LAPORAN PRAKTIKUM TEKNOLOGI BASIS DATA

MODUL 3

REPLIKASI DATA ASYNCHRONOUS DAN CONCURRENCY CONTROL

KELAS PRAKTIKUM BASIS DATA – TIK1092_B

KELOMPOK 4 :

1. HIKARU E. JONES 230211060107
2. VANCEL RENGKUNG 230211060105
3. RISKY A. IMBAT 230211060102
4. JONATHAN HENRY EMAN 230211060041
5. CHRISTIAN D.A KAREPOWAN 230 211060095
6. GRACIANO P. TILAAR 230211060031

ASISTEN: ALEXANDRIO MALETENG

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS SAM RATULANGI
MANADO

2024

LATIHAN

No. Latihan: Percobaan 1 soal no. 1

Soal Latihan:

Persiapan Docker Compose

Buatlah file docker-compose.yml , dengan isi file sebagai berikut:

```
  GNU nano 7.2                                                    docke
version: '3.8'
services:
  postgres_primary:
    container_name: prak_tbd_modul_3_primary
    image: postgres
    environment:
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: password
      POSTGRES_DB: prak_tbd_modul_3_db
    volumes:
      - db_data:/var/lib/postgresql/data
    ports:
      - "6543:6543"
    networks:
      - prak_tbd_network
    restart: unless-stopped

  postgres_standby:
    container_name: prak_tbd_modul_3_standby
    image: postgres
    environment:
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: password
      POSTGRES_DB: prak_tbd_modul_3_db
    volumes:
      - db_data_standby:/var/lib/postgresql/data
    ports:
      - "6544:6543"
    networks:
      - prak_tbd_network
    restart: unless-stopped

volumes:
  db_data:
  db_data_standby:

networks:
  prak_tbd_network:
```

setelah itu save(ctrl + o) setelah itu langsung enter dan keluar dari nano dengan(ctrl + x).

Statement SQL:	**Tujuan/Penjelasan Query:**

`version: '3.8'`
`services:`
`  postgres_primary:`
`  container_name: prak_tbd_modul_3_primary`
`  image: postgres`
`  environment:`
`  POSTGRES_USER: postgres`
	File docker-compose.yml ini mengatur dua layanan, yaitu postgres_primary sebagai server utama dan postgres_standby sebagai server cadangan, yang berjalan di dalam kontainer Docker. Kontainer utama menyimpan datanya di volume db_data, sementara kontainer cadangan menggunakan volume db_data_standby. Keduanya terhubung ke jaringan

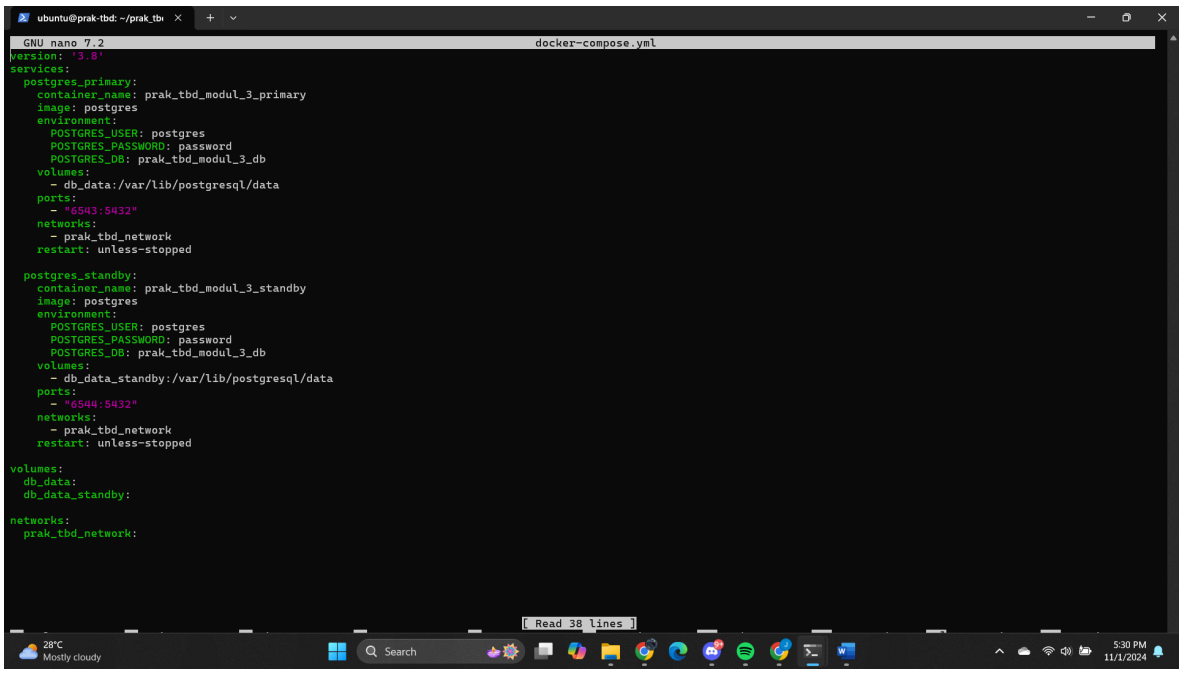
POSTGRES_PASSWORD: password POSTGRES_DB: prak_tbd_modul_3_db volumes: - db_data:/var/lib/postgresql/data ports: - "6543:5432" networks: - prak_tbd_network restart: unless-stopped postgres_standby: container_name: prak_tbd_modul_3_standby image: postgres environment: POSTGRES_USER: postgres POSTGRES_PASSWORD: password POSTGRES_DB: prak_tbd_modul_3_db volumes: - db_data_standby:/var/lib/postgresql/data ports: - "6544:5432" networks: - prak_tbd_network	prak_tbd_network dan diatur agar bisa restart otomatis kalau tiba-tiba berhenti. Dengan port yang berbeda, kedua database ini dapat diakses dari host, sehingga sistem bisa melakukan backup jika server primary mengalami masalah.

```
                                restart:
      unless-stopped


      volumes:
        db_data:
        db_data_standby:


      networks:
        prak_tbd_network:
```

Hasil Query/SQL:



Analisa Error: -

Penerapan Solusi Akhir: -

Penjelasan Statement: -

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 1 soal 2

Soal Latihan:

Menjalankan Docker Compose

Selanjutnya jalankan docker-compose [sudo docker compose up -d] ,lalu akan muncul seperti ini gambar di bawah ini.

docker-compose up -d

```
ubuntu@prak-tbd ~/prak_tbd
sudo docker compose up -d
```

```
sudo docker compose up -d
WARN[0000] /home/ubuntu/prak_tbd/docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion
[+] Running 5/5
 ✔ Network prak_tbd_prak_tbd_network   Created                                                    0.1s
 ✔ Volume "prak_tbd_db_data"           Created                                                    0.0s
 ✔ Volume "prak_tbd_db_data_standby"   Created                                                    0.0s
 ✔ Container prak_tbd_modul_3_primary  Started                                                    0.3s
 ✔ Container prak_tbd_modul_3_standby  Started                                                    0.3s
```

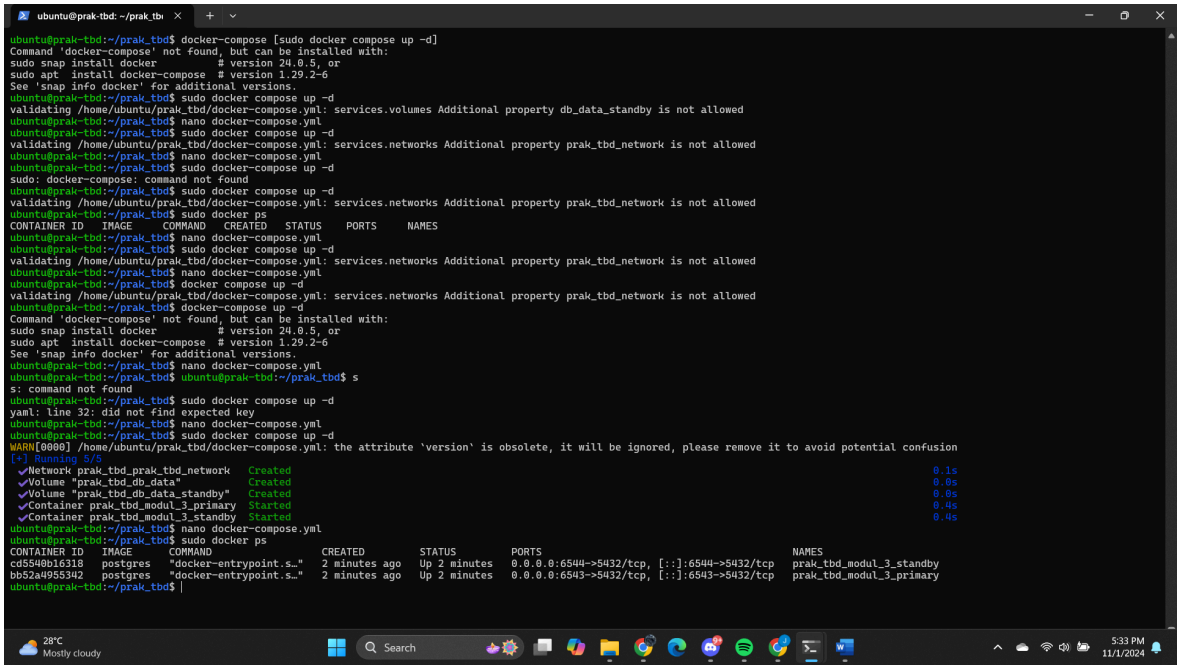
ini akan memulai dua container PostgreSQL, satu sebagai primary postgres_primary dan satu sebagai standby postgres_standby. Jika ingin melihat container docker sedang berjalan atau tidak, jalankan perintah di bawah ini.

```
ubuntu@prak-tbd ~/prak_tbd (0.072s)
sudo docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED             STATUS              PORTS                                                   NAMES
1b4b3c30d1b9   postgres   "docker-entrypoint.s…"   About a minute ago  Up About a minute   5432/tcp, 0.0.0.0:6544->6543/tcp, [::]:6544->6543/tcp   prak_tbd_modul_3_
standby
3eec710f98df   postgres   "docker-entrypoint.s…"   About a minute ago  Up About a minute   5432/tcp, 0.0.0.0:6543->6543/tcp, :::6543->6543/tcp     prak_tbd_modul_3_
primary
```

Statement SQL:	**Tujuan/Penjelasan Query:**
1. `sudo docker compose up -d` 2. `sudo docker ps`	1. sudo docker compose up -d digunakan untuk menjalankan semua layanan yang ada di dalam file docker-compose.yml secara otomatis. -d adalah singkatan dari detached mode yang berarti bahwa container akan dijalankan di latar belakang (background). 2. sudo docker ps untuk melihat daftar container yang sedang berjalan pada sistem. Menampilkan ID container, nama image, waktu

	pembuatan, status, port yang digunakan, dan nama container.

Hasil Query/SQL:



Analisa Error: -

Penerapan Solusi Akhir: -	**Penjelasan Statement: -**

Hasil Penerapan Solusi Akhir: -

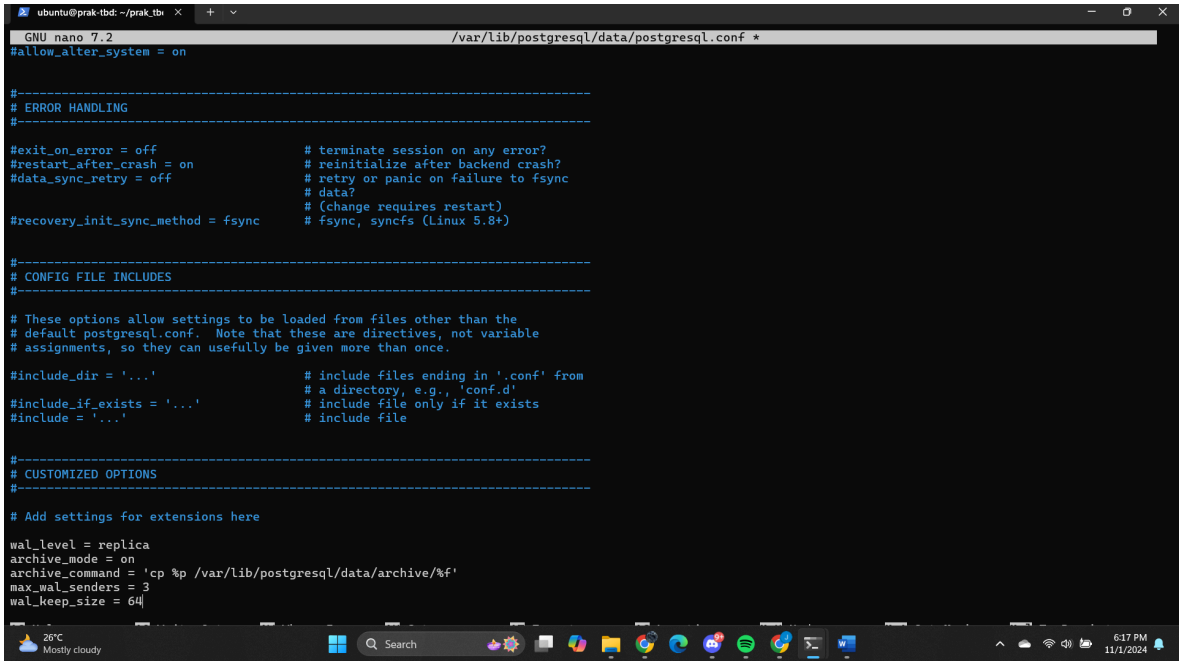
<table>
<tr><td colspan="2">No. Latihan: Percobaan 1 soal 3</td></tr>
<tr><td colspan="2">Soal Latihan:
Konfigurasi postgresql.conf di primary
Setelah container compose berjalan, jalankan perintah di bawah untuk masuk ke dalam container postgresql primary.
docker exec -it prak_tbd_modul_3_primary bash

Selanjutnya gunakan nano untuk mengedit postgresql.conf di primary database
nano /var/lib/postgresql/data/postgresql.conf

Lalu tambahkan atau sesuaikan pengaturan berikut:
wal_level = replica
archive_mode = on
archive_command = 'cp %p /var/lib/postgresql/data/archive/%f '
max_wal_senders = 3
wal_keep_size = 64
Pengaturan ini memungkinkan postgres primary untuk menyimpan cukup WAL (Write-Ahead Logging).</td></tr>
<tr><td>Statement SQL:
<ol><li><code>sudo docker exec -it prak_tbd_modul_3_primary bash</code></li><li><code>nano /var/lib/postgresql/data/postgresql.conf</code></li></ol></td><td>Tujuan/Penjelasan Query:
<ol><li>sudo docker exec -it prak_tbd_modul_3_primary bash digunakan untuk masuk ke dalam container bernama prak_tbd_modul_3_primary. Fungsi -it agar pengguna dapat masuk dan berinteraksi langsung dengan container secara interaktif di terminal, seperti menjalankan shell bash, dll..</li><li>nano /var/lib/postgresql/data/postgresql.conf untuk membuka file konfigurasi</li></ol></td></tr>
</table>

	postgresql.conf menggunakan editor teks nano.

