# Logiczna replikacja strumieniowa w SZBD Postgres

Replikacja strumieniowa zaprezentowana na poprzednich zajęciach jest replikacją fizyczną. Oznacza to, że:

* Zmiany we wszystkich obiektach podstawowej bazy danych są replikowane do czuwających baz danych.
* W danym momencie tylko jedna baza danych jest bazą podstawową, do której użytkownicy mogą wprowadzać zmiany.
* Czuwające bazy danych są uruchomione w trybie tylko do odczytu, w związku z tym nie mogą posiadać własnych obiektów oprócz tych, które są replikowane z podstawowej bazy danych.

Wady te łagodzi zastosowanie logicznej replikacji strumieniowej. Mechanizm ten, podobnie jak wcześniej poznana fizyczna replikacja strumieniowa, korzysta z dziennika bazy danych do transferu zmian w obiektach między bazami danych. W nazewnictwie systemu Postgres pliki dziennika bazy danych nazywane są WAL (ang. Write Ahead Log). Mechanizm logicznej replikacji strumieniowej wykorzystuje dwa rodzaje baz danych: *dostawcy* (ang. providers) i *subskrybenci* (ang. subscribers). *Dostawcy* dostarczają dane, natomiast *subskrybenci* replikują dane *dostawców*. Dane można modyfikować zarówno w bazach typu *dostawca* jak i *subskrybent*. Jednakże tylko zmiany wprowadzane w dostawcach są replikowane do subskrybentów. Zmiany wprowadzone w subskrybentach pozostają lokalne i mogą być przyczyną konfliktów. Logiczna replikacji strumieniowa może być kaskadowa, to znaczy, że subskrybent może być dostawcą dla innych subskrybentów.

Celem zajęć jest zapoznanie się z logiczną replikacją strumieniową dostarczaną przez system Postgres od wersji 10.

## Przygotowanie środowiska

1. Zaloguj się do maszyny wirtualnej jako użytkownik rbd używając hasła RBD#7102.
2. Otwórz okno terminala, który nazwiemy terminalem pomocniczym.
3. Usuń z klastra Kubernetes obiekty, które zostały utworzone w poprzednim tutorialu, wykonaj w terminalu pomocniczym następujące polecenia:

kubectl delete -f rbd-citus.yaml

kubectl delete pvc pgsql-rbd-citus-disk-pgsql-citus-sts-0

kubectl delete pvc pgsql-rbd-citus-disk-pgsql-citus-sts-1

kubectl delete pvc pgsql-rbd-citus-disk-pgsql-citus-sts-2

kubectl delete -f rbd-citus-coord.yaml

kubectl delete pvc postgres-db-rbd-citus-coord-1-0

kubectl delete pvc postgres-db-rbd-citus-coord-2-0

kubectl delete pvc postgres-db-rbd-citus-coord-3-0

kubectl delete pvc postgres-db-rbd-citus-coord-4-0

kubectl delete -f rbd-citus-worker1.yaml

kubectl delete pvc postgres-db-rbd-citus-worker1-1-0

kubectl delete pvc postgres-db-rbd-citus-worker1-2-0

kubectl delete -f rbd-citus-worker2.yaml

kubectl delete pvc postgres-db-rbd-citus-worker2-1-0

kubectl delete pvc postgres-db-rbd-citus-worker2-2-0

1. Skorzystamy ze StatefulSet, które zostały zdefiniowane w pierwszym tutorialu, w tym celu wykonaj w terminalu pomocniczym następujące polecenia:

k3d image import rbd/postgres17:1.1 --cluster RBDcluster

kubectl apply -f rbd1.yaml

kubectl apply -f rbd2.yaml

1. Monitoruj postęp wdrażanie manifestów:

kubectl get sts --watch

1. Po zakończeniu wdrażania manifestów otwórz 2 nowe zakładki w oknie terminala, pierwszą nazwij provider-1, a drugą subscriber-1.
2. W terminalu provider-1 uruchom poniższe polecenie w celu przyłączenia się do bazy danych, którą nazwiemy provider-1.

psql -U postgres -h localhost -p 5432

Użyj hasła rbd1 w celu uwierzytelnienia użytkownika postgres.

**Uwaga**: jeżeli narzędzie psql nieoczekiwanie traci połączenie ze serwerem zamiast adresu localhost użyj jednego z adresów węzła loadbalancer. W celu pozyskania tych adresów użyj polecenia: kubectl get svc. Wykorzystaj jeden z adresów z kolumny EXTERNAL-IP.

