ESCOLA SUPERIOR DE REDES RNP

Administração de Bancos de Dados

Tarefa

Módulo 7

Atividade 7.a – Habilitar o log de locks

Ligar parâmetro log_lock_wait;

```
file: db/data/postgresql.conf

                                         #   %q = stop here in non-session
                                         #        processes
                                         #   %% = '%'
                                         # e.g. '<%u%%%d> '
log_lock_waits = on                      # log lock waits >= deadlock_timeout
#log_parameter_max_length = -1           # when logging statements, limit logged
                                         # bind-parameter values to N bytes;
                                         # -1 means print in full, 0 disables
#log_parameter_max_length_on_error = 0   # when logging an error, limit logged
                                         # bind-parameter values to N bytes;
                                         # -1 means print in full, 0 disables
log_statement = 'all'                    # none, ddl, mod, all
#log_replication_commands = off
#log_temp_files = -1                     # log temporary files equal or larger
```

Recarregar configurações.

Atividade 7.b. – Testar funcionamento do MVCC

Abrir uma conexão com o banco benchmark;

```
postgres@debian10:~$ psql
psql (13.1)
Type "help" for help.

postgres=# \c benchmark;
You are now connected to database "benchmark" as user "postgres".
benchmark=#
```

Iniciar uma transação e fazer um update em um registro da tabela pgbench_accounts;

```
benchmark=# begin;
BEGIN
benchmark=*# update pgbench_accounts set filler = 'foo' where aid = 10;
UPDATE 1
benchmark=*# select * from pgbench_accounts where aid = 10;
 aid | bid | abalance |                                    filler
-----+-----+----------+--------------------------------------------------------------------------------------
  10 |   1 |        0 | foo
(1 row)
```

Abrir outra conexão e executar um select no mesmo registro;

```
postgres@debian10:~$ psql -d benchmark
psql (13.1)
Type "help" for help.

benchmark=# select * from pgbench_accounts where aid = 10;
 aid | bid | abalance |                                        filler
-----+-----+----------+--------------------------------------------------------------
  10 |   1 |        0 |
(1 row)
```

Executar rollback na primeira transação.

```
benchmark=*# rollback;
ROLLBACK
```

Atividade 7.c – Testar locks em updates

Abrir uma conexão com o banco benchmark;

```
postgres@debian10:~$ psql -d benchmark
psql (13.1)
Type "help" for help.

benchmark=#
```

Iniciar uma transação e executar update em um registro;

```
benchmark=# begin;
BEGIN
benchmark=*# update pgbench_accounts set filler = 'foo' where aid = 10;
UPDATE 1
benchmark=*#
```

Abrir outra conexão, tentar atualizar o mesmo registro;

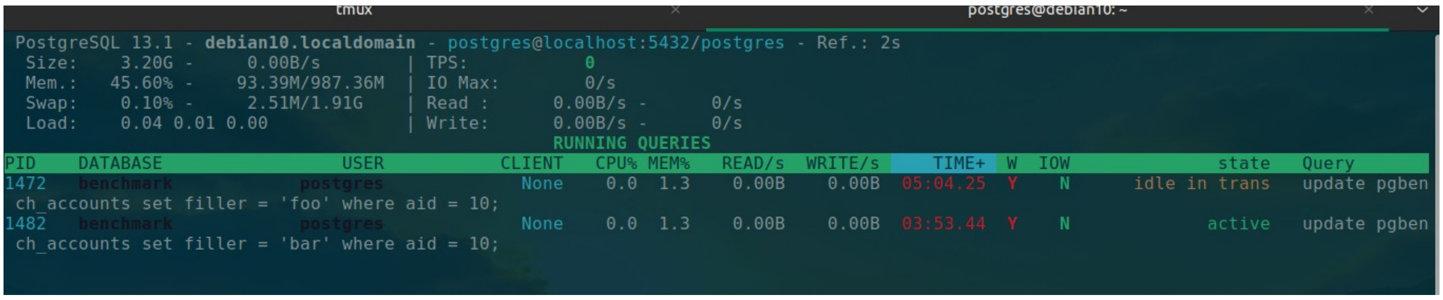
```
postgres@debian10:~$ psql -d benchmark
psql (13.1)
Type "help" for help.

benchmark=# update pgbench_accounts set filler = 'bar' where aid = 10;
```

