

# Технологии и разработка СУБД

## Лекция 7. Репликация в PostgreSQL

Анастасия Лубенникова  
Александр Алексеев

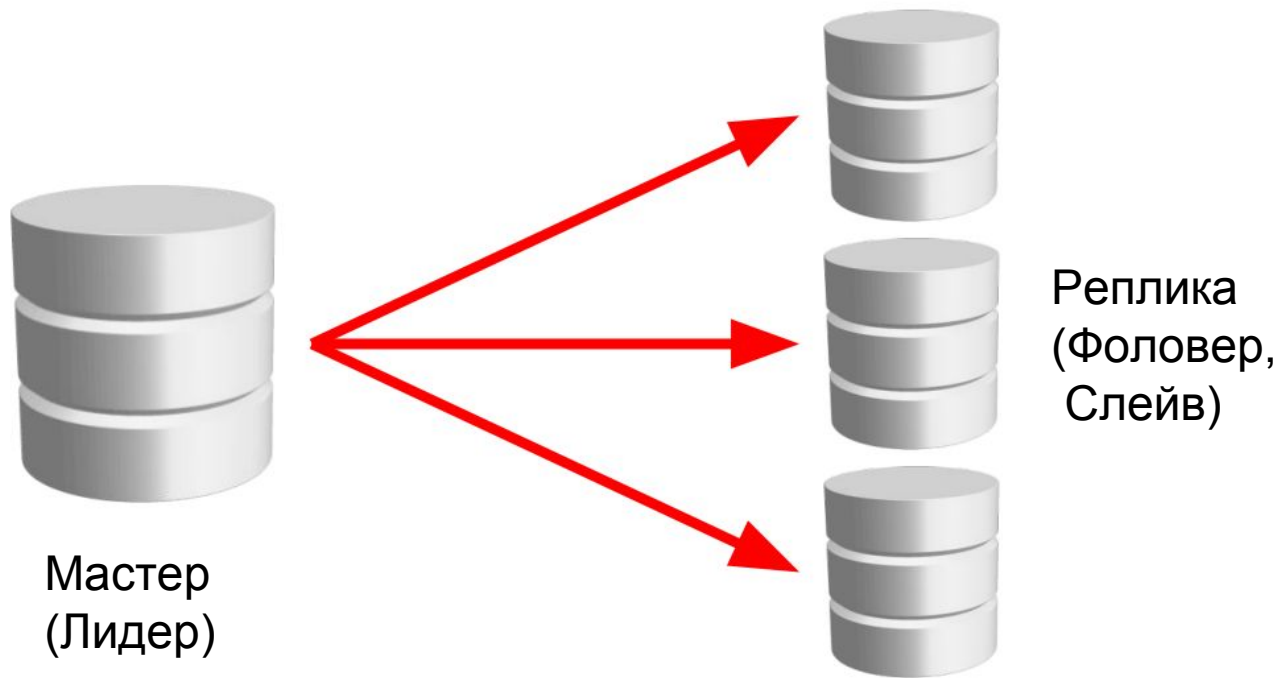
# Для кого эта лекция

- Вы считаете, что репликация - это непостижимо сложно;
- Вы никогда не настраивали физическую и/или логическую репликацию в PostgreSQL;
- Вы не знаете, как настроить фейловер;
- Вы хотели бы узнать, что нового здесь появилось у PostgreSQL в последнее время;
- Вы ищете идею для проекта =);

# Чего в ней не будет

- Скучного пересказа документации на тему что именно писать в конфигах. То есть, лекция скорее обзорная;
- Для заинтересованных в конце приводятся ссылки на дополнительные материалы;

# Репликация



# Зачем это нужно

- Распределение нагрузки
  - OLTP: на чтение ходим в реплики
  - OLAP: тяжелая аналитика на отдельной реплике
  - Снятие бэкапа с отдельной реплики
- Фейловер / High Availability
  - Бывает ручной и автоматический
- Отложенная репликация
- **Не заменяет резервное копирование!**

