Лекция нормализация данных

На этом занятии мы познакомимся с процессом *нормализации* данных — мы разобьем исходную базу данных на ее логические составляющие, называемые *таблицами*. Мы обсудим преимущества и недостатки нормализации и денормализации базы данных, а также рассмотрим то, как с точки зрения нормализации соотносятся целостность базы данных и ее производительность.

Нормализованные данные - это способ представления и законы, по которым строятся реляционные базы данных, созданные Э. Ф. Коддом. Процесс преобразования базы данных из более низшей в более высшую нормальную форму называется нормализацией БД. Нормализация осуществляется с целью оптимизации объема БД, быстродействия запросов и т.п. Вообще-то нормальных форм 5, но реально, на практике, используются 3 нормальные формы, точнее база данных в третьей нормальной форме (3НФ).

Нормализуем базу данных

Нормализация представляет собой процесс, направленный на уменьшение избыточности информации в базе данных. Кроме самих данных, в базе данных также могут быть нормализованы различные наименования, имена объектов и выражения.

Нормализация — это процесс, направленный на уменьшение избыточности информации в реляционной базе данных.

Типичная база данных до нормализации

Ненормализованная база данных содержит информацию в одной или нескольких различных таблицах; при этом создается впечатление, что включение данных в ту или иную таблицу не обусловлено никакими видимыми причинами. Такое положение дел может оказывать негативное влияние на безопасность данных, рациональное использование дискового пространства, скорость выполнения запросов, эффективность обновления базы данных и, что, наверное, является наиболее важным, на целостность хранимой информации. База данных перед нормализацией представляет собой структуру, которая логически еще не разбита на более управляемые таблицы меньшего размера. На рис.1 показана база данных до ее нормализации.

Рис. 1 База данных до нормализации

Компания

ID sotr
Фамилия
Имя
Отчество
квартира
дом
улица
город
индекс
телефон
дата рождения
должность
оклад
стаж работы
кредит
дата выплаты
ежемесячный взнос

Логическое проектирование базы данных

При проектировании любой базы данных всегда следует иметь в виду конечного пользователя. Логическое проектирование базы данных (также называемое построением ее *логической модели*) представляет собой процесс объединения данных в логически организованные группы объектов, которые можно легко поддерживать. Логическое проектирование базы данных должно приводить к уменьшению повторяющейся информации или даже полному ее устранению. В конце концов, зачем хранить одни и те же данные дважды? Кроме того, все используемые в базе данных соглашения о наименованиях также должны быть стандартными и логически обоснованными.

Избыточность данных

Данные не должны быть избыточными; существует несколько причин, почему дублирование данных следует стремиться свести к минимуму. Например, нет необходимости хранить домашний адрес сотрудника компании более, чем в одной таблице, поскольку при этом непроизводительно расходуется дисковое пространство. Кроме того, может возникнуть невообразимая путаница, когда, например, адрес сотрудника в одной таблице не соответствует его же адресу в другой. Какая информация достоверна? Есть ли у вас соответствующие документы для проверки действительного адреса этого сотрудника? Как ни сложно управление информацией само по себе, избыточность данных в этом случае может оказаться настоящим бедствием.

Нормальные формы

Несколько следующих разделов мы посвятим рассмотрению нормальных форм, понятие о которых неразрывно связано с процессом нормализации базы данных.

Нормальная форма — это своеобразный показатель уровня, или глубины, нормализации базы данных. Уровень нормализации базы данных соответствует нормальной форме, в которой она находится.

Существует три наиболее распространенных нормальных формы, в которых может находиться база данных в процессе нормализации:

Из них каждая последующая зависит от шагов, предпринятых в процессе получения предыдущей нормальной формы. Например, прежде чем вы сможете нормализовать базу данных в соответствии со второй нормальной формой, эта база данных уже должна находиться в первой нормальной форме.

Для правильного построения схемы БД традиционной реляционной модели выполняется нормализация исходной схемы, которая опирается на такие понятия, как функциональная зависимость и ключ отношения.

Определение КЛЮЧА

Пусть известно отношение R (A1, A2, ..., An) и определено множество функциональных зависимостей F на множестве атрибутов AR отношения R. Тогда, некоторое подмножество атрибутов $X \subset \{A1, A2, ..., An\}$ будем называть ключом если:

1. Каждый атрибут AR функционально зависит от X, т.е. $X \rightarrow A1, A2, .., An$ и эта зависимость принадлежит F+.
2. Ни один атрибут из набора X не может быть удален без нарушения свойства 1, т.е. ни для какого собственного подмножества $Y \subset X$, зависимость $Y \rightarrow A1, A2, ..., An$ не принадлежит F+.

Первое свойство (**идентифицируемость**), гарантирует, что в отношении не может быть двух кортежей (двух выборок) с одинаковыми ключами.