A operação não foi concluída. Ficou bloqueada

Em um terceiro terminal:

- Verificar locks com pg_activity;



```
PostgreSQL 13.1 - debian10.localdomain - postgres@localhost:5432/postgres - Ref.: 2s
 Size:     3.20G -      0.00B/s          | TPS:            0
 Mem.:    45.60% -     93.39M/987.36M    | IO Max:         0/s
 Swap:     0.10% -      2.51M/1.91G      | Read :       0.00B/s -       0/s
 Load:     0.04 0.01 0.00                | Write:       0.00B/s -       0/s
                                                    RUNNING QUERIES
PID    DATABASE                 USER           CLIENT    CPU% MEM%    READ/s  WRITE/s      TIME+  W  IOW            state   Query
1472   benchmark                postgres         None     0.0  1.3     0.00B    0.00B   05:04.25  Y    N    idle in trans   update pgben
 ch_accounts set filler = 'foo' where aid = 10;
1482   benchmark                postgres         None     0.0  1.3     0.00B    0.00B   03:53.44  Y    N           active   update pgben
 ch_accounts set filler = 'bar' where aid = 10;
```

- Localize mensagens de locks na log do PostgreSQL;

```
2022-08-17 19:55:31 UTC [1472]: [3-1] user=postgres,db=benchmark LOG:  statement: update pgbench_accounts set filler = 'foo' where aid = 10;
2022-08-17 19:56:41 UTC [1482]: [1-1] user=postgres,db=benchmark LOG:  statement: update pgbench_accounts set filler = 'bar' where aid = 10;
2022-08-17 19:56:42 UTC [1482]: [2-1] user=postgres,db=benchmark LOG:  process 1482 still waiting for ShareLock on transaction 167101 after 1000.205 ms
2022-08-17 19:56:42 UTC [1482]: [3-1] user=postgres,db=benchmark DETAIL:  Process holding the lock: 1472. Wait queue: 1482.
2022-08-17 19:56:42 UTC [1482]: [4-1] user=postgres,db=benchmark CONTEXT:  while updating tuple (0,10) in relation "pgbench_accounts"
```

- Localize a entrada relacionada no catálogo pg_locks;

```
  locktype     | database | relation | page | tuple | virtualxid | transactionid | classid | objid | objsubid | virtualtransaction | pid  |       mode       | granted | fastpath
---------------+----------+----------+------+-------+------------+---------------+---------+-------+----------+--------------------+------+------------------+---------+----------
 relation      |    12662 |    12141 |      |       |            |               |         |       |          | 5/381              | 1598 | AccessShareLock  | t       | t
 virtualxid    |          |          |      |       | 5/381      |               |         |       |          | 5/381              | 1598 | ExclusiveLock    | t       | t
 relation      |    41057 |    41080 |      |       |            |               |         |       |          | 4/79               | 1482 | RowExclusiveLock | t       | t
 relation      |    41057 |    41078 |      |       |            |               |         |       |          | 4/79               | 1482 | RowExclusiveLock | t       | t
 relation      |    41057 |    41064 |      |       |            |               |         |       |          | 4/79               | 1482 | RowExclusiveLock | t       | t
 virtualxid    |          |          |      |       | 4/79       |               |         |       |          | 4/79               | 1482 | ExclusiveLock    | t       | t
 relation      |    41057 |    41080 |      |       |            |               |         |       |          | 3/138              | 1472 | RowExclusiveLock | t       | t
 relation      |    41057 |    41078 |      |       |            |               |         |       |          | 3/138              | 1472 | RowExclusiveLock | t       | t
 relation      |    41057 |    41064 |      |       |            |               |         |       |          | 3/138              | 1472 | RowExclusiveLock | t       | t
 virtualxid    |          |          |      |       | 3/138      |               |         |       |          | 3/138              | 1472 | ExclusiveLock    | t       | t
 transactionid |          |          |      |       |            |        167101 |         |       |          | 4/79               | 1482 | ShareLock        | f       | f
 transactionid |          |          |      |       |            |        167102 |         |       |          | 4/79               | 1482 | ExclusiveLock    | t       | f
 tuple         |    41057 |    41064 |    0 |    10 |            |               |         |       |          | 4/79               | 1482 | ExclusiveLock    | t       | f
 transactionid |          |          |      |       |            |        167101 |         |       |          | 3/138              | 1472 | ExclusiveLock    | t       | f
(14 rows)
```

