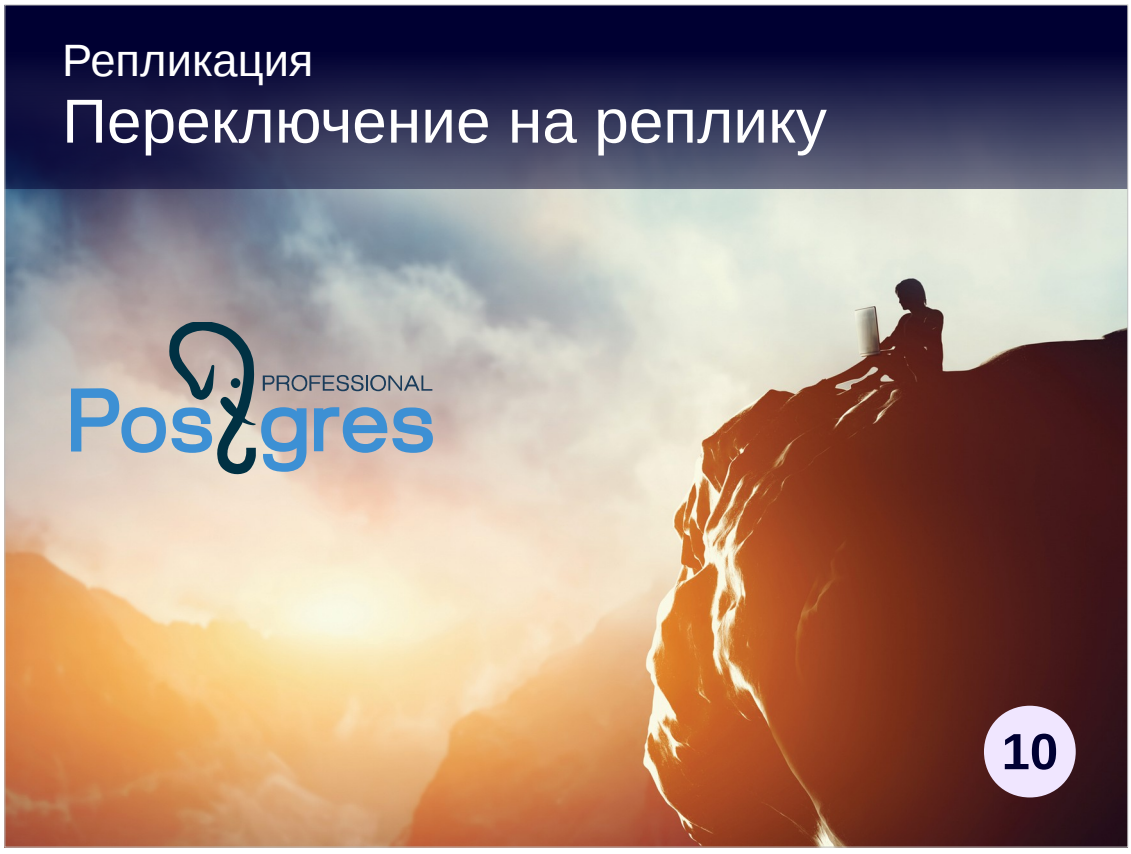


Авторы: Егор Рогов, Павел Лузанов

Обратная связь
Отзывы, замечания и предложения направляйте по адресу:
edu@postgrespro.ru

Темы

Переключение на реплику

Особенности, связанные с файловым архивом

Возвращение в строй бывшего мастера

Переключение на реплику

Причины и процедура переключения на реплику

Действия реплики при переключении на нее

Переключение на реплику

Причины

плановое переключение (switchover):
останов основного сервера для проведения технических работ

аварийное переключение (failover):
переход на реплику из-за сбоя основного сервера

Процедура

убедиться, что мастер остановлен

переключение вручную: «продвижение» или триггерный файл

автоматическое переключение: отсутствет
(для автоматизации требуется стороннее кластерное ПО)

Причиной перехода на резервный сервер может быть необходимость проведения технических работ на основном сервере — тогда переход выполняется в удобное время в штатном режиме (switchover). А может быть сбой основного сервера, и в таком случае переходить на резервный сервер нужно как можно быстрее, чтобы сократить время простоя системы (failover).

В любом случае сначала нужно убедиться, что мастер остановлен. Это очень важно, иначе данные на разных серверах «разойдутся», и свести их потом воедино — нетривиальная и неавтоматизируемая задача. Скорее всего, часть данных придется просто потерять.

