Мамонов Антон 3ИСиП-19-1

Изоляция
Изоляция и системный каталог:

1 терминал

```
access=# create function f() returns integer as $$ select 1;
access$# $$ language sql;
CREATE FUNCTION
```

2 терминал

```
postgres=# \c access
Вы подключены к базе данных "access" как пользователь "postgres".
access=# begin isolation level read committed;
BEGIN
access=# select f();
 f
---
 1
```

1 терминал:

```
access=# create or replace function f() returns integer as $$ select 2;
access$# $$ language sql;
CREATE FUNCTION
access=# select f();
 f
---
 2
```

2 терминал

```
access=# select f();
 f
---
 1
```

1 терминал:

```
access=# create table t(id integer);
CREATE TABLE
```

2 терминал:

```
access=# begin isolation level repeatable read;
BEGIN
```

1 терминал:

```
access=# insert into t values (1);
INSERT 0 1
access=#
```

2 терминал:

```
access=# select * from t;
 id
----
  1
```

1 терминал:

```
access=# insert into t values (2);
INSERT 0 1
access=#
```

2 терминал:

```
access=# select * from t;
 id
----
  1
```

Страницы и версии строк

```
access=# CREATE EXTENSION pageinspect;
ОШИБКА:  расширение "pageinspect" уже существует
access=# CREATE TABLE t(s text);
CREATE TABLE
access=# CREATE VIEW t_v AS
access-# SELECT '(0,'||lp||')' AS ctid,
access-#        CASE lp_flags
access-#          WHEN 0 THEN 'unused'
access-#          WHEN 1 THEN 'normal'
access-#          WHEN 2 THEN 'redirect to '||lp_off
access-#          WHEN 3 THEN 'dead'
access-#        END AS state,
access-#        t_xmin as xmin,
access-#        t_xmax as xmax,
access-#        CASE WHEN (t_infomask & 256) > 0  THEN 't' END AS xmin_c,
access-#        CASE WHEN (t_infomask & 512) > 0  THEN 't' END AS xmin_a,
access-#        CASE WHEN (t_infomask & 1024) > 0 THEN 't' END AS xmax_c,
access-#        CASE WHEN (t_infomask & 2048) > 0 THEN 't' END AS xmax_a,
access-#        t_ctid
access-# FROM heap_page_items(get_raw_page('t',0))
access-# ORDER BY lp;
CREATE VIEW
```

Запускаем транзакцию:

```
access=# begin;
BEGIN
access=# insert into t values ('first');
INSERT 0 1
access=# select txid_current();
 txid_current
--------------
          661
```

```
access=# select *, xmin, xmax FROM t;
   s   | xmin | xmax
-------+------+------
 first |  661 |    0
```

```
access=# select * from t_v;
 ctid  | state  | xmin | xmax | xmin_c | xmin_a | xmax_c | xmax_a | t_ctid
-------+--------+------+------+--------+--------+--------+--------+--------
 (0,1) | normal |  661 |    0 |        |        |        | t      | (0,1)
```

Создаем точку сохранения:

```
access=# select *, xmin, xmax FROM t;
   s    | xmin | xmax
--------+------+------
 first  |  661 |    0
 second |  662 |    0
```

```
access=# savepoint sp;
SAVEPOINT
access=# insert into t values ('second');
INSERT 0 1
access=# select txid_current();
 txid_current
--------------
          661
```

```
access=# select * from t_v;
 ctid  | state  | xmin | xmax | xmin_c | xmin_a | xmax_c | xmax_a | t_ctid
-------+--------+------+------+--------+--------+--------+--------+--------
 (0,1) | normal |  661 |    0 |        |        |        | t      | (0,1)
 (0,2) | normal |  662 |    0 |        |        |        | t      | (0,2)
```

Делаем откат:

```
access=# rollback to sp;
ROLLBACK
access=# insert into t values ('third');
INSERT 0 1
access=# select txid_current();
 txid_current
--------------
          661
```

```
access=# select *, xmin, xmax FROM t;
   s    | xmin | xmax
-------+------+------
 first |  661 |    0
 third |  663 |    0
```

```
access=# select * from t_v;
 ctid  | state  | xmin | xmax | xmin_c | xmin_a | xmax_c | xmax_a | t_ctid
-------+--------+------+------+--------+--------+--------+--------+--------
 (0,1) | normal |  661 |    0 | t      |        |        | t      | (0,1)
 (0,2) | normal |  662 |    0 |        | t      |        | t      | (0,2)
 (0,3) | normal |  663 |    0 | t      |        |        | t      | (0,3)
```

Завершаем транзакцию и смотри на результат:

```
access=# commit;
COMMIT
access=# select *, xmin, xmax FROM t;
   s    | xmin | xmax
-------+------+------
 first |  661 |    0
 third |  663 |    0
```

```
access=# select * from t_v;
 ctid  | state  | xmin | xmax | xmin_c | xmin_a | xmax_c | xmax_a | t_ctid
-------+--------+------+------+--------+--------+--------+--------+-------
 (0,1) | normal |  661 |    0 |        |        |        | t      | (0,1)
 (0,2) | normal |  662 |    0 |        | t      |        | t      | (0,2)
 (0,3) | normal |  663 |    0 |        |        |        | t      | (0,3)
```

Снимки данных

Создаем таблицу и запускаем первую транзакцию:

```
access=# create table t(n integer);
CREATE TABLE
access=# begin;
BEGIN
access=# insert into t(n) values (1);
INSERT 0 1
access=# select txid_current();
 txid_current
--------------
          666
```

```
access=# commit;
COMMIT
access=# BEGIN ISOLATION LEVEL REPEATABLE READ;
BEGIN
access=# select * from t;
 n
---
 1
```

```
access=# select txid_current();
 txid_current
--------------
          668
```

```
access=# SELECT txid_current_snapshot();
 txid_current_snapshot
-----------------------
 668:668:
```

