ISSN 2658 – 7505
Выпуск №4, 2020
Электронный научный журнал «Вестник молодёжной науки России»

УДК 004.658

ОБ ОДНОМ МЕТОДЕ ДВУСТОРОННЕЙ РЕПЛИКАЦИИ ДАННЫХ В РАСПРЕДЕЛЁННЫХ БАЗАХ ДАННЫХ

Панфилов А.Н., *panfiloff@rambler.ru,* ***Ситникова А.Ю.,*** *alinochka-sitnikova@mail.ru*
Южно-Российский государственный политехнический университет (НПИ) имени М.И. Платова
г. Новочеркасск

В данной статье описываются подходы и методы синхронизации данных между одноранговыми узлами в распределенных системах хранения данных и дается оценка преимуществ и недостатков каждого из них. Рассматривается возможное возникновение коллизий данных в процессе синхронизации и приводятся некоторые способы их разрешения. Формулируются рекомендации по выбору методов синхронизации в зависимости от предметной области

Ключевые слова: распределенные базы данных, репликация, синхронизация, коллизии данных, алгоритмы СУБД, непротиворечивость данных

ON A METHOD OF TWO-SIDE DATA REPLICATION IN DISTRIBUTED DATA BASES

Panfilov A.N., Sitnikova A.Y.
Platov South-Russian State Polytechnic University (NPI), Novocherkassk

This paper describes several approaches and methods of data synchronization between single rank nodes in distributed database systems and estimates pros and cons of each of them. Possible data collision occurrence is evaluated, and some of the resolving methods are described. Recommendations for picking the correct data collision resolving method depending on the application area are givenand development of new production on available floor spaces. The most important organizational economic targets of a diversification of management are presented by innovative activity of the industrial enterprise

Keywords: distributed databases, replication, synchronization, data collision, DBMS algorithms, data consistency

В различных системах информационной поддержки производственного процесса используемые данные, как правило, хранятся на удаленных узлах, что вынуждает отказываться от централизованных хранилищ данных в пользу распределенных, что предполагает использование локального хранилища узла системы. Однако, при использовании локального хранилища возникает необходимость проведения синхронизации данных с данными других узлов сети. В данном случае может возникнуть неоднородность синхронизируемых данных в разных узлах системы [1,2].

Под синхронизацией понимается процесс, при котором базы данных (БД) удаленных узлов приводятся в идентичное состояние. Передача сведений о своем текущем состоянии от одного узла к другому с последующим внесением изменений в него называется репликацией [3]. Это односторонний процесс, в котором есть распространитель информации и получатель. Односторонняя репликация решает вопросы резервирования данных и балансировки нагрузки, но для синхронизации данных в одноранговых средах необходимо использовать методы попарной двусторонней репликации между всеми узлами, имеющими разные состояния.

Синхронизацией в реальном времени, инициируемой по событию, считается та синхронизация данных, которая производится сразу после внесения изменений в БД. Этот метод сопровождается существенными затратами сетевого трафика. Так как в подавляющем большинстве распределенных систем такая

ISSN 2658 – 7505
Выпуск №4, 2020
Электронный научный журнал «Вестник молодёжной науки России»

оперативность обновления данных не требуется, данный метод применяется только в ограниченном виде. В большинстве случаев применяется отложенная синхронизация, при которой на каждом узле происходит накопление изменений, и их тиражирование начинается по некоторой служебной команде или событию.

Существуют несколько способов синхронизации данных между удаленными БД. Данные способы классифицируются по направленности. [4]. Для однонаправленной синхронизации проблема выбора не возникает, так как текущее состояние БД распространителя тиражируется в базу данных получателя в полном объеме с перезаписью данных последнего. В терминах систем управления базами данных (СУБД) такой подход называется «репликацией моментальных снимков» [5]. Для двунаправленной синхронизации различия между синхронизируемыми БД можно выявлять либо непосредственно перед началом их слияния, либо накапливать историю изменений по мере их появления.

