Homework_Lesson18_DB_2

Цель: научиться реализовывать отказоустойчивое решение для баз данных (на примере Postgresql либо MySQL)

1. Создать два сервера, установить на них PostgreSQL либо MySQL и подключить их к одной сети.

2. Настроить репликацию между серверами, чтобы изменения, вносимые на одном сервере, автоматически реплицировались на другой. (Зеркалирование)

3. Настроить отказоустойчивость, используя репликацию и механизм автоматического переключения между серверами. (Кластер*) Кластер по желанию и по возможности.

4. Проверить работу отказоустойчивого решения, симулируя отказ одного из серверов и убедившись, что второй сервер продолжает работу и все данные сохранены с обоими видами репликации (Зеркалирование подразумевает, что реплика автоматически подключится к главное ноде в случае, когда она станет вновь доступна)

Опционально:

● Настроить систему резервного копирования, чтобы регулярно создавать бэкапы данных и сохранять их на отдельном сервере (либо на отдельном диске, либо папке) через SSH.

● Документировать все шаги по настройке и проверке отказоустойчивого решения и подготовить отчет о выполненной работе.

Выполнение задания:

1. Создаем два сервера в VirtualBox с названиями VM_Ubuntu и VM2_Ubuntu с одинаковыми дистрибутивами ОС Ubuntu 22.04.

Первая ВМ VM_Ubuntu, выделено 4 GB ОЗУ, 25 GB свободного пространтсва, настроено две адаптера сети NAT и Сетевой мост.

Вторая ВМ VM2_Ubuntu аналогичная с такими же параметрами, как и в первой.

Репликация между двумя серверами применяется для создания отказоустойчивых кластеров, в которых необходимо иметь точную копию БД на другом инстансе. В нашем случае будем использовать PostgreSQL. Для настройки нужно определить, кто будет Master, а кто Slave:

1) VM_Ubuntu IP 192.168.100.14/24 - Master
2) VM2_Ubuntu IP 192.168.100.11/24 - Slave

Переходим к установке postgresql на серверах. На обоих ВМ устанавливаем PostgreSQL:

```
$ sudo apt install postgresql postgresql-contrib      \\установка PostgreSQL
$ sudo pg_config --version                            \\проверка версии PostgreSQL
```

```
root@makarov-VirtualBox:~# pg_config --version
PostgreSQL 14.15 (Ubuntu 14.15-0ubuntu0.22.04.1)
root@makarov-VirtualBox:~#
```

1.1 В начале работаем с ВМ VM_Ubuntu, сервера Master (IP - 192.168.100.14/24).

Под аккаунтом postgres необходимо создать пользователя для репликации:

```
$ sudo -i -u postgres          \\ логинимся под пользователем postgres
```

$ createuser --replication -P user_rep \\ создание пользователя для репликации

```
root@makarov-VirtualBox:~# su - postgres
postgres@makarov-VirtualBox:~$ createuser --replication -P user_rep
Enter password for new role:
Enter it again:
```

Проверяем, добавился ли пользователь user_rep в СУБД PostgreSQL:

$ psql \\ заходим в интерактивную консоль PostgreSQL
postgres=# \du \\ выводим список пользователей в СУБД

```
postgres@makarov-VirtualBox:~$ psql
psql (14.15 (Ubuntu 14.15-0ubuntu0.22.04.1))
Type "help" for help.

postgres=# \du
                                List of roles
 Role name |                         Attributes                         | Member of
-----------+------------------------------------------------------------+-----------
 postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS | {}
 user_rep  | Replication                                                | {}

postgres=#
```

Пользователь user_rep создался успешно

$ psql -c ‘SHOW config_file;’ \\ смотрим расположение конфигурационного файла

```
postgres@makarov-VirtualBox:~$ psql -c 'SHOW config_file;'
               config_file
----------------------------------------
 /etc/postgresql/14/main/postgresql.conf
(1 row)

postgres@makarov-VirtualBox:~$
```

