

# DATA WAREHOUSE

## PART 1 - Introduction to Data Warehouse

### TASK

1. Sebutkan perbedaan antara data warehouse dan data lake!

Jawab:

Data Warehouse:

- Struktur: Terstruktur dan terorganisir, biasanya dalam bentuk tabel-tabel yang saling berhubungan.
- Skema: Skema tetap (schema-on-write), di mana data diubah dan diorganisir sebelum disimpan.
- Penggunaan: Digunakan untuk analisis bisnis dan pelaporan, cocok untuk query yang kompleks.
- Jenis Data: Biasanya hanya menyimpan data yang terstruktur (misalnya, data transaksi).
- Kecepatan Akses: Dirancang untuk kecepatan query yang tinggi pada sejumlah besar data.

Data Lake:

- Struktur: Tidak terstruktur atau semi-terstruktur, dapat menyimpan data dalam berbagai format seperti JSON, video, audio, CSV, dll.
- Skema: Skema fleksibel (schema-on-read), di mana data diorganisir pada saat pengambilan.
- Penggunaan: Digunakan untuk penyimpanan data mentah dan analisis lanjutan seperti machine learning.
- Jenis Data: Menyimpan data terstruktur, semi-terstruktur, dan tidak terstruktur.
- Kecepatan Akses: Dirancang untuk menyimpan sejumlah besar data dengan biaya rendah, meskipun waktu pengambilan data mungkin lebih lama dibandingkan dengan data warehouse.

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

Jawab:

OLAP (Online Analytical Processing):

- Fokus: Analisis dan pelaporan data.
- Query: Mendukung query yang kompleks dan analisis multidimensi.
- Desain Skema: Skema bintang atau skema salju (star schema atau snowflake schema).
- Contoh Penggunaan: Analisis data historis, pelaporan, BI (Business Intelligence).
- Transaksi: Rendah, sering kali read-intensive.
- Contoh Database: Amazon Redshift, Google BigQuery, Snowflake.

OLTP (Online Transaction Processing):

- Fokus: Pemrosesan transaksi secara real-time.
- Query: Query sederhana dan cepat, biasanya CRUD (Create, Read, Update, Delete).
- Desain Skema: Skema normalisasi untuk mengurangi redundansi data.
- Contoh Penggunaan: Sistem transaksi bisnis sehari-hari seperti perbankan, e-commerce.
- Transaksi: Tinggi, read dan write-intensive.
- Contoh Database: MySQL, PostgreSQL, Oracle Database.

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

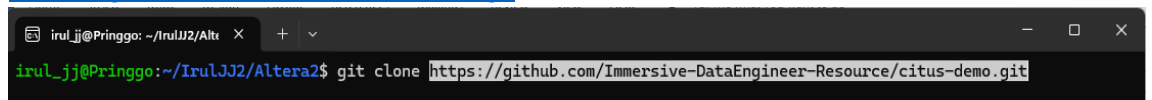
Jawab:

- AWS Redshift
- Google Big Query
- Snowflake
- Clickhouse
- Teradata
- Oracle Exadata

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

Jawab:

1. Git clone terlebih dahulu menggunakan link : <https://github.com/Immersive-DataEngineer-Resource/citus-demo.git>



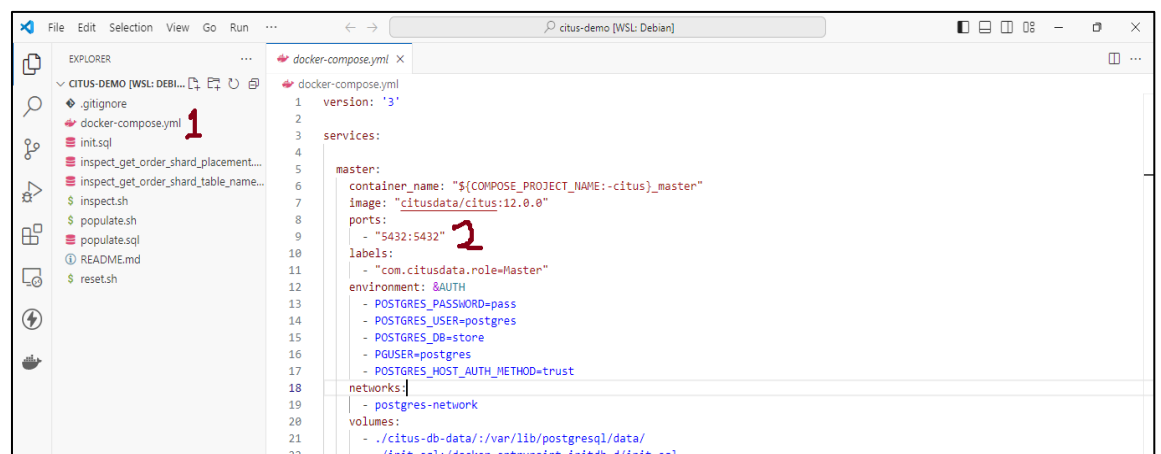
```
irul_jj@Pringgo: ~/IrulJJ2/Altera2$ git clone https://github.com/Immersive-DataEngineer-Resource/citus-demo.git
```