Затем выполняется переход на реплику в ручном режиме. Автоматизация этого процесса возможна, но требует стороннего кластерного программного обеспечения (это обсуждается в модуле «Кластерные технологии»).

Переключение состоит в разрыве цикла восстановления. Для этого реплике посылается команда promote (либо pg_ctl promote, либо pg_ctlcluster ... promote, в зависисмости от того, как установлен PostgreSQL). Другой вариант — прописать в recovery.conf параметр *trigger_file*. При появлении в системе файла с таким именем восстановление прерывается.

Еще один вариант: удалить файл recovery.conf и перезапустить резервный сервер. Это не вполне «честный» способ, поскольку в этом случае реплика не поймет, что восстановление завершено, и не перейдет на новую ветвь времени. Дальше мы будем рассматривать только два первых варианта.

Действия при переходе

Применяются журнальные записи

- уже полученные wal receiver, но еще не примененные startup

Завершаются процессы

- wal receiver
- startup

Запускаются процессы

- wal writer
- autovacuum launcher
- archiver (зависит от настройки)

Переход на новую ветвь времени

Получив сигнал, сервер завершает восстановление и переходит в обычный режим работы.

Для этого он применяет уже полученные журнальные записи, которые еще не были применены.

Процессы wal receiver и startup завершают свою работу — на основном сервере они не нужны. А вот процессы wal writer и autovacuum launcher, наоборот, запускаются.

Кроме того, сервер переходит на новую ветвь времени.

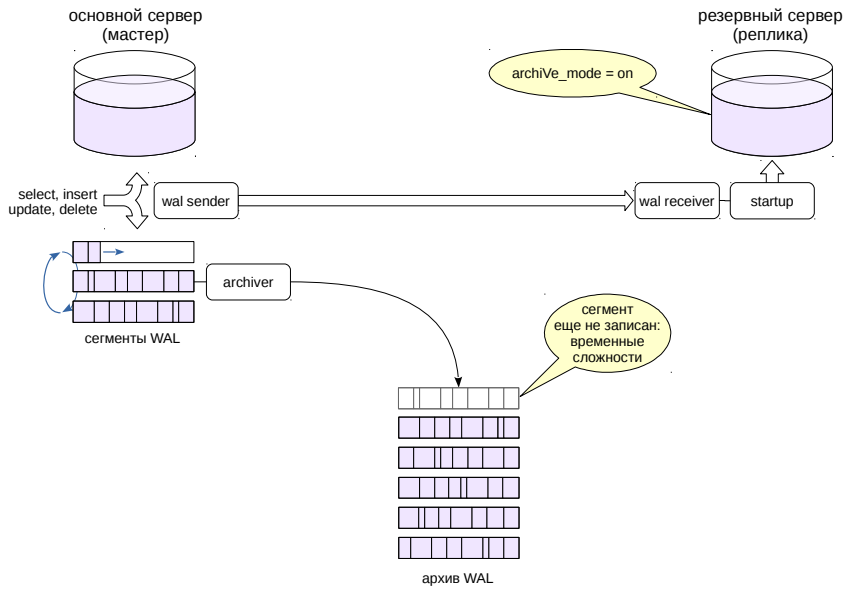
С точностью до некоторых деталей, все происходит так же, как при окончании восстановления из резервной копии.

Переключение и архив

Особенности, связанные с файловым архивом

Настройка archive_mode: режимы on и always

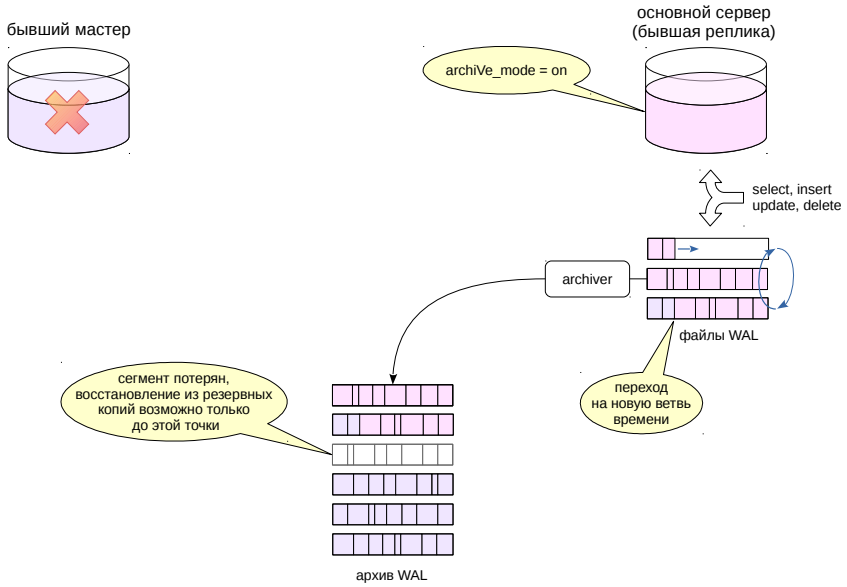
Переключение и архив



Архив файлов журнала предзаписи, наполняемый с помощью механизма непрерывного архивирования, имеет неприятную особенность в контексте потоковой репликации и переключении на реплику.

