Министерство образования и науки Российской Федерации

Национальный исследовательский университет «МИЭТ»

Институт системной и программной инженерии и

информационных технологий

Системы управления базами данных

Лабораторная работа №4

Вариант 1

Выполнил студент группы ПИН-31Д: Ельяшевич Н.А.

Проверил: Киселев Д.В.

Зеленоград, 2024 г.

**Ответы на контрольные вопросы:**

* **Из каких компонентов состоит инфологическая модель предметной области?**

Инфологическая модель предметной области включает в себя следующие компоненты:

1. Сущности (Entity) – описывают основные объекты предметной области (например, таблицы в базе данных).

2. Атрибуты (Attributes) – описывают характеристики сущностей.

3. Связи (Relationships) – описывают взаимосвязи между сущностями.

4. Ключи (Keys) – определяют уникальность записей в таблицах.

5. Ограничения (Constraints) – определяют правила целостности данных.

Инфологическая модель предметной области помогает лучше понять структуру данных и взаимосвязи между ними перед тем, как переходить к проектированию базы данных.

* **Что представляет собой целостность базы данных и как она обеспечивается?**

Целостность базы данных - это свойство, гарантирующее точность, согласованность и надежность данных в базе. Это означает, что данные в базе хранятся и обрабатываются таким образом, что они остаются непротиворечивыми и корректными во всех аспектах.

Для обеспечения целостности базы данных применяются различные механизмы и правила, такие как:

1. Ограничения целостности (Constraints) - это правила, которые накладываются на данные в базе, чтобы гарантировать их корректность. Например, ограничения уникальности, ссылочной целостности, ограничения проверки и т. д.

2. Транзакции (Transactions) - для обеспечения целостности данных базы используются транзакции, которые представляют собой логически единые и атомарные операции над данными. Транзакции должны быть выполнены полностью или не выполнены вообще, чтобы избежать недосогласованности данных.

3. Триггеры (Triggers) - это особые хранимые процедуры в базе данных, которые автоматически выполняются при наступлении определенного события. Триггеры могут использоваться для обеспечения целостности данных.

4. Нормализация базы данных - процесс проектирования базы данных таким образом, чтобы данные хранились в оптимальной форме, что способствует целостности данных и избеганию избыточности.

В целом, целостность базы данных обеспечивается комплексом мер и правил, которые гарантируют корректное хранение, доступ и обработку данных, сохраняя их целостность и надежность.

* **Какие виды ограничений целостности существуют?**

Существует несколько видов ограничений целостности в базах данных, которые играют важную роль в обеспечении правильности данных и сохранении их целостности. Различают следующие виды ограничений целостности:

1. Ограничения уникальности (UNIQUE): Ограничивают значения определенного атрибута таким образом, чтобы каждое значение в этом атрибуте было уникальным в пределах таблицы.

2. Ограничения первичного ключа (PRIMARY KEY): Определяют уникальный идентификатор для каждой записи в таблице. Позволяют однозначно идентифицировать каждую запись.

3. Ограничения внешнего ключа (FOREIGN KEY): Обеспечивают ссылочную целостность между связанными таблицами. Проверяют, что значения в столбце одной таблицы ссылаются на существующие значения в другой таблице.

4. Ограничения проверки (CHECK): Позволяют определять условия, которым должны соответствовать значения в столбцах. Эти условия могут быть сложными логическими выражениями.

5. Ограничения NOT NULL: Гарантируют, что значение атрибута не может быть NULL (пустым).

6. Ограничения на действия (CASCADE, SET NULL и другие): Устанавливают правила поведения при удалении или обновлении данных, связанных с ключами. Например, CASCADE означает автоматическое удаление или обновление связанных записей.

Эти ограничения позволяют создавать структуру базы данных с четкими правилами, обеспечивающими целостность и консистентность данных. Корректное использование ограничений целостности способствует предотвращению ошибок, обеспечению правильности данных и повышению эффективности работы с базой данных.

* **Как обеспечить быстрый доступ к данным в проектируемой базе данных?**

Для обеспечения быстрого доступа к данным в проектируемой базе данных можно применить несколько методов и стратегий:

1. Использование индексов: Создание индексов на часто используемые столбцы позволяет оптимизировать процесс поиска данных. Индексы ускоряют выполнение запросов, так как помогают базе данных быстро находить нужные записи.

