tantor

От реплики до высокодоступного Кластера PostgreSQL

О себе

С 2005 года начал свой путь в IT, и с тех пор профессиональная деятельность неразрывно связана с базами данных, в частности, с PostgreSQL в различных его аспектах..

Руководил командами разработки железа, софта и автоматизации.

Постоянный спикер на PGBootCamp

На повестке дня

Методы репликации в PostgreSQL. Асинхронная и синхронная репликация.

Patroni: управление репликацией

Комбинирование технологий для создания высокодоступного кластера

Практика

tantor

Методы репликации в PostgreSQL:

- Asynchronous Streaming Replication (Асинхронная поточная репликация)
- Synchronous Replication (Синхронная репликация)
- Logical Replication (Логическая репликация)
- Bi-Directional Replication (Двунаправленная репликация)

wal_level = replica

Назначение
Применение
Ресурсоемкость

wal_level = logical

Назначение
Применение
Ресурсоемкость

Выбор между replica и logical

Выбор уровня зависит от ваших требований к репликации и от того, какие функции вам необходимы.

Для традиционной физической репликации всего кластера подойдет replica.

Если вам необходима возможность репликации на уровне отдельных таблиц или баз данных, интеграция с другими системами через логическую репликацию, выбирайте logical.

Уровень logical также необходим, если вы планируете использовать функции, требующие декодированиялогов, например, для аудита или сложной репликации данных.

tantor

На мастере

```
wal_level = replica
max_wal_senders = 10
wal_keep_size = 1024

host  replication  replicator  адрес_сети_/маска  md5
```

WAL (Write-Ahead Logging)

Принцип работы
Цели использования WAL:
- Устойчивость к сбоям
- Репликация
- Быстродействие

Физическое хранение
Управление и настройка

На реплике

```
pg_basebackup -h master_ip_address \
              -D /var/lib/postgresql/15/main \
              -U replicator \
              -P -v -W -p 5432 \
              -R -X stream -C -S pgstandby1
```

tantor

Проверка состояния репликации с помощью SQL-запросов

На мастере: SELECT * FROM pg_stat_replication;

На реплике: SELECT * FROM pg_stat_wal_receiver;

Проверка последовательности транзакций (LSN)

На мастере: SELECT pg_current_wal_lsn();

На реплике: SELECT pg_last_wal_receive_lsn();

Проверка с помощью pg_replication_slots

На мастере: SELECT * FROM pg_replication_slots;

tantor

На мастере

synchronous_commit в on

synchronous_standby_names *

Высокая доступность и масштабируемость - это две ключевые характеристики, которые играют важную роль в проектировании и обслуживании систем баз данных, включая PostgreSQL.

Непрерывная работа: Высокая доступность означает, что система PostgreSQL остается доступной для пользователей даже в случае отказов или сбоев компонентов.
Это достигается за счет использования методов репликации, автоматического обнаружения и восстановления отказов, а также механизмов балансировки нагрузки.

Отказоустойчивость: PostgreSQL может быть настроен таким образом, чтобы автоматически обнаруживать отказы и переключаться на резервные реплики или заменять неработоспособные узлы. Это позволяет минимизировать простои и обеспечивает непрерывную работу системы.

Механизмы восстановления: PostgreSQL предоставляет механизмы для резервного копирования данных и восстановления базы данных из них в случае сбоев. Это позволяет быстро восстановить работоспособность системы после критических событий.

Patroni: Patroni - это инструмент для автоматизации управления кластером PostgreSQL. Он использует концепцию "Бесконечного Журнала" (Consul, etcd, ZooKeeper) для хранения метаданных и координации между узлами кластера. Patroni позволяет автоматически переключаться на резервные узлы в случае сбоя, обеспечивая высокую доступность и минимизацию времени простоя.

Etcd: etcd - это распределенное хранилище ключ-значение, используемое Patroni для хранения метаданных и конфигураций кластера. Он обеспечивает централизованное управление конфигурацией, а также обеспечивает согласованность в распределенной среде. Etcd играет ключевую роль в обеспечении согласованности состояния кластера и обмене информацией между узлами.

tantor

patronictl edit-config

synchronous_mode: true

synchronous_mode_strict: true

synchronous_node_count: 1

patronictl switchover

Keepalived — это программное обеспечение для обеспечения высокой доступности Linux-серверов через использование протокола маршрутизации VRRP (Virtual Router Redundancy Protocol).

Keepalived часто используется для автоматического переключения на резервный сервер в случае сбоя основного, тем самым обеспечивая непрерывную доступность сервисов и приложений.

Кроме того, Keepalived поддерживает балансировку нагрузки и мониторинг состояния через LVS (Linux Virtual Server).

Основные применения:

- **Высокая доступность:** Автоматически переносит IP-адреса и сервисы на резервные устройства при сбое.
- **Балансировка нагрузки:** Распределяет входящий трафик между несколькими серверами для оптимизации ресурсов и повышения производительности.

HAProxy — это высокопроизводительный балансировщик нагрузки и прокси-сервер для TCP и HTTP-приложений.

Он широко известен своей стабильностью, надежностью и гибкостью в обработке миллионов запросов в секунду.

HAProxy часто используется для улучшения производительности и надежности
веб-приложений путем распределения нагрузки между несколькими серверами.

Основные особенности:

- **Балансировка нагрузки:** Распределяет запросы к веб-сайтам или приложениям на несколько серверов для улучшения отклика и устойчивости
- **Масштабируемость:** Легко масштабируется для обработки возрастающего трафика, обеспечивая эффективное управление сессиями и подключениями
- **Высокая доступность:** Может автоматически определять сбои серверов и перенаправлять трафик на здоровые узлы.

PgBouncer — это легковесный пулер соединений для PostgreSQL.
Он управляет подключениями к базе данных, сохраняя их в пуле для повторного использования, что значительно уменьшает затраты на установление новых соединений и улучшает общую производительность системы.

PgBouncer особенно полезен в средах с высокой загрузкой, где частое открытие и закрытие подключений может создавать значительную нагрузку.

Основные функции включают:

- **Управление соединениями:** Снижает накладные расходы на установление соединений, переиспользуя уже существующие.
- **Мультиплексирование:** Позволяет множеству клиентских подключений использовать одно или несколько подключений к серверу баз данных.
- **Гибкая настройка:** Поддерживает различные режимы управления соединениями, включая сессионный, транзакционный и пул соединений на уровне операторов.

https://github.com/TantorLabs/pg_cluster/tree/tantor-classic

https://t.me/isomet

Спасибо за внимание!