Притязание на приоритет/перекрестная ссылка на связанные заявки

Данное изобретение заявляет приоритет предварительной заявки на патент США с регистрационным номером 60/330.842 от 1 ноября 2001 года, полностью включенной в настоящее описание посредством ссылки, и предварительной заявки на патент США с регистрационным номером 60/365.169 от 19 марта 2002 года, полностью включенной в настоящее описание посредством ссылки.

Область техники

Настоящее изобретение относится к компьютерным базам данных, более конкретно к способу и системе для достоверного обновления базы данных.

Предшествующий уровень техники

С увеличением размера баз данных и в результате их сильно распределенной структуры становится все в большей степени проблематичным обеспечение одинаковых версий данных в связанных базах данных. Если происходят существенные изменения в одной базе данных, то может потребоваться обновление других баз данных для включения указанных изменений в кратчайшие сроки. Осуществление таких обновлений может повлечь частое перемещение больших объемов данных обновления в несколько баз данных. Потенциально такой процесс может быть чрезвычайно сложным.

Указанная проблема дополнительно сочетается с ненадежной связью в системах. В этом случае данные могут быть потеряны при транспортировке. При этом требуется повторная передача данных, и другие базы данных вновь обновляются. Такое повторение существенно снижает эффективность системы и уменьшает область, в которой базы данных содержат новейшие данные.

Краткое описание чертежей

Фиг. 1 - структурная схема системы согласно варианту осуществления настоящего изобретения.

Фиг. 2 - структурная схема системы концентратора согласно варианту осуществления настоящего изобретения.

Фиг. 3 иллюстрирует возможную передачу обновлений базы данных из локальной базы данных в удаленную базу данных согласно варианту осуществления настоящего изобретения.

Фиг. 4 изображает файл отправки согласно варианту осуществления настоящего изобретения.

Фиг. 5 изображает инициализирующий файл отправки согласно варианту осуществления настоящего изобретения.

Фиг. 6 - временная диаграмма, иллюстрирующая формирование файла отправки и инициализирующего файла отправки, согласно варианту осуществления настоящего изобретения.

Фиг. 7 - блок-схема варианта осуществления настоящего изобретения, в котором могут быть сформированы файлы обновления локальной базы данных.

Фиг. 8 - блок-схема варианта осуществления настоящего изобретения, в котором удаленная база данных может получать файлы обновления из локальной базы данных.

Фиг. 9 - блок-схема другого варианта осуществления настоящего изобретения, в котором удаленная база данных может получать файлы обновления из локальной базы данных и проверять их достоверность.

Фиг. 10A - блок-схема варианта осуществления настоящего изобретения, в котором может быть проверена достоверность файлов обновления.

Фиг. 10B - блок-схема другого варианта осуществления настоящего изобретения, в котором может быть проверена достоверность файлов обновления.

Фиг. 11 - иллюстрация проверки достоверности файла обновления согласно варианту осуществления настоящего изобретения.

Подробное описание предпочтительных вариантов осуществления

Варианты осуществления настоящего изобретения обеспечивают способ и систему для достоверного обновления удаленной базы данных через сеть связи. Согласно вариантам осуществления формируется несколько периодических обновлений (далее по тексту "файл отправки"), основанных на добавочных изменениях в локальной базе данных. Каждое из периодических обновлений содержит по меньшей мере одну транзакцию. Формируется инициализирующее обновление (далее по тексту "инициализирующий файл отправки"), содержащее версию локальной базы данных в момент времени запуска. Дополнительно формируются идентификатор, соответствующий последнему периодическому обновлению, сформированному перед моментом времени запуска, и идентификатор, соответствующий последней транзакции, завершенной до момента времени запуска. Варианты осуществления преимущественно обеспечивают автономизацию файлов отправки и инициализирующего файла отправки для достоверного обновления удаленных баз данных.

Фиг. 1 - структурная схема, иллюстрирующая систему, согласно варианту осуществления настоящего изобретения. В основном, система 100 может содержать большую резидентную базу данных, через сеть связи получать запросы на поиск и обеспечивать ответы по поиску. Например, система 100 может представлять собой симметричный многопроцессорный (SMP) компьютер, например такой, как IBM RS/6000®M80 или S80, производимый компанией IBM (Armonk, штат Нью-Йорк), Sun Enterprise™10000, производимый Sun Microsystems (Santa Clara, штат Калифорния), и т.д. Система 100 также может представлять собой многопроцессорный персональный компьютер, например такой, как CompaqProLiant™ML530

(имеющий два процессора Intel Pentium® III, 866 МГц), производимый компанией Hewlett-Packard (Palo Alto, штат Калифорния). Система 100 может также содержать многопроцессорную операционную систему, например такую, как IBM AIX® 4, Sun Solaris™ 8 Operating Environment, Red Hat LINUX® 6.2, и т.д. Система 100 может получать через сеть 124 связи периодические обновления, которые могут параллельно включаться в базу данных. Варианты осуществления настоящего изобретения могут достигать очень высокой производительности обновления и поиска по базе данных посредством включения каждого обновления в базу данных без использования блокировок базы данных или средств управления доступом.

В варианте осуществления система 100 может содержать по меньшей мере один процессор 102-1, подсоединенный к шине 101. Процессор 102-1 может содержать внутренний кэш (например, кэш L1 не изображен). Между процессором 102-1 и шиной 101 может быть резидентно размещен кэш 103-1 вторичной памяти (например, кэш L2, кэши L2/L3 и т.д.). В предпочтительном варианте осуществления система 100 может содержать несколько процессоров 102-1 - 102-P, подсоединенных к шине 101. Между несколькими процессорами 102-1 - 102-P и шиной 101 также может быть резидентно размещено несколько кэшей 103-1 - 103-P вторичной памяти (например, архитектура сквозного просмотра), или, в виде другого варианта, к шине 101 может быть подсоединен по меньшей мере один кэш 103-1 вторичной памяти (например, архитектура с предысторией). Система 100 может содержать память 104, например такую, как оперативное запоминающее устройство (ОЗУ) и т.д., подсоединенную к шине 101 для хранения информации и инструкций, которые должны выполняться несколькими процессорами 102-1 - 102-P.

Память 104 может хранить большую базу данных, например, для преобразования имен доменов Интернет в адреса в Интернет, для преобразования имен или номеров телефонов в сетевые адреса, для обеспечения и обновления данных профиля абонента, для обеспечения и обновления данных присутствия пользователя и т.д. Преимущественно, размер базы данных и количество преобразований в секунду могут быть очень большими. Например, память 104 может содержать, по меньшей мере, 64 Гб ОЗУ и может содержать базу данных имен доменов в 500M (то есть $500x10^6$) записей, базу данных абонентов в 500M записей, базу данных мобильности номеров телефонов в 450M записей и т.д.

В возможной архитектуре 64-битной системы, например такой, как система, содержащая по меньшей мере один 64-битный процессор 102-1 с обратным порядком байтов, подсоединенный, по меньшей мере, к 64-битной шине 101, и 64-битную память 104, 8-байтовое значение указателя может быть записано в адрес (ячейки) памяти на границе 8-байтов (то есть адрес памяти, делимый на 8 или, например, 8N) с использованием одиночной непрерываемой операции. В основном, наличие кэша 103-1 вторичной памяти может просто задерживать запись 8-байтового указателя в память 104. Например, в одном варианте осуществления кэшем 103-1 вторичной памяти может быть кэш сквозного просмотра, действующий в режиме сквозной записи так, чтобы одиночная команда сохранения 8-байтов могла перемещать 8 байтов данных из процессора 102-1 в память 104 без прерывания и всего только за два такта системных часов. В другом варианте осуществления кэшем 103-1 вторичной памяти может быть кэш сквозного просмотра, действующий в режиме обратной записи так, чтобы 8-байтовый указатель мог быть сначала записан в кэш 103-1 вторичной памяти, который затем, в более позднее время, может записать 8-байтовый указатель в память 104, например, при записи в память 104 строки кэша, в которой хранится 8-байтовый указатель (то есть, например, когда «сбрасывается в память» определенная строка кэша или полностью кэш вторичной памяти).

В конечном счете, с точки зрения процессора 102-1, как только данные фиксируются выходными контактами процессора 102-1, все 8 байтов данных записываются в память 104 в одной непрерывной, непрерываемой передаче, которая может быть задержана действиями кэша 103-1 вторичной памяти, если он имеется в наличии. С точки зрения процессоров 102-2 - 102-P, как только данные фиксируются выходными контактами процессора 102-1, все 8 байтов данных записываются в память 104 в одной непрерывной, непрерываемой передаче, которая предписана протоколом согласованности кэша по кэшам 103-1 - 103-P вторичной памяти, которые могут задерживать запись в память 104, если имеются в наличии.

Однако если 8-байтовое значение указателя записывается в невыровненную ячейку памяти 104, например адрес памяти, который пересекает границу 8-байтов, то все 8 байтов данных не могут быть переданы из процессора 102-1 с использованием одиночной команды сохранения 8-байтов. Вместо этого процессор 102-1 может выдать две разные и отдельные команды сохранения. Например, если адрес памяти начинается за 4 байта до границы 8-байтов (например, 8N-4), то первая команда сохранения передает в память 104 4 старших байта (например, 8N-4), в то время как вторая команда сохранения передает в память 104 4 младших байта (например, 8N). Существенно, что между этими двумя отдельными командами сохранения процессор 102-1 может получить прерывание или процессор 102-1 может освободить управление шиной 101 для другого компонента системы (например, процессора 102-P и т.д.). Следовательно, значение указателя, резидентно находящееся в памяти 104, будет недействительным, пока процессор 102-1 не сможет завершить вторую команду сохранения. Если другой компонент начинает одиночное непрерываемое считывание памяти в эту ячейку памяти, то недействительное значение будет возвращено, как предположительно действительное.

Аналогично, новое 4-байтовое значение указателя может быть записано в адрес памяти, делимый на 4 (например, 4N), с использованием одиночной непрерываемой операции. Следует отметить, что в воз-

можном варианте, описанном выше, 4-байтовое значение указателя может быть записано в ячейку памяти 8N-4 с использованием одиночной команды сохранения. Безусловно, если 4-байтовое значение указателя записывается в ячейку, пересекающую границу 4-байтов, например, 4N-2, то все 4 байта данных не могут быть переданы из процессора 102-1 с использованием одиночной команды сохранения и значение указателя, резидентно размещенное в памяти 104, может быть недействительным в течение некоторого периода времени.

Система 100 также может содержать постоянное запоминающее устройство (ПЗУ) 106 или другие статические запоминающие устройства, подсоединенные к шине 101, для хранения статической информации и инструкций для процессора 102-1. Для хранения информации и команд к шине 101 может быть подсоединено запоминающее устройство 108, такое как магнитный или оптический диск. Система 100 также может содержать устройство 110 отображения (например, монитор на жидких кристаллах LCD (ЖДК)) и устройство 112 ввода данных (например, клавиатуру, мышь, шаровой указатель и т.д.), подсоединенные к шине 101. Система 100 может содержать несколько сетевых интерфейсов 114-1 - 114-O, которые могут передавать и принимать электрические, электромагнитные или оптические сигналы, несущие потоки цифровых данных, представляющие различные виды информации. В варианте осуществления сетевой интерфейс 114-1 может быть подсоединен к шине 101 и к локальной сети связи LAN (ЛС) 122, в то же время сетевой интерфейс 114-O может быть подсоединен к шине 101 и глобальной сети связи WAN (ГС) 124. Несколько сетевых интерфейсов 114-1 - 114-O могут поддерживать разные сетевые протоколы, включая, например, Gigabit Ethernet (например, IEEE Стандарт 802.3-2002, изданный в 2002), Fiber Channel (например, ANSI Стандарт X.3230-1994, изданный в 1994) и т.д. К ЛС 122 и ГС 124 может быть подсоединено несколько сетевых компьютеров 120-1 - 120-N. В одном варианте осуществления ЛС 122 и ГС 124 могут быть физически отдельными сетями связи, в то же время в другом варианте осуществления ЛС 122 и ГС 124 могут быть соединены через сетевой шлюз или маршрутизатор (для ясности не изображены). В качестве другого варианта, ЛС 122 и ГС 124 может быть одной и той же сетью связи.

