УДК 681.3

В.Ф. Третьяк, Ян Исун, Д.Ю. Голубничий

Харьковский университет Воздушных Сил им. И. Кожедуба

ИСПОЛЬЗОВАНИЕ ТЕХНОЛОГИИ РЕПЛИКАЦИИ В СИСТЕМЕ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ

Рассматривается репликация как технология приведения баз данных, функционирующих в распределенной среде в актуальное состояние, за счет выявления измененных данных, передачи и применение этих изменений в базе данных получателя.

репликация, тиражирование, системы управления базами данных

Введение

Постановка проблемы. В настоящее время информационная инфраструктура современных предприятий и организаций, носит все более распределенный характер. Уровень принимаемых управленческих решений, контроль и управление информационными ресурсами требует все большей децентрализации [1, 9 – 12]. Структура современных организаций представляет собой разветвленную схему, в состав которой входят десятки территориально разделенных подразделений, функционирование которых, друг без друга, часто, вообще не возможно, а так же взаимодействие между структурой других организаций. При этом качество функционирования такой сложной, корпоративной системы во многом зависит от применяемых в ее деятельности новейших информационных технологий. Основной проблемой при автоматизации распределенных систем является организация обработки распределенных данных.

Применение технологии тиражирования данных [3, 10, 12], является важным этапом в успешной работе организации. Внедрение данной технологии обеспечивает гарантированную доставку, своевременность и целостность передаваемых данных. Репликация данных предполагает их дублирование в различных узлах компьютерной сети. При этом любая база данных (БД) является локальной (как для СУБД, так и для пользователя, работающего с ней), т.е. данные всегда размещаются локально на том узле сети, где они обрабатываются, и все транзакции в системе завершаются локально. Эффективность технологии репликации данных в распределенной системе зависит от того, какие операции выполняются в системе. Так, операция чтения объекта в репликационной схеме выполняется локально в узле сети, это быстрее и дешевле, чем чтение объекта с удаленного компьютера сети. Поэтому, тиражирование данных в как можно большее количество узлов распределенной системы с часто повторяющимися запросами на чтение является предпочтительным, т.к. снижает нагрузку с центрального сервера. С другой стороны, если в распределенной системе происходит интенсивное обновление информации, т.е. преимущественно выполняются операции записи, то репликационная схема должна быть построена по принципу "направленного" тиражирования, при этом снижаются затраты на пересылку данных. Тем самым подход, основанный на анализе интенсивности операций "чтение-запись" в распределенных системах, обеспечивает возможность построения рациональных, в смысле технико-экономических показателей, алгоритмов тиражирования данных. В тоже время, размещение дорогостоящих СУБД в различных распределенных точках, где скорость обработки данных не играет роли, является не всегда оправданным. Рассмотрению особенностей различных механизмов тиражирования (репликации) баз данных и посвящена эта статья.

Анализ исследований и публикаций. Концепции построения систем репликации данных исследовались в СО РАН, в Санкт-Петербургском государственном университете, Томском государственном политехническом университете, Харьковском университете Воздушных Сил в научно-исследовательской лаборатории кафедры «Математического и программного обеспечения АСУ» (НИР «Репликация»), а так же в работах Новикова Б.А. «Методы и средства организации хранения в системах баз данных нового поколения»; Макарова С.В. «Управление обновлениями в СУБД расширенной архитектуры "Клиент-сервер»; Курбанова Я.З. «Разработка алгоритмов и программ для управления процессами решения взаимосвязанных задач в системе распределенной обработки данных»; Богдановой О.В. «Разработка методов моделирования и оптимизации распределенных систем обработки данных на локальных вычислительных сетях»; Мачтакова С.Г. «Разработка моделей и программного обеспечения распределенной информационной системы». Среди иностранных авторов можно выделить B. Ciciani, Matthias Nicola, James V. Hansen, Tom Swan, Peter Garrison, Tomas Delamarter.

