Nama : Reditya Nur Karomah
NIM : 245410077

1. 1. Jelaskan teorema CAP dan BASE dan keterkaitan keduanya. Jelaskan menggunakan contoh yang pernah anda gunakan.

   a. Teorema CAP menjelaskan bahwa dalam sistem terdistribusi, sebuah sistem tidak dapat sekaligus memenuhi Consistency, Availability, dan Partition Tolerance secara bersamaan. Artinya, ketika terjadi gangguan jaringan antar-node, sistem harus memilih apakah ingin tetap konsisten atau tetap tersedia. Sistem yang memprioritaskan konsistensi akan memastikan semua node memiliki data yang sama meskipun harus menolak beberapa permintaan, sedangkan sistem yang mengutamakan ketersediaan tetap merespons permintaan meskipun data yang diberikan belum tentu versi terbaru. Konsep CAP ini kemudian berkaitan dengan pendekatan BASE yang banyak digunakan oleh database NoSQL.

   b. Sementara BASE itu lebih ke pendekatan praktis yang sering dipakai ketika kita memilih ketersediaan lebih tinggi. Sistem yang saya pakai tadi akhirnya condong ke konsep BASE—server tetap melayani (basically available), datanya kadang terlihat belum sinkron (soft state), lalu beberapa detik kemudian baru menyamakan diri setelah replikasi berjalan (eventual consistency). Jadi, hubungan CAP dan BASE itu seperti teori dan praktik: CAP memberi batasan bahwa kita tidak bisa punya semuanya ketika jaringan bermasalah, sedangkan BASE adalah cara membangun sistem yang tetap “berjalan lancar” walaupun konsistensi sempurna baru tercapai beberapa saat kemudian.

   c. contohnya saya memakai dua aplikasi catatan di HP dan laptop yang tersinkron lewat internet. Pernah beberapa kali saya memperbarui catatan di HP ketika jaringan sedang jelek. Aplikasi tetap bisa dipakai (availability), tapi perubahan di HP tidak langsung muncul di laptop (consistency hilang karena “partition”, yaitu koneksi terputus). Setelah koneksi pulih, catatan di laptop akhirnya ikut berubah. Situasi ini menggambarkan konsep BASE, yaitu konsistensi yang datang belakangan, dan sekaligus menunjukkan bagaimana CAP bekerja: sistem memilih tetap tersedia

meskipun konsistensi sementara hilang. Dari pengalaman itu, terlihat jelas bahwa CAP dan BASE saling berkaitan—ketika sistem memilih tetap berjalan meskipun terjadi gangguan jaringan, maka otomatis sistem menerapkan prinsip BASE sebagai cara menjaga pengalaman pengguna tetap lancar.

2. Jelaskan keterkaitan antara GraphQL dengan komunikasi antar proses pada sistem terdistribusi. Buat diagramnya. GraphQL dapat berperan sebagai penghubung komunikasi antar proses dalam sebuah sistem terdistribusi karena ia bertindak sebagai pintu masuk tunggal yang menerima permintaan dari klien dan meneruskannya ke layanan-layanan lain di belakangnya. Dalam sistem yang terdiri dari banyak microservice, setiap layanan biasanya memiliki fungsi dan data sendiri-sendiri, sehingga klien perlu berkomunikasi dengan beberapa layanan sekaligus. Jika menggunakan REST biasa, klien harus memanggil banyak endpoint yang berbeda, sedangkan dengan GraphQL, klien cukup mengirim satu query dan GraphQL akan menyalurkannya ke layanan yang relevan. Proses seperti ini membuat komunikasi antar proses di backend menjadi lebih teratur karena semua panggilan dari klien hanya melewati satu titik koordinasi. GraphQL kemudian meneruskan permintaan itu ke service lain seperti Auth, User, Post, atau Comment sesuai bagian data yang diminta klien. Dari situ terlihat bahwa GraphQL bukan hanya alat query data, tetapi juga menjadi jembatan komunikasi antar proses dalam sistem terdistribusi yang terdiri dari banyak microservice.