Как отмечено выше, система 100 может обеспечивать услуги разрешения DNS (служба имен доменов). В варианте осуществления для разрешения (процесса определения адресов) DNS услуги разрешения DNS, в основном, могут быть разделены между транспортировкой по сети и функциями поиска данных. Например, система 100 может быть механизмом поиска (LUE) для сервера, оптимизированным для поиска данных по большим наборам данных, в то же время множество сетевых компьютеров 120-1 - 120-N могут быть множеством механизмов протокола (PE) клиента, оптимизированными для обработки сети связи и транспортировки. LUE может быть мощным многопроцессорным сервером, который хранит в памяти 104 весь набор записей DNS, чтобы способствовать высокоскоростному высокопроизводительному поиску и обновлению. В другом варианте осуществления услуги разрешения DNS могут обеспечиваться множеством мощных многопроцессорных серверов или множеством LUE, каждый из которых хранит в памяти подмножество всего набора записей DNS, чтобы способствовать высокоскоростному высокопроизводительному поиску и обновлению.

Напротив, множество PE могут быть обычными узкоспециализированными устройствами, основанными на PC, выполняющими эффективную многозадачную операционную систему (например, Red Hat LINUX® 6.2), которые минимизируют транспортную нагрузку обработки сети связи на LUE для максимизации доступных ресурсов для разрешения DNS. Устройства PE могут обрабатывать нюансы протокола DNS проводной линии связи, реагировать на недействительные запросы DNS и мультиплексировать действительные запросы DNS в LUE через ЛС 122. В другом варианте осуществления, содержащем множество LUE, хранящих подмножества записей DNS, PE могут определять, какое устройство LUE должно получить каждый действительный запрос DNS, и мультиплексировать действительные запросы DNS в соответствующие LUE. Количество PE для одного LUE может определяться, например, количеством запросов DNS, которое должно обрабатываться в секунду, и рабочими характеристиками конкретной системы. Для определения отношений соответствия и режимов также могут использоваться другие показатели.

В общем случае, могут поддерживаться другие варианты осуществления большого объема, основанные на запросах, включающие, например, разрешение номеров телефонов, обработку сигнализации SS7, определение для геолокации, установление соответствия номера телефона с абонентом, определение местоположения и присутствия абонента и т.д.

В варианте осуществления центральный сервер 140-1 оперативной обработки транзакций (OLTP) может быть подсоединен к ГС 124 и получать из разных источников добавления, изменения и удаления (то есть график обновления) для базы данных 142-1. OLTP сервер 140-1 может передавать обновления через ГС 124 в систему 100, которая содержит локальную копию базы данных 142-1. OLTP сервер 140-1 может быть оптимизирован для обработки графика обновления в различных форматах и протоколах, включая, например, Протокол Передачи Гипертекстовых файлов (HTTP), Протокол Регистратора Записи (RRP), Наращиваемый Протокол Инициализации (EPP), Систему Управления Службами/Механизированный Общий Интерфейс (MGI) 800 и другие оперативные протоколы инициализации. Совокупность LUE только для чтения может быть развернута в архитектуре типа концентратора и спицы (линии, идущей от

центра к периферии) для обеспечения возможности высокоскоростного поиска, сопровождаемого объемными добавочными обновлениями из OLTP сервера 140-1.

В альтернативном варианте осуществления данные могут быть распределены по нескольким OLTP серверам 140-1 - 140-S, каждый из которых может быть подсоединен к ГС 124. OLTP сервера 140-1 - 140-S могут получать из разных источников добавления, изменения и удаления (то есть график обновления) для соответствующих им баз данных 142-1 - 142-S (не изображены для ясносги). OLTP сервера 140-1 - 140-S могуг передавагь обновления через ГС 124 в систему 100, которая может содержагь копии баз данных 142-1 - 142-S, другие динамически создаваемые данные и т.д. Например, в варианте осуществления для геолокации OLTP сервера 140-1 - 140-S могут получать график обновления из групп удаленных датчиков. В другом альтернативном варианте осуществления множество сетевых компьютеров 120-1 - 120-N также может получать через ГС 124 или ЛС 122 добавления, изменения и удаления (то есть график обновления) из разных источников. В этом варианте осуществления множесгво сетевых компьютеров 120-1 - 120-N могуг передавагь в систему 100 обновления, а также запросы.

В варианте осуществления разрешения DNS каждый PE (например, каждый из множества сетевых компьютеров 120-1 - 120-N) может комбинировать, или мультиплексировать, несколько сообщений запросов DNS, полученных через глобальную сеть связи (например, ГС 124), в одиночный Суперпакет Запроса и передавать Суперпакет Запроса через локальную сеть связи (например, ЛС 122) в LUE (например, систему 100). LUE может комбинировать, или мультиплексировать, несколько ответов на сообщения-запросы DNS в одиночный Суперпакет Ответа и передавать Суперпакет Ответа через локальную сеть связи в соответствующий PE. В основном, максимальный размер Суперпакета Ответа или Запроса может быть ограничен максимальным блоком передачи (MTU) физического сетевого уровня (например, Gigabit Ethernet). Например, стандартные размеры сообщения запроса и ответа DNS, не превышающие 100 байтов и 200 байтов соответственно, позволяют мультиплексировать более 30 запросов в одиночный Суперпакет Запроса, а также мультиплексировать более 15 ответов в одиночный Суперпакет Ответа. Однако в одиночный Суперпакет Запроса может быть включено меньшее количество запросов (например, 20 запросов), чтобы при формировании ответа избежать переполнения MTU (например, в 10 ответов). Для MTU большего размера количество мультиплексированных запросов и ответов соответственно может быть увеличено.

Каждый многозадачный PE может иметь входящий поток и исходящий поток для управления запросами и ответами DNS соответственно. Например, входящий поток может демаршалинговать (распаковывать) компоненты запроса DNS из входящих пакетов запросов DNS, полученных через глобальную сеть связи, и мультиплексировать несколько миллисекунд запросов в одиночный Суперпакет Запроса. Затем входящий поток может передавать Суперпакет Запроса через локальную сеть связи в LUE. Обратно, исходящий поток может получать Суперпакет Ответа из LUE, демультиплексировать содержащиеся в нем ответы и маршалинговать (упаковывать в стандартный формат) различные поля в действительный ответ DNS, который затем может быть передан через глобальную сеть связи. В основном, как отмечено выше, могут поддерживаться другие варианты осуществления большого объема, основанные на запросах.

В варианте осуществления Суперпакет Запроса может также содержать информацию о состоянии, соответствующую каждому запросу DNS, например такую, как исходный адрес, вид протокола и т.д. LUE может включать в Суперпакет Ответа информацию о состоянии и соответствующие ответы DNS. Затем каждый PE может формировать и возвращать сообщения действительных ответов DNS с использованием информации, переданной из LUE. Следовательно, каждый PE может, преимущественно, функционировать как устройство, не меняющее своего состояния в процессе исполнения, то есть действительные ответы DNS могут быть сформированы из информации, содержащейся в Суперпакете Ответа. В основном, LUE может возвращать Суперпакет Ответа в PE, из которого исходил входящий Суперпакет, однако, очевидны другие возможные варианты.

В альтернативном варианте осуществления каждый PE может поддерживать информацию о состоянии, соответствующую каждому запросу DNS, и включать в Суперпакет Запроса ссылку, или дескриптор, на информацию о состоянии. LUE может включать в Суперпакет Ответа ссылки на информацию о состоянии и соответствующие ответы DNS. Затем каждый PE может формировать и возвращать сообщения действительных ответов DNS с использованием ссылок на информацию о состоянии, переданных из LUE, а также поддерживаемой им информации о состоянии. В этом варианте осуществления LUE может возвращать Суперпакет Ответа в PE, из которого исходил входящий Суперпакет.

На фиг. 2 представлена структурная схема архитектуры типа концентратора и спиц (звездообразной топологии) согласно варианту осуществления настоящего изобретения. В основном, система может содержать локальную базу данных 200 (которая может находиться в центральном OLTP концентраторе 140) и одну или большее количество удаленных баз данных 210 (которые могут находиться в устройствах LUE 100), соединенных с локальной базой данных 200 посредством любого механизма соединения, например Интернета или ЛС 122. Базы данных могут передавать и получать данные обновления.

Согласно фиг. 3 в вариантах осуществления настоящего изобретения локальная база данных 200 передает в удаленную базу данных 210 F файлов 300-1 - 300-F отправки и инициализирующий файл 310 отправки для обновления удаленной базы данных 210. Файлы обновления могут передаваться индивиду-

ально или в пакетах, например несколько файлов 300 отправки, один файл 300 отправки и один инициализирующий файл 310 отправки, множество файлов 300 отправки и один инициализирующий файл 310 отправки, одиночный файл 300 отправки или одиночный инициализирующий файл 310 отправки.

В варианте осуществления настоящего изобретения процессор 104 может получать из локальной базы данных 200 файл 300 отправки и/или инициализирующий файл 310 отправки, содержащий данные обновления. Система 150 может получать файл 300 отправки и инициализирующий файл 310 отправки в удаленной базе данных 210 через интерфейс 118 связи. Затем процессор 104 может сравнить данные обновления в файле 300 отправки или в инициализирующем файле 310 отправки с соответствующими данными в удаленной базе данных 210. Если в удаленной базе данных 210 данные другие, то процессор 104 может применить файл 300 отправки или инициализирующий файл 310 отправки к удаленной базе данных 210. Соответственно впоследствии удаленная база данных 210 может быть обновлена данными, соответствующими данным обновления в локальной базе данных 200.

Фиг. 4 изображает файл 300 отправки согласно варианту осуществления настоящего изобретения. Поля файла 300 могут включать в себя, например, идентификатор 400 файла, время 402 формирования файла, количество 404 транзакций N в файле, полный размер 406 файла, контрольную сумму 408 или любой подобный индикатор контроля ошибок и транзакции 410-1 - 410-N (содержащие идентификаторы транзакций). Указанные поля файла отправки являются возможными вариантами, предназначенными для иллюстрации, но не для ограничения контекста вариантов осуществления настоящего изобретения. В файл 300 отправки может быть включено любое поле, которое может быть полезным.

Файл 300 отправки содержит изменения в локальной базе данных 200 между двумя моментами времени. Эти изменения могут включать в себя, например, добавления новых идентификаторов (то есть идентификаторов записей данных), удаления существующих идентификаторов, изменения одной или большего количества записей данных, соответствующих идентификатору, переименование идентификатора, холостую команду и т.д. Одно или большее количество указанных изменений может производиться последовательно и может называться транзакциями. Файл 300 отправки может содержать уникальные идентификаторы этих транзакций. Транзакции могут быть записаны в файле 300 отправки в том порядке, в котором они совершены в локальной базе данных 200. Дополнительно для транзакций, содержащих более одного изменения, изменения могут быть записаны внутри транзакции в том порядке, в котором они произведены в локальной базе данных 200.

В основном, идентификаторы транзакций могут назначаться транзакциям в случайном порядке. То есть идентификаторы транзакций не обязательно монотонно возрастают во времени. Например, две последовательные транзакции могут иметь идентификаторы транзакций 10004 и следующий за ним 10002. Соответственно очередность совершения транзакции может быть определена ее размещением в текущем файле 300-F или ее размещением в предыдущем файле 300-(F-1). В основном, для полного завершения обновления удаленной базы данных с применением одного файла отправки транзакции не могут охватывать соседние файлы 300. Это предотвращает прерывание обновления из-за сетевой задержки, которая может привести к ошибочным данным в удаленной базе данных 210.