Допустим, при отказе мастера не все сегменты были записаны в архив. Например, могли возникнуть временные проблемы с доступным местом и archive_command возвращала ошибку.

Переключение и архив

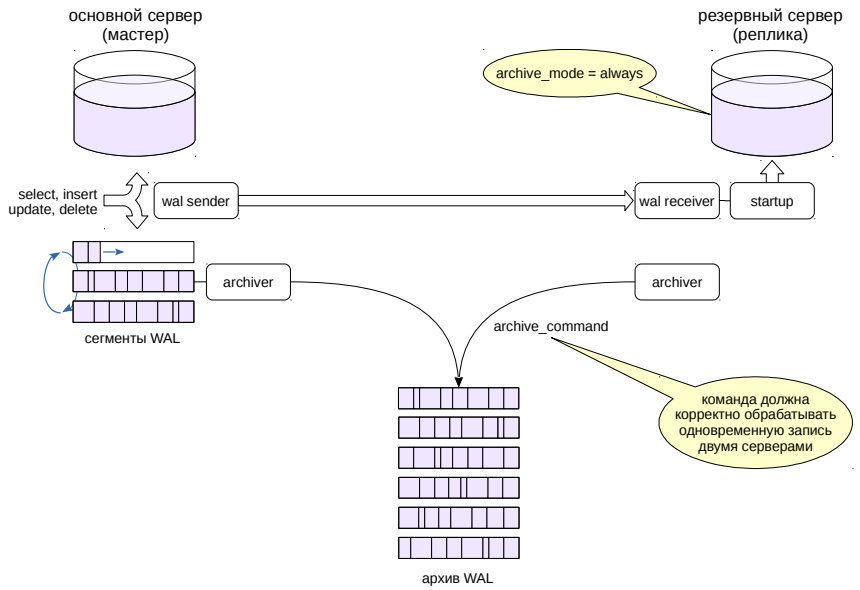


Но реплика не в курсе настроек архивирования на мастере. Когда бывшая реплика займет место мастера, она не запишет недостающие сегменты в архив (хотя они у нее есть), потому что рассчитывает на то, что архив работал без сбоев. В результате архив будет неполным.

А это означает, что из имеющихся резервных копий можно восстановить систему только до образовавшейся «дыры». Если такая ситуация возникла (а это еще нужно понять), требуется в срочном порядке выполнить резервное копирование.

Потоковый архив лишен этого недостатка, поскольку утилита pg_receivewal отслеживает содержимое каталога и запрашивает у сервера недостающие журнальные записи. Но, как отмечалось в модуле «Резервное копирование», использование этой утилиты сопряжено с поддержанием дополнительной инфраструктуры.

Переключение и архив



В версии 9.5 введено новое значение параметра *archive_mode:* always. При этом значении на реплике запускается процесс archiver, который записывает сегменты в архив наравне с мастером. Таким образом, один и тот же файл будет записан два раза: и мастером, и репликой. Это накладывает на команду *archive_command* серьезные требования:

- она не должна перезаписывать существующий файл, но должна сообщать об успехе, если файл с тем же содержимым уже есть в архиве;

- она должна корректно обрабатывать одновременный вызов с двух серверов.

При такой настройке сегмент не пропадет, поскольку реплика будет продолжать попытки записи даже после останова мастера.

(Разумеется, можно настроить *archive_command* и так, чтобы мастер и реплика сохраняли сегменты журнала в разные архивы, но вряд ли это практично.)

