регистрации на мероприятия и управления ими является актуальной задачей, решающей множество практических вопросов и предлагающей пользователям удобный и доступный способ взаимодействия.

Цель курсовой работы – разработать программное средство, позволяющее организаторам мероприятий создавать электронные формы для регистрации, управлять базами данных участников, а также осуществлять оперативное взаимодействие с участниками. Основная задача – построить надежную и гибкую систему управления базами данных (СУБД), которая будет поддерживать хранение и обработку данных о мероприятиях и пользователях. Созданное приложение будет обладать интерфейсом, обеспечивающим интуитивное взаимодействие для организаторов, возможность настроек и управления событиями в режиме онлайн.

Ключевые компоненты предлагаемого решения включают систему регистрации участников, модуль для управления мероприятиями, а также механизмы отслеживания. В основе системы лежат таблицы для хранения информации о пользователях, мероприятиях, расписаниях, билетах, ролях, категориях и платежах, что обеспечивает гибкое управление данными и удобное взаимодействие с пользователями. Для надёжного хранения изображений и управления отзывами участников о мероприятиях предусмотрены соответствующие таблицы. Использование реляционной системы управления базами данных (СУБД) гарантирует целостность данных и высокую производительность приложения.

1 АРХИТЕКТУРА ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ

1.1 Структура и архитектура вычислительной системы

PostgreSQL является одной из наиболее популярных систем управления базами данных. Сам проект postgresql эволюционировал из другого проекта, который назывался Ingres. Формально развитие postgresql началось еще в 1986 году. Тогда он назывался POSTGRES. А в 1996 году проект был переименован в PostgreSQL, что отражало больший акцент на SQL. И собственно 8 июля 1996 года состоялся первый релиз продукта. [1]

PostgreSQL – это мощная объектно-реляционная система управления базами данных с открытым исходным кодом, которая активно используется для создания сложных и высоконагруженных приложений. Благодаря своим функциям и гибкости PostgreSQL является одной из самых популярных систем управления базами данных (СУБД) среди разработчиков и администраторов, обеспечивая высокую производительность, надежность и поддержку множества стандартов SQL. Она поддерживает транзакции, расширенные типы данных, и целостность на уровне ACID (атомарность, согласованность, изолированность и долговечность), что делает её идеальной для приложений, в которых требуется гарантия сохранности данных при проведении большого количества операций.

Одним из главных преимуществ PostgreSQL является поддержка транзакций и механизмов контроля за параллельными операциями. Система использует механизм многоверсийности (MVCC), который позволяет обрабатывать несколько транзакций одновременно, предотвращая блокировки чтения и записи. Это существенно улучшает производительность в многопользовательской среде и позволяет приложениям оставаться отзывчивыми при интенсивных операциях с базой данных.

PostgreSQL поддерживает продвинутую систему индексации, которая включает традиционные B-деревья, а также специализированные индексы, такие как GiST, GIN, BRIN, SP-GiST и другие. Эти индексы позволяют оптимизировать работу запросов, особенно в приложениях, где требуется сложный поиск и фильтрация данных. Например, GIN и GiST индексы оптимальны для полнотекстового поиска, что может быть

полезным в случае анализа текстовых данных, таких как отзывы пользователей или описания мероприятий.

Еще одной важной особенностью PostgreSQL является ее расширяемость. Разработчики могут добавлять новые типы данных, создавать собственные функции, операторы и даже собственные языки процедурного программирования. Такая гибкость делает PostgreSQL подходящей для реализации сложной логики внутри базы данных, позволяя перенести часть вычислений из уровня приложения в саму СУБД. Например, можно создавать функции для специфичной обработки данных или автоматизации определенных бизнес-процессов.

PostgreSQL поддерживает репликацию данных и отказоустойчивую архитектуру, что делает её подходящей для создания приложений с высокими требованиями к доступности и масштабируемости. Поддержка как синхронной, так и асинхронной репликации позволяет создавать системы с балансировкой нагрузки и резервированием данных. Это помогает обеспечивать бесперебойную работу базы данных даже в условиях повышенной нагрузки или при отказе одного из серверов. Помимо этого, PostgreSQL поддерживает автоматическое резервное копирование и восстановление данных, что позволяет эффективно решать задачи восстановления базы данных при возникновении сбоев.

Система безопасности PostgreSQL также предоставляет широкий выбор инструментов, таких как контроль доступа на уровне строк, политика доступа на уровне таблиц, а также механизмы шифрования данных. Управление доступом осуществляется с помощью ролей и разрешений, что позволяет создавать гибкие и безопасные структуры для управления правами пользователей.