Второе свойство (**неизбыточность**) требует, чтобы ключ состоял из минимального числа атрибутов, т.е. удаление любого атрибута из множества атрибутов, составляющих возможный ключ, лишает его первого свойства.

Таким образом, в отношении может быть несколько возможных ключей, один из которых объявляется главным или первичным ключом.

Первая нормальная форма

Все рассматриваемые отношения в реляционном подходе должны находиться в первой нормальной форме (1НФ), которая предполагает, что элементы доменов отношений не являются множествами (т.е. являются атомарными) и не ограничивает наличие функциональных зависимостей между атрибутами в схеме отношения.

Целью первой нормальной формы является разделение исходных данных на логические составляющие, именуемые таблицами. После проектирования каждой таблицы для большинства из них (или даже для всех) назначается первичный ключ. Внимательно изучите рисунок 2, из которого видно, как исходная база данных, показанная на предыдущем рисунке, была перепроектирована в соответствии с первой нормальной формой.

Рис. 2 1 нормальная форма

Компания	
Сотрудник	Адрес
ID_sotr Фамилия Имя Отчество дата рождения должность оклад стаж работы кредит дата выплаты ежемесячный взнос ID_adr	ID_adr квартира дом улица город индекс телефон

Как видите, для построения первой нормальной формы нам потребовалось разбить данные на логические составляющие (таблицы), каждая из которых имеет первичный ключ и гарантирует отсутствие повторяющихся групп данных. Теперь вместо одной большой таблицы у нас есть несколько более управляемых таблиц меньшего размера: Сотрудник и Адрес. Как правило, первичный ключ указывается в списке столбцов таблицы первым; в нашем случае первичными ключами являются, соответственно, столбцы ID_sotr, и ID_adr.

Первая нормальная форма требует, чтобы на любом пересечении строки и столбца находилось единственное значение, которое должно быть атомарным, и в таблице не должно быть повторяющихся строк.

Вторая нормальная форма

Прежде чем перейти к рассмотрению второй нормальной формы (2НФ), рассмотрим понятие **полной** функциональной зависимости, используемое в теории нормализации.

Пусть имеется функциональная зависимость: {X1, X2, ..., Xn} Y.

Если из множества атрибутов, стоящих слева от знака, можно исключить некоторые имена атрибутов без разрушения функциональной зависимости, то говорят, что Y зависит от множества {X1, ..., Xn} неполно. В противном случае, имеет место полная функциональная зависимость атрибута Y от множества атрибутов {X1, ..., Xn}.

Отношение R (A1, ..., An) находится **во второй нормальной форме** (2НФ), если оно находится в 1НФ и если каждый непервичный атрибут функционально полно зависит от каждого возможного ключа.

Целью второй нормальной формы является помещение в отдельную таблицу данных, которые только частично зависят от первичного ключа. Рис. 3 иллюстрирует вторую нормальную форму.

Рис. 3

Компания			
Сотрудник	Адрес	Телефон	Должность
ID_sotr Фамилия Имя Отчество дата рождения стаж работы кредит дата выплаты ежемесячный взнос ID_adr ID_ph ID_dolg	ID_adr квартира дом улица город индекс	ID_ph телефон	ID_dolg должность оклад

Согласно этому рисунку, вторая нормальная форма может быть получена из первой путем дальнейшего разбиения таблиц на более специальные составляющие.

Третья нормальная форма

Отношение R находится в 3НФ, если оно находится в 2НФ и в нем нет транзитивных зависимостей атрибутов от возможных ключей, т.е. каждый **непервичный** атрибут **нетранзитивно** зависит от каждого возможного ключа отношения.

Целью третьей нормальной формы является устранение из таблицы данных, не зависящих от ее первичного ключа. Рис. 4 иллюстрирует третью нормальную форму.

Компания				
Сотрудник	Кредит	Адрес	Телефон	Должность
ID_sotr Фамилия Имя Отчество дата рождения стаж работы ID_kr ID_adr ID_ph ID_dolg	ID_kr кредит дата выплаты ежемесячный взнос	ID_adr квартира дом улица город индекс	ID_ph телефон	ID_dolg должность оклад

Соглашения о наименованиях

Соглашения о наименованиях являются одним из главных соображений, которыми следует руководствоваться при нормализации базы данных. Вы должны давать своим таблицам такие имена, которые бы наглядно описывали содержащуюся в них информацию. На уровне всей компании должны быть установлены соглашения о наименованиях, предусматривающие основные принципы для присвоения имен не только таблицам внутри базы данных, но и пользователям, файлам и другим связанным с ними объектам. Разработка и проведение в жизнь соглашений о наименованиях является одним из первых шагов компании на пути к успешному внедрению базы данных.

