ВикипедиЯ

Двухфазная блокировка

Материал из Википедии — свободной энциклопедии

В базах данных и обработке транзакций **двухфазная блокировка (2PL)** — это метод управления параллелизмом, который гарантирует сериализуемость[1][2]. Это также имя результирующего набора графиков транзакций базы данных (истории). Протокол использует блокировки, применяемые транзакцией к данным, которые могут блокировать (интерпретировать как сигналы для остановки) другие транзакции от доступа к тем же данным в течение жизни транзакции.

По протоколу 2PL блокировки применяются и удаляются в два этапа:

- Фаза расширения: фиксируются блокировки и не блокируются блокировки.
- Фаза сжимания: замки освобождаются, и блокировки не приобретаются.

В базовом протоколе используются два типа замков: Shared и Exclusive locks. Уточнения базового протокола могут использовать больше типов блокировок. Используя блокировки, блокирующие процессы, 2PL могут подвергаться взаимоблокировкам, которые являются результатом взаимной блокировки двух или более транзакций.

Содержание

Блокировки доступа к данным

Блокировка — системный объект, связанный с общим ресурсом, например элемент данных элементарного типа, строка в базе данных или страница памяти. В базе данных перед доступом к объекту может потребоваться блокировка объекта базы данных (блокировка доступа к данным) путем транзакции. Правильное использование блокировок предотвращает нежелательные, неправильные или непоследовательные операции с общими ресурсами другими параллельными транзакциями. Когда к объекту базы данных с существующей блокировкой, полученной одной транзакцией, необходимо получить доступ другой транзакцией, существующая блокировка объекта и тип предполагаемого доступа проверяются системой. Если существующий тип блокировки не разрешает этот конкретный тип одновременного доступа, транзакция, пытающаяся получить доступ, блокируется (в соответствии с предопределенным соглашением / схемой). На практике блокировка объекта не блокирует транзакцию непосредственно над объектом, а блокирует транзакцию от приобретения другой блокировки на одном и том же объекте, которая должна быть

сохранена / принадлежит транзакции перед выполнением этой операции. Таким образом, с блокирующим механизмом необходимая блокировка операций контролируется надлежащей схемой блокировки блокировки, которая указывает, какие блоки блокировки блокируют тип блокировки.

Используются два основных типа замков:

- **Запись-блокировка (эксклюзивная блокировка)** связана с объектом базы данных транзакцией (терминология: «транзакция блокирует объект» или «получает блокировку для нее») перед записью (вставка / изменение / удаление) этого объекта.
- **Считывание-блокировка (разделяемая блокировка)** связана с объектом базы данных транзакцией перед чтением (извлечением состояния) этого объекта.

Общие взаимодействия между этими типами блокировки определяются путем блокирования поведения следующим образом:

- Существующая блокировка записи в объекте базы данных блокирует предполагаемую запись на том же объекте (уже запрошенном / выпущенном) другой транзакцией, блокируя блокировку записи от другой транзакции. Вторая блокировка записи будет получена, и запрошенная запись объекта будет выполнена (материализоваться) после освобождения существующей блокировки записи.
- Блокировка записи блокирует предполагаемую (уже запрошенную / выпущенную), прочитанную другой транзакцией, блокируя соответствующую блокировку чтения.
- Блокировка чтения блокирует запланированную запись другой транзакцией, блокируя соответствующую блокировку записи.
- Блокировка чтения не блокирует предполагаемое чтение другой транзакцией. Соответствующая блокировка чтения для предполагаемого считывания приобретается (совместно с предыдущим чтением) сразу после запроса на чтение, а затем происходит само предназначение считывания.

Существует несколько вариантов и усовершенствований этих основных типов блокировок с соответствующими вариациями поведения блокировки. Если первый замок блокирует другой замок, два замка называются *несовместимыми*; в противном случае замки *совместимы*. Часто блокирующие взаимодействия блокировки представлены в технической литературе по таблице совместимости Lock. Ниже приведен пример с основными основными типами блокировок:

Таблица совместимости замков

Тип блокировки	блокировка чтения	блокировка записи
блокировка чтения		**X**
блокировка записи	**X**	**X**

X указывает на несовместимость, то есть случай, когда блокировка первого типа (в левом столбце) объекта блокирует блокировку второго типа (в верхней строке) от того, чтобы быть полученным на одном и том же объекте (другой транзакцией). Обычно объект имеет очередь ожидающих (по транзакциям) операций с соответствующими блокировками. Первый заблокированный замок для работы в очереди приобретается, как только существующая блокирующая блокировка удаляется из объекта, а затем выполняется соответствующая операция. Если блокировка для работы в очереди не блокируется какой-либо существующей блокировкой (возможно одновременное существование нескольких совместимых замков на одном и том же объекте), она приобретается немедленно.

Комментарий. В некоторых публикациях записи в таблице просто помечены как «совместимые» или «несовместимые», или, соответственно, «да» или «нет».