Вторая транзакция:

```
access=# begin isolation level repeatable read;
BEGIN
access=# delete from t;
DELETE 1
access=# select * from t;
 n
---
```

```
access=# SELECT txid_current();
 txid_current
--------------
          669
```

```
access=# SELECT txid_current_snapshot();
 txid_current_snapshot
-----------------------
668:668:
```

Первая транзакция, завершаем обе транзакции:

```
access=# select *, xmin, xmax FROM t;
 n | xmin | xmax
---+------+------
 1 |  667 |  669
```

Создаем функцию и запускаем транзакцию:

```
access=# create function test() returns integer as $$
access$# select pg_sleep(1);
access$# select count(*)::integer from t;
access$# $$ volatile language sql;
CREATE FUNCTION
```

```
access=# select (select count(*) from t) as cnt, test() from generate_series(1,5);
```

Второй терминал:

```
access=# insert into t values(1);
INSERT 0 1
```

Результат в транзакции:

```
access=# select (select count(*) from t) as cnt, test() from generate_series(1,5);
 cnt | test
-----+------
   0 |    0
   0 |    0
   0 |    1
   0 |    1
   0 |    1
```

Запускаем новую транзакцию и повторяем действия:

```
access=# truncate t;
TRUNCATE TABLE
access=# BEGIN ISOLATION LEVEL REPEATABLE READ;
BEGIN
access=# select (select count(*) from t) as cnt, test() from generate_series(1,5);
```

Второй терминал:

```
access=# insert into t values(1);
INSERT 0 1
access=#
```

Результат в транзакции:

```
access=# select (select count(*) from t) as cnt, test() from generate_series(1,5);
 cnt | test
-----+------
   0 |    0
   0 |    0
   0 |    0
   0 |    0
   0 |    0
```

Запускаем транзакцию:

```
access=# alter function test stable;
ALTER FUNCTION
access=# truncate t;
TRUNCATE TABLE
access=# BEGIN ISOLATION LEVEL READ committed;
BEGIN
access=# select (select count(*) from t) as cnt, test() from generate_series(1,5);
 cnt | test
-----+------
   0 |    0
   0 |    0
   0 |    0
   0 |    0
   0 |    0
```

Второй терминал:

```
INSERT 0 1
access=# insert into t values(1);
INSERT 0 1
access=#
```

Результат транзакции:

```
access=# select (select count(*) from t) as cnt, test() from generate_series(1,5);
 cnt | test
-----+------
   0 |    0
   0 |    0
   0 |    0
   0 |    0
   0 |    0
```

Запускаем транзакцию:

```
access=# truncate t;
TRUNCATE TABLE
access=# BEGIN ISOLATION LEVEL REPEATABLE READ;
BEGIN
access=# select (select count(*) from t) as cnt, test() from generate_series(1,5)
```

На втором терминале:

```
access=# insert into t values(1);
INSERT 0 1
access=#
```

Результат транзакции:

```
access=# select (select cou
 cnt | test
-----+------
   0 |    0
   0 |    0
   0 |    0
   0 |    0
   0 |    0
```

HOT- обновления

Обновления:

```
postgres=# create table t(id integer, s char(2000)) with (fillfactor=75);
CREATE TABLE
postgres=# create index t_id on t(id);
CREATE INDEX
postgres=# create extension pageinspect;
CREATE EXTENSION
postgres=# CREATE VIEW t_v AS
postgres-# SELECT '(0,'||lp||')' AS ctid,
postgres-#        CASE lp_flags
postgres-#          WHEN 0 THEN 'unused'
postgres-#          WHEN 1 THEN 'normal'
postgres-#          WHEN 2 THEN 'redirect to '||lp_off
postgres-#          WHEN 3 THEN 'dead'
postgres-#        END AS state,
postgres-#        t_xmin || CASE
postgres-#          WHEN (t_infomask & 256) > 0 THEN ' (c)'
postgres-#          WHEN (t_infomask & 512) > 0 THEN ' (a)'
postgres-#          ELSE ''
postgres-#        END AS xmin,
postgres-#        t_xmax || CASE
postgres-#          WHEN (t_infomask & 1024) > 0 THEN ' (c)'
postgres-#          WHEN (t_infomask & 2048) > 0 THEN ' (a)'
postgres-#          ELSE ''
postgres-#        END AS xmax,
postgres-#        CASE WHEN (t_infomask2 & 16384) > 0 THEN 't' END AS hhu,
postgres-#        CASE WHEN (t_infomask2 & 32768) > 0 THEN 't' END AS hot,
postgres-#        t_ctid
postgres-# FROM heap_page_items(get_raw_page('t',0))
postgres-# ORDER BY lp;
CREATE VIEW
postgres=# create view t_id_v as
postgres-# select itemoffset,
postgres-# ctid
postgres-# from bt_page_items('t_id',1);
CREATE VIEW
```

```
postgres=# insert into t(s) values ('A');
INSERT 0 1
postgres=# update t set s='B';
UPDATE 1
postgres=# update t set s='C';
UPDATE 1
postgres=# update t set s='D';
UPDATE 1
postgres=# select * from t_id_v;
 itemoffset | ctid
------------+-------
          1 | (0,1)
(1 row)

postgres=# select * from t_v;
 ctid  | state  |  xmin   |  xmax   | hhu | hot | t_ctid
-------+--------+---------+---------+-----+-----+--------
 (0,1) | normal | 573 (c) | 574 (c) | t   |     | (0,2)
 (0,2) | normal | 574 (c) | 575 (c) | t   | t   | (0,3)
 (0,3) | normal | 575 (c) | 576     | t   | t   | (0,4)
 (0,4) | normal | 576     | 0 (a)   |     | t   | (0,4)
(4 rows)
```

