Авторы: Егор Рогов, Павел Лузанов

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

Темы

Общие соображения о соединениях

Соединение вложенным циклом

Вычислительная сложность

Вложенный цикл в параллельных планах

Модификации: левые, правые, полу- и анти- соединения

Соединения

Способы соединения — не соединения SQL

- inner/left/right/full/cross join/in/exists — логические операции
- способы соединения — механизм реализации

Соединяются не таблицы, а наборы строк

- могут быть получены от любого узла дерева плана

Наборы строк соединяются попарно

- порядок соединений важен с точки зрения производительности
- обычно важен и порядок внутри пары

Мало получать данные с помощью рассмотренных методов доступа, надо еще и уметь объединять их. Для этого PostgreSQL предоставляет несколько способов.

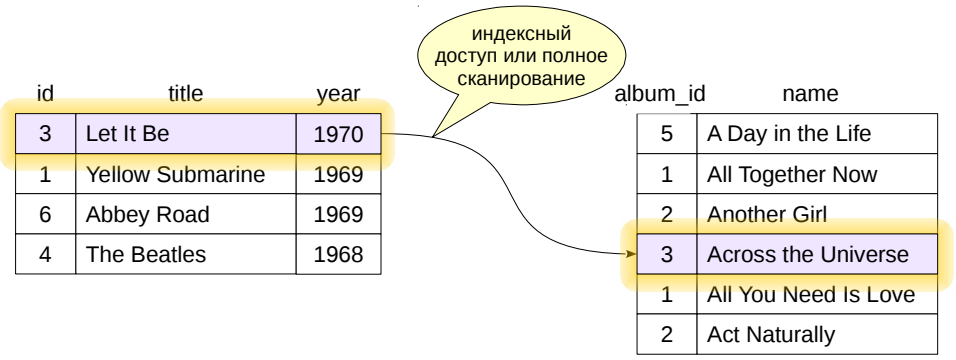
Способы соединения являются алгоритмами, с помощью которых можно объединить два набора строк. Их можно рассматривать как механизмы, с помощью которых реализуются операции соединения в языке SQL. Не стоит путать одно с другим: SQL-соединения — это логические операции с двумя множествами; способы соединения PostgreSQL — это возможные реализации таких соединений, учитывающие вопросы производительности.

Часто можно услышать, что соединяются *таблицы*. Это удобное упрощение, но на самом деле в общем случае соединяются *наборы строк*. Эти наборы действительно могут быть получены непосредственно из таблицы (с помощью одного из методов доступа), но с тем же успехом могут быть, например, результатом соединения других наборов строк.

Наконец, наборы строк всегда соединяются попарно. Порядок, в котором соединяются таблицы, не важен с точки зрения логики запроса (например, `(a join b) join c` или `(b join c) join a`), но очень важен с точки зрения производительности. Как мы увидим дальше, важен и порядок, в котором соединяются два набора строк (`a join b` или `b join a`).

Nested Loop

```
SELECT a.title, s.name
FROM albums a JOIN songs s ON a.id = s.a_id;
```



для каждой строки одного набора
перебираем подходящие строки другого набора

4

Начнем с соединения вложенными циклами (nested loop), как с самого простого. Его алгоритм таков: для каждой строки одного из наборов перебираем и возвращаем соответствующие ему строки второго набора. По сути, это два вложенных цикла, отсюда и название способа.

Заметим, что ко второму (внутреннему) набору мы будем обращаться столько раз, сколько строк в первом (внешнем) наборе. Если нет эффективного метода доступа для поиска «соответствующих» строк во втором наборе (то есть, попросту говоря, индекса на таблице), то придется неоднократно перебирать большое количество строк, не относящихся к делу. Очевидно, это будет не лучший выбор.

Рисунки проиллюстрируют этот способ соединения. На них:

- серым цветом обозначены строки, к которым ранее уже был доступ;

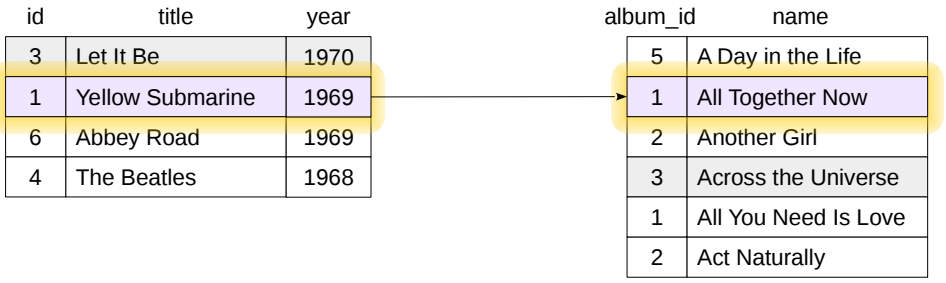
- фиолетовым цветом выделены строки, доступ к которым выполняется на шаге, показанном на слайде;

- оранжевым выделены строки, составляющие пару, подходящую по условию соединения (в данном примере — по равенству числовых идентификаторов).