- Consultar na pg_stat_activity as colunas que identificam a situação das conexões envolvidas.

```
postgres=# select state, query from pg_stat_activity;
        state         |                           query
----------------------+-----------------------------------------------------------
                      |
                      |
 idle in transaction  | update pgbench_accounts set filler = 'foo' where aid = 10;
 active               | update pgbench_accounts set filler = 'bar' where aid = 10;
 active               | select state, query from pg_stat_activity;
                      |
                      |
                      |
(8 rows)
```

Executar um Rollback na primeira transação.

```
benchmark=*# rollback;
ROLLBACK
```

Atividade 7.d – Testar deadlocks

Abrir uma conexão com o banco benchmark;
Iniciar uma transação;
Executar update em um registro;

```
postgres@debian10:~$ psql -d benchmark
psql (13.1)
Type "help" for help.

benchmark=# begin;
BEGIN
benchmark=*# update pgbench_accounts set filler = 'bar' where aid = 60;
UPDATE 1
benchmark=*#
```

Abrir outra conexão, iniciar uma transação, executar um update em outro registro;

```
postgres@debian10:~$ psql -d benchmark
psql (13.1)
Type "help" for help.

benchmark=# begin;
BEGIN
benchmark=*# update pgbench_accounts set filler = 'foo' where aid = 50;
UPDATE 1
benchmark=*#
```

Na primeira conexão, tentar atualizar o mesmo registro da segunda;

```
postgres@debian10:~$ psql -d benchmark
psql (13.1)
Type "help" for help.

benchmark=# begin;
BEGIN
benchmark=*# update pgbench_accounts set filler = 'bar' where aid = 60;
UPDATE 1
benchmark=*# update pgbench_accounts set filler = 'barbar' where aid = 50;
```

Na segunda conexão, tentar atualizar o primeiro registro da primeira conexão.

```
postgres@debian10:~$ psql -d benchmark
psql (13.1)
Type "help" for help.

benchmark=# begin;
BEGIN
benchmark=*# update pgbench_accounts set filler = 'foo' where aid = 50;
UPDATE 1
benchmark=*# update pgbench_accounts set filler = 'foofoo' where aid = 60;
ERROR:  deadlock detected
DETAIL:  Process 1641 waits for ShareLock on transaction 167105; blocked by process 1637.
Process 1637 waits for ShareLock on transaction 167106; blocked by process 1641.
HINT:  See server log for query details.
CONTEXT:  while updating tuple (0,60) in relation "pgbench_accounts"
benchmark=!#
```

Atividade 7.e – Usar EXPLAIN e criação de índices

Executar EXPLAIN na seguinte query:

```
SELECT *
FROM pgbench_accounts a
INNER JOIN pgbench_branches b ON a.bid=b.bid
INNER JOIN pgbench_tellers t ON t.bid=b.bid
WHERE a.bid=56;
```

Analise o custo

```
                                          QUERY PLAN
--------------------------------------------------------------------------------
 Nested Loop  (cost=0.43..152264.37 rows=1013070 width=813)
   ->  Index Scan using idx_accounts_bid_aid on pgbench_accounts a  (cost=0.43..139578.12
rows=101307 width=97)
         Index Cond: (bid = 56)
   ->  Materialize  (cost=0.00..22.90 rows=10 width=716)
         ->  Nested Loop  (cost=0.00..22.85 rows=10 width=716)
               ->  Seq Scan on pgbench_branches b  (cost=0.00..2.25 rows=1 width=364)
                     Filter: (bid = 56)
               ->  Seq Scan on pgbench_tellers t  (cost=0.00..20.50 rows=10 width=352)
                     Filter: (bid = 56)
(9 rows)
```

Executar EXPLAIN ANALYZE;