Фиг. 5 изображает инициализирующий файл 310 отправки согласно варианту осуществления настоящего изобретения. Поля инициализирующего файла 310 отправки могут включать в себя, например, идентификатор 500 файла, время 502 формирования файла, количество 504 транзакций N в файле, полный размер 506 файла, контрольную сумму 508 или любой подобный индикатор контроля ошибок и копию 516 (данных) полной локальной базы данных. Инициализирующий файл 310 отправки может дополнительно содержать поле 510, являющееся идентификатором 400 файла последнего файла 300 отправки, сформированного до формирования файла 310, и поле 512, являющееся идентификатором последней транзакции, фиксированной в локальной базе данных 200 до формирования инициализирующего файла 310 отправки. Данные в локальной и удаленной базах данных 200, 210 могут быть распределены по таблицам, резидентно хранящимся в базах данных 200, 210. Базы данных 200, 210 могут поддерживать произвольное количество таблиц. Следовательно, когда база данных имеет таблицы, инициализирующий файл 310 отправки может содержать для каждой таблицы поле, указывающее количество записей, записанных в таблице. Например, база данных имен доменов может содержать таблицу доменов и таблицу серверов доменных имен. Следовательно, инициализирующий файл отправки может содержать поле, указывающее количество записей в таблице доменов, и поле, указывающее количество записей в таблице серверов доменных имен. Поле может определять, например, имя таблицы, ключ, используемый для индексирования записей в таблице, и количество записей в таблице. Дополнительно инициализирующий файл 310 отправки может содержать поле, которое указывает версию инициализирующего файла 310 отправки, обычно 1.0. Указанные поля инициализирующего файла отправки являются возможными вариантами, предназначенными для иллюстрации, но не ограничения, контекста вариантов осуществления настоящего изобретения. В инициализирующий файл 310 отправки может быть включено любое поле, которое может быть полезным.

Как отмечено выше, инициализирующий файл 310 отправки может содержать, например, копию полной локальной базы данных 200, унифицированную по считыванию данных. Инициализирующий файл 310 отправки может стать унифицированным с локальной базой данных 200 в момент времени t

между ts и tf, где ts является временем начала формирования инициализирующего файла 310 отправки, а tf является временем завершения формирования. По существу, единственной операцией, которая может появиться в инициализирующем файле 310 отправки, является операция "добавления". То есть при формировании инициализирующего файла 310 отправки в него могут быть записаны копии полной локальной базы данных 200 в момент времени t. Следовательно, для записи локальной базы данных 200 в инициализирующий файл 310 отправки может быть выполнена операция "добавления". Идентификаторы могут быть записаны в инициализирующий файл 310 отправки в случайном порядке. В другом варианте, при наличии внешних идентификаторов записи данных, на которые имеются ссылки, могут быть записаны до записи данных, имеющей ссылку.

Добавление полей 510 и 512 может обеспечивать инициализирующий файл 310 отправки некоторой информацией о файлах 300 отправки, которые могут быть сформированы и переданы в удаленную базу данных 210 во время формирования инициализирующего файла 310 отправки. Однако формирование файла 300 отправки и формирование инициализирующего файла 310 отправки могут быть не связаны друг с другом в отношении отсутствия зависимости между ними при формировании. Такая структура и процесс могут предотвратить менее эффективный подход, при котором формирование и применение файла отправки может быть приостановлено до завершения формирования инициализирующего файла отправки. При продолжении формирования и применения файлов 300 отправки во время формирования инициализирующего файла 310 отправки, как в варианте осуществления настоящего изобретения, может осуществляться строгий контроль ошибок файлов 300 отправки, а также наложение ограничений на удаленную базу данных 210, например могут быть сделаны ограничения в отношении однозначности или ограничения на внешние идентификаторы. Наложение ограничений может защищать целостность данных в удаленной базе данных 210 посредством отклонения транзакций, нарушающих реляционные модели удаленной базы данных 210. Например, ограничение в отношении однозначности может препятствовать повторному сохранению в базе данных 210 одного и того же ключа.

На фиг. 6 представлена временная диаграмма, иллюстрирующая формирование файла отправки и инициализирующего файла отправки, согласно варианту осуществления настоящего изобретения. Согласно фиг. 6 файлы 300 отправки (с sf-1 no sf-21) формируются с регулярными интервалами времени. В альтернативном варианте осуществления файлы 300 отправки могут формироваться с нерегулярными интервалами времени. В общем случае формирование файла отправки не занимает интервал времени полностью. Например, если файлы формируются в 5-минутные интервалы времени, то для завершения формирования файла не требуется полностью 5 мин. Дополнительно, если в локальной базе данных 200 производятся изменения во время формирования файла 300 отправки, то эти изменения будут собраны в следующем файле 300 отправки. Например, если формирование файла отправки sf-4 начинается в 12:05:00 и завершается в 12:05:02, то любые изменения в локальной базе данных 200, произведенные между 12:05:00 И 12:05:02, собираются в файл отправки sf-5 (например, 300-5), который охватывает интервал времени с 12:05:00 до 12:10:00.

Фиг. 6 иллюстрирует файлы 300-5 и 300-19 отправки. В указанных файлах, среди других полей, изображены идентификатор 601 файла (sf-5, sf-19), время 603 формирования файла и идентификаторы 605 транзакций (например, 10002). Следует отметить, что идентификаторы транзакций могут быть не упорядоченными монотонно. Как упомянуто выше, идентификаторы транзакций могут иметь произвольные значения. Однако непосредственно соответствующие транзакции записаны в файл 300 отправки в порядке, в котором они совершены в локальной базе данных 200.

Так как формирование инициализирующего файла 310 отправки и формирование файла 300 отправки могут быть не связаны, то инициализирующий файл 310 отправки может формироваться в любой момент времени. Например, инициализирующий файл 310 отправки может формироваться до, в течение или после формирования файла 300 отправки. Фиг. 6 иллюстрирует инициализирующий файл 310 отправки, формируемый в промежутке времени между формированием четвертого и пятого файлов отправки (например, sf-4 и sf-5).

В варианте осуществления инициализирующий файл 310 отправки может среди других полей содержать идентификатор 610 файла (isf-1), идентификатор 615 файла для последнего файла отправки, сформированного перед формированием инициализирующего файла отправки, и идентификатор 620 транзакции для последней транзакции, завершенной перед формированием инициализирующего файла отправки. В данном возможном варианте последним сформированным файлом отправки является файл отправки sf-4, а последней завершенной транзакцией является транзакция 10001. Формирование инициализирующего файла 310 отправки начинается 611 в 12:07:29. Когда начинается формирование инициализирующего файла 310 отправки, первая половина транзакций в файле 300-5 отправки (sf-5), транзакции 10002, 10005 и 10001 были уже завершены в локальной базе данных 200. Соответственно инициализирующий файл 310 отправки может иметь информацию об этих транзакциях и может собрать эти транзакции в инициализирующем файле 310 отправки. Однако инициализирующий файл 310 отправки не может иметь информации о последующих транзакциях 10003 и 10004, которые совершаются после начала формирования инициализирующего файла отправки.

При формировании инициализирующего файла 310 отправки файлы отправки, начиная с файла 300-5 отправки, могут продолжать формироваться с регулярными интервалами времени. Эти файлы могут передаваться удаленной базе данных 210 и применяться.

Формирование инициализирующего файла 310 отправки может быть завершено в 1:15:29, в промежутке времени между формированием 18-го и 19-го файлов 300 отправки (sf-18 и sf-19), и не может воздействовать на формирование 19-го файла 300-19 отправки.

После получения и загрузки в удаленной базе данных 210 инициализирующего файла 310 отправки удаленная база данных 210 может не учитывать файлы отправки, сформированные до формирования инициализирующего файла 310 отправки. Это возможно, например, по причине того, что инициализирующий файл 310 отправки содержит все изменения в локальной базе данных 200, которые были записаны в предыдущие файлы 300 отправки. В этом возможном варианте удаленной базе данных 210 может не потребоваться учитывать файлы отправки с первого по четвертый (с sf-1 no sf-4). Изменения, записанные в этих файлах отправки, с sf-1 по sf-4, также могут быть записаны в инициализирующем файле 310 отправки. Указанные предыдущие файлы отправки (с sf-1 по sf-4) могут быть удалены или, в другом варианте, заархивированы.

Аналогично, удаленная база данных 210 может не учитывать транзакции, завершенные до формирования инициализирующего файла 310 отправки, которые были включены в файл 300 отправки, сформированный впоследствии. Эти транзакции могут быть включены в инициализирующий файл 310 отправки при его формировании. Например, поэтому удаленной базе данных 210 может не потребоваться учитывать первые три транзакции 10002, 10005, 10001 из файла отправки sf-5. Эти транзакции, записанные в файл отправки sf-5, могут также быть записаны в инициализирующем файле 310 отправки. Данные завершенные транзакции могут быть удалены или, в другом варианте, заархивированы.

На фиг. 7 представлена блок-схема варианта осуществления настоящего изобретения, в котором могут быть сформированы файлы обновления локальной базы данных. Система может формировать (705) несколько периодических обновлений, основанных на добавочных изменениях в локальной базе данных. Каждое обновление может содержать одну или большее количество транзакций. Затем система может передать (710) периодические обновления в удаленную базу данных. При формировании периодических обновлений система может начать формировать (715) инициализирующее обновление в момент времени запуска. Инициализирующее обновление может содержать версию полной локальной базы данных. Система может определить (720) последнее периодическое обновление, сформированное до момента времени запуска, и последнюю транзакцию, завершенную до момента времени запуска. Затем система может передать (725) инициализирующее обновление в удаленную базу данных. Инициализирующее обновление может содержать идентификатор обновления, соответствующий последнему сформированному периодическому обновлению, и идентификатор транзакции, соответствующий последней завершенной транзакции.

Например, OLTP 140 может формировать (705) файлы 300 отправки с некоторым регулярным или нерегулярным интервалом времени. Затем OLTP 140 может передать (710) файлы 300 отправки в удаленную базу данных 210. При формировании файлов 300 отправки OLTP 140 может начать формирование (715) инициализирующего файла 310 отправки в момент времени 611 запуска. Инициализирующий файл 310 отправки может содержать копию полной локальной базы данных 200. Затем OLTP 140 может определить (720) последний файл 300 отправки, сформированный до момента времени 611 запуска для формирования инициализирующего файла 310 отправки, и последнюю транзакцию, завершенную до момента времени 611 запуска для формирования инициализирующего файла 310 отправки. Затем OLTP 140 может передать (725) инициализирующий файл 310 отправки в удаленную базу данных 210. Инициализирующий файл 310 отправки может содержать идентификатор 615 файла отправки, соответствующий последнему сформированному файлу 300 отправки, и идентификатор транзакции 620, соответствующий последней завершенной транзакции.

На фиг. 8 представлена блок-схема варианта осуществления настоящего изобретения, в котором удаленная база данных может получать файлы обновлений из локальной базы данных. Система может получить (805) несколько периодических обновлений. Каждое обновление может содержать одну или большее количество транзакций. Периодические обновления могут быть получены отдельно или в пакетах. В некоторый момент времени система может получить (810) инициализирующее обновление. Инициализирующее обновление может содержать версию полной локальной базы данных. Система может считать (815) из инициализирующего обновления идентификатор последнего периодического обновления и идентификатор последней транзакции. Затем система может определить (820) последнее периодическое обновление, соответствующее идентификатору обновления, и последнюю транзакцию, соответствующую идентификатору транзакции. Периодическое обновление и транзакция могут быть соответственно последним сформированным обновлением и последней завершенной транзакцией до формирования инициализирующего обновления. Система может применить (825) к удаленной базе данных оставшиеся незавершенные транзакции в соответствующем периодическом обновлении. Затем система может применить (830) к удаленной базе данных оставшиеся периодические обновления, сформированные по-

сле последнего периодического обновления. Применение инициализирующего обновления преимущественно восполняет любые ранее потерянные периодические обновления.