Hasil Query/SQL:



Analisa Error: -

Penerapan Solusi Akhir: -	**Penjelasan Statement: -**

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 1 soal 4

Soal Latihan:

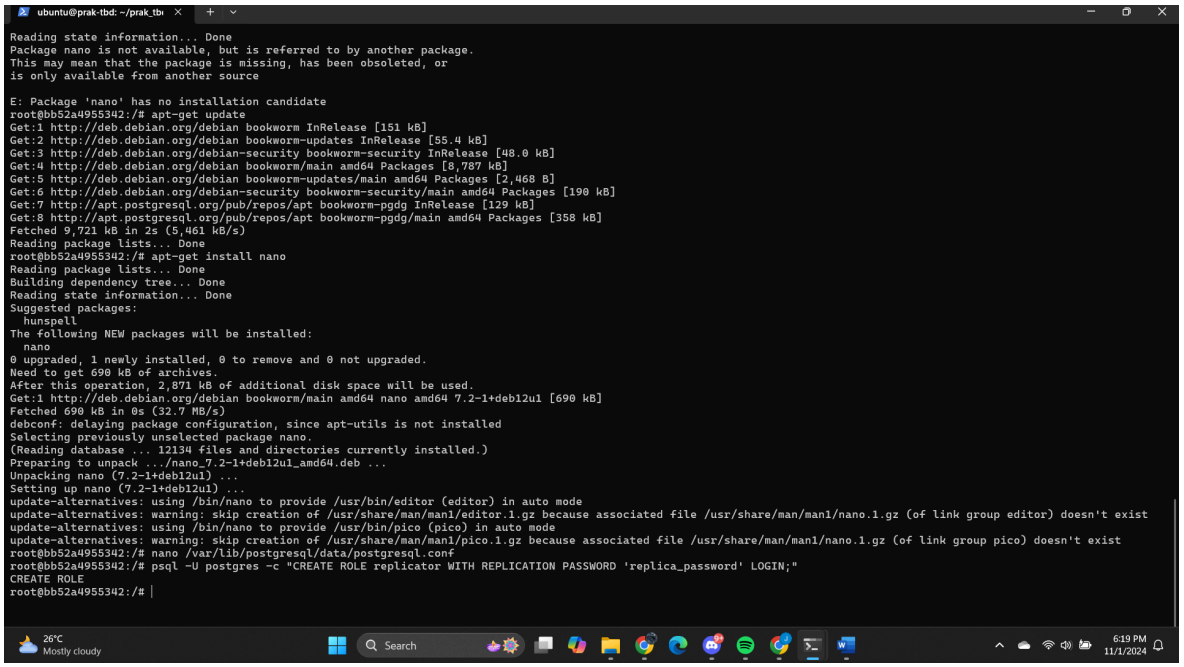
Menambah Replication User di Primary

Buatlah user khusus untuk replikasi di PostgreSQL seperti gambar dibawa ini

psql -U postgres -c "CREATE ROLE replicator WITH REPLICATION PASSWORD 'replica_password' LOGIN;"

Statement SQL:	**Tujuan/Penjelasan Query:**
`psql -U postgres -c "CREATE ROLE replicator WITH REPLICATION PASSWORD 'replica_password' LOGIN;"`	Untuk membuat role bernama replicator, dengan hak untuk melakukan replikasi data. Role ini diberi password replica_password untuk login ke dalam Postgresql.

Hasil Query/SQL:



```
ubuntu@prak-tbd: ~/prak_tbd
Reading state information... Done
Package nano is not available, but is referred to by another package.
This may mean that the package is missing, has been obsoleted, or
is only available from another source

E: Package 'nano' has no installation candidate
root@bb52a4955342:/# apt-get update
Get:1 http://deb.debian.org/debian bookworm InRelease [151 kB]
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:3 http://deb.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Get:4 http://deb.debian.org/debian bookworm/main amd64 Packages [8,787 kB]
Get:5 http://deb.debian.org/debian bookworm-updates/main amd64 Packages [2,468 B]
Get:6 http://deb.debian.org/debian-security bookworm-security/main amd64 Packages [190 kB]
Get:7 http://apt.postgresql.org/pub/repos/apt bookworm-pgdg InRelease [129 kB]
Get:8 http://apt.postgresql.org/pub/repos/apt bookworm-pgdg/main amd64 Packages [358 kB]
Fetched 9,721 kB in 2s (5,461 kB/s)
Reading package lists... Done
root@bb52a4955342:/# apt-get install nano
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  hunspell
The following NEW packages will be installed:
  nano
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 690 kB of archives.
After this operation, 2,871 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian bookworm/main amd64 nano amd64 7.2-1+deb12u1 [690 kB]
Fetched 690 kB in 0s (32.7 MB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package nano.
(Reading database ... 12134 files and directories currently installed.)
Preparing to unpack .../nano_7.2-1+deb12u1_amd64.deb ...
Unpacking nano (7.2-1+deb12u1) ...
Setting up nano (7.2-1+deb12u1) ...
update-alternatives: using /bin/nano to provide /usr/bin/editor (editor) in auto mode
update-alternatives: warning: skip creation of /usr/share/man/man1/editor.1.gz because associated file /usr/share/man/man1/nano.1.gz (of link group editor) doesn't exist
update-alternatives: using /bin/nano to provide /usr/bin/pico (pico) in auto mode
update-alternatives: warning: skip creation of /usr/share/man/man1/pico.1.gz because associated file /usr/share/man/man1/nano.1.gz (of link group pico) doesn't exist
root@bb52a4955342:/# nano /var/lib/postgresql/data/postgresql.conf
root@bb52a4955342:/# psql -U postgres -c "CREATE ROLE replicator WITH REPLICATION PASSWORD 'replica_password' LOGIN;"
CREATE ROLE
root@bb52a4955342:/#
```

Analisa Error: -

Penerapan Solusi Akhir: -	**Penjelasan Statement: -**

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 1 soal 5

Soal Latihan:

Konfigurasi pg_hba.conf di Primary

Gunakan nano untuk mengizinkan replikasi dari container standby.

lalu tambahkan host replication berikut untuk mengizinkan replikasi dari standby

host replication replicator 0.0.0.0/0 md5

```
local   replication     all                                     trust
host    replication     all             127.0.0.1/32            trust
host    replication     all             ::1/128                 trust
host    replication     replicator      0.0.0.0/0               md5
```

Statement SQL:

```
nano /var/lib/postgresql/data/pg_hba.conf
```

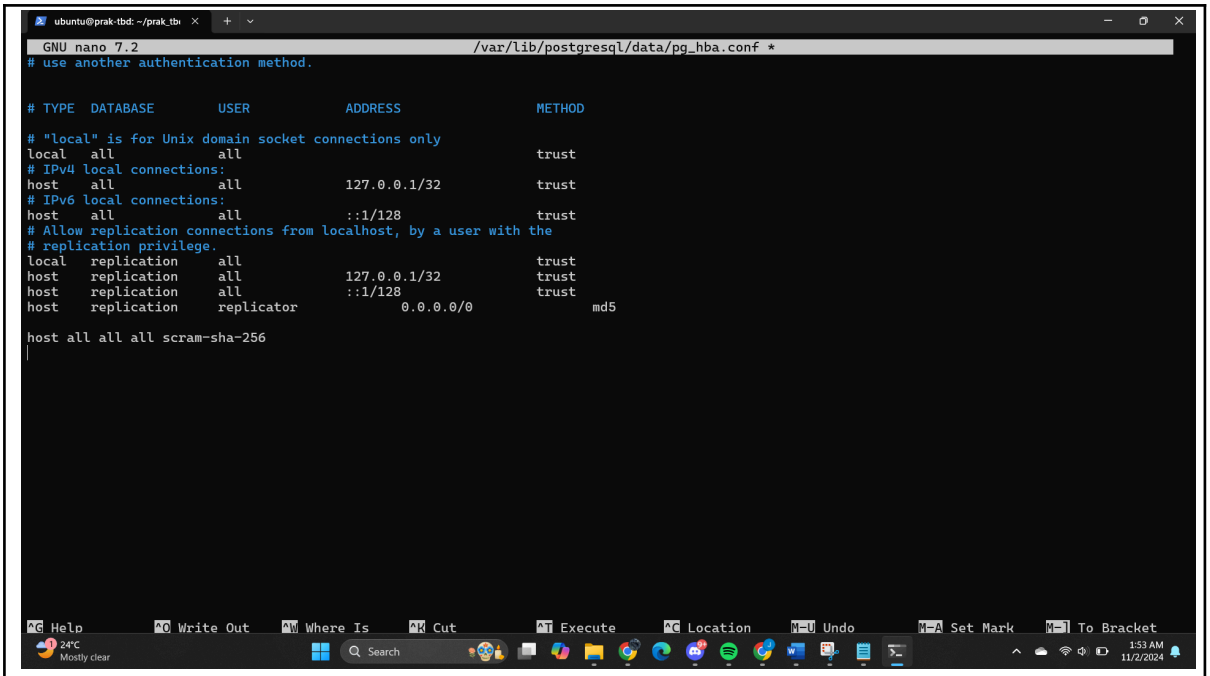
Tujuan/Penjelasan Query:

nano /var/lib/postgresql/data/pg_hba.conf untuk membuka file konfigurasi pg_hba.conf menggunakan editor teks nano.

Hasil Query/SQL:

```
ubuntu@prak-tbd: ~/prak_tbd
Reading state information... Done
Package nano is not available, but is referred to by another package.
This may mean that the package is missing, has been obsoleted, or
is only available from another source

E: Package 'nano' has no installation candidate
root@bb52a4955342:/# apt-get update
Get:1 http://deb.debian.org/debian bookworm InRelease [151 kB]
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:3 http://deb.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Get:4 http://deb.debian.org/debian bookworm/main amd64 Packages [8,787 kB]
Get:5 http://deb.debian.org/debian bookworm-updates/main amd64 Packages [2,468 B]
Get:6 http://deb.debian.org/debian-security bookworm-security/main amd64 Packages [190 kB]
Get:7 http://apt.postgresql.org/pub/repos/apt bookworm-pgdg InRelease [129 kB]
Get:8 http://apt.postgresql.org/pub/repos/apt bookworm-pgdg/main amd64 Packages [358 kB]
Fetched 9,721 kB in 2s (5,461 kB/s)
Reading package lists... Done
root@bb52a4955342:/# apt-get install nano
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  hunspell
The following NEW packages will be installed:
  nano
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 690 kB of archives.
After this operation, 2,871 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian bookworm/main amd64 nano amd64 7.2-1+deb12u1 [690 kB]
Fetched 690 kB in 0s (32.7 MB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package nano.
(Reading database ... 12134 files and directories currently installed.)
Preparing to unpack .../nano_7.2-1+deb12u1_amd64.deb ...
Unpacking nano (7.2-1+deb12u1) ...
Setting up nano (7.2-1+deb12u1) ...
update-alternatives: using /bin/nano to provide /usr/bin/editor (editor) in auto mode
update-alternatives: warning: skip creation of /usr/share/man/man1/editor.1.gz because associated file /usr/share/man/man1/nano.1.gz (of link group editor) doesn't exist
update-alternatives: using /bin/nano to provide /usr/bin/pico (pico) in auto mode
update-alternatives: warning: skip creation of /usr/share/man/man1/pico.1.gz because associated file /usr/share/man/man1/nano.1.gz (of link group pico) doesn't exist
root@bb52a4955342:/# nano /var/lib/postgresql/data/postgresql.conf
root@bb52a4955342:/# psql -U postgres -c "CREATE ROLE replicator WITH REPLICATION PASSWORD 'replica_password' LOGIN;"
CREATE ROLE
root@bb52a4955342:/# nano /var/lib/postgresql/data/pg_hba.conf
root@bb52a4955342:/#
```



Analisa Error: -

Penerapan Solusi Akhir: -	**Penjelasan Statement: -**

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 1 soal 6

Soal Latihan:

Restart Primary

Setelah melakukan Perubahan konfigurasi, restart container prak_tbd_modul_3_primary agar perubahan diterapkan.

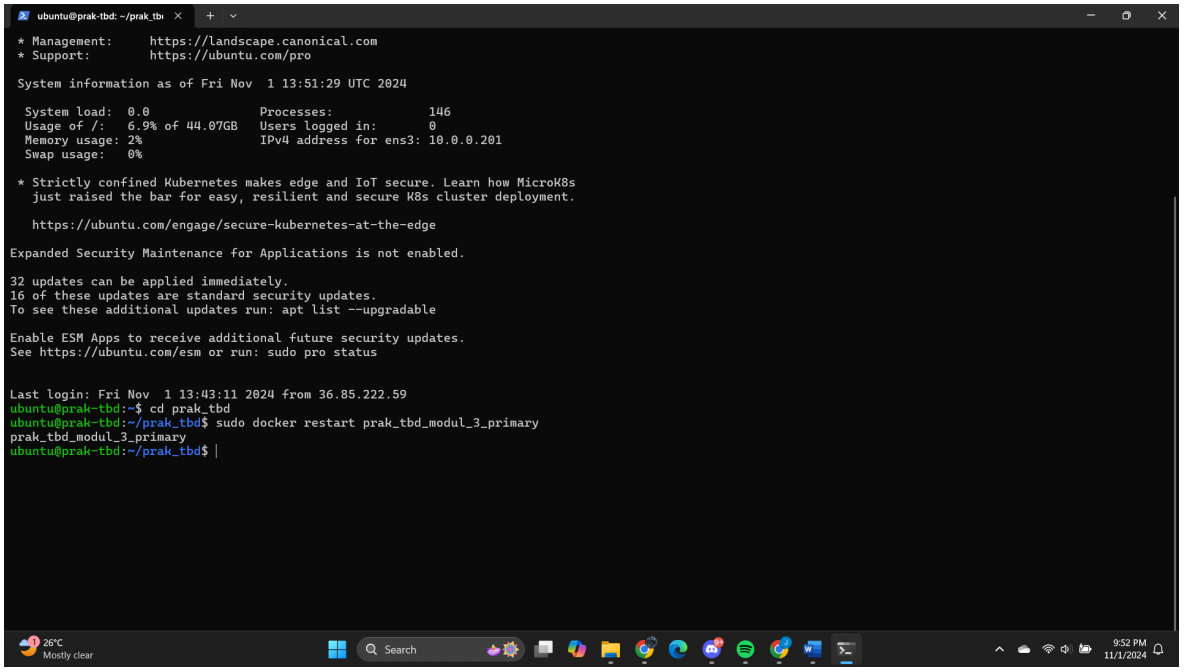
Statement SQL:

```
sudo docker restart prak_tbd_modul_3_primary
```

Tujuan/Penjelasan Query:

Untuk melakukan restart pada container prak_tbd_modul_3_primary agar perubahan pada file postgresql.conf dan pg_hba.conf dapat tersimpan dan diterapkan.

Hasil Query/SQL:



Analisa Error: -

Penerapan Solusi Akhir: -

Penjelasan Statement: -

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 1 soal 7

Soal Latihan:

Konfigurasi Standby

Setelah itu masuk ke container prak_tbd_modul_3_standby.

Hapus data directori PostgresSQL agar bisa melakukan basebackup dari primary.

