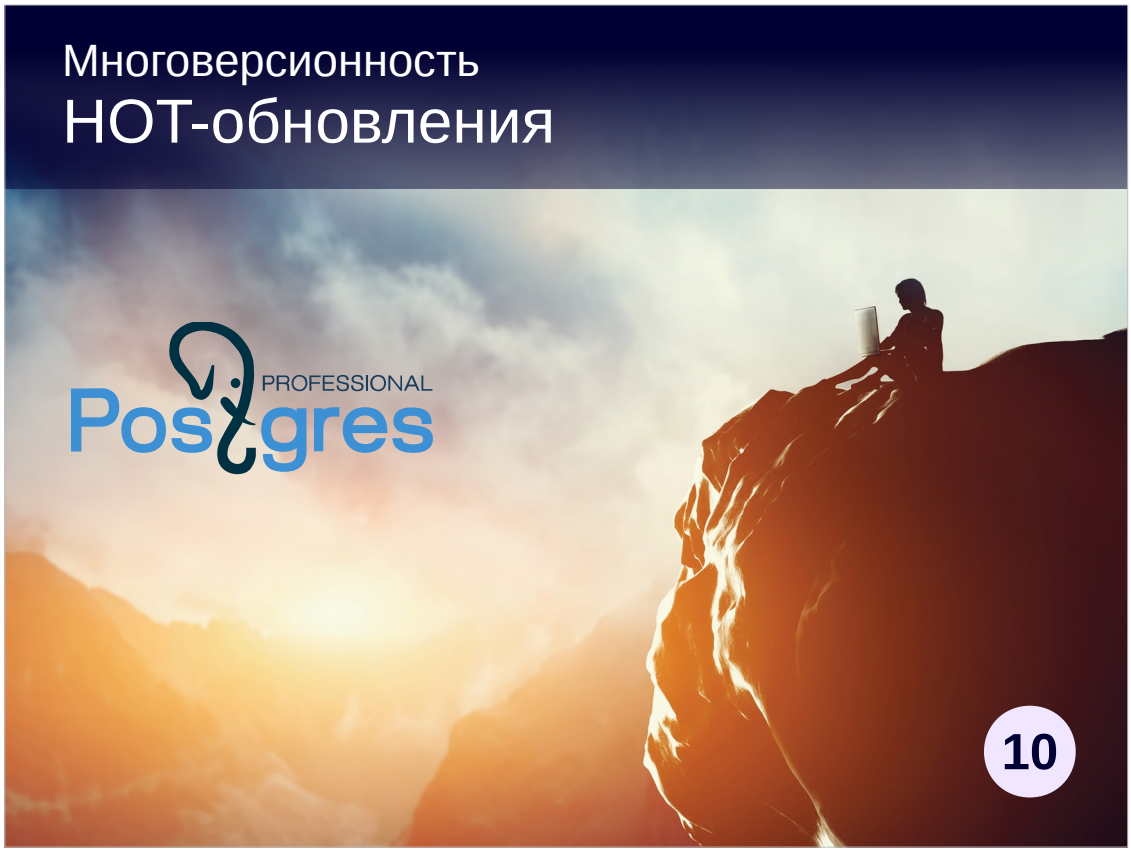


Авторы: Егор Рогов, Павел Лузанов

Обратная связь
Отзывы, замечания и предложения направляйте по адресу:
edu@postgrespro.ru