Синхронизацией по текущему состоянию называется синхронизация, перед началом которой производится анализ расхождений в удаленных БД («репликация сведением» в терминах некоторых СУБД). В таком случае требуется установление активного сетевого соединения между двумя узлами. Если синхронизация проводится путем поочередного воспроизведения внесенных изменений в базы, хранимых в специальном журнале, то ее можно назвать синхронизацией по журналу транзакций (или «репликация транзакций»).

Если при проведении синхронизации изменения могут выявляться и производиться отдельными полями записей, то такую синхронизацию можно назвать синхронизацией по полям. В противном случае единицей проведения синхронизации является запись, и такой метод можно назвать синхронизацией по записям. В таблице 1 представлена классификация предлагаемых подходов к организации синхронизации в распределенных базах данных.

Таблица 1

Классификация подходов синхронизации данных к организации синхронизации в распределенных базах данных

<table>
<tr><td colspan="2">Область сравнения данных</td><td colspan="2">Способ проведения синхронизации</td></tr>
<tr><td>отдельные экземпляры данных</td><td>атрибуты экземпляров данных</td><td>в режиме реального времени</td><td>отложенная синхронизация</td></tr>
<tr><td colspan="2">репликация снимков</td><td>синхронизация по состояниям</td><td>синхронизация по журналу</td></tr>
<tr><td colspan="2">односторонняя</td><td colspan="2">двухсторонняя</td></tr>
<tr><td colspan="4">Направление синхронизации</td></tr>
</table>

При использовании метода двусторонней репликации данных невозможно полностью исключить возникновение коллизий. В данном случае коллизия – это некоторая неопределенность или ошибка, вызванная неспособностью метода разрешить, какие из синхронизируемых данных являются приоритетными. Как правило, для разрешения возникших коллизий необходимо привлечение администратора системы, но существуют приемы, позволяющие предупредить возникновение конфликтных ситуаций [6]. Выделим виды коллизий синхронизации в распределенных системах:

– коллизии первичных ключей;

ISSN 2658 – 7505
Выпуск №4, 2020
Электронный научный журнал «Вестник молодёжной науки России»

– коллизии несвязанных таблиц;
– коллизии связанных таблиц.

В таблице 2 представлены причины возникновения и способы обработки данных коллизий. Рассмотрим каждый тип коллизии отдельно

Таблица 2

Коллизии синхронизации в распределенных системах

Тип коллизии	Причина возникновения	Метод двусторонней синхронизации	Способы обработки
Коллизии первичных ключей	Одинаковые первичные ключи для новых записей	Любой	Составные первичные ключи
			Строковые первичные ключи
			Организация допустимых диапазонов
Коллизии в самостоятельных таблицах	Дублирование данных	Любой	Ограничения уникальности
	Коллизии уникальности		Интеллектуальные алгоритмы
	Восстановление удаленных данных	Синхронизация по состоянию	Частичная централизация
			Дополнительная служебная информация
			Оперативная синхронизация
	Модификация одинаковых полей	Синхронизация по строкам	Хранение временных меток
	Модификация разных полей (в пределах записи)		Ведение журнала транзакций
Коллизии в связанных таблицах	Репликация связанных записей	Любой	Отложенные проверки целостности
			Интеллектуальные алгоритмы
	Создание дочерних записей для удаленных родительских		В зависимости от правил бизнес-логики

Коллизии первичных ключей – в большинстве случаев в таблицах баз данных в качестве первичных ключей используются суррогатные целочисленные поля, и при независимом создании новых записей в распределенных БД при синхронизации может возникнуть ошибка совпадения первичных ключей для разных записей.

Наиболее часто встречающейся причиной возникновения коллизий при синхронизации является независимая модификация одинаковых данных. Следует рассмотреть два случая: модификация данных в самостоятельных таблицах, не содержащих ссылок на внешние ключи, и модификация данных в связанных таблицах. В первом случае возможны следующие ситуации:

– независимое добавление одинаковых данных;

– удаление записей;
– модификация отдельных полей данных.