Двухфазная блокировка и ее особые случаи

Двухфазная блокировка

Согласно протоколу **двухфазной блокировки** транзакция обрабатывает свои блокировки в двух разных последовательных фазах во время выполнения транзакции:

1. **Фаза расширения** (или Growing phase): фиксируются замки и не блокируются блокировки (количество замков может только увеличиваться).
2. **Фаза сжимания**: замки освобождаются, и блокировки не приобретаются.

Правило двух фазовых блокировок можно суммировать как: никогда не получить блокировку после того, как блокировка была выпущена. Свойство сериализуемости гарантируется для расписания с транзакциями, которые подчиняются этому правилу.

Как правило, без явных знаний в транзакции по окончании фазы-1 это безопасно определяется только тогда, когда транзакция завершила обработку и запросила фиксацию. В этом случае все блокировки могут быть сразу выпущены (фаза-2).

Консервативная двухфазная блокировка

Разница между 2PL и C2PL заключается в том, что транзакции C2PL получают все блокировки, необходимые для начала транзакций. Это делается для того, чтобы транзакция, которая уже содержит некоторые блокировки, не будет блокировать ожидание других блокировок. Консервативный 2PL предотвращает взаимоблокировки.

Строгая двухфазная блокировка

Чтобы соответствовать протоколу S2PL, транзакция должна соответствовать требованиям 2PL и освобождать блокировки записи (эксклюзивные) только после ее завершения, то есть быть либо совершенной, либо прерванной. С другой стороны, блокировки чтения (общего доступа) выпускаются регулярно во время фазы 2. Этот протокол не подходит для B-деревьев, потому что он вызывает узкое место (в то время как B-деревья всегда начинают поиск от родительского корня).

Сильная строгая двухфазная блокировка

или **Строгость**, или **Строгое планирование**, или **Строгая двухфазная блокировка**

Чтобы соответствовать **строгой двухфазной блокировке** (SS2PL), протокол блокировки освобождает блокировку записи (исключение) и чтения (общего доступа), применяемую транзакцией, только после завершения транзакции, т. Е. Только после завершения выполнения (готовности) и становясь либо совершенными, либо прерванными. Этот протокол также соответствует правилам S2PL. Транзакция, подчиняющаяся SS2PL, может рассматриваться как имеющая фазу-1, которая длится всю продолжительность выполнения транзакции, а не фазу-2 (или вырожденную фазу-2). Таким образом, остается только одна фаза, и, по-видимому, «двухфазное» имя по-прежнему используется из-за исторического развития концепции из 2PL, а 2PL — суперкласса. Свойство SS2PL расписания также называется **Rigorousness**. Это также имя класса расписаний, имеющих это свойство, а расписание SS2PL также называется «строгим графиком». Термин «Жесткость» свободен от ненужного наследия «двухфазного», а также не зависит от какого-либо (блокирующего) механизма (в принципе могут быть использованы другие блокирующие механизмы). Соответствующий механизм блокировки собственности иногда называют **Rigorous 2PL**.

SS2PL является особым случаем S2PL, то есть класс расписаний SS2PL является надлежащим подклассом S2PL (каждое расписание SS2PL также является расписанием S2PL, но существуют графики S2PL, которые не являются SS2PL).

SS2PL является протоколом управления параллельным доступом для большинства систем баз данных и используется с ранних дней в 1970-х годах. Он доказал, что является эффективным механизмом во многих ситуациях и обеспечивает помимо Serializability также Strictness (особый случай использования без каскада), который помогает эффективному восстановлению базы данных, а также упорядочивание обязательств (CO) для участия в распределенных средах, где CO основанные на распределенной сериализуемости и глобальные решения для сериализации. Будучи подмножеством CO, эффективная реализация распределенного SS2PL существует без распределенного диспетчера блокировок (DLM), тогда как распределенные взаимоблокировки (см. Ниже) разрешаются автоматически. Тот факт, что SS2PL, используемый в системах с несколькими базами данных, обеспечивает глобальную сериализуемость, известен за многие годы до открытия CO, но только с CO понимали роль протокола атомных обязательств в поддержании глобальной сериализуемости, а также наблюдение за автоматическим (см. подробный пример распределенного SS2PL). На самом деле SS2PL, наследующий свойства Recoverability и CO, является более значительным, чем подмножество 2PL, которое само по себе в его общем виде, кроме того, содержит простой механизм сериализации (однако сериализуемость также подразумевается CO), в неизвестном предоставить SS2PL любые другие существенные качества. 2PL в его общем виде, а также в сочетании со строгим, то есть строгим 2PL (S2PL), как известно, не используются на практике. Популярный SS2PL не требует маркировки «конец фазы-1», как 2PL и S2PL, и, следовательно, проще реализовать. Кроме того, в отличие от общего 2PL, SS2PL предоставляет, как упоминалось выше, полезные свойства упорядочивания и фиксации обязательств.