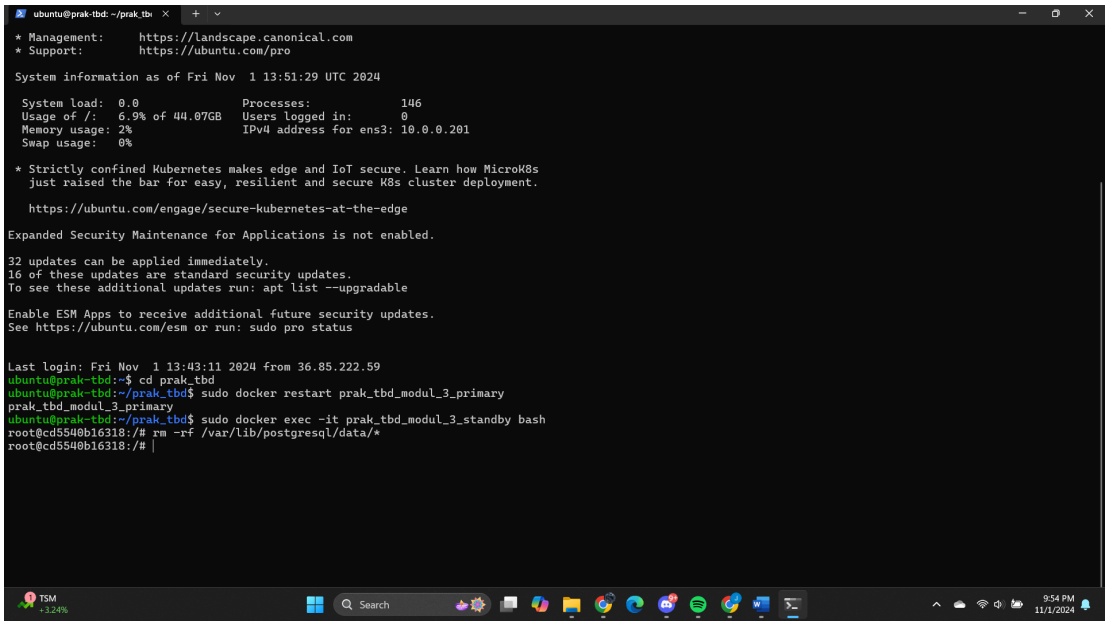
Statement SQL:

1. `sudo docker exec -it prak_tbd_modul_3_standby bash`
2. `rm -rf /var/lib/postgresql/data/*`

Tujuan/Penjelasan Query:

1. Untuk masuk ke dalam container prak_tbd_modul_3_standby dan membuka shell bash.
2. Untuk menghapus semua file dan folder yang ada di dalam direktori /var/lib/postgresql/data/ yang tersimpan dalam container prak_tbd_modul_3_standby. rm -rf digunakan untuk menghapus direktori dan semua isinya tanpa konfirmasi tambahan.

Hasil Query/SQL:



Analisa Error: -

Penerapan Solusi Akhir: -

Penjelasan Statement: -

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 1 soal 8

Soal Latihan:

Membuat Base Backup dari Primary ke Standby.

Lakukan pg_basebackup dari primary standby.

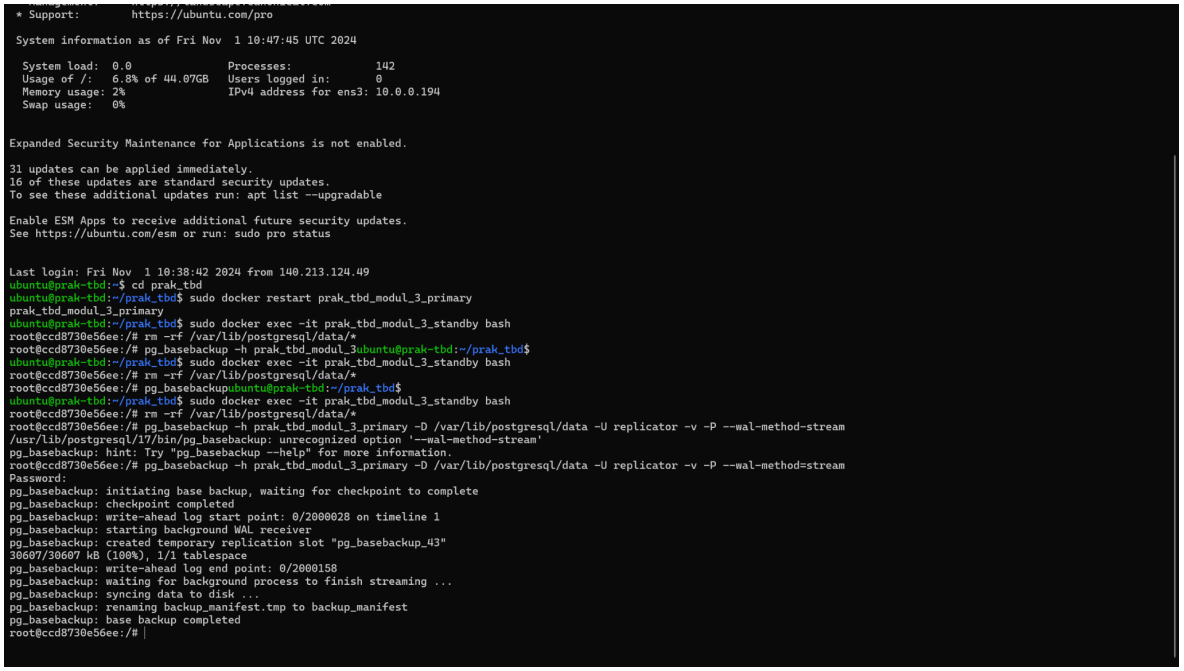
Statement SQL:

```
pg_basebackup -h prak_tbd_modul_3_primary -D /var/lib/postgresql/data -U replicator -v -P --wal-method=stream
```

Tujuan/Penjelasan Query:

Digunakan untuk melakukan backup database dari server primary ke server standby. Backup ini dilakukan secara langsung ke direktori /var/lib/postgresql/data menggunakan user replicator. Fungsi -v menampilkan proses backup secara rinci, -P menampilkan progres backup, dan --wal-method=stream memastikan bahwa log WAL (Write-Ahead Logging) untuk memastikan sinkronisasi antara server primary dan server standby.

Hasil Query/SQL:



```
 * Support:        https://ubuntu.com/pro

 System information as of Fri Nov  1 10:47:45 UTC 2024

  System load:  0.0               Processes:             142
  Usage of /:   6.8% of 44.07GB   Users logged in:       0
  Memory usage: 2%                IPv4 address for ens3: 10.0.0.194
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

31 updates can be applied immediately.
16 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


Last login: Fri Nov  1 10:38:42 2024 from 140.213.124.49
ubuntu@prak-tbd:~$ cd prak_tbd
ubuntu@prak-tbd:~/prak_tbd$ sudo docker restart prak_tbd_modul_3_primary
prak_tbd_modul_3_primary
ubuntu@prak-tbd:~/prak_tbd$ sudo docker exec -it prak_tbd_modul_3_standby bash
root@ccd8730e56ee:/# rm -rf /var/lib/postgresql/data/*
root@ccd8730e56ee:/# pg_basebackup -h prak_tbd_modul_3ubuntu@prak-tbd:~/prak_tbd$
ubuntu@prak-tbd:~/prak_tbd$ sudo docker exec -it prak_tbd_modul_3_standby bash
root@ccd8730e56ee:/# rm -rf /var/lib/postgresql/data/*
root@ccd8730e56ee:/# pg_basebackupubuntu@prak-tbd:~/prak_tbd$
ubuntu@prak-tbd:~/prak_tbd$ sudo docker exec -it prak_tbd_modul_3_standby bash
root@ccd8730e56ee:/# rm -rf /var/lib/postgresql/data/*
root@ccd8730e56ee:/# pg_basebackup -h prak_tbd_modul_3_primary -D /var/lib/postgresql/data -U replicator -v -P --wal-method=stream
/usr/lib/postgresql/17/bin/pg_basebackup: unrecognized option '--wal-method=stream'
pg_basebackup: hint: Try "pg_basebackup --help" for more information.
root@ccd8730e56ee:/# pg_basebackup -h prak_tbd_modul_3_primary -D /var/lib/postgresql/data -U replicator -v -P --wal-method=stream
Password:
pg_basebackup: initiating base backup, waiting for checkpoint to complete
pg_basebackup: checkpoint completed
pg_basebackup: write-ahead log start point: 0/2000028 on timeline 1
pg_basebackup: starting background WAL receiver
pg_basebackup: created temporary replication slot "pg_basebackup_43"
30607/30607 kB (100%), 1/1 tablespace
pg_basebackup: write-ahead log end point: 0/2000158
pg_basebackup: waiting for background process to finish streaming ...
pg_basebackup: syncing data to disk ...
pg_basebackup: renaming backup_manifest.tmp to backup_manifest
pg_basebackup: base backup completed
root@ccd8730e56ee:/#
```

Analisa Error: -

Penerapan Solusi Akhir: -

Penjelasan Statement: -

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 1 soal 9

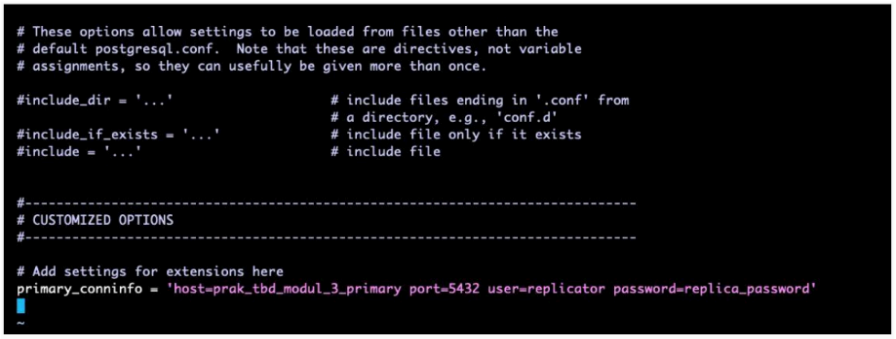
Soal Latihan:

Membuat recovery.conf di Standby

Setelah pg_basebackup selesai, buatlah file standby.signal untuk mengkonfigurasi container standby sebagai server standby sekarang gunakan nano untuk mengedit file postgresql.conf.

Lalu tambahkan atau sesuaikan konfigurasi dibawah ini

primary_conninfo = 'host=prak_tbd_modul_3_primary port=5432 user=replicator password=replica_password'



Statement SQL:	**Tujuan/Penjelasan Query:**
1. `touch /var/lib/postgresql/data/standby.signal` 2. `nano /var/lib/postgresql/data/postgresql.conf`	1. Untuk membuat file bernama standby.signal di dalam direktori /var/lib/postgresql/data/ yang berfungsi sebagai indikator server standby dalam replikasi. 2. untuk membuka file konfigurasi postgresql.conf menggunakan editor teks nano.

Hasil Query/SQL:

```
ubuntu@prak-tbd: ~/prak_tbd
ubuntu@prak-tbd:~/prak_tbd$
ubuntu@prak-tbd:~/prak_tbd$ sudo docker exec -it prak_tbd_modul_3_standby bash
root@08632f39d0dd:/# toouch /var/lib/postgresql/data/standby.signal
bash: toouch: command not found
root@08632f39d0dd:/# touch /var/lib/postgresql/data/standby.signal
root@08632f39d0dd:/# nano /var/lib/postgresql/data/standby.signal
bash: nano: command not found
root@08632f39d0dd:/# apt-get update
Get:1 http://deb.debian.org/debian bookworm InRelease [151 kB]
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:3 http://deb.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Get:4 http://deb.debian.org/debian bookworm/main amd64 Packages [8,787 kB]
Get:5 http://deb.debian.org/debian bookworm-updates/main amd64 Packages [2,468 B]
Get:6 http://deb.debian.org/debian-security bookworm-security/main amd64 Packages [190 kB]
Get:7 http://apt.postgresql.org/pub/repos/apt bookworm-pgdg InRelease [129 kB]
Get:8 http://apt.postgresql.org/pub/repos/apt bookworm-pgdg/main amd64 Packages [358 kB]
Fetched 9,721 kB in 2s (5,282 kB/s)
Reading package lists... Done
root@08632f39d0dd:/# apt-get install nano
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  hunspell
The following NEW packages will be installed:
  nano
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 690 kB of archives.
After this operation, 2,871 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian bookworm/main amd64 nano amd64 7.2-1+deb12u1 [690 kB]
Fetched 690 kB in 0s (32.6 MB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package nano.
(Reading database ... 12134 files and directories currently installed.)
Preparing to unpack .../nano_7.2-1+deb12u1_amd64.deb ...
Unpacking nano (7.2-1+deb12u1) ...
Setting up nano (7.2-1+deb12u1) ...
update-alternatives: using /bin/nano to provide /usr/bin/editor (editor) in auto mode
update-alternatives: warning: skip creation of /usr/share/man/man1/editor.1.gz because associated file /usr/share/man/man1/nano.1.gz (of link group editor) doesn't exist
update-alternatives: using /bin/nano to provide /usr/bin/pico (pico) in auto mode
update-alternatives: warning: skip creation of /usr/share/man/man1/pico.1.gz because associated file /usr/share/man/man1/nano.1.gz (of link group pico) doesn't exist
root@08632f39d0dd:/# nano /var/lib/postgresql/data/standby.signal
root@08632f39d0dd:/# nano /var/lib/postgresql/data/postgresql.conf
root@08632f39d0dd:/# root@08632f39d0dd:/#
root@08632f39d0dd:/# nano /var/lib/postgresql/data/postgresql.conf
root@08632f39d0dd:/#
```

```
ubuntu@prak-tbd: ~/prak_tbd
  GNU nano 7.2                          /var/lib/postgresql/data/postgresql.conf

#array_nulls = on
#backslash_quote = safe_encoding        # on, off, or safe_encoding
#escape_string_warning = on
#lo_compat_privileges = off
#quote_all_identifiers = off
#standard_conforming_strings = on
#synchronize_seqscans = on

# - Other Platforms and Clients -

#transform_null_equals = off
#allow_alter_system = on


#------------------------------------------------------------------------------
# ERROR HANDLING
#------------------------------------------------------------------------------

#exit_on_error = off                    # terminate session on any error?
#restart_after_crash = on               # reinitialize after backend crash?
#data_sync_retry = off                  # retry or panic on failure to fsync
                                        # data?
                                        # (change requires restart)
#recovery_init_sync_method = fsync      # fsync, syncfs (Linux 5.8+)


#------------------------------------------------------------------------------
# CONFIG FILE INCLUDES
#------------------------------------------------------------------------------

# These options allow settings to be loaded from files other than the
# default postgresql.conf.  Note that these are directives, not variable
# assignments, so they can usefully be given more than once.

#include_dir = '...'                    # include files ending in '.conf' from
                                        # a directory, e.g., 'conf.d'
#include_if_exists = '...'              # include file only if it exists
#include = '...'                        # include file


#------------------------------------------------------------------------------
# CUSTOMIZED OPTIONS
#------------------------------------------------------------------------------

# Add settings for extensions here
primary_conninfo = 'host=prak_tbd_modul_3_primary port=5432 user=replicator password=replicapas'

                                                [ Wrote 845 lines ]
^G Help      ^O Write Out   ^W Where Is    ^K Cut       ^T Execute   ^C Location   M-U Undo      M-A Set Mark  M-] To Bracket  M-Q Previous
^X Exit      ^R Read File   ^\ Replace     ^U Paste     ^J Justify   ^/ Go To Line M-E Redo      M-6 Copy      ^Q Where Was    M-W Next
```

Analisa Error: -

Penerapan Solusi Akhir: -

Penjelasan Statement: -

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 1 soal 10	
Soal Latihan: Restart Standby lalu keluar dari container dan restart postgres_standby.	
Statement SQL: `sudo docker restart prak_tbd_modul_3_standby`	**Tujuan/Penjelasan Query:** Untuk melakukan restart pada container prak_tbd_modul_3_standby agar perubahan pada file postgresql.conf dan backup dapat tersimpan dan diterapkan.

Hasil Query/SQL:
`ubuntu@prak-tbd:~/prak_tbd$ sudo docker restart prak_tbd_modul_3_standby`
`prak_tbd_modul_3_standby`	

Analisa Error: -	
Penerapan Solusi Akhir: -	**Penjelasan Statement: -**
Hasil Penerapan Solusi Akhir: -	

No. Latihan: Percobaan 1 soal 11

Soal Latihan:

Verifikasi Replikasi

sekarang untuk memastikan replikasi berjalan dengan baik,masuk ke dalam postgresql primary lalu jalankan perintah di bawah untuk melihat status replikasi.

psql -U postgres -c "SELECT * FROM pg_stat_replication;"

Perintah ini akan menampilkan informasi tentang replikasi yang berjalan di server primary. Anda seharusnya melihat informasi mengenai standby yang terhubung.

