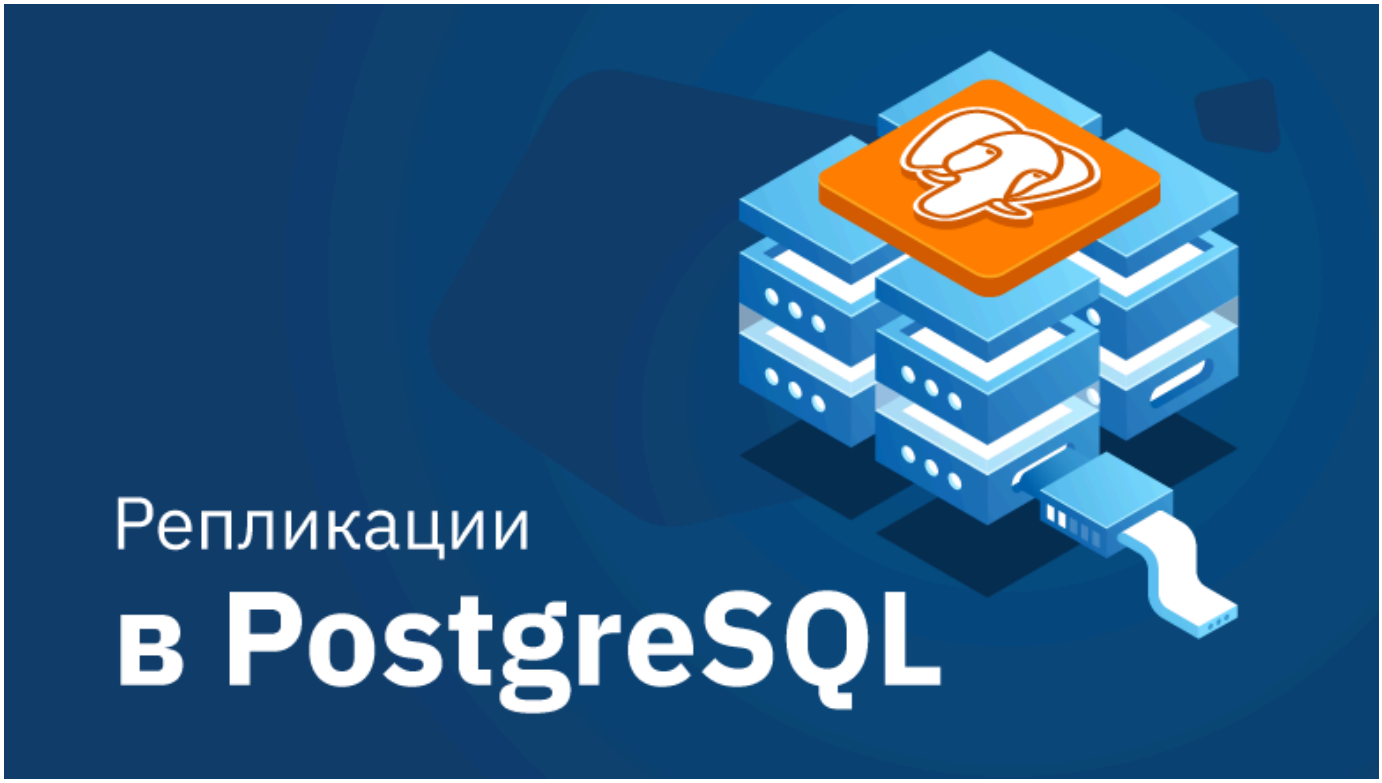

Andrey_Biryukov
17 янв 2023 в 17:55

Репликации в PostgreSQL

6 мин 53K

Блог компании OTUS, PostgreSQL*



Сейчас трудно себе представить «боевую» инсталляцию любой серьезной СУБД в виде единственного инстанса. Конечно, некоторые приложения требуют для своей работы использование локальных баз данных, но если мы говорим о сетевом многопользовательском режиме работы, то здесь использование только одной инсталляции это очень плохая идея.

Основной проблемой единственной инсталляции естественно является надежность. В случае падения сервера нам потребуется некоторое, возможно значительное, время на восстановление. Так восстановление террабайтной базы может занять несколько часов.

