1. Асинхронная репликация

1. Запустить master из dockre-compose
```
  postgres:
    container_name: postgres
    image: postgres:15.4-alpine3.18
    environment:
      POSTGRES_DB: "otusdb"
      POSTGRES_USER: "sa"
      POSTGRES_PASSWORD: "medtex"
    volumes:
      - ../migrations:/docker-entrypoint-initdb.d
      - otusdb-data:/var/lib/postgresql/data
    ports:
     - "5432:5432"
```

3. Запустить скрипт
```
bash -c "
       echo \"host    replication  replicator  172.18.0.0/16  md5\" >>
/var/lib/postgresql/data/pg_hba.conf &&
       echo \"wal_level = replica\" >> /var/lib/postgresql/data/postgresql.conf &&
       echo \"max_wal_senders = 4\" >> /var/lib/postgresql/data/postgresql.conf &&
       echo \"ssl = off\" >> /var/lib/postgresql/data/postgresql.conf
      "
```

4. Перезапустить master

```
docker restart postgres
```

5. Сделать backup для реплик

```
docker exec -it postgres bash
```

```
mkdir /pgslave
```

```
pg_basebackup -h postgres -D /pgslave -U replicator -v -P --wal-method=stream
```

6. Скопировать dir на хост

```
docker cp postgres:/pgslave pgslave
```

7. Скопировать standby.signal, чтобы сделать реплику

```
#F5 to ($PWD)\pgslave
```

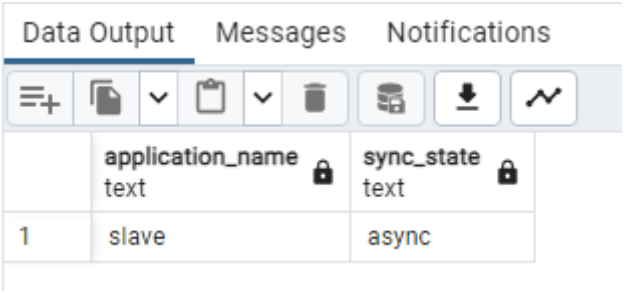
8. Изменить postgresql.conf на реплике

```
primary_conninfo = 'host=postgres port=5432 user=replicator password=medtex
application_name=slave'
```

9. Запустить реплику
```
docker run -dit -v $PWD/pgslave/:/var/lib/postgresql/data -e POSTGRES_PASSWORD=medtex -p
```

15433:5432 --network=docker_default --restart=unless-stopped --name=pgslave postgres:15.4-alpine3.18