Независимое добавление одинаковых данных в отдельные БД может возникнуть при отсутствии ограничений уникальности на прикладных полях записей. Для разрешения таких коллизий необходима разработка интеллектуальных алгоритмов, способных к анализу синхронизируемых данных;

Если в некоторой базе данных системы будет удалена отдельная запись, присутствующая в других БД системы, то она может быть восстановлена при следующей синхронизации, что по сути является ошибкой. Решением таких коллизий может выступать частичная централизация (замена полной синхронизации односторонней репликацией для отдельных таблиц), хранение дополнительной служебной информации (например, журнала транзакций) или оперативная синхронизации такого изменения данных;

Коллизии модификации отдельных полей данных могут возникнуть при использовании метода синхронизации «по строкам», когда строка с более поздней временной отметкой обновления полностью заменяет строку, в которой могли быть модифицированы другие поля. С другой стороны, применение синхронизации по полям требует гораздо больше служебной информации и более сложных алгоритмов, а также может быть ограничено функциональными особенностями используемой СУБД.

В свою очередь, модификация данных в связанных таблицах может привести к возникновению коллизий в следующих ситуациях: при одновременном создании в одной БД родительской и дочерней записи, при создании новой дочерней записи для удаленной родительской, перемещение дочерней записи к новой родительской, которая была удалена в удаленной базе. Рассмотрим каждый вариант.

В одной базе одновременно создаются родительская и дочерняя запись - здесь становится важен порядок создания записей в удаленной БД при синхронизации – сначала должны создаваться родительские записи, затем дочерние. Для предотвращения таких коллизий необходимо специальным образом обрабатывать связанные данные в синхронизирующем алгоритме, либо, при наличии возможности в СУБД, использовать отложенную проверку ограничений целостности в рамках одной транзакции. Сюда же можно отнести удаление связанных данных – в первую очередь должны удаляться дочерние записи, и лишь затем родительские;

Создание новой дочерней записи для удаленной родительской – в таком случае при синхронизации будет произведена попытка удалить родительскую запись в базе, где была создана дочерняя, и попытка создать новую запись в базе, где отсутствует требуемый родитель. Решение данного вида коллизий будет зависеть от правил бизнес-логики распределенной системы, и с большой степенью вероятности будет необходимо вмешательство администратора. Частным случаем данного вида коллизии является перемещение дочерней записи к новой родительской, которая была удалена в базе другого узла сети. Решениями являются отмена изменения ссылки на родителя либо запрет удаления родительской записи.

Синхронизация по журналу транзакций является наиболее часто применяемым методом, в котором все действия по модификации данных хранятся

ISSN 2658 – 7505
Выпуск №4, 2020
Электронный научный журнал «Вестник молодёжной науки России»

независимо от данных, в отдельном файле или самостоятельной служебной таблице. Его заполнение происходит триггерами, которые записывают в журнал информацию, достаточную для повторного выполнения произведенного действия. В случае удаления записи достаточно сохранить идентификатор таблицы и значение первичного ключа удаленной записи, а для изменения или создания записи нужно также сохранять значения полей. Обычно также сохраняется отметка времени, когда было произведено изменение.

В качестве примера синхронизации по журналу транзакций рассмотрим фрагмент распределенной базы данных и таблицу «Основные средства». Фрагмент таблицы до проведения транзакций показан на рисунке 1

ID	Название	Код	Дата выпуска	Срок эксплуатации	Признак удаления
101	Компьютер	0007820706	12.03.2014	6	*N*
207	Фрезерный станок	0006821436	23.10.2008	11	*N*
3308	Шлифовальный станок	0007235706	09.12.2015	4	*N*
4521	Протяжный станок	0007820546	04.06.2007	12	*N*
5546	Ручной считыватель штрих-кодов	-	25.05.2017	2	*N*

Рис. 1 - Значение строк данных таблицы «Основные средства» до проведения транзакций

Таблица «Основные средства» предназначена для хранения информации о средствах труда, участвующих в производственном процессе. Пусть в данной таблице произведены модификации данных с различных узлов сети. Журнал транзакций представлен на рис. 1.