Сначала мы читаем первую строку первого набора и находим ее пару во втором наборе. Соответствие нашлось, и у нас уже есть первая строка результата, которую можно вернуть вышестоящему узлу плана: («Let It Be», «Across the Universe»).

Nested Loop

```
SELECT a.title, s.name
FROM albums a JOIN songs s ON a.id = s.a_id;
```



id	title	year
3	Let It Be	1970
1	Yellow Submarine	1969
6	Abbey Road	1969
4	The Beatles	1968

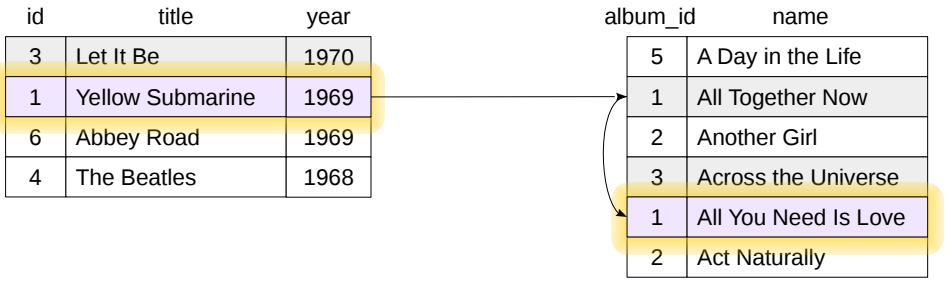
album_id	name
5	A Day in the Life
1	All Together Now
2	Another Girl
3	Across the Universe
1	All You Need Is Love
2	Act Naturally

Читаем вторую строку первого набора.

Для нее тоже перебираем пары из второго набора. Сначала возвращаем («Yellow Submarine», «All Together Now»)...

Nested Loop

```
SELECT a.title, s.name
FROM albums a JOIN songs s ON a.id = s.a_id;
```



id	title	year
3	Let It Be	1970
1	Yellow Submarine	1969
6	Abbey Road	1969
4	The Beatles	1968

album_id	name
5	A Day in the Life
1	All Together Now
2	Another Girl
3	Across the Universe
1	All You Need Is Love
2	Act Naturally

...затем вторую пару («Yellow Submarine», «All You Need Is Love»).

Nested Loop

```
SELECT a.title, s.name
FROM albums a JOIN songs s ON a.id = s.a_id;
```

id	title	year
3	Let It Be	1970
1	Yellow Submarine	1969
6	Abbey Road	1969
4	The Beatles	1968

album_id	name
5	A Day in the Life
1	All Together Now
2	Another Girl
3	Across the Universe
1	All You Need Is Love
2	Act Naturally

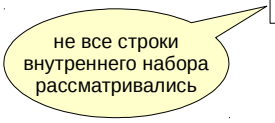
Переходим к третьей строке первого набора. Для нее соответствия нет.

Nested Loop

```
SELECT a.title, s.name
FROM albums a JOIN songs s ON a.id = s.a_id;
```

id	title	year
3	Let It Be	1970
1	Yellow Submarine	1969
6	Abbey Road	1969
4	The Beatles	1968

album_id	name
5	A Day in the Life
1	All Together Now
2	Another Girl
3	Across the Universe
1	All You Need Is Love
2	Act Naturally



Для четвертой строки тоже нет соответствий. На этом работа соединения заканчивается.

Заметим, что часть строк второго набора мы вовсе не рассматривали — на рисунке они остались белыми.

(С исходным кодом алгоритма можно познакомиться в файле src/backend/executor/nodeNestloop.c.)