1.2 Истории, версии и достоинства

PostgreSQL ведет свою историю с 1986 года, когда Майкл Стоунбрейкер из Калифорнийского университета в Беркли начал разработку системы управления базами данных под названием POSTGRES, которая позже стала основой для PostgreSQL. Основной целью было создание системы, поддерживающей работу с нестандартными типами данных и более сложными функциями, чем традиционные реляционные базы данных. В 1996 году проект переименовали в PostgreSQL после добавления полной поддержки SQL, ставшей стандартом взаимодействия с базами данных. С этого момента

PostgreSQL развивалась как одна из ведущих СУБД с открытым исходным кодом.

Основные версии PostgreSQL:

– PostgreSQL 6.0 (1997): первая версия с поддержкой SQL, обеспечивает совместимость с языком стандартного запроса данных SQL и давшая старт современной системе.

– PostgreSQL 8.0 (2005): добавлена поддержка Windows и улучшена репликация, что расширило область применения PostgreSQL в корпоративных средах.

– PostgreSQL 9.0 (2010): введены функции Hot Standby и Streaming Replication, которые позволили создавать отказоустойчивые системы с использованием асинхронной репликации.

– PostgreSQL 10 (2017): улучшена поддержка параллельных операций и добавлена логическая репликация для более гибкой настройки репликационных процессов.

– PostgreSQL 12 и последующие версии: добавлены улучшения по производительности и безопасности, такие как поддержка JSONB и расширенные возможности для работы с массивами и геоданными.

Одним из главных достоинств PostgreSQL является её высокая производительность и оптимизация для параллельной обработки запросов, что делает её эффективной для сложных приложений с высокими требованиями к скорости обработки данных. Она поддерживает многоуровневую индексацию, включая B-деревья, GiST и GIN индексы, что позволяет ускорить запросы для разнообразных типов данных и улучшает производительность даже в условиях больших объемов данных.

В дополнение к этому PostgreSQL предоставляет расширенные возможности для настройки производительности, такие как планировщики запросов, мониторинг нагрузки и инструменты для управления буферной памятью.

Расширяемость PostgreSQL предоставляет разработчикам широкие возможности для создания специализированных функций и добавления собственных типов данных, операторов и даже процедурных языков. Это позволяет переносить часть вычислений внутрь базы данных, облегчая нагрузку на приложение и предоставляя дополнительную гибкость для сложной бизнес-логики. Поддержка пользовательских расширений делает PostgreSQL подходящей для разнообразных задач, от стандартного управления данными до сложных аналитических расчетов.

PostgreSQL соответствует требованиям ACID, что обеспечивает целостность и надежность данных. Благодаря поддержке ACID гарантируется, что операции всегда будут выполняться корректно, даже в случае отказов системы. Это особенно важно для приложений, где ошибки могут привести к критическим потерям данных. Помимо стандартной транзакционной изолированности и согласованности, PostgreSQL предлагает инструменты для настройки уровня изолированности транзакций, что позволяет адаптировать СУБД к требованиям конкретного приложения.

Система безопасности PostgreSQL включает функции контроля доступа на уровне строк, шифрование, политику доступа на уровне таблиц, а также инструменты для многоуровневого управления доступом. Администраторы могут точно контролировать, какой пользователь имеет доступ к каким данным и операциям, а также применять аутентификацию и шифрование, что повышает безопасность данных на всех уровнях работы с системой. Для более высокой защиты PostgreSQL поддерживает стандартные механизмы шифрования SSL для соединений и шифрование данных на уровне полей.

Поддержка разнообразных индексов и типов данных, включая JSON, XML, массивы и географические данные, делает PostgreSQL универсальным инструментом для хранения и обработки данных. Например, JSONB позволяет хранить неструктурированные данные и выполнять быстрый поиск по ним, что делает PostgreSQL отличной базой для приложений с большими объемами JSON-данных. Поддержка геоданных и индексирования с помощью GiST и SP-GiST позволяет эффективно работать с пространственными данными, применяя PostgreSQL в геоинформационных системах и приложениях для картографии.

PostgreSQL также обладает высокой масштабируемостью и поддерживает как синхронную, так и асинхронную репликацию, что позволяет создавать системы с высокой доступностью и отказоустойчивостью. В крупных распределенных системах асинхронная репликация позволяет распределить нагрузку на разные серверы, обеспечивая масштабирование без снижения производительности. В случаях, когда требуется немедленное копирование данных, синхронная репликация гарантирует, что данные на основном и резервных серверах всегда будут актуальными.