# Потоковая (или физическая) репликация

- В сущности, заключается в передаче WAL по сети;
- Асинхронная
  - Быстро, но можно потерять данные;
- Синхронная
  - Медленнее (в рамках ДЦ не намного), но надежнее. Желательно иметь две реплики;
- Бывает еще каскадной (надо же было упомянуть об этом на каком-то слайде)

# Fun facts!

Потоковая репликация:

- Не работает между разными архитектурами;
- Не работает между разными версиями PostgreSQL [1];

[1] Согласно <https://simply.name/ru/upgrading-postgres-to-9.4.html> типичное время даунтайма при обновлении версии составляет несколько минут.

# Логическая репликация

- Начиная с PostgreSQL 10 - из коробки;
- Старые подходы: Slony, Londiste, pglogical;
  - Не рекомендуются, потому что медленные и/или плохо работают;



# Зачем нужен еще один вид репликации?

- Репликация части данных, не всего подряд;
- Обновление без даунтайма;
- На реплике можно использовать временные таблицы, да и вообще писать все что угодно, в т.ч. в реплицируемые таблицы;
- Одна реплика может тянуть данные с двух мастеров;
- *В теории* - можно изобразить multimaster;
- И другие сценарии, когда физическая репликация не подошла;

# Fun facts!

- Схема таблиц на мастере и на реплике может различаться;
- Может отличаться порядок столбцов;
- Реплика может иметь дополнительные nullable столбцы;
- НО мастер не может иметь больше столбцов, чем реплика, даже если в этих столбцах всегда NULL;

# Ограничения логической репликации

- Реплицируемые таблицы должны иметь primary key;
- DDL, TRUNCATE и sequences не реплицируются;
- Поддержка триггеров реализована не до конца [1];

[1]: <https://postgr.es/m/20171009141341.GA16999@e733.localdomain>

# Фейловер

- Ручной
  - Имеет смысл, когда серверов БД не много (~10 штук);
  - Кстати, сейчас не проблема купить машину с сотнями Гб памяти и несколькими Тб места на диске;
- Автоматический
  - Может иметь смысл, когда вы приближаетесь к масштабам Google;

# Решения для настройки автофейловера

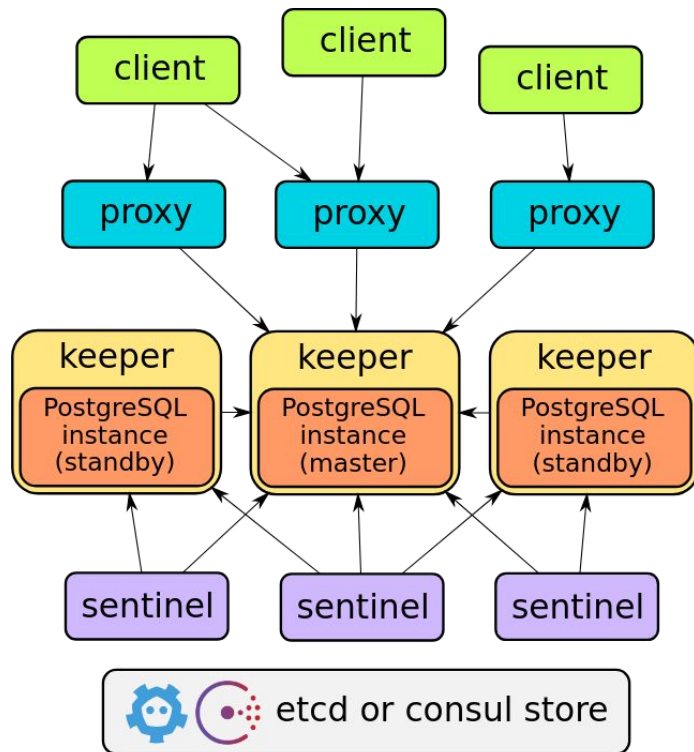
- Физическая репликация
  - Своими велосипедами на Python ;)
  - Repmgr
  - Patroni
  - Stolon (нравится мне больше всего)
- Логическая репликация
  - Еще не написали :(