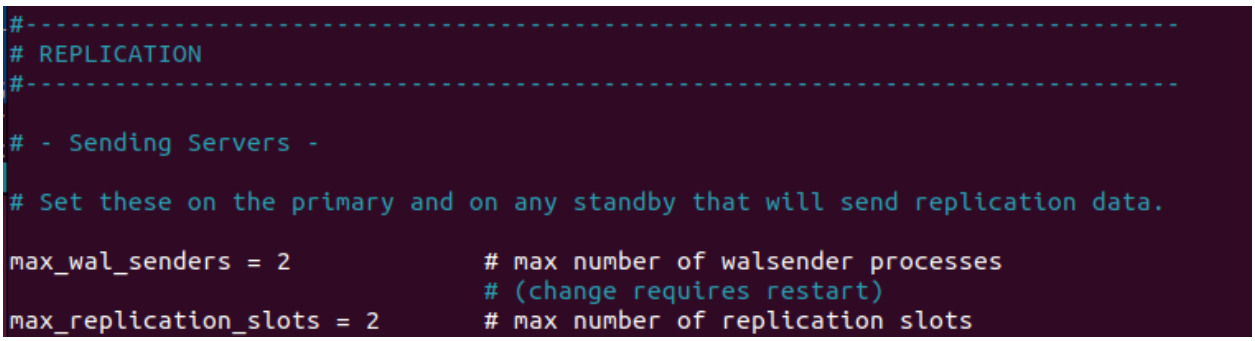
Правим файл postgresql.conf:

listen_addresses = 'localhost, 192.168.100.14'
wal_level = replica
max_wal_senders = 2
max_replication_slots = 2
hot_standby = on
hot_standby_feedback = on

```
# - Connection Settings -

listen_addresses = 'localhost, 192.168.100.14'          # what IP address(es) to listen on;
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost'; use '*' for all
```

Правки файла postgresql.conf

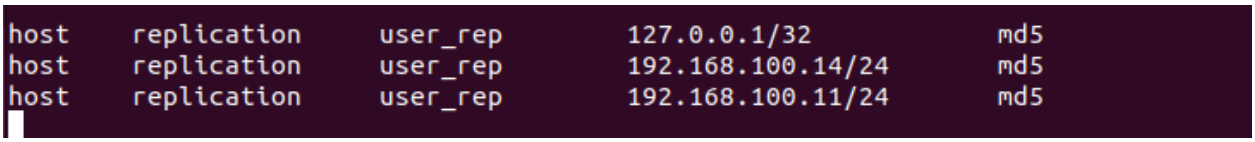
```
# - Settings -

wal_level = replica                     # minimal, replica, or logical
                                        # (change requires restart)
```



Правки файла postgresql.conf

Правки файла postgresql.conf

Также делаем правки конфигурационного файла pg_hba.conf, находится в том же каталоге, что и файл postgresql.conf. В этом файле мы разрешаем подключение к базе данных replication пользователю user_rep с локального сервера 192.168.100.14(MASTER) и сервера 192.168.100.11(SLAVE):



Правки файла pg_hba.conf

После всех добавлений/изменений перезапускаем службу postgresql:

$ systemctl restart postgresql

1.2 Переходим к настройке сервера 192.168.100.11(Slave).
Проверяем пути до конфигурационного файла:

$ su - postgres -c "psql -c 'SHOW data_directory;'"
$ su - postgres -c "psql -c 'SHOW config_file;'"

```
root@makarov2-VirtualBox:~# su - postgres -c "psql -c 'SHOW data_directory;'"
       data_directory
-----------------------------
 /var/lib/postgresql/14/main
(1 row)

root@makarov2-VirtualBox:~# su - postgres -c "psql -c 'SHOW config_file;'"
               config_file
----------------------------------------
 /etc/postgresql/14/main/postgresql.conf
(1 row)

root@makarov2-VirtualBox:~#
```