2. Оптимизация структуры базы данных: Правильное проектирование таблиц, атрибутов и связей между ними может значительно повлиять на производительность базы данных. Используйте нормализацию данных, чтобы избежать избыточности и дублирования информации.

3. Кэширование данных: Используйте кэширование, чтобы хранить результаты часто выполняемых запросов в памяти. Это позволит избежать избыточных обращений к диску и ускорит доступ к данным.

4. Партиционирование таблиц: Разделение таблиц на отдельные фрагменты (партиции) может улучшить производительность базы данных. Партиционирование позволяет уменьшить объем данных, обрабатываемых в рамках одной операции, и ускорить выполнение запросов.

5. Оптимизация запросов: Написание эффективных SQL-запросов поможет снизить нагрузку на базу данных и ускорит доступ к данным. Используйте индексы, ограничивайте выборку данных только необходимыми атрибутами и избегайте выполнения лишних операций.

6. Масштабирование: При необходимости обеспечить высокую доступность и производительность базы данных, можно использовать различные методы масштабирования, такие как вертикальное и горизонтальное масштабирование. Вертикальное масштабирование - увеличение ресурсов на одной машине, горизонтальное - распределение нагрузки на несколько серверов.

Сочетание этих подходов поможет обеспечить быстрый доступ к данным и эффективное функционирование проектируемой базы данных.

* **Что представляет собой индексный ключ?**

Индексный ключ (или ключ индекса) представляет собой структуру данных, созданную в базе данных для ускорения поиска и доступа к данным. Он позволяет эффективно организовывать данные в таблице и упрощает поиск нужных записей. Индексный ключ содержит информацию о значениях определенного столбца или набора столбцов, представленную в виде отсортированной структуры данных, которая связывает ключевые значения с физическим расположением данных в таблице.

Когда в базе данных создается индекс для определенного столбца или комбинации столбцов, информация об этих значениях организуется таким образом, что упрощается поиск конкретных записей на основе значений индексируемого столбца. Благодаря индексному ключу система базы данных может быстро определять местонахождение нужной информации, а не сканировать каждую строку в таблице. Одним из основных преимуществ использования индексов является увеличение производительности выполнения запросов к базе данных, особенно при работе с большим объемом данных.

Применение индексного ключа требует определенных компромиссов, так как создание индексов занимает дополнительное пространство в базе данных и требует поддержки для обновления индексов при изменении данных в таблице. Поэтому важно оценить, где и какие индексы необходимы для оптимизации доступа к данным, и продумать стратегию индексации для каждой таблицы в базе данных.

* **Какие виды индексов существуют?**

Существует несколько видов индексов, которые могут использоваться для оптимизации доступа к данным в базе данных. Рассмотрим основные типы индексов:

1. Индекс уникальности (Unique Index): Этот тип индекса обеспечивает уникальность значений в индексируемом столбце или комбинации столбцов. Индекс уникальности позволяет предотвратить дублирование данных и обеспечивает быстрый поиск уникальных записей.

2. Индекс-ключ (Primary Key Index): Индекс-ключ создается для столбца или набора столбцов, определенных как первичный ключ в таблице. Он обеспечивает уникальность значений ключа и ускоряет поиск данных по первичному ключу.

3. Индекс внешнего ключа (Foreign Key Index): Индекс, созданный на столбце, который используется в качестве внешнего ключа для связи с другой таблицей. Он улучшает производительность операций слияния данных и поддерживает целостность ссылочной целостности.

4. Кластеризованный индекс (Clustered Index): Этот тип индекса определяет физическое порядок данных в таблице на основе значений индексируемых столбцов. Кластеризованный индекс помогает ускорить поиск записей, поскольку данные хранятся физически упорядоченными.

5. Некластеризованный индекс (Non-Clustered Index): В отличие от кластеризованного индекса, некластеризованный индекс не определяет физический порядок данных в таблице. Он создает отдельную структуру данных для ускорения поиска и сортировки данных.

6. Полнотекстовый индекс (Full-Text Index): Индекс, предназначенный для поиска текстовой информации в больших объемах текста. Полнотекстовый индекс обеспечивает эффективный поиск по словам и фразам в текстовых полях.