Открытый исходный код и поддержка лицензии PostgreSQL позволяют использовать ее бесплатно и без ограничений для любых типов приложений, от коммерческих до образовательных и исследовательских. Активное сообщество разработчиков и пользователей PostgreSQL постоянно совершенствует систему, добавляет новые функции и улучшает производительность, что гарантирует ее актуальность и надежность в условиях современных требований.

источники

1 https://metanit.com/sql/postgresql/1.1.php

1 АНАЛИЗ ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ И ОБЗОР СУЩЕСТВУЮЩИХ АНАЛОГОВ

1.1 Анализ литературных источников

При разработке базы данных для программного средства, предназначенного для онлайн-регистрации на мероприятия и управления ими, необходимо учитывать ряд ключевых аспектов, таких как производительность, надежность, безопасность и удобство использования.

Производительность базы данных играет важную роль, так как приложения часто обрабатывают большое количество запросов одновременно. Это особенно актуально в условиях пиковых нагрузок, когда пользователи регистрируются на мероприятия. Для оптимизации производительности следует применять индексы для часто запрашиваемых полей, таких как идентификаторы пользователей, мероприятий и дат. Кроме того, правильное проектирование структуры данных позволит значительно уменьшить время отклика системы. Например, использование специализированных таблиц для хранения информации о билетах и платежах позволяет более эффективно управлять большими объемами данных.

Надежность базы данных также является важным аспектом, так как приложение должно обеспечивать защиту данных и устойчивость к сбоям. Для этого можно использовать репликацию данных, которая создает резервные копии базы в реальном времени, что гарантирует доступность информации в случае сбоев. Регулярное резервное копирование и тестирование восстановления данных помогают минимизировать риски потери информации.

Безопасность данных пользователей становится всё более актуальной в свете современных требований к защите персональной информации. База данных должна обеспечивать защиту от несанкционированного доступа, а также иметь механизмы шифрования для хранения конфиденциальных данных, таких как пароли и финансовая информация. Использование средств аутентификации и авторизации поможет ограничить доступ к чувствительным данным только для уполномоченных пользователей.

Удобство использования является еще одним важным аспектом, который необходимо учитывать при проектировании базы данных.

Интерфейс приложения должен быть интуитивно понятным для пользователей, а сама база данных – хорошо структурированной, чтобы разработчики могли легко реализовать необходимые функции. Наличие подробной документации по работе с базой данных также значительно упрощает ее использование и дальнейшую поддержку.

Масштабируемость системы имеет значение для обеспечения ее способности адаптироваться к увеличению числа пользователей и объема данных. При проектировании базы данных следует учитывать возможность ее масштабирования как вертикально (увеличение мощности серверов), так и горизонтально (добавление новых серверов для распределения нагрузки). Это поможет поддерживать высокую производительность системы по мере роста нагрузки.

Проектирование базы данных для системы онлайн-регистрации на мероприятия требует комплексного подхода, учитывающего производительность, надежность, безопасность и удобство использования. Эти аспекты являются основой для создания эффективного и устойчивого приложения, способного удовлетворить потребности пользователей и организаторов мероприятий.

1.2 Обзор существующих аналогов

В настоящее время существует множество программных средств, предназначенных для онлайн-регистрации на мероприятия и управления ими. Рассмотрим несколько популярных аналогов, их функциональные возможности, а также плюсы и минусы.

Eventbrite – одна из самых известных платформ для создания, управления и продвижения мероприятий. Она предлагает инструменты для регистрации участников, продажи билетов, а также интеграцию с различными платежными системами.

Плюсы этой платформы включают простоту использования. Интерфейс интуитивно понятен, что позволяет пользователям быстро освоить платформу. Многофункциональность также является важным преимуществом, так как она поддерживает широкий спектр типов мероприятий, включая концерты, конференции и вебинары. Возможность интеграции с социальными сетями, CRM-системами и другими приложениями добавляет ценности. Кроме того, платформа предоставляет

подробные отчёты о регистрации и посещаемости, что помогает организаторам анализировать успех мероприятий.

Однако есть и минусы. Существуют сборы за использование платформы, что может снизить прибыль организаторов. Ограниченная настройка шаблонов оформления мероприятий может быть недостаточно гибкой для уникальных запросов.

Cvent – это мощная платформа для управления мероприятиями, ориентированная на корпоративный сектор. Она предлагает комплексное решение для планирования, управления и анализа мероприятий.

К преимуществам Cvent можно отнести широкий набор инструментов для управления мероприятиями, включая управление бюджетом и аналитические отчеты. Платформа подходит как для крупных конференций, так и для небольших встреч. Кроме того, она предлагает продвинутые функции для управления приглашениями и RSVP.