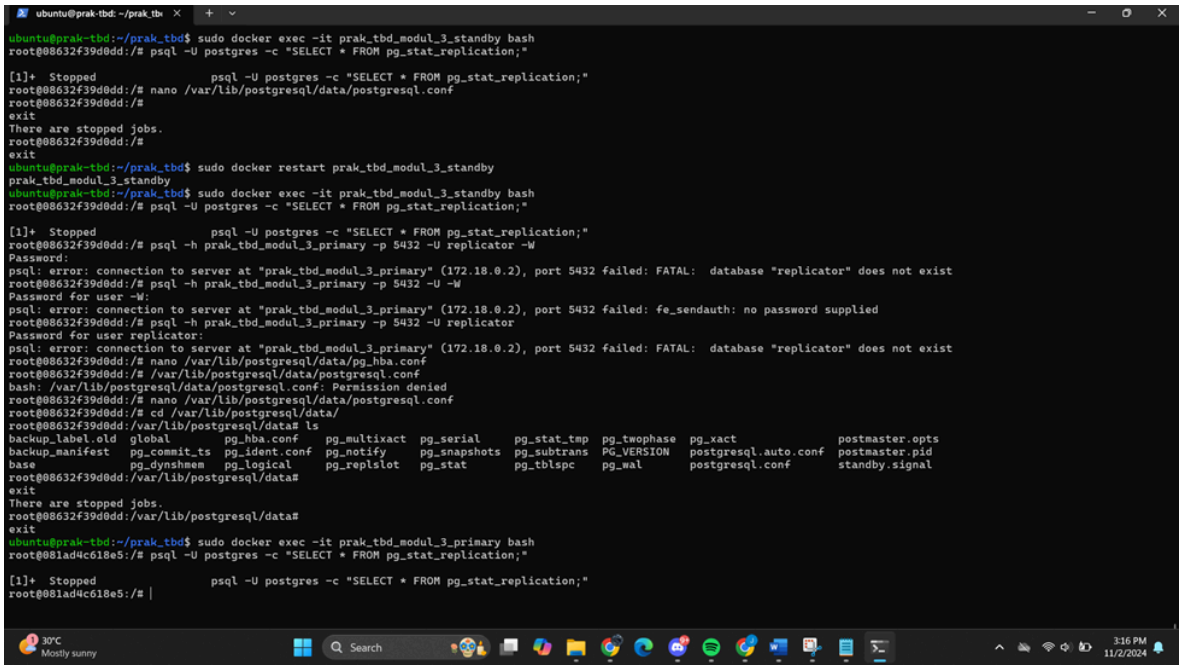
Statement SQL:

1. `sudo docker exec -it prak_tbd_modul_3_primary bash`
2. `psql -U postgres -c "SELECT * FROM pg_stat_replication;"`

Tujuan/Penjelasan Query:

1. Untuk masuk ke dalam container prak_tbd_modul_3_primary dan membuka shell bash.
2. Untuk menampilkan informasi tentang replikasi yang berjalan di server primary

Hasil Query/SQL:

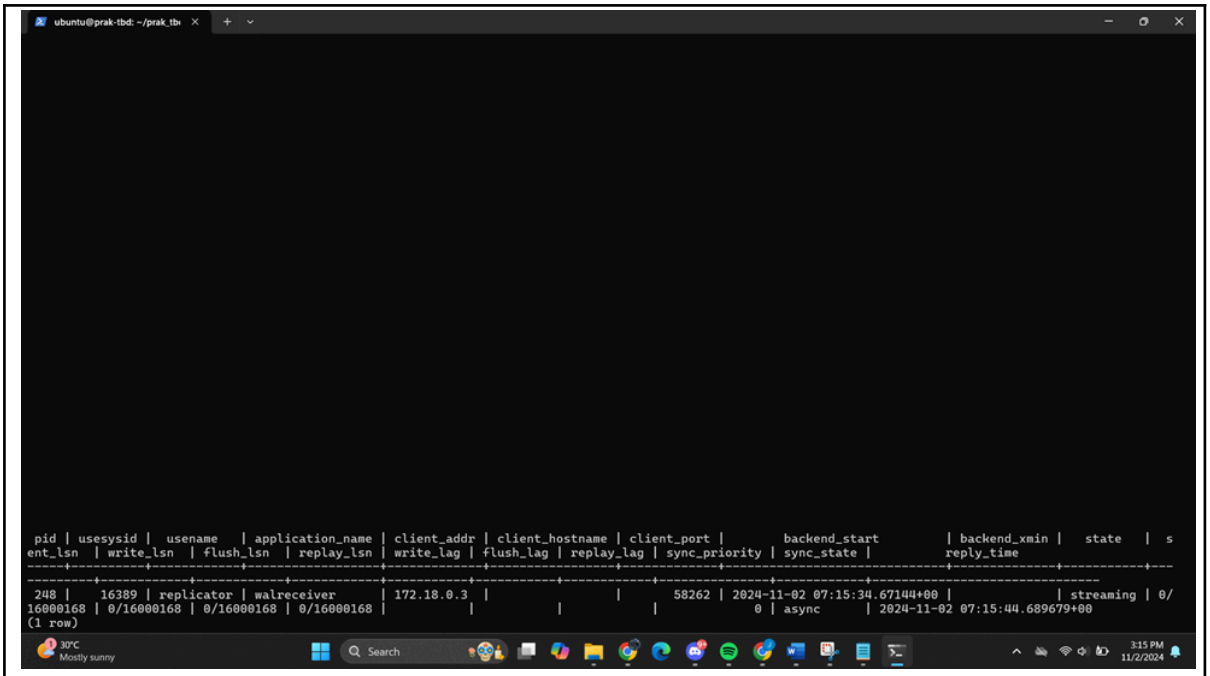


```
ubuntu@prak-tbd: ~/prak_tbd
ubuntu@prak-tbd:~/prak_tbd$ sudo docker exec -it prak_tbd_modul_3_standby bash
root@08632f39d0dd:/# psql -U postgres -c "SELECT * FROM pg_stat_replication;"

[1]+  Stopped                 psql -U postgres -c "SELECT * FROM pg_stat_replication;"
root@08632f39d0dd:/# nano /var/lib/postgresql/data/postgresql.conf
root@08632f39d0dd:/#
exit
There are stopped jobs.
root@08632f39d0dd:/#
exit
ubuntu@prak-tbd:~/prak_tbd$ sudo docker restart prak_tbd_modul_3_standby
prak_tbd_modul_3_standby
ubuntu@prak-tbd:~/prak_tbd$ sudo docker exec -it prak_tbd_modul_3_standby bash
root@08632f39d0dd:/# psql -U postgres -c "SELECT * FROM pg_stat_replication;"

[1]+  Stopped                 psql -U postgres -c "SELECT * FROM pg_stat_replication;"
root@08632f39d0dd:/# psql -h prak_tbd_modul_3_primary -p 5432 -U replicator -W
Password:
psql: error: connection to server at "prak_tbd_modul_3_primary" (172.18.0.2), port 5432 failed: FATAL:  database "replicator" does not exist
root@08632f39d0dd:/# psql -h prak_tbd_modul_3_primary -p 5432 -U -W
Password for user -W:
psql: error: connection to server at "prak_tbd_modul_3_primary" (172.18.0.2), port 5432 failed: fe_sendauth: no password supplied
root@08632f39d0dd:/# psql -h prak_tbd_modul_3_primary -p 5432 -U replicator
Password for user replicator:
psql: error: connection to server at "prak_tbd_modul_3_primary" (172.18.0.2), port 5432 failed: FATAL:  database "replicator" does not exist
root@08632f39d0dd:/# nano /var/lib/postgresql/data/pg_hba.conf
root@08632f39d0dd:/# /var/lib/postgresql/data/postgresql.conf
bash: /var/lib/postgresql/data/postgresql.conf: Permission denied
root@08632f39d0dd:/# nano /var/lib/postgresql/data/postgresql.conf
root@08632f39d0dd:/# cd /var/lib/postgresql/data/
root@08632f39d0dd:/var/lib/postgresql/data# ls
backup_label.old  global        pg_hba.conf    pg_multixact  pg_serial     pg_stat_tmp  pg_twophase  pg_xact              postmaster.opts
backup_manifest   pg_commit_ts  pg_ident.conf  pg_notify     pg_snapshots  pg_subtrans  PG_VERSION   postgresql.auto.conf  postmaster.pid
base              pg_dynshmem   pg_logical     pg_replslot   pg_stat       pg_tblspc    pg_wal       postgresql.conf       standby.signal
root@08632f39d0dd:/var/lib/postgresql/data#
exit
There are stopped jobs.
root@08632f39d0dd:/var/lib/postgresql/data#
exit
ubuntu@prak-tbd:~/prak_tbd$ sudo docker exec -it prak_tbd_modul_3_primary bash
root@081ad4c618e5:/# psql -U postgres -c "SELECT * FROM pg_stat_replication;"

[1]+  Stopped                 psql -U postgres -c "SELECT * FROM pg_stat_replication;"
root@081ad4c618e5:/#
```



Analisa Error: -

Penerapan Solusi Akhir: -	**Penjelasan Statement: -**

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 1 soal 12

Soal Latihan:

Pengujian Replikasi

Pengujian replikasi dapat dilakukan dengan membuat tabel atau memasukan data pada postgres_primary dan kemudian memeriksa apakah data tersebut muncul di postgres_standby.

Misalnya, buat tabel primary di terminal.

Kemudian, cek di postgres_standby apakah table test_replication sudah ada.

jika tabel muncul di standby, berarti replikasi berjalan dengan baik.

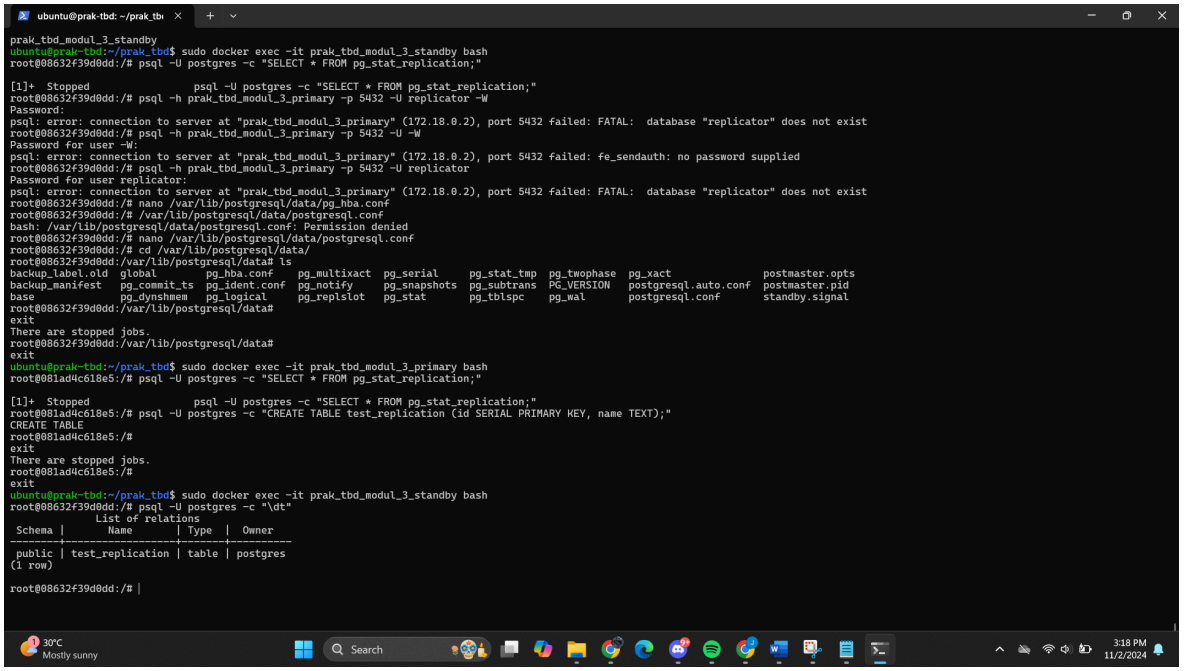
Statement SQL:

1. psql -U postgres -c "CREATE TABLE test_replication (id SERIAL PRIMARY KEY, name TEXT);"
2. psql -U postgres -c "\dt"

Tujuan/Penjelasan Query:

1. Membuat tabel dengan nama test_replication di primary yang bertujuan untuk memeriksa apakah tabel juga akan muncul di standby.
2. Untuk menampilkan tabel yang ada dalam database.

Hasil Query/SQL:



Analisa Error: -

Penerapan Solusi Akhir: -

Penjelasan Statement: -

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 2 soal 1	
Soal Latihan: Menyiapkan Lingkungan Pertama akses Server cloud Oracle menggunakan SSH(Secure Shell). ssh ubuntu@<public_ip_address> Pastikan docker-compose.yml pada percobaan 1 sudah berjalan, dan kedua container(primary dan standby) sudah aktif Masuk ke container prak_tdb_modul_3_primary dengan menggunakan perintah berikut docker exec -it prak_tbd_modul_3_primary bash akses PostgreSQL dengan perintah psql -U postgres -d prak_tbd_modul_3_db	
Statement SQL: 1. `ssh ubuntu@<public_ip_address>` 2. `sudo docker ps` 3. `sudo docker exec -it prak_tbd_modul_3_primary bash` 4. `psql -U postgres -d prak_tbd_modul_3_db`	**Tujuan/Penjelasan Query:** 1. Untuk mengakses server Oracle Cloud menggunakan SSH(Secure Shell. 2. Untuk menampilkan daftar container Docker yang sedang berjalan. 3. Untuk masuk ke dalam container Docker yang sedang berjalan dan menjalankan shell di dalamnya. 4. Untuk mengakses database PostgreSQL bernama ‘prak_tbd_modul_3_db’ sebagai user postgres.

Hasil Query/SQL:

```
ubuntu@prak-tbd: ~

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

16 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


*** System restart required ***
Last login: Sat Nov  2 08:36:35 2024 from 182.253.218.121
ubuntu@prak-tbd:~$ sudo docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS       PORTS
     NAMES
08632f39d0dd   postgres   "docker-entrypoint.s…"   16 hours ago   Up 2 hours   0.0.0.0:6544->5432/tcp, [::]:6544->5432/tcp   prak_tbd_modul_3_standby
081ad4c618e5   postgres   "docker-entrypoint.s…"   16 hours ago   Up 2 hours   0.0.0.0:6543->5432/tcp, [::]:6543->5432/tcp   prak_tbd_modul_3_primary
ubuntu@prak-tbd:~$ cd prak_tbd
ubuntu@prak-tbd:~/prak_tbd$ ls
docker-compose.yml
ubuntu@prak-tbd:~/prak_tbd$ cd
ubuntu@prak-tbd:~$ sudo docker exec -it prak_tbd_modul_3_primary bash
root@081ad4c618e5:/# psql -U postgres -d prak_tbd_modul_3_db
psql (17.0 (Debian 17.0-1.pgdg120+1))
Type "help" for help.
prak_tbd_modul_3_db=#
prak_tbd_modul_3_db=#
```

Analisa Error: -

Penerapan Solusi Akhir: -

Penjelasan Statement: -

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 2 soal 2

Soal Latihan:

Membuat Tabel dan Data Awal.