Разрыв hot- цепочки:

Запускаем транзакцию

```
postgres=# begin isolation level repeatable read;
BEGIN
postgres=# select count(*) from t;
 count
-------
     1
(1 row)
```

Открываем новый терминал:

```
postgres=# update t set s='I';
UPDATE 1
postgres=# update t set s='J';
UPDATE 1
postgres=# update t set s='K';
UPDATE 1
postgres=# select * from t_v;
 ctid  |     state     |  xmin   |  xmax   | hhu | hot | t_ctid
-------+---------------+---------+---------+-----+-----+--------
 (0,1) | redirect to 2 |         |         |     |     |
 (0,2) | normal        | 580 (c) | 581 (c) | t   | t   | (0,3)
 (0,3) | normal        | 581 (c) | 582 (c) | t   | t   | (0,4)
 (0,4) | normal        | 582 (c) | 583     | t   | t   | (0,5)
 (0,5) | normal        | 583     | 0 (a)   |     | t   | (0,5)
(5 rows)

postgres=# update t set s='L';
UPDATE 1
postgres=# select * from t_v;
 ctid  |     state     |  xmin   |  xmax   | hhu | hot | t_ctid
-------+---------------+---------+---------+-----+-----+--------
 (0,1) | redirect to 2 |         |         |     |     |
 (0,2) | normal        | 580 (c) | 581 (c) | t   | t   | (0,3)
 (0,3) | normal        | 581 (c) | 582 (c) | t   | t   | (0,4)
 (0,4) | normal        | 582 (c) | 583 (c) | t   | t   | (0,5)
 (0,5) | normal        | 583 (c) | 584     |     | t   | (1,1)
(5 rows)

postgres=# select * from t_id_v;
 itemoffset | ctid
------------+-------
          1 | (0,1)
          2 | (1,1)
(2 rows)
```

Очистка:

```
postgres=# update t set s='E';
UPDATE 1
postgres=# select * from t_v;
 ctid  |     state     |  xmin   | xmax  | hhu | hot | t_ctid
-------+---------------+---------+-------+-----+-----+--------
 (0,1) | redirect to 4 |         |       |     |     |
 (0,2) | normal        | 577     | 0 (a) |     | t   | (0,2)
 (0,3) | unused        |         |       |     |     |
 (0,4) | normal        | 576 (c) | 577   | t   | t   | (0,2)
(4 rows)

postgres=# update t set s='F';
UPDATE 1
postgres=# update t set s='G';
UPDATE 1
postgres=# select * from t_v;
 ctid  |     state     |  xmin   |  xmax   | hhu | hot | t_ctid
-------+---------------+---------+---------+-----+-----+--------
 (0,1) | redirect to 4 |         |         |     |     |
 (0,2) | normal        | 577 (c) | 578 (c) | t   | t   | (0,3)
 (0,3) | normal        | 578 (c) | 579     | t   | t   | (0,5)
 (0,4) | normal        | 576 (c) | 577 (c) | t   | t   | (0,2)
 (0,5) | normal        | 579     | 0 (a)   |     | t   | (0,5)
(5 rows)

postgres=# update t set s='H';
UPDATE 1
postgres=# select * from t_v;
 ctid  |     state     |  xmin   | xmax  | hhu | hot | t_ctid
-------+---------------+---------+-------+-----+-----+--------
 (0,1) | redirect to 5 |         |       |     |     |
 (0,2) | normal        | 580     | 0 (a) |     | t   | (0,2)
 (0,3) | unused        |         |       |     |     |
 (0,4) | unused        |         |       |     |     |
 (0,5) | normal        | 579 (c) | 580   | t   | t   | (0,2)
(5 rows)
```

Очистка

Создаем таблицу:

```
access=# CREATE TABLE t(id integer) WITH (autovacuum_enabled = off);
CREATE TABLE
access=# INSERT INTO t SELECT gen.id FROM generate_series(1,1000000) gen(id);
INSERT 0 1000000
access=# CREATE INDEX t_id ON t(id);
CREATE INDEX
access=# ALTER SYSTEM SET maintenance_work_mem = '1MB';
ALTER SYSTEM
access=# select pg_reload_conf();
 pg_reload_conf
----------------
 t
```

Обновляем строки:

```
access=#  UPDATE t SET id = id + 1;
UPDATE 1000000
```

Запускаем очистку и транзакцию:

```
access=# VACUUM VERBOSE t;
ИНФОРМАЦИЯ:  очистка "public.t"
ИНФОРМАЦИЯ:  просканирован индекс "t_id", удалено версий строк: 174472
ПОДРОБНОСТИ:  CPU: пользов.: 0.14 с, система: 0.00 с, прошло: 0.14 с
ИНФОРМАЦИЯ:  "t": удалено версий строк: 174472, обработано страниц: 772
ПОДРОБНОСТИ:  CPU: пользов.: 0.00 с, система: 0.00 с, прошло: 0.00 с
ИНФОРМАЦИЯ:  просканирован индекс "t_id", удалено версий строк: 174472
ПОДРОБНОСТИ:  CPU: пользов.: 0.14 с, система: 0.00 с, прошло: 0.13 с
ИНФОРМАЦИЯ:  "t": удалено версий строк: 174472, обработано страниц: 772
ПОДРОБНОСТИ:  CPU: пользов.: 0.00 с, система: 0.00 с, прошло: 0.00 с
ИНФОРМАЦИЯ:  просканирован индекс "t_id", удалено версий строк: 174472
ПОДРОБНОСТИ:  CPU: пользов.: 0.12 с, система: 0.00 с, прошло: 0.12 с
```

```
access=# SELECT * FROM pg_stat_progress_vacuum \gx
(0 строк)
```

