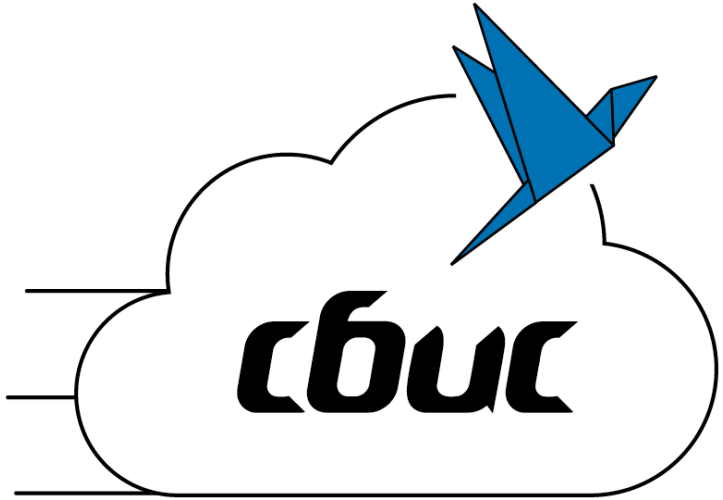


PostgreSQL для начинающих

#7: блокировки

Кирилл Боровиков / Компания «Тензор», технический директор / explain.tensor.ru, sbis.ru

Блокировки

Блокировки

Ограничение конкурентного доступа

инициируется приложением или самой СУБД

для определения строгого порядка доступа к ресурсу

https://postgrespro.ru/docs/postgresql/15/explicit-locking

Блокировки

Накладывается …

на определенный ресурс

реальный объект базы (таблица, индекс, страница данных, запись, …)

или некоторый «виртуальный» ID

Блокировки

Накладывается …

всегда последовательно

невозможно ничего заблокировать «одновременно»

если удерживать блокировку долго, это провоцирует очереди

Блокировки

Накладывается ...

изолированно для каждого соединения/процесса

невозможно управлять «чужими» блокировками

но можно «попросить» сервер «разобраться с кем надо»

```
SELECT pg_cancel_backend(pid);
-- прерывание запроса в процессе
SELECT pg_terminate_backend(pid);
-- завершение процесса
```

https://postgrespro.ru/docs/postgresql/15/functions-admin#FUNCTIONS-ADMIN-SIGNAL

Блокировки

Накладывается ...

в момент возникновения потребности

выполнение определенного запроса в ходе транзакции

изменение конкретной записи в ходе запроса

Блокировки

Снимается **вне транзакции**

- при завершении процесса/обрыве соединения
- при поступлении явной команды на снятие блокировки

```
pg_advisory_unlock*()
```

Блокировки

Снимается внутри транзакции

- при ее завершении (**COMMIT**/**ROLLBACK**)
- при откате к точке восстановления (**ROLLBACK TO sp**)
  - для блокировок, наложенных после **SAVEPOINT sp**
- при завершении **EXCEPTION**-блока в хранимой процедуре
  - для блокировок, наложенных в нем
  - только при возникновении этого исключения

Варианты блокировок

Table-Level Locks

уровня таблиц

Row-Level Locks

уровня записей

Page-Level Locks

уровня страниц

Advisory Locks

рекомендательные блокировки

deadlocks

взаимоблокировки

Table-Level

Блокируемый ресурс – таблица

- может блокировать и чтение, и запись
- распространяется и на записи **pg_catalog.***

Возникновение

- автоматически при части операций
- при вызове команды **LOCK**

https://postgrespro.ru/docs/postgresql/15/explicit-locking#TABLE-LOCK-COMPATIBILITY

Table-Level конфликты режимов

Режим блокировки – просто имя

отличаются только предопределенным набором конфликтов

Запрашиваемый режим		Текущий режим							
		AS	RS	RE	SUE	S	SRE	E	AE
Access Share	AS								x
Row Share	RS							x	x
Row Exclusive	RE					x	x	x	x
Share Update Exclusive	SUE				x	x	x	x	x
Share	S			x	x		x	x	x
Share Row Exclusive	SRE			x	x	x	x	x	x
Exclusive	E		x	x	x	x	x	x	x
Access Exclusive	AE	x	x	x	x	x	x	x	x

