PostgreSQL-Consulting

data egret

Потоковая репликация в PostgreSQL.

Alexey Lesovsky

alexey.lesovsky@dataegret.com

01 Введение

- Что такое репликация и зачем.
- Какая бывает репликация.
- Как устроена потоковая репликация в PostgreSQL.

02 Настройка

- Настройка потоковой репликации.
- Проверка результата.
- Особенности эксплуатации.

01

Введение

01 Что такое репликация

Синхронизация объектов.

Изменения распространяются на копии.

Репликация может быть *физической* или *логической*.

01 Зачем нужна репликация

Отказоустойчивость базы данных.

Масштабирование на чтение/запись.

Аналитика и BI.

01 Логическая репликация

Плюсы:

- Работает между разными версиями и архитектурами.
- Позволяет реплицировать отдельные наборы таблиц.

Минусы:

- Сложность в реализации синхронной репликации.
- Утилизация CPU (триггеры, преобразование текста, ...).

Примеры:

- Slony, Londiste (Skytools), Bucardo, Pglogical.

01 Физическая репликация

Плюсы:

- Небольшие накладные расходы на использование ресурсов.
- Легкость установки и обслуживания.

Минусы:

- Запасные узлы доступны только для чтения.
- Не работает между разными версиями и архитектурами.
- Не умеет реплицировать наборы таблиц.

01 REDO журнал

Необходимость подтверждать все изменения (Durability в ACID).

Все (*почти*) изменения записываются в REDO журнал.

REDO журнал это история *«последних»* изменений.

REDO журнал используется:

- При аварийном восстановлении;
- При резервном копировании;
- **При репликации.**

01 REDO журнал в PostgreSQL

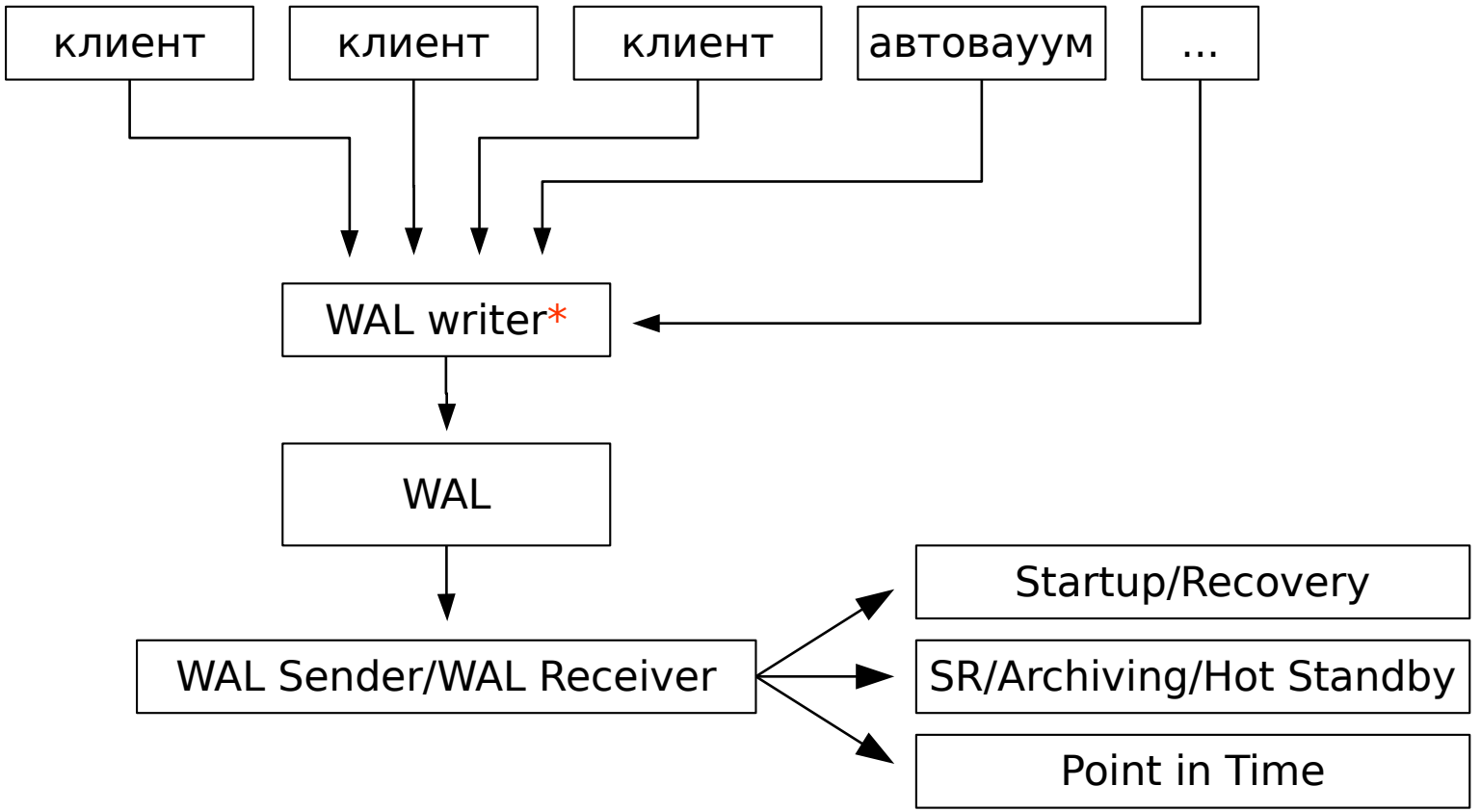
В PostgreSQL, REDO журнал называется *Write Ahead Log* (WAL).

