**Отчет по ДЗ (Репликация)**

Репликация настроена с помощью patroni

1. Запуск приложения для тестов

* docker compose --profile postgres up -d (без репликации )
* docker compose --profile patroni up -d (с репликацией)

Далее в зависимости от выбранного профиля:

* Подождать пока запустится api и вручную запустить users-loader
* Подождать пока запустится api-highload и вручную запустить users-loader-highload

2. Нагрузочное тестирование по чтению

Запускаем 2 раза с разными профилями postgres/patroni

В папках homeworks\9\replica и homeworks\9\no\_replica   
приведены исходники и результаты тестов (в виде htlm отчетов) 2-х запросов на чтение. Нагрузочное тестирование показало что с репликацией запросы стали отрабатывать без ошибок (нагрузка на чтение распределилась между узлами patroni)

2. Тестирование по аварийной остановке master

Запускаем с профилем patroni

В папке homeworks\9\write приведены исходники и результаты теста на аварийную остановку master узла для этого были проведены следующие действия через patronictl  
  
через patronictl edit-config настраиваем кворумную синхронную репликацию в секцию parameters добавляем:

synchronous\_commit: "on"

synchronous\_standby\_names: "ANY 1 (\*)"

patronictl pause

временно переводим кластер Patroni в режим обслуживания и отключаем автоматическое переключение при отказе.

postgres@haproxy:~$ patronictl list

+ Cluster: patroni (7474585561654186007) ---+----+-----------+

| Member | Host | Role | State | TL | Lag in MB |

+----------+------------+---------+---------+----+-----------+

| patroni1 | 172.18.0.3 | Replica | running | 2 | 0 |

| patroni2 | 172.18.0.2 | Replica | running | 2 | 0 |

| patroni3 | 172.18.0.4 | Leader | running | 2 | |

+----------+------------+---------+---------+----+-----------+

Проводим нагрузку на запись и останавливаем мастер  
  
postgres@haproxy:~$ patronictl list

+ Cluster: patroni (7474585561654186007) ---+----+-----------+

| Member | Host | Role | State | TL | Lag in MB |

+----------+------------+---------+---------+----+-----------+

| patroni1 | 172.18.0.3 | Replica | running | 2 | 0 |

| patroni2 | 172.18.0.2 | Replica | running | 2 | 0 |

+----------+------------+---------+---------+----+-----------+

По html отчету видим что из 1000 с ошибкой было 632 запроса (успешных стало быть 368) проверяем на patroni1  
docker exec -it patroni1 bash

postgres@patroni1:~$ psql -U highload

highload=# select count(\*) from users;

count

-------

368

(1 row)

промоутим patroni1

postgres@haproxy:~$ patronictl failover

Candidate ['patroni1', 'patroni2'] []: patroni1

Are you sure you want to failover cluster patroni? [y/N]: y

2025-02-23 12:20:38.41223 Successfully failed over to "patroni1"

+ Cluster: patroni (7474585561654186007) ---+----+-----------+

| Member | Host | Role | State | TL | Lag in MB |

+----------+------------+---------+---------+----+-----------+

| patroni1 | 172.18.0.3 | Leader | running | 2 | |

| patroni2 | 172.18.0.2 | Replica | running | 2 | 0 |

+----------+------------+---------+---------+----+-----------+

Проверяем тот же запрос на patroni2

docker exec -it patroni2 bash

postgres@patroni2:~$ psql -U highload

psql (16.8 (Debian 16.8-1.pgdg120+1))

Type "help" for help.

highload=# select count(\*) from users;

count

-------

368

(1 row)

Делаем вывод что транзакции не были потеряны