Table-Level конфликты команд

Запускаемая команда		Исполняющаяся команда							
		AS	RS	RE	SUE	S	SRE	E	AE
SELECT	AS								x
SELECT ... FOR ...	RS							x	x
INSERT / UPDATE / DELETE / MERGE	RE					x	x	x	x
VACUUM / ANALYZE CREATE INDEX CONCURRENTLY REINDEX CONCURRENTLY CREATE STATISTICS COMMENT ON ALTER INDEX / ALTER TABLE [*]	SUE				x	x	x	x	x
CREATE INDEX	S			x	x		x	x	x
CREATE TRIGGER ALTER TABLE [*]	SRE			x	x	x	x	x	x
REFRESH MATERIALIZED VIEW CONCURRENTLY	E		x	x	x	x	x	x	x
DROP TABLE VACUUM FULL TRUNCATE REINDEX CLUSTER REFRESH MATERIALIZED VIEW ALTER INDEX / ALTER TABLE [*] LOCK	AE	x	x	x	x	x	x	x	x

Table-Level

Как не ждать (совсем)

LOCK ... NOWAIT

https://postgrespro.ru/docs/postgresql/15/sql-lock

```
ALTER TABLE x ...;
-- мы ждем все предстоящие SELECT/DML
-- все последующие SELECT/DML ждут нас
```

```
BEGIN;
LOCK x IN ACCESS EXCLUSIVE MODE NOWAIT;
-- либо успешно захватим блокировку
-- либо сразу выпадем в EXCEPTION
ALTER TABLE x ...;
COMMIT;
```

Table-Level

Как не ждать (хотя иногда можно чуток)

lock_timeout

https://postgrespro.ru/docs/postgresql/15/runtime-config-client#GUC-LOCK-TIMEOUT

```
BEGIN;
LOCK x IN ACCESS EXCLUSIVE MODE NOWAIT;
-- сразу EXCEPTION при конфликте
ALTER TABLE x ...;
COMMIT;
```

```
BEGIN;
SET LOCAL lock_timeout = 100; -- ms
LOCK x IN ACCESS EXCLUSIVE MODE;
-- EXCEPTION только после 100мс ожидания
ALTER TABLE x ...;
COMMIT;
```

Table-Level

Как не ждать («мерзкие клиентс-с-с-ы!»)

idle_in_transaction_session_timeout

https://postgrespro.ru/docs/postgresql/15/runtime-config-client#GUC-IDLE-IN-TRANSACTION-SESSION-TIMEOUT

Row-Level

Блокируемый ресурс – строка таблицы

может блокировать только запись (той же строки)

не влияет на чтение

Возникновение

автоматически при **UPDATE / DELETE / MERGE**

при вызове команды **SELECT ... FOR ...**

https://postgrespro.ru/docs/postgresql/15/explicit-locking#LOCKING-ROWS

Row-Level конфликты команд

Запускаемая команда		Исполняющаяся команда			
		RS	S	SRE	E
SELECT ... FOR KEY SHARE	RS				x
SELECT ... FOR SHARE	S			x	x
SELECT ... FOR NO KEY UPDATE UPDATE [*]	SRE		x	x	x
SELECT ... FOR UPDATE DELETE UPDATE (cols of UNIQUE INDEX)	E	x	x	x	x

Row-Level

Как не ждать

```
SELECT ... FOR ... [ NOWAIT | SKIP LOCKED ]
```

https://postgrespro.ru/docs/postgresql/15/sql-select#SQL-FOR-UPDATE-SHARE

Page-Level

Блокируемый ресурс – страница таблицы/индекса

- может блокировать и чтение, и запись

Возникновение

- кратковременные системные операции
  - сброс страницы данных на диск, сплит дерева индекса, ...

https://postgrespro.ru/docs/postgresql/15/explicit-locking#LOCKING-PAGES

Page-Level конфликты режимов

Запрашиваемый режим		Текущий режим	
		S	E
Share	S		x
Exclusive	E	x	x

снимается сразу после завершения операции

Advisory

Блокируемый ресурс – произвольное число (64-bit)

```
(bigint) / (integer, integer)
```

```
pg_advisory_*(...) / pg_try_advisory_*(...)
```

https://postgrespro.ru/docs/postgresql/15/explicit-locking#ADVISORY-LOCKS

https://postgrespro.ru/docs/postgresql/15/functions-admin#FUNCTIONS-ADVISORY-LOCKS

Advisory конфликты режимов

Запрашиваемый режим		Текущий режим	
		S	E
Share	S		x
Exclusive	E	x	x

Advisory

Преимущества

можно блокировать «то, чего нет»

можно не ждать занятую блокировку

атомарность CAS-проверки наложения

можно «привязать» к транзакции, а можно – к соединению

`pg_try_advisory_xact_lock` / `pg_try_advisory_lock`

https://habr.com/ru/companies/tensor/articles/488024/

Advisory

Возможные проблемы

слишком много блокировок

«И рекомендательные, и обычные блокировки сохраняются в области общей памяти, размер которой определяется параметрами конфигурации **max_locks_per_transaction** и **max_connections**. Важно, чтобы этой памяти было достаточно, так как **в противном случае сервер не сможет выдать никакую блокировку**. Таким образом, число рекомендательных блокировок, которые может выдать сервер, ограничивается обычно десятками или сотнями тысяч в зависимости от конфигурации сервера.»