Возвращение в строй

Непосредственное подключение

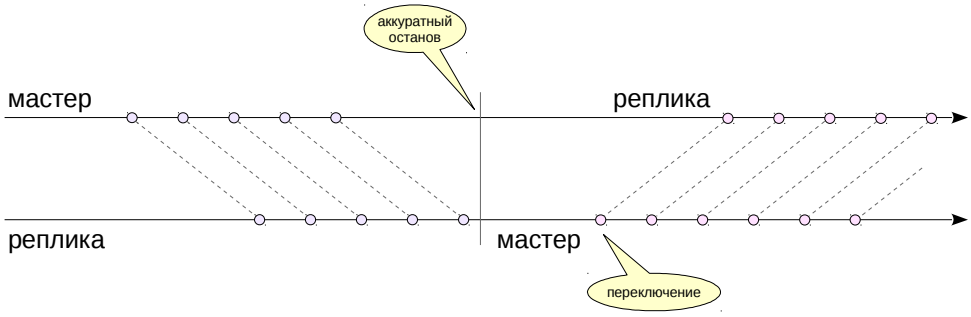
Новая реплика из резервной копии

Утилита pg_rewind

Простое подключение

Бывший мастер подключается к новому как реплика

допустимо, только если перед переключением реплика получила от мастера все журнальные записи



Если переход на реплику не был вызван выходом сервера из строя, то нужен способ быстро вернуть старый мастер в строй — теперь уже в качестве реплики (failback).

В случае аккуратной остановки мастера (останов в режимах fast или smart) все журнальные записи мастера скорее всего дойдут до реплики, хотя это и не гарантируется. Процесс останова организован таким образом, что сначала отключаются все обслуживающие процессы, затем выполняется контрольная точка, и только в самую последнюю очередь останавливается процесс wal sender, чтобы реплика успела получить запись WAL о контрольной точке.

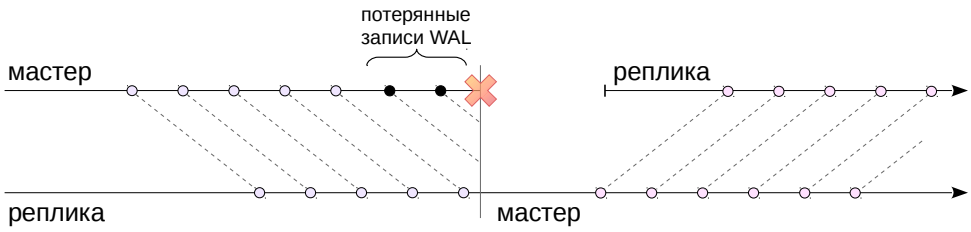
Позицию в журнале мастера можно проверить с помощью утилиты pg_controldata («Latest checkpoint location»), а позицию на реплике покажет функция pg_last_wal_receive_lsn(). Поскольку функция pg_last_wal_receive_lsn() показывает *следующую* позицию, то ее значение должно *опережать* latest_checkpoint_location на длину записи (104 байта в текущей версии PostgreSQL). Если это так, то бывший мастер можно непосредственно подключить к новому, изменив соответстующим образом конфигурационные параметры.

В случае останова мастера без выполнения контрольной точки (сбой или режим immediate), такое подключение в принципе невозможно: сервер не стартует, а в журнале сообщений будет зафиксирована ошибка.

Резервная копия

Бывший мастер восстанавливается из резервной копии

- абсолютно новая реплика, процесс занимает много времени
- можно ускорить rsync, если с момента сбоя прошло немного времени (но все равно долго для больших баз данных)



Если мастер был остановлен аварийно, велика вероятность того, что часть журнальных записей не успела дойти до реплики. В этом случае просто так подключать мастер нельзя.

Простой и надежный вариант — создать абсолютно новую реплику путем изготовления и развертывания базовой резервной копии. Однако для больших баз данных этот процесс может занимать много времени.

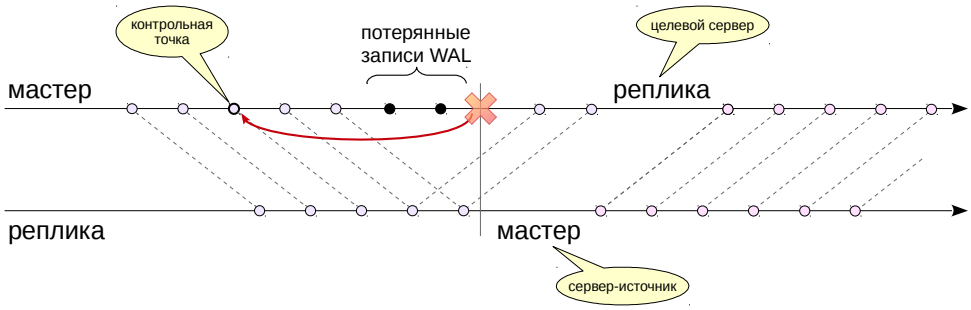
Вариант такого подхода — не использовать утилиту pg_basebackup, а сделать копию с помощью API резервирования с использованием утилиты rsync. Если выполнять копирование сразу после перехода на реплику, то большая часть файлов не должна успеть поменяться и процесс может пройти существенно быстрее. Но это, конечно, усложняет процесс.