Например, LUE 100 может получить (805) файлы 300 отправки с некоторым регулярным или нерегулярным интервалом времени. Файлы 300 отправки могут быть получены отдельно или в пакетах. В некоторый момент времени LUE 100 может получить (810) инициализирующий файл 310 отправки. LUE 100 может считать (815) из инициализирующего файла 310 отправки идентификатор 615 файла отправки и идентификатор 620 транзакции. Затем LUE 100 может определить (820) файл 300 отправки, соответствующий идентификатору 615 файла отправки, и транзакцию 605, соответствующую идентификатору 620 транзакции. Файл отправки и транзакция могут быть соответственно последним сформированным файлом отправки и последней завершенной транзакцией до формирования инициализирующего файла 310 отправки. LUE 100 может применить (825) к удаленной базе данных 210 оставшиеся незавершенные транзакции 605 в соответствующем файле 300 отправки. Затем LUE 100 может применить (830) к удаленной базе данных 210 оставшиеся файлы 300 отправки, которые следуют за последним файлом отправки sf-4.

В альтернативном варианте осуществления, например, LUE 100 может отбросить или заархивировать файлы 300 отправки, которые не были применены к удаленной базе данных 210 и/или имеют время 603 формирования до времени 611 формирования инициализирующего файла отправки. Отброшенные или заархивированные файлы 300 отправки могут включать в себя файл отправки sf-4, соответствующий идентификатору 615 файла отправки.

Понятно, что после применения инициализирующего файла 310 отправки любые более поздние файлы 300 отправки, которые могли быть уже применены к удаленной базе данных 210, могут не сохраниться, так как удаленная база данных 210 может стать унифицированной по считыванию с инициализирующим файлом 310 отправки. Соответственно эти более поздние файлы 300 отправки могут быть применены повторно.

В варианте осуществления настоящего изобретения файлы 300 отправки и инициализирующий файл 310 отправки могут передаваться из локальной базы данных 200 в удаленную базу данных 210 без уведомления об успешном получении данных, то есть без сигнала ACK/NACK для указания того, что файлы были успешно получены. Это преимущественно сокращает дополнительную служебную сигнализацию, которую может создавать сигнал ACK/NACK.

В альтернативном варианте осуществления из удаленной базы данных 210 может передаваться сигнал ACK/NACK для указания того, что файлы успешно получены. В этом варианте осуществления сигнал ACK/NACK может передаваться в системах с ненадежной связью.

На фиг. 9 представлена блок-схема другого варианта осуществления настоящего изобретения, в котором система может проверять достоверность файлов обновления, переданных из локальной базы данных и полученных в удаленной базе данных. Здесь система может передать (905) множество периодических обновлений. Каждое обновление может содержать одну или большее количество транзакций. Периодические обновления могут быть переданы отдельно или в пакетах. В некоторый момент времени система может передать (910) инициализирующее обновление и применить инициализирующее обновление к удаленной базе данных. Инициализирующее обновление может содержать версию полной локальной базы данных. Сначала система, посредством сравнения баз данных, может идентифицировать (915) расхождения между локальной и удаленной базами данных. Система может определить (920), являются ли расхождения действительными или ошибочными. Затем система может применить (925) к удаленной базе данных периодические обновления согласно варианту осуществления настоящего изобретения. Этот вариант осуществления преимущественно может обеспечивать отсутствие ошибок в удаленной базе данных, являющихся результатом получения обновлений из локальной базы данных.

Например, OLTP 140 может передать (905) в удаленную базу данных 210 файлы 300 отправки с некоторым регулярным или нерегулярным интервалом времени. Файлы 300 отправки могут быть переданы отдельно или в пакетах. В некоторый момент времени OLTP 140 может передать (910) инициализирующий файл 310 отправки в LUE 100, и LUE 100 может применить инициализирующий файл 310 отправки к удаленной базе данных 210. OLTP 140 может сравнить локальную базу данных 200 с удаленной базой данных 210 и идентифицировать (915) расхождения между ними. Затем OLTP 140 может определить (920), являются ли расхождения действительными или ошибочными. Затем OLTP 140 может уведомить LUE 100 для применения (925) к удаленной базе данных 210 файлов 300 отправки согласно варианту осуществления настоящего изобретения. Затем LUE 100 может применить файлы 300 отправки к удаленной базе данных 210.

В альтернативном варианте осуществления система может применять файлы отправки и инициализирующий файл отправки до идентификации и проверки достоверности расхождений. В виде другого варианта система может применять файлы отправки и инициализирующий файл отправки после идентификации и проверки достоверности расхождений.

Понятно, что процесс проверки достоверности может быть выполнен на любых данных, переданных из источника в адресат через сеть связи для применения переданных данных к адресату.

На фиг. 10A представлена блок-схема варианта осуществления проверки достоверности файла отправки и инициализирующего файла отправки согласно настоящему изобретению. После передачи в удаленную базу данных нескольких периодических обновлений и инициализирующего обновления система может проверить достоверность этих обновлений. Каждое обновление может содержать одну или большее количество транзакций, выполненных в локальной базе данных. Каждая транзакция может содержать одно или большее количество событий. Событие является действием или возможным осуществлением в базе данных, например, добавления, изменения, удаления и т.д., в отношении данных в базе данных.

Сначала система может сравнить (1000) запись в удаленной базе данных с соответствующей записью в локальной базе данных. Система может сформировать (1005) исключение, описывающее расхождение между записями удаленной и локальной баз данных, причем исключение может быть сформировано для каждого расхождения. Расхождением может быть любое отличие по меньшей мере в одном значении данных между двумя версиями одной записи. Например, в локальной базе данных может быть запись данных (12345, xyz.com, 123.234.345). Соответствующая запись данных в удаленной базе данных, которая предполагается идентичной, может соответствовать (12345, abc.com, 123.234.345). Соответственно во втором значении данных записи имеется расхождение. Следовательно, вариант осуществления настоящего изобретения может сформировать исключение, которое описывает указанное расхождение. Исключение может описывать расхождение, просто указывая на то, что имеется расхождение; определяя местоположение расхождения; описывая отличие между двумя значениями данных при расхождении и т.д. Запись данных в локальной базе данных соответствует записи данных в удаленной базе данных (и наоборот), если две записи обязательно содержат идентичные данные.

Понятно, что расхождение может относиться к отличию в одном или большем количестве значений данных в записи или к записи в целом.

Система может сопоставить (1010) каждому исключению идентификатор исключения, причем идентификатор исключения может соответствовать идентификатору записи. Например, запись данных (12345, xyz.com, 123.234.345) может иметь идентификатор d10. Соответственно идентификатором исключения может быть также d10. Каждое исключение может быть классифицировано как принадлежащее любому из нескольких видов исключений (или расхождений). Может быть сформирован список исключений для включения в него видов исключений и идентификатора исключения для сгруппированных там исключений. Список исключений и различные виды исключений подробно описаны ниже. Система также может сопоставить (1015) каждому событию в обновлении идентификатор события, причем идентификатор события может соответствовать идентификатору записи. Например, запись данных (12345, xyz.com, 123.234.345) может иметь идентификатор d10. Соответственно идентификатором события может быть также d10. Каждое событие в обновлении может быть найдено из предыстории событий. Предысторией событий может быть листинг и т.д. событий, выполненных на записях локальной базы данных за некоторый период времени. Предыстория событий подробно описана ниже.

Затем система может определить (1020), является ли обновление записи достоверным. На фиг. 10B представлена блок-схема варианта осуществления определения проверки достоверности. Это определение может быть сделано следующим образом. Может быть осуществлено сравнение (1022) каждого события с каждым исключением. Если каждое исключение подтверждено (1024) событием, то обновление может быть определено (1026) как достоверное, и данное обновление может быть применено к удаленной базе данных. Иначе, если каждое исключение не подтверждено (1024) событием, то обновление может быть определено (1028) как недостоверное, и исключения могут регистрироваться как ошибки. Исключение может быть подтвержденным, когда идентификатору исключения соответствует идентификатор события и соответствующее событие соответствует достоверной последовательности событий, соответствующих виду исключения. Достоверные последовательности подробно описаны ниже. Если исключение подтверждено, то система может удалить идентификатор исключения из списка исключений. Подтвержденное исключение может указывать, что расхождение является достоверным, например удаленная база данных еще не получила обновление, но при получении обновления действительно будет соответствовать локальной базе данных.

При проверке достоверности система может идентифицировать скрытые ошибки или сбои в периодическом и инициализирующем обновлениях. Система может обеспечивать возможность структурной и семантической корректности данных обновления, возможность успешного применения данных обновления, которое не вызывает формирования исключений или иного нежелательного останова, возможность точного обнаружения ошибок при сравнении локальной и удаленной баз данных между собой и невозможность случайного удаления значимых данных. Система может обеспечивать возможность успешного применения к удаленной базе данных периодического и инициализирующего обновлений.

Преимущественно, при проверке достоверности могут быть обнаружены многие ошибки, возникающие при попытке применения обновления к удаленной базе данных. Например, при попытке применения могут обнаруживаться ошибки централизации данных, предупреждения о том, что объект уже существует в удаленной базе данных, или предупреждения о том, что имеется нарушение внешнего идентификатора. Следовательно, после выполнения процесса проверки достоверности согласно варианту

осуществления настоящего изобретения система может сделать попытку применения указанных обновлений к удаленной базе данных. Попытка может быть неуспешной, что может указывать на наличие в обновлениях дополнительных ошибок, которые делают обновление недостоверным. Соответственно, дополнительные попытки применения указанных обновлений к удаленной базе данных не могут быть сделаны.

В альтернативном варианте осуществления до выполнения проверки достоверности может быть сделана попытка применить по меньшей мере одно из обновлений. Если попытка является неуспешной, то проверка достоверности может быть опущена и обновление отброшено. С другой стороны, если попытка является успешной, то может быть выполнена проверка достоверности и достоверное обновление сохраняется, а для недостоверного обновления регистрируются расхождения.

В возможном варианте осуществления OLTP 140 может осуществлять проверку достоверности файлов 300 отправки и инициализирующих файлов 310 отправки для обеспечения успешного применения к удаленной базе данных 210 файлов 300 отправки и инициализирующих файлов 310 отправки.

В альтернативных вариантах осуществления проверку валидности могут выполнять сетевые компьютеры 121, LUE 100 или любая комбинация существующих систем.

Согласно фиг. 10A OLTP 140 может сравнить локальную базу данных 200 и удаленную базу данных 210 для определения любых исключений (или расхождений) между ними. Исключения могут включать три вида: в удаленной базе данных 210 могут существовать данные, которые отсутствуют в локальной базе данных 200; в локальной базе данных 200 могут существовать данные, которые отсутствуют в удаленной базе данных 210; или соответствующие данные могут существовать и в локальной базе данных 200, и в удаленной базе данных 210, но данные могут отличаться. Безусловно, соответствующие данные могут существовать и в локальной базе данных 200, и в удаленной базе данных 210, и данные могут быть идентичными, в этом случае данные могут считаться достоверными, следовательно от OLTP 140 не требуется никакой дополнительной обработки.

Понятно, что расхождение может относиться к одному или большему количеству значений данных в записи или к записи в целом.