2. Bukalah repository tersebut ke vs code dengan command **cd citus-demo/** dan **code .**



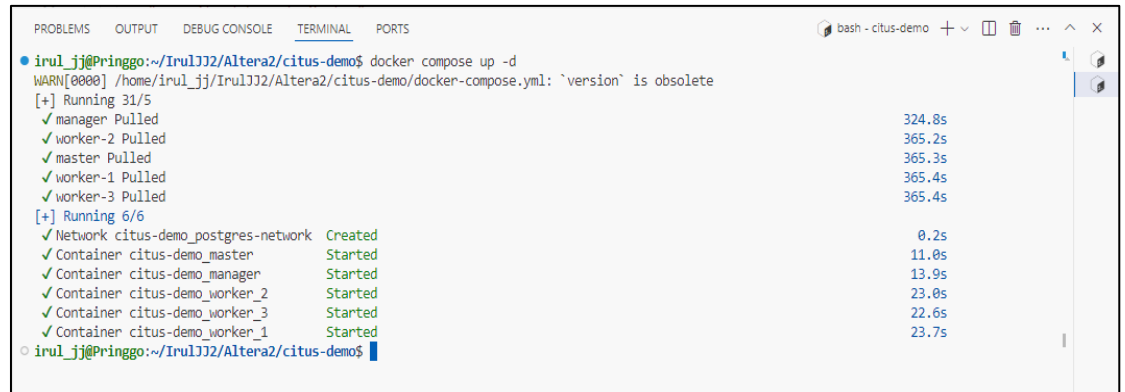
```
irul_jj@Pringgo: ~/IrulJJ2/Altera2$ cd citus-demo/
irul_jj@Pringgo: ~/IrulJJ2/Altera2/citus-demo$ code .
irul_jj@Pringgo: ~/IrulJJ2/Altera2/citus-demo$
```

3. Pilih **docker-compose.yml** dilanjutkan mengubah **"5432:5432"** menjadi **"5431:5432"**



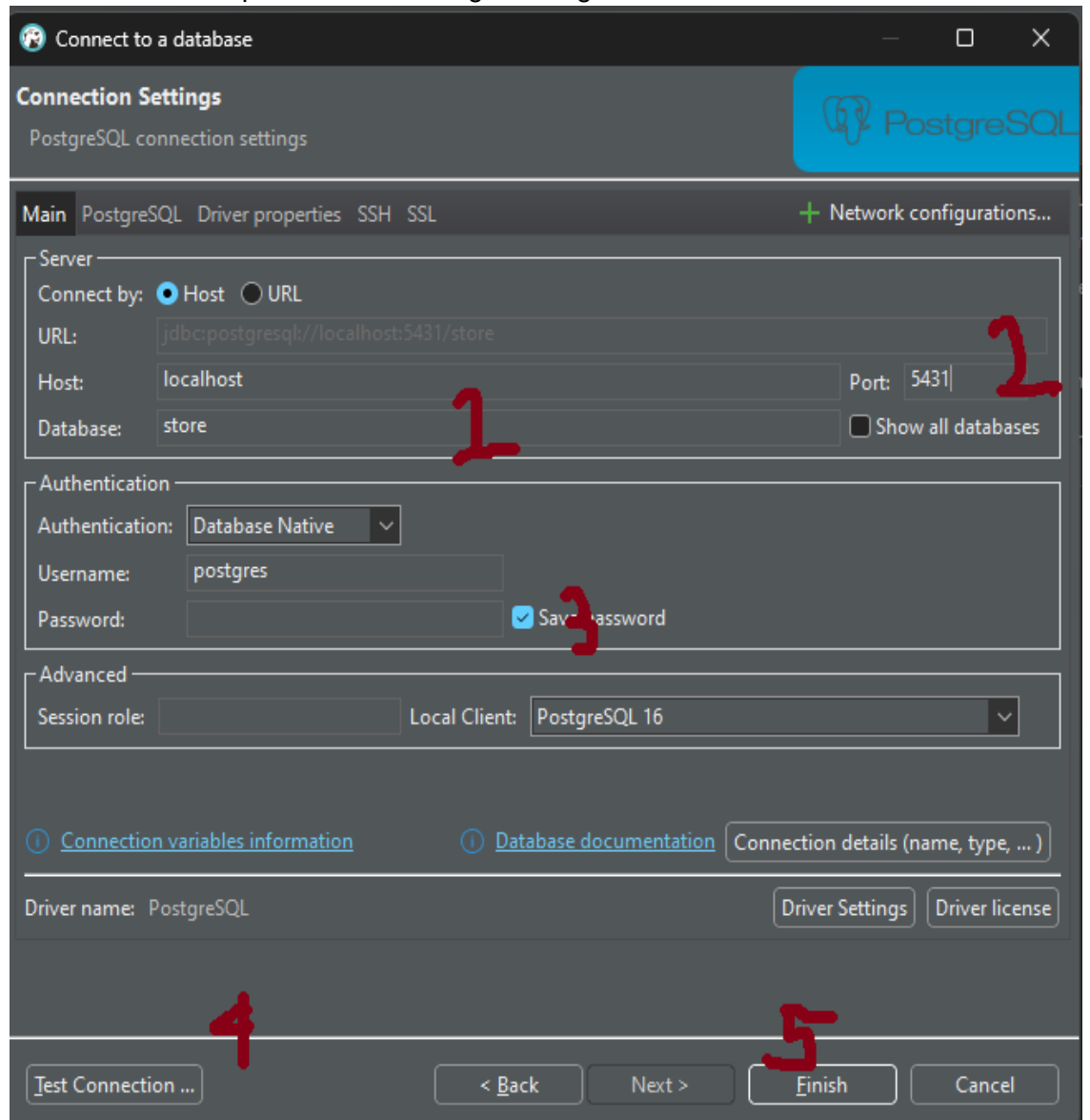
```
1 version: '3'
2
3 services:
4
5   master:
6     container_name: "${COMPOSE_PROJECT_NAME:-citus}_master"
7     image: "citusdata/citus:12.0.0"
8     ports:
9       - "5432:5432"
10    labels:
11      - "com.citusdata.role=Master"
12    environment: &AUTH
13      - POSTGRES_PASSWORD=pass
14      - POSTGRES_USER=postgres
15      - POSTGRES_DB=store
16      - PGUSER=postgres
17      - POSTGRES_HOST_AUTH_METHOD=trust
18
19    networks:
20      - postgres-network
21
22    volumes:
23      - ./citus-db-data:/var/lib/postgresql/data/
24      - ./init.sql:/docker-entrypoint-initdb.d/init.sql
```

