

image not found or type unknown



Преимущества:

- хорошая адаптация к предъявляемым требованиям;
- продукт не тиражируем (индивидуален);
- возможность быстрого изменения функциональности

Недостатки:

- разработка не закончится никогда;
- плохая адаптивность;
- необходимо создавать команду, либо отрывать от работы текущих сотрудников ИТ отдела;
- часто учитываются пожелания руководства в ущерб качеству разработки;
- проект может захлебнуться:
 - из-за нехватки квалификации внутренних специалистов;
 - из-за ухода ведущих специалистов;
 - из-за нехватки внутренних ресурсов;
- часто плохая документированность системы;

Преимущества и недостатки заказных, уникальных и тиражируемых систем.

Преимущества. Использование технологии тиражирования имеет следующие преимущества:

- сокращение сетевого трафика при выполнении запросов;
- повышение доступности данных. Доступ к локальной копии БД обеспечивается даже в случае, если доступ к центральному серверу по той или иной причине

невозможен;

- повышение производительности (за счет приближения данных к месту их использования, а также за счет специфики технологии выполнения запроса: не нужно ждать, чтобы все части распределенной БД были одновременно доступны);
- повышение автономности рабочих мест пользователей;
- повышение надежности системы (наличие множества копий повышает вероятность восстановления системы в критических ситуациях);
- уменьшение трафика (при определенных условиях);
- уменьшение конкуренции за ресурсы со стороны пользователей.

Недостатки. Дублирование данных при использовании технологии тиражирования влечет за собой следующие очевидные недостатки:

- дополнительный расход памяти;
- сложность обеспечения целостности данных; возможность возникновения конфликтов при корректировке;
- наличие временного лага между фиксацией события в БД и доступностью этой информации для всех пользователей сети;
- повышенные требования к рабочим станциям;
- системы с поддержкой тиражирования данных требуют тщательного продумывания схемы тиражирования. Само по себе проектирование современной БД представляет собой непростую задачу, а включение в нее схемы тиражирования данных требует дополнительного времени на проектирование и организацию.

Виды тиражирования.

Существуют разные виды тиражирования. Они различаются степенью транзакционной целостности и степенью автономности узлов (что, как правило, находится в обратной зависимости друг от друга). Требования к механизму тиражирования зависят от задач, которые решает вся вычислительная система в целом.

Основные проблемы, возникающие в системах с тиражированием, связаны с поддержкой согласованности состояния всех копий. Существует множество схем проведения обновления копий в РБД.

По моменту внесения изменений в реплики различают синхронное и асинхронное тиражирование.

Асинхронное тиражирование (называемое также тиражированием с промежуточным хранением) реализует схему, при которой обновление всех копий баз данных может проводиться неодновременно. При таком методе поддержки логической целостности распределенной базы данных имеет место некоторая рассинхронизация состояния локальных баз данных во времени, т.е. изменение состояния одной локальной базы данных отстает от изменения другой локальной базы данных по времени.

Асинхронность обмена данными в целом ряде случаев вполне допустима, и при этом удается обойти, в частности, проблему ограниченной пропускной способности и недостаточной надежности сети. Если один сервер системы, требующий обновления тиражируемых данных, выходит из строя, то система продолжает работать с остальными, при этом обновление данных на сбойном сервере после восстановления его работоспособности произойдет автоматически, т.е. ошибка на одном узле глобальной сети не повлияет на работу остальных узлов.

Механизм асинхронного тиражирования данных особенно актуален при работе в глобальной сети с недостаточно надежными и быстродействующими каналами связи. Например, при использовании телефонных линий связи механизм двухфазной фиксации транзакции практически нереализуем, поскольку все оборудование одновременно в работоспособном состоянии почти никогда не находится.

Асинхронное тиражирование может быть периодическим или аperiodическим. *Периодическое тиражирование* выполняется через заданные интервалы времени. Время выполнения очередных циклов тиражирования выбирается исходя из частоты возникновения изменений и их интенсивности, ограничения на допустимое время рас-синхронизации состояний источника и приемника, объема данных, передаваемых в единичном цикле тиражирования, и максимально допустимой загрузки коммуникационных ресурсов и некоторых других факторов.

Момент выполнения *аperiodического тиражирования* определяется обычно каким-либо событием (такой механизм называется также тиражированием по событиям) и может быть реализован, например, через механизм процедур или триггеров.

