Replication lab

Setup

Na początku:

```
./initdb -D ~/test_db
```

Korzystam ze swojego usera: przemek. Mogę normalnie krozystać z programów w `/usr/lib/postgresql/12/bin` ale był problem z odpalenie serwera:

```
2024-04-29 15:28:05.031 CEST [25297] FATAL:  could not create lock file
"/var/run/postgresql/.s.PGSQL.5440.lock": Permission denied
```

Rowiązanie było nadanie uprawnień do katalogu mojemu userowi:

```
sudo chmod a+w /var/run/postgresql
```

Zmienilem port i utworzyłem bazę:

```
./pg_ctl -D /home/przemek/test_db -l ~/test_db/logfile start
```

Test setupu

Łączę się z bazą:

```
psql -p 5440 -U przemek -d postgres
```

Tworzę tabelę i wykonuje kilka prostych operacji:

```
postgres=# create table tbl (id int PRIMARY KEY, name varchar);
CREATE TABLE
postgres=# select * from tbl
postgres-# ;
 id | name
----+------
(0 rows)

postgres=# INSERT INTO tbl (id, name) VALUES (1, 'Przemek');
```

```
INSERT 0 1
postgres=# INSERT INTO tbl (id, name) VALUES (2, 'Ola fasola');
INSERT 0 1
postgres=# INSERT INTO tbl (id, name) VALUES (3, 'Mati');
INSERT 0 1
postgres=# select * from tbl
;
 id |    name
----+------------
  1 | Przemek
  2 | Ola fasola
  3 | Mati
(3 rows)


postgres=# INSERT INTO tbl (id, name) VALUES (4, 'Lechu');
INSERT 0 1

postgres=# DELETE FROM tbl where id < 3;
DELETE 2
postgres=# select * from tbl;
 id | name
----+-------
  3 | Mati
  4 | Lechu
(2 rows)

postgres=# DELETE FROM tbl;
DELETE 2
postgres=# select * from tbl;
 id | name
----+------
(0 rows)

postgres=# DROP TABLE tbl;
DROP TABLE
postgres=# select * from tbl;
ERROR:  relation "tbl" does not exist
LINE 1: select * from tbl;

postgres=# \q
```

Zamykam instancję:

```
./pg_ctl -D /home/przemek/test_db -l ~/test_db/logfile stop
```

Ćwiczenie 1 - replikacja strumieniowa fizyczna

Utworzenie instancji primary

Tworzymy instancję primary:

```
./initdb -D ~/primary_db
```

Zmieniam config:

```
listen_addresses = '*'
port = 5440
```

Uruchamiam:

```
./pg_ctl -D /home/przemek/primary_db -l ~/primary_db/logfile start
```

Nowy user:

```
./createuser repuser --replication -p 5440
```

W pliku `pg_hba.conf` dodaję:

```
local replication repuser trust
```

i resetuję instancję serwera.

Utowrzenie repliki

Wykonuję:

```
./pg_basebackup -h 127.0.0.1 -U repuser -p 5440 -D ~/replica_db -R -C -S
slot_name -c fast
```

```
przemek@przemek-laptop:~/replica_db$ ls -la
total 136
drwx------ 19 przemek przemek  4096 Apr 29 16:22 .
drwxr-xr-x 99 przemek przemek  4096 Apr 29 16:22 ..
-rw-------  1 przemek przemek   225 Apr 29 16:22 backup_label
drwx------  5 przemek przemek  4096 Apr 29 16:22 base
drwx------  2 przemek przemek  4096 Apr 29 16:22 global
-rw-------  1 przemek przemek  5255 Apr 29 16:22 logfile
drwx------  2 przemek przemek  4096 Apr 29 16:22 pg_commit_ts
```

```
drwx------  2 przemek przemek  4096 Apr 29 16:22 pg_dynshmem
-rw-------  1 przemek przemek  4831 Apr 29 16:22 pg_hba.conf
-rw-------  1 przemek przemek  1636 Apr 29 16:22 pg_ident.conf
drwx------  4 przemek przemek  4096 Apr 29 16:22 pg_logical
drwx------  4 przemek przemek  4096 Apr 29 16:22 pg_multixact
drwx------  2 przemek przemek  4096 Apr 29 16:22 pg_notify
drwx------  2 przemek przemek  4096 Apr 29 16:22 pg_replslot
drwx------  2 przemek przemek  4096 Apr 29 16:22 pg_serial
drwx------  2 przemek przemek  4096 Apr 29 16:22 pg_snapshots
drwx------  2 przemek przemek  4096 Apr 29 16:22 pg_stat
drwx------  2 przemek przemek  4096 Apr 29 16:22 pg_stat_tmp
drwx------  2 przemek przemek  4096 Apr 29 16:22 pg_subtrans
drwx------  2 przemek przemek  4096 Apr 29 16:22 pg_tblspc
drwx------  2 przemek przemek  4096 Apr 29 16:22 pg_twophase
-rw-------  1 przemek przemek     3 Apr 29 16:22 PG_VERSION
drwx------  3 przemek przemek  4096 Apr 29 16:22 pg_wal
drwx------  2 przemek przemek  4096 Apr 29 16:22 pg_xact
-rw-------  1 przemek przemek   309 Apr 29 16:22 postgresql.auto.conf
-rw-------  1 przemek przemek 26753 Apr 29 16:22 postgresql.conf
-rw-------  1 przemek przemek     0 Apr 29 16:22 standby.signal
```