Соответственно OLTP 140 может сравнить (1000) соответствующие записи в локальной базе данных 200 и удаленной базе данных 210. OLTP 140 может сформировать (1005) исключение, которое описывает расхождение между записью в удаленной базе данных 210 и записью в локальной базе данных 200, причем исключение может быть сформировано для каждого расхождения. OLTP 140 может сопоставить (1010) каждому исключению идентификатор исключения, причем идентификатор исключения может соответствовать идентификатору записи. Может быть сформирован список исключений для включения видов исключений и идентификатора исключения для исключения, принадлежащего к данному виду исключения. В варианте осуществления исключение может быть определено как исключение (или расхождение) "Списка 1", если исключение принадлежит к первому виду исключений, исключение "Списка 2", если исключение принадлежит ко второму виду исключений, или исключение "Списка 3", если исключение принадлежит к третьему виду исключений. Фиг. 11 изображает возможный список 1140 исключений.

Понятно, что наличие идентификатора исключения в списке исключений не означает, что файл 300 отправки или инициализирующий файл 310 отправки не пригоден, так как, например, все три вида исключений могут возникать обоснованно из-за задержки во времени между изменениями в локальной базе данных 200 и обновлениями, применяемыми к удаленной базе данных 310. Такая задержка, например, может быть вызвана перегрузкой сети связи. По существу, проверка достоверности может обеспечивать механизм для исключения обоснованных исключений из ошибочных данных.

Для инициализирующего файла 310 отправки OLTP 140 может сравнить локальную базу данных 200 и удаленную базу данных 210 посредством выполнения двунаправленного просмотра всех таблиц в обеих базах данных 200, 210. То есть все данные в локальной базе данных 200 могут сравниваться относительно всех данных в удаленной базе данных 210. Затем все данные в удаленной базе данных 210 могут сравниваться относительно всех данных в локальной базе данных 200. Это преимущественно обеспечивает всестороннее сравнение баз данных 200, 210 для обнаружения всех расхождений.

Для файла 300 отправки OLTP 140 может сравнить только те записи данных в локальной базе данных 200 и удаленной базе данных 210, которые записаны в файле 300 отправки. Это преимущественно обеспечивает быстродействующий запрос для обнаружения заданных расхождений.

В другом варианте может быть произведена случайная выборка данных в инициализирующем файле 310 отправки и/или файле 300 отправки. Затем OLTP 140 может сравнить данные, выбранные случайным образом, в локальной базе данных 200 и удаленной базе данных 210.

Список 1140 исключений может соответствовать отсутствующим событиям, например добавления (add), изменения (mod) и удаления (del) для локальной базы данных 200, которые не согласуются с удаленной базой данных 210. Следовательно, для идентификации таких событий-кандидатов OLTP 140 может исследовать последние транзакции, фиксированные в локальной базе данных 200. В основном, в таблице регистрации, которая хранится в локальной базе данных 200, может быть сформирован элемент для каждой фиксированной транзакции. Элемент может содержать идентификатор записи, которая была

изменена, транзакцию (или событие), изменившую запись (например, add, mod и/или del), порядковый номер регистрации, указывающий очередность транзакции, и т.д.

Возможная таблица 1100 регистрации изображена на фиг. 11. В этом возможном варианте файл 300 отправки содержит транзакции 1108-1114, изображенные в таблице регистрации 1100. Первый элемент 1101 указывает, что в первой транзакции 1108 данные (сервера доменных имен) n1 и n2 были добавлены к данным (домен), соответствующим идентификатору d1. Следовательно, идентификатором является d1, событием является добавление "add", а порядковым номером регистрации является 11526. Аналогично, второй элемент 1102 указывает, что во второй транзакции 1109 данные n8 и n9 были добавлены к данным, соответствующим идентификатору d2. Третий элемент 1103 указывает, что в третьей транзакции 1110 данные, соответствующие идентификатору d3, были удалены. Четвертый элемент 1104 указывает, что в четвертой транзакции 1111 данные, соответствующие идентификатору d1, были изменены для добавления данных n5. Для пятой транзакции 1112 пятый элемент 1105 указывает, что данные n6 и n7 были добавлены к данным, соответствующим идентификатору d3. Для шестой транзакции 1113 шестой элемент 1106 указывает, что данные, соответствующие идентификатору d4, были изменены для удаления данных n3. R^{ый} элемент 1107 в R^{ой} транзакции 1114 указывает, что данные, соответствующие идентификатору d5, были удалены.

Соответственно, согласно фиг. 10A OLTP 140 может сопоставить (1015) каждому событию в обновлении идентификатор события, причем идентификатор события может соответствовать идентификатору записи. Каждое событие в обновлении может быть найдено из предыстории событий. Предыстория событий, индексированная и упорядоченная по идентификатору события, может быть сформирована из таблицы 1100 регистрации. Возможная предыстория 1120 событий изображена на фиг. 11. Здесь первый и четвертый элементы 1101, 1104 в таблице 1100 регистрации указывают изменения для данных, соответствующих идентификатору d1. Следовательно, предыстория 1120 событий содержит идентификатор d1 1121 и два события 1126, добавление "add" и последующее изменение "mod", выполненные на данных, соответствующих идентификатору d1. Второй элемент 1102 указывает изменения для данных, соответствующих идентификатору d2. Следовательно, предыстория 1120 событий содержит идентификатор d2 1122 и событие 1127 добавления "add". Предыстория 1120 событий содержит идентификатор d3 1123 и два события 1128, удаление "del" с последующим изменением "mod", указываемые третьим и пятым элементами 1103, 1105, которые содержат изменения для данных, соответствующих идентификатору d3. Шестой элемент 1106 указывает изменения для данных, соответствующих идентификатору d4. Соответственно предыстория 1120 событий содержит идентификатор d4 1124 и событие 1129 изменения "mod". R^{ый} элемент 1107 указывает изменения для данных, соответствующих идентификатору d5, и предыстория 1120 событий содержит идентификатор d5 1125 и событие 1130 удаления "del". Идентификаторы 1121-1125 упорядочены с d1 по d5.

Согласно фиг. 10A OLTP 140 может определить (1020), является ли обновление достоверным. Это определение может быть выполнено, например, согласно варианту осуществления по фиг. 10B. Сначала OLTP 140 может сравнить (1022) идентификаторы 1121-1125 событий с идентификаторами 1140 исключений для определения, какие идентификаторы имеют соответствия. Например, согласно фиг. 11 идентификатор 1121 события d1 в хронологии 1120 событий соответствует идентификатору исключения d1 в "Списке 2" списка 1140 исключений. После обнаружения соответствующих события и исключения OLTP 140 может определить (1024), подтверждено ли исключение событием. Подтверждение может быть выполнено следующим образом. Для каждого идентификатора 1121-1125 события в хронологии 1120 событий OLTP 140 может определить, является ли достоверной каждая последовательность событий 1126-1130 в хронологии 1120 событий. Это может быть сделано, например, путем исследования списка 1140 исключений для того, чтобы определить, к какому виду исключений принадлежит каждый идентификатор исключения, определения, какой должна быть достоверная последовательность событий для этого вида исключений, и затем поиска в предыстории 1120 событий соответствующего идентификатора события и последовательности событий для идентификатора события. Достоверные последовательности для каждого вида исключений подробно описаны ниже. Если последовательность событий 1126-1130 в хронологии 1120 событий соответствует достоверной последовательности, то соответствующий идентификатор события 1121-1125 имеет достоверную последовательность. По существу, исключение, соответствующее идентификатору исключения, может быть подтверждено. И соответствующая транзакция 1108-1114, содержащая указанный идентификатор события, является обоснованной, а не ошибочной. В этом случае OLTP 140 может удалить идентификатор исключения из списка 1140 исключений.

Для вида исключений "Списка 1" достоверной последовательностью событий может быть (mod)*(del). Эта последовательность может содержать последовательность из нулевого или большего количества событий изменения "mod", за которыми следует событие удаления "del", за которым следует произвольное событие. Вид исключений "Списка 1" может соответствовать данным, которые могут существовать в удаленной базе данных 210, но отсутствовать в локальной базе данных 200. В этом случае данные, возможно, были недавно удалены из локальной базы данных 200 и транзакция еще не была записана в файл 300 отправки. Следовательно, файл 300 отправки еще не мог быть применен к удаленной базе данных 210. Значит эти данные могут еще оставаться в удаленной базе данных 210. Это может считаться обосно-

ванным расхождением, так как предполагается, что в некоторый момент времени файл 300 отправки будет сформирован и применен к удаленной базе данных 210. Итак, если для идентификатора исключения в "Списке 1" из списка 1140 исключений в хронологии 1120 событий обнаружена любая такая последовательность 1126-1130, то соответствующая транзакция может считаться достоверной.

Например, согласно фиг. 11 идентификатор d5 1125 и соответствующие ему данные были удалены из локальной базы данных 200, как иллюстрирует $R^{ый}$ элемент 1114 таблицы 1100 регистрации и указывает хронология 1120 событий. Во время проверки достоверности d5 был удален из локальной базы данных 200, но не удален из удаленной базы данных 210. Итак, список 1140 исключений содержит идентификатор d5 в "Списке 1". Согласно хронологии 1120 событий событием 1130, соответствующим идентификатору d5 1125, является удаление "del". OLTP 140 может сравнить достоверную последовательность вида исключений "Списка 1", то есть (mod)*(del), с событием d5 1130 в хронологии 1120 событий. Так как достоверная последовательность "Списка 1" и событие 1130 соответствуют, то транзакция 1114 удаления, соответствующая идентификатору d5, может считаться обоснованной и не является ошибкой. Соответственно идентификатор d5 может быть удален из списка 1140 исключений.

Достоверной последовательностью событий для вида исключений "Списка 2" может быть (add). Эта последовательность может содержать событие добавления "add", за которым следует произвольное событие. Вид исключений "Списка 2" может соответствовать данным, которые существуют в локальной базе данных 200, но не существуют в удаленной базе данных 210. В этом случае данные могли быть недавно добавлены в локальную базу данных 200 и транзакция еще не была записана в файл 300 отправки. Следовательно, файл 300 отправки мог быть еще не применен к удаленной базе данных 210. Итак, данные могут не существовать в удаленной базе данных 210. Это также может считаться обоснованным расхождением, так как ожидается, что в некоторый момент времени файл 300 отправки будет сформирован и применен к удаленной базе данных 210. Соответственно, если для идентификатора исключения в "Списке 2" из списка 1140 исключений обнаружена любая такая последовательность 1126-1130 в хронологии 1120 событий, то соответствующая транзакция может считаться достоверной.

Согласно фиг. 11 идентификаторы d1 и d2 1121, 1123 могут быть сопоставлены данным, которые, например, первоначально были добавлены в локальную базу данных 200. Так как последовательности событий 1126, 1127 для них начинаются с события добавления "add", то идентификаторы d1 и d2 1121, 1123 соответствуют достоверным последовательностям для вида исключений "Списка 2". Соответственно транзакции 1108, 1109, содержащие указанные идентификаторы, могут считаться достоверными, и идентификаторы d1 и d2 могут быть удалены из списка 1140 исключений. Следует отметить, что идентификатор d3 1123 также в своей последовательности 1128 содержит событие добавления «add». Однако событие добавления «add» не является первым в последовательности 1128. Соответственно последовательность 1128 не относится к виду "Списка 2". Дополнительно, так как d3 не обозначен в "Списке 2" списка 1140 исключений, то OLTP 140 не может осуществить его проверку по достоверной последовательности для "Списка 2".