Чем выгодна нормализация

Основные преимущества нормализации:

- Лучшая общая организация базы данных
- Сокращение избыточности информации
- Непротиворечивость информации внутри базы данных
- Более гибкий проект базы данных
- Большая безопасность данных

Процесс нормализации данных приводит к улучшению их общей организации, тем самым облегчая работу каждому — от пользователя, который обращается к таблицам, до администратора базы данных (DBA), ответственного за управление всеми объектами базы данных в целом. Снижение избыточности данных влечет за собой упрощение их структуры и способствует рациональному использованию дискового пространства. Вследствие минимизации дублирующей информации значительно уменьшается вероятность появления противоречивых данных — это относится, например, к случаю,

когда в одну таблицу имя сотрудника компании введено как STEVE SMITH, в то время как в другой таблице он значится под именем STEPHEN R. SMITH. Нормализация базы данных с разбиением ее на более мелкие таблицы дает вам большую гибкость при изменении существующих структур данных. Согласитесь, что намного проще изменить несколько небольших таблиц, содержащих ограниченное количество данных, чем одну огромную таблицу, в которой хранится вся жизненно важная информация. И наконец, нормализация способствует повышению безопасности информации в том смысле, что администратор базы данных может предоставлять некоторым пользователям доступ лишь к ограниченному числу таблиц. После проведения нормализации базы данных организация защиты хранимой в ней информации значительно упрощается.

Целостность данных связана с обеспечением непротиворечивости и достоверности информации, хранящейся внутри базы данных.

Ссылочная целостность

Под *ссылочной целостностью* подразумевается, что значения в столбце одной таблицы зависят от значений в столбце другой. Например, для того, чтобы запись о кредите сотрудника можно было внести в таблицу Кредиты, необходимо наличие записи о нем в таблице Сотрудники. С помощью ограничений целостности можно также контролировать значения в столбцах - для этого вам потребуется установить пределы их диапазонов. Такие ограничения целостности должны создаваться сразу после создания таблицы. Управление ссылочной целостностью обычно происходит с помощью первичных и внешних ключей.

Для поддержки ссылочной целостности внешний ключ таблицы (обычно это единственное поле) должен непосредственно ссылаться на поле другой таблицы. В предыдущем абзаце поле ID_sotr Таблицы Кредиты является внешним ключом, ссылающимся на поле ID_sotr таблицы Сотрудники.

Недостатки нормализации

Хотя наиболее удачные базы данных всегда в той или иной степени нормализованы, у нормализованной базы данных есть один существенный недостаток, связанный со снижением ее производительности. Чтобы понять, почему это происходит, необходимо знать, что при выполнении базой данных запросов или транзакций существенную роль начинают играть такие факторы, как использование центрального процессора и памяти, а также система ввода/вывода. Короче говоря, для обработки транзакций и запросов в

случае нормализованной базы данных к центральному процессору, памяти и системе ввода/вывода предъявляются существенно большие требования, чем когда эта база данных ненормализована. Ведь для получения требуемой информации или обработки существующих данных нормализованная база данных должна сначала найти все необходимые таблицы, а затем объединить содержащуюся в них информацию.

Денормализация базы данных

Итак, денормализация базы данных — почему вообще она вам когда-нибудь может потребоваться? Ответ прост: единственной причиной денормализации базы данных является попытка улучшить ее производительность. Денормализованная база данных — это совсем не то же самое, что база данных, которая никогда не была нормализована. *Денормализация* базы данных представляет собой процесс, направленный на некоторое снижение уровня ее нормализации. Не забывайте, что нормализация базы данных на самом деле приводит к снижению ее производительности из-за частого выполнения операций, связанных с соединением таблиц. Процесс денормализации данных может включать в себя перепроектирование отдельных таблиц, а также дублирование информации с целью уменьшения числа таблиц, подлежащих объединению для поиска необходимых данных; все эти меры приводят к снижению требований к системе ввода/вывода, а также уменьшают загрузку центрального процессора.

Денормализация — это процесс изменения структуры таблиц нормализованной базы данных, направленный на получение управляемой избыточности данных с целью повышения производительности системы.

Однако денормализация данных имеет и свои издержки. Повышенная избыточность информации, хранящейся в денормализованной базе данных, может улучшить ее производительность, но потребует от вас больших усилий при отслеживании связанных между собой данных. Написание приложений в этом случае становится более сложным делом, поскольку исходные данные были разбросаны по разным таблицам и их поиск может вызвать определенные затруднения. Кроме того, теперь больших усилий потребует и поддержка ссылочной целостности — ведь родственные данные были распределены среди большого числа таблиц. Как при нормализации данных, так и при их денормализации существует золотая середина; однако в любом случае от вас требуется глубокое понимание сущности реальных данных, а также особых требований, предъявляемых к деятельности вашей компании.