УДК 681.142.2

А.В. Зубков

СИНХРОНИЗАЦИЯ МОДИФИКАЦИЙ ДЕНОРМАЛИЗОВАННЫХ ДАННЫХ В ПРИЛОЖЕНИЯХ LOTUS NOTES/DOMINO

Рассматриваются подходы для реализации синхронизации модификаций данных в приложениях Lotus Notes. Обсуждаются различия в моделях организации хранения данных, плюсы и минусы подходов к задаче синхронизации модификаций денормализованных данных в Lotus Notes. Кратко описываются алгоритм синхронизации, особенности реализации подходов.

Как известно, существуют несколько моделей организации хранения данных. Самая известная и широко используемая – это, безусловно, реляционная модель. Кроме реляционной, специалистами в разное время разработаны такие модели, как объектная, иерархическая и другие. В статье рассматривается система электронного документооборота Lotus Notes, или, кратко, Notes [1]. Если спросить специалиста, какую модель организации хранения данных имеет Notes, то, скорее всего, он затруднится дать однозначный ответ на этот вопрос, так как она по своим признакам однозначно не подходит ни под одно из определений известных моделей. Модель хранения данных в Lotus Notes конечно не реляционная. Возможно, иерархическая либо объектная. Что несомненно, модель хранения данных, используемая в Notes, достаточно удобна для реализации документооборота и удовлетворяет большинству задач в Notes-приложениях. Но иногда необходимо расширить ее возможности.

О СИСТЕМЕ LOTUS NOTES

Lotus Notes – это документо-ориентированная система групповой работы [2]. Lotus Notes имеет множество встроенных интегрированных подсистем, например встроенную систему электронной почты, которые делают эту систему чрезвычайно мощным средством автоматизации документооборота. Lotus Notes/Domino является клиент-серверной архитектурой, а сервер Lotus Domino к тому же является и вебсервером, позволяющим получить доступ к базам документов через глобальную сеть Internet.

Система Lotus Notes имеет следующие компоненты:

- систему управления базами документов,
- почтовую систему с поддержкой календаря и задач,
- систему репликации данных,
- встроенный объектно-ориентированный язык программирования Lotus Script и мощный язык формул,
- поддержку Java и JavaScript, XML.

Рассмотрим подробнее систему управления базами документов, в частности подсистему хранения данных. Для лучшего понимания будем постоянно использовать аналогии с системами управления реляционными базами данных.

Кортежем в Notes является документ. Помимо набора атрибутов, документ имеет свои собственные настройки управления доступом, может хранить код для управления своим отображением. Документ, как минимальная единица данных, практически эквивалентен записи в таблице реляционной СУБД, но имеются и отличия. Первое, и самое основное, группа документов, созданных с использованием одной формы (экранной формы как шаблона для создания документа), не соответствует таблице. Каждый документ полностью независим от других. Второе, документ может иметь набор атрибутов, отличающийся от набора в других документах.

Помимо этого, в системе Lotus Notes не определены правила формирования и поддержки связей между кортежами, или сущностями. Документ Notes может иметь неограниченное число ссылок на другие документы по некоторому ключу, определенному в конкретном приложении, но поддержка и использование этих связей полностью оставлена на разработчика приложения. Наиболее часто используется автоматически формируемый системой ключ Document Universal ID, но могут использоваться и различные искусственные ключи.

ПОСТАНОВКА ЗАДАЧИ

Сначала приведем определения используемых терминов из теории реляционных баз данных:

1. Корректной считается такая схема базы данных, в которой отсутствуют избыточные функциональные зависимости. В противном случае приходится прибегать к процедуре декомпозиции (разложения) имеющегося множества отношений. При этом порождаемое множество содержит большее число отношений, которые являются проекциями отношений исходного множества. *Обратимый* пошаговый процесс замены данной совокупности отношений другой схемой с устранением избыточных функциональных зависимостей называется *нормализацией* [3].

2. Отношение R находится в *пятой нормальной форме* (*5НФ*) тогда и только тогда, когда каждая зависимость соединения в отношении R подразумевается потенциальными ключами отношения R [4].