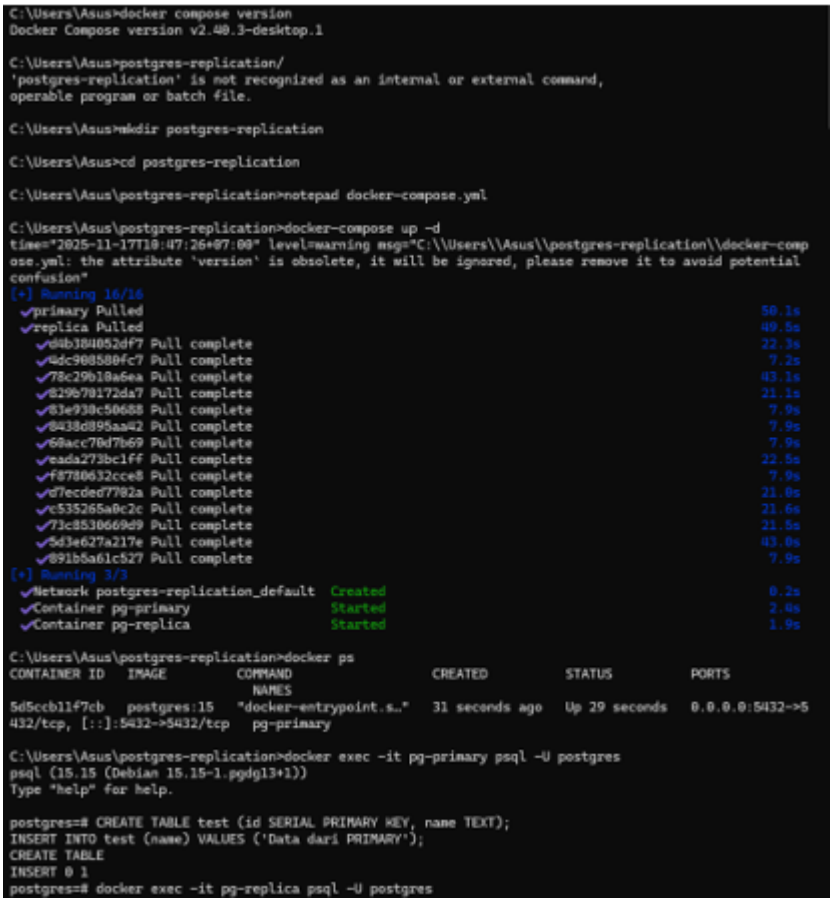
   diagramnya ada di github

3.

pembahasan :

Pertama-tama dibuat folder utama bernama *postgres-repl* dengan perintah mkdir, lalu setelah masuk ke dalamnya dibuat dua folder lagi yaitu *primary* dan *replica*. Kedua folder ini nantinya dipakai sebagai tempat penyimpanan konfigurasi dan data untuk server utama dan server cadangan. Setelah folder selesai dibuat, tampak perintah notepad docker-compose.yml dijalankan untuk mulai membuat file konfigurasi docker compose. File inilah yang nanti mengatur bagaimana container master dan slave dijalankan.

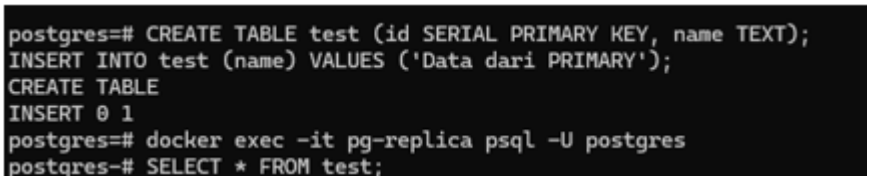
Setelah isi file disimpan, pengguna melakukan pengecekan isi directory menggunakan dir, dan di situ terlihat file docker-compose.yml sudah muncul dengan ukuran sekitar 972 bytes. Langkah-langkah di gambar ini menunjukkan persiapan awal sebelum container dijalankan. Semuanya dilakukan dari terminal Windows, mulai dari membuat struktur folder sampai menyiapkan file compose yang menjadi inti dari proses replikasi PostgreSQL nanti.



pembahasan :

proses menjalankan Docker Compose setelah file konfigurasi selesai dibuat. Awalnya terminal menampilkan pesan error karena perintah docker belum dikenali, tetapi setelah itu perintah dijalankan ulang dari direktori yang benar. Ketika docker-compose up -d dijalankan, Docker mulai menarik (pull) image PostgreSQL dari repository. Proses ini memakan waktu karena ada beberapa layer image yang harus didownload satu per satu.