- Analise o tempo

```
                                          QUERY PLAN
--------------------------------------------------------------------------------
 Nested Loop  (cost=0.43..152264.37 rows=1013070 width=813) (actual time=56.642..716.340 rows=1000000 loops=1)
   ->  Index Scan using idx_accounts_bid_aid on pgbench_accounts a  (cost=0.43..139578.12 rows=101307 width=97) (actual
time=40.824..44
         Index Cond: (bid = 56)
   ->  Materialize  (cost=0.00..22.90 rows=10 width=716) (actual time=0.000..0.001 rows=10 loops=100000)
         ->  Nested Loop  (cost=0.00..22.85 rows=10 width=716) (actual time=15.794..36.233 rows=10 loops=1)
               ->  Seq Scan on pgbench_branches b  (cost=0.00..2.25 rows=1 width=364) (actual time=13.509..13.511 rows=1
loops=1)
                     Filter: (bid = 56)
                     Rows Removed by Filter: 99
               ->  Seq Scan on pgbench_tellers t  (cost=0.00..20.50 rows=10 width=352) (actual time=2.277..22.708 rows=10
loops=1)
                     Filter: (bid = 56)
                     Rows Removed by Filter: 990
 Planning Time: 0.157 ms
 Execution Time: 758.638 ms
(13 rows)
```

Executar EXPLAIN (ANALYZE,BUFFERS);

- Analise o acerto em cache e a leitura em disco.

```
                                        QUERY PLAN
----------------------------------------------------------------------------------------------------------------------------------
 Nested Loop  (cost=0.43..152264.37 rows=1013070 width=813) (actual time=0.093..260.140 rows=1000000 loops=1)
   Buffers: shared hit=2027
   ->  Index Scan using idx_accounts_bid_aid on pgbench_accounts a  (cost=0.43..139578.12 rows=101307 width=97) (actual 
time=0.029..21.
         Index Cond: (bid = 56)
         Buffers: shared hit=2018
   ->  Materialize  (cost=0.00..22.90 rows=10 width=716) (actual time=0.000..0.001 rows=10 loops=100000)
         Buffers: shared hit=9
         ->  Nested Loop  (cost=0.00..22.85 rows=10 width=716) (actual time=0.059..0.149 rows=10 loops=1)
               Buffers: shared hit=9
               ->  Seq Scan on pgbench_branches b  (cost=0.00..2.25 rows=1 width=364) (actual time=0.018..0.019 rows=1 loops=1)
                     Filter: (bid = 56)
                     Rows Removed by Filter: 99
                     Buffers: shared hit=1
               ->  Seq Scan on pgbench_tellers t  (cost=0.00..20.50 rows=10 width=352) (actual time=0.040..0.125 rows=10 loops=1)
                     Filter: (bid = 56)
                     Rows Removed by Filter: 990
                     Buffers: shared hit=8
 Planning Time: 0.157 ms
 Execution Time: 307.363 ms
(19 rows)
```

Criar um índice na coluna bid da tabela pgbench_tellers;

```
benchmark=# CREATE INDEX idx_tellers_bid ON pgbench_tellers(bid);
CREATE INDEX
benchmark=#
```

Execute um EXPLAIN(ANALYZE,BUFFERS) novamente e analise as informações de custo, tempo e buffers;

```
                                                         QUERY PLAN
-------------------------------------------------------------------------------------------------------------------------------
 Nested Loop  (cost=4.79..152256.70 rows=1013070 width=813) (actual time=0.101..255.165 rows=1000000 loops=1)
   Buffers: shared hit=2022 read=2
   ->  Index Scan using idx_accounts_bid_aid on pgbench_accounts a  (cost=0.43..139578.12 rows=101307 width=97) (actual 
time=0.026..21.
         Index Cond: (bid = 56)
         Buffers: shared hit=2018
   ->  Materialize  (cost=4.35..15.23 rows=10 width=716) (actual time=0.000..0.001 rows=10 loops=100000)
         Buffers: shared hit=4 read=2
         ->  Nested Loop  (cost=4.35..15.18 rows=10 width=716) (actual time=0.068..0.080 rows=10 loops=1)
               Buffers: shared hit=4 read=2
               ->  Seq Scan on pgbench_branches b  (cost=0.00..2.25 rows=1 width=364) (actual time=0.017..0.019 rows=1 loops=1)
                     Filter: (bid = 56)
                     Rows Removed by Filter: 99
                     Buffers: shared hit=1
               ->  Bitmap Heap Scan on pgbench_tellers t  (cost=4.35..12.83 rows=10 width=352) (actual time=0.047..0.055 
rows=10 loops=
                     Recheck Cond: (bid = 56)
                     Heap Blocks: exact=3
                     Buffers: shared hit=3 read=2
                     ->  Bitmap Index Scan on idx_tellers_bid  (cost=0.00..4.35 rows=10 width=0) (actual time=0.041..0.041 
rows=10 loop
                           Index Cond: (bid = 56)
                           Buffers: shared read=2
 Planning:
   Buffers: shared hit=15 read=1
 Planning Time: 0.386 ms
 Execution Time: 301.588 ms
(24 rows)
```

