

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.

```

PS C:\Users\Asus> mkdir postgres-repl

Directory: C:\Users\Asus

Mode                LastWriteTime         Length Name
----                -----          ----  --
d----       11/17/2025 11:07 AM           0    postgres-repl

PS C:\Users\Asus> cd postgres-repl
PS C:\Users\Asus\postgres-repl> mkdir primary

Directory: C:\Users\Asus\postgres-repl

Mode                LastWriteTime         Length Name
----                -----          ----  --
d----       11/17/2025 11:08 AM           0    primary

PS C:\Users\Asus\postgres-repl> mkdir replica

Directory: C:\Users\Asus\postgres-repl

Mode                LastWriteTime         Length Name
----                -----          ----  --
d----       11/17/2025 11:08 AM           0    replica

PS C:\Users\Asus\postgres-repl> notepad docker-compose.yml
PS C:\Users\Asus\postgres-repl> docker compose up -d
no configuration file provided: not found
PS C:\Users\Asus\postgres-repl> docker compose up -d
no configuration file provided: not found
PS C:\Users\Asus\postgres-repl> dir *.yml

Directory: C:\Users\Asus\postgres-repl

Mode                LastWriteTime         Length Name
----                -----          ----  --
-a---       11/17/2025 11:08 AM        572  docker-compose.yml

```

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.

```

C:\Users\Asus>docker compose version
Docker Compose version v2.40.3-desktop.1

C:\Users\Asus>postres-replication/
'postgres-replication' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\Asus>mkdir postgres-replication
C:\Users\Asus>cd postgres-replication
C:\Users\Asus\postgres-replication>notepad docker-compose.yml

C:\Users\Asus\postgres-replication>docker-compose up -d
time="2025-11-17T10:47:26+07:00" level=warning msg="C:\\\\Users\\\\Asus\\\\postgres-replication\\\\docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 16/16
  ✓ primary Pulled
  ✓ replica Pulled
    ✓ e0bb384052df7 Pull complete   50.1s
    ✓ 4dc908589fc7 Pull complete   49.5s
    ✓ 78c29b18a6ea Pull complete   22.3s
    ✓ 829b78172da7 Pull complete   7.2s
    ✓ 81e930c506688 Pull complete   43.1s
    ✓ 8438d895aa42 Pull complete   21.1s
    ✓ 69acc70d7b69 Pull complete   7.9s
    ✓ eada273bc1ff Pull complete   7.9s
    ✓ f7806532cce8 Pull complete   7.9s
    ✓ fd7ccedf7782a Pull complete   22.5s
    ✓ c535265a9c2c Pull complete   7.9s
    ✓ 73c8530669d9 Pull complete   21.6s
    ✓ 5d3e627a217e Pull complete   21.5s
    ✓ 691bba1c1c527 Pull complete   43.0s
    ✓ 691bba1c1c527 Pull complete   7.9s
[+] Running 3/3
  ✓ Network postgres-replication_default Created          0.2s
  ✓ Container pg-primary   Started          2.4s
  ✓ Container pg-replica  Started          1.9s

C:\Users\Asus\postgres-replication>docker ps
CONTAINER ID IMAGE           COMMAND      CREATED        STATUS        PORTS
                NAMES
56fc0b1f47cb  postgres:15 "docker-entrypoint.s..." 31 seconds ago Up  29 seconds  0.0.0.0:5432->5432/tcp, [::]:5432->5432/tcp pg-primary

C:\Users\Asus\postgres-replication>docker exec -it pg-primary psql -U postgres
psql (15.15 (Debian 15.15-1.pgdg13+1))
Type "help" for help.

postgres=# CREATE TABLE test (id SERIAL PRIMARY KEY, name TEXT);
INSERT INTO test (name) VALUES ('Data dari PRIMARY');
CREATE TABLE
INSERT 0 1
postgres=# docker exec -it pg-replica psql -U postgres