Среди недостатков можно выделить сложность интерфейса, который может быть слишком сложным для небольших организаторов, которые хотят простое решение. Высокая стоимость подписки делает платформу менее доступной для малых предприятий и индивидуальных организаторов.

Whova – это приложение для управления мероприятиями, которое фокусируется на взаимодействии с участниками и улучшении опыта их участия.

К плюсам этой платформы относятся интерактивные инструменты для взаимодействия с участниками, такие как чаты и опросы. Наличие мобильного приложения позволяет участникам получать актуальную информацию о мероприятии на ходу. Возможность делиться контентом мероприятия в социальных сетях также является значительным преимуществом.

Тем не менее, некоторые функции доступны только в платной версии, что может ограничивать пользователей. Кроме того, кривая обучения может потребовать времени для изучения всех возможностей приложения.

Ticket Tailor – это простая в использовании платформа для продажи билетов на мероприятия, ориентированная на малый и средний бизнес.

Преимущества Ticket Tailor включают отсутствие скрытых сборов, что позволяет организаторам сохранять большую часть прибыли от продаж

билетов. Гибкость настройки страниц мероприятий под индивидуальные нужды является еще одним значительным плюсом.

Однако следует отметить ограниченный функционал. Не все функции, такие как управление мероприятиями и аналитика, представлены на уровне крупных платформ. Меньшая известность Ticket Tailor может повлиять на доверие пользователей по сравнению с более распространенными решениями.

Сравнение существующих аналогов показывает, что каждая платформа имеет свои уникальные преимущества и недостатки. При разработке собственного программного средства для онлайн-регистрации на мероприятия и управления ими важно учесть как успешные решения, так и их ограничения. Это позволит создать более адаптированное и эффективное решение, способное удовлетворить потребности пользователей и организаторов мероприятий.

2 ФОРМИРОВАНИЕ ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ И ВЫБОР ИНСТРУМЕНТОВ РАЗРАБОТКИ

2.1 Формирование функциональных требований

При разработке программного обеспечения для онлайн-регистрации на мероприятия и управления ими необходимо решить ряд задач, которые обеспечивают эффективность и удобство использования приложения как для участников, так и для организаторов. Основные функциональные требования и задачи, которые должно решать приложение, можно разделить на несколько категорий.

Первой задачей является создание системы регистрации пользователей. Пользователи должны иметь возможность создавать свои аккаунты, вводя такие данные, как имя, адрес электронной почты и пароль. Это позволит обеспечить уникальность каждого пользователя и возможность аутентификации.

В дополнение к регистрации важно предусмотреть функции управления профилем, такие как изменение пароля, обновление контактной информации и просмотр истории покупок. Для этой задачи необходимо хранить данные о пользователях в таблице "User", где будут содержаться поля для имени, электронной почты, пароля и идентификатора роли (например, участник или организатор).

Следующей ключевой задачей является возможность создания, редактирования и удаления мероприятий организаторами. Каждое мероприятие должно иметь уникальные атрибуты, такие как название, описание, дата и время проведения, место, стоимость билетов и изображение. Для управления мероприятиями данные будут храниться в таблице "Event", которая будет связана с таблицей "EventSchedule" для учета расписания и доступных временных промежутков.

Организаторы должны также иметь возможность добавлять категории к мероприятиям для упрощения их поиска и фильтрации. Для этого используется таблица "Category", связанная с таблицей "EventCategories", что позволяет устанавливать связь между мероприятиями и их категориями.

Третьей важной задачей является разработка системы продажи билетов. Приложение должно позволять пользователям просматривать доступные мероприятия и приобретать билеты через интегрированные

платежные системы. Необходимо предусмотреть разные типы билетов (например, стандартные, VIP), а также возможность выбора количества.

Для реализации этой функциональности потребуется создать таблицу "Ticket", которая будет хранить информацию о каждом проданном билете, включая уникальный код, идентификатор мероприятия и пользователя. Связь с таблицей "EventSchedule" позволит организовать данные о времени и месте проведения мероприятия, а также учесть возможные изменения в расписании.

Четвертая задача связана с управлением отзывами пользователей о мероприятиях. Пользователи должны иметь возможность оставлять свои комментарии и оценки, которые будут храниться в таблице "Review". Это поможет организаторам получать ценную обратную связь и улучшать качество своих мероприятий на основе мнений участников.

Система должна поддерживать функционал модерации отзывов, чтобы предотвратить появление недостоверной или неприемлемой информации. Удобный интерфейс для просмотра и анализа отзывов будет способствовать улучшению качества мероприятий.