Да и исправный бэкап есть не всегда, но об этом мы уже говорили в предыдущей статье.

Кроме надежности, второй существенной проблемой единственной инсталляции является производительность. Даже при использовании виртуальной инфраструктуры мы не можем до бесконечности «одалживать» ресурсы у гипервизора. Что делать когда закончились память и ядра у сервера? Нужно горизонтальное масштабирование, то есть установка

дополнительных инстансов. Также на производительность конкретного инстанса влияют ресурсоемкие операции, такие как построение отчетов, обработка данных и т. д. Да и бэкап тоже лучше делать с менее нагруженного инстанса.

Таким образом, мы приходим к тому, что для полноценного функционирования серьезной БД нам необходимо делать реплики. Наличие второго инстанса позволит использовать его в случае выхода из строя основного, то есть обеспечить отказоустойчивость. Кроме того, все ресурсоемкие операции можно перенести на реплику, чтобы максимально разгрузить мастер.

Но и при репликации не следует забывать о некоторых важных моментах. Так, если у вас произошло повреждение данных на логическом уровне (повреждены индексы, некорректные данные в таблицах) то все эти неприятные изменения сохраняться и на реплику. Поэтому репликация это хорошая защита от физических сбоев, а вот от логических сбоев нам поможет бэкап.

И завершая теоретическую часть этой статьи хотелось бы прояснить различия между понятиями резервирование и резервное копирование. Готовя материал к предыдущей статье я сталкивался с тем, что на некоторых ресурсах бэкап тоже называют термином резервирование. Это неверно, резервирование это обеспечение отказоустойчивости работы ресурсов в режиме реального времени. Резервирование не допускает простоев системы, по крайней мере таких масштабных как при восстановлении из бэкапов. Так что резервирование это репликации и кластеры, а резервное копирование это бэкап.

Ну а теперь перейдем уже непосредственно к репликациям в PostgreSQL.

Виды репликаций

Для начала напомню о WAL — журналом предзаписи транзакций. Во избежание нарушений целостности в структуре баз данных, PostgreSQL сначала записывает эти изменения в файлы журнала WAL и только потом в базу. Журналы WAL нужны для того, чтобы в случае сбоя сервера можно было восстановить незафиксированные данные. Ну а применительно к теме сегодняшней статьи, WAL используется и для репликации данных.

Репликация на серверах СУБД PostgreSQL бывает двух видов: физическая и логическая. При физической репликации у нас на сервер реплики передается поток WAL записей. Одним из основных достоинств физической репликации является простота в конфигурировании и использовании, так как используется простое побайтовое копирование с одного сервера на другой. Также из-за своей простоты, физическая репликация потребляет меньше ресурсов.

Но у физических репликаций есть и свои недостатки. По аналогии с физическим бэкапом здесь также требуются одинаковые версии PostgreSQL и операционной системы.

При этом, также должны быть идентичны в том числе и аппаратные компоненты, такие как архитектура процессора. Также при физических бэкапах возможна репликация только всего кластера, на подчиненном инстансе нельзя создать никакую отдельную таблицу, даже временную. Полная идентичность основному серверу.

В зависимости от архитектуры самой БД возможна существенная нагрузка на инфраструктуру, так как нужно передавать все изменения в файлах данных полностью.

Логическая репликация работает по принципу подписки. Мастер сервер выступает в роли поставщика, который публикует изменения, происходящие в базе, а серверы реплики, выступающие в роли подписчиков получают и применяют эти изменения у себя.

На программном уровне логические репликации используют репликационные идентификаторы (как правило, это первичный ключ). Снимок данных с таблицы на основном инстансе публикуется и передается подписчику. После этого, все изменения, происходящие с данными на стороне основного сервера также публикуются и передаются подписчику, как любят писать во многих источниках «в режиме реального времени». Хотя если быть занудным то термин «работа в реальном времени» применим только к операционным системам реального времени RTOS, таким как QNX. А для классических ОС Linux/Windows все-таки уместен термин «режим близкий к реальному времени».

Так или иначе, как только изменения происходят на мастере, они сразу же реплицируется на слейва. Соответственно, подчиненный сервер применяет изменения в той же последовательности, что и публикующий узел, тем самым гарантируется транзакционная целостность.