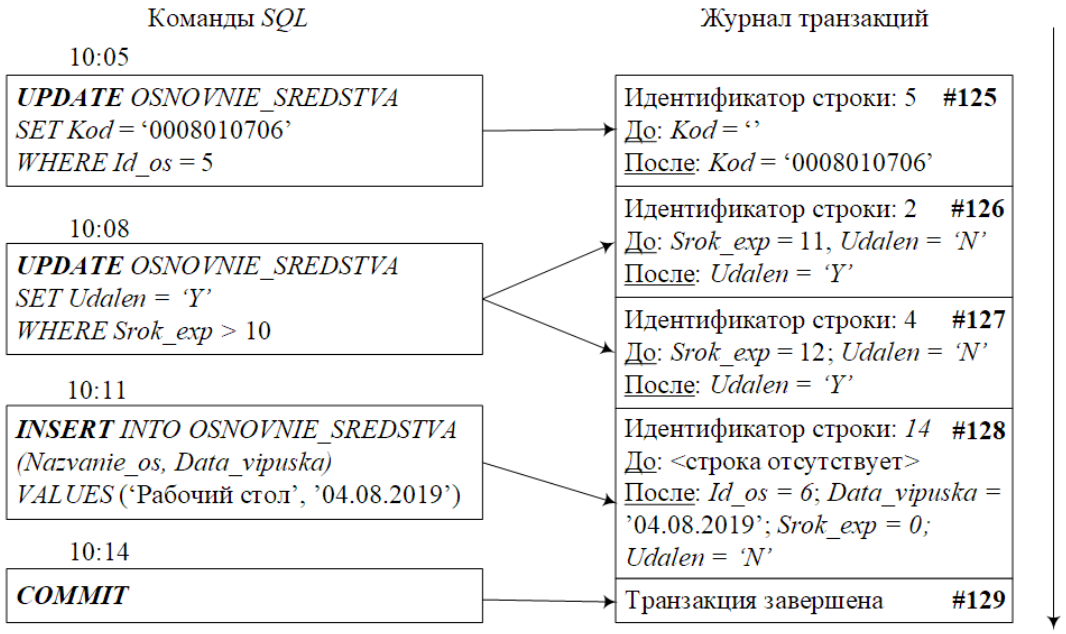

Рис. 2 – Схема построения журнала транзакций

На рисунке 3 показана часть данных из таблицы «Основные средства» после проведения транзакций. В ходе проведения транзакций были изменены строки 207, 4521, 5546 и добавлена строка 8067.

ID	Название	Код	Дата выпуска	Срок эксплуатации	Признак удаления
101	Компьютер	0007820706	12.03.2014	6	*N*
207	Фрезерный станок	0006821436	23.10.2008	11	*Y*
3308	Шлифовальный станок	0007235706	09.12.2015	4	*N*
4521	Протяжный станок	0007820546	04.06.2007	12	*Y*
5546	Ручной считыватель штрих-кодов	0008010706	25.05.2017	2	*N*
8067	Рабочий стол	-	04.08.2019	0	*N*

Рис. 3 – Значение строк данных «Основные средства» после проведения транзакций

Эффективным решением является хранение текста выполненных *SQL*-команд транзакций, позволяющим наиболее точно воспроизвести модификацию данных [8]. Данный метод обладает следующими достоинствами:

– синхронизацию можно производить без установления активного соединения между синхронизируемыми БД;

– возможность восстановления состояния БД на любой момент;

– высокая скорость синхронизации, пропорциональная числу изменений.

Однако следует отметить и недостатки:

– большие объемы хранимых данных, пропорциональные числу изменений;

– рост сложности правил применения журналов при росте числа распределенных узлов;

– коллизии практически неизбежны.

Синхронизация по состоянию менее распространена среди встроенных средств администрирования современных промышленных СУБД, тем не менее метод находит применение в определенных задачах [9]. Синхронизация выполняется с установлением активного соединения, поэтому не может быть выполнена в офлайн режиме. После установления соединения между обоими БД (если баз больше двух, то синхронизация выполняется попарно, например, «цепочкой» или «звездой») проводится анализ синхронизируемых таблиц на предмет их различий. Уже здесь проявляется первый недостаток метода – скорость синхронизации пропорциональная полному количеству записей в БД.