WAL гарантирует что информация об изменениях будет зафиксирована **ДО** реальных изменений.

Как это работает:

- LSN (*log sequence number*) – положение записи внутри WAL;
- Страницы маркируются LSN;
- Перед записью страницы на диск, проверяем что LSN уже записан в журнал.

01 Упрощенная схема



* - опционально

01 Startup процесс

Главный компонент который запускает СУБД.

Запускается восстановление по WAL журналу.

Чтение конфигурации и определение источника WAL.

REDO цикл:

- Чтение WAL из pg_xlog/ или WAL архива;
- Установка соединения с upstream.

01 WAL Receiver процесс

WAL receiver:

- Определение с какого места начать прием WAL;
- Подключение к мастеру и отправка LSN отметки;
- Принимает WAL и записывает на диск;
- Обновляет *особую переменную* в shared memory;
- Отправляет статистику на мастер.

Startup процесс использует *особую переменную* чтобы воспроизвести WAL до этого места.

01 WAL Sender процесс

Для каждого клиента, создается отдельный backend-процесс.

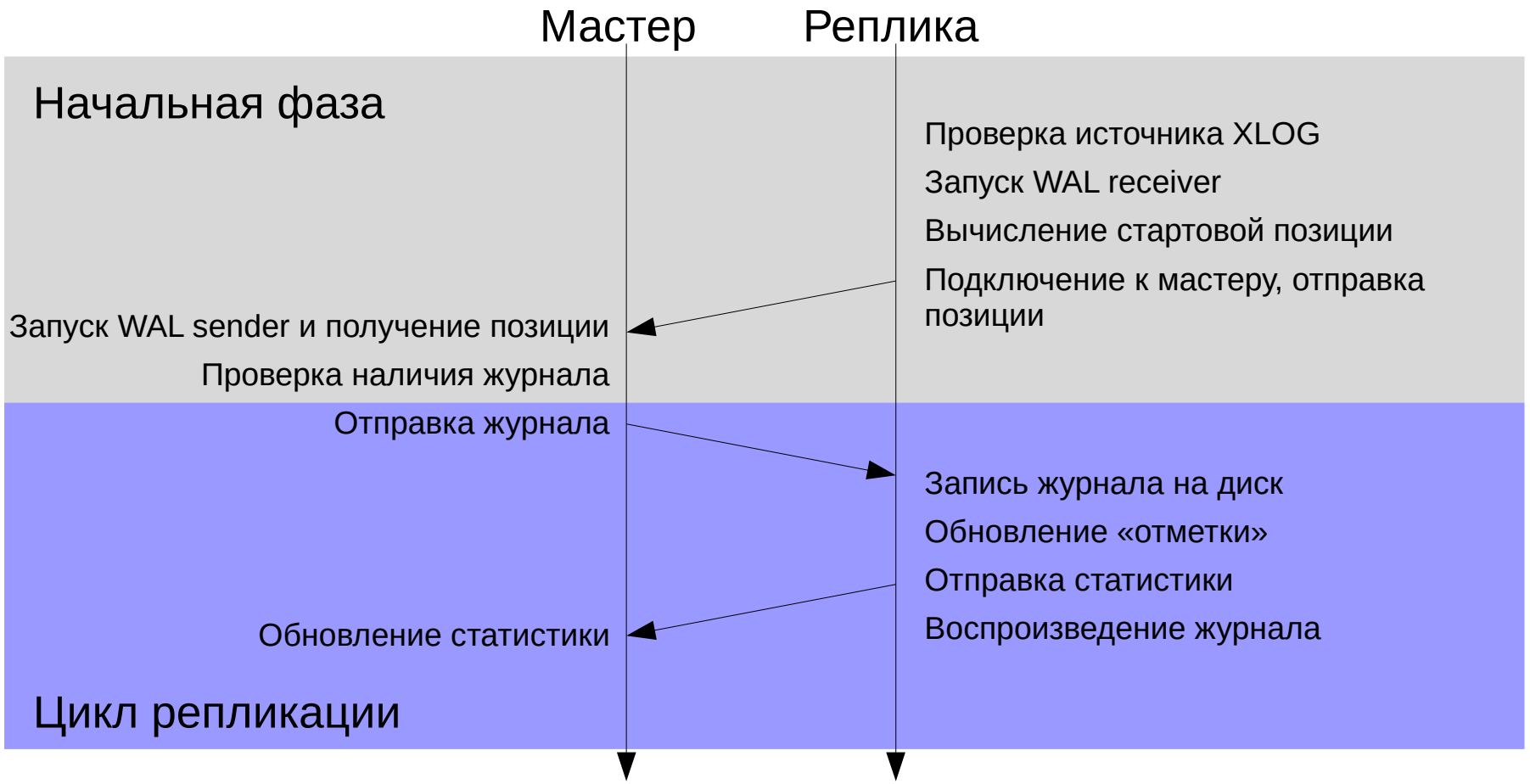
WAL sender это тоже backend.

WAL sender запускает репликацию.

Отправляет WAL журнал клиенту.

Или спит если нет новых журналов.

01 Упрощенный порядок работы



02

Настройка

02 План

Варианты настройки.

Подготовка мастера.

Запуск репликации.

Проверка результата.

Особенности эксплуатации.

02 Варианты настройки

Синхронная или асинхронная репликация.

Каскадная конфигурация.

02 Стандартный алгоритм

Подготовка мастера (настройка конфигурации).