Восстанавливаем значения измененного параметра:

```
access=# SELECT pg_reload_conf();
 pg_reload_conf
----------------
 t
```

Смотрим размер файла:

```
access=# SELECT pg_size_pretty(pg_table_size('t'));
 pg_size_pretty
----------------
 69 MB
```

```
access=# DELETE FROM t WHERE random() < 0.9;
DELETE 899579
access=# VACUUM t;
VACUUM
access=# SELECT pg_size_pretty(pg_table_size('t'));
 pg_size_pretty
----------------
 69 MB
```

Полная очистка:

```
access=# TRUNCATE t;
TRUNCATE TABLE
access=# INSERT INTO t SELECT gen.id FROM generate_series(1,1000000) gen(id)ж
access-# ;
ОШИБКА:  ошибка синтаксиса (примерное положение: "|")
СТРОКА 1: ...NTO t SELECT gen.id FROM generate_series(1,1000000) gen(id)|
                                                                       ^
access=# INSERT INTO t SELECT gen.id FROM generate_series(1,1000000) gen(id);
INSERT 0 1000000
access=# SELECT pg_size_pretty(pg_table_size('t'));
 pg_size_pretty
----------------
 35 MB
```

```
access=# DELETE FROM t WHERE random() < 0.9;
DELETE 900081
access=# VACUUM FULL t;
VACUUM
access=# SELECT pg_size_pretty(pg_table_size('t'));
 pg_size_pretty
----------------
 3544 kB
```

Автоочистка

Настраиваем очистку:

```
access=# alter system set autovacuum_vacuum_threshold = 0;
ALTER SYSTEM
access=# alter system set autovacuum_vacuum_scale_factor = 0.1;
ALTER SYSTEM
access=# alter system set autovacuum_naptime = '1s';
ALTER SYSTEM
access=# select pg_reload_conf();
 pg_reload_conf
----------------
 t
```

Создаем и заполняем таблицу:

```
access=# create table t(n integer);
CREATE TABLE
access=# insert into t(n) select 1 from generate_series(1,100000);
INSERT 0 100000
access=#
```

```
access=# select pg_size_pretty(pg_table_size('t'));
 pg_size_pretty
----------------
 3568 kB
```

Делаем обновление таблицы

```
access=# update t set n=n+1 where random()<0.055;
UPDATE 5503
access=# update t set n=n+1 where random()<0.055;
UPDATE 5485
access=# update t set n=n+1 where random()<0.055;
UPDATE 5539
access=# update t set n=n+1 where random()<0.055;
UPDATE 5599
access=# update t set n=n+1 where random()<0.055;
UPDATE 5535
access=# update t set n=n+1 where random()<0.055;
UPDATE 5487
access=# update t set n=n+1 where random()<0.055;
UPDATE 5524
access=# update t set n=n+1 where random()<0.055;
UPDATE 5470
access=# update t set n=n+1 where random()<0.055;
UPDATE 5449
access=# update t set n=n+1 where random()<0.055;
UPDATE 5382
access=# update t set n=n+1 where random()<0.055;
UPDATE 5545
access=# update t set n=n+1 where random()<0.055;
UPDATE 5445
access=# update t set n=n+1 where random()<0.055;
UPDATE 5379
access=# update t set n=n+1 where random()<0.055;
UPDATE 5438
access=# update t set n=n+1 where random()<0.055;
UPDATE 5433
access=# update t set n=n+1 where random()<0.055;
UPDATE 5580
access=# update t set n=n+1 where random()<0.055;
UPDATE 5507
access=# update t set n=n+1 where random()<0.055;
UPDATE 5509
access=# update t set n=n+1 where random()<0.055;
UPDATE 5441
access=# update t set n=n+1 where random()<0.055;
```

Оценка разности

```
access=# SELECT * FROM pg_stat_all_tables WHERE relname='t' \gx
-[ RECORD 1 ]-------+------------------------------
relid               | 16540
schemaname          | public
relname             | t
seq_scan            | 20
seq_tup_read        | 2000000
idx_scan            |
idx_tup_fetch       |
n_tup_ins           | 100000
n_tup_upd           | 109709
n_tup_del           | 0
n_tup_hot_upd       | 78509
n_live_tup          | 100000
n_dead_tup          | 5459
n_mod_since_analyze | 0
last_vacuum         |
last_autovacuum     | 2021-10-28 15:02:42.54631+03
last_analyze        |
last_autoanalyze    | 2021-10-28 15:02:45.597224+03
vacuum_count        | 0
autovacuum_count    | 4
analyze_count       | 0
autoanalyze_count   | 11


access=# SELECT pg_size_pretty(pg_table_size('t'));
 pg_size_pretty
----------------
 4416 kB
```