Достоверными последовательностями событий для вида исключений "Списка 3" могут быть (del), (add) или (mod). Эти последовательности могут содержать событие удаления "del", за которым следует событие добавления "add", за которым следует произвольное событие, или событие изменения "mod", за которым следует произвольное событие. Вид исключений "Списка 3" может соответствовать данным, которые существуют в обеих базах данных 200, 210, но отличны. В этом случае данные могли быть недавно изменены в локальной базе данных 200 и транзакция еще не была записана в файл 300 отправки. Следовательно, файл 300 отправки еще не мог быть применен к удаленной базе данных 210. Следовательно, данные, соответствующие идентификатору, могут быть еще не изменены в удаленной базе данных 210. Вновь это может считаться обоснованным расхождением, так как ожидается, что в некоторый момент времени файл 300 отправки будет сформирован и применен к удаленной базе данных 210. Соответственно, если для идентификатора исключения в "Списке 3" из списка 1140 исключений обнаружена любая такая последовательность 1126-1130 в хронологии 1120 событий, то соответствующая транзакция может считаться достоверной.

Например, согласно фиг. 11 идентификаторы d3 и d4 1123, 1124 могут быть сопоставлены данным, которые были изменены в локальной базе данных 200. В случае идентификатора d3 1123, идентификатор d3 1123 и его данные первоначально был удален, а затем добавлен обратно с новыми данными, так что последовательность событий 1128 может содержать удаление "del", за которым следует добавление "add". В случае идентификатора d4 1124, данные d4 были изменены для удаления данных, так что последовательность событий 1129 может содержать изменение "mod". Так как указанные последовательности событий 1128, 1129 соответствуют достоверным последовательностям для вида исключений "Списка 3", то соответствующие им транзакции 1110, 1112, 1113 могут считаться достоверными и идентификаторы d3 и d4 могут быть удалены из списка 1140 исключений.

Согласно фиг. 10B, если все исключения, обозначенные в списке 1140 исключений своими идентификаторами, были обоснованы (1024) событиями, например, если список 1140 исключений является пустым, то OLTP 140 может определить (1026) файл 300 отправки или инициализирующий файл 310 отправки как достоверный и уведомить LUE 100 для применения файла 300 отправки или инициализи-

рующего файла 310 отправки к удаленной базе данных 210. Затем LUE 100 может применить файл 300 отправки или инициализирующий файл 310 отправки к удаленной базе данных 210.

И наоборот, если все исключения не были подтверждены (1024) событиями, например, если список 1140 исключений не является пустым, то оставшиеся исключения могут указывать на ошибки в файле 300 отправки или инициализирующем файле 310 отправки. Соответственно OLTP 140 может определить (1028) файл 300 отправки или инициализирующий файл 310 отправки как недостоверный и зарегистрировать ошибки в файле ошибок.

В альтернативном варианте осуществления, если, например, файл 300 отправки или инициализирующий файл 310 отправки был определен как недостоверный, то после предварительно определенного периода времени OLTP 140 может повторить процесс проверки достоверности в отношении недостоверного файла 300 отправки или инициализирующего файла 310 отправки для подтверждения того, что расхождения действительно являются ошибками. Указанная предварительно определенная задержка обеспечивает сети большее время для передачи какого-либо медленного файла 300, 310 отправки и большее время на то, чтобы базы данных 200, 210 стали унифицированным по считыванию.

В варианте осуществления настоящего изобретения данные в удаленной базе данных 210 могут "отставать" от данных в локальной базе данных 200 на значительный интервал времени. Соответственно для сравнения баз данных 200, 210 и для обнаружения ошибок базы данных 200, 210 могут быть сделаны унифицированными по считыванию в тот момент времени, когда они станут точными копиями друг друга. В основном, удаленная база данных 210 может быть приведена с повтором всех завершенных транзакций к локальной базе данных 200, причем данные в удаленной базе данных 210 могут быть сделаны, по существу, идентичными данным в локальной базе данных 200.

Соответственно, для ускорения проверки достоверности любой сформированный в текущее время инициализирующий файл 310 отправки и последующие файлы 300 отправки могут быть применены к удаленной базе данных 210 до начала проверки достоверности. По существу, количество расхождений может быть существенно уменьшено. Такая пакетная обработка файлов 300, 310 отправки может быть определена как образование блоков. Первый и последний из этих файлов 300, 310 отправки в блоке может быть назван нижним и верхним «водяным знаком» соответственно. Первый блок, называемый начальным блоком, может содержать инициализирующий файл 310 отправки. Все последующие блоки, называемые конечными блоками, могут содержать только файлы 300 отправки.

Образование блоков может обеспечивать проверку достоверности группы вместо проверки достоверности в отдельности. Соответственно при обнаружении в блоке ошибки недостоверным может быть обозначен весь блок, а не только файл 300 отправки или инициализирующий файл 310 отправки, где возникла ошибка.

Механизмы и способы, соответствующие вариантам осуществления настоящего изобретения, могут быть реализованы с использованием универсального микропроцессора, запрограммированного согласно вариантам осуществления. Следовательно, варианты осуществления настоящего изобретения также включают носитель информации, считываемый компьютером, который может содержать инструкции, которые могут использоваться для программирования процессора для выполнения способа, соответствующего вариантам осуществления настоящего изобретения. Указанный носитель может включать в себя любой вид диска, включая гибкий диск, оптический диск, компакт-диски CD-ROM и т.д.

Выше подробно описано и проиллюстрировано несколько вариантов осуществления настоящего изобретения. Однако должно быть очевидно, что без отклонения от сущности и объема настоящего изобретения могут быть осуществлены изменения и модификации изобретения, охватываемые приведенным выше описанием и приложенной формулой изобретения.

ФОРМУЛА ИЗОБРЕТЕНИЯ

1. Способ обновления удаленной базы данных через сеть, включающий

формирование множества периодических обновлений, основанных на добавочных изменениях в локальной базе данных, причем каждое из множества периодических обновлений содержит по меньшей мере одну транзакцию,

передачу множества периодических обновлений в удаленную базу данных через сеть и,

при формировании множества периодических обновлений,

формирование инициализирующего обновления, содержащего версию локальной базы данных в момент времени запуска,

определение последнего периодического обновления из множества периодических обновлений на основе момента времени запуска,

определение последней транзакции на основе момента времени запуска и

передачу через сеть в удаленную базу данных инициализирующего обновления, идентификатора последнего периодического обновления и идентификатора последней транзакции.

2. Способ по п.1, в котором передача инициализирующего обновления включает

сопоставление последнему периодическому обновлению идентификатора последнего периодического обновления и

сопоставление последней транзакции идентификатора последней транзакции.

3. Способ по п.1, в котором множество периодических обновлений формируется с регулярными или нерегулярными интервалами времени.

4. Способ по п.1, в котором момент времени запуска формирования инициализирующего обновления идентичен моменту времени запуска формирования периодического обновления.

5. Способ по п.1, в котором момент времени запуска формирования инициализирующего обновления более поздний, чем момент времени запуска формирования периодического обновления.

6. Способ по п.1, в котором периодическое обновление содержит множество транзакций, каждая из которых имеет уникальный идентификатор транзакции.

7. Способ обновления удаленной базы данных через сеть, включающий

получение через сеть множества периодических обновлений, основанных на добавочных изменениях в локальной базе данных, причем каждое из множества периодических обновлений содержит по меньшей мере одну транзакцию,

получение через сеть инициализирующего обновления, содержащего версию локальной базы данных в момент времени запуска,

считывание из инициализирующего обновления идентификатора последнего периодического обновления,

считывание из инициализирующего обновления идентификатора последней транзакции,

определение последнего периодического обновления из идентификатора последнего периодического обновления, причем упомянутое определение последнего периодического обновления основано на моменте времени запуска,

определение последней транзакции из идентификатора последней транзакции, причем упомянутое определение последней транзакции основано на моменте времени запуска,

применение к удаленной базе данных транзакций, сформированных после последней транзакции, и

применение к удаленной базе данных периодических обновлений, сформированных после последнего периодического обновления.

8. Способ по п.7, дополнительно включающий

отбрасывание периодических обновлений, сформированных до момента времени запуска инициализирующего обновления.

9. Способ по п.7, в котором множество периодических обновлений принимаются по одному в периодические интервалы времени.

10. Способ по п.7, в котором множество периодических обновлений принимаются в пакетах в периодические интервалы времени.

11. Способ по п.7, в котором периодическое обновление содержит множество транзакций, каждая из которых имеет уникальный идентификатор транзакции.

12. Способ обновления удаленной базы данных через сеть, включающий

формирование множества периодических обновлений, основанных на добавочных изменениях в локальной базе данных, каждое из множества периодических обновлений содержит по меньшей мере одну транзакцию, и

формирование инициализирующего обновления, содержащего версию локальной базы данных в момент времени запуска, идентификатор обновления, соответствующий последнему периодическому обновлению, сформированному до момента времени запуска, и идентификатор транзакции, соответствующий последней транзакции, завершенной до момента времени запуска.

13. Способ по п.12, в котором множество периодических обновлений формируются с регулярными интервалами времени.

14. Способ по п.12, в котором множество периодических обновлений формируются с нерегулярными интервалами времени.

15. Способ по п.12, в котором момент времени запуска формирования инициализирующего обновления идентичен моменту времени запуска формирования периодического обновления.

16. Способ по п.12, в котором момент времени запуска формирования инициализирующего обновления более поздний, чем момент времени запуска формирования периодического обновления.

17. Система для обновления удаленной базы данных через сеть, содержащая

по меньшей мере один процессор, связанный с сетью, и

память, связанную с процессором, которая содержит локальную базу данных и команды, адаптированные для выполнения процессором, для реализации способа обновления удаленной базы данных через сеть, при этом способ включает

формирование множества периодических обновлений, основанных на добавочных изменениях в локальной базе данных, причем каждое из множества периодических обновлений содержит по меньшей мере одну транзакцию,

передачу множества периодических обновлений в удаленную базу данных через сеть и,

при формировании множества периодических обновлений,

формирование инициализирующего обновления, содержащего версию локальной базы данных в момент времени запуска,

определение последнего периодического обновления из множества периодических обновлений на основе момента времени запуска,

определение последней транзакции на основе момента времени запуска и

передачу через сеть в удаленную базу данных инициализирующего обновления, идентификатора последнего периодического обновления и идентификатора последней транзакции.

18. Система по п.17, в которой передача инициализирующего обновления включает

сопоставление последнему периодическому обновлению идентификатора последнего периодического обновления и

сопоставление последней транзакции идентификатора последней транзакции.

19. Система по п.17, в которой множество периодических обновлений формируется с регулярными или нерегулярными интервалами времени.

20. Система по п.17, в которой момент времени запуска формирования инициализирующего обновления идентичен моменту времени запуска формирования периодического обновления.

21. Система по п.17, в которой момент времени запуска формирования инициализирующего обновления более поздний, чем момент времени запуска формирования периодического обновления.

22. Система по п.17, в которой периодическое обновление содержит множество транзакций, каждая из которых имеет уникальный идентификатор транзакции.

23. Система для обновления через сеть удаленной базы данных, содержащая

по меньшей мере один процессор, связанный с сетью, и

память, связанную с процессором, которая содержит локальную базу данных и команды, адаптированные для выполнения процессором, для реализации способа обновления удаленной базы данных через сеть, при этом способ включает

получение через сеть множества периодических обновлений, основанных на добавочных изменениях в локальной базе данных, причем каждое из множества периодических обновлений содержит по меньшей мере одну транзакцию,

получение через сеть инициализирующего обновления, содержащего версию локальной базы данных в момент времени запуска,

считывание из инициализирующего обновления идентификатора последнего периодического обновления,

считывание из инициализирующего обновления идентификатора последней транзакции,

определение последнего периодического обновления из идентификатора последнего периодического обновления, причем последнее периодическое обновление основано на моменте времени запуска,

определение последней транзакции из идентификатора последней транзакции, причем последняя транзакция основана на моменте времени запуска,

применение к удаленной базе данных транзакций, сформированных после последней транзакции, и

применение к удаленной базе данных периодических обновлений, сформированных после последнего периодического обновления.

