Postgres PROFESSIONAL

Очистка

Темы

Внутристраничная очистка

Очистка при HOT-обновлениях

Очистка вручную (vacuum)

Полная очистка (vacuum full)

Внутристраничная очистка

Выполняется при любом обращении к странице

- если ранее выполненное обновление не нашло места на странице
- если страница заполнена больше, чем на fillfactor

Очищает старые версии строк, но не указатели

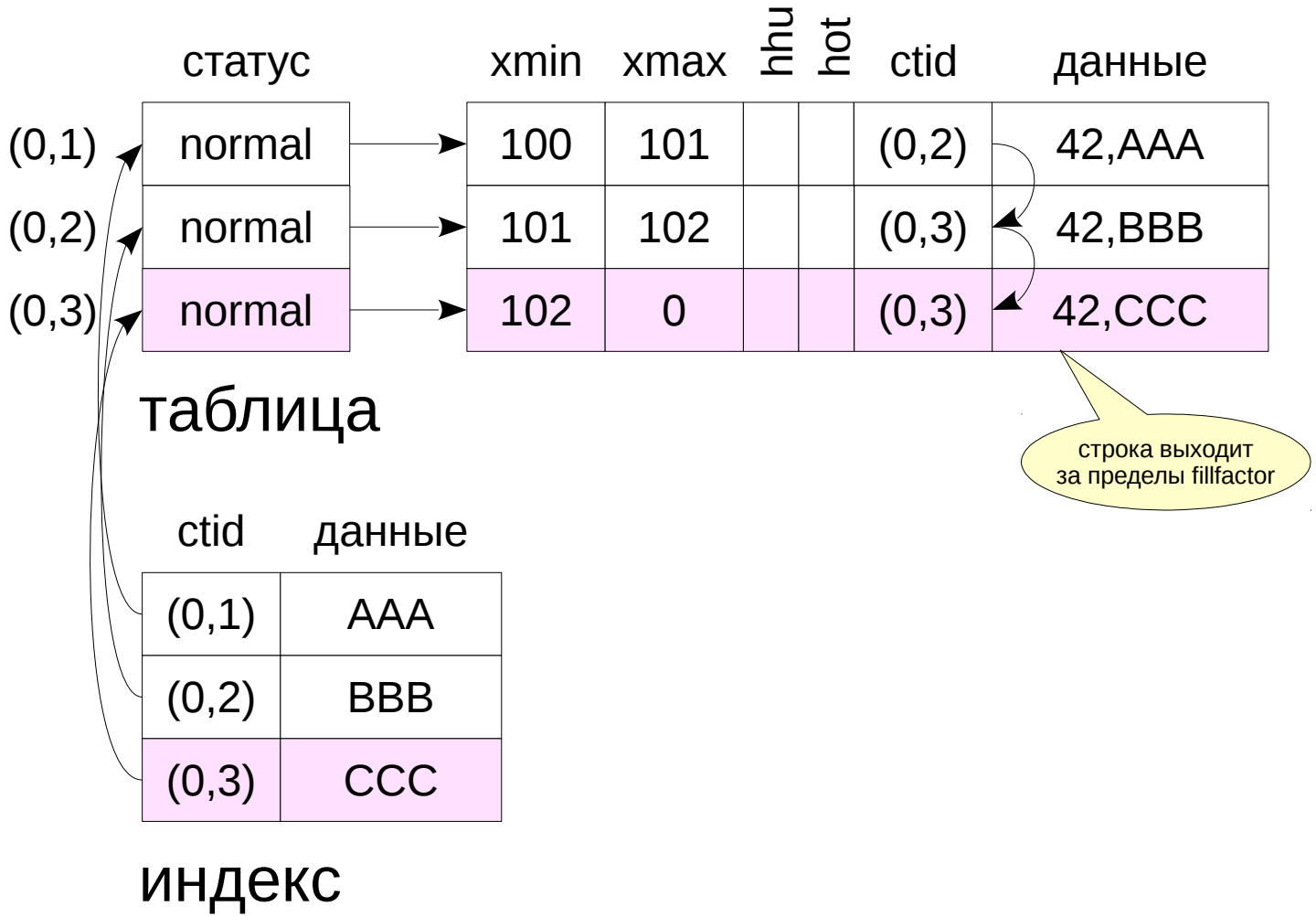
- так как на них могут ссылаться индексы

Не обновляет карту свободного пространства

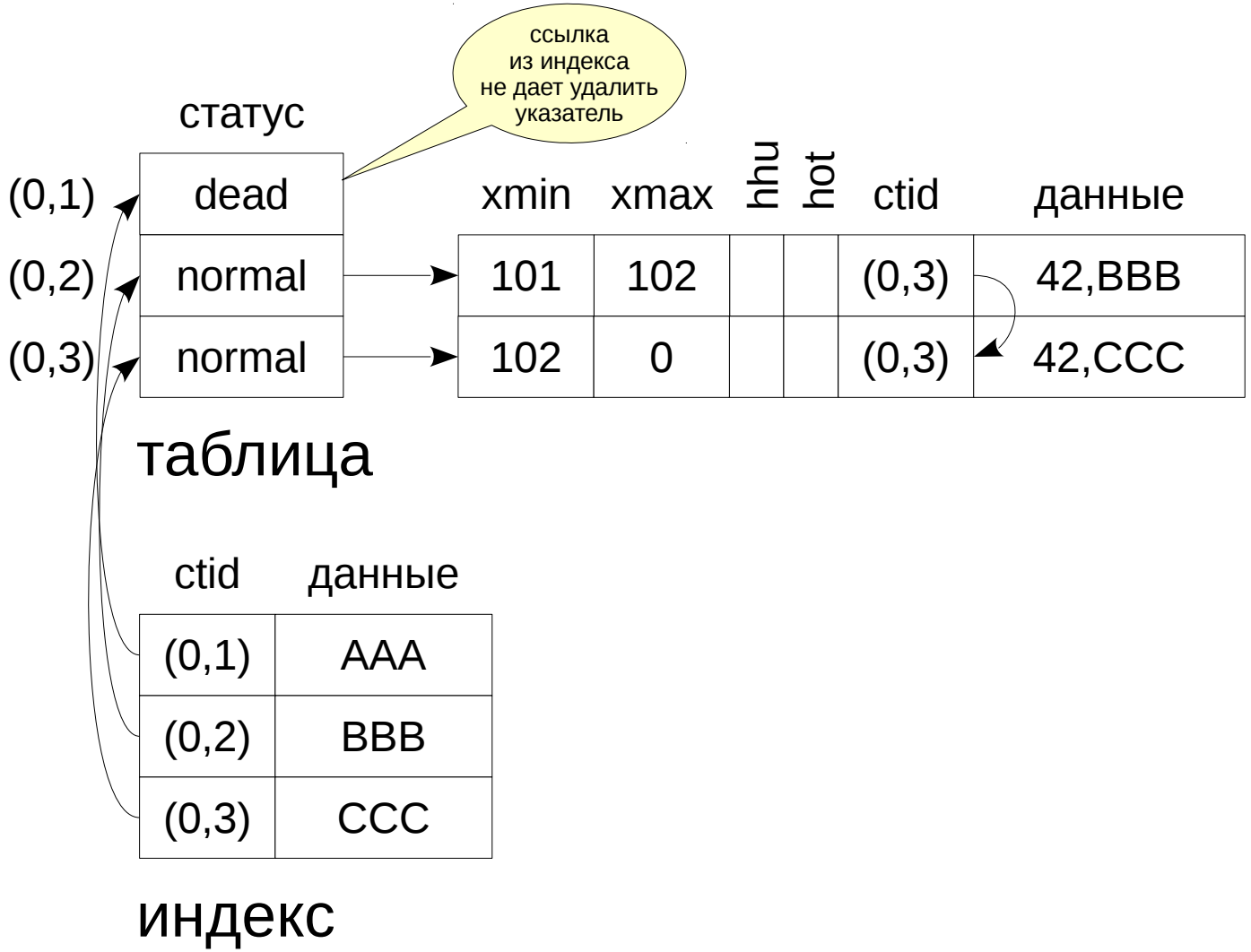
- освобождается место для обновлений,
  но новые строки не будут попадать на эту страницу

