Мамонов Антон 3ИСиП-19-1

База данных и таблица

```
postgres=# CREATE DATABASE admin_monitoring;
CREATE DATABASE
postgres=# \c admin_monitoring
Вы подключены к базе данных "admin_monitoring" как пользователь "postgres".
admin_monitoring=# CREATE TABLE t(n numeric);
CREATE TABLE
admin_monitoring=# INSERT INTO t SELECT 1 FROM generate_series(1,1000);
INSERT 0 1000
admin_monitoring=# DELETE FROM t;
DELETE 1000
```

Статистика обращений

```
admin_monitoring=# \c
Вы подключены к базе данных "admin_monitoring" как пользователь "postgres".
admin_monitoring=# \x
Расширенный вывод включён.
admin_monitoring=# SELECT * FROM pg_stat_all_tables WHERE relid='t'::regclass;
-[ RECORD 1 ]-------+------------------------------
relid               | 16444
schemaname          | public
relname             | t
seq_scan            | 1
seq_tup_read        | 1000
idx_scan            |
idx_tup_fetch       |
n_tup_ins           | 1000
n_tup_upd           | 0
n_tup_del           | 1000
n_tup_hot_upd       | 0
n_live_tup          | 0
n_dead_tup          | 0
n_mod_since_analyze | 0
n_ins_since_vacuum  | 0
last_vacuum         |
last_autovacuum     | 2021-10-26 23:00:17.856142+03
last_analyze        |
last_autoanalyze    | 2021-10-26 23:00:17.95234+03
vacuum_count        | 0
autovacuum_count    | 1
analyze_count       | 0
autoanalyze_count   | 1
```

Мы вставили 1000 строк (n_tup_ins = 1000), удалили 1000 строк (n_tup_del = 1000).

После этого не осталось активных версий строк (n_live_tup = 0), все 1000 строк не актуальны на текущий момент (n_dead_tup = 1000).

Очистка

```
admin_monitoring=# VACUUM;
VACUUM
admin_monitoring=# SELECT * FROM pg_stat_all_tables WHERE relid='t'::regclass;
-[ RECORD 1 ]-------+------------------------------
relid               | 16444
schemaname          | public
relname             | t
seq_scan            | 1
seq_tup_read        | 1000
idx_scan            |
idx_tup_fetch       |
n_tup_ins           | 1000
n_tup_upd           | 0
n_tup_del           | 1000
n_tup_hot_upd       | 0
n_live_tup          | 0
n_dead_tup          | 0
n_mod_since_analyze | 0
n_ins_since_vacuum  | 0
last_vacuum         | 2021-10-26 23:03:57.95087+03
last_autovacuum     | 2021-10-26 23:00:17.856142+03
last_analyze        |
last_autoanalyze    | 2021-10-26 23:00:17.95234+03
vacuum_count        | 1
autovacuum_count    | 1
analyze_count       | 0
autoanalyze_count   | 1
```

Неактуальные версии строк убраны при очистке (n_dead_tup = 0), очистка обрабатывала таблицу один раз (vacuum_count = 1).

Взаимоблокировка

```
admin_monitoring=# INSERT INTO t VALUES (1),(2);
INSERT 0 2
```

Одна транзакция блокирует первую строку таблицы...

```
postgres=# \c admin_monitoring
Вы подключены к базе данных "admin_monitoring" как пользователь "postgres".
admin_monitoring=# BEGIN;
BEGIN
admin_monitoring=*# UPDATE t SET n=10 WHERE n=1;
UPDATE 1
```

Затем другая транзакция блокирует вторую строку...

```
postgres=# \c admin_monitoring
Вы подключены к базе данных "admin_monitoring" как пользователь "postgres".
admin_monitoring=# BEGIN;
BEGIN
admin_monitoring=*# UPDATE t SET n=200 WHERE n=2;
UPDATE 1
```

Теперь первая транзакция пытается изменить вторую строку и ждет ее освобождения...

А вторая транзакция пытается изменить первую строку...

```
Вы подключены к базе данных "admin_monitoring" как пользователь "postgres".
admin_monitoring=# BEGIN;
BEGIN
admin_monitoring=*# UPDATE t SET n=10 WHERE n=1;
UPDATE 1
admin_monitoring=*# UPDATE t SET n=20 WHERE n=2;
UPDATE 1
```

```
admin_monitoring=# BEGIN;
BEGIN
admin_monitoring=*# UPDATE t SET n=200 WHERE n=2;
UPDATE 1
admin_monitoring=*# UPDATE t SET n=100 WHERE n=1;
ОШИБКА:  обнаружена взаимоблокировка
ПОДРОБНОСТИ:  Процесс 11148 ожидает в режиме ShareLock блокировку "транзакция 781"; заблокирован
процессом 3720.
Процесс 3720 ожидает в режиме ShareLock блокировку "транзакция 782"; заблокирован процессом 1114
.
ПОДСКАЗКА:  Подробности запроса смотрите в протоколе сервера.
КОНТЕКСТ:  при изменении кортежа (0,1) в отношении "t"
```

...и происходит взаимоблокировка.