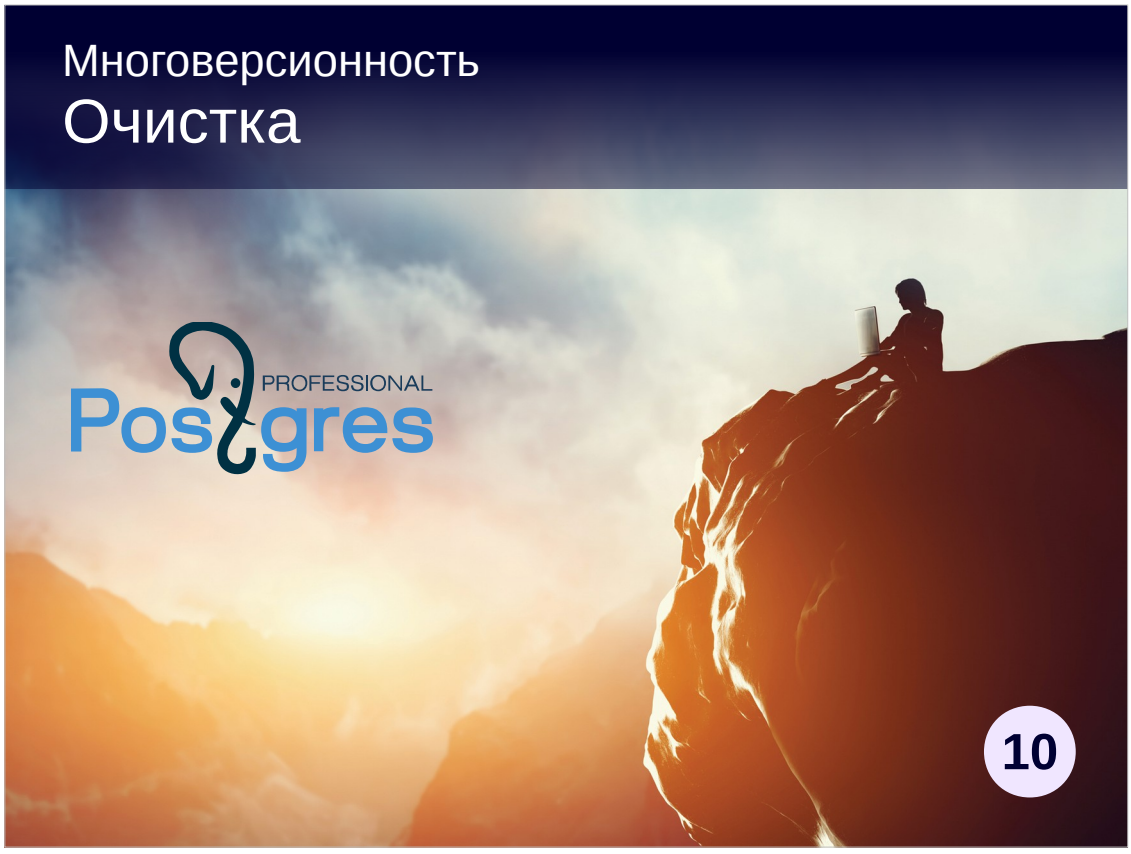


Авторы: Егор Рогов, Павел Лузанов

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

Темы

Обычная очистка

Полная очистка

Обычная очистка

VACUUM

Схема работы очистки

Регулирование нагрузки

Анализ

Обычная очистка

Выполняется командой VACUUM

- не конфликтует с обычной активностью в системе

Обрабатывает таблицу и все ее индексы

- очищает ненужные версии строк в табличных страницах (пропуская страницы, уже отмеченные в карте видимости)
- очищает индексные записи, ссылающиеся на очищенные версии строк
- освобождает указатели
- обновляет карту свободного пространства
- обновляет карту видимости

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

Очистка, выполняемая командой 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 и для чтения, и для изменения (однако одновременное выполнение для таблицы таких команд, как CREATE INDEX, ALTER TABLE и др. будет невозможно — подробнее см. модуль «Блокировки»).