Не обновляет карту видимости

Внутристраничная очистка



Внутристраничная очистка



Внутристраничная очистка

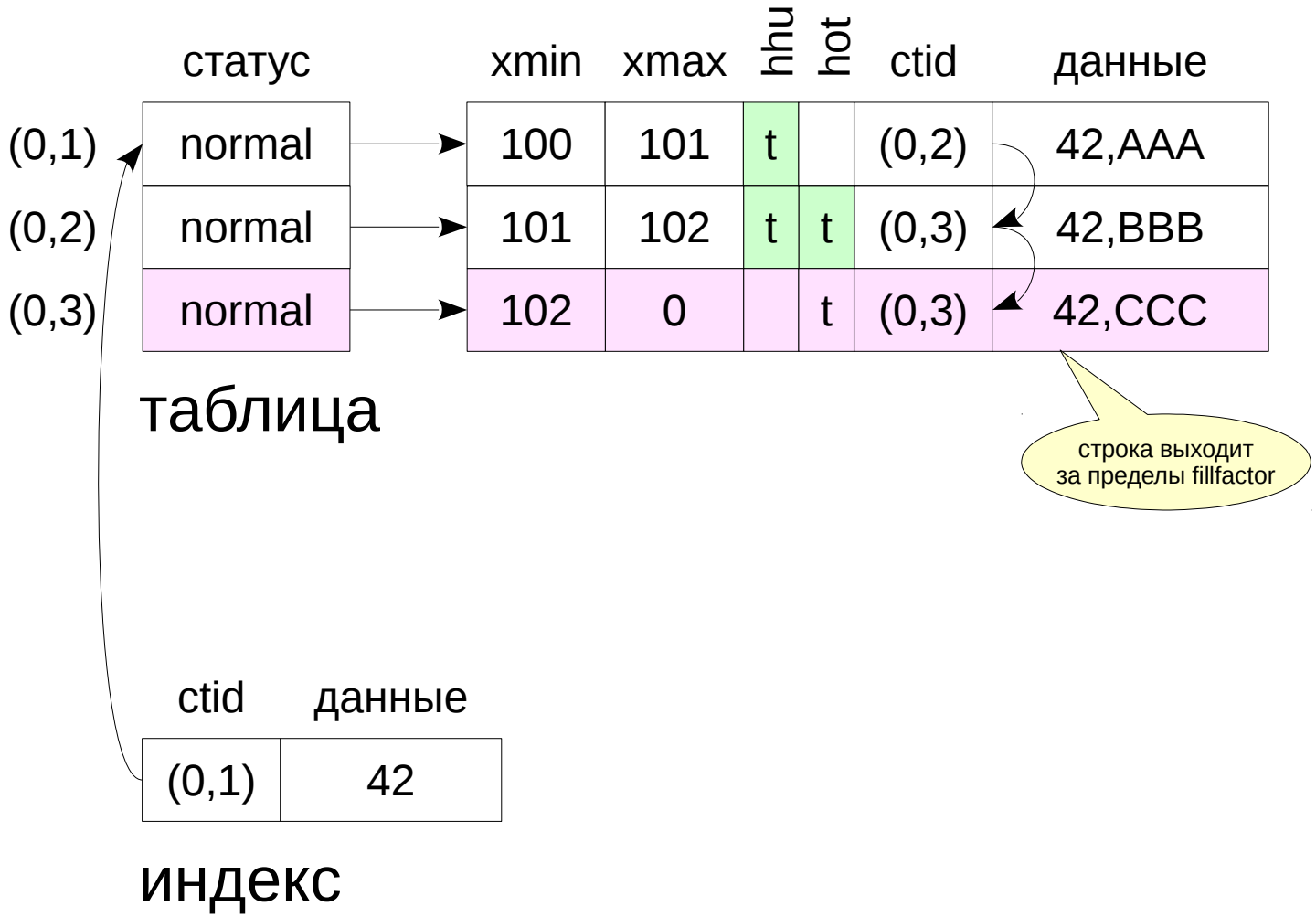
HOT-обновления

- элемент индекса ссылается на один указатель
- внутри страницы поддерживается цепочка обновлений

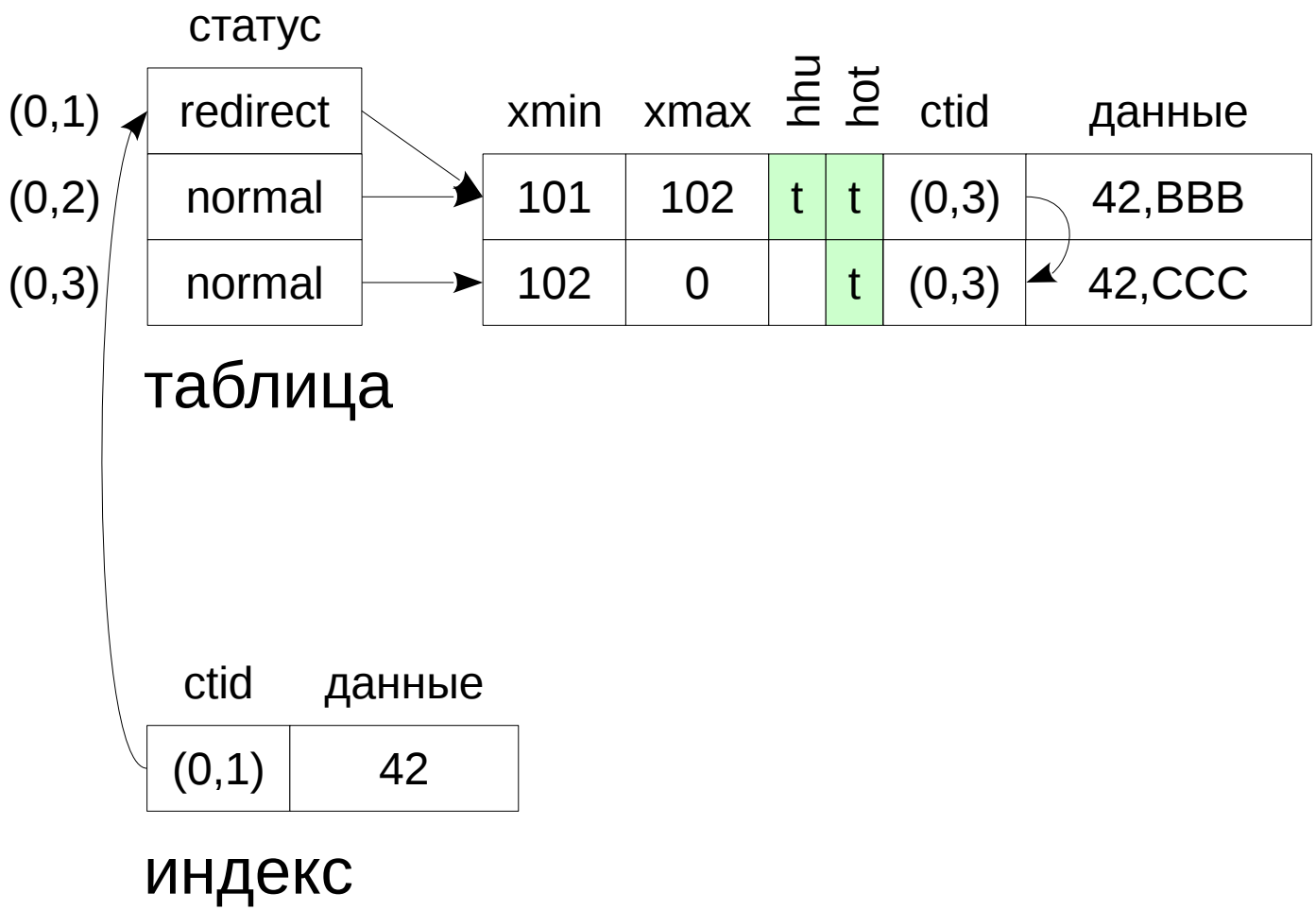
Особенности очистки

- указатель, на который ссылается индекс, должен остаться активным, даже если соответствующая версия строки больше не нужна
- поэтому вместо статуса «dead» используется «redirect» — перенаправление