Buatlah tabel akun dengan saldo awal seperti di bawah ini

CREATE TABLE akun (

id SERIAL PRIMARY KEY,

nama VARCHAR(50),

saldo INT);

INSERT INTO akun (nama, saldo) VALUES ('User1', 1000);

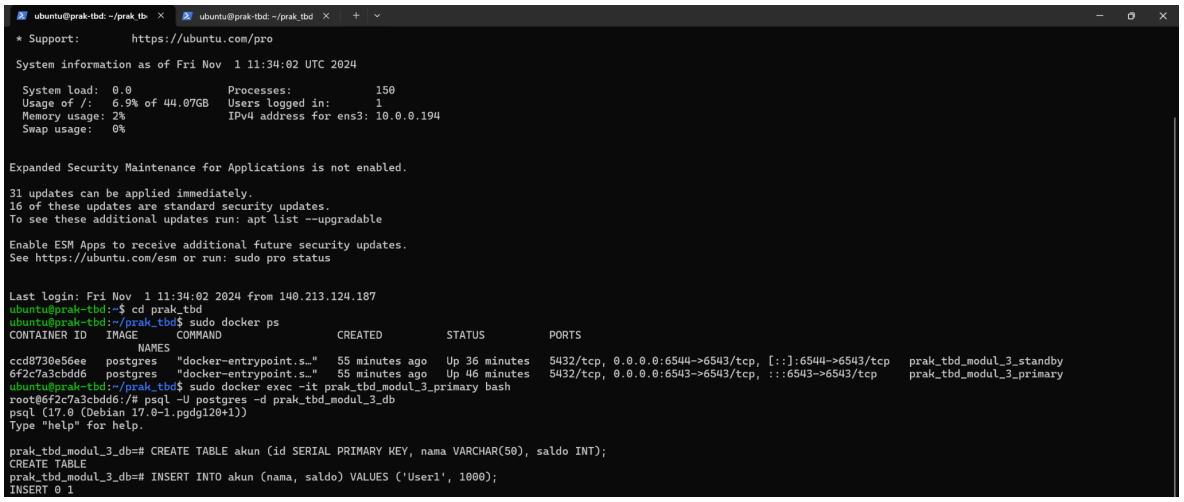
Statement SQL:

```
1. CREATE TABLE akun (
   id SERIAL PRIMARY KEY,
   nama VARCHAR(50),
   saldo INT
   );
2. INSERT INTO akun (nama, saldo) VALUES ('User1', 1000);
```

Tujuan/Penjelasan Query:

1. Untuk membuat tabel ‘akun’ dengan kolom-kolom dan tipe data yang telah ditentukan.
2. Untuk menambahkan data(saldo awal) ke tabel akun dengan nilai-nilai yang telah ditentukan.

Hasil Query/SQL:



```
ubuntu@prak-tbd: ~/prak_tbd
 * Support:        https://ubuntu.com/pro

 System information as of Fri Nov  1 11:34:02 UTC 2024

  System load:  0.0               Processes:             150
  Usage of /:   6.9% of 44.07GB   Users logged in:       1
  Memory usage: 2%                IPv4 address for ens3: 10.0.0.194
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

31 updates can be applied immediately.
16 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


Last login: Fri Nov  1 11:34:02 2024 from 140.213.124.187
ubuntu@prak-tbd:~$ cd prak_tbd
ubuntu@prak-tbd:~/prak_tbd$ sudo docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED          STATUS          PORTS
                    NAMES
ccd8730e56ee   postgres   "docker-entrypoint.s…"   55 minutes ago   Up 36 minutes   5432/tcp, 0.0.0.0:6544->6543/tcp, [::]:6544->6543/tcp   prak_tbd_modul_3_standby
6f2c7a3cbdd6   postgres   "docker-entrypoint.s…"   55 minutes ago   Up 46 minutes   5432/tcp, 0.0.0.0:6543->6543/tcp, :::6543->6543/tcp     prak_tbd_modul_3_primary
ubuntu@prak-tbd:~/prak_tbd$ sudo docker exec -it prak_tbd_modul_3_primary bash
root@6f2c7a3cbdd6:/# psql -U postgres -d prak_tbd_modul_3_db
psql (17.0 (Debian 17.0-1.pgdg120+1))
Type "help" for help.

prak_tbd_modul_3_db=# CREATE TABLE akun (id SERIAL PRIMARY KEY, nama VARCHAR(50), saldo INT);
CREATE TABLE
prak_tbd_modul_3_db=# INSERT INTO akun (nama, saldo) VALUES ('User1', 1000);
INSERT 0 1
```

Analisa Error: -

Penerapan Solusi Akhir: -

Penjelasan Statement: -

Hasil Penerapan Solusi Akhir: -

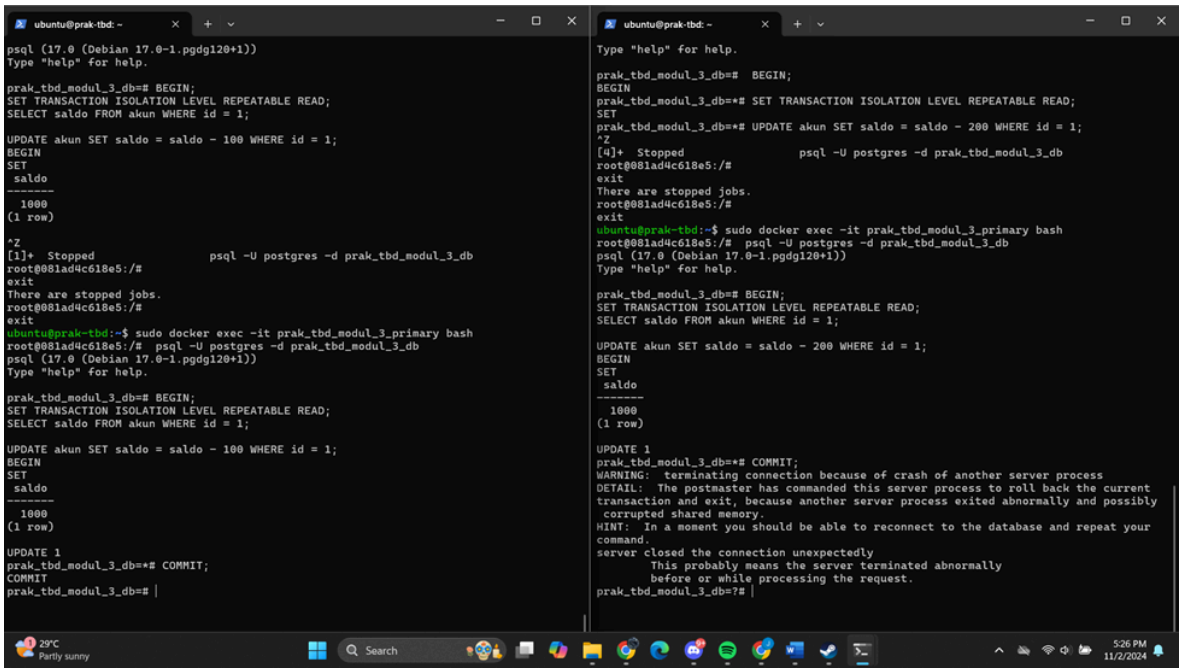
No. Latihan: Percobaan 2 soal 3-5	
Soal Latihan: Sekarang lakukan dua transaksi yang mencoba mengubah saldo akun secara bersamaan dengan membuat dua terminal PostgreSQL (Session A dan Session B) pada prak_tbd_modul_3_primary untuk mewakili dua pengguna yang mencoba mengakses data akun. Transaksi pertama (Terminal A) Pada Session A, lakukan langkah-langkah berikut BEGIN; SET TRANSACTION ISOLATION LEVEL REPEATABLE READ; SELECT saldo FROM akun WHERE id = 1; UPDATE akun SET saldo = saldo - 100 WHERE id = 1; Transaksi Kedua (Terminal B) Pada Session B, lakukan langkah-langkah berikut BEGIN; SET TRANSACTION ISOLATION LEVEL REPEATABLE READ; SELECT saldo FROM akun WHERE id = 1; UPDATE akun SET saldo = saldo - 200 WHERE id = 1; Mengamati Hasil Pada Terminal A, Lakukan COMMIT; Pada Terminal B, Lakukan COMMIT;	
Statement SQL:	**Tujuan/Penjelasan Query:**

1. `BEGIN;`
`SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;`
`SELECT saldo FROM akun WHERE id = 1;`
 	
`UPDATE akun SET saldo = saldo - 100 WHERE id = 1;`
 	
2. `BEGIN;`
`SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;`
	1. `BEGIN;`
	agar semua perubahan data yang dilakukan tidak akan langsung diterapkan pada database sampai kita melakukan COMMIT;
	
	`SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;`
	agar semua data yang dibaca dalam transaksi tetap konsisten sepanjang transaksi berlangsung.

`SELECT saldo FROM akun WHERE id = 1;`
 	
`UPDATE akun SET saldo = saldo - 200 WHERE id = 1;`
 	
3. SESSION A `COMMIT;`
4. SESSION B `COMMIT;`
	`SELECT saldo FROM akun WHERE id = 1;`
	untuk Mengambil nilai saldo dari akun dengan id = 1.
	
	`UPDATE akun SET saldo = saldo - 100 WHERE id = 1;`
	untuk mengupdate tabel akun, mengurangi saldo dari akun yang memiliki id = 1 sebesar 100;
	2. `BEGIN;`
	agar semua perubahan data yang dilakukan tidak akan langsung diterapkan pada database sampai kita melakukan COMMIT;
	
	`SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;`
	agar semua data yang dibaca dalam transaksi tetap konsisten sepanjang transaksi berlangsung.
	
	`SELECT saldo FROM akun WHERE id = 1;`
	untuk Mengambil nilai saldo dari akun dengan id = 1.
	
	`UPDATE akun SET saldo = saldo - 200 WHERE id = 1;`

	untuk mengupdate tabel akun, mengurangi saldo dari akun yang memiliki id = 1 sebesar 200; 3. Untuk menyelesaikan transaksi yang sedang berlangsung dan menyimpan semua perubahan yang telah dilakukan selama sesi A ke dalam database secara permanen. 4. Untuk menyelesaikan transaksi yang sedang berlangsung dan menyimpan semua perubahan yang telah dilakukan selama sesi B ke dalam database secara permanen.

Hasil Query/SQL:



```
ubuntu@prak-tbd: ~
psql (17.0 (Debian 17.0-1.pgdg120+1))
Type "help" for help.

prak_tbd_modul_3_db=# BEGIN;
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SELECT saldo FROM akun WHERE id = 1;

UPDATE akun SET saldo = saldo - 100 WHERE id = 1;
BEGIN
SET
 saldo
-------
  1000
(1 row)

^Z
[1]+  Stopped                 psql -U postgres -d prak_tbd_modul_3_db
root@081ad4c618e5:/#
exit
There are stopped jobs.
root@081ad4c618e5:/#
exit
ubuntu@prak-tbd:~$ sudo docker exec -it prak_tbd_modul_3_primary bash
root@081ad4c618e5:/#  psql -U postgres -d prak_tbd_modul_3_db
psql (17.0 (Debian 17.0-1.pgdg120+1))
Type "help" for help.

prak_tbd_modul_3_db=# BEGIN;
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SELECT saldo FROM akun WHERE id = 1;

UPDATE akun SET saldo = saldo - 100 WHERE id = 1;
BEGIN
SET
 saldo
-------
  1000
(1 row)

UPDATE 1
prak_tbd_modul_3_db=*# COMMIT;
COMMIT
prak_tbd_modul_3_db=#
```

```
ubuntu@prak-tbd: ~
Type "help" for help.

prak_tbd_modul_3_db=#  BEGIN;
BEGIN
prak_tbd_modul_3_db=*# SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SET
prak_tbd_modul_3_db=*# UPDATE akun SET saldo = saldo - 200 WHERE id = 1;
^Z
[4]+  Stopped                 psql -U postgres -d prak_tbd_modul_3_db
root@081ad4c618e5:/#
exit
There are stopped jobs.
root@081ad4c618e5:/#
exit
ubuntu@prak-tbd:~$ sudo docker exec -it prak_tbd_modul_3_primary bash
root@081ad4c618e5:/#  psql -U postgres -d prak_tbd_modul_3_db
psql (17.0 (Debian 17.0-1.pgdg120+1))
Type "help" for help.

prak_tbd_modul_3_db=# BEGIN;
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
SELECT saldo FROM akun WHERE id = 1;

UPDATE akun SET saldo = saldo - 200 WHERE id = 1;
BEGIN
SET
 saldo
-------
  1000
(1 row)

UPDATE 1
prak_tbd_modul_3_db=*# COMMIT;
WARNING:  terminating connection because of crash of another server process
DETAIL:  The postmaster has commanded this server process to roll back the current
transaction and exit, because another server process exited abnormally and possibly
 corrupted shared memory.
HINT:  In a moment you should be able to reconnect to the database and repeat your
command.
server closed the connection unexpectedly
        This probably means the server terminated abnormally
        before or while processing the request.
prak_tbd_modul_3_db=?#
```

Analisa Error: -

Penerapan Solusi Akhir: -	**Penjelasan Statement: -**

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 2 soal 6

Soal Latihan:

Memeriksa Konsistensi Data.

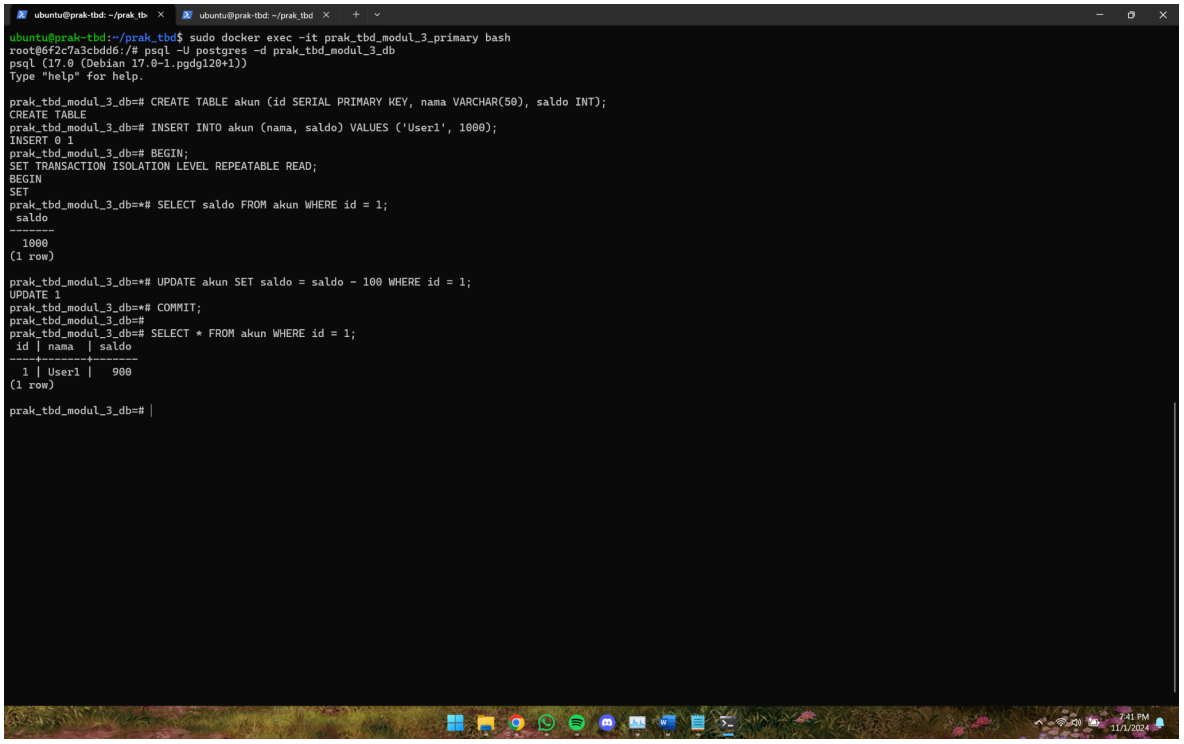
Setelah kedua transaksi berhasil, periksa saldo akun di PostgreSQL untuk memastikan hasil akhirnya benar dan tidak terjadi dirty read atau anomali.