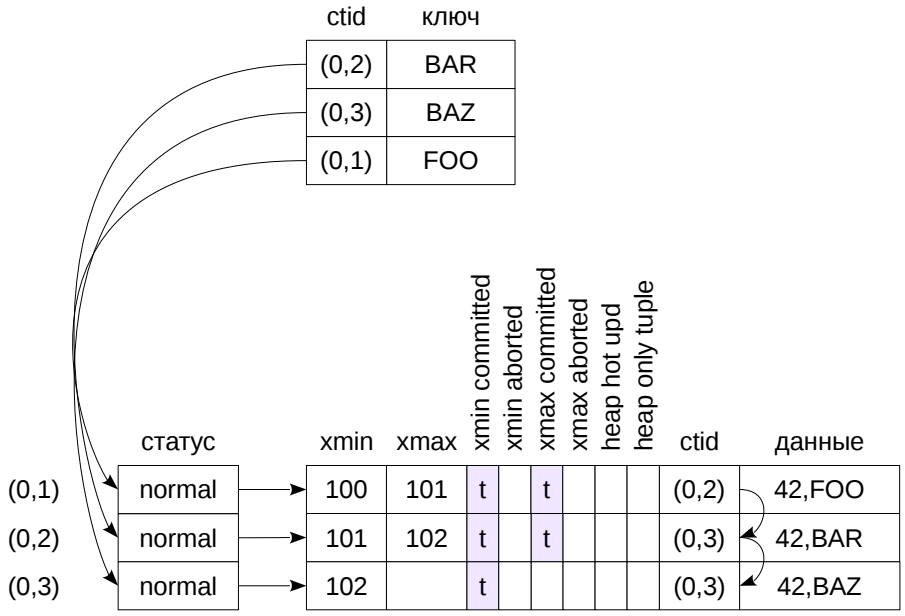
Из таблицы читаются только те страницы, в которых происходила какая-то активность. Для этого используется карта видимости (visibility map), в которой отмечены страницы, содержащие только достаточно старые версии строк, которые гарантированно видимы во всех снимках. Обрабатываются только страницы, не отмеченные в карте, а сама карта при этом обновляется.

В процессе работы обновляется и карта свободного пространства (free space map, в которой отражается наличие свободного места в страницах.

https://postgrespro.ru/docs/postgresql/10/sql-vacuum

https://postgrespro.ru/docs/postgresql/10/routine-vacuuming

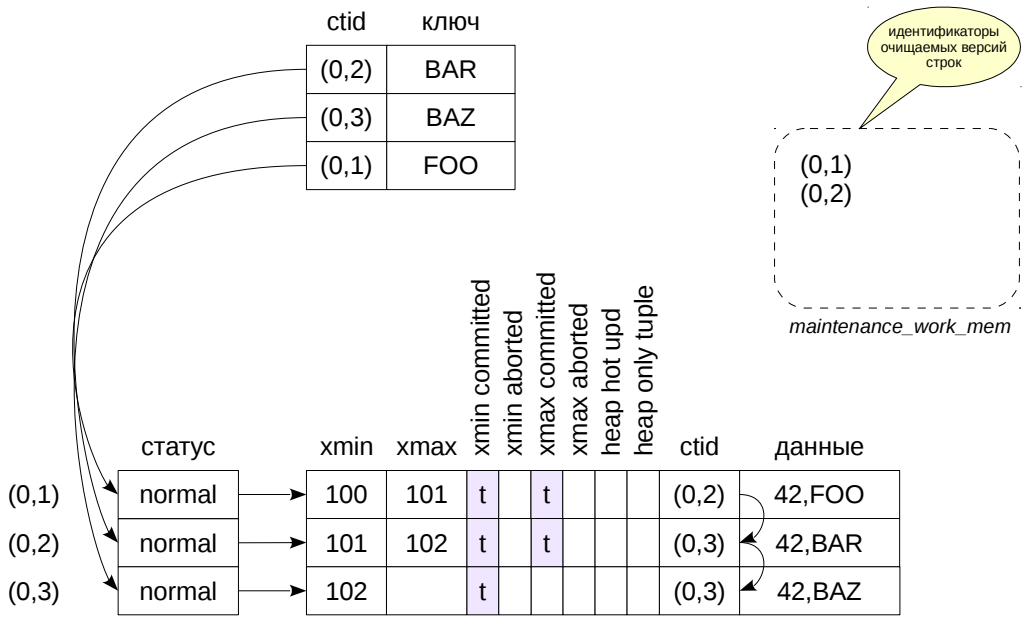
До очистки



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.