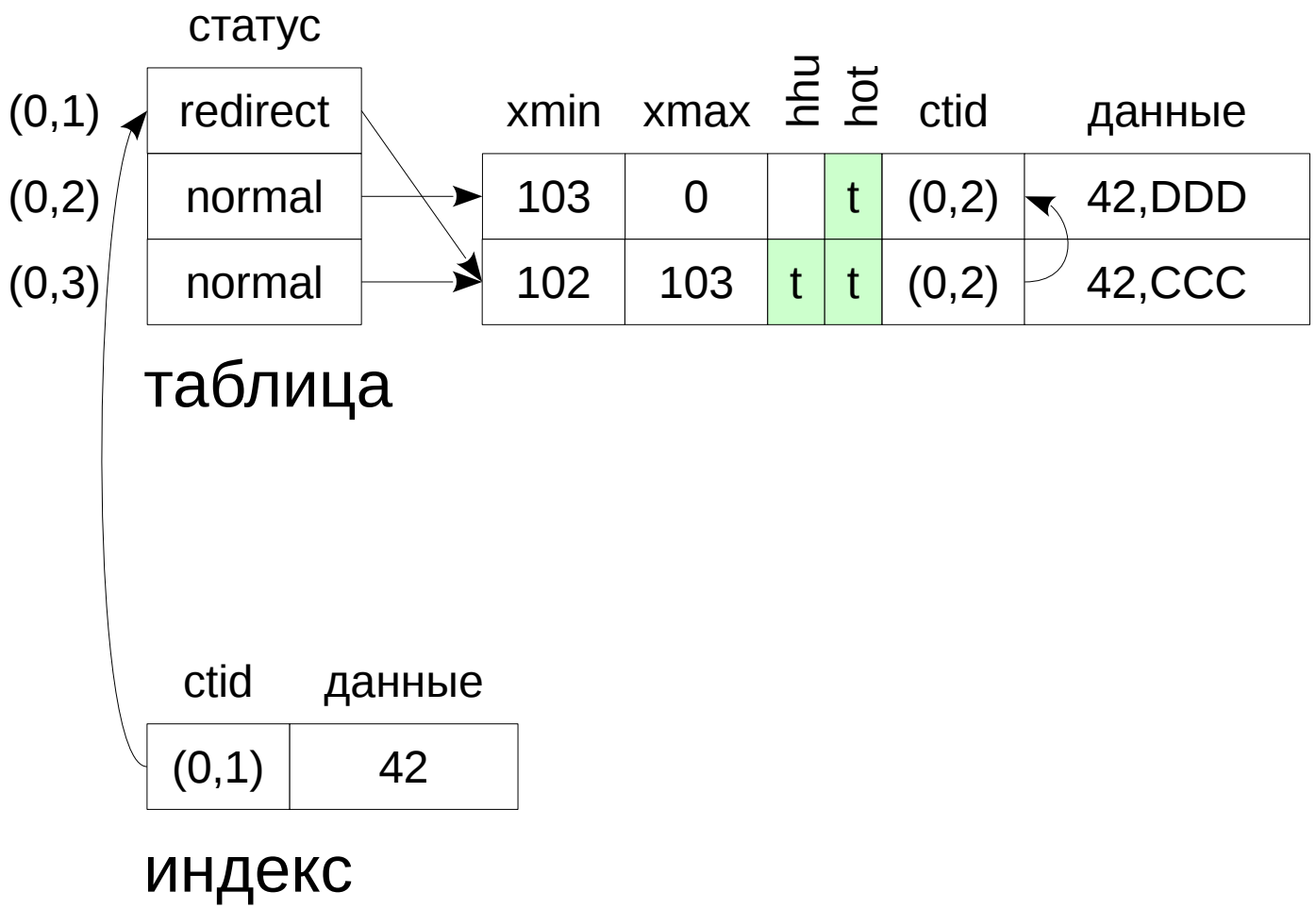
HOT-обновление



HOT-обновление



HOT-обновление



Ручная очистка

Выполняется при ручном запуске VACUUM

- для отдельных страниц таблицы и соответствующих индексов
- не рассматривает страницы, отмеченные в карте видимости

Очищает старые версии строк

- еще не прошедшие внутристраничную очистку

Очищает указатели

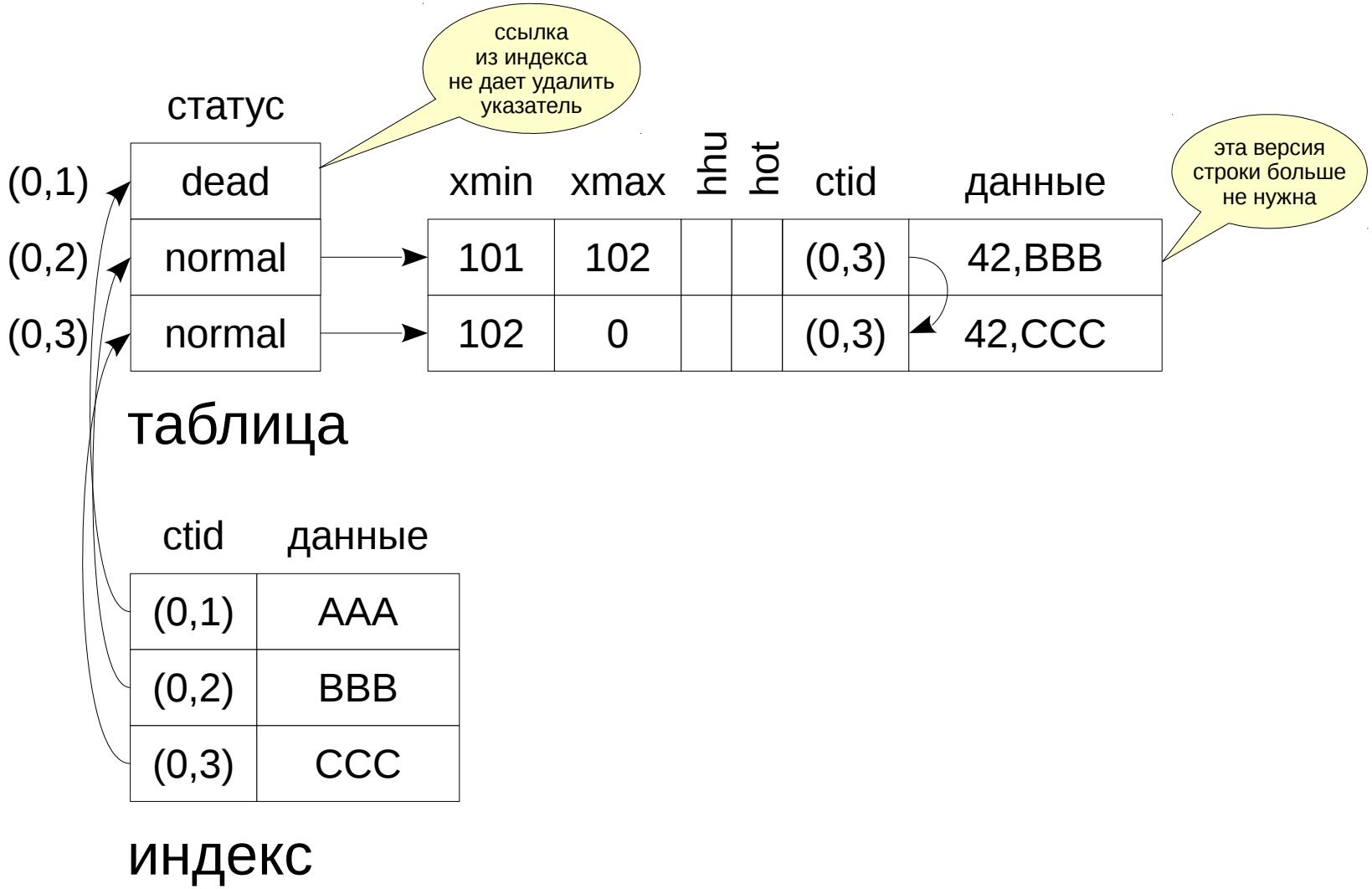
- так как одновременно очищает и индексные страницы