https://postgrespro.ru/docs/postgresql/15/explicit-locking#ADVISORY-LOCKS

Advisory

Возможные проблемы

мультиналожение одной advisory-блокировки

«Когда поступает несколько запросов на блокировку сеансового уровня, **они накапливаются, так что если один идентификатор ресурса был заблокирован три раза, должны поступить три запроса на освобождение блокировки**, чтобы ресурс был разблокирован до завершения сеанса.»

https://postgrespro.ru/docs/postgresql/15/functions-admin#FUNCTIONS-ADVISORY-LOCKS

Advisory

Возможные проблемы

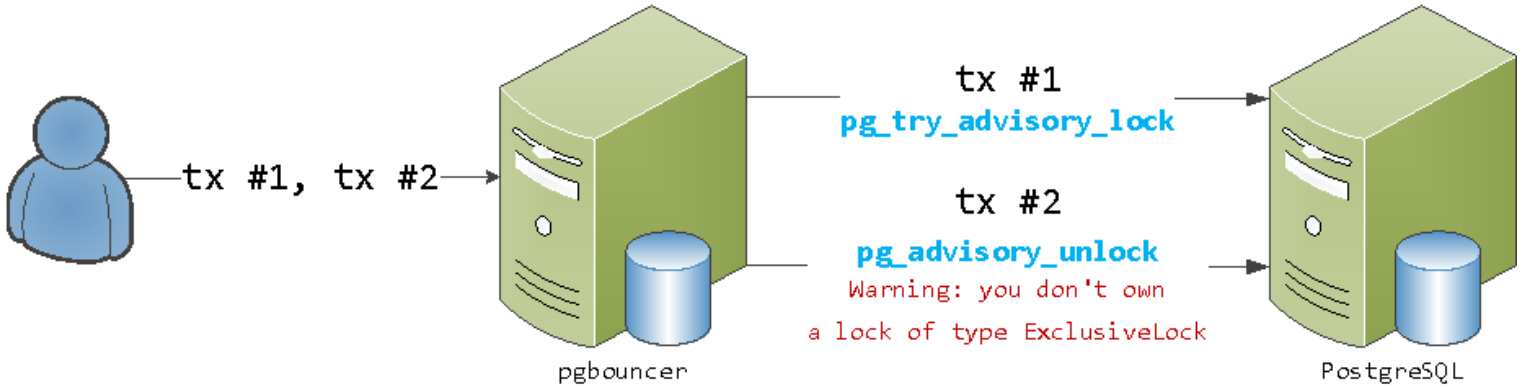
утечки при фильтрации записей

```
SELECT pg_advisory_lock(id) FROM foo WHERE id = 12345;           -- ok
SELECT pg_advisory_lock(id) FROM foo WHERE id > 12345 LIMIT 100; -- опасно!
...
SELECT pg_advisory_unlock_all(); -- «Гильотина – лучшее средство от головной боли!»
```

Advisory

Возможные проблемы

утечки через pgbouncer в transaction mode



Advisory

Возможные проблемы

забыть **try**

pg_try_advisory_lock – не ждет, сразу возвращает **FALSE**

pg_advisory_lock – ждет освобождения блокировки

забыть **xact**

pg_try_advisory_xact_lock – снимется завершением транзакции

pg_try_advisory_lock – останется «висеть» на соединении

Взаимоблокировки (deadlock)

Deadlock

Несколько соединений ждут друг друга

https://postgrespro.ru/docs/postgresql/15/explicit-locking#LOCKING-DEADLOCKS

```
ALTER TABLE x ...;
-- ok
ALTER TABLE y ...;
-- ждем «соседа» справа
```

```
...
ALTER TABLE y ...;
-- ok
ALTER TABLE x ...;
-- ждем «соседа» слева

-- ERROR:  deadlock detected
-- откат транзакции и снятие блокировок
```

Deadlock

Порядок важен!

https://habr.com/ru/companies/tensor/articles/567514/

```
UPDATE tbl SET val = val + 1 WHERE id IN (1, 2, 3);
-- ERROR:  deadlock detected
```

```
UPDATE tbl SET val = val + 1 FROM (
  SELECT ctid FROM tbl WHERE id IN (1, 2, 3)
  ORDER BY id -- явная сортировка
  FOR UPDATE  -- принудительная блокировка
) lc
  WHERE tbl.ctid = lc.ctid; -- поиск по физической позиции записи
```

Deadlock

Как не ждать

deadlock_timeout

https://postgrespro.ru/docs/postgresql/15/runtime-config-locks#GUC-DEADLOCK-TIMEOUT

«Эта проверка довольно дорогостоящая, поэтому сервер не выполняет её при всяком ожидании блокировки. Мы оптимистично полагаем, что взаимоблокировки редки в производственных приложениях, и поэтому **просто ждём некоторое время, прежде чем пытаться выявить взаимоблокировку**. При увеличении значения этого параметра сокращается время, уходящее на ненужные проверки взаимоблокировки, но замедляется реакция на реальные взаимоблокировки.»