Темы

HOT-обновления

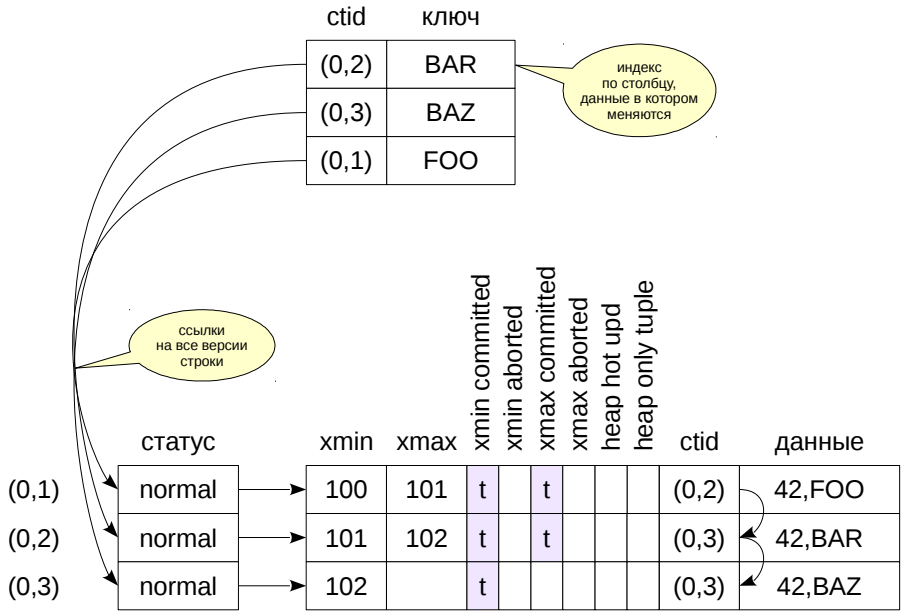
Внутристраничная очистка

HOT-обновления

Проблемы обычных обновлений

Устройство HOT-обновления

Обычное обновление



Напомним, что при обычном обновлении в индексе создаются ссылки на все версии строки, присутствующие в табличной странице.

Проблемы обновления

При любом обновлении строки надо изменять индекс

- страдает производительность вставок и изменений

В индексе накапливаются ссылки на неактуальные версии

- размер индекса растет, требуется очистка

Все сложности умножаются на количество индексов, построенных по таблице

Чем это плохо?

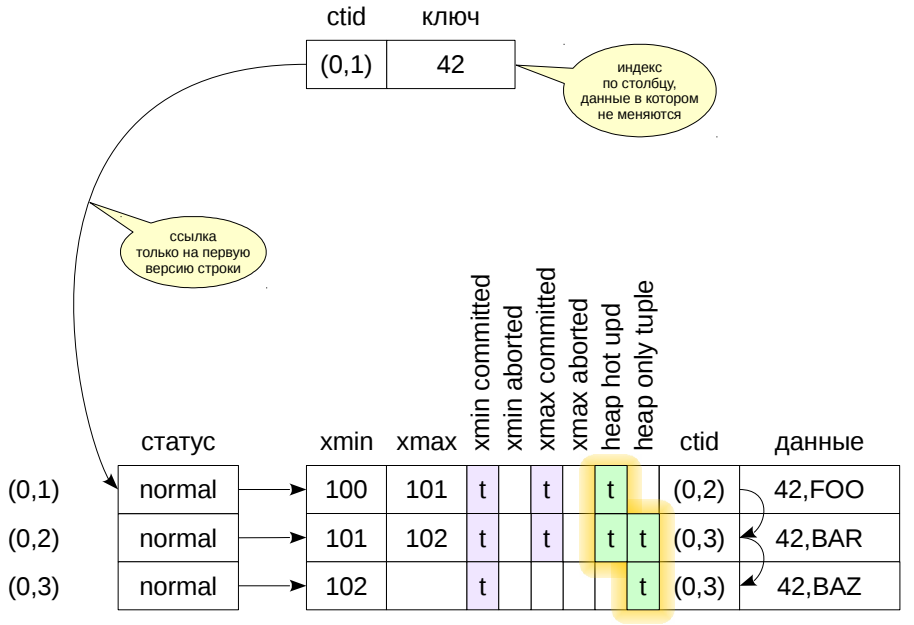
Во-первых, при любом изменении строки приходится обновлять все индексы, созданные для таблицы (даже если измененные поля не входят в индекс). Очевидно, это снижает производительность.

Во-вторых, в индексах накапливаются ссылки на исторические версии строки, которые потом приходится очищать.

Более того, есть особенность реализации B-дерева в PostgreSQL. Если на индексной странице недостаточно места для вставки новой строки, страница делится на две и все данные перераспределяются между ними. Однако при удалении (или очистке) строк две индексные страницы не «склеиваются» в одну. Из-за этого размер индекса может не уменьшиться даже при удалении существенной части данных.