Обновляет карту свободного пространства

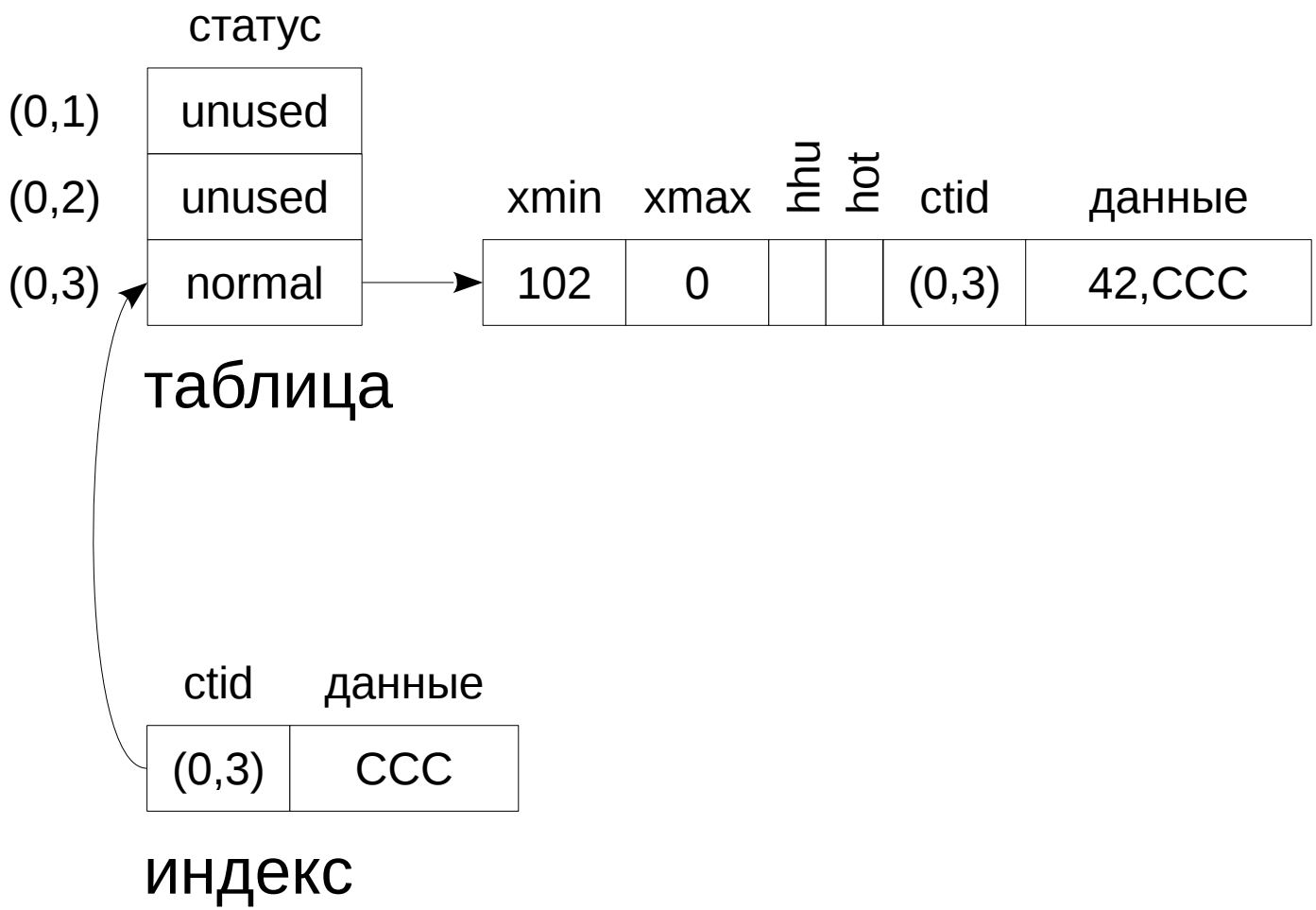
- освобождается место и для обновлений, и для новых строк

Обновляет карту видимости

Ручная очистка



Ручная очистка



Регулирование нагрузки

Процесс чередует работу и ожидание

примерно *vacuum_cost_limit* условных единиц работы,
затем засыпает на *vacuum_cost_delay* мс

Настройки

vacuum_cost_limit = 200

vacuum_cost_delay = 0 ms

стоимость обработки

vacuum_cost_page_hit = 1	*страницы в кэше*
vacuum_cost_page_miss = 10	*страницы на диске*
vacuum_cost_page_dirty = 20	*грязной страницы*

Полная очистка

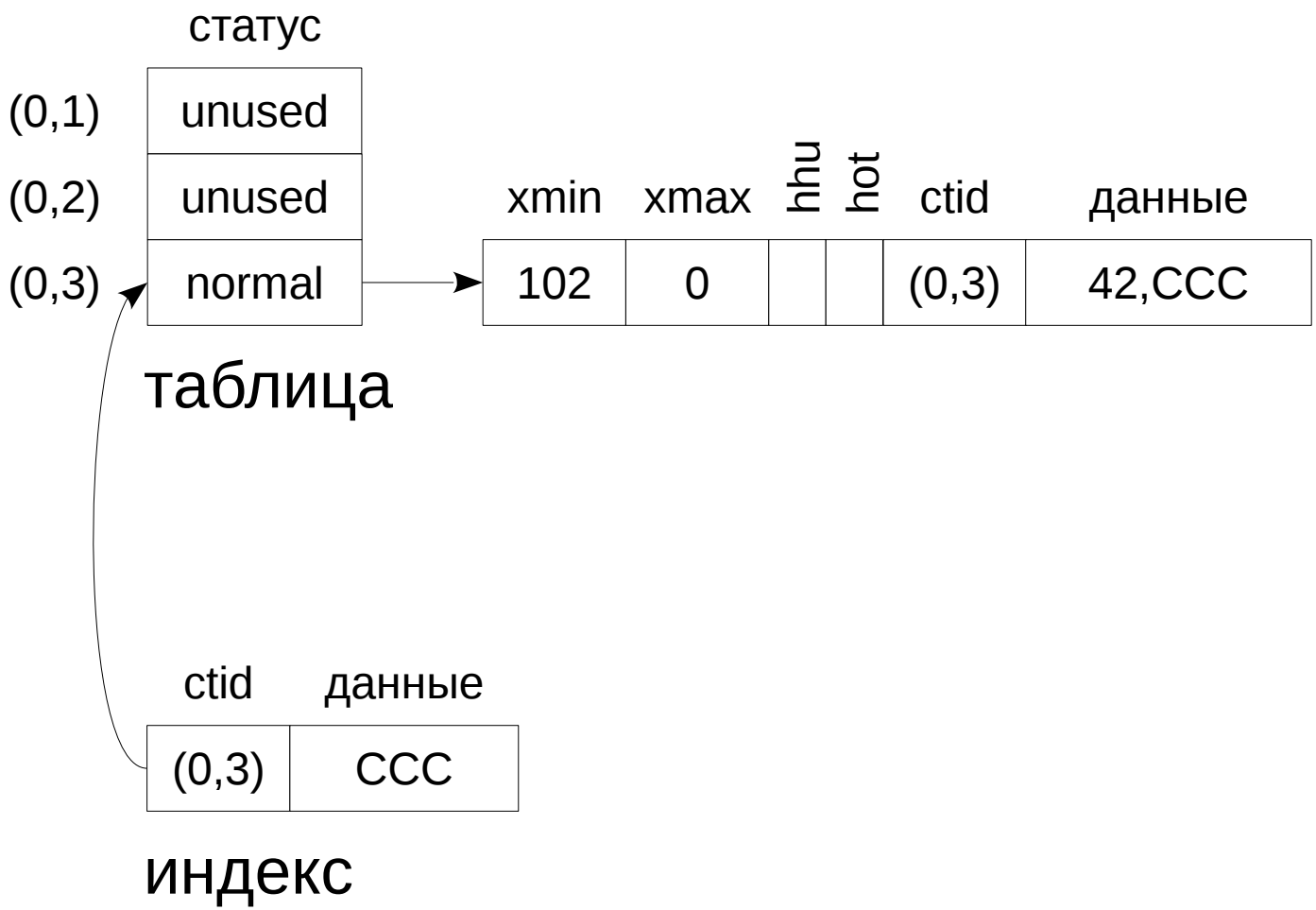
Выполняется при ручном запуске VACUUM FULL