Утилита pg_rewind

Подготовка к восстановлению

«откатывает» потерянные записи WAL, заменяя соответствующие страницы целевого сервера страницами с сервера-источника

копирует с сервера-источника все служебные файлы



Еще более быстрый вариант состоит в использовании утилиты pg_rewind. Она входит в состав PostgreSQL с версии 9.5.

Утилита определяет место расхождения между двумя серверами, определяет ближайшую к нему контрольную точку, и, просматривая журнал, определяет все страницы, измененные с момента этой контрольной точки.

Найденные страницы (которых должно быть немного) заменяются страницами с сервера-источника (нового мастера). Кроме того, утилита копирует с сервера-источника все служебные файлы.

Дальше применяются все необходимые записи WAL с нового мастера. Фактически, это выполняет уже не утилита, а обычный процесс восстановления после запуска сервера. Чтобы восстановление началось с нужного момента, утилита создает управляющий файл backup_label.

Ограничения pg_rewind

Целевой сервер

- все необходимые журнальные файлы должны сохраниться в pg_wal
- должны быть включены контрольные суммы или *wal_log_hints* = on
- сервер должен быть остановлен через контрольную точку

Сервер-источник

- должен быть включен параметр *full_page_writes* = on

К сожалению, утилита имеет ряд особенностей, ограничивающих ее применение. Необходимо, в числе прочего:

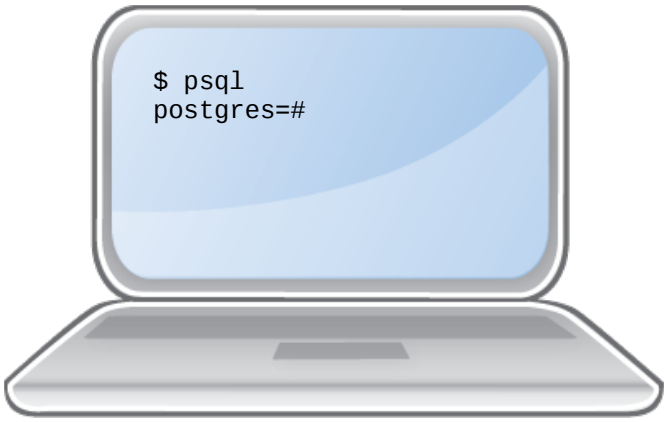
- Все сегменты WAL от текущего момента до найденной контрольной точки должны находиться в каталоге pg_wal целевого сервера. Обычно это не проблема, если бывший мастер был остановлен сразу перед переключением на реплику, или хотя бы вскоре после этого.

- Первое изменение данных после контрольной точки должно вызывать запись в WAL полной страницы. Параметра *full_page_writes* = on недостаточно, поскольку он не учитывает «незначительные» изменения страниц (hint bits). Требуется, чтобы либо кластер был инициализирован с контрольными суммами страниц, либо нужно устанавливать параметр *wal_log_hints* = on.

- Целевой сервер должен быть остановлен аккуратно, с выполнением контрольной точки. В случае аварийного переключения можно попробовать запустить целевой сервер и тут же остановить его уже в штатном режиме.

- На сервере-источнике должен быть установлен параметр *full_page_writes* = on — причина та же, что и при восстановлении из резервной копии: утилита может скопировать страницы в рассогласованном состоянии.

Демонстрация



Итоги

Переключение используется как в штатных,
так и в нештатных ситуациях

После переключения бывший мастер надо вернуть в строй

Обе процедуры должны быть заранее отработаны

Файловый архив журнала предзаписи требует внимания

Практика

1. Выполните необходимую настройку мастера и реплики для потоковой репликации с использованием слота, без непрерывного архивирования.
2. Имитируйте сбой основного сервера.
3. Переключитесь на реплику.
4. Верните в строй бывший основной сервер, выполнив резервную копию с нового мастера и настроив необходимые параметры.
5. Убедитесь, что репликация работает и использует слот.
6. Переключитесь на новую реплику, чтобы бывший мастер снова стал основным сервером.

2. Для имитации сбоя можно остановить сервер в режиме immediate:

pg_ctlcluster 10 *имя_кластера* stop -m immediate --skip-systemctl-redirect

(или pg_ctl stop -m immediate, если PostgreSQL собран из исходных кодов).