После обнаружения некоторого различия, алгоритм принимает решение о том, какая из БД содержит более актуальные данные по текущей обрабатываемой записи и какую операцию *DML* необходимо выполнить. Данный алгоритм может извлекать дополнительные сведения через вспомогательные запросы в обе БД для максимально эффективного принятия решения – например, поиск в других таблицах записей, ссылающихся на текущую. Задав определенные правила для алгоритма, можно автоматически разрешать большую часть коллизий [10]. Таким образом, метод синхронизации по состоянию позволяет провести анализ различий и максимально безопасное их устранение, вместо слепого повторения действий из другой БД. Для проведения синхронизации по данному методу обычно для

каждой записи БД вводятся одно или несколько дополнительных полей (обычно это отметка времени), по которым алгоритм сможет определить, какая из двух записей содержит более актуальные данные. Размер дополнительных данных также пропорционален общему количеству записей, но он достаточно мал и не увеличивается при многократных изменениях данных, как в методе синхронизации по журналу, когда записи чаще всего целиком дублируют строки таблиц и накапливаются при последовательных изменениях.

Преимущество метода синхронизации по состояниям – возможность исключения большинства коллизий. Недостатки – низкая скорость синхронизации, пропорциональная количеству всех записей, отсутствие возможности офлайн синхронизации, использование дополнительных служебных полей.

Таким образом, метод синхронизации состояний наиболее эффективен для сильно связанных распределенных систем, в которых общее количество записей относительно мало, но каждый узел генерирует большое число событий модификации данных. Это системы поддержки производственного процесса, складского учета, в том числе и на базе технологии радиочастотной идентификации, а также распределенные системы аналитической обработки данных. Для повышения эффективности работы метода может быть применен его частный случай – синхронизация наборов требуемых таблиц, в которой имеют место смешанные потоки данных. В таком случае синхронизации подвержены не базы данных целиком, а лишь таблицы тех данных, в которых заинтересованы другие узлы системы, с использованием методов односторонней репликации, где это возможно.

Список цитируемой литературы

1. Скоба А.Н., Состина Е.В. Математическая модель оптимального размещения распределённой базы данных по узлам ЛВС на базе файл-серверной архитектуры // Инженерный вестник Дона, 2015, №2 URL: ivdon.ru/magazine/archive/n1y2009/250/.

2. G.L. Sanders, Seungkyoon Shin. Denormalization effects on performance of RDBMS // Proceedings of the 34th Annual Hawaii International Conference on System Sciences, 2003. URL: ieeexplore.ieee.org/abstract/document/926306/.

3. Шаталова Ю.Г., Жиглов Я.В. Разработка системы репликации для распределенной базы данных предприятия // Символ науки. 2017. №03-2. С. 137-141.

4. M. Wiesmann, F. Pedone, A. Schiper, B. Kemme, G. Alonso. Database replication techniques: a three parameter classification // Proceedings 19th IEEE Symposium on Reliable Distributed Systems, 2000. URL: ieeexplore.ieee.org/abstract/document/885408/.

5. А. Н. Земцов, Н. В. Болгов, С. Н. Божко. Многокритериальный выбор оптимальной системы управления базы данных с помощью метода анализа иерархий // Инженерный вестник Дона, 2014, №2. URL: ivdon.ru/ru/magazine/archive/n2y2014/2360.

6. Ильин А.В., Пищик Б.Н. Методы репликации в распределенных системах // Вести Новосибирского гос. ун-та. Серия: Информационные технологии. 2016. Т. 14, №2. С. 52-58.

7. Коновалов М.В. Обзор и сравнительный анализ промышленных хранилищ данных и баз данных // Молодой ученый. 2018. №24. С. 24-28.

8. Ивутин А.Н., Терехин И.С. Повышение надежности и производительности баз данных при помощи репликации // Известия Тульского гос. ун-та. Технические науки. 2013. №9-2. С. 118-123.

9. Khuzaima Daudjee, Kenneth Salem. Lazy Database Replication with Snapshot Isolation // VLDB Conference, 2006. URL: vldb.org/conf/2006/p715-daudjee.pdf.

10.Гришмановский П.В., Базилевский Е.В. Анализ технологий репликации данных и методы повышения эффективности разрешения конфликтов репликации // Вестник Волжского ун-та им В.Н. Татищева. 2012. №2 (19). С. 97-106.