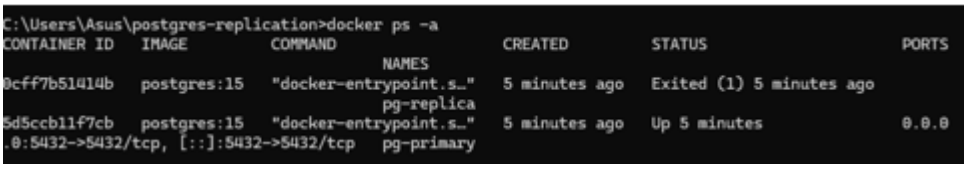
Setelah semua image berhasil didownload, kedua container—yaitu *primary* dan *replica*—langsung dibuat dan dijalankan. Pada bagian bawah tampak perintah docker ps -a untuk memastikan bahwa semua container berjalan. Hasilnya, kedua container tampil dengan status *Up*, yang berarti server utama dan server replikasinya sudah aktif. Proses ini menunjukkan bahwa konfigurasi docker-compose yang dibuat sebelumnya sudah benar dan berhasil dijalankan oleh Docker tanpa error.



pembahasan :

Gambar ini menampilkan tahap pengujian replikasi. Pertama, pengguna masuk ke dalam container *primary* dan membuka PostgreSQL dengan perintah psql -U postgres. Di dalam database utama tersebut dibuat tabel baru bernama test dengan dua kolom, yaitu id dan name. Setelah tabel selesai dibuat, satu baris data dimasukkan menggunakan perintah INSERT.

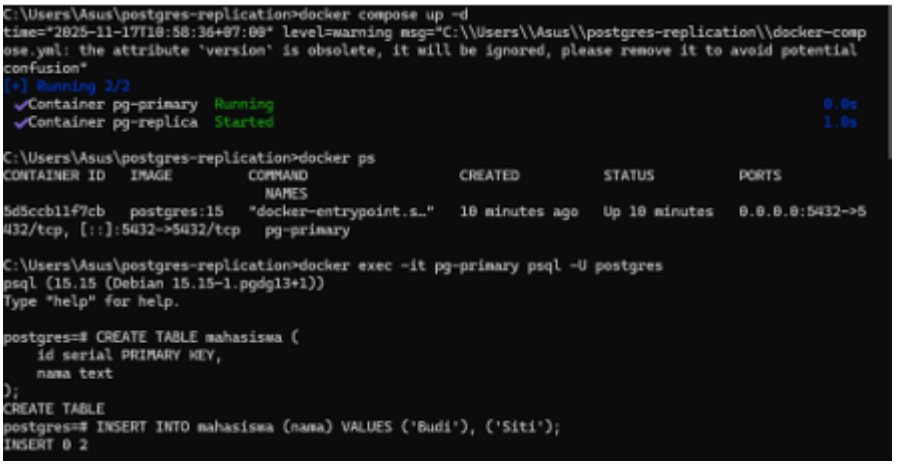
Setelah melakukan operasi pada primary, pengguna keluar dan masuk ke container *replica* menggunakan perintah docker exec -it pg-replica psql -U postgres. Di dalam server replica ini dijalankan query SELECT * FROM test; untuk melihat apakah data yang dimasukkan sebelumnya sudah tersalin. Jika data muncul di sini, berarti mekanisme streaming replication berhasil berjalan. Gambar ini menunjukkan inti dari proses sinkronisasi database, yaitu memastikan perubahan yang dilakukan di primary otomatis diteruskan ke replica tanpa harus melakukan input ulang.



pemabhasan :

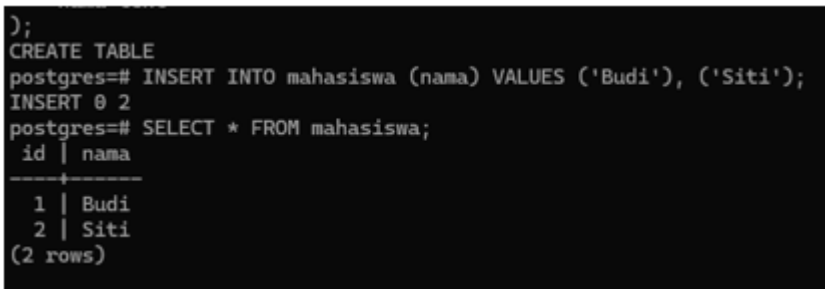
Gambar terakhir memperlihatkan status container setelah semua proses replikasi diuji. Perintah docker ps digunakan untuk melihat container mana saja yang sedang aktif. Pada output terlihat dua container: *pg-replica* dan *primary*. Keduanya menampilkan status *Up*, menandakan bahwa sistem sudah stabil dan kedua server berjalan normal. Selain itu, terlihat juga waktu aktif (Up 5 minutes), yang menunjukkan bahwa container tetap berjalan setelah proses pengetesan replikasi dilakukan. Informasi port juga terlihat jelas; primary berjalan di port standar PostgreSQL 5432, sementara replica menggunakan port berbeda agar tidak bentrok. Bagian ini menjadi penutup bahwa seluruh rangkaian konfigurasi—mulai dari pembuatan folder, penulisan