Приложение должно поддерживать важные бизнес-процессы, включая регистрацию пользователей, управление мероприятиями, продажу билетов и сбор отзывов. Для эффективного выполнения этих процессов необходимо хранить следующие данные:

– Информация о пользователях (таблица "User") с данными о профилях, ролях и аутентификации.

– Данные о мероприятиях (таблица "Event"), включая название, описание, дату, время, место и организатора.

– Информация о билетах (таблица "Ticket"), содержащая уникальные коды и информацию о связанных мероприятиях.

– Данные отзывов (таблица "Review"), позволяющие пользователям оценивать мероприятия и оставлять комментарии.

– Записи об активности пользователей (таблица "ActivityLog"), помогающие отслеживать действия пользователей в системе.

Пользователи должны иметь возможность выполнять различные операции с данными. Участники смогут:

– Регистрироваться и создавать личные профили.

– Просматривать список доступных мероприятий с фильтрацией по категориям и дате.

– Приобретать билеты на выбранные мероприятия и получать подтверждения.

– Оставлять отзывы и оценки о мероприятиях, которые они посетили.

– Получать уведомления о предстоящих мероприятиях и изменениях.

Организаторы смогут:

– Создавать и редактировать мероприятия с указанием всех необходимых атрибутов.

– Управлять продажами билетов, просматривать статистику и получать отчёты.

– Модерировать отзывы и взаимодействовать с участниками через комментарии.

– Настраивать уведомления для своих участников.

Нормализация базы данных является важным аспектом проектирования, поскольку она способствует уменьшению избыточности данных и обеспечению их целостности.

Нормализация базы данных представляет собой процесс организации данных с целью уменьшения их избыточности и повышения целостности. Этот процесс включает структурирование данных и создание таблиц, которые минимизируют дублирование информации и предотвращают аномалии при вставке, обновлении и удалении данных. Нормализация обеспечивает логичное распределение данных и облегчает их управление.

Основные цели нормализации заключаются в сокращении избыточности данных, обеспечении целостности и упрощении структуры базы данных. Существует несколько нормальных форм, каждая из которых имеет свои критерии и условия.

Первая нормальная форма (1NF) требует, чтобы все поля таблицы содержали атомарные, или неделимые, значения. Это означает, что каждое поле должно хранить только одно значение, и строки должны быть уникальными. Например, если в таблице содержатся данные о пользователях, нельзя иметь поле с множественными значениями, такими как списки телефонов. Каждое значение должно быть представлено в отдельной строке.

Вторая нормальная форма (2NF) требует, чтобы таблица соответствовала первой нормальной форме и все неключевые атрибуты были полностью функционально зависимы от первичного ключа. Это

означает, что каждый неключевой атрибут должен зависеть от всего первичного ключа, а не только от его части. Например, если у вас есть таблица с данными о заказах, и первичный ключ состоит из нескольких полей, все остальные поля должны зависеть от комбинации всех этих полей, а не только от одного из них.

Третья нормальная форма (3NF) требует, чтобы таблица соответствовала второй нормальной форме и все неключевые атрибуты были независимы друг от друга. Это значит, что данные, содержащиеся в таблице, не должны зависеть от других неключевых атрибутов. Например, если в таблице о заказах есть поле с именем клиента и поле с его адресом, адрес не должен зависеть от имени клиента, так как это создает зависимость между неключевыми полями.

Каждая из нормальных форм направлена на повышение качества структуры базы данных и обеспечение ее надежности. Процесс нормализации может продолжаться и далее, охватывая более высокие нормальные формы, такие как Бойс-Кодд нормальная форма (BCNF) и четвертая нормальная форма (4NF), которые решают более специфические проблемы, связанные с зависимостями между данными. Нормализация является важным этапом проектирования баз данных, так как она способствует созданию эффективных и легко управляемых систем хранения данных.

В данной базе данных применены различные нормальные формы, что позволяет логически организовать данные в соответствующих таблицах. Например, информация о пользователях хранится в таблице "User", а данные о мероприятиях и их расписании распределены между таблицами "Event" и "EventSchedule".

Данный подход позволяет минимизировать дублирование информации, улучшить производительность запросов и упростить управление данными. Каждая таблица связана с другими через внешние ключи, что обеспечивает целостность и непротиворечивость данных, а также упрощает выполнение сложных запросов для получения необходимой информации.

Таким образом, приложение должно решать множество задач, связанных с регистрацией пользователей, управлением мероприятиями, продажей билетов и получением отзывов. Хранение соответствующих данных и поддержка бизнес-процессов в системе обеспечат её

функциональность и удобство использования для всех категорий пользователей.