## Лабораторная работа №3

1) Создать конфигурационные файлы и запустить docker compose

```
pg-slave | 2024-12-23 23:54:04,212 INFO: no action. I am (postgresql1), the leader with the lock
pg-master | 2024-12-23 23:54:05,969 INFO: no action. I am (postgresql0), a secondary, and following a leader (postgresql1)
```

2) Создаем таблицу и записываем данные в мастер ноду

```
pg-slave public ___ *<pg-slave> Script X

create table test_table (
   id int,
   data varchar(255)
);

insert into test table values(1, 'тестовая запись')
```

3) Убеждаемся, что во второй ноде таблица так же создалась

```
    Фруктивней профессионня профе
```

4) Проверяем, что мы не можем записывать в зависимую ноду

```
insert into test table values (2, 'test test'):

□ Pesynьтат 1 X

□ SQL Error [25006]: ERROR: cannot execute INSERT in a read-only transaction
Позиция ошибки:
```

5) Добавляем балансировщик в docker-compose, перезапускаем контейнеры и проверяем, что все запустилось ок

```
pg-slave | 2024-12-24 00:11:01.454 UTC [22] LOG: received SIGHUP, reloading configuration files

pg-slave | server signaled

pg-slave | 2024-12-24 00:11:01.455 UTC [22] LOG: parameter "synchronous_standby_names" changed to "postgresql0"

pg-slave | 2024-12-24 00:11:01.664 UTC [49] LOG: standby "postgresql0" is now a synchronous standby with priority 1

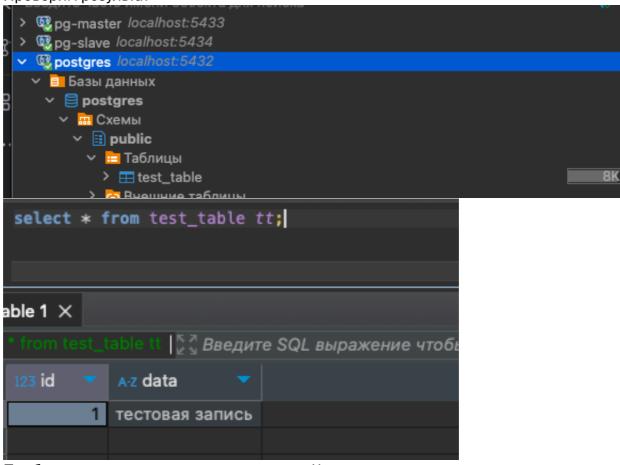
pg-slave | 2024-12-24 00:11:01.664 UTC [49] STATEMENT: START_REPLICATION SLOT "postgresql0" 0/7000000 TIMELINE 5

pg-slave | 2024-12-24 00:11:03,576 INFO: Synchronous standby status assigned to ('postgresql0')

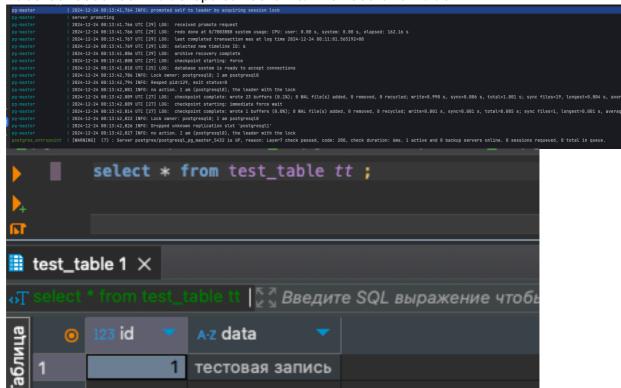
pg-slave | 2024-12-24 00:11:03,576 INFO: no action. I am (postgresql1), the leader with the lock

postgres_entrypoint | [WARNING] (7): Server postgres/postgresqlpg_slave_5432 is UP, reason: Layer7 check passed, code: 200, check duration: 8ms.
```

6) Проверим результат



7) Пробуем отключить текущую мастер ноду. Через некоторое время вторичная нода самопровозгласилась мастером, а балансировщик обновил путь. Работоспособность кластера самостоятельно восстановилась.



## Ответы на доп вопросы:

- 1) Разница между директивами заключается в том, что expose только уведомляет Docker о портах, которые контейнер слушает, но не делает их доступными извне. Это может быть полезно для внутренней связи между контейнерами в одной сети. Напротив, ports пробрасывает порты контейнера на хост-машину, что позволяет к ним подключаться из внешней среды.
- 2) При простом перезапуске контейнера образ не будет собираться заново. Так же изменения конфигов не будут отражены в контейнере, пока образ не будет сбилжен заново. При изменении Dockerfile образвсе еще не будет автоматически пересобран, пока явно не будет указан флаг build при запуске compose или билд не будет запущен вручную.

3)