Заморозка

Запуск транзакции:

```
access=# begin;
BEGIN
access=# create table t(n integer);
CREATE TABLE
access=# copy t from stdin with freeze;
Вводите данные для копирования, разделяя строки переводом строки.
Закончите ввод строкой '\.' или сигналом EOF.
>> 1
>> 2
>> 3
>> \.
COPY 3
access=# commit;
COMMIT
```

```
access=# CREATE VIEW t_vs AS
access-# SELECT '(0,'||lp||')' as ctid,
access-# CASE lp_flags
access-# WHEN 0 THEN 'unused'
access-# WHEN 1 THEN 'normal'
access-# WHEN 3 THEN 'dead'
access-# END AS state,
access-# t_xmin AS xmin,
access-# age(t_xmin) AS xmin_age,
access-# CASE WHEN (t_infomask & 256) > 0 THEN 't' END AS xmin_c,
access-# CASE WHEN (t_infomask & 512) > 0 THEN 't' END AS xmin_a,
access-# t_xmax AS xmax,
access-# t_ctid
access-# FROM heap_page_items(get_raw_page('t',0))
access-# ORDER BY lp;
CREATE VIEW
access=# select * from t_vs;
 ctid  | state  | xmin | xmin_age | xmin_c | xmin_a | xmax | t_ctid
-------+--------+------+----------+--------+--------+------+--------
 (0,1) | normal |  727 |        2 | t      | t      |    0 | (0,1)
 (0,2) | normal |  727 |        2 | t      | t      |    0 | (0,2)
 (0,3) | normal |  727 |        2 | t      | t      |    0 | (0,3)
```

Изоляция:

Транзакция на втором терминале

```
access=# begin isolation level repeatable read;
BEGIN
access=# select txid_current();
 txid_current
--------------
          729
```

Запускаем транзакцию на первом терминале:

```
access=# begin;
BEGIN
access=# truncate t;
TRUNCATE TABLE
access=# copy t from stdin with freeze;
Вводите данные для копирования, разделяя строки переводом строки.
Закончите ввод строкой '\.' или сигналом EOF.
>> 10
>> 20
>> 30
>> \.
COPY 3
access=# commit;
COMMIT
```

На второй транзакции:

```
access=# select * from t;
 n
----
 10
 20
 30
```

```
access=# ALTER SYSTEM SET autovacuum = off;
ALTER SYSTEM
access=# ALTER SYSTEM SET vacuum_freeze_min_age = 1000;
ALTER SYSTEM
access=# ALTER SYSTEM SET vacuum_freeze_table_age = 10000;
ALTER SYSTEM
access=#  ALTER SYSTEM SET autovacuum_freeze_max_age = 100000;
ALTER SYSTEM
access=# SELECT datname, datfrozenxid, age(datfrozenxid) FROM pg_databasex

access-# ;
ОШИБКА:  отношение "pg_database|" не существует
СТРОКА 1: ...ECT datname, datfrozenxid, age(datfrozenxid) FROM pg_databas.
..
                                                             ^
access=# SELECT datname, datfrozenxid, age(datfrozenxid) FROM pg_database;

    datname    | datfrozenxid | age
---------------+--------------+-----
 postgres      |          562 | 169
 access        |          562 | 169
 template0     |          562 | 169
 template1     |          562 | 169
 data_lowlevel |          562 | 169
```

```
access=# VACUUM t;
VACUUM
access=# SELECT datname, datfrozenxid, age(datfrozenxid) FROM pg_database;

    datname    | datfrozenxid | age
---------------+--------------+-----
 postgres      |          562 | 169
 access        |          562 | 169
 template0     |          562 | 169
 template1     |          562 | 169
```

Буферный кэш

Создаем, заполняем таблицу и смотрим размеры:

```
ostgres=# CREATE TABLE t(n integer);
REATE TABLE
ostgres=# INSERT INTO t SELECT 1 FROM generate_series(1,10000);
NSERT 0 10000
ostgres=# SELECT setting FROM pg_settings WHERE name = 'block_size';
setting
--------
8192
1 row)

ostgres=# SELECT pg_table_size('t') / 8192;
?column?
---------
      48
1 row)

ostgres=# SELECT pg_relation_size('t','main') / 8192;
?column?
---------
      45
1 row)
```

```
postgres=# SELECT pg_relation_size('t','fsm') / 8192;
 ?column?
----------
        3
(1 row)
```

```
postgres=# CREATE EXTENSION pg_buffercache;
CREATE EXTENSION
postgres=# SELECT CASE relforknumber
postgres-#          WHEN 0 THEN 'main'
postgres-#          WHEN 1 THEN 'fsm'
postgres-#          WHEN 2 THEN 'vm'
postgres-#        END relfork,
postgres-#        count(*)
postgres-# FROM   pg_buffercache b,
postgres-#        pg_class c
postgres-# WHERE  b.reldatabase = (
postgres(#          SELECT oid FROM pg_database WHERE datname = current_database
()
postgres(#        )
postgres-# AND    c.oid = 't'::regclass
postgres-# AND    b.relfilenode = c.relfilenode
postgres-# GROUP BY 1;
 relfork | count
---------+-------
 fsm     |     2
 main    |    45
(2 rows)
```

Количество грязных буферов:

```
postgres=# select count(*) from pg_buffercache b where isdirty;
 count
-------
     0
(1 row)
```

Контрольная строка:

```
postgres=# select count(*) from pg_buffercache b where isdirty;
 count
-------
     0
(1 row)
```

