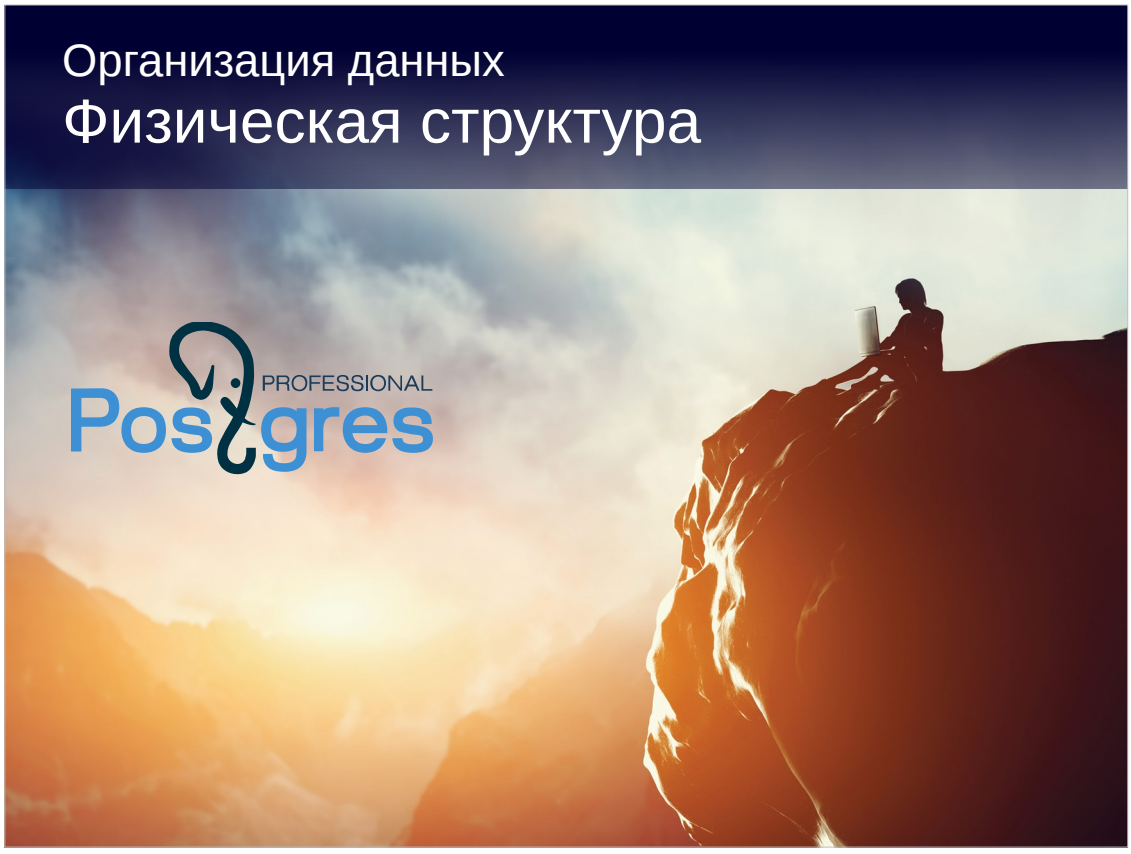


Авторы: Егор Рогов, Павел Лузанов

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

Темы

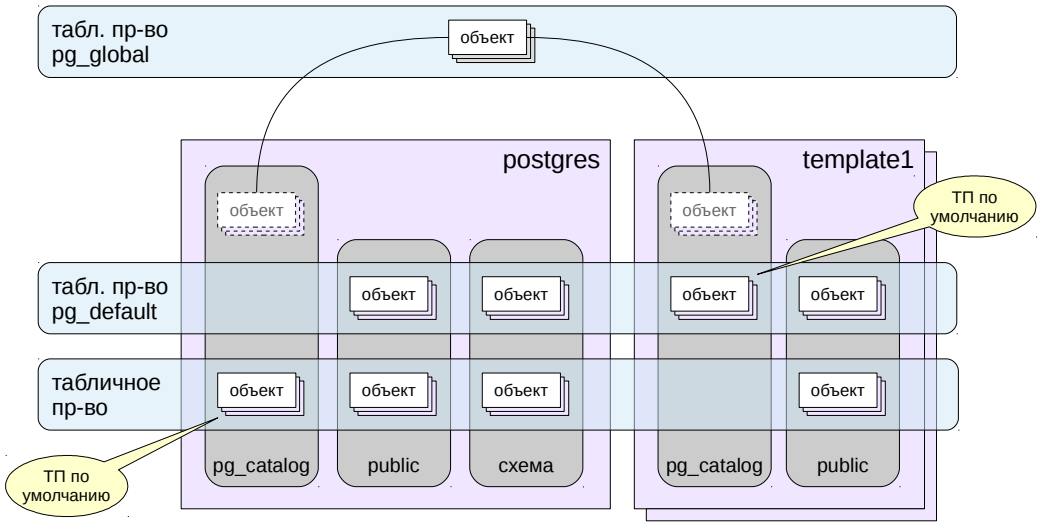
Табличные пространства и каталоги

Файлы и страницы данных

Слои: данные, карты видимости и свободного пространства

Технология TOAST

Табличные пространства



Табличные пространства (ТП) служат для организации физического хранения данных и определяют расположение данных в файловой системе.