```

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.

```
postgres=# CREATE TABLE test (id SERIAL PRIMARY KEY, name TEXT);
INSERT INTO test (name) VALUES ('Data dari PRIMARY');
CREATE TABLE
INSERT 0 1
postgres=# docker exec -it pg-replica psql -U postgres
postgres# SELECT * FROM test;
```

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.

```
C:\Users\Asus\postgres-replication>docker ps -a
CONTAINER ID        IMAGE       COMMAND             CREATED          STATUS           PORTS
 NAMES
0cff7b51414b      postgres:15   "docker-entrypoint.s..."  5 minutes ago   Exited (1) 5 minutes ago
                                         pg-replica
5d5ccb11f7cb      postgres:15   "docker-entrypoint.s..."  5 minutes ago   Up 5 minutes          0.0.0
                                         .0:5432->5432/tcp, [::]:5432->5432/tcp pg-primary
```

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.

```
C:\Users\Asus\postgres-replication>docker compose up -d
time="2025-11-17T10:58:36+07:00" level=warning msg="C:\Users\Asus\postgres-replication\docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion"
[*] Running 2/2
  ✓ Container pg-primary  Running
  ✓ Container pg-replica  Started

C:\Users\Asus\postgres-replication>docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS
5d5ccb11f7cb      postgres:15      "docker-entrypoint.s..."   10 minutes ago   Up 10 minutes   0.0.0.0:5432->5432/tcp, [::]:5432->5432/tcp
pg-primary

C:\Users\Asus\postgres-replication>docker exec -it pg-primary psql -U postgres
psql (15.15 (Debian 15.15-1.pgdg13+1))
Type "help" for help.

postgres=# CREATE TABLE mahasiswa (
    id serial PRIMARY KEY,
    nama text
);
CREATE TABLE
postgres=# INSERT INTO mahasiswa (nama) VALUES ('Budi'), ('Siti');
INSERT 0 2
postgres=# SELECT * FROM mahasiswa;
 id | nama
----+
  1 | Budi
  2 | Siti
(2 rows)
```

pembahasan :

kemudian buat tablenya

```
);

CREATE TABLE
postgres=# INSERT INTO mahasiswa (nama) VALUES ('Budi'), ('Siti');
INSERT 0 2
postgres=# SELECT * FROM mahasiswa;
 id | nama
----+
  1 | Budi
  2 | Siti
(2 rows)
```

pembahasan :

Menampilkan tablenya

The screenshot shows the Docker Desktop interface with the 'Compose file viewer' tab selected. The left sidebar shows a tree view with 'postgres replication' expanded, and 'docker-compose.yml' selected. The main pane displays the contents of the docker-compose.yml file:

```
version: "3.9"
services:
  primary:
    image: postgres:13
    container_name: pg-primary
    environment:
      POSTGRES_PASSWORD: postgres
      POSTGRES_USER: postgres
      POSTGRES_DB: pgdb
    volumes:
      - ./primary:/var/lib/postgresql/data
    ports:
      - "5432:5432"
    command:
      - -wal_level=replica
      - -max_wal_size=1G
      - -max_main_wal_size=1G
      - -max_reorg_wal_size=1G
      - -hot_standby=on
  replica:
    image: postgres:13
    container_name: pg-replica
    environment:
      POSTGRES_PASSWORD: postgres
    volumes:
      - ./replica:/var/lib/postgresql/data
    depends_on:
      - primary
    command:
      - hash -r
      - rm -rf /var/lib/postgresql/data/*
      - pg_basebackup -h primary -D /var/lib/postgresql/data -U postgres -F -R -P
      - echo "primary_conninfo = 'host=primary user=postgres password=postgres'" > /var/lib/postgresql/data/postgresql.auto.conf
    ports:
      - "5432:5432"
```

On the right side, there are buttons for 'Open in VSCode', 'Convert and deploy to Kubernetes', and a 'Learn more' section titled 'Run Docker Hub images' with links to 'Search for the image', 'Run the image', 'Explore the container', and 'What's next'. A note says 'Next, learn how to use Docker to run multi-container applications.'