4. Jalankan command **docker compose up -d** pada terminal di vs code



```
irul_jj@Pringo:~/Iru1J2/Alter2/citus-demo$ docker compose up -d
WARN[0000] /home/irul_jj/Iru1J2/Alter2/citus-demo/docker-compose.yml: `version` is obsolete
[+] Running 31/5
✓ manager Pulled                                324.8s
✓ worker-2 Pulled                                365.2s
✓ master Pulled                                  365.3s
✓ worker-1 Pulled                                365.4s
✓ worker-3 Pulled                                365.4s
[+] Running 6/6
✓ Network citus-demo_postgres-network Created      0.2s
✓ Container citus-demo_master Started             11.0s
✓ Container citus-demo_manager Started             13.9s
✓ Container citus-demo_worker_2 Started            23.0s
✓ Container citus-demo_worker_3 Started            22.6s
✓ Container citus-demo_worker_1 Started            23.7s
irul_jj@Pringo:~/Iru1J2/Alter2/citus-demo$
```

5. Buat koneksi baru pada dbeaver dengan mengikuti urutan:



- Membuat database **store**
- Mengubah port menjadi **5431**
- Masukkan Password
- Lakukan Test Connection
- Pilih Finish

```

create table events_columnar (
  device_id bigint,
  event_id bigserial,
  event_time timestamptz default now (),
  data jsonb not null
)
using columnar;

```

6.

Masukkan query tersebut. Lalu table berhasil dibuat. Hasilnya seperti ini:

The screenshot shows a database management tool interface. At the top, there's a search bar with the text 'events\_columnar'. Below it, a table is displayed with the following columns: 'device\_id', 'event\_id', 'event\_time', and 'data'. The table is currently empty, showing only the column headers. The interface includes a sidebar on the left with icons for 'Grid', 'Text', and 'Record' views. The 'Grid' view is selected, showing the table structure. The 'event\_time' column has a clock icon, and the 'data' column has a document icon.

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

Jawab:

Heap:

- Menyimpan data dalam baris (row-oriented).
- Efisien untuk operasi insert, update, dan delete.
- Cocok untuk OLTP di mana operasi write-intensive lebih dominan.
- Cepat untuk transaksi yang sering mengubah data.
- Kurang efisien untuk query analitis yang memerlukan akses kolom tertentu dari banyak baris.

Columnar:

- Menyimpan data dalam kolom (column-oriented).
- Efisien untuk query read-intensive yang memerlukan akses kolom tertentu.
- Cocok untuk OLAP di mana operasi read-intensive lebih dominan.
- Lebih cepat untuk query analitis, kompresi data lebih baik.
- Kurang efisien untuk operasi write-intensive karena data harus ditulis ke beberapa kolom.