Полностью перестраивает таблицу

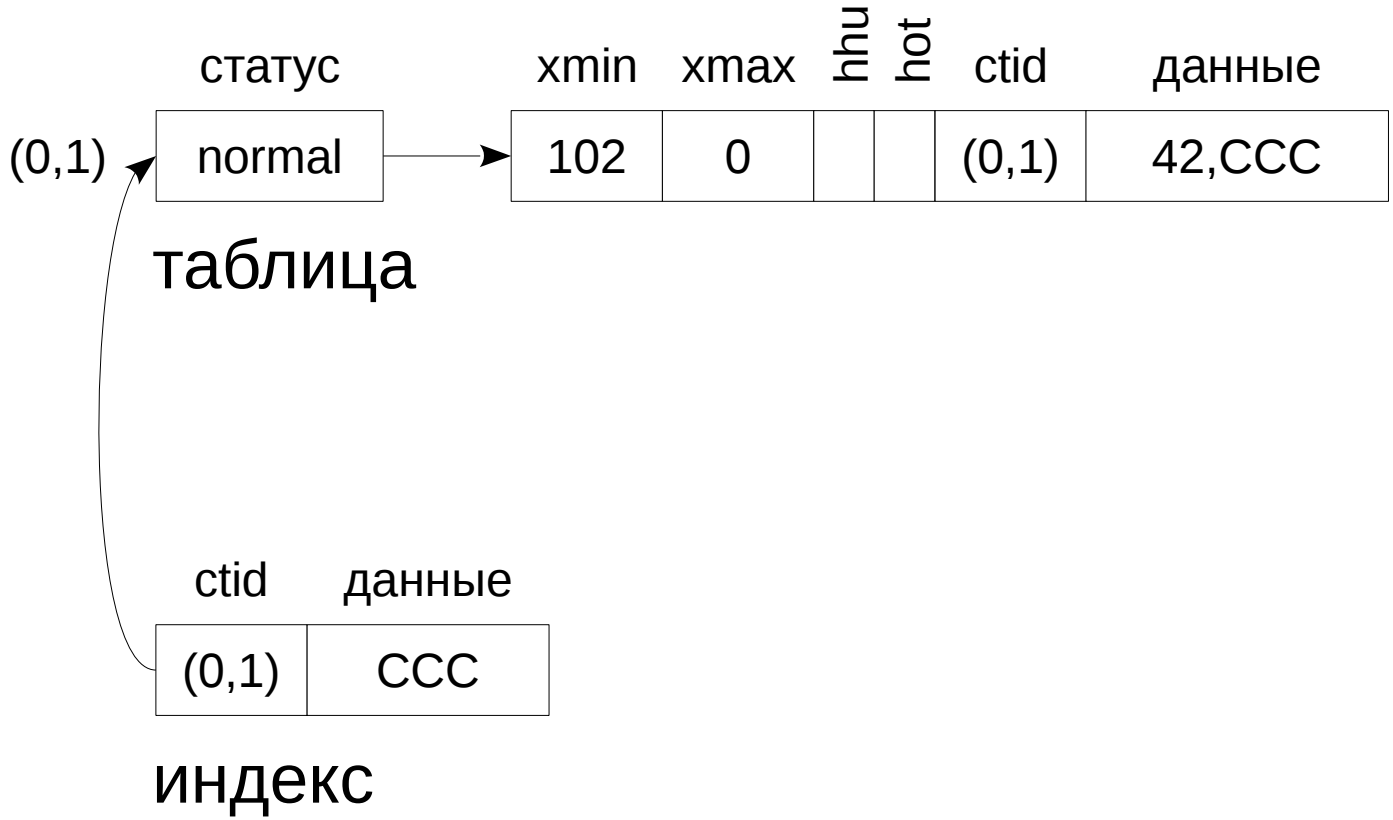
- перестраивает также все индексы на таблице
- освобожденное место возвращается операционной системе
- выполняется дольше, чем обычная очистка