Журнал предзаписи

```
postgres=# SELECT pg_current_wal_insert_lsn();
 pg_current_wal_insert_lsn
---------------------------
 0/8865B08
(1 row)

postgres=# CREATE TABLE t(id integer PRIMARY KEY);
CREATE TABLE
postgres=# INSERT INTO t VALUES (1),(2),(3);
INSERT 0 3
postgres=# SELECT pg_current_wal_insert_lsn();
 pg_current_wal_insert_lsn
---------------------------
 0/8878B70
(1 row)

postgres=# SELECT '0/2C480D70'::pg_lsn - '0/2C46450C'::pg_lsn;
 ?column?
----------
   116836
(1 row)

postgres=# SELECT pg_walfile_name('0/2C46450C');
     pg_walfile_name
--------------------------
 00000001000000000000002C
(1 row)
```

Контрольная точка

Настройки контрольной точки:

```
postgres=# ALTER SYSTEM SET checkpoint_timeout = '30s';
ALTER SYSTEM
postgres=# ALTER SYSTEM SET min_wal_size = '16MB';
ALTER SYSTEM
postgres=# ALTER SYSTEM SET max_wal_size = '16MB';
ALTER SYSTEM
postgres=# SELECT pg_reload_conf();
 pg_reload_conf
----------------
 t
(1 row)
```

Нагрузка:

```
postgres=# SELECT pg_stat_reset_shared('bgwriter');
 pg_stat_reset_shared
----------------------

(1 row)

postgres=# SELECT pg_current_wal_insert_lsn();
 pg_current_wal_insert_lsn
---------------------------
 0/8878C58
(1 row)

postgres=# SELECT pg_current_wal_insert_lsn();
 pg_current_wal_insert_lsn
---------------------------
 0/8878C58
(1 row)

postgres=# SELECT pg_size_pretty('0/327F3210'::pg_lsn - '0/2D0B6BF8'::pg_lsn);
 pg_size_pretty
----------------
 87 MB
(1 row)

postgres=# SELECT checkpoints_timed, checkpoints_req FROM pg_stat_bgwriter;
 checkpoints_timed | checkpoints_req
-------------------+-----------------
                 2 |               0
(1 row)
```

Возврат настроек:

```
postgres=# ALTER SYSTEM RESET ALL;
ALTER SYSTEM
postgres=#  SELECT pg_reload_conf();
 pg_reload_conf
----------------
 t
(1 row)
```

Настройка журнала

ON:

```
postgres=# ALTER SYSTEM SET full_page_writes = on;
ALTER SYSTEM
postgres=# ALTER SYSTEM SET wal_compression = on;
ALTER SYSTEM
postgres=#  SELECT pg_reload_conf();
 pg_reload_conf
----------------
 t
(1 row)

postgres=# checkpoint;
CHECKPOINT
postgres=# SELECT pg_current_wal_insert_lsn();
 pg_current_wal_insert_lsn
---------------------------
 0/8878F50
(1 row)

postgres=# SELECT pg_current_wal_insert_lsn();
 pg_current_wal_insert_lsn
---------------------------
 0/8878F50
(1 row)

postgres=# SELECT pg_size_pretty('0/3623DD38'::pg_lsn - '0/35A67878'::pg_lsn);
 pg_size_pretty
----------------
 8025 kB
(1 row)
```

OFF:

```
postgres=# ALTER SYSTEM SET full_page_writes = off;
ALTER SYSTEM
postgres=#  SELECT pg_reload_conf();
 pg_reload_conf
----------------
 t
(1 row)

postgres=# checkpoint;
CHECKPOINT
postgres=# SELECT pg_current_wal_insert_lsn();
 pg_current_wal_insert_lsn
---------------------------
 0/8878E48
(1 row)

postgres=# SELECT pg_current_wal_insert_lsn();
 pg_current_wal_insert_lsn
---------------------------
 0/8878E48
(1 row)

postgres=# SELECT pg_size_pretty('0/35A677C0'::pg_lsn - '0/347BC348'::pg_lsn);
 pg_size_pretty
----------------
 19 MB
(1 row)

postgres=# SHOW data_checksums;
 data_checksums
----------------
 off
(1 row)
```

Сжатие:

```
postgres=# SHOW full_page_writes;
 full_page_writes
------------------
 on
(1 row)

postgres=# checkpoint;
CHECKPOINT
postgres=# SELECT pg_current_wal_insert_lsn();
 pg_current_wal_insert_lsn
---------------------------
 0/8878D40
(1 row)

postgres=# SELECT pg_current_wal_insert_lsn();
 pg_current_wal_insert_lsn
---------------------------
 0/8878D40
(1 row)

postgres=# SELECT pg_size_pretty('0/347BC290'::pg_lsn - '0/33429A54'::pg_lsn);
 pg_size_pretty
----------------
 20 MB
(1 row)
```

Блокировки объектов

Создаем базу данных, таблицу и заносим данные:

```
postgres=# create database locks_obj;
CREATE DATABASE
postgres=# \c locks_obj
Вы подключены к базе данных "locks_obj" как пользователь "postgres".
locks_obj=# create table accounts(acc_no integer primary key, amount);
ОШИБКА:  ошибка синтаксиса (примерное положение: ")")
СТРОКА 1: create table accounts(acc_no integer primary key, amount);
                                                             ^
locks_obj=# create table accounts(acc_no integer primary key, amount numeric);
CREATE TABLE
locks_obj=# insert into accounts values (1,1000.00), (2, 2000.00), (3, 3000.00);
INSERT 0 3
```

Начинаем транзакцию, выводим одну строку:

```
locks_obj=# \c locks_obj
Вы подключены к базе данных "locks_obj" как пользователь "postgres".
locks_obj=# select pg_backend_pid();
 pg_backend_pid
----------------
         345996
(1 строка)
```

```
locks_obj=# begin;
BEGIN
locks_obj=*# select * from accounts where acc_no=1;
 acc_no | amount
--------+---------
      1 | 1000.00
(1 строка)
```