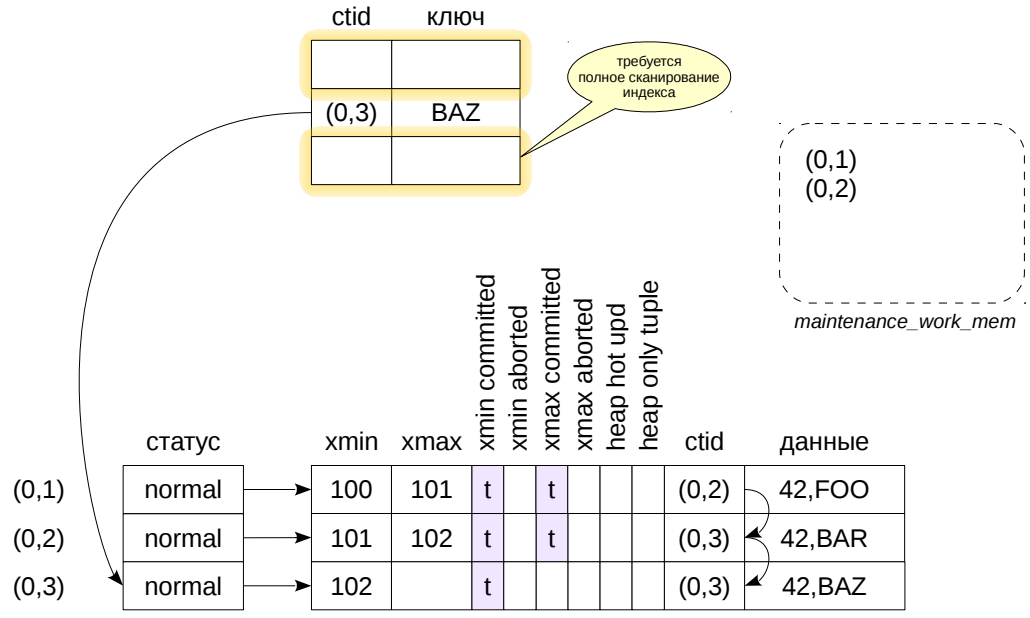
1. Чтение таблицы



Сначала VACUUM выполняет сканирование таблицы (пропуская те страницы, которые помечены в карте видимости).

В прочитанных страницах процесс определяет ненужные версии строк и записывает их идентификаторы в специальный массив. Для этого в локальной памяти процесса VACUUM выделяется фрагмент размером *maintenance_work_mem* (или меньше, если таблица небольшая). Значение этого параметра по умолчанию — 64 МБ. Отметим, что это память выделяется сразу в полном объеме, а не по мере необходимости.

