

Nama : Karno

Jurusan : data engineer 01

1. Perbedaan data warehouse dan data lake :

- Data warehouse adalah basis data yang telah diatur dengan struktur yang terdefinisi dengan baik. Sedangkan Data lake adalah tempat penyimpanan data mentah, yang memungkinkan menyimpan berbagai jenis data, termasuk data terstruktur, semi-terstruktur, dan tidak terstruktur.
- Data warehouse berfokus pada data yang sudah diproses dan disiapkan untuk analisis. Data lake dapat menyimpan data mentah yang mencakup data masa lalu, saat ini, dan masa depan yang membutuhkan analisis lebih mendalam.
- Data warehouse telah mengalami proses ETL (Extract, Transform, Load) untuk mempersiapkan data sebelum disimpan. Data lake memungkinkan penyimpanan data mentah tanpa perlu transformasi sebelumnya.

2. Perbedaan antara OLAP dan OLTP :

- OLAP adalah teknologi yang digunakan untuk analisis data, pemrosesan data besar, dan pengambilan keputusan. OLTP (Online Transaction Processing): OLTP adalah teknologi yang digunakan untuk memproses transaksi bisnis sehari-hari.
- struktur Data dalam OLAP umumnya diatur dalam format yang mendukung analisis kompleks, struktur Data dalam OLTP diatur dalam bentuk yang sesuai untuk menjalankan transaksi. Dan merupakan struktur normalisasi yang menghindari duplikasi data
- Data warehouse OLAP seringkali mengelola volume data yang besar dan kompleks, Database OLTP biasanya mengelola volume data yang relatif lebih kecil dan relevan dengan transaksi harian.
- OLAP sering menggunakan indeks kolom untuk mengoptimalkan query analisis yang kompleks.
- OLTP menggunakan indeks untuk mengoptimalkan operasi pencarian dan penggantian data dalam transaksi.

3. Teknologi yang biasa dipakai untuk data warehouse :

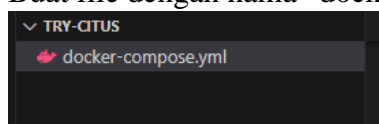
- Sistem Manajemen Basis Data (DBMS) seperti Microsoft SQL Server, Oracle Database, IBM Db2, Postgresql
- Alat ETL (Extract, Transform, Load): Perangkat lunak ETL seperti Apache Nifi, Talend, Informatica, Microsoft SSIS (SQL Server Integration Services), dan Apache Spark
- Perangkat lunak BI seperti Tableau, Power BI, QlikView, dan MicroStrategy

4. Tahapan instalasi citus menggunakan docker compose :

- Buat folder baru dengan nama bebas (try-citus)



- Buat file dengan nama “docker-compose.yml”



- Salin perintah dibawah ini dari link “[docker/docker-compose.yml at master · citusdata/docker \(github.com\)](https://github.com/citusdata/docker/blob/master/docker-compose.yml)”

```

1 # This file is auto generated from it's template,
2 # see citusdata/tools/packaging_automation/templates/docker/latest/docker-compose.tpl.yml.
3 version: "3"
4
5 services:
6   master:
7     container_name: "${COMPOSE_PROJECT_NAME:-citus}_master"
8     image: "citusdata/citus:12.0.0"
9     ports: ["${COORDINATOR_EXTERNAL_PORT:-5432}:5432"]
10    labels: ["com.citusdata.role=Master"]
11    environment: &AUTH
12      POSTGRES_USER: "${POSTGRES_USER:-postgres}"
13      POSTGRES_PASSWORD: "${POSTGRES_PASSWORD}"
14      PGUSER: "${POSTGRES_USER:-postgres}"
15      PGPASSWORD: "${POSTGRES_PASSWORD}"
16      POSTGRES_HOST_AUTH_METHOD: "${POSTGRES_HOST_AUTH_METHOD:-trust}"
17   worker:
18     image: "citusdata/citus:12.0.0"
19     labels: ["com.citusdata.role=Worker"]
20     depends_on: [manager]
21     environment: *AUTH
22     command: "/wait-for-manager.sh"
23     volumes:
24       - healthcheck-volume:/healthcheck
25   manager:
26     container_name: "${COMPOSE_PROJECT_NAME:-citus}_manager"
27     image: "citusdata/membership-manager:0.3.0"
28     volumes:
29       - "${DOCKER_SOCKET:-/var/run/docker.sock}:/var/run/docker.sock"
30       - healthcheck-volume:/healthcheck
31     depends_on: [master]
32     environment: *AUTH
33 volumes:
34   healthcheck-volume:

```

- isi beberapa perintah pada code seperti container\_name dari master, ports, postgres\_user, postgrs\_password, pguser, pgpassword, dan container\_name dari manager sesuai dengan data yang anda mau

```





1 # This file is auto generated from it's template,
2 # see citusdata/tools/packaging_automation/templates/docker/latest/docker-compose.tpl.yml.
3 version: "3"
4
5 services:
6   master:
7     container_name: "citus_master"
8     image: "citusdata/citus:12.0.0"
9     ports: ["5432:5432"]
10    labels: ["com.citusdata.role=Master"]
11    environment: &AUTH
12      POSTGRES_USER: "mysuperuser"
13      POSTGRES_PASSWORD: "passuper123"
14      PGUSER: "myuser"
15      PGPASSWORD: "pas123"
16      POSTGRES_HOST_AUTH_METHOD: "${POSTGRES_HOST_AUTH_METHOD:-trust}"
17   worker:
18     image: "citusdata/citus:12.0.0"
19     labels: ["com.citusdata.role=Worker"]
20     depends_on: [manager]
21     environment: *AUTH
22     command: "/wait-for-manager.sh"
23     volumes:
24       - healthcheck-volume:/healthcheck
25   manager:
26     container_name: "citus_manager"
27     image: "citusdata/membership-manager:0.3.0"
28     volumes:
29       - "${DOCKER_SOCKET:-/var/run/docker.sock}:/var/run/docker.sock"
30       - healthcheck-volume:/healthcheck
31     depends_on: [master]
32     environment: *AUTH
33 volumes:
34   healthcheck-volume:

```

- Jalankan perintah `docker-compose -p citus up -d` dan tunggu hingga proses pembuatan container selesai

```
PS F:\altera_academy\try-citus> docker-compose ps -a
NAME          IMAGE          COMMAND          SERVICE    CREATED   STATUS    PORTS
PS F:\altera_academy\try-citus> docker-compose -p citus up -d
[+] Building 0.0s (0/0)
[+] Running 4/4
✓ Network citus_default      Created
✓ Container citus_master     Created
✓ Container citus_manager    Created
✓ Container citus-worker-1   Created
```

Apabila sudah berhasil maka akan muncul container di aplikasi docker desktop

<input type="checkbox"/>	Name	Image	Status	CPU (%)	Port(s)	Last started
<input type="checkbox"/>	 <b>citus</b>		Running (3/3)	N/A		7 minutes ago
<input type="checkbox"/>	 <b>citus_master</b> 4478e2d75819	<a href="#">citusdata/citus:12.0.0</a>	Running	N/A	<a href="#">5432:5432</a>	7 minutes ago
<input type="checkbox"/>	 <b>citus_manager</b> 5e6201695c90	<a href="#">citusdata/membership-man</a>	Running	N/A		7 minutes ago
<input type="checkbox"/>	 <b>worker-1</b> 9c61546e17a7	<a href="#">citusdata/citus:12.0.0</a>	Running	N/A		7 minutes ago

- Selanjutnya kita perlu menjalankan citus di dalam docker menggunakan perintah “`docker exec -it citus_master bash`”

```
PS F:\altera_academy\try-citus> docker exec -it citus_master bash
root@4478e2d75819:/#
```

- Hubungkan ke postgresql yang tadi di code docker compose gunakan perintah “`psql -U mysuperuser -d postgres`”

```
root@4478e2d75819:/# psql -U mysuperuser -d postgres
psql (15.3 (Debian 15.3-1.pgdg120+1))
Type "help" for help.

postgres=#
```

- Selanjutnya kita perlu terhubung ke citus extension menggunakan perintah “`create extension citus;`”

```
postgres=# create extension citus;
CREATE EXTENSION
postgres=#
```

- Untuk membuat table kita perlu menuliskan perintah create table seperti gambar dibawah

```
postgres=# CREATE TABLE events_columnar (
postgres(#   device_id bigint,
postgres(#   event_id bigserial,
postgres(#   event_time timestamptz default now(),
postgres(#   data jsonb not null
postgres(# )
postgres=# USING columnar;
CREATE TABLE
postgres=#
```

- Table sudah dibuat untuk melihat table tinggal tuliskan perintah seperti didalam postgresql biasa “select \* from events\_columnar;

```
postgres=# select * from events_columnar;
 device_id | event_id | event_time | data
-----+-----+-----+-----
(0 rows)

postgres=#
```

- Insert Table

```
postgres=# INSERT INTO events_columnar (device_id, data)
postgres=# SELECT d, '{"hello": "columnar"}' FROM generate_series(1,100) d;
INSERT 0 100
postgres=#
```

- Lihat 10 table teratas menggunakan perintah select \* from events\_columnar

```
postgres=# select * from events_columnar limit 10;
 device_id | event_id | event_time | data
-----+-----+-----+-----
          1 |         1 | 2023-10-31 15:56:41.540262+00 | {"hello": "columnar"}
          2 |         2 | 2023-10-31 15:56:41.540262+00 | {"hello": "columnar"}
          3 |         3 | 2023-10-31 15:56:41.540262+00 | {"hello": "columnar"}
          4 |         4 | 2023-10-31 15:56:41.540262+00 | {"hello": "columnar"}
          5 |         5 | 2023-10-31 15:56:41.540262+00 | {"hello": "columnar"}
          6 |         6 | 2023-10-31 15:56:41.540262+00 | {"hello": "columnar"}
          7 |         7 | 2023-10-31 15:56:41.540262+00 | {"hello": "columnar"}
          8 |         8 | 2023-10-31 15:56:41.540262+00 | {"hello": "columnar"}
          9 |         9 | 2023-10-31 15:56:41.540262+00 | {"hello": "columnar"}
         10 |        10 | 2023-10-31 15:56:41.540262+00 | {"hello": "columnar"}
(10 rows)
```

##### 5. Perbedaan antara acces method heap dan columnar :

- Acces method heap data disimpan dalam format baris secara berurutan dalam satu tempat, sedangkan penyimpanan columnar data disimpan dalam format kolom. Setiap kolom dalam tabel disimpan terpisah dalam blok penyimpanan.
- Acces data heap memerlukan membaca seluruh baris data sedangkan columnar hanya perlu membaca kolom yang diperlukan.