Блокировка вне транзакции:

```
locks_obj=# select locktype, relation::Regclass, virtualxid as virtxid, transactioni
d as xid,mode,granted from pg_locks where pid = 345996;
  locktype  | relation | virtxid | xid |      mode       | granted
------------+----------+---------+-----+-----------------+---------
 relation   | pg_locks |         |     | AccessShareLock | t
 virtualxid |          | 4/7117  |     | ExclusiveLock   | t
(2 строки)
```

Блокировка в транзакции:

```
locks_obj=# begin;
BEGIN
locks_obj=*# select * from accounts where acc_no=1;
 acc_no | amount
--------+---------
      1 | 1000.00
(1 строка)

locks_obj=*#
locks_obj=*# select locktype, relation::Regclass, virtualxid as virtxid, transaction
id as xid,mode,granted from pg_locks where pid = 345996;
  locktype  |   relation    | virtxid | xid |      mode       | granted
------------+---------------+---------+-----+-----------------+---------
 relation   | pg_locks      |         |     | AccessShareLock | t
 relation   | accounts_pkey |         |     | AccessShareLock | t
 relation   | accounts      |         |     | AccessShareLock | t
 virtualxid |               | 4/7119  |     | ExclusiveLock   | t
(4 строки)
```

Выполняем транзакцию, читаем одну строку, блокируем:

```
locks_obj=# begin;
BEGIN
locks_obj=*# select * from accounts where acc_no=1;
 acc_no | amount
--------+---------
      1 | 1000.00
(1 строка)

locks_obj=*#
locks_obj=*# select locktype, relation::Regclass, virtualxid as virtxid, transaction
id as xid,mode,granted from pg_locks where pid = 345996;
  locktype  |   relation    | virtxid | xid |      mode       | granted
------------+---------------+---------+-----+-----------------+---------
 relation   | pg_locks      |         |     | AccessShareLock | t
 relation   | accounts_pkey |         |     | AccessShareLock | t
 relation   | accounts      |         |     | AccessShareLock | t
 virtualxid |               | 4/7119  |     | ExclusiveLock   | t
(4 строки)
```

Изменяем параметры:

```
locks_obj=# alter system set log_lock_waits=on;
ALTER SYSTEM
locks_obj=# alter system set deadlock_timeout='100ms';
ALTER SYSTEM
locks_obj=# select pg_reload_conf();
 pg_reload_conf
----------------
 t
(1 строка)
```

Первый терминал:

```
locks_obj=# begin;
BEGIN
locks_obj=*# update accounts set amount=10.00 where acc_no=1;
UPDATE 1
```

Второй терминал:

```
locks_obj=# begin;
BEGIN
locks_obj=*# update accounts set amount=100.00 where acc_no=1;
```

Первый терминал:

```
locks_obj=*# SELECT PG_SLEEP(1);
 pg_sleep
----------

(1 строка)

locks_obj=*# COMMIT;
COMMIT
```

На втором терминале завершаем транзакцию.

Блокировки строк

Создаем представление:

```
locks_obj=# CREATE VIEW locks AS SELECT pid, locktype, CASE locktype WHEN 'relat
ion' THEN relation::REGCLASS::text WHEN 'virtualxid' THEN virtualxid::text WHEN
'transactionid' THEN transactionid::text WHEN 'tuple' THEN relation::REGCLASS::t
ext||':'||tuple::text END AS lockid, mode, granted FROM pg_locks;
CREATE VIEW
```

Первый терминал:

```
locks_obj=# begin;
BEGIN
locks_obj=*# select txid_current(), pg_backend_pid();
 txid_current | pg_backend_pid
--------------+----------------
          606 |         356219
(1 строка)

locks_obj=*# update accounts set amount=amount+100.00 where acc_no=1;
UPDATE 1
```

Второй терминал:

```
locks_obj=# begin;
BEGIN
locks_obj=*# select txid_current(), pg_backend_pid();
 txid_current | pg_backend_pid
--------------+----------------
          607 |         356862
(1 строка)

locks_obj=*# UPDATE accounts SET amount = amount + 100.00 WHERE acc_no = 1;
```

Третий терминал:

```
locks_obj=# begin;
BEGIN
locks_obj=*# SELECT txid_current(), pg_backend_pid();
 txid_current | pg_backend_pid
--------------+----------------
          608 |         357061
(1 строка)

locks_obj=*# UPDATE accounts SET amount = amount + 100.00 WHERE acc_no = 1;
```

Блокировка для первой транзакции:

```
locks_obj=*# SELECT * FROM locks WHERE pid =356219;
  pid   |   locktype    |    lockid     |       mode       | granted
--------+---------------+---------------+------------------+---------
 356219 | relation      | accounts_pkey | RowExclusiveLock | t
 356219 | relation      | accounts      | RowExclusiveLock | t
 356219 | virtualxid    | 6/1456        | ExclusiveLock    | t
 356219 | tuple         | accounts:9    | ExclusiveLock    | t
 356219 | transactionid | 609           | ExclusiveLock    | t
 356219 | transactionid | 608           | ShareLock        | f
(6 строк)
```

Блокировка для второй транзакции:

```
locks_obj=*# SELECT * FROM locks WHERE pid =356862;
  pid   |   locktype    |    lockid     |       mode       | granted
--------+---------------+---------------+------------------+---------
 356862 | relation      | accounts_pkey | RowExclusiveLock | t
 356862 | relation      | accounts      | RowExclusiveLock | t
 356862 | virtualxid    | 8/143         | ExclusiveLock    | t
 356862 | transactionid | 610           | ExclusiveLock    | t
 356862 | tuple         | accounts:9    | ExclusiveLock    | f
(5 строк)
```