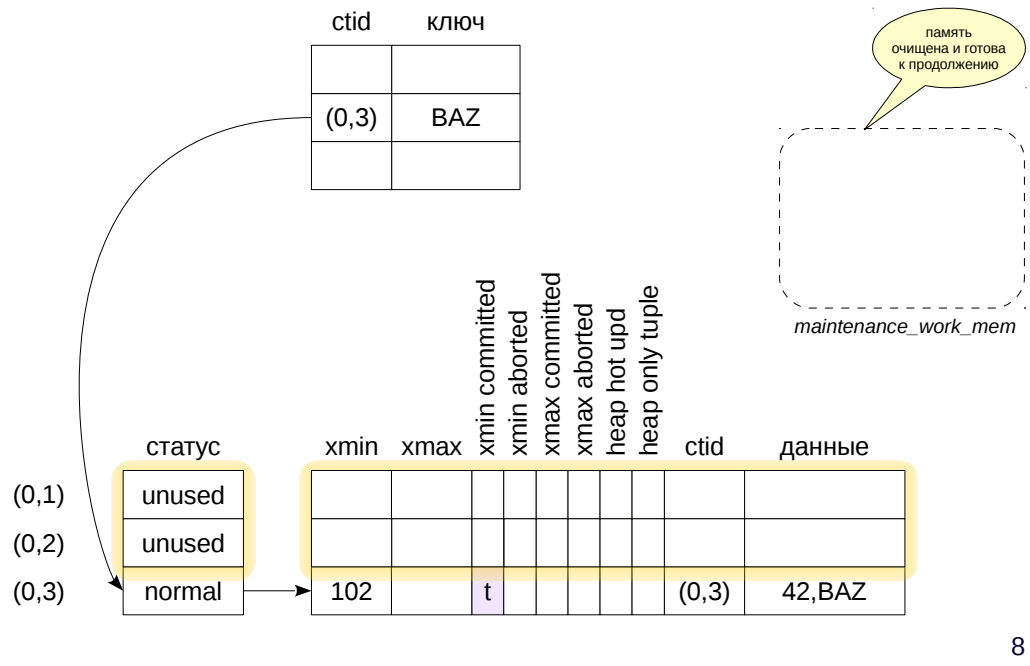
2. Очистка индексов



Когда выделенная под массив память заканчивается (или если мы уже дошли до конца таблицы), выполняется очистка индексов.

Для этого *каждый* из индексов, созданных на таблице, *полностью сканируется* в поисках записей, которые ссылаются на очищаемые версии строк. Найденные записи очищаются из индексных страниц.

3. Очистка таблицы



После этого необходимые табличные страницы повторно читаются, чтобы очистить в них ненужные версии строк и освободить указатели, на которые больше нет ссылок из индексов.

Если на первом проходе таблица не была просканирована полностью, то VACUUM очищает массив в памяти и продолжает работу с этой точки.

Таким образом, если при очистке удаляется так много версий строк, что все они не помещаются в память размером *maintenance_work_mem,* то все индексы будут полностью сканироваться несколько раз. На больших таблицах это может занимать существенное время и создавать существенную нагрузку на систему. Чтобы ускорить процесс, имеет смысл либо вызывать очистку чаще (чтобы за каждый раз очищалось не очень большое количество версий строк), либо выделить больше памяти.

Мониторинг

VACUUM VERBOSE

Представление pg_stat_progress_vacuum

- полный размер таблицы
- число прочитанных страниц и число очищенных страниц
- количество уже завершенных циклов очистки индексов
- число идентификаторов версий строк, помещающихся в память, и текущее число идентификаторов в памяти
- текущая фаза очистки

Если очистка выполняется долго, может потребоваться узнать текущее состояние дел.

Для этого можно вызывать VACUUM с указанием VERBOSE — на консоль будет выводиться информация о ходе выполнения.

Кроме того, начиная с версии 9.6 имеется представление pg_stat_progress_vacuum, которое также содержит всю необходимую информацию. Процесс очистки таблицы можно отслеживать, сравнивая число очищенных страниц (heap_blks_vacuumed) с общим количеством (heap_blks_total). Ход очистки индексов детально не отображается, но основное внимание надо обращать на количество циклов очистки (index_vacuum_count) — значение больше 1 означает, что памяти *maintenance_work_mem* не хватило для того, чтобы завершить очистку за один проход.

https://postgrespro.ru/docs/postgresql/10/progress-reporting

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

Поскольку процесс очистки интенсивно работает с таблицами и индексами, может оказаться необходимым распределить действия во времени и тем самым сгладить пики нагрузки. Для этого существует возможность выполнять очистку порциями, чередуя работу и ожидание.

Размер порции *vacuum_cost_limit* указывается в условных единицах. Даже это число является примерным, поскольку VACUUM определяет, сколько страниц обрабатывать, исходя из своих оценок, а не по реально проделанной работе.

Настройка по умолчанию фактически отключает этот механизм, так как время ожидания выставлено в ноль. Так сделано из тех соображений, что если администратору пришлось запускать VACUUM вручную, он, скорее всего, хочет выполнить очистку как можно быстрее.

https://postgrespro.ru/docs/postgresql/10/runtime-config-resource#RUNTIME-CONFIG-RESOURCE-VACUUM-COST

Анализ

Выполняется при VACUUM ANALYZE или ANALYZE

- не конфликтует с обычной активностью в системе

Собирает статистику для планировщика

Еще одна задача, которую обычно совмещают с очисткой, — анализ, то есть сбор статистической информации для планировщика запросов. Анализируется число строк в таблице, распределение данных по столбцам и т. п. Более подробно это рассматривается в курсе QPT-10.