Например, можно создать одно ТП на медленных дисках для архивных данных, а другое – на быстрых дисках для данных, с которыми идет активная работа.

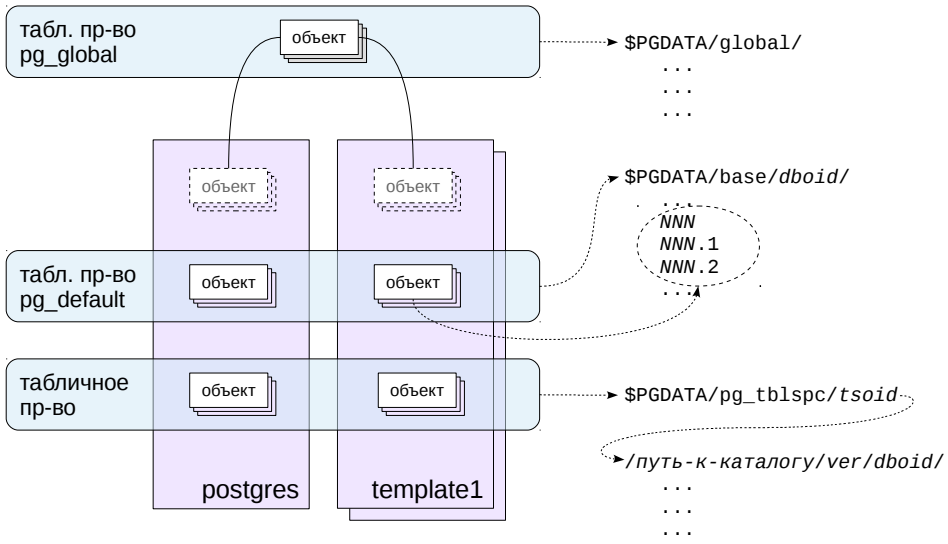
При инициализации кластера создаются два ТП: pg_default и pg_global.

Одно и то же ТП может использоваться разными базами данных, а одна база данных может хранить данные в нескольких ТП.

При этом у каждой БД есть так называемое «ТП по умолчанию», в котором создаются все объекты, если явно не указать иное. В этом же ТП хранятся и объекты системного каталога. Изначально в качестве «ТП по умолчанию» используется ТП pg_default, но можно установить и другое.

ТП pg_global особенное: в нем хранятся те объекты системного каталога, которые являются общими для кластера.

Каталоги, файлы, страницы



По сути, табличное пространство — это указание на каталог, в котором располагаются данные. Стандартные ТП pg_global и pg_default всегда находятся в $PGDATA/global/ и $PGDATA/base/ соответственно. При создании пользовательского ТП указывается произвольный каталог; для собственного удобства PostgreSQL создает на него символьную ссылку в каталоге $PGDATA/pg_tblspc/.

Внутри каталога $PGDATA/base/ данные дополнительно разложены по подкаталогам баз данных (для $PGDATA.global/ это не требуется, как так данные в нем относятся к кластеру в целом).

Внутри каталога пользовательского ТП появляется еще один уровень вложенности: версия сервера PostgreSQL. Это сделано для удобства обновления сервера на другую версию.

Собственно объекты хранятся в файлах внутри этих каталогов — каждый объект в отдельных файлах.

Каждый файл (называемый *сегментом*) занимает не более 1 ГБ, потому каждому объекту может соответствовать несколько файлов. Необходимо учитывать влияние потенциально большого количества файлов на используемую файловую систему.

Все файлы-сегменты разбиты на отдельные страницы, обычно по 8 КБ (размер можно установить для всего кластера только при сборке сервера). Страницы разных объектов считываются с диска совершенно однотипно через общий механизм буферного кэша.

https://postgrespro.ru/docs/postgresql/9.6/storage-file-layout.html

Слои

Основной

- собственно данные

Карта видимости (vm)

- отмечает страницы, на которых все версии строк видны во всех снимках
- используется для оптимизации работы процесса очистки и ускорения индексного доступа
- существует только для таблиц

Карта свободного пространства (fsm)

- отмечает свободное пространство в страницах после очистки
- используется при вставке новых версий строк

Обычно каждому объекту соответствует несколько слоев. Каждый слой — это набор сегментов (то есть файл или несколько файлов, размером не более 1 ГБ).