Требует эксклюзивной блокировки

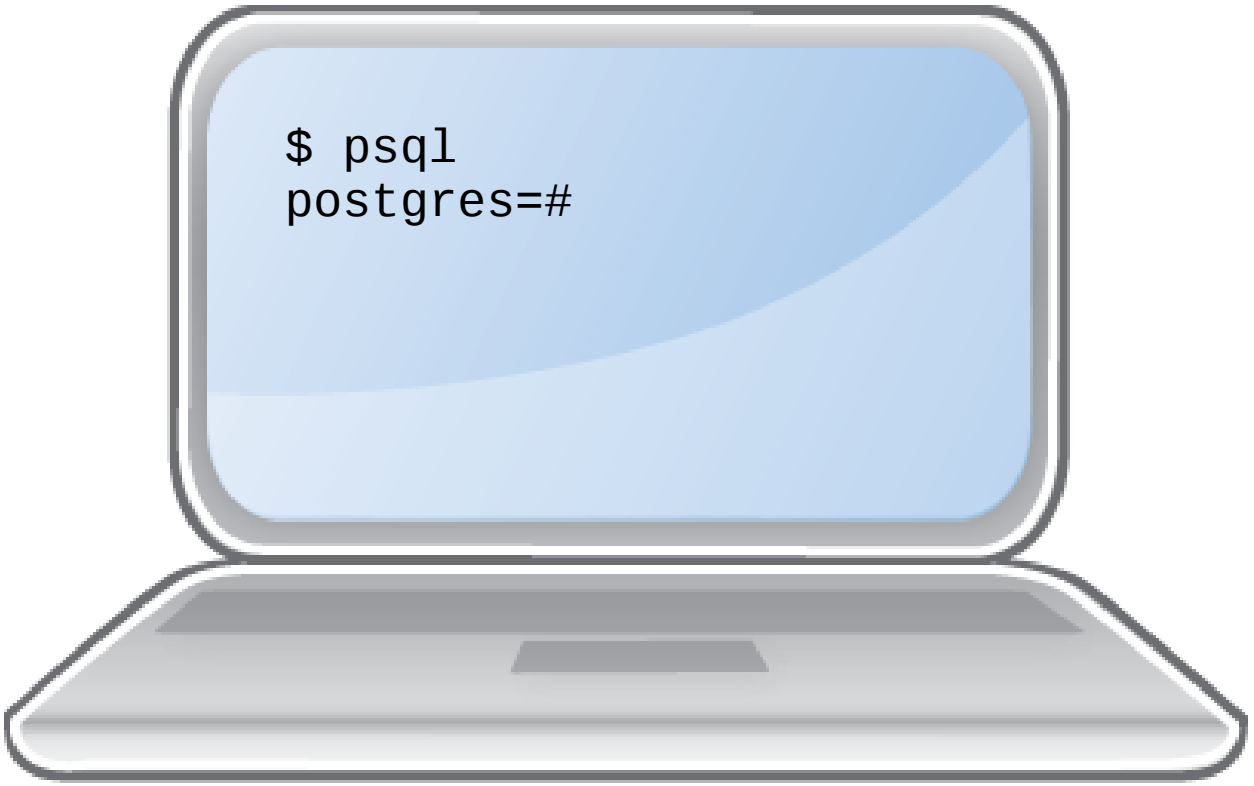
Полная очистка



Полная очистка



Демонстрация



Итоги

Многоверсионность требует очистки ненужных версий

При обращении к странице может происходить внутристраничная очистка

При частых обновлениях полезно уменьшить fillfactor

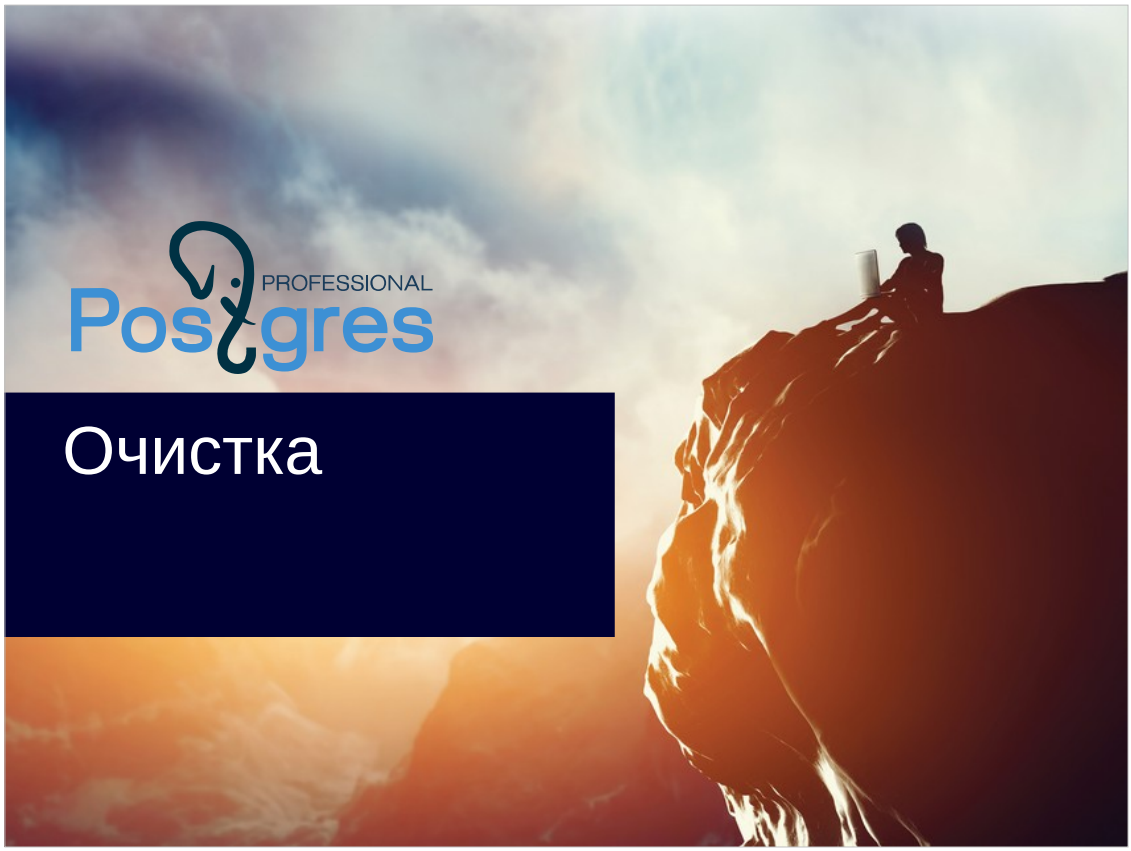
При ручной очистке (vacuum) освобождается место во всех страницах таблиц и индексов

Полная очистка (vacuum full) полностью перестраивает таблицы и индексы

Практика

1. В базе данных DB5 создать таблицу.
2. Воспроизвести ситуацию HOT-обновления. Проконтролировать содержимое страницы с помощью pageinspect.
3. Воспроизвести ситуацию внутристраничной очистки при HOT-обновлении.

4*. Воспроизвести ситуацию HOT-обновления, при которой внутристраничная очистка не освобождает достаточно места и новая версия помещается на другую страницу.



Авторы: Егор Рогов, Павел Лузанов

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

Темы

Внутристраничная очистка

Очистка при HOT-обновлениях

Очистка вручную (vacuum)

Полная очистка (vacuum full)

Внутристраничная очистка

Выполняется при любом обращении к странице

- если ранее выполненное обновление не нашло места на странице
- если страница заполнена больше, чем на fillfactor

Очищает старые версии строк, но не указатели

- так как на них могут ссылаться индексы

Не обновляет карту свободного пространства

- освобождается место для обновлений,
  но новые строки не будут попадать на эту страницу

Не обновляет карту видимости

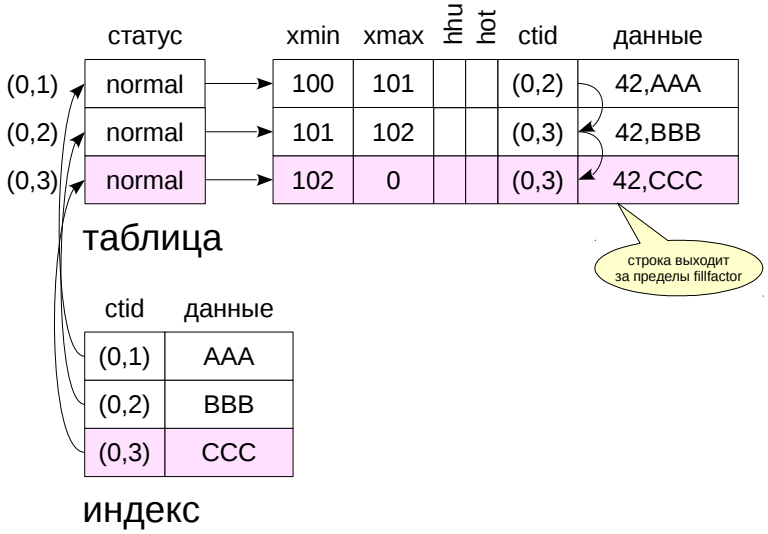
При обращении к странице — как при обновлении, так и при чтении — может происходить быстрая внутристраничная очистка. Для этого PostgreSQL должен понять, что место на странице заканчивается (например, если ранее выполненное обновление не обнаружило достаточного места, или если страница заполнена больше, чем на fillfactor).

Внутристраничная очистка убирает «достаточно старые» версии строк, не видимые ни в одном снимке. Указатели на версии строк не трогаются, так как на них могут быть ссылки из индексов, а это уже другая страница.

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.

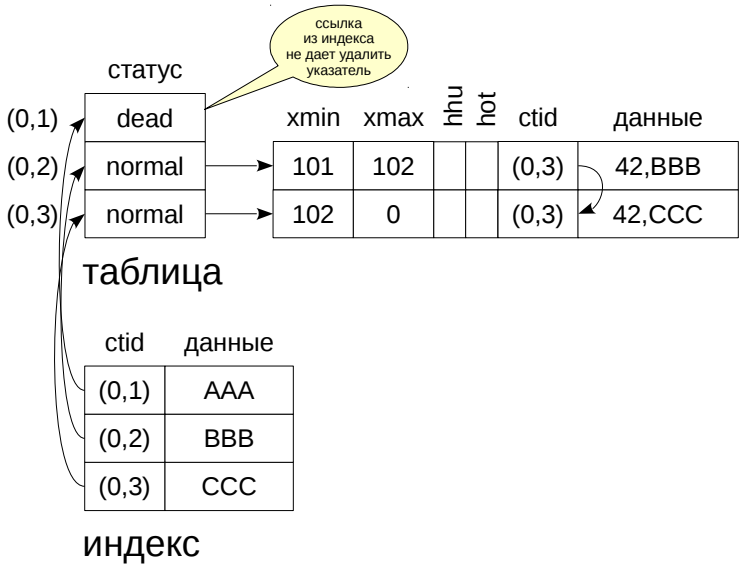
Тот факт, что страница может очищаться при чтении, означает, что запрос select может вызвать изменение данных. Это еще один такой случай, в дополнение к рассмотренному в теме «Страницы и версии строк» изменению битов-подсказок.

Внутристраничная очистка



На рисунке приведена ситуация, в которой попытка изменить версию строки (0,2) не удается, так как в табличной странице не хватает свободного пространства.

Внутристраничная очистка



В таком случае может быть выполнена внутристраничная очистка. Допустим, версия строки (0,1) уже не видна ни в одном снимке. В таком случае версия строки удаляется, указатель получает статус «dead», а освободившееся место может быть использовано для вставки новой версии.

Сам указатель удалить нельзя, поскольку на него существует ссылка из индексной страницы: при индексном доступе PostgrSQL может получить (0,1) в качестве идентификатора версии строки, попробует получить саму строку из табличной станицы, но благодаря статусу указателя обнаружит, что эта версия уже не существует.

Но указатель занимает всего 4 байта, так что его удаление в любом случае не освободит много места. Принципиальнее то, что внутристраничная очистка работает только в пределах одной табличной страницы и не очищает индексные страницы.