Вручную анализ выполняется командой ANALYZE (только анализ) или VACUUM ANALYZE (и очистка, и анализ).

Как и с обычной очисткой, обработка происходит в фоновом режиме и не мешает обычному использованию таблиц и индексов.

https://postgrespro.ru/docs/postgresql/10/sql-analyze

Полная очистка

VACUUM FULL

Команды, схожие по механизму работы

Полная очистка

Выполняется командой VACUUM FULL

- не совместима ни с какими операциями над таблицей, включая чтение

Полностью перестраивает таблицу и все ее индексы

- при работе потребуется дополнительное место для новых файлов
- освобожденное место возвращается операционной системе
- выполняется дольше, чем обычная очистка

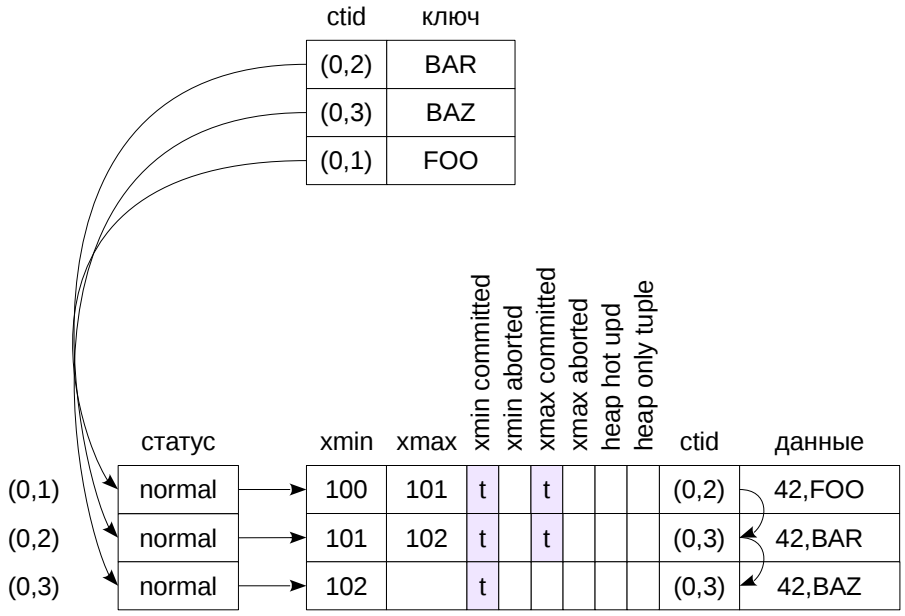
Обычная очистка не решает всех задач по освобождению места. Если таблица или индекс сильно выросли в размерах, то очистка не приведет к сокращению числа страниц (и к уменьшению файлов). Вместо этого внутри существующих страниц появятся «дыры», которые могут быть использованы для вставки новых строк или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

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

При перестройке PostgreSQL последовательно перестраивает сначала таблицу, а затем и каждый из индексов. Для них создаются новые файлы, а старые удаляются. Следует учитывать, что в процессе работы на диске потребуется дополнительное место.

Полная очистка не предполагает регулярного использования, так как полностью блокирует всякую работу с таблицей (включая и выполнение запросов к ней) на все время своей работы. На активно используемой системе это может быть неприемлемым; в таком случае может помочь расширение pg_repack (https://github.com/reorg/pg_repack), не входящее в поставку.

