

Nama: Farhan Riyandi

Data Engineer Batch 4

1. Sebutkan perbedaan antara data warehouse dan data lake

Jawab:

Data Warehouse

- Data Warehouse adalah tempat penyimpanan untuk data terstruktur dan terfilter yang telah diproses untuk tujuan tertentu seperti digunakan untuk laporan keuangan, analisis penjualan dan lain-lain.
- Data disimpan dalam format terstruktur, biasanya dalam tabel dengan baris dan kolom. Data harus sesuai dengan skema yang telah didefinisikan (schema-on-write) sebelum dimuat ke dalam data warehouse. Terutama data terstruktur dan semi-terstruktur yang telah diproses sebelumnya.
- Biasanya menggunakan ETL (Extract, Transform, Load). Data diekstrak dari sumber, kemudian diubah (dibersihkan dan diatur) sebelum dimuat ke dalam data warehouse.

Data Lake

- Data Lake cenderung mencakup sejumlah besar data mentah, yang tujuannya mungkin belum ditentukan
- Mampu menyimpan data mentah tanpa memerlukan skema tetap sebelum penyimpanan (schema-on-read). Ini memungkinkan penyimpanan berbagai jenis data, termasuk terstruktur, semi-terstruktur, dan tidak terstruktur.
- Menggunakan ELT (Extract, Load, Transform). Data diekstrak dan dimuat ke dalam data lake dalam bentuk aslinya, dan transformasi dilakukan saat data diakses atau diproses.

2. Apa yang membedakan teknologi database untuk data warehouse (OLAP) dari teknologi database konvensional (OLTP)

Jawab:

Tujuan

OLAP

Tujuan utama pemrosesan analitik online (OLAP) adalah untuk menganalisis data agregat. Sistem OLAP untuk menghasilkan laporan, melakukan analisis data yang kompleks, dan mengidentifikasi tren.

OLTP

Tujuan utama pemrosesan transaksi online (OLTP) adalah untuk memproses transaksi basis data. Sistem OLTP untuk memproses pesanan, memperbarui inventaris, dan mengelola akun pelanggan

Pemformatan Data

OLAP

OLAP menggunakan model data multidimensional, sehingga dapat terlihat data yang sama dari sudut yang berbeda. Basis data OLAP menyimpan data dalam format kubus, di mana setiap dimensi mewakili atribut data yang berbeda. Setiap sel dalam kubus mewakili nilai atau ukuran untuk persimpangan dimensi.

OLTP

Sistem OLTP memiliki satu dimensi dan berfokus pada satu aspek data. Sistem ini menggunakan basis data relasional untuk menyusun data ke dalam tabel. Setiap baris dalam tabel mewakili instans entitas, dan setiap kolom mewakili atribut entitas.

Arsitektur Data

OLAP

Arsitektur basis data OLAP lebih memprioritaskan operasi pembacaan data daripada penulisan data. Anda dapat dengan cepat dan efisien melakukan kueri kompleks pada volume data yang besar. Ketersediaan merupakan masalah dengan prioritas rendah karena kasus penggunaan utamanya adalah analitik.

OLTP

Arsitektur basis data OLTP memprioritaskan operasi penulisan data. Ini dioptimalkan untuk beban kerja berat tulis dan dapat memperbarui data transaksional berfrekuensi tinggi dan bervolume tinggi tanpa mengorbankan integritas data.

Performa

OLAP

Waktu pemrosesan OLAP dapat bervariasi mulai dari beberapa menit hingga beberapa jam, tergantung jenis dan volume data yang dianalisis. Untuk memperbarui basis data OLAP, secara berkala, Memproses data dalam batch besar lalu mengunggah batch tersebut ke sistem sekaligus. Frekuensi pembaruan data juga bervariasi antara sistem satu dengan lainnya, mulai dari harian hingga mingguan atau bahkan bulanan.

OLTP

Mengukur waktu pemrosesan OLTP dalam milidetik atau kurang. Basis data OLTP mengelola pembaruan basis data secara waktu nyata. Pembaruan dilakukan dengan cepat, singkat, dan dipicu oleh pengguna. Pemrosesan aliran sering kali digunakan selama pemrosesan batch.

3. Teknologi apa saja yang biasanya dipakai untuk data warehouse?

Jawab:

- AWS Redshift
- Google Big Query
- Clickhouse
- Snowflake
- Databricks
- Apache Doris
- Postgre (with Citus Extension)