Естественно, чем больше индексов создано на таблице, тем с большими сложностями приходится сталкиваться.

HOT-обновление



Однако если индекс создан по полю, значение которого не изменилось в результате обновления строки, то нет смысла создавать дополнительную строку в B-дереве, содержащую то же самое значение ключа. Именно так работает оптимизация, называемая HOT-обновлением — Heap-Only Tuple Update.

При таком обновлении в индексной странице находится лишь одна строка, ссылающаяся на первую версию строки табличной страницы. А внутри табличной страницы организуется цепочка версий:

- строки, которые изменены и входят в цепочку, маркируются битом Heap Hot Updated;

- строки, на которые нет ссылок из индекса, маркируются битом Heap Only Tuple (то есть — «только табличная версия строки»);

- поддерживается обычная связь версий строк через поле ctid.

Если при сканировании индекса PostgreSQL попадает в табличную страницу и обнаруживает версию, помеченную как Heap Hot Updated, он понимает, что надо пройти дальше по цепочке обновлений. (Разумеется, для всех полученных таким образом версий строк проверяется видимость, прежде чем они будут возвращены клиенту.)

https://git.postgresql.org/gitweb/?p=postgresql.git;a=blob;f=src/backend/access/heap/README.HOT;hb=HEAD

HOT-обновление

Обновляемый столбец не должен входить ни в один индекс

- иначе на версию строки будет ссылка из индекса
  и ее нельзя пометить как «heap only»

Цепочка обновлений — только в пределах одной страницы

- не требуется обращение к другим страницам,
  обход цепочки не ухудшает производительность
- если в табличной странице не хватает места для новой версии,
  цепочка обрывается (как если бы оптимизация не работала)
- место в странице можно зарезервировать, уменьшив параметр
  хранения таблицы *fillfactor* (100 % → 10 %)

Подчеркнем, что HOT-обновления работают в случае, если обновляемые поля не входят *ни в один индекс.* Иначе в каком-либо индексе оказалась бы ссылка непосредственно на новую версию строки, что противоречит идее этой оптимизации.

Оптимизация действует только в пределах одной страницы, поэтому дополнительный обход цепочки не требует обращения к другим страницам и не ухудшает производительность.

Однако если на странице не хватит свободного места, чтобы разместить новую версию строки, цепочка прервется. На версию строки, размещенную на другой странице, придется сделать и ссылку из индекса.

Поэтому при частых обновлениях неиндексированных полей может иметь смысл уменьшать параметр хранения *fillfactor*. Этот параметр определяет пороговый процент занятого на странице места, после которого вставка новых строк в эту страницу будет запрещена. Оставшееся место остается зарезервированным для обновлений: при обновлении новая версия строки может занять свободное место на странице. (С другой стороны, чем выше *fillfactor,* тем компактнее располагаются записи и, соответственно, размер таблицы получается меньше.)

Внутристраничная очистка

При обычных обновлениях

При HOT-обновлениях

Внутристраничная очистка

Выполняется при любом обращении к странице

- если ранее выполненное обновление не нашло места для новой версии строки на этой же странице
- если страница заполнена больше, чем на *fillfactor*

Действует в пределах одной табличной страницы

- не освобождает указатели, на которые могут ссылаться индексы
- не обновляет карту свободного пространства
- не обновляет карту видимости