Внутристраничная очистка

HOT-обновления

- элемент индекса ссылается на один указатель
- внутри страницы поддерживается цепочка обновлений

Особенности очистки

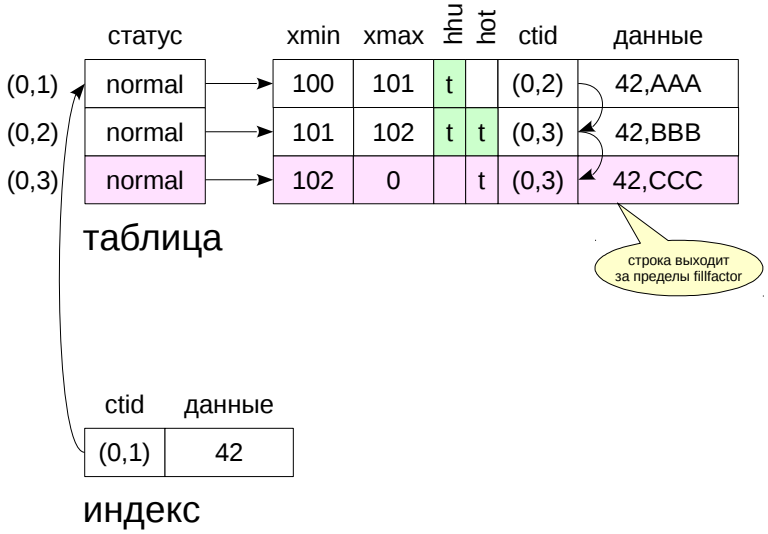
- указатель, на который ссылается индекс, должен остаться активным, даже если соответствующая версия строки больше не нужна
- поэтому вместо статуса «dead» используется «redirect» — перенаправление

Частный, но важный случай внутристраничной очистки представляет собой очистка при HOT-обновлениях (см. тему «Страницы и версии строк»).

HOT-обновления возникают при изменении в строке только не индексированных столбцов. При этом в индексе будет только одна ссылка на строку, независимо от того, сколько версий этой строки существует на странице. Версии строки, на которые нет ссылок извне, называются «heap-only tuples».

Таким образом при любых изменениях строки, указатель должен оставаться на своем месте и указывать на самую старую версию строки на странице (остальные можно получить по цепочке ссылок). Поэтому в данном случае применяется двойная адресация: для указателя используется статус «redirect», перенаправляющий на другой указатель, который уже и ведет на нужную версию строки.

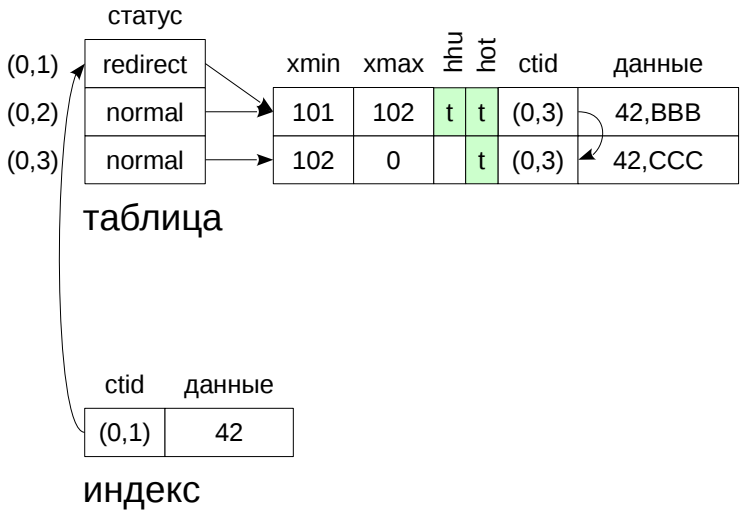
HOT-обновление



На рисунке приведен пример HOT-обновления. Изначальная версия строки (0,1) больше не видна ни в одном снимке, но именно на этот указатель ссылается индекс. Версия (0,2) не имеет на себя ссылок извне (признак heap-only tuple).

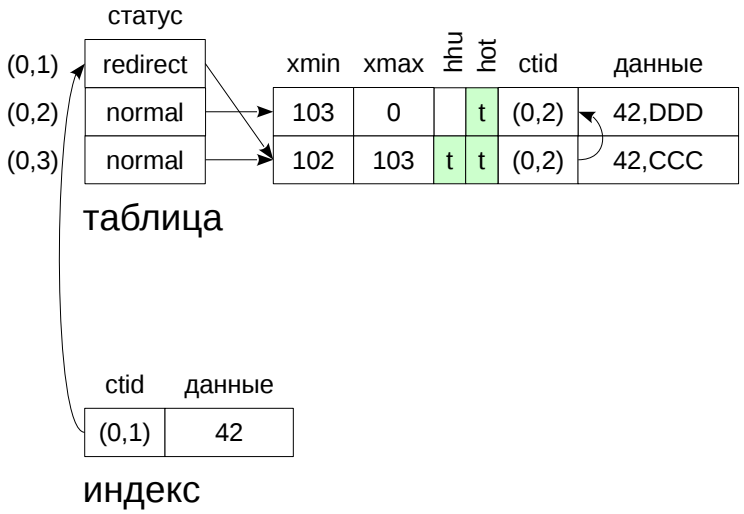
При попытке изменения обнаруживается, что на странице недостаточно места для еще одной версии строки. В такой ситуации возможно два варианта. Во-первых, новая версия строки может быть размещена на другой странице, но при этом цепочка HOT-обновлений будет прервана. Во-вторых, может помочь внутристраничная очистка.

HOT-обновление



В данном случае версия (0,1) может быть удалена, что освобождает место под новую версию (0,3). Первый указатель не может быть поставлен в статус «dead», как это было при обычном обновлении, так как ссылка из индекса должна оставаться активной. Поэтому используется перенаправление с помощью статуса «redirect».

HOT-обновление



Если снова изменить строку, будет удалена версия (0,2), и ее место займет новая версия. Указатель будет перенаправлен на версию (0,3), которая теперь является началом цепочки.

Ручная очистка

Выполняется при ручном запуске VACUUM

- для отдельных страниц таблицы и соответствующих индексов
- не рассматривает страницы, отмеченные в карте видимости

Очищает старые версии строк

- еще не прошедшие внутристраничную очистку

Очищает указатели

- так как одновременно очищает и индексные страницы

Обновляет карту свободного пространства

- освобождается место и для обновлений, и для новых строк

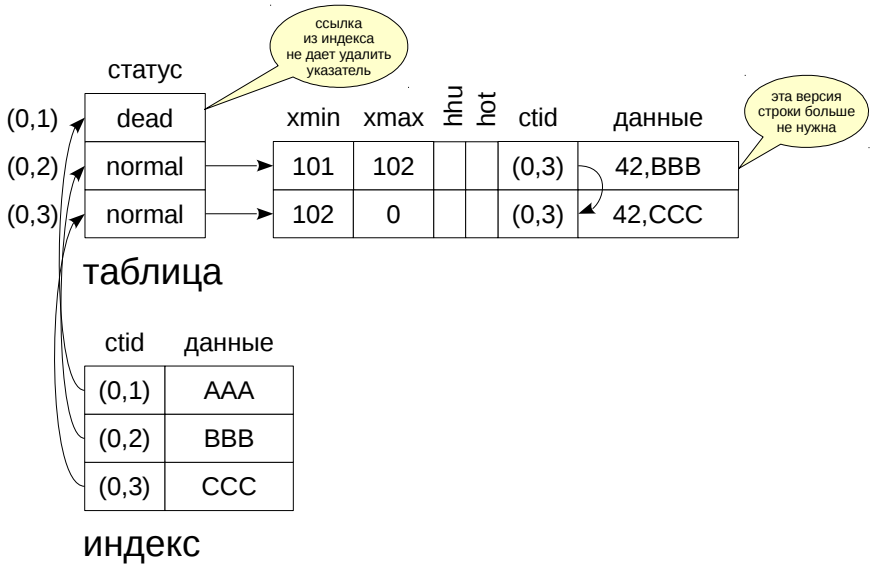
Обновляет карту видимости

Внутристраничная очистка выполняется быстро, но не решает всех задач. Она работает только в пределах одной табличной страницы и не затрагивает индексны.

Процесс очистки (vacuum) обрабатывает таблицу полностью, включая все созданные на ней индексы, освобождая место за счет удаления и ненужных версий строк, и указателей. Обработка происходит в фоновом режиме, таблица при этом может использоваться обычным образом.

