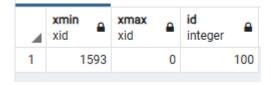
Практика №3 Настройка и поддержка транзакций в СУБД PostgreSQL

Дисциплина	Базы данных для индустриальных задач			
Институт	Перспективных технологий и индустриального			
	программирования			
Кафедра	Индустриального программирования			
Вид учебного материала	Практика			
Преподаватель	Евдошенко Олег Игоревич			
Семестр	1 семестр, 2025-2026			

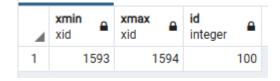
Часть 1. Вставка, обновление и удаление строки

- 1. Создадим произвольную таблицу.
- 2. Вставим первую строку в таблицу.
- 3. Посмотрим, какой номер транзакции xmin: SELECT xmin, xmax, * FROM a;



Получился 1593 — это номер транзакции, в которой была создана первая версия строки.

- 4. Начнем явную транзакцию: *BEGIN*;
- 5. Обновим первую строчку.
- 6. Обратимся и посмотрим, что получилось: SELECT xmin, xmax, * FROM a;
- 7. Во втором терминале обращаемся к таблице. Посмотрим, какой номер транзакции xmin: *SELECT xmin, xmax, * FROM a;*



Обратите внимание, что хтах изменился — это значит, что уже существует вторая версия строки, но она еще не зафиксирована.

- 8. В первом терминале фиксируем транзакцию
- 9. Во втором терминале теперь видим вторую строку: *SELECT xmin, xmax, *FROM*

4	xmin xid	xmax xid	à	id integer	<u></u>
1	1594		0		200

- 10. Теперь посмотрим, как выглядит удаление. Откроем транзакцию в первом терминале.
 - 11. Удаляем строчку.
- 12. Посмотрим результат: *SELECT xmin, xmax, * FROM a;* Первая транзакция не видит строчку, она удалена, но изменение пока не зафиксировано.
 - 13. Посмотрим результат во втором терминале: SELECT xmin, xmax, * FROM a;



Строка еще видна, но хтах опять изменился.

- 14. В первом терминале фиксируем транзакцию
- 15. Во втором терминале теперь видим изменение

Часть 2. Видимость версии строки на различных уровнях изоляции

- 16. Начнем явную транзакцию в первом терминале: *BEGIN*
- 17. Посмотрим уровень изоляции: SHOW transaction_isolation;
- 18. Добавим новую строку.
- 19. Начнем вторую транзакцию во втором терминале и обратимся к таблице.
- 20. Посмотрим уровень изоляции
- 21. Пока новая строка не видна. Зафиксируем первую транзакцию
- 22. Во втором окне повторно обратимся к таблице. Что увидим?
- 23. Зафиксируем вторую транзакцию

Изменения стали видны. Это и есть аномалия неповторяющегося чтения.

Теперь в первом окне начнем транзакцию на уровне repeatable read.

- 24. Вставим еще одну строку: BEGIN ISOLATION LEVEL REPEATABLE READ
- 25. Во второй транзакции обратимся к таблице в новой транзакции на том же уровне.
 - 26. Теперь фиксируем первую транзакцию
 - 27. Обратимся ко второй транзакции еще раз.

Изменения не видны. на этом уровне операторы транзакции работают только с одним снимком данных.

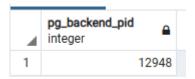
28. Зафиксируем вторую транзакцию.

Часть 3. Состояние транзакции по CLOG

- 29. Откроем первую транзакцию, добавим строку и посмотрим после вставки состояние.
- 30. Видим вставку третьей строки. Посмотрим статус транзакции: *SELECT pg_xact_status('1598');*
- 31. Зафиксируем транзакцию и посмотрим статус.
- 32. Теперь посмотрим, как поведет себя CLOG при откате транзакции: откроем транзакцию, добавим новую строку, просмотрим статус транзакции, выполним отказ транзакции, еще раз посмотрим статус

Часть 4. Блокировки таблицы

33. В первой транзакции вставим новую строку и посмотрим блокировки с помощью pg_locks, для этого нам нужен pid обслуживающего процесса: *SELECT* pg backend pid();

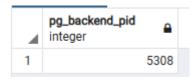


- 34. Откроем транзакцию и обратимся к таблице.
- 35. Выполним обновление данных: $UPDATE\ a\ SET\ id=id+1;$
- 36. Выполним запрос: SELECT locktype, transactionid, mode, relation::regclass as obj FROM pg_locks where pid = 12948;



Появилась блокировка на уровне таблицы RowExclusiveLock — накладывается в случае обновления строк.

37. Во втором окне построим индекс по таблице, предварительно посмотрим pid процесса: SELECT pg backend pid();



- 38. Создадим индекс: CREATE INDEX ON a (id);
- 39. Транзакция подвисла. В первом терминале посмотрим, что происходит во втором процессе: SELECT locktype, transactionid, mode, relation::regclass as obj FROM pg_locks where pid = 5308;

4	locktype text	transactionid xid	mode text	obj regclass
1	virtualxid	[null]	ExclusiveLock	[null]
2	relation	[null]	ShareLock	a

Появилась блокировка ShareLock она не совместима RowExclusiveLock возникла блокировочная ситуация.

40. Зафиксируем первую транзакцию. Тут же срабатывает команда во втором окне.