В случае *синхронного тиражирования* предполагается завершение транзакции только после успешной модификации всех копий. Синхронное тиражирование

использует механизм двухфазной фиксации (2PC) - Two-Phase Commit: основная система связывается с подчиненными копиями баз данных и одновременно вносит в них изменения, блокируя соответствующие записи. Если хотя бы одна из таких копий недоступна, изменения не выполняются. Поскольку механизм двухфазной фиксации предъявляет высокие требования к системе и снижает степень автономности узлов, то он в технологии тиражирования используется не часто.

Чтобы как-то компенсировать недостатки синхронного тиражирования, используют *тиражирование во времени, близком к реальному*, которое заключается в том, что сразу после завершения транзакции на основном узле изменения автоматически передаются тиражируемым копиям. Но при данном виде тиражирования в случае недоступности одного из узлов изменения хранятся до восстановления связи с получателем. Такая схема создает меньшую загрузку сети и подходит для приложений, не требующих строгой синхронности копий данных.

Наиболее жестким критерием согласованности является критерий *полной эквивалентности копий*, который требует, чтобы по завершении транзакции все копии логического элемента данных были идентичны. Такая технология синхронизации не позволяет достичь одной из основных целей применения технологии тиражирования - повышения производительности за счет обеспечения большей автономности отдельных частей системы - и поэтому используется в системах с тиражированием редко.

Типичный протокол тиражирования, обеспечивающий сериализуемость по критерию полной эквивалентности копий, известен под названием Read-Once/Write-All (ROWA) - одно чтение, запись во все копии. Операция чтения в этом протоколе сводится к чтению одной какой-либо копии (вопрос о том, из какой именно копии будет выполняться чтение, может решаться из соображений эффективности). Каждая операция записи в логический элемент данных отображается на множество операций записи в физические копии этих элементов.

Протокол ROWA прост, но он требует доступности всех копий элемента данных, для того чтобы транзакция была завершена. Сбой на одном из узлов приведет к дублированию транзакции, что снижает доступность данных.

Было предложено несколько альтернативных алгоритмов, смягчающих требования ROWA относительно одновременности внесения изменения во все копии.

Реплики в РБД с тиражированием могут быть *равноправными* и *неравноправными*. Чаще всего они являются неравноправными. В этом случае одна из реплик

считается основной. Даже при наличии неравноправных реплик возможны ситуации, при которых корректировать разрешается любую реплику, но часто изменения позволено вносить только в основную реплику, а другие реплики доступны пользователям только по чтению. Такая схема тиражирования называется *тиражированием из основного узла* (primary site). При этом данные асинхронно копируются из основной в иные реплики.

Поскольку при такой схеме тиражирования предъявляются повышенные требования к сохранности основной реплики и надежности функционирования основного узла, то иногда вышеописанную модель дополняют *горячим резервированием* (failover) основного узла. В этом случае основной узел тиражирует изменения как на подчиненные узлы, так и на узел с резервной копией. Если основной узел выходит из строя, то владельцем данных становится резервный, который с этого момента выполняет все его функции.

Одной из технологий, которые могут использоваться, если изменения вносятся в локальные реплики, является *тиражирование слиянием* (merge replication). Суть метода заключается в том, что операции выполняются на удаленном компьютере, который может быть даже полностью отключен от компьютерной сети. Автономная СУБД записывает все операции с данными и их очередность. Затем в определенный момент автономный компьютер связывается с издателем (сервером, на котором корректируется БД, подлежащая дальнейшему тиражированию) и согласовывает с ним свои данные, пересылая издателю последовательность операций, проведенных в удаленной базе данных. При возникновении конфликтов они разрешаются с помощью различных алгоритмов.

В некоторых системах функции формирования «эталонной» базы и рассылки реплик конечным пользователям (подписчикам) распределены между разными узлами системы. В этом случае после слияния издатель передает изменения на сервер-дистрибьютер для дальнейшего распространения по подписчикам.

Тиражирование слиянием обеспечивает максимальную автономность удаленной базы данных.

При использовании этого, метода могут возникнуть коллизии, которые необходимо себе хорошо представлять и предотвращать их разнообразными средствами, как техническими, так и организационными.