# Stolon

Коротко о главном:

- Разрабатывается с 2015 года компанией Sorint.lab
- Написан на Go
- Полагается на Consul или etcd
- Умеет интегрироваться с Kubernetes
- Настраивается быстро и просто
- Корректно обрабатывает любые падения машин и нетсплиты

# Stolon: как это работает?



# Fun facts!

- Stolon направляет и чтение, и запись в мастер. Но есть воркэраунд [1];
- Использует Consul или etcd чисто как key-value, в частности не знает про поддержку Consul'ом DNS;

[1]: <https://github.com/sorintlab/stolon/issues/132>



# Consul

Коротко о главном:

- Разрабатывается HashiCorp, подарившей миру Packer и Vagrant;
- Написан на языке Go, использует протокол Raft;
- Решение для service discovery, как ZooKeeper или etcd;
- Распределенное key-value хранилище с REST интерфейсом;
- Имеет CAS, встроенный мониторинг, локи, подписки на обновления, ...;
- Умеет отдавать информацию о сервисах по DNS;
- Тестируется Jepsen'ом [1];

[1]: <https://www.consul.io/docs/internals/jepsen.html>

# Fun facts!

- У Consul есть красивый веб-интерфейс с информацией о зарегистрированных сервисах;
- Поверх него (ровно как и поверх Cassandra или Couchbase) можно довольно легко написать выбор лидера, используя подход под названием leader lease [1];

[1]: <https://eax.me/go-leader-election/>

# synchronous\_commit

- `synchronous_commit = off`
  - Не ждем записи в WAL, можно потерять часть последних изменений
  - В отличие от `fsync = off` не приведет к неконсистентности базы
- `synchronous_commit = on`
  - Ждем подтверждения записи в WAL - свой и синхронной реплики
- `synchronous_commit = remote_write`
  - Аналогично `on`, но не дожидаемся `fsync()` на реплике
- `synchronous_commit = local`
  - Не ждем записи на реплике, пишем только локально
- `synchronous_commit = remote_apply ( >= 9.6 )`
  - Ждем, когда данные попадут в WAL реплики *и применятся к данным*

# Fun fact!

- `synchronous_commit` можно менять не только в `postgresql.conf`, но и в рамках сессии с помощью команды `SET`;

# synchronous\_standby\_names

- `synchronous_standby_names = '*'`
  - Ждем подтверждения от одной любой реплики
- `synchronous_standby_names = ANY 2(node1,node2,node3);`
  - Коммит на кворум
  - Появилось в версии 10
- Другие варианты [1] не очень полезны

[1]: <https://www.postgresql.org/docs/current/static/runtime-config-replication.html>

# Что осталось за кадром

- Шардинг и решардинг
  - Реализуемы при помощи логической репликации и словаря на базе Consul;
- Распределенные транзакции
  - Percolator-like транзакции достаточно просто реализуемы;
  - Дают snapshot isolation, возможна аномалия write skew;
  - Подойдет для большинства приложений, в частности Oracle только SI и предлагает;
  - **Важно!** Нужно использовать и при записи, и при чтении;
- Готовых решений нет, или я про них не знаю
- Если вам повезет, то никогда не придется всем этим заниматься
- Кое-какие подробности - <https://eax.me/sharding/>

# Дополнительные материалы

- <https://www.postgresql.org/docs/10/static/index.html>
- <https://www.consul.io/>
- <https://github.com/sorintlab/stolon/>
- <https://raft.github.io/>
- <https://jepsen.io/>
- <https://kubernetes.io/>
- + есть статьи на <https://eax.me/>

## Вопросы и ответы.

- [a.lubennikova@postgrespro.ru](mailto:a.lubennikova@postgrespro.ru)
- [a.alekseev@postgrespro.ru](mailto:a.alekseev@postgrespro.ru)
- Telegram: <https://t.me/dbmsdev>