В приложениях, построенных с использованием РСУБД, разработчики стараются реализовать такую схему данных, которая бы соответствовала одной из нормальных форм, как правило, пятой НФ. Это делается для того, чтобы хранение и использование данных было наиболее эффективно и просто. Это также означает, что среди данных отсутствуют избыточные функциональные зависимости. В отличие от РСУБД, в Lotus Notes данные хранятся даже не в первой нормальной форме, так как любой атрибут документа Notes в общем случае является сложным, то есть может иметь несколько значений. Ясно, что использование денормализованных данных требует некоторых дополнительных действий. Попытаемся определить, какие действия необходимы, или желательны, в связи с декларированной денормализацией. Исходя из практического опыта создания различных Notes-приложений, можно выделить одно действие, которое совершенно необходимо для нормальной работы.

Рассмотрим подробнее поведение системы Lotus Notes в одном частном, но очень часто используемом случае. Если в некотором документе (назовем его документом-приемником) определенное поле получило свое текущее значение из поля в другом документе

(документе-источнике) то при изменении значения в источнике, значение в приемнике не изменится. Это происходит потому, что ссылка на значение источника никаким образом не хранится в документе-приемнике (если не делать этого специально с помощью кода). Становится понятно, что значение приемника совершенно независимо от источника, и нет специальных функций, производящих синхронизацию модификаций. То есть автоматически производимой системой синхронизации в Lotus Notes нет. Поэтому и возникает задача дополнения подсистемы хранения данных функцией синхронизации модификаций денормализованных данных.

ОПРЕДЕЛЕНИЕ ВОЗМОЖНОСТИ РЕШЕНИЯ

Попытаемся определить, можно ли вместо обеспечения непротиворечивости денормализованных данных просто хранить нормализованные данные в базах Notes. Можно ли применить точно такой же метод, какой применяется в приложениях, построенных на реляционных базах данных: хранение не самого значения, а только ссылки на него, и затем отображение значений вместо ссылок с помощью запросов. Это возможно, но с некоторыми, достаточно сильными ограничениями. Показ самих значений вместо ссылок в реляционных базах производится исполнением SQL запросов и соединения таблиц по ключу (JOIN). Так как представление в реляционных базах – это тоже запрос, то такой запрос – единственный способ показать данные.

В Notes, как и в большинстве систем управления данными, мы имеем аналогичные презентационные возможности: показ данных в форме (аналог запроса к базе) и показ данных в представлении. В форме мы можем показать значения из других документов, просто исполнив определенный код на событии открытия документа. В поле хранится не само значение, а некоторое ключевое значение для поиска документа-источника, например его Universal ID. Находим этот документ, затем получаем значение нужного нам поля и помещаем это значение в видимое нередактируемое поле на форме. Нередактируемое оно именно потому, что значения для такого поля задаются в другом месте. Повторяем эти действия для всех полей документа-приемника, которые требуют данные из внешних документов.

С представлением же все иначе, ведь, в отличие от реляционных баз данных, в Notes представление – это не совсем запрос. Это, скорее, статический индекс, обновляемый по событию, причем значения, показываемые в представлении, также хранятся в индексе. Плюс в Notes невозможно сделать представление, в одной строке которого показывались бы данные более чем из одного документа, т.е. нет аналога операции JOIN.

Как же решить эту проблему? Существует способ, который возвращает нас к денормализации данных. Нужно хранить не только ссылки на источник значений, но и сами значения. Тогда мы можем показывать значения как в формах, так и в представлениях, а также иметь связи для обновления этих значений. Естественно, становится необходимым постоянно поддерживать непротиворечивость данных. Когда поле в документе-источнике изменилось, нужно изменить и значение в поле документа-приемника. Возникает вопрос, когда и как обновлять значения в документах-приемниках.

ОПИСАНИЕ ПОДХОДОВ РЕШЕНИЯ

Существует много различных вариантов реализации. В результате обобщения этих вариантов стало возможно предложить 3 общих подхода:

Первый подход: «Синхронизация по запросу»

При сохранении документа-источника проверяется, изменилось ли значение поля, используемого в документах-приемниках, и если да, то выполняется операция обновления значений во всех документах-приемниках.