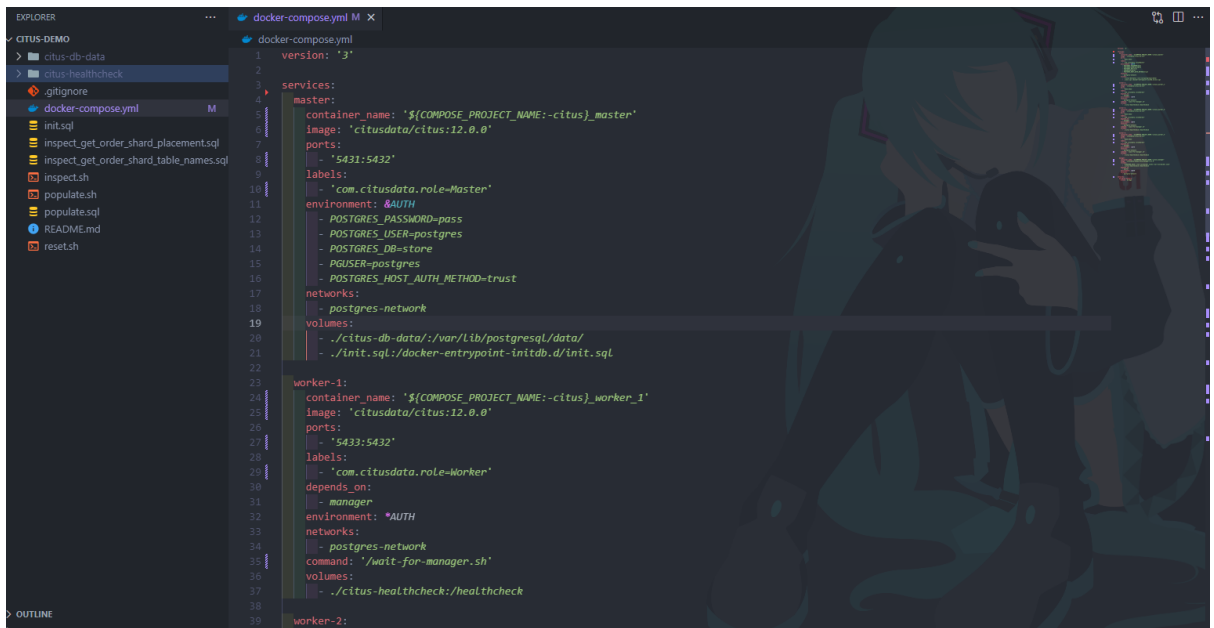
4. Tuliskan setiap perintah dari proses instalasi citus menggunakan docker compose sampai tabel terbentuk, berikan juga tangkapan layar untuk setiap langkah dan hasilnya!

Jawab:

Pertama git clone terlebih dahulu dari link berikut:

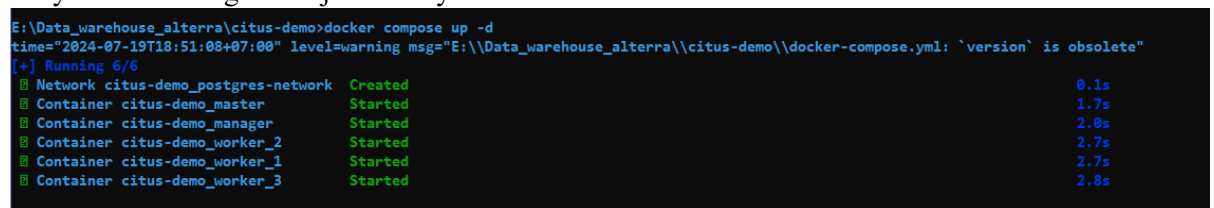
<https://github.com/Immersive-DataEngineer-Resource/citus-demo.git>

Kedua pada github tersebut terdapat file docker-compose.yml yang sudah didefinisikan untuk menginstall citus



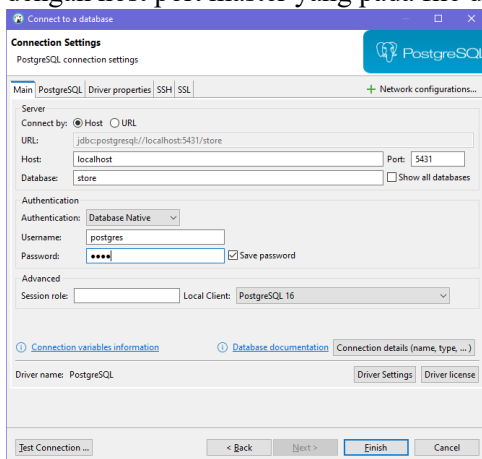
```
1 version: '3'
2
3 services:
4   master:
5     container_name: '${COMPOSE_PROJECT_NAME}-citus_master'
6     image: 'citusdata/citus:12.0.0'
7     ports:
8       - '5431:5432'
9     labels:
10      - 'com.citusdata.role=Master'
11     environment: &AUTH
12      - POSTGRES_PASSWORD=pass
13      - POSTGRES_USER=postgres
14      - POSTGRES_DB=store
15      - PGUSER=postgres
16      - POSTGRES_HOST_AUTH_METHOD=trust
17     networks:
18      - postgres-network
19     volumes:
20      - ./citus-db-data:/var/lib/postgresql/data/
21      - ./init.sql:/docker-entrypoint-initdb.d/init.sql
22
23   worker-1:
24     container_name: '${COMPOSE_PROJECT_NAME}-citus_worker_1'
25     image: 'citusdata/citus:12.0.0'
26     ports:
27       - '5433:5432'
28     labels:
29      - 'com.citusdata.role=Worker'
30     depends_on:
31      - manager
32     environment: *AUTH
33     networks:
34      - postgres-network
35     command: '/wait-for-manager.sh'
36     volumes:
37      - ./citus-healthcheck:/healthcheck
38
39   worker-2:
```