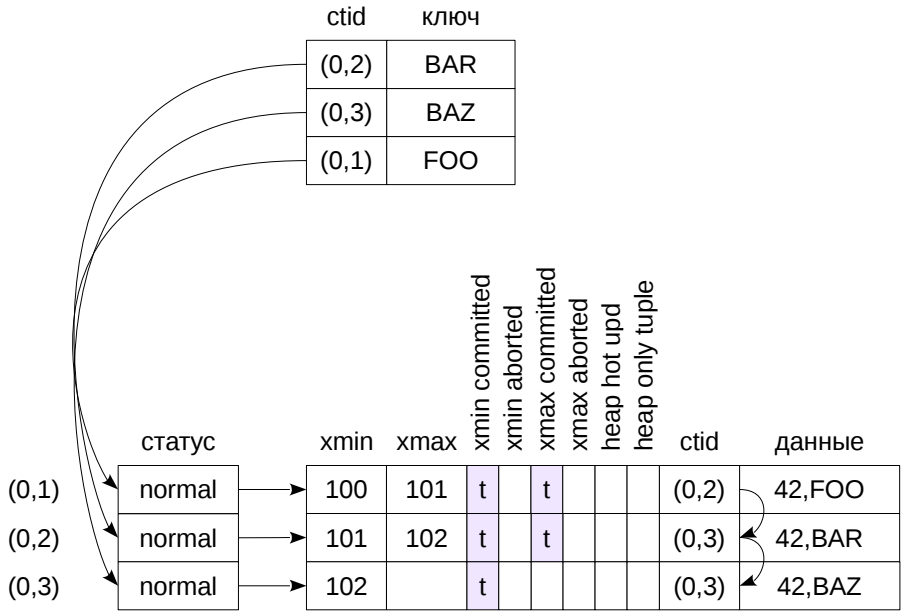
При обращении к странице — как при обновлении, так и при чтении — может происходить быстрая внутристраничная очистка, если PostgreSQL поймет, что место на странице заканчивается. Критериев два. Первый: ранее выполненное на этой странице обновление не обнаружило достаточного места, чтобы разместить новую версию строки на той же странице. Второй: если страница заполнена больше, чем на *fillfactor*.

Внутристраничная очистка убирает версии строк, не видимые ни в одном снимке (находящиеся за «горизонтом событий» базы данных), но работает строго в пределах одной табличной страницы. Указатели на версии строк не освобождаются, так как на них могут быть ссылки из индексов, а это уже другая страница — она не очищается.

Карта свободного пространства не обновляется — из экономии ресурсов, а также из соображения, что освобожденное место лучше приберечь для обновлений, а не для вставок. Также не обновляется и карта видимости.

Тот факт, что страница может очищаться при чтении, означает, что запрос SELECT может вызвать изменение страниц. Это еще один такой случай, в дополнение к рассмотренному в теме «Страницы и версии строк» изменению битов-подсказок.

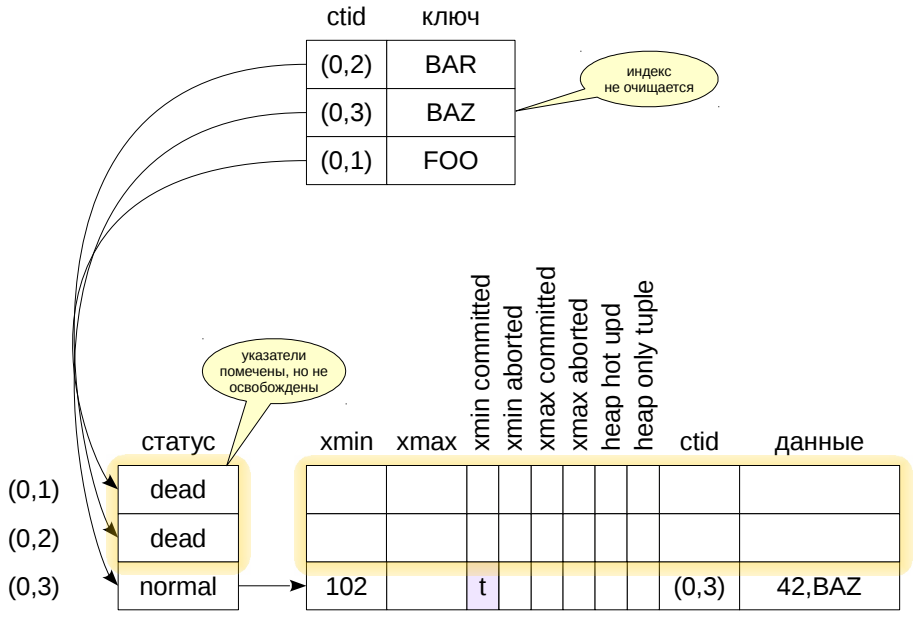
До очистки



На рисунке приведена ситуация до очистки. В табличной странице три версии одной строки. Две из них неактуальны, не видны ни в одном снимке и могут быть удалены.