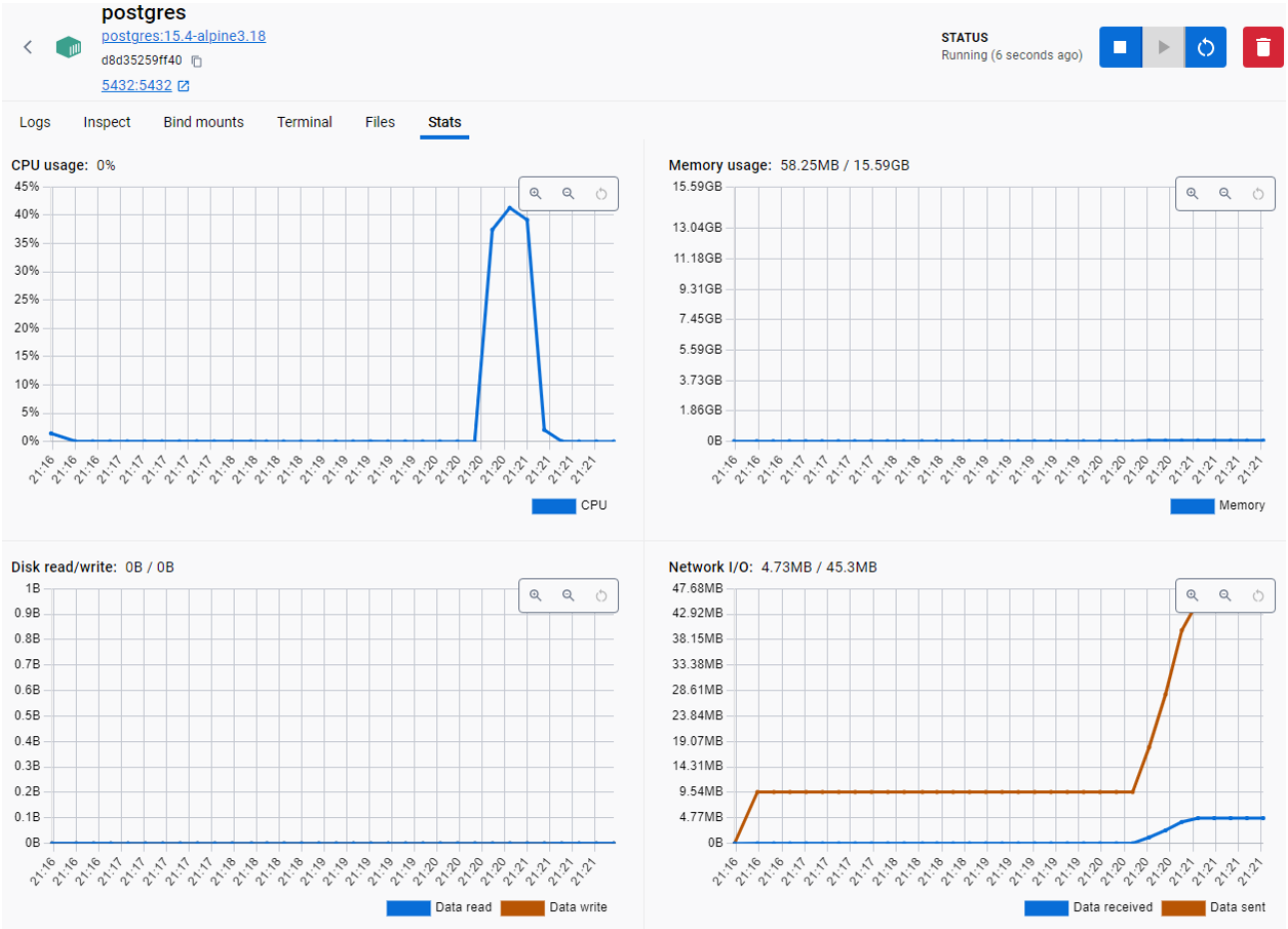
8 Проверить мастер
select application_name, sync_state from pg_stat_replication;



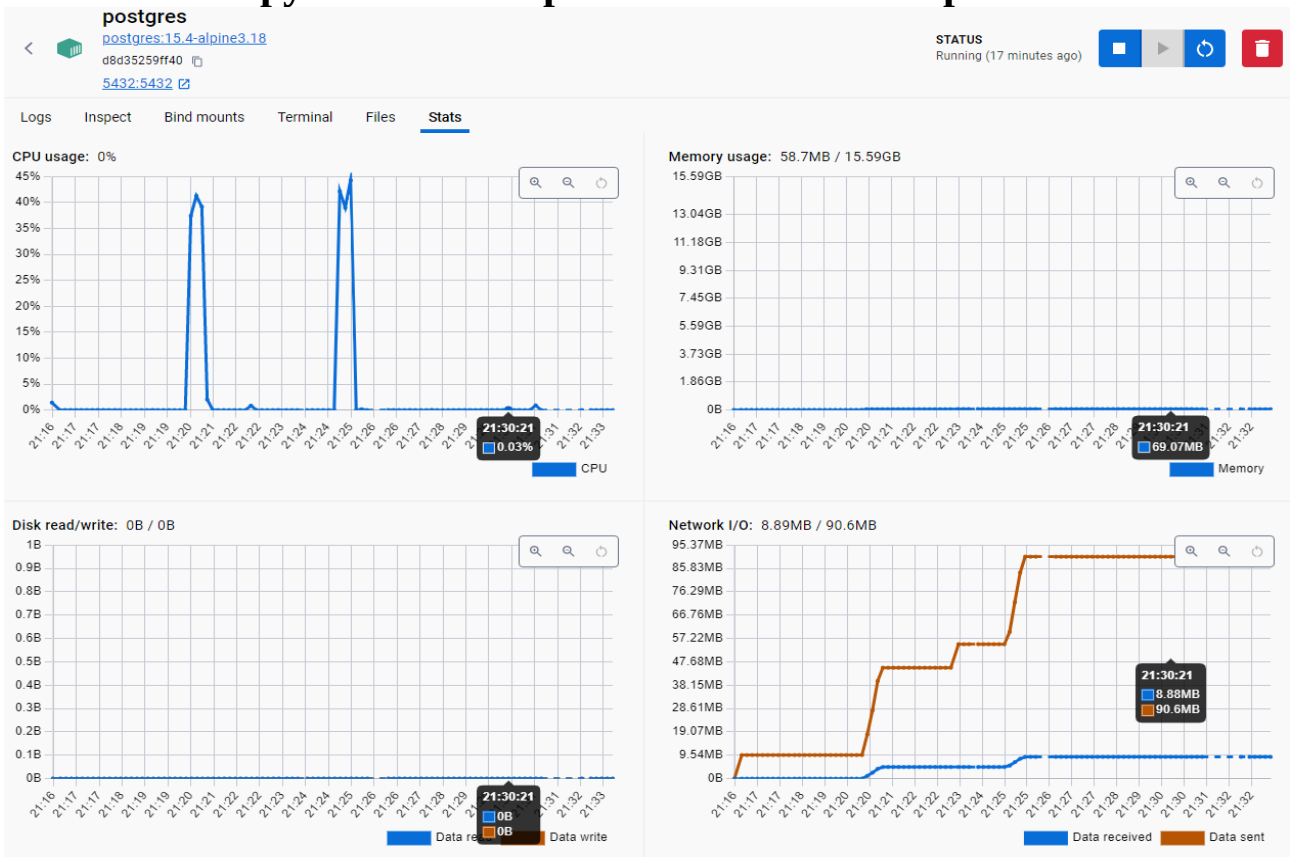
Проверяем передачу нагрузки с мастера на реплику