Существует множество вариантов SS2PL, которые используют различные типы блокировок с различной семантикой в разных ситуациях, включая случаи изменения блокировки во время транзакции. Известны варианты, которые используют блокировку множественного зернистости.

Тупики в двухфазной блокировке

Блокирует блокировку доступа к данным. Взаимная блокировка между транзакциями приводит к тупиковой ситуации, когда выполнение этих транзакций застопоривается и завершение не может быть достигнуто. Таким образом, необходимо устранить взаимоблокировки, чтобы выполнить эти транзакции и высвободить связанные с ними вычислительные ресурсы. Тупик — это отражение потенциального цикла в графе приоритетов, который будет происходить без блокировки. Тупик разрешается путем прерывания транзакции, связанной с таким потенциальным циклом, и прерывания цикла. Он часто обнаруживается с использованием графика ожидания (график конфликтов, заблокированных блокировками от материализации, конфликты, не материализованные в базе данных из-за заблокированных операций, не отражаются в графе приоритетов и не влияют на сериализуемость), что указывает, какая транзакция «ждет» блокировки, с помощью которой транзакция, а цикл означает тупик. Прерывание одной транзакции за цикл является достаточным для разрыва цикла. Если транзакция была прервана из-за ошибки взаимоблокировки, приложение может решить, что делать дальше. Обычно приложение перезапускает транзакцию с самого начала, но может задержать это действие, чтобы предоставить другим транзакциям достаточное время для завершения, чтобы избежать возникновения другого тупика. [6]

В распределенной среде для атомарности используется протокол атомных обязательств, как правило, двухфазный протокол фиксации (2PC). Когда восстанавливаемые данные (данные под управлением транзакций) разделены между участниками 2PC (т. Е. Каждый объект данных контролируется одним участником 2PC), то распределенные (глобальные) взаимоблокировки, взаимоблокировки с участием двух или более участников в 2PC автоматически решаются следующим образом:

Когда SS2PL эффективно используется в распределенной среде, то глобальные взаимоблокировки, связанные с блокировкой, генерируют блокировки голосования в 2PC и автоматически разрешаются с помощью 2PC (см. «Упорядочение обязательств» (CO) «в» Точной характеристике блокировок для голосования по глобальным циклам "; за исключением статей CO, как известно, это замечает). Для общего случая 2PL глобальные взаимоблокировки аналогичным образом автоматически разрешаются протоколом точки синхронизации для

этапа 1 в распределенной транзакции (точка синхронизации достигается путем «голосования» (уведомление о локальном фазе-1) и распространения на участников в распределенной транзакции так же, как точка принятия решения в атомном режиме, по аналогии с точкой принятия решения в CO, конфликтная операция в 2PL не может произойти до момента синхронизации фазы-1, с той же результирующей тупиковой ситуацией в случае глобальный тупик доступа к данным, тупик (который также является глобальным тупиком на основе блокировки) автоматически разрешается протоколом, прерывающим некоторые транзакции, с отсутствующим голосованием, обычно использующим тайм-аут).

Комментарий:

Когда данные распределяются между участниками протокола атомного обязательства (например, 2PC), автоматическая глобальная ошибка взаимоблокировки была упущена в литературной литературе базы данных, хотя взаимоблокировки в таких системах были довольно интенсивной областью исследований:

- Для CO и его специального случая SS2PL автоматическое разрешение по протоколу атомного обязательства было замечено только в CO-статьях. Однако на практике было замечено, что во многих случаях глобальные взаимоблокировки очень редко обнаруживаются с помощью специальных механизмов разрешения, меньше, чем можно было ожидать («Почему мы видим так мало глобальных взаимоблокировок?»). Причина в том, что это, вероятно, тупики, которые автоматически разрешаются и, таким образом, не обрабатываются и не учитываются механизмами;
- Для 2PL в общем случае автоматическое разрешение (обязательным) протоколом точки синхронизации по завершении фазы один (с тем же механизмом голосования, что и протокол атомных обязательств, и одинаковое отсутствие обработки голосования при тупике в результате голосования, что приводит к глобальному разрешению взаимоблокировки) не упоминалось до сегодняшнего дня (2009). Практически используется только специальный случай SS2PL, где в дополнение к протоколу атомарного коммита не требуется синхронизация в конце фазы фазы.

В распределенной среде, где восстановимые данные не разделяются между участниками протокола по использованию атомных обязательств, такого автоматического разрешения не существует, а распределенные взаимоблокировки должны решаться с помощью специальных методов.

Примечания

1. Philip A. Bernstein, Vassos Hadzilacos, Nathan Goodman (1987): *Concurrency Control and Recovery in Database Systems*, Addison Wesley Publishing Company, ISBN 0-201-10715-5
2. Gerhard Weikum, Gottfried Vossen (2001): *Transactional Information Systems*, Elsevier, ISBN 1-55860-508-8

Ссылки

- Граф предшествования
- Блокировка (программирование)
