DBS, CP4

Volodymyr Semenyug

INDEX:

Použití indexu na atributech bloodGroup, Rh a phone nam umožní rychlé odfiltrovat potřebné řádky v tabulkách blood a person. Změna rychlosti ve vykonání queurie viz. následující stránka.

```
drop index if exists bloodGroupIdx;
drop index if exists RhIdx;
drop index if exists phoneIdx;
create index bloodGroupIdx on blood(bloodGroup);
create index RhIdx on blood(Rh);
create index phoneIdx on person(phone);
explain
SELECT donormedcard, bloodGroup, Rh, NAME, phone
FROM blood
    JOIN donor
    ON (blood.donormedcard = donor.medcard)
    JOIN person
    ON (person.pass = donor.pass)
WHERE (bloodGroup = 1) AND (Rh = TRUE) AND (phone IS NOT NULL)
OFFSET 2;
```

Before index:

	QUERY PLAN text
1	Limit (cost=2110.45..2854.97 rows=3487 width=45)
2	[...] -> Hash Join (cost=2110.03..2854.97 rows=3489 width=45)
3	[...] Hash Cond: ((person.pass)::text = (donor.pass)::text)
4	[...] -> Seq Scan on person (cost=0.00..602.00 rows=28815 width=40)
5	[...] Filter: (phone IS NOT NULL)
6	[...] -> Hash (cost=2061.59..2061.59 rows=3875 width=27)
7	[...] -> Hash Join (cost=764.44..2061.59 rows=3875 width=27)
8	[...] Hash Cond: ((donor.medcard)::text = (blood.donormedcard)::text)
9	[...] -> Seq Scan on donor (cost=0.00..513.60 rows=31360 width=24)
10	[...] -> Hash (cost=716.00..716.00 rows=3875 width=16)
11	[...] -> Seq Scan on blood (cost=0.00..716.00 rows=3875 width=16)
12	[...] Filter: (rh AND (bloodgroup = 1))

After:

	QUERY PLAN text
1	Limit (cost=1963.61..2708.13 rows=3487 width=45)
2	[...] -> Hash Join (cost=1963.18..2708.13 rows=3489 width=45)
3	[...] Hash Cond: ((person.pass)::text = (donor.pass)::text)
4	[...] -> Seq Scan on person (cost=0.00..602.00 rows=28815 width=40)
5	[...] Filter: (phone IS NOT NULL)
6	[...] -> Hash (cost=1914.74..1914.74 rows=3875 width=27)
7	[...] -> Hash Join (cost=617.59..1914.74 rows=3875 width=27)
8	[...] Hash Cond: ((donor.medcard)::text = (blood.donormedcard)::text)
9	[...] -> Seq Scan on donor (cost=0.00..513.60 rows=31360 width=24)
10	[...] -> Hash (cost=569.16..569.16 rows=3875 width=16)
11	[...] -> Bitmap Heap Scan on blood (cost=147.72..569.16 rows=3875 width=16)
12	[...] Recheck Cond: (bloodgroup = 1)
13	[...] Filter: rh
14	[...] -> Bitmap Index Scan on bloodgroupidx (cost=0.00..146.75 rows=7795 width=0)
15	[...] Index Cond: (bloodgroup = 1)

Trigger:

Trigger blood_th_Rh_group použijme při přídání nových, nebo při změně existujících položek v tabulce blood. Konkrétně provede kontrolu, že skupina krve je v rozmezí [1, 4], rhesus faktor nabývá hodnot true nebo false.

```
create function check_Rh_group()
returns trigger
as $$
begin
    if (not (new.Rh between true and false) then
        raise exception 'invalid Rh factor';
    elseif (not (new.bloodGroup between 1 and 4)) then
        raise exception 'invalid blood group';
    end if;
    return new;
end;
    $$
language plpgsql;
create trigger blood_tg_Rh_group before insert or update on blood
for each row execute procedure check_Rh_group();
```

View:

Umožní používat hodnoty tohoto querie.

```
create view countOfDonatedBlood as
SELECT NAME, COUNT(*) AS counter
FROM blood
    JOIN donor
    ON (blood.donormedcard = donor.medcard)
    JOIN person
    ON (donor.pass = person.pass)
GROUP BY (NAME)
HAVING (NAME NOT LIKE 'A%')
ORDER BY (counter) DESC
```

Transaction:

Bez použití transakce na new_transfer by mohlo dojít k problému R-W: nový převod bude vytvořen ještě před tím, než sklad krve v pobočce bude uplně obsazen.

```
create function new_transfer(brNumber INTEGER, artKey INTEGER, cid integer, ceo varchar(100))
returns boolean
as $$
    declare
        usedPlace INTEGER;
        capacity INTEGER;
    begin
        if (brNumber is none) then return false;
        else
            usedPlace := (select bankUsedPlace from branch where(branch.branchNumber = brNumber));
            capacity := (select bankCapacity from branch where(branch.branchNumber = brNumber));
            if (usedPlace < capacity) then return false; end if;
            update branch set bankUsedPlace = 0 where (branch.branchNumber = brNumber);
            insert into transfers values (artKey, brNumber, cid, ceo, usedPlace);
            return true;
        end if;
    end;
    $$
language plpgsql;
begin transaction isolation level read committed;
select new_transfer(0, 105, 6618, 'semenvol');
commit transaction;
```