Результаты исследований

Распределенная база данных (DDB – distributed database) – это совокупность множества взаимосвязанных баз данных, распределенных в компьютерной сети. Система управления распределенной базой данных определяется как программная система, которая позволяет управлять базой данных таким образом, чтобы ее распределенность была прозрачна для пользователей. В этом определении следует уточнить два отличительных условия. Первое заключается в том, что система состоит из (возможно, пустого) множества узлов приема запросов (query site) и непустого множества узлов данных (data site). Узлы данных обладают средствами для хранения данных, а узлы приема запросов – нет; на них лишь выполняются программы, реализующие пользовательский интерфейс для доступа к данным, хранящимся в узлах данных. Второе условие заключается в том, что узлы логически представляют собой независимые компьютеры, на которых установлены собственные операционные системы (может быть, одинаковые на всех узлах, а возможно, и разные) и могут выполнять независимые приложения. Важнейший отличительный признак – слабосвязанный характер среды, где каждый узел имеет собственную операционную систему и функционирует независимо.

База данных физически распределяется по узлам данных при помощи фрагментации и репликации, или тиражирования**,** данных. Отношения, принадлежащие реляционной базе данных, могут быть фрагментированы на горизонтальные или вертикальные разделы. Горизонтальная фрагментация реализуется при помощи операции селекции, которая направляет каждый кортеж отношения в один из разделов, руководствуясь предикатом фрагментации. При реализации вертикальной фрагментации отношение делится на разделы при помощи операции проекции. За счет фрагментации данные приближаются к месту их наиболее интенсивного использования, что потенциально снижает затраты на пересылки; уменьшаются также размеры отношений, участвующих в пользовательских запросах.

Фрагменты данных могут также тиражироваться с учетом спроса на доступ к ним. Это полезно, если доступ к одним и тем же данным нужен из приложений, выполняющихся на разных узлах. В таком случае, с точки зрения экономии затрат, более эффективно будет поддерживать копии данных на всех узлах, чем непрерывно пересылать данные между узлами.

Использование репликации позволяет привести несколько баз данных с одинаковой структурой в одно и то же непротиворечивое (не вызывающее нарушений целостности) состояние, которое сопровождается взаимным внесением изменений. В [4, 8] репликацию определяют как процесс генерации и воспроизведения нескольких копий данных, размещаемых на одном или нескольких сайтах, в [1] под репликацией данных СУБД понимают приведение баз данных функционирующих в распределенной среде в актуальное состояние, за счет выявления измененных данных, передачи и применение этих изменений в базе данных получателя. Решение данной задачи при наличии постоянного канала между серверами баз данных сводится к неоднократно описанному в специальной литературе механизму двухфазной фиксации (2PC – two-phase commit) [9].

Среди функций, реализуемых службой репликации, выделяют[8]:

– масштабируемость**.** Служба репликации должна эффективно обрабатывать как малые, так и большие объемы данных;

– отображение и трансформация. Служба репликации должна поддерживать репликацию данных в гетерогенных системах, использующих несколько платформ. Это может быть связано с необходимостью отображения и преобразования данных из одной модели данных в другую или же для преобразования некоторого типа данных в соответствующий тип данных, но для среды другой СУБД;

– репликация объектов. В службе репликации должна существовать возможность реплицировать объекты, отличные от обычных данных;

– средства определения схемы репликации. Система должна предоставлять механизм, позволяющий привилегированным пользователям задавать данные и объекты, подлежащие репликации;

– механизм подписки. Система должна включать механизм, позволяющий привилегированным пользователям оформлять подписку на данные и другие подлежащие репликации объекты;

– механизм инициализации**.** Система должна включать средства, обеспечивающие инициализацию вновь создаваемой реплики.