Jika transaksi bersamaan ditangani dengan benar, maka dapat dilihat bahwa PostgreSQL telah menjaga konsistensi saldo melalui kontrol bersamaan.

Pengujian ini berhasil jika setiap transaksi hanya melihat data yang sudah di-commit dan tidak terjadi race condition atau ketidakkonsistenan data.

Statement SQL:	**Tujuan/Penjelasan Query:**
`SELECT * FROM akun WHERE id = 1;`	Mengambil data dengan id = 1 dari tabel akun, memastikan hasil akhir setelah kedua transaksi benar dan tidak terjadi dirty read/anomali.

Hasil Query/SQL:



Analisa Error: -

Penerapan Solusi Akhir: -	**Penjelasan Statement: -**

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 3 soal 1

Soal Latihan:

Siapkan Database di Primary dan Standby.

Pada server primary,pastikan database telah disiapkan dengan data yang sesuai (menggunakan tabel akun pada percobaan 2 sebelumnya)

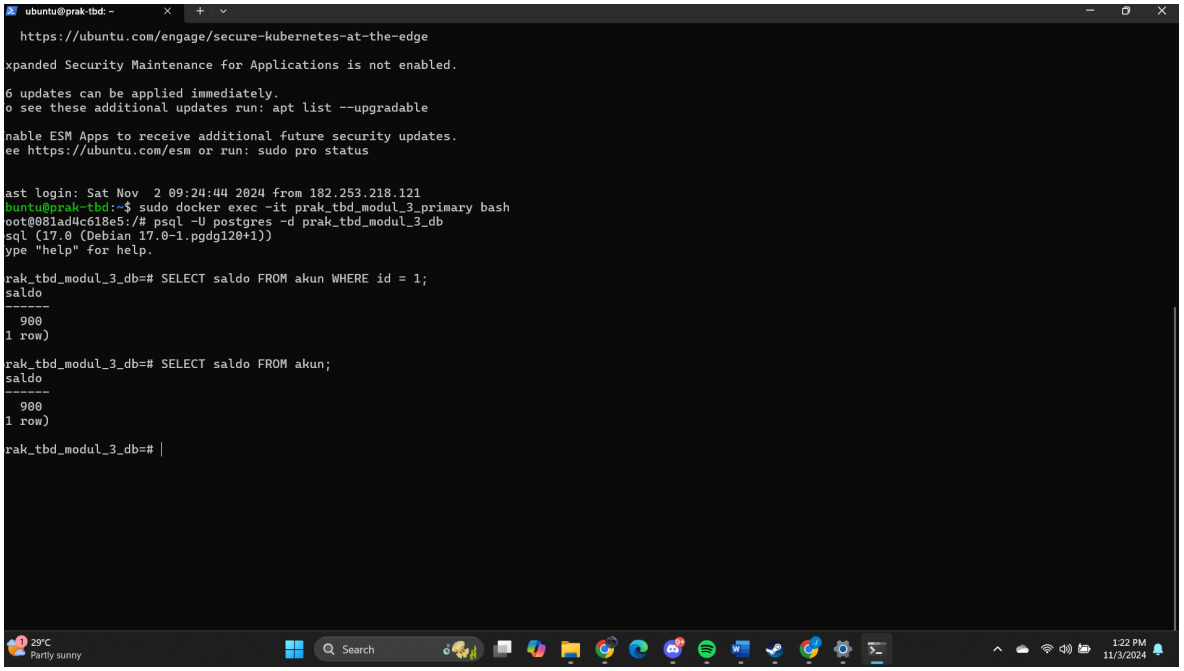
Statement SQL:

1. `sudo docker exec -it prak_tbd_modul_3_primary bash`
2. `psql -U postgres -d prak_tbd_modul_3_db`
3. `SELECT saldo FROM akun;`

Tujuan/Penjelasan Query:

1. Masuk ke dalam container prak_tbd_modul_3_primary dan menjalankan shell bash di dalamnya.
2. Mengakses database PostgreSQL bernama prak_tbd_modul_3_db sebagai user postgres.
3. Mengambil semua nilai saldo dari tabel akun.

Hasil Query/SQL:



Analisa Error: -

Penerapan Solusi Akhir: -

Penjelasan Statement: -

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 3 soal 2	
Soal Latihan: Hentikan Layanan PostgreSQL pada Server Primary. Hentikan layanan PostgreSQL pada container primary untuk mensimulasikan kegagalan pada server primary. docker stop prak_tbd_modul_3_primary	
Statement SQL: `sudo docker stop prak_tbd_modul_3_primary`	**Tujuan/Penjelasan Query:** Untuk menghentikan container primary yang sedang berjalan.

Hasil Query/SQL:
`ubuntu@prak-tbd:~$ sudo docker stop prak_tbd_modul_3_primary`
`prak_tbd_modul_3_primary`	

Analisa Error: -	
Penerapan Solusi Akhir: -	**Penjelasan Statement: -**
Hasil Penerapan Solusi Akhir: -	

No. Latihan: Percobaan 3 soal 3

Soal Latihan:

Promosikan Standby Menjadi Primary Baru.

Masukan ke container standby dan jalankan perintah dibawah untuk mengkonversi server standby menjadi primary baru

docker exec -it prak_tbd_modul_3_standby bash

psql -U postgres -d prak_tbd_modul_3_db -c "SELECT pg_promote();"

Statement SQL:

1. `sudo docker exec -it prak_tbd_modul_3_standby bash`
2. `psql -U postgres -d prak_tbd_modul_3_db -c "SELECT pg_promote();"`

Tujuan/Penjelasan Query:

1. Untuk masuk ke dalam container standby yang sedang berjalan.
2. Untuk mempromosikan container standby menjadi primary.

Hasil Query/SQL:

```
ubuntu@prak-tbd:~$ sudo docker exec -it prak_tbd_modul_3_standby bash
root@08632f39d0dd:/# psql -U postgres -d prak_tbd_modul_3_db -c "SELECT pg_promote();"
 pg_promote
------------
 t
(1 row)
```

Analisa Error: -

Penerapan Solusi Akhir: -

Penjelasan Statement: -

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 3 soal 4

Soal Latihan:

Verifikasi Peran Primary di Server Standby.

Verifikasi ke container standby telah menjadi primary dengan menjalankan

SELECT pg_is_in_recovery();

Jika hasilnya false, maka server standby sekarang berfungsi sebagai primary.

Statement SQL:	**Tujuan/Penjelasan Query:**
`psql -U postgres -d prak_tbd_modul_3_db -c "SELECT pg_is_in_recovery();"`	Untuk memeriksa status recovery, mengembalikan true jika server saat ini dalam mode recovery, dan false jika server berfungsi sebagai primary.

Hasil Query/SQL:

```
root@08632f39d0dd:/# psql -U postgres -d prak_tbd_modul_3_db -c "SELECT pg_is_in_recovery();"
 pg_is_in_recovery
-------------------
 f
(1 row)
```

Analisa Error: -

Penerapan Solusi Akhir: -	**Penjelasan Statement: -**

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 3 soal 5

Soal Latihan:

Lakukan Operasi Tulis di Primary Baru(Standby Lama).

Di server standby(yang sekarang menjadi primary baru), lakukan beberapa operasi tulis untuk memastikan bahwa failover berjalan dengan baik

INSERT INTO akun (nama, saldo) VALUES ('User2', 500);

UPDATE akun SET saldo = saldo + 200 WHERE nama = 'User1';

Statement SQL:

1. `psql -U postgres -d prak_tbd_modul_3_db`
2. `INSERT INTO akun (nama, saldo) VALUES ('User2', 500);`
3. `UPDATE akun SET saldo = saldo + 200 WHERE nama = 'User1';`

Tujuan/Penjelasan Query:

1. Mengakses database PostgreSQL bernama ‘prak_tbd_modul_3_db’ sebagai user postgres.
2. Memasukkan data ke tabel akun sesuai dengan nilai-nilai yang telah ditentukan
3. Mengupdate saldo ‘user1’ dalam tabel akun sebanyak saldo+200.

Hasil Query/SQL:

```
root@08632f39d0dd:/# psql -U postgres -d prak_tbd_modul_3_db
psql (17.0 (Debian 17.0-1.pgdg120+1))
Type "help" for help.

prak_tbd_modul_3_db=# INSERT INTO akun (nama, saldo) VALUES ('User2', 500);
INSERT 0 1
prak_tbd_modul_3_db=# UPDATE akun SET saldo = saldo + 200 WHERE nama = 'User1';
UPDATE 1
prak_tbd_modul_3_db=#
```

Analisa Error: -

Penerapan Solusi Akhir: -

Penjelasan Statement: -

Hasil Penerapan Solusi Akhir: -

No. Latihan: Percobaan 3 soal 6	
Soal Latihan: Restart Server Primary Lama dan Konfigurasi Sebagai Standby Baru. Setelah selesai menulis operasi di atas, nyalakan ulang server primary lama docker start prak_tbd_modul_3_primary. Kemudian, untuk membuat primary lama kembali menjadi standby maka lakukan ulang langkah-langkah pada percobaan 1 tapi kali ini standby yang lama akan menjadi primary baru sedangkan primary lama akan menjadi standby baru.	
Statement SQL: 1. `sudo docker start prak_tbd_modul_3_primary` 2. `sudo docker exec -it prak_tbd_modul_3_standby bash` 3. `nano /var/lib/postgresql/data/postgresql.conf` 4. `psql -U postgres -c "CREATE ROLE replicator WITH REPLICATION PASSWORD 'replica_password' LOGIN;"` 5. `nano /var/lib/postgresql/data/pg_hba.conf` 6. `sudo docker restart prak_tbd_modul_3_standby` 7. `sudo docker exec -it prak_tbd_modul_3_primary bash` 8. `rm -rf /var/lib/postgresql/data/*`	**Tujuan/Penjelasan Query:** 1. Menyalakan container Docker prak_tbd_modul_3_primary yang sebelumnya dihentikan. 2. Untuk masuk ke dalam container Docker prak_tbd_modul_3_standby dan menjalankan shell bash di dalamnya. 3. Untuk membuka file konfigurasi PostgreSQL bernama postgresql.conf menggunakan editor teks nano. 4. Membuat role(peran) baru dalam PostgreSQL dengan nama replicator. 5. Untuk membuka file konfigurasi PostgreSQL bernama postgresql.conf menggunakan editor teks nano. 6. Untuk merestart container Docker prak_tbd_modul_3_standby. 7. Untuk masuk ke dalam container Docker prak_tbd_modul_3_primary bash dan menjalankan shell bash di dalamnya.

9. `pg_basebackup -h prak_tbd_modul_3_standby -D /var/lib/postgresql/data -U replicator -v -P --wal-method=stream` 10. `touch /var/lib/postgresql/data/standby.signal` 11. `nano /var/lib/postgresql/data/postgresql.conf` 12. `sudo docker restart prak_tbd_modul_3_primary` 13. `sudo docker exec -it prak_tbd_modul_3_primary bash` 14. `psql -U postgres -c "SELECT * FROM pg_stat_replication;"` 15. `psql -U postgres -c "CREATE TABLE test_replication (id SERIAL PRIMARY KEY, name TEXT);"` 16. `psql -U postgres -c "\dt"`	8. untuk menghapus semua file dan direktori yang ada di dalam direktori /var/lib/postgresql/data/. 9. Untuk membuat salinan cadangan (backup) dari database PostgreSQL secara fisik. 10. Untuk membuat file kosong bernama standby.signal di dalam direktori /var/lib/postgresql/data/. 11. Untuk membuka file konfigurasi postgresql.conf menggunakan editor teks nano. 12. Untuk merestart container Docker 'prak_tbd_modul_3_primary'. 13. Untuk masuk ke dalam container Docker prak_tbd_modul_3_primary dan menjalankan shell bash di dalamnya. 14. Untuk memberikan informasi tentang semua koneksi replikasi yang aktif ke server PostgreSQL. 15. Untuk membuat tabel 'test_replication' di database PostgreSQL dengan struktur tabel yang telah ditentukan. 16. Untuk melihat daftar tabel yang ada dalam PostgreSQL.

Hasil Query/SQL:

```
ubuntu@prak-tbd:~$ sudo docker start prak_tbd_modul_3_primary
prak_tbd_modul_3_primary
```

```
ubuntu@prak-tbd:~$ sudo docker exec -it prak_tbd_modul_3_standby bash
root@08632f39d0dd:/# nano /var/lib/postgresql/data/postgresql.conf
root@08632f39d0dd:/# psql -U postgres -c "CREATE ROLE replicator WITH REPLICATION PASSWORD
'replica_password' LOGIN;"
CREATE ROLE
root@08632f39d0dd:/# nano /var/lib/postgresql/data/pg_hba.conf
root@08632f39d0dd:/#
```

```
ubuntu@prak-tbd: ~
  GNU nano 7.2                    /var/lib/postgresql/data/postgresql.conf *
# - Other Platforms and Clients -

#transform_null_equals = off
#allow_alter_system = on


#------------------------------------------------------------------------------
# ERROR HANDLING
#------------------------------------------------------------------------------

#exit_on_error = off                    # terminate session on any error?
#restart_after_crash = on               # reinitialize after backend crash?
#data_sync_retry = off                  # retry or panic on failure to fsync
                                        # data?
                                        # (change requires restart)
#recovery_init_sync_method = fsync      # fsync, syncfs (Linux 5.8+)


#------------------------------------------------------------------------------
# CONFIG FILE INCLUDES
#------------------------------------------------------------------------------

# These options allow settings to be loaded from files other than the
# default postgresql.conf.  Note that these are directives, not variable
# assignments, so they can usefully be given more than once.

#include_dir = '...'                    # include files ending in '.conf' from
                                        # a directory, e.g., 'conf.d'
#include_if_exists = '...'               # include file only if it exists
#include = '...'                        # include file


#------------------------------------------------------------------------------
# CUSTOMIZED OPTIONS
#------------------------------------------------------------------------------

# Add settings for extensions here
wal_level = replica
archive_mode = on
archive_command = 'cp %p /var/lib/postgresql/data/archive/%f'
max_wal_senders = 3
wal_keep_size = 64
```

```
GNU nano 7.2                    /var/lib/postgresql/data/pg_hba.conf
CAUTION: Configuring the system for local "trust" authentication
allows any local user to connect as any PostgreSQL user, including
the database superuser.  If you do not trust all your local users,
use another authentication method.


TYPE  DATABASE        USER            ADDRESS                 METHOD

"local" is for Unix domain socket connections only
cal   all             all                                     trust
IPv4 local connections:
st    all             all             127.0.0.1/32            trust
IPv6 local connections:
st    all             all             ::1/128                 trust
Allow replication connections from localhost, by a user with the
replication privilege.
cal   replication     all                                     trust
st    replication     all             127.0.0.1/32            trust
st    replication     all             ::1/128                 trust
st    replication     replicator      0.0.0.0/0               md5

st all all all scram-sha-256
```

```
ubuntu@prak-tbd:~$ sudo docker restart prak_tbd_modul_3_standby
prak_tbd_modul_3_standby
ubuntu@prak-tbd:~$ sudo docker exec -it prak_tbd_modul_3_primary bash
root@331a35dee3a1:/# rm -rf /var/lib/postgresql/data/*
root@331a35dee3a1:/# pg_basebackup -h prak_tbd_modul_3_standby -D /var/lib/postgresql/data -U replicator -v -P --wal-method=stream
Password:
pg_basebackup: initiating base backup, waiting for checkpoint to complete
pg_basebackup: checkpoint completed
pg_basebackup: write-ahead log start point: 0/2000028 on timeline 1
pg_basebackup: starting background WAL receiver
pg_basebackup: created temporary replication slot "pg_basebackup_38"
30607/30607 kB (100%), 1/1 tablespace
pg_basebackup: write-ahead log end point: 0/2000158
pg_basebackup: waiting for background process to finish streaming ...
pg_basebackup: syncing data to disk ...
pg_basebackup: renaming backup_manifest.tmp to backup_manifest
pg_basebackup: base backup completed
root@331a35dee3a1:/# ubuntu@prak-tbd:~$
ubuntu@prak-tbd:~$ sudo docker exec -it prak_tbd_modul_3_primary bash
root@331a35dee3a1:/# touch /var/lib/postgresql/data/standby.signal
root@331a35dee3a1:/# nano /var/lib/postgresql/data/postgresql.conf

root@331a35dee3a1:/#
```

```
ubuntu@prak-tbd: ~
  GNU nano 7.2                                  /var/lib/postgresql/data/postgresql.conf *
# CONFIG FILE INCLUDES                         /var/lib/postgresql/data/postgresql.conf *
#------------------------------------------------------------------------------

# These options allow settings to be loaded from files other than the
# default postgresql.conf.  Note that these are directives, not variable
# assignments, so they can usefully be given more than once.

#include_dir = '...'                    # include files ending in '.conf' from
                                        # a directory, e.g., 'conf.d'
#include_if_exists = '...'              # include file only if it exists
#include = '...'                        # include file


#------------------------------------------------------------------------------
# CUSTOMIZED OPTIONS
#------------------------------------------------------------------------------

# Add settings for extensions here
primary_conninfo = 'host = prak_tbd_modul_3_standby port = 5432 user = replicator password = replica_password'
```

```
ubuntu@prak-tbd:~$ sudo docker restart prak_tbd_modul_3_standby
prak_tbd_modul_3_standby
ubuntu@prak-tbd:~$ sudo docker restart prak_tbd_modul_3_primary
prak_tbd_modul_3_primary
ubuntu@prak-tbd:~$ sudo docker exec -it prak_tbd_modul_3_primary bash
root@331a35dee3a1:/# psql -U postgres -c "SELECT * FROM pg_stat_replication;"
root@331a35dee3a1:/#
```

```
| usesysid |  usename   | application_name | client_addr | client_hostname | client_port |         backend_start         | backend_xmin |   state   | s
lsn  | write_lsn | flush_lsn | replay_lsn | write_lag | flush_lag | replay_lag | sync_priority | sync_state |          reply_time
-+----------+------------+------------------+-------------+-----------------+-------------+-------------------------------+--------------+-----------+--
-----+-----------+-----------+-----------+----------+----------+----------+---------------+------------+-------------------------------
|    16389 | replicator | walreceiver      | 172.18.0.3  |                 |       36982 | 2024-11-03 06:19:45.906122+00 |              | streaming | 0
00D8 | 0/40000D8 | 0/40000D8 | 0/40000D8  |           |           |           |             0 | async      | 2024-11-03 06:20:22.740112+00
ow)
```

```
root@08632f39d0dd:/# psql -U postgres -c "CREATE TABLE test_replication (id SERIAL PRIMARY
KEY, name TEXT);"
CREATE TABLE
root@08632f39d0dd:/# psql -U postgres -c "\dt"
              List of relations
 Schema |       Name       | Type  |  Owner
--------+------------------+-------+----------
 public | test_replication | table | postgres
(1 row)
```