Логическая репликация также имеет ряд преимуществ перед физической. Прежде всего, это независимость от используемых непосредственно на серверах форматов хранения данных. То есть, мастер и слейв могут иметь различные представления данных на диске, разные ОС и архитектуры. Оба сервера участника репликации могут выступать для разных объектов, как в роли поставщика, так и в роли подписчика. Таким образом мы можем использовать двусторонний обмен, например одна таблица может реплицироваться с первого сервера на второй, а другая со второго сервера на первый. К преимуществам логической репликации можно отнести возможность использовать разные ОС на разных инстансах, а также возможность выборочной репликации отдельных объектов кластера. В результате мы можем снизить нагрузку на сеть, так как при логической репликации объем передаваемых данных меньше.

Сценарии использования

Для физических репликаций основное применение это создание отказоустойчивых кластеров, когда необходимо иметь точную копию БД на другом инстансе.

Логическая репликация предполагает больше различных вариантов использования. Например, мы можем объединить несколько баз в одну, для целей анализа, реплицировать данные между разными кофигурациями PostgreSQL, развернутыми как под Windows, так и под Linux. Можно настроить срабатывание триггеров для отдельных изменений, когда их получает подписчик и другие интересные манипуляции с реплицируемыми данными.

Перейдем к практике.

Для начала настроим физическую репликацию между двумя серверами Ubuntu 22.04.

Далее в примерах

192.168.222.142 - Master

192.168.222.136 - Slave

Прежде всего необходимо установить две абсолютно одинаковые инсталляции ОС. В моем примере это будет Ubuntu. Далее обновляемся и устанавливаем СУБД на обоих серверах:

```
sudo apt install postgresql postgresql-contrib
```

Далее работаем в консоли сервера Master.

Под аккаунтом postgres необходимо создать пользователя для репликации:

```
otus@otus:~$ sudo -i -u postgres
[sudo] password for otus:
postgres@otus:~$ createuser --replication -P rep_user
Enter password for new role:
Enter it again:
```

В моем примере таким пользователем является rep_user. Далее, смотрим расположение conf файла:

```
postgres@otus:~$ psql -c 'SHOW config_file;'
              config_file
----------------------------------------
 /etc/postgresql/14/main/postgresql.conf
(1 row)
```

Нам необходимо внести в этот файл некоторые правки:

```
postgres@otus:~$ nano /etc/postgresql/14/main/postgresql.conf
```

В файл необходимо внести следующие правки:

```
archive_mode = on

archive_command = 'cp %p /oracle/pg_data/archive/%f'

max_wal_senders = 10

wal_keep_segments = 50

wal_level = replica

wal_log_hints = on
```

Далее необходимо внести дополнения в файл pg_hba.conf, добавив имя пользователя для репликаций и IP адрес подчиненного сервера:

```
host replication rep_user 192.168.222.136/32 scram-sha-256
```

И в завершении настройки перезапускаем Postgresql:

```
postgres@otus:~$ systemctl restart postgresql
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ===
Authentication is required to restart 'postgresql.service'.
Authenticating as: otus
Password:
==== AUTHENTICATION COMPLETE ===
```

На этом настройка сервера Master завершена. Теперь перейдем к настройке сервера Slave. Прежде всего правим файл postgresql.conf:

```
listen_addresses = 'localhost, 192.168.1.136'
```

Для внесения дальнейших изменений нам необходимо остановить сервер:

```
postgres@otus:~$ systemctl stop postgresql
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ===
Authentication is required to stop 'postgresql.service'.
Authenticating as: otus
Password:
==== AUTHENTICATION COMPLETE ===
```

Так как в режиме подчиненного сервера у нас все данные реплицируются с основного, нам необходимо удалить файлы из каталога main. Если у вас инсталляция не новая, можно предварительно сделать бэкап содержимого этого каталога.

```
postgres@otus:~$ rm -rf /var/lib/postgresql/14/main/*
```

Теперь проведем проверку работы процесса репликации. Для этого используем команду pg_basebackup с адресом основного сервера и именем пользователя для репликаций:

```
postgres@otus:~$ pg_basebackup -R -h 192.168.222.142 -U rep_user -D /var/lib/postgresql/14/main -P
Password:
26251/26251 kB (100%), 1/1 tablespace
```