Блокировка для третьей:

```
  pid   |   locktype    |          lockid           |       mode       | granted
--------+---------------+---------------------------+------------------+---------
 357061 | relation      | pg_stat_activity          | AccessShareLock  | t
 357061 | relation      | pg_locks                  | AccessShareLock  | t
 357061 | relation      | locks                     | AccessShareLock  | t
 357061 | relation      | accounts_pkey             | RowExclusiveLock | t
 357061 | relation      | accounts                  | RowExclusiveLock | t
 357061 | virtualxid    | 9/21                      | ExclusiveLock    | t
 357061 | relation      | pg_database_oid_index     | AccessShareLock  | t
 357061 | relation      | pg_authid_oid_index       | AccessShareLock  | t
 357061 | transactionid | 608                       | ExclusiveLock    | t
 357061 | relation      | pg_authid_rolname_index   | AccessShareLock  | t
 357061 | relation      | pg_database               | AccessShareLock  | t
 357061 | relation      | pg_authid                 | AccessShareLock  | t
 357061 | relation      | pg_database_datname_index | AccessShareLock  | t
(13 строк)
```

Посмотрим блокирующие процессы:

```
locks_obj=*# SELECT pid, wait_event_type, wait_event, pg_blocking_pids(pid) FROM
 pg_stat_activity WHERE backend_type = 'client backend';
  pid   | wait_event_type |  wait_event   | pg_blocking_pids
--------+-----------------+---------------+------------------
 356366 | Client          | ClientRead    | {}
 356200 |                 |               | {}
 133337 | Client          | ClientRead    | {}
 356219 | Lock            | transactionid | {357061}
 356862 | Lock            | tuple         | {356219}
 357061 |                 |               | {}
(6 строк)
```

Взаимодействие трех транзакций

Первая транзакция:

```
locks_obj=# begin;
BEGIN
locks_obj=*# UPDATE accounts SET amount = amount - 100.00 WHERE acc_no = 1;
UPDATE 1
locks_obj=*#
```

Вторая транзакция:

```
locks_obj=# begin;
BEGIN
locks_obj=*# UPDATE accounts SET amount = amount - 100.00 WHERE acc_no = 2;
UPDATE 1
locks_obj=*#
```

Третья транзакция:

```
locks_obj=# begin;
BEGIN
locks_obj=*# UPDATE accounts SET amount = amount - 100.00 WHERE acc_no = 3;
UPDATE 1
locks_obj=*#
```

Первая транзакция:

```
locks_obj=*# UPDATE accounts SET amount = amount + 100.00 WHERE acc_no = 2;
```

Вторая транзакция:

```
locks_obj=*# UPDATE accounts SET amount = amount + 100.00 WHERE acc_no = 3;
```

Третья транзакция, обновление строки на 1 и 2 транзакции завершатся:

```
locks_obj=*# UPDATE accounts SET amount = amount + 100.00 WHERE acc_no = 1;
ОШИБКА:  обнаружена взаимоблокировка
ПОДРОБНОСТИ:  Процесс 357061 ожидает в режиме ShareLock блокировку "транзакция 6
14"; заблокирован процессом 356219.
Процесс 356219 ожидает в режиме ShareLock блокировку "транзакция 615"; заблокиро
ван процессом 356862.
Процесс 356862 ожидает в режиме ShareLock блокировку "транзакция 616"; заблокиро
ван процессом 357061.
ПОДСКАЗКА:  Подробности запроса смотрите в протоколе сервера.
КОНТЕКСТ:  при изменении кортежа (0,11) в отношении "accounts"
```

Взаимоблокировка двух операций UPDATE

Первая транзакция:

```
locks_obj=# begin;
BEGIN
locks_obj=*# DECLARE c1 CURSOR FOR SELECT * FROM accounts ORDER BY acc_no FOR UP
DATE;
DECLARE CURSOR
```

Вторая транзакция:

```
locks_obj=# begin;
BEGIN
locks_obj=*# DECLARE c2 CURSOR FOR SELECT * FROM accounts ORDER BY acc_no DESC F
OR UPDATE;
DECLARE CURSOR
```

Первая транзакция:

```
locks_obj=*# FETCH c1;
 acc_no | amount
--------+--------
      1 | 300.00
(1 строка)
```

Вторая транзакция:

```
locks_obj=*# FETCH c2;
 acc_no | amount
--------+--------
      3 | 3000.00
(1 строка)

locks_obj=*#
```

Первая транзакция:

```
locks_obj=*# FETCH c1;
 acc_no | amount
--------+---------
      2 | 2000.00
(1 строка)
```

Вторая транзакция (задержка):

```
locks_obj=*#
locks_obj=*# FETCH c2;
```

Первая транзакция:

```
locks_obj=*# FETCH c1;
ОШИБКА:  обнаружена взаимоблокировка
ПОДРОБНОСТИ:  Процесс 356219 ожидает в режиме ShareLock блокировку "транзакция 6
18"; заблокирован процессом 356862.
Процесс 356862 ожидает в режиме ShareLock блокировку "транзакция 617"; заблокиро
ван процессом 356219.
ПОДСКАЗКА:  Подробности запроса смотрите в протоколе сервера.
КОНТЕКСТ:  при блокировке кортежа (0,3) в отношении "accounts"
locks_obj=!#
```

Появление на второй:

```
 acc_no | amount
--------+---------
      3 | 3000.00
(1 строка)

locks_obj=*#
```