Вывод путей конфигурационных файлов

Перед настройкой, чтоб не испортить данные, останавливаем postgres:

```
root@makarov2-VirtualBox:~# systemctl stop postgresql
root@makarov2-VirtualBox:~# systemctl status postgresql
○ postgresql.service - PostgreSQL RDBMS
     Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; vendor pr>
     Active: inactive (dead) since Sat 2025-01-18 16:58:26 +03; 4s ago
   Main PID: 995 (code=exited, status=0/SUCCESS)
        CPU: 3ms

сту 16 18:17:52 makarov2-VirtualBox systemd[1]: Starting PostgreSQL RDBMS...
сту 16 18:17:52 makarov2-VirtualBox systemd[1]: Finished PostgreSQL RDBMS.
сту 18 16:58:26 makarov2-VirtualBox systemd[1]: postgresql.service: Deactivated>
сту 18 16:58:26 makarov2-VirtualBox systemd[1]: Stopped PostgreSQL RDBMS.
lines 1-10/10 (END)
```

Удаляем содержимое каталога /vat/lib/postgresql/14/main/* *(если на данном сервере, есть какие-нибудь важные данные (базы данные), то сохраняем содержимое, перед удалением)*:

$ rm -rf /vat/lib/postgresql/14/main/* \\ удаление содержимого каталога */main

Запускаем команду копирования данных с сервера 192.168.100.14 (Master) с директории var/lib/postgresql/14/main:

$ su - postgres -c "pg_basebackup --host=192.168.100.14 --username=user_rep --pgdata=/var/lib/postgresql/14/main --wal-method=stream --write-recovery-conf"

```
root@makarov2-VirtualBox:~# su - postgres -c "pg_basebackup --host=192.168.100.14 --username=user_rep --pgdata=/var/lib/postgr
esql/14/main --wal-method=stream --write-recovery-conf"
pg_basebackup: error: connection to server at "192.168.100.14", port 5432 failed: Connection timed out
        Is the server running on that host and accepting TCP/IP connections?
```

Выдало ошибку, что не может связаться с сервером 192.168.100.14. В нашей ситуации оказалось, что на сервере 192.168.100.14 (Master) был поднят брандмауэр ufw, который блокировал соединение из вне. Есть два варианта, первый, остановить/выключить ufw-брандмауэр, второй - добавить порт 5432:

1) ufw disable \\ небезопасно делать
2) ufw allow 5432 \\ добавляем порт 5432 в брандмауэр ufw

Пробуем ещё раз запустить нашу команду:

$ su - postgres -c "pg_basebackup --host=192.168.100.14 --username=user_rep --pgdata=/var/lib/postgresql/14/main --wal-method=stream --write-recovery-conf"

```
root@makarov2-VirtualBox:~# su - postgres -c "pg_basebackup --host=192.168.100.14 --username=user_
rep --pgdata=/var/lib/postgresql/14/main --wal-method=stream --write-recovery-conf"
Password:
```

После удачного запроса и удачном коннекте к хосту 192.168.100.14 запросит пароль пользователя user_rep

Успешное клонирование данных.
Далее нужно отредактировать параметр в конфигурационном файле postgresql.conf

listen_addresses = ‘localhost, 192.168.100.11’

```
#------------------------------------------------------------------------------
# CONNECTIONS AND AUTHENTICATION
#------------------------------------------------------------------------------

# - Connection Settings -

listen_addresses = 'localhost, 192.168.100.11'          # what IP address(es) to listen on;
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost'; use '*' for all
                                        # (change requires restart)
port = 5432                             # (change requires restart)
```

Запускаем сервис Postgresql:

$ systemctl start postgresql

Проверка работоспособности зеркалирования между серверами VM_Ubuntu IP 192.168.100.14/24 - Master и VM2_Ubuntu IP 192.168.100.11/24 - Slave.