Среди достоинств применения механизма репликации выделяют:

– надежность**.** Процесс репликации значительно увеличивает надежность системы, предоставляя приложениям альтернативный доступ к данным. При потере соединения с одним из серверов, пользователь может продолжать работу с данными, используя другой;

– производительность. Используя репликацию, можно обеспечить высокую скорость доступа к данным, так как нагрузка будет равномерно распределена между множеством сайтов репликации;

– возможность работы без постоянного соединения с базой. Снимки позволяют пользователю работать с набором данных без наличия постоянного соединения с базой данных. Позже, когда установится соединение, пользователь сможет синхронизировать (обновить) снимки, если необходимо и таким образом внесенные им изменения попадут в базу данных;

– уменьшение сетевой нагрузки. Репликация может быть использована для распределения данных между различными региональными центрами.

Приложения будут обращаться не к центральному серверу, а к региональному, тем самым, уменьшая нагрузку на сеть.

При разработке вариантов тиражируемой системы учитывают:

– загрузку и вычислительную мощность каждого узла как предполагаемого места расположения репликационных серверов – источников тиражируемых данных;

– качество и полосу пропускания каналов связи между парами удаленных узлов при определении пути распространения данных и частоты репликации;

– требования к надежности системы, необходимость дублирования актуальных данных, в том числе и многоконтурного;

– возможные конфликты и предполагаемые способы их автоматического разрешения или разрешения вручную.

Репликацию баз данных будем классифицировать [1 – 3, 5 – 8]:

I. По направлению репликации:

– однонаправленная, когда данные изменяются только в одной из БД, а в другой данные только хранятся и не подвергаются изменениям;

– мультинаправленная, когда данные могут изменяться и вводиться на всех БД.

II. По времени проведения сеанса репликации:

– репликация реального времени. Данные должны быть засинхронизированны немедленно после изменений;

– отложенная репликация. Процесс репликации запускается по какому либо событию во времени или по действию администратора БД.

III.По способу передачи информации во время процесса репликации:

– прямая передача. Соединение серверов, хранящих распределенные БД, происходит при помощи программы клиента, которая с одной стороны соединяется к своему серверу, а с другого конца имеет прямую связь с БД другого сервера и может подключиться напрямую к данным другого сервера, для прямого изменения и анализа реплицируемых данных с обоих концов, имея при этом гарантированный устойчивый канал связи.

– недетерминированная или вероятностная передача. Канал неустойчивый и не гарантирует устойчивую связь без падений во время процесса синхронизации и данные приходится передавать цельными пакетами, при этом принимающая сторона во время закачки и анализа данных не имеет немедленной возможности опросить источник при возникновении различных ситуаций.

IV. По стратегии репликации:

– клиент-сервер. Единственная копия состояния объекта хранится в серверной реплике. Остальные реплики являются клиентами. Все вызовы методов реплик-клиентов направляются серверу (рис. 1).

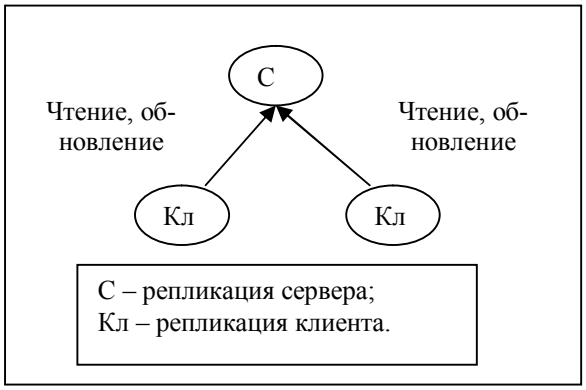

Рис.1. Стратегия репликации клиент-сервер

Данная стратегия в большинстве случаев является неэффективной, поскольку не обеспечивает локальность доступа к ресурсам. Другой недостаток стратегии клиент-сервер - низкая надежность как результат централизованного хранения и доступа к объекту. Существуют, однако, объекты, для которых клиент-сервер является единственной возможной стратегией репликации;

– пассивная репликация. В случае пассивной репликации каждая реплика хранит копию состояния объекта. Одна из реплик назначается *первичной*. Операции чтения выполняются локально во всех узлах. Операции, модифицирующие состояние объекта, направляются первичной реплике, которая, после выполнения метода, обновляет все остальные реплики (рис. 2);