Atividade 7.f – Criar índice composto

Execute o EXPLAIN da query:

```
SELECT DISTINCT *
FROM pgbench_accounts
WHERE bid=81 AND aid NOT IN (1,46,28,04,77,93);
```

```
benchmark=# explain
benchmark-# SELECT DISTINCT *
benchmark-# FROM pgbench_accounts
benchmark-# WHERE bid=81 AND aid NOT IN (1,46,28,04,77,93);

                                                  QUERY PLAN
---------------------------------------------------------------------------------
 HashAggregate  (cost=147107.22..148174.14 rows=106692 width=97)
   Group Key: aid, bid, abalance, filler
   ->  Index Scan using idx_accounts_bid_aid on pgbench_accounts  (cost=0.43..146040.30 rows=106692 width=97)
         Index Cond: (bid = 81)
         Filter: (aid <> ALL ('{1,46,28,4,77,93}'::integer[]))
(5 rows)
```

Crie um índice composto para query.

```
benchmark=# CREATE INDEX idx_branch_aid_bid ON pgbench_accounts(aid,bid);
CREATE INDEX
```

Teste novamente a query e veja o plano de execução

```
benchmark=# explain
SELECT DISTINCT *
FROM pgbench_accounts
WHERE bid=81 AND aid NOT IN (1,46,28,04,77,93);

                                                  QUERY PLAN
---------------------------------------------------------------------------------
 HashAggregate  (cost=146042.34..147099.01 rows=105667 width=97)
   Group Key: aid, bid, abalance, filler
   ->  Index Scan using idx_accounts_bid_aid on pgbench_accounts  (cost=0.43..144985.67 rows=105667 width=97)
         Index Cond: (bid = 81)
         Filter: (aid <> ALL ('{1,46,28,4,77,93}'::integer[]))
(5 rows)
```

Atividade 7.g – Criar uma Visão Materializada

Abra uma conexão com o banco benchmark;

```
postgres@debian10:~$ psql -d benchmark
psql (13.1)
Type "help" for help.

benchmark=#
```

Crie uma visão materializada com todos os registros da tabela pgbench_accounts cujo abalance seja maior que zero;

NOTA: Caso não haja nenhum registro, rode um teste do pgbench primeiro para alterar registros aleatoriamente

```
$ pgbench -T 60 benchmark
```

```
postgres@debian10:~$ psql -d benchmark
psql (13.1)
Type "help" for help.

benchmark=# create materialized view mv_accounts
benchmark-# as select * from pgbench_accounts
benchmark-# where abalance > 0;
SELECT 2178
benchmark=#
```

Selecione todos os registros da visão medindo o tempo de execução

Compare com o tempo de execução buscando na tabela original pgbench_accounts os registros positivos