1. W bazie danych provider-1 zwiększ ilość informacji generowanych do plików dziennika bazy danych, które umożliwią logiczną replikację. W tym celu wykonaj polecenie:

alter system set wal\_level=logical;

1. W terminalu pomocniczym zrestartuj Pod obsługujący bazę danych provider-1. Użyj polecenia:

kubectl rollout restart sts pgsql-rbd1

1. W terminalu pomocniczym zrestartuj połączenie do bazy danych provider-1. Możesz w tym celu wykonać dwukrotnie to samo polecenie, np.:

show wal\_level;

## Strumieniowa replikacja logiczna z jednym dostawcą.

1. W bazie danych provider-1 utwórz użytkownika *repl*, który będzie służyć do uwierzytelnienia połączenia z bazy danych subskrybenta do bazy danych dostawcy. Uruchom polecenie:

CREATE ROLE repl WITH REPLICATION LOGIN PASSWORD 'rbd1repl';

1. W bazie danych provider-1 utwórz obiekty, których zawartość będzie replikowana, skorzystaj z poniższego polecenia:

\i ~/loggers/loggers.sql

1. W bazie danych provider-1 nadaj użytkownikowi *repl* prawa odczytu do obiektów, które zostały utworzone w poprzednim kroku, użyj poniższe polecenie:

GRANT select ON ALL TABLES IN SCHEMA public TO repl;

Polecenie to nadaje uprawnienia do wszystkich obecnie istniejących tabel w schemacie public, czyli tylko tych, które zostały utworzone w poprzednim kroku

1. W celu udostępnienia wybranych tabel mechanizmowi logicznej replikacji utwórz w bazie danych provider-1 publikację o nazwie meas\_publication.

CREATE PUBLICATION meas\_publication;

1. W bazie danych provider-1 dodaj do publikacji meas\_publication tabelę organizations. Skorzystaj z poniższego polecenia:

ALTER PUBLICATION meas\_publication ADD TABLE organizations;

1. W bazie danych provider-1 sprawdź za pomocą poniższego polecenia jakie publikacje znajdują się w bazie danych:

select pubname from pg\_publication;

1. W bazie danych provider-1 zobacz jakie tabele zostały opublikowane, w tym celu uruchom poniższe polecenie:

select \* from pg\_publication\_tables;

1. W terminalu subscriber-1 uruchom poniższe polecenie w celu przyłączenia się do bazy danych, którą nazwiemy subscriber-1.

psql -U postgres -h localhost -p 5433

1. W bazie danych subscriber-1 utwórz obiekty, do których będą replikowana dane z obiektów dostawcy, skorzystaj z poniższego polecenia:

\i ~/loggers/loggers.sql

Mechanizm logicznej replikacji nie propaguje operacji DDL. Dzięki temu podejściu schematy obiektów w bazie danych dostawcy i subskrybenta mogą być różne.

1. W bazie danych subscriber-1 utwórz subskrypcję meas\_subscription, która umożliwi replikację danych z bazy danych dostawcy do bazy danych subskrybenta, uruchom poniższe polecenie:

CREATE SUBSCRIPTION meas\_subscription CONNECTION

'host=pgsql-rbd1-lb port=5432 user=repl password=rbd1repl dbname=postgres'

PUBLICATION meas\_publication;

1. W bazie danych subscriber-1 wyświetl utworzone subskrypcje, skorzystaj z poniższego polecenia:

select \* from pg\_subscription;

1. W bazie danych provider-1 wstaw dane do tabeli organizations:

\i ~/loggers/organizations.dmp

1. W bazie danych subscriber-1 wyświetl status replikacji:

select subname as substription\_name, relname as table\_name,

case srsubstate

when 'i' then 'initialize'

when 'd' then 'data is being copied'

when 's' then 'synchronized'

when 'r' then 'ready (normal replication)'

end

from pg\_subscription\_rel r join pg\_subscription s on r.srsubid=s.oid

join pg\_class c on r.srrelid=c.oid;

1. W bazie danych subscriber-1 sprawdź liczbę wierszy w tabeli organizations. Czy replikacja zmian się powiodła? [Raport]
2. W bazie danych provider-1 dodaj tabelę loggers do publikacji meas\_publication. [Raport]
3. W bazie danych provider-1 wstaw dane do tabeli loggers:

\i ~/loggers/loggers.dmp

1. W bazie danych subscriber-1 sprawdź liczbę wierszy w tabeli loggers. Czy replikacja zmian się powiodła? [Raport]
2. Powtórz krok 13, czy w wyniku znajdują się informacje o statusie replikacji tabeli loggers? [Raport]
3. W bazie danych subscriber-1 odśwież subskrypcję meas\_subscription:

ALTER SUBSCRIPTION meas\_subscription REFRESH PUBLICATION;

1. Powtórz krok 17 oraz 18.
2. Sprawdź replikację wyników operacji aktualizacji i usuwania danych z tabeli loggers w bazie danych provider-1. [Raport]
3. Sprawdź co się stanie jeżeli w trakcie wykonywania zmian w bazie danych provider-1 baza danych subscriber-1 będzie niedostępna, wykonaj następujący eksperyment [Raport]:
   1. Zmniejsz liczbę replik Pod bazy danych subscriber-1 do zera
   2. W bazie danych provider-1 wprowadź zmiany do replikowanego obiektu.
   3. Zwiększ liczbę replik Pod bazy danych subscriber-1 do jednej
   4. W bazie danych subscriber-1 sprawdź czy zmiany zostały zreplikowane.
4. Sprawdź definicję parametru [wal\_keep\_size](https://www.postgresql.org/docs/14/runtime-config-replication.html#GUC-WAL-KEEP-SIZE). Jak jego wartość może mieć wpływ na wynik eksperymentu z kroku 22? [Raport]

