1.5 Разграничение прав доступа, создание ролей и пользователей

1) ER-модель (Entity-Relationship model) — это модель данных, которая используется для описания структуры базы данных с точки зрения сущностей (entities) и их взаимосвязей (relationships). ER-модель позволяет определить сущности, их атрибуты и связи между этими сущностями.

В PostgreSQL ER-модель представляется в виде диаграммы, где сущности изображаются в виде прямоугольников с названиями таблиц, а атрибуты указываются внутри каждого прямоугольника. Связи между сущностями обозначаются стрелками, которые указывают на тип отношений (один-к-одному, один-ко-многим, многие-ко-многим).

ER-модель помогает разработчикам понять структуру базы данных, определить правильные отношения между таблицами, а также создать оптимальные запросы для извлечения данных. В PostgreSQL могут быть использованы дополнительные инструменты для создания ER-модели, такие как pgModeler или ER/Studio.

ER-модель является важным инструментом при проектировании баз данных в PostgreSQL, поскольку позволяет сделать структуру данных более прозрачной, понятной и легко изменяемой.

2) Логическая архитектура баз данных PostgreSql представляет собой модель данных, которая описывает структуру и отношения между данными в базе данных. В PostgreSql данные организованы в виде таблиц, которые содержат столбцы с определенными типами данных.

В базе данных PostgreSql используется SQL (Structured Query Language) для работы с данными. SQL позволяет выполнять различные операции над данными, такие как добавление, удаление, изменение и извлечение данных.

Логическая архитектура баз данных PostgreSql также включает в себя ключевые понятия, такие как индексы, ограничения, хранимые процедуры и триггеры. Индексы позволяют ускорить выполнение запросов к базе данных, ограничения обеспечивают целостность данных, хранимые процедуры позволяют выполнять сложные операции над данными, а триггеры позволяют автоматизировать выполнение определенных действий при изменении данных.

Логическая архитектура баз данных PostgreSql также поддерживает транзакции, что обеспечивает целостность и надежность данных. Транзакция представляет собой последовательность операций, которые либо полностью выполнены, либо отменены в случае ошибки.

В целом, логическая архитектура баз данных PostgreSql обеспечивает эффективное и надежное хранение и обработку данных, что делает эту систему удобной и популярной среди разработчиков и администраторов баз данных.

3) PostgreSQL - это объектно-реляционная система управления базами данных (СУБД), которая использует мощный язык запросов SQL для работы с данными. Физическая архитектура баз данных PostgreSQL состоит из нескольких основных компонентов:

1. Сегменты данных: PostgreSQL хранит данные в различных сегментах данных, таких как таблицы, индексы, представления и т. д. Каждый сегмент данных может занимать отдельное пространство на диске.
2. Файловая система: PostgreSQL использует файловую систему для хранения данных и журналов транзакций. Это позволяет эффективно управлять доступом к данным и обеспечивать согласованное чтение и запись.
3. Процессы: В PostgreSQL существует несколько типов процессов, включая процессы сервера, процессы фоновой работы, процессы резервного копирования и т. д. Каждый тип процесса выполняет определенные задачи для обеспечения надежности и производительности СУБД.
4. Буферный кэш: PostgreSQL использует буферный кэш для кеширования часто используемых данных, чтобы ускорить доступ к ним. Буферный кэш хранится в оперативной памяти и содержит данные из таблиц, индексов и других объектов базы данных.
5. Журналы транзакций: PostgreSQL использует журналы транзакций для обеспечения целостности данных и восстановления после сбоев. Журналы записывают все изменения данных перед их фактическим применением к базе данных, что позволяет восстанавливать состояние базы данных до любой точки в прошлом.

Физическая архитектура баз данных PostgreSQL разработана с учетом высокой производительности, надежности и масштабируемости. Благодаря использованию современных технологий и методов хранения данных, PostgreSQL способна обрабатывать большие объемы информации и поддерживать высокую доступность сервисов.

1.6 Разграничение прав доступа, создание ролей и пользователей

При выборе средства разработки для курсового проекта было решено использовать СУБД PostgreSQL в виду следующих причин:

1. Открытый исходный код: PostgreSQL является open source решением, что позволяет бесплатно использовать его и иметь доступ к исходному коду для улучшения и расширения функционала.

2. Надежность: PostgreSQL известен своей высокой стабильностью и надежностью. Он обеспечивает целостность данных и обеспечивает защиту от сбоев.

3. Мощные функциональные возможности: PostgreSQL поддерживает множество передовых функций, таких как поддержка различных типов данных, поддержка транзакций, хранилище процедур, многопоточность и многие другие.

4. Большое сообщество пользователей: PostgreSQL имеет крупное сообщество пользователей и разработчиков, что означает наличие большого количества документации, форумов, обучающих материалов и готовых решений.

5. Совместимость с различными операционными системами: PostgreSQL поддерживает большинство популярных операционных систем, включая Linux, Windows, macOS, Unix и другие.

Исходя из перечисленных причин, использование PostgreSQL в качестве СУБД для курсового проекта является оптимальным выбором, который обеспечивает надежность, гибкость и удобство в разработке и поддержке проекта.

2.6 Разграничение прав доступа, создание ролей и пользователей

Предусмотрено разграничение прав доступа и ролей.

Таблица … – роли и права

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Роль | Пользователь в БД | Функции в приложении | Справочные таблицы | | | | | | Операционные таблицы | | | | | | Представления | Хранимые функции и процедуры | Триггеры |
| Услуги | Партнеры | Клиенты | Номера | Корпуса | Сотрудники | Лист услуг | Лист жалоб | История проживания | Бюджет | Бронь | График |
| Администратор базы данных | admin | Редактирует всю информацию | all | all | all | all | all | all | all | all | all | all | all | all |  |  |  |
| Менеджер | manager | Просмотр и редактирование клиентов и номеров | - | all | all | all | s | - | s | s | s | - | all | s |  |  |  |
| Директор | director | Просматривает всю информацию, формирует отчеты | s | s | s | s | s | s | s | s | s | s | s | s |  |  |  |

Окно авторизации (все роли)

Окно бронирования (manager-all, director - s)

Окно заселения (manager-all, director - s)