Вычислительная сложность

$\sim N \times M$,

где N и M — число строк во внешнем и внутреннем наборах данных

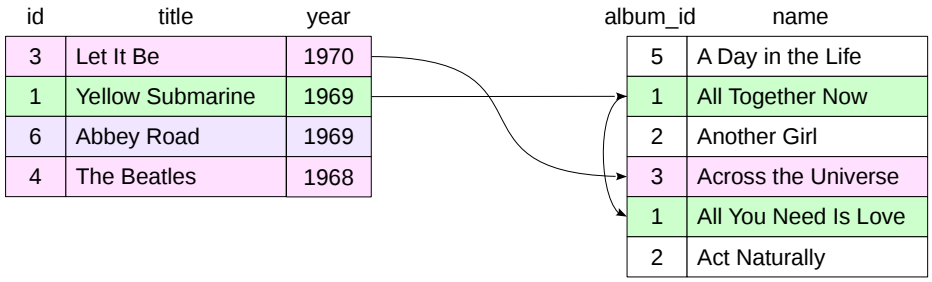
Эффективно только для небольшого числа строк

Если принять за *N* число строк во внешнем наборе данных, а за *M* — среднее число строк внутреннего набора, приходящееся на одну итерацию, то общая сложность соединения будет пропорциональна произведению *N* × *M*.

Из-за этого метод соединения вложенным циклом эффективен только для небольшого числа строк.

В частности, такой метод (в сочетании с индексным доступом) характерен для OLTP-запросов, в которых надо очень быстро вернуть очень небольшое число строк.

В параллельном режиме

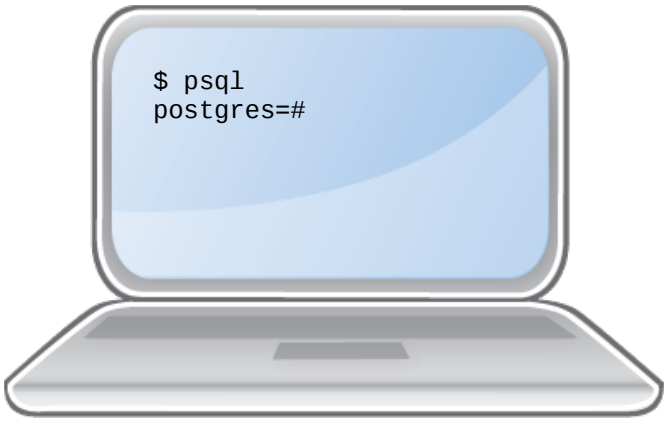


внешний набор строк сканируется параллельно,
внутренний — последовательно одним из рабочих процессов

Начиная с PostgreSQL 9.6, соединение вложенным циклом может выполняться в параллельном режиме.

Внешний набор строк сканируется несколькими рабочими процессами параллельно. Получив строку из внешнего набора, процесс затем перебирает соответствующие ему строки внутреннего набора последовательно.

Демонстрация



Итоги

Вложенный цикл не требует подготовительных действий

- может отдавать результат соединения без задержек

Эффективен для небольших выборок

- внешний набор строк не очень велик
- к внутреннему есть эффективный доступ (обычно по индексу)

Зависит от порядка соединения

- обычно лучше, если внешний набор меньше внутреннего

Поддерживает соединение по любому условию

- как эквисоединения, так и любые другие

Сильной стороной способа соединения вложенными циклами является его простота: не требуется никаких подготовительных действий, мы может начать возвращать результат практически моментально.

Обратная сторона состоит в том, что этот способ крайне неэффективен для больших объемов данных. Ситуация та же, что и с индексами: чем больше выборка, чем больше накладных расходов.

Таким образом, соединение вложенными циклами имеет смысл применять, если:

- один из наборов строк небольшой;

- к другому набору есть эффективный доступ по условию соединения;

- общее количество строк результата не велико.

Это обычная ситуация для OLTP-запросов (например, запросов от пользовательского интерфейса, где веб-страница или экранная форма должны открыться быстро и не выводят большой объем информации).

Еще одна особенность, которую стоит отметить: соединение вложенными циклами может работать для любого условия соединения. Подходит как эквисоединение (по условию равенства, как в примере), так и любое другое.

Практика

1. Создайте индекс на таблице рейсов (flights) по аэропортам отправления (departure_airport).
   Найдите все рейсы из Ульяновска и проверьте план выполнения запроса.
2. Соедините любые две таблицы без указания условий соединения (иными словами, выполните декартово произведение таблиц).
   Какой способ соединения будет выбран планировщиком?
3. Постройте таблицу расстояний между всеми аэропортами (так, чтобы каждая пара встречалась только один раз).
   Какой способ соединения используется в таком запросе?

3. Используйте оператор <@> из расширения earthdistance.