Теперь можно запустить сервис PosthreSQL на подчиненном сервере:

```
postgres@otus:~$ systemctl restart postgresql
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ===
Authentication is required to restart 'postgresql.service'.
Authenticating as: otus
Password:
==== AUTHENTICATION COMPLETE ===
```

Логические репликации

В завершении статьи рассмотрим настройку логической репликации в PostgreSQL. В качестве примера я выполню репликацию базы Otus. Прежде всего необходимо в файле postgresql.conf сменить значение параметра wal_level:

```
wal_level = logical_                    # minimal, replica, or logical
```

В уже знакомом нам файле pg_hba.conf на мастере добавляем строку с IP адресом подчиненного сервера:

```
host otus postgres 192.168.222.136/32 trust
```

Далее делаем дамп всей БД и дамп схемы базы Otus.

```
postgres@otus:~$ pg_dumpall --database=otus --host=192.168.222.142 --no-password --globals-only --no
-privileges | psql_
```

```
postgres@otus:~$ pg_dump --dbname otus --host=192.168.222.142 --no-password --create --schema-only |
 psql
```

Теперь необходимо создать публикацию на стороне сервера мастер:

```
postgres=# CREATE PUBLICATION db_pub FOR ALL TABLES;
CREATE PUBLICATION
```

И подписку на стороне подчиненного сервера:

```
postgres=# CREATE SUBSCRIPTION db_sub CONNECTION 'host=192.168.222.142 dbname=otus' PUBLICATION db_p
ub;
```

Теперь все изменения в базе Otus будут реплицироваться на Slave И данный сервер уже можно будет использовать для выполнения резервного копирования, построения отчетов и других ресурсоемких задач.

Заключение

В этой статье мы рассмотрели такую важную тему как базовая настройка физических и логических репликаций в СУБД PostgreSQL. Дальше, с помощью репликаций уже можно строить более сложные схемы взаимодействия между серверами БД.

Также напоминаю про бесплатный вебинар курса "*PostgreSQL для администраторов баз данных и разработчиков*" посвященный маленьким хитростям GROUP BY. На вебинаре вспомним, как устроен GROUP BY и рассмотрим его на наглядных примерах, оптимизируем работу группировки в связке с индексами, разберемся с особенностями группировки строк в PostgreSQL, а также изучим несколько полезных приемов для работы с GROUP BY.

- Зарегистрироваться на бесплатный вебинар

Теги: postgresql, postgresql scaling, postgresql replication

Хабы: Блог компании OTUS, PostgreSQL

21 Карма 134 Рейтинг

@Andrey_Biryukov
Пользователь

Комментарии 16

baldr
17 янв 2023 в 19:07

Да-да, команды для консоли скриншотами, криво обрезанные и разного масштаба! Классика. Предлагаю всю статью загнать в jpg!

Вроде и тема интересная, и материал авторский.. Но скриншоты - просто кровь из глаз.. Плюс красные предупреждения еще мешают.

```
archive_command = 'cp %p /oracle/pg_data/archive/%f'
```

Вот тут oracle как-то режет глаз. Понятно что просто имя папки, но для статьи про PostgreSQL лучше б заменить.

Ответить

JuriM
18 янв 2023 в 00:26

Статья явно проходная для набора очков

Ответить

OptimumOption
18 янв 2023 в 06:32

Скриншоты - отличная идея для исключения бездумного копипаста (rm -rf /)

Ответить

dbax
18 янв 2023 в 09:27

Прям как у Джека Воробья:

- Это вывод консоли?

- Лучше - это скриншот вывода консоли!

+2 Ответить

Koyotter
17 янв 2023 в 20:04

Вопрос: можно ли реализовать автоматическое переключение с мастера на реплику при отказе мастера? т. е если сервер с мастер-базой упал, реплика сама становится новым мастером, а упавший сервер после поднятия становится репликой? Сколько гуглил и смотрел оф. доки постгреса, ответа точного не нашёл.

0 Ответить

Volkodlak
17 янв 2023 в 21:05

repmgr, patroni, и так далее, правда это варианты с тремя нодами (мастер и две реплики, или мастер,реплика, свидетель) ..для двух нод похоже только мультимастер от PGPro

+2 Ответить

oxff
18 янв 2023 в 00:38

Без третьей ноды/свидетеля не разрулить split brain.