На Master-сервере создаем БД newdb_analysis. Из под пользователя postgres, запускаем psql:

$ psql	\\ заходим в psql
postgres=# create database newdb_analysis;	\\ создаем БД newdb_analysis
postgres=# \l	\\ выводим список БД в СУБД

Выведя список БД, видимо, что есть наша созданная БД newdb_analysis:

```
postgres=# \l
                                 List of databases
      Name      |  Owner   | Encoding |   Collate   |    Ctype    |   Access privileges
----------------+----------+----------+-------------+-------------+-----------------------
 newdb_analysis | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 postgres       | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 template0      | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
                |          |          |             |             | postgres=CTc/postgres
 template1      | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
                |          |          |             |             | postgres=CTc/postgres
(4 rows)

postgres=# \connect newdb_analysis;
You are now connected to database "newdb_analysis" as user "postgres".
newdb_analysis=#
```

Также заходим на вторичный сервер и выводим список созданных БД:

```
postgres@makarov2-VirtualBox:~$ psql
psql (14.15 (Ubuntu 14.15-0ubuntu0.22.04.1))
Type "help" for help.

postgres=# \l
                                    List of databases
      Name      |  Owner   | Encoding |   Collate   |    Ctype    |   Access privileges
----------------+----------+----------+-------------+-------------+-----------------------
 newdb_analysis | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 postgres       | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 template0      | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
                |          |          |             |             | postgres=CTc/postgres
 template1      | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
                |          |          |             |             | postgres=CTc/postgres
(4 rows)

postgres=#
```

БД появилась на вторичном сервере

Зеркалирование между двумя ВМ настроено успешно, все данные внесённые на первичном сервере, добавятся автоматически и на вторичном сервере.

Проверить репликацию, можно запросами:

На первичном сервере :

postgres=# select * from pg_stat_replication;

```
  pid  | usesysid | usename  | application_name |  client_addr   | client_hostname | client_port |         backend_star
t         | backend_xmin |   state   | sent_lsn  | write_lsn | flush_lsn | replay_lsn | write_lag | flush_lag | replay_
lag | sync_priority | sync_state |          reply_time
-------+----------+----------+------------------+----------------+-----------------+-------------+-----------------------
---------+-------------+-----------+-----------+-----------+-----------+------------+-----------+-----------+--------
----+---------------+------------+-------------------------------
 12175 |    16384 | user_rep | 14/main          | 192.168.100.11 |                 |       40856 | 2025-01-18 17:40:59.
368002+03 |              | streaming | 0/5000AF0 | 0/5000AF0 | 0/5000AF0 | 0/5000AF0  |           |           |
    |             0 | async      | 2025-01-19 13:39:50.184184+03
(1 row)
```

либо,

postgres=# select client_addr, state, sent_location, write_location, flush_location, replay_location from pg_stat_replication;

```
newdb_analysis=# select client_addr, state, sent_lsn, write_lsn, flush_lsn, replay_lsn from pg_stat_replication;;
  client_addr   |   state   | sent_lsn  | write_lsn | flush_lsn | replay_lsn
----------------+-----------+-----------+-----------+-----------+------------
 192.168.100.11 | streaming | 0/5000AF0 | 0/5000AF0 | 0/5000AF0 | 0/5000AF0
(1 row)

newdb_analysis=#
```

На вторичном сервере:

postgres=# select * from pg_stat_wal_receiver;

```
  pid  |  status   | receive_start_lsn | receive_start_tli | written_lsn | flushed_lsn | received_tli |      last_msg_send_time
  |     last_msg_receipt_time     | latest_end_lsn |        latest_end_time        | slot_name |  sender_host   | sender_port |
                                                                                                                                con
ninfo

-------+-----------+-------------------+-------------------+-------------+-------------+--------------+-------------------------
--+-------------------------------+----------------+-------------------------------+-----------+----------------+-------------+-----
-------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------------------------------
-------
 12724 | streaming | 0/5000000         |                 1 | 0/5000AF0   | 0/5000AF0   |            1 | 2025-01-19 13:34:29.920956+0
3 | 2025-01-19 13:34:29.894191+03 | 0/5000AF0      | 2025-01-18 17:57:01.448555+03 |           | 192.168.100.14 |        5432 | user
=user_rep password=******** channel_binding=prefer dbname=replication host=192.168.100.14 port=5432 fallback_application_name=14/mai
n sslmode=prefer sslcompression=0 sslsni=1 ssl_min_protocol_version=TLSv1.2 gssencmode=prefer krbsrvname=postgres target_session_att
rs=any
(1 row)
```

Кластеризация, будет делатся с использованием repmgr:
Будет дополнено....