Это лишь некоторые из основных видов индексов, которые могут использоваться в базах данных для оптимизации доступа к данным. Каждый тип индекса имеет свои преимущества и особенности, и выбор подходящего индекса зависит от специфики приложения и требований к производительности запросов.

* **Структура каких таблиц описывается в первую очередь при со-здании базы данных?**

При создании базы данных одной из первостепенных задач является создание структуры таблиц, которые будут хранить данные. При этом обычно описывается следующее:

1. Название таблицы: Каждая таблица должна иметь уникальное название, которое ясно указывает на ее содержимое или назначение.

2. Столбцы таблицы: Для каждой таблицы определяются столбцы, которые будут хранить различные атрибуты или свойства объектов, информацию о сущностях. Для каждого столбца указывается его имя, тип данных (например, целые числа, строки, даты и т. д.), и другие атрибуты (например, допустимые значения, ограничения, уникальность и т. д.).

3. Определение первичного ключа: Каждая таблица должна иметь первичный ключ, который однозначно идентифицирует каждую запись в таблице. Первичный ключ обеспечивает уникальность данных в таблице и связь между таблицами.

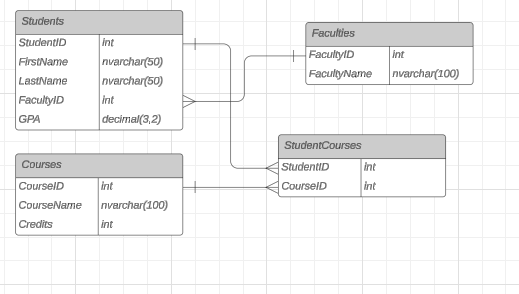
4. Определение внешних ключей: Если в базе данных присутствуют связи между таблицами, то для каждой такой связи определяются внешние ключи. Внешний ключ обеспечивает связь между записями в разных таблицах.

5. Индексы: При создании таблицы также могут определяться индексы для ускорения поиска и сортировки данных. Индексы могут быть созданы как на первичном ключе, так и на других столбцах таблицы в зависимости от требуемой производительности запросов.

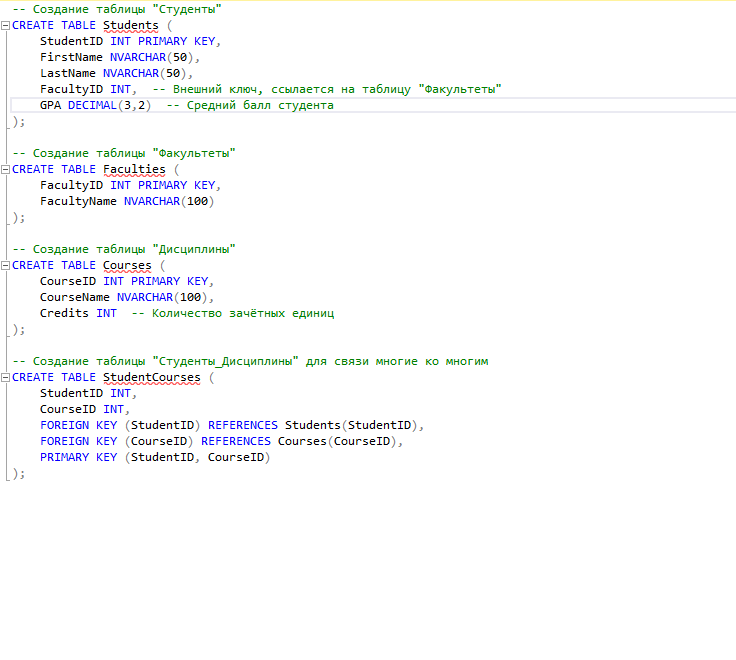
6. Ограничения целостности данных: Описываются правила, которые гарантируют целостность данных в таблице. Например, ограничения целостности могут запрещать пустые значения в определенном столбце, проверять соответствие значений определенным условиям и т. д.

При создании базы данных описание структуры таблиц является ключевым этапом, поскольку от правильного проектирования этой структуры зависит эффективность работы с данными, производительность запросов и целостность информации.

Диаграмма базы данных



create\_tables.sql



Заполнение таблиц данными