Копирование каталога DATADIR.

Подготовка реплики (настройка конфигурации).

Запуск реплики.

Проверка результата.

02 Настройка мастера

Создание отдельного пользователя для репликации.

Правка postgresql.conf.

Правка pg_hba.conf.

Создание слота репликации (*необязательно*).

02 Настройка мастера

Отдельный пользователь для репликации (*psql* или *createuser*).

- CREATE ROLE replica WITH LOGIN REPLICATION PASSWORD '123';

Правка postgresql.conf.

Правка pg_hba.conf.

Создание слота репликации (*необязательно*).

02 Настройка мастера

Выделенный пользователь для репликации.

Правка postgresql.conf.

- wal_level = replica (or logical)
- max_wal_senders = 8
- wal_keep_segments = 200
- Рестарт обязателен.

Правка pg_hba.conf.

Создание слота репликации (необязательно).

02 Настройка мастера

Отдельный пользователь для репликации.

Правка postgresql.conf.

Правка pg_hba.conf.

- *host replication username client_addr/mask authtype*
- host replication replica 10.1.0.99/32 md5
- Требуется reload.

Создание слота репликации (необязательно).

02 Настройка мастера

Выделенный пользователь для репликации.

Правка postgresql.conf.

Правка pg_hba.conf.

Создание слота репликации (*опциональный шаг*).

- postgresql.conf — max_replication_slots = 4.
- Создание слота с *pg_create_physical_replication_slot('name')*;
- recovery.conf — primary_slot_name = 'name'.

02 Копирование DATADIR

pg_basebackup (с версии 9.1)

-h, --host=...; -p, --port=...; -U, --username=...; -d, --dbname=...; -D, --pgdata=...