Odpalamy:

```
./pg_ctl -D /home/przemek/replica_db -l ~/replica_db/logfile start
```

Sprawdzamy `pg_stat_replication` w bazie primary.

```
postgres=# select * from pg_stat_replication;
-[ RECORD 1 ]----+------------------------------
pid              | 67016
usesysid         | 16385
usename          | repuser
application_name | walreceiver
client_addr      | 127.0.0.1
client_hostname  |
client_port      | 40880
backend_start    | 2024-04-29 20:06:32.955219+02
backend_xmin     |
state            | streaming
sent_lsn         | 0/30001C0
write_lsn        | 0/30001C0
flush_lsn        | 0/30001C0
replay_lsn       | 0/30001C0
write_lag        |
flush_lag        |
replay_lag       |
sync_priority    | 0
sync_state       | async
```

```
reply_time       | 2024-04-29 20:09:33.65523+02
```

Test replikacji

W primary bazie:

```
postgres=# create table test_tbl (id int primary key, name varchar);
CREATE TABLE
postgres=# insert into test_tbl (id, name) values (1, 'Ola');
INSERT 0 1
postgres=# insert into test_tbl (id, name) values (2, 'ALa');
INSERT 0 1
postgres=# insert into test_tbl (id, name) values (3, 'Kot');
INSERT 0 1
postgres=#
```

w bazie replice:

```
postgres=# select * from test_tbl;
 id | name
----+------
  1 | Ola
  2 | ALa
  3 | Kot
(3 rows)
```

Baza primary:

```
postgres=# delete from test_tbl where id >= 1;
DELETE 3
postgres=# truncate table test_tbl;
TRUNCATE TABLE
```

Replika:

```
postgres=# select * from test_tbl;
 id | name
----+------
(0 rows)
```

Awaria bazy głównej:

```
./pg_ctl -D /home/przemek/primary_db -l ~/primary_db/logfile stop
```

Promocja repliki:

```
przemek@przemek-laptop:/usr/lib/postgresql/12/bin$ ./pg_ctl -D
/home/przemek/replica_db -l ~/replica_db/logfile promote
waiting for server to promote.... done
server promoted
```

Zadanie dodatkowe - multi-standby setup

Tworzę dwa dodatkowe backupy:

```
./pg_basebackup -h 127.0.0.1 -U repuser -p 5440 -D ~/replica_db2 -R -C -S
slot_name2 -c fast

./pg_basebackup -h 127.0.0.1 -U repuser -p 5440 -D ~/replica_db3 -R -C -S
slot_name3 -c fast
```

Zmieniam port w plikach `postgresql.conf` na 5442 i 5443. Następnie startuje:

```
./pg_ctl -D /home/przemek/replica_db2 -l ~/replica_db2/logfile start
./pg_ctl -D /home/przemek/replica_db3 -l ~/replica_db3/logfile start
```

Tworzę nową tabelę w primary:

```
postgres=# create table test_tbl2 (id int primary key, cost int);
CREATE TABLE

postgres=# insert into test_tbl2 (id, cost) values (1, 10);
INSERT 0 1
postgres=# insert into test_tbl2 (id, cost) values (2, 200);
INSERT 0 1
postgres=# insert into test_tbl2 (id, cost) values (3, 25);
INSERT 0 1
```

Otwieram postgresa w replica_db3:

```
psql -p 5443 -U przemek -d postgres
```

Sprawdzam czy replika się udała:

```
postgres=# select * from test_tbl2;
 id | cost
----+------
  1 |   10
  2 |  200
  3 |   25
(3 rows)
```

Dane zostały zbackupowane!

Zadanie dodatkowe - cascade setup

Tworzymy replikę (UWAGA - jako primary podajemy port 5441, czyli naszą pierwszą replikę), zmieniamy port na 5444 i uruchamiamy:

```
./pg_basebackup -h 127.0.0.1 -U repuser -p 5441 -D ~/replica_db4 -R -C -S
slot_name4 -c fast
./pg_ctl -D /home/przemek/replica_db4 -l ~/replica_db4/logfile start
```

Insertuje dodatkowe wiersze z primary:

```
postgres=# insert into test_tbl2 (id, cost) values (4, 50);
INSERT 0 1
postgres=# insert into test_tbl2 (id, cost) values (5, 1000);
INSERT 0 1
postgres=# insert into test_tbl2 (id, cost) values (6, 22000);
INSERT 0 1
```

Wchodzimy na replikę:

```
psql -p 5444 -U przemek -d postgres
```

Widzimy wyniki po replikacji:

```
ostgres=# select * from test_tbl2;
 id | cost
----+-------
  1 |    10
  2 |   200
  3 |    25
  4 |    50
  5 |  1000
  6 | 22000
```