24. Система по п.23, дополнительно включающая

отбрасывание периодических обновлений, сформированных до момента времени запуска инициализирующего обновления.

25. Система по п.23, в которой множество периодических обновлений принимаются и по одному с регулярными или нерегулярными интервалами времени.

26. Система по п.2.3, в которой множество периодических обновлений принимаются в пакетах с регулярными или нерегулярными интервалами времени.

27. Система по п.23, в которой периодическое обновление содержит множество транзакций, каждая из которых имеет уникальный идентификатор транзакции, идентификаторы транзакций упорядочены случайным образом.

28. Носитель информации, считываемый компьютером, содержащий команды, адаптированные для выполнения процессором, для реализации способа обновления удаленной базы данных через сеть, причем способ включает

формирование множества периодических обновлений, основанных на добавочных изменениях в локальной базе данных, каждое из множества периодических обновлений содержит по меньшей мере одну транзакцию,

передачу множества периодических обновлений в удаленную базу данных через сеть и,

при формировании множества периодических обновлений,

формирование инициализирующего обновления, содержащего версию локальной базы данных в момент времени запуска,

определение последнего периодического обновления из множества периодических обновлений на основе момента времени запуска,

определение последней транзакции на основе момента времени запуска,

сопоставление последнему периодическому обновлению идентификатора последнего периодического обновления,

сопоставление последней транзакции идентификатора последней транзакции и

передачу через сеть в удаленную базу данных инициализирующего обновления, идентификатора последнего периодического обновления и идентификатора последней транзакции.

29. Носитель информации, считываемый компьютером, содержащий инструкции, адаптированные для выполнения процессором, для реализации способа обновления удаленной базы данных через сеть, причем способ включает

получение через сеть множества периодических обновлений, основанных на добавочных изменениях в локальной базе данных, каждое из множества периодических обновлений содержит по меньшей мере одну транзакцию,

получение через сеть инициализирующего обновления, содержащего версию локальной базы данных в момент времени запуска,

считывание из инициализирующего обновления идентификатора последнего периодического обновления,

считывание из инициализирующего обновления идентификатора последней транзакции,

определение последнего периодического обновления из идентификатора последнего периодического обновления, причем последнее периодическое обновление основано на моменте времени запуска,

определение последней транзакции из идентификатора последней транзакции, причем последняя транзакция основана на моменте времени запуска,

применение к удаленной базе данных транзакций, сформированных после последней транзакции, и

применение к удаленной базе данных периодических обновлений, сформированных после последнего периодического обновления.

30. Способ обновления удаленной базы данных через сеть, включающий

формирование множества периодических обновлений, основанных на добавочных изменениях в локальной базе данных, причем каждое из множества периодических обновлений содержит по меньшей мере одну транзакцию,

формирование инициализирующего обновления, содержащего версию локальной базы данных в момент времени запуска, и

увязку инициализирующего обновления с периодическими обновлениями, идентификатор обновления, соответствующий последнему периодическому обновлению, сформированному до момента времени запуска, и идентификатор транзакции, однозначно соответствующий последней транзакции, завершенной до момента времени запуска.

31. Способ по п.30, в котором периодические обновления содержат обновления в сервере доменных имен.

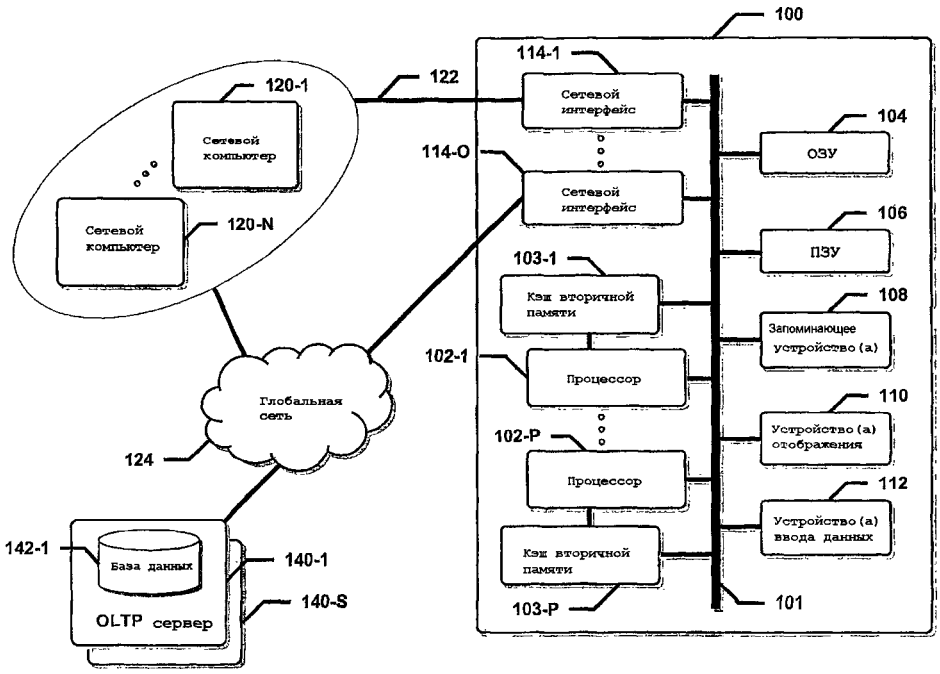
32. Способ по п.30, в котором идентификатор обновления соответствует последнему периодическому обновлению в сервере доменных имен.

33. Способ по п.30, в котором идентификатор транзакции соответствует последней транзакции, завершенной в сервере доменных имен.

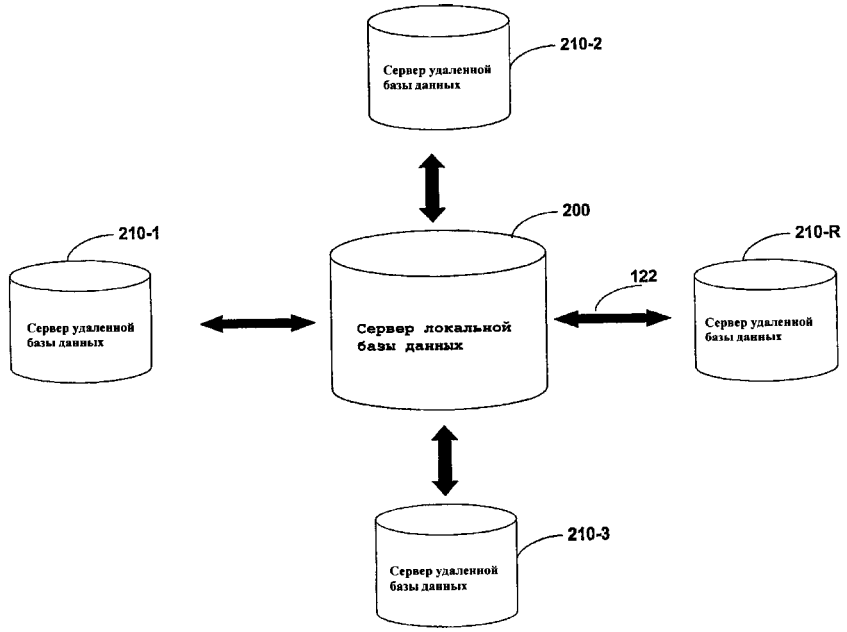
34. Формирователь обновлений, содержащий

средство формирования множества периодических обновлений, основанных на добавочных изменениях в локальной базе данных, причем каждое из множества периодических обновлений содержит, по меньшей мере, одну транзакцию, и

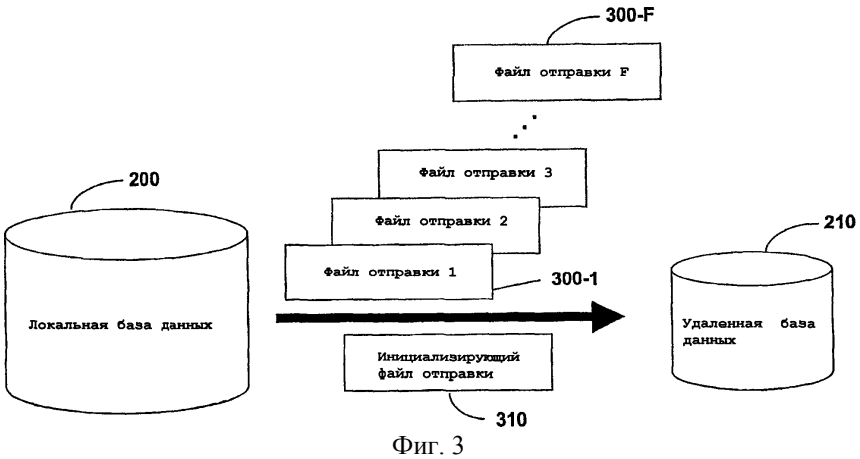
средство формирования инициализирующего обновления, содержащего версию локальной базы данных в момент времени запуска, идентификатор обновления, соответствующий последнему периодическому обновлению, сформированному до момента времени запуска, и идентификатор транзакции, соответствующий последней транзакции, завершенной до момента времени запуска.