Analisa Error: -

Penerapan Solusi Akhir: -	**Penjelasan Statement: -**

Hasil Penerapan Solusi Akhir: -

TUGAS

No. Tugas: 1

Soal Tugas:

Gunakan docker-compose.yml yang disediakan, tetapi modifikasi konfigurasi agar postgres_primary dan postgres_standby menyimpan data pada volume terpisah. buat tabel sederhana di primary dan isi dengan beberapa data. Verifikasi bahwa standby telah berhasil mereplikasi tabel dan data dari primary. Dokumentasikan konfigurasi yang telah dibuat dan berikan tangkapan layar hasil uji replikasi awal.

Statement SQL:	**Tujuan/Penjelasan Query:**
1. `nano docker-compose.yml` 2. `sudo docker exec -it prak_tbd_modul_3_primary_tugas bash` 3. `psql -U postgres -c "CREATE TABLE testugas1 (id SERIAL PRIMARY KEY, name TEXT);"` 4. `psql -U postgres -c "\dt"` 5. `psql -U postgres -c "INSERT INTO testugas1 values(1, 'A');"` 6. `psql -U postgres -c "SELECT * FROM testugas1;"` 7. `sudo docker exec -it prak_tbd_modul_3_standby_tugas bash` 8. `psql -U postgres -c "\dt"` 9. `psql -U postgres -c "SELECT * FROM testugas1;"`	1. Untuk membuka file konfigurasi Docker Compose yang bernama docker-compose.yml menggunakan editor teks nano. 2. Untuk menjalankan shell interaktif di dalam container Docker 'prak_tbd_modul_3_primary_tugas'. 3. Untuk membuat tabel testugas1 di database PostgreSQL dengan struktur tabel yang telah ditentukan. 4. Untuk menampilkan daftar tabel yang ada dalam database PostgreSQL. 5. Untuk memasukkan data ke dalam tabel testugas1 dengan nilai yang telah ditentukan. 6. Untuk menampilkan semua data pada tabel testugas1 di database PostgreSQL. 7. Untuk menjalankan shell interaktif dalam container Docker 'prak_tbd_modul_3_standby_tugas'. 8. Untuk menampilkan daftar tabel yang ada dalam database PostgreSQL.Untuk menampilkan

	semua data pada tabel testugas1 di database PostgreSQL.

Hasil Query/SQL:

```
GNU nano 7.2                                    docker-compose.yml
version: '3.8'
services:
  postgres_primary:
    container_name: prak_tbd_modul_3_standby
    image: postgres
    environment:
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: password
      POSTGRES_DB: prak_tbd_modul_3_db
    volumes:
      - db_data:/var/lib/postgresql/data
    ports:
      - "7543:5432"
    networks:
      - prak_tbd_network
    restart: unless-stopped

  postgres_standby:
    container_name: prak_tbd_modul_3_primary
    image: postgres
    environment:
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: password
      POSTGRES_DB: prak_tbd_modul_3_db
    volumes:
      - db_data_standby:/var/lib/postgresql/data
    ports:
      - "7544:5432"
    networks:
      - prak_tbd_network
    restart: unless-stopped

  postgres_primary_tugas:
    container_name: prak_tbd_modul_3_primary_tugas
    image: postgres
    environment:
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: password
      POSTGRES_DB: prak_tbd_modul_3_db
    volumes:
      - db_data_tugas:/var/lib/postgresql/data
    ports:
      - "7542:5432"
    networks:
      - prak_tbd_network
    restart: unless-stopped

  postgres_standby_tugas:
    container_name: prak_tbd_modul_3_standby_tugas
    image: postgres
    environment:
      POSTGRES_USER: postgres
      POSTGRES_PASSWORD: password
      POSTGRES_DB: prak_tbd_modul_3_db
    volumes:
      - db_data_standby_tugas:/var/lib/postgresql/data
    ports:
      - "7541:5432"
    networks:
      - prak_tbd_network
    restart: unless-stopped

volumes:
  db_data:
  db_data_standby:
  db_data_tugas:
  db_data_standby_tugas:

networks:
  prak_tbd_network:
```

```
ubuntu@prak-tbd:~/prak_tbd$ sudo docker exec -it prak_tbd_modul_3_primary_tugas bash
root@5e750c671bd0:/# psql -U postgres -c "SELECT * FROM pg_stat_replication;"

[1]+  Stopped                 psql -U postgres -c "SELECT * FROM pg_stat_replication;"
root@5e750c671bd0:/#
exit
There are stopped jobs.
root@5e750c671bd0:/#
exit
ubuntu@prak-tbd:~/prak_tbd$ sudo docker exec -it prak_tbd_modul_3_primary_tugas bash
root@5e750c671bd0:/# psql -U postgres -c "CREATE TABLE testugas1 (id SERIAL PRIMARY KEY, name TEXT);"
CREATE TABLE
root@5e750c671bd0:/# psql -U postgres -c "\dt"
           List of relations
 Schema |   Name    | Type  |  Owner
--------+-----------+-------+----------
 public | testugas1 | table | postgres
(1 row)

root@5e750c671bd0:/# psql -U postgres -c "INSERT INTO testugas1 values(1, 'A');"
INSERT 0 1
root@5e750c671bd0:/# psql -U postgres -c "SELECT * FROM testugas1;"
 id | name
----+------
  1 | A
(1 row)

root@5e750c671bd0:/#
exit
ubuntu@prak-tbd:~/prak_tbd$ sudo docker exec -it prak_tbd_modul_3_standby_tugas bash
root@a62017e642d2:/# psql -U postgres -c "\dt"
           List of relations
 Schema |   Name    | Type  |  Owner
--------+-----------+-------+----------
 public | testugas1 | table | postgres
(1 row)

root@a62017e642d2:/# psql -U postgres -c "SELECT * FROM testugas1;"
 id | name
----+------
  1 | A
(1 row)
```

No. Tugas: 2

Soal Tugas:

Konfigurasikan replikasi asynchronous antara primary dan standby. Kemudian, lakukan batch insert sejumlah data secara bertahap ke primary dan observasi proses replikasi data ke standby. Selama proses insert berlangsung, lakukan pengujian konsistensi data antara primary dan standby. Apakah ada perbedaan atau delay yang signifikan?

Dokumentasikan hasil observasi dan jelaskan penyebab potensi delay pada replikasi asynchronous.

Statement SQL:

```
for j in {0..4}; do
  for i in {1..500}; do
    id=$((i + (j * 500)))
     letter=$(echo $((i % 26 + 1)) | awk '{printf "%c", $1 + 64}') psql -U postgres -c "INSERT INTO testugas1 VALUES ($id, '$letter');"
  done
done
```

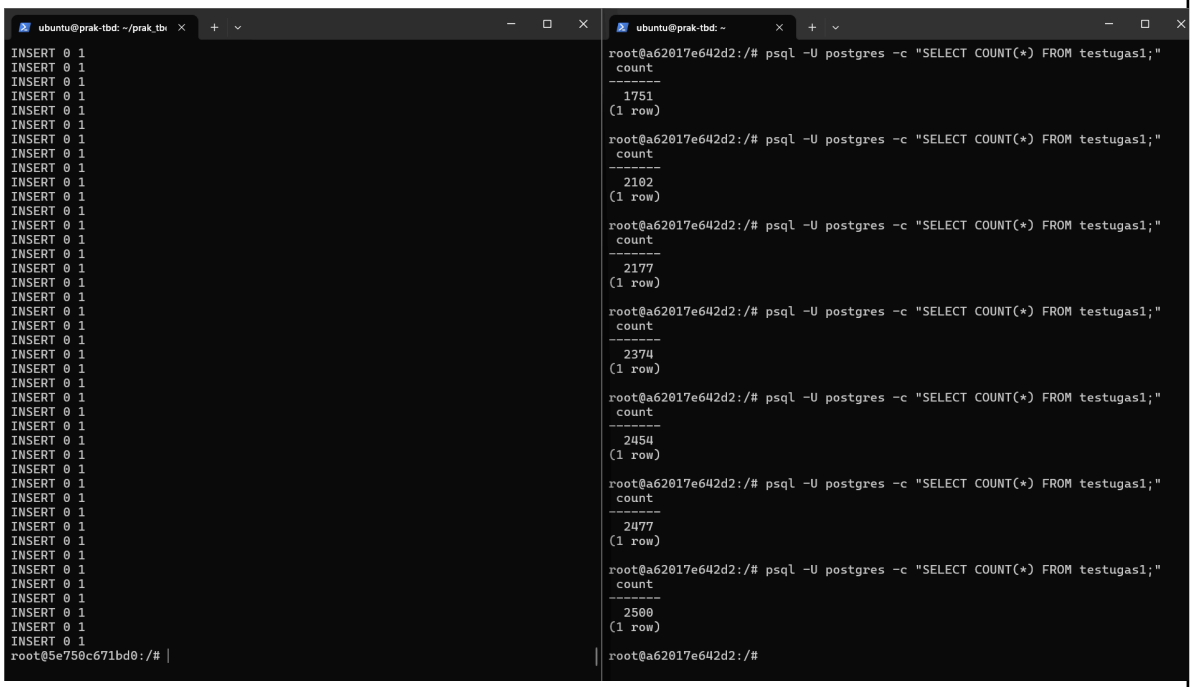
Tujuan/Penjelasan Query:

Mengisi 2500 data ke dalam tabel testugas1, digunakan looping untuk mempercepat pemasukkan data.

Dalam pengujian tidak terlihat ada delay ketika memasukkan 2500 data.

Delay bisa terjadi ketika data yang dimasukkan dalam jumlah yang sangat besar dan 2500 data belum cukup.

Hasil Query/SQL:



No. Tugas: 3

Soal Tugas:

Buat dua skenario update data secara paralel di primary yang mencoba mengakses tabel yang sama. Gunakan teknik locking untuk memastikan transaksi dapat berjalan tanpa deadlock. Catat bagaimana PostgreSQL menangani proses locking ini dan amati apakah ada konflik yang terjadi. Jelaskan hasil pengujian yang dilakukan dan dokumentasikan strategi yang digunakan PostgreSQL untuk menangani concurrency control dalam replikasi asynchronous.

Statement SQL:

1. `psql -U postgres -d prak_tbd_modul_3_db`
2. `CREATE TABLE testugas3 (id SERIAL PRIMARY KEY, name TEXT);`
3. `BEGIN;`

   `LOCK TABLE testugas3 IN EXCLUSIVE MODE;`

   `UPDATE testugas3 SET name = 'sesi 1' WHERE id = 1;`
4. `BEGIN;`

   `LOCK TABLE testugas3 IN EXCLUSIVE MODE;`

   `UPDATE testugas3 SET name = 'sesi 1' WHERE id = 1;`

Tujuan/Penjelasan Query:

1. Mengakses database PostgreSQL bernama ‘prak_tbd_modul_3_db’ sebagai user postgres.
2. Membuat tabel bernama “testugas3” dengan kolom:
   a. ID dengan tipe data serial dan sebagai primary key.
   b. NAME dengan tipe data text.
3. Memulai sebuah transaksi, mengunci tabel testugas3 secara eksklusif, dan memperbarui kolom name menjadi 'sesi 1' untuk baris dengan nilai kolom id = 1.
4. Melakukan hal yang sama dengan nomor 3 namun dengan niat melihat apa yang akan terjadi saat query ini dijalankan namun pada sesi bash yang berbeda.

Hasil pengujian menunjukkan bahwa PostgreSQL menggunakan mekanisme locking untuk menangani concurrency control, terutama dalam skenario replikasi

	asynchronous. Pada gambar, terdapat dua transaksi paralel yang mencoba mengakses tabel yang sama di primary. Setiap transaksi mencoba melakukan pembaruan (UPDATE) dan memulai dengan mengunci tabel menggunakan LOCK TABLE ... IN EXCLUSIVE MODE. Ketika transaksi pertama mengunci tabel dan memulai pembaruan, transaksi kedua yang mencoba mengakses tabel yang sama akan menunggu sampai kunci dilepaskan. Ini menunjukkan bahwa PostgreSQL mengelola kontrol konkurensi dengan menerapkan kunci tingkat tabel, sehingga menghindari terjadinya deadlock. Jika transaksi pertama selesai dan di-commit, maka kunci dilepaskan, dan transaksi kedua dapat dilanjutkan. Dalam replikasi asynchronous, strategi ini memastikan bahwa hanya satu transaksi yang dapat memodifikasi data pada satu waktu di primary, sementara standby akan menerima perubahan setelah commit. Dengan demikian, PostgreSQL menghindari konflik data antara transaksi paralel di primary, tetapi perubahan tersebut mungkin tidak langsung terlihat di standby karena adanya jeda waktu (latency) dalam replikasi asynchronous. Strategi ini memberikan keseimbangan antara konsistensi data dan kinerja,

	menjaga agar transaksi di primary tetap konsisten tanpa mengorbankan kemampuan standby untuk mengikuti perubahan secara asinkron.