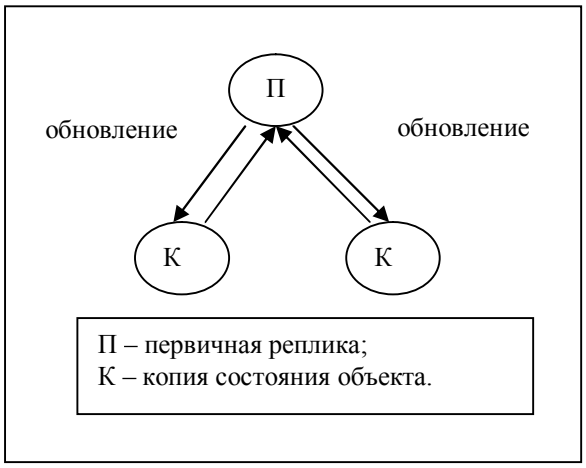

Рис. 2. Пассивная репликация

– активная репликация. Каждая реплика хранит копию состояния объекта. Операции чтения и модификации выполняются локально в каждом узле. Для поддержания когерентности реплик операции модификации рассылаются всем репликам объекта, которые выполняют их над локальной копией состояния (рис. 3).

К активным стратегиям репликации относится широкий класс алгоритмов, обеспечивающих различную степень согласованности реплик распределенного объекта (последовательная, каузальная, временная, слабая, пассивная (lazy) согласованности).

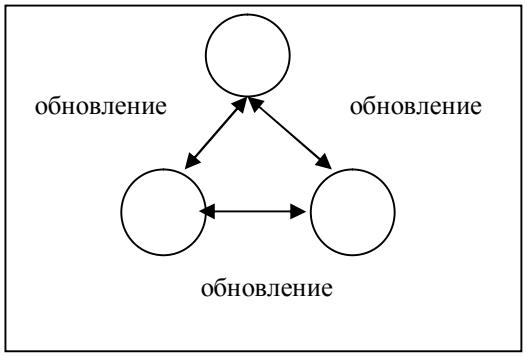

Рис.3. Активная репликация

V. По способу анализа реплицируемой информации:

– репликация по текущему состоянию – процесс, когда ядро алгоритма работает по принципу сравнения записей одной таблицы с записями другой, и на основании этого принимается решение о синхронизации;

– дельта репликация – процесс, когда в базе предусмотрен журнал вносимых изменений в БД, и алгоритм репликации переносит изменения по дельтам изменений накопленным в журнале.

VI. По типу тиражирования данных:

– базовое тиражирование. Тиражируемые копии обеспечивают доступ типа «только для чтения» к табличным данным, источником которых является первичный узел, иногда называемый основным узлом;

– усовершенствованное тиражирование. Считывание и обновление тиражируемых копий таблиц во всей системе.

VII. По способу тиражирования данных:

– тиражирование слиянием (merge replication). Суть его заключается в том, что все операции выполняются на удаленном компьютере, который может быть полностью отключен от компьютерной сети. Автономная СУБД записывает все операции с данными и их очередность. Затем в определенный момент, автономный компьютер связывается с издателем и согласовывает с ним свои данные, пересылая издателю последовательность операций, произведенных в удаленной базе данных. При возникновении конфликтов они разрешаются с помощью различных алгоритмов, например, на основе приоритета. После такого слияния издатель передает изменения на сервер-дистрибутор для дальнейшего распространения по подписчикам;

– тиражирование моментального снимка (snapshot replication). Способ тиражирования моментального снимка отличается от слияния данных тем, что издатель распространяет не последовательность транзакций, а снимок всех данных, в том числе и системных. В результате, подписчики не имеют возможности изменять полученные данные, и поэтому они доступны только на чтение. Все операции изменения данных выполняются только через издателя. Механизм тиражирования в этом случае работает так: дистрибьютор запрашивает у издателя моментальный снимок данных, а когда получает, то распространяет его по подписчикам. В этом варианте тиражирования, кроме распространения по запросу можно использовать схему принудительной рассылки тиража. При этом распространение данных инициируется не подписчиком, а дистрибьютором. Таким образом, при хорошей связи подписчик начинает прием данных, как только они стали доступны дистрибьютору;