В индексе есть три ссылки на каждую из версий строки.

После очистки



Если выполнены условия срабатывания внутристраничной очистки, две неактуальные версии строки (0,1) и (0,2) могут быть удалены. Указатели на удаленные версии получает статус «dead», а освободившееся место может быть использовано для вставки новой версии.

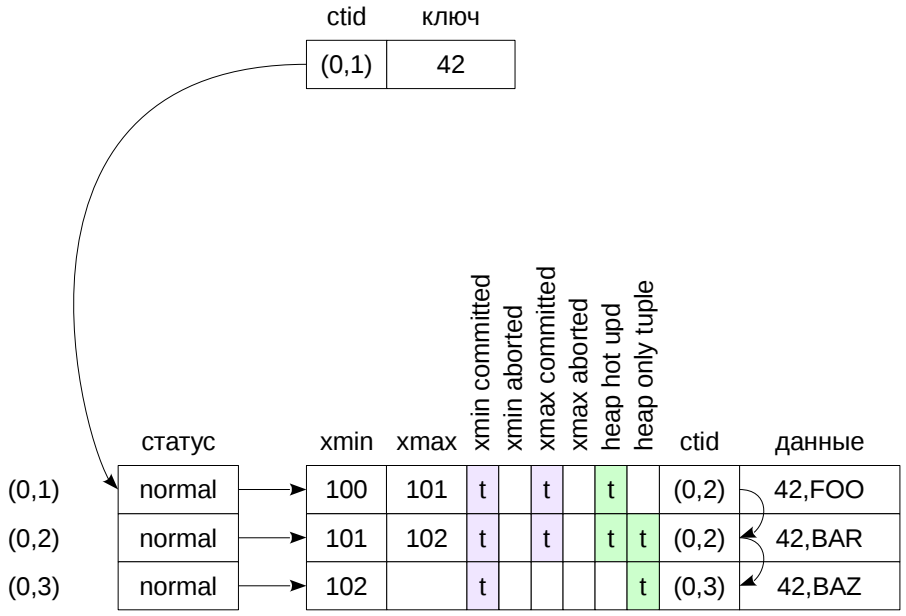
Все оставшиеся версии строк сдвигаются вместе так, чтобы свободное место на странице было представлено одним фрагментом. Соответствующим образом изменяются и значения указателей. Благодаря этому не возникает проблем с фрагментацией свободного места в странице.

Указатели на удаленные версии строк освободить нельзя, поскольку на них существует ссылки из индексной страницы. При индексном доступе PostgreSQL может получить (0,1) или (0,2) в качестве идентификатора версии строки, попробует получить саму строку из табличной станицы, но благодаря статусу указателя обнаружит, что эта версия уже не существует.

(На рисунке не показаны указатели на индексные строки. На самом деле, в первый раз обнаружив отсутствие версии табличной строки, PostgreSQL изменит и статус указателя в индексной странице, чтобы повторно не обращаться к табличной странице.)

Принципиально то, что внутристраничная очистка работает только в пределах одной табличной страницы и не очищает индексные страницы.