Hasil Query/SQL:

```
prak_tbd_modul_3_db=# BEGIN;
LOCK TABLE testugas3 IN EXCLUSIVE MODE;
UPDATE testugas3 SET name = 'sesi 1' WHERE id = 1;
BEGIN
LOCK TABLE
UPDATE 1
prak_tbd_modul_3_db=*# COMMIT;
COMMIT
prak_tbd_modul_3_db=# ;
prak_tbd_modul_3_db=#
\q
root@5e750c671bd0:/# '
> ;
>
>
>
bash: unexpected EOF while looking for matching `''
```

```
prak_tbd_modul_3_db=# BEGIN;
LOCK TABLE testugas1 IN EXCLUSIVE MODE;
UPDATE testugas1 SET name = 'sesi 2' WHERE id = 1;
BEGIN
ERROR:  relation "testugas1" does not exist
ERROR:  current transaction is aborted, commands ignored until end of transa
ction block
prak_tbd_modul_3_db=!# BEGIN;
LOCK TABLE testugas3 IN EXCLUSIVE MODE;
UPDATE testugas3 SET name = 'sesi 2' WHERE id = 1;
ERROR:  current transaction is aborted, commands ignored until end of transa
ction block
ERROR:  current transaction is aborted, commands ignored until end of transa
ction block
ERROR:  current transaction is aborted, commands ignored until end of transa
ction block
prak_tbd_modul_3_db=!# ;
```

No. Tugas: 4	
Soal Tugas: Buat user dengan hak akses read-only untuk standby yang akan digunakan hanya untuk melakukan query data di standby, bukan untuk insert atau update. Uji bahwa user ini hanya bisa melakukan operasi baca pada standby. Buat tangkapan layar hasil pengujian dan analisis implikasi keamanan dari pembatasan akses ini. Jelaskan mengapa user dengan akses terbatas penting dalam konteks replikasi dan distribusi data.	
Statement SQL: 1. `sudo docker exec -it prak_tbd_modul_3_standby_tugas bash` 2. `psql -U postgres -c "SELECT * FROM testugas1;"` 3. `psql -U postgres -c "INSERT INTO testugas1 values(2, 'B');"` 4. `psql -U postgres -c "SELECT * FROM testugas1;"`	**Tujuan/Penjelasan Query:** 1. Menggunakan privilege superuser dan menjalankan serta masuk ke dalam shell bash container docker bernama prak_tbd_modul_3_standby_tugas. 2. Menjalankan query untuk menampilkan seluruh data atau nilai-nilai dalam tabel ‘testugas1’ menggunakan ‘psql’ sebagai user ‘postgres’. 3. Menjalankan query untuk memasukkan data atau nilai (2,B) ke dalam tabel testugas1 menggunakan ‘psql’ sebagai user ‘postgres’. 4. Menjalankan query untuk menampilkan seluruh data atau nilai-nilai dalam tabel ‘testugas1’ menggunakan ‘psql’ sebagai user ‘postgres’. Secara default, standby atau replika hanya memiliki akses read-only oleh karena itu perintah INSERT ada error: “ERROR:

	cannot execute INSERT in a read-only transaction”. Pembatasan akses read-only pada user di standby memiliki dampak positif yang signifikan terhadap keamanan. Dengan akses yang terbatas, data di standby terlindungi dari perubahan yang tidak sah, menjaga agar data tetap konsisten dengan primary. Hal ini penting untuk memastikan bahwa standby berfungsi sebagai replika yang akurat tanpa risiko modifikasi yang dapat merusak integritas data. Dengan menjaga standby dalam mode read-only, fungsi utamanya sebagai cadangan terpercaya dari primary tetap terjaga, siap untuk digunakan jika primary mengalami kegagalan. Pembatasan akses ini memastikan standby dapat digunakan untuk query berat tanpa mengganggu operasi utama atau mengorbankan keamanan data di lingkungan terdistribusi.

Hasil Query/SQL:

```
ubuntu@prak-tbd:~$ sudo docker exec -it prak_tbd_modul_3_standby_tugas bash
root@a62017e642d2:/# psql -U postgres -c "SELECT * FROM testugas1;"
 id | name
----+------
  1 | ABC
(1 row)

root@a62017e642d2:/# psql -U postgres -c "INSERT INTO testugas1 values(2, 'B');"
ERROR:  cannot execute INSERT in a read-only transaction
root@a62017e642d2:/# psql -U postgres -c "SELECT * FROM testugas1;"
 id | name
----+------
  1 | ABC
(1 row)
```

No. Tugas: 5	
Soal Tugas: Simulasikan failover manual dengan menghentikan container primary. Setelah standby berfungsi sebagai server utama, tambahkan data baru di standby. Setelah itu, hidupkan kembali primary sebagai standby untuk instance yang sekarang aktif. Pastikan data yang ditambahkan tetap konsisten di kedua container. Dokumentasikan proses failover ini dan analisis bagaimana failover manual dapat menjadi solusi darurat dalam menjaga ketersediaan data.	
Statement SQL: 1. `sudo docker stop prak_tbd_modul_3_primary_tugas` 2. `psql -U postgres -d prak_tbd_modul_3_db -c "SELECT pg_promote();"` 3. `psql -U postgres -c "SELECT pg_is_in_recovery();"` 4. `psql -U postgres -c "INSERT INTO testugas1 values(2, 'B');"` 5. `psql -U postgres -c "SELECT * FROM testugas1;"` 6. `sudo docker start prak_tbd_modul_3_primary_tugas` 7. `sudo docker exec -it prak_tbd_modul_3_standby_tugas bash` 8. `nano /var/lib/postgresql/data/postgresql.conf`	**Tujuan/Penjelasan Query:** 1. Menghentikan container docker yang bernama prak_tbd_modul_3_primary_tugas. 2. Menggunakan psql untuk mempromosikan server standby menjadi primary pada database prak_tbd_modul_3_db. 3. Memeriksa apakah server saat ini berada dalam recovery mode. 4. Memasukkan data baru ke dalam tabel testugas1 dengan nilai (2, 'B'). 5. Menampilkan semua data dari tabel testugas1. 6. Memulai kembali container docker yang bernama prak_tbd_modul_3_primary_tugas. 7. Menggunakan privilege superuser dan menjalankan serta masuk ke dalam shell bash container docker bernama prak_tbd_modul_3_standby_tugas. 8. Membuka file configurasi postgresql menggunakan text editor nano.

9. `sudo docker restart prak_tbd_modul_3_standby_tugas` 10. `sudo docker exec -it prak_tbd_modul_3_primary_tugas bash` 11. `rm -rf /var/lib/postgresql/data/*` 12. `pg_basebackup -h prak_tbd_modul_3_standby_tugas -D /var/lib/postgresql/data -U replicator_tugas -v -P --wal-method=stream` 13. `touch /var/lib/postgresql/data/standby.signal` 14. `nano /var/lib/postgresql/data/postgresql.conf` 15. `sudo docker restart prak_tbd_modul_3_primary_tugas` 16. `sudo docker exec -it prak_tbd_modul_3_primary_tugas bash` 17. `psql -U postgres -c "SELECT * FROM pg_stat_replication;"` 18. `psql -U postgres -c "SELECT * FROM testugas1;"`	9. Merestart container docker prak_tbd_modul_3_standby_tugas. 10. Menggunakan privilege superuser dan menjalankan serta masuk ke dalam shell bash container docker bernama prak_tbd_modul_3_primary_tugas. 11. Menghapus semua data dalam directory /var/lib/postgresql/data. 12. Mengambil backup dari server standby dan menyimpannya ke direktori data di server primary menggunakan pengguna 'replicator_tugas'. 13. Membuat file standby.signal untuk menandakan bahwa server harus berjalan sebagai standby. 14. Membuka file konfigurasi postgresql.conf menggunakan editor teks nano. 15. Merestart container docker prak_tbd_modul_3_primary_tugas. 16. Menggunakan privilege superuser dan menjalankan serta masuk ke dalam shell bash container docker bernama prak_tbd_modul_3_primary_tugas. 17. Menampilkan semua data dari tabel pg_stat_replication; 18. Menampilkan semua data dalam tabel testugas1

	Failover manual dapat menjadi solusi darurat yang efektif dalam menjaga ketersediaan data, terutama saat terjadi gangguan pada server utama. Dengan melakukan failover secara manual, administrator memiliki kendali langsung untuk memastikan bahwa standby beralih menjadi server utama sehingga operasi dapat berlanjut tanpa waktu henti yang signifikan. Dalam skenario ini, failover memungkinkan akses data tetap tersedia meskipun primary mengalami gangguan, menjaga kelancaran aplikasi dan mencegah kehilangan data yang dapat berdampak negatif pada layanan..

Hasil Query/SQL:

```
ubuntu@prak-tbd:~$ sudo docker stop prak_tbd_modul_3_primary_tugas
prak_tbd_modul_3_primary_tugas
ubuntu@prak-tbd:~$ sudo docker exec -it prak_tbd_modul_3_standby_tugas bash
root@a62017e642d2:/# psql -U postgres -d prak_tbd_modul_3_db -c "SELECT
pg_promote();"
 pg_promote
------------
 t
(1 row)

root@a62017e642d2:/# psql -U postgres -c "SELECT pg_is_in_recovery();"
 pg_is_in_recovery
-------------------
 f
(1 row)

root@a62017e642d2:/# psql -U postgres -c "INSERT INTO testugas1 values(2, 'B');"
INSERT 0 1
root@a62017e642d2:/# psql -U postgres -c "SELECT * FROM testugas1;"
 id | name
----+------
  1 | ABC
  2 | B
(2 rows)

root@a62017e642d2:/#
exit
```

```
ubuntu@prak-tbd:~$ sudo docker start prak_tbd_modul_3_primary_tugas
prak_tbd_modul_3_primary_tugas
ubuntu@prak-tbd:~$ sudo docker exec -it prak_tbd_modul_3_primary_tugas bash
root@5e750c671bd0:/# psql -U postgres -c "SELECT * FROM testugas1;"
 id | name
----+------
  1 | ABC
(1 row)

root@5e750c671bd0:/#
exit
ubuntu@prak-tbd:~$ sudo docker exec -it prak_tbd_modul_3_standby_tugas bash
root@a62017e642d2:/# nano /var/lib/postgresql/data/postgresql.conf
root@a62017e642d2:/# nano /var/lib/postgresql/data/pg_hba.conf
root@a62017e642d2:/# nano /var/lib/postgresql/data/postgresql.conf
root@a62017e642d2:/#
exit
ubuntu@prak-tbd:~$ sudo docker restart prak_tbd_modul_3_standby_tugas
prak_tbd_modul_3_standby_tugas
ubuntu@prak-tbd:~$ sudo docker exec -it prak_tbd_modul_3_primary_tugas bash
root@5e750c671bd0:/# rm -rf /var/lib/postgresql/data/*
root@5e750c671bd0:/# pg_basebackup -h prak_tbd_modul_3_standby_tugas -D /var/lib/postgresql/data -U replicator_tugas -v -P --wal-method=stream
Password:
pg_basebackup: initiating base backup, waiting for checkpoint to complete
pg_basebackup: checkpoint completed
pg_basebackup: write-ahead log start point: 0/6000060 on timeline 2
pg_basebackup: starting background WAL receiver
pg_basebackup: created temporary replication slot "pg_basebackup_37"
30902/30902 kB (100%), 1/1 tablespace
pg_basebackup: write-ahead log end point: 0/6000158
pg_basebackup: waiting for background process to finish streaming ...
pg_basebackup: syncing data to disk ...
pg_basebackup: renaming backup_manifest.tmp to backup_manifest
pg_basebackup: base backup completed
```

```
ubuntu@prak-tbd: ~
  GNU nano 7.2                                  /var/lib/postgresql/data/postgresql.conf
# These options allow settings to be loaded from files other than the
# default postgresql.conf.  Note that these are directives, not variable
# assignments, so they can usefully be given more than once.

#include_dir = '...'                     # include files ending in '.conf' from
                                         # a directory, e.g., 'conf.d'
#include_if_exists = '...'               # include file only if it exists
#include = '...'                         # include file


#------------------------------------------------------------------------------
# CUSTOMIZED OPTIONS
#------------------------------------------------------------------------------

# Add settings for extensions here
wal_level = replica
archive_mode = on
archive_command = 'cp %p /var/lib/postgresql/data/archive/%f'
max_wal_senders = 3
wal_keep_size = 64
```

```
root@5e750c671bd0:/# ubuntu@prak-tbd:~$ sudo docker exec -it prak_tbd_modul_3_primary_tugas bash
root@5e750c671bd0:/# touch /var/lib/postgresql/data/standby.signal
root@5e750c671bd0:/# nano /var/lib/postgresql/data/postgresql.conf
root@5e750c671bd0:/# root@5e750c671bd0:/#
exit
```

```
ubuntu@prak-tbd: ~
  GNU nano 7.2                                  /var/lib/postgresql/data/postgresql.conf
#------------------------------------------------------------------------------
# CONFIG FILE INCLUDES
#------------------------------------------------------------------------------

# These options allow settings to be loaded from files other than the
# default postgresql.conf.  Note that these are directives, not variable
# assignments, so they can usefully be given more than once.

#include_dir = '...'                     # include files ending in '.conf' from
                                         # a directory, e.g., 'conf.d'
#include_if_exists = '...'               # include file only if it exists
#include = '...'                         # include file


#------------------------------------------------------------------------------
# CUSTOMIZED OPTIONS
#------------------------------------------------------------------------------

# Add settings for extensions here
primary_conninfo = 'host = prak_tbd_modul_3_primary_tugas port = 5432 user = replicator_tugas password = replica_password'
```

```
ubuntu@prak-tbd: ~

 pid | usesysid |     usename      | application_name | client_addr | client_hostname | client_port |         backend_start         | backend_xmin |   state
   | sent_lsn  | write_lsn | flush_lsn | replay_lsn | write_lag | flush_lag | replay_lag | sync_priority | sync_state |          reply_time
-----+----------+------------------+------------------+-------------+-----------------+-------------+-------------------------------+--------------+--------
---+-----------+-----------+-----------+-----------+-----------+-----------+-----------+---------------+------------+-------------------------------
  31 |    16389 | replicator_tugas | walreceiver      | 172.18.0.4  |                 |       32860 | 2024-11-04 16:29:04.024693+00 |              | streami
ng | 0/80000A0 | 0/80000A0 | 0/80000A0 | 0/80000A0  |           |           |           |             0 | async      | 2024-11-04 16:29:14.043289+00
(1 row)
```

```
ubuntu@prak-tbd:~$ sudo docker restart prak_tbd_modul_3_primary_tugas
prak_tbd_modul_3_primary_tugas
ubuntu@prak-tbd:~$ sudo docker exec -it prak_tbd_modul_3_primary_tugas bash
root@5e750c671bd0:/# psql -U postgres -c "SELECT * FROM pg_stat_replication;"

[1]+  Stopped                 psql -U postgres -c "SELECT * FROM pg_stat_replication;"
root@5e750c671bd0:/# psql -U postgres -c "SELECT * FROM testugas1;"
 id | name
----+------
  1 | ABC
  2 | B
(2 rows)
```