+1 Ответить

Volkodlak
18 янв 2023 в 16:24

вроде я об этом и написал))

0 Ответить

oxff
18 янв 2023 в 16:33

Я просто решил пояснить для задавшего вопрос. Что это не техническое ограничение существующих решений, а скорее жизненная необходимость. Две ноды это однозначно ручной фэйловер.

Тут всё упирается в кворум. Можно иметь 2 ноды БД с автофэйловером, но потребуется как минимум 3 ноды DCS (etcd и т.п.).

0 Ответить

PrinceKorwin
17 янв 2023 в 21:13

Patroni? Правда автоматом, вроде, только в одну сторону.

0 Ответить

Volkodlak
17 янв 2023 в 21:38

когда работа бывшего мастера будет восстановлена, он увидит нового мастера и станет репликой, естественно тут возможны разные ситуации и их решения, но в целом может и splitbrain разрулить (после того как вы наигрались с какими-то "синтетическими" остановками кластера), отмотав "лишнее" назад)

0 Ответить

JuriM
18 янв 2023 в 00:25

Патрони работает в любом направлении в любое время, вы наверное с repmgr путаете

0 Ответить

erogov
18 янв 2023 в 00:02

> Журналы WAL нужны для того, чтобы в случае сбоя сервера можно было восстановить незафиксированные данные.

Вы уверены?

+2 Ответить

Throwable
18 янв 2023 в 10:29

То есть в обоих случаях сначала дампим и копируем все данные с основного сервера на реплику, а затем слушаем изменения. Не совсем понятно можно ли это делать online, и что будет происходить в промежутке между копированием реплики и началом подписки, ведь данные на мастере постоянно изменяются. Или для поднятия реплики нужно мастер тоже останавливать?

P.S. вроде как были автоматические решения для HA типа PgPool.

0 Ответить

НЛО прилетело и опубликовало эту надпись здесь

sirbusby
18 янв 2023 в 17:33

> В файл необходимо внести следующие правки:
>
> `wal_keep_segments`

Начиная с PostgreSQL 13 - wal_keep_size

> `wal_log_hint`

Зачем?

+1 Ответить

Публикации

ЛУЧШИЕ ЗА СУТКИ ПОХОЖИЕ

unxed
2 часа назад

far2l в Ubuntu 24.04 LTS готов для повседневного использования

Простой 5 мин 1.7K

Дайджест

+30 20 25

bodyawm
22 часа назад

Игры на промышленных девайсах? Легко! Портируем эмуляторы и Wolfenstein 3D на ТСД за 500 рублей на практике

Средний 9 мин 3K

Ретроспектива

+23 25 11

GidraVydra
22 часа назад

Всё делается из нефти и газа. Или нет?

Простой 6 мин 4.1K

Мнение

+19 14 31

ImagineTables
17 часов назад

Ещё один набор параметров для разблокировки Ютуба: теперь мобильный

Простой 2 мин 4.6K

Туториал

+18 54 7

R3EQ
5 часов назад

История моего увлечения астрономией

Простой 8 мин 824

+16 11 8

Andrey2008
23 часа назад

SafeCode – конференция, которой мне не хватало

3 мин 2K

Мнение

+14 5 1

Affdey
17 часов назад

Новые технологии и древние звуки

Средний 12 мин 1.1K

Туториал

+13 11 10

Bright_Translate
3 часа назад

Как хакеры «красной команды» в роли злоумышленников проникают в здания и системы

Простой 6 мин 576

Кейс Перевод

+11 4 0

smart_alex
4 часа назад

Что показывает ваш мультиметр (напряжение или погоду на Марсе)?

Простой 8 мин 1.9K

Туториал

+11 19 25

kliment_01
4 часа назад

Не Flipper Zero единым: хакерский мультитул из старого смартфона

7 мин 1.6K

+11 22 2

Показать еще

ВАКАНСИИ КОМПАНИИ «OTUS»

Преподаватель онлайн курса PostgreSQL Advanced
OTUS · Можно удаленно

Наставник онлайн курса Highload Architect
OTUS · Можно удаленно

Преподаватель онлайн курса Cassandra для разработчиков и администраторов
OTUS · Можно удаленно