Фиг. 1

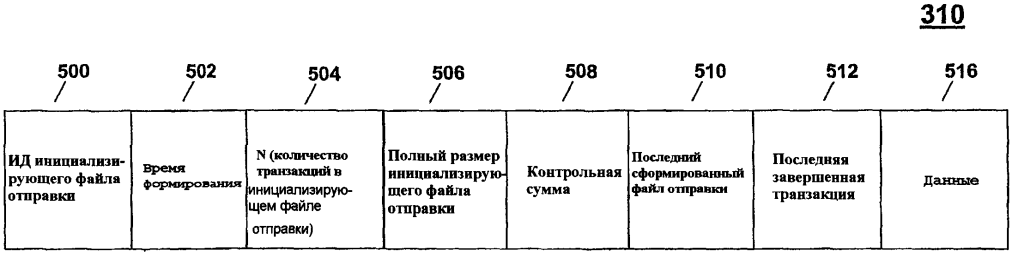


Фиг. 2

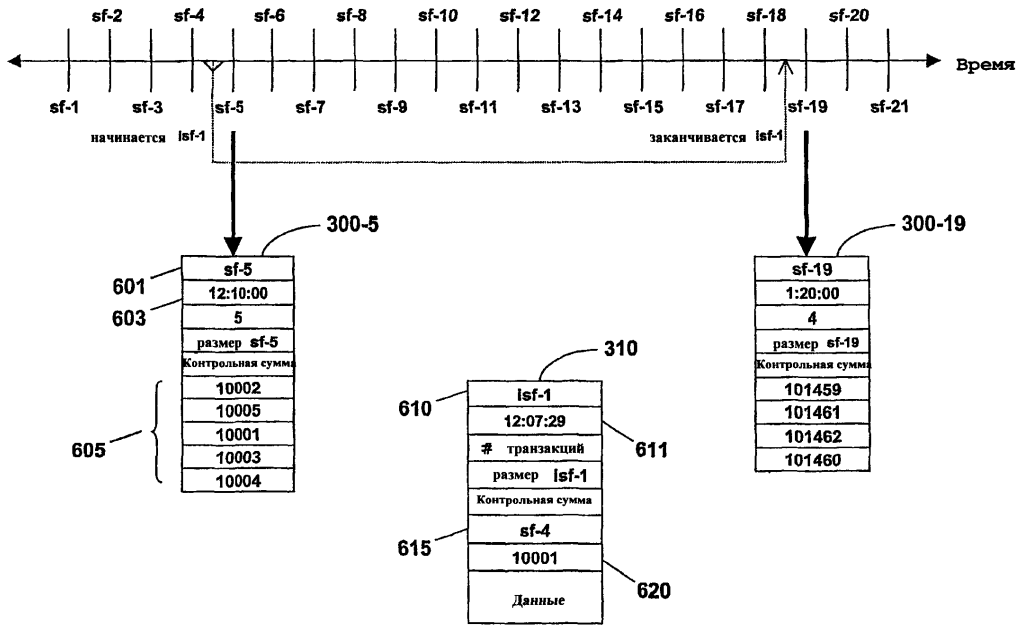


Фиг. 3

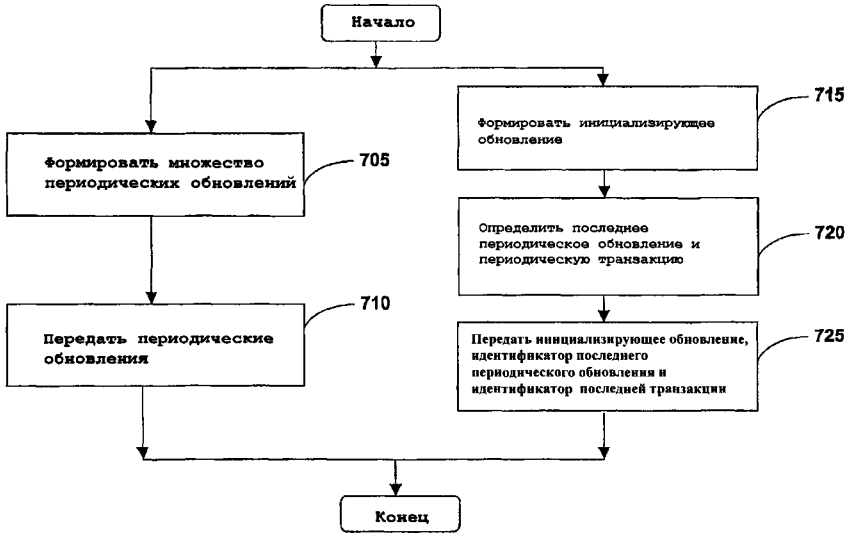
Фиг. 4



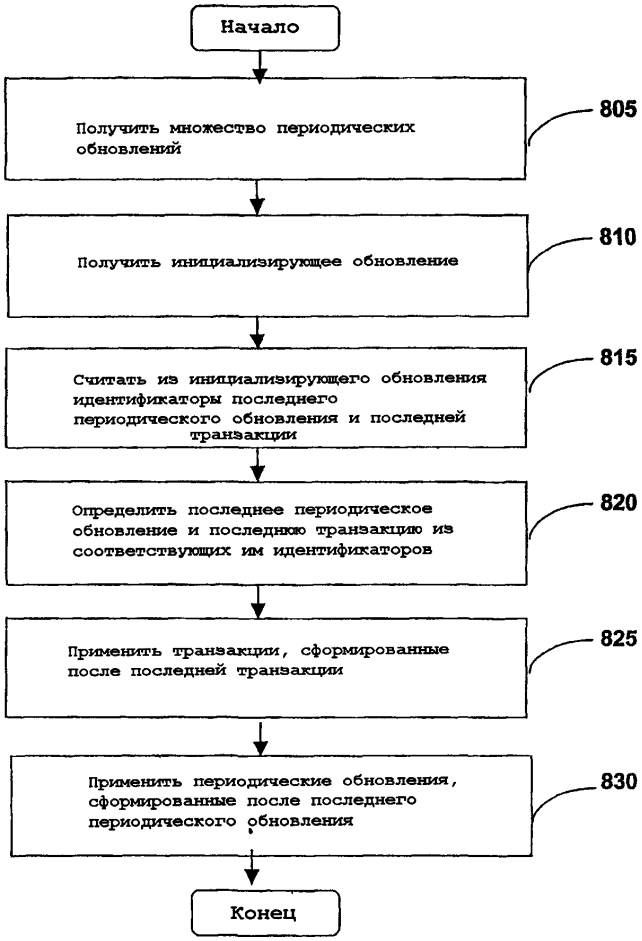
Фиг. 5



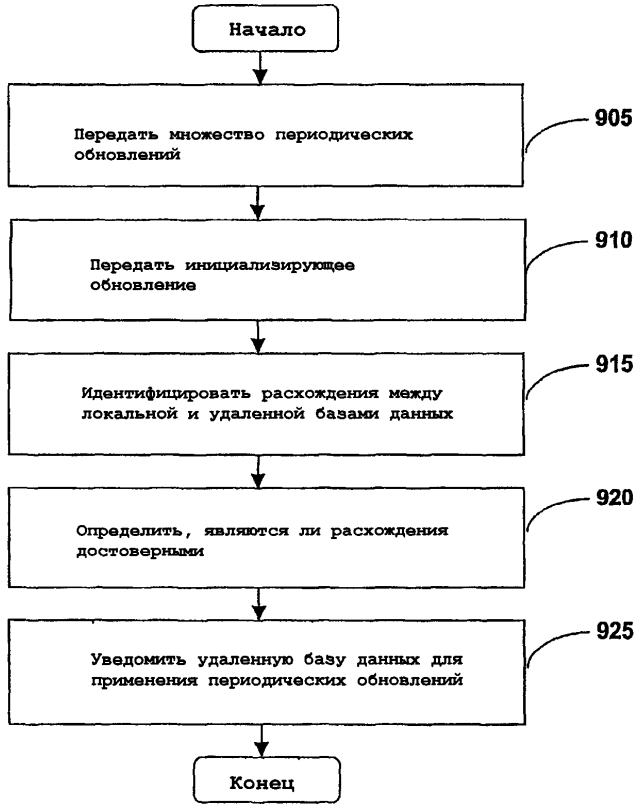
Фиг. 6



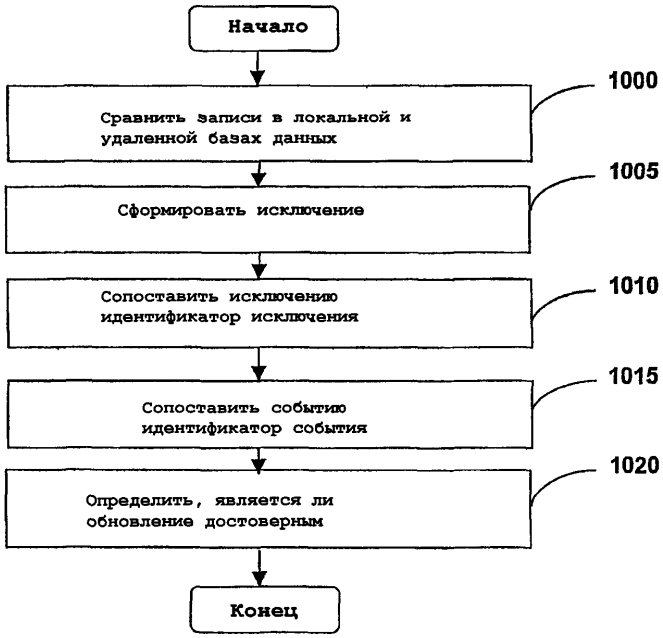
Фиг. 7


Начало
Получить множество периодических обновлений
805
Получить инициализирующее обновление
810
Считать из инициализирующего обновления идентификаторы последнего периодического обновления и последней транзакции
815
Определить последнее периодическое обновление и последнюю транзакцию из соответствующих им идентификаторов
820
Применить транзакции, сформированные после последней транзакции
825
Применить периодические обновления, сформированные после последнего периодического обновления
830
Конец

Фиг. 8

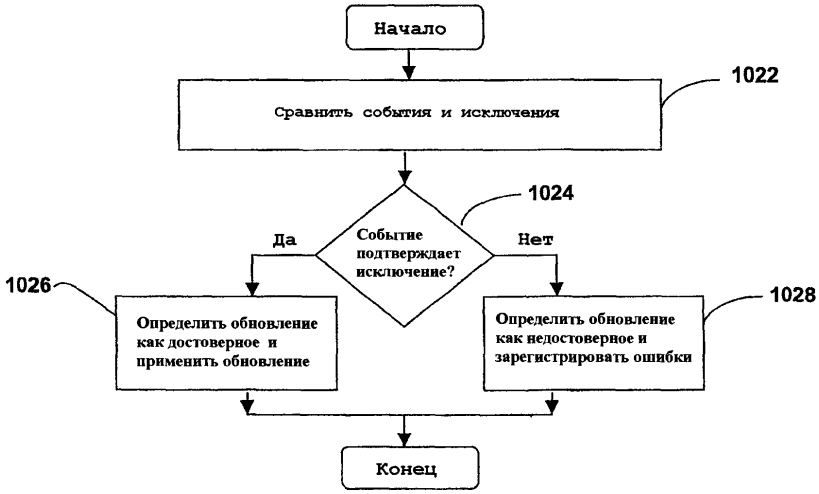

Начало
Передать множество периодических обновлений
905
Передать инициализирующее обновление
910
Идентифицировать расхождения между локальной и удаленной базами данных
915
Определить, являются ли расхождения достоверными
920
Уведомить удаленную базу данных для применения периодических обновлений
925
Конец

Фиг. 9

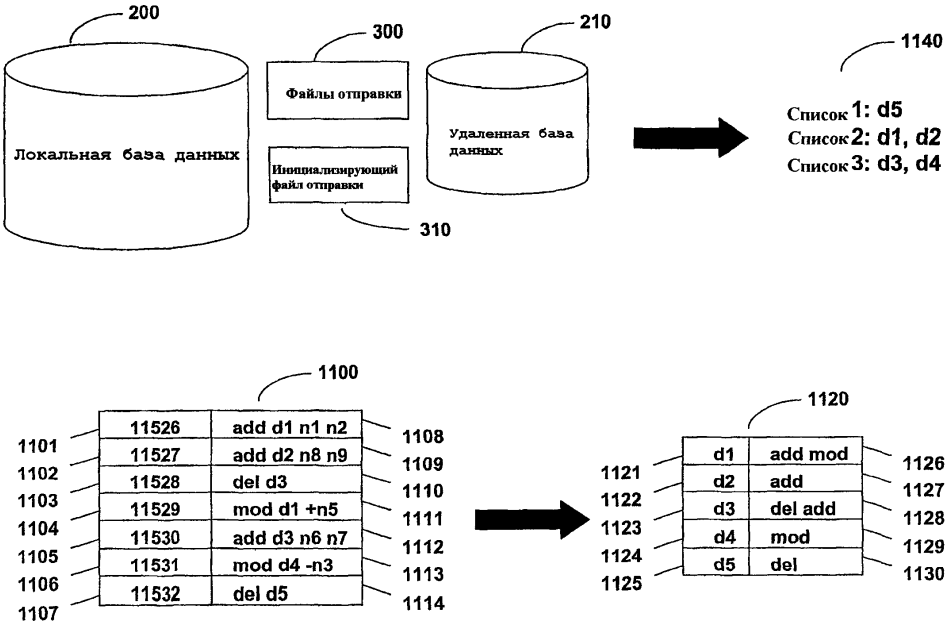


Фиг. 10A

1020



Фиг. 10B


200
Локальная база данных
300
Файлы отправки
Инициализирующий файл отправки
310
210
Удаленная база данных
1140
Список 1: d5
Список 2: d1, d2
Список 3: d3, d4
1100
1101 11526 add d1 n1 n2 1108
1102 11527 add d2 n8 n9 1109
1103 11528 del d3 1110
1104 11529 mod d1 +n5 1111
1105 11530 add d3 n6 n7 1112
1106 11531 mod d4 -n3 1113
1107 11532 del d5 1114
1120
1121 d1 add mod 1126
1122 d2 add 1127
1123 d3 del add 1128
1124 d4 mod 1129
1125 d5 del 1130

Фиг. 11