Второй подход: «Ленивая синхронизация»

При открытии документа-приемника на редактирование проверяется, изменилось ли значение в документе-источнике, и если да, то обновляются значения в текущем документе-приемнике.

Третий подход: «Полная синхронизация»

По расписанию либо по запросу выполняется операция проверки всех изменившихся документов, и если в них изменились значения, используемые в документах-приемниках, то производится обновление значений во всех документах-приемниках.

Каждый из приведенных трех подходов имеет определенные плюсы и минусы. Рассмотрим основные из них:

1) «Синхронизация по запросу»:

+ данные всегда актуальны (при допущении, что операция обновления значений занимает несущественное время);

– возможны конфликты, когда обновляется документ, открытый на редактирование другими пользователями;

– при большом количестве документов со ссылками изменяется много данных. Так как изменение документов обычно происходит во время активной работы с приложением, то выполнение этой операции может значительно замедлить работу с системой.

2) «Ленивая»:

+ операция обновления выполняется очень быстро;

– пока документ не открыт на редактирование, данные в нем неактуальны;

– для актуализации всех данных необходимо открыть и пересохранить все документы-приемники.

3) «Полная»:

+ может выполняться в моменты бездействия системы, например ночью;

– до запуска данные неактуальны.

С учетом всех плюсов и минусов указанных подходов, можно решить, какой либо какие из подходов использовать в разрабатываемом приложении. Однако делать это можно только индивидуально, учитывая особенности функционирования этого приложения. Конкретные рекомендации об использовании приведенных подходов выходят за рамки данной статьи и не будут рассмотрены.

КРАТКОЕ ОПИСАНИЕ РЕАЛИЗАЦИИ

Возможная реализация всех трех подходов практически одинаковая: поиск документов-приемников (документов-источников в случае 2-го подхода), обновление значений в них, сохранение документов. Итак, нам необходимо иметь структуру для поиска документов-приемников по известному документу-источнику и наоборот. Такая структура может представлять собой, например, специальное поисковое представление. Однако в приложениях с большим количеством документов индекс такого представления может быть слишком велик, что повлечет за собой проблемы в функционировании приложения. Эта проблема решается разделением одного представления на несколько, но приводит к накладным расходам на определение нужного представления. Обновление и сохранение документов очень простая операция, но, тем не менее, может привести к конфликтам сохранения, если обновляемый документ открыт на редактирование пользователем в данный момент времени работающим с системой. При наличии в документе флага редактирования можно пропускать такие документы и обрабатывать их позже, в момент сохранения документа открывшим его пользователем. Однако это также приводит к определенным накладным расходам, и время реакции системы может перестать удовлетворять пользователей.

ЗАКЛЮЧЕНИЕ

Ранее уже было замечено, что ни один из подходов не обеспечивает достаточно надежную реализацию синхронизации модификаций денормализованных данных. Только при изменении архитектуры подсистемы хранения данных в Lotus Notes можно говорить о простом и удобном решении задачи синхронизации. В последнее время в прессе появились слухи об использовании РСУБД IBM DB2 в качестве хранилища данных в Notes. Если эти слухи имеют под собой реальную основу, то задача решится сама собой.

Тем не менее, и сегодня для широкого круга задач можно создать работающую схему синхронизации данных. Единственное, что необходимо, это эффективная реализация необходимых улучшений механизмов поиска и обработки документов.

ЛИТЕРАТУРА

1. *Линд Д., Керн С.* Lotus Notes и Domino 5/6. Киев: ДиаСофт ЮП, 2003.
2. *Линд Д., Керн С.* Lotus Notes и Domino R5. Энциклопедия пользователя. Киев: ДиаСофт ЮП, 2003.
3. *Зеленков Ю.А.* Введение в базы данных, 1997; http://alpha.netes.ru/win/db/toc.html
4. *Дейт К.* Введение в системы баз данных. Киев: Диалектика, 1998.

Статья представлена кафедрой теоретических основ информатики факультета информатики Томского государственного университета, поступила в научную редакцию «Кибернетика и информатика» 23 мая 2003 г.