Кроме того, существуют так называемые *поточные модели тиражирования* (workflow). В этой модели владелец данных меняется динамически: данные могут

обновляться на различных узлах, но в любой момент времени такими полномочиями обладает только один из них. Узлы последовательно обновляют принадлежащие им данные и тиражируют изменения на всю систему.

Некоторые системы позволяют любым узлам с тиражируемыми копиями одновременно обновлять их. Такая модель называется моделью с *произвольным обновлением данных*. В этом случае возможно возникновение конфликтов между отдельными репликами, и необходим развитый механизм выявления и разрешения этих конфликтов, предотвращающий потерю целостности данных. При этом могут использоваться разнообразные алгоритмы разрешения возникающих коллизий. В некоторых системах реализован какой-либо один из возможных алгоритмов, в других - имеется возможность выбора из нескольких вариантов.

Еще одним критерием классификации механизмов тиражирования является признак того, кто является инициатором обновления реплик. Внесение изменений в реплики может инициироваться рабочими станциями. Такой процесс обновления называется *обновлением по запросу*. Обновление может выполняться как *по определенному графику*, так и *вручную*.

Другой подход заключается в том, что инициатором проведения изменений является сервер (*принудительная рассылка тиража*). Такая рассылка может осуществляться либо в момент появления изменений (*немедленное тиражирование*), либо *по графику* (через заданный интервал, в указанное время).

Немедленное тиражирование (называемое также *тиражированием по событиям*) означает, что сразу после завершения транзакции на основном узле изменения автоматически передаются тиражируемым копиям.

Метод, при котором считывание данных проводится удаленными узлами из основных, называется *вытягиванием* изменений, а метод, заключающийся в выгрузке данных на узлы с копиями - *выталкиванием* изменений.

Обновление содержания реплик может быть обеспечено следующими способами:

- копированием моментального снимка базы данных;
- копированием и выполнением очереди подтвержденных издателем транзакций;
- копированием изменений из журнала БД.

Моментальный снимок базы данных (snapshot) отражает состояние базы данных в целом или ее фрагмента на момент получения снимка. При копировании моментального снимка базы данных часто передаются не только собственно данные, но и дополнительная служебная информация, например идентификаторы столбцов и строк.

В процессе *тиражирования транзакций* (transactional replication) от издателя к подписчикам передаются не данные, а операции над ними. Само обновление происходит на рабочей станции. Передачу транзакций можно использовать только в том случае, когда в автономной базе данных уже хранится копия основной БД. При использовании этой схемы периодически необходимо выполнять и полную синхронизацию данных, которая выполняется по методу тиражирования моментального снимка данных. Тиражирование транзакций лучше всего использовать, когда объем базы данных велик, а операций над ними выполняется немного.

В БД вообще, а в распределенных - особенно одной из важнейших задач является обеспечение целостности баз данных, в том числе восстановление баз после их разрушения. Для этих целей используются *журналы изменений баз данных* - протокол, в котором регистрируются в хронологическом порядке исходные и обновленные состояния всех записей базы данных, модифицированных в процессе исполнения транзакций. Ведение журнала обеспечивается многими СУБД и применяется при процедурах восстановления.

В случае *обновления на подписчике* (immediate updating subscribers) тиражирование иницируется издателем. Как только издатель подтверждает транзакцию, он сообщает дистрибьютеру о том, что данные изменены. Дистрибьютер забирает подтвержденную транзакцию и рассылает ее подписчикам. Если во время завершения транзакции связь между дистрибьютером и издателем была прервана, то транзакция записывается в очередь и будет передана дистрибьютеру, как только связь восстановится. Дальнейшее распространение данных выполняет дистрибьютер либо принудительно, либо по запросу. Транзакции проводятся только через издателя.

Список использованной литературы

1 Автоматизированные информационные технологии в экономике: учебник / под ред. Г. А. Титоренко. — М.: Компьютер, ЮНИТИ, 2011.

2 Дзюбенко, А. Л. Корпоративные информационные системы. Особенности внедрения. Часть 1 / А. Л. Дзюбенко // Информационные системы в экономике — (http://www.e-college.ru/xbooks/xbook_018/book/index/index.html?go=part-014*page.htm).

3 Корпоративные информационные системы — (http://iablov.narod.ru/igupit/kislec.htm#_Тoc_151_864_579).