./ab -n 10000 -c 100 http://localhost:5266/user/search?Name=%25%D0%B5%D0%B2"&"Surname=%25%D1%80%D0%B0%D0%BC%D0%BE%25

Нагрузка на мастере до включения реплики



Нагрузка на мастере после включения реплики



Нагрузка перешла на реплику

2. Синхронная репликация

---

1. Запустить 2ую реплику
docker exec -it postgres bash

mkdir /pgslave2

pg_basebackup -h postgres -D /pgslave2 -U replicator -v -P --wal-method=stream

docker cp postgres:/pgslave2 pgslave2

#Copy standby.signal, to make replica

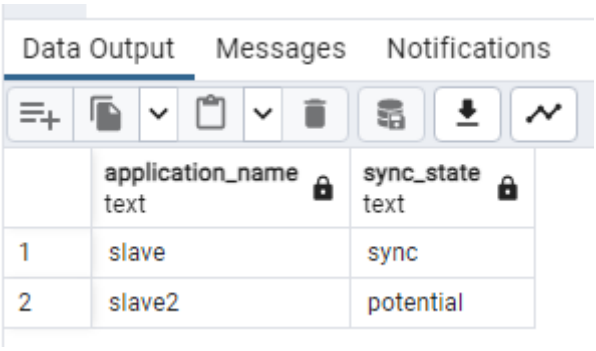
primary_conninfo = 'host=postgres port=5432 user=replicator password=medtex application_name=slave2'

docker run -dit -v $PWD/pgslave2/:/var/lib/postgresql/data -e POSTGRES_PASSWORD=medtex -p 25433:5432 --network=docker_default --restart=unless-stopped --name=pgslave2 postgres:15.4-alpine3.18

2. Включаем синхронную репликацию на мастере:
Изменить postgresql.conf

synchronous_commit = on
synchronous_standby_names = 'FIRST 1 (slave, slave2)'

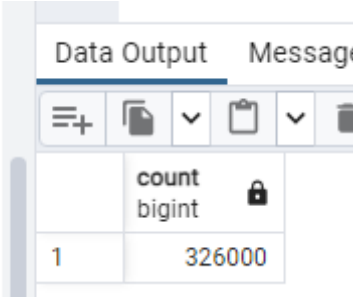
Перезапустить конфигурацию
select pg_reload_conf();



	application_name text	sync_state text
1	slave	sync
2	slave2	potential

3. Создаем нагрузку и останавливаем мастер
4. Смотрим результат транзакции на обоих репликах

SELECT count(*) FROM public.users;



	count bigint
1	326000

5. Смотрим результат транзакции на мастере

SELECT count(*) FROM public.users;

Результат аналогичен репликам. **Потерь транзакций нет.**

3. Повышение реплики

1. Запромоутим реплику pgslave

select * from pg_promote();

Изменить postgresql.conf

synchronous_commit = on
synchronous_standby_names = 'ANY 1 (postgres, slave2)'

2. Подключим вторую реплику к новому мастеру

primary_conninfo = 'host=pgslave port=5432 user=replicator password=pass
application_name=slave2'

3. Проверяем что реплика переключилась на новый мастер

select application_name, sync_state from pg_stat_replication;