

Использование TOAST в PostgreSQL

 chatgpt.com/c/68303b9a-d360-8006-8038-5e050f40ce0d



```
SELECT width_bucket(pg_column_size(attributes), ARRAY[5, 256, 512, 1024, 2048, 4096, 8192, 16384]) AS size_bucket, count(*) FROM messages GROUP BY 1 ORDER BY 1;
```

показало: size_bucket;count 1;144091374 2;31502 3;2 4;2 какой вывод использования TOAST

Ваш SQL-запрос анализирует распределение размеров значения столбца `attributes` таблицы `messages`, используя `pg_column_size()` и функцию `width_bucket`, чтобы сгруппировать строки по размеру. Это полезный способ понять, как PostgreSQL хранит данные и используется ли механизм TOAST (The Oversized-Attribute Storage Technique).

Результаты:

size_bucket	count
1	144,091,374
2	31,502
3	2
4	2

Выводы о TOAST:

1. TOAST почти не используется

- Почти все значения (~144 млн) попадают в первую корзину — это `pg_column_size(attributes) <= 5 байт`. Это очень маленькие значения, которые хранятся **inline** прямо в основной строке (внутри 8 KB страницы).
- Следующая корзина (5–256 байт) содержит всего 31 тысячу строк. Это тоже значения, хранящиеся **inline**, но чуть больше по размеру.
- TOAST начинает применяться, когда значение превышает **2 KB** (по умолчанию). В этом наборе данных видно, что:
 - Лишь **четыре строки** имеют размеры, выходящие за пределы 512 байт.
 - Это говорит о том, что **почти никакие строки не превышают порог TOAST** и PostgreSQL не использует TOAST-хранилище.

2. TOAST может быть включен, но не нужен

- PostgreSQL автоматически использует TOAST для хранения больших значений в специальных TOAST-таблицах (например, `pg_toast.pg_toast_<oid>`), но в вашем случае значения почти всегда малы.
- Возможно, тип `attributes` — это `json`, `jsonb`, `text` или `bytea`, но значения почти всегда короткие.

3. Оптимизация хранения

- Хороший признак: маленькие значения экономят место и ускоряют доступ.
- TOAST — это «медленное» хранилище (относительно), поэтому его избегание может положительно сказаться на производительности.

Резюме:

TOAST не используется в вашей таблице практически совсем, потому что почти все значения `attributes` — очень короткие. Это хорошо с точки зрения производительности и места на диске.

Если вы хотите проверить, используется ли TOAST в принципе, можно сделать:

sql

```
SELECT reltoastrelid
FROM pg_class
WHERE relname = 'messages';
```

Если `reltoastrelid` != 0, то TOAST-таблица существует, но это не означает, что она используется активно.

Хотите, я помогу проверить, сколько строк реально лежат в TOAST-таблице?

Да

Чтобы проверить, используется ли TOAST реально, можно сделать следующие шаги:

1. Найти имя TOAST-таблицы для **messages**

sql

```
SELECT c.oid::regclass AS table_name,  
       c.reltoastrelid::regclass AS toast_table  
FROM pg_class c  
WHERE c.relname = 'messages';
```

- Это покажет имя основной таблицы и её связанной TOAST-таблицы (если она есть).
- Если **reltoastrelid** показывает что-то вроде **pg_toast.pg_toast_123456**, значит TOAST-таблица существует.

2. Посчитать строки в TOAST-таблице

sql

```
SELECT COUNT(*) FROM pg_toast.pg_toast_123456;
```

- Замените **pg_toast_123456** на имя, найденное на предыдущем шаге.
- Это покажет **сколько строк реально вынесено в TOAST-хранилище**.

Пример всего вместе:

sql

```
-- Шаг 1: Найти TOAST-таблицуSELECT c.relname AS main_table,
      t.relname AS toast_table
FROM pg_class c
JOIN pg_class t ON c.reltoastrelid = t.oid
WHERE c.relname = 'messages';

-- Шаг 2: Подставить имя TOAST-таблицыSELECT COUNT(*) FROM pg_toast.pg_toast_123456; -- замените
на имя из шага 1
```

Если результат будет 0, это означает, что **TOAST-таблица есть, но не используется** — PostgreSQL создал её автоматически, но значения туда не попадали (все помещаются inline).

Если нужно, я помогу интерпретировать вывод — просто пришлите результат.

Нравится ли вам эта личность?