До HOT-очистки

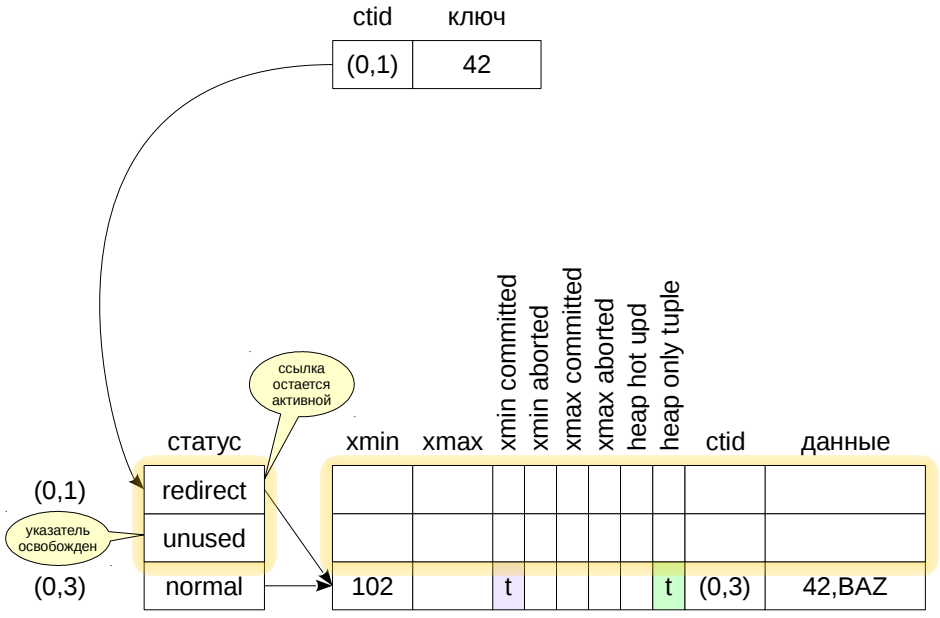


Частный, но важный случай внутристраничной очистки представляет собой очистка при HOT-обновлениях.

На рисунке приведена ситуация до очистки. В таблице три версии одной и той же строки. На первую из них (0,1) ссылается индекс, остальные две (0,2) и (0,3) помечены как «только табличные».

Версии строк (0,1) и (0,2) неактуальны, не видны ни в одном снимке и могут быть удалены.

После HOT-очистки



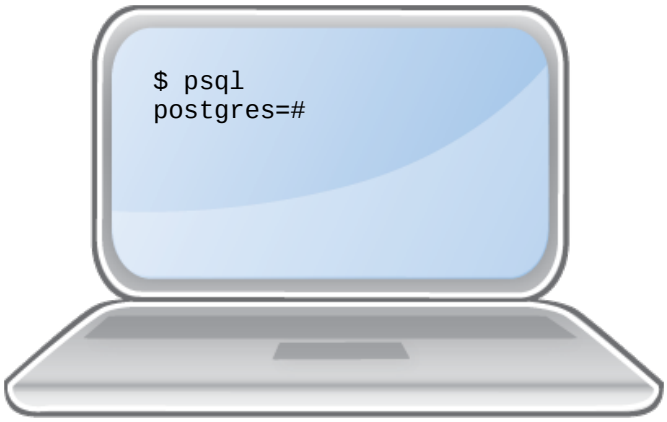
После срабатывания внутристраничной очистки неактуальные версии строк удаляются.

При HOT-обновлениях в индексе может быть только одна ссылка на версию строки, представляющую собой «голову» HOT-цепочки, которая поддерживается внутри одной табличной страницы. При любых изменениях версий строки, указатель должен оставаться на своем месте и ссылаться на голову цепочки.

Поэтому применяется двойная адресация: для указателя, на который ссылается индекс — в данном случае (0,1), — используется статус «redirect», перенаправляющий на нужную версию строки.

Указатель на вторую версию (0,2) больше не нужен. Он получает статус «unused» и будет использован при вставке какой-нибудь новой версии строки.

Демонстрация



Итоги

Если изменяемый столбец не входит ни в один индекс и на странице есть место — применяется HOT-обновление

При удобном случае автоматически выполняется быстрая внутристраничная очистка

При частых обновлениях можно подумать об уменьшении *fillfactor*

Практика

1. Воспроизведите ситуацию HOT-обновления на таблице с индексом по некоторым полям.
   Проверяйте содержимое табличной и индексной страниц с помощью расширения pageinspect.
2. Воспроизведите ситуацию внутристраничной HOT-очистки.
3. Воспроизведите ситуацию HOT-обновления, при которой внутристраничная очистка не освобождает достаточно места на странице и новая версия создается на другой странице. Сколько строк будет в индексе в этом случае?