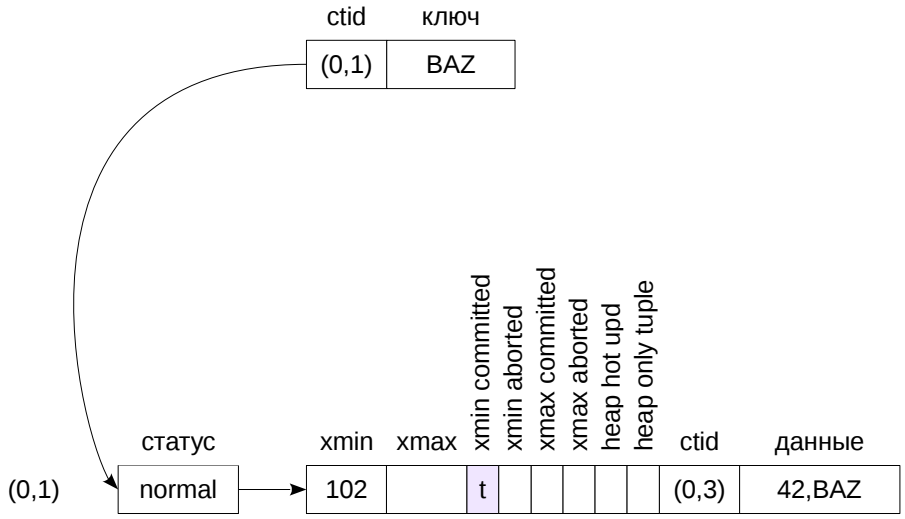
До полной очистки



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 полной очистки



После выполнения полной очистки содержимое таблицы и индекса полностью перестроено и физически находится в других файлах. При этом OID таблицы в системном каталоге сохраняется.

Похожие команды

CLUSTER

- полностью перестраивает таблицу и все ее индексы
- дополнительно физически упорядочивает версии строк в соответствии с одним из индексов

REINDEX

- полностью перестраивает отдельный индекс

TRUNCATE

- «опустошает» таблицу

Особенности

- все команды полностью блокируют работу с таблицей
- все команды создают новые файлы для данных

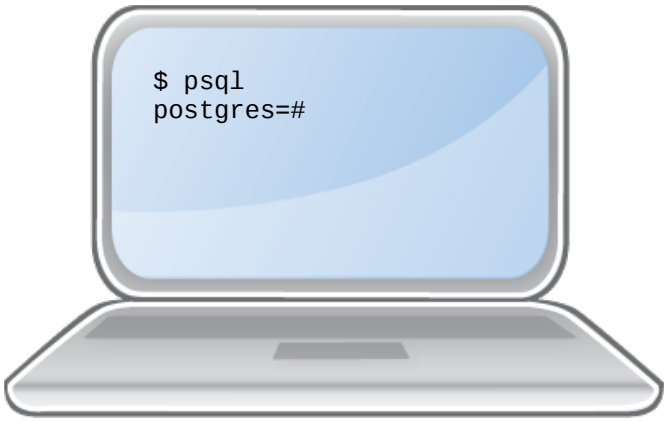
Есть несколько команд, которые работают, используя схожий механизм. Все они полностью блокируют работу с таблицей, все они удаляют старые файлы данных и создают новые.

Команда CLUSTER во всем аналогична VACUUM FULL, но дополнительно физически упорядочивает версии строк в соответствии с одним из индексов. Это дает планировщику возможность более эффективно использовать индексный доступ в некоторых случаях. Однако надо понимать, что кластеризация не поддерживается: при последующих изменениях таблицы физический порядок версий строк будет нарушаться.

Команда REINDEX перестраивает отдельный индекс на таблице. Фактически, VACUUM FULL и CLUSTER используют эту команду для того, чтобы перестроить индексы.

Команда TRUNCATE логически работает так же, как и DELETE — удаляет все табличные строки. Но DELETE, как уже было рассмотрено, помечает версии строк как удаленные, что требует дальнейшей очистки. TRUNCATE же просто создает новый, чистый файл. Это работает быстрее, но надо учитывать, что TRUNCATE заблокирует работу с таблицей на все время до конца транзакции.

Демонстрация



Итоги

Обычная очистка освобождает место в страницах

- выполняется в фоновом режиме
- может потребоваться несколько раз сканировать индексы

Полная очистка перестраивает таблицу и индексы

- блокирует работу с таблицей на все время работы

Практика

1. Создайте большую таблицу с индексом. Временно уменьшите значение *maintenance_work_mem* так, чтобы потребовалось несколько проходов для очистки индекса. Проконтролируйте, запуская VACUUM VERBOSE.
2. Удалите из большой таблицы 90 % случайных строк и проверьте, как изменился объем, который таблица занимает на диске, после выполнения обычной очистки.
3. Повторите п. 2 с полной очисткой.

1. Чтобы исключить срабатывание автоматической очистки (рассматривается в следующей теме), при создании таблицы укажите параметр хранения autovacuum_enabled:

```
CREATE TABLE ... WITH (autovacuum_enabled = off);
```