kemudian pada command prompt sesuaikan direktori yang sudah diclone tersebut lalu jalankan docker compose up -d yang digunakan untuk membuat container yang sudah didefinisikan pada file yml dan sekaligus menjalankannya.



```
E:\Data_warehouse_alterra\citus-demo>docker compose up -d
time="2024-07-19T18:51:08+07:00" level=warning msg="E:\\Data_warehouse_alterra\\citus-demo\\docker-compose.yml: `version` is obsolete"
[+] Running 6/6
  Network citus-demo_postgres-network Created                                0.1s
  Container citus-demo_master           Started                             1.7s
  Container citus-demo_manager          Started                             2.0s
  Container citus-demo_worker_2         Started                             2.7s
  Container citus-demo_worker_1         Started                             2.7s
  Container citus-demo_worker_3         Started                             2.8s
```

Kemudian buka dbeaver Buat database dengan nama store atau lainnya sesuaikan host port dengan host port master yang pada file docker-compose.yml seperti di bawah ini.



Connect to a database

PostgreSQL connection settings

Main | PostgreSQL | Driver properties | SSH | SSL

Server

Connect by: ☒ Host ☐ URL

URL: jdbc:postgresql://localhost:5431/store

Host: localhost Port: 5431

Database: store ☐ Show all databases

Authentication

Authentication: Database Native

Username: postgres

Password: Save password

Advanced

Session role: Local Client: PostgreSQL 16

[Connection variables information](#) [Database documentation](#) [Connection details \(name, type, ...\)](#)

Driver name: PostgreSQL Driver Settings Driver license

Kemudian untuk membuat table dengan syntax sql berikut:

```

CREATE TABLE events_columnar (
  device_id bigint,
  event_id bigserial,
  event_time timestamptz default now(),
  data jsonb not null
)
USING columnar;
--insert some data
INSERT INTO events_columnar (device_id, data)
SELECT d, '{"hello": "columnar"}' FROM generate_series (1, 100) d;

--create a row-based table to compare
CREATE TABLE events_row AS SELECT * FROM events_columnar;

```

Berikut adalah table yang terbentuk
Tabel events_columnar

	123 device_id	123 event_id	event_time	data
1	1	1	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
2	2	2	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
3	3	3	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
4	4	4	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
5	5	5	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
6	6	6	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
7	7	7	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
8	8	8	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
9	9	9	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
10	10	10	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
11	11	11	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
12	12	12	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
13	13	13	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
14	14	14	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
15	15	15	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
16	16	16	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}

Tabel events_row

	123 device_id ▼	123 event_id ▼	🕒 event_time ▼	📄 data ▼
1	1	1	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
2	2	2	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
3	3	3	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
4	4	4	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
5	5	5	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
6	6	6	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
7	7	7	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
8	8	8	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
9	9	9	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
10	10	10	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
11	11	11	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
12	12	12	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
13	13	13	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
14	14	14	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}
15	15	15	2024-07-19 14:43:14.902 +0700	{"hello": "columnar"}

5. Jelaskan perbedaan antara access method heap dan columnar citus!

Jawab:

Method heap

- Heap adalah metode akses di mana data disimpan dalam format baris (row-based storage)
- Waktu Akses yang Lebih Lambat untuk Query Analitik: Ketika hanya sebagian kolom yang diperlukan, metode heap mungkin memerlukan pembacaan seluruh baris, yang dapat memperlambat query analitik.
- Kompresi yang Kurang Efisien: Data dalam baris cenderung memiliki pola yang lebih bervariasi dibandingkan data dalam kolom, sehingga kompresi data kurang efektif.
- Waktu Penulisan yang Lebih Cepat: Karena data disimpan dalam baris yang teratur, penambahan dan penghapusan baris dapat dilakukan dengan cepat dan efisien.

Metode Columnar

- Columnar metode akses di mana data disimpan dalam format kolom (column-based storage)
- Waktu Akses yang Lebih Baik: Columnar storage sangat efisien untuk query analitik karena hanya membaca kolom yang diperlukan, bukan seluruh baris, sehingga mempercepat waktu akses data.
- Kompresi: Data dalam kolom seringkali memiliki pola yang sama, sehingga memungkinkan kompresi yang lebih efektif dan mengurangi ruang penyimpanan.
- Waktu Penulisan yang Lebih Lambat: Penulisan data dalam format kolom bisa lebih lambat dibandingkan dengan format baris karena penulisan memerlukan pembaruan di beberapa tempat yang tersebar.