

Postgresql: нормализация, индексы, транзакции

онлайн-воркшоп с Кириллом Храпковым

День 2:
Индексы

ПРИВЕТ :)

Повторим, что уже сделали

Рефлексия

День 2:

- познакомимся с термином “индекс”
- разберемся с тем, что это такое, для чего нужен и как правильно его использовать
- узнаем какие типы бывают
- поделимся личным опытом

Индекс - это специальные структуры в базах данных, которые позволяют ускорить поиск и сортировку по определенному полю или набору полей.

Зачем нужен индекс:

1. сократить время поиска запрошенных данных
2. для ограничения уникальных значений в таблице
(напр. для Btree-индекса)

Основные способы сканирования:

1. последовательное сканирование
2. индексное сканирование
3. сканирование по битовой карте

Какие бывают индексы:

1. по одному полю
2. по нескольким полям (составной индекс)
3. по выражениям (функциональный индекс)
4. частичные индексы
5. покрывающие индексы

Типы индексов:

1. Hash
2. Btree
3. GiST
4. SP-GiST
5. GIN
6. BRIN

Hash-индексы:

являются одностолбцовыми индексами, хранящими 32-битные результаты, полученные из значения индексированного столбца. Хэш-значение сопоставляется с сегментом, в котором хранится указатель на строку в таблице кучи

Hash-индекс использует страницы четырех видов:

Метастраница (meta page) — нулевая страница, содержит информацию о том, что находится внутри индекса

Страницы корзин (bucket page) — основные страницы индекса, хранят данные в виде пар «хеш-код — TID»

Страницы переполнения (overflow page) — устроены так же, как страницы корзин, и используются в случае, когда одной страницы для корзины не хватает

Страницы битовой карты (bitmap page) — в них отмечаются освободившиеся страницы переполнения, которые можно использовать для других корзин

Ограничения хэш-индексов:

- поддерживают только сравнение на равенство при выполнении поиска
- поддерживаются только для ключа из единственного столбца
- не имеют возможности накладывать ограничение уникальности

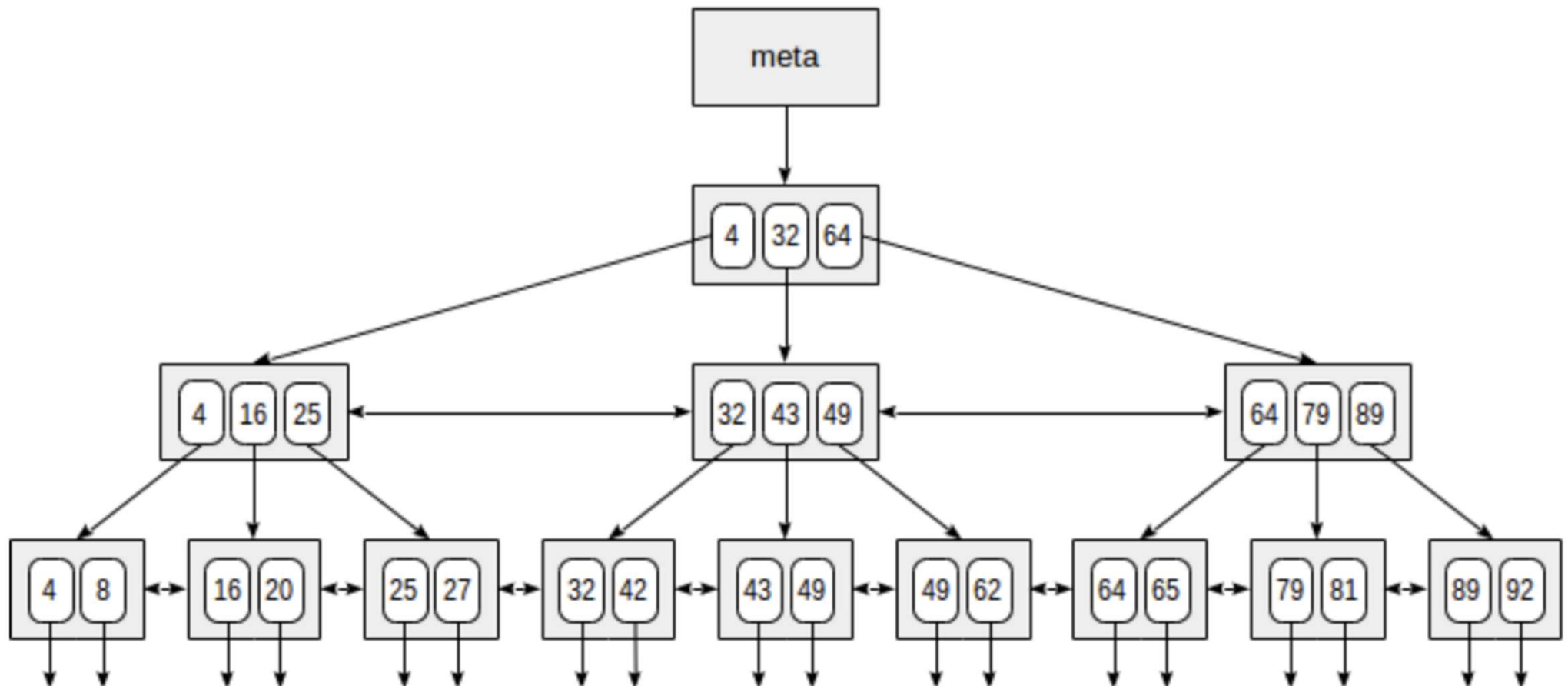
Преимущества хэш-индексов перед B-Tree:

- очень быстрое обновление
- скорость чтения почти в 3 раза выше
- размер ключа
- размер хэш-индекса не влияет, насколько селективным является значение ключа индекса

Btree-индексы

алгоритм индексирования, основанный на идее разделения данных на несколько секций, которые называются узлами дерева. Каждый узел содержит набор ключей и ссылок на дочерние узлы.

схематичный пример индекса по одному полю с целочисленными ключами



Преимущества B-Tree над Hash-индексами:

- работает с операторами сравнения `>`, `<`, `=`, `>=`, `<=`, `BETWEEN` и `IN`
- работает с `IS NULL` и `IS NOT NULL`
- работает с `LIKE` и `~`, если искомая строка закреплена в начале шаблона (например `name LIKE 'foo%'`);
- работает с регистронезависимыми операторами поиска подстроки `ILIKE` и `~*`. Но только в том случае, если искомая строка начинается с символа, который одинаков и в верхнем и в нижнем регистре (например числа)
- работает с сортировкой

Gin-индексы (*Generalized Inverted Index*)

применимы к составным типам, работа с которыми осуществляется с помощью ключей. Это массивы, jsonb и tsvector.

GiST-индексы (*Generalized Search Tree*)

для построения индексов использует один из нескольких алгоритмов, наиболее подходящих под тип индексируемого поля. Поэтому набор операторов, при работе с которыми может быть задействован этот индекс, зависит от типа поля. По умолчанию PostgreSQL предоставляет индексы для некоторых типов данных, таких как геометрические типы, сетевые адреса, диапазоны и т.д. Так же этот список можно расширить, установив соответствующие модули.

На базе GiST могут быть реализованы B-деревья, R-деревья и многие другие схемы индексации

Преимущества GiST:

- работает с геометрическими типами (напр. точками)
- работает с диапазонами
- работает с полнотекстовым поиском
- работает со сложными запросами, включающими операции включения, перекрытия или расстояния.

GIN или GiST?

- GIN выигрывает в точности и скорости поиска у GiST. Если данные изменяются не часто, а искать надо быстро — скорее всего выбор падет на GIN.
- GiST выигрывает в скорости обновления. Если данные изменяются активно, придется сравнивать оба варианта и выбирать тот, чьи показатели будут лучше сбалансированы.

SP-GiST-индексы (*Space Partitioning Generalized Search Tree*)

поддерживает деревья поиска с разбиением, что облегчает разработку широкого спектра различных несбалансированных структур данных, в том числе деревьев квадрантов, а также k-мерных и префиксных деревьев.

Преимущества SP-GiST:

- работает с деревьями квадрантов
- работает с K-мерными деревьями
- работает с префиксными деревьями

BRIN (*Block Range Index*)

хранят обобщённые сведения о значениях, находящихся в физически последовательно расположенных блоках таблицы. Поэтому такие индексы наиболее эффективны для столбцов, значения в которых хорошо коррелируют с физическим порядком столбцов таблицы.

ВЫВОДЫ

1. что ценного сегодня узнали?
2. какие методы будете использовать для решения своих задач?
3. для закрепления материала дз для практиков будет выложено на следующий день до обеда вместе с записью встречи

ВОПРОСЫ

Оставьте отзыв!