docker-compose, menjalankan container, hingga pengujian replikasi—semuanya berhasil dijalankan dengan baik.



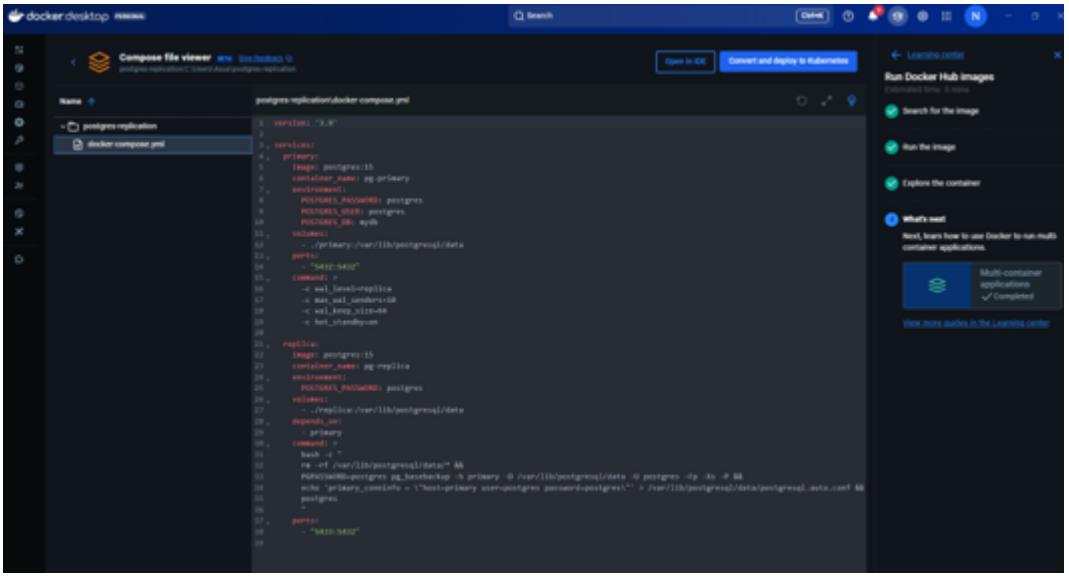
pembahasan :

kemudian buat tablenya



pembahasan :

Menampilkan tablenya


docker desktop
Search
Ctrl+K
Compose file viewer
Open in IDE
Convert and deploy to Kubernetes
Name
postgres-replication
docker-compose.yml
postgres-replication\docker-compose.yml
version: "3.9"
services:
primary:
image: postgres:15
container_name: pg-primary
environment:
POSTGRES_PASSWORD: postgres
POSTGRES_USER: postgres
POSTGRES_DB: mydb
volumes:
- ./primary:/var/lib/postgresql/data
ports:
- "5432:5432"
command: >
-c wal_level=replica
-c max_wal_senders=10
-c wal_keep_size=64
-c hot_standby=on
replica:
image: postgres:15
container_name: pg-replica
environment:
POSTGRES_PASSWORD: postgres
volumes:
- ./replica:/var/lib/postgresql/data
depends_on:
- primary
command: >
bash -c "
rm -rf /var/lib/postgresql/data/* &&
PGPASSWORD=postgres pg_basebackup -h primary -D /var/lib/postgresql/data -U postgres -Fp -Xs -P &&
echo 'primary_conninfo = \"host=primary user=postgres password=postgres\"' > /var/lib/postgresql/data/postgresql.auto.conf &&
postgres
"
ports:
- "5433:5432"
Learning center
Run Docker Hub images
Search for the image
Run the image
Explore the container
What's next
Next, learn how to use Docker to run multi-container applications.
Multi-container applications
Completed
View more guides in the Learning center