Новые возможности PgPro Enterprise 11

И.А. Фролков

www.postgrespro.ru

Postgres PROFESSIONAL

О чем пойдет речь

- Новые возможности
  - Оптимизатор
    - Статистика по индексам
    - Соединение таблицы самом с собой по первичному ключу
  - Executor
    - FULL OUTER JOIN
  - Разное
    - Изменение параметров другой сессии
    - Встроенный пул
    - Ограничение числа активных prepared statements
    - autoprepare

Статистика по индексам

- Кардинальность и селективность
- pg_stats
- Статистика по каждой колонке
- null_frac, n_distinct, most_common_vals…
- Каждая колонка по отдельности

Статистика по индексам

```
create table tt as select n, 0 as n1, 0 as n2, 0 as n3 from
generate_series(1,1000000) as g(n);
update tt set n1=n/100, n2=n/200, n3=n/1000;
vacuum full tt;
explain analyze
select * from tt
where n1=3 and n2=1 and n3=0;
Gather  (cost=1000.00..13697.77 rows=1 width=16) (actual
time=0.501..67.274 rows=100 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Parallel Seq Scan on tt  (cost=0.00..12697.67 rows=1
width=16) (actual time=39.779..61.753 rows=33 loops=3)
        Filter: ((n1 = 3) AND (n2 = 1) AND (n3 = 0))
        Rows Removed by Filter: 333300
Planning Time: 0.116 ms
Execution Time: 67.316 ms
```

Статистика по индексам

```
create index on tt(n1,n2,n3);
vacuum analyze tt;
Index Scan using tt_n1_n2_n3_idx on tt  (cost=0.42..183.93 rows=100
width=16) (actual time=0.081..0.128 rows=100 loops=1)
  Index Cond: ((n1 = 3) AND (n2 = 1) AND (n3 = 0))
Planning Time: 0.320 ms
Execution Time: 0.171 ms
```

Статистика по индексам

- Чем это плохо?

```
create table ttt as select 1 as n1, 1 n2, repeat('X',20) from
generate_series(1,100000) as g(n)
insert into ttt select  2 as n1, 2 n2, repeat('X',20) from
generate_series(1,100000) as g(n)
insert into ttt select 2 as n1, 1 n2, repeat('Z',20)
create index on ttt(n1,n2)
vacuum analyze ttt
explain analyze
select * from ttt where n1=2 and n2=1;
Seq Scan on ttt  (cost=0.00..4482.01 rows=50000 width=29) (actual
time=17.063..17.063 rows=1 loops=1)
  Filter: ((n1 = 2) AND (n2 = 1))
  Rows Removed by Filter: 200000
Planning Time: 0.058 ms
Execution Time: 17.080 ms
```

Статистика по индексам

- `vacuum analyze ttt;`
-
```
Index Scan using ttt_n1_n2_idx on ttt  (cost=0.42..6.19 rows=1
width=29) (actual time=0.047..0.049 rows=1 loops=1)
  Index Cond: ((n1 = 2) AND (n2 = 1))
Planning Time: 0.272 ms
Execution Time: 0.079 ms
```

Соединение таблицы самой с собой по первичному ключу

- Казалось бы, очень странное изобретение
- Нет, и вот почему:
  - Пусть есть таблица «работники»
  - ```
    create table hr.emp(
      id bigserial primary key,
      first_name text not null,
      middle_name text not null,
      last_name text not null,
      birth_date date not null,
      sex char(1) check (sex in ('M','F')),
      occupation_id bigint not null references
    hr.occupation(id)
    )
    ```
    ```

Соединение таблицы самой с собой по первичному ключу

- Представление «работники в подробностях»
- `create or replace view hr.emp_details as select id, ... from hr.emp`
- Представление «работники предпенсионного возраста»
-
```
create or replace view
hr.emp_at_preretirement_age as
select …?
where (now()-birth_date between
make_interval(years:=60) and
make_interval(years:=65) and sex='M')
or (now()-birth_date between
make_interval(years:=55) and
make_interval(years:=60) and sex='F')
```

Соединение таблицы самой с собой по первичному ключу

- Представление «работники в подробностях»
- `create or replace view hr.emp_details as select format('%s %s %s', last_name, first_name, middle_name) from hr.emp`
- Представление «работники предпенсионного возраста»
- `create or replace view hr.emp_at_preretirement_age as select id !`

Соединение таблицы самой с собой по первичному ключу

- ```
  select *
  from hr.emp_details d
  where |
  d.id in(select p.id
              from
          hr.emp_at_preretirement_age p)
  ```
- ```
  Hash Join  (cost=325.33..423.12 rows=18 width=111) (actual
  time=4.375..5.225 rows=69 loops=1)
    Hash Cond: (emp.id = e.id)
    ->  Seq Scan on emp  (cost=0.00..84.59 rows=3459 width=55)
  (actual time=0.010..0.600 rows=3459 loops=1)
    ->  Hash  (cost=325.11..325.11 rows=18 width=71) (actual
  time=3.706..3.706 rows=69 loops=1)

  ...
  ```

Соединение таблицы самой с собой по первичному ключу

- ```
  Seq Scan on emp e  (cost=0.00..240.29 rows=17 width=111) (actual
  time=0.547..1.162 rows=69 loops=1)
  ...
  ```
  ```

Соединение таблицы самой с собой по первичному ключу

- Другой источник подобных запросов — ORM
- Там вообще бывает много интересного

FULL OUTER JOIN

- Типовой случай
  - Первый случай
    - Сущность поступает
    - Сущность выдается
  - Второй случай
    - Сущность сдается на склад
    - Сущноть отправляется куда-то
  -

Получена клавиатура	Выдана Пупкину
Пупкин сдал клавиатуру	Отправлена на утилизацию
Получена волшебная палочка	
	Выданы валенки из запасов

FULL OUTER JOIN

- Как ни странно, обычный Postgres не умеет FULL OUTER JOIN не по равенству:
- ```
  select * from  generate_series(1,10) as g1(n)
  full outer join generate_series(1,10) as g2(n)
  on g1.n-g2.n=0
  ```
- ```
  SQL Error [0A000]: ERROR: FULL JOIN is only supported
  with merge-joinable or hash-joinable join conditions
  ```
- ```
  select * from  generate_series(1,10) as g1(n)
   left outer join generate_series(1,10) as g2(n)
   on g1.n-g2.n=0
   union all
  select * from  generate_series(1,10) as g2(n)
  left outer join generate_series(1,10) as g1(n)
   on g1.n-g2.n=0
  ```
  ```

Изменение параметров другой сессии

- В PgPro Enterprise 11 можно будет изменять параметры другой сесии.
- Доступно только суперпользователю
- Отладка, мониторинг

Встроенный пул соединений

- Есть внешние пулы
  - PgBouncer, pgPool
  - Встроенные пулы в Java/C#
- Иногда это не совсем удобно:
  - Для pgBouncer режим statement требует определенных усилий
  - Для пулов Java подобное только с JdbcTemplate
- В любом случае проблемы с транзакциями

Ограничение числа prepared statements

- Каждый prepared statement расходует память
- Если у вас их 10 или 100 — ничего страшного
- Что делать, если у вас их 20000?
- Только PgPro Enterprise

autoprepare

- Унаследованные приложения
- Что делать?
- autoprepare

Спасибо за внимание!

Контакты:

v.pupkinne@postgrespro.ru

+71234567890