-c, --checkpoint=fast | spread

-X, --xlog-method=fetch | stream – *stream* с версии 9.2

-R, --write-recovery-conf – с версии 9.3

-r, --max-rate=... – с версии 9.4

--xlogdir=... – с версии 9.4

-T, --tablespace-mapping=olddir=newdir – с версии 9.4

-P, --progress

pg_basebackup -P -R -X stream -c fast -h 1.2.3.4 -U replica -D /pgdb

02 Копирование DATADIR

Утилиты файлового копирования - cp, scp, tar, rsync...

Снимки:

- ZFS send/receive;
- LVM + dd.

pg_start_backup() + pg_stop_backup().

02 Настройка реплики

Файлы конфигурации:

- Должны быть одинаковыми *(желательно)*;
- postgresql.conf;
- recovery.conf.

02 Настройка реплики

Файлы конфигурации (postgresql.conf):

- hot_standby = on;

02 Настройка реплики

Файлы конфигурации (recovery.conf):

- primary_conninfo = 'host=... port=...' – обязателен
- standby_mode = on – обязателен
- primary_slot_name = 'slotname' - слоты?
- trigger_file = '...' – рекомендуется
- recovery_min_apply_delay. - отложенная реплика

02 Запуск реплики

pg_ctl – штатная утилита PostgreSQL.

pg_ctlcluster – perl обертка над pg_ctl в Debian/Ubuntu Linux.

sysvinit, upstart, openrc, systemd...

02 Проверка результата

Наличие процессов WAL sender и WAL receiver.

Проверка системного журнала.

Простое подключение через *psql*.

Системное представление *pg_stat_replication*.

02 Проверка результата

Наличие процессов WAL sender и WAL receiver.

master $ ps aux |grep -i wal
postgres: wal sender process postgres [10.1.0.99] streaming 4/EA000060

standby $ ps aux |grep -i wal
postgres: wal receiver process   streaming 4/EA000060

02 Проверка результата

Проверка системного журнала.

LOG: database system was interrupted; last known up at 2017-02-10 12:28:54
LOG: entering standby mode
LOG: redo starts at 4/E9000028
LOG: consistent recovery state reached at 4/E9000130
LOG: database system is ready to accept read only connections
LOG: started streaming WAL from primary at 4/EA000000 on timeline 1

02 Проверка результата

Подключение через psql.

```
$ psql -h replica -U postgres
psql (9.6.2)
Type "help" for help.
postgres=# select pg_is_in_recovery();
true
```

02 Проверка результата

Системное представление pg_stat_replication.

```
postgres=# select * from pg_stat_replication;
-[ RECORD 1 ]----+------------------------------
pid              | 29351
usesysid         | 10
usename          | postgres
application_name | walreceiver
client_addr      | 10.1.0.99
client_hostname  |
client_port      | 5432
backend_start    | 2017-04-15 12:52:54.639356+05
backend_xmin     |
state            | streaming
sent_location    | 4/EA000060
write_location   | 4/EA000060
flush_location   | 4/EA000060
replay_location  | 4/EA000060
sync_priority    | 0
sync_state       | async
```

02 Особенности эксплуатации

Поставил и забыл.

02 Особенности эксплуатации

Мониторинг и поиск проблем:

- pg_stat_replication — лаг репликации.
- pg_current_xlog_location(), pg_xlog_location_diff().
- pg_stat_activity — запросы на реплике.

02 Особенности эксплуатации

Использование слотов:

- wal_keep_segments не нужен.
- pg_replication_slots – мониторинг слотов.
- мониторинг дискового пространства.

02 Особенности эксплуатации

Долгие запросы на реплике могут быть причиной лага:

- Неизбежное зло.
- Переписывать запросы или отстреливать их.
- Или вообще забить.

02 Особенности эксплуатации

DDL и autovacuum может аффектить запросы на реплике:

- Конфликты репликации.
- pg_stat_database_conflicts.
- hot_standby_feedback = on.
- max_standby_streaming_delay = ...

02 Особенности эксплуатации

Нет встроенных средств автофайловера.

- trigger_file (recovery.conf).
- Скрипты на Shell/Python/Ansible/whatever.
- Repmgr, Patroni, Stolon.