– транзакционное тиражирование (transactional replication). В процессе тиражирования транзакций, от издателя к подписчикам передаются не данные, а только операции над ними. При этом периодически необходимо выполнять и полную синхронизацию данных, которая выполняется по методу тиражирования моментального снимка данных. Транзакции, которые подтверждены издателем, собираются дистрибутором и копируются дальше по подписчикам, где те же операции выполняются с заранее распространенными снимками центральной базы данных. Здесь следует отметить, что тиражируются не отдельные операции над данными, а их логические группы - транзакции. Если же подписчику требуется изменить какие-либо данные в основной базе, то он должен провести транзакцию с издателем;

– тиражирование с обновлением на подписчике (immediate updating subscribers). Тиражирование в случае немедленного обновления инициируется издателем. Как только издатель подтверждает транзакцию, он сообщает дистрибутору о том, что данные изменены. Дистрибутор забирает подтвержденную транзакцию и рассылает ее подписчикам. Если во время завершения транзакции связь между дистрибутором и издателем была прервана, то транзакция записывается в очередь и будет передана дистрибутору, как только связь восстановится. Дальнейшее распространение данных выполняет дистрибутор;

– распределенные транзакции (distributed transaction). В распределенных транзакциях участвуют несколько равноправных серверов баз данных, поэтому модель «издатель-дистрибьютор-подписчик» в этом случае не применима. Для подтверждения каждой транзакции все команды распространяются всем серверам, от которых должны прийти подтверждения о возможности проведения данной транзакции. Только после этого транзакция является принятой. Так работает двух фазная система подтверждения транзакции. В случае, когда хотя бы одна база данных недоступна, вся система перестает работать, поскольку неработающая система не может ни подтвердить, ни опровергнуть транзакцию. Поскольку все сервера равноправны, то транзакции можно проводить через любой сервер СУБД, а в процессе подтверждения они становятся доступны всем остальным серверам.

Следует отметить, что почти все перечисленные методы тиражирования укладываются в единую модель «издатель-дистрибьютор-подписчик». Функции

издателя выполняет центральная база данных, которая хранит основной вариант публикации: группы столбцов и строк. Для каждой публикации должен быть только один издатель. Дистрибьютор следит за изменениями данных, которые вносит издатель, и распространяет их по подписчикам. Таким образом, каждому элементу системы достаточно иметь связь только с дистрибутором, а не со всеми другими фрагментами. Следует отметить, что при небольших объемах данных, функции издателя и дистрибутора может выполнять один компьютер. Характеристики тиражирования меняются в зависимости от организации связи между издателем и дистрибутором, а также между дистрибьютором и подписчиками. Такая модель не работает только в случае распределенных транзакций, когда любая база данных системы, прежде чем подтвердить изменение данных, должна получить разрешение от всех остальных серверов системы.

VIII. По реализации механизмов разрешения конфликтов:

– самая ранняя или самая поздняя временная отметка. Изменяются соответственно данные с самой ранней или самой поздней временной отметкой;

– приоритеты сайтов. Применяется обновление, поступившее с сайта с наибольшим приоритетом;

– дополняющие и усредненные обновления. Введенные изменения обобщаются. Этот вариант разрешения конфликтов может использоваться в тех случаях, когда обновление атрибута выполняется операциями, записанными в форме отклонений;

– минимальное или максимальное значение. Применяются обновления, соответствующие столбцу с минимальным или максимальным значением;

– по решению пользователя. АБД создает собственную процедуру разрешения конфликта. Для устранения различных типов конфликтов могут быть подготовлены различные процедуры;

– сохранение информации для принятия решения вручную. Сведения о конфликте записываются в журнал ошибок для последующего анализа и устранения администратором базы данных вручную.