Deadlock

Не всегда может распознаться

```
DO $$
BEGIN
  SELECT dblink_exec(
    format('dbname=%s user=%s', current_database(), current_user)
  , 'CREATE INDEX CONCURRENTLY idx_cic ON x(pk);'
  );
END
$$;
-- CONCURRENTLY ждет на своем соединении окончания всех транзакций, включая наш DO
```

Мониторинг блокировок

Мониторинг блокировок

```
SELECT * FROM pg_stat_activity WHERE wait_event_type IS NOT NULL; -- кто ждет
```

https://postgrespro.ru/docs/postgresql/15/monitoring-stats#MONITORING-PG-STAT-ACTIVITY-VIEW

wait_event_type	чего ждем
Activity	пауза в системных фоновых процессах (autovacuum, archiver, checkpointer)
BufferPin	эксклюзивный доступ к странице данных
Client	ожидание активности со стороны клиента
Extension	разрешение условия внутри расширения
IO	ожидание ввода/вывода
IPC	ожидание межпроцессного обмена
Lock	ожидание классической «тяжелой» блокировки (объект БД/ID)
LWLock	ожидание «легкой» блокировки в памяти
Timeout	ждем таймаута (пауза в системных фоновых процессах или pg_sleep)

https://postgrespro.ru/docs/postgresql/15/monitoring-stats#WAIT-EVENT-TABLE

Мониторинг блокировок

```
SELECT * FROM pg_stat_activity WHERE wait_event_type = 'Lock'; -- чего ждет
```

wait_event	чего ждем
advisory	пользовательская рекомендательная блокировка
extend	запись в TOAST
frozenid	«заморозка» записей на таблице
object	блокировка на нереляционном объекте БД
page	блокировка на странице данных
relation	блокировка на таблице/индексе
spectoken	блокировка при INSERT ON CONFLICT
transactionid	ожидание завершения транзакции
tuple	блокировка на строке таблицы
userlock	пользовательская блокировка
virtualxid	блокировка по ID транзакции

https://postgrespro.ru/docs/postgresql/15/monitoring-stats#WAIT-EVENT-LOCK-TABLE

Мониторинг блокировок

```
SELECT * FROM pg_locks WHERE NOT granted; -- кто чего конкретно ждет
```

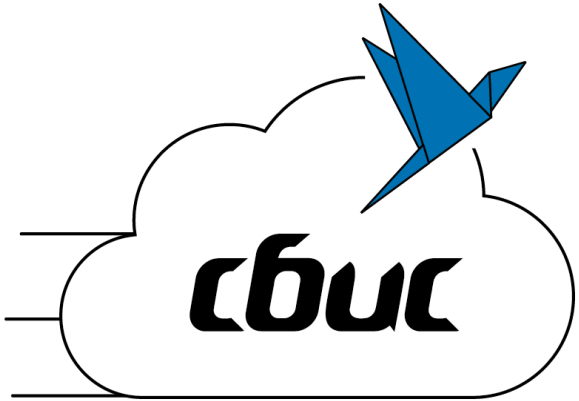
https://postgrespro.ru/docs/postgresql/15/view-pg-locks

```
advisory      | (classid, objid, objsubid)
extend        | (relation)
frozenid      | (relation)
object        | (classid, objid, objsubid)
page          | (relation, page)
relation      | (relation)
spectoken     | (relation, page, tuple)
transactionid | (transactionid)
tuple         | (relation, page, tuple)
userlock      | (classid)
virtualxid    | (virtualxid)
```

https://habr.com/ru/companies/tensor/articles/506360/

Егор Рогов, серия «Блокировки в PostgreSQL»

Блокировки отношений	https://habr.com/ru/companies/postgrespro/articles/462877/
Блокировки строк	https://habr.com/ru/companies/postgrespro/articles/463819/
Блокировки других объектов	https://habr.com/ru/companies/postgrespro/articles/465263/
Блокировки в памяти	https://habr.com/ru/companies/postgrespro/articles/466199/



Спасибо за внимание!

Боровиков Кирилл

kilor@tensor.ru / https://n.sbis.ru/explain

sbis.ru / tensor.ru