02 Особенности эксплуатации

Бэкап:

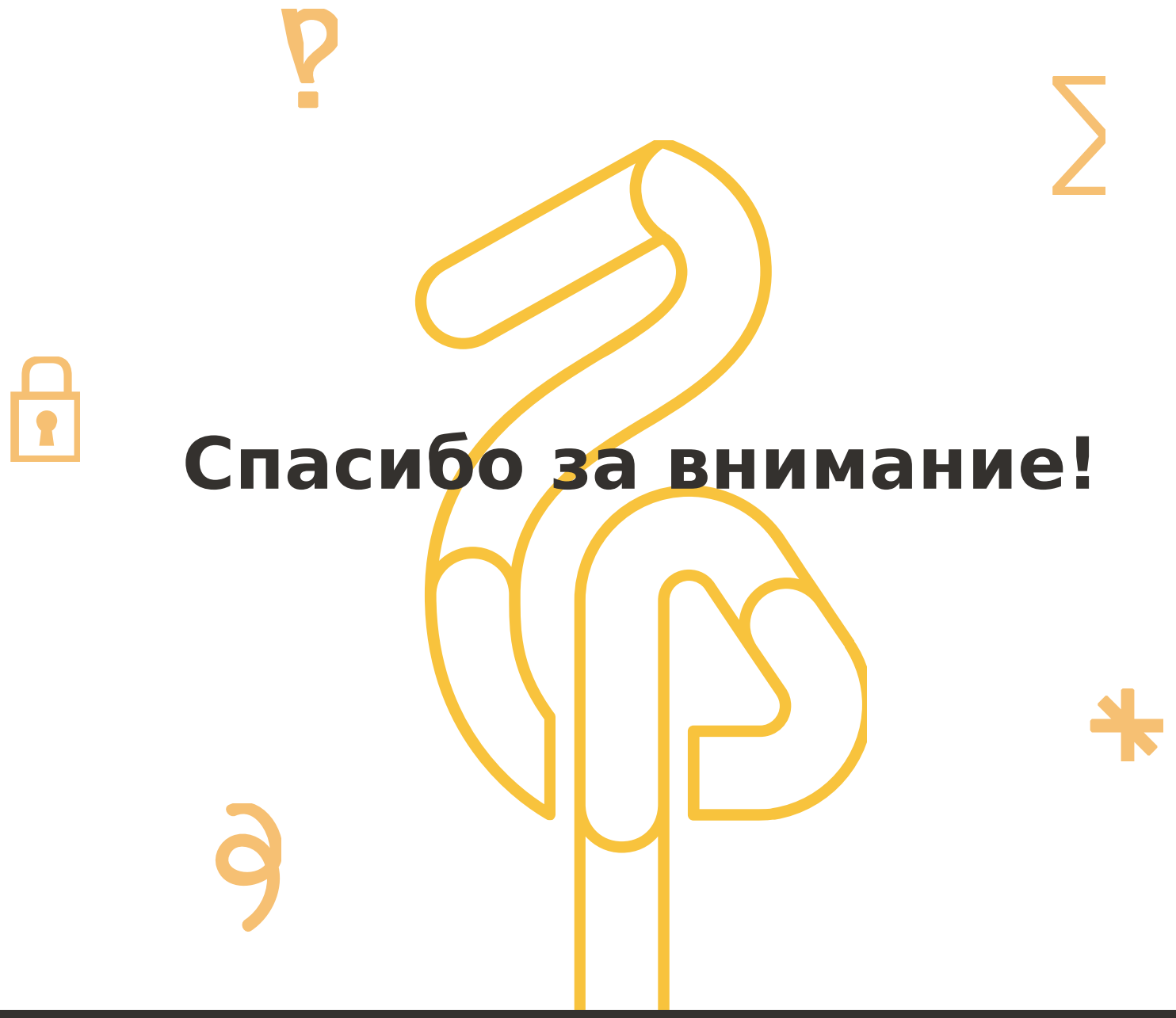
- Реплика **!=** Бэкап.
- pg_basebackup + WAL архив.
- pgBarman, pgBackRest.

02 Резюме

Репликация это нужно и полезно.

Настроить репликацию легко.

Репликация проста в обслуживании.



Спасибо за внимание!

alexey.lesovsky@dataegret.com