Основной слой — это собственно данные: версии строк таблиц или строки индексов.

Слой vm (visibility map) — битовая карта видимости. В ней отмечены страницы, которые содержат только актуальные версии строк, видимые во всех снимках данных. Иными словами, это страницы, которые давно не изменялись и успели полностью очиститься от неактуальных версий.

Карта видимости применяется для оптимизации очистки (отмеченные страницы не нуждаются в очистке) и для ускорения индексного доступа. Дело в том, что информация о версионности хранится только для таблиц, но не для индексов (поэтому у индексов не бывает карты видимости). Получив из индекса ссылку на версию строки, нужно прочитать табличную страницу, чтобы проверить ее видимость. Но если в самом индексе уже есть все нужные столбцы, и при этом страница отмечена в карте видимости, то к таблице можно не обращаться.

Слой fsm (free space map) — карта свободного пространства. В ней отмечено доступное место внутри страниц, образующееся, например, при работе очистки. Эта карта используется при вставке новых версий строк, чтобы быстро найти подходящую страницу.

TOAST

Версия строки должна помещаться на одну страницу

- можно сжать часть атрибутов,
- или вынести в отдельную TOAST-таблицу,
- или сжать и вынести одновременно

TOAST-таблица

- схема pg_toast
- поддержана собственным индексом
- «длинные» атрибуты разделены на части размером меньше страницы
- читается только при обращении к «длинному» атрибуту
- собственная версионность
- работает прозрачно для приложения

Любая версия строки в PostgreSQL должна целиком помещаться на одну страницу. Для «длинных» версий строк применяется технология TOAST — The Oversized Attributes Storage Technique. Она подразумевает несколько стратегий. Подходящий «длинный» атрибут может быть сжат так, чтобы версия строки поместилась на страницу. Если это не получается, версия строки может быть отправлена в отдельную служебную таблицу. Могут применяться и оба подхода.

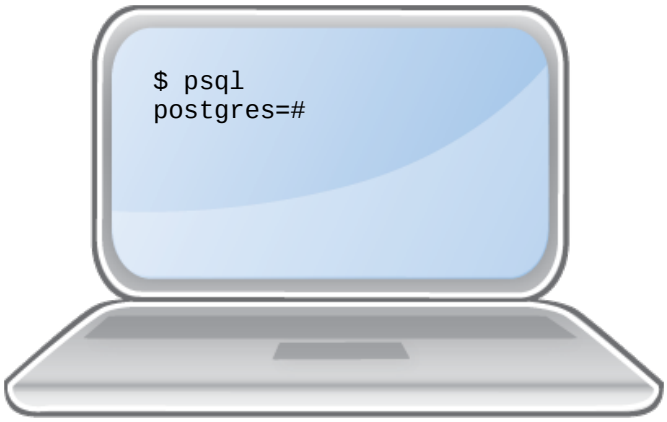
Для каждой основной таблицы при необходимости создается отдельная TOAST-таблица (и к ней специальный индекс). Такие таблицы и индексы располагаются в отдельной схеме pg_toast и поэтому обычно не видны.

Версии строк в TOAST-таблице тоже должны помещаться на одну страницу, поэтому «длинные» значения хранятся порезанными на части. Из этих частей PostgreSQL прозрачно для приложения «склеивает» необходимое значение.

TOAST-таблица используется только при обращении к «длинному» значению. Кроме того, для toast-таблицы поддерживается своя версионность: если обновление данных не затрагивает «длинное» значение, новая версия строки будет ссылаться на то же самое значение в TOAST-таблице — это экономит место.

https://postgrespro.ru/docs/postgresql/9.6/storage-toast.html

Демонстрация



Итоги

Физически

- данные распределены по табличным пространствам (каталогам)
- объект представлен несколькими слоями
- каждый слой состоит из одного или нескольких файлов-сегментов

Табличными пространствами управляет администратор

Слои, файлы, TOAST — внутренняя кухня PostgreSQL

Практика

1. Создайте новую базу данных и подключитесь к ней.
2. Создайте табличное пространство ts.
3. Создайте таблицу t в табличном пространстве ts и добавьте в нее несколько строк.
4. Вычислите объем, занимаемый базой данных, таблицей и табличными пространствами ts и pg_default.
5. Перенесите таблицу в табличное пространство pg_default.
6. Как изменился объем табличных пространств?
7. Удалите табличное пространство ts.