IX. По типу схем владения данными:

– "ведущий/ведомый". При организации владения данными по схеме "ведущий/ведомый" асинхронно реплицируемые данные принадлежат одному из сайтов, называемому ведущим, или первичным, и могут обновляться только на нем. Здесь можно провести аналогию между издателем и подписчиками. Издатель (ведущий сайт) публикует свои данные, а все остальные сайты только лишь подписываются на данные, принадлежащие ведущему сайту, т.е. имеют собственные локальные копии, доступные им только для чтения. Потенциально каждый из сайтов может играть роль ведущего для различных, не перекрывающихся наборов данных. Однако в системе может существовать только один сайт, на котором располагается ведущая обновляемая копия каждого конкретного набора данных, а это означает, что конфликты обновления данных в системе полностью исключены;

– "рабочий поток". Схема владения "рабочий поток" позволяет передавать право обновления реплицируемых данных от одного сайта другому. Однако в каждый конкретный момент времени существует только один сайт, имеющий право обновлять некоторый конкретный набор данных;

– "повсеместное обновление". Схема владения с повсеместным обновлением создает равноправную среду, в которой множество сайтов имеют одинаковые права на обновление реплицируемых данных. В результате локальные сайты получают возможность работать автономно, даже в тех случаях, когда другие сайты недоступны.

Выводы

В настоящее время является актуальным создание системы репликации данных и разработка на этой основе алгоритмов и методов тиражирования информации в распределенных системах, а так же исследование методов адаптации системы репликации данных к различным платформам СУБД.

Список литературы

1. Дейт К.Дж. Введение в системы баз данных, 6-е издание: Пер. с англ. – К.: Вильямс, 2000. – 848 с.

2. Ciciani B., Dias D.M., Yu P.S. Analysis of Replication in Distributed Database Systems // IEEE Trans. on Knowledge and Data Engineering. – June 1990. – Vol. 2, No. 2. – P. 247-261.

3. Голубятников И.В., Матвеев Ю.В., Сергеев И.В. Создание программного комплекса репликации СУБД-независимых данных // Системный анализ, информатика и оптимизация. – М.: РЗИТЛП, 2001. – С. 106-116.

4. Ciciani B., Dias D.M., Yu P.S. Analysis of Concurrency-Coherency Control Protocols for Distributed Transaction Proc. Systems with Regional Locality // IEEE Trans. on Software Engineering. – October 1992. – Vol. 18, No. 10. – P. 899-914.

5. Использование Oracle 8 / Вильям Г. Пейдж и др. – К.; М.; С.-Пб.: Вильямс, 1998. – 752 с.

6. Мейер М. Теория реляционных баз данных. – М.: Мир, 1987. – 608 с.

7. Саймон А.Р. Стратегические технологи баз данных: менеджмент на 2000 год. – М.: Финансы и статистика, 1999. – 479 с.

8. Томас Конноли. Базы данных: проектирование, реализация и сопровождение. Теория и практика. – М.: Вильямс, 2000. – 1120 с.

9. Третяк В.Ф. Голубничий Д.Ю., Челенко Ю.В. Тиражирование данных в системе управления базами данных // Управління розвитком. – Х.: ХНЕУ, 2005. – № 3. – С. 94-95.

10. Третяк В.Ф., Приходько В.М., Голубничий Д.Ю. Актуальність побудови grid систем // Перша НТК ХУ ПС. – Х.: ХУ ПС, 2005. – С. 240-241.

11. Третяк В.Ф. Технология репликации в распределенных системах управления базами даннях // ІКТЗТ. – 2004. – № 1. – С. 7-10.

12. Третяк В.Ф.., Голубничий Д.Ю., Золотарьова І.О. Проблеми розвитку програмного забезпечення середовища розподілених обчислень GRID // Матеріали Третя НТК ХУ ПС. – Х.: ХУ ПС, 2007. – С. 77.

Поступила в редколлегию 26.02.2007

Рецензент: д-р техн. наук, доцент С.В. Листровой, Харьковская академия железнодорожного транспорта, Харьков.