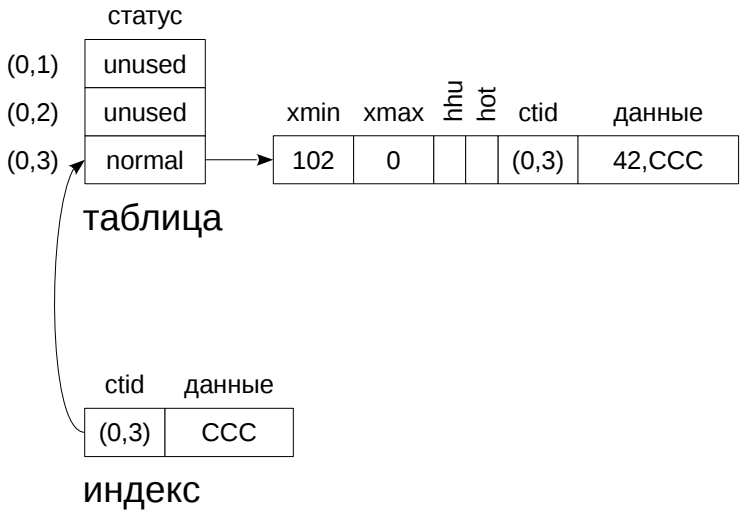
Из таблицы читаются только те страницы, в которых происходила какая-то активность. Для этого используется карта видимости. В ней отмечены те страницы, которые содержат только достаточно старые версии строк, которые гарантированно видимы во всех снимках. Таким образом, обрабатываются только страницы, не отмеченные в карте (а карта при этом обновляется).

Ручная очистка



На рисунке представлена ситуация, в которой первый указатель был помечен статусом «dead» при внутристраничной очистке, а версия (0,2) больше не нужна, но внутристраничная очистка до нее еще не добралась.

Ручная очистка



После очистки первый и второй указатели получают статус «unused», версия строки (0,2) удаляется. Одновременно в индексном блоке удаляются ссылки на первый и второй указатели.

Регулирование нагрузки

Процесс чередует работу и ожидание

примерно *vacuum_cost_limit* условных единиц работы,
затем засыпает на *vacuum_cost_delay* мс

Настройки

vacuum_cost_limit = 200
vacuum_cost_delay = 0 ms

стоимость обработки

vacuum_cost_page_hit = 1	*страницы в кэше*
vacuum_cost_page_miss = 10	*страницы на диске*
vacuum_cost_page_dirty = 20	*грязной страницы*

Процесс очистки интенсивно работает с таблицами и индексами и может вызвать серьезную дополнительную нагрузку на систему. Может оказаться удобным распределить процесс очистки по времени, чтобы сгладить пик нагрузки. Для этого существует возможность выполнять очистку порциями, чередуя работу и ожидание. Размер порции определяется исходя из настраиваемых оценок (а не по реально проделанной работе).

Настройка по умолчанию фактически отключает этот механизм, так как время ожидания выставлено в ноль.

http://www.postgresql.org/docs/current/static/runtime-config-resource.html

Полная очистка

Выполняется при ручном запуске VACUUM FULL

Полностью перестраивает таблицу

- перестраивает также все индексы на таблице
- освобожденное место возвращается операционной системе
- выполняется дольше, чем обычная очистка

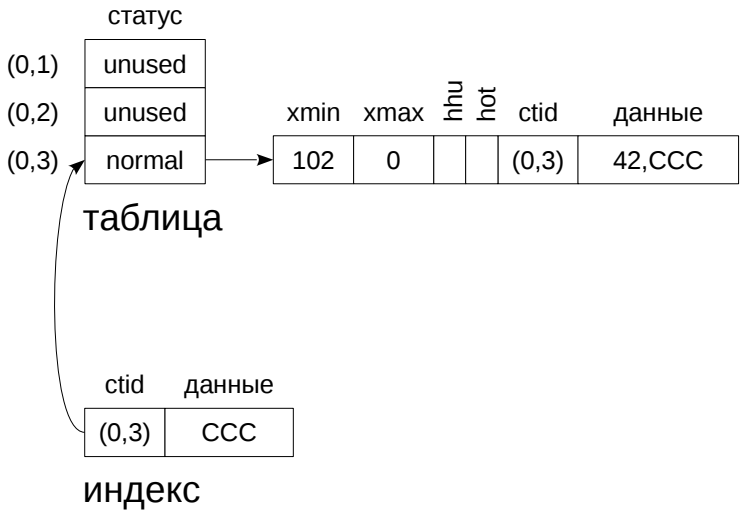
Требует эксклюзивной блокировки

Обычная очистка также не решает всех задач по освобождению места. Если таблица или индекс сильно выросли в размерах, то очистка не приведет к сокращению чиста страниц (и к уменьшению файлов). Вместо этого внутри существующих страниц появятся «дыры», которые могут быть использованы для вставки новых строк или изменения существующих. Единственно исключение составляют полностью очищенные страницы, находящиеся в конце файла.

Если размер файлов превышает некие разумные пределы, может быть выполнена полная очистка. При этом таблица и соответствующие индексы перестраиваются полностью с нуля, а данные будут упакованы максимально компактно (разумеется, с учетом fillfactor).

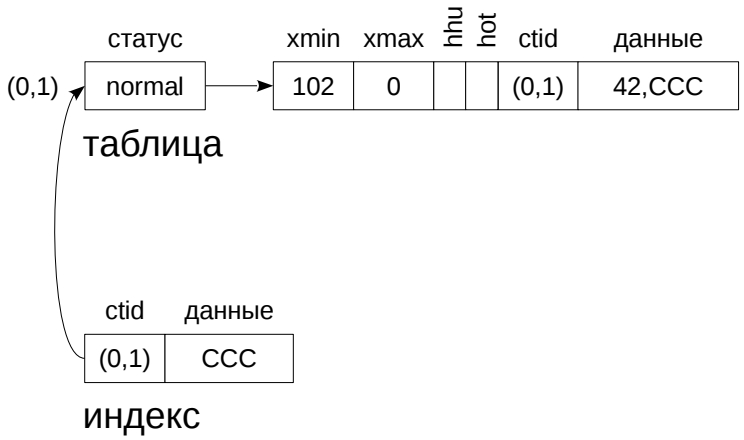
Тем не менее полная очистка не предполагает регулярного использования, так как выполняется дольше обычной и требует эксклюзивную блокировку на таблицу. Это означает, что сама очистка будет выполнена после завершения всех запросов к таблице, а запросы, отправленные после запуска полной очистки, приступят к работе только после ее завершения. На активно используемой системе это может быть неприемлемым; в таком случае может помочь расширение pg_repack, не входящее в поставку.

Полная очистка



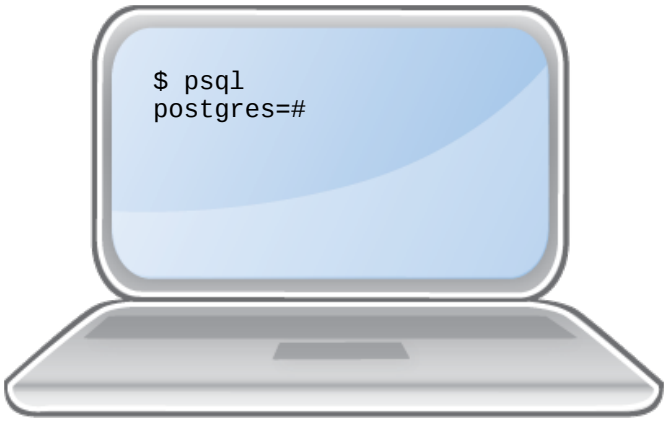
На рисунке показана ситуация, возникшая после выполнения обычной очистки.

Полная очистка



После выполнения полной очистки содержимое таблицы и индекса полностью перестроено; указатели со статусом «unused» исчезли.

Демонстрация



Итоги

Многоверсионность требует очистки ненужных версий

При обращении к странице может происходить внутристраничная очистка

При частых обновлениях полезно уменьшить fillfactor

При ручной очистке (vacuum) освобождается место во всех страницах таблиц и индексов

Полная очистка (vacuum full) полностью перестраивает таблицы и индексы

Практика

1. В базе данных DB5 создать таблицу.
2. Воспроизвести ситуацию HOT-обновления. Проконтролировать содержимое страницы с помощью pageinspect.
3. Воспроизвести ситуацию внутристраничной очистки при HOT-обновлении.

4*. Воспроизвести ситуацию HOT-обновления, при которой внутристраничная очистка не освобождает достаточно места и новая версия помещается на другую страницу.