DICA: Você pode usar EXPLAIN ANALYZE antes da query ou usar \timing no psql

```
benchmark=# explain (analyze)
benchmark-# select * from mv_accounts;
                                                 QUERY PLAN
--------------------------------------------------------------------------------
 Seq Scan on mv_accounts  (cost=0.00..57.78 rows=2178 width=97) (actual
time=0.013..0.533 rows=2178 loops=1)
 Planning Time: 0.229 ms
 Execution Time: 0.825 ms
(3 rows)

benchmark=# explain (analyze)
select * from pgbench_accounts where abalance > 0;
                                                                      QUERY PLAN
--------------------------------------------------------------------------------
 Gather  (cost=1000.00..218657.43 rows=1 width=97) (actual time=102.021..19350.573 rows=2178
loops=1)
   Workers Planned: 2
   Workers Launched: 2
   ->  Parallel Seq Scan on pgbench_accounts  (cost=0.00..217657.33 rows=1 width=97) (actual
time=95.398..19333.292 rows=726 loops=3)
         Filter: (abalance > 0)
         Rows Removed by Filter: 3332607
 Planning Time: 0.173 ms
 Execution Time: 19351.225 ms
(8 rows)
```

Atividade 7.h -Testar opções de SELECTs com locks

Abrir uma conexão com o banco benchmark;
Iniciar uma transação;
Executar update em um registro qualquer; Por exemplo:

```
UPDATE pgbench_accounts SET abalance=100 WHERE aid=1273;
```

```
postgres@debian10:~$ psql -d benchmark
psql (13.1)
Type "help" for help.

benchmark=# begin;
BEGIN
benchmark=*# update pgbench_accounts set filler = 'foo/bar' where aid = 500;
UPDATE 1
benchmark=*#
```

Abrir outra conexão, testes os seguintes comandos:

```
SELECT * FROM pgbench_accounts WHERE aid=500 FOR UPDATE NOWAIT;
```

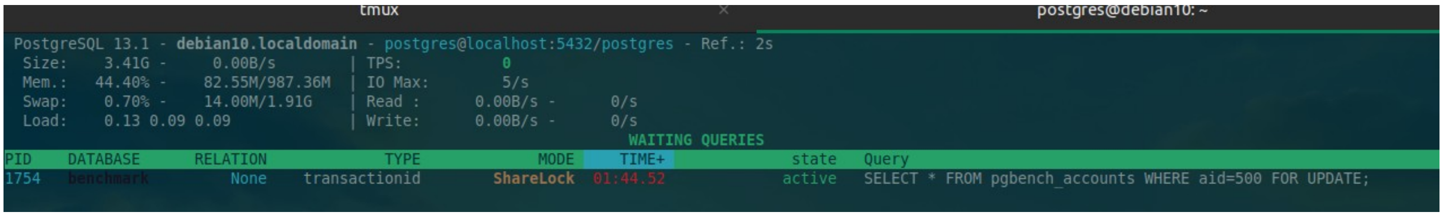
```
benchmark=# SELECT * FROM pgbench_accounts WHERE aid=500 FOR UPDATE NOWAIT;
ERROR:  could not obtain lock on row in relation "pgbench_accounts"
```

```
SELECT * FROM pgbench_accounts WHERE aid=500 FOR UPDATE SKIP LOCKED;
```

```
benchmark=# SELECT * FROM pgbench_accounts WHERE aid=500 FOR UPDATE SKIP LOCKED;
 aid | bid | abalance | filler
-----+-----+----------+--------
(0 rows)
```

```
SELECT * FROM pgbench_accounts WHERE aid=500 FOR UPDATE;
```

```
benchmark=# SELECT * FROM pgbench_accounts WHERE aid=500 FOR UPDATE;
```



```
PostgreSQL 13.1 - debian10.localdomain - postgres@localhost:5432/postgres - Ref.: 2s
 Size:    3.41G -      0.00B/s        | TPS:             0
 Mem.:   44.40% -     82.55M/987.36M  | IO Max:          5/s
 Swap:    0.70% -     14.00M/1.91G    | Read :       0.00B/s -      0/s
 Load:    0.13 0.09 0.09              | Write:       0.00B/s -      0/s
                                                                 WAITING QUERIES
PID    DATABASE      RELATION          TYPE            MODE     TIME+           state    Query
1754   benchmark           None     transactionid      ShareLock  01:44.52        active   SELECT * FROM pgbench_accounts WHERE aid=500 FOR UPDATE;
```

Interprete as diferenças entre os comandos

A opção **for update** faz com que a consulta fique bloqueada até o registro ser liberado. **Nowait**, gera um erro, mas não bloqueia a consulta. **Skipe locked** ignora os registros bloqueados.