

ChatGPT - SQL Expert (QueryGPT)

 chatgpt.com/g/g-m5lMeGif-sql-expert-querygpt/c/6830a3c4-0944-8006-b7dd-5677fb950e86



[postgres-A4.pdf](#)

[PDF](#)

Что такое "мертвые строки", причины их возникновения с указанием номеров страниц.

В PostgreSQL термин "мёртвые строки" относится к строкам таблиц, которые были удалены или обновлены, но физически всё ещё находятся в таблице. Это результат механизма MVCC (многоверсионного управления конкурентным доступом), который сохраняет старые версии строк для обеспечения изоляции транзакций.

Причины возникновения мёртвых строк:

1. **Обновления:** При обновлении строки PostgreSQL не изменяет её на месте, а создаёт новую версию строки, помечая старую как мёртвую.
2. **Удаления:** При удалении строки она также просто помечается как мёртвая, но физически остаётся в таблице.
3. **НОТ-оптимизация** (Heap-Only Tuples): применяется, если изменение не затрагивает индексируемые столбцы и на той же странице достаточно места для новой версии. Это помогает уменьшить количество индексаций и сократить число мёртвых строк, но не устраняет их полностью.

Для чего нужно удаление мёртвых строк:

PostgreSQL использует команду **VACUUM**, чтобы очищать мёртвые строки, освобождать занимаемое ими место и поддерживать актуальность статистики таблиц. Это особенно важно для предотвращения "раздувания" таблиц и потери производительности запросов.

Полезные страницы:

- Раздел о HOT: стр. 2497
- Принципы очистки и роль мёртвых строк: стр. 2137 и далее

Follow GPT creator profile on LinkedIn <https://linkedin.com/in/khanukov/>

В каких случаях VACUUM НЕ удаляет мертвые строки?

Команда **VACUUM** в PostgreSQL не удаляет мёртвые строки в следующих случаях:

1. **Ограничения прав доступа** – VACUUM не обрабатывает таблицы, на которые у пользователя нет прав **MAINTAIN** или если пользователь не является владельцем базы данных.
2. **Конфликтующие блокировки** – при использовании параметра **SKIP_LOCKED**, VACUUM пропускает таблицы (или секции таблиц), если не удаётся немедленно получить блокировку. Это особенно актуально для секционированных таблиц, где VACUUM может пропустить все секции.
3. **Отключённая или ограниченная очистка индексов** – если установлено **INDEX_CLEANUP = OFF**, VACUUM не удаляет мёртвые кортежи из индексов. При **INDEX_CLEANUP = AUTO** (по умолчанию) индексы могут быть пропущены, если выгода от очистки мала.
4. **Страницы без мёртвых строк** – VACUUM использует карту видимости и может пропускать страницы, на которых, по его мнению, нет мёртвых строк, даже если там находятся старые версии строк. Это означает, что VACUUM не всегда замораживает все старые версии, особенно без агрессивной стратегии очистки.
5. **Неактуальные настройки автоочистки** – если автоочистка (autovacuum) отключена или неправильно настроена, VACUUM может не запускаться для нужных таблиц или запускаться слишком редко, не удаляя мёртвые строки вовремя.
6. **Недостаточная активность** – VACUUM может не удалять мёртвые строки, если их количество мало и их удаление нецелесообразно с точки зрения производительности, особенно при автоочистке.

Эти ограничения могут приводить к накоплению мёртвых строк и, как следствие, к ухудшению производительности, если не принять меры (например, ручной запуск VACUUM FULL или настройка параметров автоочистки).

Follow GPT creator profile on LinkedIn <https://linkedin.com/in/khanukov/>

Какая связь с наличием мёртвых строк и проблемами с транзакциями?

Наличие мёртвых строк в PostgreSQL напрямую связано с системой управления транзакциями (MVCC) и может вызывать следующие проблемы:

1. Повышенная нагрузка на систему MVCC

- Каждая транзакция должна проверять видимость строк, используя метаданные транзакций (XMIN/XMAX).
- Чем больше мёртвых строк, тем больше времени тратится на проверки, что снижает производительность чтения.

2. Рост времени выполнения запросов

- Планировщик запросов может переоценивать количество строк в таблице из-за большого числа мёртвых строк.
- Это может привести к неэффективному выбору плана запроса (например, использование последовательного сканирования вместо индекса).

3. Повышенное потребление ресурсов

Хранение большого количества мёртвых строк увеличивает использование памяти, дискового пространства и кэша.

4. Замедление транзакций

- При длительных транзакциях мёртвые строки не могут быть удалены, так как VACUUM должен убедиться, что никакая активная транзакция не нуждается в этих версиях строк.
- Это может блокировать очистку и вызывать «раздутие» таблицы (table bloat).

5. Риск истечения возраста транзакции (transaction wraparound)

PostgreSQL использует 32-битные идентификаторы транзакций. Если мёртвые строки не очищаются, их XMIN может блокировать использование новых XID, что в критических случаях требует срочной заморозки (**FREEZE**) таблиц.

Практический пример:

Если в системе есть долгоживущая транзакция, она «замораживает» минимальный XID, и VACUUM не может очистить мёртвые строки, даже если они больше никому не видны. Это приводит к ухудшению производительности всей базы данных.

Таким образом, мёртвые строки не только следствие транзакционной модели, но и активный фактор, влияющий на надёжность и производительность системы транзакций.

Follow GPT creator profile on LinkedIn <https://linkedin.com/in/khanukov/>