## Strumieniowa replikacja logiczna z wieloma dostawcami.

1. Utwórz nowy StatefulSet, który będzie zawierał bazę danych provider-2. W tym celu skopiuj plik manifestów rbd2.yaml do pliku rbd3.yaml. Za pomocą ulubionego edytora zamień w pliku rbd3.yaml wszystkie napisy rbd2 na rbd3, zamień również port wykorzystywany przez loadbalancer z 5433 na 5434. Wykonaj wdrożenie manifestów znajdujących się w pliku rbd3.yaml.
2. W oknie terminalu otwórz nową zakładkę w oknie terminala, nazwij ją provider-2.
3. W terminalu provider-2 uruchom poniższe polecenie w celu przyłączenia się do bazy danych, którą nazwiemy provider-2.

psql -U postgres -h localhost -p 5434

1. W bazie danych provider-2:
   1. Zmień odpowiednio wartość parametru wal\_level i zrestartuj bazę danych provider-2
   2. Utwórz tabele korzystając ze skryptu ~/loggers/loggers.sql
   3. Utwórz użytkownika repl
   4. Nadaj użytkownikowi repl uprawnienia do obiektów utworzonych za pomocą skryptu ~/loggers/loggers.sql
   5. Utwórz publikację meas2\_publication i dodaj do niej tabelę organizations
2. W bazie danych subscriber-1 utwórz subskrypcję meas2\_subscription, która umożliwi replikację danych z bazy danych provider-2.
3. W bazie danych provider-2 dodaj organizację o identyfikatorze -40. Czy zmiany przepropagowały się do baz danych subscriber-1? [Raport]
4. Przeprowadź eksperyment wprowadzenia konfliktu przez wstawienie do bazy danych provider-2 organizacji, która już istnieje w bazie danych provider-1 i subscriber-1. Wykonaj bazie danych provider-2 sekwencję następujących operacji:

insert into organizations values(50, 'Konflitowa', 'CLIENT');

select pg\_current\_wal\_lsn ();

insert into organizations values(-50, 'Nie Konflitowa', 'CLIENT');

Funkcja pg\_current\_wal\_lsn() zwraca bieżącą pozycję zapisu w pliku WAL.

1. W bazie danych subscriber-1 sprawdź wartości dla organizacji o identyfikatorach 50 i -50. Czy propagacja zmian się powiodła? [Raport]
2. Sprawdź w logu bazy danych subscriber-1 co się wydarzyło. W tym celu w terminalu pomocniczym najpierw zwiększ limit otwartych plików:

ulimit -n 65536

i następnie uruchom poniższe polecenie:

kubectl logs pgsql-rbd2-0 -f

1. W celu rozwiązania konfliktu pominiemy w bazie danych subscriber-1 wstawienie "konfliktowej" organizacji.
   1. Wykonaj poniższe polecenie:

select s.subname, r.\*

from pg\_subscription s join pg\_replication\_origin\_status r

on concat('pg\_',s.oid)=r.external\_id;

Odczytaj wartość kolumny external\_id dla subskrypcji meas2\_subscription.

* 1. Wykonaj poniższe polecenie, użyj odpowiednio external\_id uzyskanego w punkcie 10.1 oraz WAL LSN uzyskanego w punkcie 7:

select pg\_replication\_origin\_advance('external\_id','WAL LSN');

1. Sprawdź w terminalu pomocniczym nowe wpisy w logu bazy danych subscriber-1. Czy replikacja została wznowiona? [Raport]
2. W bazie danych subscriber-1 sprawdź wartości dla organizacji o identyfikatorach 50 i -50. Wyjaśnij uzyskany wynik [Raport]

## Przydatne polecenia do diagnostyki błędu zbyt wielu otwartych plików

Sprawdzenie liczby aktualnie otwartych plików przez procesy użytkownika rbd:

sudo lsof -u rbd | wc -l

Podwyższenie liczby otwartych w bieżącej powłoce

ulimit -n 65536

Podwyższenie liczby otwartych plików przez użytkownika, wymaga restartu systemu

sudo tee -a /etc/systemd/user.conf <<EOF

# Zwiększenie limitu otwartych plików (nofile) użytkownika

DefaultLimitNOFILE=65536

EOF

Podwyższenie liczby otwartych plików w systemie, wymaga restartu systemu

sudo tee -a /etc/security/limits.conf <<EOF

\* soft nofile 65536

\* hard nofile 65536

root soft nofile 65536

root hard nofile 65536

EOF

sudo systemctl daemon-reexec

relogin

reboot